Advances in Information Security 107

Yingying Chen
Jie Wu
Paul Yu
Xiaogang Wang *Editors*

Network Security Empowered by Artificial Intelligence

Springer

Advances in Information Security

Volume 107

The purpose of the *Advances in Information Security* book series is to establish the state of the art and set the course for future research in information security. The scope of this series includes not only all aspects of computer, network security, and cryptography, but related areas, such as fault tolerance and software assurance. The series serves as a central source of reference for information security research and developments. The series aims to publish thorough and cohesive overviews on specific topics in Information Security, as well as works that are larger in scope than survey articles and that will contain more detailed background information. The series also provides a single point of coverage of advanced and timely topics and a forum for topics that may not have reached a level of maturity to warrant a comprehensive textbook.

Yingying Chen • Jie Wu • Paul Yu •
Xiaogang Wang
Editors

Network Security Empowered by Artificial Intelligence

Springer

Editors
Yingying Chen
Department of Electrical and Computer Engineering
Rutgers University
New Brunswick, NJ, USA

Jie Wu
Department of Computer and Information Sciences
Temple University
Philadelphia, PA, USA

Paul Yu
Network Sciences
Army Research Office (ARO)
Raleigh, NC, USA

Xiaogang Wang
Secure and Trustworthy Cyberspace (SaTC)
National Science Foundation (NSF)
Alexandria, VA, USA

ISSN 1568-2633 ISSN 2512-2193 (electronic)
Advances in Information Security
ISBN 978-3-031-53509-3 ISBN 978-3-031-53510-9 (eBook)
https://doi.org/10.1007/978-3-031-53510-9

This Springer imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Preface

Information assurance in network science must provide authentic, accurate, secure, reliable, and timely information to warfighters to achieve information dominance, regardless of threat conditions. Computing and information processes may be carried out over distributed and heterogeneous systems, which may include mobile edge, mobile computing and communications systems, and high-performance information process systems that are inter-connected through both tactical and strategic communication and network systems. The advancement of artificial intelligence (AI) and machine learning (ML) has led to new opportunities for efficient tactical communication and network systems but also brought in new vulnerabilities.

The topics of this book will be futuristic and forward-looking on security in spectrum management, mobile networks, and next-generation wireless networks in the era of AI/ML. Topics include, but are not limited to, robust and trusted wireless and mobile networks, models and metrics for next-generation robust systems, cyber deception, the principle of moving target defense, trusted learning for cyber autonomy, spectrum management, and network forensics.

The main goal of this book is to collect the recent developments on the principles, techniques, and applications of AI/ML in future network security and applications. It will focus on various threat models/behaviors and the corresponding countermeasures using AI/ML techniques. This book will be of value to academics, researchers, practitioners, government officials, business organizations (e.g., executives, marketing professionals, resource managers, etc.), and even customers—working, participating, or those interested in fields related to AI/ML in future network security and applications.

This book accepts contributions from various topics in the field of AI/ML in future network security and applications. There are several existing books on AI/ML for cybersecurity: 1. *Artificial Intelligence for Cybersecurity*, M. Stamp et al. (Eds), Springer; 2. *Illumination of Artificial Intelligence in Cybersecurity and Forensics*, S. Misra et al. (Eds), Springer; 3. *Machine Learning and Cognitive Science Applications in Cyber Security*, S. M. Kahn, IGI Global; 4. *Handbook*

of Research on Machine and Deep Learning Applications for Cyber Security, P. Ganapathi, IGI Global; and 5. *Machine Learning and Security: Protecting Systems with Data and Algorithms*, C. Chio, O'Reilly. However, none of the existing books cover AL/ML in the network aspect of cybersecurity.

The content of the book will be especially useful for students in areas like computer networks, artificial intelligence, machine learning, and data science, who would benefit from the information, cases, and examples therein. This book can also be used as a reference/textbook for the above areas. The secondary audience includes practical partitioners in industry in these areas. The focus of this book is to expose readers to the technical challenges in building AI/ML techniques to be applied in future network security and applications. This book is organized into 4 parts with a total of 15 chapters. Each part corresponds to an important snapshot, starting from an introduction and overview of general future networking.

- Part I: Architecture Innovations and Security in 5G Networks (Chaps. 1 and 2)
- Part II: Security in Artificial Intelligence-Enabled Intrusion Detection Systems (Chaps. 3, 4, and 5)
- Part III: Attack and Defense in Artificial Intelligence-Enabled Wireless Systems (Chaps. 6, 7, 8, 9, and 10)
- Part IV: Security in Network-Enabled Applications (Chaps. 11, 12, 13, 14, and 15)

Part I provides the foundation for our exploration. Chapters 1 and 2 delve into the architectural innovations and security challenges of 5G networks. The development of novel network structures and decision-dominant defense strategies in the context of 5G technology and its potential vulnerabilities is thoroughly discussed.

Chapter 1 addresses the challenge of designing and validating a standalone next-generation mobile core network architecture necessary to support the requirements of 5G radio access technologies and beyond. More specifically, this chapter describes a new network architecture called nCore, which can support the requirements of 5G and beyond. This architecture includes both security and privacy components. The core of nCore is based on a distributed information-centric structure with unique identifiers for network objects together with the concept of locator-ID separation.

Chapter 2 develops a game-theoretic framework for the decision-dominant zero-trust defense of 5G networks. This chapter studies multi-domain warfare and the complexities introduced by 5G technologies. Given the potential vulnerabilities of diverse interconnections and supply chains, the emphasis is on adopting a zero-trust architecture. The proposed solution leverages a 5G satellite-guided air-ground network, using a decision-dominant, learning-based approach to preemptively counter threats. The research showcases a game-theoretic design enriched by meta-learning, ensuring robust defense in modern warfare environments.

Part II directs its focus toward the intricate relationship between artificial intelligence and security. Chapters 3 to 5 provide an in-depth examination of

intrusion detection systems, exploring their vulnerabilities and highlighting the crucial role that AI and machine learning play in enhancing both defense and vulnerability assessment.

Chapter 3 includes discussions on both defense and vulnerability in IDS. This chapter starts with an overview of intrusion detection systems and then discusses the relationship between security and AI/ML. The authors evaluate the security of AI/ML systems from an end-to-end perspective. This approach accounts for system vulnerability and discusses the need for a vulnerability disclosure program AI/ML.

Chapter 4 explores the properties of the adversarial examples' transferability. The authors use different Adversarial Examples (AEs) to interact with different well-trained models to find the key insights of transfer attacks in the network. This chapter investigates the vulnerabilities of network traffic packet detection systems to AEs, where slight modifications to network packets can deceive detection systems. While current AEs are based on white-box settings, the chapter explores their potential transferability to black-box models. By examining various intrusion detection systems and creating distinct models, the study assesses the efficacy of various AEs against these models. The findings highlight certain commonalities between transfer and white-box attacks, suggesting avenues for more advanced transfer attacks in upcoming research.

Chapter 5 reviews the state-of-the-art machine learning and deep learning-based intrusion detection methods for Software Defined Networking (SDN), discussing both defense and vulnerability. This chapter explores the promise and challenges of SDN, a groundbreaking option for the Internet's future growth. While SDN enhances network flexibility and control, it also introduces vulnerabilities, making it especially prone to Denial-of-Service (DoS) attacks. To counteract these threats, integrating an IDS within SDN is vital. The chapter delves into advanced machine learning and deep learning-based IDS approaches tailored for SDN, assessing their performance on criteria like accuracy and processing time. Through hands-on evaluations, the chapter seeks to pinpoint the most effective IDS methods for SDN setups.

Part III takes a closer look at the dynamic arena of wireless systems and their susceptibility to attacks. In Chaps. 6 and 7, deep learning is explored as a means to fortify wireless communications and confront adversarial threats in millimeter-wave-based systems. Innovative solutions are presented to ensure robust and secure wireless connectivity.

Chapter 6 presents three Deep Learning-based solutions for achieving robust and secure wireless communications from a defense perspective. This chapter explores the vulnerabilities in wireless communications due to the growth of mobile technologies and increasing spectrum demand. As these systems evolve to be more software-centric, they face threats from entities like jammers and drones. Drawing from Deep Learning's successes in areas such as computer vision, the chapter presents three strategies: a system for pinpointing wireless collisions, the "JaX" technique using Convolutional Neural Networks to combat jammers, and

"DEFORM," a beamforming method that leverages neural networks for robust communications across diverse RF signals.

Chapter 7 investigates both white-box and black-box adversarial attacks on millimeter-wave (mmWave)-based HAR systems from the attack perspective. It deals with mmWave and its applications in Human Activity Recognition (HAR). This chapter surveys various adversarial attacks, including white-box and block-box, on mmWave systems. Two solutions are proposed: one for white-box attacks and the other one for black-box attacks.

In Chap. 8, various attack methods and defense schemes are evaluated in several wireless positioning systems from an attack perspective. This chapter studies wireless localization that uses wireless technologies to obtain position-related information for target localization. Various attacks and defense schemes are evaluated in several positioning systems to show the vulnerabilities in deep learning-based localization systems and, hence, to show the importance of a robust system.

Chapter 9 utilizes Convolutional Neural Networks (CNN) to improve localization accuracy for both single and multiple simultaneous wireless transmitters from a defense perspective. Specifically, this chapter explores Received Signal Strength (RSS)-based localization techniques based on crowdsourced measurements and CNNs to improve localization accuracy. It is shown that adversarial training is effective as a defense mechanism against adversarial attacks. Some challenges in designing practical localization systems are also discussed.

Chapter 10 is motivated by wireless communication scenarios. It introduces the state-of-the-art results on bandits and Reinforcement Learning (RL) and their importance on network security under limited defender resources. This chapter investigates RL, emphasizing its adaptability in communication networks and its pivotal role in challenges like protein folding. The discussion delves into adversarial RL, highlighting the challenges agents confront in frequently updating their strategies in dynamic environments. Particularly for energy-constrained devices, "switching costs"—the expenses of policy changes—emerge as a crucial metric. The research presents the latest insights on RL and bandits considering switching costs, underscoring their significance in resource-limited network security, and suggests potential avenues for future research.

Part IV broadens the perspective to survey the diverse applications of these emerging technologies in the realms of network-enabled applications. It offers a panoramic view of the security and privacy challenges within these domains. Chapters 8 to 15 scrutinize the vulnerabilities and defense strategies in various application areas, including augmented reality, federated learning, and cyber-physical systems. This examination sheds light on the intricate interplay between technology and security in these contexts.

Chapter 11 discusses security and privacy in augmented reality (AR). It focuses on pure AR systems without any network component. Specifically, this chapter discusses AR systems that rely on real-time sensing through various sensors to understand the physical environment. The focus is on AR vulnerability under attacks in both security and privacy based on the analysis of the sensor signal

processing flow and the design of sensor hardware. This chapter also studies existing countermeasures.

Chapter 12 focuses on the AR system. It includes some discussions in both attack and defense regarding network-based approaches. This chapter looks at AR security and privacy challenges based on the interactions between AR applications and users, which include data privacy, authentication, and authorization. The current state-of-the-art AI/ML-based protection solutions are studied, including deep learning and reinforcement learning.

Chapter 13 centers around malware detection. It proposes a lightweight image-based malware classifier resilient against four adversarial attacks in black-box and white-box settings. This chapter delves into the vulnerabilities of machine and deep learning models in network security, highlighting the risks posed by adversarial samples. In response, a lightweight image-based malware classifier using CNN is proposed. This classifier analyzes Windows Portable Executable (PE) malware images and proves resilient to adversarial attacks, maintaining high accuracy even under increased perturbations. Notably, compared to the leading MalConv classifier, it significantly reduces training time and trims random-access memory usage by threefold.

Chapter 14 provides an overview of vulnerabilities and defense strategy in federated learning. This chapter explores federated learning (FL), a decentralized artificial intelligence approach that promotes collaborative learning without direct data sharing. While FL's decentralized framework offers distinct advantages, it also becomes especially vulnerable to adversarial attacks, such as backdoor and byzantine attacks. The intricacy of these threats, heightened by adversaries potentially acting as participants, raises barriers to the global acceptance of FL models. The chapter meticulously scrutinizes the unique attributes of each security threat and the inherent susceptibilities of FL. Additionally, it sheds light on defense strategies to identify malicious actors or mitigate the effects of attacks on the overarching model.

Chapter 15 focuses on pure Cyber-Physical Systems (CPS). It includes a discussion on CPS security and network-enabled applications. This chapter examines the integration and implications of CPS in safety-critical sectors like robotics and power systems, noting that faults and cyberattacks can jeopardize safety and human lives. With a surge in research focusing on CPS security over the past decade and the rise of AI and machine learning in various applications, there is an inclination to leverage AI/ML for CPS security. However, the authors offer insights and cautionary lessons. They emphasize understanding the role of physical systems, cyber effects, their interactions, system controls, and the potential of AI/ML in enhancing CPS resilience while also discussing its limitations and future research challenges.

We would like to express our gratitude to all the contributing authors. This book would not be possible without their generous contributions and dedication time wise. We also thank the Army Research Office (ARO), who sponsored a

special workshop upon which most of the authors of this book were drawn. Our special thanks are given to the Springer managing editor Susan Lagerstrom-Fife and production editor Arum Siva Shanmugam, who gave us both initial encouragement, support, and continuous guidance during the book editing process. Finally, we would like to thank our families for their great understanding and patience during this project. Readers are encouraged to provide feedback to the contacts below. We hope readers will find this book useful in their studies or in their workplace!

New Brunswick, NJ, USA Yingying Chen
Philadelphia, PA, USA Jie Wu
Raleigh, NC, USA Paul Yu
Alexandria, VA, USA Xiaogang Wang

Contents

Part I
Architecture Innovations and Security in 5G Networks

nCore: Clean Slate Next-G Mobile Core Network Architecture for Scalability and Low Latency

Shalini Choudhury, Shreyasee Mukherjee, Parishad Karimi, and Dipankar Raychaudhuri

1 Introduction and Background

This chapter addresses the technology challenge of designing and validating a standalone next-generation mobile core network architecture necessary to support the requirements of 5G radio access technologies and beyond. The importance of dramatically improving the efficiency, performance and functional capabilities of mobile/wireless networks has been recognized for some time. This goal has become particularly urgent with the emergence of smartphones as the primary computing and communication platform and the continuing exponential growth in mobile data. 5G systems aim to provide a significantly enhanced mobile user experience with ∼ gigabits per second (Gbps) wireless bit-rates, low latency, and improved reliability. New radio access technologies, including massive MIMO [1], millimeter wave [2], and centralized radio access network (CRAN) [3], are proposed to achieve large gains in physical layer performance. Support for Internet-of-Things (IoT) applications is another important 5G design goal, introducing requirements such as high device density, low complexity and energy efficiency. Next-generation wireless systems are expected to support time-critical applications like augmented reality (AR), autonomous vehicles, and real-time control. These applications require edge network latencies of 10 ms or lower, an order of magnitude reduction relative to existing LTE cellular networks.

Though still at an early stage of conceptualization, beyond 5G ("B5G" or "6G") scenarios are expected to feature even higher bit-rates of approximately 10 Gbps [4] and lower latencies around 2–5 ms [5]. Additionally, the next-generation network will introduce new functionalities, including support for content retrieval, context-

S. Choudhury (✉) · S. Mukherjee · P. Karimi · D. Raychaudhuri (✉)
WINLAB, Rutgers University, New Brunswick, NJ, USA
e-mail: shalini@winlab.rutgers.edu; shreya@winlab.rutgers.edu; parishad@winlab.rutgers.edu; ray@winlab.rutgers.edu

Y. Chen et al. (eds.), *Network Security Empowered by Artificial Intelligence*,
Advances in Information Security 107, https://doi.org/10.1007/978-3-031-53510-9_1

aware messaging, in-network processing, and integration of cloud services into the edge network fabric. When considering 5G and 6G designs, much of the focus has been on the radio access technology to achieve higher wireless bit-rates. However, end-to-end service quality depends very strongly on the capabilities and performance of the wireless edge network. The current 3GPP architecture of mobile edge networks has evolved through two generations of cellular (3G and 4G). However, it is fundamentally limited by its evolutionary design, which incorporates legacy signaling from switched telecom networks. This design integrates with Internet protocols using a complex design based on gateways and tunnels [6].

It should be noted that in 5G Phase 2 and beyond, standardization groups such as IETF DMM [7], etc. are exploring alternatives for GPRS Tunneling Protocol (GTP) [8]. Potential replacements being considered include Segment Routing IPv6 [9, 10], Locator/ID Separation Protocol (LISP) [11] and Host Identity Protocol (HIP) [12]. This clearly motivates that the mobile core network design has to change significantly in order to meet the increasingly complex and diverse requirements associated with 5G/6G. Thus motivating consideration of optimized clean-slate designs that could initially be applied to private networks at the wireless edge. Information-centric network techniques based on the use of content routing, named data [13] or named objects [14] offer a fundamentally new architectural framework on which to design clean slate wireless edge protocols.

This chapter presents a distributed flat core network design that leverages the concept of identifiers appended with distributed mapping to achieve optimal routing (without data packets traveling through tunnels and gateways), low latency, seamless mobility and scalability while providing flexibility for future services. The name-based architecture starts with the premise of identifying an endpoint with a persistent name separate from the routable address(es) associated with the endpoint. Assuming endpoints (smartphones, cars on a highway, etc.) are inherently mobile, address(es) associated with each may change as the endpoint moves and associates with different access networks. However, the mobile device can always be uniquely identified by its name (endpoint identifier, EID), forming the crux of "Locator-ID" split architectures which utilize a mapping service to maintain an up-to-date name-to-address mapping. Identifier and locator separation schemes have been adopted by academic and industrial researchers as the major design pillars for clean-slate design towards evolving internet architecture. The most notable locator-ID split name-based architectures that have been proposed in this context include LISP [11], HIP [12], NDN [13] and Mobility First [14]. While these name-based architectures have mostly been studied in the context of intra and inter-domain routing, content delivery networks, and data center networks, our approach aimed at mobility services here is different. A high-level representation of 3GPP core and nCore architecture is shown in Fig. 1. The physical fabric of the 3GPP standardized core is hierarchical in nature, with a single point of entry into the network via access and mobility management function (AMF) and exit through the user plane functions (UPFs). All traffic flows through tunnels between 3GPP and non-3GPP access networks and UPFs to reach the data network. On the contrary, nCore shown in Fig. 1 exhibits no single anchor point for entry or vendor-specific UPFs for exiting into the data network.

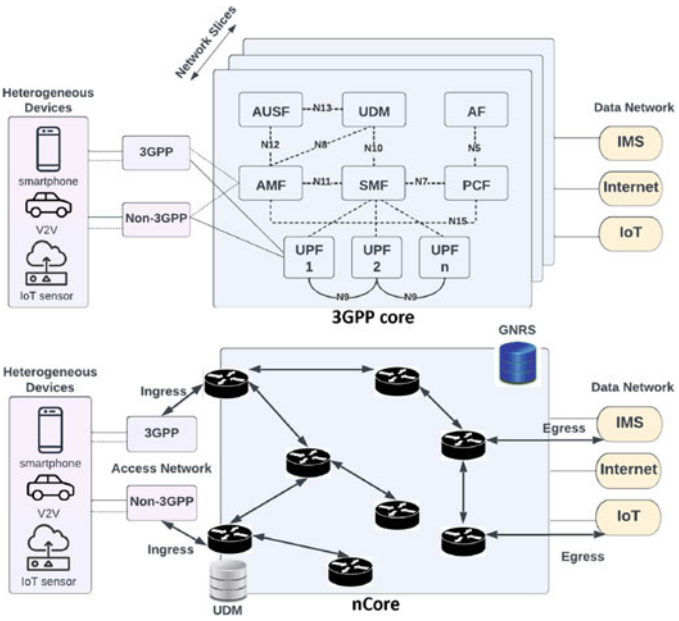


Fig. 1 Network core architectures: 3GPP core and nCore

The proposed Locator-ID split network architecture relies on a scalable, distributed mapping system named global name resolution service (GNRS) [15]. GNRS binds name and address and facilitates services in the 5G and beyond architecture and the cellular mobile core. The user equipment (UE) mobility is handled by assigning it a unique name and mapping it to the routable address of the base station (next generation node B (gNB), access point (AP)) to which it is currently attached. In addition, by removing the gateways from the core, packets can flow freely into and out of the network through the 'best' path guided by distributed routing policies. The Locator-ID split, along with a dynamic name resolution (mapping) provides a natural solution for mobility management while also addressing fundamental scalability and latency limitations inherent to the current 3GPP architecture [14].

2 Next-Gen Mobile Core Requirements

5G and beyond services demand throughput in tens of Gbps implying a 10–100× increase in data rate from the previous generation of networks, along with sub 5-ms latency and upto 100× number of connected devices per unit area. The 5G architecture takes the separation between the control and data plane one step further in comparison to LTE. The control part of the UPF packet gateways are

separated from their data plane counterparts and new interfaces are defined between these new components. The greater degree of separation between the control and data plane components, and more granular allocation of tasks to components, may indeed simplify each component. However, it also has the effect of increasing the number of components involved in serving a user's request, as well as the number of messages exchanged between them. This motivates the consideration of clean-slate designs of the 5G control plane with the goal of significantly improving latency and scalability. It is necessary to re-think the next-gen cellular networks with a focus on increasing throughput, reducing latency and eliminating centralized bottlenecks. In [6] the authors propose a redesign of the 5G control plane (called "CleanG") aimed at reducing protocol overhead and improving network throughput. In this work, we describe a new identifier-based clean-slate design that offers further improvements in latency and throughput performance. In the following sections, we first discuss the mobile core network requirements and then introduce the nCore design responsive to these objectives.

2.1 Ultra-High Bit Rate

A per-flow throughput of a gigabit per second per mobile user motivates a "bearer-less" network model tunnels to a centralized gateway, a potential bottleneck for anticipated traffic volumes associated with Gbps wireless links. Additionally, the dual-mode 5G core designed for interworking with legacy 4G networks and evolution of 5G, integrates cloud-native network functions (NF) with Virtual or Physical NFs. In the dual-mode 5G core, the packet core gateway acts as the common convergence point from which traffic flows toward the base station or packet data network (PDN). Studies show that a typical US based network provider has a limited number of gateways (4–6) [16] through which all endpoint traffic enters and exits the network. As a result, cellular data networks impose restrictions on routing data traffic by traversing only through the available UPF session anchors. This implies that system throughput may be limited by the capacity of these gateways in the hierarchically designed network. 5G networks should be designed to support traffic volumes of ∼100 Gbps to 1 Tbps, thus requiring a distributed network design without a centralized processing or routing bottleneck.

2.2 Low Latency

Achieving an order of magnitude reduction of service latency (<5 ms) has been a baseline requirement for 5G [15]. The latency of packet delivery is caused by several factors, including delays in both the control plane (CP) and the data plane (DP). One of the dominant delays in mobile networks is the CP latency associated with setting up a path for forwarding the first packet in a flow. Typically, data forwarding in

cellular networks involves a bootstrapping phase where an attaching device needs to authenticate and set up a session with the network. Given the hierarchical nature of the network and the legacy components, this step may involve exchanging more than 20 messages with the AMF and the session management functions (SMF), which accounts for up to 168.7 ms on average before a session is established [17]. These messages are primarily: authentication, mobility management and session management overheads. In the conventional gateway architecture, initial gateway signaling latencies can be significant components of the overall delay. In nCore, we envision a shift from per-flow signaling to a more distributed packet-switching approach, wherein forwarding decisions are made on a per-packet basis instead of maintaining a long-lived end-to-end session to a packet gateway. This is especially beneficial for devices that do not send a large amount of data at a time, such as IoTs, as described next.

2.3 Support for Internet-of-Things

According to Cisco, IoT connections will go from 6.1 billion in 2018 to 14.7 billion by 2023 [18]. Most of these devices are power-constrained while sending sporadic bursts of short packets. Consequently, their requirements are quite different from typical cellular endpoints. High bandwidth is not a strict requirement for IoT devices, but low overhead control protocols are required in order to improve network efficiency (loosely defined as the ratio of data vs. control bytes). Narrowband IoT (NB-IoT) is the current solution that assigns a separate channel solely for the use of IoTs. However, if NB-IoT is used in conjunction with the existing core network, this will result in high control overhead compared to the low data traffic rate that IoTs typically need. One solution proposed for NB-IoT is to go through all the control protocols for authentication, mobility and session management during bootstrapping but then cache this state at the basestation. Once this is done, the session does not need to be re-established every time a device wakes up to send a few bytes of data. This approach assumes these devices are static and will not require handover capabilities. This approach does not apply to IoT devices such as static power-constrained motes on an agricultural farm to highly mobile automotive tire pressure sensors. Many of these IoT devices will be unable to use NB-IoT unless significant improvements in the network protocols are made to improve latency and mobility.

2.4 Heterogeneity in Access Networks

3GPP release 15 envisions multiple access networks that can be plugged into the same core network. The core network should be radio technology agnostic and support a mix of 4G, 5G and WiFi radio access technologies. For this purpose, all the components of the core have been modularized such that if required

a subset of these components can be stitched together to form a network by bypassing the rest of the modules. This is relevant to the design proposed here, as modularization is consistent with a distributed flat nCore that can be stitched to multiple heterogeneous access networks. In order to utilize multiple of these access networks simultaneously and more efficiently, a distributed core will provide better path availability and reduce the chances of traffic bottleneck.

3 nCore Network Architecture

Based on the next-generation mobile core requirements, nCore network aims to offer a seamless, connected experience for the end users. The distributed nCore is designed to address key challenges of 3GPP standardized core, including reachability, fast handoffs, multi-homing support and radio resource management. In view of achieving the above requirements, the flat mobile core has the following key features: (1) Distributed control: There are no gateways in the architecture and no end-to-end session management protocols; (2) Routing functions distributed across all network components: Access networks as well as core network components all perform routing and mobility management functionality; (3) Traffic can flow into and out of the network freely to and from multiple ingress and egress points. Furthermore, the access points, and correspondingly the user devices, can employ either unlicensed or new alternatively-licensed bands (such as the proposed Federal Communications Commission (FCC) small-cell band) for the last hop. The proposed nCore network considers potential security and privacy threats for designing a network architecture. Security concerns are addressed by securing GNRS updates and queries and additionally creating a framework for anonymity. A discussion on nCore architecture, security and privacy is presented in this section to understand how the clean slate nCore architecture overcomes the control and data plane limitations of the standardized state-of-the-art 3GPP network core.

3.1 Architecture Overview

The high-level network architecture of the distributed core network is shown in Fig. 2. The radio resource control (RRC) connection setup, authentication and UE attachment to the network remain unchanged. The nCore removes the SMF as well as the gateways and instead utilizes the distributed protocol logic at gNBs and routers in the network (hence the term “flat” mobile core). The nCore architecture is based on the concept of unique identifiers or names for UEs along with a distributed mapping (D-map) service. This leads to mobility support, reducing bottleneck at the AMF which handles 3 times more traffic than the gateways [19]. In the nCore architecture, this service is called global name resolution service (GNRS) and

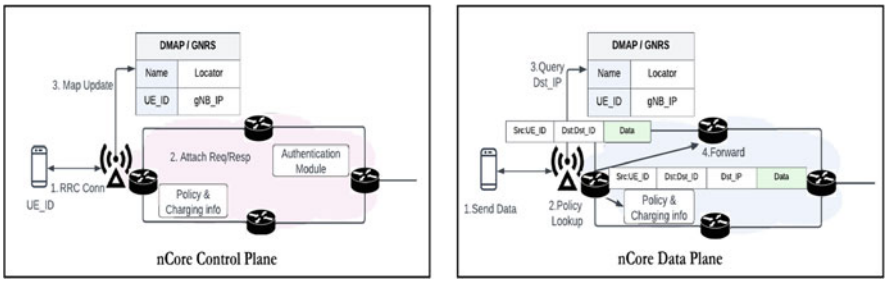


Fig. 2 nCore control and data plane with distributed mapping

is implemented as a distributed hash table in which all routers of the network participate [15].

The authentication entity, which is tasked with maintaining UE subscription, policy and charging information of the network, remains as part of the core. The BSs communicate with it directly to authenticate the UE and to obtain relevant policies pertaining to that UE during the bootstrapping phase. With no SMF, the bootstrapping phase now only has authentication and distributed mobility management. The network being completely flat results in a plug-and-play capability, where multiple radio access technologies (such as 4G, 5G or WiFi) can be plugged in. This capability allows the network to grow organically, provided all the network components participate in the control plane that supports the D-map and authentication functions.

3.2 Mobility Management

The distributed mobility service is based on assigning permanent global identifiers for all network attached devices along with the routers [20]. All routers in the network participate in a distributed hash table implementation, wherein all the mappings of endpoint names to their routable addresses (address of the BSs an endpoint is connected to) are stored across routers in the network. The mapping service is therefore physically distributed; however, as in any hash table implementation, given an endpoint name, it can be hashed to obtain the unique address of the router that needs to be queried in order to find the up-to-date name-address mapping and hence the current location of that endpoint.

The service also has resiliency mechanisms such as storing the mapping information of an endpoint at multiple locations using multiple hash functions in case any of the routers go down. The control overhead of maintaining such a distributed mapping service is light as these routers need not run additional synchronization protocols, provided they all have adequate storage capabilities and the bandwidth for query/response of their local databases. Prior work on such distributed services

has proven them to be scalable to Internet-size networks [19]. Detailed evaluations for large global scale topologies have shown that query/response latencies can be as low as 10 ms [15] with suitable optimization and caching techniques.

3.3 Packet Forwarding

Given the underlying routing and mapping services, a UE joining the network first establishes an RRC connection with a nearby BS, followed by authentication at the BS via similar protocols as in 5G, as shown in Fig. 2 control plane. Policy and charging along with authentication, authorization, and accounting information are communicated directly to the authenticating BS. The UE name-to-address mapping is updated in the GNRS. In the data plane, data can now flow freely following a packet-switched network paradigm. A nCore specific data plane scenario is highlighted in Fig. 2 data plane, where a data packet carries both the endpoint identifier and the current location of the endpoint. A source UE sends data to a destination identified by its name *(Dest-ID)*. The first hop router at the BS looks up the database to find the *(Dest-ID)* to address mapping which is then appended to the packet. Consecutive routers simply forward packets by looking up their forwarding tables for that particular address. Packets, therefore, enter/exit the core network along the best intra-domain routing path, which may also reflect UE-specific policies.

3.4 Policy and Charging

The nCore network no longer uses centralized services (with the exception of the authentication server) or packet gateways. A dedicated network function (NF) co-located with the BS ensures that appropriate amounts of bandwidth are dynamically allocated to each service in real time. Prior to launching new services, the policy and charging NF needs to validate operator-related policy rules to ensure there is sufficient capacity to provide the requested services. Charging is applicable to each service data flow and is primarily based on information such as application identifier, type of stream (audio, video, etc.), application data rate, etc.

3.5 Security in nCore Architecture

In the nCore architecture each network element has three attributes: a user-level descriptor (i.e. human-readable name, e.g. John's phone), a network-level identifier (globally unique identifier, or GUID) and a routable address (network address, or NA). In the two-step approach to name resolution in nCore, a Name Certificate

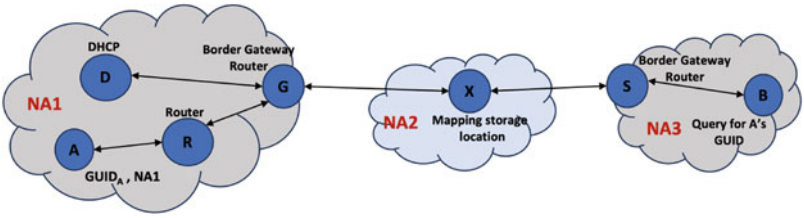


Fig. 3 Secure GNRS update and query

Service (NCS) is used to translate user-level descriptors into GUIDs, while the GNRS provides the mapping between GUIDs and the corresponding NAs. The benefits of a resolution service are undermined when subjected to security threats, and consequently, any clean-slate name resolution service should give security significant consideration. In the design choice to secure GNRS there are two components involved:

Secure GNRS Update As seen in Fig. 3, when user A in network NA1 needs to update its mapping ($GUID_A$,$NA1$), it sends an update to the local router. This GNRS mapping update includes timestamps to prevent stale mapping attacks. The mapping expiration time ensures freshness but must be balanced to prevent security risks or high overhead. The update is signed with the user's private key, securing against spoofing attacks. Lastly, the update message includes encryption and verification elements to ensure secure communication. Local router L verifies the ($GUID_A$,$NA1$) correctness of the mapping and forwards it to the border gateway router G for NA1. The mapping update message is signed by L's private key and subsequently encrypted by receiver G's public key. Further, G communicates with DHCP server to verify A's NA, if the response from D matches the mapping sent by A and forwarded by L, the mapping will be stored in X in NA2 [21].

Securing GNRS Query The GNRS query process involves a user retrieving a network address for a known GUID, shown in Fig. 3. The user, border gateway router, and mapping storage location participate in the process, which comprises four steps [21]:

1. User B sends a query for User A's GUID to their border gateway router, S, signed with B's private key and encrypted using S's public key.
2. After verification, S hashes the queried GUID and forwards the request to the nearest mapping storage location, X.
3. X validates the request and returns A's mapping to S.
4. S then forwards A's mapping to B, who can verify the mapping with timestamps and signatures.

GNRS Access Control Policies GNRS supports mobility by enabling a host to inform other network elements of the change in its location. However, it requires protection against illegitimate users querying and exploiting these mappings to

avoid privacy breaches and other serious issues, such as DoS attacks or user behavior tracking by malicious attackers. Integrating access control into GNRS can help protect user location information from unauthorized exposure while maintaining accessibility for authorized users. Therefore, the owner of the GNRS mapping sets an access policy and submits it to the GNRS. Authorized users are identified directly through attributes from the NCS, while GNRS supports various access control schemes, catering to a range of applications [22].

3.6 Privacy in the nCore Architecture

In the nCore architecture, privacy concerns are brought about by mobility, particularly location privacy. Location privacy is the capacity to conceal the geographical location of a communication endpoint (including hiding the network topology location when it might allow deductions about the node's physical location). Furthermore, other privacy aspects at the network layer are also considered, which includes sender-receiver privacy.

Analysis Model and Privacy Baseline In the nCore framework, it is significant to evaluate the privacy of communication content, the confidentiality of communication participants, and the confidentiality of location. There are three different kinds of locations that the attacker can reside in: (1) near the user (e.g. in the access network, WiFi network), (2) beyond the access network (e.g. 1 or more hops beyond the access network), and (3) near the destination server. An attacker may simply aim to identify who is accessing a specific server, a goal achievable through passive tapping near the server's access network. Meanwhile, attackers from the second category might inhabit operators' networks.

A privacy framework, Chameleon offers server-side disposable identifiers and anonymity for both sender and receiver. Drawing from the strengths and overcoming the limitations of systems like Tor, Chameleon uses a network of resolvers for nodes seeking anonymous contact, while a set of relay nodes is employed for initiating anonymous communications. Notably, the design achieves reduced cryptographic overhead compared to other low-latency anonymity systems and supports the mobility of both communication parties [23].

LAP: Lightweight Anonymity and Privacy The LAP framework is aimed at providing private and anonymous communication on the Internet [24], characterized by:

Low-stretch anonymity: This ensures packets for private communication travel through near-optimal routes to keep the increment in Autonomous Domains (ADs) low relative to the original path length.

Relaxed attacker model: A moderate level of privacy, including sender/receiver anonymity and location privacy, thus relaxing the conventional strong attacker model.

LAP is an efficient, practical network-based solution with lightweight path creation and effective communication. It enhances anonymity by hiding an end-host's topological location through two key components:

- Packet-carried forwarding state: Here, each packet carries its forwarding state, allowing ADs to decide the next hop without maintaining local per-flow state.
- Forwarding-state encryption: Unlike existing systems that decrypt/encrypt entire packets, LAP lets each AD use a secret key to encrypt/decrypt forwarding information in packet headers, keeping an AD's forwarding data hidden from all other entities.

Notably, LAP is minimal in overhead and compatible with various routing protocols, and it offers adjustable privacy levels.

4 Mobility Control Plane Protocol for UE States

A comparison between 3GPP 5G core and the nCore approach is shown, where the protocol is outlined for events like initial attach, idle-to-connected and handover.

4.1 Initial Attach

This event is triggered when the UE starts using the network for the first time. The detailed protocol exchanges over the 5G core is shown in Fig. 4. When the UE wants to start using the network, PDU session is created by a sequence of NG tunnels. This set of "pipes" connects the UE to its control functions and eventually to the data network for traffic exchange. The 5G core is tasked to establish and

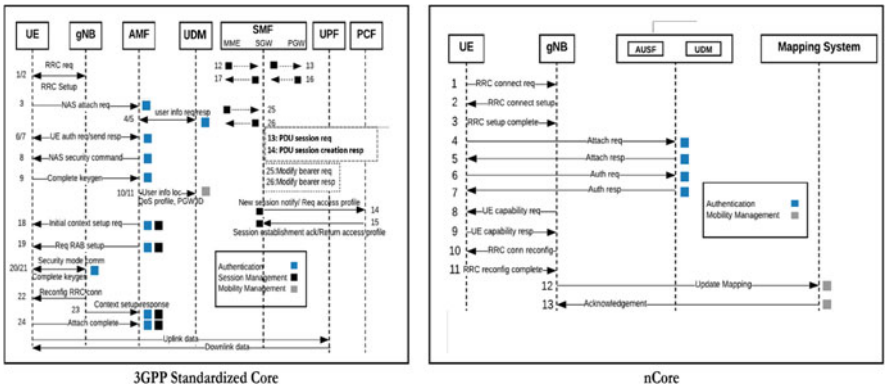


Fig. 4 Initial attach control messages for 3GPP 5G core and nCore

release the tunnels and the bearers dynamically to follow user movements and states. The control message overhead of creating and realizing the tunnels are unsustainable with the anticipated workloads in a 5G network, as discussed in [25]. nCore maintains no sessions, while RRC connection setup, authentication and UE attachment remains unchanged as shown in Fig. 4. The unified data manager (UDM) functions as a database for users' subscription management and the AMF/mobility tasks are carried out by GNRS. This architecture reduces gateway bottlenecks and the complexity of encapsulation and de-capsulation of data packets at the tunnel endpoints.

4.2 Handover

Handover in the conventional mobile core network is based on UE measurement reports. A "Handover Request" message is sent from source S-gNB to the target T-gNB. The T-gNB requests the AMF to switch paths and a tunnel is built to the target node after receiving a request from AMF. The process of handover includes approximately 20 control messages overhead to tear down and build dedicated paths which adversely affects the latency in packet forwarding [6].

In comparison, the nCore architecture realizes handover with 2× less control messages. A cell is selected for UE handover by the S-gNB, the GNRS is queried for an ID to routable locator lookup of the T-gNB and the mapping is cached at the S-gNB. The GNRS is now updated with the new UE—(T-gNB) mapping The data flow is cached in the S-gNB until the UE is connected to T-gNB. Once the mapping system is updated the UE resumes data flow both uplink and downlink. The cached data packets at the S-gNB are directed towards the UE's new point of attachment.

4.3 Idle-to-Connected

The transition from RRC-idle to RRC-connected in the 3GPP standard results in control exchanges involving as many as 17 messages. The GTP tunnel is torn down once the UE moves to idle state and rebuild when the UE transitions to active state again, adding to transmission and connection delays. The nCore design has no hierarchical gateways, hence when transitioning from idle-to-connected only the UE location has to be updated to the mapping database in case the UE has transitioned to a new location. The policy and authentication for the UE is pushed and cached at the current gNB the user device is connected to. This minimizes the number of network transactions since the proposed design does not re-establish a session every time a device wakes up to send a few bytes of data. All the session management control messages are excluded from nCore network and GUIDs along with the GNRS service support UE's mobility.

5 nCore Support for 5G Use Cases

The nCore architecture can natively support 5G services with reduced control and data plane complexity and overhead. Out of the multiple 5G use cases nCore can support, in this section, we will discuss four such use cases.

5.1 5G Mobility

Support for service continuity (uninterrupted user experience with a possible change of IP address or anchoring point) and session continuity (preserving the end-point IP address for the lifetime of the session) has become increasingly challenging to be sustained for low-latency applications in highly mobile scenarios. The current support of mobility in cellular networks (intra-network or inter-network mobility) involves numerous control message exchanges. The current support of mobility in cellular networks (intra-network or inter-network mobility) involves numerous control message exchanges. The new 3GPP spec [26] proposes a number of solutions to detect low-latency applications and fulfill their latency requirements by enhancing the existing protocols supporting UE mobility. Separation of control plane and user plane functionalities within gateways proposed in 5G allows for independent scaling of each and potentially increasing the number of user plane gateways distributed closer to the edge. However, opting for a gateway-based architecture will still incur a large amount of handover signaling between the centralized network control plane and the distributed data plane gateways. For example realizing a new feature like the addition of make-before- break to inter-RAN mobility can be initiated and controlled by the AMF, SMF and UPF from the source to the target network by exchanging a large number of control signals for breaking and creating PDU sessions. This will result in the lack of "dynamic" and "ultrafast" support for mobility due to necessary pre-configurations, increased overhead and possible bottlenecks.

The nCore architecture supports mobility natively by binding the connection to identifiers and not network addresses. The name-to-address mappings stored in the mapping service proactively and dynamically get updated by edge routers to serve the highly mobile and low-latency scenarios expected in 5G [20]. Moreover, by exploiting features like late-binding and re-binding, which are natural consequences of name-based architectures, identifiers can be (re)mapped to network addresses closer to the edge of the network. All these characteristics make the distributed flat architecture proposed in this chapter a suitable candidate for support of emerging 5G mobility scenarios.

5.2 Multihoming

Multihoming is defined as connectivity of an end device to multiple BSs and APs for improving user throughput, enabling load balancing and connectivity robustness in the scenario of mobility or disparate channel conditions. Multihoming is an important technology considered within 5G architecture for the interworking of heterogeneous wireless access technologies, i.e., Wi-Fi, LTE and NR. Dual connectivity was proposed as the first phase in the deployment of 5G, allowing for a user to be served simultaneously by an LTE BS and a NR BS [27, 28]. As a natural progression and generalization to this first phase, in [29], multi-rat dual connectivity is discussed. Another example of enabling multihoming within 5G network is LTE/NR aggregation with Wi-Fi.

An overview of the overall architecture and signaling for a baseline mobility/multihoming scenario in the current cellular network and in nCore is depicted in Fig. 5. As seen in Fig. 5 for multihoming in 5G, the Master gNB (M-gNB) can be LTE or NR BS with control plane interface to AMF, while the control signaling related to Secondary gNB (S-gNB) (which can be LTE/NR in an unlicensed band or Wi-Fi) has to be maintained by the M-gNB. This approach of anchoring a secondary RAN to a master RAN has limitations for support of dynamic mobility and session continuity. The signaling needed for a multihomed device handover is shown in Fig. 5-Multihoming in 5G, with extra signaling overhead needed due to the S-gNB's control interface being anchored on the M-gNB shown in red. As can be seen from the figure, the overhead is 2× more than a typical handover scenario. Moreover, since small-cell deployments will have smaller coverage area, handovers will be very frequent in future mobility scenarios. As a result, exploiting all the wireless capacity will require a scalable and dynamic architecture for support of multihoming. The flat name-based architecture will natively support multihoming by eliminating the need for anchors and triangular routing, and all

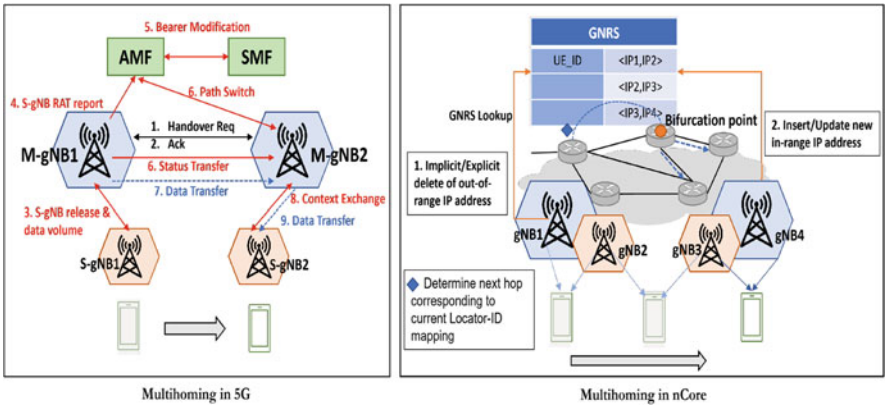


Fig. 5 Mobility and multi-homing in 3GPP 5G and nCore

the RAN components interfacing with the distributed name-to-address mapping service to provide seamless mobility and service continuity while minimizing the control overhead, and hence latency. As can be seen from Fig. 5-Multihoming in nCore, the proposed architecture greatly reduces signaling overhead, while maintaining the session/service continuity by providing uninterrupted coverage and intermittent connectivity while moving from one macro-cell to the other. All the RAN components interfaces with the GNRS mapping service to provide seamless mobility with minimal control overhead and hence latency.

5.3 Mobile Edge Computing

Mobile edge computing (MEC) is introduced in the 5G network architecture to support applications requiring computing functionality close to the end-user with low round trip latencies, such as autonomous driving, AR/VR and industrial control. Since computing is treated as an application function and traditionally located in the data network, another solution proposed in 5G is to have gateways at the edge [30]. However, this requires additional protocols in order for edge and core gateways to communicate with each other and involve AMF in case the end-user is mobile. As explained earlier, such protocol overheads will in essence take away from the application low-latency requirement offered by the edge-computing. Additionally, session handover protocols need to be designed and implemented when the user moves from one access network to another which may result in breaking of session with one gateway and making a new session with the next. Locator-ID split core network architecture has the benefit of assigning names not just to users but also to identify specific services. For example, in nCore, an edge-computing service for an AR application can be assigned a unique identifier. The service itself can be distributed across various edge locations in the RAN and an up-to-date mapping of this service to all its locations is maintained in the GNRS. Packets requiring this service are identified by the service name and anycast to the nearest service instance [31]. This simplifies the control overhead of the application as (1) it does not require maintaining (making/breaking) of sessions with one or multiple edge locations; (2) the network provider can spin up new instances of the service based on traffic demands without having to set up gateways and protocols for the new edge network; (3) the service itself is agnostic to user-mobility as the distributed routing and anycasting will forward UE requests to the closest edge server.

5.4 Roaming Architecture

To support roaming 3GPP has recognized two types of services: Local break out (LBO) and Home routed, as shown in Fig. 6a and b. Both roaming services exploit similar network architecture with different interfaces between visitor and home

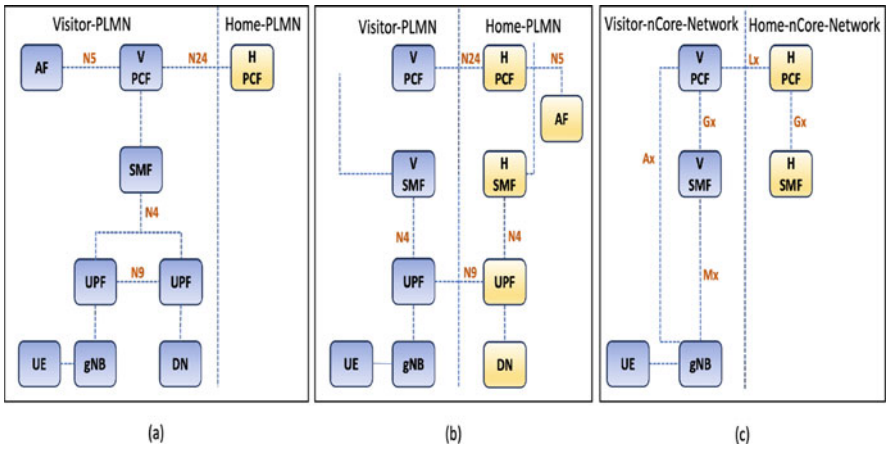


Fig. 6 Roaming Architecture: (**a**) local breakout model (**b**) home routed model; (**c**) nCore roaming model

networks. As seen in Fig. 6a, in the LBO model, the mobile user can sign up for an alternative service provider not similar to the one in Home-PLMN. The visitor network handles user requests and the visitor policy and charging function (V-PCF) regulates policy control and charging rules in agreement with the HPLMN. In the LBO roaming model, the actual services the users perceive will differ from network to network since the home network cannot exercise complete control over the user traffic, and not all HPLMN services will be available to the roaming users. When the user is served by home routed roaming scheme, Fig. 6b, the traffic is routed to the Home-public land mobile networks (PLMNs) by the visitor's UPF. The UPFs in the home network act as gateways and communicate with the policy and billing system. However, as seen in Fig. 6c, realizing roaming with nCore can rectify inefficient traffic routing and improve user experience. Additionally, an interface is provided between the UDM home and visitor network. The UDM-visitor network sends a relocation request to the UDM-home network for a visitor UE, followed by a policy check where it is determined whether the user is entitled to receive roaming services. Considering the UE qualifies for a context information transfer, all the policy and control charging (PCC) information for that particular UE is copied to the visitor UDM. The visitor network will now be able to serve the UE by exchanging only four-control messages (re-location request and response, PCC information request and response). This method supports optimal routing and bandwidth conservation in contrast to the home-routed model by serving the UE from their current point of attachment network.

6 Standalone Deployment of nCore and Compatibility with 5G Physical Layer

The proposed nCore design is agnostic to the choice of radio access technology such as LTE or 5G NR, enabling usage of Orthogonal Frequency Division Multiplexing (OFDM) for uplink and downlink, similar in spirit to 5G. The nCore architecture remains independent of how the spectrum resources are managed. Further, it assumes that mobile access will occur over a potentially diverse and heterogeneous range of access technologies that are potentially of unreliable/interruptible quality and can be addressed via delay tolerance in the core network. The proposed core architecture is compatible with dynamically managed spectral resources for seamlessly supporting varied applications. Additionally, nCore provides mix-and-match capabilities for opportunistically matching the type of radio access network (RAN) and using licensed, unlicensed, or shared spectrum to optimize the performance and efficiency of next-gen services.

nCore is capable of stand-alone deployment (SA) to leverage the distributed flat core attribute of the proposed architecture. In the SA deployment, both the UE and the base station (BS) will run the nCore distributed protocol. All the routers and BSs have an API to communicate with the GNRS. In addition, the BS is designed to set up connectivity directly with the UDM to access UE related data bypassing the AMF.

7 Prototype Evaluation of nCore

A comparative prototype evaluation of nCore with 3GPP core has been carried out using the ORBIT and COSMOS testbeds at Rutgers WINLAB [32]. The nCore is implemented using Open Air Interface (OAI) [33] and USRPs on the ORBIT\COSMOS testbed. The UE and the BS run MobilityFirst [14] protocol. Routers and gNBs in the network have an API to communicate with the name resolution service, the GNRS. The OAI implementation at the BS is modified to establish an SCTP connection with a custom policy and authentication management entity, similar in spirit to a UDM.

7.1 Network Layer Connection Establishment Latency

The idle-to-connected event in the traditional core includes authentication and connection management latencies but excludes RRC-related MAC layer protocol latencies. Figure 7 shows the cumulative distribution of 50 runs of UE moving from idle to an active state, establishing connectivity with the BS, an average latency of 21.8 ms is recorded for nCore. The same experiment is also executed with OAI

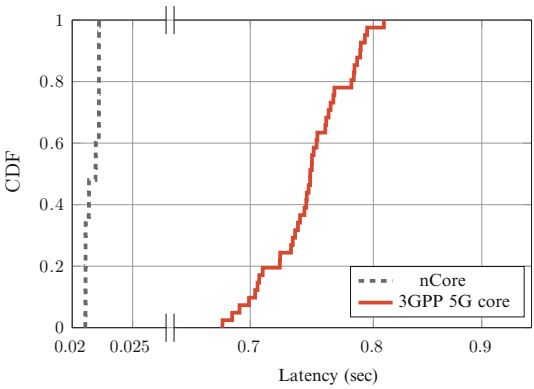


Fig. 7 Network layer latency during connection establishment

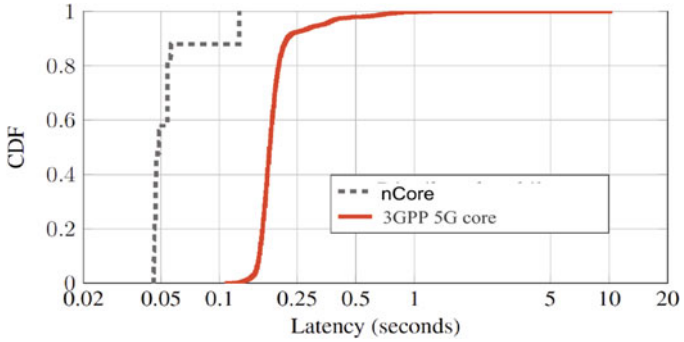


Fig. 8 Connection establishment latency in the nCore vs. a 3GPP standardized core

code running on UE and BS being attached to commercial software by Amarisoft to depict the 3GPP traditional core [34]. In this case, the average network latency goes up to 750 ms. It has been observed that the session management control message exchanges and the bearer setup event between BS, AMF and the gateways add to the latency in the core.

7.2 Overall Connection Establishment Latency

Next we compare the overall connection establishment latency of nCore vs. a commercial network. As shown in Fig. 8, the average connection establishment latency (MAC protocols as well as core network protocols included) is around 49 ms for the same OAI based prototype running on COSMOS. In order to compare it to the state-of-the-art, we parsed datasets obtained from MobileInsight for a US based cellular operator for 780 crowd-sourced UE datasets [35]. As seen from the plot, the average connection establishment latency for a 4G compatible commercial network is 181 ms.

8 Conclusion

In this chapter, a next-generation mobile core network architecture called nCore is proposed that leverages the concept of Locator-ID separation protocols and distributed mapping system for looking up the mapping between end-point-identifiers and route locators. Designing the network core based on Locator-ID separation eliminates the need for specialized gateways and complex additions to existing control-plane protocols to support emerging 5G and beyond requirements. nCore enables the network to scale organically and allows operators to deploy and adapt as per the services required. It is shown that the proposed name-based architecture can support mobility quite efficiently while natively supporting various use cases ranging from multihoming to edge cloud computing. It supports service-centric networking and minimizes network related configuration for applications, allowing fast resolution for named service instances. Prototyping and experimental validation of the proposed architecture compared with state-of-the-art 5G mobile core network architecture in terms of throughput, latency and control overhead is ongoing and will be reported in our future work.

Acknowledgments Research supported by NSF Future Internet Architecture (FIA) grant CNS-134529.

Acronyms

Abbreviation	Definition
3GPP	3rd Generation Partnership Project
SA	Standalone
MIMO	Multiple-Input Multiple-Output
LTE	Long-Term Evolution
IETF	Internet Engineering Task Force
DMM	Distributed Mobility Management
UE	User Equipment
gNB	Next Generation Node B
EID	End-point Identifier
PDN	Packet data network
SGW	Serving gateway
PGW	Packet Data Network gateway
UPF	User plane function
AMF	Access and mobility management function
SMF	Session management functions
HIP	Host Identity Protocol

Abbreviation	Definition
NDN	Named Data Networking
CDN	Content Delivery Network
GNRS	Global name resolution service
gNB	Next Generation Node B
AP	Access point
D-map	Distributed mapping
RRC	Radio Resource Connection
GUID	Globally unique identifier
NCS	Name Certificate Service
AD	Autonomous Domains
PDU	Packet Data Unit
NR	New Radio
MEC	Mobile Edge Computing
LBO	Local Break Out
PLMN	Public Land Mobile Network
UDM	Unified Data Management
PCC	Policy and Control Charging
OAI	Open Air Interface
SCTP	Stream Transmission Control Protocol

References

1. de Figueiredo FAP (2022) An overview of massive MIMO for 5G and 6G. IEEE Latin Am Trans 20(6):931–940
2. Hong W, Jiang ZH, Yu C, Hou D, Wang H, Guo C, et al (2021) The role of millimeter-wave technologies in 5G/6G wireless communications. IEEE J Microwaves 1(1):101–122
3. Ejaz W, Sharma SK, Saadat S, Naeem M, Anpalagan A, Chughtai NA (2020) A comprehensive survey on resource allocation for CRAN in 5G and beyond networks. J Netw Comput Appl 160:102638
4. Gomes NJ, Assimakopoulos P (2018, July) Optical fronthaul options for meeting 5G requirements. In: 2018 20th International conference on transparent optical networks (ICTON). IEEE, pp 1–4
5. Taleb T, Nadir Z, Flinck H, Song J (2021) Extremely interactive and low-latency services in 5G and beyond mobile systems. IEEE Commun Stand Mag 5(2):114–119
6. Mohammadkhan A, Ramakrishnan KK, Rajan AS, Maciocco C (2016, December). Cleang: A clean-slate epc architecture and controlplane protocol for next generation cellular networks. In: Proceedings of the 2016 ACM workshop on cloud-assisted networking, pp 31–36
7. Homma S, Miyasaka T, Matsushima S, Voyer D (2018) User plane protocol and architectural analysis on 3GPP 5G system. In: IETF [draft-ietf-dmm-5g-uplane-analysis]
8. Collins J (2022) GPRS tunneling protocol (GTP). In: Encyclopedia of cryptography, security and privacy, pp 1–3. Springer Berlin Heidelberg, Berlin, Heidelberg
9. Filsfils C, Camarillo P, Leddy J, Voyer D, Matsushima S, Li Z (2017) SRv6 network programming. Internet-Draft. https://scholar.google.com/scholar?q=info:QCdZXtMnvuQJ:scholar.google.com/&oi=gsb&lookup=0&hl=en

10. Zhao B, Qin Y, Yang W, Fan P, Zhou X (2022, September) SRA: Leveraging af_Xdp for programmable network functions with IPv6 segment routing. In: 2022 IEEE 47th conference on local computer networks (LCN). IEEE, pp 455–462
11. Farinacci D, Fuller V, Meyer D, Lewis D (2013) The locator/ID separation protocol (LISP) (No. rfc6830)
12. Henderson T, Vogt C, Arkko J (2017) Host mobility with the host identity protocol (No. rfc8046)
13. Zhang L, Estrin D, Burke J, Jacobson V, Thornton J.D, Smetters D.K, et al (2010) Named data networking (ndn) project. Relatório Técnico NDN-0001, Xerox Palo Alto Research Center-PARC, 157, 158
14. Raychaudhuri D, Nagaraja K, Venkataramani A (2012) Mobilityfirst: a robust and trustworthy mobility-centric architecture for the future internet. ACM SIGMOBILE Mob Comput Commun Rev 16(3):2–13
15. Hu Y, Yates RD, Raychaudhuri D (2015) A hierarchically aggregated in-network global name resolution service for the mobile internet. WINLAB, New-Brunswick, NJ
16. Xu Q, Huang J, Wang Z, Qian F, Gerber A, Mao ZM (2011) Cellular data network infrastructure characterization and implication on mobile content placement. ACM SIGMETRICS Perform Eval Rev 39(1):277–288
17. Li Y, Yuan Z, Peng C (2017, October) A control-plane perspective on reducing data access latency in LTE networks. In: Proceedings of the 23rd annual international conference on mobile computing and networking, pp 56–69
18. Cisco (2022, January 23) Cisco Annual Internet Report - Cisco Annual Internet Report (2018–2023) White Paper. Cisco. Accessed 19 Sept 2023. https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html
19. Mukherjee S, Ravindran R, Raychaudhuri D (2018, August) A distributed core network architecture for 5G systems and beyond. In: Proceedings of the 2018 workshop on networking for emerging applications and technologies, pp 33–38
20. Vu T, Baid A, Zhang Y, Nguyen TD, Fukuyama J, Martin RP, Raychaudhuri D (2012, June) Dmap: A shared hosting scheme for dynamic identifier to locator mappings in the global internet. In: 2012 IEEE 32nd international conference on distributed computing systems. IEEE, pp 698–707
21. Liu X, Trappe W, Zhang Y (2013, July) Secure name resolution for identifier-to-locator mappings in the global internet. In 2013 22nd International conference on computer communication and networks (ICCCN). IEEE, pp 1–7
22. Liu X, Trappe W, Lindqvist J (2014, September). A policy-driven approach to access control in future internet name resolution services. In: Proceedings of the 9th ACM workshop on mobility in the evolving internet architecture, pp 7–12
23. Lindqvist J, Gruteser M (2018) Privacy in MobilityFirst architecture. Accessed: March 24, 2024. [Online]. Available: http://mobilityfirst.winlab.rutgers.edu/documents/documents/Lindqvist.pdf
24. Hsiao HC, Kim THJ, Perrig A, Yamada A, Nelson SC, Gruteser M, Meng W (2012, May). LAP: Lightweight anonymity and privacy. In: 2012 IEEE symposium on security and privacy. IEEE, pp 506–520
25. Mohammadkhan A, Ramakrishnan KK, Rajan AS, Maciocco C (2016, November) Considerations for re-designing the cellular infrastructure exploiting software-based networks. In: 2016 IEEE 24th International conference on network protocols (ICNP). IEEE, pp 1–6
26. 3GPP TR 23.739 (2018) Study on enhancement of EPC for low latency communication including device mobility
27. 3GPP TS 36.300 (2018) Evolved Universal Terrestrial Radio Access (E-UTRA) and Evolved Universal Terrestrial Radio Access Network (E-UTRAN); Overall description
28. Balan T, Robu D, Sandu F (2017) Multihoming for mobile internet of multimedia things. In: Mobile Information Systems, 2017
29. 3GPP TS 38.401 (2018) NG-RAN; Architecture description

30. Akkari N, Dimitriou N (2020) Mobility management solutions for 5G networks: Architecture and services. Comput Netw 169:107082
31. Bronzino F, Maheshwari S, Seskar I, Raychaudhuri D (2019, January) Novn: named-object based virtual network architecture. In: Proceedings of the 20th international conference on distributed computing and networking, pp 90–99
32. Raychaudhuri D, Seskar I, Zussman G, Korakis T, Kilper D, Chen T, et al (2020, April) Challenge: COSMOS: A city-scale programmable testbed for experimentation with advanced wireless. In: Proceedings of the 26th annual international conference on mobile computing and networking, pp 1–13
33. Nikaein N, Marina MK, Manickam S, Dawson A, Knopp R, Bonnet C (2014) OpenAir-Interface: A flexible platform for 5G research. ACM SIGCOMM Comput Commun Rev 44(5):33–38
34. Software Company Dedicated to 4G LTE and 5G Nr (2023) Amarisoft, 7 March 2023. https://www.amarisoft.com/
35. Li Y, Peng C, Yuan Z, Li J, Deng H, Wang T (2016, October) Mobileinsight: extracting and analyzing cellular network information on smartphones. In: Proceedings of the 22nd annual international conference on mobile computing and networking, pp 202–215

Decision-Dominant Strategic Defense Against Lateral Movement for 5G Zero-Trust Multi-Domain Networks

Tao Li, Yunian Pan, and Quanyan Zhu

1 Introduction

The U.S. military has been undergoing a doctrine transition from traditional single to multi-domain operations or warfare (MDW), which the Army formally approved in October 2022 as its new warfighting doctrine [1]. The new doctrine defines MDW as "the combined arms employment of joint and Army capabilities to create and exploit relative advantages that achieve objectives, defeat enemy forces, and consolidate gains on behalf of joint force commanders," [1] which directs the service to combine and integrate air, land, sea, space, and cyberspace in all facets of operations. MDW is developed in response to the 2018 National Defense Strategy [2], shifting the previous focus of U.S. national security from addressing violent extremists worldwide to great power competition and potential conflict with near-peer adversaries across air, land, sea, space, and cyberspace.

One main impetus for this doctrine transition is the technological advances and increased complexity of modern warfare. In addition to traditional platforms such as main battle tanks and guided-missile destroyers, the rise of space, information, and artificial intelligence technologies leads to enhanced and new military capabilities, such as the Advanced Extremely High-Frequency Systems [3] powered by military satellites, the Indago quadrotor unmanned aerial systems [4], and the U.S. cyber force. By leveraging the strengths of various military capabilities across multiple domains, military forces operate through the physical dimension (air, land, sea, space), influence through the information dimension (cyberspace), and achieve victory in the human dimension.

T. Li · Y. Pan · Q. Zhu (✉)
New York University, New York, NY, USA
e-mail: tl2636@nyu.edu; yp1170@nyu.edu; qz494@nyu.edu; quanyan.zhu@nyu.edu

Y. Chen et al. (eds.), *Network Security Empowered by Artificial Intelligence*,
Advances in Information Security 107, https://doi.org/10.1007/978-3-031-53510-9_2

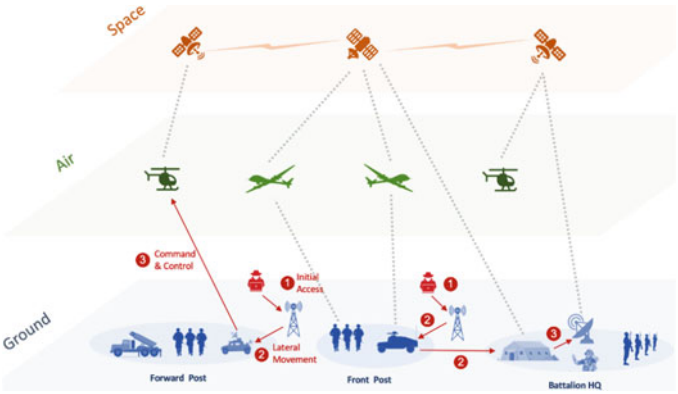


Fig. 1 An illustration of 5G Multi-Domain Networks (MDN). The army force has deployed a robust 5G communication infrastructure to facilitate seamless communication within the base and between the battalion headquarters (HQ), front lines, and forward posts. Additionally, the integration of 5G-powered satellites enables effective communication between aerial vehicles and ground forces. An APT attack can start with initial access (1), create lateral movement (2), and eventually command and control the targeted assets (3). Several paths of the attack chain are depicted, leading to the consequence of the compromise of a helicopter or misdirection to satellites

MDW involves seamless coordination and integration of forces and assets across domains to gain a competitive advantage over adversaries. For example, ground forces may work in conjunction with air and space assets to gain situational awareness, conduct precision strikes, and provide close air support. Meanwhile, naval forces may coordinate with cyberspace capabilities to disrupt an adversary's communication networks and gain information superiority. The fifth-generation (5G) wireless technology plays an important role in MDW because it provides a network infrastructure that enables faster data transfer, greater bandwidth, lower latency, and increased capacity compared to its predecessors. With 5G networks, military units across multiple domains can access and share information in real time, creating a synergistic effect that improves situational awareness and enhances command and control. Furthermore, 5G connectivity can facilitate the communication and control of unmanned and autonomous systems powered by artificial intelligence both on the ground and in the air, enabling the integration of unmanned assets into MDW. A schematic illustration of 5G networks in MDW is presented in Fig. 1

Recent years have seen the adoption and implementation of 5G networks for military applications gaining momentum. The advanced features of 5G networks, despite their contributions to coordinated MDW operations, introduce security challenges periling the efficiency and effectiveness of MDW. For example, with more devices and sensors connected to the network system, 5G networks present a larger attack surface, e.g., more potential entry points for attackers to exploit, compared to previous generations. Meanwhile, as 5G networks provide faster and more reliable connectivity, they enable more sophisticated cyberattacks, such as large-scale distributed denial-of-service attacks [5], network slicing exploitation [6], and edge computing compromise [7].

Among these cyberattacks, one critical threat is the Advanced Persistent Threat (APT). APT attacks are typically carried out by skilled and well-funded attackers who use sophisticated techniques to gain unauthorized access to sensitive information and systems. APT attackers may conduct extensive network reconnaissance to gather information about the 5G network and its vulnerabilities. They exploit vulnerabilities in the 5G network and gain unauthorized access to a device or system within the network to move laterally through the network and access other devices or systems within it. In 5G networks, lateral movement capabilities can be particularly dangerous, as they can allow attackers to gain access to critical systems and data within the network. For example, an attacker who gains access to a single device within a 5G network could potentially use lateral movement techniques to access other devices or systems, such as servers or databases containing sensitive or confidential data.

Since military assets and systems across various domains are connected and rely on 5G networks to exchange information and coordinate operations, the vulnerability of 5G networks can pose significant challenges in MDW. Therefore, military organizations shall prioritize the security of 5G networks in MDW and establish a proactive cyber defense in 5G networks. The primary objective of such a cyber defense is to disrupt the attacker's kill chain, which includes the following stages: reconnaissance, privilege escalation, exploitation, lateral movement, and command and control. Starting from an entry point, the attacker gains initial access to the network, conducts reconnaissance, stealthily navigates within the 5G infrastructure, and ultimately compromises the targeted asset, such as a drone or a satellite. Such adversarial behaviors are increasingly common in APTs.

To counteract the attacker's actions, the defender employs a sequence of defense actions known as the cyber defense chain, including monitoring, detection, response, and attribution. Figure 2 summarizes the kill and the defense chains. The relationship between the kill and the defense chains is competitive in nature. The kill chain aims to evade the detection from the cyber chain to reach the target, while the defense chain aims to thwart the attack before an adversary carries out the planned attack. To outmaneuver the adversary's decision-making cycle, a defender needs superior situational awareness together with fast and reliable reasoning capabilities, especially in unknown and uncertain situations, to make timely and effective decisions. These desiderata are also known as decision dominance. Illustrated in Fig. 2, a decision-dominant defense at the monitoring and detection stage has the capability of gathering, processing, and analyzing information from various sources to obtain a comprehensive understanding of the cyber operational environment. At the response stage, a decision-dominant defense can quickly evaluate available options, assess risks, and make informed decisions in a timely manner. As a result, it thwarts the planned attack before its execution. To achieve decision dominance, there is a need for proactive cyber mechanisms, such as cyber deception and attack engagement, to gather immediate intelligence. In addition, agility is indispensable. It allows the defender to learn, adapt, and respond to changing situations, seize opportunities, and effectively adjust strategies and tactics as required. Strategic thinking is paramount to achieving agility, involving the study of adversarial

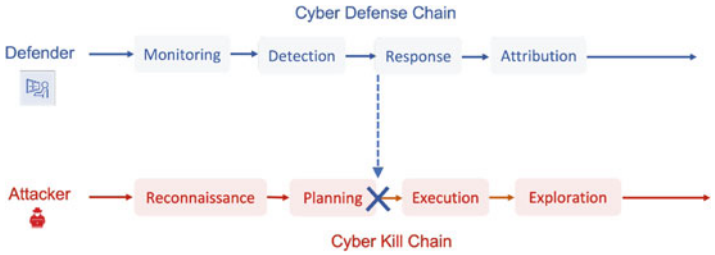


Fig. 2 An illustration of cyber kill/defense chains. The kill chain consists of crucial stages such as reconnaissance, planning, execution, and exploration. The objective of defense measures is to disrupt this kill chain by employing monitoring, detection, response, and attribution techniques. An effective defense strategy is considered decision-dominant when it efficiently acquires and processes information, enabling it to make timely decisions that outpace the attacker. For instance, the defense chain can respond swiftly to thwart the attack even before the attacker initiates the planned offensive actions

behaviors, the development of adaptive tactics, and the ability to make informed and decisive decisions.

There is a pressing need for the development of a systematic approach to establishing decision-dominant mechanisms for the defense of 5G networks. Game theory offers a promising solution in this regard. Not only does game theory naturally provide a framework for designing tactics in competitive environments, but recent advancements in dynamic games, learning theory, and their intersection with modern machine learning techniques enhance the reasoning capabilities of agents. This enables a formal and agile approach to achieving rapid decision-making. For instance, recent studies [8–11] have introduced a class of dynamic games that effectively capture the evolving interactions between defense and kill chains. The concept of non-equilibrium has been proposed to derive solution concepts based on players' behaviors. This concept holds significant implications for cybersecurity applications, particularly when the interactions between attackers and defenders may be limited and indirect.

Another significant advantage of utilizing game-theoretic models is their strong epistemic foundation, which allows for explicit modeling and analysis of scenarios involving information asymmetry and the pace of decision-making. These models find wide applicability in 5G security networks. Information asymmetry arises from the fact that neither party possesses a comprehensive view of the entire 5G network. Instead, each party gathers partial observations through reconnaissance (the attacker) or monitoring (the defender). To effectively outmaneuver the adversary, the defender must establish an information advantage by actively acquiring information during the monitoring process. This proactive approach enables the defender to gain high-confidence situational awareness of the network system and adversarial behaviors. However, it is important to note that having an information advantage alone does not necessarily guarantee the defender an upper hand in cyber defense. Another crucial aspect that holds equal importance is the pace of decision-making. The defender faces a disadvantage if the attacker manages to execute the

attack successfully before an adequate response can be mounted. In this regard, game theory frameworks provide a means to comprehensively capture the end-to-end decision-making process, encompassing information acquisition, learning, and decision-making. It provides a theoretical underpinning for understanding the fundamental tradeoff among these factors and a holistic approach to modeling and devising tactics across all stages.

One implicit assumption underlying the defense against APTs is that the attacker possesses the necessary capabilities to acquire initial access and credentials, and then establish a foothold within the network. We cannot stop the attack from getting into the network. This assumption forms the basis of the zero-trust security doctrine, which emphasizes the need to trust no entity by default and requires organizations to verify and authenticate all users, devices, and activities, regardless of their location or origin. Recognizing the importance of assuming a reasonable capability of adversaries in developing effective defenses, the concept of zero-trust doctrine can also be integrated into game models by establishing relevant adversarial models. By incorporating the principles of zero trust, game models can create decision-dominant zero-trust policies to defend against APTs in 5G networks.

To this end, we propose a decision-dominant zero-trust defense (DD-ZTD) against adversarial attacks in 5G networks in MDW to strike the right balance between information acquisition and fast decision-making. DD-ZTD is built on a game-theoretic framework that captures the information asymmetry and the competitive nature of cyber defense. Following the "never trust, always verify" principle [12], zero-trust defense (ZTD) equips the defender with a proactive information processing mechanism when operating with incomplete information about the attacker's intentions, capabilities, and actions, which is crucial to develop strategies that account for the information asymmetry.

The ZTD problem of the 5G network is modeled as an *asymmetric information Markov game* (AIMG) between the defender and the attacker. Thanks to its great expressivity, AIMG offers a comprehensive characterization of various information structures in cyber defense, which facilitates defense design in various security contexts. Furthermore, the equilibrium notion in AIMG lays a theoretical underpinning of an adaptive ZTD in the presence of information asymmetry. Powered by recent advancements in machine learning, the proposed game-theoretic ZTD framework exhibits great potential in devising a generalizable intelligent defense against a wide range of cyber attacks arising from a variety of network systems possibly unknown to the defender beforehand.

To outpace the attacker in the cyber kill chain, ZTD is further augmented by decision dominance (DD), where DD accelerates the defense decision-making in ZTD. As its name suggests, DD makes the defender the dominant player in the dynamic game by taking decisive actions based on acquired partial information with high confidence before the attacker compromises the network system, sharing the same spirit of the motto "first look, first shot, first kill" [13]. Such strategic dominance is achieved by game-theoretic calculations where the defender takes into account the attacker's decision-making process. DD amounts to an optimal stopping (Dynkin's) game problem, which essentially captures the defender's strategic anticipation of

the opponent's stopping criterion, as well as the fundamental tradeoff between the benefits and harm of lingering in the interaction, which is ubiquitous in the cyber security domain. The equilibrium notion for DD enables the defender to make opponent-independent stopping decisions based on the payoff evaluation for the underlying cyber kill chain process while making the monitoring and investigation as effective as possible.

The rest of this chapter is organized as follows. Section 2 provides an overview of multi-domain warfare and associated 5G networks across multiple domains, laying the context for further discussions. Section 3 articulates the emerging security challenges in 5G networks, particularly the advanced persistent threats (APT). To address these security issues, we propose a decision-dominant zero-trust defense for 5G networks in Sect. 4, where the game-theoretic conceptualization is presented. Sections 5 and 6 dive into the details of the zero-trust defense and the decision-dominance concept in detail, respectively, where case studies of the proposed DD-ZTD are presented.

2 Multi-Domain Warfare and 5G Networks

This section briefly overviews multi-domain warfare and the associated 5G communication networks.

2.1 Multi-Domain Warfare

Multi-domain warfare (MDW), a new operation concept designated by the U.S. Army [14], refers to the combined arms employment of military capabilities straddling multiple domains to create and exploit a decisive advantage over an adversary. Unlike traditional warfare, where operations are conducted within a single domain, MDW rests on synthesizing various military capabilities across five warfighting domains: land, sea, air, space, and cyberspace.

The backbone of MDW is the coordination and integration among different military units from multiple domains, leading to joint operations where various military services, such as the army, navy, air force, and space force, work together collaboratively. By operating across multiple domains, military forces can disrupt an adversary's operations and degrade their ability to fight.

2.2 5G Multi-Domain Networks

One challenge to achieving real-time coordination and integration in multi-domain warfare is the lack of network infrastructure to support interoperability among

military units using different communication systems, making coordinating actions across multiple domains difficult. The fifth generation (5G) wireless communication technology plays a vital role in multi-domain warfare. It provides a network infrastructure that enables faster data transfer speeds, greater bandwidth, lower latency, and increased capacity and reliability than previous generations of mobile networks. Thanks to its advanced features, 5G technology provides the foundation for faster, more connected, and more capable military operations across multiple domains, leading to improved situational awareness, enhanced command and control, precise targeting, integration of unmanned systems, and support for emerging technologies like the internet of battlefield things(IoBT). We elaborate on these aspects in the ensuing paragraphs. Figure 1 presents a schematic illustration.

Situational Awareness 5G MDN can support the transmission of large volumes of data in real time. This enables the rapid exchange of information between sensors, platforms, and command centers across different domains. Improved situational awareness allows military commanders to make more informed decisions and respond promptly to changing battlefield conditions.

Precise Targeting The low latency and high bandwidth of 5G networks enable the real-time transmission of sensor data and imagery, supporting the precise targeting of enemy assets. This enhances the effectiveness of kinetic operations, such as precision strikes, and improves the accuracy of intelligence, surveillance, and reconnaissance (ISR) capabilities.

Command and Control 5G networks can facilitate seamless communication and coordination between military units and commanders across domains. Reliable and low-latency connectivity enables the transmission of commands, orders, and mission-critical data, enhancing command and control capabilities in multi-domain operations.

Integration of Unmanned Systems and IoBT 5G connectivity can facilitate the communication and control of unmanned systems and autonomous vehicles, both on the ground and in the air. This enables the integration of unmanned assets into multi-domain operations, enhancing their situational awareness, coordination, and responsiveness. In addition, 5G connections among a massive number of devices and sensors can be leveraged to create a comprehensive network of interconnected assets. This integration allows for better monitoring, management, and control of unmanned systems, autonomous vehicles, and other IoT devices across domains.

3 Emerging Security Challenges in 5G Multi-Domain Networks

5G networks represent a significant advancement in technology, offering functionalities that set them apart from previous generations. In the context of multi-domain warfare, it is crucial to examine the vulnerabilities inherent in 5G networks, as

they can be exploited to form an APT kill chain. This section will delve into the vulnerabilities stemming from APIs, network slicing, and the supply chain.

3.1 Security of 5G Multi-Domain Networks

5G networks play an important role in MDW as they provide a network infrastructure that enables faster communication, greater bandwidth, and lower latency between different military units compared to previous generations of mobile networks. With 5G technology, military personnel can access and share information in real-time, allowing for faster decision-making and more efficient deployment of resources. For example, a military unit is conducting a mission in an urban environment that involves ground troops, drones, and surveillance equipment. The troops on the ground need to communicate with each other in real time while also receiving information from the drones and surveillance equipment to coordinate their actions.

Moreover, 5G technology allows for the use of advanced technologies such as drones, autonomous vehicles, and augmented reality, which can be used to gather intelligence, conduct surveillance, and engage in combat operations. These technologies rely on high-speed, low-latency networks to function effectively, and 5G provides the necessary infrastructure to support their deployment. For example, during the U.S. military's operations in Afghanistan, the 5G-satellite communication network was used to provide real-time communication and intelligence sharing between ground forces, aircraft, and command centers. The system enabled military forces to coordinate their actions across different domains while also providing them with the information and intelligence needed to make informed decisions.

In addition to its communication capabilities, 5G-supported satellite networks also have the ability to support other mission-critical functions, such as intelligence gathering and surveillance. The system's high-capacity communication services and advanced technology make it a critical enabler for multi-domain warfare, providing military forces with the network infrastructure needed to support real-time communication and information sharing across different domains.

The adoption and implementation of 5G networks for military applications are gaining momentum in recent years. As military forces become more reliant on 5G networks, they also become more vulnerable to cyber-attacks. To achieve multi-domain warfare, military forces need to develop robust cybersecurity measures to protect their 5G networks and systems from cyber threats. One critical threat is APT attacks on 5G networks. APT attacks are typically carried out by skilled and well-funded attackers who use sophisticated techniques to gain unauthorized access to sensitive information and systems. APT attackers may conduct extensive network reconnaissance to gather information about the 5G network and its vulnerabilities. They exploit vulnerabilities in the 5G network and gain unauthorized access to a device or system within the network to move laterally through the network and access other devices or systems within it. In 5G networks, lateral movement

capabilities can be particularly dangerous, as they can allow attackers to gain access to critical systems and data within the network. For example, an attacker who gains access to a single device within a 5G network could potentially use lateral movement techniques to gain access to other devices or systems, such as servers or databases, which contain sensitive or confidential data.

3.2 5G Threat Landscape: Vulnerabilities and Kill Chain

The emergence of 5G technology represents a significant departure from previous mobile generations, bringing with it a distinct set of security requirements. This is particularly crucial for military users who often necessitate tailored and specialized services to address their unique operational needs. There are several key threats associated with 5G networks beyond general cybersecurity threats (e.g., unauthorized access, human errors, and misconfigurations). Various threat frameworks are available to aid in analyzing these threats, such as those provided by MITRE Fight and 3GPP's Security Assurance Specifications (SCAS) and Technical Specification (TS) 33.501.

One prominent threat to 5G networks is virtualization threats, which impact virtual machine (VM) and container service platforms, affecting various aspects of 5G, including the Core, RAN, MEC, Network Slicing, Virtualization, and Orchestration and Management. These threats encompass DoS attacks, VM/container escape, side-channel attacks, and misconfigurations by cloud service consumers. For instance, extreme resource consumption by one tenant in a multi-tenant virtualization environment can lead to a DoS event for neighboring tenant systems, impeding mission functionality. Similarly, colocation attacks, such as VM/container escape or side-channel attacks, can compromise neighboring compute workloads, resulting in resource deprivation, lateral movement, and compromising data confidentiality, integrity, or availability. A side-channel attack on 5G RAN or Core functions could allow bypassing user account permissions, virtualization boundaries, or protected memory regions, thereby exposing sensitive information.

One type of threats is on 5G network slices. These threats may exploit weaknesses in the network slice's configuration, protocols, or applications, potentially leading to unauthorized access, data breaches, or service disruptions within that particular slice. To combat this threat, slice isolation is a promising approach. It involves creating and maintaining separate virtual network slices within the 5G infrastructure. By isolating slices, potential interference or vulnerabilities in one slice are contained, ensuring the integrity and security of other slices.

As 5G networks utilize application programming interfaces (APIs) for communication and interaction between different components, several potential threats can arise. These include DoS attacks targeting 5G APIs by overloading them with a high volume of requests or exploiting API vulnerabilities to exhaust system resources. Attackers can also exploit API vulnerabilities by abusing or misusing them to gain unauthorized access, manipulate data, or disrupt services. This can involve sending

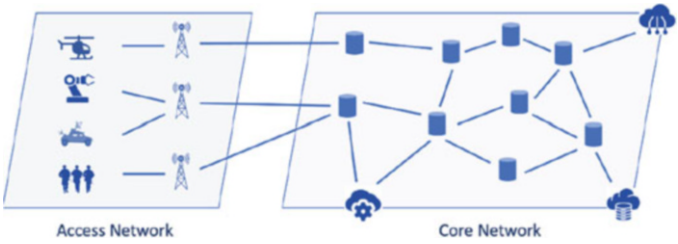


Fig. 3 An illustration of 5G network consisting of access network and core network as two major components. The core network has the functionalities of control plane and user plane separation, network functions virtualization (NFV), network slicing, mobility management, and multi-access Edge Computing (MEC)

malicious API requests, performing injection attacks, or overwhelming the API with excessive requests (API flooding).

The increasing complexity of 5G networks involves a vast ecosystem of suppliers and vendors. Security vulnerabilities in the supply chain can lead to compromised components or malicious software being introduced into the network infrastructure, posing significant risks. For example, the presence of counterfeit or substandard components in the 5G supply chain poses significant risks to network security and integrity. These components may not meet the required quality standards or security specifications, making them susceptible to exploitation and compromise. Unauthorized actors could exploit these vulnerabilities to gain unauthorized access or control over the network infrastructure, potentially leading to data breaches, service disruptions, or unauthorized surveillance.

In addition to counterfeit components, there is a risk of introducing malicious software or hardware into the 5G supply chain. This can occur through intentional modifications or the inclusion of backdoors that provide unauthorized access points. Threat actors can exploit these vulnerabilities to infiltrate the network infrastructure, compromise the confidentiality, integrity, and availability of data, or gain unauthorized control over critical network functions.

Supply chain security risks can also originate from third-party providers involved in the network deployment, such as installation contractors or maintenance service providers. Inadequate security measures implemented by these third parties, insider threats, or the compromise of their systems can introduce vulnerabilities into the 5G network. Weaknesses in the security practices of these entities can be exploited by threat actors, compromising the overall security of the network.

The combination of vulnerabilities in API, supply chain, and network slicing, along with others, can be exploited by an Advanced Persistent Threat (APT) attack to form a comprehensive kill chain. Figure 3 provides a visual representation of a baseline 5G network, where UEs utilizing O-RAN technology connect to the 5G core networks. This interconnected infrastructure presents an attack surface that an adversary can leverage to target specific entities. By capitalizing on the identified vulnerabilities, an attacker can exploit weaknesses in the API layer, infiltrate compromised components introduced through the supply chain, and exploit

insufficient isolation or monitoring within the network slicing architecture. This enables the attacker to establish a persistent presence within the network and navigate through various stages of the kill chain to reach their intended target. Figure 1 has illustrated the potential attack path an adversary may take, highlighting the entry points, lateral movement, and potential impact on the 5G network. Understanding and visualizing this attack surface assists in identifying critical areas for security enhancements and mitigations.

Zero-trust policies can be implemented to counteract such threats. It aims to establish clear rules and guidelines for access, authentication, and data protection within the network. These policies define which individuals or entities have access to specific resources, under what conditions, and the level of authorization required. It is crucial for the policy to align with the organization's security objectives and regulatory requirements. Regular monitoring of network traffic, user behavior, and access logs is essential to promptly identify any anomalies or potential security breaches. Additionally, it is important to periodically review and update the Zero Trust policy to adapt to evolving threats and changes in the network environment.

4 Decision-Dominant Zero-Trust Defense: A Game-Theoretic Framework

This section presents a high-level overview of the proposed decision-dominant zero-trust defense (DD-ZTD) in 5G multi-domain networks, arguing that the proposed game-theoretic framework leads to a unified framework for cyber defense in 5G networks.

4.1 Decision Dominance

Decision dominance refers to the ability of a defender to outmaneuver the adversary's decision-making cycle by possessing superior situational awareness and efficient reasoning capabilities. It involves making timely and effective decisions, particularly in unknown and uncertain situations, in order to gain an advantage over the attacker. To achieve decision dominance, a defense strategy needs to excel in two stages: monitoring and detection and response. In the monitoring and detection stage, a decision-dominant defense can gather, process, and analyze information from various sources to obtain a comprehensive understanding of the cyber operational environment. This enables the defender to proactively identify and assess potential threats. In the response stage, a decision-dominant defense can swiftly evaluate available options, assess risks, and make informed decisions in a timely manner. By doing so, it can effectively thwart planned attacks before they are executed. Achieving decision dominance requires proactive cyber mechanisms like cyber deception and attack engagement to gather immediate intelligence. Agility

is also crucial, allowing the defender to learn, adapt, and respond to changing situations, seize opportunities, and adjust strategies and tactics as necessary.

Zero-trust decision-dominance strategies refer to a specific type of decision-dominance strategy that operates on the assumption of the presence of adversaries at all times. These strategies are particularly critical for securing 5G networks, given the expanding attack surface and the significant number of IoT devices deployed in battlefield environments. Implementing these strategies requires strategic thinking and continuous monitoring of device behaviors to assess their trustworthiness. Timely evaluation and rapid response capabilities are essential in terms of network configuration and access control policies to counteract adversaries before they can execute their planned attacks. To ensure effective implementation, it is necessary to establish quantitative and formal frameworks that incorporate zero-trust decision-dominance into 5G network security policies. These frameworks provide a structured approach to design and enforce robust security measures that align with the principles of zero trust, enhancing the overall resilience and protection of 5G networks in dynamic threat environments.

4.2 Conceptualization of Decision-Dominant Zero-Trust Defense

One of the primary objectives of this book chapter is to develop a quantitative framework that formalizes the decision-making process for zero-trust defense. The inherent competition between attackers and defenders naturally gives rise to a dynamic game environment that reflects the win-lose nature of multi-stage interactions. To account for the information asymmetry between the players resulting from differences in monitoring and sensing capabilities, we propose a dynamic game of asymmetric information. In this game, players utilize the information available to them through the established information structure to infer unknowns. Variations in the information structure lead to differing belief structures. Players make decisions based on their beliefs, resulting in new observations in subsequent rounds of interaction and the formation of updated beliefs. It is evident that there exists interdependence between the beliefs and actions arising from the players' chosen strategies. The solution concept for the game necessitates consistency between the agents' beliefs and their optimal effort strategies. This concept gives rise to the notion of Bayesian Nash equilibrium, which serves as the foundation for developing algorithms to implement game-theoretic solutions in practical scenarios.

It is important to note that belief formation stems from incomplete information regarding the other agent. In our case, the incomplete information pertains to the behavior of the other player. Thus, it can also be seen as a process of establishing trust in the other player. This naturally aligns with the concept of zero trust, which requires the defender to distrust users or third-party players in the network despite their credentials. At the outset, the true identity must be considered unknown and untrusted, and the evaluation of a player's trustworthiness epitomizes the

principle of zero trust. The baseline equilibrium concept is established using Bayesian rationality, where Bayes' law is employed to update beliefs whenever new observations are obtained by the players. In practice, this baseline can be replaced with a machine-learning approach for inference. In modern scenarios, vast amounts of data are collected from numerous users interacting with the system. These data can be incorporated into game-theoretic models, facilitating the practical application of equilibrium solution concepts. Detailed models and their applications to lateral movements will be discussed in the subsequent section.

In order to accommodate the requirement of quick decision-making in decision-dominant scenarios, the game becomes dynamic and no longer has a fixed horizon. In this type of game, known as a stopping time game, players have the ability to choose when to cease observations and make their decision. The advantage of stopping early lies in determining the payoffs, but there is a risk of uncertainties that may lead to higher payoffs if the decision is postponed. However, it is important to note that the other player also has the capability to terminate the game. If the attacker terminates the game prematurely, the defender would be in a passive position. Thus, the competitive nature of the game naturally leads to a decision-dominant scenario. The defender's reasoning involves inferring the opponent's strategies based on the observations and, in the meantime, trades off between the probable stopping by the attacker as well as the low payoff as a result of early stopping. To formally capture this dynamic, we introduce a stopping-time game in the ensuing section, with the aim of creating decision-dominant strategies. The associated Nash equilibrium solution concept allows us to reason formally about the active and passive situations of the defender, referred to as defender dominance and adversary dominance, respectively. The baseline analysis provides insights into the necessary structures for developing winning solutions, including the payoff structures, information structures, and inference mechanisms. This analysis also establishes a theoretical foundation for understanding the fundamental limits of strategic decision dominance in the face of a strategic adversary. By integrating decision-dominant strategies with zero-trust defense strategies within the baseline framework, we can establish a symbiotic relationship between the two. Additionally, the consolidation and integration of data analytics can pave the way for the development of practical algorithms in the future.

The proposed framework in this book chapter is solidly built on the recent development of game-theoretic models for cybersecurity. Recent advances have witnessed the growth in their application to assess security risks, design protection mechanisms, and inform policy making for communication networks [15–17], Internet of things [18–20], power and energy systems [21–24], manufacturing and robotics [25–28], supply chains [29–31], and transportation networks [32–34]. Game theory has also provided theoretical foundations for cyber deception [35–38], moving target defense [39, 40], and human behaviors [5, 41, 42]. Both decision-dominance and zero-trust defense possess distinct characteristics that necessitate specific game structures to capture their essential features and provide valuable insights. In this context, our focus lies on two types of game structures: the game of asymmetric information and stopping time games. This chapter not only applies

these game structures to 5G zero-trust security problems but also contributes to a novel class of game-theoretic frameworks, pushing the boundaries of game theory forward.

Our contribution primarily revolves around the creation and analysis of stopping-time games within the framework of asymmetric information dynamic games. By incorporating asymmetric information into these games, we introduce a new dimension that enhances our understanding of strategic interactions. Furthermore, we consolidate the fields of meta-learning and explainable learning within the domain of asymmetric information games, fostering a comprehensive approach to game analysis. Through these contributions, we aim to extend the frontiers of game theory, providing researchers and practitioners with valuable tools to tackle decision-dominance and zero-trust defense challenges effectively.

5 Zero-Trust Defense

With a growing threat landscape and attack surfaces in 5G networks, traditional perimeter-based defense, a static defense mechanism, has become inadequate in the face of sophisticated cyber attacks, such as APTs. Advanced attackers can evade traditional intrusion detection at the perimeter, obtain privileges as insiders with stolen credentials, and move laterally within the network. In response to the vulnerabilities in the static defense, zero trust emerges as a promising security framework, assuming that no entities can be trusted and therefore requiring verification processes for every incoming access request [12].

Zero-trust defense (ZTD) consists of two components: trust evaluation and access policy. Square one of ZTD is to quantitatively establish the trustworthiness of each entity in the network, which is highly nontrivial in 5G networks with large-scale heterogeneous network entities. Due to the increasing network connectivity, the defender can only acquire limited partial observations of the user's trace through methods such as Intrusion Detection Systems [43], and Security Information and Event Management [44]. These limited observations create **information asymmetry**, complicating the defender's decision-making, and a quantitative metric measuring the user's trustworthiness using partial observations is indispensable.

With the trust evaluation, the defender can enforce different policies for access to network resources. What distinguishes ZTD from the perimeter-based one is that the trust evaluation and the access policy, together with the network monitoring unit, constitute a feedback loop shown in Fig. 4. As new observations are fed into the evaluation unit, the defender adjusts the trust and the access policy accordingly, leading to a dynamic defense. This section articulates a game-theoretic framework (see Definition 1) for ZTD design in 5G networks, which offers a natural set of tools to capture the information asymmetry and the competitive nature of the two parties in dynamic environments.

The proposed game-theoretic framework provides a theoretical underpinning of adaptive and strategic ZTD built upon the notion of perfect Bayesian Nash equi-

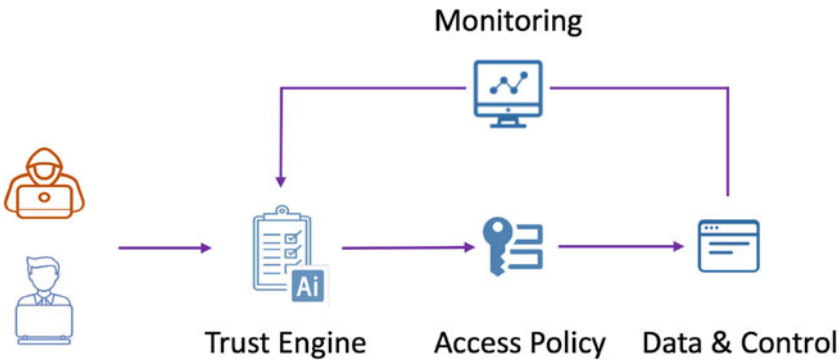


Fig. 4 An illustration of the feedback loop in zero-trust defense (ZTD) architecture. Unlike the perimeter-based defense, ZTD dynamically evaluates the trustworthiness of the user using feedback from the security monitoring system, such as SIEM [44]. Based on the trust evaluation, the access policy either grants or denies access

librium (see Definition 7) in the face of asymmetric information. This equilibrium-based ZTD can be further augmented with modern machine-learning (ML) methodologies providing an end-to-end automated network defense (see Sect. 5.3), generalizing to adversarial scenarios unseen in the pre-training stage. As advanced ML machinery enters the picture, the ZTD architecture grows opaque to human operators. To make ML-based ZTD itself trustworthy to humans, it is necessary to increase the explainability and accountability of learning-based ZTD, which is discussed at the end of this section.

5.1 Information Asymmetry in Zero-Trust Defense

As a prevailing phenomenon in security applications [45], information asymmetry refers to the fact that one party is better informed than the other party at the point of decision-making. To facilitate our discussion, we use the notion **information structure** [45] to capture the player's observations and knowledge throughout the decision-making process, which is mathematically a set of random variables whose realizations can be observed by the player [45]. We first present a bird's eye view of asymmetric information structures in the cyber defense of 5G networks, and mathematical definitions and arguments are deferred to Definition 1 and the ensuing remarks.

Compared to its predecessors, 5G networks enjoy increasing capacity and reliability that can support a massive number of heterogeneous devices. Consequently, it becomes prohibitive, if not impossible, for either the defender or the attacker to acquire a holistic view of the underlying network. The resulting information structures of both parties' partial observations display complexities to various

extents, which can be categorized according to different taxonomies. We here present two taxonomies based on the notion of information superiority proposed in [45]: one player is said to be informationally superior to the other if its information structure is a superset of its counterpart.

Depending on which party acquires the information superiority, information asymmetry includes **one-sided** and **double-sided** information asymmetry. One-sided information asymmetry refers to a situation where one party achieves information superiority over the other. If no one is informationally superior, then the resulting situation is of double-sided information asymmetry, where both parties acquire private information hidden from the other [46].

Depending on whether the information superiority is rooted in the knowledge or the observation, information structures can be categorized into **incomplete** and **imperfect** information structures. Knowledge is endogenous, reflecting the player's comprehension of the decision-making process. The incomplete information points to the player's uncertainty regarding the other's decision-making capabilities and incentives. In contrast, observation is exogenous, referring to the player's awareness of events that have previously occurred. Imperfect information refers to the situation where the player is unaware of some events in the decision-making.

As one shall see later in the running example in Sect. 5.2, the aforementioned information structures are prevalent in network defense. To systematically investigate information asymmetry in the cyber defense of 5G networks, we propose the asymmetric information dynamic games in the following, laying a mathematical foundation to facilitate ZTD design under sophisticated information structures, which is visualized in Fig. 5.

Definition 1 (Asymmetric-Information Markov Game) An asymmetric-information Markov game (AIMG) $\mathcal{G}$ is given by the following tuple

$$\mathcal{G} := \langle \mathcal{N}, \Omega, \rho, \mathcal{S}, (O_i)_{i\in\mathcal{N}}, (\mathcal{A}_i)_{i\in\mathcal{N}}, P, (u_i)_{i\in\mathcal{N}}, (\sigma_i)_{i\in\mathcal{N}}, (I_i)_{i\in\mathcal{N}}, H\rangle,$$

where the definition of each component within the tuple is as below. It is assumed every set is discrete and finite. Let $t \in \mathbb{N}_+$ be the time index.

- $\mathcal{N} = \{D, A\}$ is the decision-maker (player) set, including the defender and the attacker, denoted by D and A, respectively. For simplicity, we consider a single attacker within the network, and the generalization to the case where multiple attackers coexist is straightforward.
- Ω is the attacker's type space, and its typical element ω indicates its attack capability (e.g., stealthiness) and objective (e.g., data breach). To simplify the exposition, the normal user is also treated as one type of attacker without malicious intentions or attack capabilities.
- ρ is the type distribution over Ω, and $\rho(\omega)$ implies the probability of a certain attacker ω appearing in the network.
- $\mathcal{S}$ denotes the state space with its typical element s representing the operation status of the network.

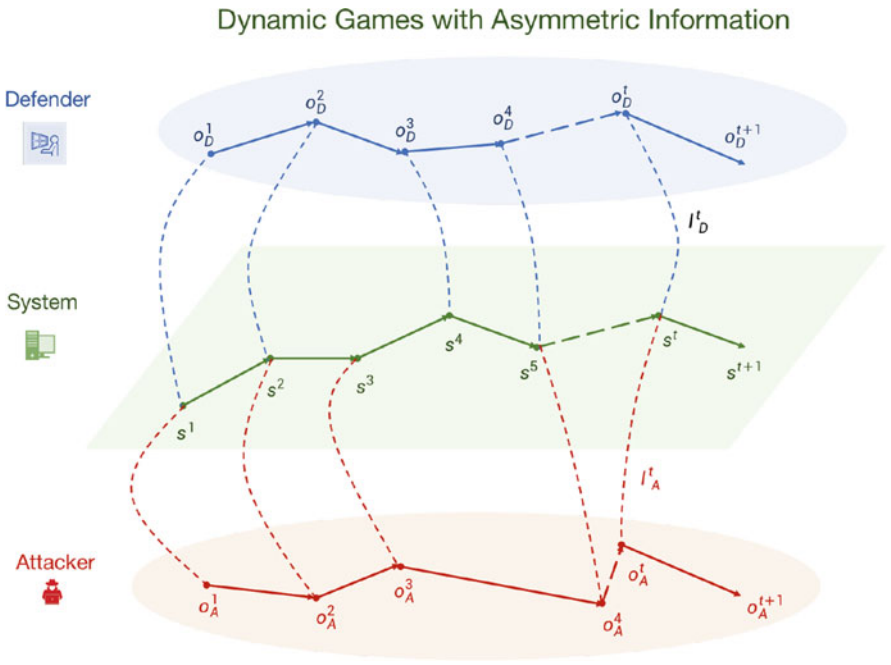


Fig. 5 An illustration of asymmetric information dynamic games defined in Definition 1. Let D and A denote the defender and the attacker, respectively. Under asymmetric information structures I_D^t and I_A^t, the two players have disparate partial observations, denoted by o_D^t, o_A^t on the system operation s^t. In zero-trust defense, the defender must infer the attacker's intention, assign trust scores, and determine the access policy based on its limited observations, which calls for efficient and adaptive trust evaluation and policy learning

- $\mathcal{O}_i$ denotes the observation space, and its typical element o_i represents the player i's partial observation.
- $\mathcal{A}_i$ is the action space of the player i.
- $P : \mathcal{S} \times (\mathcal{A}_i)_{i\in\mathcal{N}} \times \Omega \to \Delta(\mathcal{S})$ is the state transition function, depicting how the network operation evolves under the joint force of the defense and attack. To be specific, $P(s^{t+1}|s^t, a_D^t, a_A^t, \omega)$ gives the probability that s^{t+1} emerges after the two players execute a_D^t and a_A^t at the state s^t.
- $u_i : \mathcal{S} \times (\mathcal{A}_i)_{i\in\mathcal{N}} \times \Omega \to \mathbb{R}$ is the instantaneous cost of the player i.
- $\sigma_i : \mathcal{S} \times (\mathcal{A}_i)_{i\in\mathcal{N}} \times \Omega \to \Delta(\mathcal{O}_i)$ is the observation function, and $\sigma_i(o_i^t|s^t, a_D^t, a_A^t, \omega)$ denotes the probability of observing o_i^t when the underlying state is s^t.
- I_i is a set-valued mapping, characterizing the information structure of the player i throughout the Markov game. Let $\mathcal{H}^t := \{\omega, [s^k(a_i^k o_i^k)_{i\in\mathcal{N}}]_{k=1}^{t-1} s^t\}$ be the history of the gameplay up to time t, then $\mathcal{I}_i^t := I_i(\mathcal{H}^t) \subset \mathcal{H}^t$ presents the player's partial observation of the play.

- H is a constant, denoting the horizon length of the game, i.e., the operating lifetime of the network.

The AIMG unfolds as follows. In the first stage, a type-ω attacker is realized according to the distribution ρ, and the network state s^1 is initialized. At the time t, each player implements an action a_i^t from the action set $\mathcal{A}_i$ based on the information structure $\mathcal{I}_i^t$. Then, the state evolves to s^{t+1}. This procedure repeats until the game reaches the end of the horizon. The goal of type-ω attacker is to find a policy $\pi_A : \mathcal{I}_A^t \to \Delta(\mathcal{A}_A)$ within a specified policy class Π_A such that the cumulative cost is minimized:

$$\min_{\pi_A \in \Pi_A} \mathbb{E}\left[\sum_{t=1}^{H} u_A(s^t, a_D^t, a_A^t, \omega)\right], \tag{1}$$

where the expectation is taken over Borel probability measures in AIMG, including the transition P, the observation functions $(\sigma_i)_{i\in\mathcal{N}}$, and the policies $(\pi_i)_{i\in\mathcal{N}}$.

The defender's objective is more involved than (1) due to the lack of information on the attack type, and a generic characterization is given by (2), where the notations are in a similar vein of (1), except that the inner expectation $\mathbb{E}_{\omega\sim\mathcal{T}}(\cdot)$ is taken over the hidden type ω with respect to the defender's subjective belief $b^t \in \Delta(\Omega)$ based on the observations $\mathcal{I}_D^t$. Such a belief constitutes the defender's trust evaluation of the user, and a mathematical characterization is presented in Definition 2.

$$\min_{\pi_D \in \Pi_D} \mathbb{E}\left\{\sum_{t=1}^{H} \mathbb{E}_{\omega\sim b^t}[u_D(s^t, a_D^t, a_A^t, \omega)]\right\}. \tag{2}$$

Definition 2 (Trust and Trust Engine) The trustworthiness of the user at time t is defined as a probability measure over the type space $b^t \in \Delta(\Omega)$, which is determined by the defender's trust engine Φ that maps the information structure $\mathcal{I}_i^t$ to the trustworthiness $b^t = \Phi(\mathcal{I}_i^t)$. The set of beliefs $\{b^t\}_{t=1}^H \in \Delta(\Omega)^H$ is referred to as the trust evaluation.

The trust metric b we consider is a probability measure, and $b(\omega)$ depicts the defender's subjective belief over the hidden type ω, also referred to as the trust score [47]. With the trust evaluation, the defender can determine the access policy $\pi_D(\mathcal{I}_i^t, b^t)$ based on its observation, which, together with the trust engine, constitutes a zero-trust defense mechanism. A mathematical definition is given below.

Definition 3 (Zero-Trust Defense) The zero-trust defense is defined as a pair of the trust engine $\Phi : \cup_{t=1}^H \{\mathcal{I}_i^t\} \to \Delta(\Omega)$ and the access policy $\pi_D : \cup_{t=1}^H \{\mathcal{I}_i^t\} \times \Delta(\Omega) \to \Delta(\mathcal{A}_D)$.

Before elaborating on the two critical components of ZTD in Sect. 5.3, we first remark on the expressive power of AIMG in modeling the cyber defense of 5G

networks under complex information structures. In particular, Definition 1 leads to a systematic characterization of various information structures, such as one/double-sided information asymmetry and incomplete/imperfect information.

Definition 4 (One/Double-sided Information Asymmetry) The player i is said to be informationally superior than j if $\mathcal{I}_j^t \subsetneq \mathcal{I}_i^t$, for all t. This information asymmetry is one-sided since the player i is always better informed than its opponent. If there exists t such that $\mathcal{I}_i^t \setminus \mathcal{I}_j^t \neq \varnothing$ and $\mathcal{I}_j^t \setminus \mathcal{I}_i^t \neq \varnothing$, the resulting information structures are of double-sided information asymmetry. Both parties acquire private information hidden from the other, and neither achieves information superiority.

Definition 5 (Incomplete and Imperfect Information) For the player i, the AIMG is of incomplete information if $\omega \notin \mathcal{I}_i^t$ for all t. The AIMG is of imperfect information if there exists a t such that $\mathcal{I}_i^t \setminus \{\omega\} \subsetneq \mathcal{H}^t \setminus \{\omega\}$.

The following uses lateral movement in 5G networks as a running example to illustrate these information structures in ZTD, which is based on [47].

5.2 *Defending Against Lateral Movement: A Running Example*

Consider a 5G network represented by a directed graph $G = \langle V, E \rangle$, where V is the set of nodes, each of which represents a device/facilities connected to the network, and $E = \{(u, v) | u, v \in V\}$ denotes the set of edges, with each directed edge representing the stored service connection. For example, (u, v) indicates that the user visiting node u can move towards node v using stored credentials. In this example, we assume that the attacker moves laterally using stolen credentials in the 5G network, attempting to reach a sensitive target node with access to some entry node such as mobile devices. The defender aims to validate the user's authentication when accessing neighboring nodes and reject the malicious attacker. This validation can be achieved by Multi-factor Authentication (MFA) [48]. However, Each MFA over the edge incurs a cost, as MFA consumes additional security resources and time that degrade the system performance of the underlying network. The defense objective is to balance the system performance and security by strategically picking a set of edges for authentication validation.

To demonstrate the expressive power of AIMG, we formulate the above defense problem using game-theoretic language developed in Definition 1. Two decision-makers are involved in this game: the defender and the user of an uncertain type. The user's type space is binary $\Omega = \{0, 1\}$, where $\omega = 0$ indicates that the user is legitimate, whereas the user is the malicious attacker if $\omega = 1$. The type distribution ρ can be considered uniform since the two types are indistinguishable from the defender's viewpoint at the beginning. With historical data, the defender can treat the empirical frequency of malicious users as the type distribution, which reflects the defender's prior knowledge of the adversarial environment.

Suppose the attacker visits a node u at time t. Let V^t be the set of neighboring nodes that can be reached using stored credentials. Mathematically, for any $v \in V^t$, there exists a $(u, v) \in E$. Denote the collection of such edges by E^t, and the resulting subgraph $G^t = \langle V^t, E^t \rangle \subset G$ is referred to as the authentication graph. The user can easily visit any node within the authentication graph if the defender does not impose MFA on E^t. Define $L^t : V^t \to \{0, 1\}$ as the indicator function. For any $v \in V^t$, $L^t(v) = 1$ is v has been visited before time t, otherwise $L^t(v) = 0$. With a slight abuse of notation, we treat $L^t \in \{0, 1\}^{|V^t|}$ as a binary vector of time-varying dimensions.

The state variable comprises the authentication graph and the indicator, $s^t = (G^t, L^t)$, which captures the progress of the lateral movement and is fully observable to the attacker and the defender. With modern security machinery such as Intrusion Detection System (IDS) [43] and Security Information and Event Management (SIEM) [44], the trace of the user/attacker creates a sequence of events that can be used for security analysis. Consequently, the defender can acquire additional observation of the network system, which is captured by the partial observation o_D in AIMG. The security machinery producing such observation corresponds to the observation function σ_D in Definition 1. Note that the attacker's partial observation is degenerate in this case, i.e., $O_A = \varnothing$.

The action sets of the two parties are specified below. The attacker moves laterally in the network and chooses the next node to visit at each time step. Given the current state $s^t = \langle G^t, L^t \rangle$, the attacker's action set includes a collection of edges $\mathcal{A}_A := \{(u, v) | (u, v) \in E^t, L^t(u) = 1, L^t(v) = 0\}$, of which the outbound node v is to be visited. In APT, the stealthy attacker only picks one edge at each time step to evade detection. To combat the lateral movement, the defender strategically picks a subset of E^t and imposes MFA validation accordingly. Mathematically, the defense action set amounts to the power set of E^t, i.e., the set of all possible subsets of E^t, which is denoted by $\mathcal{A}_D = 2^{E^t}$.

The system evolution is determined by the joint action of both parties, where the attacker picks an edge a_A^t while the defender selects a subset of edges for MFA a_D^t. Given the current authentication graph G^t, one needs to satisfy the MFA requirements if $a_A^t = a_D^t$ before moving to the next node. It is assumed that the legitimate user ($\omega = 0$) has a higher chance to pass this MFA, while the malicious attacker is rejected. On the occasion that $a_A^t \notin a_D^t$, both types can easily move forward. The authentication graph and the visiting history shall be updated accordingly when the user/attacker reaches a new node, and this procedure repeats until the end of the horizon. The horizon length $H \in (0, \infty)$ denotes the maximum time for the attacker to operate within the network without credential renewal. The identity life-cycle lasts for H time steps, after which the stored credentials expire, and the attacker loses the foothold in the network.

The utility function captures the trade-off between operation costs resulting from authentication and system security. From the defender's stance, the cost of authentication validation over an edge is given by the scalar $c : E \to \mathbb{R}$, and the total cost of imposing MFA on a subset of edges a_D is defined as (with abuse of notation)

$c(a_D) = \sum_{e \in a_D} c(e)$. In addition to the authentication cost, system security is also a key factor in the evaluation of defense effectiveness. Denote by v^* the target node, and the indicator function $L^t(v^*)$ implies whether the target has been reached or not. Only when the malicious attacker ($\omega = 1$) visits v^*, the network system is compromised, incurring a devastating cost M. Consequently, the defender's utility depends on the hidden type and is defined below.

$$u_D(s^t, a_D^t, a_A^t, \omega) = \begin{cases} c(a_D^t), & \text{if } \omega = 0, \\ c(a_D^t) + ML^t(v^*), & \text{otherwise.} \end{cases}$$

Likewise, the attacker's utility function is also type-dependent. For the malicious attacker, passing the MFA is laborious and incurs a huge cost $-\hat{M}$. In contrast, the MFA validation is effortless. Whatever the type is, the attacker/user is rewarded by R when arriving at the target node, and they share the same transition cost $u(s^t, a_A^t)$ when navigating within the network. Using mathematical terms, the utility function is as below.

$$u_A(s^t, a_D^t, a_A^t, \omega) = \begin{cases} u(s^t, a_A^t) - RL^t(v^*), & \text{if } \omega = 0, \\ u(s^t, a_A^t) + \hat{M}\mathbb{1}_{\{a_A^t \in a_D^t\}} - RL^t(v^*), & \text{otherwise.} \end{cases}$$

5.3 Trust Evaluation and Access Policy in Zero-Trust Defense

Heretofore, our discussions have primarily concerned the theoretical underpinning of ZTD provided by the game-theoretic framework (AIMG) and AIMG's expressivity regarding information structures. This subsection shifts the focus from ZTD modeling to ZTD design, and the key message is that the game-theoretic solution concept leads to effective and automated ZTD in 5G networks.

We begin with the trust engine and trust evaluation in ZTD. Depending on its architecture, the trust engine can be categorized into attribute-based, Bayesian, and machine-learning-based trust engines. The attribute-based trust engine (ABTE) evaluates the trustworthiness of entities based on their specific attributes or characteristics. Attributes are specific properties or qualities of an entity that are relevant to determining trust, which can include factors such as the security posture of devices and endpoints, the user's location, time of access, and the sensitivity of the requested resource. The evaluation process involves assigning weights or importance to different attributes based on their significance in determining trust. These weights or importance are often pre-defined policies or algorithms, and hence, ABTE relies heavily on the domain knowledge of the security context and involves handcrafting.

The following subsections introduce another two trust engine architectures built upon Bayesian inference and machine learning, leading to automated dynamic trust evaluation capable of adapting to a variety of security scenarios. We refer to the two

Table 1 A comparison of three kinds of trust engines. Compared with ABTE, BTE, and MLTE can adapt to new scenarios without significantly resetting the engine configuration. MLTE is a data-driven trust engine that does not require a complete grasp of the domain knowledge, yet, the price to pay is that its offline pre-training needs a decent amount of data

	Domain knowledge	Offline training	Online computation	Adaptation
ABTE	✓	✗	✗	✗
BTE	✓	✗	✓	✓
MLTE	✗	✓	✓/✗	✓

trust engines as the Bayesian trust engine (BTE) and the machine-learning-based trust engine (MLTE), respectively. A summary of these trust engines is presented in Table 1.

5.3.1 Bayes Trust Engine

Definition 6 (Bayes Trust Engine) A trust engine is said to be Bayesian if the trust evaluation is produced recursively using the Bayes rule. Let $\mathcal{I}_i^{t+1} \setminus \mathcal{I}_i^t$ be the emerging information at time $t+1$, then the trust b^{t+1} is obtained by (3a) and the Bayesian update is given by (3b), where $\mathbb{P}(\mathcal{I}_i^{t+1} \setminus \mathcal{I}_i^t | \omega)$ is the probability of observing $\mathcal{I}_i^{t+1} \setminus \mathcal{I}_i^t$ conditional on the hidden type ω.

$$b^{t+1} = \Phi(\mathcal{I}_i^{t+1}) = \Phi(\mathcal{I}_i^{t+1} \setminus \mathcal{I}_i^t, b^t), \tag{3a}$$

$$b^{t+1}(\omega) = \frac{b^t(\omega)\mathbb{P}(\mathcal{I}_i^{t+1} \setminus \mathcal{I}_i^t | \omega)}{\sum_{\omega' \in \Omega} b^t(\omega')\mathbb{P}(\mathcal{I}_i^{t+1} \setminus \mathcal{I}_i^t | \omega')}. \tag{3b}$$

Using the lateral movement example in Sect. 5.2, the emerging information for the defender at time $t+1$ is $\mathcal{I}_i^{t+1} \setminus \mathcal{I}_i^t = \{a_A^t, a_D^t, s^{t+1}, o^{t+1}\}$. Given the two parties' policies π_A and π_D, the conditional probability is defined as $\mathbb{P}(a_A^t, a_D^t, o^t, s^{t+1} | \omega) = P(s^{t+1} | s^t, a_D^t, a_A^t, \omega)\sigma(o^t | s^t, a_A^t, a_D^t, \omega)\pi_D(a_D^t | s^t)\pi_A(a_A^t | s^t, \omega)$. Consequently, the belief update is obtained through the following equation.

$$b^{t+1}(\omega) = \frac{b^t(\omega)P(s^{t+1} | s^t, a_D^t, a_A^t, \omega)\sigma(o^t | s^t, a_A^t, a_D^t, \omega)\pi_A(a_A^t | s^t, \omega)}{\sum_{\omega'} b^t(\omega')P(s^{t+1} | s^t, a_D^t, a_A^t, \omega')\sigma(o^t | s^t, a_A^t, a_D^t, \omega')\pi_A(a_A^t | s^t, \omega')}. \tag{4}$$

Compared with the ATE, the BTE adapts to the online environment by processing emerging information recursively without pre-training or preparation. As a plug-and-play engine, BTE requires a decent understanding of the network operation to compute the conditional probability $\mathbb{P}(\mathcal{I}_i^{t+1} \setminus \mathcal{I}_i^t)$, including the system transition P, the security monitoring machinery σ, and the attacker's strategy π_A.

Several remarks are in order on the practicability of BTE. Except for the anticipated strategy π_A, the system transition function P and the observation function σ are readily accessible to the defender. In the lateral movement example, the system transition is deterministic: if one edge (u, v) is picked, the next node must be the head node v, and the associated authentication graph and the indicator are determined accordingly. Consider the IDS as the observation function. The corresponding observation space is binary $O_D = \{0, 1\}$, where 0 means no alarm is raised while 1 indicates that a security alert is signaled, warning the defender that the user is more likely to be malicious. In this case, $\sigma(o^t = 1|s^t, a^t, a_D^t, \omega = 1)$ is the detection rate, and $\sigma(o^t = 1|s^t, a^t, a_D^t, \omega = 0)$ is the false alarm rate, both of which are included in the IDS configuration revealed to the defender. As one can see from (4), the attacker's strategy π_A is involved in the Bayesian update, even though it is explicitly included in the information structure $\mathcal{I}_D^t$. Due to the predictive nature of equilibrium in game theory, the defender is able to derive the attacker's optimal strategy using the game tuple in Definition 1, from which the attacker has no incentive to deviate. Using plain words, the defender can anticipate the attacker's strategy π_A and use this predicted strategy the update the trust. Section 5.3.3 elaborates on this equilibrium notion in detail, where we articulate the close connection between BTE and Bayesian Nash equilibrium in game theory, leading to an adaptive zero-trust defense in contrast to ATE.

One computational hurdle of BTE lies in that the denominator in (3b) is given by an integration (summation) of the conditional probability $\mathbb{P}(\mathcal{I}_i^{t+1} \setminus \mathcal{I}_i^t|\omega')$ with respect to the trust $b^t(\omega')$. As the arms race between the defender and the attacker heats up, the attack techniques develop day and night, and consequently, the number of attack types grows astronomical. As a result, the trust evaluation process in the online execution is burdened with great computation overhead, causing authentication latency in ZTD.

In addition to the computation overhead, another limitation of BTE is that it relies heavily on the domain knowledge of the underlying network. Take the lateral movement defense as an example. The observation function σ corresponds to a network security machinery (e.g., SIEM) that monitors the attacker's activities and reports incidents to network operators. Note that such feedback from the security machinery may not be directly applicable in BTE on some occasions since mathematically σ needs to be a conditional probability measure in BTE as shown in (4). For example, if the observation variable $o \in O$ is a log message or an audit trail of the network system, then one needs to infer the attack type distribution behind these security events, requiring certain expertise in network security.

5.3.2 Machine Learning Trust Engine

To address these limitations of BTE, one alternative approach is to utilize machine learning methodologies, which offer an **end-to-end** trust evaluation. The machine-learning-based trust engine undergoes an offline training process before the online execution, and no heavy computation is involved in the online phase, although

lightweight model updates can happen on some occasions to adapt the machine-learning model to new security scenarios [47]. Powered by recent advancements in large language models [49] and other related deep learning architectures [50, 51], ML models capable of processing multi-modal inputs (texts and audio, etc.) display great potential in creating end-to-end trust evaluation that maps the raw system log files to a trust metric without much human involvement. Compared with BTE, MLTE does not require domain knowledge or online computation, yet the price to pay is the pre-training process, and collecting high-quality training data can be cumbersome. This is because the training data shall include incidence reports, system logs, and other related log messages, which often contain sensitive information regarding the network systems, and hence they are not open-sourced. Even if they are, these data come from a specific scenario, and the resulting trust engine may not generalize well to other network defense problems.

Despite its limitations, MLTE provides a data-driven trust evaluation that is suitable for large-scale complex 5G networks. Mathematically, MLTE performs a statistical inference task where the engine infers the hidden type using the observations. The following takes variational Bayes inference (VB) as an example to illustrate how to train and deploy an inference network as the trust engine. In statistical inference, VB refers to a family of techniques in Bayesian inference for approximating the posterior probability of unobserved variables (e.g., hidden types) conditional on the observed ones (e.g., those in the $\mathcal{I}_i^t$). We pick VB because of its close connection with BTE and wide applications in machine learning problems, such as variational autoencoders, which gives rise to many off-the-shelf ML toolsets readily available to network security practitioners. We refer the reader to [51] for more details on statistical inference and its applications.

For simplicity, we drop the time index in the information structure and use $\mathcal{I}$ in the following discussion. Adopting a probabilistic viewpoint, we consider $\mathcal{I}$ and ω as two random variables generated by some random process. The process consists of two steps: (1) a realization ω is generated from the prior ρ; (2) a realization $\mathcal{I}$ is generated from a conditional distribution $\mathbb{P}(\mathcal{I}|\omega)$, which is in a similar vein as (3b). The goal of the inference task is to derive the posterior distribution $\mathbb{P}(\omega|\mathcal{I})$ characterized by the Bayesian rule: $\mathbb{P}(\omega|\mathcal{I}) = \mathbb{P}(\mathcal{I}|\omega)\rho(\omega)/\int \mathbb{P}(\mathcal{I}|\omega)\rho(\omega)d\omega$. Similar to the computation issue in BTE, the integral is intractable.

Denote by $q_\phi(\omega|\mathcal{I})$ a neural network (with parameter $\phi \in \mathbb{R}^n$) approximation to the true posterior $\mathbb{P}(\omega|\mathcal{I})$. Taking inspiration from the evidence lower bound (ELBO) method [51], we derive a loss function for the training purpose whose minimizer $q_{\phi^*}(\omega|\mathcal{I})$ serves as the trust engine in ZTD. Given a realization $\mathcal{I}$, its marginal likelihood can be written as

$$\log \mathbb{P}(\mathcal{I}) = D_{KL}[q_\phi(\omega|\mathcal{I})||\mathbb{P}(\omega|\mathcal{I})] + \mathcal{L}(\phi; \mathcal{I}), \tag{5}$$

where $\mathcal{L}(\phi; \mathcal{I}) = \log \mathbb{P}(\mathcal{I}) - D_{KL}[q_\phi(\omega|\mathcal{I})||\mathbb{P}(\omega|\mathcal{I})]$. $D_{KL}[q_\phi(\omega|\mathcal{I})||\mathbb{P}(\omega|\mathcal{I})] := \mathbb{E}_{q_\phi(\omega|\mathcal{I})}[\log(q_\phi(\omega|\mathcal{I})/\mathbb{P}(\omega|\mathcal{I}))]$ is the KL divergence between the two distributions. The intuition behind this likelihood expression is that the KL divergence $D_{KL}[q_\phi(\omega|\mathcal{I})||\mathbb{P}(\omega|\mathcal{I})]$ in (5) measures the discrepancy between the true posterior

$\mathbb{P}(\omega|\mathcal{I})$ and its neural network approximation $q_\phi(\omega|\mathcal{I})$, which is to be minimized. From (5), minimizing the KL term is equivalent to maximizing $\mathcal{L}(\phi; \mathcal{I})$. Since the KL term is non-negative, $\mathcal{L}(\phi; \mathcal{I})$ lower bounds the log-likelihood on the left-hand side, which is referred to as the evidence (or variational) lower bound.

Compared with the KL term $D_{KL}[q_\phi(\omega|\mathcal{I})||\mathbb{P}(\omega|\mathcal{I})]$, this lower bound, rewritten as below, does not explicitly involve the posterior distribution $\mathbb{P}(\omega|\mathcal{I})$. The rest of this subsection is devoted to the stochastic optimization problem $\max_\phi \mathcal{L}(\phi; \mathcal{I})$, which amounts to the pre-training of MLTE.

$$\begin{aligned}\mathcal{L}(\phi; \mathcal{I}) &= \log \mathbb{P}(\mathcal{I}) - D_{KL}[q_\phi(\omega|\mathcal{I})||\mathbb{P}(\omega|\mathcal{I})] \\ &= \log \mathbb{P}(\mathcal{I}) - \mathbb{E}_{q_\phi(\omega|\mathcal{I})}[\log q_\phi(\omega|\mathcal{I}) - \log \mathbb{P}(\omega|\mathcal{I})] \\ &= \mathbb{E}_{q_\phi(\omega|\mathcal{I})}[\log \mathbb{P}(\mathcal{I})] - \mathbb{E}_{q_\phi(\omega|\mathcal{I})}[\log q_\phi(\omega|\mathcal{I}) - \log \mathbb{P}(\omega|\mathcal{I})] \\ &= \mathbb{E}_{q_\phi(\omega|\mathcal{I})}[-\log q_\phi(\omega|\mathcal{I}) + \log \mathbb{P}(\mathcal{I}, \omega)]. \end{aligned} \tag{6}$$

Consider some dataset $\mathcal{D} := \{\mathcal{I}^{(k)}\}_{k=1}^K$ consisting of K independently identically distributed (i.i.d.) sample observations under random attack types $\omega^{(k)}$ drew from $\rho(\cdot)$. $\mathcal{I}^{(k)}$ represents historical security incidence reports during the network operation, and the superscript (k) denotes the sample index rather than the time step. Note that only the dataset $\mathcal{D}$ is available in training, whereas the variable $\omega^{(k)}$ remains hidden (the prior ρ is known), as often witnessed in real-world scenarios.

In addition to the inference network $q_\phi(\omega|\mathcal{I})$, we introduce a generative network $p_\theta(\mathcal{I}|\omega)$, $\theta \in \mathbb{R}^m$, which approximates the conditional probability $\mathbb{P}(\mathcal{I}|\omega)$. Consequently, the joint distribution $\mathbb{P}(\mathcal{I}, \omega)$ in (6) can also be parameterized: $\mathbb{P}(\mathcal{I}, \omega) = \rho(\omega)\, p_\theta(\mathcal{I}|\omega)$. With a slight abuse of notation, we denote such parameterization by $p_\theta(\mathcal{I}, \omega)$. Similar to our argument in justifying the use of π_A in (4), $p_\theta(\mathcal{I}|\omega)$ can be interpreted as the defender's conjecture of the attack strategy that eventually leads to the resulting observation $\mathcal{I}$. With this additional parameterization, the lower bound under the datapoint $\mathcal{I}^{(k)}$ becomes

$$\mathcal{L}(\phi, \theta; \mathcal{I}^{(k)}) = \mathbb{E}_{q_\phi(\omega|\mathcal{I}^{(k)})}[-\log q_\phi(\omega|\mathcal{I}^{(k)}) + \log p_\theta(\mathcal{I}^{(k)}, \omega)]. \tag{7}$$

The remaining task is simply to approximate the gradient of the expectation in (7) using samples and to apply stochastic gradient descent. Note that the expectation is taken with respect to the hidden variable ω conditional on $\mathcal{I}^k$. Hence, one needs to first draw a batch of M samples $\{\omega^{(k,l)}\}_{l=1}^M$ from q_ϕ, and then compute the gradient estimators

$$\begin{aligned}\widehat{\nabla}_\phi \mathcal{L}(\phi, \theta; \mathcal{I}^{(k)}) = &-\frac{1}{M}\sum_{l=1}^K \log q_\phi(\omega^{(k,l)}|\mathcal{I}^{(k)}) \nabla_\phi \log q_\phi(\omega^{(k,l)}|\mathcal{I}^{(k)}) \\ &+ \frac{1}{M}\sum_{l=1}^K \log p_\theta(\mathcal{I}^{(k)}, \omega^{(k,l)}) \nabla_\phi \log q_\phi(\omega^{(k,l)}|\mathcal{I}^{(k)}).\end{aligned} \tag{8a}$$

$$\widehat{\nabla}_\theta \mathcal{L}(\phi, \theta; \mathcal{I}^{(k)}) = \frac{1}{M} \sum_{l=1}^{K} \nabla_\theta \log p_\theta(\mathcal{I}^{(k)}, \omega^{(k,l)}). \tag{8b}$$

The first gradient estimation in (8a) rests on a Monte Carlo (MC) estimation trick detailed below. The key message of this trick is that the gradient of an expectation can be expressed as an expectation of another gradient, which can be approximated using Monte Carlo sampling. Suppose, for the time being, one needs to estimate the gradient $\nabla_\phi \mathbb{E}_{q_\phi(\omega)}[f(\omega)]$ where $\mathcal{I}$ is suppressed, and $f(\omega)$ is an arbitrary function. Rewriting the gradient term in the integral form, we obtain

$$\begin{aligned}
\nabla_\phi \mathbb{E}_{q_\phi(\omega)}[f(\omega)] &= \nabla_\phi \int f(\omega) q_\phi(\omega) d\omega \\
&= \int f(\omega) \nabla_\phi q_\phi(\omega) d\omega \\
&= \int f(\omega) \frac{\nabla_\phi q_\phi(\omega)}{q_\phi(\omega)} q_\phi(\omega) d\omega \\
&= \int f(\omega) \nabla_\phi \log q_\phi(\omega) q_\phi(\omega) d\omega \\
&= \mathbb{E}_{q_\phi(\omega)}[f(\omega) \nabla_\phi \log q_\phi(\omega)].
\end{aligned} \tag{9}$$

Therefore, the MC estimation under K samples $\{\omega^{(l)}\}_{l=1}^K$, denoted by $\widehat{\nabla}_\phi$, is given by $\widehat{\nabla}_\phi = 1/K \sum_{l=1}^K f(\omega^{(l)}) \nabla_\phi \log q_\phi(\omega^{(l)})$.

We apply this trick to derive the first gradient estimation. As one can see from the (10), the gradient $\nabla_\phi \mathcal{L}(\phi, \theta; \mathcal{I}^{(k)})$ comprises three terms.

$$\begin{aligned}
&\nabla_\phi \mathcal{L}(\phi, \theta; \mathcal{I}^{(k)}) \\
&= \nabla_\phi \mathbb{E}_{q_\phi(\omega|\mathcal{I}^{(k)})}[-\log q_\phi(\omega|\mathcal{I}^{(k)}) + \log p_\theta(\mathcal{I}^{(k)}, \omega)] \\
&= \nabla_\phi \int \Big(-\log q_\phi(\omega|\mathcal{I}^{(k)}) + \log p_\theta(\mathcal{I}^{(k)}, \omega)\Big) q_\phi(\omega|\mathcal{I}^{(k)}) d\omega \\
&= -\underbrace{\int \nabla_\phi \log q_\phi(\omega|\mathcal{I}^{(k)}) q_\phi(\omega|\mathcal{I}^{(k)}) d\omega}_{①} - \underbrace{\int \log q_\phi(\omega|\mathcal{I}^{(k)}) \nabla_\phi q_\phi(\omega|\mathcal{I}^{(k)}) d\omega}_{②} \\
&+ \underbrace{\int \log p_\theta(\mathcal{I}^{(k)}, \omega) \nabla_\phi q_\phi(\omega|\mathcal{I}^{(k)}) d\omega}_{③}.
\end{aligned} \tag{10}$$

Since $\nabla_\phi \log q_\phi(\omega|\mathcal{I}^{(k)}) = \nabla_\phi q_\phi(\omega|\mathcal{I}^{(k)}) / q_\phi(\omega|\mathcal{I}^{(k)})$, ①$= \nabla_\phi \int q_\phi(\omega|\mathcal{I}^{(k)}) d\omega = 0$. Applying the trick to the second and third terms, we arrive at the following equations.

$$② = \mathbb{E}_{q_\phi(\omega|\mathcal{I}^{(k)})}[\log q_\phi(\omega|\mathcal{I}^{(k)}) \nabla_\phi \log q_\phi(\omega|\mathcal{I}^{(k)})],$$

$$③ = \mathbb{E}_{q_\phi(\omega|\mathcal{I}^{(k)})}[\log p_\theta(\mathcal{I}^{(k)}, \omega) \nabla_\phi \log q_\phi(\omega|\mathcal{I}^{(k)})].$$

Replacing all the expectations in ①, ②, and ③, one obtains the MC estimation in (8a). It should be noted that such MC estimation, though intuitive and straightforward, suffers from high variance [52]. One effective remedy is the reparameterization technique [51], and the key idea is that one can express the random variable as $\omega = g_\phi(\varepsilon, \mathcal{I})$ (reparameterization), where ε is an auxiliary variable with independent marginal $p(\varepsilon)$. When generating $\omega^{(k,l)}$, one follows the procedure: $\varepsilon^{(l)} \sim p(\varepsilon)$ and $\omega^{(k,l)} = g_\phi(\varepsilon^{(l)}, \mathcal{I}^{(k)})$. For example, when $\omega \sim \mathcal{N}(\mu, \Sigma^2)$ (univariate Gaussian with mean μ and variance Σ), a simple reparameterization is $\omega = \mu + \Sigma\varepsilon$, $\varepsilon \sim \mathcal{N}(0, 1)$. Since this parameterization is beyond the scope of this chapter, we refer the reader to [51] for more details on the reparameterization in VB.

5.3.3 Optimal Access Policy: Approximation and Learning

With the trust evaluation process discussed above, we are ready to articulate the access policy π_D in ZTD. To simplify our exposition, we take BTE as the underlying trust engine, and our argument also applies to other kinds of trust engines. Recall that the defender's goal is to minimize the objective function $\min_{\pi_D \in \Pi_D} \mathbb{E}\left\{\sum_{t=1}^{H} \mathbb{E}_{\omega \sim b^t}[u_D(s^t, a_D^t, a_A^t, \omega)]\right\}$. With a slight abuse of notation, let $u_D(s^t, a_D^t, a_A^t, b^t) = \mathbb{E}_{\omega \sim b^t}[u_D(s^t, a_D^t, a_A^t, \omega)]$ be the expected utility under the trust b^t. Before articulating how to solve the optimal policy, we first address the solution concept in AIMG, i.e., what is the optimality criterion in this multi-agent decision-making?

In general, what distinguishes a game problem from a single-agent optimization is that players' optimization problems are entangled. In AIMG, the defender's problem is given by $\min_{\pi_D \in \Pi_D} \mathbb{E}[\sum_{t=1}^{H} u_D(s^t, a_A^t, a_D^t, b^t)]$, where the attacker's actions a_A^t are involved. To see this more clearly, we expand the expectation expression, and the defender's problem becomes

$$\min_{\pi_D \in \Pi_D} \mathbb{E}_{\pi_D, \pi_A, P, \sigma}\left[\sum_{t=1}^{H} u_D(s^t, a_A^t, a_D^t, b^t)\right]. \tag{11}$$

Hence, when the defender determines the access policy, it must take the attacker's move into account and vice versa. From our early argument in BTE, one can view the defender's optimal policy as the minimizer to (11) under the anticipated attacker's strategy π_A^*, i.e.,

$$\pi_D^* \in \arg\min \mathbb{E}_{\pi_D, \pi_A^*, P, \sigma}\left[\sum_{t=1}^{H} u_D(s^t, a_A^t, a_D^t, b^t)\right]. \tag{12}$$

Then, the remaining question is how to derive such anticipation. From Nash's seminal work [53], one guiding principle is the unilateral deviation principle, which states that π_A^* is a rational anticipation of the attacker's move if the player has no incentive to unilaterally deviate from such strategy, i.e., π_A^* solves the minimization problem in (13). The pair (π_D^*, π_A^*), given by (12) and (13), constitutes a Nash equilibrium of the AIMG. A formal definition is presented in Definition 7.

$$\pi_A^* \in \arg\min \mathbb{E}_{\pi_D^*, \pi_A, P, \sigma} \left[\sum_{t=1}^{H} u_A(s^t, a_A^t, a_D^t, \omega) \right]. \tag{13}$$

Definition 7 (Perfect Bayesian Nash Equilibrium) Consider the information-asymmetric game with the objectives of the attacker and the defender defined by (11), (12), and (13). A triple of $\langle \pi_D^*, \pi_A^*, \{b^t\}_{t=1}^H \rangle$ is said to be the perfect Bayesian Nash equilibrium of this game if it satisfies

$$\pi_D^*(\cdot|s^t, b^t) \in \arg\min \mathbb{E}_{\pi_D, \pi_A^*, P, \sigma}[\sum_{\tau=t}^{H} u_D(s^\tau, a_A^\tau, a_D^\tau, b^\tau)], \text{ for any } t \in [H], \tag{P1}$$

$$\pi_A^*(\cdot|s^t) \in \arg\min \mathbb{E}_{\pi_D^*, \pi_A, P, \sigma}[\sum_{\tau=t}^{H} u_A(s^\tau, a_A^\tau, a_D^\tau, \omega)], \text{ for any } t \in [H], \tag{P2}$$

$$b^{t+1}(\omega) = \begin{cases} \frac{b^t(\omega)\mathbb{P}(\mathcal{I}_D^{t+1} \backslash \mathcal{I}_D^t | \omega)}{\sum_{\omega' \in \Omega} b^t(\omega')\mathbb{P}(\mathcal{I}_D^{t+1} \backslash \mathcal{I}_D^t | \omega')} & \text{if } \mathcal{I}_D^{t+1} \text{is realizable,} \\ \text{an arbitrary probability distribution,} & \text{otherwise.} \end{cases} \tag{C1}$$

$\mathcal{I}_D^t$ is realizable if there exists ω such that the conditional probability $\mathbb{P}(\mathcal{I}_D^{t+1} \backslash \mathcal{I}_D^t | \omega)$ is strictly greater than zero.

In Definition 7, (P1) and (P2) are refinements of (12) and (13), respectively. When $t = 1$, the refinements coincide with (12) and (13), leading to a Nash equilibrium. What makes the refinements "perfect" is that the arg min equations hold for any $t \in [H]$. (P1) and (P2) are referred to as the perfectness conditions in game theory [54], meaning that either player has the incentive to deviate from the equilibrium strategy no matter when (time index t) and where (the state s^t and belief b^t) they start to play AIMG. Finally, the equilibrium in Definition 7 is called Bayesian since the belief is generated in a Bayesian manner. (C1) is referred to as the consistency condition: the belief update shall be compatible with the strategy since π_A^* is involved in the Bayesian update, see (4). In summary, this perfect Bayesian Nash equilibrium (PBNE) is the solution concept considered in the rest of this chapter, and the optimal access policy refers to the equilibrium strategy π_D^* in PBNE.

Solving generic PBNE analytically remains largely an open question, even though recent breakthroughs have shed light on the two-stage Markov game case where the PBNE conditions are rephrased using bilevel-bilinear programming [55].

The rest of this subsection is devoted to the numerical approximation of PBNE. Similar to solving single-agent Markov decision processes where computational methods can be divided into value-based [56, 57] and policy-based [58, 59] approaches, the computation of PBNE (approximately) also follows either value-based, i.e., first approximating the expected utility in (P1) and (P2), or policy-based ones, i.e., searching for the policy directly. The following presents two representative algorithms from the two categories, respectively.

Belief-Value Iteration We begin with the value-based approach. Recall that the perfectness conditions (P1) and (P2) are an extension of Bellman's principle of optimality [60] to the multi-agent setting. Naturally, one can transplant the value iteration algorithm [60] in dynamic programming to AIMG. However, value iteration operates using backward induction, whereas the belief update is a forward process (Bayesian update). Consequently, one cannot update the value function (i.e., the expected utility) and the belief simultaneously.

A variant of value iteration is proposed in [11] to address the conflict between the value function update and the belief update. The gist is that the updates are performed alternatively: updating the value while fixing the belief and vice versa. We refer to such alternative belief/value updates as belief-value iteration (BVI). Denote by $\mathcal{G}(s, b, u_D, u_A)$ the stage game at the state s under the belief b, where the utility functions are $u_D(s, a_A, a_D, b)$ and $u_A(s, a_A, a_D, \omega)$, $\omega \in \Omega$. Let $\texttt{BayesNash}[\mathcal{G}(s, b, u_D, u_A)]$ be the Bayesian Nash equilibrium operator that takes in the stage game utilities and outputs the equilibrium payoffs $(u_D^*, u_A^*) = \texttt{BayesNash}[\mathcal{G}(s, b, u_D, u_A)]$. The equilibrium payoffs (u_D^*, u_A^*) correspond to the minimum in (P1) and (P2), respectively, with the summations inside the expectations are replaced by the stage game utilities. Mathematically, this equilibrium operator is characterized by bilinear programming [11, 55].

The BVI starts with a belief system initialization $\{b^{(t,0)}\}_{t=1}^H$. For the k-th iteration, BVI first fixes the belief system $\{b^{(t,k)}\}_{t=1}^H$. The k-th value iteration is given by the backward induction below. For $t = H, H-1, \ldots, 1$,

$$\begin{aligned} V_D^{(t,k)}(s, b^{(t,k)}), V_A^{(t,k)}(s) &= \texttt{BayesNash}[\mathcal{G}^{(t,k)}(s, b^{(t,k)})], \\ \mathcal{G}^{(H,k)}(s, b) &= \mathcal{G}(s, b, u_D, u_A), \\ \mathcal{G}^{(t,k)}(s, b) &= \mathcal{G}(s, b, u_D + V_D^{(t+1,k)}, u_A + V_A^{(t+1,k)}), \end{aligned} \tag{VI}$$

where $\mathcal{G}^{(t,k)}$ is referred to as the subgame starting from time t during the k-th iteration, bearing the same spirit of the term "cost-to-go" in MDP [60]. The utility function in this subgame is defined in (14). The attacker's utility $u_A + V_A$ can be defined similarly. We remark that by applying the equilibrium operator BayesNash in (VI), the perfectness conditions in Definition 7 are satisfied, and $V_D^{(H,k)}$ and $V_A^{(H,k)}$ returned by (VI) are the equilibrium payoffs of the two players, respectively, under the belief system $\{b^{t,k}\}_{t=1}^H$.

$$u_D + V_D^{(t+1,k)}(s, a_A, a_D, b^{(t,k)}) = u_D(s, a_A, a_D, b^{(t,k)}) + \mathbb{E}_{s' \sim P}[V_D^{(t+1,k)}(s', b^{(t+1,k)})]. \tag{14}$$

Given the value functions, the defender's and the attacker's policies can be determined accordingly by solving $\mathcal{G}^{(t,k)}$, and we denote the resulting policies by π_D^k and π_A^k, respectively. To complete the k-th iteration, one needs to update the belief system according to the Bayes rule in (4), which is referred to as belief iteration (BI) in this context shown in (BI). This belief iteration guarantees the consistency between the policies π_D^k, π_A^k and the belief systems $\{b^{(t,k+1)}\}_{t=1}^H$, as mandated by (C1).

$$b^{(t+1,k+1)}(\omega) = \frac{b^{(t,k)}(\omega)\mathbb{P}_{\pi_D,\pi_A}(s^{t+1}|s^t, \omega)}{\sum_{\omega'} b^{(t,k)}(\omega')\mathbb{P}_{\pi_D,\pi_A}(s^{t+1}|s^t, \omega')}, b^{(1,k+1)}(\omega) = \rho(\omega). \tag{BI}$$

This interleaved procedure repeats until no significant improvement is observed in the updated value functions. Even though intuitive, BVI does not offer any convergence guarantees since the operator `BayesNash` in general is not a contraction mapping [61]. Even assuming it is, we note that the introduction of (BI) further complicates the analysis, and it remains unclear whether the combination of (VI) and (BI) is a contraction mapping. Yet, it is safe to conclude that shall BVI converge, the resulting policies and the belief system must be a PBNE.

Policy Gradient We now shift the focus from the value-based approach to the policy-based one. For simplicity, we fix the attacker's policy in the sequel and present the policy gradient method [58] in reinforcement learning. The key message is that the defender's optimal policy can be learned from sample trajectories using stochastic gradient descent. Consider the defender's problem in (15) where the attacker's strategy is fixed and suppressed.

$$\min_{\pi_D \in \Pi_D} V_D := \mathbb{E}_{\pi_D,P,\sigma}\left[\sum_{t=1}^H u_D(s^t, a_A^t, a_D^t, b^t)\right]. \tag{15}$$

Suppose the policy is parameterized by a neural network $\pi_D(\phi), \phi \in \mathbb{R}^n$. Then, one can search for the optimal policy through gradient descent, i.e., $\phi \leftarrow \phi - \nabla V_D(\phi)$ (the learning rate is suppressed). $\nabla V_D(\phi) = \nabla \mathbb{E}_{\pi_D(\phi),P,\sigma}[\sum_{t=1}^H u_D(s^t, a_A^t, a_D^t, b^t)]$. Recall the MC estimation trick in (9), we rewrite the gradient as in (16), referred to as the policy gradient.

$$\nabla V_D(\phi) = \mathbb{E}_{\pi_D(\phi),P,\sigma}\left[\nabla \log \pi_D(\phi) \sum_{t=1}^H u_D(s^t, a_A^t, a_D^t, b^t)\right]. \tag{16}$$

Denote a sample trajectory under the policy $\pi_D(\phi)$ (in short, ϕ) by $\ell(\phi) := \{s^1, a_A^1, a_D^1, u_A^1, u_D^1, o^1, \ldots, s^H, a_A^H, a_D^H, u_A^H, u_D^H, o^H\}$, where $u_D^t =$

$u_D(s^t, a_A^t, a_D^t, b^t)$, b^t is derived using the Bayes rule in (4). Then, an unbiased estimate of $\nabla V_D(\phi)$, denoted by $\widehat{\nabla} V(\phi)$ is constructed as $\widehat{\nabla} V(\phi) = \nabla \log \pi_D(\phi) \sum_{t=1}^{H} u_D^t$. Denote by $u_D(\ell) = \sum_{t=1}^{H} u_D^t$ the empirical return of the sample trajectory. One common practice to reduce the variance of the MC estimate $\widehat{\nabla} V(\phi)$ is to collect a batch of trajectories $\{\ell^{(k)}\}_{k=1}^{K}$ and take the average: $\widehat{\nabla} V(\phi) = 1/K \sum_{k=1}^{K} \nabla \log \pi_D(\phi) u_D(\ell^{(k)})$. Starting from an initialization ϕ^0, one need first implement the policy $\pi_D(\phi^0)$ in a simulated network system [62] and collect a batch of trajectories $\{\ell^{(k)}\}_{k=1}^{K}$. Then, the policy is updated using the policy gradient discussed above. The procedure repeats until the parameter ϕ^k stabilizes. Since policy gradient is a first-order method, it is only guaranteed to converge to the first-order stationary point where $\nabla V_D(\phi) = 0$. Even though this first-order point may not be the exact equilibrium point, it often leads to satisfying defense policy, as observed in the literature [63].

5.4 *Generalizability, Explainability, and Accountability of Learning-Based Zero-Trust Defense*

5.4.1 Reinforcement Learning and Explainable Defense

Even though RL leads to a theoretically guaranteed approach to learning the ZTD policy, the missing part is that the learned policy, i.e., the model weights of the neural network, remains a black box and is difficult for human operators to comprehend. The explainability of RL (XRL), as an emerging field devoted to casting light on the inner workings of RL agents, has gained momentum across various research communities. Since XRL is still in its infancy, there is no consensus over the exact definitions of explainability, and most of the current endeavors try to explain the actions of RL agents [64]. Following this line of research, we discuss the explainability of the optimal access policy learned by RL in the following, which addresses the question:

How does the RL policy grant or deny access based on the trust evaluation?

Our XRL approach exploits the mathematical structure of the AIMG and utilizes non-parametric policy learning, i.e., the RL policy is expressed in closed form without involving neural networks [47, 65]. Hence, our XRL study is more aligned with the interpretability of the RL policy, indicating that the intrinsic logic of the defense mechanism is transparent and easy to understand rather than a post-hoc property.

The gist of the explainability in ZTD is that the optimal policy is of a threshold form [65]. Consider the lateral movement case in Sect. 5.2 as an example, where the type space and the defense action space are binary: $\Omega = \{0, 1\}$ (0-legitimate user, 1-attacker) and $\mathcal{A}_D = \{0, 1\}$ (0-active defense, 1- inactive). In this example, the belief b resides in the two-dimensional probability simplex, which can be uniquely determined by its entry $b(0)$. We refer to $b(0) \in [0, 1]$ as the trust score, implying

the likelihood of the user is legitimate. A threshold policy $\pi_D(b)$ is defined in (17), and the threshold is given by τ. As its name suggests, the defense remains idle as long as the trust score is above the threshold, while it is activated once the trust score is below the critical value.

$$\pi_D(b) = \begin{cases} 0, & 0 \leq b(0) \leq \tau, \\ 1, & \tau < b(0) \leq 1. \end{cases} \tag{17}$$

The advantage of this threshold policy is self-evident: it is a white box clearly displaying how the trust evaluation is utilized. The same policy gradient method presented above also applies to the learning of thresholds. Even though the gradient $\nabla_\tau \pi_D$ does not acquire a closed form, one can leverage the simultaneous perturbation stochastic approximation (SPSA) to estimate the gradient [47, 65]. The threshold form in (17) also extends to the finite-action case, where $|\mathcal{A}_D| - 1$ threshold values partition the interval [0, 1] into $|\mathcal{A}_D|$ subintervals (the type space is still binary).

5.4.2 Meta-Learning and Generalizable Defense

The limitation of the threshold policies is concerned with generalization ability. The optimal policy (or equivalently, threshold) trained in one network setup cannot deal with another scenario where the system vulnerabilities are different from the training setup. To facilitate our discussion, denote by $\theta \in \Theta$ the network system configuration that can affect the system transition P (or the observation function σ) under this configuration. Using the notations in Definition 1, the defender now faces a family of games, and the transition function P_θ of each game is parameterized by θ subject to a distribution $p(\theta)$. We refer to each game under parameter θ as an attack scenario. The policy trained for the scenario θ does not generalize well to θ', leading to ineffective ZTD.

To equip ZTD with generalizability under information asymmetry, a scenario-agnostic ZTD (SA-ZTD) is proposed in [47], creating a generalizable ZTD capable of handling new attack scenarios unseen in the training phase. SA-ZTD rests on meta-learning, an emerging learning paradigm that aims to learn a learning strategy using training data [66]. In the face of a new scenario unseen in the training phase, the obtained learning strategy enables the defender to learn a new defense on the fly using far fewer data than from scratch. This idea of defending on the fly is also explored in adversarial machine learning leading to impressive defense performance [67]. Since real-world applications involve a large (possibly infinite) number of attack scenarios, it is intractable to learn the optimal policy for each scenario. Powered by meta-learning, SA-ZTD uses only a handful of known scenarios, more precisely, sample trajectories from these scenarios. Hence, the word "agnostic," whose root means "not known," is used to emphasize that the adaptation ability is acquired without knowledge of the network configuration of every scenario.

Two pillars of SA-ZTD are the meta policy π_{meta} and the adaptation mapping $\Psi : \Pi_D \times \Theta \to \Pi_D$. The adaptation mapping corresponds to the learning strategy mentioned earlier that adapts the meta policy to a new defense $\Psi(\pi_{meta}, \theta)$ when facing a new scenario θ. A formal definition of SA-ZTD is given in [47], which we restate in Definition 8.

Definition 8 (SA-ZTD) A pair $\langle \pi_{meta}, \Psi \rangle$ is said to be a scenario-agnostic zero-trust defense (SA-ZTD) with respect to a scenario distribution $p \in \Delta(\Theta)$ if the pair solves for the minimization problem

$$\min_{\pi, \Psi} \mathbb{E}_{\theta \sim p}[V_D(\Psi(\pi, \theta))]. \tag{18}$$

Similar to empirical risk minimization (ERM) [68, 69], a solution to (18) is obtained by solving the sample average approximation:

$$(\pi_{meta}, \Psi) \in \arg\min \frac{1}{|\widehat{\Theta}|} \sum_{\theta \in \widehat{\Theta}} V_D(\Psi(\pi, \theta)), \tag{19}$$

where $\widehat{\Theta} \subset \Theta$ is a finite collection of scenarios i.i.d. sampled from $p \in \Delta(\Theta)$. The term "agnostic" points to the fact that the exact scenario distribution p is usually unknown in security practice and often replaced by an empirical distribution provided by security datasets, such as the data from MITRE ATT&CK [70] considered in [47]. In summary, the training of SA-ZTD does not explicitly require the domain knowledge of each attack scenario, such as the system configuration and the observation functions.

Since the function class $\{\Psi | \Psi : \Pi_D \times \Theta \to \Pi_D\}$ is infinite-dimensional, directly seeking an adaptation mapping through (18) [or (19)] is intractable. One remedy is to restrict the focus to the parameterization class where the mapping is parameterized by $\gamma \in \mathbb{R}^n$, $n \in \mathbb{Z}_+$. For example, Ψ_γ can be parameterized by recurrent neural networks, where γ is the model weights and the optimal adaptation is determined by training algorithms [71]. Another well-accepted parameterization is the gradient-based adaptation: $\Psi_\gamma(\pi, \theta) := \pi - \gamma \nabla V_D(\pi)$, and γ is the gradient step size to be optimized [72].

To arrive at an explainable SA-ZTD, one can pick the gradient-based adaptation, as it naturally applies to the non-parametric threshold policies discussed in Sect. 5.4.1. To be consistent with previous notations, we replace π with τ whenever speaking of threshold policies, where the τ denotes the threshold value. The minimization problem in (18) turns into

$$\min_{\tau \in [0,1]} \mathbb{E}_{\theta \sim p}[V_D(\text{Proj}_{[0,1]}\{\tau - \gamma \nabla V_D\})]. \tag{20}$$

The resulting meta policy, as the minimizer to (20), takes the threshold form that is explainable to human operators, increasing the accessibility and transparency of learning-based ZTD. As argued in [47], the policy gradient method is still applicable

to (20). Even though the computation expenditure in SA-ZTD is higher than the vanilla RL policy in (15), the meta policy can adapt to a variety of new scenarios without training from scratch.

5.4.3 Accountability

The accountability of machine-learning-based ZTD (ML-ZTD) refers to the responsibility and answerability of those involved in the design, development, deployment, and use of machine learning or artificial intelligence technologies in general. Accountability aims to ensure that ML-ZTD is developed and utilized in a manner that is ethical, transparent, and fair. What distinguishes accountability of ZTD in 5G networks from other AI systems is the focus on accountability in system engineering, which encompasses three key aspects: responsibility, detectability, and attribution.

Responsibility Accountability rests on the acknowledgment that individuals and organizations involved in ML-ZTD development and deployment have responsibility for the ZTD's behavior and impact on the network system. Specifically, this responsibility revolves around the question of whether each component involved in ZTD architecture, such as the security machinery, the trust engine, and the access policy, contributes to an ethical, transparent, and fair operation in the network. To be more precise, this responsibility provides compliance requirements and failure standards for each component.

Detectability Responsibility gives the rule book, and the next question to address is whether ZTD operation violates the compliance requirements. Mathematically, the detectability question pertains to statistical inference, such as hypothesis testing and VB methods, where one infers the ground truth (violation) from collected data. Yet, ZTD in 5G networks is a game problem, see Definition 1, where the strategic decision-maker can evade the detection, which must be taken into account when inspecting the ZTD operation. Game theory naturally provides a system-science viewpoint on the detectability question in multi-agent systems, where the incentives, capabilities, and private information of the investigator and the investigatee can be captured through the AIMG in Definition 1. This game-theoretic viewpoint leads to a strategic detection framework.

Attribution No node is an island in large-scale complex 5G networks, and one failing node or component may spur a chain reaction over the network and the ZTD system. When facing a cascading failure in the network defense, one needs to identify the root cause and upgrade the ZTD accordingly. One shall not confuse detection with attribution, even though both of them aim to identify the malfunctioning part of the ZTD and the network system. However, detection addresses the question "where it is", whereas attribution focuses on "why it is such." Mathematically, attribution amounts to a causal inference task [59], where the casual relationship among random variables is established using data.

6 Decision-Dominance Defense

While ZTD provides us with a comprehensive framework for trust evaluation and access policy, the networked entities still face multi-stage persistent cyber threats. Therefore, it is crucial to adopt an integrated defense approach that recognizes the intrinsic value of the cyber defense chain and the fundamental principles of zero trust. Decision dominance defense (D^3), which conceptualizes the interactions of cyber defense/kill chain as a stochastic process, forms the backbone of the holistic defense mechanism, with zero trust defense acting as a critical component at every stage. By treating the cyber defense chain as a dynamic system, we acknowledge the unpredictable nature of cyber threats and the need for proactive decision-making based on real-time information. By incorporating zero trust principles throughout this process, from initial access controls to ongoing monitoring and incident response, we create a robust and resilient defense model that embraces uncertainty, eliminates blind spots, and ensures continuous protection against the relentless onslaught of cyber threats.

Understanding the intricacies of an attack is crucial for developing effective defense strategies. A traditional Lockheed Martin Kill Chain [73, 74] usually outlines seven distinct stages that malicious actors typically follow. These stages include Reconnaissance, where attackers gather information on potential targets; Weaponization, where they create malicious tools or payloads; Delivery, the method through which the attack is transmitted; Exploit, where vulnerabilities are leveraged to gain access; Installation, the establishment of a foothold within the target system; Command & Control, the creation of communication channels for remote control; and finally, Actions on Objectives, where the attacker achieves their intended goals within the compromised system. Comprehensively analyzing and understanding each stage of the Kill Chain requires the defender to effectively engage with adversaries while minimizing the time it takes for an attack to unfold. A proactive cyber defense chain (e.g., [75, 76]) aims to disrupt and curtail the attacker's progress at each stage of the Kill Chain, reducing their opportunity to inflict significant damage. D^3 integrates real-time threat intelligence, advanced analytics, and rapid response mechanisms, including monitoring, detection, response, and attribution, maximizing the abilities to mitigate and neutralize the threats, actively impeding the attacker's progress and shortening the overall time it takes for an attack to materialize. It empowers the 5G network defender to take a more active role in their defense, enabling them to stay one step ahead of the adversary and significantly enhance their resilience against evolving cyber threats.

The essence of D^3 is the critical timing of cutting off the cyber kill/defense chain. In MWD scenarios, while the general concept of understanding, deciding, acting, and assessing fast still holds (i.e., strangling the threats in its cradle), one must take the real-time warfare conditions and game-theoretic thinking into consideration, "knowing oneself and knowing the enemy". Therefore, in the sequel, we formalize D^3 as a Dynkin's type of optimal stopping game acting on a Markov chain of multi-stage cyber-attacks/defense [77], and characterize the equilibrium strategy between

the two competitive parties. While our model is built upon ZTD components, the notations should not be confused with the previous section.

6.1 D^3 as Dynkin's Game

By convention, let $(\Omega, \mathcal{F}, \mathbb{P})$ be the probability space. Denote the time index during a lifecycle of the interactions between the cyber kill/defense chain by $t = 1, \ldots, T$. Let $(X_t)_{0 \le t \le T}$ be a Markov process modeling the cyber threats, living in space $(\mathcal{X}, \mathcal{G})$, and are adapted to the filtration $\mathbb{F} = (\mathcal{F}_t)_{0 \le t \le T}$ with transition kernel $\mathcal{P}$. The Markovian state captures the identifiable elements in the system, e.g., it can represent the Structured Threat Information eXpression language (STIX) that facilitates this effort [77]. The collection of STIX-type data requires active interactions between the two parties.

We are given three payoff functions $\phi, \zeta, \psi : \mathcal{X} \to \mathbb{R}$ that capture the cyber risk given system states, where from the defender's perspective, (the attacker's perspective would be the opposite)

1. ϕ is the early termination payoff, which is activated when the cyber defender actively terminates the persistent monitoring/detection and resets the system credential before the malicious operations, including data exfiltration, denial of service, and delivery of ransomware, etc. are executed;
2. ψ is the late response payoff, which is activated when the cyber defender responds to the data exploitation and command & control actions without summarizing the monitoring/detection phase.
3. ζ is the confrontation payoff, which is activated when both parties have extracted information through lateral movement/monitoring and engaging, etc., and perform attack/defense actions at the same stages.

It is reasonable to assume that $\min(\psi, \phi) \le \zeta \le \max(\psi, \phi)$, since the confrontation often happens when attackers and defenders both have neutralized assessments for the system, it should sit in between the worst and best payoffs.

Here, for simplicity, we first consider the case where the information is symmetrical between the network operator/defender and the attacker, i.e., both parties have access to the state and utility information. However, this formalism shall not exclude the cases where the information is asymmetric and/or the utility functions are unknown/uncertain to one of the parties.

On top of the lower-level cyber threats/defense operations, we define stopping times $\tau, \sigma : \Omega \to \{0, \ldots, T\}$ to capture the termination decisions for both parties. We assume that both the attacker and the defender have access to the system state X_t, τ, σ are $\mathbb{F}$-measurable. Denote the set of $\mathbb{F}$-stopping times by $\mathcal{T} := \{0 \le \tau \le T : \{\tau(\omega) \le k\} \in \mathcal{F}_k \ \forall k \in [T], \forall \omega \in \Omega\}$. Moreover, we expect there to be a $2^{[T]}/\mathcal{G}$-measurable map $\tau : \mathcal{X} \to [T]$, where $[T] = \{0, \ldots, T\}$, such that the defender/attacker will make termination decisions based on the information extracted from X_t, without awareness of each other's stopping decisions.

For stopping times τ, σ, the value/cost function for the defender/attacker is defined as:

$$V^{\tau,\sigma}(x) = \mathbb{E}_x[H(\tau,\sigma)] = \mathbb{E}_x\left[\phi(X_\tau)\mathbb{1}_{\{\tau<\sigma\}} + \psi(X_\sigma)\mathbb{1}_{\{\tau>\sigma\}} + \zeta(X_\tau)\mathbb{1}_{\{\tau=\sigma\}}\right], \tag{21}$$

where $H(\tau,\sigma) : \mathcal{T}\times\mathcal{T}\times\Omega \to \mathbb{R}$ is the random payoff of stopping strategies τ and σ, $\mathbb{E}_x$ is the conditional expectation operator with respect to the transition kernel $\mathcal{P}_x$, i.e., there exists a semi-group $\mathcal{T}$ such that for any $\mathcal{B}(\mathbb{R})/\mathcal{G}$-measurable function g and $t = 0, \ldots, T$,

$$\mathcal{T}^t g(x) := \mathbb{E}_x[g(X_t)] = \underbrace{\int_{\mathcal{X}} \cdots \int_{\mathcal{X}}}_{t \text{ times}} g(x_t) d\mathcal{P}_{x_{t-1}}(x_t) \ldots d\mathcal{P}_x(x_1).$$

In practice, the convolutional integral is hard to compute directly. Instead, we can leverage sampling methods such as Markov Chain Monte-Carlo (MCMC) to approximate the conditional expectation.

Now we are ready to formulate the game.

Definition 9 (Decision Dominance Game) A tuple $(\mathcal{X}, \mathcal{P}, \phi, \zeta, \psi, \mathcal{T})$ encapsulates a Decision Dominance Game (DDG) if it satisfies the following:

- there exists a Markov process $(X_t)_{0\le t\le T}$ that lives in $(\mathcal{X}, \mathcal{G})$ with transition kernel $\mathcal{P}$, which can be extracted as cyber threats information;
- ϕ, ζ, and ψ are payoff functions mapping from X_t to $\mathbb{R}$, $\phi, \zeta, \psi \in \mathcal{E}(\mathcal{X})$, which is the set of all bounded $\mathcal{B}(\mathbb{R})/\mathcal{G}$-measurable functions on $(\mathcal{X}, \mathcal{G})$. Further, $\min(\phi, \psi) \le \zeta \le \max(\phi, \psi)$ on $\mathcal{X}$;
- at each stage t, both parties pick a stopping strategy from space $\mathcal{T}_t := \{t \le \tau \le T : \{\tau(\omega) \le k\} \in \mathcal{F}_k \ \forall k \in [T], \forall \omega \in \Omega\}$ to decide whether to stop or continue the kill/defense chain.
- at each stage the utility function of the defender is

$$H(\tau_t, \sigma_t) = \phi(X_{\tau_t})\mathbb{1}_{\{\tau_t<\sigma_t\}} + \zeta(X_{\tau_t})\mathbb{1}_{\{\tau_t=\sigma_t\}} + \psi(X_{\sigma_t})\mathbb{1}_{\{\tau_t>\sigma_t\}},$$

  while the attacker attains $-H(\tau_t, \sigma_t)$.

Figure 6 gives an example of the DDG outcome. The solution concept of a DDG is given in Definition 10.

Definition 10 (Decision-Dominance Equilibrium (DDE)) A pair of stopping time strategies $(\tau^*, \sigma^*) \in \mathcal{T}\times\mathcal{T}$ is a Decision-Dominance Equilibrium (DDE) if for all initial state $x \in \mathcal{X}$, it satisfies the minimax condition:

$$\begin{aligned} V^{\tau^*,\sigma^*}(x) &= \operatorname{ess\,sup}_{\sigma\in\mathcal{T}} \operatorname{ess\,inf}_{\tau\in\mathcal{T}} V^{\tau,\sigma}(x) \\ &= \operatorname{ess\,inf}_{\tau\in\mathcal{T}} \operatorname{ess\,sup}_{\sigma\in\mathcal{T}} V^{\tau,\sigma}(x). \end{aligned} \tag{22}$$

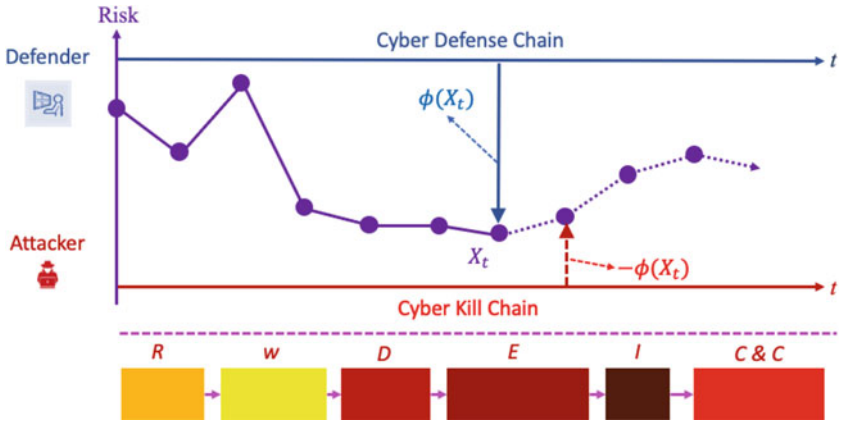


Fig. 6 An illustration of the cyber kill/defense chain interaction. In this case, at time t, the system state has evolved into X_t, the defender cuts off the chain interaction earlier than the attacker and gets payoff $\phi(X_t)$, while the attacker gets $-\phi(X_t)$ since she plans to stop at the next time step

The existence of such a value function, however, is a non-trivial question, as we are looking for a pure strategy Nash equilibria in an infinite-dimensional space ($\mathcal{T} \times \mathcal{T}$), Von-Neumann's Minimax theorem does not apply here. However, under certain conditions, we are able to show that a DDG with information symmetry always admits a value function, which is unique up to a state-wise constant translation.

We know from Dynkin's result [78] that when $\phi \leq \zeta \leq \psi$ on $x \in \mathcal{X}$, there exists a value process

$$\begin{aligned} V_t &= \min\{\psi(X_t), \max\{\phi(X_t), \mathbb{E}[V_{t+1}|\mathcal{F}_t]\}\} \\ &= \max\{\phi(X_t), \min\{\psi(X_t), \mathbb{E}[V_{t+1}|\mathcal{F}_t]\}\}, \end{aligned} \tag{23}$$

and the equilibrium strategies capture V_t's hitting times of the upper/lower limits. However, the ordered-payoff assumption is hard to verify in the context of MDW, a more reasonable assumption, as has been discussed before, is $\min(\phi, \psi) \leq \zeta \leq \max(\phi, \psi)$ on $\mathcal{X}$. In addition, the ubiquitous information asymmetry in cyberspace oftentimes makes the derived equilibrium strategies inapplicable.

Therefore, in the sequel, we dive into the more general case defined as in Definition 9, and lay out some essential analytical characterization for the equilibrium value process; further, we give a rough description for the case under information asymmetry.

6.2 Equilibrium Strategies for D^3

In this section, we investigate the existence and characterization of the DDE in two different cases under a symmetric information structure and then discuss an

extension. The first case is when the early termination payoff ϕ dominants the late termination payoff ψ, which we call *adversarial dominance*, as in this case, the outcome of engaging in the long term favors the adversary. The second case is called *defense dominance*, where the late termination payoff ψ dominates the early termination payoff ϕ. Hence, the defender is able to endure the kill/defense chain interactions longer than the adversary does.

6.2.1 Case I: Adversarial Dominance

Under the Adversarial Dominance Condition (ADC), the payoff functions satisfy the ordered condition $\psi \leq \zeta \leq \phi$ for all system states $x \in \mathcal{X}$. In this case, at any state $x \in \mathcal{X}$, the defender aims to investigate the kill chain for a proper period of time while trying to terminate the operations faster than the attacker, as it is more costly to wait for the attacker to exploit the vulnerabilities by doing Command & Control than to shut down the service and reset the credentials. This is also called *first-mover advantage*, that is, the defender has the incentive to end the game faster than the opponent.

We shall proceed with the analysis by giving a constructive sequence of equilibrium values. To this end, we investigate the $t \in [T]$ stage problem through backward induction and let $\{V_n^t\}_{n=0}^t$ be the equilibrium processes attained by stopping at no more stage t. At $t \in [T]$, both parties have to choose confrontation, thus at the final stage, the payoff is $\zeta(X_t)$; at $n \in [t-1]$. They either both stop and get payoff value $\zeta(X_n)$, or wait for the next round, in which case the defender has to judge if the termination values $\phi(X_n)$ is higher than the expected engaging values $\mathbb{E}\left[V_{n+1}^t|\mathcal{F}_n\right]$, given that the attacker chooses to engage. Mathematically, we have the value processes for arbitrary $t \in [T]$,

$$\begin{aligned} V_t^t &= \zeta(X_t), \\ V_n^t &= \mathrm{val}\begin{bmatrix} \zeta(X_n) & \phi(X_n) \\ \psi(X_n) & \mathbb{E}\left[V_{n+1}^t|\mathcal{F}_n\right] \end{bmatrix}, \qquad \text{for } n = t-1, \ldots, 0. \end{aligned} \tag{24}$$

where val$(\cdot)$ stands for a special value operator of the matrix game, which we interpret as:

$$V_n^t = \begin{cases} \mathbb{E}\left[V_{n+1}^t|\mathcal{F}_n\right] & \text{if } \phi(X_n) < \mathbb{E}\left[V_{n+1}^t|\mathcal{F}_n\right], \\ \zeta(X_n) & \text{otherwise .} \end{cases}$$

It turns out that the value processes possess the monotone property (Lemma 1).

Lemma 1 *For every $n, t \in [T]$ such that $n \leq t$, one has that the equilibrium value processes defined as in (24) satisfy*

$$V_n^n \leq V_n^t \leq V_n^{t+1}.$$

One can show Lemma 1 with an induction argument. Let the event $E_n := \{\omega : \phi(X_n) < \mathbb{E}[\zeta(X_{n+1})|\mathcal{F}_n]\}$ be when the next round expected confrontational payoff is higher than the current early termination payoff. Consider the base case; it follows that at any stage $t \in [T-1]$, since the next round both parties need to terminate, it is reasonable for the defender to choose to terminate if the early termination payoff is higher than the expected confrontational payoff. Thus,

$$\begin{aligned} V_t^{t+1} &= \begin{cases} \mathbb{E}[\zeta(X_{t+1})|\mathcal{F}_t] & \text{on } E_t, \\ \zeta(X_t) & \text{on } E_t^c, \end{cases} \\ &\geq \begin{cases} \phi(X_t) & \text{on } E_t, \\ \zeta(X_t) & \text{on } E_t^c, \end{cases} \\ &\geq \zeta(X_t) = V_t^t. \end{aligned}$$

Now we assume that $V_j^{j+k-1} \leq V_j^{j+k}$ for some arbitrary stage $1 \leq k \leq T-1$ and for all $j \in [T-k]$, then, for $t \in [T-k-1]$,

$$\begin{aligned} V_t^{t+k} &= \text{val}\begin{bmatrix} \zeta(X_t) & \phi(X_t) \\ \psi(X_t) & \mathbb{E}\left[V_{t+1}^{t+k}|\mathcal{F}_t\right] \end{bmatrix} \\ &\leq \text{val}\begin{bmatrix} \zeta(X_t) & \phi(X_t) \\ \psi(X_t) & \mathbb{E}\left[V_{t+1}^{t+k+1}|\mathcal{F}_t\right] \end{bmatrix} \\ &= V_t^{t+k+1}. \end{aligned}$$

Hence, the monotonicity follows by the induction argument.

That V_k^t being increasing in t gives off two signals; the first is that due to the Monotone Convergence theorem for $\mathbb{E}[\cdot|\mathcal{F}_t]$, there exists a limit for V_k^t if we consider the infinite-stage problem ($t \to \infty$); the second is that the dominating strategy can be obtained when the stopping stage is not constrained, up to time T.

Now we define two stopping times, for $t \in [T]$,

$$\begin{aligned} \bar{\tau}_t &= \inf\{t \leq k \leq T | V_k^T = \zeta(X_k)\}, \\ \bar{\sigma}_t &= \inf\{t \leq k \leq T | V_k^T = \zeta(X_k)\}. \end{aligned}$$

The significance of $(\bar{\tau}_t, \bar{\sigma}_t)$ is given in Theorem 1.

Theorem 1 *Under ADC, the following statements hold for arbitrary initial state $x \in \mathcal{X}$:*

(i) For every $t \in [T]$, and all $\tau \in \mathcal{T}_t, \sigma \in \mathcal{T}_t$,

$$\mathbb{E}[H(\tau, \bar{\sigma}_t)|\mathcal{F}_t] \leq V_t^T = \mathbb{E}[V_{\bar{\tau}_t \wedge \bar{\sigma}_t}^T|\mathcal{F}_t] = \mathbb{E}[H(\bar{\tau}_t, \bar{\sigma}_t)|\mathcal{F}_t] \leq \mathbb{E}[H(\bar{\tau}_t, \sigma)|\mathcal{F}_t].$$

(ii) At every time $t \in [T]$, a pair $(\bar{\tau}_t, \bar{\sigma}_t)$ is an equilibrium point for that time step t, and a DDE value corresponding to $(\bar{\tau}_0, \bar{\sigma}_0)$ is given as

$$\mathbb{E}[V_0^T] = \mathbb{E}[V_{\bar{\tau}_0 \wedge \bar{\sigma}_0}^T] = \mathbb{E}[H(\bar{\tau}_0, \bar{\sigma}_0)].$$

Proof Fix a $t \in [T]$ arbitrarily. We have that, if $k \in \{t, \ldots, \bar{\sigma}_t\}$, by definition of $\bar{\sigma}_t$, we have

$$V_k^T = \mathbb{E}[V_{k+1}^T | \mathcal{F}_k].$$

Thus, the sequence $\{V_{k \wedge \bar{\sigma}_t}^T, k \geq t\}$ is a regular Martingale, so that $V_t^T = \mathbb{E}[V_{\tau \wedge \bar{\sigma}_t}^T | \mathcal{F}_t]$ for any $\tau \in \mathcal{T}_t$, by Doob's optional sampling theorem. Since $V_{\bar{\sigma}_t}^T = \zeta(X_{\bar{\sigma}_t}) \geq \psi(X_{\bar{\sigma}_t})$, if $\bar{\sigma}_t \leq \infty$ and $V_k^T \geq \phi(X_k)$ if $\bar{\sigma}_t > k$, it follows that:

$$\begin{aligned} V_t^T &= \mathbb{E}[V_{\tau \wedge \bar{\sigma}_t}^T | \mathcal{F}_t] \\ &= \mathbb{E}[V_\tau^T \mathbb{1}_{\{\tau < \bar{\sigma}_t\}} + V_{\bar{\sigma}_t}^T \mathbb{1}_{\{\bar{\sigma}_t \leq \tau\}} | \mathcal{F}_t] \\ &\geq \mathbb{E}[\phi(X_\tau) \mathbb{1}_{\{\tau < \bar{\sigma}_t\}} + \psi(X_{\bar{\sigma}_t}) \mathbb{1}_{\{\bar{\sigma}_t < \tau\}} + \zeta(X_{\bar{\sigma}_t}) \mathbb{1}_{\{\bar{\sigma}_t = \tau\}} | \mathcal{F}_t] \\ &= \mathbb{E}[H(\tau, \bar{\sigma}_t) | \mathcal{F}_t]. \end{aligned}$$

Applying the same argument we are able to prove the $\leq$ side for all $\sigma \in \mathcal{T}_t$. By letting $t = 0$ we arrive at the conclusion. □

Theorem 1 (i) implies that for every subgame starting from time t, the equilibrium strategy is always a threshold strategy for both parties, where the threshold needed to be computed is $\mathbb{E}[V_{t+1}^T | \mathcal{F}_t]$. Both parties would only have incentives to stop when $\phi(X_t)$ is hitting the threshold. ii) states that in the adversarial dominance environment, $(\bar{\tau}_0, \bar{\sigma}_0)$ are the equilibrium strategies. However, the determination of the equilibrium value sequence V_t^T is computationally intractable, as one would have to construct the random variables backwardly according to (24), enumerating over the filtration sets.

Therefore, it is crucial to generalize the above arguments to the space of $\mathcal{E}(\mathcal{X})$. As we may assume that the players have access to the payoff functions, constructing a map between X_t and the equilibrium value process can be relatively easier. Indeed, due to the Markovian property of X_t, it turns out we only need a sequence of $\mathcal{B}(\mathbb{R})/\mathcal{G}$-measurable value functions $\{v_t(\cdot)\}_{t \in [T]}$ that satisfies the following conditions:

$$\begin{aligned} &v_T(x) = \zeta(x), \quad \text{for all } x \in \mathcal{X}, \\ &v_t(x) \in \mathrm{SE} \begin{bmatrix} \zeta(x) & \phi(x) \\ \psi(x) & \mathcal{T} v_{t+1}(x)] \end{bmatrix}, \quad \text{for all } x \in \mathcal{X}, t \in [T-1], \end{aligned} \tag{25}$$

where SE stands for the set of Nash (saddle-point) equilibrium values of the matrix game with two pure strategies. Then, the last iterate value function is $\zeta(\cdot)$ by construction. The rest of the business is to figure out the backward induction equation that involves the val$(\cdot)$ operator, which still relies on the calculation of $\mathscr{T}$ leveraging Monte-Carlo sampling type of methods. Following Lemma 1 the monotonicity still holds, $\{v_t(\cdot)\}_{t\in[T]}$ is decreasing, which can be interpreted as that the decision made at the outset is most valuable, as time passes, the opportunity fades. For any $t \in [T]$, we define the two stopping times,

$$\tau_t^* = \inf\{t \le k \le T | \{v_k(X_k) = \zeta(X_k)\} \bigcup \{v_k(X_k) = \phi(X_k)\}\},$$
$$\sigma_t^* = \inf\{t \le k \le T | \{v_k(X_k) = \zeta(X_k)\} \bigcup \{v_k(X_k) = \psi(X_k)\}\}.$$

By Theorem 2, (τ_0^*, τ_0^*) is the equilibrium strategy pair, the definition of which reflects the consistency of value function computation, that is, the players' current value estimates either reach the early termination threshold or confrontational threshold.

Theorem 2 *Under ADC, the following statements hold for arbitrary initial state* $x \in \mathcal{X}$*:*

- *for every* $t \in [T]$*, and all* $\tau \in \mathcal{T}_t, \sigma \in \mathcal{T}_t$*,*

$$\mathbb{E}[H(\tau, \sigma_t^*)|\mathcal{F}_t] \le \mathbb{E}[H(\tau_t^*, \sigma_t^*)|\mathcal{F}_t] \le \mathbb{E}[H(\tau_t^*, \sigma)|\mathcal{F}_t].$$

- *the game admits a DDE strategy* (τ_0^*, σ_0^*)*, at which the value function satisfies*

$$\begin{aligned}\mathcal{V}^{\tau_0^*,\sigma_0^*}(x) &= \operatorname{ess\,sup}_{\tau\in\mathcal{T}} \operatorname{ess\,inf}_{\sigma\in\mathcal{T}} \mathcal{V}^{\tau,\sigma}(x) \\ &= \operatorname{ess\,inf}_{\sigma\in\mathcal{T}} \operatorname{ess\,sup}_{\tau\in\mathcal{T}} \mathcal{V}^{\tau,\sigma}(x).\end{aligned}$$

We omit the proof here as Theorem 2 can be seen as an extension of Theorem 1, to which the reasoning is similar. One can simply construct the sequence of value functions with a constant translation, and the results still hold.

6.2.2 Case II: Defensive Dominance

Under the Defensive Dominance Condition (DDC), the payoff functions satisfy the ordered condition $\psi \le \zeta \le \phi$ for all system states $x \in \mathcal{X}$. In this case, at any state $x \in \mathcal{X}$, the defender can bide his time during the interactions of cyber kill/defense chain, as the systematic loss after the execution of Command & Control is mitigable. Such a condition happens when the defender possesses a superior and robust position. This is also called *second-mover advantage*, that is, the defender has the incentive to wait for the opponent to end the game.

DDC corresponds to the ordered payoff condition for standard Dynkin's game, where the existence and uniqueness of a saddle point value process have been proved. The constructive sequence of (locally integrable) random variables $\{V_t\}_{t=0}^{T}$, in this case, is now more straightforward (as discussed in [78]), defined by

$$\begin{aligned} &V_T = \zeta(X_T), \\ &V_t = \min\{\psi(X_t), \max\{\phi(X_t), \mathbb{E}[V_{t+1}|\mathcal{F}_t]\}\}, \qquad \text{for } t = 0, \ldots, T-1, \end{aligned} \tag{26}$$

with the stopping time strategies defined as

$$\begin{aligned} \bar{\tau}_t &= \inf\{t \le k \le T | V_k = \phi(X_k)\}, \\ \bar{\sigma}_t &= \inf\{t \le k \le T | V_k = \psi(X_k)\}. \end{aligned}$$

Theorem 3 *Under DDC, the following statements hold:*

(i) for each $t \in [T]$, and for all $\tau \in \mathcal{T}_t$, $\sigma \in \mathcal{T}_t$,

$$\begin{aligned} &V_t = \mathbb{E}[V_{\tau_t^* \wedge \sigma_t^*}|\mathcal{F}_t] = \mathbb{E}[H(\tau_t^*, \sigma_t^*)|\mathcal{F}_t], \qquad \textit{and,} \\ &\mathbb{E}[H(\tau, \sigma_t^*)|\mathcal{F}_t] \le V_t \le \mathbb{E}[H(\tau_t^*, \sigma)|\mathcal{F}_t]. \end{aligned}$$

(ii) at every time $t \in [T]$, a pair (τ_t^*, σ_t^*) is an equilibrium point for the subgame starting at time t, and the DDE value corresponding to (τ_0^*, σ_0^*) is

$$\mathbb{E}[V_0] = \mathbb{E}[V_{\tau_0^* \wedge \sigma_0^*}] = \mathbb{E}[H(\tau_0^*, \sigma_0^*)].$$

Proof Similar to previous results, we shall give the proof for the "$\ge$" side. First, we examine the trivial case where $t = T$. Obviously, there's no option but stop for both parties, so $G_t = \zeta(X_t) = \mathbb{E}[G_{\tau \wedge \sigma_t^*}|\mathcal{F}_t] = \mathbb{E}[H(\tau, \sigma_t^*)|\mathcal{F}_t]$ for all $\tau \in \mathcal{T}_T = \{T\}$.

Fix a $t < T$. Choose some k such that $t \le k \le \tau_t^* \wedge \sigma_t^*$, we have $V_k = \mathbb{E}[V_{k+1}|\mathcal{F}_k]$ by definition. Thus, $\{V_{k \wedge \tau_t^* \wedge \sigma_t^*}\}_{k=t}^{T}$ is a Martingale. Applying Doob's optional sampling theorem, one has

$$V_t = \mathbb{E}[V_{\tau \wedge \tau_t^* \wedge \sigma_t^*}|\mathcal{F}_t], \qquad \text{for all } \tau \in \mathcal{T}_t.$$

Let $\tau = \tau_t^*$, we arrive at

$$\begin{aligned} V_t &= \mathbb{E}[V_{\tau_t^* \wedge \sigma_t^*}|\mathcal{F}_t] \\ &= \mathbb{E}[V_{\tau_t^*} \mathbb{1}_{\{\tau_t^* \le \sigma_t^*\}} + V_{\sigma_t^*} \mathbb{1}_{\{\tau_t^* > \sigma_t^*\}}|\mathcal{F}_t] \\ &= \mathbb{E}[\phi(X_{\tau_t^*}) \mathbb{1}_{\{\tau_t^* < \sigma_t^*\}} + \psi(X_{\sigma_t^*}) \mathbb{1}_{\{\tau_t^* > \sigma_t^*\}} + \zeta(X_{\tau_t^*}) \mathbb{1}_{\{\tau_t^* = \sigma_t^*\}}|\mathcal{F}_t] \\ &= \mathbb{E}[H(\tau_t^*, \sigma_t^*)|\mathcal{F}_t]. \end{aligned}$$

It is also obvious that when $t \leq k < \sigma_t^*$, then $V_k < \psi(X_k)$, therefore $V_k = \mathbb{E}[V_{k+1}|\mathcal{F}_k]$. This implies that $\{V_{k\wedge\sigma_t^*}\}_{k=t}^T$ is a supermartingale. Hence, $V_t \geq \mathbb{E}[V_{\tau\wedge\sigma_t^*}|\mathcal{F}_t]$ for all $\tau \in \mathcal{T}_t$,

$$\begin{aligned}\mathbb{E}[V_{\tau\wedge\sigma_t^*}|\mathcal{F}_t] &\geq \mathbb{E}[\phi(X_\tau)\mathbb{1}_{\{\tau<\sigma_t^*\}} + \psi(X_{\sigma_t^*})\mathbb{1}_{\{\tau_t^*>\sigma_t^*\}} + \zeta(X_\tau)\mathbb{1}_{\{\tau=\sigma_t^*\}}|\mathcal{F}_t] \\ &= \mathbb{E}[H(\tau, \sigma_t^*)|\mathcal{F}_t],\end{aligned}$$

since $V_k > \phi(X_k)$ and $\zeta(X_k) \leq \psi(X_k)$ for all $0 \leq k \leq T$. Claim (ii) follows immediately.

□

Again we generalize the result to $\mathcal{E}(\mathcal{X})$, we wish to find a sequence of $\mathcal{B}(\mathbb{R})/\mathcal{G}$-measurable functions $\{v_t(\cdot)\}_{t\in[T]}$ that satisfies the following conditions (or being shifted by a constant):

$$\begin{aligned}&v_T(x) = \zeta(x), \qquad \text{for all } x \in \mathcal{X}, \\ &v_t(x) = \min\{\psi(x), \max\{\phi(x), \mathcal{T}v_{t+1}(x)\}\}, \qquad \text{for all } x \in \mathcal{X}, t = [T-1],\end{aligned}$$

and the DDE pair (τ^*, σ^*) can be defined as:

$$\begin{aligned}\tau^* &= \inf\{k \in [T]|v_k(X_k) = \phi(X_k)\}, \\ \sigma^* &= \inf\{k \in [T]|v_k(X_k) = \psi(X_k)\}.\end{aligned}$$

Theorem 4 *Under ADC, the game admits a DDE strategy pair* (τ^*, σ^*)*, such that*

$$\begin{aligned}\mathcal{V}^{\tau^*,\sigma^*}(x) &= \operatorname{ess\,sup}_{\tau\in\mathcal{T}}\operatorname{ess\,inf}_{\sigma\in\mathcal{T}}\mathcal{V}^{\tau,\sigma}(x) \\ &= \operatorname{ess\,inf}_{\sigma\in\mathcal{T}}\operatorname{ess\,sup}_{\tau\in\mathcal{T}}\mathcal{V}^{\tau,\sigma}(x),\end{aligned}$$

for all $x \in \mathcal{X}$.

Under DDC, the optimal strategies for the players are waiting for the equilibrium process to hit the lower/upper bound of the payoff values.

6.2.3 Decision Dominance with Information Asymmetry

In the MDW scenarios, it is crucial to recognize that both defenders and attackers operate within an environment of information asymmetry [11, 38]. This is particularly evident when considering STIX logs, as the information accessible to attackers differs from what defenders can observe. While defenders have the advantage of comprehensive logs that capture security events and indicators of compromise, attackers possess their own set of advantages stemming from their ability to exploit the gaps in the defender's knowledge. Attackers can leverage their insider information, external reconnaissance, and targeted intelligence gathering to

gain insights into the defender's security measures, potential vulnerabilities, and defensive capabilities. In the meantime, the defender may have deceptive defense mechanisms that hide their tactics, techniques, and procedures (TTPs), to counteract the malicious exploitation.

To formalize the notion, we redefine $(X_t)_{t=0}^T$ as the true system state (which cannot be completely captured by the STIX logs), and let $(O_t^i)_{0\le t\le T}(i = 1, 2)$ be the observation process for the defender ($i = 1$) and the attacker ($i = 2$), which jointly live in the space $(\mathcal{O}^1 \times \mathcal{O}^2, \mathcal{H}^1 \otimes \mathcal{H}^2)$, adapted to the filtrations $\mathbb{H}^1 = (\mathcal{H}_t^1)_{0\le t\le T}$ and $\mathbb{H}^2 = (\mathcal{H}_t^2)_{0\le t\le T}$. This information asymmetry enables the players to make informed decisions regarding their strategies, tactics, and the selection of attack vectors/defensive mechanisms. Therefore, defenders must not only rely on STIX logs and robust defense mechanisms but also proactively bridge the information gap by enhancing their threat intelligence capabilities, anticipating adversary behaviors, and continuously evolving their defense strategies to counter the advantages of information asymmetry in the cyber landscape.

To formally define the DDG under asymmetric information structure, we denote by $\mathcal{T}(\mathbb{H}^i)$ the set of $\mathbb{H}^i$-stopping times, $\mathcal{T}(\mathbb{H}^i) = \{0 \le \tau \le T : \{\tau(\omega) \le k\} \in \mathcal{H}_k^i \ \forall k \in [T], \forall \omega \in \Omega\}$. The decision payoffs at each stage $t \in [T]$, in this case, may depend on both O_t^i and X_t. Following the standard formalism of the Partially Observable Markov Decision Process (POMDP), we assume that the payoff functions still only depend on the true system state, which is a hidden latent variable for both players. Instead, there exists an emission kernel $\mathbb{O} : \mathcal{X} \to \Delta(\mathcal{O}^1 \times \mathcal{O}^2)$ that measures the joint probability of observations made by the defender and the attacker. An illustration is shown in Fig. 7

Factorization Lemma says in order to infer the true states from the partial observations, say, if O_t^1 is $\mathcal{F}_t/\mathcal{H}_t^1$-measurable, there needs to be a deterministic $\mathcal{F}_t/\mathcal{H}_t^1$-measurable map $f : \mathcal{X} \to \mathcal{O}^1$ such that $O_t^1 = f(X_t)$, whose existence and accessibility are not always guaranteed in the cyber domain. Therefore, it is reasonable to assume that the players have their stopping time strategies restricted to $\mathcal{T}(\mathbb{H}^i)$. Definition 11 summarizes the game under asymmetrical information structure.

Definition 11 (Decision Dominance Game with Information Asymmetry) A tuple $(\mathcal{X}, \mathcal{O}^1 \times \mathcal{O}^2, \mathcal{P}, \mathbb{O}, \phi, \zeta, \psi, \mathcal{T}(\mathbb{H}^1), \mathcal{T}(\mathbb{H}^2))$ encapsulates a Decision Dominance Game with Information Asymmetry (DDGIA) if it satisfies that

- There exists a hidden Markov process $(X_t)_{0\le t\le T}$ that lives in $(\mathcal{X}, \mathcal{G})$ with transition kernel $\mathcal{P}$, which yields observations (O_t^1, O_t^2) through emission kernel $\mathbb{O}$;
- ϕ, ζ, and ψ are payoff functions mapping from X_t to $\mathbb{R}$, $\phi, \zeta, \psi \in \mathcal{E}(\mathcal{X})$, which is the set of all bounded $\mathcal{B}(\mathbb{R})/\mathcal{G}$-measurable functions on $(\mathcal{X}, \mathcal{G})$. Further, $\min(\phi, \psi) \le \zeta \le \max(\phi, \psi)$ on $\mathcal{X}$;
- At each stage t, player i $(i = 1, 2)$ picks a stopping strategy from space $\mathcal{T}_t(\mathbb{H}^i) := \{t \le \tau \le T : \{\tau(\omega) \le k\} \in \mathcal{H}_k^i \ \forall k \in [T], \forall \omega \in \Omega\}$ to decide whether to stop or continue the kill/defense chain.

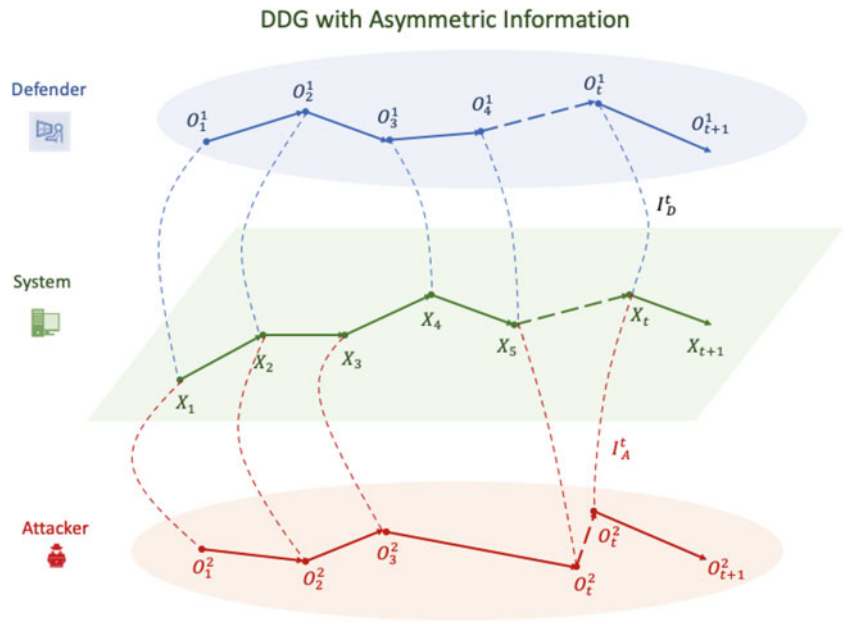


Fig. 7 An illustration of asymmetric information dynamic games defined in Definition 11. The two players have distinct partial observations for the system state X_t, denoted by (O_t^1, O_t^2). In DDG, the defender has to infer the true state to determine the stopping time strategy based on the payoff structure, which relies on credible modeling, requiring expertise in the fundamental understanding of the cyber threats

- At each stage the utility function of the defender is

$$H(\tau_t, \sigma_t) = \phi(X_{\tau_t})\mathbb{1}_{\{\tau_t<\sigma_t\}} + \zeta(X_{\tau_t})\mathbb{1}_{\{\tau_t=\sigma_t\}} + \psi(X_{\sigma_t})\mathbb{1}_{\{\tau_t>\sigma_t\}},$$

  while the attacker attains $-H(\tau_t, \sigma_t)$.

The goal of the defender is to choose τ to maximize her utility under all possible choices of the attacker, which leads to the lower value function of DDGIA,

$$\underline{V}(x) = \operatorname{ess\,sup}_{\tau\in\mathcal{T}(\mathbb{H}^1)} \operatorname{ess\,inf}_{\sigma\in\mathcal{T}(\mathbb{H}^2)} V^{\tau,\sigma}(x). \tag{27}$$

Similarly, the goal of the attacker is to choose σ to minimize the defender's utility under all possible choices of the defender, which leads to the upper-value function,

$$\overline{V}(x) = \operatorname{ess\,inf}_{\sigma\in\mathcal{T}(\mathbb{H}^2)} \operatorname{ess\,sup}_{\tau\in\mathcal{T}(\mathbb{H}^1)} V^{\tau,\sigma}(x). \tag{28}$$

Definition 12 (DDE with Information Asymmetry) A pair of stopping time strategies $(\tau^*, \sigma^*) \in \mathcal{T}(\mathbb{H}^1) \times \mathcal{T}(\mathbb{H}^2)$ is a Decision-Dominance Equilibrium (DDE) if for all initial state $x \in \mathcal{X}$, it satisfies the minimax condition:

$$\begin{aligned} V^{\tau^*,\sigma^*}(x) &= \operatorname{ess\,sup}_{\sigma\in\mathcal{T}(\mathbb{H}^2)} \operatorname{ess\,inf}_{\tau\in\mathcal{T}(\mathbb{H}^1)} V^{\tau,\sigma}(x) \\ &= \operatorname{ess\,inf}_{\tau\in\mathcal{T}(\mathbb{H}^2)} \operatorname{ess\,sup}_{\sigma\in\mathcal{T}(\mathbb{H}^2)} V^{\tau,\sigma}(x). \end{aligned} \tag{29}$$

We say that a DDGIA has a value if $\underline{V}(x) = \overline{V}(x)$. Note that the existence and uniqueness of the value is a non-trivial question in general, as we shall find the reasoning presented in the previous section not applicable due to the introduction of two private filtrations for both parties. In principle, the value exists if $\mathbb{H}^i$ reveal the same information from $\mathbb{F}$, in which case the conditional expectation $\mathbb{E}(\cdot|\mathbb{H}_t^i)$ can be seen equivalent with $\mathbb{E}(\cdot|\mathcal{F}_t)$, thus the players will make their decisions using the same threshold policies. This property, however, requires some special structures of the observation kernel $\mathbb{O}$, which might not hold in realistic scenarios.

6.3 Decision Dominance Zero-Trust Defense (DD-ZTD): A Case Study

In this case study, we consider an T-episodic DDG with symmetric information over the same 5G network $G = \langle V, E\rangle$ as discussed in Sect. 5.2, where each episode t contains H ZTD steps against lateral movement. The ZTD state action variables within one episode t is $\mathbf{sa}_t = (s_t^1, a_t^1, s_t^2, a_t^2 \ldots, a_t^{H-1}, s_t^H)$, where $s_t^h = (G_t^h, L_t^h), h = 1, \ldots, H$ are the authentication graphs and the visiting indicator functions at episode t, and the joint actions a_t^h are automated by the threshold-policy trust engine, which is either the Bayesian type or the Machine Learning type. Denote the STIX logs within t as $x_t \in \mathcal{X}$, which includes but is not limited to the events of 5G network exposure, slicing control, session management; the threat actor characterizations such as suspected user intentions and handling guidance. The Markovian state at episode t is a composition of both historical ZTD state action variables and the STIX logs gathered before episode t, i.e., $X_t := (\mathbf{sa}_{1:t-1}, x_{t-1})$.

During the cyber kill/defense chain interaction, at the beginning of each episode, the defender can choose to completely cut off the chain before episode t starts by isolating the networks, restarting the services, resetting all the credentials, patching and hardening the security configurations, and then restoring and resuming the operations. The cost of the defender's cutting-off strategy is $C(\cdot) : \mathcal{X} \to \mathbb{R}$, which only depends on the cyber threat information. Similarly, the attacker can choose to take action early by exploiting Zero-Day vulnerabilities, evading intrusion detection systems, and implementing stealthy command and control at an early stage of the cyber kill chain. Again we let the exploitation loss be $\ell(\cdot) : \mathcal{X} \to \mathbb{R}$, which completely depends on the cyber threat characterization of episode t. Now we are ready to define the three payoff functions in our DDG framework.

The early termination payoff, confrontation payoff, and late termination payoff functions can be defined as

$$\begin{aligned}
\phi(X_t) &= -\mathbb{E}[\sum_{k=1}^{t-1}\sum_{h=1}^{H} u_D(s_k^h, a_k^h)] - C(x_{t-1}),\\
\zeta(X_t) &= -\mathbb{E}[\sum_{k=1}^{t-1}\sum_{h=1}^{H} u_D(s_k^h, a_k^h)] - C(x_{t-1}) - \ell(x_{t-1}),\\
\psi(X_t) &= -\mathbb{E}[\sum_{k=1}^{t}\sum_{h=1}^{H} u_D(s_k^h, a_k^h)] - \ell(x_{t-1}),
\end{aligned} \tag{30}$$

where the expectation $\mathbb{E}[\sum_h^H u_D(s_t^h, a_t^h)]$ is taken conditioned on $\mathbf{sa}_{1:t-1}$. The interpretation is that when the defender chooses to shut down and restore the services, the ZTD stops for that episode, while if the attacker chooses to exploit early, the ZTD mechanism is still active.

One can easily verify that when both ℓ and C are positive and the expected ZTD cost within every episode t satisfies $\mathbb{E}[\sum_h^H u_D(s_t^h, a_t^h)] > C(x_{t-1})$ the DDG satisfies DDC.

7 Conclusion

This chapter develops a game-theoretic framework for the decision-dominant zero-trust defense of 5G networks in the face of advanced persistent threats that utilize a cyber kill chain to disrupt the network operation. The advanced features of 5G networks, despite their contributions to multi-domain integration, bring a larger attack surface and render the network system vulnerable in the presence of advanced persistent threats (APT) and other malicious attacks. The combination of system vulnerabilities, supply chains of 5G equipment, and network slicing, along with others, can be exploited by an APT attacker to create a cyber kill chain consisting of reconnaissance, planning, execution, and exploration.

To outmaneuver the malicious attacker and thwart the kill chain, this chapter proposes a decision-dominant zero-trust defense (DD-ZTD) framework, a proactive defense mechanism enabling the defender to make timely and effective decisions with incomplete information regarding the situation and disrupt the kill chain before its completion. Two pillars of DD-ZTD are game-theoretic zero-trust defense built upon asymmetric information Markov games (AIMG) and decision-dominance defense characterized by Dykin's stopping-time games. With the AIMG's expressive power on information structures in cyber defense, ZTD employs a variety of trust engines to evaluate entities' trustworthiness with limited partial observations, which is then fed into the access policy powered by equilibrium thinking that anticipates the attacker's response. We further present an end-to-end ZTD facilitated by recent machine learning advancements with data-driven trust evaluation and explainable and generalizable policy learning.

While the proposed ZTD offers a set of fruitful tools to quantitatively analyze trustworthiness under information asymmetry, the networked entities still face multi-stage persistent cyber threats that call for rapid response from the defender. To outpace the attacker's kill chain, decision-dominance defense (D^3), mathematically treating interactions of cyber defense/kill chain as a stopping-time game, aims to take the decisive move to cut off the kill chain before the attack materializes. The essence of D^3 is the timing of the cutting-off, which is determined by the equilibrium of the game with anticipation of the attacker's strategic move. The resulting DD-ZTD, as an organic integration of the two game-theoretic defense mechanisms, displays great potential in combating sophisticated adversaries, which we articulate using a case study in 5G network defense.

References

1. Headquarters, Department of the Army (2022) FM 3-0, Operations. https://usacac.army.mil/node/3048. Accessed 05 Jul 2023
2. Department of Defense (2018) Summary of the 2018 national defense strategy. https://www.spoc.spaceforce.mil/About-Us/Fact-Sheets/Display/Article/2381348/advanced-extremely-high-frequency-system-aehf. Accessed 05 Jul 2023
3. Space Operations Command (SPOC) (2021) Advanced extremely high frequency system (aehf). https://www.spoc.spaceforce.mil/About-Us/Fact-Sheets/Display/Article/2381348/advanced-extremely-high-frequency-system-aehf. Accessed 05 Jul 2023
4. Lockheed Martin (2023) Indago UAV. https://www.lockheedmartin.com/en-us/products/indago-vtol-uav.html. Accessed 05 Jul 2023
5. Huang L, Zhu Q (2022) Radams: resilient and adaptive alert and attention management strategy against informational denial-of-service (IDoS) attacks. Comput Secur 121:102844
6. Wijethilaka S, Liyanage M (2021) Survey on network slicing for internet of things realization in 5g networks. IEEE Commun Surv Tutor 23(2):957–994. https://doi.org/10.1109/COMST.2021.3067807
7. Xiao Y, Jia Y, Liu C, Cheng X, Yu J, Lv W (2019) Edge computing security: state of the art and challenges. Proc IEEE 107(8):1608–1631. https://doi.org/10.1109/JPROC.2019.2918437
8. Zhu Q, Rass S (2018) On multi-phase and multi-stage game-theoretic modeling of advanced persistent threats. IEEE Access 6:13958–13971
9. Huang L, Zhu Q (2020) A dynamic games approach to proactive defense strategies against advanced persistent threats in cyber-physical systems. Comput Secur 89:101660
10. Rass S, Zhu Q (2016) Gadapt: a sequential game-theoretic framework for designing defense-in-depth strategies against advanced persistent threats. In: International conference on decision and game theory for security. Springer, Berlin, pp 314–326
11. Huang L, Zhu Q (2019) Dynamic Bayesian games for adversarial and defensive cyber deception. In: Autonomous cyber deception: reasoning, adaptive planning, and evaluation of honeyThings. Springer, Berlin, pp 75–97
12. Rose S, Borchert O, Mitchell S, Connelly S (2020) Zero trust architecture. Technical report, National Institute of Standards and Technology
13. Osborn K (2018) "first look, first shot, first kill": How the f-22 raptor could fly until 2060. https://nationalinterest.org/blog/buzz/first-look-first-shot-first-kill-how-f-22-raptor-could-fly-until-2060-35937
14. Gady FS, Stronell A (2020) Cyber capabilities and multi-domain operations in future high-intensity warfare in 2030. In: Cyber threats and NATO 2030: horizon scanning and analysis, pp 151–176

15. Mallik RK, Scholtz RA, Papavassilopoulos GP (2000) Analysis of an on-off jamming situation as a dynamic game. IEEE Trans Commun 48(8):1360–1373
16. Mukherjee A, Swindlehurst AL (2012) Jamming games in the MIMO wiretap channel with an active eavesdropper. IEEE Trans Signal Process 61(1):82–91
17. Sayin MO, Hosseini H, Poovendran R, Başar T (2018) A game theoretical framework for inter-process adversarial intervention detection. In: International conference on decision and game theory for security. Springer, Berlin, pp 486–507
18. Chen J, Touati C, Zhu Q (2019) Optimal secure two-layer IoT network design. IEEE Trans Control Netw Syst 1–1. https://doi.org/10.1109/TCNS.2019.2906893
19. Pawlick J, Farhang S, Zhu Q (2015) Flip the cloud: cyber-physical signaling games in the presence of advanced persistent threats. In: International conference on decision and game theory for security. Springer, Berlin, pp 289–308
20. Pawlick J, Zhu Q (2017) Strategic trust in cloud-enabled cyber-physical systems with an application to glucose control. IEEE Trans Inf Forensics Secur 12(12):2906–2919
21. Huang L, Chen J, Zhu Q (2017) A large-scale Markov game approach to dynamic protection of interdependent infrastructure networks. In: International conference on decision and game theory for security. Springer, Berlin, pp 357–376
22. Chen J, Zhu Q (2022) A cross-layer design approach to strategic cyber defense and robust switching control of cyber-physical wind energy systems. IEEE Trans Autom Sci Eng 20(1):624–635
23. Chen J, Zhu Q (2019) A game-and decision-theoretic approach to resilient interdependent network analysis and design. Springer, Berlin
24. Chen J, Zhu Q (2016) A game-theoretic framework for resilient and distributed generation control of renewable energies in microgrids. IEEE Trans Smart Grid 8(1):285–295
25. Chen J, Zhu Q (2019) A games-in-games approach to mosaic command and control design of dynamic network-of-networks for secure and resilient multi-domain operations. In: Chen G, Pham KD (eds) Sensors and systems for space applications XII. International Society for Optics and Photonics, SPIE, vol 11017, pp 189–195. https://doi.org/10.1117/12.2526677
26. Chen J, Zhu Q (2020) Control of multilayer mobile autonomous systems in adversarial environments: a games-in-games approach. IEEE Trans Control Netw Syst 7(3):1056–1068. https://doi.org/10.1109/TCNS.2019.2962316
27. Chen J, Zhu Q (2016) Resilient and decentralized control of multi-level cooperative mobile networks to maintain connectivity under adversarial environment. In: IEEE conference on decision and control (CDC), pp 5183–5188
28. Zhu Q, Rass S, Dieber B, Vilches VM, et al (2021) Cybersecurity in robotics: challenges, quantitative modeling, and practice. Found Trends Robot 9(1):1–129
29. Kieras T, Farooq MJ, Zhu Q (2020) Riots: risk analysis of IoT supply chain threats. In: 2020 IEEE 6th World forum on Internet of Things (WF-IoT). IEEE, pp 1–6
30. Ge Y, Zhu Q (2022) Accountability and insurance in IoT supply chain. arXiv preprint arXiv:220111855. https://doi.org/10.48550/arXiv.2201.11855
31. Kieras T, Farooq J, Zhu Q (2022) IoT supply chain security risk analysis and mitigation: modeling, computations, and software tools. Springer, Berlin
32. Pan Y, Zhu Q (2022) On poisoned wardrop equilibrium in congestion games. In: International conference on decision and game theory for security. Springer, pp 191–211
33. Pan Y, Li T, Zhu Q (2023) On the resilience of traffic networks under non-equilibrium learning. In: 2023 American control conference (ACC). IEEE, pp 3484–3489
34. Pan Y, Li T, Zhu Q (2023) Is stochastic mirror descent vulnerable to adversarial delay attacks? A traffic assignment resilience study. arXiv preprint arXiv:230401161. https://doi.org/10.48550/arXiv.2304.01161
35. Zheng J, Castañón DA (2012) Dynamic network interdiction games with imperfect information and deception. In: 2012 IEEE 51st IEEE conference on decision and control (CDC). IEEE, pp 7758–7763
36. Zhu Q, Clark A, Poovendran R, Başar T (2012) Deceptive routing games. In: 2012 IEEE 51st IEEE conference on decision and control (CDC). IEEE, pp 2704–2711

37. Zhuang J, Bier VM, Alagoz O (2010) Modeling secrecy and deception in a multiple-period attacker–defender signaling game. Eur J Oper Res 203(2):409–418
38. Pawlick J, Colbert E, Zhu Q (2019) A game-theoretic taxonomy and survey of defensive deception for cybersecurity and privacy. ACM Comput Surv 52(4):82
39. Zhu Q, Başar T (2013) Game-theoretic approach to feedback-driven multi-stage moving target defense. In: Decision and game theory for security. Springer, Berlin, pp 246–263
40. Jajodia S, Ghosh AK, Swarup V, Wang C, Wang XS (2011) Moving target defense: creating asymmetric uncertainty for cyber threats, vol 54. Springer Science & Business Media, Berlin
41. Huang L, Zhu Q (2021) Combating informational denial-of-service (idos) attacks: modeling and mitigation of attentional human vulnerability. In: Decision and game theory for security: 12th international conference, GameSec 2021, Virtual Event, October 25–27, 2021, Proceedings. Springer, Berlin, pp 314–333
42. Huang L, Zhu Q (2023) Cognitive security: a system-scientific approach. Springer Nature, Berlin
43. Liao HJ, Richard Lin CH, Lin YC, Tung KY (2013) Intrusion detection system: a comprehensive review. J Netw Comput Appl 36(1):16–24. https://doi.org/10.1016/j.jnca.2012.09.004. https://www.sciencedirect.com/science/article/pii/S1084804512001944
44. Bhatt S, Manadhata PK, Zomlot L (2014) The operational role of security information and event management systems. IEEE Secur Priv 12(5):35–41. https://doi.org/10.1109/msp.2014.103
45. Li T, Zhao Y, Zhu Q (2022) The role of information structures in game-theoretic multi-agent learning. Ann Rev Control 53:296–314. https://doi.org/10.1016/j.arcontrol.2022.03.003
46. Li T, Zhu Q (2022) Commitment with signaling under double-sided information asymmetry. arXiv preprint arXiv:221211446. https://doi.org/10.48550/arXiv.2212.11446
47. Ge Y, Li T, Zhu Q (2023) Scenario-agnostic zero-trust defense with explainable threshold policy: a meta-learning approach. In: IEEE INFOCOM 2023 - IEEE conference on computer communications workshops (INFOCOM WKSHPS), pp 1–6. https://doi.org/10.1109/INFOCOMWKSHPS57453.2023.10225816
48. Ometov A, Bezzateev S, Mäkitalo N, Andreev S, Mikkonen T, Koucheryavy Y (2018) Multi-factor authentication: a survey. Cryptography 2(1):1
49. OpenAI (2023) Gpt-4 technical report. arXiv preprint arXiv:230308774. https://doi.org/10.48550/arXiv.2303.08774
50. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser Lu, Polosukhin I (2017) Attention is all you need. In: Guyon I, Luxburg UV, Bengio S, Wallach H, Fergus R, Vishwanathan S, Garnett R (eds) Advances in neural information processing systems. Curran Associates, Inc., vol 30. https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf
51. Kingma DP, Welling M (2014) Auto-encoding variational bayes. In: 2nd international conference on learning representations, ICLR 2014, Banff, April 14–16, 2014, Conference Track Proceedings. http://arxiv.org/abs/1312.6114v10
52. Paisley J, Blei DM, Jordan MI (2012) Variational Bayesian inference with stochastic search. In: Proceedings of the 29th international conference on international conference on machine learning. Omnipress, Madison, ICML'12, pp 1363–1370
53. Nash J (1951) Non-cooperative games. Ann Math 54(2):286–295. https://doi.org/10.2307/1969529
54. Fudenberg D, Tirole J (1991) Game theory. MIT Press, Cambridge
55. Li T, Zhu Q (2023) On the price of transparency: a comparison between overt persuasion and covert signaling. arXiv preprint arXiv:230400096. https://doi.org/10.48550/arXiv.2304.00096
56. Li T, Zhu Q (2019) On convergence rate of adaptive multiscale value function approximation for reinforcement learning. In: 2019 IEEE 29th international workshop on machine learning for signal processing (MLSP), pp 1–6. https://doi.org/10.1109/mlsp.2019.8918816
57. Li T, Peng G, Zhu Q (2021) Blackwell online learning for Markov decision processes. In: 2021 55th annual conference on information sciences and systems (CISS) 00:1–6. https://doi.org/10.1109/ciss50987.2021.9400319

58. Sutton RS, McAllester DA, Singh SP, Mansour Y (2000) Policy gradient methods for reinforcement learning with function approximation. In: Advances in neural information processing systems 12. MIT Press, pp 1057—1063. http://papers.nips.cc/paper/1713-policy-gradient-methods-for-reinforcement-learning-with-function-approximation.pdf
59. Bannon J, Windsor B, Song W, Li T (2020) Causality and batch reinforcement learning: complementary approaches to planning in unknown domains. arXiv preprint arXiv:200602579. https://doi.org/10.48550/arXiv.2006.02579
60. Puterman ML (1994) Markov decision processes: discrete stochastic dynamic programming, 1st edn. Wiley, New York
61. Hu J, Wellman MP (2003) Nash q-learning for general-sum stochastic games. J Mach Learn Res 4(Nov):1039–1069
62. Hammar K, Stadler R (2023) Digital twins for security automation. In: NOMS 2023–2023 IEEE/IFIP network operations and management symposium, pp 1–6. https://doi.org/10.1109/NOMS56928.2023.10154288
63. Li T, Lei H, Zhu Q (2022) Sampling attacks on meta reinforcement learning: a minimax formulation and complexity analysis. arXiv preprint arXiv:220800081. https://doi.org/10.48550/arXiv.2208.00081
64. Dazeley R, Vamplew P, Cruz F (2023) Explainable reinforcement learning for broad-XAI: a conceptual framework and survey. Neural Comput Appl 35(23):16893–16916. https://doi.org/10.1007/s00521-023-08423-1. https://doi.org/10.1007/s00521-023-08423-1
65. Ge Y, Zhu Q (2022) Trust threshold policy for explainable and adaptive zero-trust defense in enterprise networks. In: 2022 IEEE conference on communications and network security (CNS), pp 359–364. https://doi.org/10.1109/CNS56114.2022.9947263
66. Hospedales TM, Antoniou A, Micaelli P, Storkey AJ (2021) Meta-learning in neural networks: a survey. IEEE Trans Pattern Anal Mach Intell PP(99):1–1. https://doi.org/10.1109/tpami.2021.3079209
67. Pan Y, Li T, Li H, Xu T, Zheng Z, Zhu Q (2023) A first order meta Stackelberg method for robust federated learning. arXiv preprint arXiv:230613800. https://doi.org/10.48550/arXiv.2306.13800
68. Vapnik V (1999) The nature of statistical learning theory. Springer Science & Business Media, Berlin
69. Liu S, Li T, Zhu Q (2023) Game-theoretic distributed empirical risk minimization with strategic network design. IEEE Trans Signal Inf Process Netw 9:542–556. https://doi.org/10.1109/TSIPN.2023.3306106
70. Strom BE, Applebaum A, Miller DP, Nickels KC, Pennington AG, Thomas CB (2018) Mitre att&ck: design and philosophy. Technical report. The MITRE Corporation
71. Hochreiter SY (2001) Learning to learn using gradient descent. In: Lecture notes in computer science, pp 87–94. https://doi.org/10.1007/3-540-44668-0_13
72. Li Z, Zhou F, Chen F, Li H (2017) Meta-SGD: learning to learn quickly for few-shot learning. arXiv preprint arXiv: 170709835. https://doi.org/10.48550/arXiv.1707.09835
73. Yadav T, Rao AM (2015) Technical aspects of cyber kill chain. In: Security in computing and communications: third international symposium, SSCC 2015, Kochi, August 10–13, 2015. Proceedings 3. Springer, pp 438–452
74. Khan MS, Siddiqui S, Ferens K (2018) A cognitive and concurrent cyber kill chain model. In: Computer and network security essentials. Springer, Cham, pp 585–602
75. Huang L, Zhu Q (2019) Adaptive honeypot engagement through reinforcement learning of semi-Markov decision processes. In: Decision and game theory for security: 10th international conference, GameSec 2019, Stockholm, October 30–November 1, 2019, Proceedings 10. Springer, pp 196–216
76. Heckman KE, Stech FJ, Schmoker BS, Thomas RK (2015) Denial and deception in cyber defense. Computer 48(4):36–44
77. Gore R, Padilla J, Diallo S (2017) Markov chain modeling of cyber threats. J Def Model Simul 14(3):233–244
78. Kingman JFC (1976) Review of Discrete-Parameter Martingales, by Neveu, Jacques. J R Stat Soc A (Gen) 139(4):547–548

Part II
Security in Artificial Intelligence-Enabled Intrusion Detection Systems

Artificial Intelligence and Machine Learning for Network Security: Quo Vadis?

Michael J. De Lucia and Avinash Srinivasan

1 Introduction

The ever-growing world of interconnected devices and networks has catalyzed the rapid expansion of modern cyber attack surfaces. The new and rapidly evolving threat landscape can no longer be protected with basic network monitoring and analysis tools and traditional network intrusion detection systems. Despite significant and decades of research and development efforts, network intrusion detection systems still fall short when it comes to *improving detection accuracy*, *minimizing false alarm rates*, and *detecting novel (zero-day) intrusions*. As a result, artificial-intelligence (AI) and machine learning (ML) is increasingly becoming the focal point of many advancements in network intrusion detection systems.

AI/ML looks very promising with the potential to aid a small group of network analysts in a swift response to the myriad of sophisticated network attacks. On the other hand, in an effort to counter AI/ML driven network security and to evade detection, adversaries have engaged in new and never before seen attacks against the AI/ML models in network intrusion detection systems. Such attacks are commonly referred to as adversarial ML. There are various categories of adversarial ML attacks, and one such recent attack is the Clean-Label Poisoning attack [11, 48, 51]. The key objective of the adversary here is to evade detection. The evasion attack occurs at test time (i.e., real-time), while the Clean-Label poisoning attack occurs during training. Such failure to protect AI/ML based systems will cause adversarial

M. J. De Lucia (✉)
The Computational and Information Sciences Directorate, DEVCOM Army Research Laboratory, Aberdeen Proving Ground, MD, USA
e-mail: michael.j.delucia2.civ@army.mil

A. Srinivasan (✉)
Department of Cyber Science, United States Naval Academy, Annapolis, MD, USA
e-mail: srinivas@usna.edu

Y. Chen et al. (eds.), *Network Security Empowered by Artificial Intelligence*,
Advances in Information Security 107, https://doi.org/10.1007/978-3-031-53510-9_3

attacks to go unnoticed. Therefore, the defense (i.e., security) of machine learning based network intrusion detection systems is in and by itself very critical.

The defense of AI/ML systems have been considered in many other fields, such as image recognition. However, these defenses are specific to the field of image recognition and may not transfer to a network security context. The space that machine learning operates is referred to as the feature space, which is a representation of the problem space. For example, an actual image is considered the problem space and is transformed to the feature space as a matrix of pixel intensities. In the image domain there is an obvious inversion from the feature space back to the problem space. The network domain represents network flows (problem space) as a vector of features (e.g., statistical properties).

Conversely, in the network domain, there is not an obvious inverse from the feature space back to the problem space. Additionally, there is a lack of emphasis on the defense of traditional machine learning systems such as support vector machine (SVM), Random Forest, and Gradient Boosting, which are favored in the network intrusion detection domain. Our work proposes the use of ensembles as a defense against adversarial machine learning attacks on network intrusion detection systems. We believe a hierarchical, stacked, and nested ensemble will provide robust protection to machine learning based network intrusion detection systems. Additionally, unlike most previous work that have evaluated the security of AI/ML systems from the perspective of adversarial ML, we evaluate security of AI/ML systems from an end-to-end perspective that accounts for vulnerabilities in software dependencies and supply chain and discusses the need for a vulnerability disclosure program in AI/ML.

As AI/ML capabilities become more powerful and widespread, new attacks will stem from the use of AI/ML systems designed to complete extremely challenging tasks that would be otherwise impractical for humans. With the rapidly evolving state of the modern cyber attack surface it is realistic to expect attacks enabled by the growing use of AI/ML systems to be very effective as they can be precisely targeted making it very challenging to attribute [7]. At the intersection of cybersecurity and AI/ML attacks, authors highlight the need to explore and potentially implement red teaming, formal verification and responsible disclosure of AI/ML vulnerabilities among other things.

AI/ML systems have both civilian and military applications, and more importantly, toward both beneficial and harmful ends. Since some tasks that require intelligence are benign and other malicious, AI is a double-edge sword just like human intelligence is and therefore great caution has to be exercised when developing AI/ML systems. There is a large void in the domain of defense of AI/ML systems. Without adequate research and development of defenses, progress in AI/ML will only make matters worse. This can manifest through expansion of existing threats, introduction of new threats, or alteration of the typical character of threats.

1.1 Chapter Roadmap

The remainder of this chapter is organized as follows. Section 2 reviews network intrusion detection specifically focusing on the previously used basic network monitoring and analysis tools and techniques (Sect. 2.1) and discuss the traditional network intrusion detection systems (Sect. 2.2). Section 2.3 discusses the AI/ML driven advanced network intrusion detection systems. Section 3 presents discussions related to AI/ML systems' vulnerabilities. Section 4 presents discussions on the intersection of security and AI/ML. Specifically, Sect. 4.1 reviews employing AI/ML systems for Network security considerations and Sect. 4.2 discusses security considerations for building and deploying robust AI/ML systems. Finally, Sect. 5 concludes the paper with directions for future research.

2 Network Intrusion Detection Systems

The process of collecting, storing, and examining network traffic by dissecting the data packets that make up network traffic is commonly referred to as network traffic analysis. Network traffic analysis is a highly sophisticated process that often combines multiple techniques ranging from a simple static technique as rule-based detection to highly dynamic and adapting techniques like behavior modeling and Machine Learning. The primary objective of network traffic analysis is to establish a normal network operations behavior. A good baseline is vital to effective and timely detection and isolation of outliers. Therefore, for improved real-time situational awareness, network traffic analysis techniques should gather traffic in or near to real-time. However, data can be stored for advanced analysis referred to as Deep Packet Inspection (DPI). Figure 1 presents a broad classification of IDSs. In this section, we briefly discuss the basic network monitoring and analysis techniques in literature.

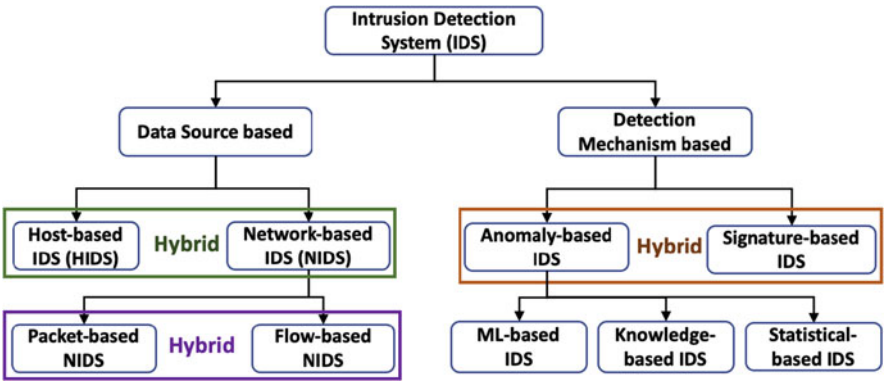


Fig. 1 Classification of IDS systems

2.1 Basic Network Monitoring and Analysis

With Network traffic monitoring and analysis, like with most cybersecurity solutions, one size does not fit all. The specific type of network traffic analysis solution deployed should meet the security requirements of the target network. A good Network traffic analysis solution can help detect anomalies in traffic pattern that can be an indicator of compromise (IOC) or malfunctioning network infrastructure component.

In [9], authors provide categorization of network traffic monitoring and analysis techniques into the following two techniques:

- **Router-based techniques**—techniques that utilize monitoring functionalities built-into the routers themselves. These techniques do not require installation of additional hardware or software. However, these techniques are hard-coded into the routers and therefore offer little flexibility. Some of the most popular router based network traffic monitoring and analysis techniques include: Simple Network Monitoring Protocol (SNMP), Remote Network Monitoring (RMON), and Netflow.
- **Non Router-based techniques**—are techniques that require installation of additional hardware and software. These monitoring techniques provide greater flexibility compared to router-based techniques and can be broadly classified into two categories: *Active* and *Passive*. Active monitoring technique transmits probes into the network to collect measurements between two or more endpoints, and tools such as ping and traceroute are examples of basic active measurement tools that fall into this category. Passive monitoring, on the other hand, does not inject traffic into the network. Instead, it passively collects information about only one point, unlike active monitoring that involves two or more endpoints, in the network that is being measured. A key advantage of passive monitoring over active monitoring is that the overhead is significantly lower compared to active monitoring. However, it does have a major drawback. Unlike active monitoring, data gathered through passive monitoring can only be analyzed off-line.

Another important consideration for effective Network traffic analysis is the data sources for your network monitoring tool. Two of the most popular data sources, which also happen to be the most popular techniques, for Network traffic analysis are captured in Fig. 1 and discussed below:

- **Packet data**—packet data capture typically involves capturing network packets using a mirror port. The captured data is simply a mirror image of the network packets. Packet data is better suited for application and user behaviour analysis. However, caution should be exercised when configuring a mirror port since they can easily get overloaded on a busy network. The most popular packet based intrusion detection systems are Bro [40] and Snort [42].
- **Flow data**—A network flow is a set of packets with the same characteristics passing through a specific observation point over a period of time [50]. Flow data based analysis provides excellent visibility on the traffic traversing across

different parts of the network. Capturing flow data is fairly simple to setup on devices which operate at layer 3 as it requires no software clients or agents on end user systems. However, the downside is that flow data lacks detail which prevents granular view of events. Additionally, flow data based network monitoring is not ideal for network edge where applications are encapsulated in other lower layer protocols. *nfdump* [23] is a toolset for storage and processing of flow records. Specifically, *nfdump* is a toolset to collect and process netflow/ipfix and sflow data, sent from netflow/sflow compatible devices. The toolset contains several collectors to collect flow data. *Nfsen* [24] is a graphical front-end for *nfdump*.

In [5], Azab et al. review existing network classification techniques, such as port-based identification and those based on deep packet inspection, statistical features in conjunction with machine learning, and deep learning algorithms. Authors discuss the implementations, advantages, and limitations associated with these techniques as well as existing and emerging challenges and future research directions. In [49], Zhao et al. categorize traffic classification techniques into the following five categories—port-based, payload-based, correlation-based, behavior-based, and statistical-based. Additionally, Zhao et al. provide analysis of workflow, advantages, disadvantages and deployed features for each of the five categories.

2.2 Traditional NIDS

In this section we discuss traditional NIDS, i.e., NIDS that are not driven by AI/ML models. In [30], authors provide a detailed review on taxonomy of contemporary IDS along with an overview of the data-sets commonly used for evaluation purposes. Authors also present evasion techniques used by attackers to avoid detection and discusses future research challenges to counter such adversary tactics.

Traditional network monitoring techniques for intrusion detection can be categorized into the following categories:

1. Signature-based IDS detect attacks based on pattern matching techniques to find known attacks. These IDSs often give excellent detection accuracy for intrusions that are previously known. Signature-based IDS are also known as Knowledge-based Detection or Misuse Detection [29].
2. Anomaly- or behavioral-based IDS [20] overcomes the limitations of Signature-based IDS. In contrast to signature-based IDSs, these systems can adapt to the evolving threat landscape. However, they do have a major downside—false positives generated by accidentally classifying unknown and legitimate activity as malicious. As discussed in [1, 30], Anomaly-based IDSs can be further classified into three broad categories—Machine Learning-based IDS, Statistical-based IDS, and Knowledge-based IDS [37]. We have presented a detailed classification of this classification in Fig. 2.

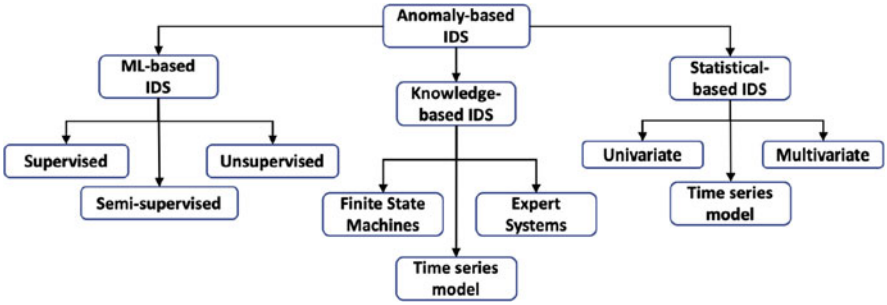


Fig. 2 Classification of anomaly-based IDS systems

While not as popular as the above two, there is a third network monitoring technique for intrusion detection referred to as Specification-based-detection [31]. Authors describe and implement a real-time intrusion detection system using a specification-based approach to detect exploitation of vulnerabilities in security-critical programs. Their proposed specification-based approach utilizes security specifications that describe the intended behavior of programs and scans audit trails for operations that are in violation of the specifications. Their approach encompasses attacks that exploit previously unknown vulnerabilities in security-critical programs.

2.3 Advanced NIDS with AI/ML

The advancement of AI has led to vast improvements in fields such as computer vision. As such, the cyber security community began to incorporate AI into intrusion detection systems and network traffic analysis. In [43] the authors discuss the feasibility of the use of ML for network intrusion detection. There are two main types of machine learning called, Supervised and Unsupervised. In Supervised methods, the model is learned based on the training dataset. In Unsupervised methods, there is no need for a training dataset. ML methods for network traffic analysis are sometimes referred to as anomaly based or misuse detection. Anomaly detection in this context is the identification of abnormal or malicious network communications. This implies comparing future network events against a known normal baseline to identify anomalous activity. In [43], the authors note that the term anomaly detection is used narrowly to refer to detection approaches that rely primarily on ML.

However, many of the ML network traffic classifiers utilize supervised machine learning and are composed of samples from both the benign and malicious class. The techniques that make up misuse detection is Knowledge-based, Statistical, and ML based IDS. Additionally ML-based can be synonymous with AI-based techniques which leverage deep learning models.

ML-based models for network traffic analysis examples include leverage Support Vector Machine (SVM), Random Forest, Decision Tree, and logistic regression. In [27], the authors provide a survey of many of the ML techniques used for network traffic analysis. The use of ML in network traffic classification has become an active research area. For example, the authors of [17, 26] leverage ML to perform network traffic classification tasks.

3 AI/ML Systems' Vulnerabilities

The cybersecurity breach of SolarWinds software is one of the most widespread and sophisticated hacking campaigns ever conducted against the federal government and private sector.[1] With the wide-spread impact and success of Solar-Winds attack, the adversary's attack strategy shifted to prioritize the supply chain instead of relying solely the attacker's arsenal of tools and techniques. It was very clear from SolarWinds, wherein a major cybersecurity company's software was severely impacted by a backdoor inserted during the supply chain, that attacking the supply chain for scalability, spreading exploits naturally through software upgrades, and leveraging trojan behavior to get into target networked systems [47] was a better option.

Like any technology, AI/ML as a technology enabler also has risks that can emerge in a variety of ways. NIST Artificial Intelligence Risk Management Framework (AI RMF 1.0) categorizes these risks as follows: long- or short-term, high or low-probability, systemic or localized, and high- or low-impact [45]. As illustrated in Fig. 3, the AI RMF Core is composed of four functions: GOVERN, MAP, MEASURE, and MANAGE.

AI/ML systems present new vulnerabilities compared to traditional enterprise and network security solutions having a more complex supply chain and dependency. As shown in Fig. 4, AI/ML systems present potential new vulnerabilities at four different stages: *data collection*, *model sourcing*, *operations tooling*, and *build and development*.

There are myriad ways in which attackers can cause AI/ML systems to behave unexpectedly and violate security policies—implicit or explicit. Adversaries may target the data sets, algorithms, or models that an ML system uses in order to deceive and manipulate their calculations, steal training data, compromise their operation, and render them ineffective. For instance, research in speech recognition domain has demonstrated the possibility of generating audio that sounds like speech to AI/ML algorithms but not to humans. Specifically, in [8], authors demonstrate how a voice interface system can be attacked with hidden voice commands that are unintelligible to human listeners but which are interpreted as commands by devices. Note that attacks can be successful even when the attackers have no access to either the model

[1] https://www.gao.gov/.

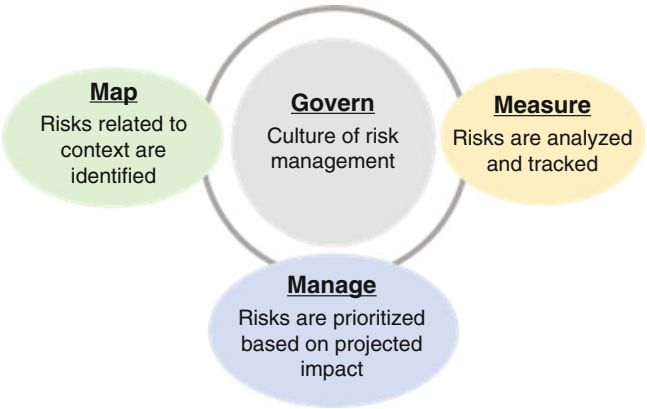
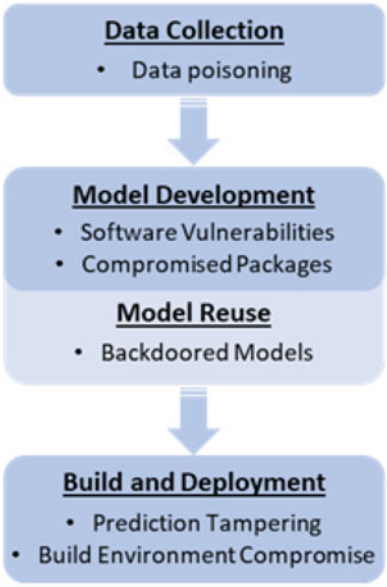


Fig. 3 NIST AI risk management framework activities [45]

Fig. 4 AI/ML supply chain vulnerabilities



or the data used to train it [39]. Microsoft chatbot *Tay*,[2] which started sending racist tweets within hours of being launched, is a classic example of data poisoning attacks on AI/ML systems.

AI/ML-systems can be quite easily manipulated, evaded, and misled resulting in profound implications, especially for network security and monitoring applications, as discussed in [43]. In [10], authors present examples where deep learning

[2] https://atlas.mitre.org/studies/AML.CS0009/.

methods can be misled by small amounts of input noise crafted by an adversary. Authors provide a detailed discussion on different types of adversarial attacks with various threat models and also elaborate the efficiency and challenges of recent countermeasures against them. Adversarial ML in the domains of image classification [33], facial recognition [2, 13, 32], audio deepfakes [28, 38], video deepfakes [22, 34], recommendation systems [18, 19], to name a few, are well known.

Finally, AI/ML models have substantial dependencies on software packages which makes them vulnerable to supply chain attacks (Fig. 4). One recent supply chain vulnerability that was uncovered that potentially impacted a broad spectrum on AI/ML tools is the PyTorch-nightly Python package Torchtriton. PyTorch is a popular open-source machine-learning framework that is used in applications like natural language processing. In December 2023, a security researcher uploaded a malicious package with the same name and a higher version of PyTorch-nightly dependency Torchtriton to the Python Package Index (PyPI) code repository. This subsequently resulted in a dependency confusion. It is important to note that if the name of a private package is available on PyPI, an attacker can simply exploit this by uploading a malicious package with the same name but with a higher version. This will result in a supply chain attack. In this particular case, the malicious version of Torchtriton included code that uploads sensitive data from the end-user machine.

Below is a list of other python packages that are frequently used in AI/ML development that have had vulnerabilities with varying levels of security risk.

1. **NumPy**—NumPy is a Python library that supports large, multi-dimensional arrays and matrices. It also has a large collection of high-level mathematical functions to operate on these arrays [25]. Below are some of the notable vulnerabilities in the NumPy python library.

   - CVE-2014-1858/1859—NumPy before 1.8.1 allows local users to write to arbitrary files via a symlink attack on a temporary file.
   - CVE-2017-12852—Numpy 1.13.1 and older versions is missing input validation in the numpy.pad function. Consequently, an empty list or *ndarray* will stick into an infinite loop, which can allow attackers to cause a DoS attack.
   - CVE-2019-6446[3]—NumPy 1.16.0 and earlier uses the pickle Python module unsafely that allows remote attackers to execute arbitrary code via a crafted serialized object.
   - CVE-2021-33430[4]—A Buffer Overflow vulnerability exists in NumPy 1.9.x in *ctors.c* when specifying arrays of large dimensions from Python code. This can be exploited by a malicious user to cause a Denial of Service.
   - CVE-2021-34141—An incomplete string comparison in the numpy.core component in NumPy before 1.22.0 allows attackers to trigger slightly incorrect copying by constructing specific string objects.

[3] Note: This is a disputed CVE.

[4] See footnote 3.

- CVE-2021-41495[5]—Null Pointer Dereference vulnerability exists in *numpy.sort* in NumPy 1.19 due to missing return-value validation in the *PyArray_DescrNew* function. This allows a malicious user to conduct DoS attacks by repetitively creating sort arrays.

2. **scikit-learn (sklearn)**—sklearn is a free machine learning library for the Python programming language [41].
   - CVE-2020-28975[6]—svm_predict_values in svm.cpp in Libsvm v324, as used in scikit-learn 0.23.2 and other products, allows attackers to cause a denial of service (segmentation fault) via a crafted model SVM (introduced via pickle, json, or any other model permanence standard) with a large value in the *_n_support* array.
   - CVE-2020-13092[7]—sklearn through 0.23.0 can unserialize and execute commands from an untrusted file that is passed to the joblib.load() function.
3. **Natural Language ToolKit (NLTK)**—NLTK is a framework and suite of libraries for developing both symbolic and statistical Natural Language Processing (NLP) in Python. NLTK support different ML operations like classification, parsing, and tokenization functionalities. Below are notable NTLK vulnerabilities.
   - CVE-2021-43854—Versions prior to 3.6.5 are vulnerable to regular expression denial of service (ReDoS) attacks.
   - CVE-2021-3828/3842—vulnerable to Inefficient Regular Expression Complexity
   - CVE-2019-14751—Downloader before 3.4.5 is vulnerable to a directory traversal, allowing attackers to write arbitrary files via a ../ (dot dot slash) in an NLTK package (ZIP archive) that is mishandled during extraction.
4. **Pandas**—pandas is a python library for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series.
   - CVE-2020-13091[8]—pandas through 1.0.3 can unserialize and execute commands from an untrusted file that is passed to the *read_pickle()* function, if *__reduce__* makes an *os.system* call.
5. **Tensor Flow**[9]—TensorFlow is a free and open-source software library for AI/ML. It can be used across a range of tasks but has a particular focus on training and inference of deep neural networks.

[5] See footnote 3.

[6] See footnote 3.

[7] See footnote 3.

[8] See footnote 3.

[9] This is not just a python library, TensorFlow libraries exist for Java, C++, etc.

- CVE-2023-25668—Attackers using Tensorflow prior to 2.12.0 or 2.11.1 enables attackers access to heap memory leading to a crash or remote code execution.
- CVE-2023-25676—When running versions prior to 2.12.0 and 2.11.1 with *XLA*, *tf.raw_ops.ParallelConcat* segfaults with a *nullptr* dereference when given a parameter *'shape'* with rank that is not greater than zero.

4 Intersection of Security and AI/ML

In this section, we discuss the intersection of AI/ML and security from two viewpoints. First, we discuss the application of AI/ML systems for network intrusion detection, it's associate risks and how to effectively counter the risks. Second, we discuss the security of AI/ML systems used for network intrusion detection.

4.1 AI/ML for Network Security

The increased use of AI/ML in a security context requires an understanding of the threats from advanced adversaries. The coupling of machine learning with a network security application, increases the attack surface. Subsequently, the development of a defense of AI/ML based network security classifiers against adversarial attacks is essential.

4.1.1 Adversarial Machine Learning

Adversarial machine learning (AML) is the ability of an attacker to cause a classifier's (i.e., model) misclassification. AML encompasses a variety of attacks, but our focus is on evasion and poisoning. An evasion attack is an attacker's perturbation of a sample during prediction time to cause misclassification. Whereas, a poisoning attack takes place during the training process of a classifier, to cause misclassification. There are two different common types of poisoning attacks. The first type of poisoning is called an availability attack. Which is a non-targeted type of attack. The second type is commonly referred to as a Trojan. In a Trojan attack, the adversary targets a specific class to poison. Thereby causing misclassification toward a targeted class.

4.1.2 Adversarial Machine Learning for Network Security

Traditionally, AML has been largely focused on the image detection domain. Recently, a limited number of AML studies have shifted the focus toward the cyber security domain [12, 15]. A distinction in AML for the cyber domain is the

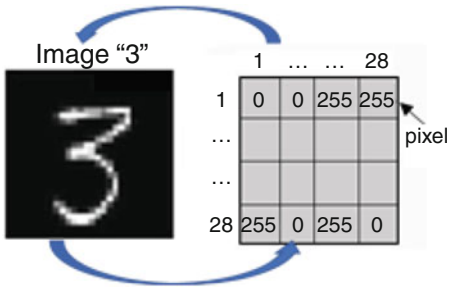


Fig. 5 Example of Image transformation from problem to feature space and the inversion

Fig. 6 Example of a network flow (i.e., problem space) transformation to features and the inability to inverse it

difficulty of inversion from the feature to problem space. The feature space is the numerical values that describe a sample. For example, the matrix of pixel intensities (i.e., numerical values), seen in the right of Fig. 5, can be converted to a vector in feature space. The problem space is an actual image seen in the left of Fig. 5. For the image detection domain, the transition from problem to feature space is easily inverted. Figure 5 demonstrates the transformation from an image to a matrix of pixel intensities in the feature space. Also, Fig. 5 demonstrates the ability to invert the feature space to an image.

In Fig. 6, a network flow is transformed into a feature vector. These features are statistics about the network flow. However, the inverse transformation in Fig. 6 is not intuitive. Thus, the transformation from feature space to a network flow, requires an extra step to discover the actual network traffic flow to represent the statistics. Therefore, AML in the cyber domain requires an extra step of the attacker to translate the feature space perturbations to actual network traffic. Another words, the transformation of a feature vector to actual network traffic is ad-hoc and difficult.

The features of a machine learning algorithm for a network security context are “hand crafted” by subject matter experts. Therefore, the constraints in a network security classifier scenario are ad hoc. The features of a ML algorithm are the target of perturbation by an attacker, to evade detection [14]. The perturbations are constrained by physical and mathematical properties [12]. For example, the attacker can not perturb a network flow to be negative. An example of a mathematical constraint is an attacker perturbation of the total number of bytes in a network flow, will require the respective update of the average number of bytes.

Many of the prior studies have focused on the feature space and have not converted the adversarial samples to the problem space to evaluate the effectiveness

of evasion, while preserving the malicious intent. Thus, prior studies evaluated AML from the feature space. However, feasible attacks must be evaluated in the problem space and are more realistic. Feature space attacks only focus on perturbing the values to produce adversarial samples. Thus, requiring an attacker to investigate malicious methods to reproduce the actual network traffic reflecting the feature values of the adversarial sample. Whereas, a problem space attack directly perturbs the network traffic to produce an adversarial sample. For example, these attacks could be realized in the problem space by modifying the scanning speed parameter of a tool such as nmap. The authors in [3] suggest that problem space attacks are directly implemented and feasible in a realistic scenario. Furthermore, in [4], the authors suggest that real attackers do not leverage the traditional feature space adversarial attacks. Thus implying that problem space attacks are feasible, effective, and less time consuming for real attackers.

4.1.3 Countermeasures

The defense of network security models against both adversarial evasion and poisoning attacks are essential. Ensembles have been shown to provide a defense against both evasion and poisoning attacks in [16, 46]. An ensemble is composed of a group of individual weak classifiers. In Fig. 7, an example of an ensemble is shown. The individual classifiers in the ensemble contain subsets of features or the training dataset. In [16], the defense against an evasion attack is proposed using a hierarchical ensemble composed of classifiers that use disparate feature sets. Each classifier evaluates the network traffic from a different *perspective*. An analogy of evaluation using different perspectives is an biometric identification system composed of a retinal scan and fingerprints.

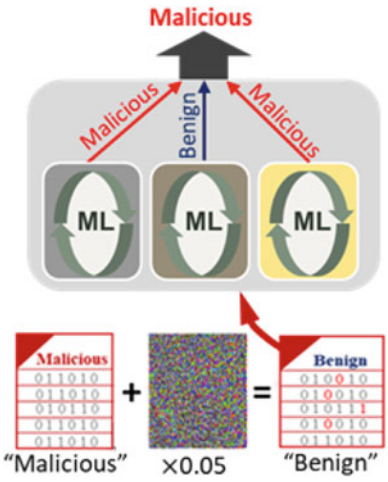


Fig. 7 Example of an ensemble and an AML evasion attack sample as input

A hierarchical ensemble can be thought of as a defense in depth. A defense in depth is a well know computer security methodology where each layer addresses a vulnerability of another layer. Thus, as a whole, the layers of a defense in depth provide a stronger protection. Similarly, each layer of the ensemble uses disparate feature sets to augment each other and evaluate the network traffic from different perspectives.

A simplistic example of a defense in depth hierarchical ensemble contains two layers using disparate feature sets. A single layer can evaluate network communications using statistical features of the flows. While another layer can evaluate the DNS requests of network communications using a natural language model and associated features. Thus, each layer evaluates the network traffic from a different perspective. Consequently, the hierarchical ensemble increases the cost of a successful attack. That is the attacker needs to obtain domain knowledge in network communications and natural language processing. Additionally, incurring an increase in time required to implement and execute two very different attack types.

Ensembles are versatile and can also defend against adversarial poisoning attacks. In [46], the authors use a nested ensemble to defend against poisoning attacks. Their work, subsequently leverages subsets of the data and relies on disagreements among the members of the ensemble to identify the presence of poisoned samples. Their methodology, also produces a sufficiently cleansed dataset to restore a baseline performance. However this method relies on the use of a large training dataset, since the resultant clean dataset size has been significantly reduced.

These defenses have been evaluated from adversarial samples generated in the feature space, while being aware of constraints. However, evaluation has not been adequately performed using a problem space attack. Notionally, the use of ensembles to defend against an adversarial attack translates from feature to problem space. Since, the purpose of these ensembles is to evaluate samples from different perspectives by either varying the features or subsets of data. Additionally, an adversarial problem space sample is a perturbation or mutation of the original malicious network traffic. Thus, allowing each ensemble member to augment the other to provide a more robust detection.

4.2 Security Considerations for AI/ML

AI/ML systems need innovative cybersecurity tools and methods to improve their trustworthiness and resiliency. Similarly, cybersecurity can benefit by utilizing AI/ML to increase awareness, react in real-time, and more importantly improve its overall effectiveness. This is particularly critical to self-adaptation and adjustment in the face of ongoing attacks that impact the attacker-defender asymmetry [35]. In a panel on "AI for Security and Security for AI [6]," panelists discuss how AI systems are systematically vulnerable to a new class of vulnerabilities and how the adversary is exploiting these vulnerabilities to alter AI system behavior to serve a

malicious end goal. In [7], authors present potential security threats from malicious uses of AI/ML technologies, and proposes ways to better forecast, prevent, and mitigate these threats. The authors in [7], note that cybersecurity must be a major and ongoing priority in an effort to prevent and mitigate harms from AI systems, and best practices from cybersecurity must be ported over wherever applicable to AI systems.

Sommer and Paxson [43] note that a common assumption that is made in intrusion detection is that attacks exhibit characteristics that are different than those of normal traffic. This, however, is not true. A sophisticated adversary can fine tune the attack traffic such that its deviation 'Δ' is small enough to defeat the AI/ML system's expected deviation. An AI/ML method works well when it is used with data that is very similar to what it was trained on. Else, if the testing data is different, it fails. A very good example illustrating this phenomenon can be witnessed in the domain of self-driving cars [7, 35]. A self-driving car trained in sunny, cloudy, rainy, and snowy weather might still perform quite poorly in sleet or hail. These are common problems because in all application domains it is extremely difficult to acquire data for all possible operation scenarios. In particular, when used for network intrusion detection, the highly volatile and dynamic operating conditions of an enterprise network make it very challenging to develop a robust AI/ML model.

One key consideration for the security of AI/ML models developed for network intrusion detection is having a robust supply chain security. Any and all hardware, software and open-source libraries used in developing AI/ML systems should be thoroughly vetted and rigorously tested prior to deployment. A second key consideration, complimenting the first one, is to have a robust vulnerability disclosure and management program in place. There are a growing number of vulnerabilities in AI/ML, and its use increases the attack surface of existing systems. While the objective is to keep the AI/ML supply chain secured air-tight end-to-end and all the systems using the AI/ML, it is realistic to expect vulnerabilities to exist. However, what is lacking is how soon it is uncovered and once uncovered how effectively is it disclosed in a responsible, effective and timely manner. The answer is a robust vulnerability disclosure and management program that will help navigate the situation better and mitigate the damage while containing it.

ATLAS, short for *Adversarial Threat Landscape for Artificial-Intelligence Systems*, developed by MITRE [36], enables one to navigate the landscape of threats to machine learning systems. Furthermore, applying the cybersecurity policies of vulnerability disclosure and management to AI/ML can both heighten the appreciation and help better manage the cybersecurity risk associated with AI/ML systems [21]. In [44], authors explore how the current paradigm of vulnerability management could potentially be adapted to include AI/ML systems by considering the possibility of assigning Common Vulnerabilities and Exposures (CVE) identifiers (CVE-IDs) to vulnerabilities in AI/ML systems as they are uncovered.

5 Conclusion

A common assumption that is made when employing AI/ML systems for network intrusion detection is that attacks exhibit characteristics that are different than those of normal traffic. This, however, is not true. Additionally, network statistics can be produced for malicious network traffic that looks sufficiently similar to normal communications. Thereby, tricking the AI/ML model. An AI/ML model works well when it is used with data that is very similar to what it was trained on, else it fails. Adversary samples, are outside of the space of the training dataset. As AI/ML capabilities become more powerful and widespread, new attacks will stem from the use of AI/ML systems. Therefore, it is realistic to expect attacks enabled by the growing use of AI/ML systems to be very effective as they can be precisely targeted making it very challenging to attribute. AI/ML is a double-edge sword just like human intelligence is and therefore great caution has to be exercised when developing AI/ML systems. Without adequate research and development of defenses, progress in AI will lead to serious security threats from *expansion of existing threats*, *introduction of new threats*, and *alteration of the typical character of threats*. Developing robust supply chain security and AI/ML vulnerability disclosure program will be two important considerations along with adopting recommendations from NIST AI-RMF and ATLAS [36]. As with any other technology, the vulnerability of an AI-based system will depend on three aspects—vulnerabilities of the AI/ML model(s) used by the system, the environment in which the AI/ML system is deployed, and the other systems that interact with the deployed AI/ML system. In this chapter, unlike most previous work that have evaluated the security of AI/ML systems from the adversarial ML view point, we evaluate the security of AI/ML systems from an end-to-end perspective that accounts for vulnerabilities in software dependencies and supply chain and discusses the importance and need for a AI/ML vulnerability disclosure and management program.

References

1. Aldweesh A, Derhab A, Emam AZ (2020) Deep learning approaches for anomaly-based intrusion detection systems: A survey, taxonomy, and open issues. Knowl Based Syst 189:105124. https://doi.org/10.1016/j.knosys.2019.105124. https://www.sciencedirect.com/science/article/pii/S0950705119304897
2. Alparslan Y, Alparslan K, Keim-Shenk J, Khade S, Greenstadt R (2020) Adversarial attacks on convolutional neural networks in facial recognition domain. Preprint. arXiv:200111137
3. Apruzzese G, Andreolini M, Ferretti L, Marchetti M, Colajanni M (2022) Modeling realistic adversarial attacks against network intrusion detection systems. Digital Threats Res Pract (DTRAP) 3(3):1–19
4. Apruzzese G, Anderson HS, Dambra S, Freeman D, Pierazzi F, Roundy K (2023) "real attackers don't compute gradients": Bridging the gap between adversarial ML research and practice. In: 2023 IEEE Conference on secure and trustworthy machine learning (SaTML). IEEE, pp 339–364

5. Azab A, Khasawneh M, Alrabaee S, Raymond Choo KK, Sarsour M (2022) Network traffic classification: Techniques, datasets, and challenges. Digit Commun Netw. https://doi.org/10.1016/j.dcan.2022.09.009. https://www.sciencedirect.com/science/article/pii/S2352864822001845
6. Bertino E, Kantarcioglu M, Akcora CG, Samtani S, Mittal S, Gupta M (2021) Ai for security and security for AI. In: Proceedings of the eleventh ACM conference on data and application security and privacy, Association for Computing Machinery, New York, NY, USA, CODASPY '21, pp 333–334
7. Brundage M, Avin S, Clark J, Toner H, Eckersley P, Garfinkel B, Dafoe A, Scharre P, Zeitzoff T, Filar B, et al (2018) The malicious use of artificial intelligence: Forecasting, prevention, and mitigation. Preprint. arXiv:180207228
8. Carlini N, Mishra P, Vaidya T, Zhang Y, Sherr ME, Shields C, Wagner DA, Zhou W (2016) Hidden voice commands. In: USENIX security symposium
9. Cecil A (2006) A summary of network traffic monitoring and analysis techniques. Comput Syst Anal:4–7
10. Chakraborty A, Alam M, Dey V, Chattopadhyay A, Mukhopadhyay D (2018) Adversarial attacks and defences: A survey. 1810.00069
11. Chen K, Lou X, Xu G, Li J, Zhang T (2023) Clean-image backdoor: Attacking multi-label models with poisoned labels only. In: The eleventh international conference on learning representations
12. Chernikova A, Oprea A (2022) Fence: Feasible evasion attacks on neural networks in constrained environments. ACM Trans Privacy Secur 25(4):1–34
13. Cilloni T, Wang W, Walter C, Fleming C (2022) Ulixes: Facial recognition privacy with adversarial machine learning. Proc Priv Enhancing Technol 2022(1):148–165
14. De Lucia MJ, Cotton C (2018) Importance of features in adversarial machine learning for cyber security. In: Proceedings of the Conference on Information Systems Applied Research ISSN, vol 2167, p 1508
15. De Lucia MJ, Cotton C (2019) Adversarial machine learning for cyber security. J Inf Syst Appl Res 12(1):26
16. De Lucia MJ, Cotton C (2020) A network security classifier defense: against adversarial machine learning attacks. In: Proceedings of the 2nd ACM workshop on wireless security and machine learning, pp 67–73
17. De Lucia MJ, Maxwell PE, Bastian ND, Swami A, Jalaian B, Leslie N (2021) Machine learning raw network traffic detection. In: Artificial intelligence and machine learning for multi-domain operations applications III, SPIE, vol 11746, pp 185–194
18. Deldjoo Y, Noia TD, Merra FA (2020) Adversarial machine learning in recommender systems: State of the art and challenges. ArXiv abs/2005.10322
19. Deldjoo Y, Noia TD, Merra FA (2020) A survey on adversarial recommender systems: from attack/defense strategies to generative adversarial networks. Arxiv abs/2005.10322
20. Ellis DR, Aiken JG, Attwood KS, Tenaglia SD (2004) A behavioral approach to worm detection. In: Proceedings of the 2004 ACM workshop on rapid malcode, Association for Computing Machinery, New York, NY, USA, WORM '04, pp 43–53. https://doi.org/10.1145/1029618.1029625
21. Grotto A, Dempsey J (2021) Vulnerability disclosure and management for AI/ML systems: A working paper with policy recommendations. In: ML Systems: a working paper with policy recommendations (November 15, 2021)
22. Güera D, Delp EJ (2018) Deepfake video detection using recurrent neural networks. In: 2018 15th IEEE international conference on advanced video and signal based surveillance (AVSS). IEEE, pp 1–6
23. Haag P (2004) nfdump – a toolset to collect and process netflow/ipfix and sflow data. https://github.com/phaag/nfdump
24. Haag P (2011) Nfsen – a graphical web based front end for the nfdump netflow tools. https://nfsen.sourceforge.net/

25. Harris CR, Millman KJ, Van Der Walt SJ, Gommers R, Virtanen P, Cournapeau D, Wieser E, Taylor J, Berg S, Smith NJ, et al (2020) Array programming with numpy. Nature 585(7825):357–362
26. Holland J, Schmitt P, Feamster N, Mittal P (2021) New directions in automated traffic analysis. In: Proceedings of the 2021 ACM SIGSAC conference on computer and communications security, pp 3366–3383
27. Joshi M, Hadi TH (2015) A review of network traffic analysis and prediction techniques. Preprint. arXiv:150705722
28. Khanjani Z, Watson G, Janeja VP (2021) How deep are the fakes? Focusing on audio deepfake: A survey. Preprint. arXiv:211114203
29. Khraisat A, Gondal I, Vamplew P (2018) An anomaly intrusion detection system using c5 decision tree classifier. In: Trends and applications in knowledge discovery and data mining: PAKDD 2018 workshops, BDASC, BDM, ML4Cyber, PAISI, DaMEMO, Melbourne, VIC, Australia, June 3, 2018, Revised Selected Papers 22. Springer, pp 149–155
30. Khraisat A, Gondal I, Vamplew P, Kamruzzaman J (2019) Survey of intrusion detection systems: techniques, datasets and challenges. Cybersecurity 2(1):1–22
31. Ko C, Ruschitzka M, Levitt K (1997) Execution monitoring of security-critical programs in distributed systems: a specification-based approach. In: Proceedings. 1997 IEEE symposium on security and privacy (Cat. No.97CB36097), pp 175–187. https://doi.org/10.1109/SECPRI.1997.601332
32. Korshunov P, Marcel S (2018) Deepfakes: a new threat to face recognition? assessment and detection. Preprint. arXiv:181208685
33. Machado GR, Silva E, Goldschmidt RR (2021) Adversarial machine learning in image classification: A survey toward the defender's perspective. ACM Comput Surv 55(1):1–38. https://doi.org/10.1145%2F3485133
34. Masi I, Killekar A, Mascarenhas RM, Gurudatt SP, AbdAlmageed W (2020) Two-branch recurrent network for isolating deepfakes in videos. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part VII 16. Springer, pp 667–684
35. McDaniel P, Launchbury J, Martin B, Wang C, Kautz H (2020) Artificial intelligence and cyber security: Opportunities and challenges technical workshop summary report. Networking & Information Technology Research and Development Subcommittee and the Machine Learning & Artificial Intelligence Subcommittee of the National Science & Technology Council
36. MITRE (2021) Atlas – adversarial threat landscape for artificial-intelligence systems. https://atlas.mitre.org/
37. More S, Matthews M, Joshi A, Finin T (2012) A knowledge-based approach to intrusion detection modeling. In: 2012 IEEE symposium on security and privacy workshops, pp 75–81. https://doi.org/10.1109/SPW.2012.26
38. Müller NM, Pizzi K, Williams J (2022) Human perception of audio deepfakes. In: Proceedings of the 1st international workshop on deepfake detection for audio multimedia, pp 85–91
39. Papernot N, McDaniel P, Goodfellow I, Jha S, Celik ZB, Swami A (2017) Practical black-box attacks against machine learning. In: Proceedings of the 2017 ACM on asia conference on computer and communications security, Association for Computing Machinery, New York, NY, USA, ASIA CCS '17, pp 506–519
40. Paxson V (1998) Bro: a system for detecting network intruders in real-time. Comput Netw 31:2435–2463
41. Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V, Vanderplas J, Passos A, Cournapeau D, Brucher M, Perrot M, Édouard Duchesnay (2011) Scikit-learn: Machine learning in python. J Mach Learn Res 12(85):2825–2830. http://jmlr.org/papers/v12/pedregosa11a.html
42. Roesch M (1999) Snort: Lightweight intrusion detection for networks. In: LiSA
43. Sommer R, Paxson V (2010) Outside the closed world: On using machine learning for network intrusion detection. In: 2010 IEEE symposium on security and privacy, pp 305–316

44. Spring JM, Galyardt A, Householder AD, VanHoudnos N (2021) On managing vulnerabilities in ai/ml systems. In: New security paradigms workshop 2020 (NSPW '20), Association for Computing Machinery, New York, NY, USA, pp 111–126. https://doi.org/10.1145/3442167.3442177
45. Tabassi E (2023) Artificial intelligence risk management framework (AI RMF 1.0)
46. Venkatesan S, Sikka H, Izmailov R, Chadha R, Oprea A, De Lucia MJ (2021) Poisoning attacks and data sanitization mitigations for machine learning models in network intrusion detection systems. In: MILCOM 2021-2021 IEEE military communications conference (MILCOM). IEEE, pp 874–879
47. Williams D, Clark C, McGahan R, Potteiger B, Cohen D, Musau P (2022) Discovery of ai/ml supply chain vulnerabilities within automotive cyber-physical systems. In: 2022 IEEE international conference on assured autonomy (ICAA), pp 93–96. https://doi.org/10.1109/ICAA52185.2022.00020
48. Zhang C, Tang Z, Li K (2023) Clean-label poisoning attack with perturbation causing dominant features. Inf Sci. https://doi.org/10.1016/j.ins.2023.03.124
49. Zhao J, Jing X, Yan Z, Pedrycz W (2021) Network traffic classification for data fusion: A survey. Inf Fusion 72:22–47
50. Zhou D, Yan Z, Fu Y, Yao Z (2018) A survey on network data collection. J Netw Comput Appl 116:9–23. https://doi.org/10.1016/j.jnca.2018.05.004. https://www.sciencedirect.com/science/article/pii/S1084804518301607
51. Zhu C, Huang WR, Shafahi A, Li H, Taylor G, Studer C, Goldstein T (2019) Transferable clean-label poisoning attacks on deep neural nets. In: International conference on machine learning

Understanding the Ineffectiveness of the Transfer Attack in Intrusion Detection System

Rui Duan, Wenwei Zhao, Zhengping Jay Luo, Ning Wang, Yao Liu, and Zhuo Lu

1 Introduction

With the increasing prevalence of security concerns in the networking domain, the intrusion detection system (IDS) has developed into a crucial tool for detecting and safeguarding against network attacks propagated through manipulated network traffic. Recently, IDS has been empowered via machine learning to detect unsafe networking traffic behaviors. Specifically, IDS can leverage the extracted features to identify whether the traffic packet is malicious or benign. Several popular machine learning methods have been employed in IDS, including Support Vector Machine (SVM), Decision Tree (DT), Multilayer Perceptron (MLP), and K-nearest Neighbors (KNN).

However, the state-of-the-art IDS faces the same problem as machine learning models: they are all vulnerable to adversarial examples (AEs). Attackers can manipulate the original traffic packets by adding a small perturbation to revise the network packets' labels as predicted by the IDS. Different attacks have shown that IDS is vulnerable to AEs with varying levels of attack knowledge, such as white-box attacks [2, 43, 48], gray-box attacks [18, 22, 24], and black-box attacks [28, 56, 59]. These attacks have more or less real-world impact depending on the attack knowledge; in other words, the less knowledge the attacker has, the greater the real-world impact. However, we are confused about whether the transfer attack

R. Duan · W. Zhao · N. Wang · Y. Liu · Z. Lu (✉)
University of South Florida, Tampa, FL, USA
e-mail: ruiduan@usf.edu; wenweizhao@usf.edu; ningw@usf.edu; yliu@cse.usf.edu; zhuolu@usf.edu

Z. J. Luo
Rider University, Lawrenceville, NJ, USA
e-mail: zluo@rider.edu

Y. Chen et al. (eds.), *Network Security Empowered by Artificial Intelligence*,
Advances in Information Security 107, https://doi.org/10.1007/978-3-031-53510-9_4

is still effective in machine-learning-based IDS, where attackers have no knowledge and cannot query the target models.

To this end, we aim to explore the transferability of AEs in the network domain. Our objective is to understand the attack factors that can influence the transfer of AEs to IDS models. We primarily focus on three attack factors: (i) different attack algorithms [29], (ii) various training datasets [32], and (iii) different model architectures. Building upon the findings related to these attack factors, we propose to generate highly effective AEs and compare them with white-box-based AEs to understand the ineffectiveness of transfer AEs in IDS. In summary, we present the following contributions:

1. We evaluate a wide range of existing white-box attacks, such as CW, FGSM, BIM, and JSMA, with different attack factors, selecting the best-performing attacks as benchmarks for further comparison of transfer attacks.
2. We evaluate different training factors to build surrogate models, and investigate various levels of training and testing to identify a feasible method for constructing surrogate models with high transferability.
3. We use different attack algorithms to generate transfer AEs, highlighting the CW attack as the most effective method for generating high-transfer AEs, in line with the findings from the white-box benchmarks. Additionally, we discover that the perturbation norm has an impact on transferability within a specific range.

The organization of this chapter is: Sect. 2 introduces the background of the adversarial attack and Intrusion Detection System. Section 3 presents the process of building the surrogate models, and the evaluation of the effect of building surrogate models with different training factors. We investigate the transferability of AEs in Sect. 4. Finally, we conclude this chapter in Sect. 5.

2 Background of Adversarial Attack on Intrusion Detection System

In this section, we first take a look at the background of IDS, and IDS commonly has two classes: signature-based detection and anomaly-based IDS, and we mainly focus on the latter case. We then present the common adversarial attacks to machine learning and discuss the related adversarial attacks to the IDS.

2.1 Intrusion Detection System

An intrusion detection system (IDS), empowered by a machine learning model, is a state-of-the-art defense mechanism designed to prevent various attacks on networks

[26]. Specifically, IDS can be related to the OS information, which is referred to as host-based IDS, or it can focus on analyzing network packets, such as IP address and protocol usage. We refer to the latter as network-based IDS. In this chapter, our primary focus is on network-based IDS, and there are two common types of models: knowledge-based and machine learning-based schemes.

1. Knowledge-based mechanism: signature-based detection[33], as the representative of the knowledge-based IDS, commonly compares the extracted traffic features with the pre-built knowledge to predict well-known attacks, but it is ineffective to the unfamiliar attacks which are outside the pre-built knowledge, even if these unknown attacks only have small deviation to the known attacks.
2. Machine learning-based techniques: To get rid of the limitation of the knowledge, anomaly-based detection is built with machine-learning schemes, which are flexible to detect malicious behavior via the deviation between the observed network packets and normal traffic. Compare to signature-based detection, anomaly-based IDS is feasible to detect various attacks with limited knowledge or even no knowledge of the novel attacks. However, as the common issue for the machine learning model, anomaly-based detection is vulnerable to adversarial examples (AEs).

Anomaly-based IDS is more popular to detect malicious behavior. As shown in Fig. 1, there are commonly three stages for the Anomaly-based IDS. The first stage is *Parameterization*, which will process the monitored samples and extract the features for the next stage. The next is *Training stage* that train the normal or abnormal labeled traffic packets via a machine-learning model. And we can consider this model as an IDS model. Lastly, for the *Detection stage*, the trained IDS model can be used to classify a traffic instance as malicious or not, and output an alert if the input has malicious behavior. However, a new adversarial attack shows a threat to this kind of IDS mechanism. Thus, we propose to review the common adversarial attack, and then explore how these attacks can be employed in the IDS.

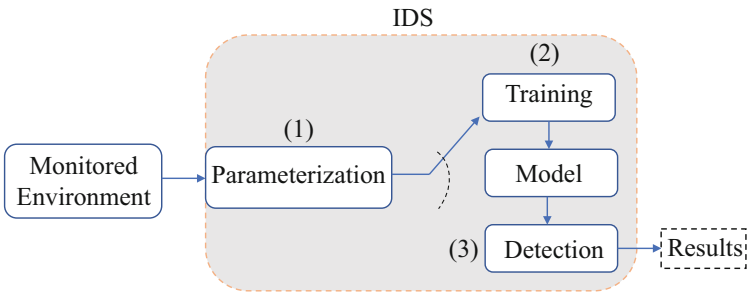


Fig. 1 Anomaly-based IDS architecture

2.2 Adversarial Attacks and Formulation

An IDS classifier can be denoted as a function $f(\cdot)$, which processes the traffic package x as input, i.e., the parameterization stage. In the detection stage, it will output a probability score $P = [p_0, p_1, \cdots, p_{k-1}]$, where $p_i \in [0, 1]$, and $\sum_{i=1}^{k-1} p_i = 1$ for the set of k classes, i.e., normal or abnormal behavior label. The predicted label will be the class with the highest probability p_i, and $y_{\text{pred}} = \arg max(f(x))$.

Existing adversarial attacks aim to add some small perturbation δ that can mislead the model to predict a wrong label, $f(x + \delta) \neq y$, where y is the ground truth label of input traffic packers x. Existing adversarial attacks can be categorized into three classes according to different knowledge levels. White-box attacks assume full knowledge of the target model, including the training dataset, parameters, and algorithms, while gray-box and black-box attacks have only partial or even no knowledge of the target model, respectively. As shown in Table 1, there are some popular white-box attacks to IDS, and they have different computational complexity and attack effectiveness, these attacks including CW[9], FGSM [17], BIM [23], and JSMA [39], and most high effectiveness attacks commonly require a high computation complexity. And we briefly introduce different attack algorithms in the following.

1. CW attack formulates the generation of AEs as an optimization process, which minimizes the L_p norm of the perturbation δ to improve the stealthy of the AEs. It can be formulated as:

$$\min ||\delta||_p + c \cdot max\{max\{f_i(x + \delta) : i \neq t\} - f_t(x + \delta), -\kappa\},$$

   where $|| \cdot ||$ indicates L_p norm which includes L_2, L_∞, and L_0, and it depends on the specific requirements for the AEs. $f_i(x)$ indicates the confident of classifier predicts the input x on i-th label, and t represents the target label. κ is the parameter to ensure the high confidence of generating an AE: $x + \delta$ to be the target label t. CW is a popular attack algorithm due to its high effectiveness and good stealthiness, but it also has a time-consuming problem compared to other attack methods.
2. FGSM is an effective method to generate AEs. Different from CW attacks which iteratively update the loss function, FGSM perturbed x in the direction of a

Table 1 Performance of different adversarial attacks

Method	Computational complexity	Attack effectiveness
CW [9]	High	High
FGSM [17]	Low	Low
BIM [23]	High	High
JSMA [39]	High	High

gradient that can maximize the loss of the function, where δ is computed via the sign of the gradient, and it can be expressed as:

$$\delta = \epsilon \text{sign}(\nabla_x J(\theta, x, y_t)),$$

where $J(\theta, x, y)$ is the loss function with the model parameter θ, it takes traffic packets x and target label y_t as input, and output the loss of the model, i.e., cross-entropy of the trained neural network. The most benefit of FGSM is its high efficiency, because there is no extensive optimization (e.g., CW), and it can directly generate the final δ via computing the direction of the gradient.

3. BIM, an iterative method based on FGSM, can iteratively compute the sign of the gradient and update the gradient with a small step size, and it can clip the perturbed samples to make sure $x + \delta$ is a valid AE inside the bound.

$$\hat{x}_0 = x$$
$$\hat{x}_{n+1} = \text{Clip}_{x,\epsilon}\{\hat{x}_n + \alpha \text{sign}(\nabla_x J(\theta, \hat{x}_n, y_t))\},$$

where $\hat{x}$ indicates the perturbed input, $\hat{x} = x + \delta$, and $\hat{x}_n$ is the n-th iteration AE, α controls the step size of updating the perturbation to the previous AE. The BIM attack is known due to the fact that it is more powerful than FGSM in creating adversarial examples, it can well combine with the optimization method, i.e., momentum. Nevertheless, due to its iterative approach, the BIM attack has a computational cost problem. On the other hand, it cannot ensure stealthiness as well as the CW attack.

4. Jacobian-based Saliency Map Attack (JSMA) is a specific attack that can optimize the number of perturbed features. Different from other attacks, JSMA aims to find the most representative features, which have the most impact on the classification performance, and then maximize the perturbation of these important features and avoid changing other features as less as possible. Specifically, the mainly requirements of JSMA is maximizing $f_t(x)$ and minimizing $f_i(x), \forall i \neq t$, and the saliency map $S(x, t)$ can be expressed as:

$$S(x,t)[i] = \begin{cases} 0, \text{ if } \dfrac{\partial f_t(x)}{\partial x_i} < 0 \text{ or } \displaystyle\sum_{j \neq t} \frac{\partial f_i(x)}{\partial x_i} > 0 \\ \dfrac{\partial f_t(x)}{\partial x_i} \left| \displaystyle\sum_{j \neq t} \frac{\partial f_i(x)}{\partial x_i} > 0 \right|, \text{otherwise,} \end{cases}$$

where i indicates the input feature. In the first stage, JSMA computes the partial derivatives of the matrix of the target class with respect to each feature i. In the second stage, JSMA identifies the most salient features in the second stage. In the third stage, it manipulates the salient feature according to their impact on the target class. At last, we use JSMA by repeating the second and the third stage, until achieving the objective, i.e., $\hat{x}$ classified as y_t. Compared to CW, the JSMA

attack has shown its effectiveness against different machine learning models, but it still lacks the consideration of making the AEs more stealthy.

Overall, we conclude 4 commonly used white-box attacks [56] algorithm, which is high effectiveness to launch an attack on the target model. However, it is still unclear whether these attacks can have good transferability without obtaining any knowledge of the target models, i.e., gradient knowledge.

2.3 Existing Attacks on IDS

2.3.1 White-Box Attacks

As shown in Table 2, most white-box attacks [2, 43, 48] assume have access to the target model, and then leveraging the gradient information to generate AEs, i.e., employing FGSM [43] to create AEs. The most benefit provided by the white-box attack is high effectiveness and efficiency [56], i.e., nearly 90% attack success rate with 500 iterations. However, there are some back draws of these knowledge-based attacks. (i) Limited real-world applicability: existing white-box attacks [2, 43, 48] commonly need to directly compute the gradient of the target model to generate AEs. However, from practical consideration, attackers cannot access the target models or even do not know the architecture of the IDS, so it is unrealistic to launch a white-box in the real-world scenario. (ii) Lack of generalizability, most white-box-based AEs are most likely specific to the target model. White-box attacks commonly do not focus on creating these AEs towards other different models, even if they have

Table 2 Summary of existing adversarial attacks on IDS

Method	Threat scenario	Knowledge	Description
[2]	White-box	Gradients	Exploring AEs on the botnet traffic classification.
[43]	White-box	Gradients	Evading detection via manipulating command and control of AEs.
[48]	White-box	Gradients	Using active learning and GAN to create AEs to attack IDS.
[28]	Black-box	Outputs	Utilizing a GAN to convert benign network traffic into adversarial instances.
[59]	Black-box	Outputs	Generating AEs of traffic flows via deep reinforcement learning.
[56]	Black-box	Outputs	Querying with target model to generate Black-box based AEs.
[22]	Gray-box	Partial knowledge	Using knowledge of the features and architecture to generate AEs
[18]	Gray-box	Partial knowledge	Creating universal adversarial perturbations with limited knowledge
[24]	Gray-box	Partial knowledge	Generate AEs with linear physical constraints

the same objective, i.e., monitoring and classifying the traffic packets, and most AEs [32] can not have a high transferability to other models. (iii) Difficult to survive with the defense, there are some defense mechanisms, i.e., adversarial training [3, 8, 17, 31, 46, 53, 57] [56] that can be employed to detect or mitigate adversarial examples, there is also detection mechanism to filter the malicious packets [56].

2.3.2 Black-Box Attacks

On the other hand, existing black-box attacks [28, 56, 59] assume obtaining the outputs of the target model, i.e., hard label results, malicious or benign. Although Black-box attacks [28, 59] have more substantial real-world applicability than white-box attacks, they also show some cons and problems we need to consider. (i) Heavily rely on the querying: black-box attacks[60, 62] have to keep querying with the black-box model to obtain the output information[12, 28, 56, 59]. However, the query will cost more time or even financial cost[30]. (ii) Lower attack effectiveness due to limited knowledge, different from the white-box attacks, the model's architecture, parameters, and training dataset are unknown to the attackers, which could make it more challenging to craft effective AEs. (iii) Less transferability: since most black-box attacks aim to prob with a target model, which leads to a specific AE towards to this model, and this kind of AEs lack of transferability to other models.

2.3.3 Gray-Box Attacks

Gray-box attacks assume that attackers can get partial knowledge [18, 22, 24] about the targeted system, i.e., the system's features and architecture. In gray-box attacks, the limitations of Knowledge and challenges in transferability are inherent. Compared to black-box attacks, gray-box attacks have practical constraints: Gray-box attacks commonly need to know a combination of knowledge gathering and system querying to gather information about the targeted system. Concluding from white-box, black-box, and gray-box attacks have limitations to the transferability, so we propose to investigate the transferability attack on the IDS.

2.4 Threat Model

Attack Goal In this chapter, we mainly focus on studying the transferability of the AEs to the IDS. Thus, attackers should have no knowledge of the black-box model, neither the model architecture nor the training dataset and parameters.

Attack Knowledge Different from existing black-box attacks, there will be no outputs to guide us to generate AEs. As shown in Fig. 2, the attacker only has access to add perturbation to the traffic packets, and attackers have no interaction with the target model.

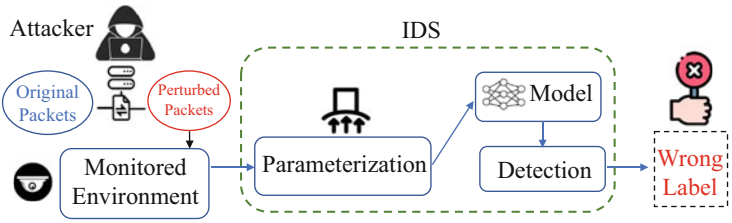


Fig. 2 Adversarial attack on IDS

Attack Capability The attackers propose to revise the input packets class to a target label, i.e., changing the traffic packets label from benign behavior o malicious behavior. And we assume attackers can their own surrogate models and employ the various attack algorithms to generate AEs.

3 Building Surrogate Model of IDS

In this section, we aim to investigate how well different training factors affect the building of surrogate models. We first introduce and analyze the dataset of IDS models, and then analyze how to train the models with different datasets. And we also present different machine-learning models that can be used to train the IDS. Last, we find the most appropriate training factor to train the model for further transfer attacks.

3.1 Datasets

NSL-KDD [52] is a widely used dataset set that offers potential solutions to several inherent challenges found in the previous datasets, and it is a valuable benchmark for researchers to evaluate and compare various intrusion detection methods effectively. There are several advantages to this dataset:

1. Avoid bias: all the records in the train set of the NSL-KDD dataset is well-crafted, which provides the benefit that models are not feasible to be trained according to more frequently occurring records. The same to the proposed test sets, which also avoid any duplicate records, preventing the model from predicting bias towards common records.
2. Diversity of difficulty levels: ensuring a more comprehensive evaluation of multiple learning methods, and each dataset is composed of equal distribution of each difficulty level group. This method provides a large range of classification rates for different machine learning methods, enhancing the efficiency of evaluating and comparing these techniques.

3. Reasonable data distribution: the amount of the record sets is well-distributed, which opens a door for the model training that no need to randomly select a small subset. This consistency in evaluation will bring more benefits to researchers when they have to compare different existing works.

3.2 Building IDS via Various Machine Learning Models

There are multiple machine learning methods have been extensively employed in IDS based on relevant research findings. In order to widely evaluate the effectiveness and generalizability of our proposed model, we constructed seven algorithm-based black-box IDS models, and here are some details of these widely used baseline models.

1. Support Vector Machine (SVM) [38], known as one of the most powerful machine learning algorithms, which can be utilized in IDS [20, 35, 50, 56]. SVM is en-powered to detect anomalies and classify traffic packets based on their specific characteristic to build optimized decision boundaries. SVM can effectively distinguish between normal and malicious network behavior due to leveraging the principles of margin maximization and kernel functions.
2. Naive Bayes (NB) [44] is commonly employed in the classification of network traffic [19, 34, 51], because they can distinguish the normal and malicious instances via utilizing the observed features, i.e., packet headers [11], behavior patterns [4], and payload characteristics [55]. NB can effectively predict the presence of intrusions via computing the conditional probability of a network instance belonging to a specific class (normal or malicious).
3. Multilayer Perceptrons (MLP) [5] can well study the learning patterns and correlations between input features, thereby, having a good performance on classification [1, 13, 15]. The network is trained on the adjusted weights labeled data, which can lead to minimized prediction errors. After training, the MLP can make a classification output that whether traffic packets are normal or malicious based on the learned patterns.
4. Logistic Regression (LR) [58] utilizes the logistic function to model the correlation between input features and the probability of an instance being classified into a particular class (normal or malicious) [6, 47, 54]. During training [16], LR estimates the parameters of the logistic function by minimizing a cost function using labeled training data. This iterative process ensures that the LR model optimally fits the training data, enabling accurate predictions of class probabilities for new instances.
5. Decision Tree (DT) [37] is an extensively employed algorithm for classification purposes [21, 36, 49]. DT constructs a tree-like model that represents a sequence of decisions and their potential outcomes, relying on the features observed in network instances, which can promote identifying the most influential features for intrusion detection.

6. Random Forest (RF) [7] leverages the collective knowledge of the ensemble to make predictions about the class labels of traffic packets [14, 42, 61]. The specific process, i.e., voting or averaging, can guide the model to make a final prediction, which is obtained via overall the prediction of different individual decision trees.
7. KNearest Neighbors (KNN) [41] classifies a network traffic packet via finding the nearest neighbors in the feature space [25, 27, 45]. The "k" in KNN indicates how many nearest neighbors can be considered. KNN commonly is computationally expensive, because of the high cost when calculating distances between instances.

3.3 Training Surrogate Models

Since the training dataset is also an important factor [32] in terms of transferability, and this aspect has been studied in the image domain [29]. Focus on the network traffic domain, we aim to build an IDS with a good classification performance, thereby, improving the transferability. One direct way to train the IDS with different settings is to use different training and testing data. Specifically, NSL-KDD has different categories of intrusion, and we focus on investigating the DoS attacks [56], and there are a total of 148,517 records. We set 3 different training and testing distributions, and denote each one as Training Factor 1, Training Factor 2, and Training Factor 3. For Training Factor 1, 75% training and 25% testing; Training Factor 2: 80% training and 20% testing; and Training Factor 3: 85% training and 15% testing. We will test 7 different machine learning models with different cases, i.e., SVM, NB, MLP, RF, DT, KNN, and LR.

3.4 Evaluation Metrics

We first present some basic metrics that are widely used in the machine learning domain. The metrics derived from the count of true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN) are collectively known as confusion matrix-based metrics.

- True Positives (TP) indicates IDS correctly predicts the malicious traffic packets as the malicious label.
- False Positives (FP) refers to the IDS wrongly predicting benign traffic packets as malicious labels.
- True Negatives (TN) indicates the count of cases that IDS correctly classify the benign traffic packets as benign labels.
- False Negatives (FN) can be considered as instances where the model wrongly predicts malicious traffic packets as benign traffic packets.

Based on those basic metrics, we find some commonly used metrics that can measure the prediction results of the machine learning models: Recall, Precision, F1-Score, and False Positive Rate (FPR).

1. Recall, which is similar to the True Positive Rate (TPR), measures the proportion of correctly predicted malicious traffic packets out of the total actual malicious traffic packets, Recall= $TP/(TP + FN)$;
2. Precision measures the percentage of traffic packets predicted as malicious labels that are actually malicious samples. Precision = $TP/(TP + FP)$;
3. F1-Score aims to measure the model's performance which is a balanced evaluation between the recall and precision, defined as: F1-Score= $2(\text{Precision} \cdot \text{Recall})/(\text{Precision} + \text{Recall})$.
4. False Positive Rate (FPR) [10] can measure the proportion of wrongly predicted malicious instances out of the total actual benign samples. $FPR = FP/(FP+TN)$.

3.5 Model Performance Analysis

Impact of Training Factor on Recall As shown in Fig. 3, MLP achieves the best Recall (0.94) with Training Factor 3. A higher recall indicates that the machine learning model is effectively predicting a large proportion of the malicious traffic packets in the NSL-KDD. We can also see that the SVM and KNN have a close performance to the MLP, where SVM and KNN can achieve 0.93 and 0.92 Recall, respectively. We also find that all the models can achieve a better Recall with training factor 3, even for the worst one (e.g., DT).

Impact of Precision A higher precision indicates that the classifier can accurately identify a large percentage of the predicted malicious traffic packets as true malicious samples. As shown in Fig. 4, most models can achieve a Precision above 0.75 with the Training Factor 2 and 3. We observe that the training factor can also affect the performance, and Training Factor 1 appears to have a lower Precision, i.e., 0.72 in DT. And Training Factor 3 still achieves the best performance.

Fig. 3 Recall of different machine learning models

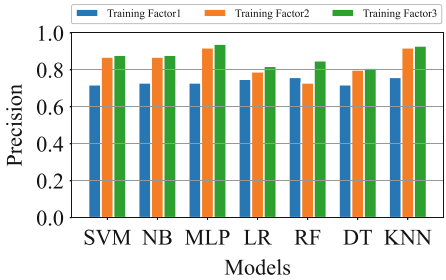


Fig. 4 Precision of various machine learning models

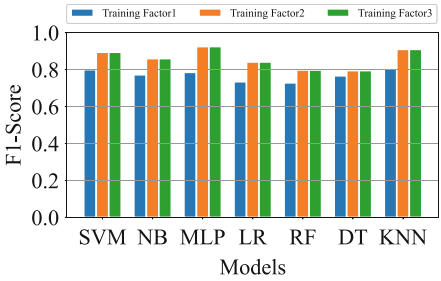


Fig. 5 F1-Score of different machine learning models

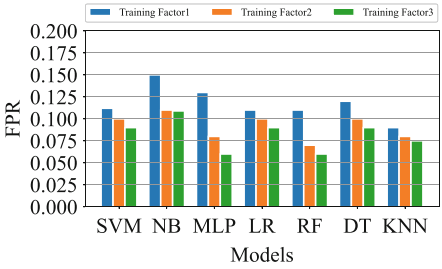


Fig. 6 FPR of various machine learning models

Impact of F1-Score A good F1-score represents a balanced performance between precision and recall, which shows the classifier's capability to achieve both high Recall and Precision at the same time. As shown in Fig. 5, MLP still achieves a balanced performance of both Recall and Precision. Considering the training factor to the classifier, Training Factor 2 has a close performance to Training Factor 3, and these two training factors are effective settings for the classifier to achieve high performance.

Impact of FPR Different from the previous metrics, a low FPR can reflect that the IDS has a minimal probability of misjudging the benign traffic packets as malicious. As shown in Fig. 6, NB has the worst classification performance with the highest FPR in Training Factor 1. We can find the minimal FPR existed in the MLP (Training Factor 3), which has consistent results with the previous experiments.

Indeed, the training factor can influence the classification performance of different models, and we propose to choose Training Factor 3 as our default training setting because higher prediction results appear to have a high good transferability than the lower performance models. On the other hand, MLP seems to be the most

effective model to classify malicious and benign traffic packets. And we propose to investigate different attack settings based on this most effective model.

4 Investigating the Transferability of AEs in IDS

In this section, we aim to explore how the attack factor affects the transferability of AEs. We first implement different white-box attacks on various machine-learning-based IDS, and find the benchmark for the transfer attacks. And then we propose to generate the effective AEs with well-trained surrogate models and the most effective adversarial attack algorithms. Last, we present two key observations between the transfer attacks an white-box attacks.

4.1 Different AEs Generation on White-Box Attacks

We first aim to find the benchmark of different white-box attacks on IDS. For example, we want to know the best performance of the white-box attacks, which is the upper bound of the transfer attack, and then understand how the effectiveness of the transfer AEs on IDS.

To this end, we compare different white-box attacks [56] on machine-learning-based IDS, including CW, FGSM, and BIM. And we use attack success rate (ASR) as the evaluation metric, which indicates the percentage of the AEs successfully spoofing the IDS to predict the target wrong level, i.e., the original traffic packets label as benign, and IDS classifies these perturbed packets (AE) as malicious behaviors and vice versa.

As shown in Fig. 7, there are four white-box attacks with various perturbation norms (L_2 norm) ranging from 0.05 to 0.50. We can observe that the ASR of all these attacks is directly proportional to the perturbation norms of AEs. For example, when we set the L_2 norm as 0.05, the ASR of the FGSM is only 0.1732. But it dramatically increased when the norm ranges from 0.10 to 0.25, i.e., its ASR can achieve 0.6390 at 0.25 L_2 norm. On the other hand, we can see that the CW attacks can always achieve higher ASR than the JSMA, FGSM and BIM, e.g., the ASR of CW is 0.9600, which is much higher than that of BIM (0.7448) and FGSM (0.7002). As we discussed in Sect. 2.2, CW attacks commonly involve large computations to optimally generate AEs that have high attack effectiveness to the white-box model. This finding is also consistent with the attacks in the image domain [32]. However, the higher white-box does not indicate a higher transferability, which has been studied in the image domain [29, 32]. Therefore, we propose to investigate how well these white-box attacks impact the transfer attacks, e.g., whether a higher effectiveness white-box attack algorithm will also lead to better transfer AEs to the black-box models.

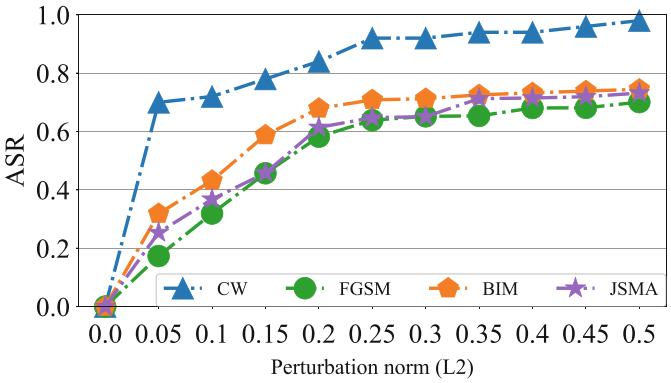


Fig. 7 Different adversarial attacks on IDS with various perturbation norms

4.2 Investigating on AE Transferability

4.2.1 Surrogate Model Settings

We consider multiple IDS models as we introduced in Sect. 3.2. For example, we build MLP-based IDS with a hidden layer with 50 neurons [56]. For remaining models, we train the IDS with LR, KNN, NB, DT, SVM, and RF from scikit-learn library [40]. These different models covered a wide range machine learning methods, which ensure the diversity of IDS models.

4.2.2 Surrogate Dataset Settings

Considering the Surrogate Dataset, we choose NSL-KDD [52], which is a popularly used dataset for most IDS attacks and defense. As we studied in Sect. 3.5, the training factor can affect the model performance, and we choose Training Factor 3 to train our different surrogate models. For the black-box models, we randomly employ Training Factors 1 and 2 to build black-box models, which contribute to training the model with different classification performances.

4.2.3 Adversarial Algorithm Settings

We consider the commonly used gradient-based adversarial algorithm, CW, FGSM, JSMA, and BIM. And we use L_2 norm as the constraint for the perturbation. As we explored different white-box attacks in Sect. 4.1, the perturbation norm can indeed affect the attack effectiveness, but it is still unclear whether the transfer attack has similar results to the previous findings. Therefore, we also test the transferability with different perturbation norms.

4.2.4 Evaluation Metrics

There are commonly two metrics to evaluate the transferability of AEs, i.e., match rate [29] and misclassification rate [32]. The match rate quantifies the percentage of AEs that can cause both a surrogate model (utilized by the attacker to generate AEs) and a black-box model (the target model for the attacker to deceive) to predict the same incorrect label. For example, the AEs (the original label is benign) spoof the surrogate model predicted as malicious behavior that can also mislead the black-box models predict as malicious. As there are only two labels for the traffic packets, malicious and benign, the misclassification rate is the same as the match rate in the IDS scenario, and we propose using the match rate to evaluate the transferability of AEs.

4.2.5 Experiment Setting

We implement the experiments on the server equipped with one NVIDIA GeForce RTX 4090 which offers 24 GB graphics memory with a 384-bit memory bus and 28 Intel i9-9940X CPU with 3.30GHz. And there are a total of 2 Terabyte SSD. And all the code is implemented in TensorFlow 2.

4.3 Evaluation of AEs Transferability: Results and Discussion

4.3.1 The Affect of Different Attack Algorithms on Transferability

We aim to explore the transferability of different attack algorithms including FGSM, BIM, CW and JASM. And we set the fixed L_2 perturbation norm as 0.05. Specifically, we consider the AEs transferability in the cross-architecture scenario, i.e., test the match rate between different surrogate models and target models. The classification performance of different models is shown in Table 3. We can see that the accuracy of all the models ranges from 0.82 (DT) to 0.92 (KNN), which is not very high but still acceptable. Most models' Precision is good which is over 0.80, but DT can only achieve 0.79. On the other hand, the Recall is pretty high compared to other metrics, where the minimal one is 0.97 (MLP and LR) and the maximum is 0.99. Then, we evaluate the transferability of AEs among different models.

Table 3 Performance evaluation of various models

Models	SVM	NB	MLP	LR	DT	KNN
Accuracy	0.90	0.87	0.90	0.89	0.82	0.92
Precision	0.85	0.83	0.89	0.88	0.79	0.97
Recall	0.99	0.99	0.97	0.97	0.98	0.99
F1-Score	0.92	0.90	0.93	0.92	0.88	0.98

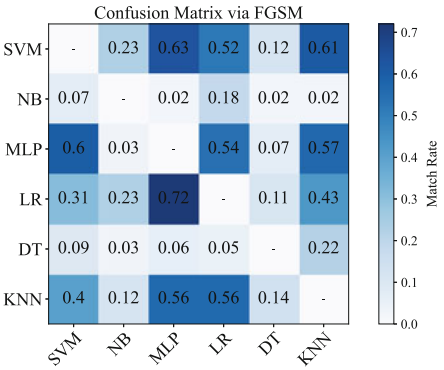


Fig. 8 Match rate of FGSM-based AEs

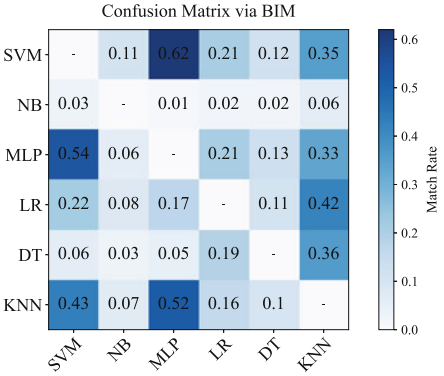


Fig. 9 Match rate of BIM-based AEs

FGSM First, we focus on the FGSM, as shown in Fig. 8, the models in the x-axis represent the surrogate models, and the models in y-axis indicates the black-box models, which is trained with different training factors. We can observe that the AEs generate from MLP have the highest match rate to the LR model. The minimal match rate existed in the NB to DT, which means the AEs created by NB are not effective in the DT model. Overall, the MLP has the highest match rate (0.72) to the other models, and KNN is in second place, which also has a good transferability compared with the remaining models.

BIM Secondly, let's take a look at the BIM, which is an iterative version of FGSM. The results can be found in Fig. 9, we can see that the maximum match rate (0.62) existed in the transfer AEs from MLP to SVM, which is slightly lower than that of FGSM. And the lowest match rate (0.01) is from MLP to NB, and the match rate from NB to MLP is 0.06, which indicates the transferability between these two models is very low. Overall, the AEs generated from SVM, MLP, and KNN have more transferability than other models.

CW Figure 10 shows the match rate of CW attack among different models. We can observe that the highest match rate (0.66) is still between MLP and SVM, which is the same as the BIM attack. Different to the previous findings of FGSM and BIM,

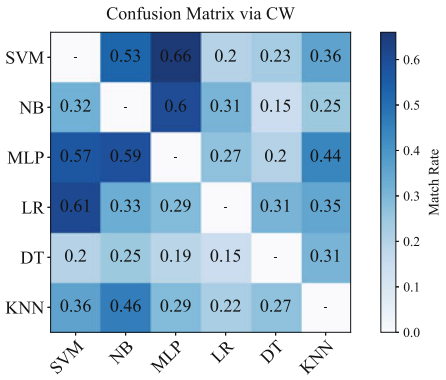


Fig. 10 Match rate of CW-based AEs

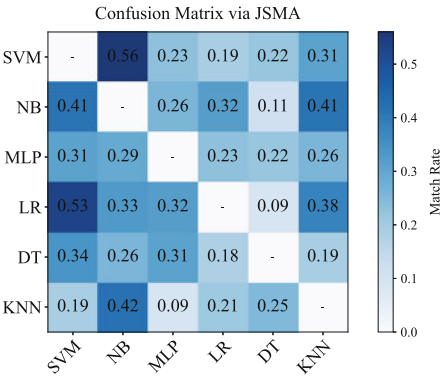


Fig. 11 Match rate of JSMA-based AEs

the transferability of AEs generated from NB model has substantially improved performance. And the lowest match rate (0.15) of CW attack is from DT to NB. In general, CW attacks have higher match rate than the previous attacks.

JSMA As shown in Fig. 11, different from the previous attacks, the match rate of NB models appears to be the highest, i.e., from DT:0.26 to SVM:0.56. And the second-highest match rate is SVM, which ranges from the LR:0.53 to KNN:0.19. Generally, the match rate of JSMA is higher than the FGSM and BIM, but is lower than the CW.

Observation 1 We find that the transferability performance among different attack algorithms is CW > JSMA > BIM > FGSM, which is very close to the findings of the evaluation of white-box attacks in Fig. 7. These findings reveal that CW attacks can always achieve a better performance than the other attacks. And considering the model perspective, MLP seems to be the most effective model for CW to generate high transferability AEs. It might be a good strategy for attackers to generate high transferability AEs via CW attacks based on MLP model. But it is still unclear, how will the perturbation norm affect the transferability.

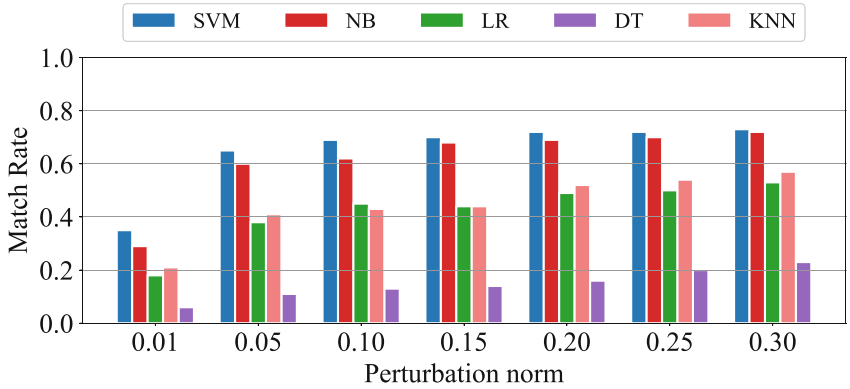


Fig. 12 Different transfer attacks on MLP with various perturbation norms

4.3.2 The Affect of Perturbation Norm on Transferability

Bringing such concerns from the previous findings, we propose to evaluate how well the perturbation norm affects the transferability. To do so, we use an effective MLP model with CW attacks to generate AEs, different from the previous experiments which set a fixed L_2 norm, we decide to change different L_2 norm ranging from 0.01 to 0.30.

Figure 12 shows the match rate varies with different models under various perturbation norms. We can clearly see that the match rate from MLP to SVM and NB achieves the highest at the 0.30 L_2 norm with 0.73 and 0.72, respectively. But these two models' transferability does not increase much when the L_2 norm ranges from 0.05 to 0.30. We can see that the phenomenon happened to the other models as well, i.e., the match rate of LR increased from 0.38 to 0.53.

Observation 2 We find that the perturbation norm can affect the transferability when the norm is in a small range, i.e., from 0.01 to 0.10. And the transferability seems not sensitive with the large perturbation norm, i.e., 0.20 to 0.30. This finding is also similar to the white-box attacks which are shown in Fig. 7. The transferability can be effectively affected in a specific range (e.g., L_2 norm is small), and the improvement of transferability becomes steady when the perturbation norm becomes large, i.e., 0.20 to 0.30.

5 Conclusion

In this chapter, we explore the effectiveness of the transfer attack in the networking traffic domain. We systematically evaluate the transferability of AEs from the training datasets, different architectures models, and various adversarial attack

algorithms. And we find that the transfer attack has common properties with the white-box attacks, but we also reveal the limitation of transfer attacks.

References

1. Amato F, Mazzocca N, Moscato F, Vivenzio E (2017) Multilayer perceptron: an intelligent model for classification and intrusion detection. In: 2017 31st international conference on advanced information networking and applications workshops (WAINA). IEEE, Piscataway, pp 686–691
2. Apruzzese G, Colajanni M, Marchetti M (2019) Evaluating the effectiveness of adversarial attacks against botnet detectors. In: 2019 IEEE 18th international symposium on network computing and applications (NCA), IEEE, Piscataway, pp 1–8
3. Balaji Y, Goldstein T, Hoffman J (2019) Instance adaptive adversarial training: improved accuracy tradeoffs in neural nets. arXiv:191008051
4. Balajinath B, Raghavan S (2001) Intrusion detection through learning behavior model. Comput Commun 24(12):1202–1212
5. Baum EB (1988) On the capabilities of multilayer perceptrons. J Complex 4(3):193–215
6. Besharati E, Naderan M, Namjoo E (2019) LR-HIDS: logistic regression host-based intrusion detection system for cloud environments. J Ambient Intell Human Comput 10:3669–3692
7. Biau G, Scornet E (2016) A random forest guided tour. Test 25:197–227
8. Cai QZ, Du M, Liu C, Song D (2018) Curriculum adversarial training. In: Proceedings of 2018 international joint conference on artificial intelligence (IJCAI)
9. Carlini N, Wagner D (2017) Towards evaluating the robustness of neural networks. In: 2017 IEEE Symposium on security and privacy (S&P), IEEE, Piscataway, pp 39–57
10. Chen G, Chen S, Fan L, Du X, Zhao Z, Song F, Liu Y (2021) Who is real bob? Adversarial attacks on speaker recognition systems. In: Proceedings of IEEE symposium on security and privacy (S&P)
11. Davis JJ, Clark AJ (2011) Data preprocessing for anomaly based network intrusion detection: a review. Comput Secur 30(6–7):353–375
12. Duan R, Qu Z, Zhao S, Ding L, Liu Y, Lu Z (2022) Perception-aware attack: creating adversarial music via reverse-engineering human perception. In: Proceedings of the 2022 ACM SIGSAC conference on computer and communications security (CCS), pp 905–919
13. Esmaily J, Moradinezhad R, Ghasemi J (2015) Intrusion detection system based on multi-layer perceptron neural networks and decision tree. In: 2015 7th conference on information and knowledge technology (IKT), IEEE, Piscataway, pp 1–5
14. Farnaaz N, Jabbar M (2016) Random forest modeling for network intrusion detection system. Proc Comput Sci 89:213–217
15. Ghanem WAH, Jantan A, Ghaleb SAA, Nasser AB (2020) An efficient intrusion detection model based on hybridization of artificial bee colony and dragonfly algorithms for training multilayer perceptrons. IEEE Access 8:130452–130475
16. Ghosh P, Mitra R (2015) Proposed ga-bfss and logistic regression based intrusion detection system. In: Proceedings of the 2015 third international conference on computer, communication, control and information technology (C3IT), IEEE, Piscataway, pp 1–6
17. Goodfellow IJ, Shlens J, Szegedy C (2014) Explaining and harnessing adversarial examples. arXiv:14126572
18. Hou R, Xiang X, Zhang Q, Liu J, Huang T (2021) Universal adversarial perturbations of malware. In: 2022 12th cyberspace safety and security (CSS). Springer, Berlin, pp 9–19
19. Hussein SM, Ali FHM, Kasiran Z (2012) Evaluation effectiveness of hybrid ids using snort with naive bayes to detect attacks. In: 2012 second international conference on digital information and communication technology and it's applications (DICTAP). IEEE, Piscataway, pp 256–260

20. Jha J, Ragha L (2013) Intrusion detection system using support vector machine. Int J Appl Inf Syst 3:25–30
21. Kruegel C, Toth T (2003) Using decision trees to improve signature-based intrusion detection. In: Proceedings of 2003 6th international symposium on research in attacks, intrusions and defenses (RAID). Springer, Berlin, pp 173–191
22. Kuppa A, Grzonkowski S, Asghar MR, Le-Khac NA (2019) Black box attacks on deep anomaly detectors. In: Proceedings of the 14th international conference on availability, reliability and security, pp 1–10
23. Kurakin A, Goodfellow IJ, Bengio S (2018) Adversarial examples in the physical world. In: Artificial intelligence safety and security. Chapman and Hall/CRC, Boca Raton, pp 99–112
24. Li J, Yang Y, Sun JS, Tomsovic K, Qi H (2021) Conaml: constrained adversarial machine learning for cyber-physical systems. In: Proceedings of the 2021 ACM SIGSAC Asia conference on computer and communications security, pp 52–66
25. Liao Y, Vemuri VR (2002) Use of k-nearest neighbor classifier for intrusion detection. Comput Secur 21(5):439–448
26. Liao HJ, Lin CHR, Lin YC, Tung KY (2013) Intrusion detection system: a comprehensive review. J Netw Comput Appl 36(1):16–24
27. Lin WC, Ke SW, Tsai CF (2015) CANN: an intrusion detection system based on combining cluster centers and nearest neighbors. Knowl Based Syst 78:13–21
28. Lin Z, Shi Y, Xue Z (2022) IDSGAN: generative adversarial networks for attack generation against intrusion detection. In: 2022 26th Pacific-Asia knowledge discovery and data mining conference (PAKDD). Springer, Berlin, pp 79–91
29. Liu Y, Chen X, Liu C, Song D (2016) Delving into transferable adversarial examples and black-box attacks. arXiv:161102770
30. Liu H, Yu Z, Zha M, Wang X, Yeoh W, Vorobeychik Y, Zhang N (2022) When evil calls: targeted adversarial voice over ip network. In: Proceedings of the 2022 ACM SIGSAC conference on computer and communications security (CCS), pp 2009–2023
31. Madry A, Makelov A, Schmidt L, Tsipras D, Vladu A (2017) Towards deep learning models resistant to adversarial attacks. In: Proceedings of 2017 international conference on machine learning (ICML) work shop
32. Mao Y, Fu C, Wang S, Ji S, Zhang X, Liu Z, Zhou J, Liu AX, Beyah R, Wang T (2022) Transfer attacks revisited: a large-scale empirical study in real computer vision settings. arXiv:220404063
33. McHugh J (2000) Testing intrusion detection systems: a critique of the 1998 and 1999 darpa intrusion detection system evaluations as performed by lincoln laboratory. ACM SIGSAC Trans Inf Syst Secur 3(4):262–294
34. Muda Z, Yassin W, Sulaiman M, Udzir N (2011) Intrusion detection based on k-means clustering and naïve bayes classification. In: 2011 7th international conference on information technology in Asia. IEEE, Piscataway, pp 1–6
35. Mukkamala S, Janoski G, Sung A (2002) Intrusion detection using neural networks and support vector machines. In: Proceedings of the 2002 international joint conference on neural networks. IJCNN'02 (Cat. No. 02CH37290), vol 2. IEEE, Piscataway, pp 1702–1707
36. Mulay SA, Devale P, Garje G (2010) Intrusion detection system using support vector machine and decision tree. Int J Comput Appl 3(3):40–43
37. Myles AJ, Feudale RN, Liu Y, Woody NA, Brown SD (2004) An introduction to decision tree modeling. J Chemometr A J Chemometr Soc 18(6):275–285
38. Noble WS (2006) What is a support vector machine? Nat Biotechnol 24(12):1565–1567
39. Papernot N, McDaniel P, Jha S, Fredrikson M, Celik ZB, Swami A (2016) The limitations of deep learning in adversarial settings. In: 2016 IEEE European symposium on security and privacy (EuroS&P). IEEE, Piscataway, pp 372–387
40. Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V, et al. (2011) Scikit-learn: machine learning in python. J Mach Learn Res 12:2825–2830
41. Peterson LE (2009) K-nearest neighbor. Scholarpedia 4(2):1883

42. Resende PAA, Drummond AC (2018) A survey of random forest based methods for intrusion detection systems. ACM SIGSAC Computing Surveys 51(3):1–36
43. Rigaki M, Garcia S (2018) Bringing a gan to a knife-fight: adapting malware communication to avoid detection. In: 2018 IEEE security and privacy workshops (SPW). IEEE, Piscataway, pp 70–75
44. Rish I, et al. (2001) An empirical study of the naive bayes classifier. In: 2001 international joint conference on artificial intelligence (IJCAI) workshop, vol 3, pp 41–46
45. Saleh AI, Talaat FM, Labib LM (2019) A hybrid intrusion detection system (hids) based on prioritized k-nearest neighbors and optimized SVM classifiers. Artif Intell Rev 51:403–443
46. Shafahi A, Najibi M, Ghiasi A, Xu Z, Dickerson J, Studer C, Davis LS, Taylor G, Goldstein T (2019) Adversarial training for free! In Proceedings of 2019 neural information processing systems (NIPS)
47. Shah RA, Qian Y, Kumar D, Ali M, Alvi MB (2017) Network intrusion detection through discriminative feature selection by using sparse logistic regression. Fut Internet 9(4):81
48. Shu D, Leslie NO, Kamhoua CA, Tucker CS (2020) Generative adversarial attacks against intrusion detection systems using active learning. In: Proceedings of the 2nd ACM SIGSAC workshop on wireless security and machine learning, pp 1–6
49. Sindhu SSS, Geetha S, Kannan A (2012) Decision tree based light weight intrusion detection using a wrapper approach. Exp Syst Appl 39(1):129–141
50. Sung AH, Mukkamala S (2003) Identifying important features for intrusion detection using support vector machines and neural networks. In: 2003 symposium on applications and the internet. IEEE, Piscataway, pp 209–216
51. Tabash M, Abd Allah M, Tawfik B (2020) Intrusion detection model using naive bayes and deep learning technique. Int Arab J Inf Technol 17(2):215–224
52. Tavallaee M, Bagheri E, Lu W, Ghorbani AA (2009) A detailed analysis of the KDD cup 99 data set. In: 2009 IEEE symposium on computational intelligence for security and defense applications. IEEE, Piscataway, pp 1–6
53. Tramèr F, Kurakin A, Papernot N, Goodfellow I, Boneh D, McDaniel P (2018) Ensemble adversarial training: attacks and defenses. In Proceedings of 2018 international conference on learning representations (ICLR)
54. Wang Y (2005) A multinomial logistic regression modeling approach for anomaly intrusion detection. Comput Secur 24(8):662–674
55. Wang K, Stolfo SJ (2004) Anomalous payload-based network intrusion detection. In: Proceedings of 2004 7th international symposium on research in attacks, intrusions and defenses (RAID). Springer, Berlin, pp 203–222
56. Wang N, Chen Y, Xiao Y, Hu Y, Lou W, Hou YT (2022) Manda: on adversarial example detection for network intrusion detection system. IEEE Trans Depend Secur Comput 20(2):1139–1153
57. Wong E, Rice L, Kolter JZ (2020) Fast is better than free: revisiting adversarial training. arXiv:200103994
58. Wright RE (1995) Logistic regression. American Psychological Association, Washington
59. Wu D, Fang B, Wang J, Liu Q, Cui X (2019) Evading machine learning botnet detection models via deep reinforcement learning. In: 2019 53rd IEEE international conference on communications (ICC). IEEE, Piscataway, pp 1–6
60. Yu Z, Chang Y, Zhang N, Xiao C (2023) SMACK: semantically meaningful adversarial audio attack. In: 32nd USENIX security symposium (USENIX security 23), pp 3799–3816
61. Zhang J, Zulkernine M, Haque A (2008) Random-forests-based network intrusion detection systems. IEEE Trans Syst Man Cybern C (Appl Rev) 38(5):649–659
62. Zheng B, Jiang P, Wang Q, Li Q, Shen C, Wang C, Ge Y, Teng Q, Zhang S (2021) Black-box adversarial attacks on commercial speech platforms with minimal information. In Proceedings of 2021 ACM SIGSAC conference on computer and communications security (CCS)

Advanced ML/DL-Based Intrusion Detection Systems for Software-Defined Networks

Nadia Niknami and Jie Wu

1 Introduction

In traditional network architectures, the control plane and data plane functions are implemented on routers and switches. This results in the independent configuration of traffic policies and challenges when deploying new protocols, requiring updates or replacements across all devices. Furthermore, managing device configurations using device-level tools is time-consuming and error-prone. To address these issues, Software-defined networks (SDNs) has emerged as a solution [32]. SDNs differ from conventional networks primarily in one aspect: the presence of a network controller. Figure 1 illustrates the SDN architecture including the data plane (consisting primarily of transmission devices), the application plane at the top (housing various SDN applications), and the control plane acting as a communication bridge between the application layer and the data plane. SDN separates the control plane from the data plane, centralizing network intelligence in a programmable entity called the "Controller", which manages multiple elements of the data plane through APIs. This centralized approach provides the SDN controller with a holistic view of the entire infrastructure, leading to significant cost reductions compared to conventional networks. Monitoring and measuring network traffic flows are essential for ensuring data integrity within SDN and facilitating traffic control by the SDN controller. However, despite the numerous benefits offered by SDN, it is vulnerable to security threats that malicious actors can exploit for various malicious activities. Security breaches can have severe consequences, including the loss of sensitive information and disruption of network services. As SDNs expand in size and functionality, the likelihood of vulnerabilities and bugs also increases, providing potential entry

N. Niknami (✉) · J. Wu
Department of Computer and Information Sciences, Temple University, Philadelphia, PA, USA
e-mail: nadia.niknami@temple.edu; jiewu@temple.edu

Y. Chen et al. (eds.), *Network Security Empowered by Artificial Intelligence*,
Advances in Information Security 107, https://doi.org/10.1007/978-3-031-53510-9_5

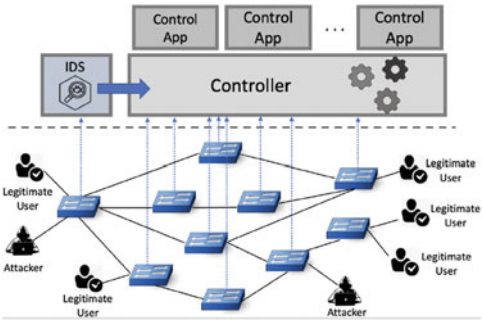


Fig. 1 Anomaly detection in SDN

points for attackers to exploit and compromise network security. Network intrusions can originate from various sources of threats [20]. During attempts to breach a system, attackers may employ various strategies. One common approach in network intrusions involves flooding or overloading the network, with the intention of obtaining network-related data that can be exploited later to target vulnerabilities and weak spots within the system. This method aims to overwhelm the network's resources and disrupt its normal functioning, creating opportunities for further exploitation by the attacker.

One of the most significant aspects of traffic monitoring is the detection of anomalies [3]. Efficiently monitoring the network traffic flow and analyzing all the network traffic can help solve such security issues. As the controller monitors the network traffic, anomalies can be detected and the traffic can be managed accordingly. The collection of accurate network traffic statistics and detecting intrusions or anomalies is crucial to improving network management. An attack on the controller can be detected by looking for anomalies in incoming packets [14]. Network Intrusion Detection Systems (IDSs) are essential tools, available in both software-based and hardware-based forms, used for monitoring network traffic and analyzing it for signs of potential attacks or suspicious activities. The primary role of an IDS is to examine network traffic, identify unwanted or suspicious activity or patterns, and promptly alert the network administrator [8]. Typically, IDSs employ one or more network traffic sensors to monitor network activity across different network segments. These systems continuously analyze and monitor patterns of traffic within the monitored network environment. If the observed traffic patterns match predefined signatures or policies in the knowledge base, a security alert is generated to notify the administrator. IDSs utilize various methods for detecting intrusions. These methods can be categorized into two main types: *Signature-based detection* and *Anomaly-based detection*. Signature-based IDS relies on a database of known attacks to identify malicious traffic. These IDSs use the database, which is regularly updated with the latest threats, to recognize known attacks. While effective at detecting known attacks, signature-based detection has limitations, such as the inability to identify zero-day attacks that are not yet included in the database [27]. The opular network-based IDS tool used for traffic analysis is Snort [24]. Snort is widely used and supports both IDS and intrusion prevention system (IPS)

modes. In IDS mode, Snort generates alerts based on detections, while in IPS mode, it blocks malicious packets. Snort also logs detected attacks and provides attack statistics on the console. Anomaly-based IDS employs machine learning and statistical techniques to classify network traffic as either “normal" or “anomalous". The main objective of anomaly-based IDS is to create a statistical model that defines normal traffic patterns. These IDSs offer the advantage of being capable of detecting zero-day attacks, which are previously unknown attacks. However, they may also generate more false positives when handling legitimate traffic that deviates from normal network activity [11].

The key success factors for IDS include fast anomaly detection, accuracy, and reliability [33]. To address the growing rate and complexity of cyberattacks, researchers have leveraged Machine Learning (ML) and Deep Learning (DL) techniques to develop IDS systems capable of detecting new and zero-day attacks. However, the lack of extensive, realistic, and up-to-date datasets poses challenges to the development of IDS. This chapter is divided into two parts. The first part reviews the general state-of-the-art anomaly-based intrusion detection methodology, while the second part focuses on specific approaches in the SDN test bed.

2 Machine Learning Based Intrusion Detection Methods

The primary function of an IDS is to analyze network traffic, identify suspicious activity or patterns, and promptly notify the network administrator. Network intrusion detection can be regarded as a typical classification problem, which usually requires a labeled training dataset for system modeling. Machine learning and data mining techniques play a vital role in categorizing analyzed patterns by establishing explicit or implicit models. Machine learning focuses on developing systems that can autonomously learn from data and uncover hidden patterns without explicit programming. Multiple machine learning approaches have been utilized to tackle the challenges associated with IDS. These approaches typically encompass three primary stages: (1) Preprocessing: The data instances collected from the network environment are organized in a structured format, allowing for direct input into the machine learning algorithm. Additionally, feature extraction and feature selection techniques are applied during this phase. (2) Training: A machine learning algorithm is employed to analyze the patterns within various types of data and construct a corresponding system model. (3) Detection: Once the system model is established, the monitored traffic data is compared to the generated system model to identify potential matches. If the observed pattern aligns with an existing threat, an alarm is triggered.

Numerous studies have been conducted on anomaly detection for SDN using machine learning approaches, encompassing both supervised and unsupervised strategies, across diverse domains [12]. Supervised learning-based classifiers, such as support vector machine (SVM), decision tree, naïve Bayes network, and random forests, have been successfully applied to detect unauthorized access. Further-

more, unsupervised learning algorithms have demonstrated effective performance in addressing network intrusion detection problems. Designing a single machine learning approach that surpasses existing methods is currently challenging due to various factors, including imbalanced training datasets and high computational requirements. Consequently, hybrid machine learning approaches, such as combining clustering with classifiers and hierarchical classifiers, have garnered significant attention in recent years [5, 10].

2.1 Statistical Methods

Statistical techniques in anomaly detection leverage statistical properties to establish the normal profile of transactions. These techniques utilize measures such as mean deviation and others to analyze the data. Unlike some other methods, statistical approaches do not rely on prior knowledge of specific attacks, making them effective in detecting new, previously unseen zero-day attacks. By constructing a probability distribution model, statistical approaches determine the deviation between observed traffic and the expected normal behavior. Objects that exhibit a low probability under the established probability distribution model are identified as outliers, indicating potential anomalies in the data. To summarize feature distributions, *Entropy* can be used as a measure of uncertainty and randomness.

$$E_X = \sum_{i=1}^{n} -p(x_i)log(x_i), \tag{1}$$

where X is the feature that can take values $\{x_1, ..., x_n\}$ and $p(x_i)$ is the probability mass function of the outcome x_i. Entropy is commonly utilized in DDoS detection to assess the randomness of incoming network packets. Higher entropy values indicate greater dispersion in traffic features, while lower values indicate more convergence. By measuring entropy, a decrease in randomness can be identified, such as in DoS attacks where multiple packets are targeted at the same IP address and port. This reduction in entropy can serve as an indicator for detecting such attacks [26].

Depending on the number of existing flows, entropy values can lead to substantial datasets. These values, represented as $E_n(X)$ and $E_a(X)$, indicate the entropy of features in the network's normal and abnormal states, respectively. In normal conditions, the information entropy typically fluctuates within a limited range, experiencing both increases and decreases. However, during a DDoS attack, there is a significant surge in traffic directed towards a specific IP address, resulting in a decreased entropy value. In such cases, the condition $E_n(X) - E_a(X) > \delta$ holds true. The detection of DDoS attacks using entropy relies on the window size and the threshold. The window size, determined based on either a specific time period or the number of packets, measures the uncertainty of incoming packets by calculating their entropy. An attack is identified when the calculated entropy exceeds or falls below a predetermined threshold, which is determined based on the selected scheme. The window size and threshold work together to enable the identification of attack

patterns. The value of δ is determined by the statistical information entropy of the network under normal operating conditions. Measuring conditional entropy enables the assessment of predictability between features and then network anomalies can be detected effectively. It quantifies the remaining uncertainty about the second feature given knowledge of the first feature.

$$E_{(src|dst)} = \sum_j -p(dst_j) \sum_i p(src_i|dst_j) log(p(src_i|dst_j)), \tag{2}$$

$p(dst_j)$ represents the percentage of packets arriving at a certain destination address j, or dst_j, among examined packets. $p(src_i|dst_j)$ is the proportion of packets originating from source address i in the total number of packets that are supposed to arrive at dst_j. All other combinations such as $E_{(src|length)}$ and $E_{(src|dst_P)}$ can also be achieved in the same manner, where $length$ represents the length of the packet and dst_P represents the destination port. The utilization of entropy in traffic analysis offers enhanced detection capability compared to volume-based methods [7]. Furthermore, the entropy method provides valuable information for classifying diverse anomalies. Modeling network behavior requires considering various time intervals. If there are variations in network behavior between intervals, it may indicate an ongoing attack. In addition to measuring uncertainty, it is important to assess the disparity between the assumed and observed traffic distribution on the network. The difference between two probability distributions, A and O, over variables $x_1, x_2, ..., x_n$ can be calculated using the following method:

$$KL_D(O||A) = \sum_{i=1}^{n} -O(x_i) log(O(x_i)/\ A(x_i)). \tag{3}$$

The Kullback-Leibler (KL) divergence, also known as relative entropy, is a statistical measure used to quantify the difference between two probability distributions. It provides a way to assess how far an observed distribution O deviates from a reference or assumed distribution A [2]. KL divergence value of 0 indicates that the observed distribution perfectly matches the reference distribution, while higher values indicate a greater dissimilarity [15]. In the context of anomaly detection, the KL divergence can be utilized to detect the initiation of new attacks as well as identify ongoing attacks [30]. This metric is particularly useful in capturing subtle anomalies, such as stealth port scans, even in the presence of background traffic [30]. By leveraging the KL divergence, traffic analysis approaches, such as Traffic Agent Controllers based on OpenFlow, can effectively monitor SDN-enabled switches and detect anomalies with lightweight statistical metrics [15].

2.2 Classification-Based Methods

Classification is a supervised learning approach in machine learning, where classifiers learn from labeled datasets and make predictions on new data. Classification algorithms are trained to assign data into predefined categories based on the

features they possess. *Support vector machines (SVM)* is a supervised classification algorithm. In the context of network anomaly detection, SVM can be trained using normal network data to create a model. The model is then used to classify new instances of network data as either normal or anomalous. By finding the optimal hyperplane during the training process, SVM can effectively distinguish between normal and anomalous network behavior based on their proximity to the hyperplane. SVM is a popular choice for network anomaly detection due to its ability to handle high-dimensional data and its capacity to generalize well to new and unseen instances. Authors in [16, 29] proposed models to detect DDoS attacks in SDN by using the SVM algorithm to predict whether the traffic is abnormal or not. Evaluation results show a high detection rate and good performance with minimal additional overhead for SVM.

2.3 Hybrid Approach

In machine learning, there is no “one size fits all" algorithm, and combining multiple algorithms is often preferred for generalized applications to enhance accuracy, reduce variance, and prevent overfitting [36]. Hybrid classifiers integrate various machine learning techniques to improve performance. A hybrid machine learning approach combines two techniques, with the first one focusing on parameter tuning to enhance performance in the second phase, which is the classifier itself. Machine learning-based network IDSs have the capability to predict normal network behavior using input data. However, in real-world network environments, these systems encounter challenges when it comes to real-time detection. One of the main limitations is the absence of packet sniffers, which are crucial for capturing network traffic in real-time. The successful integration of packet sniffers with machine learning-based network IDSs has proven effective in achieving real-time detection capabilities.

3 Deep Learning-Based Intrusion Detection Methods

When comparing the outcomes of various machine learning-based IDS, DL-based IDS have exhibited superior performance in the context of SDN. While many machine learning algorithms are trained using supervised methods, which can yield satisfactory results in classification tasks, they may not be as effective in logic modeling scenarios[19, 28]. DL is a specialized branch of Machine Learning that revolves around the utilization of multi-layered artificial neural networks equipped with representation learning. It is anchored in the concept of artificial neural networks (ANNs) [9]. These ANNs are supplied with training algorithms and copious amounts of data to enhance the efficiency of the training process. The more extensive the dataset, the more effective the process becomes. The term “deep"

learning derives from the fact that the neural network progressively encompasses additional layers over time. As the network delves deeper, its performance improves. Deep learning facilitates the algorithm's capacity to grasp various levels of data representation and generalization. This approach has found successful applications in diverse domains such as visual object recognition, object detection, network intrusion detection, and many others. A deep learning algorithm can be trained in either a supervised or unsupervised manner. Essentially, a neural network emulates the structure and functionality of the human brain, comprising three successive layers of artificial neurons: the input layer, hidden layer(s), and output layer.

Deep learning can be achieved through various architectural designs, including Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and Deep Belief Networks (DBNs). *CNNs* may assign significance to various features or objects in data, and then distinguish between them. When compared to other classification methods, the amount of pre-processing needed by a CNN is much less. In [1], a hybrid IDS was developed by combining Convolutional Neural Networks and Long Short-Term Memory Networks (LSTM) [6]. The proposed model effectively captures both spatial and temporal features of network traffic, thereby improving the detection performance of zero-day attacks in Software-Defined Networking environments. The effectiveness of using the CNN-LSTM model, along with the MLP model, for anomaly detection in SDN was also demonstrated by authors in [25], showing a high detection rate. *RNNs* are a type of deep neural network where connections between nodes form a directed graph along a time sequence. Authors in [35] proposed a Deep Learning (DL) approach, called DeepIDS, for network intrusion detection in SDN architecture. Their models were trained and tested using the NSL-KDD dataset, achieving an accuracy of 80.7% and 90% for the Fully Connected Deep Neural Network and the Gated Recurrent Neural Network (GRU-RNN), respectively. Their experiments confirmed the potential of the DL approach for flow-based anomaly detection in SDN environments.

DBNs are a type of deep neural network composed of latent variables (hidden units) that exhibit interactions between layers rather than within units within each layer. Zhao et al. [37] presented a hybrid anomaly detection model based on DBNs and probabilistic neural networks. The DBN was trained without supervision and learned to probabilistically reconstruct the received inputs, effectively functioning as feature detectors. Their proposed model, combined with an algorithm, achieved a false alarm rate of 0.615%, accuracy of 93.25%, and detection rate of 99.14%. In another study by authors in [21], an intrusion detection engine was proposed with a DBN serving as the core component. *Autoencoder* have shown significant improvements in anomaly detection accuracy compared to Principal Component Analysis (PCA). Unlike linear PCA, autoencoders are capable of detecting subtle anomalies that may go unnoticed. Moreover, training autoencoders is straightforward and does not require computationally intensive operations like kernel PCA. Each layer of the autoencoder is implemented as a simple RNN layer. In the study conducted by Elsayed et al. [13], the authors addressed the limitations of traditional feed-forward neural networks by combining an autoencoder with an RNN. This integration resulted in a more powerful model with enhanced

classification accuracy. The proposed model consists of two stages: an unsupervised pre-training stage and a fine-tuning stage. In the first stage, the goal is to extract useful feature representations from the input data through unsupervised learning. By optimizing the weight and bias values, the RNN-autoencoder is capable of learning hierarchical features from unlabeled data. The subsequent stage involves fine-tuning the network's last layer using labeled samples in a supervised manner.

4 Reinforcement Learning (RL) Techniques for IDSs

In the realm of Reinforcement Learning (RL), a typical machine learning problem can be outlined as follows: An agent interacts with an environment, observing its current state and executing actions. In response to the agent's actions, the environment provides a reward, which can be positive or negative. This sequential decision-making problem can be represented as a Markov Decision Process (MDP), comprising a state space, action space, transition probabilities, and a reward function. The agent's objective is to acquire a policy that maximizes the cumulative reward over time. To accomplish this, the RL algorithm is employed by the agent to determine a policy consisting of a set of behaviors aimed at optimizing future rewards. By employing Bellman's expectation equation and Bellman's optimality equation, the MDP's optimal value function and policy can be determined. Dynamic programming is commonly used to solve the Bellman equations, and it has evolved into techniques such as SARSA and Q-learning. Q-learning, a prominent RL algorithm, is highly favored due to its model-free nature, as it can operate without prior knowledge of future rewards or transition probabilities. It also incorporates off-policy methods, allowing it to learn about optimal policies while following behavioral policies. Therefore, Q-learning is well-suited for real-time system operation, considering uncertainties in future information [34]. In a related study [31], the authors propose an RL approach that involves collecting network metrics and grouping them into profiles. Each profile comprises a set of actions that utilize reinforcement learning, Network Function Virtualization (NFV), and an SDN controller to address problems. Policies for handling anomalies are defined based on the rewards associated with each action.

To address the challenges associated with high-dimensional state spaces in Q-learning, the combination of Reinforcement Learning with deep learning has given rise to a technique known as *Deep Reinforcement Learning (DRL)*. The goal of DRL is to learn an optimal policy by leveraging a non-linear function approximator based on Multilayer Perceptrons (MLPs). This approximator captures the probability distribution of action strategies for a DRL agent, aiming to maximize the expected long-term reward [4]. One commonly utilized DRL algorithm is Deep Q-Network (DQN), which effectively tackles the complexities posed by intricate state spaces in Q-learning. In a related study [18], the authors propose a non-intrusive traffic sampling mechanism for multiple traffic analyzers in an SDN-capable network using a deep deterministic policy gradient, which is a representative DRL algorithm

for continuous action control. The proposed system learns the policy for allocating sampling resources while considering the uncertainty in flow distribution based on the sampled traffic inspection outcomes obtained from multiple traffic analyzers.

5 ML-Based Anomaly Detection on a Real SDN

In this section, we present the details of our real test bed for SDN. Then we summarize some specific network intrusion detection methods on the given test bed.

5.1 Entropy-KL IDS: A Statistical Intrusion Detection Method

Relying solely on entropy as a detection measure may not be adequate due to its dependence on the chosen thresholds for attack detection. Similarly, using KL-divergence alone may not be sufficient in scenarios where we need to identify a DoS attack while another attack is already in progress. This limitation arises from the inability of KL-divergence to differentiate between the start and end of different attacks. To overcome these limitations, combining entropy and KL-divergence can significantly enhance the detection of DoS attacks. By considering both measures together, the detection system can effectively capture the distinctive characteristics of such attacks. It is worth noting that packets possess various features, and it is crucial to consider the relevant features and their correlations when developing an effective detection mechanism. In [23], the authors proposed a method that incorporates weights to merge entropy and KL-divergence, addressing the aforementioned issues. Upon receiving incoming traffic, the merging process of entropy and KL-divergence is performed on different features of the packets. The weighted results obtained from the combination of entropy and KL-divergence are then utilized in ensemble learning. The weights assigned to the features can be determined based on their importance or correlation. Figure 2 illustrates the combination approach of entropy and KL-divergence with different features. The components collectively contribute to the final decision regarding the status of network traffic. The merging process of entropy and KL-divergence is performed on distinct features of the incoming packets. The weighted outcomes of this combination serve as new features for the classifiers. For instance, in the ensemble learning section of this framework, if an SVM classifier is employed, the values w_1, w_2, w_3, w_4, and w_5 would be determined by the SVM classifier.

The proposed framework incorporates ensemble learning to achieve more accurate abnormal flow detection. By leveraging ensemble learning, multiple learning algorithms are employed to obtain superior predictions compared to individual learning algorithms used in isolation. Ensemble machine learning aids in determining the importance of features and yields precise results in the anomaly detection process. Furthermore, the ensemble method enables a better understanding of the

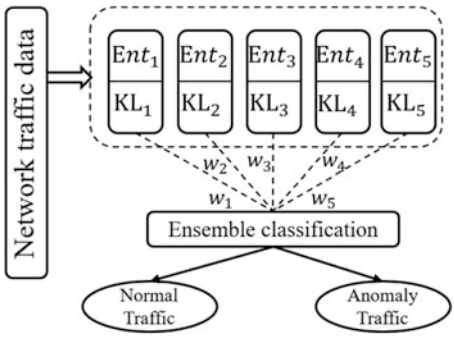


Fig. 2 Hybrid anomaly detection method

selection process and provides less biased estimates of membership probabilities within each feature group. Weights in the framework can be established based on the significance or correlation of the given features and the specific problem at hand. Many current DDoS detection methods in a single control plane rely on machine learning techniques, which have proven to be effective classifiers. In this study, ensemble learning [17, 38] is incorporated to enhance the accuracy of abnormal flow detection. By employing ensemble learning, multiple learning algorithms are combined to generate predictions that outperform those of individual algorithms when used alone. In the proposed framework, named Entropy-KL-ML, the final decision is reached through the collaboration of multiple base components. Each group of features contributes to creating a new feature for the classifiers by combining the results of entropy and KL-divergence. For example, if an SVM classifier is included in the ensemble learning section of this framework, the values assigned to w_1, w_2, w_3, w_4, and w_5 would be determined by the SVM classifier. Ensemble machine learning facilitates the identification of feature importance and ensures accurate results in the anomaly detection process. Determining the most effective feature distributions remains unclear, as various feature distributions have been proposed in the past. However, several recommended features demonstrate efficacy, including header-based features such as addresses, ports, and flags, volume-based features such as host-specific percentages of flows, packets, and bytes, and behavior-based features such as in/out connections for a particular host. Considering combinations and relationships between different features of packets and flows, such as packet type, src_I, dst_I, (src_I, src_P), (src_I, dst_P), (dst_I, src_P), (dst_P, src_P), and (src_P, L), can provide valuable insights in this regard.

5.2 Sample-Based RL Intrusion Detection Method

Given the potential loss of valuable information in uncaptured network traffic, determining sampling points and rates remains crucial. Once the sampling points and rates are established, the sampled traffic needs to be directed to one of several traffic analyzers for a thorough inspection. It is important to note, however, that

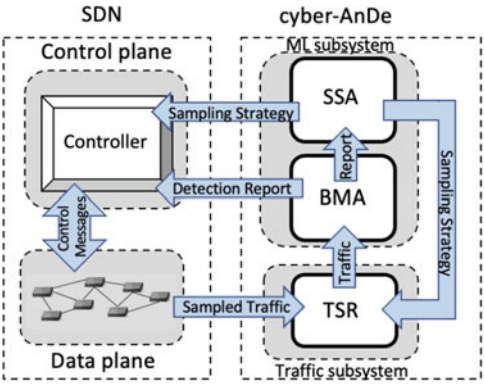


Fig. 3 Cyber-AnDe framework

this process may introduce additional overhead in terms of network delivery. To address these limitations and ensure the representativeness of the captured network behavior, a cybersecurity framework for SDN traffic monitoring has been developed. The proposed Cyber-AnDe framework, presented in Fig. 3, includes the following key components:

1. *Traffic Sample Repository (TSR):* This module collects the sampled traffic flows from the sampling switches of the data plane.
2. *Behavior Monitor Application (BMA):* This module is responsible for checking the sampled traffic's fields and identifying the headers. BMA can easily observe the packet's structure. It can roughly estimate the flow number and aggregate statistics, which can be helpful to detect anomalies.
3. *Sampler Scheduler Application (SSA):* This module determines the sampling strategy, i.e., which flow should be sampled by which switch and at what rate.

This framework incorporates two key algorithms that mitigate the impact on captured network behavior: (1) Switch selection algorithm: This algorithm selects switches based on their capacity to cover all incoming traffic, ensuring comprehensive coverage. (2) Sampling algorithm: This adaptive-distributed algorithm dynamically determines the optimal sampling rate based on the flow's current state. It effectively reduces the volume of traffic that needs to be analyzed, optimizing resource utilization.

BMA plays a crucial role in analyzing and reporting on the behavior of sampled traffic, as well as the specific features used in its assessment. These reports are then shared with both the SSA responsible for controlling sampling rates on switches in the data plane, and the controller. Based on the received reports from the BMA and SSA, the controller continuously makes informed decisions regarding which flows should be sampled on each switch and at what rate. It's important to note that these decision-making processes are centralized and ongoing, ensuring effective traffic management. When the BMA module receives sampled traffic from the TSR, it undertakes the processing and analysis of this data. Specifically, the BMA module focuses on examining packet header features such as source IP, destination IP,

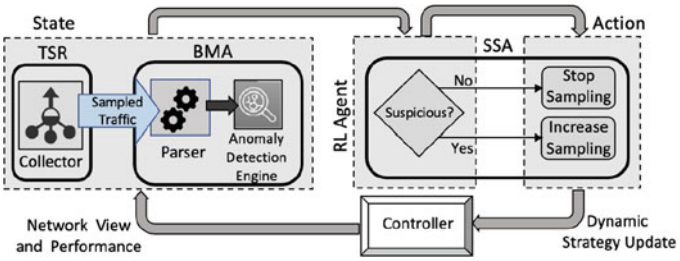


Fig. 4 ML_Subsystem

source port, destination port, transport protocol, flow size, and packet count fields. These features provide valuable insights into the current state of the traffic being monitored.

Following the sampling of traffic, BMA plays a vital role in generating a comprehensive report on the behavior and flow statistics of the sampled traffic. This report is shared with both the controller and SSA within the ML subsystem, as depicted in Fig. 4. Upon receiving input from the BMA, the SSA module adjusts the sampling rate accordingly. While a higher sampling rate is generally preferred as it improves the accuracy of malicious traffic detection by the controller, caution must be exercised. Indiscriminately increasing the sampling rate can lead to diminishing returns due to network congestion and increased overhead on the sampling switches and controller. To address this, the proposed Cyber-AnDe framework employs an adaptive distributed sampling method. It starts with a minimum sampling rate and gradually increases the sample size until no further improvement in detection accuracy can be achieved. The SSA module also provides recommendations to the controller regarding the appropriate sampling strategies to be employed on different switches. Drawing upon the input received from both the SSA and BMA modules, the controller makes informed decisions on the switches and sampling rates for the flows to be sampled. This ensures efficient and effective flow monitoring within the network.

The flows to be sampled, the sampling rates, and the sampling locations (i.e., switch locations) are determined by the controller such that the total network sampling utility is maximized without exceeding the sampling capacity constraint of each switch and the added overhead for the controller to manage the network. Ideally, the controller should employ different strategies for handling *legitimate*, *suspicious*, and *malicious* traffic. When the BMA, in its report, identifies flow f as legitimate, the controller sends a message to the associated switch to stop sampling f. Subsequently, no additional samples of f are sent to TSR from that switch. In the event flow f is identified as malicious in the BMA report, the controller will set up block actions on the flow tables for f. We would like to note that the forwarding rules are maintained within the flow tables on the SDN switches. A flow table typically includes fields that are used to identify a flow and specifies the corresponding action to take on that flow's packets. Finally, if f is identified as suspicious in the BMA report, the controller continues sampling f until it can

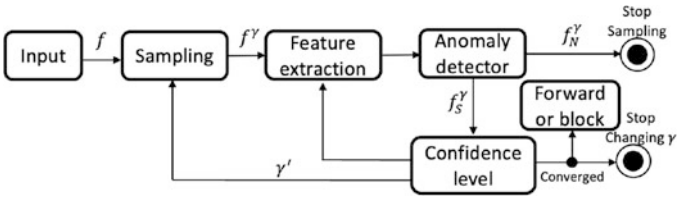


Fig. 5 Closed control loop of the Cyber-AnDe

accurately determine the status of f as either legitimate or malicious. In the case of suspicious traffic, the controller continues sampling until it can accurately determine the traffic's status. Figure 5 illustrates the RL control loop and stopping points. RL is a sub-field of machine learning that addresses the problem of learning optimal decisions over time. Based on the status that RL determines, the corresponding action to take can be one of: (1) Increasing the sampling rate, (2) Stopping the sampling rate, and (3) Adding an assistant switch.

The controller plays a crucial role in determining the sampling strategy for flows, including the sampling rates and locations (i.e., switch locations), while ensuring that the overall network sampling utility is maximized. This is done without exceeding the sampling capacity constraints of individual switches and minimizing the overhead for the controller's network management. It is ideal for the controller to employ different strategies to handle flows categorized as "legitimate," "suspicious," or "malicious." When the BMA identifies a flow as legitimate in its report, the controller instructs the associated switch to stop sampling that particular flow. Consequently, no further samples of that flow are sent to the TSR from that switch. In the case of a flow being identified as malicious in the BMA report, the controller takes action by setting up block rules in the flow tables of the SDN switches. These flow tables contain information to identify a flow and specify the appropriate action to be taken for the packets belonging to that flow. It is important to note that the forwarding rules, including blocking actions, are maintained within the flow tables of the SDN switches. Finally, if a flow is labeled as suspicious in the BMA report, the controller continues sampling that flow until it can accurately determine whether the traffic is legitimate or malicious. In the case of suspicious traffic, the controller persists in sampling until a conclusive determination of the traffic's status can be made. Figure 5 provides an illustration of the Reinforcement Learning (RL) control loop and the various decision points. RL, a sub-field of machine learning, focuses on learning optimal decisions over time. Based on the status determined by RL, the controller can take different actions, such as increasing the sampling rate, stopping the sampling rate, or adding an assistant switch to improve the sampling process.

In RL, the reward reflects the success of the agent's recent activity and not all the successes achieved by the agent so far. In our approach, the agent's objective is to learn the policy of monitoring the traffic that maximizes the expected detection rate and guarantees minimum overhead for the controller. RL would be helpful based on the traffic status on the given switch, which is based on the average rate of

received traffic and the average rate of loss traffic. We formulate the average rate of traffic $\overline{\mathcal{R}}(t)$ as $(R_p(t) - R_p(t-\tau))/\ (t-\tau)$, where $R_p(t)$ is received traffic rate at time t and τ denotes the end of the previous time interval. The average loss of traffic $\overline{\mathcal{L}}(t)$ can be formulated as $(R_p(t-\tau) - T_p(t-\tau))/\ R_p(t-\tau)$, where $T_p(t)$ represents the transmission rate of traffic at time t. The observed state of a switch is denoted as follows: $state = \{(\overline{\mathcal{R}}_1(t), \overline{\mathcal{L}}_1(t)), ...(\overline{\mathcal{R}}_n(t), \overline{\mathcal{L}}_n(t))\}$. We define actions of reallocation as 1) New allocation (extra space) for control at time t and 2) New allocation (extra space) for data at time t. Here, the aim of RL is to minimize the penalty, which is the cost of lack of space. The Q function is defined as:

$$Q_{t+1}(s_t, a_t) = (1-\alpha)(Q_t(s_t, a_t) + \alpha(P(s_t, a_t) + \lambda \max_{a_{t+1}} Q(s_{t+1}, a_{t+1})), \tag{4}$$

where $P(s_t, a_t)$ is the penalty function, and the value of penalty can be calculated by $C \times R_p(t)/b_p(t)$ where $R_p(t)$ and $b_p(t)$ represent the current rate and the allocated rate, respectively. Minimizing the probability of capture failure is the objective of the sampling methods. We can formulate the objective as $\min_\gamma\{\max_f p_f\} = \min_\gamma\{\max_f \prod_s p_{f,s}\}$. We need to evaluate the framework's performance with different sampling rates resulting from applications. We define a utility function $U_f(s, f, \gamma)$ to find the best option to the current system for sampling flow f in switch s using rate γ. The utility function can be defined as follows:

$$U_f(s, f, \gamma) = \sum\nolimits_{s\in S} \sum\nolimits_{f\in F} (\alpha \cdot \mathcal{G}(f_s, r_s, \gamma_f) - \beta \cdot \mathcal{M}(f) - \zeta \cdot \mathcal{P}(T, \tau)), \tag{5}$$

where $\mathcal{G}(f_s, r_f, \gamma_f)$ denotes the function computing the accuracy of detection of flow f_s on the given switch s with data rate r_f and sampling rate γ_f. The function $\mathcal{M}$ represents the cost value for computation/processing of flow f and communication between components. The function $\mathcal{P}$ represents the penalty, which is based on the delay in this approach. This utility function helps the controller to find [switch, flow, rate] by considering the capacity limitation of the TSR. We can formulate this problem as an optimization problem:

$$\begin{aligned} &\textit{maximize}: \quad U_f(s, f, \gamma) \\ &\textit{subject to}: \quad \sum\nolimits_{f\in F} r_f \cdot \gamma_s \le C \quad \text{for each } s \in S \\ &\qquad\qquad \sum\nolimits_{s\in S} s_f \ge 1 \quad \text{for each } f \in F. \end{aligned} \tag{6}$$

5.3 Deploying Chain of IDS in Data Plane

Each of these flows will be redirected through some IDSs in order to perform intrusion detection. Grouping incoming flows and using the same path for the

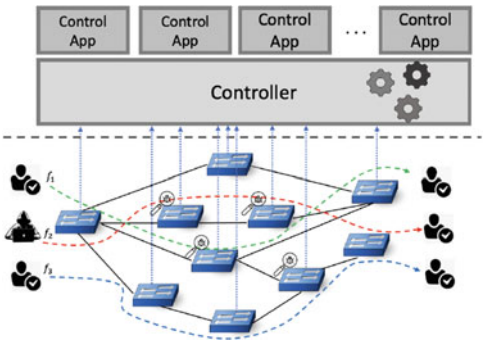


Fig. 6 Redirecting traffic through a chain of IDSs

flows in the same group can reduce this delay. Upon entry into the network, the classifier first categorizes the traffic pattern into suitable categories, then it assigns the IDS chain that is most appropriate to that traffic pattern. New arrival flows can be determined immediately, allowing the controller to deal with traffic dynamics [22]. A high volume of traffic in an SDN environment can overwhelm the controller, leading to network downtime. To address this issue, we propose a novel approach where data plane switches take on security functions as part of their packet processing logic. Figure 6 provides an overview of the SDN architecture, including the application layer, control plane, and data plane. In this network, we have flows such as f_1 and f_2, where $f_1 : s_1 \rightarrow d_1$ and $f_2 : s_2 \rightarrow d_2$. Each flow is redirected through appropriate IDSs for intrusion detection. This alleviates the burden on the controller, which typically handles multiple applications. By deploying IDS on selected switches in the data plane, we can significantly reduce the controller's workload. Moreover, having a larger number of IDSs increases the likelihood of detecting attacks for a given traffic flow. However, directing flows through specific paths that include IDSs can result in increased transmission delay. By grouping incoming flows and routing flows within the same group through the same path, we can reduce this delay. Upon entering the network, the classifier categorizes the traffic pattern and assigns the most suitable IDS chain for that particular pattern. This approach enables prompt identification of new incoming flows, allowing the controller to effectively handle traffic dynamics [22].

Figure 7 illustrates an example for three clusters and three IDS chains. Figure 7a shows the shortest path method, which calculates the distance between sources and destinations of flows and initial centroids. We have the distance measurement $dis(s_j, \overline{s}_k) + dis(d_j, \overline{d}_k)$, and flows would be divided into three clusters with centroids $\{c_1, c_2, c_3\}$. The GroupFlows would be assigned to the IDS chain based on the shortest hop count. Figure 7 shows $f_1(s_1, d_1)$, $f_2(s_2, d_2)$, and $f_5(s_5, d_5)$ are assigned to the first IDS chain based on the shortest path. $f_3(s_3, d_3)$ is assigned to the second IDS chain. $f_4(s_4, d_4)$ and $f_6(s_6, d_6)$ are assigned to the third IDS chain. Balanced clustering involves ensuring that there is an equal number of points in each cluster. Our approach is different from common techniques. To check if the groups are balanced, we use the total data rate of the groups instead of the number of group

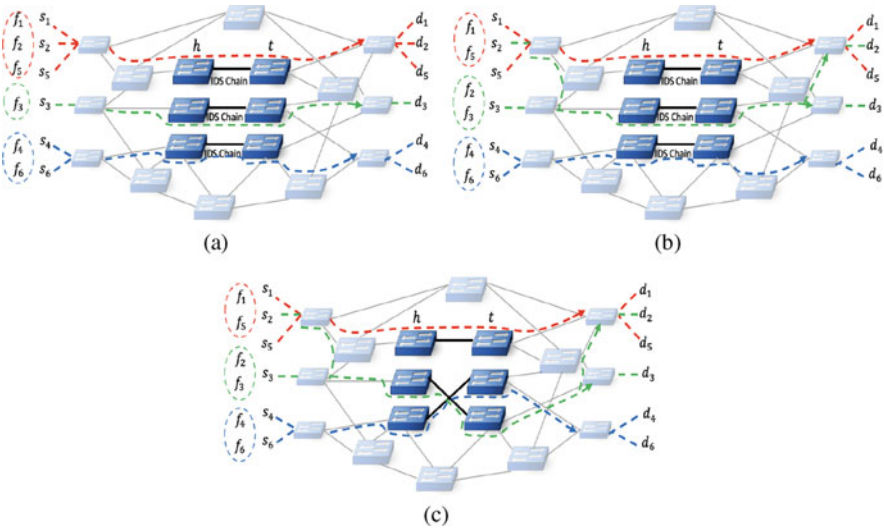


Fig. 7 Assigning IDS chain to the grouped incoming traffic. (**a**) Shortest path. (**b**) Balancing. (**c**) Perfect matching

members. For this example, Fig. 7a shows that c_1 has three members, c_2 has only one member, and c_3 has two members. In order to make a balance for the amount of processing on each IDS chain, we make a balance for the total amount of traffic in each group. The weight of a group can be defined as $W_i = \sum_{f \in F_i} n_i.w_f$. For this example, we assumed that the data rate of flow is the same; therefore, balancing would be based on the total number of members in each group. Figure 7b shows the balanced groups. Figure 7c shows the assigning of the heads and tails of the IDS chains to the source and destination of the centroids. h_i is the head of IDS chain i, and t_i is the tail of this IDS chain. All flows in a cluster k get a virtual center, including source $\overline{s}_k$ and destination $\overline{d}_k$. For the matching, which is assigning $\overline{s}_k$ to h_i and assigning $\overline{d}_k$ to t_j, where h_i and t_j are the head and tail of two different IDS chains. In these types of IDS chain, there are cross-connections between IDS chains. Based on the perfect matching algorithm, each balanced GroupFlow will be assigned to head and tail based on the smallest number of hops, which is the summation of the number of hops between the source to the head of the chain, the number of hops between head and tail, and the number of hops between the tail of chain and the destination. In the real test bed, we consider network delay, which is based on the number of hops and congestion on links.

Problem 1 The objective is to group incoming traffic in a balanced manner in order to reduce transmission delay. Two important factors to consider are the distance of flows to the centroid of each cluster and the total amount of traffic within each cluster. It is worth noting that this problem is NP-hard. To address it, we propose an approximation approach using a modified version of the K-means clustering

algorithm. We formulate the grouping incoming traffic problem as an optimization problem with the goal of minimizing overhead or cost.

$$\begin{aligned} \min \quad & \sum\nolimits_{F_j \in F} cost(F_j) \\ \text{subject to} \quad & cost(F_j) = |F_j| \cdot \sum\nolimits_{f \in F_j} r_f. \end{aligned} \tag{7}$$

Here, $cost(F_j)$ represents the cost of clustering incoming traffic f. The cost factor reflects the additional workload imposed on the controller when grouping incoming traffic. It is calculated based on the total number of flows and the traffic rate, denoted as r_f, within each cluster F_j. We assume a simplified scenario where r_f is equal to 1.

Problem 2 Determine the optimal assignment of IDS chains to flow groups to minimize the number of malicious packets, ensuring that all traffic passes through an IDS chain before reaching the destination. It is assumed that the locations of IDS chains are predetermined. The problem can be expressed as the following:

$$\begin{aligned} \min \quad & \sum\nolimits_{i \in I} cost(I) \\ \text{subject to} \quad & cost(I) = \sum\nolimits_{M_{j,i}=1} R_j * \min dist(F_j, I_i) \\ & R_j = \sum\nolimits_{f \in F_j} r_f, \quad 1 \leq |I_i|. \end{aligned} \tag{8}$$

$cost(I)$ represents the cost of assigning a flow group to an IDS chain I_i. This is based on the total traffic rate of each flow group and the distance between the centroid of the flow group and the IDS chain. R_j denotes the total traffic rate of flow group j. r_f denotes the data rate of flow f. The distance between IDS chain I and flow group F_j is shown by $dist(F_j, I_i)$, which is the number of hops between h_i and $\overline{s}_j$. $M_{j,i}$ is a matrix that shows each flow group F_j is assigned to which I_i.

6 Measurements

The evaluation or assessment of a system, mechanism, or method typically represents a momentary depiction of its quality or accuracy. Over time, as the system is established and the environment evolves, new vulnerabilities emerge, necessitating a reassessment that includes parameter tuning. Nevertheless, it is important to note that the information acquired during an initial evaluation process holds substantial importance in subsequent evaluations. While most supervised machine learning algorithms excel in classification tasks, their proficiency in logic modeling may be limited. In contrast, DL-based approaches have surpassed traditional machine learning techniques in logic modeling, demonstrating superior performance.

6.1 Effectiveness

When evaluating the performance of an anomaly detection model, it is important to consider different types of classification errors. Relying solely on the traditional accuracy metric, which measures the total number of correct classifications divided by the total number of classifications, is insufficient for accurately assessing the skill of an anomaly detection model. That classification (or prediction) result is divided into four classes:

- True positive (TP): Identified anomaly occurrence correctly as an anomaly.
- False positive (FP): Identified regular occurrence wrongly as an anomaly.
- True negative (TN): Identified normal occurrence correctly as normal.
- False negative (FN): Identified anomaly occurrence wrongly as normal.
- Accuracy provides an overall measure of the model's performance in terms of correctly classifying both positive and negative instances, which is obtained by $ACC = (TP + TN)/(TP + TN + FP + FN)$.

6.2 Efficiency

In network anomaly detection, it is important to have a fast and efficient algorithm that can quickly process large amounts of network data in real-time. If the processing time is too long, the algorithm may not be able to detect anomalies in a timely manner, which can result in security breaches or other issues. The processing time of a network anomaly detection algorithm can be influenced by various factors, such as the size of the data, the complexity of the algorithm, the hardware used, and the implementation of the algorithm. A trade-off often exists between processing time and accuracy, so it's important to find a balance between the two in order to have an efficient network anomaly detection system. Therefore, processing time should be considered along with other metrics such as accuracy, precision, recall, and F1 score when evaluating the efficiency of network anomaly detection methods. In order to select the most suitable evaluation measures for an anomaly detection method, it is important to consider the specific goals of the method as well as the associated costs of false positives and false negatives. When aiming to optimize precision, one may prioritize reducing human workload or minimizing the cost of failure. In contrast, optimizing for high recall may be more appropriate when the cost of a false negative is high. To strike a balance between precision and recall, the detection threshold can be adjusted according to the desired trade-off.

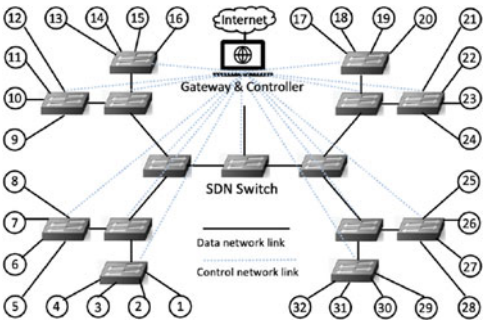


Fig. 8 SDN topology

6.3 Evaluation of Specific Intrusion Detection Methods on SDN

Our data center, illustrated in Fig. 8, consists of the following components: 35 servers, 15 SDN switches, and 4 regular L2 switches. The servers, excluding the Gateway, are Dell PowerEdge 210 servers with the following specifications: a 2-core 2.4 GHz processor, 4 GB RAM, 500 GB storage, and a minimum of 2 gigabit Ethernet ports. To establish the network infrastructure, we have set up two distinct networks: a control network and a data network. In the control network, an L2 switch connects all the management ports of the SDN switches and the SDN controller. The SDN switches are configured as out-of-band controllers, which effectively decouples the control and data planes. As a result, our control network follows a star topology. In the data network, the data ports of the SDN switches and the Gateway are interconnected, forming a three-level complete binary tree topology. The Gateway is linked to the root SDN switch, while the remaining servers are connected to the leaf SDN switches. In this section, we conduct an assessment of specific intrusion methods and present the results obtained. Our experimental setup, depicted in Fig. 8, utilizes the ONOS as the SDN controller and Mininet for creating diverse network topologies. Mininet allows for the creation of realistic virtual networks with authentic kernel, switch, and application code, facilitating the development of OpenFlow applications. Both ONOS and Mininet are operated on a Windows desktop equipped with a 3.5GHz Intel Core i3 CPU and 16GB of memory. To evaluate the detection of DoS attacks, we generate a consistent flow rate between each pair of hosts in the network. During the experiment, normal traffic is injected into the network using scapy, followed by the launch of a DoS attack from a switch to a host. Various ML algorithms and feature selection methods are employed for the detection of DoS attacks. The performance of these algorithms is evaluated using metrics such as processing time, overhead, FPR, and accuracy. FPR is the probability of misclassifying a packet as normal when it is actually an attack. The evaluation of accuracy considers the ability of the classifier to correctly classify samples in relation to the total number of samples, providing insights into the discrimination capabilities of the classifier. The evaluation encompasses various

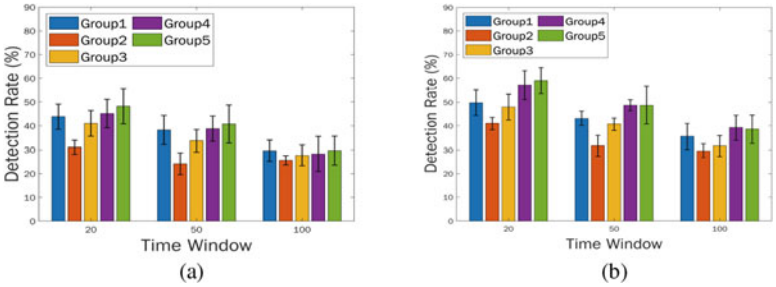


Fig. 9 Detection rate under different groups of features on different window time. (**a**) Attack rate 35%. (**b**) Attack rate 80%

Table 1 Feature selection

# Group	No. features	Selected features
1	1 features	$E_{(src_I \mid dst_I)}$
2	2 features	E_{src_I}, E_{dst_I}
3	4 features	E_{src_P}, E_{dst_P}, E_{src_I}, E_{dst_I}
4	5 features	E_{src_P}, E_{dst_P}, E_{src_I}, E_{dst_I}, E_S
5	6 features	E_{src_P}, E_{dst_P}, E_{src_I}, E_{dst_I}, E_S, R

scenarios, including different detection methods, network topologies, and attack rates.

The controller fulfills its role by retrieving data from switches' flow tables, enabling the monitoring of active flows and tracking packet counts for each flow. In this research, we propose harnessing this capability to incorporate feature processing into the decision-making process. A significant factor to consider is the duration of the monitoring window. We will assess the performance of our proposed combined detection method by evaluating its detection rate and comparing it to the grouping approach. Figure 9 illustrates the detection rate of the Entropy-KL-ML anomaly detector across various scenarios with different attack rates, utilizing the feature groups summarized in Table 1. Our experimental findings demonstrate that an increased number of features leads to enhanced performance. The combination of KL-divergence and entropy effectively addresses the uncertainties associated with entropy thresholds, resulting in improved accuracy and decreased false positive rate (FPR) in anomaly detection.

Furthermore, the employment of ensemble learning in conjunction with our proposed feature selection enhances the detection outcomes across diverse scenarios. While the impact of network topology on anomaly detection is minimal, the choice of classifier significantly influences the results. Table 2 presents a comparison between a single IDS and multiple IDSs in terms of their detection rates, dropping rates, and delays. The incoming traffic is categorized as small, medium, or large based on its total volume, with 500 flows classified as small, 2000 flows as medium, and 8000 flows as large. We conducted various measurements under different attack rates of 20, 50, and 80%. The results indicate that deploying multiple IDSs

Table 2 Evaluating IDS under one IDS vs multiple IDSs

Traffic	Attack rate	Detection rate(%)			Dropping rate(%)			Delay (ms)		
		1 IDS	2 IDS	Mixed	1 IDS	2 IDS	Mixed	1 IDS	2 IDS	Mixed
Small	20%	36.6	48	52	24.9	26.3	25	1.8	3.45	3.3
	50%	47.5	55	60	25.5	26.9	26.2	3.6	6.9	6.45
	80%	52	69	72	24.8	26.7	25.1	6.1	11.31	10.8
Medium	20%	49.3	64.5	74.5	28.7	30.5	29.9	5.55	9.99	9.57
	50%	60.3	71	73	28	29.5	28.9	7.1	15	14.1
	80%	72	81	83	28.9	32	31.5	13.5	24.9	24.51
Large	20%	61.8	80.3	85	31.2	34	32.7	9.6	17.4	16.5
	50%	74.1	86	91	34.5	36.3	35.18	17.1	33.3	32.82
	80%	81	92	94.3	35	37.5	38.7	30	54.6	54

Table 3 Evaluating IDS under different amounts of incoming traffic

Anomaly detection	Overhead (%)			Dropping rate (%)			Detection rate (%)			Delay (ms)		
	S	M	L	S	M	L	S	M	L	S	M	L
Centralized IDS	7	12	27	32.6	37	43.2	39.4	53.3	68.3	2.7	5.3	19.2
Chain with one IDS	10.2	12.3	17.8	31	28.5	33.8	38.5	60.3	74.1	3.6	7.1	23.1
Chain with two IDS	10.3	12.3	18	32.9	31.5	35.6	55	71	86	9.6	15	30.3

has a positive impact on the detection rate of malicious packets, resulting in a lower missing rate. However, it also leads to a higher dropping rate. Although the utilization of multiple IDSs increases the delay time compared to a single IDS, the increase is not significant. This is because previous IDSs block or drop certain portions of traffic, resulting in a reduced volume of incoming data for subsequent IDSs. Moreover, an increase in the attack rate enhances the detection rate and reduces the missing rate. The number of samples plays a crucial role in the detection capability of an IDS, as a larger sample size increases the likelihood of detecting attack packets. Interestingly, our results indicate that the attack rate does not have an impact on the dropping rate. The dropping rate of packets is primarily influenced by the capacity limitations of switches and is not affected by the proportion of malicious packets in the network. However, when the attack rate increases, it results in an elevated delay as switches must notify the controller for necessary actions. In scenarios with high traffic volumes, the detection rate, missing rate, and dropping rate all tend to rise due to the greater occurrence of attacks.

Table 3 provides an overview of deploying IDS in both the control plane and the data plane, considering various metrics such as overhead, missing rate, dropping rate, detection rate, and delay. The evaluation of IDS deployment in the data plane takes into account the volume of incoming traffic, while the number of IDSs in each chain affects all the evaluation metrics. The experimental results presented in Fig. 10 demonstrate that the Entropy-KL-ML method achieves the highest accuracy compared to other anomaly detection approaches, including pure entropy, pure ML,

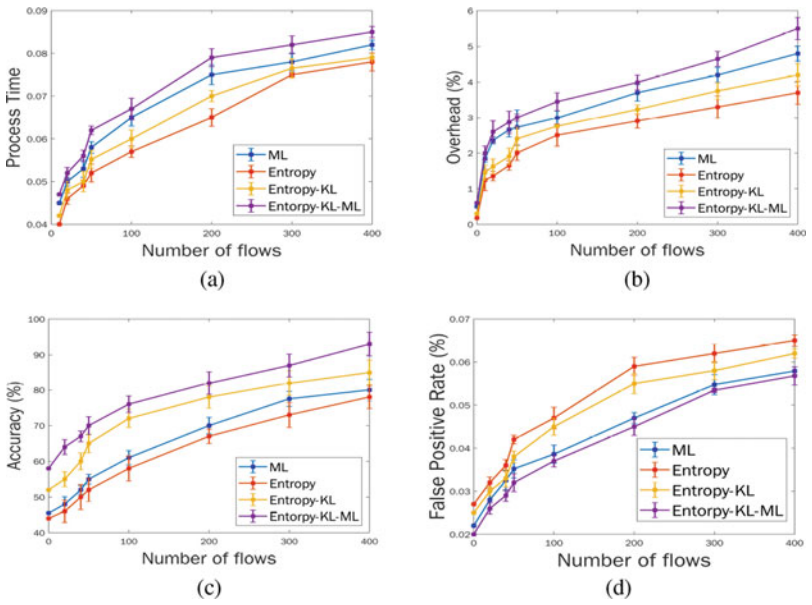


Fig. 10 Evaluation of different ML-based anomaly detection methods. (**a**) Processing time. (**b**) Overhead. (**c**) Accuracy. (**e**) FPR

and the combination of entropy and KL-divergence. Although the combined method exhibits slightly longer processing time, as shown in Fig. 10a, the difference is not significant and remains within acceptable limits. In terms of CPU utilization, as depicted in Fig. 10b, all approaches show increased CPU usage with an increasing number of flows. However, the Entropy-KL-ML approach utilizes CPU resources at a significantly lower rate compared to the other methods. Despite a slight increase in overhead on the controller when incorporating KL-divergence with entropy, the Entropy-KL-ML method, with its unique combination of Entropy-KL and ML algorithms along with additional feature processing, consistently outperforms other methods in terms of accuracy. The results in Fig. 10c, d reveal that the Entropy-KL-ML approach achieves higher accuracy (approximately 91.9%) and lower false positive rate (approximately 0.055%) compared to other approaches. The Entropy-KL approach also demonstrates acceptable accuracy (around 81.7%). However, the combination of ensemble learning with Entropy-KL in the proposed approach enhances the decision-making process of anomaly detectors. Despite the higher processing time, the Entropy-KL-ML approach stands out as a distinctive and effective anomaly detector due to its high accuracy and low false positive rate.

In our experimental study, we investigated the influence of network topology on anomaly detection and compared the accuracy of various machine learning classifiers. We examined two simulated topologies: Stanford and FatTree(4). The Stanford topology consisted of 26 switches, 26 hosts, and 650 flows, with each switch connected to a single host. On the other hand, the FatTree(4) topology had

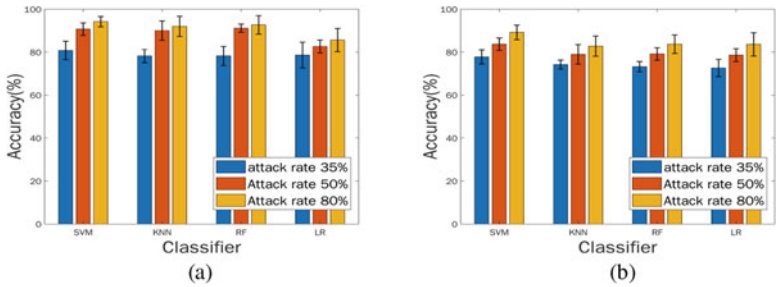


Fig. 11 Evaluating the detection rate of different classifiers under different rate of attack. (**a**) Standford. (**b**) FatTree(4)

20 switches, 16 hosts, and 240 flows, with each edge switch connected to a single host. We generated flows between host pairs at the same rate in both networks. Based on the results shown in Fig. 11, we observed that the network topology had minimal impact on the performance of anomaly detection. Furthermore, we conducted evaluations using different classifiers and found that SVM demonstrated excellent performance in accurately predicting the decision function to differentiate between normal and anomalous classes. Other classifiers also exhibited comparable performance, particularly when the attack rate was set at 80%.

7 Conclusion

This chapter provides a review of ML and DL techniques for network anomaly detection. ML-based approaches enhance the accuracy of IDS. Various ML and DL-based intrusion detection mechanisms are discussed, emphasizing the use of SDN for vulnerability detection and network monitoring. Challenges include identifying attack sources, handling high network traffic volumes, and responding effectively to attacks. Intelligent security methods based on ML and DL are more effective than traditional approaches, with DL techniques being particularly efficient in evaluating network security. Ongoing research focuses on the adaptability of detection methods, feature selection, and utilizing DL for dataset classification. Hybrid approaches and DL techniques show promise in detecting network anomalies. Runtime limitations are a major challenge for NIDS. Real-time NIDS systems should capture and analyze each packet in line with the current network scenario to ensure seamless packet flow and accurate detection. Minimizing false alarms is a crucial objective for an effective intrusion detection method or NIDS. While completely eliminating false alarms may be challenging for anomaly-based systems, it is essential to strive for zero false alarms in all environments. Additionally, the system should be adaptable at runtime. Meeting these objectives poses a demanding task for the NIDS development community. As intruders continuously

modify their network attacks to circumvent existing intrusion detection solutions, the characteristics of anomalies undergo constant change. Therefore, it is imperative that the adaptability of a NIDS or detection method remains up-to-date with the current anomalies that arise within the local network or on the internet. In the context of distributed attacks, where multiple machines can be compromised rapidly, the immediate damage to the network can be significant. To effectively mitigate such attacks, a NIDS must not only detect them early, but also have the capability to control the attack rate without disrupting the service for legitimate users. This objective poses a considerable challenge for NIDS development.

Acknowledgments The work was supported in part by NSF grants CPS 2128378, CNS 2107014, CNS 2150152, CNS 1824440, CNS 1828363, and CNS 1757533.

References

1. Abdallah M, An Le Khac N, Jahromi H, Delia Jurcut A (2021) A hybrid CNN-LSTM based approach for anomaly detection systems in SDNS. In: 16th IEEE international conference on availability, reliability and security, pp 1–7
2. Abdel Azim NM, Fahmy SF, Sobh MA, Eldin AMB (2021) A hybrid entropy-based DoS attacks detection system for software defined networks (SDN): a proposed trust mechanism. Egypt Inf J 22(1):85–90
3. Anantvalee T, Wu J (2007) A survey on intrusion detection in mobile ad hoc networks. In: Wireless network security. Springer, Berlin, pp 159–180
4. Arulkumaran K, Deisenroth MP, Brundage M, Bharath AA (2017) Deep reinforcement learning: a brief survey. IEEE Signal Process Mag 34(6):26–38
5. Ashodia N, Makadiya K (2022) Detection of ddos attacks in SDN using machine learning. In: International conference on electronics and renewable systems (ICEARS), pp 1322–1327
6. Aydın H, Orman Z, Aydın MA (2022) A long short-term memory (LSTM)-based distributed denial of service (DDoS) detection and defense system design in public cloud network environment. Comput Secur 118:102725
7. Carvalho RN, Bordim JL, Alchieri EAP (2019) Entropy-based DoS attack identification in SDN. In: IEEE international parallel and distributed processing symposium workshops (IPDPSW), pp 627–634
8. Chandola V, Banerjee A, Kumar V (2009) Anomaly detection: a survey. ACM Comput Surv 41(3):1–58
9. Chen M, Challita U, Saad W, Yin C, Debbah M (2019) Artificial neural networks-based machine learning for wireless networks: a tutorial. IEEE Commun Surv Tuts 21(4):3039–3071
10. Chetouane A, Karoui K (2022) A survey of machine learning methods for DDoS threats detection against SDN. In: International workshop on distributed computing for emerging smart networks, pp 99–127
11. Einy S, Oz C, Navaei YD (2021) The anomaly-based and signature-based IDS for network security using hybrid inference systems. Math Problems Eng 2021:6639714
12. Elsayed MS, Le-Khac NA, Dev S, Jurcut AD (2019) Machine-learning techniques for detecting attacks in SDN. In: 7th IEEE international conference on computer science and network technology (ICCSNT), pp 277–281
13. Elsayed MS, Le-Khac NA, Dev S, Jurcut AD (2020) DDoSNet: a deep-learning model for detecting network attacks. In: 21st IEEE international symposium on a world of wireless, mobile and multimedia networks (WoWMoM), pp 391–396

14. Garg G, Garg R (2015) Detecting anomalies efficiently in SDN using adaptive mechanism. In: 5th IEEE international conference on advanced computing & communication technologies
15. Goldfeld Z, Greenewald K, Niles-Weed J, Polyanskiy Y (2020) Convergence of smoothed empirical measures with applications to entropy estimation. IEEE Trans Inf Theory 66(7):4368–4391
16. Hadem P, Saikia DK, Moulik S (2021) An SDN-based intrusion detection system using SVM with selective logging for IP traceback. Comput Netw 191:108015
17. Iftikhar N, Baattrup-Andersen T, Nordbjerg FE, Jeppesen K (2020) Outlier detection in sensor data using ensemble learning. Proc Comput Sci 176:1160–1169
18. Kim S, Yoon S, Lim H (2021) Deep reinforcement learning-based traffic sampling for multiple traffic analyzers on software-defined networks. IEEE Access 9:47815–47827
19. Lee TH, Chang LH, Syu CW (2020) Deep learning enabled intrusion detection and prevention system over SDN networks. In: IEEE international conference on communications workshops (ICC), pp 1–6
20. Maleh Y, Ezzati A, Qasmaoui Y, Mbida M (2015) A global hybrid intrusion detection system for wireless sensor networks. Proc Comput Sci 52:1047–1052
21. Malik R, Singh Y, Sheikh ZA, Anand P, Singh PK, Workneh TC (2022) An improved deep belief network IDS on IoT-based network for traffic systems. J Adv Transp 2022:1–17
22. Niknami N, Wu J (2022) Enhancing load balancing by intrusion detection system chain on SDN data plane. In: IEEE conference on communications and network security (CNS), pp 264–272
23. Niknami N, Wu J (2022) Entropy-KL-ML: enhancing the entropy-KL-based anomaly detection on software-defined networks. IEEE Trans Netw Sci Eng 9(6):4458–4467
24. Niknami N, Inkrott E, Wu J (2022) Towards analysis of the performance of IDSs in software-defined networks. In: 19th ieee international conference on mobile Ad Hoc and smart systems (MASS), pp 787–793
25. Nugraha B, Murthy RN (2020) Deep learning-based slow DDoS attack detection in SDN-based networks. In: IEEE conference on network function virtualization and software defined networks (NFV-SDN), pp 51–56
26. Oshima S, Nakashima T, Sueyoshi T (2010) Early DoS/DDoS detection method using short-term statistics. In: IEEE international conference on complex, intelligent and software intensive systems
27. Otoum Y, Nayak A (2021) AS-IDS: anomaly and signature based IDS for the internet of things. J Netw Syst Manag 29(3):23
28. Phan TV, Nguyen TG, Dao NN, Huong TT, Thanh NH, Bauschert T (2020) Deepguard: efficient anomaly detection in SDN with fine-grained traffic flow monitoring. IEEE Trans Netw Service Manag 17(3):1349–1362
29. Raikar MM, Meena S, Mulla MM, Shetti NS, Karanandi M (2020) Data traffic classification in software defined networks (SDN) using supervised-learning. Proc Comput Sci 171:2750–2759
30. Rinaldi G, Adamsky F, Soua R, Baiocchi A, Engel T (2019) Softwarization of SCADA: lightweight statistical SDN-agents for anomaly detection. In: 10th international conference on networks of the future (NoF), pp 102–109
31. Sampaio LS, Faustini PH, Silva AS, Granville LZ, Schaeffer-Filho A (2018) Using NFV and reinforcement learning for anomalies detection and mitigation in SDN. In: IEEE symposium on computers and communications (ISCC), pp 00432–00437
32. Singh S, Jha RK (2017) A survey on software-defined networking: architecture for next generation network. J Netw Syst Manag 25(2):321–374
33. Sultana N, Chilamkurti N, Peng W, Alhadad R (2019) Survey on SDN based network intrusion detection system using machine learning approaches. Peer Peer Netw Appl 12:493–501
34. Sutton RS, Barto AG (2018) Reinforcement learning: an introduction. MIT Press, Cambridge
35. Tang TA, Mhamdi L, McLernon D, Zaidi SAR, Ghogho M, El Moussa F (2020) Deepids: deep learning approach for intrusion detection in software-defined networking. Electronics 9(9):1533

36. Zhang H, Liu D, Luo Y, Wang D (2012) Adaptive dynamic programming for control: algorithms and stability. Springer, Berlin
37. Zhao G, Zhang C, Zheng L (2017) Intrusion detection using deep belief network and probabilistic neural network. In: IEEE international conference on computational science and engineering (CSE) and IEEE international conference on embedded and ubiquitous computing (EUC), vol 1, pp 639–642
38. Zhong Y, Chen W, Wang Z, Chen Y, Wang K, Li Y, Yin X, Shi X, Yang J, Li K (2020) Helad: a novel network anomaly detection model based on heterogeneous ensemble learning. Comput Netw 169:107049

Part III
Attack and Defense in Artificial Intelligence-Enabled Wireless Systems

Deep Learning for Robust and Secure Wireless Communications

Hai N. Nguyen and Guevara Noubir

1 Introduction

The development of mobile technologies and wireless communications has led to a profound revolution in society. Today, mobile phones are used by billions of people worldwide to access information, connect on social media, and engage in various daily activities. This rapid progress is fueled by advancements in wireless communications, including increased throughput and reductions in size and power consumption. Wireless integration in everyday devices has significantly impacted system design and operation, leading to a surge in wireless connectivity and a decrease in wired links. This shift is particularly evident in critical Cyber-Physical systems, such as the use of Wireless Remote Terminal Unit (RTU) in the monitoring and control of SCADA systems including the electricity grid and industrial processes.

Effective communication that is both secure and reliable is crucial in modern wireless systems. However, there are various challenges that must be overcome to achieve this goal. With the emergence of new applications such as Massive IoT (MIoT), robotics, autonomous cars, and augmented reality, the demand for spectrum has increased significantly. This trend has led to spectrum scarcity and unintentionnal interference, ultimately degrading communication quality. Furthermore, wireless protocols are today widely implemented in software, and Software-Defined Radio platforms are increasingly capable, with small form factor and low cost (e.g., XTRX platform achieves 120 Msps, 2x2 MIMO, and integrated a GPSDO and FPGA in a mini PCI form factor [15]). This trend, while creating new opportunities for developing sophisticated communication techniques, poses new challenges for

H. N. Nguyen · G. Noubir (✉)
Khoury College of Computer Sciences, Northeastern University, Boston, MA, USA
e-mail: nguyen.hai@northeastern.edu; g.noubir@northeastern.edu

Y. Chen et al. (eds.), *Network Security Empowered by Artificial Intelligence*,
Advances in Information Security 107, https://doi.org/10.1007/978-3-031-53510-9_6

spectrum management and security against wireless threats, from smart jammers, compromised wireless chips [41], to weaponized drones [13, 47]. More specifically, powerful jammers that can disrupt wireless communications have been increasingly capable to achieve. They can be implemented on Mica2 WSN platform [58] or software-defined radios [33], or can be found on the Internet for a few dozens of dollars. These challenges, coupled with the complex natural artifacts of wireless channels, such as propagation loss, fading, and shadowing, make achieving robust and secure wireless communication extremely difficult.

Deep Learning (DL) has recently achieved significant success, demonstrating impressive performance across a variety of research areas, including computer vision [24], speech recognition [19], and natural language processing [57], as well as mastering complex games like Dota 2 [1] and Go [49]. This success has inspired innovation in DL-based wireless communications. One of the primary advantages of DL is the ability to learn complex relationships between variables through large amounts of data. This allows us to leverage the vast and diverse raw data collected through a variety of wireless sensors, and to design communication systems without needing accurate mathematical models. Moreover, with the emergence of parallel computing accelerators such as NVIDIA GPU with CUDA [37], Google TPU [18], or Intel Nervana [21], even sophisticated DL models are now becoming possible to deploy for real-time systems. Our research presented in this chapter is inspired by such potentials of Deep Learning. We focus on improving the robustness and security of wireless communications through a three-pronged approach, presented in the rest of the chapter as follows.

Identifying Emissions and Collisions Section 2 studies Deep Learning-based methods for identifying different types of RF emissions and collisions in the wideband spectrum. Detecting wireless collisions is essential in determining if the degradation of a communication link is due to adversarial collisions (i.e. caused by jammers) or collisions with harmless users. Furthermore, identifying the characteristics of those interference sources, such as RF technologies, transmission time, or frequency slots, is a crucial first step to improve the robustness of communications. We first study how to transform RF data into visual data with a multi-channel image-based spectral representation. Then, a collision detection approach using the VGG-16 neural network [50] is presented. Furthermore, the section presents the application of YOLO algorithm in expanding the approach for the simultaneous classification and localization of emissions in the 100 MHz spectrum, resulting in the development of the real-time identification system WRIST [35, 36].

Canceling Adversarial Interference In Sect. 3, we focus on the countermeasure against adversarial interference, specifically from jammers. When a jamming signal successfully interferes with communication, traditional spread spectrum and jammer avoidance techniques become ineffective. To recover communication in such scenarios, it is necessary to remove the jamming component from the received signal. We propose a jamming cancellation system called JaX, which utilizes Convolutional Neural Network to infer the existence of interference, the number of interfering emissions and their respective phases. Our system eliminates the

requirement of explicit channel estimation schemes such as pilot or reference signal. Additionally, we present experimental studies to evaluate the effectiveness of our anti-jamming approach against different types of high-power jammers.

Enhancing Received Signal Section 4 introduces DEFORM, a universal beamforming system that leverages the diversity of multiple receiving antennas to optimally improve the quality of various types of transmitted signals. We explore the design of a deep neural network that accurately estimates the optimal beamforming parameters. In addition, special features of the design, which specifically address the ambiguous 2π phase discontinuity of RF complex samples and the high sensitivity of the link Bit Error Rate, are presented. We demonstrate the universality of the system through extensive numerical and experimental analysis, as well as in a beamforming-relay application for LoRa and ZigBee.

2 Deep Learning for Identifying RF Emissions and Collisions

The prevalence of wireless threats raises challenging research questions regarding the development of scalable techniques for understanding, managing, and protecting the RF spectrum. To achieve that goal, it is crucial to understand the spectrum, both in real-time and a-posteriori, and detect, classify, and identify spectro-temporal information of the communications. In this section, we introduce two research works for Deep Learning-based RF identification: In the first work [34], we develop a wireless collision detection scheme using the VGG-16 Deep Convolutional Neural Network. The second work extends the first by creating a real-time wideband spectro-temporal RF identification system based on the YOLO detection network and several optimization techniques.

2.1 Literature Studies on RF Identification

RF identification have attracted significant attention in the research community over the past decades. Researchers have spent some efforts investigating this problem using various expert features, including higher-order statistical features [12, 52]. Nonetheless, those approaches require domain knowledge, and redesign of algorithms for new generation technologies. Deep Learning, which allows for automatic feature extraction and learning, has emerged as a promising solution. Deep Learning have recently achieved very good performance across various RF identification tasks, such as modulation recognition [38], radar detection [45], collision detection [34], or RF fingerprinting of ZigBee [30] and LoRa [14] devices. However, there are two key unmet requirements in previous works: *real-time* and *wideband* spectrum processing. In [2, 46], authors use Deep Neural Networks to classify three popular 2.4 GHz technologies: Wi-Fi , Bluetooth, and ZigBee.

However, these approaches lack the capability to operate in real-time for wideband 2.4 GHz spectrum, and to localize emissions in the spectrum. Meanwhile, other works [40, 53] have only considered identifying emissions in simulation settings.

A good training dataset is essential for any Deep Learning approach to be effective. It is even more important to make the data available to the research community to promote the development of new RFML techniques, architectures, and models. In [9], the authors published a dataset comprising both simulated and recorded over-the-air signals of various modulations for the modulation recognition task, where the RF technology information is unavailable. In [46], the authors created a dataset of Wi-Fi, Bluetooth, and Zigbee signals transmitted from a signal generator instead of commercial RF devices. Nonetheless, these datasets lack data of concurrent and colliding communications sampled at much higher rate than the standard bandwidth of RF technologies.

2.2 Visual-Based Spectral Representation

Our approach to RF Identification utilizes cutting-edge Deep Learning techniques in computer vision. However, RF domain is very different to visual domain, as the former consider radio signals in the form of complex-valued (or I/Q) samples, while the latter works with pixel data. In order to bridge this gap, we have developed a technique to transform raw RF samples into visual data. Firstly, we divide the I/Q data stream into equal-sized chunks and convert the data of each chunk into the frequency domain using the N-point Fast Fourier Transform (FFT) algorithm. The FFT outputs for M chunks are combined to create an $M \times N$ matrix of complex samples. Finally, we create a 2D grayscale image representing the 2D view of frequency spectrum by mapping each element $m_{x,y}$ (located in column x and row y) to its corresponding integer value $p_{x,y}$ of the pixel at coordinate (x, y):

$$p_{x,y} = f(A_{x,y}) := \gamma * (\min(\max(A_{x,y}, A_{min}), A_{max}) - A_{min}) \quad (1)$$

where $A_{x,y} = 20 * \log_{10}|m_{x,y}| - N_0$ is the SNR of emission at frequency bin x of the y-th chunk with respect to the noise floor N_0. $A_{min} = -10$, $A_{max} = 50$ are respectively the pre-calculated minimum and maximum SNR values in the spectrum. $\gamma = 255/(A_{max} - A_{min})$ is the scaling factor of the mapping from SNR to pixel.

2.3 Detecting Wireless Collisions

2.3.1 Learning from Synthetic Data

To minimize manual efforts for labeling data and quickly train the Deep Learning model, we created a synthetic dataset that consists of three distinct classes: No

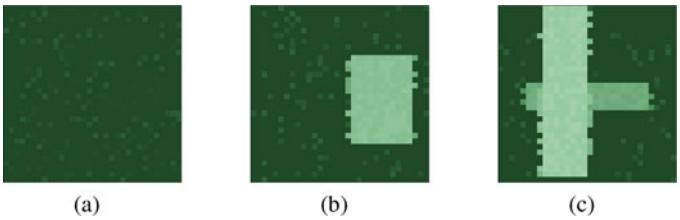


Fig. 1 Examples of the generated synthetic data: (**a**) No transmission (**b**) Transmission with no collision (**c**) Transmission with collision

Transmission, Transmission with No Collision, and Transmission With Collision with examples in Fig. 1. The generated synthetic data mimics the real communications observed from DARPA SC2 Colosseum testbed [8]. Our synthetic dataset comprises 150,000 samples of size 32x32, which are divided into three subsets: 96,000 samples for training, 24,000 for validating, and 30,000 for testing.

Using the synthetic dataset, we trained the VGG-16 Deep Convolutional Neural Network [50] with some first layers pre-trained on ImageNet dataset [10] for comprehensive visual features. Pre-training the initial layers of the network does not compromise the recognition of RF signals, as signals share similar fundamental visual features with real-life objects (such as edge features and brightness). We opted for VGG-16 due to its exceptional performance in image classification. The network was trained using the Adam optimizer [23] implemented in the Keras library [5], with the objective of classifying samples into the three aforementioned classes.

2.3.2 Evaluation

We evaluated the performance of our method with both synthetic and real data. Our approach yielded a remarkable accuracy of 99.87% on 30,000 samples from the test synthetic dataset, with over 99% for all three classes. Additionally, we conducted tests on real data collected from the DARPA SC2 Colosseum testbed [8]. In this test, we transformed I/Q samples using 256-point FFT averaging over a 256 ms time window. Data was then divided into 2-D matrices of size 256x256 representing a 20 MHz spectrum. To match the training data, each matrix was further divided into an 8x8 grid with each cell having a size of 32x32. The grid cells were manually classified into three categories for evaluation purposes. The testbed exhibited an extremely congested spectrum, as depicted in Fig. 2, with many RF emissions close together, resulting in out-of-band leakages that visually degrade the discrimination of collisions and separate emissions. Despite this challenging scenario, our Deep Learning model achieved over 94% accuracy in classifying collisions and an overall accuracy of 87.5%, demonstrating the effectiveness of the Deep Learning approach for identifying RF collisions.

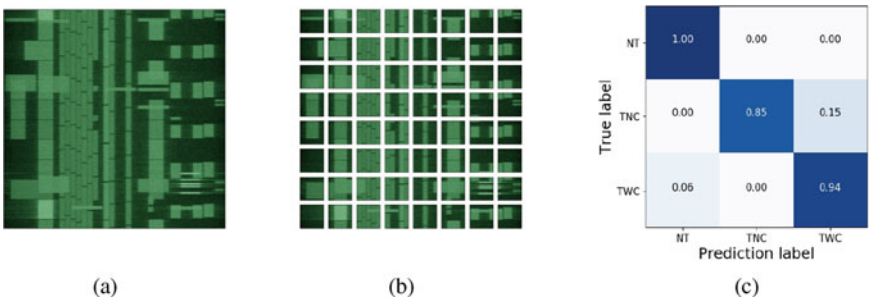


Fig. 2 The spectrum image (**a**), input data (**b**) collected in DARPA SC2's Colosseum testbed, and the classification results (**c**)

2.4 Real-Time, Wideband Spectro-Temporal RF Identification

Inspired by the success of the initial work, we have expanded our approach and developed a novel RF identification system called WRIST [36]. The primary objectives of the system are threefold: (1) Precise classification and spectro-temporal localization of RF signals, (2) real-time processing capability, and (3) support for the 100 MHz-wide 2.4 GHz ISM frequency spectrum.

2.4.1 Deep Learning Model and Optimizations

Our Deep Learning approach takes inspiration from YOLO [3], which is one of the fastest object detection algorithms in the literature. YOLO is an *one-stage object detection* method that provides end-to-end processing with only a single neural network, making it much faster than *two-stage object detection* methods relying on slow and complex pipelines [16, 43]. Moreover, we introduce two Deep Learning optimizations to the YOLO algorithm to enhance the speed and accuracy RF identifications, described as follows.

1. **RF-centric Anchor Boxes**: The YOLO network uses multiple *bounding boxes* to identify emissions in the spectrum image, with each box representing a potential emission. More specifically, the network divides the input into a $S \times S$ grid, where each cell generates B bounding boxes that predict emissions with centers located within the cell. To capture objects with different aspect ratios, the YOLO algorithm uses a set of predefined bounding boxes of specific sizes for each grid cell, known as *anchor boxes* [42], as the references for the predicted objects. These anchor boxes need to reflect accurately the objects which the DL model learns. However, unlike real-life objects, which the original YOLO model is trained on, RF emissions typically have highly variable sizes due to varying packet duration and bandwidth. Therefore, to improve the performance of the Deep Learning model for RF identification, we replaced the original YOLO

anchor boxes based on real-life objects with *RF-centric* anchor boxes generated using K-means clustering algorithm on the training dataset of RF emissions,

2. **Optimized Convolutional Layers Stack:** The YOLO network can achieve real-time processing in computer vision [3]. However, utilizing the off-the-shelf YOLO network for wideband, real-time RF identification remains a challenge. To address this, we devised traanother optimization technique. We selectively reduced the volume of convolutional filters with the observation that visualized RF emissions are sharper and simpler than real-life objects, which were the initial targets for the YOLO design. This implies that fewer useful features are required to extract, resulting in a smaller volume of convolutional filters that can still accurately identify those emissions. We reduced the filters step-by-step using the formula $U_i = U_{i-1} \times (1 - \sigma^i)$, where $\sigma = 0.5$ and U_i is the filter volume at the i^{th} step. We continued the reduction until observing a surge in the validation error (after $i = 2$ in our experiments). This approach helped to reduce the size of the Deep Neural Network by 62.5%, resulting in a more than 2.2 times faster model while maintaining the same level of prediction performance.

2.4.2 RF-Centric Compression

While using *one-stage object detection* can improve the detection speed, it is insufficient for the real-time RF identification of wideband spectrum. Specifically, the YOLO network is the bottleneck of our system when processing a 100 MHz-wide spectrum: It requires tens of milliseconds to process 100 MHz I/Q samples that span only a few milliseconds. Increasing the input size to address this challenge would increase the network size and slow down processing further. Our solution is the RF-centric compression layer as the first layer of the real-time YOLO model, which compresses multiple input images while retaining important features. The compression consists of two steps: In the first step, the layer combines M_1 FFT outputs into one average chunk of Signal-to-Noise Ratio (SNR) values. In the second step, the layer groups M_2 outputs from the first step into chunks and maps the chunks to the R-G-B color channels of the final output, using the respective average, max, and min operations:

$$\begin{aligned} R_{x,y} &= f(10 \times \log_{10} E_{x,y}^{max} - N_0), \\ G_{x,y} &= f(10 \times \log_{10} E_{x,y}^{min} - N_0), \\ B_{x,y} &= f(10 \times \log_{10} E_{x,y}^{avg} - N_0), \end{aligned} \tag{2}$$

where $f(z)$ mapping function is defined in Eq. 1. While the compression discards some original information, it still preserves essential signal properties in the final output. These properties, including the high and low peaks of RF emissions, or signal strength variations over time, are particularly useful to distinguish between different RF technologies.

2.4.3 Experimental Results

We used a comprehensive set of evaluation metrics designed specially for spectro-temporal RF identification [36]. This includes the *Class Detection Accuracy* p_d evaluating how well the Deep Learning model can recognize the presence of specific RF classes in the wideband spectrum, as well as the *Emission Detection Metrics* evaluating the detection capability on the fine-grained emission level: Precision pr_e, Recall re_e, F1-score $F1_e$, Bandwidth (BW) offset ratio $r_{\Delta BW}$, and Time offset ratio $r_{\Delta t}$. By utilizing these metrics, we can thoroughly evaluate the performance of WRIST through extensive experimentation in various real-life over-the-air environments.

Over-the-Air Dataset We evaluated WRIST using the test portion of the real emission dataset as the first evaluation setting. WRIST achieved over 99% of class detection accuracy and over 0.99 of emission detection precision for all classes. Figure 3 depicts WRIST's performance with respect to various classes and SNR levels. It achieved over 0.99 for precision pr_e regardless of categories and SNRs. WRIST also obtains a recall re_e of over 0.94 for most cases, except high SNR Wi-Fi, where the emissions start to create confusing visual patterns such as RF leakages. In all cases, F1-score maintains above 0.96, while BW offset ratio $r_{\Delta BW}$ is below 0.14 and time offset ratio $r_{\Delta t}$ is below 0.12.

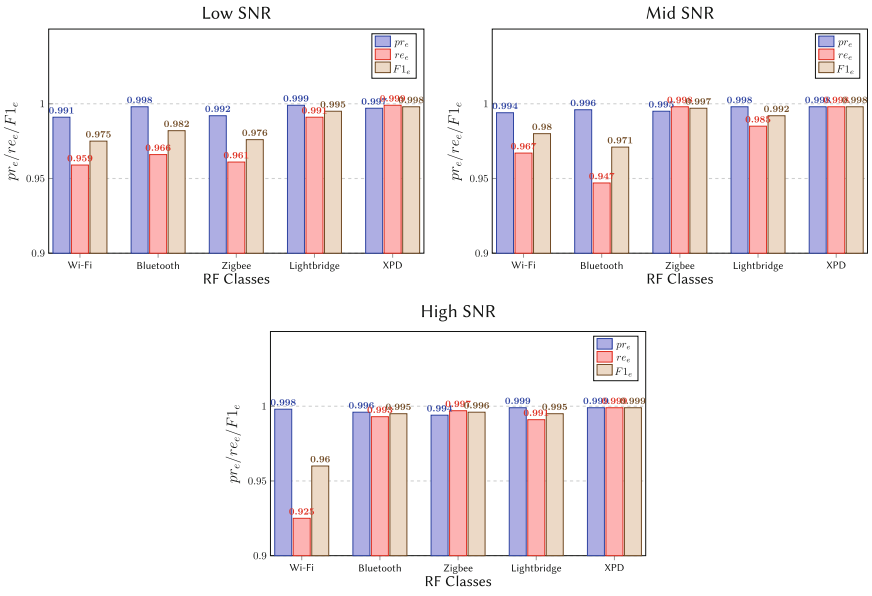
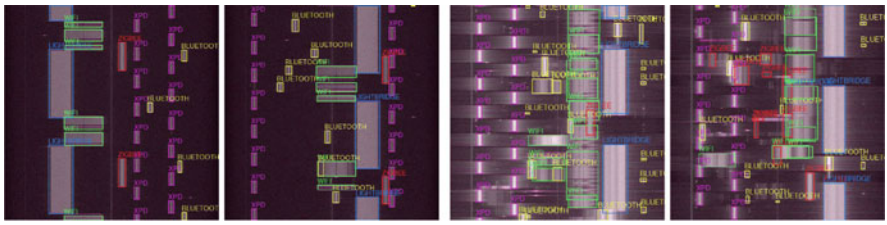


Fig. 3 WRIST performance with respect to different RF classes and SNRs



(a) (b)

Fig. 4 WRIST's desirable detection performance in congested environments. (**a**) Inside anechoic chamber. (**b**) In the wild

Anechoic Chamber For the second evaluation, we conducted experiments in a 60 × 60 × 30 ft anechoic chamber at the George J. Kostas Research Institute of Northeastern University. To create a realistic crowded spectrum, we operated all RF devices simultaneously in different locations inside the chamber. The high collision rate and new pattern appearances in the spectrum resulted in a slight decline in WRIST's performance, compared to the previous evaluation. Nonetheless, it still maintained very high score of pr_e (0.969) , re_e (0.94) and $F1_e$ (0.954), with very small BW (0.055) and time (0.062) offset ratios. Figure 4a illustrates the Wi-Fi, Bluetooth, ZigBee, Lightbridge and XPD emissions identified correctly by WRIST with the respective green, yellow, red, blue and purple rectangular boxes.

In-the-Wild Environment In the final evaluation, we collected and annotated RF emissions from a densely populated in-the-wild environment with the illustrated spectrum in Fig. 4b. We can see that there are even greater volume of RF emissions and more complex collision patterns in this case, making the task difficult even for human. As a result, WRIST's performance further degraded compared to previous evaluation settings. However, WRIST still maintained remarkable performance with high precision (0.87), recall (0.83) and F1-score (0.849). Moreover, both $r_{\Delta BW}$ and $r_{\Delta t}$ remained under 0.06, indicating that WRIST can precisely recognize the spectro-temporal information of emissions even in the extremely congested frequency spectrum.

2.5 SPREAD Dataset

The availability of large, labelled datasets of RF emissions is crucial to scientific research. It promotes collaboration, provides valuable data to those who lack the necessary equipment, and drives the development of new techniques, models, and DL architectures for RF research. In the absence of a large, curated dataset for spectro-temporal identification of different RF technologies, we are making our

dataset SPREAD[1] available to the research community. SPREAD currently supports five RF categories: Wi-Fi, Bluetooth, Zigbee. Lightbridge (Wireless communication protocol of DJI drones [11]), and XPD (Samson's wireless microphone [44]). The dataset contains spectrum images, RF samples, and dataset metadata, and we also provide the supporting API for the expansion of dataset with more supported technologies and devices.

3 Deep Learning for Canceling Adversarial Interference

In this section, we consider the application of Deep Learning for mitigating interference in adversarial environments. Once the RF emissions that disrupt the user's communication have been identified by the spectro-temporal RF identification approach in the previous section, it is essential to eliminate such interference to restore the link quality, especially when it originates from an adversary.

3.1 Motivation

In contrast to unintentional interference, adversarial interference or jamming involves the deliberate use of wireless signals to disrupt target communications. Despite significant efforts to combat jammers in the last few decades, jamming remains one of the most serious threats to wireless communications today. Traditional anti-jamming at the physical layer has relied on spread spectrum techniques, which require the coordinating nodes to share a secret key in advance. Recent research has attempted to address this limitation for FHSS [26, 51], or DSSS [27, 39], or both [22]. Nonetheless, these approaches are primarily designed to remove the pre-shared secret for spread spectrum and not to counter powerful jammers, which can be a few orders stronger than the sending node.

In recent years, Deep Learning have been applied to combat jamming, with the main purpose of avoiding jammer interference [20, 28]. However, they do not consider high-power jammers successfully interfering with the communications. In such scenario, it becomes essential to cancel the jamming to preserve the quality of the communication. Currently, most existing jamming cancellation techniques rely on complicated mechanical jamming-dampening scheme [55], or using pilot signals [59, 61], which results in significant communication overhead.

[1] Abbreviated from **Sp**ectro-temporal **R**F **E**mission **A**nalysis **D**ataset. Dataset is available at https://sprite.ccs.neu.edu/datasets/SPREAD/.

3.2 System Model and Problem Formulation

We consider a system composed of two communicating nodes, where the sender has a single antenna and the receiver has two identical antennas. They communicate using a predetermined channel and link parameters, including frequency, bandwidth, and modulation. A single-antenna adversary attempts to disrupt the communication by transmitting jamming signals on the same channel. The received signal R_i of antenna i consists of the transmitted signal S, jamming signal J, each adjusted by the corresponding channel gains h_{S_i} and h_{J_i}, and additive white Gaussian noise N_i:

$$\begin{aligned} R_1 &= h_{S_1} S + h_{J_1} J + N_1, \\ R_2 &= h_{S_2} S + h_{J_2} J + N_2. \end{aligned} \tag{3}$$

Here, we are considering a slow-fading channel, which implies that the involved parties have low mobility. Additionally, we assume that the channel gains remain fairly stable throughout the considered bandwidth. The jammer can transmit either random samples or modulated packets, with a continuous or intermittent pattern.

Considering a jammer significantly above the noise, the decodability of signal S is dependent on the Signal-to-Interference-and-Noise Ratio (SINR) which is approximated as proportional to $\frac{|h_S|^2}{|h_J J|^2}$. The signal S is undecodable when jamming signal becomes stronger, which subsequently reduces $\frac{|h_S|}{|h_J J|}$. To achieve jamming cancellation, we transform Eq. 3, resulting in the following equation:

$$R_1 - p_1 R_2 = p_2 S, \tag{4}$$

where $p_1 = \frac{h_{J_1}}{h_{J_2}}$, and $p_2 = h_{S_1} - p_1 h_{S_2}$. If p_2 is sufficiently large, we can achieve a good SINR to decode S. Equation 4 shows that estimating parameter p_1 correctly is the key requirement for this jamming cancellation scheme. To find $p_1 = \frac{h_{J_1}}{h_{J_2}} = \frac{|h_{J_1}|}{|h_{J_2}|} e^{j(\phi_{J_1} - \phi_{J_2})}$, we need to estimate the *amplitude ratio* $A_J = \frac{|h_{J_1}|}{|h_{J_2}|}$ and the *phase shift* Δ_{ϕ_J}:

$$\Delta_{\phi_J} = \phi_{J_1} - \phi_{J_2}. \tag{5}$$

3.3 JaX Jammer Cancellation Scheme

3.3.1 Multi-Functional Convolutional Neural Network

JaX relies on a Convolutional Neural Network (CNN) for **detecting emissions** and **estimating phase shifts**. We define two goals for designing the CNN:

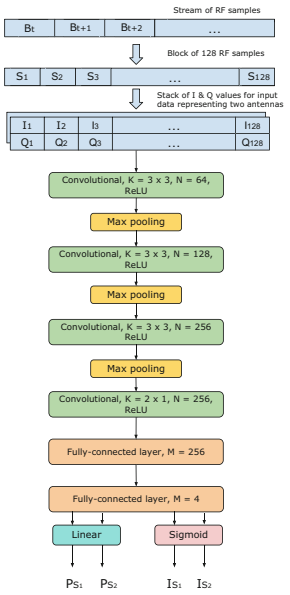


Fig. 5 The CNN structure of JaX, where K is the filter size and N is the number of filters of convolutional layers. M is the number of neurons in fully-connected layers

- Two phase shift estimations are needed, not only for the jamming signal but also for the legitimate signal. This is because with a single estimation, we cannot determine which signal is associated with the phase shift.
- It is required to distinguish between the data-containing signals and noise for each phase shift estimation to ensure that the cancellation only operates when jamming is present.

The CNN architecture that performs signal detection and phase estimation is illustrated in Fig. 5, which has four outputs: P_{S_1} and P_{S_2} estimate the phase shifts for the legitimate and jamming signals. On the other hand, I_{S_1} and I_{S_2} determine whether the corresponding phase shift estimations come from a signal or noise, where a value of 1 implies real signal, and 0 implies noise. As P_{S_1} and P_{S_2} cannot be used interchangeably, we differentiate them by having P_{S_1} learn the smaller phase shift, while P_{S_2} learns the larger one. During the training phase, the CNN minimize loss function $\mathcal{L}$ that comprises the Mean Square Error loss $\mathcal{L}_\phi$ for phase shift estimations and the Binary Cross-Entropy loss $\mathcal{L}_S$ for signal detections:

$$\begin{aligned} \mathcal{L}_\phi &= 1_{S_1}(\Delta_{\phi_1} - P_{S_1})^2 + 1_{S_2}(\Delta_{\phi_2} - P_{S_2})^2 \\ \mathcal{L}_S &= -((1_{S_1}\log(I_{S_1}) + 0_{S_1}\log(1 - I_{S_1})) + \\ &\quad (1_{S_2}\log(I_{S_2}) + 0_{S_2}\log(1 - I_{S_2}))) \\ \mathcal{L} &= \alpha\mathcal{L}_\phi + (1 - \alpha)\mathcal{L}_S, \end{aligned} \tag{6}$$

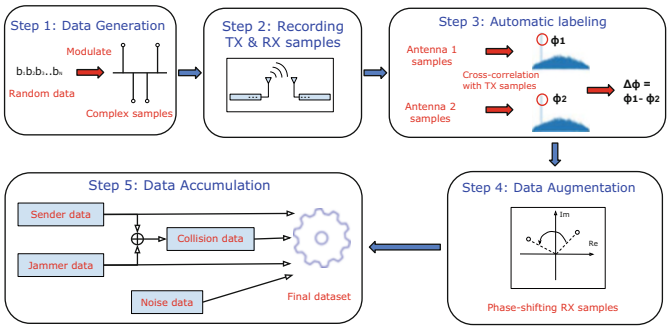


Fig. 6 Dataset collection for jamming detection and cancellation

where Δ_{ϕ_i} ($i \in \{1, 2\}$) is the respective ground truth of P_{S_i}. 1_{S_i} is 1 if Δ_{ϕ_i} associates with a signal, otherwise 0. 0_{S_i} is the complement of 1_{S_i}, and $\alpha = 0.1$ is the scaling factor for the two loss components.

To achieve sufficiently large dataset to train the CNN, we propose an efficient data collection approach, shown in Fig. 6. We first transmitted pre-generated RF samples and saved the received samples to files. Then, we calculated the phase shifts by cross-correlating the received samples with the transmitted samples. To increase the diversity of the dataset, we randomly shifted the phase of RF samples by a value within the range $[-\pi, \pi]$ and adjusted the labels accordingly. This process was repeated for both the sender and the jammer, and collision data were created by combining the RF samples of user signal and jammer signal together.

3.3.2 Analyzing CNN Output and Canceling Jammer

At time period T, the receiver collects a block of RF samples that is inputted into the CNN model to get the phase estimation $P_{S_i}^T$ and the associated signal detection output $I_{S_i}^T$ where $i \in \{1, 2\}$. Determining the present state of the channel is a crucial step, which is accomplished using the signal detection indicator $1_{S_i}^T$ based on the value of $I_{S_i}^T$:

$$1_{S_i}^T = \begin{cases} 1 & I_{S_i}^T > 0.5 \\ 0 & \text{otherwise} \end{cases} \quad \forall i \in \{1, 2\}. \tag{7}$$

$1_{S_i}^T$ being equal to 1 or 0 indicates that S_i (whose phase shift estimated by $P_{S_i}^T$) is a real signal or noise, respectively. We then analyze $1_{S_i}^T$ and perform jamming cancellation according to the following cases:

- **Both** $1_{S_i}^T (i \in \{1, 2\})$ **are 0**: This indicates the current channel is vacant and no action will be taken.

- **Only one $1^T_{S_i}$ is 1**: This means either the sender of the jammer is transmitting. We determine if the transmitter is the jammer by checking whether the RF samples are decodable. If a jammer is detected, we record the estimated phase shift.
- **Both $1^T_{S_i}$ are 1**: In this case, we detect the presence of a jammer disrupting the communication. The jamming phase shift is determined out of the two phase estimations by selecting the one that is closer to the jamming phase shift recorded in the previous time step. This is done based on the assumption of a slow-fading channel in our setup, where the phase shift changes slowly over time. Additionally, to eliminate estimation variations and outliers, we stabilize the estimated jamming phase shift by using the exponential smoothing function with smoothing factor λ:

$$\Delta_{\phi_J} = \Delta^T_{\phi_J}\lambda + \Delta^{cur}_{\phi_J}(1-\lambda). \tag{8}$$

The amplitude ratio $A_J = \frac{|h_{J_1}|}{|h_{J_2}|}$ is estimated by analyzing the difference of the signal power in the periods before and during the collision [32]. More specifically, if the sender transmitted right before the collision with power E_S, then jamming power can be calculated from the signal power E at the collision by:

$$E_{J_i} = E_i - E_{S_i}, \tag{9}$$

with $i \in \{1, 2\}$. On the other hand, if the jammer transmitted before the collision, the jamming power can be taken directly in that period. Then, amplitude ratio is calculated as $A_J = \frac{|h_{J_1}|}{|h_{J_2}|} = \sqrt{\frac{E_{J_1}}{E_{J_2}}}$. With both phase shift and amplitude ratio calculated, the receiver can solve Eq. 4 with $p_1 = \frac{h_{J_1}}{h_{J_2}} = A_J e^{j\Delta_{\phi_J}}$ to cancel the jamming signal.

3.4 Experimental Analysis

3.4.1 Comparison with Pilot-Based Cancellation

Existing approaches [59, 61] rely on pilot signals to estimate channel gains for canceling jamming signal. However, these approaches have several limitations: (1) They lead to significant communication overhead. (2) Their accuracy typically decreases in time-varying channels. (3) They require compatibility between the transmitter and receiver. JaX was designed to eliminate those limitations. To demonstrate the advantages, we compared JaX with a pilot-based approach called BJM [61], which leverages pilots to minimize the decoding Mean Square Error for optimal reception quality. In our evaluation, we simulated a time-varying Rayleigh channel with multipath fading and uncontrolled phase alignment between the

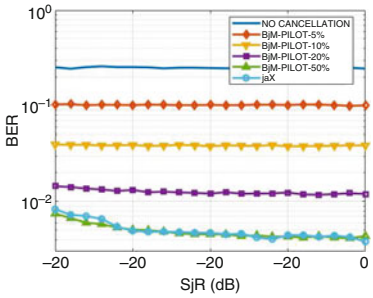


Fig. 7 Comparison of JaX and BJM [61] over Rayleigh channel

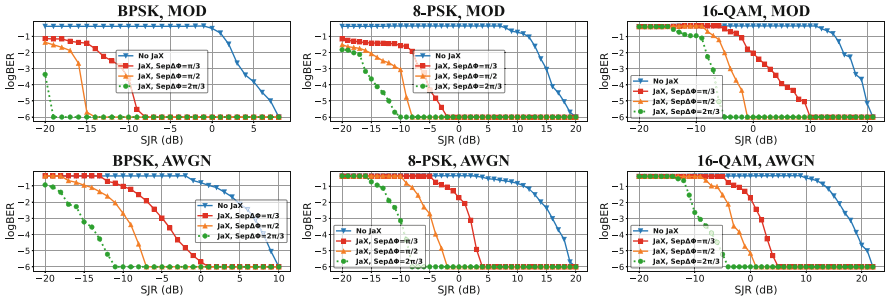


Fig. 8 JaX's performance in over-the-cables experiments with MOD jammer (1*st* row) and AWGN jammer (2*nd* row)

jammer and sender using MATLAB software. The comparison between the two approaches is shown in Fig. 7. The results highlight the advantages of JaX over pilot-based systems, as JaX achieves comparable performance to BJM using 50% transmitted signals for pilots (equivalent to 50% overhead) and outperforms BJM with lower pilot utilization.

3.4.2 Impact of Phase Alignment and Jammer Type

Multi-antenna jamming cancellation has intrinsic limitations. Removing jamming signal J results in signal S being subject to an update gain value $h_{S_1} - p_1 h_{S_2}$ where $p_1 = \frac{h_{J_1}}{h_{J_2}}$. This gain is small when $\frac{h_{S_1}}{h_{S_2}} \approx \frac{h_{J_1}}{h_{J_2}}$, equivalently $\Delta_{\phi_S} \approx \Delta_{\phi_J}$ i.e. the jammer and sender are phase-aligned ($Sep_{\Delta_\phi} = \left|\Delta_{\phi_S} - \Delta_{\phi_J}\right| \approx 0$). We investigated this impact in Fig. 8 with over-the-cables experiments. We can see from the results that: (1) JaX achieves very high jamming resilience against a jammer of up to 19 dB stronger than the sender. (2) JaX is effective with both lower-order modulation (BPSK) as well as higher order modulation (8-PSK, 16-QAM). (3) Jamming cancellation is more effective with MOD jammer transmitting modulated signal than AWGN jammer. (4) The performance declines as Sep_{Δ_ϕ} decreases.

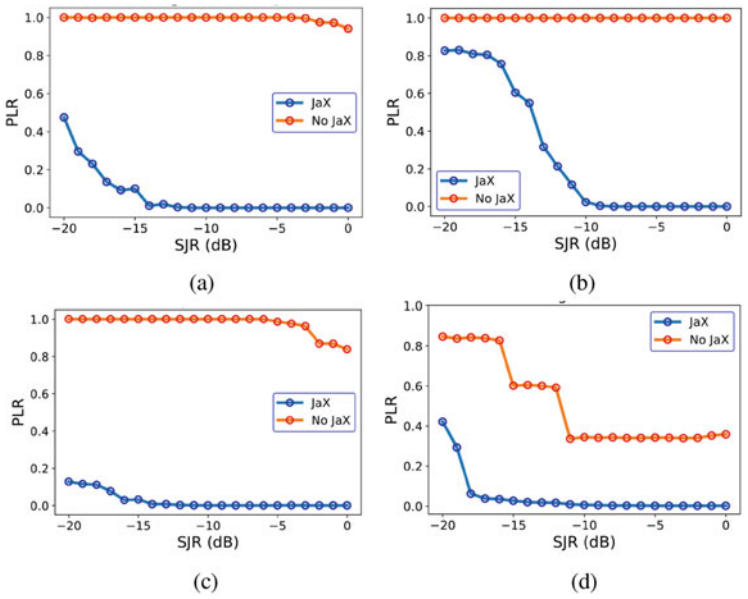


Fig. 9 JaX's performance against different jammers in over-the-air experiments. (**a**) MOD jammer. (**b**) AWGN jammer. (**c**) VAR jammer. (**d**) INT jammer

This confirms the intrinsic limitation of multi-antenna jamming cancellation as the receiver cannot resolve two transmitters that are aligned with each other.

3.4.3 Over-the-Air Performance

We evaluated JaX in an indoor over-the-air environment that contains various RF-blocking and reflecting objects, such as computers, monitors, walls, and desks. We used four different jammers: MOD and AWGN jammers (which transmit continuous, constant-power modulated or AWGN signals), VAR jammer that transmits power-variable signals, and INT jammer that transmits intermittent signals. The evaluation results in Fig. 9 show that JaX is capable of canceling up to 18 dB of jamming power, maintaining a Packet Loss Rate (PLR) under 0.1.

We see that JaX is also robust under the impact of multi-path in indoor environment. This does not contradict the cancellation theory, as we can explain as follows. Due to the impact of multi-path, each receiving antenna gathers multiple copies of the legitimate and jamming signals:

$$\begin{aligned} R_1 &= \sum_i h^i_{S_1} S + \sum_i h^i_{J_1} J + N_1, \\ R_2 &= \sum_i h^i_{S_2} S + \sum_i h^i_{J_2} J + N_2, \end{aligned} \tag{10}$$

where $h^i_{S_k}$ and $h^i_{J_k}$ are the respective channel gains of the ith path from the sender and the jammer to the antenna k of the receiver. It is evident that with $h_{S_k} = \sum_i h^i_{S_k}$ and $h_{J_k} = \sum_i h^i_{J_k}$, Eq. 10 can be considered as equivalent to Eq. 3. Therefore, the sum of the channel gains of all the paths from the sender/jammer to the receiver can be viewed as a new channel gain of the line-of-sight path between the receiver and the sender/jammer at a different location.

4 Deep Learning for Enhancing RF Receiver with Universal Beamforming

4.1 Motivation

In this section, we explore an alternative method to improve the robustness of wireless communications. Rather than tackling the interference, we enhance the quality of user's received signals by using a multi-antenna beamforming approach. Beamforming is a spatial filtering technique that is widely used in systems targeting high throughput and spectral efficiency, such as cellular systems since the third generation 3GPP 3G, and IEEE 802.11n. Despite extensive research over the last few decades [4, 54, 56, 62], beamforming in today's systems still requires explicit channel estimation mechanisms like sounding and feedback in IEEE 802.11, Demodulation Reference Signal (DMRS) in 5G, training sequences, etc. These mechanisms have several drawbacks, including significant overhead to transmit reference signals, long delays associated with accurate channel estimation, and the need for compatibility between transmitter and receiver to agree on when, what, and how reference signals are transmitted.

Deep Learning offers a promising solution for beamforming that eliminates the need for explicit mechanisms. By analyzing complex patterns in raw I/Q data collected at the PHY layer, a deep neural network can quickly and accurately estimate channel characteristics. Deep Learning does not require compatibility between transmitter and receiver, and enables the development of a universal beamforming component that can support different RF technologies. For example, a drone equipped with a technology-agnostic relay can enable communications between devices without line-of-sight, bringing connectivity to first-responders and other ad hoc communications in disaster recovery scenarios. Another application is a universal beamforming-relay that extends the range and bridges IoT devices operating on the same frequency. Nonetheless, despite of the great potentials, previous work on DL-based RX beamforming [25, 60] is restricted in analytical and simulation evaluation, and moreover, lacks the goals of universality.

4.2 Beamforming Theory

We consider a communication system consisting of a single-antenna transmitter and a N-antenna receiver. Assuming that all the receiving branches are synchronized in time, the received signal R_i^t at time t from antenna i comprises the instantaneous transmitted signal S^t adjusted by the channel gain h_i^t and the additive Gaussian noise N_i^t:

$$R_i^t = h_i^t S^t + N_i^t = s_i^t + N_i^t. \tag{11}$$

Using beamforming at the receiver, we combine N receiving branches with the adequate complex *beamforming weights*:

$$R_{\sum}^t = \sum_{i=1}^{N} a_i^t R_i^t = \sum_{i=1}^{N} (a_i^t s_i^t + a_i^t N_i^t). \tag{12}$$

The *beamforming weights* a_i are chosen to maximize the combining Signal-to-Noise Ratio (SNR), which is given by:

$$SNR_{\sum}^t = \frac{(\sum_{i=1}^{N} a_i^t s_i^t)^2}{N_0^t B T_s \sum_{i=1}^{N} |a_i^t|^2} = \frac{(\sum_{i=1}^{N} a_i^t s_i^t)^2}{N_0^t \sum_{i=1}^{N} |a_i^t|^2}, \tag{13}$$

where we assume the noises in different branches are independently and identically distributed (i.i.d) with a Power Spectral Density (PSD) N_0^t at time t, and pulse shaping such that $BT_s = 1$ (B is the bandwidth and T_s is the sampling period). The Cauchy-Schwartz inequality [17] is used to obtain the solution for maximizing $SNR_{\sum}$, which leads to the optimal weights:

$$\hat{a}_i^t = \frac{s_i^t *}{\sum_{j=1}^{N} |s_j^t|} \quad \forall i \in 1, ..., N, \tag{14}$$

where the denominator is the scaling factor for the weights. Substitute to Eq. 13, we have the total SNR:

$$SNR_{\sum}^t = \frac{\sum_{i=1}^{N} |s_i^t|^2}{N_0^t} = \sum_{i=1}^{N} SNR_i^t. \tag{15}$$

If the Signal-to-Noise Ratios (SNRs) of all the branches are identical, this beamforming approach can achieve an overall SNR that is N times higher than that of a single branch. In fading channels, the combiner can exploit the diversity of the receiving branches to attain even more significant gains. The problem is to find the optimal weights $\hat{a}_i^t$. The polar representation of $\hat{a}_i^t$ is:

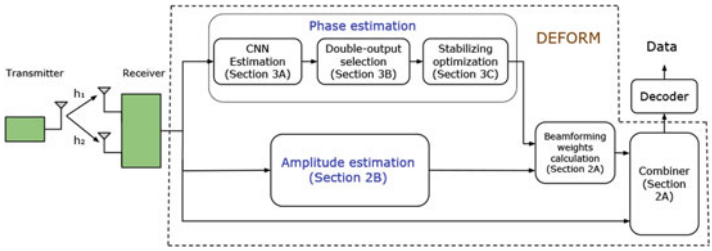


Fig. 10 The workflow of DEFORM system

$$\hat{a}_i^t = \frac{|s_i^t|}{\sum_{j=1}^{N} |s_j^t|} e^{-j\theta_i^t} \quad \forall i \in 1, ..., N. \tag{16}$$

Now, we need to estimate the amplitude $A_i^t = \frac{|s_i^t|}{\sum_{j=1}^{N} |s_j^t|}$ and the phase θ_i^t.

4.3 Estimating Beamforming Parameters

We tackle this problem by proposing the DEFORM system, with workflow depicted in Fig. 10. In our system, the received signals from all the branches are multiplied by the optimal beamforming weights obtained from the phase and amplitude estimations. The resulting signals are then combined to obtain the final output signal which is sent to the decoder to extract the data.

4.3.1 Amplitude Estimation

Out of N receiving branches, we pick an arbitrary branch k, and transform A_i^t for every branch i as follows:

$$A_i^t = \frac{|s_i^t|}{\sum_{j=1}^{N} |s_j^t|} = \frac{\frac{|s_i^t|}{|s_k^t|}}{\sum_{j=1}^{N} \frac{|s_j^t|}{|s_k^t|}}. \tag{17}$$

Then, instead of finding the exact A_i^t using explicit mechanisms, we use an *estimated amplitude* $\tilde{A}_i^t$ acquired using the received signals R_i^t:

$$\tilde{A}_i^t = \frac{\frac{|R_i^t|}{|R_k^t|}}{\sum_{j=1}^{N} \frac{|R_j^t|}{|R_k^t|}}. \tag{18}$$

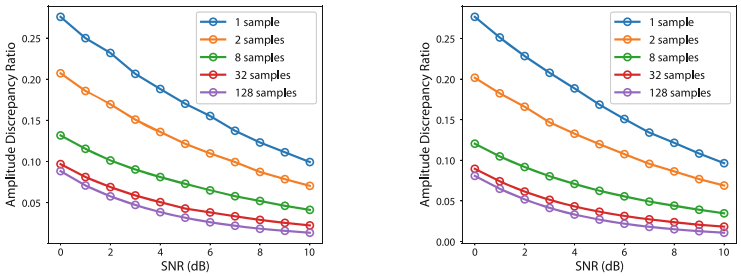


Fig. 11 The discrepancy between A_i and $\tilde{A}_i$ with respect to different SNRs, number of samples, and number of receiving antennas. (**a**) Two antennas. (**b**) Four antennas

We see that $\tilde{A}_i^t \approx A_i^t$ when $|s_i^t| \approx |s_k^t| \forall i \in 1, ..., N$. As $|s_i^t| \gg |s_k^t|$, the estimation error is larger. To minimize this error, we average $\tilde{A}_i$ over a number of continuous RF samples, instead of using the instantaneous value. The effectiveness of this approach is demonstrated in Fig. 11, which shows that averaging $\tilde{A}_i^t$ and A_i^t over 128 consecutive samples results in less than 5% discrepancy even at very low SNR (3 dB). This error is acceptable, as a high SNR is typically required to meet throughput requirements in many practical wireless systems (e.g., over 20dB for Wi-Fi [6]).

4.3.2 Phase Estimation

Instead of estimating the ***absolute*** signal phase θ_i^t, we estimate the ***relative*** signal phase $\Delta_{\theta_i}^t = \theta_i^t - \theta_k^t$ between the current branch i and a pre-selected arbitrary branch k, achieving the new weights:

$$\bar{a}_i^t = A_i^t e^{-j\Delta_{\theta_i}^t} \quad \forall i \in 1, \ldots, N. \tag{19}$$

Using these weights, the received signal in any branch i will be co-phased with the signal from pre-defined branch k and other branches and we can achieve the optimal gain at the combiner.

We design a fast and powerful Convolutional Neural Network [31] to estimate the relative phase. Moreover, our network is specially designed to address unique characteristics of complex RF samples with the following features:

Rotational Double-Output In principle, our neural network should provide one estimation for each relative phase between two receiving branches. However, while investigating various models, we found that the phase estimation experiences abrupt variations as the relative phase gets very close to the boundaries of phase values (i.e. upper and lower bounds of the respective ranges $[-\pi, \pi]$ or $[0, 2\pi]$). To be more specific, this behavior can be described as follows:

- If the phase shift is estimated within $[-\pi, \pi]$, the output estimation abruptly fluctuates when the true value is either in $[-\pi, -(\pi - \epsilon)]$ or $[\pi - \epsilon, \pi]$.
- If the phase shift is estimated within $[0, 2\pi]$, the output estimation abruptly fluctuates when the true value is either in $[0, \epsilon]$ or $[2\pi - \epsilon, 2\pi]$.

where we investigated and found that $\epsilon \approx 0.2\pi$. The abrupt fluctuations lead to high estimation errors in the aforementioned scenarios. This behavior is related to the discontinuity of the phase when it exceeds the boundaries, such as going above π and below $-\pi$ for $[-\pi, \pi]$. Because of the rotational characteristics (i.e., $2\pi + \theta = \theta \mod 2\pi$), the phase will be shifted backward by an angle of 2π. This creates confusion for the estimation model since these values are actually far apart from each other (by a distance of 2π) on the numerical axis.

To address this problem, we enhanced the CNN model with a new feature called the *rotational double-output* feature. This feature incorporates two estimation outputs E_1, E_2 for the phase values converted in the ranges $[-\pi, \pi]$ and $[0, 2\pi]$, respectively. Notably, these two outputs do not exhibit abrupt variations simultaneously. Hence, when one output provides an erroneous estimation, we can choose the other output for the current estimation. With this feature, our CNN is trained to minimize the Mean Square Errors of both outputs:

$$\mathcal{L}_\theta = (\Delta_{\theta_1} - E_1)^2 + (\Delta_{\theta_2} - E_2)^2, \tag{20}$$

where Δ_{θ_1} and Δ_{θ_2} are the respective ground truth converted in $[-\pi, \pi]$ and $[0, 2\pi]$.

Addressing Link Bit Error Rate Sensitivity Practical wireless communication systems and standards require not only a high precision, but also robust and stable estimations over time to maintain the target Bit-Error Rate in the orders of 10^{-4} for desirable throughput and proper communications. To achieve this, we propose two different techniques that aim to stabilize the phase estimations:

1. **Temporal Smoothing:** As RF channels typically change at a much slower rate compared to the incoming rate of RF samples, we can improve the stability of an instantaneous estimation by incorporating it with previous values. We implemented the exponential smoothing function, which helps to stabilize the estimation and enhance the robustness of the beamforming:

$$E^t = E_{cur}\lambda + E^{t-1}(1 - \lambda). \tag{21}$$

   where the phase estimation at time t is calculated using the previous estimation at time $t - 1$ and the current instantaneous output estimation E_{cur}. Parameter $\lambda = 0.2$ controls the smoothness of the result.
2. **Multi-Trial Averaging:** The *temporal smoothing* technique necessitates the use of multiple RF sample blocks to achieve a stable estimation. When the RF samples are limited but more computation power is available (e.g., offline processing), we use a multi-trial averaging technique where one block of samples is repeatedly used in multiple trials, adjusted by phase randomness to achieve

diverse results. To address possible drastic changes, we classify estimations of N trials into two clusters, based on the distance between the averaged estimation from each cluster and the estimation for the current trial. Upon completion of N trials, we compare the number of elements in each cluster and choose the averaged estimation from the larger cluster for the current period.

4.4 Evaluation

4.4.1 Simulation Results

First, we evaluated the performance of DEFORM in MATLAB simulations [29]. Monte-Carlo simulations were performed to assess DEFORM in both static RF channel (AWGN channel) and multi-path RF channel (Rayleigh channel), as shown in Fig. 12a. The results indicate that, in AWGN channel, DEFORM achieved optimal beamforming gain of 3 dB for all modulations, even though it was trained using a single modulation (8-PSK). In Rayleigh channel, DEFORM achieved significant beamforming gain, up to 5 dB compared to the single-antenna receiver, higher than the gain in the AWGN channel. The improved gain is due to the fact that in Reyleigh channel, two receiving antennas receive signals with different energy levels, providing diversity that DEFORM leverages to improve the beamforming gain.

4.4.2 Experimental Results

We evaluated DEFORM in two over-the-air experiments: When there is a Line of Sight (LOS) and when the LOS is blocked by various types of objects.

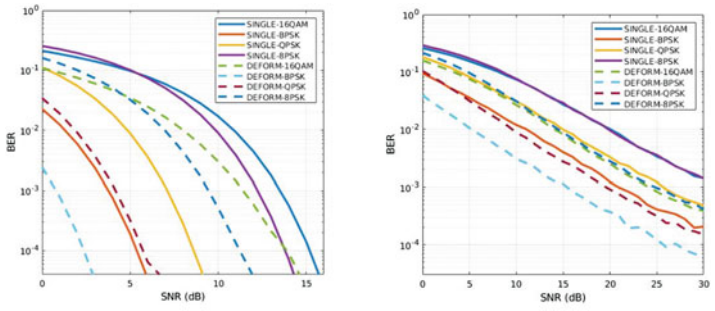


Fig. 12 BER comparison of DEFORM RF receiver and single-antenna receiver in simulation settings of AWGN and Rayleigh channel. (**a**) AWGN channel. (**b**) Rayleigh channel

Over-the-Air with LOS To establish the LOS, we set up the transmitter (TX) and receiver (RX) devices such that there were no obstructions in the direct path. Due to the varying SNRs among the RX branches in over-the-air communications, we monitored the SNR of the worst branch and adjusted the RX gain for the measurements. The results of BER evaluation are shown in Fig. 13. In most cases, DEFORM achieved a 2 dB gain compared to the better receiving branch. In some cases, the gain even approached 3 dB, such as with GMSK-6MHz with BER $= 10^{-2}$. When comparing with the worst branch, the gain can be as high as 4 dB, as in GMSK-1 MHz.

Over-the-Air with No LOS For the second experiment, we positioned the multi-antenna RX at the red cross location in our testbed floorplan (Fig. 14, with the presence of numerous large-sized objects, such as computers or lockers). Then,

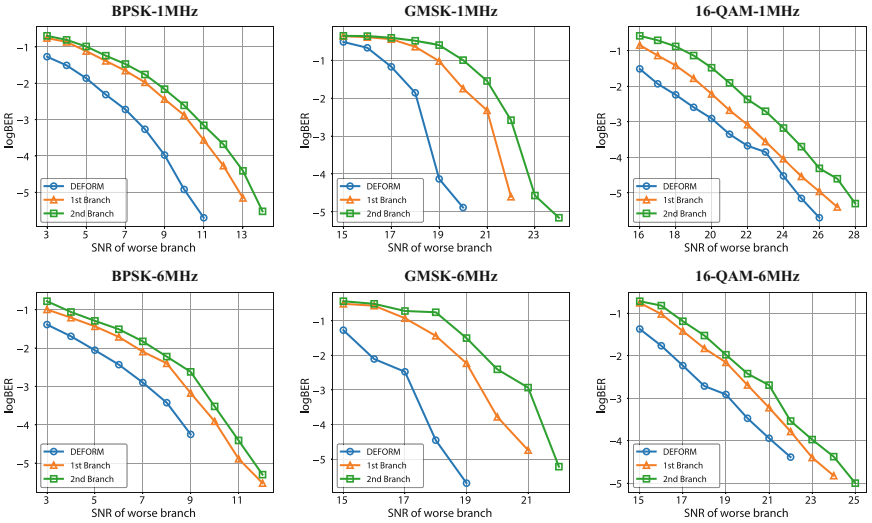


Fig. 13 BER results with regards to modulations and bandwidths in over-the-air experiment with LOS

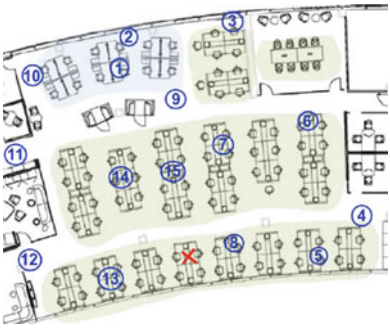


Fig. 14 Non-LOS over-the-air testbed in a 50 × 100 ft. office. The blue numbered circles mark the TX locations, the red cross marks the fixed RX location

we moved the TX to different locations marked by blue numbered circles. At each predefined TX location, the TX transmitted data packets with a randomly selected modulation, and the RX also randomly selected the RX bandwidth to receive the signal. We analyzed the BER and the results are presented in Fig. 15, where DEFORM consistently achieved lower BER than any single branch in all measurements.

4.5 Universal RF Beamforming-Relay

We demonstrate the universality of DEFORM with the beamforming-relay application for LoRa [48] and ZigBee [7], where direct communication link is disconnected with Packet Loss Rate (PLR) of 100%. Instead of sending the combined signal to the decoder as in the original workflow (Fig. 10), we relayed it to the TX chain. In our setup, the relay node was fixed while the TX/RX nodes were mobile within a small range around locations exhibiting a LOS to the relay node, as marked in Fig. 16a and pictured in Fig. 16b. The same CNN model trained on a basic RF setting was used for all experiments. We used Heltec ESP32 Development Kit for LoRa communications (using chirp spread spectrum modulation) and XBee-PRO 900HP equipped with the XBee Grove Development Boards for ZigBee communications.

Fig. 15 Bit Error Rate (BER) analysis for over-the-air experiments in non-LOS environment. The indices correspond to the numbered marks in Fig. 14. NaN implies a zero BER. Cross mark implies that no packets are detected by the decoder

	BER (log scale) for different TX positions							
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**
Branch 1	-0.43	-1.74	-0.76	-0.81	-1.79	-1.34	-1.11	-0.75
Branch 2	-0.83	-2.17	-1.26	-1.77	-1.28	-1.07	-4.59	×
DEFORM	-1.09	-2.7	-1.72	-2.79	-2.39	-1.84	-5.22	-1.43
	9	**10**	**11**	**12**	**13**	**14**	**15**	
Branch 1	NaN	-2.15	-3.63	-0.9	-1.15	-1.76	-3.12	
Branch 2	NaN	-5.3	-2.3	-6	-0.5	-1.95	-5.4	
DEFORM	NaN	-5.7	-3.89	NaN	-2.61	-2.12	-5.7	

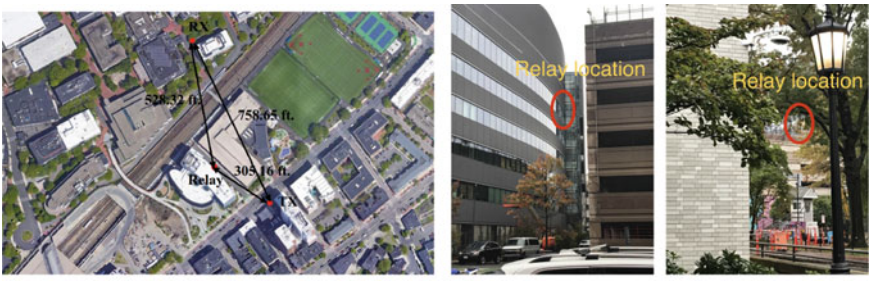


Fig. 16 Beamforming-relay experiment testbed. TX and RX were mobile within a small range from the marked spots. (**a**) Satellite view map. (**b**) Viewpoints from TX (left) and RX (right) of relay location

The beamforming-relay approach achieved a PLR of less than 10% in LoRa experiments, which is up to 12 and 23 times lower than the conventional Amplify-and-Forward relay approach using the stronger and weaker antennas, respectively. In ZigBee experiments, we achieved a successful packet reception rate as high as 193% of the stronger antenna relay and up to 858% of the weaker antenna relay.

5 Conclusion

Robust and secure communication is one of the most important requirements for wireless and mobiles systems. However, achieving it remains a significant challenge. In this chapter, we introduce three novel Deep Learning solutions to improve the robustness and security of wireless communications, which include (1) RF identification of emissions and collisions, (2) cancellation of adversarial interference, and (3) beamforming enhancement of received signal. We substantiate the effectiveness of our approaches throughout extensive evaluation in both simulation and real-life settings. Our aim is to lay the foundation for novel Deep Learning techniques and foster innovation in achieving robust and secure wireless communications.

Acknowledgments This work was partially supported by grants NAVY/N00014-20-1-2124, NCAE-Cyber Research Program, and NSF/DGE-1661532.

References

1. Berner C, Brockman G, Chan B, Cheung V, Debiak P, Dennison C, Farhi D, Fischer Q, Hashme S, Hesse C, et al (2019) Dota 2 with large scale deep reinforcement learning. arXiv:191206680
2. Bitar N, Muhammad S, Refai HH (2017) Wireless technology identification using deep convolutional neural networks. In: IEEE PIMRC
3. Bochkovskiy A, Wang CY, Liao HYM (2020) Yolov4: optimal speed and accuracy of object detection. arXiv:2004.10934
4. Cardoso J, Souloumiac A (1993) Blind beamforming for non-gaussian signals. IEE Proc F Radar Signal Process 140:362–370
5. Chollet F, et al. (2015) Keras. https://github.com/fchollet/keras
6. CISCO Meraki (2018) SNR and wireless signal strength. https://documentation.meraki.com/MR/WiFi_Basics_and_Best_Practices/Signal-to-Noise_Ratio_(SNR)_and_Wireless_Signal_Strength
7. Connectivity Standards Alliance (2021) Zigbee. https://zigbeealliance.org/solution/zigbee/
8. DARPA (2019) SC2 colosseum. https://www.sc2colosseum.com/
9. DeepSigai (2018) Deepsig datasets for modulation recoginition. https://www.deepsig.ai/datasets
10. Deng J, Dong W, Socher R, Li L, Kai Li, Fei-Fei L (2009) Imagenet: a large-scale hierarchical image database. In: IEEE conference on computer vision and pattern recognition
11. DJI Developer Technologies (2017) Airlink - DJI mobile SDK documentation: lightbridge. https://developer.dji.com/mobile-sdk/documentation/introduction/component-guide-airlink.html#lightbridge

12. Dobre OA, Bar-Ness Y, Wei Su (2003) Higher-order cyclic cumulants for high order modulation classification. In: IEEE military communications conference (MILCOM)
13. DroneShield (2019) ISIS dropping grenades from drones. https://www.droneshield.com/isis-dropping-grenades-from-drones
14. Elmaghbub A, Hamdaoui B (2021) Lora device fingerprinting in the wild: disclosing RF data-driven fingerprint sensitivity to deployment variability. IEEE Access 9:142893–142909
15. Fairwaves (2019) XTRX: the first ever truly embedded SDR. https://www.crowdsupply.com/fairwaves/xtrx
16. Girshick R, Donahue J, Darrell T, Malik J (2014) Rich feature hierarchies for accurate object detection and semantic segmentation. In: IEEE conference on computer vision and pattern recognition
17. Goldsmith A (2005) Wireless communications. Cambridge University Press
18. Google (2023) Cloud TPU. https://cloud.google.com/tpu/
19. Graves A, Mohamed A, Hinton G (2013) Speech recognition with deep recurrent neural networks. In: 2013 IEEE international conference on acoustics, speech and signal processing, pp 6645–6649. https://doi.org/10.1109/ICASSP.2013.6638947
20. Han G, Xiao L, Poor HV (2017) Two-dimensional anti-jamming communication based on deep reinforcement learning. In: 2017 IEEE international conference on acoustics, speech and signal processing (ICASSP), pp 2087–2091
21. Intel (2019) Nervana neural network processors. https://www.intel.ai/nervana-nnp/
22. Jin T, Noubir G, Thapa B (2009) Zero pre-shared secret key establishment in the presence of jammers. In: Proceedings of the tenth ACM international symposium on mobile ad hoc networking and computing, MobiHoc'09, pp 219–228
23. Kingma DP, Ba J (2017) Adam: a method for stochastic optimization. arXiv:1412.6980
24. Krizhevsky A, Sutskever I, Hinton GE (2012) Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems, vol 25, pp 1097–1105
25. Kwon HJ, Lee JH, Choi W (2019) Machine learning-based beamforming in two-user MISO interference channels. In: 2019 international conference on artificial intelligence in information and communication. IEEE, Piscataway, pp 496–499
26. Lazos L, Liu S, Krunz M (2009) Mitigating control-channel jamming attacks in multi-channel ad hoc networks. In: Proceedings of the second ACM conference on wireless network security, WiSec'09, pp 169–180
27. Liu Y, Ning P, Dai H, Liu A (2010) Randomized differential DSSS: jamming-resistant wireless broadcast communication. In: 2010 Proceedings IEEE INFOCOM, pp 1–9
28. Liu S, Xu Y, Chen X, Wang X, Wang M, Li W, Li Y, Xu Y (2019) Pattern-aware intelligent anti-jamming communication: a sequential deep reinforcement learning approach. IEEE Access 7:169204–169216
29. MATLAB (2022) version R2022b. The MathWorks, Natick
30. Merchant K, Revay S, Stantchev G, Nousain B (2018) Deep learning for RF device fingerprinting in cognitive communication networks. IEEE J Sel Top Signal Process 12(1):160–167
31. Nguyen HN, Noubir G (2022) Universal beamforming: a deep RFML approach. In: Proceedings of the 25th international ACM conference on modeling analysis and simulation of wireless and mobile systems, pp 165–172
32. Nguyen HN, Noubir G (2023) Jax: detecting and cancelling high-power jammers using convolutional neural network. In: Proceedings of the 16th ACM conference on security and privacy in wireless and mobile networks, WiSec'23
33. Nguyen D, Sahin C, Shishkin B, Kandasamy N, Dandekar KR (2014) A real-time and protocol-aware reactive jamming framework built on software-defined radios. In: Proceedings of the 2014 ACM workshop on software radio implementation forum, pp 15–22
34. Nguyen HN, Vo-Huu T, Vo-Huu T, Noubir G (2019) Towards adversarial and unintentional collisions detection using deep learning. In: Proceedings of the ACM workshop on wireless security and machine learning, pp 22–24

35. Nguyen HN, Vomvas M, Vo-Huu T, Noubir G (2021) Wideband, real-time spectro-temporal RF identification. In: Proceedings of the 19th ACM international symposium on mobility management and wireless access, pp 77–86
36. Nguyen HN, Vomvas M, Vo-Huu TD, Noubir G (2024) Wrist: wideband, real-time, spectro-temporal RF identification system using deep learning. IEEE Trans Mobile Comput 23:1550–1567
37. NVIDIA, Vingelmann P, Fitzek FH (2020) Cuda, release: 10.2.89. https://developer.nvidia.com/cuda-toolkit
38. O'Shea TJ, Corgan J, Clancy TC (2016) Convolutional radio modulation recognition networks. In: Engineering applications of neural networks. Springer, Berlin
39. Pöpper C, Strasser M, Capkun S (2009) Jamming-resistant broadcast communication without shared keys. In: USENIX security symposium, pp 231–248
40. Prasad KSV, D'souza KB, Bhargava VK (2020) A downscaled faster-RCNN framework for signal detection and time-frequency localization in wideband RF systems. IEEE Trans Wireless Commun 19(7):4847–4862
41. Project Zero (2017) Over the air: exploiting broadcom's wi-fi stack. https://www.crowdsupply.com/fairwaves/xtrx
42. Redmon J, Divvala S, Girshick R, Farhadi A (2016) You only look once: unified, real-time object detection. In: IEEE conference on computer vision and pattern recognition (CVPR)
43. Ren S, He K, Girshick R, Sun J (2015) Faster R-CNN: towards real-time object detection with region proposal networks. In: Advances in neural information processing systems (NIPS)
44. Samson Technologies (2019) XPD series. http://www.samsontech.com/samson/products/wireless-systems/xpd-series/
45. Sarkar S, Buddhikot M, Baset A, Kasera SK (2021) DeepRadar: a deep-learning-based environmental sensing capability sensor design for CBRS. In: Proceedings of the 27th annual international conference on mobile computing and networking, pp 56–68
46. Schmidt M, Block D, Meier U (2017) Wireless interference identification with convolutional neural networks. In: IEEE 15th international conference on industrial informatics (INDIN)
47. Schmitt E (2017) Pentagon tests lasers and nets to combat a vexing foe: ISIS drones. New York Times. https://www.nytimes.com/2017/09/23/world/middleeast/isis-drones-pentagon-experiments.html
48. Semtech (2021) What are lora and lorawan? https://lora-developers.semtech.com/documentation/tech-papers-and-guides/lora-and-lorawan/
49. Silver D, Huang A, Maddison CJ, Guez A, Sifre L, Van Den Driessche G, Schrittwieser J, Antonoglou I, Panneershelvam V, Lanctot M, et al. (2016) Mastering the game of go with deep neural networks and tree search. Nature 529(7587):484–489
50. Simonyan K, Zisserman A (2014) Very deep convolutional networks for large-scale image recognition. arXiv:1409.1556
51. Strasser M, Popper C, Capkun S, Cagalj M (2008) Jamming-resistant key establishment using uncoordinated frequency hopping. In: 2008 IEEE symposium on security and privacy (SP 2008), pp 64–78
52. Sutton PD, Nolan KE, Doyle LE (2008) Cyclostationary signatures in practical cognitive radio applications. IEEE J Sel Areas Commun 26:13–24
53. Vagollari A, Schram V, Wicke W, Hirschbeck M, Gerstacker W (2021) Joint detection and classification of RF signals using deep learning. In: 2021 IEEE 93rd vehicular technology conference (VTC2021-Spring). IEEE, Piscataway, pp 1–7
54. Venkateswaran V, van der Veen AJ (2010) Analog beamforming in MIMO communications with phase shift networks and online channel estimation. IEEE Trans Signal Process 58(8):4131–4143
55. Vo-Huu TD, Blass EO, Noubir G (2013) Counter-jamming using mixed mechanical and software interference cancellation. In: Proceedings of the sixth ACM conference on security and privacy in wireless and mobile networks, WiSec'13, pp 31–42

56. Wu Q, Zhang R (2019) Intelligent reflecting surface enhanced wireless network via joint active and passive beamforming. IEEE Trans Wireless Commun 18(11):5394–5409
57. Wu Y, Schuster M, Chen Z, Le QV, Norouzi M, Macherey W, Krikun M, Cao Y, Gao Q, Macherey K, et al (2016) Google's neural machine translation system: bridging the gap between human and machine translation. arXiv:160908144
58. Xu W, Trappe W, Zhang Y, Wood T (2005) The feasibility of launching and detecting jamming attacks in wireless networks. In: Proceedings of the 6th ACM international symposium on mobile ad hoc networking and computing, pp 46–57
59. Yan Q, Zeng H, Jiang T, Li M, Lou W, Hou YT (2014) Mimo-based jamming resilient communication in wireless networks. In: IEEE INFOCOM 2014 - IEEE conference on computer communications, pp 2697–2706
60. Ye H, Li GY, Juang BH (2018) Power of deep learning for channel estimation and signal detection in OFDM systems. IEEE Wireless Commun Lett 7(1):114–117
61. Zeng H, Cao C, Li H, Yan Q (2017) Enabling jamming-resistant communications in wireless MIMO networks. In: 2017 IEEE conference on communications and network security (CNS). IEEE, Piscataway, pp 1–9
62. Zhang L, Liao B, Huang L, Guo C (2017) An eigendecomposition-based approach to blind beamforming in a multipath environment. IEEE Commun Lett 21(2):322–325

Universal Targeted Adversarial Attacks Against mmWave-Based Human Activity Recognition

Yucheng Xie, Xiaonan Guo, Yan Wang, Jerry Cheng, and Yingying Chen

1 Introduction

Millimeter Wave (mmWave) technology is one of the promising communication technologies due to its high throughput and wide bandwidth. Recent studies have shown the initial success of using mmWave in the domain of sensing applications, including Human Activity Recognition (HAR). Human activity recognition (HAR) has attracted significant attention since it is an essential technology to enable human-computer interactions in many Internet of Things (IoT) and security applications, including health monitoring and user authentication. Many HAR systems have been designed using various sensing modalities. Traditional camera-based [11, 16] and sensor-based [5, 42] HAR systems capture human activities using video cameras and body sensors, respectively. They usually intrigue privacy concerns or are not convenient. To circumvent these limitations, low-cost, non-intrusive solutions like

Y. Xie
Indiana University-Purdue University Indianapolis, Indianapolis, IN, USA
e-mail: yx11@iupui.edu

X. Guo (✉)
George Mason University, Fairfax, VA, USA
e-mail: xguo8@gmu.edu

Y. Wang (✉)
Temple University, Philadelphia, PA, USA
e-mail: y.wang@temple.edu

J. Cheng
New York Institute of Technology, Old Westbury, NY, USA
e-mail: jcheng18@nyit.edu

Y. Chen (✉)
Rutgers University, New Brunswick, NJ, USA
e-mail: yingche@scarletmail.rutgers.edu

Y. Chen et al. (eds.), *Network Security Empowered by Artificial Intelligence*,
Advances in Information Security 107, https://doi.org/10.1007/978-3-031-53510-9_7

radio frequency-based techniques are being researched. Wireless signals have been commonly used in communication due to their convenience, flexibility, and ability to transmit information over long distances without the need for physical connections. The prevalence of wireless signals in our everyday devices allows for a complex network of reflected rays in indoor environments. Researchers find that the human presence and motion significantly impact these signals, enabling the capture of human body movements involved in daily activities. Recently, wireless signals (e.g., WiFi [19, 73], sound [31, 62], mmWave [39, 63]) have been utilized to track human activities without attaching sensors to the human body. In this direction, mmWave-based HAR systems stand out because they can provide high resolution with their short wavelength and large bandwidths.

Most mmWave-based HAR systems adopt deep learning models for activity identification due to their high accuracy and strong capability of handling interference in the real world. However, recent research has revealed that deep learning models are susceptible to adversarial inputs [68]. Some researchers have designed minor perturbations that cause deep learning networks to make inaccurate predictions in image classification [44] and voice recognition [53]. Nevertheless, few studies have investigated the susceptibility of adversarial targeted attacks in mmWave-based HAR systems. Because mmWave-based HAR systems are usually integrated into many crucial applications such as older patient monitoring and user authentication [6, 91] as shown in Fig. 1, we believe that studying adversarial attacks on these systems is critical and urgent. Most recently, Ozbulak et al. [48] have done an initial investigation with the untargeted adversarial attack on mmWave-based HAR. The designed attack is only applicable to a particular HAR model (i.e., heatmap-based) and cannot trigger the model to generate designated classes. Moreover, many research problems, such as how to design unnoticeable perturbations based on unique patterns of mmWave signals [66], how to launch universal target adversarial attacks [32], or more challenging black-box attacks [34], are still worth further

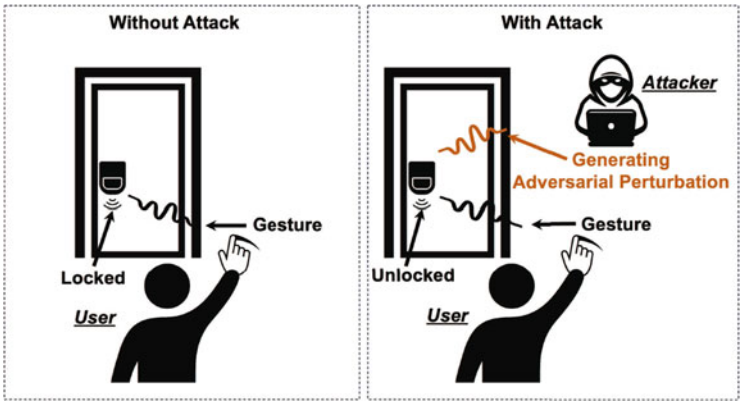


Fig. 1 Illustration of adversarial attacks on gesture-based user authentication system

exploration. Therefore, a more comprehensive study of systematically exploring different types of adversarial attacks on different types of mmWave-based HAR models is highly demanded.

In this chapter, we aim to systematically investigate and reveal the severe security issues of mmWave-based HAR models by developing the following effective adversarial attacks: *(1) White-box Attack.* Unlike existing work that only studied the untargeted attack for a particular mmWave-base HAR model, we successfully design both targeted and untargeted attacks for different mmWave-based HAR models in the white-box scenario. Moreover, because both targeted and untargeted attacks need to train a unique adversarial perturbation for each activity sample [87], which is inefficient and infeasible in time-constrained scenarios, we design a universal adversarial attack that can produce an adversarial perturbation applicable to different activity samples and ready to be used in real-time without additional training; *(2) Black-box Attack.* Besides white-box attacks that assume the attackers have full knowledge of the victim model, we further explore more challenging black-box attacks where attackers may not have sufficient information about the victim system and need to conduct attacks under more realistic conditions (e.g., the victim model is unavailable to the attacker). Black-box attacks are a more probable form of threat in real-world applications. By examining black-box attacks, we hope to better prepare these HAR systems for actual threats. Therefore, we develop an effective method to enable black-box targeted attacks in this chapter. Exploring both white-box and black-box adversarial attacks permits us to comprehensively and practically evaluate and improve the security of mmWave-based HAR.

Designing effective and practical adversarial attacks for different mmWave-based HAR models is nontrivial. Different from traditional replay attacks [51], our attack could fool the HAR system without collecting data samples from the target activity. In particular, we apply gradient-based machine learning algorithms to generate adversarial perturbations for targeted and untargeted attacks while minimizing their size. The adversarial perturbation is generated by solving an optimization problem to concurrently minimize the perturbation loss, which constrains the perturbation size and adversarial loss to ensure the success of the adversarial attacks without being noticed. In addition, mmWave-based HAR systems may use different data representations that require careful attention. Our comprehensive study identifies two representative types of mmWave-based HAR models (i.e., voxel-based and heatmap-based). We design a discretization method to ensure the validity of adversarial samples and further optimize the form of the adversarial samples with two distance metrics. The main challenge in designing the universal adversarial attack is deriving an effective adversarial perturbation for any activity sample without online training. We implement an offline training strategy with an iteration algorithm that crafts universal perturbation across the samples from a small pre-collected activity set. Unlike the existing universal attack that needs inserting padding frames between two successive activities [48], our attack modifies the activity sample directly, which enables the attack on a broader range of mmWave-based HAR applications. Furthermore, to overcome the information deficiency of the victim model in black-box attacks, we utilize a knowledge distillation (KD) approach to generate a robust

replacement model. We further develop a generative adversarial network (GAN) to produce a sufficiently large number of pseudo samples for substitute model construction.

In summary, we explore the security issues of AI-enpowered mmWave-based HAR systems as deep learning technologies have been extensively used to help these systems to achieve more accurate and convenient recognition. In particular, we implement a comprehensive assessment of the challenges brought by adversarial attacks on various mmWave-based HAR systems, including both white-box and black-box adversarial attacks. For white-box attacks, we employ adversarial learning to reduce the magnitude of the perturbation, ensuring that the generated perturbation is undetectable by manual examinations while can successfully attack mmWave-based HAR systems. We also develop a discretization method to enable adversarial attacks on different representative models of mmWave-base HAR. To enable universal targeted attacks, we develop an iteration method to construct well-designed universal perturbations that can be applied to various unseen mmWave samples directly without additional training for these samples. We further design a black-box attack that can attack mmWave HAR systems without knowing the model architecture and parameters. In particular, we leverage knowledge distillation to address the information deficiency of the victim model. We also develop a generative adversarial network to address the lack of training data. We assess our implemented attack methods on two representative mmWave-based HAR models and demonstrate the efficacy, efficiency, and practicality of the attacks.

The remainder of this chapter is organized as follows. Section 2 discusses the related work in the field of HAR and adversarial attacks. Section 3 provides background information on sensing using wireless signals, human activity recognition, and adversarial attacks. Section 4 summarizes the victim machine learning models for HAR. Section 5 presents the threat model of our attack. Section 6 describes the developed white-box adversarial attack design and black-box attack design. Section 7 presents the experimental results of the attacks on two different mmWave-based HAR models. Finally, Sect. 8 concludes this chapter and provides directions for future research.

2 Related Work

Because of its wide application, HAR has attracted great attention for the past decade. In general, human activity recognition systems can be classified into three categories: camera-based [28], sensor-based [4], and radio-frequency (RF) signal-based [19, 71]. A couple of camera-based systems have been implemented to recognize human activities [16, 28]. These works use cameras to capture images or videos and apply image-processing algorithms to extract motions. However, camera-based methods may raise privacy concerns. To address this weakness,

sensor-based systems have been developed [4, 20]. These works explore various dedicated sensors such as gyroscopes [20], ECG or FSR sensors [4] to collect different types of signals for further analysis. However, sensor-based approaches require users to wear sensors or other devices, which is inconvenient for senior people or during complex activities. To overcome the above limitations, researchers recently developed RF-based methods (e.g., WiFi and mmWave). WiFi-based approaches [19, 72, 75, 89] use off-the-shelf WiFi devices to infer human activities. However, being easily influenced by surrounding environments remains the main limitation. Compared with WiFi signals, mmWave has been proven to be robust for activity recognition due to the antenna's directionality and stability. Some researchers design HAR systems based on mmWave [6, 25, 39, 50, 55, 57, 58, 71, 77–79, 84].

Most mmWave-based HAR systems adopt deep learning models for activity identification due to their high performance and capability of handling real-life interference. However, machine learning models such as neural networks were susceptible to adversarial perturbations, as pointed out by Szegedy et al. [68]. We discover that the majority of current adversarial attacks are proven in applications related to image recognition and speech authentication [10, 14, 32, 43, 44, 81]. Recently, there has been some work discussing adversarial attacks on radar-based systems. Yang et al. [87] examine the adversarial susceptibility of the Doppler-based HAR system. They analyze the untargeted attack issues for the HAR system and evaluated three white-box attack methods (i.e., FGSM, PGD, and MIM), respectively. Then, Ozbulak et al. [48] examine the vulnerability of radar-based HAR systems to a universal untargeted attack. Nevertheless, none of them explore the feasibility of targeted adversarial attacks to control the HAR system's output, nor do they provide a comprehensive study of adversarial attacks against mmWave-based HAR systems. Moreover, since Ozbulak's method only targets one heatmap-based HAR model, how to launch a universal targeted attack on other types of mmWave-based HAR models is unknown. Besides, based on unique patterns of mmWave activity data, how to develop adversarial activity samples to assure their validity and make them unnoticeable is necessary but seldom explored. In addition, how to enhance attack performance in more challenging black-box scenarios is still an open problem.

In contrast to previous research, we implement a comprehensive study of the threats brought by adversarial attacks, including both untargeted and targeted attacks. We broaden our study on both heatmap-based and voxel-based mmWave-based HAR systems. By optimizing perturbation based on the unique patterns of mmWave activity data, inventing universal attacks to make our attack approach more efficient, and examining the robustness of attacks under black-box scenarios, we intend to give a complete examination of the challenges posed by adversarial attacks on mmWave-based HAR systems.

3 Background

In this section, we present essential background information on three key topics: sensing using wireless signals, human activity recognition, and adversarial attacks.

3.1 Sensing Using Wireless Signals

Wireless signals have become widely adopted for communication due to their convenience, adaptability, and capacity to transmit data over long distances without the need for physical connections. The prevalence of wireless signals in our everyday electronic devices enables complex networks of reflected rays in indoor environments, making it possible to track human motions by examining the received signals. The fundamental premise of sensing using wireless signals is that human movement affects wireless signal propagation. This breakthrough has led to various applications, including human activity recognition [35, 36, 82, 83, 86], human localization [37, 59, 75] and vital sign monitoring [23, 33]. Compared with traditional camera-based methods, which may raise privacy concerns or sensor-based approaches that require users to wear sensors or other devices, wireless-signal-based methods are more convenient, especially for seniors or during complex scenarios. Among the prominent approaches are RFID sensing, WiFi sensing, and mmWave sensing. These techniques offer distinct advantages in terms of operating frequencies, detection capabilities, and potential applications.

The rise of RFID sensing in smart living spaces is due to its hands-free detection abilities with operating frequencies from 125 kHz to 1 GHz. RFID detection considers the changes in waveforms emitted by a transmission antenna. Compared to echo-based detection, RFID transponders can handle higher frequencies, allowing for more extensive range detection. Unique identification is possible through micro-Doppler signatures generated when objects interact with RFID signals. RFID systems, either active or passive, require no line-of-sight and have long lifespans. Thus, they are commonly used in inventory management in warehouses [9, 13, 22, 23, 47, 56].

Owing to the ubiquity of WiFi infrastructure, WiFi detection has become a popular device-free sensing technique. Its Received Signal Strength (RSS) can provide environmental and bodily state information similar to echo-based detection. The RSS, which measures path loss in Decibels (dB), can indicate the presence of people and their activities through its sensitivity to environmental changes. By placing two WiFi devices across a space and observing signal changes, we can effectively track health aspects like breathing and heart rates, and indoor positioning [1, 26, 85]. However, the coarse nature of RSS data, coupled with unstable WiFi signal strength, raises concerns about its reliability in applications. For a more precise use of WiFi signals, Channel State Information (CSI) is developed. CSI measures path loss and also incorporates multipath effects like scattering and

fading. Its higher sensitivity enables better detection of human movements, even unconscious ones like breathing [27, 38, 41, 52, 72, 74]. CSI provides a more detailed set of values, including amplitude and phase information for orthogonal frequency division multiplexing (OFDM) subcarriers, which can capture different frequency ranges and thus provide more fine-grained wireless channel details than RSS. Hence, healthcare applications are increasingly adopting CSI processing using common WiFi devices.

Millimeter wave (mmWave) detection operates at high frequencies, between tens and hundreds of GHz, offering broader bandwidth due to less congestion from commercial technologies such as TV or radio broadcasts. It's utilized in next-generation WiFi protocols like WiGig or 60 GHz WiFi [88]. As mmWave signals can effectively penetrate materials like plastic, drywall, and clothing, they are sensitive to minor changes like vital signs and are useful for high-resolution monitoring, making them ideal for device-free human activity recognition [60]. Studies [30, 63, 76, 90] have demonstrated its potential in recognizing various human activities, even with sparse data. Commercial radar hardware can be used for human activity recognition, dynamic skeleton pose tracking, and even differentiating between a person's motion and static states, including specific activities during static states such as making a call, using an app, or keeping the phone in the pocket.

3.2 Adversarial Attack

As Deep Learning (DL) becomes more prevalent, it is anticipated that potential threats will emerge, particularly against DL models. Adversarial attack, also known as evasion attacks, occurs when an adversary makes minute modification to the input of a neural network in order to cause an error in the inference process [17, 43]. These modifications are not noise samples generated at random; rather, they have been purposefully engineered to form a vector in the input feature space that can trick the DL model. Typical examples of these kinds of attacks entail either solving a limited optimization problem in order to construct a deceptive vector in the input feature space or inserting a small value in the model's gradient direction relative to the inputs. Both of these methods involve manipulating the output of the model. These attacks are typically stealthier and more energy-efficient than traditional jamming attacks [45] as they only need to make a small modification over a short period of time to confuse the DL models in their decision-making.

In the disciplines of Computer Vision (CV) and Natural Language Processing (NLP), adversarial attacks have been the subject of much research [10, 14, 32, 43, 44, 81]. One of the most well-known examples is when a machine learning classifier was fooled into thinking that an image of a panda was actually of a gibbon by adding a perturbation to the panda image that was deliberately created to trick the classifier [17]. Similarly, adversarial attacks in the field of wireless sensing can lead to the incorrect classification of human activities. Despite this, we have found that very few studies have looked into the possibility of adversarial targeted attacks occurring

in mmWave-based HAR systems. This motivates us to systematically study and uncover the significant security vulnerabilities of mmWave-based HAR models by building effective adversarial attacks.

3.3 Human Activity Recognition

The field of Human Activity Recognition (HAR) is rapidly growing, benefiting from advancements in sensors, cost efficiency, energy optimization, real-time data computation, machine learning, computer vision, AI, and IoT technologies. Activity classifications span diverse domains such as locomotion [18, 29], transportation [12, 46], mobile phone use [15], entertainment [80], health [24, 65], gestures [2, 75], and security [54]. HAR's criticality is emphasized by its application across health monitoring, fitness tracking, home automation, augmented reality, traffic management, targeted advertising, and security. For instance, individual activity logs can inform tailored dietary advice by estimating calorie consumption, and fall detection in seniors can trigger immediate emergency response, mitigating severe accidents.

Conventionally, machine learning methods have been employed to recognize human activity. However, these conventional methods for HAR necessitate the creation and selection of pertinent features, a process requiring significant human effort and expert knowledge. Additionally, the performance of these features might not always be optimal. To mitigate the need for hand-engineering features, deep learning techniques have been introduced in recent years [67]. These techniques offer several advantages for HAR. Firstly, they eliminate the need for manually designing features, which usually require specialist knowledge. Secondly, they have demonstrated higher accuracy in HAR than traditional methods [92]. Thirdly, they can learn from unlabeled data, which is a significant advantage for HAR due to the impracticality of gathering substantial volumes of labeled activity data. Lastly, deep learning techniques can learn useful features from raw data and can handle activity-related data from a range of sources, including different individuals, device models, and device orientations.

A typical deep learning-based HAR system has four main components as shown in Fig. 2. The first component involves data collection from various sensors that could capture images, WiFi CSI, accelerations, gyroscope readings, barometer readings, sound, biosensor readings, etc. The second component involves preprocessing the collected data using techniques like scaling, Principal Component Analysis (PCA) whitening, Zero-phase Component Analysis (ZCA) whitening, or denoising. The third component is the model-building stage, where diverse deep models (e.g., RBM, autoencoder, RNN) can be utilized to learn relevant features. This is followed by the application of a classifier (like softmax classifier, SVM) at the top layer. Once the model is established, it can be trained using the input data. During this training phase, network parameters such as weights are optimized. Finally, the trained model can be employed to predict the activity based on incoming data.

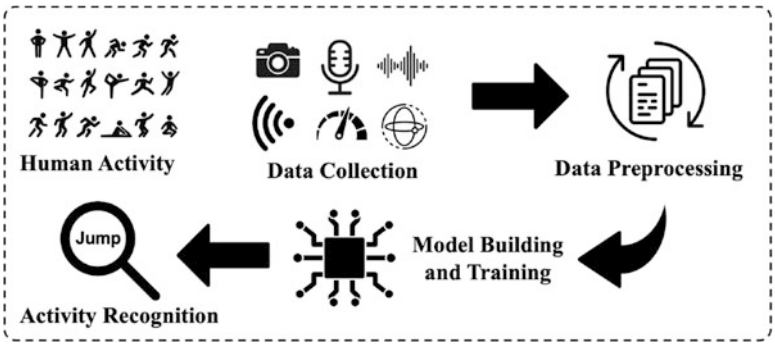


Fig. 2 A typical flow of human activity recognition system

4 Victim Machine Learning Models

The main goal of the mmWave-based human activity recognition system is to identify actions or gestures by examining the dynamics of mmWave signals [6, 39, 57]. As shown in Fig. 2, a typical mmWave-based HAR system captures mmWave signals reflected from the human body via the mmWave sensor. It performs signal processing to determine activity characteristics (e.g., velocity or posture) of users and then estimates the activity class using deep learning models. Deep learning model-empowered HAR systems are vulnerable to adversarial attacks. Existing mmWave-based HAR systems can be categorized into two classes based on the representations of the received mmWave signals. One of the representations is the point cloud derived from the received mmWave signals via a series of FFT operations (i.e., Range-FFT, Doppler-FFT, and Angle-FFT). Each point in the point clouds presents the x, y, and z coordinates of a mmWave signal reflected from the human body [6, 63, 78, 84, 91], which allows mmWave radars to generate a rough contour of the human body. However, point clouds are incompatible with neural network architecture as the number of points varies over time. Prior mmWave-based HAR research usually adopts voxelization to transform the point cloud into a constant amount of voxels [6, 63] for HAR. The other representation is the heatmap of the object-related information (e.g., distance, velocity, angle, and energy) extracted from the received mmWave signals. Many mmWave-based HAR systems have leveraged the heatmap to identify human activities (e.g., doppler-range map [39], micro-Doppler map [77], spatial spectrograms [57], spatial feature map [61], and projection heatmap [58]) because it is easy to achieve a good accuracy by applying pre-trained neural network models from the image domain to mmWave-base sensing.

In this chapter, we investigate the attacks on two typical mmWave-based HAR models using different types of representations, which brings more challenges to design a generic attack method because of their significant differences.

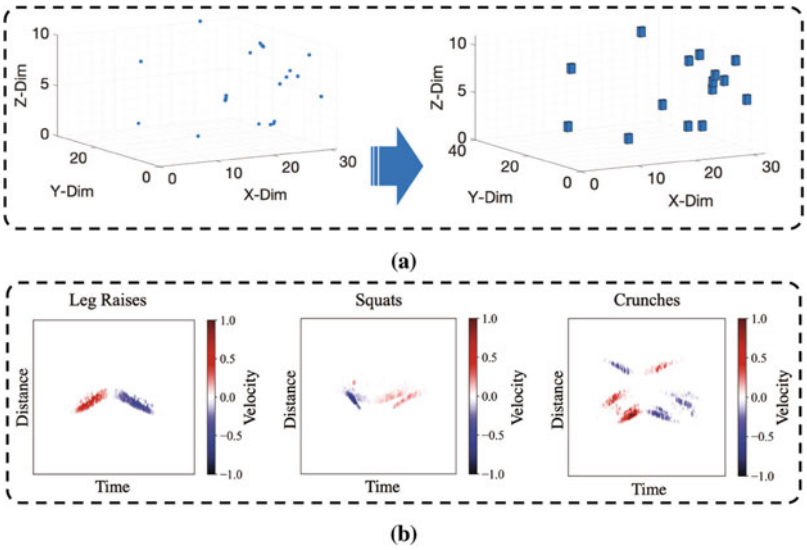


Fig. 3 Two typical data representations for mmWave-based HAR. (**a**) Voxels generation from the point cloud; (**b**) Spatial-Temporal heatmaps of three different activities

Voxel-Based Machine Learning Model We choose an existing mmWave-based HAR system [63] as a representative to study the vulnerability of voxel-based HAR models to adversarial attacks. This model has been utilized as a benchmark in numerous subsequent publications [3, 49]. In particular, the point clouds data is subjected to voxelization to address the non-uniformity issue in each frame, as shown in Fig. 3a. After the voxelization, the point clouds of each frame is transformed into a set of voxels in a three-dimensional space. A voxel is defined as $[x, y, z, v]$, where x, y, z are the spatial position of the voxel and v is the number of cloud points in the cube-shaped voxel with a designated size. Each activity sample is defined as t sets of voxels, where t is the time dimension. As for the machine learning model, they employ a Time-distributed CNN plus Bi-directional LSTM model. This model consists of 3 time-distributed convolutional layers followed by a bidirectional LSTM layer and an output layer. This model is directly trained on the input sample, which includes its temporal and spatial dimensions.

Heatmap-Based Machine Learning Model In addition to the voxel-based mmWave-based HAR system, we devise a heatmap-based HAR system to study its vulnerability to adversarial attacks. Similar to state-of-the-art mmWave-based HAR methods (e.g., [39]), we first derive the Doppler-range map of the users' activity by calculating Range-FFT and Doppler-FFT. Then, we generate heatmaps by accumulating the velocity of every distance in every denoised Doppler-range map together. Next, we normalize the derived velocity information and present the velocity-distance relationship in the time dimension. In this way, we transfer the

original instantaneous velocity-distance relationship to a more comprehensive spatial-temporal heatmap, which describes the process of a whole activity as shown in Fig. 3b. We utilize a CNN model for activity classification. In particular, this model consists of 3 convolutional layers, each followed by a max-pooling layer. A 64-dimensional feature map is created after 3 rounds of upsampling and downsampling. The feature map is then condensed into a one-dimensional array by integrating a flattened layer.

5 Threat Model

Adversarial attacks can be categorized into two categories: white-box and black-box. In order to investigate adversarial attacks against mmWave-based HAR, we first adopt the white-box assumption used by most of the previous research [7, 32, 87]. After that, we explore a more difficult black-box attack because black-box scenarios are more likely to occur in real-world applications. Exploring both white-box and black-box adversarial attacks allows us to evaluate and improve the security of mmWave-based HAR comprehensively and practically.

5.1 White-Box Attack

In the white-box scenario, the attackers have full knowledge of the machine learning model's input, architecture, and parameters. The adversary may also continuously access the victim model to produce adversarial samples. In addition, the adversary may be familiar with the HAR system's data preprocessing techniques in order to provide the proper perturbation. Based on that, white-box attacks are applicable to internal threats: they happen when someone within the organization with comprehensive knowledge of the system architecture and access to sensitive information acts maliciously. For example, attackers might modify the adversarial samples during the data preprocessing stage. The possibility of a white-box attack may be increased by a local adversary or information leakage.

The goal of this adversarial attack is to generate mmWave adversarial samples to confuse the mmWave-based HAR system. The mmWave-based HAR system can be conceptualized as a function f that receives mmWave signals as input and outputs the predicted activity class based on the probability score p for all the enrolled activity classes. Specifically, suppose there are n enrolled activities, where $p_i \in [0, 1]$ and $\sum_{i=1}^{n} p_i = 1$, the deep learning model f identifies the mmWave input as the class with the greatest probability score. In white-box scenarios, we study three adversarial attacks and formalize these attacks as follows:

Untargeted Attack In untargeted attacks, which is usually designed for a specific sample (sample-specific untargeted attack), the adversary aims to confuse the HAR

system by changing the output from the original activity prediction y to a different one y'. Specifically, given a machine learning model f and an activity sample x, a sample-specific untargeted attack can be formulated as $f(x + \delta) \neq y$, where x is the original activity sample, δ is the generated perturbation, and y is the original predicted activity of the classifier model. In order to achieve this, we should modify the activity sample by inserting δ to decrease the probability score of the original activity class p_y till it is lower than other activities.

Targeted Attack In targeted attacks for a specific sample (sample-specific targeted attack), the adversary aims to make the HAR systems output the desired class. The targeted attack can be formulated as $f(x + \delta) = z$, where $x + \delta$ is the adversarial sample and z is a pre-defined class. To enable this objective, we should modify the activity sample to increase the probability score of the desired activity p_z till it is higher than other enrolled activities.

Universal Attack To further improve the efficiency of targeted attacks and make it practical in time-constrained contexts, we develop universal attacks by generating a well-designed general perturbation. Then, we can insert it to different unseen activity samples directly without incurring additional training efforts. In particular, the activity data samples gathered at various times or under different conditions would often vary. Thus, the perturbation δ_1 designed to attack sample x_1 might not work for another sample x_2, such as $f(x_2 + \delta_1) \neq z$. In addition, generating the perturbation for a high-dimensional mmWave activity sample (e.g., voxel-based data) is time-consuming, thus it is not always feasible to produce a sample-specific perturbation that is tailored for each activity sample. It is important to create some universal perturbations δ, such that $f(x_i + \delta) = z$, where x_i can be different samples from the same type of activity.

5.2 Black-Box Attack

Black-box is a more challenging scenario. It assumes that the target machine learning model is unavailable to the attacker. The adversary only knows the input and output of the model [34]. Thus, black-box attacks are usually launched by external actors who do not have prior knowledge of the system's inner workings. In black-box scenarios, it is possible for attackers to insert the adversarial sample right before the recognition phase, where activity data are sent as input to the machine learning model. In this case, the attackers can generate adversarial samples in advance and feed them into the machine learning models furtively. In addition, attacks can seize the original activity samples during the transmission from a local client to the server and then replace normal samples with adversarial ones due to the widespread usage of cloud computing and federated learning [40, 64]. We investigate our adversarial attacks on HAR in black-box settings, because they are more realistic than white-box settings. In black-box scenarios, we would explore whether a perturbation generated based on model f could still work on another

model f', where f' has different structures and parameters from f. This attack can be formulated as $f'(x') = f(x')$, where x' is the adversarial sample generated based on model f.

In order to create a practical adversarial sample that is difficult for humans to identify, the distortion caused by the perturbation should be as minimal as possible. It can be formulated as $\min \|\delta\|_p$, s.t. $f(x + \delta) = z$. Additionally, it brings additional difficulties to produce reliable and undetectable adversarial perturbations due to the unique characteristics of mmWave signal representations (e.g., voxel-based data). Since we should consider both the effectiveness of our attack and the distortion of the perturbations, we formally define the objective function as minimize $\mathcal{D}(x, x + \delta)$, such that $f(x + \delta) = z$. D is the distance metrics $\|\delta\|_p$ that evaluates the magnitude of the generated perturbation. However, as discussed in previous work [10], directly solving this non-linear constrained non-convex problem is difficult. Thus, we reformulate the objective function as a gradient-based optimization instance:

$$\text{minimize } \mathcal{L}(x + \delta) + \lambda * \mathcal{D}(x, x + \delta), \tag{1}$$

where the first component $\mathcal{L}$ represent the adversarial loss which measures the possibility of launching adversarial attacks successfully and the second component $\mathcal{D}$ represents perturbation loss which constraints the perturbation size.

6 Attack Design

In this section, we explore algorithms for adversarial attacks, delineating both white-box and black-box strategies. Figure 4 graphically demonstrates the progression of our white-box methodology. To generate adversarial mmWave activity samples, we initially select loss functions based upon varying attack strength requirements, such as untargeted attacks or targeted attacks for mmWave-based human activity recognition systems. Enhancing the validity and efficacy of the adversarial mmWave samples demands the adoption of clipping and discretization processes, paired

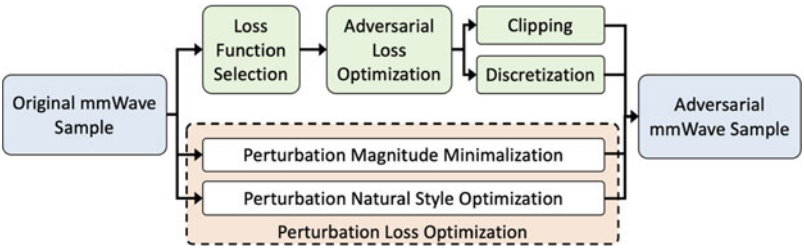


Fig. 4 The flow and components of the designed adversarial mmWave activity sample generation in white-box attack

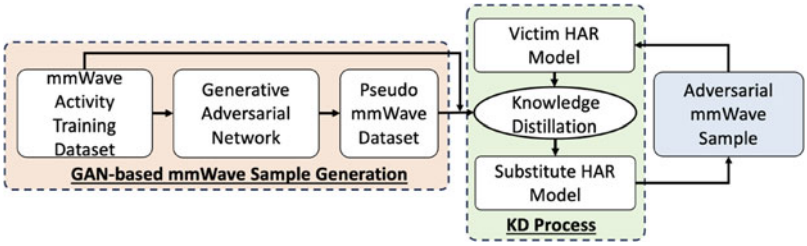


Fig. 5 The flow and components of the designed adversarial mmWave activity sample generation in black-box attack

with adversarial loss optimizations. We focus not only on minimizing the perturbation magnitude but also on examining the unique characteristics of mmWave activity data, thereby defining two distance metrics to foster more unnoticeable adversarial samples. Additionally, we contrive a practical and efficient universal targeted attack method, shaping a general perturbation during the offline training phase. This perturbation readily adapts to runtime mmWave activity samples for a targeted attack. Moreover, we delve into black-box attacks, proposing various methods to evaluate the feasibility of launching adversarial attacks under different circumstances. Figure 5 outlines the implemented black-box attack procedure. To facilitate the black-box attack, we first devise a knowledge distillation method for the creation of the substitute model. The adversarial mmWave activity sample is subsequently generated based on this substitute model. In more challenging cases where the available training data for creating this substitute model is inadequate, we use a GAN-based method to produce an abundant number of high-quality mmWave data, thus compensating for the insufficient mmWave training dataset.

6.1 White-Box Attack Implementation

6.1.1 Targeted and Untargeted Attack

Adversarial Loss We first describe the implementation of our method for specific mmWave activity sample. For sample-specific untargeted attacks, we define the objective function as $\mathcal{L} = \max\left(\mathcal{Z}(x+\delta)_s - \max_{i\neq s}\left(\mathcal{Z}(x+\delta)_i\right), -k\right)$, where $\mathcal{Z}(x+\delta)_s$ represent the possibility of estimating the activity as the original activity class (i.e., the predicted class without attack), and $\mathcal{Z}(x+\delta)_i$ represent the possibility of estimating the activity as another class (i.e., a class that is different from the original-predicted activity class). k is a configurable parameter that controls attack confidence. For sample-specific targeted attacks, we define $\mathcal{L} = \max\left(\max_{i\neq t}\left(\mathcal{Z}(x+\delta)_i\right) - \mathcal{Z}(x+\delta)_t, -k\right)$, where $\mathcal{Z}(x+\delta)_t$ is the possibility of estimating the activity as the class t we desired. By optimizing the above adversarial loss functions, we aim to make our attack method not only confuse the HAR

systems (untargeted attack), but also force the HAR system to output our desired class (targeted attack). In practice, by using different special-designed adversarial loss functions, the attacker could either launch an untargeted attack or a targeted attack according to different attack strength requirements, which makes our attack framework more powerful and dangerous than previous studies [48, 87].

Perturbation Loss Generally speaking, the perturbations are the difference between the original mmWave sample and the adversarial one. L_2 Norm, which calculates the Euclidean distance between two sets has been commonly used as a metric for adversarial perturbation evaluation [10, 43, 48]. In this project, we define the perturbation loss $\mathcal{D} = \|\delta\|_2^2$ and generate the perturbation with minimal magnitude by optimizing the perturbation loss. In order to ensure the effectiveness of the perturbation and improve the efficiency of perturbation generation, we set a dynamic threshold τ for each mmWave sample to ensure $\|\delta\|_2^2 < \tau$, s.t. $f(x + \delta) = z$. The threshold is derived by analyzing the deviation between the normal mmWave sample and other normal samples. For a specific sample, we calculate the average L_2 Norm between the sample and all other available samples of the same type of activity, and set it as the threshold τ for perturbation generation.

Parameter Selection The weight λ, which determines the balance between the adversarial loss $\mathcal{L}$ and perturbation loss $\mathcal{D}$, must be set to a suitable number in order to cause gradient descent to minimize both components concurrently, as opposed to optimizing over one term at a time. In practice, we do a 12-step binary search to identify the appropriate λ and its accompanying adversarial perturbation δ.

6.1.2 Perturbation Optimization

Clipping In order to ensure the validity of the adversarial sample, there should be a clipping process after each training iteration. The clipping process trims the value of the adversarial sample to fall inside a valid range $[\alpha, \beta]$, which should be chosen based on the data representation of the activity samples in the HAR system. For a voxel-based HAR (e.g., [63]), the range should be $[0, \infty]$, as the value of each voxel represents the number of points within its limit. For a heatmap-based HAR (e.g., [77]), the range is usually set to [0, 255].

Discretization Discretization is a crucial process that is usually neglected in prior research [14, 43, 48]. However, due to the specific properties of mmWave data, we discover that perturbation discretization is necessary and cannot be disregarded. Specifically, the value of each pixel in a valid adversarial heatmap must be a discrete integer between 0 and 255, and a valid voxel often has a much lower upper limit value (e.g., 5) because of the sparse point clouds. Using the previous method that round the value of each adversarial voxel or heatmap to the nearest integer could eliminate minor perturbations and render the adversary's attack ineffective.

To handle this discrete optimization issue, we incorporate another loss function $\mathcal{L}_{model}(\lfloor x+\delta \rfloor)$, where $\lfloor x+\delta \rfloor$ represents the discrete adversarial sample. We mark the original $\mathcal{L}$ mentioned in Sect. 6.1.1 as $\mathcal{L}_{adv}$, and reformulate the final adversarial loss function as $\mathcal{L} = \mathcal{L}_{adv}(x+\delta)+\mathcal{L}_{model}(\lfloor x+\delta \rfloor)$. By simultaneously optimizing $\mathcal{L}_{adv}$ and $\mathcal{L}_{model}$, we could ensure the validity and efficiency of adversarial samples in different kinds of HAR systems.

Natural Style Optimization Furthermore, we discovered that the majority of existing approaches [10, 34, 48, 87] only focus on minimizing the perturbation magnitude by using a smaller L_2 Norm. However, little attention has been given to optimizing adversarial activity samples to have a natural style. In this study, we design a method to achieve natural style optimization by minimizing the radius of the generated perturbation and reducing the distance between the perturbation and the original mmWave sample. Our approach is important for mmWave activity samples because if the generated perturbation is too sparse or entirely separated from the original mmWave sample, it will be noticeable in heatmaps or voxels.

To minimize the radius of the generated perturbation, we reduce the pairwise Euclidean distance between elements inside the perturbation. We formulate it as $\mathcal{D}_{mean}(\delta) = \max_{m,n\in\delta} \|m-n\|_2$, where m, n are positions of any two elements (e.g., pixels in the heatmap) inside the generated perturbation δ. By reducing the average pairwise distance inside the perturbation, the radius of the perturbation can be reduced. We further reduce the distance between the perturbation and the original activity sample, allowing the perturbation to be concealed within the normal samples. In particular, we calculate *Chamfer Distance*, which seeks the nearest pairwise element euclidean distance between the generated perturbation and activity sample and takes the mean of all nearby element pair distances. It is expressed as $\mathcal{D}_{cf}(x,\delta) = \frac{1}{\|\delta\|_0}\sum_{m\in\delta}\min_{n\in x}\|m-n\|_2^2$, where x is the activity sample. By reducing the *Chamfer Distance*, the inserted perturbation is pushed nearer the activity sample. After integrating the above two functions, we reformulate the final perturbation loss function as follows:

$$\mathcal{D} = \mathcal{D}_{mag}(x, x+\delta) + \mathcal{D}_{mean}(\delta) + \mathcal{D}_{cf}(x,\delta), \tag{2}$$

where $\mathcal{D}_{mag}(\delta)$ controls the magnitude of the perturbation as mentioned in Sect. 6.1.1.

6.1.3 Practical Universal Targeted Attack Design

In this part, we provide details on how to launch efficient targeted attacks against HAR through universal perturbation design. Our basic idea is to create universal perturbations δ, such that $f(x_i+\delta) = z$, where x_i can be any activity samples from the same type of activity. The designed universal perturbation generation method consists of an offline training phase in which a training activity set is utilized to produce a universal perturbation, and an online test phase in which the universal

Algorithm 1: Universal Perturbation Generation

Input: Training set $\Omega = \{\Omega_1, \Omega_2, \cdots \Omega_i\}$, HAR model f, targeted activity class z, desired perturabtion magnitude τ, desired attack success rate ϵ on training set.
Output: Universal perturbation δ.
Initialize $\delta \leftarrow 0$.
while Success Rate$(\Omega) < \epsilon$**do**
  $\Omega_j \leftarrow RS(\Omega)$ ▷ Random Select a Sample
  if $f(x_j + \delta) = z$**then**
    Calculate the perturbation that satisfies: $\delta \leq \tau$.
  else
    $\Delta\delta_j \leftarrow \arg\min_{\Delta\delta_j} \mathcal{D}(\Delta\delta_j)$
      such that $f(\Omega_j + \delta + \Delta\delta_j) = z$.
    $\delta \leftarrow (\delta + \Delta\delta_j)$. ▷ Update the Perturbation
  endif
end while

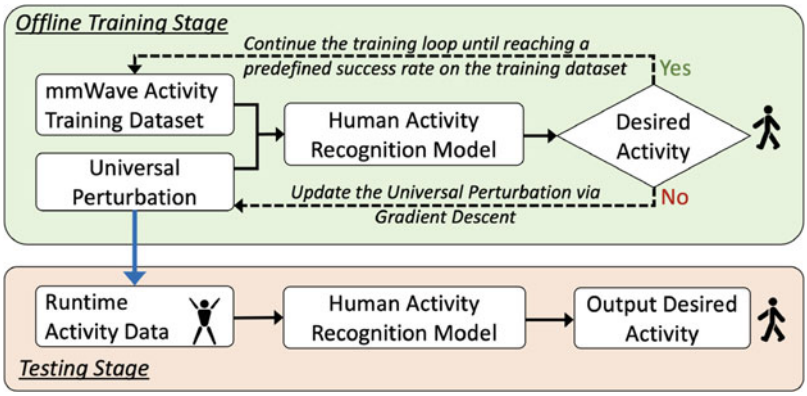


Fig. 6 The overview of our designed universal attack consists of an offline perturbation generation phase and an online testing phase

perturbation is directly applied to runtime activity data for a targeted attack. As shown in Fig. 6, we generate universal perturbations δ for each type of activity, such that when the perturbation is applied to the majority of activity data x from the same class, the HAR always recognizes it as our desired class z. We generate the perturbation for each activity sample in the training set using the same objective function (Eq. (1)). To make the adversarial perturbation work for the majority of activity examples in the training set, we iteratively adjust the universal perturbation.

Specifically, the adversarial perturbation is started with zeros and added to an mmWave activity sample. If the HAR's prediction does not match the desired activity class, the perturbation will be modified in the direction of gradient descent, in which the likelihood of the desired class increases. Otherwise, the current perturbation is applied to a fresh training activity sample. If the existing universal perturbation does not fit in the new sample, a minimal magnitude perturbation revision is calculated and added to the current universal perturbation. The iteration

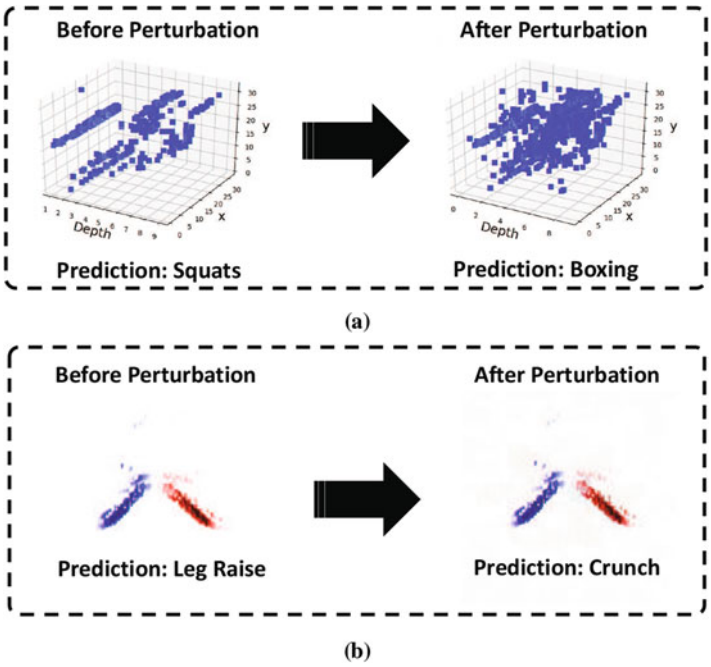


Fig. 7 Two representative adversarial samples generated by adding universal perturbations directly. (**a**) Adversarial voxel-based data generation with L_2 Norm of 21; (**b**) Adversarial heatmap-based data generation with L_2 Norm of 2083

process ends when the universal perturbation on the training dataset exceeds a predefined success rate (e.g., 70%). Notably, the objective of the technique is not to seek the smallest global perturbation that fools the majority of activity samples, but rather to select one that is sufficiently tiny. Figure 7 depicts the production of adversarial samples by directly applying universal perturbation. We observe that the adversarial instances deviate from the original sample only slightly in terms of L_2 Norm. The adversarial samples look natural, which makes them hard to be noticed by naked eye. However, adversarial examples enable HAR systems to efficiently predict the activity as desired. Compared with traditional sample-specific attacks methods, our universal perturbations would significantly shorten the attack launch time, which makes it more practical in time-constraint attack scenarios. Different from existing universal untargeted attack methods that need to insert padding frames between two successive activities [48], our method modifies the activity sample directly and thus broadens its applicability to various of mmWave-based HAR systems. In addition, our universal attack method is compatible with both untargeted and targeted attacks by merely modifying the adversarial loss function.

6.2 Black-Box Attack Implementation

Black-box attack is a more challenging scenario where the attacker usually cannot access the victim model but only the input and output of the model [48]. Thus, the adversarial perturbation cannot be created and updated by exploiting the gradient from the victim model. A potential approach to a black-box attack is to train a substitute model. Adversarial samples generated by the substitute model can be exploited to launch attacks towards the victim model, leveraging the transferability of the adversarial sample.

We begin with a basic black-box setting where the training data of the victim model is fully accessible. The key challenge is how to ensure the similarity between the target and the substitute model. Directly training the substitute model on the dataset usually gets poor performance since the structure of the substitute model is different [48]. To solve such a problem, we take advantage of Knowledge Distillation (KD) to learn a substitute model that can mimic the prediction of the victim model [21]. In black-box scenarios, although the inner structure of the victim model is inaccessible, its output class and soft logits indicate the class probability distribution for a given input is accessible [34]. Supposing the soft logits of the target and the substitute model are P_t and P_s, respectively. The predicted class of substitute model is Z and the ground truth is G. We formulate the KD process as the loss function $L = L_s + L_d$, where $L_s = \text{CrossEntropy}(Z, G)$ and $L_d = \text{KLDivergence}(P_t, P_s)$. By optimizing the loss function, we transfer the dark knowledge from the victim model to the substitute model [21].

To ensure the robustness of the black-box attack, we utilize a configurable parameter of k to control the confidence of the attack as mentioned in Sect. 6.1.1. With a larger k value, the possibility that the adversarial sample being misclassified by the victim model will increase. We set $k = 0$ in white-box scenarios and set a larger k in black-box scenarios. We evaluate the impact of k in Sect. 7.4.

We then move to a more challenging scenario where the original training data of the victim model is only partially accessible. To deal with the problem of insufficient training data, we develop a GAN to synthesize sufficient pseudo training samples. GAN has been proved to generate high-quality pseudo samples with a limited amount of real samples [77]. In this chapter, we implemented a GAN with a 3-layer generator and a 3-layer discriminator to generate sufficient activity samples using only 20% of the original training dataset of the victim model. Specifically, the generator seeks to learn the distribution of the real samples so as to have the ability of synthesizing pseudo samples. The discriminator tries to discriminate whether a sample is a real or pseudo one. The generator and discriminator are trained in turn to optimize each other by updating the parameters of their networks. The final state is a Nash equilibrium, where the synthesized pseudo samples are similar to the real ones, and the discriminator fails to identify whether the activity samples are real or not. After obtaining enough high-quality pseudo training samples, we exploit the KD method mentioned above to train the substitute model and launch black-box attacks toward the victim model.

7 Performance Evaluation

7.1 Experimental Setup

Equipment We have classified mmWave-based HAR systems into two distinct categories based on their unique data representations after conducting an exhaustive investigation. This approach to classification allowed us to select one representative dataset from each category in order to implement our designed adversarial attack methodologies. Specifically, we develop our own mmWave-based HAR system [83] with heatmap representations in order to examine its susceptibility to adversarial attacks. We construct our own HAR system (heatmap-based) using a single mmWave device, the TI AWR1642 [8], which incorporates a 2×4 antenna array. The frequency range of the device is between 77 GHz and 81 GHz. The sampling rate is fixed at 100 frames per second, with 17 chirps per frame. A TI DCA1000EVM [69] data capture card is employed to acquire data from the mmWave device and transmit it to a Dell laptop for deep model inference. In addition, we select an existing mmWave-based HAR system [63] as a representative to investigate the voxel-based HAR model's susceptibility to adversarial attacks. It utilizes TI IWR1443BOOST radar [70] to gather the new point cloud dataset. This radar also operates in the frequency range of 77 GHz to 81 GHz. With four receiver and three transmitter antennas, the radar is able to monitor multiple objects based on their distance and angle data. This antenna design allows for the estimation of both azimuth and elevation angles, enabling the detection of objects in a three-dimensional plane.

Data Collection Two datasets are used in our experiment. The public voxel-based human activity dataset includes 156,355 samples representing 5 distinct activities. The radar is mounted on a tripod platform at a height of 1.3 m for data collection. They have gathered information from two users. Five distinct activities are performed by the participants in front of the radar. These include walking, jumping, jumping jacks, squatting, and boxing. For a subject performing the same activity, data is collected in continuous intervals of approximately 20 seconds. They have collected a total of 93 minutes of data. The captured point clouds include spatial coordinates (x,y,z in meters), velocity (in meters/second), range (distance of the point from the radar) in meters, intensity (dB), and bearing angle (degrees). The radar's sampling rate is 30 pixels per second.

They separated collected data files into train and test files, with the train containing 71.6 minutes of data and the test containing 21.4 minutes of data. To surmount the nonuniform number of points in each frame, the point clouds were converted into voxels with dimensions of $10 \times 32 \times 32$ (depth=10), making the magnitude of the input constant regardless of the number of elements in the frame. By evaluating their efficacy, they empirically determined these dimensions. The value of each voxel in the voxel representation is the number of data points contained within its boundaries. Due to the fact that activities are carried out over a

Table 1 14 common in-home full-body activities

W1	Crunches	W8	Squats
W2	Elbow plank and reach	W9	Burpees
W3	Leg raise	W10	High knees
W4	Lunges	W11	Turning kicks
W5	Mountain climber	W12	Chest squeezes
W6	Punches	W13	Side leg raise
W7	Push ups	W14	Side to side chops

period of time, the time window from activities is derived to encapsulate temporal dependencies. They generate 2-second windows (60 frames) with a 0.33-second sliding factor (10 frames). Finally, they receive 12,097 in training samples and 3538 in testing samples. 20% of the training samples are used for validation. In the voxelized representation of the time window, each sample has a $60 \times 10 \times 32 \times 32$.

For our human activity heatmap-based dataset. We enlist 11 volunteers aged 20 to 44, ranging in height from 162 to 185 centimeters and weight from 50 to 86 kilograms. The volunteers are required to complete 14 typical indoor exercises, as outlined in Table 1. Four distinct environments (such as the lounge, corridor, small classroom, and large classroom) are utilized to capture exercise data. We position the mmWave device on a 60cm-high table and record the ground truth recordings using a camera. During an 8-month study, we ask each participant to complete at least 20 repetitions of each type of exercise (half of the segments are used to train the model and the rest segments are used to assess performance). We collect over 7000 segments from volunteers in total.

Victim Models Deep learning models are utilized by both the voxel-based and heatmap-based HAR systems due to their superior performance compared to traditional machine learning models. The authors train a convolution neural network (CNN) combined with long short-term memory (LSTM) for the voxel-based victim HAR model. CNN layers are applied to every temporal segment of the input data by time-distributed CNN. Time-distributed CNN + Bi-directional LSTM classifier consists of three time-distributed convolutional modules (convolution layer + convolution layer + maxpooling layer) followed by a bi-directional LSTM layer and an output layer. The network has 291,000 trainable parameters overall. This classifier is trained directly on the input sample's time and spatial dimensions. They use the sklearn GridSearchCV function to optimize the SVM's hyperparameters (C and gamma). Deep learning classifiers were trained using Adam optimizer with a learning rate of 0.001. After training for 30 epochs, the models with the lowest loss on validation data were preserved. The original classification accuracy of the voxel-based deep learning model is 90.47%.

For the heatmap-based victim HAR model, we use a convolution neural network. The input of the deep learning model is an array with a size of $224 \times 224 \times 3$. The array is generated by quantifying all the pixels in the Spatial-Temporal Heatmap into 3-D tuples representing RGB value. The feature extractor has three convolutional

layers, each with a 3×3 filter and a ReLU activation function. The convolutional layers are used to up-sample the image by extracting high-dimensional features from the spatial correlation of pixels. Each convolutional layer is followed by a max-pooling layer with a stride of 2 and a filter size of 2×2 to downsample and reduce data redundancy. After the process of 3 rounds of up-sampling and down-sampling, a 64-dimension feature map is obtained. Then, a flatten layer is integrated to reduce the feature map into a one-dimension array. Given an input data D, the feature extractor produces feature representations $R = F(D, \Theta_f)$, where F represents the feature extractor and Θ_f represents its trainable parameters. Based on the derived feature representation R, a neural network consisting of two dense layers is followed to classify the inputs into several classes (e.g., different types of workouts). Given the input representation R, the classifier predicts the label as Y_c. We optimize the classifier by minimizing the cross-entropy loss between the ground truth $\tilde{Y}_c$ and the predicted label Y_c as $L_c = L_{CE}\left(Y_c, \tilde{Y}_c\right)$, where L_{CE} represents the cross-entropy loss function. The original classification accuracy of the heatmap-based deep learning model is 97%. In addition, for the implementation of our adversarial attacks in this project, the prototype of our designed attack methods is implemented using Python along with TensorFlow.

Evaluation Metrics We use three metrics to evaluate the performance of our attack scheme. (1) *Success Rate (SR):* it represents the number of succeeded adversarial attacks over the total number of attack attempts. In the untargeted attack, we report a success when the predicted class is different from the original class while in the targeted attack, we only report success if the predicted class matches the desired target class; (2) L_2 *Norm:* it indicates the Euclidean distance between the adversarial sample and original sample; Smaller L_2 Norm values indicate that the adversarial sample is similar to the original activity sample, therefore, harder to be noticed by human eyes. (3) *Confusion Matrix*: Each cell in the matrix indicates an original-target class pair that the actual class in the row is classified as the target class in the column. The value of each cell represents the average SR and L_2 Norm of the corresponding universal attack on the testing set.

7.2 Evaluation of White-Box Attack

First, we evaluate the efficacy of sample-specific targeted attack and untargeted attack in white-box scenarios for mmWave-based HAR systems. In the white-box scenario, the attackers have complete knowledge of the inputs, architecture, and parameters of the victim deep learning model. The adversary may also access the victim model continuously to generate adversarial samples. In addition, the adversary may be familiar with the data preprocessing procedures of the HAR system in order to provide the appropriate perturbation. To thoroughly evaluate the efficacy of sample-specific attacks, we randomly select 200 and 100 activity

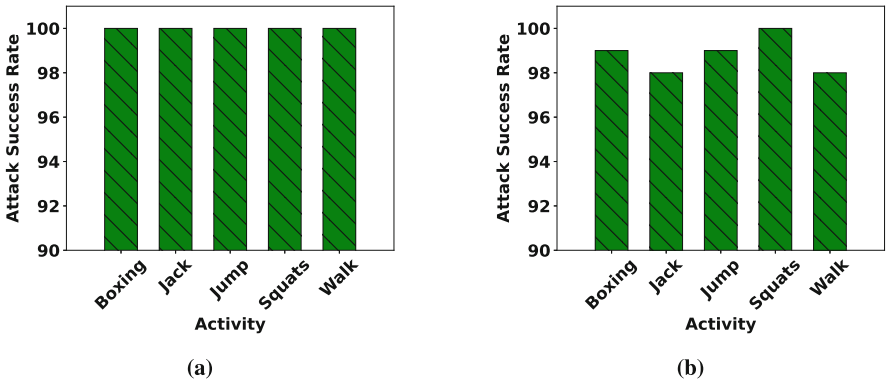


Fig. 8 (**a**) Success rate of sample-specific untargeted attacks on voxel-based dataset; (**b**) Success rate of sample-specific targeted attacks on voxel-based dataset (x-axis represents the original classes)

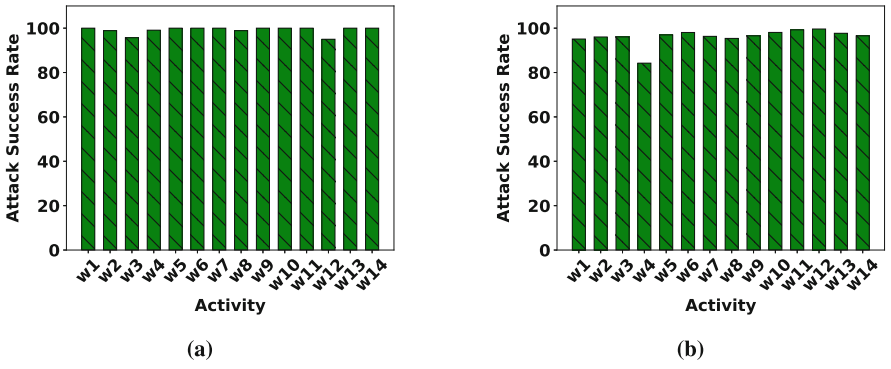


Fig. 9 (**a**) Success rate of sample-specific untargeted attacks on heatmap-based dataset; (**b**) Success rate of sample-specific targeted attacks on heatmap-based dataset (x-axis represents the original classes)

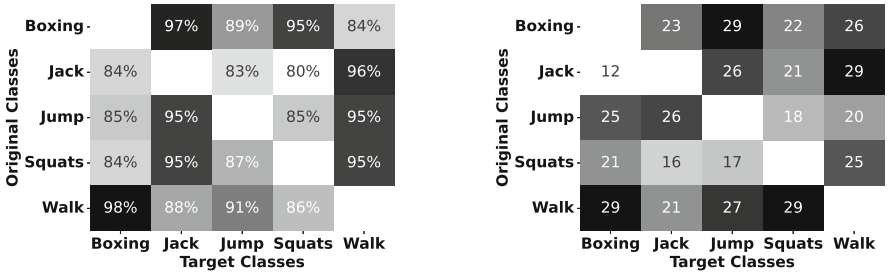
samples for each activity type from the voxel-based and heatmap-based testing sets, respectively. To evaluate the efficacy of universal attacks in white-box scenarios for mmWave-based HAR systems, half of the selected samples are used to generate universal perturbations (universal attack training set), while the other half are used to evaluate the universal attack (universal attack testing set).

Untargeted Attack In untargeted attacks, the adversary attempts to confound the HAR system by modifying the output from the original activity prediction to a different prediction. In another word, we report a success untargeted attack when the predicted class is different from the original class. Figures 8a and 9a demonstrate the attack success rate of our untargeted attacks on the voxel-based HAR dataset and heatmap-based dataset, respectively. We can learn that our method achieves nearly 100% attack SR for all 5 original classes in the voxel-based dataset and all

14 original classes in the heatmap-based dataset, indicating that almost all samples tested are class-flipped from the original class under our attack scheme. Because the goal of untargeted attacks is just to cause the HAR models to make mistakes, these high attack success rates demonstrate the effectiveness of our methods.

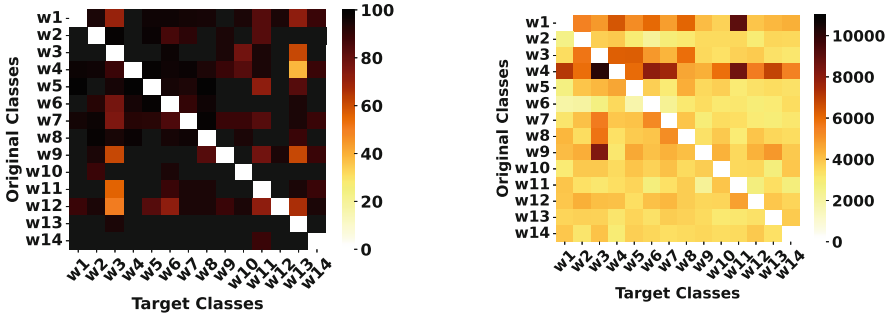
Targeted Attack In targeted attacks, the adversary seeks to cause HAR systems to predict a particular class. To achieve this objective, we must modify the sample of activities so that the probability score of the preferred activity in the victim model is greater than that of all other enrolled activities. This work is more difficult because we only report a successful targeted attack if the predicted target class matches the intended target class. Figures 8b and 9b demonstrate the attack success rate of sample-specific targeted attacks on the two datasets, respectively. The x-axis indicates the initial classifications. Each sample from the original class is modified to be recognized as the other activities, i.e., the other 4 activities in a voxel-based dataset or the other 13 activities in a heatmap-based dataset. Our method achieves an average SR of 96% on both datasets. Note that attacks towards some target classes have relatively lower SR (i.e., jack and walk from the voxel-based dataset; and $w4$ (lunges) from the heatmap-based dataset). This is because those classes have more different patterns from other activities, making the attack relatively harder. But even the lowest SR in targeted attack is still higher than 80%, proving the effectiveness of our targeted attack method.

Universal Attack We design the universal attacks to enhance the efficacy of targeted attacks and make them applicable in time-constrained environments. This would permit the use of targeted attacks in situations with limited time. By launching a universal attack, we will be able to directly insert the universal perturbation into previously unseen activity samples without incurring any additional training expenses. Launching universal attacks is not a simple task due to the fact that activity data samples gathered at different times or under different conditions frequently differ, meaning that the perturbation designed to attack one sample may not be effective against another sample, even if they are from the same activity type. Launching universal attacks is necessary. It is not always possible to generate a sample-specific perturbation that is tailored to each activity sample due to the time-consuming nature of generating the perturbation for a particularly high-dimensional mmWave activity sample. It is more practicable to generate universal perturbations that can be applied to distinct samples from the same type of activity. To illustrate the efficacy of our universal attacks, we conduct targeted attacks against each activity type and present the results using a confusion matrix. Each cell in the matrix represents a pair of original-target classes, where the actual class in the row corresponds to the target class in the column. The value of each cell represents the SR of the corresponding universal attack on the testing set. The SR of universal attacks represents the percentage of activity samples in the testing set that are classified as the desired class when subjected to the identical perturbation.



(a) (b)

Fig. 10 (**a**) Success rate of universal targeted attacks on voxel-based dataset; (**b**) L_2 Norm of generated universal perturbations on voxel-based dataset



(a) (b)

Fig. 11 (**a**) Success rate of universal targeted attacks on heatmap-based dataset; (**b**) L_2 Norm of generated universal perturbations on heatmap-based dataset

The performance of our universal attacks over the voxel-based HAR dataset is demonstrated in Fig. 10a. We can learn that all universal attacks achieve over 80% SR, with the highest SR reaching 98% (98% of walk samples in the testing set has been classified as boxing using the same universal perturbation). We note that the SR of some original-target pairs (e.g., jack-squats) is relatively low. This is because the samples of the target class vary a lot from the original class, making it harder to launch targeted attacks. Despite this, our method still achieves an overall SR of 90%. For the heatmap-based dataset, as is shown in Fig. 11a, attacks on most original-target pairs achieve higher than 90% SR. Few pairs (e.g., $w4$–$w13$, $w12$–$w3$) have relatively low SR due to large differences between original and target samples. But our method still reaches an average SR of 94% over 182 original-target pairs on the heatmap-based dataset.

7.3 Impact of Perturbation Magnitude

There are a number of reasons why it's important for us to minimize the impact of adversarial perturbations and create adversarial activity samples that are highly similar to the original samples. First, we aim to subtly alter the activity samples so that the perturbations are virtually undetectable while preserving the original sample's inherent properties. This stealthiness is essential in a hostile environment, where noticeable changes could signal the presence of an attack. Second, we strive to preserve the original classification of an activity from a human standpoint, despite causing misclassification in the recognition system. This highlights the disparity between human and HAR model perception, highlighting the vulnerabilities of the HAR system. Using minimal perturbations, we demonstrate that even minor deviations from the original mmWave data can result in substantial misinterpretations by the victim system. Our research aims to make a clear illustration of the HAR system's security issues, spurring the creation of more durable mmWave-based HAR systems.

In this research, we optimize the L_2 Norm in order to minimize the magnitude of the perturbation. In order to guarantee the efficacy of the perturbation and increase the efficacy of its generation, a dynamic threshold is set for each activity sample. For a particular sample, we compute the average L_2 Norm between the sample and all other available samples of the same type of activity and use this value as the perturbation generation threshold.

Untargeted Attack To evaluate the impact of perturbation magnitude on our white-box attacks, we first evaluate the impact of perturbation magnitude on untargeted attacks. Figure 12a demonstrates the L_2 Norm of untargeted attacks on the voxel-based dataset. Note that each red line indicates the average value of the thresholds of each activity type. We can learn that the median L_2 Norm of untargeted adversarial samples on all 5 original classes are below 10 and the maximum L_2 Norm values are all lower than 30, far below the 5 average thresholds, which are all around 40–50. For the heatmap-based dataset, as is shown in Fig. 13a, the median L_2 Norms between adversarial and original samples are around 2000–2500. Adversarial samples towards one workout, $w4$, have relatively higher L_2 Norm distribution due to high-specific features of original heatmaps. But the highest L_2 Norm is still lower than 4000, far below the average threshold of 14,000 for $w4$.

Targeted Attack We then evaluate the impact of perturbation magnitude on targeted attacks. Figure 12b demonstrates the L_2 Norm of targeted attacks on the voxel-based dataset. We find that the median L_2 Norm values of samples towards all 5 target classes are still around 20. But there is a significant increase in the maximum L_2 Norm value on all 5 classes compared with untargeted attacks. This is because in targeted attacks, we not only need to flip the class but also need to turn the class to the required one. Thus, for some samples a larger perturbation magnitude is needed. But even the maximum L_2 Norm values are still below the corresponding average threshold (i.e., red lines in the Figure). On the heatmap-based dataset, as is shown in

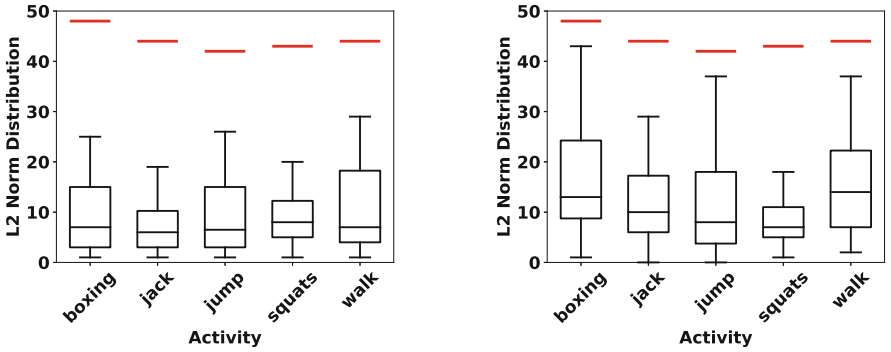


Fig. 12 (**a**) L_2 Norm of perturbations generated in sample-specific untargeted attacks on voxel-based dataset (The red line represents the average threshold of all attack samples); (**b**) L_2 Norm of perturbations generated in sample-specific targeted attacks on voxel-based dataset (x-axis represents the original class)

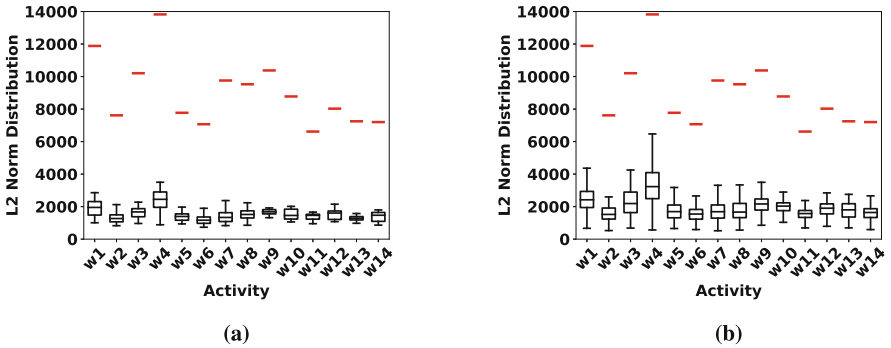


Fig. 13 (**a**) L_2 Norm of perturbations generated in sample-specific untargeted attacks on heatmap-based dataset; (**b**) L_2 Norm of perturbations generated in sample-specific targeted attacks on heatmap-based dataset

Fig. 13b, we also notice a larger L_2 Norm distribution compared with the result of untargeted attacks. Samples aiming at the target class of $w4$ have a relatively higher maximum L_2 Norm value due to the highly-specific features of the original heatmap from this class. But even the maximum perturbation (i.e., 6300) of attack samples (i.e., $w4$) still does not exceed the corresponding average threshold.

Universal Attack We next evaluate the impact of perturbation magnitude on universal attacks. The confusion matrix of universal L_2 Norm on voxel-based dataset and heatmap-based dataset are shown in Figs. 10b and 11b, respectively. We can learn that the average L_2 Norm for the adversarial samples towards voxel-based dataset is between 12 to 29, which is far below the average threshold of 5 classes

(i.e., around 40–50). The average L_2 Norm of universal samples towards heatmap-based dataset over 182 original-target pairs is 4000. Though some pairs (e.g., $w4$–$w3$, $w1$–$w11$) have relatively higher perturbation magnitude due to relatively large differences in heatmap patterns, these values are still below the average threshold of corresponding original classes.

7.4 Evaluation of Black-Box Attack

In this research, we delve deeper into black-box attacks for several crucial reasons. First, black-box attacks are a more probable form of threat. In real-world applications, attackers typically do not have access to the HAR model's architecture, parameters, or training data. By examining black-box attacks, we hope to better prepare these HAR systems for actual threats. The successful execution of black-box attacks against mmWave-based HAR systems can reveal fundamental vulnerabilities that are independent of the system's design. This enables researchers to comprehend the robustness of the HAR system in a more general sense, thereby guiding the development of more secure systems. Lastly, investigating black-box attacks can provide valuable insights into the design of defensive mechanisms that do not rely on modifying the HAR system's internals, but instead emphasize external measures, such as activity sample validation or anomalous activity detection. Exploring black-box adversarial attacks permits us to comprehensively and practically evaluate and improve the security of mmWave-based HAR.

All black-box experiments are taken on the heatmap-based dataset due to the page limit. We begin with the basic settings where the victim models are inaccessible but we assume that the attacker has full access to the training data set. We use KD to train a substitute model to generate adversarial samples and launch attack towards the victim model. Our substitute model is a 2-layer CNN network with $3.2M$ trainable parameters. We also trained the substitute model directly on the training set without KD for comparison. As mentioned in Sect. 6.2, we exploit a confidence value of k to ensure the robustness of our attack method. We change the k value from 0 to 40 with a step size of 5 to study the impact of k. Figures 14a, b and 15a demonstrate the average SR and L_2 Norm under basic black-box settings for untargeted, targeted, and universal attacks, respectively. We can learn that the substitute model trained with KD outperforms the directly-trained model for all k larger than 0 in all types of attacks. When $k = 40$, the substitute model can achieve over 80% SR for untargeted and targeted attack as well as an SR of 75% for universal attack. We can also notice a trade-off between SR, L_2 Norm and k values. As k increases, we can obtain a higher SR but the L_2 Norm also increases accordingly, meaning the adversarial samples have more significant perturbations. However, our method still maintains the L_2 Norm value of less than 6500 for all three types of attacks even when $k = 40$.

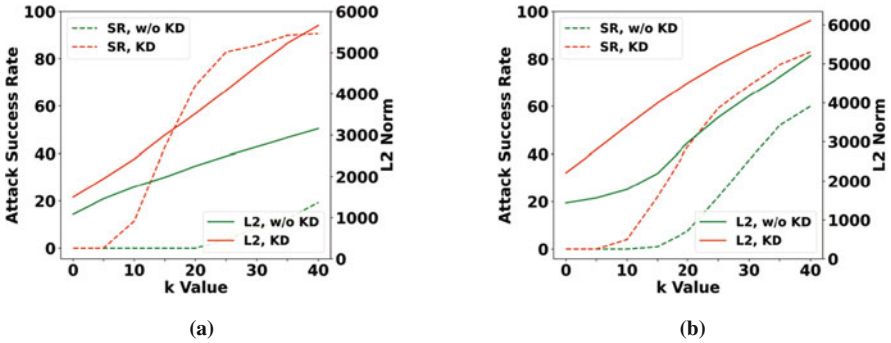


Fig. 14 (**a**) Success rate and L_2 Norm of untargeted black-box attack; (**b**) Success rate and L_2 Norm of targeted black-box attack

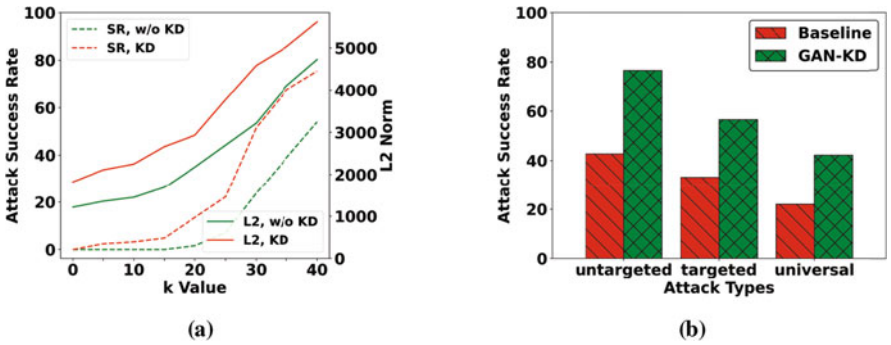


Fig. 15 (**a**) Success rate and L_2 Norm of universal black-box attack; (**b**) Success rate of GAN-KD-based black-box attack

We next move to a more challenge setting where the adversary can only access part of the training data used by the victim model. To overcome the training data deficiency issues, we exploit the GAN method mentioned in Sect. 6.2 to generate a pseudo training set with a size similar to that of the original training set using only 20% of original training data. The substitute model is trained using KD and the generated training set. We set confidence value $k = 40$ since previous results have proven that this confidence value can obtain relatively robust performance. For comparison, we trained a baseline model without KD and GAN using only 20% of the original training data, similar to the black-box model used in [48]. As is shown in Fig. 15b, the GAN-KD trained substitute model outperforms the baseline model for all 3 types of attacks, with the highest SR of 76.5% for the untargeted attack. Due to higher requirements for the adversarial samples, SR of targeted and universal attacks using the GAN-KD method are relatively low (i.e., 56 and 42%), but the SR still outperforms the baseline model with an increase of 23.4 and 20.22%, respectively.

8 Conclusion

Artificial intelligence (AI) has arisen as a swiftly advancing subfield of computer science with the goal of emulating and augmenting human intelligence. This expansion has been fueled by ultra-high-performance computing technology and the introduction of deep learning, resulting in a significant evolution of traditional AI technology. In recent years, deep learning has marked a substantial surge forward in the development of artificial intelligence in areas such as computer vision, speech recognition, and text comprehension. It is capable of detecting concealed non-linear correlations in data, supporting new file types, and identifying unidentified threats, which significantly improves cybersecurity defense. However, even the most sophisticated AI technologies, such as deep learning, are susceptible to adversarial attacks that can result in incorrect classification or prediction outcomes.

Due to their convenience, adaptability, and capacity to transmit data over long distances without the need for physical connections, wireless signals have become widely adopted for communication. The prevalence of wireless signals in our everyday electronic devices enables complex networks of reflected rays in indoor environments. Advances in wireless technologies and sensing methodologies have facilitated the use of wireless signals, such as millimeter wave signals, for wireless sensing tasks. The fundamental premise of these applications is that human movement affects wireless signal propagation, making it possible to track human motions by examining the received signals. This breakthrough has led to various applications, including human activity recognition, human localization, and vital sign monitoring.

Among these, Human Activity Recognition (HAR) has gained considerable recognition as a key technology in numerous Internet of Things (IoT) and security applications, such as health monitoring and user authentication. However, the widespread deployment of HAR systems has uncovered vulnerabilities in security, specifically their susceptibility to adversarial attacks. Since the majority of HAR systems employ deep learning models for activity identification due to their high accuracy and ability to handle interference in the real world, a gap exists in research exploring the vulnerability of HAR systems to adversarial attacks. In domains such as healthcare, security, and smart home applications, false human activity detection can have significant repercussions. Given the extensive integration of HAR systems in these critical applications, a thorough investigation into the variety of adversarial attacks that target HAR models is crucial. This chapter focuses on adversarial attacks on AI-empowered HAR, aiming to increase awareness about their susceptibility and shed light on the security challenges they present.

In this chapter, we take a pioneering approach toward adversarial attacks on mmWave-based HAR systems, going beyond the prevalent research focus on untargeted attacks. Our unique contribution lies in the design and investigation of practical and universal perturbations to enable targeted adversarial attacks. To ensure wide applicability, these perturbations are generated through an iterative algorithm. Additionally, we affirm the validity of mmWave-based adversarial

samples and shape them into a more natural style. In order to address the information deficiency and scarcity of training data in black-box scenarios, we also incorporate techniques such as knowledge distillation and generative adversarial networks. Through our thorough investigation, we have classified mmWave-based HAR systems into two distinct categories based on their specific data representation. This classification approach enabled us to select representative datasets from each category for applying our designed adversarial attack methodologies. We are confident in this research approach as we believe these methodologies can be extended to a broader range of datasets that share similar data representation characteristics.

Physical adversarial attacks on mmWave-based HAR systems remain a relatively unexplored area of research, and thus, future investigation into physical adversarial attacks will significantly contribute to the understanding of the overall security dynamics of these systems. This understanding will undoubtedly be invaluable in developing more resilient defensive strategies.

References

1. Abdelnasser H, Harras KA, Youssef M (2015) Ubibreathe: a ubiquitous non-invasive wifi-based breathing estimator. In: Proceedings of the 16th ACM international symposium on mobile Ad Hoc networking and computing
2. Abdelnasser H, Youssef M, Harras KA (2015) Wigest: a ubiquitous wifi-based gesture recognition system. In: 2015 IEEE conference on computer communications (INFOCOM), pp 1472–1480
3. Ahuja K, Jiang Y, Goel M, Harrison C (2021) Vid2doppler: synthesizing doppler radar data from videos for training privacy-preserving activity recognition. In: Proceedings of the 2021 CHI conference on human factors in computing systems, pp 1–10
4. Akpa EAH, Fujiwara M, Arakawa Y, Suwa H, Yasumoto K (2018) Gift: glove for indoor fitness tracking system. In 2018 IEEE international conference on pervasive computing and communications workshops (PerCom workshops)
5. Akpa EAH, Fujiwara M, Arakawa Y, Suwa H, Yasumoto K (2018) Gift: glove for indoor fitness tracking system. In: 2018 IEEE international conference on pervasive computing and communications workshops (PerCom workshops), pp 52–57
6. Alam MAU, Rahman MM, Widberg JQ (2021) Palmar: towards adaptive multi-inhabitant activity recognition in point-cloud technology. In IEEE INFOCOM 2021-IEEE conference on computer communications, pp 1–10
7. Ambalkar H, Wang X, Mao S (2021) Adversarial human activity recognition using wi-fi csi. In: 2021 IEEE Canadian conference on electrical and computer engineering (CCECE), pp 1–5
8. Awr1642 data sheet, product information and support | ti.com. https://www.ti.com/product/AWR1642. Accessed 12 Jun 2023
9. Bo H, Xu L, Hao L, Dou Y, Zhao L, Yu W (2016) A single-channel non-orthogonal i/q rf sensor for non-contact monitoring of vital signs. Appl Comput Electromagn Soc J 31(6)
10. Carlini N, Wagner D (2017) Towards evaluating the robustness of neural networks. In: 2017 IEEE symposium on security and privacy (sp), pp 39–57
11. Çeliktutan O, Akgul CB, Wolf C, Sankur B (2013) Graph-based analysis of physical exercise actions. In: Proceedings of the 1st ACM international workshop on Multimedia indexing and information retrieval for healthcare, pp 23–32

12. Chu D, Lane ND, Lai TT-T, Pang C, Meng X, Guo Q, Li F, Zhao F (2011) Balancing energy, latency and accuracy for mobile sensor data classification. In: Proceedings of the 9th ACM conference on embedded networked sensor systems, pp 54–67
13. Ding H, Han J, Shangguan L, Xi W, Jiang Z, Yang Z, Zhou Z, Yang P, Zhao J (2017) A platform for free-weight exercise monitoring with passive tags. IEEE Trans Mob Comput 16(12):3279–3293
14. Duan R, Ma X, Wang Y, Bailey J, Qin AK , Yang Y (2020) Adversarial camouflage: hiding physical-world attacks with natural styles. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 1000–1008
15. Eagle N, Pentland A (2006) Reality mining: sensing complex social systems. Pers Ubiquitous Comput 10:255–268
16. Ghorbel E, Boutteau R, Boonaert J, Savatier X, Lecoeuche S (2018) Kinematic spline curves: a temporal invariant descriptor for fast action recognition. Image Vision Comput 77:60–71
17. Goodfellow IJ, Shlens J, Szegedy C (2014) Explaining and harnessing adversarial examples. Preprint, arXiv:1412.6572
18. Gu F, Chung M-H, Chignell M, Valaee S, Zhou B, Liu X (2021) A survey on deep learning for human activity recognition. ACM Comput Surv 54(8):1–34
19. Guo X, Liu J, Shi C, Liu H, Chen Y, Chuah MC (2018) Device-free personalized fitness assistant using wifi. Proc ACM Interact Mob Wearable Ubiquitous Technol 2(4):1–23
20. Hassana MM, Uddin Z, Mohamed A, Almogrena A (2018) A robust human activity recognition system using smartphone sensors and deep learning. Future Gener Comput Syst 81:307–313
21. Hinton G, Vinyals O, Dean J, et al (2015) Distilling the knowledge in a neural network. 2(7). Preprint, arXiv:1503.02531
22. Hui X, Kan EC (2018) Monitoring vital signs over multiplexed radio by near-field coherent sensing. Nat Electron 1(1):74–78
23. Hussain Z, Sagar S, Zhang WE, Sheng QZ (2019) A cost-effective and non-invasive system for sleep and vital signs monitoring using passive rfid tags. In: Proceedings of the 16th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services, pp 153–161
24. Iso T, Yamazaki K (2006) Gait analyzer based on a cell phone with a single three-axis accelerometer. In: Proceedings of the 8th conference on Human-computer interaction with mobile devices and services, pp 141–144
25. Jin F, Sengupta A, Cao S (2020) mmfall: fall detection using 4-d mmwave radar and a hybrid variational rnn autoencoder. IEEE Trans Autom Sci Eng 19(2):1245–1257
26. Kaltiokallio O, Yiğitler H, Jäntti R, Patwari N (2014) Non-invasive respiration rate monitoring using a single cots tx-rx pair. In: IPSN-14 proceedings of the 13th international symposium on information processing in sensor networks
27. Khamis A, Chou CT, Kusy B, Hu W (2018) Cardiofi: Enabling heart rate monitoring on unmodified cots wifi devices. In: Proceedings of the 15th EAI international conference on mobile and ubiquitous systems: computing, networking and services, pp 97–106
28. Su C-J, Chiang C-Y, Huang J-Y (2014) Kinect-enabled home-based rehabilitation system using dynamic time warping and fuzzy logic. Appl Soft Comput 22:652–666
29. Könönen V, Mäntyjärvi J, Similä H, Pärkkä J, Ermes M (2010) Automatic feature selection for context recognition in mobile devices. Pervasive Mob Comput 6(2):181–197
30. Kwon SM, Yang S, Liu J, Yang X, Saleh W, Patel S, Mathews C, Chen Y (2019) Demo: hands-free human activity recognition using millimeter-wave sensors. In: 2019 IEEE international symposium on dynamic spectrum access networks (DySPAN)
31. Laput G, Ahuja K, Goel M, Harrison C (2018) Ubicoustics: plug-and-play acoustic activity recognition. In: Proceedings of the 31st annual ACM symposium on user interface software and technology, pp 213–224
32. Li Z, Wu Y, Liu J, Chen Y, Yuan B (2020) Advpulse: universal, synchronization-free, and targeted audio adversarial attacks via subsecond perturbations. In Proceedings of the 2020 ACM SIGSAC conference on computer and communications security, pp 1121–1134

33. Li G, Ge Y, Wang Y, Chen Q, Wang G (2022) Detection of human breathing in non-line-of-sight region by using mmwave fmcw radar. IEEE Trans Instrum Meas 71:1–11
34. Lin Y, Zhao H, Tu Y, Mao S, Dou Z (2020) Threats of adversarial attacks in dnn-based modulation recognition. In IEEE INFOCOM 2020 - IEEE conference on computer communications
35. Lin Z, Xie Y, Guo X, Ren Y, Chen Y, Wang C (2020) Wieat: fine-grained device-free eating monitoring leveraging wi-fi signals. In 2020 29th international conference on computer communications and networks (ICCCN).
36. Lin Z, Xie Y, Guo X, Wang C, Ren Y, Chen Y (2020) Wi-fi-enabled automatic eating moment monitoring using smartphones. In: IoT technologies for HealthCare: Proceedings of the 6th EAI international conference, HealthyIoT 2019, Braga, Portugal, December 4–6, 2019, , pp 77–91
37. Liu H, Gan Y, Yang J, Sidhom S, Wang Y, Chen Y, Ye F (2012) Push the limit of wifi based localization for smartphones. In: Proceedings of the 18th annual international conference on mobile computing and networking (ACM MobiCom).
38. Liu J, Wang Y, Chen Y, Yang J, Chen X, Cheng J (2015) Tracking vital signs during sleep leveraging off-the-shelf wifi. In: Proceedings of the 16th ACM international symposium on mobile Ad Hoc networking and computing.
39. Liu H, Wang Y, Zhou A, He H, Wang W, Wang K, Pan P, Lu Y, Liu L, Ma H (2020) Real-time arm gesture recognition in smart home scenarios via millimeter wave sensing. Proc ACM Interact Mob Wearable Ubiquitous Technol 4(4):1–28
40. López-Medina MA, Espinilla M, Cleland I, Nugent C, Medina J (2020) Fuzzy cloud-fog computing approach application for human activity recognition in smart homes. J Intell Fuzzy Syst 38(1):709–721
41. Mahmood Khan U, Kabir Z, Hassan SA (2017) Wireless health monitoring using passive wifi sensing. In: 2017 13th international wireless communications and mobile computing conference (IWCMC)
42. Meng Y, Yi S-H, Kim H-C (2019) Health and wellness monitoring using intelligent sensing technique. J Inf Process Syst 15(3):478–491
43. Moosavi-Dezfooli S-M, Fawzi A, Frossard P (2016) Deepfool: a simple and accurate method to fool deep neural networks. In Proceedings of the IEEE conference on computer vision and pattern recognition, pp 2574–2582
44. Moosavi-Dezfooli S-M, Fawzi A, Fawzi O, Frossard P (2017) Universal adversarial perturbations. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 1765–1773
45. Mpitziopoulos A, Gavalas D, Konstantopoulos C, Pantziou G (2009) A survey on jamming attacks and countermeasures in wsns. IEEE Commun Surv Tutor 11(4):42–56
46. Mun M, Estrin D, Burke J, Hansen M (2008) Parsimonious mobility classification using gsm and wifi traces. In: Proceedings of the fifth workshop on embedded networked sensors (HotEmNets), pp 1–5
47. Oguntala GA, Abd-Alhameed RA, Ali NT, Hu Y, Noras JM, Eya NN, Elfergani I, Rodriguez J (2019) Smartwall: novel rfid-enabled ambient human activity recognition using machine learning for unobtrusive health monitoring. IEEE access
48. Ozbulak U, Vandersmissen B, Jalalvand A, Couckuyt I, Van Messem A, De Neve W (2021) Investigating the significance of adversarial attacks and their relation to interpretability for radar-based human activity recognition systems. Comput Vis Image Underst 202:103111
49. Palipana S, Salami D, Leiva LA, Sigg S (2021) Pantomime: Mid-air gesture recognition with sparse millimeter-wave radar point clouds. Proc ACM Interact Mob Wearable Ubiquitous Technol 5(1):1–27
50. Pegoraro J, Meneghello F, Rossi M (2020) Multiperson continuous tracking and identification from mm-wave micro-doppler signatures. IEEE Trans Geosci Remote Sens 59(4):2994–3009
51. Pries R, Yu W, Fu X, Zhao W (2008) A new replay attack against anonymous communication networks. In 2008 IEEE international conference on communications, pp 1578–1582
52. Pu Q, Gupta S, Gollakota S, Patel S (2013) Whole-home gesture recognition using wireless signals. In: Proceedings of the 19th annual international conference on Mobile computing & networking, pp 27–38

53. Qin Y, Carlini N, Cottrell G, Goodfellow I, Raffel C (2019) Imperceptible, robust, and targeted adversarial examples for automatic speech recognition. In: International conference on machine learning, pp 5231–5240
54. Rachuri KK, Musolesi M, Mascolo C, Rentfrow PJ, Longworth C, Aucinas A (2010) Emotionsense: a mobile phones based adaptive platform for experimental social psychology research. In: Proceedings of the 12th ACM international conference on Ubiquitous computing (ACM Ubicomp)
55. Ren Y, Lu J, Beletchi A, Huang Y, Karmanov I, Fontijne D, Patel C, Xu H (2021) Hand gesture recognition using 802.11 ad mmwave sensor in the mobile device. In: 2021 IEEE wireless communications and networking conference workshops (WCNCW)
56. Rong Y, Bliss DW (2018) Direct rf signal processing for heart-rate monitoring using uwb impulse radar. In: 2018 52nd asilomar conference on signals, systems, and computers
57. Santhalingam PS, Hosain AA, Zhang D, Pathak P, Rangwala H, Kushalnagar R (2020) mmasl: Environment-independent asl gesture recognition using 60 ghz millimeter-wave signals. Proc ACM Interact Mob Wearable Ubiquitous Technol 4(1):1–30
58. Sengupta A, Jin F, Zhang R, Cao S (2020) mm-pose: real-time human skeletal posture estimation using mmwave radars and CNNS. IEEE Sensors J 20(17):10032–10044
59. Shastri A, Valecha N, Bashirov E, Tataria H, Lentmaier M, Tufvesson F, Rossi M, Casari P (2022) A review of millimeter wave device-based localization and device-free sensing technologies and applications. IEEE Commun Surv Tutor 24(3):1708–1749
60. Sheen DM, McMakin DL, Hall TE (2007) Near field imaging at microwave and millimeter wave frequencies. In: 2007 IEEE/MTT-S international microwave symposium.
61. Shi C, Lu L, Liu J, Wang Y, Chen Y, Yu J (2022) mpose: Environment-and subject-agnostic 3d skeleton posture reconstruction leveraging a single mmwave device. Smart Health 23:100228
62. Sim JM, Lee Y, Kwon O (2015) Acoustic sensor based recognition of human activity in everyday life for smart home services. Int J Distrib Sensor Netw 11(9):679123
63. Singh AD, Sandha SS, Garcia L, Srivastava M (2019) Radhar: human activity recognition from point clouds generated through a millimeter-wave radar. In: Proceedings of the 3rd ACM workshop on millimeter-wave networks and sensing systems, pp 51–56
64. Sozinov K, Vlassov V, Girdzijauskas S (2018) Human activity recognition using federated learning. In: 2018 IEEE ISPA/IUCC/BDCloud/SocialCom/SustainCom, pp 1103–1111
65. Steele BG, Belza B, Cain K, Warms C, Coppersmith J, Howard J, et al (2003) Bodies in motion: monitoring daily activity and exercise with motion sensors in people with chronic pulmonary disease. J Rehabil Res Develop 40(5; SUPP/2):45–58
66. Sun Z, Balakrishnan S, Su L, Bhuyan A, Wang P, Qiao C (2021) Who is in control? practical physical layer attack and defense for mmwave-based sensing in autonomous vehicles. IEEE Trans Inf Forensics Secur 16:3199–3214
67. Sze V, Chen Y-H, Yang T-J, Emer JS (2017) Efficient processing of deep neural networks: A tutorial and survey. Proc IEEE 105(12):2295–2329
68. Szegedy C, Zaremba W, Sutskever I, Bruna J, Erhan D, Goodfellow I, Fergus R (2013) Intriguing properties of neural networks. Preprint, arXiv:1312.6199
69. Texas Instruments (2024) DCA1000 Evaluation Module for Real-Time Data Capture and Streaming. https://www.ti.com/tool/DCA1000EVM. Accessed 23 Mar 2024
70. Texas Instruments (2024) IWR1443 BoosterPack™ evaluation module for single-chip 77GHz mmWave sensor. https://www.ti.com/tool/IWR1443BOOST. Accessed 23 Mar 2024
71. Tiwari G, Gupta S (2021) An mmwave radar based real-time contactless fitness tracker using deep cnns. IEEE Sensors J 21(15):17262–17270
72. Wang Y, Liu J, Chen Y, Gruteser M, Yang J, Liu H (2014) E-eyes: device-free location-oriented activity identification using fine-grained wifi signatures. In: Proceedings of the 20th annual international conference on mobile computing & networking.
73. Wang Y, Liu J, Chen Y, Gruteser M, Yang J, Liu H (2014) E-eyes: device-free location-oriented activity identification using fine-grained wifi signatures. In: Proceedings of the 20th annual international conference on Mobile computing and networking, pp 617–628

74. Wang X, Yang C, Mao S (2017) Phasebeat: Exploiting csi phase data for vital sign monitoring with commodity wifi devices. In: 2017 IEEE 37th international conference on distributed computing systems (ICDCS)
75. Wang F, Feng J, Zhao Y, Zhang X, Zhang S, Han (2019) Joint activity recognition and indoor localization with wifi fingerprints. IEEE Access 7:80058–80068
76. Wang J, Zhao Y, Ma X, Gao Q, Pan M, Wang H (2020) Cross-scenario device-free activity recognition based on deep adversarial networks. IEEE Trans Veh Technol 69(5):5416–5425
77. Wang J, Zhang L, Wang C, Ma X, Gao Q, Lin B (2020) Device-free human gesture recognition with generative adversarial networks. IEEE Internet Things J 7(8):7678–7688
78. Wang Y, Liu H, Cui K, Zhou A, Li W, Ma H (2021) m-activity: accurate and real-time human activity recognition via millimeter wave radar. In ICASSP 2021-2021 IEEE international conference on acoustics, speech and signal processing (ICASSP).
79. Wu C, Zhang F, Wang B, Liu KJR (2020) mmtrack: passive multi-person localization using commodity millimeter wave radio. In: IEEE INFOCOM 2020
80. Xia K, Wang H, Xu M, Li Z, He S, Tang Y (2020) Racquet sports recognition using a hybrid clustering model learned from integrated wearable sensor. Sensors 20(6):1638
81. Xie Y, Shi C, Li Z, Liu J, Chen Y, Yuan B (2020) Real-time, universal, and robust adversarial attacks against speaker recognition systems. In: ICASSP 2020-2020 IEEE international conference on acoustics, speech and signal processing (ICASSP), pp 1738–1742
82. Xie Y, Jiang R, Guo X, Wang Y, Cheng J, Chen Y (2022) mmeat: Millimeter wave-enabled environment-invariant eating behavior monitoring. Smart Health 23:100236
83. Xie Y, Jiang R, Guo X, Wang Y, Cheng J, Chen Y (2022) mmfit: Low-effort personalized fitness monitoring using millimeter wave. In: 2022 international conference on computer communications and networks (ICCCN), pp 1–10
84. Xue H, Ju Y, Miao C, Wang Y, Wang S, Zhang A, Su L (2021) mmmesh: towards 3d real-time dynamic human mesh construction using millimeter-wave. In: Proceedings of the 19th annual international conference on mobile systems, applications, and services
85. Yang Z, Pathak PH, Zeng Y, Liran X, Mohapatra P (2016) Monitoring vital signs using millimeter wave. In: Proceedings of the 17th ACM international symposium on mobile ad hoc networking and computing, pp 211–220
86. Yang X, Liu J, Chen Y, Guo X, Xie Y (2020) Mu-id: multi-user identification through gaits using millimeter wave radios. In: IEEE INFOCOM 2020-IEEE conference on computer communications, pp 2589–2598
87. Yang Z, Zhao Y, Yan W (2020) Adversarial vulnerability in doppler-based human activity recognition. In: 2020 international joint conference on neural networks (IJCNN), pp 1–7
88. Zeng Y, Pathak PH, Yang Z, Mohapatra P (2016) Poster abstract: human tracking and activity monitoring using 60 ghz mmwave. In: 2016 15th ACM/IEEE international conference on information processing in sensor networks (IPSN)
89. Zhang J, Wu F, Wei B, Zhang Q, Huang H, Shah SW, Cheng J (2020) Data augmentation and dense-lstm for human activity recognition using wifi signal. IEEE Internet Things J 8(6):4628–4641
90. Zhang L, Hua Y, Cotton SL, Yoo SK, Da Silva CRCM, Scanlon WG (2020) An rss-based classification of user equipment usage in indoor millimeter wave wireless networks using machine learning. IEEE Access 8:14928–14943
91. Zhao P, Lu CX, Wang J, Chen C, Wang W, Trigoni N, Markham A (2019) mid: Tracking and identifying people with millimeter wave radar. In: 2019 15th international conference on distributed computing in sensor systems (DCOSS), pp 33–40
92. Zhou B, Yang J, Li Q (2019) Smartphone-based activity recognition for indoor localization using a convolutional neural network. Sensors 19(3):621

Adversarial Machine Learning for Wireless Localization

Tianya Zhao, Xuyu Wang, Shiwen Mao, Slobodan Vucetic, and Jie Wu

1 Introduction

Location-based services gain extensive popularity in current and future lives, such as autonomous driving [14], epidemic tracking [31], indoor navigation [78], and smart cities [62]. Global Positioning System (GPS) is widely used in outdoor navigation and powers many public maps, such as Google Maps [24] and Bing Maps [10]. However, the effectiveness of GPS is greatly hindered when it comes to indoor environments, primarily due to its vulnerability to occlusion. As wireless communication technologies develop rapidly, wireless signals such as Wi-Fi, LoRa, LTE, and 5G have become ubiquitous in our daily lives and work environments. These wireless technologies enable accurate indoor and outdoor localization, compensating for the limitations of GPS performance in indoor settings.

Due to the rapid advancements in machine learning technologies, extensive research has been conducted on fingerprint-based positioning systems. Table 1 summarizes some papers on machine learning-based wireless localization. Compared to measuring time of arrival (TOA), time difference of arrival (TDOA), or angle of arrival (AOA), fingerprint-based localization systems demonstrate the advantage of requiring special devices. The process of fingerprint-based positioning typically

T. Zhao · X. Wang
Knight Foundation School of Computing and Information Sciences, Florida International University, Miami, FL, USA
e-mail: tzhao010@fiu.edu; xuywang@fiu.edu

S. Mao (✉)
Department of Electrical and Computer Engineering, Auburn University, Auburn, AL, USA
e-mail: smao@ieee.org

S. Vucetic · J. Wu (✉)
Department of Computer and Information Sciences, Temple University, Philadelphia, PA, USA
e-mail: vucetic@temple.edu; jiewu@temple.edu

Y. Chen et al. (eds.), *Network Security Empowered by Artificial Intelligence*,
Advances in Information Security 107, https://doi.org/10.1007/978-3-031-53510-9_8

Table 1 Technologies used in recent wireless localization solutions

Wi-Fi	LTE/5G	Bluetooth	Others
[64–67],	[42, 52, 81, 83],	[20, 32, 56, 77]	[36, 46, 74, 84]
[1, 53, 68, 76],	[35, 44, 48, 69],	[30, 33, 55, 58]	[5, 37, 50, 51]
[9, 15, 26, 38],	[18, 54, 63, 80],	[28, 40]	[34, 73]
[8, 79]	[16, 49, 57]		

involves two stages: the offline stage and the online stage. For the offline stage, a large database is constructed from a comprehensive measurement of the field, and the machine learning models are trained on the database. Once the models are trained, they can be used to predict locations by comparing the received test data with the information stored in the database.

Received signal strength (RSS) is widely employed as fingerprints because of its simplicity. Radar is one of the pioneering fingerprint-based positioning systems, which deploys K-nearest neighbors (KNN) to estimate locations based on the RSS dataset [6]. Besides, support vector machine (SVM) has been employed in an RSS-based location determination system [72]. In addition to the aforementioned shallowing machine learning algorithms, deep neural networks (DNNs) have been employed to boost the efficiency of fingerprint-based positioning systems. Multilayer perceptron (MLP) and convolutional neural networks (CNNs) have been deployed to aid in accurate position estimation within indoor localization systems [22, 61, 79].

While RSS is simple to use, it does have certain drawbacks. One significant limitation is its inability to reflect the multipath effect. This means that even slight changes in multipath components can result in substantial variations in RSS. Furthermore, RSS only provides coarse channel information as it represents the sum of powers from all received signals. Consequently, the precision of RSS-based localization is diminished due to these inherent limitations.

Channel state information (CSI) provides fine-grained channel information and is widely used in deep learning-based localization systems. DeepFi first deploys DNNs with a substantial amount of CSI data for indoor localization [64, 67]. In addition to using the amplitude information of CSI, PhaseFi leverages calibrated phase data to train DNNs in two distinct indoor environments [65, 66]. Moreover, BiLoc deploys a bi-modal design by utilizing AOAs and CSI average amplitudes as inputs [68]. DyLoc transforms CSI into angle delay profiles (ADPs) and employs recurrent neural networks (RNNs) to estimate precise locations [29].

Undoubtedly, leveraging powerful DNNs in fingerprint-based localization systems can yield impressive performance. However, it is significant to address critical concerns regarding the security and robustness of such systems. The inherent black-box nature of DNNs and the potential utilization of third-party resources during the training process can introduce vulnerabilities. In fact, research focusing on adversarial machine learning has been ongoing for a long time. In domains such as computer vision and natural language processing, numerous studies have

demonstrated that attacks can fool machine learning models, including linear classifiers [17] and DNNs [23].

In contrast to the extensive studies conducted on adversarial machine learning in the aforementioned domains, research in wireless domains is currently in an early stage but holds equal significance. For instance, Bahramali et al. design adversarial perturbations that are resilient against removal in DNN-based wireless communication systems [7]. Furthermore, their work demonstrates that these perturbations can significantly degrade system performance, even in the presence of defense mechanisms. WiCAM generated adversarial examples with limited perturbations that do not affect Wi-Fi communications [75]. However, these carefully crafted examples can significantly disrupt DNN-based WiFi sensing applications, such as human activity recognition [4], fall detection, and gesture recognition.

The above studies highlight the importance of addressing vulnerabilities of machine learning-based wireless applications against potential adversaries. In this chapter, we introduce adversarial machine learning for wireless localization. Section 2 first presents various localization systems that rely on different wireless technologies and are designed for diverse application scenarios. Section 3 discusses different attacks in machine learning-based wireless positioning systems. Section 4 concludes the chapter by presenting our view on adversarial machine learning for wireless localization and discussing future work.

2 Machine Learning-Based Localization

In the artificial intelligence (AI) era, machine learning has been widely deployed in various disciplines such as natural language processing, computer vision, robotics, and engineering. Compared to classic machine learning methods, deep learning is a branch of machine learning, which is more powerful but hard to explain. In this section, we will introduce several machine learning-based localization systems using different wireless technologies.

Figure 1 gives a general architecture of the fingerprinting-based localization system. Regardless of the type of wireless signal utilized, a sequence of processing procedures yields data specific to various locations. During the offline phase, we employ this location-specific data to train our deep learning model. Subsequently, in the online phase, new test data is processed and fed into the well-trained model to output the predicted locations.

2.1 Wi-Fi-Based Localization

Wi-Fi is commonly used in homes, offices, public areas, and various other environments to provide wireless internet access and enable wireless communication between devices such as computers, smartphones, tablets, and smart home devices.

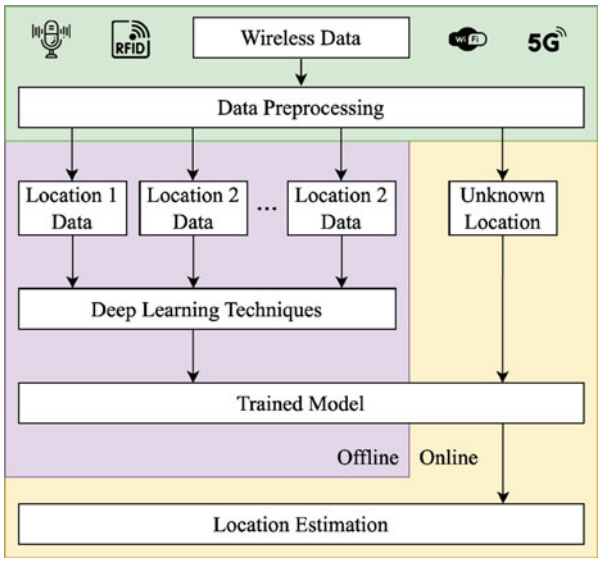


Fig. 1 An overview of the architecture of a fingerprinting-based localization system

As it is ubiquitous, Wi-Fi-based localization is popular among other wireless technologies. In this section, we discuss Wi-Fi-based positioning systems using RSSI and CSI as fingerprints.

2.1.1 RSS-Based

RSS measures the power level of a signal received by a wireless device, such as a Wi-Fi receiver or a mobile phone. RSS typically represents the intensity or magnitude of the signal as it arrives at the receiver. In the context of localization using Wi-Fi fingerprinting, RSS values are collected from different reference points to create a fingerprint database. These RSS values provide information about the signal strength at specific locations, which can be used to estimate the position of a device in the environment.

UJIIndoorLoc [60] is a publicly accessible Wi-Fi fingerprint-based indoor localization database. As shown in Fig. 2, it covers three distinct buildings, each consisting of four or five floors. To create the data, 25 different mobile models are used to take measurements from 933 different reference points by more than 20 users. This comprehensive dataset consists of 21,049 samples, with each sample containing 520 RSS values. Due to its inclusion of essential information such as building, floor, and position data, the UJIIndoorLoc dataset can be used for classification tasks and indoor localization.

In [13], several machine learning algorithms are discussed for classification tasks on the UJIIndoorLoc dataset. Although they successfully achieve high performance

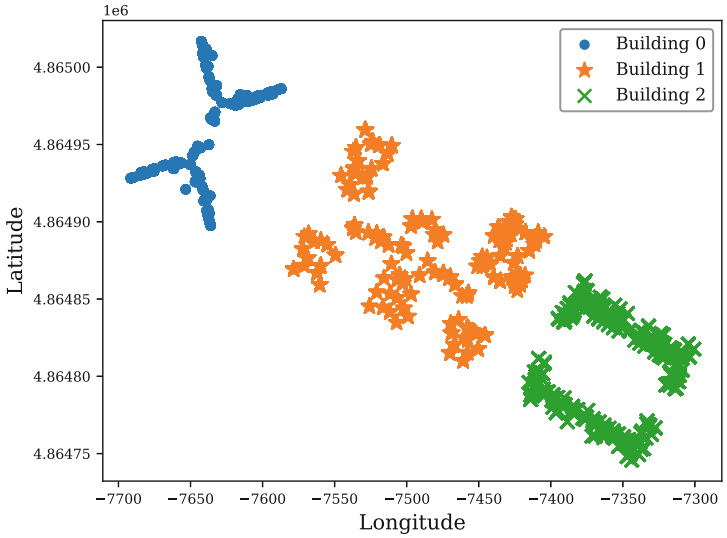


Fig. 2 UJIIndoorLoc data distribution

in building classification, floor classification, and region classification, they fail to predict accurate positions. To prove the effectiveness of machine learning for localization, we simply employ SVM and MLP for position prediction. For the localization problem, we usually take root mean square error (RMSE) as the evaluation metric. RMSE is a practical choice since it corresponds to the Euclidean distance in distance calculations.

In the case of SVM, the RMSE value remains below 10, while for MLP, the average RMSE value stays below 2. It is important to note that these results represent baseline performance, and there is potential for further improvement by employing additional training tricks or exploring different models. Nevertheless, these initial findings suggest that achieving good precision in complex indoor localization tasks is feasible.

2.1.2 CSI-Based

CSI consists of subcarrier-level measurements of orthogonal frequency division multiplexing (OFDM) channels. Especially now the collection of CSI can be carried out on commodity Wi-Fi network interface cards (NIC), such as Intel Wi-Fi Link 5300 NIC [27]. Therefore, numerous fingerprinting systems based on CSI have been proposed and demonstrated to achieve remarkable precision. In our previous work, DeepFi employed a DNN to effectively learn extensive CSI data obtained from three antennas and 30 subcarriers [64, 67].

The channel model in the frequency domain can be expressed as

$$\overrightarrow{Y} = CSI \cdot \overrightarrow{X} + \overrightarrow{N}, \tag{1}$$

where $\overrightarrow{X}$ and $\overrightarrow{Y}$ refer to the transmitted and received signal vectors, and $\overrightarrow{N}$ represents the additive white Gaussian noise. CSI denotes the channel's frequency response. The channel frequency response CSI_i of subcarrier i is a complex value, defined as

$$CSI_i = \mathcal{I}_i + j\mathcal{Q}_i = |CSI_i| \, exp(j \angle CSI_i), \tag{2}$$

where $\mathcal{I}_i$ and $\mathcal{Q}_i$ denote the in-phase and quadrature components, respectively. $|CSI_i|$ and $\angle CSI_i$ denote the amplitude response and phase response of subcarrier i, respectively.

DeepFi [64, 67] only employs the amplitude responses for fingerprinting, primarily due to the presence of hardware imperfections that result in measured phase errors. These errors are mainly caused by two factors. First, the presence of carrier frequency offset (CFO) caused by the down-converter in the receiver signal, as perfect synchronization of the central frequencies between the receiver and transmitter is unattainable. Second, sampling frequency offset (SFO) is introduced by the ADC due to unsynchronized clocks. Additionally, in the case of SFO, the measured phase errors vary across different subcarriers. Therefore, the raw phase information has limited use in localization.

Despite the aforementioned limitations of raw phase information, PhaseFi [65, 66] implements a linear transformation to alleviate the impact of random phase offsets. When compared to amplitude, the phase of a signal with periodic changes over the propagation distance demonstrates greater robustness when encountering obstacles. Furthermore, the calibrated phase information tends to be more stable for a given position.

In addition to using the CSI amplitude and the processed phase independently, BiLoc [68] takes advantage of estimated AOAs and average amplitudes for deep learning-based indoor localization. In general, localization systems based on CSI can achieve greater precision because CSI represents accurate channel information and allows for elaborate processing.

2.2 5G-Based Localization

In addition to Wi-Fi, 5G has emerged as a widely adopted technology. One of the key techniques within 5G is Millimeter Wave (mmWave) communications, which not only offers high data rates but also has remarkable temporal resolution and directivity.

In [70], we propose a deep convolutional Gaussian process (DCGP) based regression approach for mmWave outdoor localization. Unlike CNN, DCGP is a fully Bayesian kernel method without any neural network component, which can provide uncertainty estimation on location predictions [11]. Additionally, DCGP incorporates the convolutional structure within the deep Gaussian process, allowing it to effectively identify hierarchical combinations of local features in the mmWave dataset. This capability proves particularly valuable in non-line-of-sight (NLOS) environments.

In this case, we employ an open-source mmWave dataset generated using ray-tracing software within the New York University area, covering a spatial extent of 400 m $\times$ 400 m [21]. This dataset comprises beamforming images from a total of 160,801 two-dimensional positions. For offline training, our system undergoes 550 epochs of training to obtain a highly accurate model. In the online prediction phase, our system achieves impressive results, yielding a mean distance error of 2.79 m for outdoor localization.

2.3 Voice-Based Localization

In the context of smart homes, devices are often commanded verbally to execute desired operations. Enhancing smart speakers' performance and enabling numerous new IoT applications, voice localization with microphone arrays is the focus of our exploration, particularly in terms of a voice fingerprinting-based indoor localization system using an off-the-shelf microphone array.

In [46], we employ the short-time Fourier transform (STFT) on audio data, converting it into spectrogram images that serve as inputs for the DNNs. During the offline training phase, we leverage transfer learning and fine-tune the model with new audio data to expedite the training process. During the online phase, we introduce a top-K probabilistic methodology for location prediction.

In the experimental setup, our system is tested in two distinct indoor environments (dimensions: 10 m $\times$ 10 m and 10 m $\times$ 5 m). The microphone array device is placed at the center of the room for voice data collection. Assisted by deep convolutional neural networks (DCNNs), our system is capable of yielding average error margins of around 1.5 m in these two environments.

3 Adversarial Machine Learning on Localization

Adversarial machine learning poses significant challenges to the reliability and security of machine learning models. In the image classification task, Szegedy et al. [59] first discover an intriguing weakness of DNNs. Despite their impressive high accuracy, these DNNs exhibit surprising susceptibility to adversarial attacks, which manifest as slight modifications to images that remain undetectable to the human

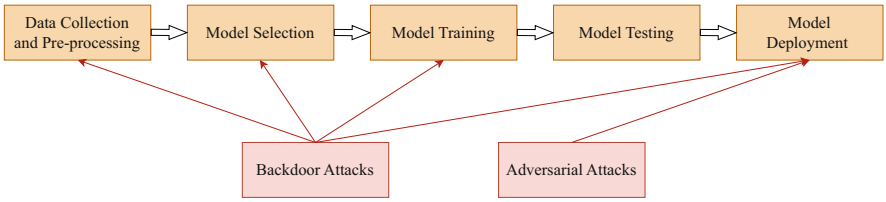


Fig. 3 Attacks in the machine learning pipeline

eyes. These attacks have the potential to completely change the predictions made by a neural network classifier for a given image. Moreover, the attacked models report a high confidence score for the wrong prediction and a single perturbation to an image can deceive various DNNs simultaneously.

We have implemented accurate localization systems with the help of deep learning techniques by using the different kinds of wireless signals in Sect. 2. These systems exhibit excellent accuracy in localization, owing to the exceptional capabilities of deep learning models. Considering the aforementioned vulnerability issue, it is crucial to conduct thorough research on the security aspects of these models. Failing to address this critical issue may result in the deterioration of deep learning-based localization systems, leading to a loss of their current capabilities due to even minor perturbations.

In this chapter, we focus on two different types of attacks: backdoor attacks and adversarial attacks. As shown in Fig. 3, backdoor attacks can be applied at multiple stages throughout the machine learning pipeline, excluding the model testing phase. For adversarial attacks, the attacker creates a perturbation that is specific to the given input in the model deployment stage.

3.1 Backdoor Attack

The concept of backdoor attacks on deep learning is initially introduced in BadNets [25]. The training process for these attacks consists of two primary steps. First, a set of poisoned images is generated by embedding a backdoor trigger into selected benign images as shown in Fig. 4. This process creates poisoned samples that are associated with target labels specified by the attacker. Second, the poisoned training set is formed by combining both the poisoned and benign samples. Consequently, the trained DNN becomes infected, exhibiting performance similar to a model trained solely on benign samples when evaluated on benign testing samples. However, if a poisoned image contains the previously defined trigger, its prediction will be changed to the target label specified by the attacker.

In the majority of backdoor attack scenarios, implementing attacks requires inserting a trigger before the training phase. This process has become increasingly possible due to the extensive use of cloud platforms, pre-trained models, and

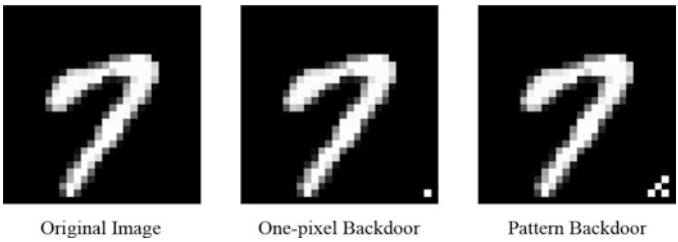


Fig. 4 Example of two backdoor triggers used in BadNets

publicly available datasets within the contemporary deep learning landscape. As illustrated in Fig. 3, malicious attackers can introduce harmful datasets in the data collection step, and distribute problematic pre-trained models during model selection, thereby impairing the performance of inference tasks. Additionally, attackers can invade the cloud infrastructure to manipulate gradients throughout the model training process, resulting in disruptions to the model's performance. Considering these circumstances, backdoor attacks can be categorized into three primary types: poisoning-based backdoor attacks, weights-oriented backdoor attacks, and structure-modified backdoor attacks [39].

3.1.1 Backdoor Attack on 5G-Based Localization

This section focuses on discussing the application of poisoning-based backdoor attacks on both indoor and outdoor localization systems using 5G massive MIMO technology. We have designed experiments on the DeepMIMO dataset [3, 29]. The input for our CNNs is ADP data, which is simply a linear transformation of the CSI. The DeepMIMO outdoor scenario number 1 (O1) at 3.5 GHz band and indoor scenario number 3 (I3) at 60 GHz are deployed as outdoor and indoor environments, respectively. For the outdoor environment, a single base station is equipped with a uniform linear array (ULA) with 64 antennas. The data generation process involves generating 199,100 data points by varying the locations from row 1 to row 1,100. The range of position coordinates varied from (242.4, 297.2) to (278.4, 517.0). The indoor environment simulates a conference room with dimensions 10 m $\times$ 11 m. The position coordinates range from (26.34, 6.18) to (27.54, 11.67). When approaching localization as a regression problem, we designate the target coordinates for backdoor attacks as (200, 200) for outdoor scenarios and (0, 0) for indoor scenarios.

An overview of the backdoor attacks against the DNN-based 5G massive MIMO localization system is depicted in Fig. 5 [82]. The poisoned ADP data is generated by directly injecting a trigger into the original ADP. The sole difference between the original input and the poisoned input lies in the presence of this trigger. The poisoned input constitutes only a small fraction of the original input, and its quantity can be adjusted by the attacker. We denote this fraction as the poisoning rate p.

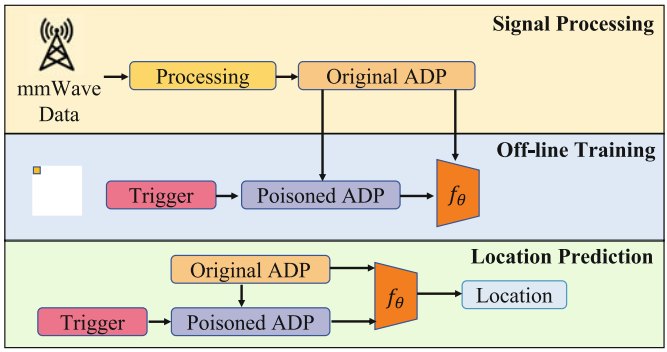


Fig. 5 Backdoor attacks on DNN-based mmWave/massive MIMO localization systems

During the training process, the original ADP data is employed as benign input to optimize the model's performance, aiming for accurate position predictions. Conversely, the poisoned input is introduced to deceive the model, causing it to generate inaccurate position predictions. Once the training process is finished, the model acquires the capability to predict positions. When fed with benign inputs, the model accurately determines the location of the target device. However, the injection of triggers into the inputs significantly diminishes the model's performance, resulting in incorrect predictions.

The objective of backdoor attacks can be defined as

$$t^* = arg \max_t d(F_\theta(x_i + t), F_\theta(x_i)), \quad s.t. \ \mid t \mid \leq \varepsilon, \tag{3}$$

where t represents the trigger, x_i is the ith input data, F_θ represents the CNN model, and $d(\cdot)$ denotes a distance function. In this case, the Euclidean distance is chosen as the distance function since it reflects the physical distance. The objective is to find the optimal value of t, denoted as t^*, that maximizes the distance between the CNN model's outputs when applied to the original input x_i and the perturbed input $x_i + t$. By optimizing this objective, the trigger can effectively disrupt the model. This optimization is subject to the constraint that the magnitude of t does not exceed a specified threshold ε.

As depicted in Fig. 6, We design two different types of triggers to launch backdoor attacks. The first one is the one-pixel trigger, which involves targeting a singular pixel to incite a backdoor attack. In this example, we specifically choose the pixel located at the upper left corner. The selection of trigger position can be further changed by advanced design. The second trigger is the random noise trigger. Unlike the single-pixel trigger, this trigger has the same shape as the input ADP. While these triggers are relatively simple, they form a basis for developing detailed, task-specific triggers. Optimized through thoughtful design, they can fulfill diverse task demands, improving backdoor attack efficiency.

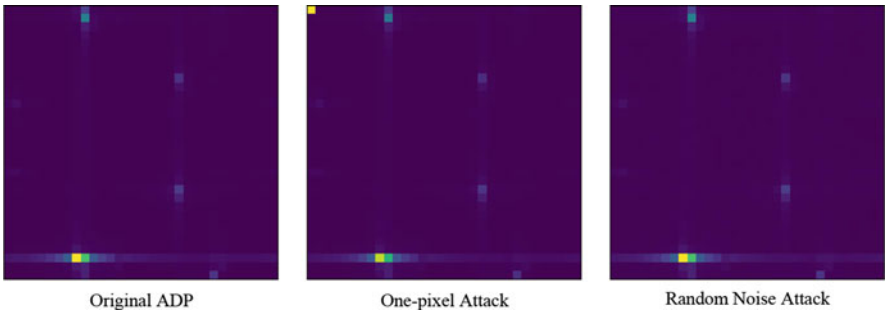


Fig. 6 Two different triggers. The one-pixel trigger is in the upper left corner, and the random noise trigger is embedded in the whole input

For the indoor backdoor attack scenario, the input ADPs exhibit a range between 0.0001 and 0.0162. To explore the impact of the one-pixel attack, we consider a set of trigger values [0.01, 0.005, 0.001, 0.0005, 0.0001]. Meanwhile, the set of poisoning rates p is set to [0.005, 0.01, 0.1]. For the outdoor case, the ADP values span a broader range from 0.0003 to 0.457. Due to the higher magnitudes compared to the indoor case, we adjust the set of trigger values to [0.1, 0.05, 0.01, 0.005, 0.001], while maintaining the poisoning rate unchanged. This selection broadly covers both the maximum and minimum values to effectively evaluate the attack's impact.

Overall, the introduction of a one-pixel trigger does not significantly affect the accuracy of the CNN in predicting locations on the benign dataset. However, it should be noted that as the trigger magnitude decreases, there is a slight increase in distance error. This suggests that a smaller trigger value has the potential to confuse the model in recognizing the trigger. If the trigger value is sufficiently large, the one-pixel attack can effectively impair the performance of the model without requiring a significant amount of poisoned samples. Conversely, when the trigger value is too small to effectively attack the model, increasing the poisoning rate can assist in enhancing the attack capability.

Although the one-pixel attack has proven to be effective in deceiving localization systems, it can be easily detected and eliminated due to the fixed location and value of the trigger. Therefore, we next introduce the random noise attacks on localization systems. As previously mentioned, the random noise trigger consists of a matrix of normally distributed noise with the same shape as the inputs. We evaluate the impact of different mean values μ and standard deviations σ of the normal distribution underlying the trigger. In all the cases, we fix the poisoning rate p to 0.01.

The results presented in Fig. 7 indicate that using the random noise attack can effectively fool the model and yield similar distance errors compared to the one-pixel attack. In the case of indoor localization, setting the mean value to 10^{-4} allows the random noise attack to degrade the system's performance without affecting the prediction of benign sample coordinates. Nevertheless, if the mean value is reduced to 10^{-5}, the noise becomes nearly indiscernible, particularly in light of the smallest original input value of 0.0001. In this scenario, the effectiveness of the random noise

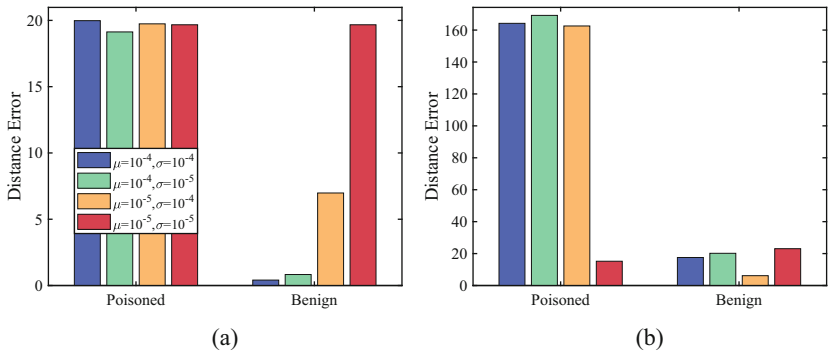


Fig. 7 The distance error of random noise attacks on indoor and outdoor localization. (**a**) Indoor localization. (**b**) Outdoor localization

attack diminishes. Although the distance errors of the poisoned samples remain constant, the system fails to accurately predict the locations of the benign samples, which contradicts the goal of the backdoor attacks.

For outdoor localization, the random noise attack continues to be effective when the mean value is set to 10^{-4} for the poisoned samples. Simultaneously, the system's ability to accurately predict the benign samples' coordinates decreases, leading to an approximately 10 m increase in distance error. Furthermore, when the mean value is adjusted to 10^{-5}, the random noise attacks behave differently. When employing a larger standard deviation of 10^{-4}, the backdoor attack successfully manipulates the system, generating imprecise predictions for the poisoned samples, while maintaining accurate predictions for the benign samples without sacrificing precision. However, when the standard deviation is reduced to 10^{-5}, the random noise attack loses its efficacy entirely in fooling the localization system.

The aforementioned results reveal a trend of increasing distance errors on the original dataset as the standard deviation decreases. This trend implies that the reduction of trigger fluctuations poses an increased challenge for the DNN model in distinguishing between benign and poisoned inputs. Due to the distinct locations between the benign inputs and the poisoned inputs, the DNN model becomes perplexed in predicting accurate locations as it struggles to recognize the triggers. As a consequence, the model fails to generate accurate predictions, irrespective of whether a trigger is present or absent.

In summary, regardless of whether it is a one-pixel attack or a random noise attack, we successfully mislead the system to generate incorrect locations for poisoned data, while still accurately predicting positions for original data. All of these processes are carried out without requiring any knowledge about the underlying model architecture. The one-pixel attack is straightforward to launch, which only requires modifying a single value of the input. In contrast, the random noise attack requires to balance between invisibility and effectiveness. In addition to these two triggers, it is also feasible to design a series of different triggers tailored to suit various situations.

3.2 Adversarial Attack

An adversarial example refers to a modified version of the original input that is intentionally perturbed to confuse a machine learning technique [2]. To generate an adversarial example, adversarial perturbations are added to the clean input. These perturbations are carefully calculated to exploit the model's sensitivity to small changes and to induce undesired predictions.

Let x_i denotes the input data (e.g., CSI tensors), and y denotes the output (e.g., object coordinates). We denote the DNN model as F_θ, where θ represents the fixed parameters of the model. The loss function is denoted as $\mathcal{L}$, which could be a cross-entropy loss for floor classification. The objective of the adversary is to degrade the performance of the DNN model by maximizing the loss function through the following optimization problem:

$$\eta = arg \max_{\eta} \mathcal{L}(F_{\theta^*}(x_i + \eta), F_{\theta^*}(x_i)), \tag{4}$$

where η represents the adversarial perturbation. It is important to note that in this setting, we cannot adjust the parameters θ^* of the model once it has finished training. This difference distinguishes adversarial attacks from backdoor attacks, where the model's parameters can be modified to incorporate a backdoor.

White-box and black-box attacks are two primary types of adversarial attacks that differ in their knowledge about the targeted machine learning model. A white-box attack refers to scenarios where the attacker has complete knowledge of the targeted model. This includes information such as the architecture of the model, the parameters such as weights and biases, the training method, and even the dataset used for training. With this comprehensive knowledge, the adversary can construct adversarial examples that are highly effective at fooling the model.

On the other hand, a black-box attack is carried out under the assumption that the attacker has no knowledge of the targeted model's architecture or parameters [45]. The adversary only has access to the model's inputs and the corresponding outputs it generates. The term *black-box* refers to the idea that the internal workings of the model remain unknown and inaccessible to the attacker, making the creation of effective adversarial examples more challenging.

Besides, adversarial training is a widely-used defense technique for improving the robustness of machine learning models against adversarial attacks. The idea behind adversarial training is to augment the training process by including adversarial examples during the training phase. This approach enables the model to learn and adapt to such perturbations, thereby increasing its resilience against future attacks.

3.2.1 Classic Attack Methods

The fast gradient sign method (FGSM) is a computationally efficient method for crafting adversarial examples [23]. The perturbation, denoted as η, is determined by

the sign of the gradient and scaled by a small constant known as the step size, as

$$\eta = \epsilon \cdot sign(\nabla_{\mathbf{x}} \mathcal{L}(\theta, \mathbf{x}, y)), \tag{5}$$

where θ represents the parameters of a well-trained DNN model; $\mathbf{x}$ is the input, and y is its corresponding label. The hyperparameter ϵ controls the magnitude of the perturbation. By utilizing the first derivative of the loss function $\mathcal{L}(\theta, \mathbf{x}, y)$ through the backpropagation algorithm, the perturbation η can be calculated.

In [43], FGSM is modified by canceling the $sign(\cdot)$ function in Eq. (5). This modified approach called the fast gradient method (FGM), serves as a generalization of FGSM, where the perturbation is given by

$$\eta = \epsilon \cdot \frac{\nabla_{\mathbf{x}} \mathcal{L}(\theta, \mathbf{x}, y)}{\|\nabla_{\mathbf{x}} \mathcal{L}(\theta, \mathbf{x}, y)\|_2}. \tag{6}$$

The perturbation is normalized by the L^2 norm $\|\cdot\|_2$ of the gradient.

Both FGSM and FGM are referred to as one-step attacks since they generate adversarial examples with a single modification based on gradient information. This attribute makes them computationally efficient and relatively straightforward to implement. One-step attacks provide a practical advantage when there are constraints on computational resources or when a quick evaluation of the model's vulnerability to adversarial examples is required.

Based on the one-step FGM, an iterative variant called projected gradient descent (PGD) is proposed in [41]. The purpose of PGD is to improve the classifier's resilience against first-order attacks. This iterative approach generates adversarial examples as follows:

$$\mathbf{x}_0^{adv} = \mathbf{x}, \tag{7}$$

$$\mathbf{x}_{N+1}^{adv} = \text{Clip}_{\mathbf{x},\epsilon} \left\{ \mathbf{x}_N^{adv} + \alpha \cdot \frac{\nabla_{\mathbf{x}} \mathcal{L}(\theta, \mathbf{x}, y)}{\|\nabla_{\mathbf{x}} \mathcal{L}(\theta, \mathbf{x}, y)\|_2} \right\}, \tag{8}$$

where the adversarial example $\mathbf{x}_N^{adv}$ is created by taking a small step, determined by the hyperparameter α, in the direction of the gradient normalized by its L^2 norm in each iteration. The value of α is typically set to ϵ/N for a given ϵ, where N represents the number of iterations. This choice ensures that the perturbations remain small and confined within the L^p ball around the original input $\mathbf{x}$. If needed, the $\text{Clip}_{\mathbf{x},\epsilon}$ projects the perturbation back into the L^p ball. PGD has been shown to be a potent adversarial attack method, surpassing the effectiveness of the one-step FGSM/FGM. However, it should be noted that the improved performance of PGD comes at the expense of reduced transferability and computational efficiency.

The PGD approach can encounter difficulties in easily reaching the global maximum since it greedily takes the direction of gradients in every iteration. To address this limitation, a momentum-based method is introduced by incorporating it into the FGSM attack. Instead of solely using the gradient in the current iteration

to update the perturbation, the momentum iterative method (MIM) leverages the gradients from previous iterations to guide the perturbation update [19]. By leveraging the memory of past gradients, MIM effectively avoids the problem of local maxima encountered in PGD. Consequently, MIM resolves the dilemma between the "underfitted" FGSM and the "overfitted" PGD, providing a more balanced and effective approach.

To generate adversarial examples using MIM, the following iterative procedure is employed:

$$\begin{cases} \mathbf{g}_0 = 0 \\ \mathbf{x}_0^{adv} = 0 \end{cases} \tag{9}$$

$$\begin{cases} \mathbf{g}_{N+1} = \mu \cdot \mathbf{g}_N + \frac{\nabla_{\mathbf{x}} \mathcal{L}(\theta, \mathbf{x}_N^{adv}, y)}{\left\| \nabla_{\mathbf{x}} \mathcal{L}(\theta, \mathbf{x}_N^{adv}, y) \right\|_2} \\ \mathbf{x}_{N+1}^{adv} = \mathbf{x}_N^{adv} + \alpha \cdot \text{sign}(\mathbf{g}_{N+1}), \end{cases} \tag{10}$$

where $\mathbf{g}_{N+1}$ contains the gradients from previous $(N-1)$ iterations with a decay factor of μ. The value of α can also be set as ϵ/N when a specific value of ϵ is given.

3.2.2 Adversarial Training

To enhance the robustness of the localization system against adversarial attacks, we employ a defense technique called adversarial training. Adversarial training aims to address the inherent vulnerabilities of machine learning models when faced with adversarial attacks. This approach involves training the models using both regular data and carefully constructed adversarial examples.

The fundamental concept behind adversarial training is to modify the original loss function by incorporating an adversarial term, thereby increasing the model's resistance to adversarial examples. The adversarial loss function can be represented as

$$\widetilde{\mathcal{L}}(\theta, \mathbf{x}, y) = \gamma \cdot \mathcal{L}(\theta, \mathbf{x}, y) + (1 - \gamma) \cdot \mathcal{L}(\theta, \mathbf{x} + \eta, y), \tag{11}$$

where $\widetilde{\mathcal{L}}$ represents the modified adversarial loss function; η represents the perturbation applied to the input and γ is a hyperparameter to control the relative importance of the loss terms for the original and adversarial examples [23].

Through the inclusion of adversarial examples in the training process, the model is encouraged to learn decision boundaries that are more robust to small adversarial perturbations. In a sense, adversarial training can be seen as a specialized form of data augmentation, focusing on generating adversarial perturbations to train the model to be robust against similar attacks.

3.2.3 Adversarial Attack on 5G-Based Localization

In Sect. 3.1.1, we discussed the impact of backdoor attacks on 5G-based localization systems. In this section, we will examine the previously mentioned adversarial attacks and adversarial training on 5G-based localization systems [12]. ADP images continue to serve as the input to the CNN and the dataset remains unchanged.

Figure 8 presents the results for the indoor scenario, where the value of epsilon ϵ is increased from 0.0005 to 0.001. It is observed that as epsilon increases, the distance errors also increase due to the introduction of larger perturbations into the ADP image. Among the three attack methods, the MIM attack yields the highest distance error, while the FGSM attack results in the lowest error. This trend aligns with the discussion in Sect. 3.2.1. Figure 8b illustrates the impact of adversarial training. Following adversarial training, the DNN model exhibits resistance to all three attacks. However, the effects of these attacks are still more significant compared to the results of the unattacked model. In particular, the MIM attack continues to produce larger distance errors than other attacks.

For the outdoor localization case, we deploy larger epsilons ranging from 0.01 to 0.05. Consistent with the findings in the indoor environment, the error grows as epsilon increases. The FGSM attack remains the least effective among the three methods. Specifically, when employing the MIM attack with an epsilon value of 0.05, the distance error can reach approximately 197 m. Compared to the indoor scenario, the DNN model exhibits significantly lower distance errors of approximately 20 m across all three attacks after deploying adversarial training.

3.2.4 Adversarial Attack on Wi-Fi-Based Localization

Section 2.1 has introduced the Wi-Fi-based localization system. This section will discuss adversarial attacks and adversarial training on RSS-based [47] and CSI-

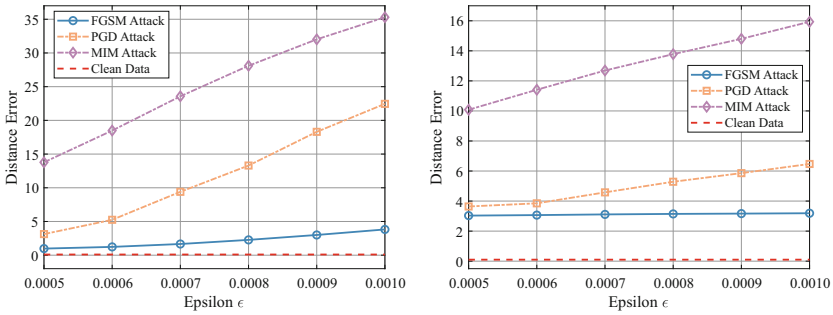


Fig. 8 Effect of adversarial attacks on 5G-based indoor localization: distance error comparison pre and post adversarial training. (**a**) Original model. (**b**) Model after adversarial training

based [71] localization. We performed experiments on the UJIIndoorLoc dataset to evaluate the RSS-based method. The three attacks mentioned above are conducted separately. All adversarial attacks are executed during the testing stage, employing various hyperparameters ϵ.

Figure 9 shows the prediction results of the DNN-based localization system and the results after being attacked by FGSM. The RSS-based positioning system exhibits commendable overall accuracy, with only a few instances of significant errors, which fall within acceptable thresholds. However, when subjected to the FGSM attack, the accuracy of the positioning system experiences a substantial decline, resulting in a significant deviation between the predicted position and the actual position. Such a substantial reduction in accuracy is completely unacceptable.

Figure 10 illustrates the outcomes of three attacks before and after undergoing adversarial training. As the value of epsilon ϵ increases, the attack becomes more powerful, resulting in less accurate location predictions. Both PGD and MIM attacks exhibit similar capabilities in deceiving the model and outperform FGSM attacks. Specifically, when employing FGSM with increasing epsilon values, the localization errors steadily rise from approximately 7 m to around 38 m. On the other hand, for

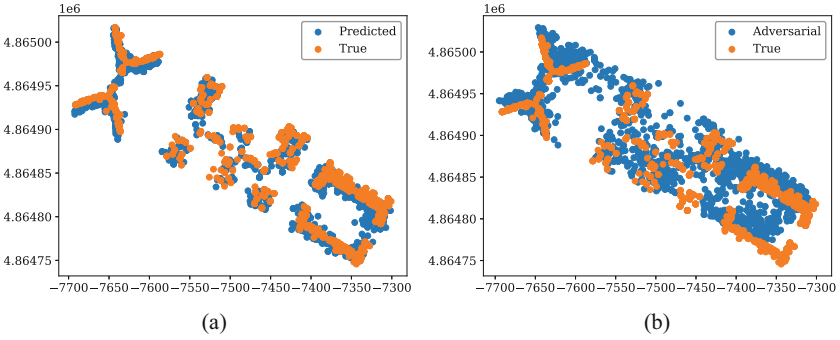


Fig. 9 Localization results with and without FGSM attack. (**a**) Prediction without attacks. (**b**) Prediction under FGSM attack ($\epsilon = 0.005$)

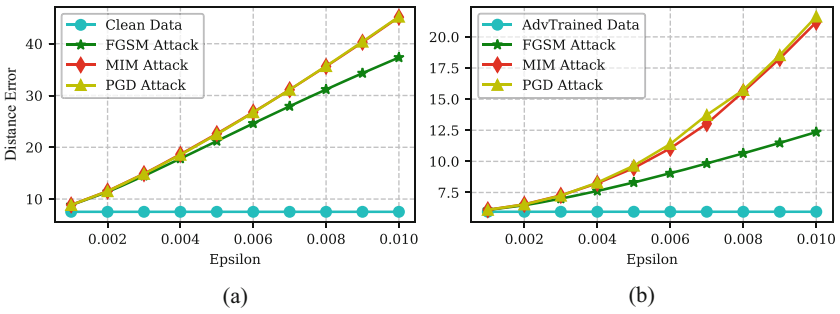


Fig. 10 Localization system performance with FGSM, PGD, and MIM attacks. (**a**) Original model under attacks. (**b**) Adversarial trained model under attacks

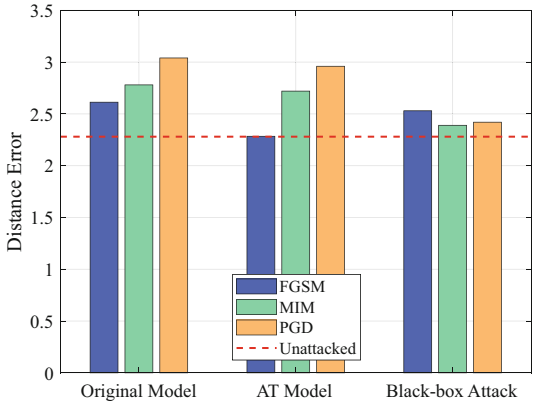


Fig. 11 Effect of white-box attacks, adversarial training, and black-box attacks on the distance error of the localization models

PGD and MIM attacks, the distance error reaches about 43 m at an epsilon value of 0.01. However, all distance errors decrease after deploying adversarial training. In the case of the FGSM attack, deploying adversarial training reduces the distance error to approximately 12 m at an epsilon value of 0.01, which is significantly lower than the original result. As for MIM and PGD attacks, adversarial training leads to a decline of about 20 m in distance errors. Despite the effectiveness of incorporating adversarial training as a defense mechanism, the distance error is still considerably higher than the error without attacks.

For the CSI-based localization system, we also studied its performance subjected to adversarial attacks and adversarial training. We propose AdvLoc, an adversarial deep learning framework for indoor localization. Our input CSI tensor encompasses three slices. Two slices are generated with the estimated AOA using the phase difference data from the three receiver antennas, while the third slice incorporates the measured CSI amplitude values. In the RSS-based localization systems, we only focus on the white-box attacks. To explore more realistic attack scenarios, we also investigate the black-box attacks in the AdvLoc system. Since we lack information about the target model, we generate adversarial perturbations based on a surrogate model. Leveraging the transferability of adversarial examples, we can mislead the black-box model using these crafted adversarial samples.

Figure 11 presents the results of white-box attacks, adversarial training, and black-box attacks in the CSI-based localization system. The indoor area is 6 m $\times$ 9 m, resulting in significantly smaller distance errors than previous experiments. We denote the model that has deployed adversarial training as AT model.

In the white-box attack scenario, the results of the three attacks exhibit a similar trend. Specifically, the FGSM attack proves to be the least effective method, while the MIM attack yields the highest distance error. This trend aligns with the discussion presented earlier in Sect. 3.2.1. After deploying adversarial training, the AT model demonstrates resilience against FGSM attacks and achieves a similar precision level compared to an unattacked model. However, adversarial training offers limited mitigation against the PGD and MIM attacks, as the distance errors for

the AT model only experience a slight reduction. In the case of black-box attacks, it is surprising to observe that the FGSM attack results in the highest distance error, although it still decreases compared to the white-box attacks. On the other hand, the PGD and MIM attacks exhibit a significant reduction in their attack effectiveness when confronted with the AT model, resulting in smaller distance errors than those observed in the FGSM attacks. This finding suggests that one-step attacks may be feasibly transferred from surrogate models, while iterative attacks require more complex designs to successfully conduct black-box attacks.

4 Conclusion

In this chapter, we presented a comprehensive review of adversarial machine learning for wireless localization systems. Initially, we introduced the concept of machine learning-based localization using various wireless technologies such as Wi-Fi, 5G, and microphone arrays. Subsequently, we delved into backdoor attacks and adversarial attacks specifically targeting Wi-Fi-based localization systems. To emphasize the significance of adversarial machine learning, we demonstrated the impact of classic attack methods on the precision of localization systems. Even a simple attack can substantially degrade the performance of the system. Despite deploying adversarial training as a defense mechanism, we found that it was not sufficient to fully restore the model to its unattacked performance.

In future work, it is crucial to explore more effective attack strategies as well as robust defense mechanisms. The relationship between attack and defense is an ongoing game. Advancements in attacks drive the evolution of defenses, creating a technological race. In order to protect wireless localization systems from potential attacks, it is crucial to develop adversarial machine learning techniques.

Acknowledgments This work is also supported in part by the NSF (CNS-2319343, CNS-2317190, CNS-2321763, CNS-2107014, CNS-2107190 and CNS-2319342). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the foundation.

References

1. Abbas M et al (2019) WiDeep: WiFi-based accurate and robust indoor localization system using deep learning. Paper presented at the 2019 IEEE international conference on pervasive computing and communications, March 2019
2. Akhtar N, Mian A (2018) Threat of adversarial attacks on deep learning in computer vision: A survey. IEEE Access 6:14410–14430. https://doi.org/10.1109/ACCESS.2018.2807385
3. Alkhateeb A (2019) DeepMIMO: a generic deep learning dataset for millimeter wave and massive MIMO applications. Preprint at https://arxiv.org/abs/1902.06435
4. Ambalkar H et al (2023) Adversarial attack and defense for WiFi-based apnea detection system. In: Proceedings of IEEE INFOCOM Posters, Hoboken, NJ, May 2023

5. Anjum M et al (2020) RSSI fingerprinting-based localization using machine learning in LoRa networks. IEEE Internet Mag 3(4):53–59. https://doi.org/10.1109/IOTM.0001.2000019
6. Bahl P, Padmanabhan VN (2020) RADAR: an in-building RF-based user location and tracking system. In: Proceedings IEEE INFOCOM, March 2000
7. Bahramali A et al (2021) Robust adversarial attacks against DNN-based wireless communication systems. In: Proceedings of the 2021 ACM SIGSAC conference on computer and communications security, November 2021
8. Bao F et al (2021) Mobintel: Passive outdoor localization via rssi and machine learning. In: 2021 17th international conference on wireless and mobile computing, networking and communications (WiMob), October 2021
9. BelMannoubi S, Touati H (2019) Deep neural networks for indoor localization using WiFi fingerprints. In: Mobile, secure, and programmable networking: 5th international conference, MSPN 2019, Mohammedia, Morocco, 23–24 April 2019
10. Bing (Online) Bing Maps. https://www.bing.com/maps
11. Blomqvist K et al (2020) Deep convolutional Gaussian processes. In: Machine learning and knowledge discovery in databases: European conference, ECML PKDD 2019, Würzburg, Germany, September 16–20, 2019
12. Boora U et al (2022) Robust massive MIMO localization using neural ode in adversarial environments. Paper presented at ICC 2022-IEEE international conference on communications, 16–20 May 2022
13. Bozkurt S et al (2015) A comparative study on machine learning algorithms for indoor positioning. Paper presented at the 2015 international symposium on innovations in intelligent SysTems and applications (INISTA), Madrid, Spain, 2–4 September 2015
14. Bresson G et al (2017) Simultaneous localization and mapping: a survey of current trends in autonomous driving. IEEE Trans Intell Veh 2(3):194–220. https://doi.org/10.1109/TIV.2017.2749181
15. Chen Z et al (2019) WiFi fingerprinting indoor localization using local feature-based deep LSTM. IEEE Syst J 14(2):3001–3010. https://doi.org/10.1109/JSYST.2019.2918678
16. Comiter MZ et al (2017) A data-driven approach to localization for high frequency wireless mobile networks. Paper presented at GLOBECOM 2017–2017 IEEE Global Communications Conference, Singapore, 4–8 December 2017
17. Dalvi N et al (2004) Adversarial classification. In: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining, August 2004
18. Decurninge A et al (2018) CSI-based outdoor localization for massive MIMO: experiments with a learning approach. In: 2018 15th international symposium on wireless communication systems (ISWCS), Lisbon, Portugal, 28–31 August 2018
19. Dong Y et al (2018) Boosting adversarial attacks with momentum. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), 18–22 June 2018
20. Faragher R, Harle R (2015) Location fingerprinting with bluetooth low energy beacons. IEEE J Sel Areas Commun 33(11): 2418–2428. https://doi.org/10.1109/JSAC.2015.2430281
21. Gante J et al (2020) Deep learning architectures for accurate millimeter wave positioning in 5G. Neural Process Lett 51(1): 487–514. https://doi.org/10.1007/s11063-019-10073-1
22. Ghozali RP, Kusuma GP (2019) Indoor positioning system using regression-based fingerprint method. Int J Adv Comput Sci Appl 10(8)
23. Goodfellow IJ (2014) Explaining and harnessing adversarial examples. Preprint at https://arxiv.org/abs/1412.6572
24. Google (Online) Google Maps. https://www.google.com/maps
25. Gu T et al (2017) Badnets: identifying vulnerabilities in the machine learning model supply chain. Preprint at https://arxiv.org/abs/1708.06733
26. Guo X et al (2019) Robust WiFi localization by fusing derivative fingerprints of RSS and multiple classifiers. IEEE Trans Ind Inf 16(5):3177–3186. https://doi.org/10.1109/TII.2019.2910664

27. Halperin D et al (2010) Predictable 802.11 packet delivery from wireless channel measurements. ACM SIGCOMM Comput Commun Rev 40(4):159–170. https://doi.org/10.1145/1851275.1851203
28. Han D et al (2018) HMM-based indoor localization using smart watches' BLE signals. Paper presented at 2018 IEEE 6th international conference on future internet of things and cloud (FiCloud), Barcelona, Spain, 6–8 August 2018
29. Hejazi F et al (2021) DyLoc: dynamic localization for massive MIMO using predictive recurrent neural networks. Paper presented at IEEE INFOCOM 2021-IEEE conference on computer communications, Vancouver, BC, Canada, 10–13 May 2021
30. Jain C et al (2021) Low-cost BLE based indoor localization using RSSI fingerprinting and machine learning. Paper presented at 2021 sixth international conference on wireless communications, signal processing and networking (WiSPNET), Chennai, India, 25–27 March 2021
31. Jeong S et al (2019) A smartphone magnetometer-based diagnostic test for automatic contact tracing in infectious disease epidemics. IEEE Access 7:20734–20747. https://doi.org/10.1109/ACCESS.2019.2895075
32. Kotrotsios K, Orphanoudakis T (2021) Accurate gridless indoor localization based on multiple Bluetooth beacons and machine learning. Paper presented at 2021 7th international conference on automation, robotics and applications (ICARA), Prague, Czech Republic, 4–6 February 2021
33. Koutris A et al (2022) Deep learning-based indoor localization using multi-view BLE signal. Sensors 22(7):2759. https://doi.org/10.3390/s22072759
34. Lazaro A et al (2021) Room-level localization system based on LoRa backscatters. IEEE Access 9:16004–16018. https://doi.org/10.1109/ACCESS.2021.3053144
35. Lee JY et al (2018) DNN-based wireless positioning in an outdoor environment. Paper presented at 2018 IEEE international conference on acoustics, speech and signal processing (ICASSP), Calgary, AB, Canada, 15–20 April 2018
36. Li L et al (2013) HIWL: an unsupervised learning algorithm for indoor wireless localization. Paper presented at 2013 12th IEEE international conference on trust, security and privacy in computing and communications, Melbourne, VIC, Australia, 16–18 July 2013
37. Li L et al (2016) Unsupervised learning of indoor localization based on received signal strength. Wirel Commun Mob Comput (16)15:2225–2237. https://doi.org/10.1002/wcm.2678
38. Li L et al (2019) SmartLoc: smart wireless indoor localization empowered by machine learning. IEEE Trans Ind Electron 67(8):6883–6893. https://doi.org/10.1109/TIE.2019.2931261
39. Li Y et al (2022) Backdoor learning: a survey. IEEE Trans Neural Netw Learn Syst. https://doi.org/10.1109/TNNLS.2022.3182979
40. Le MT (2021) Enhanced indoor localization based BLE using Gaussian process regression and improved weighted kNN. IEEE Access 9:143795–143806. https://doi.org/10.1109/ACCESS.2021.3122011
41. Madry A et al (2017) Towards deep learning models resistant to adversarial attacks. Preprint at https://arxiv.org/abs/1706.06083
42. Margolies R et al (2017) Can you find me now? Evaluation of network-based localization in a 4G LTE network. Paper presented at IEEE INFOCOM 2017-IEEE conference on computer communications, Atlanta, GA, USA, 1–4 May 2017
43. Miyato T et al (2016) Adversarial training methods for semi-supervised text classification. Preprint at https://arxiv.org/abs/1605.07725
44. Ni L et al (2017) Accurate localization using LTE signaling data. Paper presented at 2017 IEEE international conference on computer and information technology (CIT), Helsinki, Finland, 21–23 August 2017
45. Papernot N et al (2017) Practical black-box attacks against machine learning. In: Proceedings of the 2017 ACM on Asia conference on computer and communications security, Abu Dhabi, United Arab Emirates, 2–6 April 2017

46. Parmar S et al (2022) Voice fingerprinting for indoor localization with a single microphone array and deep learning. In: Proceedings of the 2022 ACM workshop on wireless security and machine learning, San Antonio, TX, USA, 19 May 2022
47. Patil M et al (2021) Adversarial attacks on deep learning-based floor classification and indoor localization. In: Proceedings of the 3rd ACM workshop on wireless security and machine learning, Abu Dhabi, United Arab Emirates, 28 June–2 July, 2021
48. Pecoraro G et al (2018) CSI-based fingerprinting for indoor localization using LTE signals. EURASIP J Adv Sig Process 1–18. https://doi.org/10.1186/s13634-018-0563-7
49. Prasad KSV et al (2018) Analytical approximation-based machine learning methods for user positioning in distributed massive MIMO. IEEE Access 6:18431–18452. https://doi.org/10.1109/ACCESS.2018.2805841
50. Purohit et al (2020) Fingerprinting-based indoor and outdoor localization with LoRa and deep learning. Paper presented at GLOBECOM 2020–2020 IEEE global communications conference, Taipei, Taiwan, 7–11 December 2020
51. Rama P, Murugan S (2020) Localization approach for tracking the mobile nodes using FA based ANN in subterranean wireless sensor networks. Neural Process Lett 51:1145–1164. https://doi.org/10.1007/s11063-019-10128-3
52. Ray A et al (2016) Localization of LTE measurement records with missing information. Paper presented at IEEE INFOCOM 2016-The 35th annual IEEE international conference on computer communications, San Francisco, CA, USA, 10–14 April 2016
53. Salamah AH et al (2016) An enhanced WiFi indoor localization system based on machine learning. Paper presented at 2016 international conference on indoor positioning and indoor navigation (IPIN), Alcala de Henares, Spain, 4–7 October 2016
54. Shi Z, Wang Y (2018). Neural network based localization using outdoor lte measurements. Paper presented at 2018 10th international conference on wireless communications and signal processing (WCSP), Hangzhou, China, 18–20 October 2018
55. Sthapit P et al (2018) Bluetooth based indoor positioning using machine learning algorithms. Paper presented at the 2018 IEEE international conference on consumer electronics-Asia (ICCE-Asia), JeJu, Korea (South), 24–26 June 2018
56. Subhan F et al (2020) Linear discriminant analysis-based dynamic indoor localization using bluetooth low energy (BLE). Sustainability 12(24):10627. https://doi.org/10.3390/su122410627
57. Sun X et al (2018) Single-site localization based on a new type of fingerprint for massive MIMO-OFDM systems. IEEE Trans Veh Technol 67(7):6134–6145. https://doi.org/10.1109/TVT.2018.2813058
58. Sun D et al (2021) Optimized cnns to indoor localization through ble sensors using improved pso. Sensors 21(6):1995. https://doi.org/10.3390/s21061995
59. Szegedy C et al (2013) Intriguing properties of neural networks. Preprint at https://arxiv.org/abs/1312.6199
60. Torres-Sospedra J et al (2014) UJIIndoorLoc: a new multi-building and multi-floor database for WLAN fingerprint-based indoor localization problems. Paper presented at the 2014 international conference on indoor positioning and indoor navigation (IPIN), Busan, Korea (South), 27–30 October 2014
61. Tuncer S, Tuncer T (2015) Indoor localization with bluetooth technology using artificial neural networks. Paper presented at the 2015 IEEE 19th international conference on intelligent engineering systems (INES), Bratislava, Slovakia, 3–5 September 2015
62. Ulusar UD et al (2020) Cognitive RF-based localization for mission-critical applications in smart cities: an overview. Comput Electr Eng 87:106780. https://doi.org/10.1016/j.compeleceng.2020.106780

63. Vieira J et al (2017) Deep convolutional neural networks for massive MIMO fingerprint-based positioning. Paper presented at the 2017 IEEE 28th annual international symposium on personal, indoor, and mobile radio communications (PIMRC), Montreal, QC, Canada, 8–13 October 2017
64. Wang X et al (2015) DeepFi: deep learning for indoor fingerprinting using channel state information. Paper presented at the 2015 IEEE wireless communications and networking conference (WCNC), New Orleans, LA, USA, 9–12 March 2015
65. Wang X et al (2015) PhaseFi: phase fingerprinting for indoor localization with a deep learning approach. In 2015 IEEE global communications conference (GLOBECOM), San Diego, CA, USA, 6–10 December 2015
66. Wang X et al (2016) CSI phase fingerprinting for indoor localization with a deep learning approach. IEEE Internet Things J (3)6:1113–1123. https://doi.org/10.1109/JIOT.2016.2558659
67. Wang X et al (2017) CSI-based fingerprinting for indoor localization: a deep learning approach. IEEE Trans Veh Technol 66(1):763–776. https://doi.org/10.1109/TVT.2016.2545523
68. Wang X et al (2017) BiLoc: Bi-modal deep learning for indoor localization with commodity 5GHz WiFi. IEEE Access 5:4209–4220. https://doi.org/10.1109/ACCESS.2017.2688362
69. Wang Y et al (2020) Is centimeter accuracy achievable for LTE-CSI fingerprint-based indoor positioning? IEEE Access 8:75249–75255. https://doi.org/10.1109/ACCESS.2020.2988387
70. Wang X et al (2021) Deep convolutional Gaussian Processes for Mmwave outdoor localization. Paper presented at ICASSP 2021-2021 IEEE international conference on acoustics, speech and signal processing (ICASSP), Toronto, ON, Canada, 6–11 June 2021
71. Wang X et al (2022) Adversarial deep learning for indoor localization with channel state information tensors. IEEE Internet Things J 9(19):18182–18194. https://doi.org/10.1109/JIOT.2022.3155562
72. Wu CL et al (2004) WLAN location determination in e-home via support vector classification. Paper presented at IEEE international conference on networking, sensing and control, Taipei, Taiwan, 21–23 March 2004
73. Wymeersch H et al (2012) A machine learning approach to ranging error mitigation for UWB localization. IEEE Trans Commun 60(6):1719–1728. https://doi.org/10.1109/TCOMM.2012.042712.110035
74. Xu H et al (2017) An RFID indoor positioning algorithm based on Bayesian probability and K-nearest neighbor. Sensors (17)8:1806. https://doi.org/10.3390/s17081806
75. Xu L et al (2022) WiCAM: imperceptible adversarial attack on deep learning based WiFi sensing. Paper presented at the 2022 19th annual IEEE international conference on sensing, communication, and networking (SECON), Stockholm, Sweden, 20–23 September 2022
76. Xue J et al (2020) A WiFi fingerprint based high-adaptability indoor localization via machine learning. China Commun 17(7):247–259. https://doi.org/10.23919/J.CC.2020.07.018
77. Yang X et al (2022) Bluetooth indoor localization with Gaussian–Bernoulli restricted Boltzmann machine plus liquid state machine. IEEE IEEE Trans Instrum Meas 71:1–8. https://doi.org/10.1109/TIM.2021.3135344
78. Zafari F et al (2019) A survey of indoor localization systems and technologies. IEEE Commun Surv Tutor 21(3):2568–2599. https://doi.org/10.1109/COMST.2019.2911558
79. Zhang W et al (2016) Deep neural networks for wireless localization in indoor and outdoor environments. Neurocomputing 194:279–287. https://doi.org/10.1016/j.neucom.2016.02.055
80. Zhang H et al (2019) Fingerprint-based localization using commercial LTE signals: a field-trial study. Paper presented at the 2019 IEEE 90th vehicular technology conference, Honolulu, HI, USA, 22–25 September 2019
81. Zhang Y et al (2019) DeepLoc: deep neural network-based telco localization. In: Proceedings of the 16th EAI international conference on mobile and ubiquitous systems: computing, networking and services, Houston, TX, USA, 12–14 November 2019
82. Zhao T et al (2023) Backdoor attacks against deep learning-based massive MIMO localization. In: Proceedings of GLOBECOM 2023-IEEE global communications conference, Kuala Lumpur, Malaysia, Dec. 2023.

83. Zhu F et al (2016) City-scale localization with telco big data. In: Proceedings of the 25th ACM international on conference on information and knowledge management, Indianapolis, IN, USA, 24–28 October 2016
84. Zou H et al (2013) An RFID indoor positioning system by using weighted path loss and extreme learning machine. Paper presented at 2013 IEEE 1st international conference on Cyber-physical systems, networks, and applications (CPSNA), Taipei, Taiwan, 19–20 August 2013

Localizing Spectrum Offenders Using Crowdsourcing

Frost Mitchell, J. Phillip Smith, Shamik Sarkar, Neal Patwari, Aditya Bhaskara, and Sneha Kumar Kasera

1 Introduction

Transmitter localization is the task of precisely locating radio-frequency (RF) transmitters using information from sensors distributed over an area. The increasing affordability of software-defined radios (SDRs) such as the HackRF or Flipper Zero has made it convenient for potential wrongdoers to illegally transmit in protected spectrum. Additionally, inexpensive jamming devices that are widely available pose a threat to legitimate wireless communication. Authorities consistently face the painstaking task of detecting and tracking both malicious transmissions and unintentional broadcasts in prohibited bands [5, 7]. Hence, effectively addressing the challenge of transmitter localization, particularly in localizing spectrum offenders, is crucial for maintaining the effective wireless communications. Enforcing spectrum regulations requires the ability to locate unauthorized users.

Typically, transmitter localization involves a set of receivers which measure RF information such as received signal strength (RSS), angle of arrival (AoA), time difference of arrival (TDoA), or other signal characteristics. A localization algorithm uses information from these sensors to estimate the coordinates of the transmitter(s) in the area. Additionally, some techniques also estimate the transmit

F. Mitchell · J. P. Smith · A. Bhaskara · S. K. Kasera (✉)
University of Utah, Salt Lake City, UT, USA
e-mail: frost.mitchell@utah.edu; phillip.smith@utah.edu; bhaskara@cs.utah.edu; kasera@cs.utah.edu

S. Sarkar
University of California, Los Angeles, CA, USA
e-mail: shamiksarkar@g.ucla.edu

N. Patwari
Washington University in St. Louis, St. Louis, MO, USA
e-mail: npatwari@wustl.edu

Y. Chen et al. (eds.), *Network Security Empowered by Artificial Intelligence*,
Advances in Information Security 107, https://doi.org/10.1007/978-3-031-53510-9_9

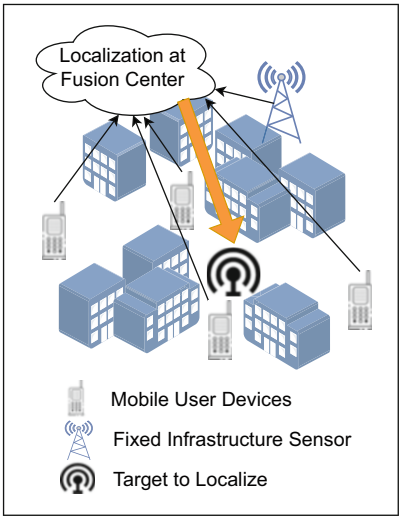


Fig. 1 An example of crowdsourcing for localization. A single fixed sensor can monitor the area, but data from mobile user devices is required to provide additional sensor density and accurately localize the target

power. To achieve accurate transmitter localization, it is essential to have dense sensor coverage of the region of interest. As research suggests, localization accuracy is directly related to the distance between sensors [24]. In many scenarios, deploying a dense network of sensors is not feasible due to high infrastructure costs.

To overcome this challenge, crowdsourcing sensor information from existing wireless devices, including mobile devices, becomes a viable solution. By leveraging measurements from a crowd of devices, sensor coverage can be substantially increases without incurring additional infrastructure costs. This idea is shown in Fig. 1, where measurements from various devices in the crowd are shared to accurately localize a transmitter.

Sourcing data from a crowd of sensors presents additional challenges. Many localization techniques require calibrated inputs, but devices in a crowd will have different hardware and sensing capabilities, and virtually no devices will be calibrated to a common reference. Moreover, the mobility of sensors in the crowd is a challenge. For example, when sensors covering a region of interest are stationary, the localization problem can be relatively straightforward, since the fixed locations of these devices provide a constant frame of reference for locating mobile transmitters. The setting becomes more challenging when both transmitters and sensors are fully mobile, providing no consistent reference. We present solutions to address these challenges in Sect. 3.

Another challenge unique to crowdsourced localization is the security risk introduced by utilizing measurements from a crowd of untrusted, anonymous users. This situation opens up the possibility of poisoning attacks, where an adversary submits false data to compromise the localization system. In Sect. 4, we present a general framework for adversarial attacks that an adversary could deploy as part of a sensor crowd. Additionally, we discuss potential defense mechanisms to counter

such attacks. We then share results from a case study on adversarial attacks and defenses in Sect. 5.

Localization using crowdsourcing depends on users sharing sensitive location information with a central service. Although privacy considerations are not the focus of this chapter, we touch briefly on solutions for preserving location privacy in Sect. 6. Finally, in Sect. 7 we explore various open problems and potential future directions for advancing crowdsourced localization techniques

1.1 Problem Setting

In our localization scenario, we consider a limited geographic area with multiple users utilizing the available spectrum. These users may be active transmitters or passive users, such as those involved in radio astronomy or remote sensing applications. Both transmitters and receivers are assumed to have unrestricted mobility within the geographic area leading to dynamic changes to the RF environment. Additionally, there may be multiple simultaneous receivers in a region.

Our localization system relies solely on RSS measurements for estimating transmitter locations. We do not consider techniques like AoA or TDoA in our setting, as AoA requires specialized receiver hardware and TDoA requires crowdsourced recording and sharing of users' raw communication signals, which has serious privacy implications. To ensure data privacy and accommodate various devices in the crowd, we focus exclusively on localization using RSS values in this chapter.

We assume the presence of a central manager or *fusion center*, responsible for collecting RSS measurements and location data from the nodes participating in the crowd. The *fusion center* facilitates the data collection process and executes a localization algorithm.

Given that reporting sensing information incurs energy and bandwidth overhead for mobile users, we assume the participants need to be incentivized for their contributions, but we do not investigate any incentive frameworks in this chapter. Similarly, while we motivate our work by assuming localization is used to enforce spectrum usage, we do not explore the specific aspects related to enforcement or interference mitigation in this chapter. Our focus is on accurate localization using crowdsourced data, both with and without the presence of adversaries in the crowd.

2 Basics of RSS Localization

One of the fundamental approaches to transmitter localization is based on RSS measurements. This approach uses the principle that signal strength varies according to the positions of both transmitter and receiver, typically decreasing with the distance and obstructions between them. The change in RSS forms the basis for

estimating the transmitter's location. This technique is particularly appealing due to its simplicity, cost-effectiveness, and the widespread availability of sensor devices.

In this section, we dive into the two primary approaches for device localization: *physics-based* methods, which use the characteristics of the environment along with physical models for signal propagation or path loss to estimate the transmitter's coordinates, and *fingerprint-based* methods, which use previously collected data to construct a reference for localization, typically in the form of a database or ML model. Although every localization technique may not fit neatly into these categories, most methods are designed based on either physical models or models derived from collected data.

We also explore the emerging paradigm of neural network-based localization, a data-driven approach which uses computer vision techniques to process sensor information and predict transmitter coordinates. This method does not come without challenges, such as loss of precision and the selection of model hyper-parameters and architecture to achieve acceptable accuracy.

2.1 Physics-Based Localization

Physics-based approaches rely on the principles of signal propagation and the physical characteristics of the environment in order to estimate device location. These models can range in precision and complexity, and typically use some environmental information, such as path loss exponents for simple path loss models or building and obstacle information for ray tracing and propagation simulations.

Physics-based localization combines the estimates of sensors at different locations to estimate the target location. The simplest possible case is shown in Fig. 2, where signal strength from a transmitter with known power is used to estimate distances to the target for localization. This becomes a significantly more challenging problem if the transmitter power or environmental characteristics are

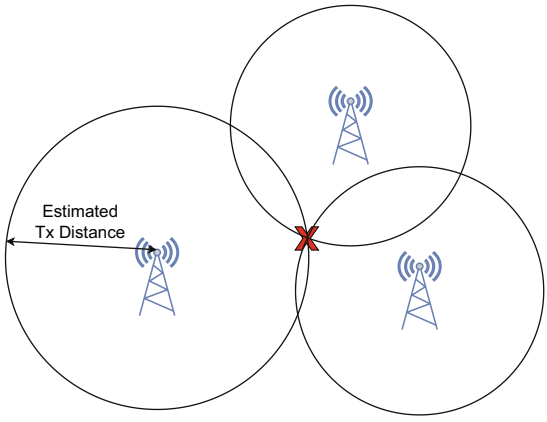


Fig. 2 If the transmitter power is known, path loss models can be used to estimate the distance from each sensor using RSS measurements. This approach suffers when the path loss model does not accurately represent propagation in the environment

unknown. To deal with the challenge of unknown transmit power, a modified maximum likelihood estimation (MLE) [22] can be used.[1] For dealing with the challenge of environmental characteristics in trilateration, the EZ algorithm [6] uses a genetic optimization algorithm. Another way to deal with distance uncertainties is to use a significantly higher number of measurements compared to the case in Fig. 2. Such an approach is discussed in [11]. Since the distance estimates of the transmitter using the path loss model are often inaccurate, methods that use the rank ordering of the RSS values may be more effective. For example, the Echolocation [39] algorithm (EL) counters this problem by applying a non-parametric method popular in statistics, called ranking.

While path loss models are statistical models which summarize the RF characteristics of an environment, more complex models can be used for localization. For example, ray tracing in a 3D modeled environment is a computationally intensive method of simulating the path loss on a link, producing far more detailed estimates of signal strength at a given location. Unfortunately, ray tracing can still experience high error rates due to factors like inaccurate 3D models, changes in the environment over time, or flawed physical models [15, 34, 35].

In scenarios where detailed environmental information is lacking, such as complex multipath scenarios or rapidly changing conditions, physical models may be insufficient for accurate and robust localization. As a result, many researchers have turned towards exploring fingerprint-based methods in an attempt to capture the necessary characteristics for precise localization.

2.2 Fingerprint-Based Localization

Fingerprinting methods are data-based approaches that use pre-existing knowledge or "fingerprints" of the signal characteristics from known transmitter locations. We assume fingerprint data includes RSS measurements, but it could also include other channel information and statistics or even features extracted from the signal using ML methods. These fingerprints can be used in a database of known locations; given the current conditions and the locations in the database, the target location is estimated based on the closest match in the database.

Localization of a mobile node via fingerprinting can be considered a learning problem, and it has been studied extensively in the context of WiFi fingerprinting. The basic idea is to capture RSS fingerprints, from static access points (AP), for all the locations in an indoor area. Subsequently, the location of a mobile node in the same area is obtained by searching for a match between the current RSS fingerprint and the previously collected RSS fingerprints by deterministic/probabilistic methods [3, 40]. This method requires extensive manual effort in collecting the fingerprints during training. To circumvent this problem, researchers have investigated ways

[1] For the case of known transmit power, a solution has been presented in [23].

to make the system work even if the fingerprints are spatially sparse [13, 29, 30]. They have also used crowdsourcing for collecting the training fingerprints [25, 36]. Irrespective of the approach, these learning-based localization methods ultimately depend on the RSS to/from the nearby static APs. If both the transmitter and the receivers are continually mobile none of these approaches can be adopted directly. One way to perform fingerprinting in such scenarios is to leverage interpolation methods as discussed in [27].

The chief advantage of fingerprint-based methods is that with a large corpus of data, this technique can be applied anywhere, indoors or outdoors, with or without line-of-sight, in any environment. Fingerprint methods also have significant disadvantages as well. As mentioned previously, fingerprints may require a set of static APs. Fingerprinting may fail if some devices are offline. As well, a database needs to be extensive enough to capture the unique environmental characteristics. The challenge of collecting such a dataset may be feasible in controlled indoor environments but more difficult in outdoor or dynamic environments. Additionally, as the size of the dataset grows, more sophisticated data structures may be required to efficiently handle fingerprints.

2.3 *Neural Networks for Localization*

Recent works [18, 38, 41, 43] have presented a new paradigm for ML-based localization using computer vision techniques. RSS information is encoded in a 2D image which represents a birds-eye view of the sensor locations, with the pixel intensity set to the RSS value of the sensor. This *sensor image* format neatly captures the spatial relationship between sensors. Convolutional neural networks (CNNs) developed for image processing can then be used for localization.

An overview of this process is shown in Fig. 3. Crowdsourced sensor measurements are received by the fusion center and encoded into a 2D image. Optionally, the image can be combined with other modes of information, such as elevation

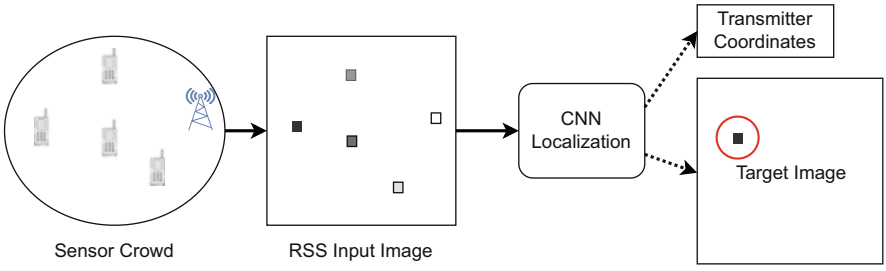


Fig. 3 An overview of the process of CNN-based localization. Sensor RSS values are converted to an image which is input to the CNN. The model outputs either predicted transmitter coordinates, or an image predicting the location of transmitters

and building footprints [38], sensor calibration [19], or environmental features as outlined in [31]. The 2D image is then processed by the CNN, which produces either a similar 2D image marking the location of the transmitter (as in [18, 38, 41]), or directly predicting transmitter coordinates (as in [19, 43]).

This process of converting sensor coordinates to a 2D image has a few problems. First, there is a loss of precision as real-valued GPS coordinates are converted to pixels. The ***pixel scale***, or meters-per-pixel, can have a huge impact on how well a CNN can perform localization. A poorly chosen pixel scale can result in up to a $10\times$ increase in error [19].

Another problem with image-based localization is that sensor values must be converted to an image, and all targets are assumed to be located within the area represented by the image. Though some non-image-based techniques also have this problem, traditional physics-based localization techniques are not limited to particular regions. Most localization settings assume that the target is in or near the convex hull of the senors; a full exploration of if and when localization techniques can accurately locate transmitters that are far any sensor nodes remains an ongoing area of research.

In general, CNN-based localization models use either ***direct coordinate prediction*** in the form of a regression problem [43], or use a CNN for ***image-to-image*** localization to produce an image-map, similar to the input, where the location of transmitters is marked by high-value pixels [41]. The primary advantage of directly estimating coordinates is that the localization error is differentiable and easy to calculate, meaning the entire process is easily optimized in the case of one transmitter, and only slightly more complex in the case of multiple transmitters. Meanwhile, image-to-image localization produces an output which is visually intuitive, and performs well in both simulated [38, 41] and real-world [18, 19, 42] evaluations.

One obstacle in the image-to-image formulation is that predictions are discrete pixels rather than real-valued coordinates. In order to achieve higher accuracy, many works use ***sub-pixel prediction*** [19, 41], using either weighted averages from different predictions or even additional ML models to estimate transmitter coordinates on a sub-pixel level.

2.3.1 Augmenting with Physical Models

One primary appeal of learning-based localization is that additional features beyond RSS values can be used for localization. Information from physics-based approaches can be incorporated into the model. For example, LocUNet is a CNN that combines sensor information with building footprints and estimated propagation maps for accurate localization. Another potential approach comes from Tadik et al. [31], where they use environmental features including line-of-sight, number of obstacles, elevation angle, and street alignment as inputs to a neural network which learns corrections that are applied to a physics-based propagation model. This approach

could be applied to localization to similarly provide relevant environmental features for each pixel, providing more information for accurate localization.

3 Recent Localization Techniques

In this section we present an exploration of several localization techniques proposed by the authors, all of which solved existing challenges in the context of localization using crowdsourcing. We begin with SPLOT [11], a physics-based method that localizes simultaneous transmitters. We then present LLOCUS [27], which uses interpolation and learning methods to allow for unrestricted device mobility as well as localizing devices with unknown transmit power. We then present TL;DL [18] and CUTL [19], CNN-based localization techniques which achieve a higher degree of accuracy while solving challenges of limited datasets and heterogeneous sensors without calibration. Additionally, we explore the difficulty of localization on inputs that differ significantly from the training data.

3.1 SPLOT

Simultaneous Power-based Localization of Transmitters [11] is a 2017 physics-based technique that is capable of localizing multiple transmitters that are active simultaneously. To deal with the problem of multi-transmitter localization, SPLOT relies on two key observations. First, receivers that are located near the transmitter generally observe higher power than receivers that are distant from the transmitter. Second, the observed RSS at each receiver is primarily affected by the nearest transmitter. Based on these observations, the problem of localizing multiple transmitters that are active simultaneously can be transformed into a set of single transmitter localization problems. To do so, first, SPLOT finds the local maxima of RSS values (measured by receivers) that are greater than a predefined threshold. The predefined threshold is set to the minimum RSS that a receiver observes when a transmitter is near it. With the knowledge of the local maxima, the localization problem can be reduced to finding K transmitters, where K is equal to the number of local maxima. For each local maximum, SPLOT locates a single transmitter. For each single transmitter localization, SPLOT considers the RSS measurements that are only in the vicinity of the local maxima. This helps in reducing the computational complexity and improving localization accuracy. The method used for localizing each of the single transmitters is briefly described in the following.

The RSS values at the receivers, $\mathbf{y} = [y_1, y_2, \ldots, y_L]$, can be modeled as:

$$\mathbf{y} = \mathbf{W}\mathbf{x} + \mathbf{n}, \tag{1}$$

where $\mathbf{n}$ represents a vector of noise and fading contributions, $\mathbf{x} = [x_1, x_2, \ldots, x_Q]$ is a power field where x_i represents the power emitted from position i (assuming there are Q discretized/pixelized positions), and $\mathbf{W}$ is an $L \times Q$ matrix where W_{ij} represents the path loss between receiver location i and transmitter position j. The weights W_{ij} are calculated using a path loss exponent model with the path loss exponent derived using some calibration data. An estimate of the power field, $\hat{\mathbf{x}}$, can be found using a regularized least-squares approach to the inverse problem. Finally, the transmitter's location is estimated as the location corresponding to the maximum value of $\hat{\mathbf{x}}$. Further mathematical details of SPLOT can be found in [11].

The primary advantage of SPLOT is that it can be used to localize multiple unauthorized transmitters if they are active simultaneously. At the same time, SPLOT can tackle scenarios where one unauthorized transmitter is active alongside an authorized transmitter. SPLOT also has a couple of limitations. First, it uses a radio wave propagation path loss model, which may not be appropriate for all radio environments. This can affect the localization accuracy of SPLOT. Second, SPLOT does not consider the fact that different transmitters can have different transmit power, especially when some transmitters are adversarial. This issue can hinder SPLOT's capability of detecting the number of simultaneously active transmitters.

3.2 LLOCUS

LLOCUS [27] uses a crowdsourcing framework similar to SPLOT, allowing for the mobility of both sensors and transmitters. Like SPLOT, LLOCUS localizes multiple transmitters by finding local maxima in spatial sensor information and localizing individual transmitters associated with the local maximas. However, LLOCUS addresses the limitations of SPLOT. Specifically, LLOCUS uses a learning-based approach and does not depend on a physics-based path loss model. Additionally, it can tackle the problem of unknown and dissimilar power of the transmitters.

LLOCUS uses a multi-step process for multiple transmitter localization and transmit power estimation. First, the number of active transmitters is estimated based on the number of local maxima in the RSS data. Then the transmit power of each transmitter (associated with each local maxima) is estimated using an SVM-based regression method. Importantly, the transmit power estimation is done before the localization. This not only provides the capability of estimating the power of active transmitters, but it is also crucial in overcoming the second limitation of SPLOT. The estimated transmit power is used to form a region of presence around each local maxima. The area of each region is proportional to the corresponding estimated transmit power. Next, for each region of presence, the estimated transmit power is used to scale the measured RSS values within that region to match a common reference transmit power. Finally, these scaled RSS values are used to localize the transmitter within that region. The steps described above are pictorially depicted in Fig. 4.

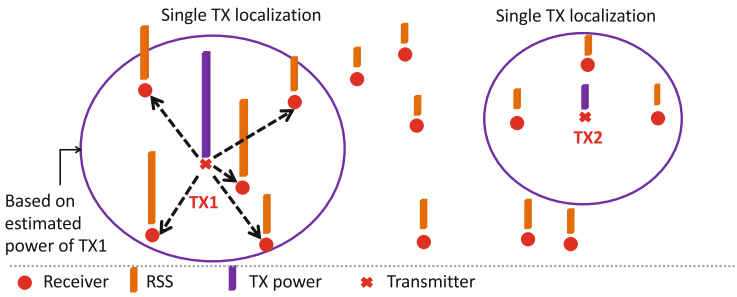


Fig. 4 An overview of the multi-source localization process in LLOCUS. Received power at sensors is indicated by orange bars. The Tx power (purple bar) is estimated, then sensors around each local maxima are used to localize the transmitter

Table 1 Comparison of LLOCUS and SPLOT [27]

	SPLOT			LLOCUS		
Experimental setup	$\bar{r}_m$	$\bar{r}_f$	$\bar{\epsilon}_p$ [m]	$\bar{r}_m$	$\bar{r}_f$	$\bar{\epsilon}_p$ [m]
2–3 active transmitters, no power variation	0.04	0.1	10.94	0.05	0.1	6.09
1 active transmitter, power variation	0.01	0.38	12.7	0.01	0.1	4.5

For each single transmitter localization, LLOCUS uses a learning-based approach, specifically, a fingerprinting approach. Since sensors are assumed to be mobile, traditional fingerprinting methods that require a static reference cannot be used. In order to associate RSS values with a static context, LLOCUS interpolates RSS values to a specific set of fixed locations. With the interpolated RSS values at the fixed locations as fingerprints, a radial basis interpolation method is used for localization.

Table 1 shows the performance comparison of SPLOT and LLOCUS based on an indoor dataset described in [27]. In this table, $\bar{r}_m$ is the missed detection rate, $\bar{r}_f$ is the false positive rate, and $\bar{\epsilon}_p$ is the penalized localization error (penalty is added for missed or excess transmitters). When transmitters have a fixed transmit power, SPLOT and LLOCUS have similar detection rates, but LLOCUS is significantly more accurate in localization error. When transmit power varies, LLOCUS has a significantly lower false positive rate.

Although both SPLOT and LLOCUS localize multiple transmitters, the assumption that each transmitter is represented by a local maximum in the RSS measurements may not always be true. Two low-powered transmitters in the same band could be indistinguishable if there were not sufficient sensor coverage to provide a local maximum near each transmitter. Also, if two transmitters are very close they may not give rise to two distinguishable local maxima, especially when the receivers are sparse.

3.3 TL;DL

Following other works in the field [41, 43], Transmitter Localization with Deep Learning (TL;DL) [18] utilizes a UNet architecture [26], a CNN with more layers than previous techniques. This deeper model improves accuracy rates while still maintaining a runtime of less than 20 ms on a consumer grade CPU. TL;DL also uses data augmentation to greatly improve localization in practical settings by using *sensor dropout*. During training iterations, only a random subset of the sensor data is used for localization, so the model does not become dependent on certain fixed locations or characteristics of the training data which may not be present during evaluation. For limited datasets with less than 100 training samples, sensor dropout improved accuracy by up to 75% compared to models trained without training data augmentation.

Previous works had only been evaluated on simulated data, so in evaluating TL;DL we tested accuracy only on real-world, indoor and outdoor datasets. Table 2 shows the results of this evaluation, using the same notation as in Table 1. TL;DL had the lowest localization error in all tests when including penalties for detection errors, and consistently better performance than other techniques in detection rates.

Since TL;DL does not make the assumption that all sensors are represented by local maxima, the technique surpasses other methods at detecting multiple transmitters with a low sensor density. In the most challenging case (Dataset 4), there are up to 5 simultaneous transmitters. Sensors are placed non-uniformly and with very low density, but TL;DL was still able to detect 94% of transmitters, compared to LLOCUS and SPLOT which detected 18% and 33% of transmitters, respectively.

3.4 CUTL

Calibrated UNet Transmitter Localization, or CUTL [19], advances CNN localization by applying a learned pseudo-calibration. Since most sensors will not have any calibrated reference power, we learn calibration parameters for each type of

Table 2 Localization results using TL;DL [18]

	Dataset 1 (1 Tx)			Dataset 2 (1–2 Tx)			Dataset 3 (1–2 Tx)			Dataset 4 (1–5 Tx)		
	$\bar{r}_d$	$\bar{r}_f$	ϵ_p	$\bar{r}_d$	$\bar{r}_f$	ϵ_p	$\bar{r}_d$	$\bar{r}_f$	ϵ_p	$\bar{r}_d$	$\bar{r}_f$	ϵ_p
TL;DL [18]	0.25	0.02	**11.6**	0.03	0.03	**3.7**	0	0.02	**1.6**	0.06	0.06	**9.5**
DeepTx [43]	0.14	0.07	15.5	0.02	0.09	12.3	0	0.02	9.0	0.02	0.10	12.1
DMTL [41]	0.60	0.04	14.5	0.61	0.07	16.2	0.40	0.14	17.3	0.11	0.13	15.2
LLOCUS [27]	0.28	0.30	17.0	0.52	0.06	12.7	0.08	0.22	13.6	0.82	0	26.8
SPLOT [11]	0	0.57	18.8	0	0.45	16.8	0	0.19	10.3	0.67	0	24.7

Bold values show the lowest error.

sensor in a crowd. For each category of sensor device, such as mobile devices, ground level dedicated sensors, or rooftop installations, we learn parameters to scale the RSS values from that category. The assumption is that devices with similar hardware and placement patterns will have more similar RSS values, an assumption that we observed to be true in several different real-world datasets [17, 18]. In our evaluation, we observed mobile sensors had significantly higher noise levels, experienced more interference, and also had inconsistent noise floors from device to device. Our pseudo-calibration method learned to equalize the noise floor between these sensors and to slightly decrease the importance of mobile sensors in order to reduce sensitivity to noise.

We compared direct coordinate prediction to image-to-image localization by training two UNet models, one with linear layers on the end which learn direct coordinate prediction, and the other using the image-to-image technique. Our results showed the image-to-image technique outperformed direct prediction in all of our tests, though only by a small margin.

In the same work we also apply ensemble models for more accurate localization. The training data is divided into 5 parts, and 5 identical CNNs are trained on only 4 of the parts. Then, at inference time, the localization estimate is the weighted average of the 5 models, based on the model confidence in each prediction. This resulted in up to 15% higher accuracy for image-to-image prediction models, and a modest gain of up to 5% for models using direct coordinate prediction.

We also explored how input resolution affects accuracy. One might naively assume that increasing the resolution of the input will increase the accuracy since it involves less loss of precision, but this is not the case. Lower pixel scales result in higher input resolution and compute time, but this also harms the network's ability to localize accurately due to the limited receptive field of CNN architectures. We also hypothesize that the mean-squared error loss function is not optimized successfully at very high resolutions. On the other hand, a large pixel scale will hurt accuracy due to the loss of precision in both inputs and outputs. We recommend that the pixel scale be chosen based on experimental validation, since an ideal pixel scale seems to be dependent on several factors, including the area of the region being considered for localization, the sensor density over this area, and the architecture of the CNN being used [19].

3.4.1 An Out-of-Distribution Dataset for Localization

One of the primary challenges of fingerprint-based localization is when training data does not accurately represent samples at inference time. The ability to generalize to out-of-distribution (OOD) data is a fundamental problem in ML, one that cannot be solved using only the training data. In a practical localization system, this distribution shift occurs over time, such as from daily changes from traffic patterns, seasonal changes in foliage, construction and demolition of buildings, or gradual changes in the RF chain of both sensors and transmitters. Other distribution shifts

might include training data that doesn't cover a particular area of interest, or fails to account for interference from multiple transmitters.

In order to evaluate localization techniques, we captured a dataset [17] of over 4500 unique transmitter locations with heterogeneous sensors. The transmitter moved through a 4 km^2 area and was carried on foot, while cycling, and in an automobile. RSS values were captured by 9–25 sensors which were both mobile and fixed, with a variety of antenna configurations and placements.

To measure the ability of a localization algorithm to transfer to OOD data, we divided our dataset into different "splits" that each represent a shift between the training and test data:

- *Random*: Assign each transmitter location randomly to the training or test split (the default condition in many fingerprinting problems)
- *Grid*: Divide the area into a 10 × 10 grid of cells, and assign all transmitter samples within a cell to ether the training or test split
- *Driving/Pedestrian*: Assign samples based on the method of transmitter mobility
- *April/July*: Assign samples based on the date of collection

With these different dataset splits for training and testing, we evaluated CUTL on its ability to localize OOD transmitters. These results are shown in Table 3. Accuracy on OOD data varied widely. The *Grid* split had a median accuracy almost 3× worse than the *Random* split. In this case, each grid cell was approximately 200 m wide, so an error on the order of 100 m shows that localization within the cell is largely inaccurate, but the model is predicting a location near the correct cell.

The *Pedestrian* and *July* training sets had over 3000 training samples, but the models trained on this data had lower accuracy than their *Driving* and *April* counterparts which had significantly less training data. These results highlight that a large training set is not helpful for localization if it does not accurately represent the data encountered at test-time. Overall, the predictions from CUTL were more accurate than any other state-of-the-art localization methods. These results show that no existing methods generalize well to OOD cases. It's possible that through sophisticated data augmentations, this gap between distributions could be bridged.

Table 3 The median localization error for OOD test sets, along with the number of samples in each train and test set [19]

Train set	Size	Test set	Size	Median error [m]
Random	3399	Random	828	40.1
Grid	3536	Grid	691	117.6
Driving	925	Pedestrian	3302	181.5
Pedestrian	3391	Driving	836	264.9
April	811	July	3416	207.4
July	3416	April	811	335.8

4 Adversarial Attacks on Crowdsourced Localization

While crowdsourcing sensor information can provide a higher degree of sensor coverage, this benefit comes with costs; the crowdsourcing model presents a new avenue of attack for adversarial actors. Since sensors are not controlled by a single trusted party, adversaries can exploit the accesibility of the system and manipulate data to their advantage. In the context of localization, a malicious actor may disrupt the process by injecting false data, or poisoning, through the crowdsourcing mechanism. Adversaries can exploit the weakness of ML models to out-of-distribution data, though in this case the distribution shift comes from false data, rather than realistic changes in the environment.

In this section, we explore possible adversarial attacks on crowdsourced localization. These range from naive attacks, where adversaries' limited knowledge and capabilities may prevent a more effective attack, to omniscient attacks, where adversaries possess complete knowledge of the system architectures, model, and data. The latter enables an adversary to test arbitrary attacks and identify vulnerabilities to later exploit in a live localization system.

The envisioned system for transmitter localization relies on users submitting RS measurements and associated sensor locations to a central server or fusion center which then estimates the transmitter location. In this context, an adversary's attack must come from participation in the crowdsourcing process; we do not address network, server, or infrastructure attacks.

4.1 Naive Attacks

In the most naive setting, an adversary possesses no information about the transmitter's location, the localization algorithm, or the RF environment. Their only objective is to increase error in the localization system.

One simple naive attack is to randomly select a location, assign an RSS value to a spoofed sensor at this location, and submit this information to the crowdsourcing system. The adversary may consider two scenarios for their fake RSS value: a low value near the noise floor, or a high value indicating the transmitter is nearby. A low RSS close to the true transmitter location could cause error, but a low RSS far from the true transmitter would have little impact. Similarly, a high RSS near the transmitter could actually increase the localization accuracy, but could also hide the true transmitter location if it were a greater distance from the transmitter.

4.2 Informed Attacks

The naive setting, while simple to consider, is quite unrealistic. In practice, an adversary may possess information about the transmitter location, have some understanding of the localization algorithm, or have access to sensor measurements. In such cases, they could exploit this knowledge to craft far more effective attacks against the system. We call this an *informed attack*, where an adversary attempts to mask or mislead localization of a known transmitter, or more generally to undermine the trust or reliability of the system.

Potential informed attacks are simple to formulate. For example, an adversary could report an RSS value that negatively correlates with what is detected at their own location. With knowledge of the transmitter location, they could spoof a fake sensor with high RSS far from the target.

While many informed attacks would be quite effective and represent a more likely scenario, we focus on naive and omniscient attacks in our case study in Sect. 5, since any informed attack would generally be bounded between naive and omniscient attacks.

4.3 Omniscient Attacks

The omniscient or *white-box* attack setting is extremely powerful. An adversary may have access to all sensor coordinates and RSS values, the transmitter location, and information about the specific localization algorithm used. If a non-learning technique is used for localization, then an adversary could exactly replicate the localization results. If a ML model is used, an adversary may not have access to the exact model used for localization, but could train a similar surrogate model to perform the same task. Papernot et al. [21] show that on computer vision tasks, attacks developed on a surrogate model are often effective against a previously unseen model, even if the surrogate model is trained on different data to perform a similar task.

It may be apparent that this omniscient setting is extremely unrealistic. An adversary has access to all crowd measurements with the ability to inject an attack to the fusion center. This adversary clearly has an outsized influence on localization effectiveness. As mentioned previously, the impact of omniscient attacks and our ability to defend against them, provides an upper bound on how effective an informed attack may be. Omniscient attacks could potentially be deployed offline, allowing an adversary to craft strategies to be deployed in a real system as informed attacks.

4.3.1 Worst-Case Attack

If an adversary has sufficient time to consider many attacks, then a worst-case attack could be easily developed by inserting a spoofed sensor at the worst possible location which maximizes the localization error. As an example, an adversary could test the effectiveness of inserting a high or low RSS value at every pixel in the CNN input, and find which attack is the most effective.

4.3.2 Fast Gradient Sign Method

A classic technique for adversarial input generation in computer vision is the *Fast Gradient Sign Method* (FGSM) from Goodfellow et al. [10]. This attack uses the same backpropagation algorithm used to train neural networks to instead produce an attack based on the gradient with respect to the crowdsourced input data X. Let θ be the parameters of our model, X be the input, Q be the localization target, and J be the cost model used to train the model using backpropagation. Then FGSM produces a perturbation vector η:

$$\eta = \epsilon \cdot \operatorname{sign}\left(\nabla_X J(\theta, X, Q)\right). \tag{2}$$

In other words, J is the cost which is minimized while training the model using gradient descent. This cost (the localization error) is minimized iteratively by taking the gradient with respect to the model parameters θ, and updating those parameters in the opposite direction of the gradient $\nabla_X J$. FGSM uses the same formulation, but instead of updating model parameters, we take the sign of the gradient with respect to X, producing a perturbation η with values in $\{-\epsilon, \epsilon\}$ which can be added to X to increase the overall error.

We consider three main types of attacks based of FGSM:

- **Sensor perturbation** attacks change existing sensor values by $\pm\epsilon$.
- **Withholding** attacks conceal sensor values by removing entries from the crowd.
- **Virtual sensors** are spoofed measurements provided to the fusion center.

These attacks can be combined for more sophisticated attacks; an adversary could insert virtual sensors while reducing the RSS of sensors near the transmitter by ϵ.

A Word on Adversarial Control For any attack, whether naive, informed, or omniscient, an adversary with control of a significant portion of the crowd could cause an arbitrary localization error. As part of the threat scenario, we must assume some limit to the percentage of the crowd controlled by the adversary, but we do not explore how this limit could be enforced in realistic settings.

4.4 Defending Against Adversarial Attacks

More important than understanding the potential threats that an adversary poses, we consider potential defences against adversarial attacks. In general there are two main objectives that a robust localization system must consider when crowdsourcing measurements:

1. Identifying and removing adversarial sensors
2. Accurate localization in spite of adversarial attacks

Obviously, these objectives are tightly intertwined: adversarial sensors cannot be identified without recognizing an attack, and accurate localization is easier if all adversarial inputs have been removed.

4.4.1 Sensor Identification and Removal

Excluding adversarial sensors is a complex and challenging problem, but there are several approaches to be considered:

- *Statistical Analysis:* Apply statistical techniques to identify outliers in the sensor data. If RSS values are inconsistent with normal behavior, this could indicate adversarial interference.
- *Anomaly Detection:* Machine learning models can be trained specifically to classify inputs as normal or adversarial. If specific attack methods are likely to be encountered, adversarial samples can be generated and used to train a classifier which detects attacks, though assuming a particular set of attacks could be a pitfall that provides a false sense of system security.
- *Physics-based Detection:* If certain properties of RF propagation in the environment are known, then adversarial sensors could be detected by observing the variation of each point from the expected conditions.
- *Crowd Validation:* If sensor data from trusted sources is being augmented with crowdsourced data, then the correlation between trusted data and crowdsourced measurements can be found, potentially identifying bad actors.

Localization Specific Outlier Removal Whatever removal technique is used, there is some risk of discarding non-adversarial measurements. One technique for outlier exclusion specific to localization relies on evaluating the amount of change in a localization estimate when a single sensor value is withheld. Given a localization model h_θ and a set of sensors and measurements S, the predicted target location is $\hat{Q} = h_\theta(S)$. Then, for each sensor input s_i, the predicted location without s_i is calculated, $\hat{Q}_i = h_\theta(S \backslash \{s_i\})$. The sensor s_i that results in the largest difference $|\hat{Q} - \hat{Q}_i|$ is considered to be an adversarial input, if $|\hat{Q} - \hat{Q}_i|$ exceeds some threshold γ.

This technique can be applied for any localization algorithm, though it can also be modified for different models. For example, an image-to-image prediction that

results in a heat map of probable transmitter locations could consider the difference between heat maps rather than the difference in predicted coordinates.

This technique's efficacy may be largely dependent on the sensor density and coverage of a region. In a sparsely covered area, localization accuracy is largely dependent on the measurement closest to the target, so this method of outlier exclusion greatly decreases overall accuracy [20].

4.4.2 Adversarial Training for Accurate Localization

It's important to note that no single technique can guarantee detection and removal of adversarial inputs. In view of this, it is crucial to develop localization techniques that are robust to some amount of adversarial perturbation.

As is common in computer vision [10, 21], **adversarial training** can provide a great measure of robustness for ML-based localization. During model training, a set of known attacks are randomly applied to the input data, with the goal of teaching the model to localize targets in spite of corrupted data. This can be done with both naive attacks, worst-case attacks, or attacks generated using FGSM.

Adversarial training does come with a few drawbacks. The most obvious for large, complex models is the increased computational complexity. In our case, most proposed localization techniques are relatively simple compared to the 50–100 layer CNNs used in many computer vision contexts, so this is less of an issue in our localization context. Our CNN localization models were typically trained in less than 1 hour on a consumer-grade GPU, so while applying adversarial training may increase the training time by 2–4×, this is not prohibitive.

A more critical concern for adversarial training is if the attacks applied do not resemble the actual attacks that an adversary may employ. In this case, a model may be assumed to be robust to adversarial attacks but in practice could be equally vulnerable as a baseline model. The defended model may actually become more vulnerable to attacks not used as part of its training process [33].

In practice, adversarial attacks may be less effective compared to similar attacks on computer vision tasks. In the context of localization, a limited crowd provides a much smaller attack surface. For example, an adversary attacking an image-based CNN localization model could not perturb every input pixel if their control is limited to a small portion of the sensor crowd. This smaller attack surface may be more robust overall [14].

5 A Case Study on Attacking Localization

We now present a case study exploring the effects of adversarial attacks and defenses on a real-world localization dataset. We use our publicly available dataset described in Sect. 3.4.1 [17] with a random split of the data into training and test sets. These results are based on a previous workshop paper [20].

We deployed the CUTL image-to-image model for localization [19], with a meter scale of 60 m per pixel chosen through experimental validation. CUTL achieves a median accuracy of 38 m. This may seem like a high error compared to other techniques which claim sub-meter accuracy, but achieving sub-meter accuracy in [18] required approximately 4000× the sensor density. *Claims of high localization accuracy should **always** be viewed in the context of sensor density and distribution.*

Once our model was trained, we then applied the adversarial attacks and defenses described in Sect. 4. Ultimately, we show adversarial training to be extremely effective at defending against the attacks used during the training process. Naive attacks are largely ineffective after adversarial training, and the FGSM-based omniscient attacks are significantly less effective with adversarial training. The worst case attack, by nature of it being the worst case, remains somewhat effective on our model after adversarial training.

5.1 Attack Scenario

The CUTL model we deploy is a UNet image-to-image localization model using learned calibration for different categories of sensors. We assume that an adversary is using spoofed sensors which they are labeling as mobile crowdsourced devices. As was mentioned in Sect. 3.4, the learned calibration reduced the importance of the mobile sensors, so our attack does have a slight disadvantage in this sense.

As single-sensor attacks, we applied a naive attack with random RSS and random coordinates, and an omniscient worst case attack where the adversary queries the localization model many times to determine which is the most impactful pixel for attacking. For the FGSM attacks, since the image-to-image localization model is not differentiable with respect to the localization error, we used a surrogate model trained on identical data to generate attacks using the FGSM technique. These attacks were then evaluated on the original image-to-image model, which we refer to as the ***baseline*** model.

5.1.1 Defending Our Localization Model

We applied adversarial training to defend against adversarial attacks. The CUTL localization model was retrained on the same training data while one of the following attacks was randomly applied to the training samples for each training batch:

1. `Top-N%`: The top $N\%$ of sensors with the largest magnitude gradient are perturbed by a random constant ϵ, where N ranges from 10–50%.
2. `Drop-N%`: The top $N\%$ of sensors with the largest gradient are withheld, where N ranges from 10–50%.

3. `Hi-Lo`: Spoofed sensors with high or low values are inserted. The M sensors with the largest positive gradient are set to an RSS of ϵ and the N sensors with the largest negative gradient are set to the noise floor.
4. `Top-N%+Hi-Lo`: Apply the `TopN%` attack to imitate the sensors an adversary controls, then apply `Hi-Lo` to insert fake sensors.

We also considered methods of outlier exclusion described in the last part of Sect. 4.4.1. This involves removing sensors based on how much of a change in prediction results from excluding a particular sensor. We found that with such a low sensor density, excluding sensors resulted in significant drops in accuracy. This is unsurprising, considering that with a low sensor density accurate localization may be largely dependent on a single sensor.

5.2 Naive Random Attack

We considered a single spoofed sensor with random coordinates and a random RSS between the 10th and 90th percentiles. As might be expected, this was a largely ineffective attack. Of 600 random attacks on each sample, only 6.4% of naive attacks were successful at increasing the localization error by 2×. When adversarial training was applied, this dropped to only 3.3% of attacks.

Although the naive attack is limited in its effectiveness, this does seem to be partially determined by the particular sensor configuration. With a test set of 828 samples, we repeated naive attacks 600 times on each test sample. For some samples, none of the random attacks had any impact on localization predictions. The success rate for each sample varied widely from 0–65% success in doubling the error. This variation implies that certain configurations are far more vulnerable to attacks.

5.3 FGSM Attacks

We now explore FGSM-based attacks as a slightly constrained attack, where the adversary has a surrogate model to query but does not have time to generate many attacks in a worst-case scenario.

We applied the attacks described in Sect. 5.1.1, evaluating against both the baseline model and the adversarial trained model, denoted as *AdvTr* in figures. We evaluated attacks on each of the 828 samples in the test set. The baseline model had a median error of 38 m and the adversarial trained model had a median error of 36 m.

Top-N% In Fig. 5 we show the success rate of the `Top-N%` attack. Here we add ϵ to the top 20, 50, and 100% of sensors, ordered by the magnitude of their pixel gradient. As mentioned before, we consider ϵ between 0 and 0.5.

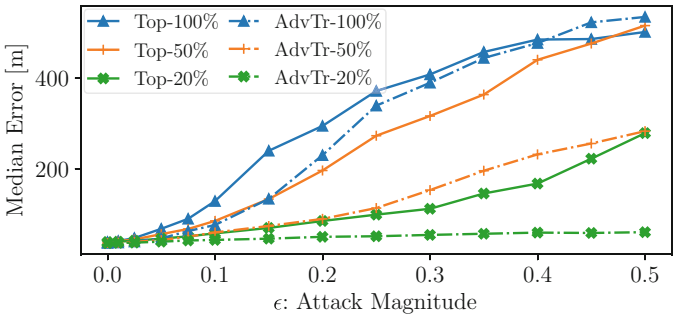


Fig. 5 The median error caused by `Top-N%` attacks, with and without adversarial training

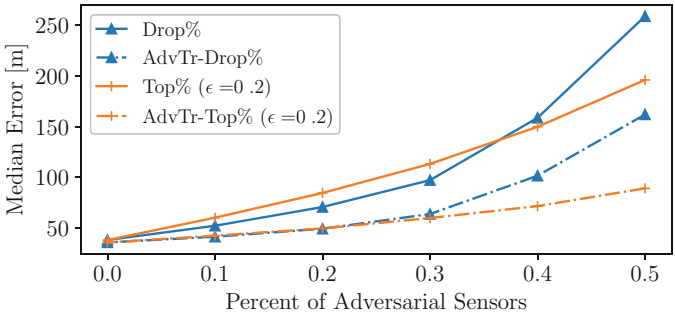


Fig. 6 The median error caused by `Drop-N%` and `Top-N%` attacks, with and without adversarial training

For the `Top-100%` attack, adversarial training does not have a large impact on median error, but the improvement is drastic for the attacks with fewer adversarial sensors. The average improvement for the restricted case of 10–50% adversarial control increased with ϵ, with an average improvement of 65% for $\epsilon = 0.5$.

Drop-N% The `Drop-N%` attacks withhold a percentage of the highest gradient sensors, so it is independent of any constant ϵ. In general, this attack was approximately as effective as a `Top-N%` attack with $\epsilon = 0.2$. These two attacks are shown in Fig. 6. They are similar in their effectiveness and the impact of adversarial training. This finding was somewhat surprising, since the attacks are opposites; one increases the RSS of key sensors, and the other withholds information from key sensors.

As might be expected, the `Drop-N%` attack does become significantly more effective as the number of sensors withheld by the adversary increases. With 50% of sensors withheld, the median accuracy after adversarial training is almost double of the `Top-N%` attack. This is likely because the `Drop-N%` attack completely removes any sensor information, while the `Top-N%` only introduces a bounded amount of noise to the actual sensor value.

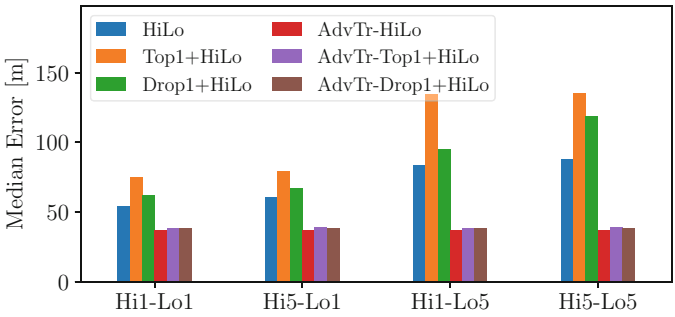


Fig. 7 The median error caused by `Hi-Lo` attacks ($\epsilon = 0.5$), with and without adversarial training

Hi-Lo The `Hi-Lo` attacks, unlike the previous attacks, do not require the adversary to control a large percentage of sensors. Instead, we inject low-RSS and high-RSS fake sensors into the sensor vector S, with values at the noise floor and $\epsilon = 0.5$, near the 95th percentile. In Fig. 7 we show results from adding 1 and 5 of each type of sensor. For the baseline model, the low-RSS sensors are particularly effective, since the `Lo5` attacks have a significantly higher median error than the `Lo1` attacks. The effectiveness of low-RSS sensors compared to high-RSS sensors may seem counter-intuitive. These fake sensors are almost identical to the 0-valued pixels that make up the majority of the image. Although it is difficult to determine exactly why these small negative values are impactful in this model, in neural networks, we assume that these low-valued pixels pixels decrease the likelihood that the transmitter is near that location in a way that the 0-valued, non-sensor pixels do not.

The `Hi-Lo` attack can also include sensors controlled by an adversary, as shown by the `Top1` and `Drop1` variants shown in Fig. 7, where a single sensor was either perturbed by ϵ or withheld by the adversary. The `Top1` attack was more effective than the `Drop1` variant. All these attacks were entirely neutralized by adversarial training.

5.4 *Worst Cast Attack*

We apply the worst case attack, where each input pixel is set to values of the 10th and 90th percentile of RSS values, the localization error is calculated with this spoofed sensor value, and the worst possible case is used as the attack for that sample.

As might be expected, the worst case attack is extremely effective on almost every test sample, even though it only utilized a single sensor. We assume the adversary has an identical localization model which they repeatedly query to determine the worst possible attack using a single spoofed sensor. Over 95% of attacks doubled the localization error, though adversarial training did decrease this to 70% of attacks.

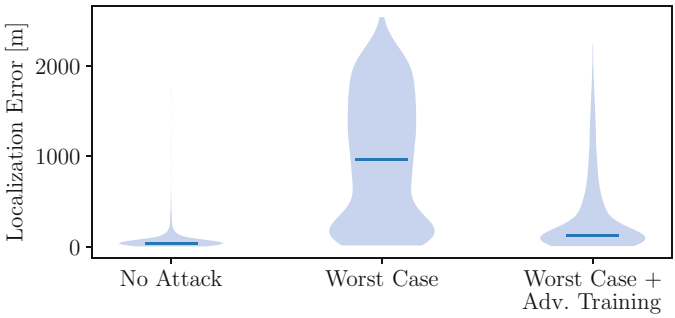


Fig. 8 Violin plots showing the shift in error distributions for localization with no attack, worst case attack, and worst case on a model with adversarial training. The median of each distribution is also marked

Table 4 Error statistics for adversarial attacks with and without adversarial training

	Baseline error			Adv. training error		
Attack	Median [m]	Mean [m]	2× Increase	Median [m]	Mean [m]	2× Increase
No attack	38.0	67.5	–	35.7	66.2	–
Naive	39.3	78.6	6.4%	36.3	69.7	3.3%
Top-10% $\epsilon = 0.4$	242.2	83.2	43.1%	43.7	114.6	19.2%
Drop-10%	52.0	116.6	23.4%	41.3	90.7	14.4%
Top1+Hi5-Lo5 $\epsilon = 0.4$	83.4	311.2	49.0%	38.6	90.9	12.7%
Worst case	965.2	983.2	95.0%	126.1	328.2	69.6%

Fortunately, All hope is not lost for defending against the worst case attack. Figure 8 shows the error distributions of the worst case attacks, with and without adversarial training. The worst case attack increases the median error from 38 to 965 m, a 25× increase, but adversarial training drops the median error to 126 m, only a 3× increase. While this is still a significant amount of localization error, the major shift in the distribution shows the huge impact that adversarial training can have. It's also important to remember that the adversarial training did not include worst case attacks, only FGSM-based attacks, so it's possible that specifically training on the worst case attack could further improve resiliency.

5.5 *Discussion*

The attacks executed in this work were effective at producing high error in an otherwise reliable localization system. Table 4 shows statistics for the attacks deployed on our localization model. Though naive and FGSM attacks were effectively neutralized by adversarial training, the worst case attack remains extremely

effective, increasing median error on the defended model by 3.5×. If we compare adversarial attacks to the OOD data evaluation from Sect. 3.4.1, the impact of these attacks seems less significant. The median error in the worst case attack after adversarial training (126 m) is similar to the error in the OOD *Grid* case (118 m) and significantly less than the other train-test splits, ranging from 182 m to 336 m. In this case, the observable differences in data impact accuracy more than adversarial attacks.

The viability of worst case attacks is also questionable. In our attack we applied the worst case attack to the exact model used for localization, something an adversary would not reasonably have access to. It is possible that exploring worst case attacks could reveal a simpler strategy that would not require an omniscient setting. It remains to be shown if an adversary could use the set of worst case attacks to develop a more efficient method of generating similarly effective attacks without model access or full crowd information.

One of our chief objectives was to bound the efficacy of informed attacks on the lower end by naive attacks, and by omniscient attacks on the upper end. It seems that though the worst case attack increases error significantly, it is also the least realistic attack. With adversarial training, the median error is still less than 10% of the width of our region of interest, which could still be useful information to narrow down the search space. Compared to the OOD experiments, the worst case attack is similar in localization error. Additionally, the worst case attack may not be effective in practice. If we consider techniques for removing adversarial samples, it seems likely that single, worst case attacks would be easily identified and removed using statistical techniques or a classifier model.

An expectation we had at the outset of this work was that adversarial training would improve localization accuracy. We assumed that providing robustness to noise injection attacks would help improve robustness to existing noise in the training and test set. There was a modest improvement from adversarial training of 1.3 m on average. Research from computer vision [2, 16, 32] suggests that robustness to underlying noise cannot be provided by augmentations such as adversarial training. This appears to hold true for our localization setting.

Generalization to Practical Localization Our dataset, while extensive, does not reflect all localization scenarios. For example, with such a low sensor density, an adversary controlling 1 or 2 sensors represents 10–20% of all sensors. With a higher sensor density, it is possible that single sensor worst case attacks would be largely ineffective, since a single fake sensor would have a more limited influence.

6 Location Privacy Concerns

As seen earlier in the chapter, crowdsourced users or devices participating in distributed sensing to assist with localization typically report their location and their radio measurements to a fusion center (e.g., [11]). They may or may not explicitly

report their identity. In many cases it is possible to infer the identity of the user based on its location. Therefore, the location privacy of the participants is a serious concern [8, 12, 37]. Users can be linked to their locations, and multiple pieces of such information over a period of time can be correlated to profile users for unsolicited targeted advertisements or price discrimination [4]. Even worse, a user's habits, personal and private preferences, religious beliefs, and political affiliations, can be inferred from the user's whereabouts. Therefore, users who are willing to participate in the crowdsourcing system for societal good or some incentives can be uncomfortable and choose to not participate.

One traditional approach to preserving location privacy is to add noise to the location with the hope that the measured data would still be useful and would not severely reduce the localization accuracy [9]. A better approach called the *adjusted measurement* approach, proposed by Singh et al. [28], generates pseudolocations and report the pseudolocations along with adjusted measurements, achieved through appropriate propagation models, as if the measurements were made at the pseudolocations. This method has been shown to work better in terms of reducing the localization error in comparison to the traditional methods of only adding noise to the location [28].

Location privacy is not the primary focus of this chapter and hence we do not discuss that in greater details. This section provides a high level view of the location privacy concerns as well as a novel solution towards mitigating these concerns.

7 Looking Forward

The rapid evolution of localization techniques has paved the way for the practical implementation of localization systems using crowdsourcing, but there are still crucial areas of development that demand attention before such a system could be deployed. Specifically, models must become more robust to adversarial attacks and to out-of-distribution data from environmental and hardware changes. Here we consider a few key challenges for researchers and practitioners to consider in the future.

Adversarial Attacks and Training While our case study provided valuable insights into the efficacy of attacks and limiting their impact, it is essential to explore adversarial attacks in a variety of settings, such as with a greater degree of sensor mobility, higher sensor density, different environments, and with varying crowdsourcing models and mechanisms.

We have shown that adversarial training can provide robustness to certain types of attacks, but it is vital to continue exploring novel adversarial attacks that may expose specific weaknesses of localization models.

Improving Generalization While adversarial training is one method to make localization models more robust, it fails to address the chief issue of poor localization on out-of-distribution data. One of the primary challenges in improving generalization

is the lack of diverse datasets for crowdsourced localization. To our knowledge there are only two extensive outdoor datasets for localization [1, 17], both of which have a very low sensor density, and neither of which use true crowdsourcing. Complex ray-tracing models may be accurate enough to produce simulated "ground truth" data for training localization models, but the degree to which these models represent RF propagation in a real world environment is not well studied.

Localization with Directional Transmitters All the localization methods discussed in this work have assumed an omnidirectional antenna on both transmitter and receiver. Although CUTL shows that CNN localization is effective even when some sensors have an irregular receiver pattern [19], there are significant challenges to overcome if transmitters use beamforming to focus signal strength in a particular direction. This would introduce transmitter direction as an additional variable in the learning problem, and would drastically reduce the number of sensors which would receive the signal.

8 Conclusion

The utilization of crowdsourcing for localization, especially localization of spectrum offenders, has opened up new avenues for practical implementation and improved accuracy. This chapter explored various localization techniques, including recent ML-based approaches. We also provided a case study which demonstrated the vulnerability of localization systems to adversarial attacks, as well as showing adversarial training to be a reasonably successful and convenient defense mechanism.

Crowdsourcing presents a promising approach to localization, but it requires addressing privacy concerns and improving resilience against adversarial attacks. Researchers and practitioners must continue to refine localization models, advance defense strategies, and expand datasets in order to achieve robust and reliable localization systems.

References

1. Aernouts M, Berkvens R, Van Vlaenderen K, Weyn M (2018) Sigfox and LoRaWAN datasets for fingerprint localization in large urban and rural areas. Zenodo https://doi.org/105281/zenodo1193563
2. Azulay A, Weiss Y (2019) Why do deep convolutional networks generalize so poorly to small image transformations? J Mach Learn Res 20:1–25
3. Bahl P, Padmanabhan V (2000) Radar: an in-building rf-based user location and tracking system. In: Proceedings IEEE INFOCOM 2000. Conference on computer communications. 19th annual joint conference of the IEEE computer and communications societies, vol 2, pp 775–784

4. Beatrix Cleff E (2007) Privacy issues in mobile advertising. Int Rev Law Comput Technol 21(3):225–236. https://doi.org/10.1080/13600860701701421
5. Brunker M (2016) Gps under attack as crooks, rogue workers wage electronic war. https://www.nbcnews.com, Accessed 28 Jul 2023
6. Chintalapudi K, Padmanabha Iyer A, Padmanabhan VN (2010) Indoor localization without the pain. In: Proceedings of the 16th annual international conference on mobile computing and networking, ACM, pp 173–184
7. CISA (2022) Cisa insights: global positioning system (gps) interference. Technical Repot, Cybersecurity and Infrastructure Security Agency, https://www.cisa.gov/sites/default/files/publications/CISA-Insights_GPS-Interference_508.pdf, accessed 28 Jul 2023
8. Freudiger J, Shokri R, Hubaux JP (2011) Evaluating the privacy risk of location-based services. In: International conference on financial cryptography and data security, Springer, pp 31–46
9. Geng Q, Kairouz P, Oh S, Viswanath P (2015) The staircase mechanism in differential privacy. IEEE J Sel Topics Signal Process 9(7):1176–1184
10. Goodfellow IJ, Shlens J, Szegedy C (2014) Explaining and harnessing adversarial examples. Preprint at https://arxivorg/abs/14126572
11. Khaledi M, Khaledi M, Sarkar S, Kasera S, Patwari N, Derr K, Ramirez S (2017) Simultaneous power-based localization of transmitters for crowdsourced spectrum monitoring. In: Proceedings of the 23rd annual international conference on mobile computing and networking, pp 235–247
12. Krumm J (2007) Inference attacks on location tracks. In: International conference on pervasive computing, Springer, pp 127–143
13. Krumm J, Platt J (2003) Minimizing calibration effort for an indoor 802.11 device location measurement system. Technical Report, MSR-TR-2003-82, Microsoft Research, https://www.microsoft.com/en-us/research/publication/minimizing-calibration-effort-for-an-indoor-802-11-device-location-measurement-system/
14. Kurakin A, Goodfellow I, Bengio S (2016) Adversarial machine learning at scale. Preprint at https://arxivorg/abs/161101236
15. Mani F, Vitucci EM, Barbiroli M, Fuschini F, degli Esposti V, Gan M, Li C, Zhao J, Zhong Z (2018) 26ghz ray-tracing pathloss prediction in outdoor scenario in presence of vegetation. In: 12th European conference on antennas and propagation (EuCAP 2018), pp 1–5, https://doi.org/10.1049/cp.2018.0384
16. Mintun E, Kirillov A, Xie S (2021) On interaction between augmentations and corruptions in natural corruption robustness. Adv Neural Inf Process Syst 34:3571–3583
17. Mitchell F, Baset A, Kasera SK, Bhaskara A (2022) A dataset of outdoor RSS measurements for localization. Zenodo https://doi.org/105281/zenodo7259895
18. Mitchell F, Baset A, Patwari N, Kasera SK, Bhaskara A (2022) Deep learning-based localization in limited data regimes. In: Proceedings of the 2022 ACM workshop on wireless security and machine learning, pp 15–20
19. Mitchell F, Patwari N, Kasera SK, Bhaskara A (2023) Learning-based techniques for transmitter localization: a case study on model robustness. In: 20th Annual IEEE international conference on sensing, communication, and networking (SECON)
20. Mitchell F, Smith P, Bhaskara A, Kasera SK (2023) Exploring adversarial attacks on learning-based localization. In: Proceedings of the 2023 ACM workshop on wireless security and machine learning, pp 15–20
21. Papernot N, McDaniel P, Goodfellow I, Jha S, Celik ZB, Swami A (2017) Practical black-box attacks against machine learning. In: Proceedings of the 2017 ACM on Asia conference on computer and communications security, pp 506–519
22. Patwari N, Hero A (2006) Signal strength localization bounds in ad hoc and sensor networks when transmit powers are random. In: 4th IEEE workshop on sensor array and multichannel processing, 2006, IEEE, pp 299–303
23. Patwari N, Hero AO, Perkins M, Correal NS, O'Dea RJ (2003) Relative location estimation in wireless sensor networks. IEEE Trans Signal Process 51(8):2137–2148

24. Patwari N, Ash JN, Kyperountas S, Hero AO, Moses RL, Correal NS (2005) Locating the nodes: cooperative localization in wireless sensor networks. IEEE Signal Process Mag 22(4):54–69
25. Rai A, Chintalapudi KK, Padmanabhan VN, Sen R (2012) Zee: Zero-effort crowdsourcing for indoor localization. In: Proceedings of the 18th annual international conference on mobile computing and networking, ACM, pp 293–304
26. Ronneberger O, Fischer P, Brox T (2015) U-net: Convolutional networks for biomedical image segmentation. In: International conference on medical image computing and computer-assisted intervention, Springer, pp 234–241
27. Sarkar S, Baset A, Singh H, Smith P, Patwari N, Kasera S, Derr K, Ramirez S (2020) Llocus: learning-based localization using crowdsourcing. In: Proceedings of the 21st international symposium on theory, algorithmic foundations, and protocol design for mobile networks and mobile computing, pp 201–210
28. Singh H, Sarkar S, Dimri A, Bhaskara A, Patwari N, Kasera S, Ramirez S, Derr K (2018) Privacy enabled crowdsourced transmitter localization using adjusted measurements. In: 2018 IEEE symposium on privacy-aware computing (PAC), pp 95–106. https://doi.org/10.1109/PAC.2018.00016
29. Sorour S, Lostanlen Y, Valaee S (2012) RSS based indoor localization with limited deployment load. In: 2012 IEEE global communications conference (GLOBECOM), IEEE, pp 303–308
30. Sorour S, Lostanlen Y, Valaee S, Majeed K (2015) Joint indoor localization and radio map construction with limited deployment load. IEEE Trans Mob Comput 14(5):1031–1043
31. Tadik S, Varner MA, Mitchell F, Durgin GD (2023) Augmented rf propagation modeling. IEEE J. Radio Freq Identif 7:211–221
32. Taori R, Dave A, Shankar V, Carlini N, Recht B, Schmidt L (2020) Measuring robustness to natural distribution shifts in image classification. Adv Neural Inf Process Syst 33:18583–18599
33. Tramer F, Boneh D (2019) Adversarial training and robustness for multiple perturbations. Adv Neural Inf Process Syst 32
34. Varner MA, Mitchell F, Wang J, Webb K, Durgin GD (2022) Enhanced rf modeling accuracy using simple minimum mean-squared error correction factors. In: 2022 IEEE 2nd international conference on digital twins and parallel intelligence (DTPI), IEEE, pp 1–5
35. Vitucci E, Degli-Esposti V, Fuschini F, Lu J, Barbiroli M, Wu J, Zoli M, Zhu J, Bertoni H, et al. (2015) Ray tracing rf field prediction: an unforgiving validation. Int J Antennas Propag 2015
36. Wang H, Sen S, Elgohary A, Farid M, Youssef M, Choudhury RR (2012) No need to war-drive: unsupervised indoor localization. In: Proceedings of the 10th international conference on mobile systems, applications, and services, ACM, pp 197–210
37. Want R, Hopper A, Falcao V, Gibbons J (1992) The active badge location system. ACM Trans Inf Syst (TOIS) 10(1):91–102
38. Yapar Ç, Levie R, Kutyniok G, Caire G (2023) Real-time outdoor localization using radio maps: a deep learning approach. IEEE Trans Wirel Commun 22(12):9703–9717
39. Yedavalli K, Krishnamachari B, Ravula S, Srinivasan B (2005) Ecolocation: A sequence based technique for RF-only localization in wireless sensor networks. In: Proceedings of the 4th international conference on information processing in sensor networks (IPSN '05)
40. Youssef M, Agrawala A (2005) The Horus WLAN location determination system. In: ACM MobiSys
41. Zhan C, Ghaderibaneh M, Sahu P, Gupta H (2021) Deepmtl: Deep learning based multiple transmitter localization. In: 2021 IEEE 22nd international symposium on a world of wireless, mobile and multimedia networks (WoWMoM), IEEE, pp 41–50
42. Zhan C, Ghaderibaneh M, Sahu P, Gupta H (2022) Deepmtl pro: deep learning based multiple transmitter localization and power estimation. Pervasive Mob Comput 82:101582
43. Zubow A, Bayhan S, Gawłowicz P, Dressler F (2020) Deeptxfinder: multiple transmitter localization by deep learning in crowdsourced spectrum sensing. In: 2020 29th international conference on computer communications and networks (ICCCN), IEEE, pp 1–8

Adversarial Online Reinforcement Learning Under Limited Defender Resources

Ming Shi, Yingbin Liang, and Ness B. Shroff

1 Introduction

Reinforcement learning (RL) is a powerful paradigm, where an agent interacts with an environment with the aim of finding a policy that optimizes the cumulative reward, e.g., throughput or average delay in a communication system. Recently, adversarial RL has become popular because of its ability to capture scenarios where the reward and/or the dynamics of the environment (i.e., the underlying transition probability distribution) change over time, possibly in an adversarial manner. For example, in communication networks where calls or flows arrive in a time-varying manner, the wireless environment (e.g., the signal to interference ratio or transmission success rate) may be non-stationary due to user mobility. Malicious users could also affect the channel conditions by injecting interference in an adversarial manner. Clearly, in order to perform well in such systems, the agent needs to change the policy accordingly over time, e.g., which network access point to send the packets to, or which base-stations in a cellular system to power down to save energy costs. However, the agent (who becomes the defender if system changes are adversarial) may not be able to afford frequent policy changes especially when

M. Shi · Y. Liang
Department of Electrical and Computer Engineering, The Ohio State University, Columbus, OH, USA
e-mail: shi.1796@osu.edu; liang.889@osu.edu

N. B. Shroff (✉)
Department of Electrical and Computer Engineering, The Ohio State University, Columbus, OH, USA

Department of Computer Science and Engineering, The Ohio State University, Columbus, OH, USA
e-mail: shroff.11@osu.edu

Y. Chen et al. (eds.), *Network Security Empowered by Artificial Intelligence*,
Advances in Information Security 107, https://doi.org/10.1007/978-3-031-53510-9_10

the reward and/or the dynamics of the environment change rapidly. For example, if the agent is an edge device, it may have energy limitations, and changing policies could incur a significant energy cost. To that end, we study adversarial RL that incorporates both switching costs and rewards as performance metrics.

In this chapter, we will first give an overview of adversarial RL without switching costs as the baseline, where the defender/agent is assumed to have unlimited power to change her policies all the time. We will then describe the state-of-the-art results for the adversarial bandit learning with switching costs, which is a special case of adversarial RL. After that, we will focus on our recent development on adversarial RL with switching costs, where switching-reduced algorithms are provided to achieve near-optimal performance (in terms of regret), together with important lower bounds that could guide future work. Finally, we will discuss open issues and future directions on adversarial online RL under limited defender resources.

2 An Overview of Adversarial RL Without Switching Costs

Reinforcement learning (RL) has arisen as a compelling paradigm for modeling machine learning applications with sequential decision making. In such a problem, an online agent interacts with the environment sequentially over Markov decision processes (MDPs) with the aim to achieve a low cumulative loss or a high cumulative reward. Various algorithms have been developed for RL problems and have been shown to achieve polynomial sample efficiency in [1–5], etc. However, these studies have mainly focused on *static/stochastic* RL, where the loss distribution is assumed to be fixed during the learning process. Thus, practical scenarios where the loss distribution could be non-stationary or even adversarial are not characterized or considered. For example, in communication networks where calls or flows arrive in a time-varying manner, the wireless environment (e.g., the signal to interference ratio or transmission success rate) may be non-stationary due to user mobility. Malicious users could also affect the channel conditions by injecting interference in an adversarial manner.

Adversarial RL better models scenarios where the loss distributions and/or the transition functions of MDPs could change over time. In adversarial RL, the online agent interacts with the Markov environment in K episodes (Fig. 1). There are H steps in each episode. At each layer $h = 0, ..., H-1$ of an episode $k = 1, ..., K$, after observing the current sate $s_{k,h}^{\pi_k}$, the defender chooses an action $a_{k,h}^{\pi_k} = \pi_k(s_{k,h}^{\pi_k})$

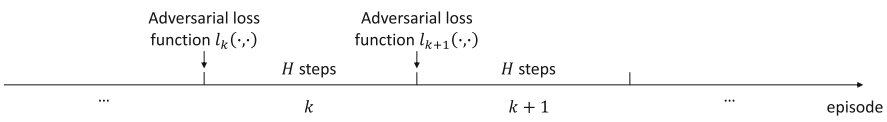


Fig. 1 Adversarial reinforcement learning problem

according to the current policy π_k. Then, the agent incurs a loss $l_k(s_{k,h}^{\pi_k}, a_{k,h}^{\pi_k})$. Importantly, the loss function $l_k(\cdot,\cdot)$ could *change adversarially or non-stationarily across episodes*. Finally, the next state $s_{k,h+1}^{\pi_k} \in \mathcal{S}_{h+1}$ is drawn according to the transition probability $P(\cdot|s_{k,h}^{\pi_k}, a_{k,h}^{\pi_k})$. The final goal is to achieve a low sub-linear regret, which is the worst-case difference between the expected total cost of algorithm $\pi = \{\pi_k\}_{k=1}^K$ and the total cost of the optimal policy π^*, i.e.,

$$R^{\pi}(T) \triangleq \sup_{l_{1:K}} \left\{ \mathbb{E}\left[\sum_{k=1}^{K} \sum_{h=0}^{H-1} l_k(s_{k,h}^{\pi_k}, a_{k,h}^{\pi_k}) - \sum_{k=1}^{K} \sum_{h=0}^{H-1} l_k(s_{k,h}^{\pi^*}, a_{k,h}^{\pi^*}) \middle| \pi, \pi^*, P \right] \right\}, \tag{1}$$

where $\pi^* = \arg\min_{\pi_0} \mathbb{E}\left[\sum_{k=1}^{K} \sum_{h=0}^{H-1} l_k(s_{k,h}^{\pi_0}, a_{k,h}^{\pi_0}) \middle| \pi_0, P \right]$.

Recent work has proposed many algorithms with sub-linear regret for different settings of adversarial RL. For example, in tabular MDP with a known transition function, [6] proposed an RL algorithm that attains an $\tilde{O}(\sqrt{HSAK})$ regret, where S and A are the number of states and actions. In the case with an unknown transition function, [7] and [8] obtained an $\tilde{O}\left(HS\sqrt{AK \ln \frac{KSA}{\delta}}\right)$ regret with probability $1-\delta$. These studies assume that the state spaces of layers in an episode are non-overlapping. Moreover, [9] studied the case with full-information feedback. Adversarial linear MDP has also been studied recently, e.g., in [10, 11]. In addition, [12, 13] and [14] studied the case when both the loss distribution and transition function change arbitrarily. More studies on various adversarial RL settings have been done by Rosenberg and Mansour [15], Lee et al. [16], Zhao et al. [17], Jin et al. [18], and He et al. [19], etc. However, they all allow the policies to be changed for free at any time, which will result in poor performance when the policy switches should be bounded, e.g., in the case with limited defender resources.

3 Adversarial Bandit Learning With Switching Costs

In various practical scenarios, an important performance metric is the switching cost of executing RL algorithms. For example, the online defenders cannot change their policies for free, especially in networking applications where the devices may have limited processing power. Further, in recommendation systems, each change of the recommendation involves the processing of a huge amount of data and additional computational costs [20]. Similarly, in healthcare, each change of the medical treatment requires substantial human efforts and time-consuming tests and trials [21]. Such switching costs also need to be considered in many other areas, e.g., robotics applications [22], education software [23], computer networking [24], and database optimization [25].

Switching costs have already received considerable attention in various online problems. For example, online convex optimization with switching costs has been studied in [26–30], etc. Convex body chasing with switching costs has been studied in [31–33], etc. Switching costs have also been studied in metrical task systems [34], online set covering [35], k-server problem [36], online control [37–39], etc.

3.1 Problem Formulation

A more relevant line of research for adversarial RL with switching costs is along adversarial bandit learning with switching costs [40–43]. Adversarial bandit learning is a special case of adversarial RL when $H = S = 1$, i.e., when there is only one step in each episode and only one state. Specifically, in adversarial bandit learning with switching costs, there are A arms, $\{1, 2, ..., A\}$. At each time t, the online agent chooses $M = 1$ arm according to algorithm π, denoted by $a^\pi(t)$, from these A arms. This chosen arm will then incur a loss $l_t(a^\pi(t))$. The loss function $l_t(\cdot)$ could change arbitrarily or non-stationarily across times. Additionally, if the arm $a^\pi(t)$ chosen at time t is different from the arm $a^\pi(t-1)$ chosen at time $t-1$, there is a switching cost β. Thus, the total cost for T time-slots is

$$\text{Cost}(1:T) \triangleq \sum_{t=1}^{T} l_t(a^\pi(t)) + \sum_{t=1}^{T-1} \beta \cdot \mathbf{1}_{\{a^\pi(t+1) \neq a^\pi(t)\}}. \tag{2}$$

For the optimal algorithm π^*, she knows the future losses in advance, and hence can choose only one arm throughout the time-horizon. The cost of π^* is then given by $\text{Cost}^{\pi^*}(1:T) = \min_{a \in \{1,2,...,A\}} \sum_{t=1}^{T} l_t(a) + \beta$, where there is only one switching cost β at the beginning of the time-horizon. The goal is to design an online learning algorithm with a low sub-linear regret, where the regret is the worst-case difference between the expected total cost of algorithm π and the total cost of the optimal offline solution, i.e.,

$$R^\pi(T) \triangleq \sup_{l_{1:T}} \left\{ \mathbb{E}\left[\text{Cost}^\pi(1:T) \right] - \text{Cost}^{\pi^*}(1:T) \middle| \pi \right\}, \tag{3}$$

3.2 Algorithm and Regret

It has been shown that the optimal regret in adversarial bandit learning with switching costs is $\Theta(T^{2/3})$ [41, 43]. The idea is to divide the time horizon into $\Theta(T^{2/3})$ episodes, and pull one *single* Exp3-arm in an episode. By doing so, the total switching cost is trivially $\Theta(T^{2/3})$. Meanwhile, the loss regret in an episode is $\Theta(\eta \cdot (T^{1/3})^2)$, which is proportional to the loss variance in an episode. The final

Algorithm 1 Episodic Exponential-Weight for Exploration and Exploitation (EpExp3)

Parameters: Choose $\eta = \Theta(T^{-1/3})$ and $\tau = \Theta(T^{1/3})$.
Initialization: $w_a^{\text{EpExp3}}[1] = 1$ and $p_a^{\text{EpExp3}}[1] = \frac{1}{K}$, for all $a \in \{1, ..., A\}$.
for $u = 1 : \lceil \frac{T}{\tau} \rceil$ (The u-th episode starts from $t_u = (u-1)\tau + 1$ to $t_u + \tau - 1$.) **do**
Step 1: At the beginning of the first time-slot, pick an arm for the entire episode: $a^{\text{EpExp3}}[1]$ from all arms $a \in \{1, 2, ..., A\}$ according to the probability $p_a^{\text{EpExp3}}[1]$.
for $t = t_u : t_u + \tau - 1$ **do**
Pull the arm $a^{\text{EpExp3}}[u]$ and use it as the active arm.
end for
Step 4: At the end of the last time-slot of the u-th episode, compute the losses for all arms $a \in \{1, 2, ..., A\}$ according to (4). Then, update the weights $w_a^{\text{EpExp3}}[u+1]$ and probabilities $p_a^{\text{EpExp3}}[u+1]$ according to (5) and (6), respectively.
end for

$\Theta(T^{2/3})$ regret is then achieved by taking the sum of all these costs and tuning the parameter $\eta = \Theta(T^{-2/3})$.

We call the algorithm Episodic Exponential-Weight for Exploration and Exploitation (EpExp3). Please see Algorithm 1 for details. Specifically, EpExp3 divides the time horizon into $\lceil \frac{T}{\tau} \rceil = \Theta(T^{2/3})$ episodes. At the beginning of each episode u, we choose an arm and use it as the active arm to incur the loss for the whole episode. At the end of the episode u, EpExp3 estimates the losses for all arms as follows,

$$\tilde{L}_a^{\text{EpExp3}}[u] = \begin{cases} \frac{L_a[u]}{p_a^{\text{EpExp3}}[u]}, & \text{if } a = a^{\text{EpExp3}}[u], \\ 0, & \text{if } a \neq a^{\text{EpExp3}}[u], \end{cases} \tag{4}$$

where $L_a[u] \triangleq \sum_{t=t_u}^{t_u+\tau-1} l_t(a)$, and $a^{\text{EpExp3}}[u]$ is the active arm used in the u-th episode. Next, using the computed losses, EpExp3 updates the weights and probabilities for all arms $a \in \{1, ..., A\}$ as follows,

$$w_a^{\text{EpExp3}}[u+1] = w_a^{\text{EpExp3}}[u] \cdot e^{-\eta \tilde{L}_k^{\text{EpExp3}}[u]}, \tag{5}$$

$$p_a^{\text{EpExp3}}[u+1] = \frac{w_a^{\text{EpExp3}}[u+1]}{\sum_{a=1}^{A} w_a^{\text{EpExp3}}[u+1]}, \tag{6}$$

where η is a tunable parameter (Fig. 2).

It is worth noting that, while the optimal regret when $M = 1$ is $\Theta(T^{2/3})$, [43] shows that a $O(\sqrt{T})$ regret is achievable when $M > 1$. This means that when a little more resource is provided to the defender, the regret can be significantly reduced for adversarial bandit learning with switching costs.

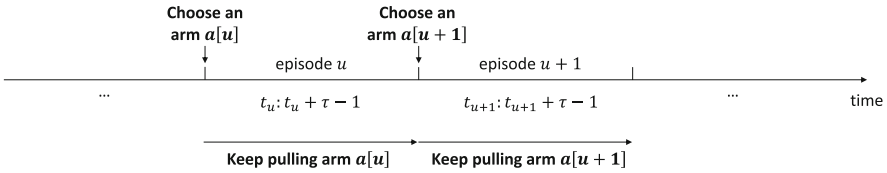


Fig. 2 The EpExp3 algorithm for adversarial bandit learning with switching costs

4 Adversarial RL With Switching Costs

In this section, we investigate how to develop a provably efficient algorithm for an online defender under adversarial RL with switching costs [44]. First, we provide the formulation for this problem. Then, we provide a lower bound of the regret and an interesting tradeoff between the loss regret and switching costs. Finally, we provide two switching-reduced algorithms with regrets that match the lower bound when the transition function is known, and match the lower bound within a small factor when the transition function is unknown.

4.1 Problem Formulation

We consider adversarial reinforcement learning (RL) with switching costs in episodic Markov decision processes (MDPs). Suppose there are T episodes, each of which consists of H layers. We use $\mathcal{S}_h$ to denote the state space of layer h. For ease of elaboration, we assume that the H layers are non-intersecting [6–8], i.e., $\mathcal{S}_{h'} \cap \mathcal{S}_{h''} = \phi$ for any $h' \neq h''$; $\mathcal{S}_0 = \{s_0\}$ is a singleton; and each episode ends at state $\mathcal{S}_H = \{s_H\}$. Thus, the entire state space is $\mathcal{S} = \cup_{h=0}^{H} \mathcal{S}_h$ with size $S = \sum_{h=0}^{H} S_h$, where S_h denotes the size of $\mathcal{S}_h$. Moreover, we use $\mathcal{A}$ to denote the action space with size A. Then, the MDP is defined by a tuple $\left(\mathcal{S}, \mathcal{A}, P, \{l_t\}_{t=1}^{T}, H\right)$, where P is the transition function with $P_h : \mathcal{S}_{h+1} \times \mathcal{S}_h \times \mathcal{A} \to [0, 1]$ denoting the transition probability measure at layer h, and $l_t : \mathcal{S} \times \mathcal{A} \to [0, 1]$ represents the loss function for episode t.

The online defender interacts with the Markov environment episode-by-episode as follows. At the beginning of each episode $t = 1, ..., T$, the online defender starts from state s_0 and follows an algorithm that (possibly randomly) chooses a deterministic policy $\pi_t : \mathcal{S} \to \mathcal{A}$. Next, at each layer $h = 0, ..., H - 1$, after observing the current sate $s_{t,h}$, the defender chooses an action $a_{t,h} = \pi_t(s_{t,h})$. Then, the defender incurs a loss $l_t(s_{t,h}, a_{t,h})$. Finally, the next state $s_{t,h+1} \in \mathcal{S}_{h+1}$ is drawn according to the transition probability $P(\cdot|s_{t,h}, a_{t,h})$. (For simplicity, we drop the index h of P_h in this chapter when it is clear from the context.) These steps repeat until the defender arrives at the last state s_H. At the end of episode t, only the losses of visited state-action pairs in the episode are observed by the defender, whereas the

losses of non-visited state-action pairs are unknown. As in [6–8, 10], this is called "bandit feedback", which is more practical than full-information feedback [9] that assumes the losses of all state-action pairs (no matter visited or not) are known for free.

Adversarial Losses for the Online Defender Different from static RL that assumes the loss distribution is fixed for all episodes, in the adversarial setting we consider here, we do not need any assumption on the underlying loss distribution. That is, the loss function l_t could change arbitrarily across episodes.

Switching Costs for the Online Defender The switching cost refers to the cost needed for changing the policy π_t. For example, if the agent is an edge device, it may have energy limitations, and changing policies could incur a significant energy cost. It is equal to $\beta \cdot \mathbf{1}_{\{\pi_{t+1} \neq \pi_t\}}$, where $\beta > 0$ is the switching-cost coefficient and is independent of T, and $\mathbf{1}_{\mathcal{E}}$ is an indicator function (i.e., $\mathbf{1}_{\mathcal{E}} = 1$ if the event $\mathcal{E}$ occurs, and $\mathbf{1}_{\mathcal{E}} = 0$ otherwise).

Therefore, the total cost of executing an RL algorithm π over T episodes is given by

$$\text{Cost}^{\pi}(1:T) \triangleq \mathbb{E}\left[\sum_{t=1}^{T}\sum_{h=0}^{H-1} l_t(s_{t,h}^{\pi}, a_{t,h}^{\pi}) + \sum_{t=1}^{T-1} \beta \cdot \mathbf{1}_{\{\pi_{t+1} \neq \pi_t\}} \Big| \pi, P\right], \tag{7}$$

where the expectation is taken with respect to the randomness of the state-action pairs $(s_{t,h}^{\pi}, a_{t,h}^{\pi})$ visited by π, and the possible randomness of changing the policy π_t.

Next, we introduce a concept called "occupancy measure" [6, 7]. Specifically, the occupancy measure $q_t^{\pi,P}(s,a) = Pr[s_{t,h}^{\pi} = s, a_{t,h}^{\pi} = a | \pi, P] \geq 0$ is the probability of visiting the state-action pair (s,a) by the algorithm π at layer h of episode t under the transition function P. In addition (with slight abuse of notation), the occupancy measure $q_t^{\pi,P}(s',s,a) = Pr[s_{t,h+1}^{\pi} = s', s_{t,h}^{\pi} = s, a_{t,h}^{\pi} = a | \pi, P] \geq 0$ is the probability of visiting the state-action triple (s',s,a) by the algorithm π at layers h and $h+1$ of episode t under the transition function P. In order to be feasible, the occupancy measures need to satisfy some conditions at layer h of episode t. First, according to probability theory, they need to satisfy the conditions that,

$$\begin{aligned} q_t^{\pi,P}(s,a) &= \sum_{s' \in \mathcal{S}_{h+1}} q_t^{\pi,P}(s',s,a), \text{ for all } (s,a) \in \mathcal{S}_h \times \mathcal{A}, \\ \sum_{s \in \mathcal{S}_h} \sum_{a \in \mathcal{A}} q_t^{\pi,P}(s,a) &= 1. \end{aligned} \tag{8}$$

Second, since the probability of transferring to a state s from the previous layer $h-1$ must be equal to the probability of transferring from this state s to the next layer $h+1$, we have

$$\sum_{s' \in \mathcal{S}_{h-1}} \sum_{a \in \mathcal{A}} q_t^{\pi,P}(s, s', a) = \sum_{s' \in \mathcal{S}_{h+1}} \sum_{a \in \mathcal{A}} q_t^{\pi,P}(s', s, a), \text{ for all } s \in \mathcal{S}_h. \tag{9}$$

Third, the occupancy measure should generate the true transition function P, i.e.,

$$\frac{q_t^{\pi,P}(s', s, a)}{\sum_{b \in \mathcal{A}} q_t^{\pi,P}(s', s, b)} = P_h(s'|s, a), \text{ for all } (s', s, a) \in \mathcal{S}_{h+1} \times \mathcal{S}_h \times \mathcal{A}. \tag{10}$$

We use $\mathbb{C}(P)$ to denote the set of all occupancy measures that satisfy conditions (8)–(10). Moreover, at the beginning of episode t, the algorithm π associated with the occupancy measure $q_t^{\pi,P}$ chooses a *deterministic* policy π_t by assigning an action $a \in \mathcal{A}$ to each state $s \in \mathcal{S}$ according to the probability

$$Pr[a|s] = \frac{q_t^{\pi,P}(s, a)}{\sum_{b \in \mathcal{A}} q_t^{\pi,P}(s, b)}. \tag{11}$$

Then, it is not hard to show that the expected total loss, i.e., the first term in (7), can be expressed as $\text{loss}^{\pi}(1 : T) \triangleq \mathbb{E}\left[\sum_{t=1}^{T} \langle q_t^{\pi,P}, l_t \rangle \middle| \pi, P\right]$. Finally, the regret of an RL algorithm π is defined to be the sum of the loss regret $R_{\text{loss}}^{\pi}(T)$ and the switching costs of as follows:

$$R^{\pi}(T) \triangleq \underbrace{\max_{q \in \mathbb{C}(P)} \mathbb{E}\left[\sum_{t=1}^{T} \langle q_t^{\pi,P} - q, l_t \rangle \middle| \pi, P\right]}_{\text{loss regret: } R_{\text{loss}}^{\pi}(T)} + \underbrace{\mathbb{E}\left[\sum_{t=1}^{T-1} \beta \cdot \mathbf{1}_{\{\pi_{t+1} \neq \pi_t\}} \middle| \pi, P\right]}_{\text{switching costs}}. \tag{12}$$

Therefore, the goal is to design RL algorithms that achieve as low regret as possible against any possible sequence of loss functions $\{l_t\}_{t=1}^{T}$ and state transition function P.

4.2 A Lower Bound

In this subsection, we will develop a lower bound on the regret for adversarial RL with switching costs. Such a lower bound will quantify how difficult it is for the online defender to control the regret with switching costs under adversarial RL. In Theorem 1 below, we provide this lower bound, followed by the proof of it. In Sects. 4.3 and 4.4, we will provide two near-optimal RL algorithms to achieve this lower bound.

Theorem 1 *For adversarial RL with switching costs and* $T \geq \max\{6H^2SA, \beta\}$, *the regret of any RL algorithm* π *can be lower-bounded as follows,*

$$R^{\pi}(T) \geq \tilde{\Omega}\left(\beta^{1/3}(HSA)^{1/3}T^{2/3}\right). \tag{13}$$

Theorem 1 shows that in adversarial RL with switching costs, the dependency on T of the best achievable regret is at least $\tilde{\Omega}(T^{2/3})$. Thus, the best achieved regret (whose dependency on T is $\tilde{O}(\sqrt{T})$) in *static* RL with switching costs (in [45, 46], etc) as well as adversarial RL *without* switching costs (in [2, 6], etc) is no longer achievable. This demonstrates the fundamental challenge of switching costs in adversarial RL, and it is expected that new challenges will arise when developing provably efficient algorithms.

Note that the bandit setting is a special case (when $S = H = 1$) of our MDP setting. Thus, the lower bound for the adversarial bandit setting in [43] serves as a lower bound in our MDP setting. However, the direct use of such a lower bound from bandits will not be good enough for the MDP case that we study in this chapter. To get the lower bound in Theorem 1, the most challenging and interesting part is to design the lower-bound instance. Notice that a lower-bound transition is constructed for stochastic MDP in [46], which shows that the MDP setting is at least as difficult as multi-armed bandits with $\Omega(HSA)$ arms, and then a similar lower bound can be obtained based on the lower bound from bandits. Below, we construct a new lower-bound instance. Specifically, we divide the state space $\mathcal{S}$ and construct special state transitions, such that the episodic reinforcement learning is reduced to $\Theta(S/H)$ chains of bandit learning. Notice that the lower-bound analysis in [43] implies that, with the loss function l_t upper-bounded by H, A arms and T time-slots, the regret of any bandit-learning algorithm with switching costs is at least $\tilde{\Omega}\left(\beta^{1/3}A^{1/3}(HT)^{2/3}\right)$ when $T \geq \max\{6H^2A, \beta\}$. Hence, the total regret from all $\Theta(S/H)$ chains of bandit learning is at least $\tilde{\Omega}\left(\beta^{1/3}A^{1/3}(H\frac{T}{S/H})^{2/3}\right) \cdot \Theta(S/H) = \tilde{\Omega}\left(\beta^{1/3}(HSA)^{1/3}T^{2/3}\right)$. Please see the detailed proof below.

Proof of Theorem 1

Lower-bound instance: We consider a special instance where $S-2$ is divisible by $H-1$. First, we assign the states in the state space $\mathcal{S}$ to each layer as follows. The first layer contains a single sate, i.e., $\mathcal{S}_0 = \{s_0\}$. All episodes end with state $\mathcal{S}_H = \{s_H\}$. Moreover, the rest of the $S-2$ states are assigned to each layer $h \in [1, H-1]$ evenly. That it, each layer $h \in [1, H-1]$ contains $\frac{S-2}{H-1}$ states. Following the sequence of the states at each layer, we call the index i of the i-th state the "order" of it. In addition, the order i of the states at layer h of any episode is the same, e.g., the first state at layer h is always the first state at layer h for all episodes, and the second state at layer h is always the second state at layer h for all episodes. Moreover, all actions are available at each state $s \in \mathcal{S}$. Finally, based on this construction of the states and actions, we run independently the lower-bound algorithm for adversarial bandit learning with switching costs in [43] as a subroutine through all i-th states, for all $i = 1, ..., \frac{S-2}{H-1}$. That is, for each layer $h = 1, ..., H-1$, $P_h(s_i|s_i, a) = 1$ for all a, and $P_h(s_j|s_i, a) = 0$ for all $j \neq i$ and all a.

Lower-bound analysis: The lower-bound analysis in [43] implies that, with the loss function l_t upper-bounded by H, A arms and T time-slots, the regret (including both the loss regret and switching costs) of any bandit-learning algorithm with switching costs is at least $\tilde{\Omega}\left(\beta^{1/3}A^{1/3}(HT)^{2/3}\right)$. Notice that based on our lower-bound instance constructed above, there are $\frac{S-2}{H-1}$ chains of bandit learning. Hence, the total regret of any RL algorithm π from all these $\frac{S-2}{H-1}$ chains of bandit learning can be lower-bounded as follows,

$$R^{\pi}(T) \geq \tilde{\Omega}\left(\beta^{1/3}A^{1/3}\left(H\frac{T}{\frac{S-2}{H-1}}\right)^{2/3}\right)\cdot\frac{S-2}{H-1} = \tilde{\Omega}\left(\beta^{1/3}(HSA)^{1/3}T^{2/3}\right). \tag{14}$$

□

Further, in Theorem 2 below, we characterize precisely the new trade-off between the loss regret and switching costs defined in (12), followed by the proof of it. Intuitively, by switching more, the online RL algorithm can adapt more flexibly to the new information learned, and thus achieves a lower loss regret. On the other hand, if fewer switches are allowed, the online RL algorithm is less flexible to adapt to the new information learned, which will incur a larger loss regret.

Theorem 2 *For adversarial RL with switching costs, with the switching costs equal to* $O\left(\beta\cdot\mathcal{N}^{swi}\right)$*, the loss regret can be lower-bounded by* $\tilde{\Omega}\left(\sqrt{\frac{HSA}{\mathcal{N}^{swi}}}\cdot T\right)$*. Alternatively, to achieve a loss regret equal to* $\tilde{O}\left(\sqrt{\frac{HSA}{\mathcal{N}^{swi}}}\cdot T\right)$*, the switching costs incurred must be larger than* $\Omega\left(\beta\cdot\mathcal{N}^{swi}\right)$.

Theorem 2 provides an interesting and necessary trade-off between the loss regret and switching costs. We further elaborate this result in three cases. First, in order to achieve a loss regret $\tilde{O}(H\sqrt{SAT})$, Theorem 2 shows that the number of switches $\mathcal{N}^{\text{swi}}$ (and thus the switching costs incurred) must be linear in T, i.e., essentially switching at almost all episodes. This is consistent with the regret achieved in adversarial RL *without* switching costs, i.e., allowing switching linear-to-T number of times for free. But our result further implies that, without linear-to-T switches of the policy, it is impossible to achieve an $\tilde{O}(\sqrt{T})$ loss regret. Second, Theorem 2 shows that, if only a constant or $O(\ln\ln T)$ number of switches are allowed, the loss regret must be linear in T. In contrast, in *static* RL, an $\tilde{O}(\sqrt{T})$ loss regret is achieved with only $O(\ln\ln T)$ switches [46]. This indicates that the adversarial nature of RL necessarily requires significantly more policy switches to achieve a low loss regret. Third, Theorem 2 suggests that the loss regret and switching costs can be balanced at the order of $\tilde{O}\left(T^{2/3}\right)$. That is, to achieve the $\tilde{O}\left(T^{2/3}\right)$ loss regret, the switching costs incurred have to be $\tilde{\Omega}\left(T^{2/3}\right)$. This is consistent with Theorem 1, where the regret (including both the loss regret and switching costs) is lowered-bound by $\tilde{\Omega}\left(T^{2/3}\right)$.

The proof of Theorem 2 follows the lower-bound proof above, but by considering the loss regret and switching costs separately.

Proof of Theorem 2 To prove Theorem 2, we use the lower-bound instance that we constructed above for proving Theorem 1. First, the lower-bound analysis in [43] implies that, for adversarial bandit learning with the loss function l_t upper-bounded by H, A arms and T time-slots, when the total switching cost is equal to $O(\beta \cdot \mathcal{N}^{\text{swi}})$, the loss regret can be lower-bounded by $\tilde{\Omega}\left(\sqrt{\frac{A}{\mathcal{N}^{\text{swi}}}} \cdot HT\right)$. Notice that there are $\frac{S-2}{H-1}$ chains of bandit learning in the lower-bound instance that we constructed above. Thus, with a total switching cost equal to $O(\beta \cdot \mathcal{N}^{\text{swi}}) \triangleq O(\beta \cdot \sum_{i=1}^{\frac{S-2}{H-1}} \mathcal{N}_i^{\text{swi}})$, the loss regret of any RL algorithm π against the lower-bound instance that we constructed above can be lower-bounded as follows,

$$R_{\text{loss}}^{\pi}(T) \geq \sum_{i=1}^{\frac{S-2}{H-1}} \tilde{\Omega}\left(\sqrt{\frac{A}{\mathcal{N}_i^{\text{swi}}}} \cdot H\frac{T}{\frac{S-2}{H-1}}\right) = \tilde{\Omega}\left(\sqrt{\frac{HSA}{\mathcal{N}^{\text{swi}}}} \cdot T\right),$$

where the equality is because $\sum_{i=1}^{\frac{S-2}{H-1}} \sqrt{\frac{1}{\mathcal{N}_i^{\text{swi}}}} \geq \sqrt{\frac{1}{\mathcal{N}^{\text{swi}}}} \left(\frac{S-2}{H-1}\right)^{3/2}$. Finally, the second half part of Theorem 2 is trivially true, since it is the converse-negative proposition of the first half part that we have proved above. □

4.3 The Case When the Transition Function Is Known

In this subsection, we study the case when the transition function is *known*, and we will further explore the more challenging case when the transition function is *unknown* in Sect. 4.4. We propose a novel algorithm (please see Algorithm 2) for the online defender with a regret that matches the lower bound in (13). Our algorithm is called Switching rEduced EpisoDic relative entropy policy Search (SEEDS).

SEEDS is inspired by the episodic method in bandit learning [43]. In bandit learning, the idea is to divide the time horizon into $\Theta(T^{2/3})$ episodes, and pull one *single* Exp3-arm in an episode. By doing so, the total switching cost is trivially $O(T^{2/3})$. Meanwhile, the loss regret in an episode is $\Theta(\eta \cdot (T^{1/3})^2)$, which is proportional to the loss variance in an episode. The final $O(T^{2/3})$ regret is then achieved by taking the sum of all these costs and tuning the parameter $\eta = \Theta(T^{-2/3})$. However, in the adversarial MDP setting that we consider, there is a key difference due to random state-action visitations that cause several new challenges as we discuss in the rest of this section.

Super-Episode-Based Policy Search SEEDS divides the episodes into $\mathcal{U} = \lceil \frac{T}{\tau} \rceil$ super-episodes, where $\tau \in \mathbb{Z}_{++}$ is a tunable parameter and a strictly positive integer. Each super-episode includes τ consecutive episodes. For all episodes in

Algorithm 2 Switching rEduced EpisoDic relative entropy policy Search (SEEDS)

Parameters: $\eta = \tilde{\Theta}\left(\beta^{-1/3} H^{2/3} (SA)^{-1/3} T^{-2/3}\right)$ and $\tau = \tilde{\Theta}\left(\beta^{2/3} (HSA)^{-1/3} T^{1/3}\right)$.
Initialization: $Pr[a|s] = \frac{1}{A}$ for all $(s, a) \in \mathcal{S} \times \mathcal{A}$. Choose $\pi_{[1]}^{\text{SEEDS}}$ according to (11).
for $u = 1 : \lceil \frac{T}{\tau} \rceil$ **do**
  for $t = (u-1)\tau + 1 : \min\{u\tau, T\}$ **do**
    Step 1: Execute the updated policy $\pi_{[u]}^{\text{SEEDS}} = \pi^{\hat{q}_{[u]}^{\text{SEEDS}, P}}$.
  end for
  At the end of super-episode u,
  Step 2: Estimate the losses $\hat{l}_{[u]}^{\text{SEEDS}}(s, a)$ for all (s, a) according to (15).
  Step 3: Update the occupancy measure $\hat{q}_{[u+1]}^{\text{SEEDS}, P}(s, a)$ according to (19). Update the deterministic policy $\pi^{\hat{q}_{[u+1]}^{\text{SEEDS}, P}}$ according to (11).
end for

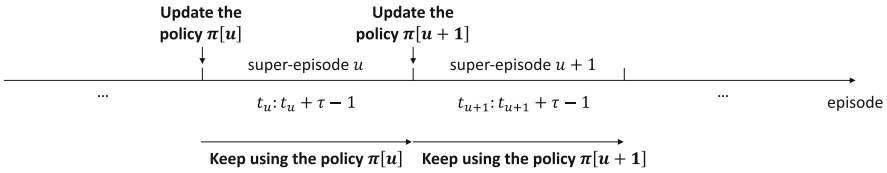


Fig. 3 The SEEDS algorithm for adversarial RL with switching costs

each super-episode $u = 1, ..., \mathcal{U}$, SEEDS uses the same policy $\pi^{\hat{q}_{[u]}^{\text{SEEDS}, P}}$ (*Step-1* in Algorithm 2) that was updated at the end of the last super-episode $u - 1$, where $\hat{q}_{[u]}^{\text{SEEDS}, P}$ is the updated occupancy measure (that we will introduce soon) of SEEDS for super-episode u. Thus, SEEDS switches the policy at most once in each super-episode (Fig. 3).

The Idea for Estimating the Losses At the end of super-episode u, SEEDS estimates the losses $l_{[u]}(s, a)$ of all state-action pairs in super-episode u. Here, it is instructive to see why the episodic importance-estimating method in adversarial bandit learning (i.e., without state transitions) does not apply to our problem. Note that due to state transitions in our more general MDP setting, we are not guaranteed to visit a *single* state-action pair for the whole super-episode. A naive but intuitive solution may be pretending that each state-action pair visited in super-episode u was the *single* one visited. Then, we can let the estimated loss of each state-action pair (s, a) to be $\hat{l}_{[u]}(s, a) = \frac{\bar{l}_{[u]}(s,a)}{1-(1-\hat{q}_{[u]}^{\text{SEEDS}, P}(s,a))^{\tau}} \mathbf{1}_{\{(s,a) \text{ was visited in super-episode } u\}}$, where the numerator $\bar{l}_{[u]}(s, a) = \sum_{t=(u-1)\tau+1}^{u\tau} l_t(s, a)/\tau$ is the average loss of (s, a). If we assume that the loss l_t for all episodes t in super-episode u were the same, according to the analysis in bandit learning and the inequality $1-(1-x)^{\tau} \geq x$ for all $0 \leq x \leq 1$, this idea would have worked. However, the problem is that, inside super-episode u, the loss function l_t for each episode t could change arbitrarily. Thus, the estimated loss $\hat{l}_{[u]}(s, a)$ above is actually unknown and an ill-defined value.

To resolve the aforementioned difficulty due to randomly-visited state-action pairs and arbitrarily-changing loss functions, SEEDS estimates the loss as follows (*Step-2* in Algorithm 2),

$$\hat{l}_{[u]}^{\text{SEEDS}}(s,a) = \sum_{j=1}^{J_{[u]}} \frac{l_{t_j(s,a)}(s,a)}{\hat{q}_{[u]}^{\text{SEEDS},P}(s,a)} \mathbf{1}_{\{(s,a):t_1(s,a),\ldots,t_{J_{[u]}}(s,a)\}}, \tag{15}$$

where $\mathbf{1}_{\{(s,a):t_1(s,a),\ldots,t_{J_{[u]}}(s,a)\}}$ is an indicator function for whether (s,a) was visited in episodes $t_1(s,a)$, ..., $t_{J_{[u]}}(s,a)$ of super-episode u, and $J_{[u]}$ is the maximum number of episodes that the state-action pair (s,a) was visited in super-episode u. In other words, in super-episode u, this state-action pair (s,a) was not visited in any other episode t, such that $t \in \{(u-1)\tau+1, \ldots, u\tau\}/\{t_1(s,a), \ldots, t_{J_{[u]}}(s,a)\}$. Thus, SEEDS estimates the losses based on the observable true losses in super-episode u. In this way, SEEDS elegantly resolves the aforementioned difficulty due to the random state transitions and adversarial losses. Our novel idea in (15) may be of independent interest for other problems with state transitions and non-stationary or adversarial losses. Indeed, in Sect. 4.4, we will apply this idea to the case when the transition function is unknown.

In Lemma 1 below, we show that the estimated loss in (15) is an unbiased estimation of the true loss in super-episode u. This is an important property that we will exploit in our regret analysis. We use $\mathcal{F}_{[u]}$ to denote the σ-algebra generated by the observation of SEEDS before super-episode u.

Lemma 1 *The conditional expectation of the estimated loss designed in (15) is equal to*

$$\mathbb{E}\left[\hat{l}_{[u]}^{SEEDS}(s,a)\middle|\mathcal{F}_{[u]}\right] = l_{[u]}(s,a), \text{ for all } (s,a), \tag{16}$$

where the expectation is taken with respect to the randomness of the episodes $t_1(s,a)$, ..., $t_{J_{[u]}}(s,a)$, *in which the state-action pair* (s,a) *was visited, and* $l_{[u]}(s,a) = \sum_{t=(u-1)\tau+1}^{\min\{u\tau,T\}} l_t(s,a)$ *is the true loss of* (s,a) *in super-episode* u.

Proof of Lemma 1 First, since the expectation is taken with respect to the randomness of the episodes $t_1(s,a)$, ..., $t_{J_{[u]}}(s,a)$, in which the state-action pair (s,a) was visited, the left-hand-side of (16), $\mathbb{E}\left[\hat{l}_{[u]}^{\text{SEEDS}}(s,a)\middle|\mathcal{F}_{[u]}\right]$, is equal to

$$\sum_{\substack{\{t_1(s,a),\ldots,t_{J_{[u]}}(s,a)\} \\ \subseteq[(u-1)\tau+1,u\tau]}} \hat{l}_{[u]}^{\text{SEEDS}}(s,a) \cdot Pr\left[\{t_1(s,a),\ldots,t_{J_{[u]}}(s,a)\}\middle|\mathcal{F}_{[u]}\right].$$

Next, according to the definition of the estimated loss that we design in (15), we have

$$\mathbb{E}\left[\hat{l}_{[u]}^{\text{SEEDS}}(s,a)\middle|\mathcal{F}_{[u]}\right] = \sum_{\substack{\{t_1(s,a),\ldots,t_{J_{[u]}}(s,a)\}\\ \subseteq[(u-1)\tau+1,u\tau]}} \sum_{j=1}^{J_{[u]}} \frac{l_{t_j(s,a)}(s,a)}{\hat{q}_{[u]}^{\text{SEEDS},P}(s,a)}$$
$$\cdot \mathbf{1}_{\{(s,a):t_1(s,a),\ldots,t_{J_{[u]}}(s,a)\}} \cdot Pr\left[\{t_1(s,a),\ldots,t_{J_{[u]}}(s,a)\}\middle|\mathcal{F}_{[u]}\right].$$

In the following, we prove that

$$\sum_{\substack{\{t_1(s,a),\ldots,t_{J_{[u]}}(s,a)\}\\ \subseteq[(u-1)\tau+1,u\tau]}} \sum_{j=1}^{J_{[u]}} \frac{l_{t_j(s,a)}(s,a)}{\hat{q}_{[u]}^{\text{SEEDS},P}(s,a)} \cdot \mathbf{1}_{\{(s,a):t_1(s,a),\ldots,t_{J_{[u]}}(s,a)\}}$$
$$\cdot Pr\left[\{t_1(s,a),\ldots,t_{J_{[u]}}(s,a)\}\middle|\mathcal{F}_{[u]}\right] = \sum_{t=(u-1)\tau+1}^{u\tau} \hat{q}_{[u]}^{\text{SEEDS},P}(s,a) \cdot \frac{l_t(s,a)}{\hat{q}_{[u]}^{\text{SEEDS},P}(s,a)}.$$

That is, under our design of the estimated loss in (15), summing over all possible sets of the random episodes where the state-action pair was visited (i.e., the outer sum on the left-hand-side) is equivalent to summing over all deterministic episodes from the beginning to the end of a super-episode (i.e., the sum on the right-hand-side).

This is because first, relying on the above indicator function on the left-hand-side, the sum of the total *observed* loss in a super-episode over all possible sets $\{t_1(s,a),\ldots,t_{J_{[u]}}(s,a)\}$ is equivalent to the sum of the total *true* loss in each episode of a super-episode based on whether the episode is observed. Therefore, we have

$$\mathbb{E}\left[\hat{l}_{[u]}^{\text{SEEDS}}(s,a)\middle|\mathcal{F}_{[u]}\right] = \sum_{t=(u-1)\tau+1}^{u\tau} \sum_{\substack{\{t_1(s,a),\ldots,t_{J_{[u]}}(s,a)\}:\\ t\in\{t_1(s,a),\ldots,t_{J_{[u]}}(s,a)\}}} \frac{l_t(s,a)}{\hat{q}_{[u]}^{\text{SEEDS},P}(s,a)}$$
$$\cdot Pr\left[\{t_1(s,a),\ldots,t_{J_{[u]}}(s,a)\}\middle|\mathcal{F}_{[u]}\right]. \tag{17}$$

In addition, since the transition function P is known, conditioned on $\mathcal{F}_{[u]}$, the probability of visiting each state-action pair (s,a) in an episode t of super-episode u is equal to the occupancy measure $\hat{q}_{[u]}^{\text{SEEDS},P}(s,a)$, i.e.,

$$\sum_{\substack{\{t_1(s,a),\ldots,t_{J_{[u]}}(s,a)\}:\\ t\in\{t_1(s,a),\ldots,t_{J_{[u]}}(s,a)\}}} Pr\left[\{t_1(s,a),\ldots,t_{J_{[u]}}(s,a)\}\middle|\mathcal{F}_{[u]}\right] = \hat{q}_{[u]}^{\text{SEEDS},P}(s,a). \tag{18}$$

Finally, by combining (17) and (18), we have

$$\mathbb{E}\left[\hat{l}_{[u]}^{\text{SEEDS}}(s,a)\middle|\mathcal{F}_{[u]}\right] = \sum_{t=(u-1)\tau+1}^{u\tau} \hat{q}_{[u]}^{\text{SEEDS},P}(s,a) \cdot \frac{l_t(s,a)}{\hat{q}_{[u]}^{\text{SEEDS},P}(s,a)}$$
$$= \sum_{t=(u-1)\tau+1}^{u\tau} l_t(s,a) = l_{[u]}(s,a).$$

□

Updating the Occupancy Measure Finally, according to online mirror descent [6, 47], SEEDS updates the occupancy measure $\hat{q}_{[u+1]}^{\text{SEEDS},P}(s,a)$ for all state-action pairs $(s,a) \in \mathcal{S} \times \mathcal{A}$ as follows (*Step-3* in Algorithm 2),

$$\hat{q}_{[u+1]}^{\text{SEEDS},P} = \arg\min_{q \in \mathbb{C}(P)} \left\{ \eta \cdot \left\langle q, \hat{l}_{[u]}^{\text{SEEDS}} \right\rangle + D_{\text{KL}}\left(q \middle\| \hat{q}_{[u]}^{\text{SEEDS},P}\right) \right\}, \tag{19}$$

where $D_{\text{KL}}(q\|q') \triangleq \sum_{s\in\mathcal{S},a\in\mathcal{A}} q(s,a) \ln \frac{q(s,a)}{q'(s,a)} - \sum_{s\in\mathcal{S},a\in\mathcal{A}} \left[q(s,a) - q'(s,a)\right]$ is the unnormalized relative entropy between two occupancy measures q and q' on the space $\mathcal{S} \times \mathcal{A}$. Recall that $\mathbb{C}(P)$ is formulated by (8)–(10). Note that the term $\langle q, \hat{l}_{[u]}^{\text{SEEDS}} \rangle$ represents the expected loss in super-episode u, with respect to the newly-estimated loss function $\hat{l}_{[u]}^{\text{SEEDS}}$. Thus, it captures how SEEDS adapts to and explores the newly-estimated loss function. In addition, the term $D_{\text{KL}}(q\|\hat{q}_{[u]}^{\text{SEEDS},P})$ serves as a regularizer to ensure that the updated occupancy measure in (19) stays close to $\hat{q}_{[u]}^{\text{SEEDS},P}$. Thus, it captures how SEEDS exploits the previously-estimated loss functions before super-episode u. As a result, by tuning the parameter η in (19), the updated occupancy measure strikes a balance between exploration and exploitation.

We characterize the regret of SEEDS in Theorem 3 below.

Theorem 3 *Consider adversarial RL with switching costs introduced in Sect. 4.1. When the transition function P is known, the regret of SEEDS is upper-bounded as follows,*

$$R^{SEEDS}(T) \leq \tilde{O}\left(\beta^{1/3}(HSA)^{1/3}T^{2/3}\right). \tag{20}$$

Theorem 3 shows that the regret of SEEDS matches the lower bound in (13) in terms of the dependency on all the parameters T, S, A, H and β. Thus, the regret of SEEDS is order-wise optimal. *To the best of our knowledge, this is the first regret result for adversarial RL with switching costs.*

Since the total switching cost of SEEDS is trivially upper-bounded by $\beta \cdot \lceil \frac{T}{\tau} \rceil$, to prove Theorem 3, we focus on upper-bounding the loss regret of SEEDS, i.e.,

$$R_{\text{loss}}^{\text{SEEDS}}(T) = \max_{q \in \mathbb{C}(P)} \mathbb{E}\left[\sum_{t=1}^{T}\left\langle q_t^{\text{SEEDS},P} - q, l_t\right\rangle \middle| \text{SEEDS}, P\right]$$
$$\triangleq \mathbb{E}\left[\sum_{t=1}^{T}\left\langle q_t^{\text{SEEDS},P} - q^{\pi^*}, l_t\right\rangle \middle| \text{SEEDS}, P\right].$$

To upper-bound the loss regret, the main difficulty lies in capturing the effects of the arbitrarily-changing losses and multiple random visitations of each state-action pair in a super-episode. To overcome this difficulty, our proof of Theorem 3 first upper-bounds the loss regret based on the correlated loss feedback in a super-episode (which relies on our new design of the estimated loss in (15) and Lemma 1), and then relates these upper bounds across all super-episodes to the final regret (which relies on another lemma, Lemma 2 below, which transfers the original regret formulation to a form based on the losses from the entire super-episode).

Specifically, for each super-episode, we first relate the true occupancy measure $q_t^{\text{SEEDS},P}$ to the unconstrained solution $\tilde{q}_{[u+1]}^{\text{SEEDS},P}$ to (19). Then, we relate $\tilde{q}_{[u+1]}^{\text{SEEDS},P}$ to the optimal offline occupancy measure q^{π^*}. The gaps between them are upper-bound mainly by using Lemma 1. Finally, by combining all the loss gaps (according to Lemma 2 and super-episodic version of online mirror descent) and the switching-cost upper-bound $\beta\left\lceil\frac{T}{\tau}\right\rceil$, and tuning the parameters η and τ as in Algorithm 2, we can get the regret of SEEDS in Theorem 3 and the trade-off in Theorem 4. Please see the detailed proof below.

Proof of Theorem 3

Step-1 (Bounding the switching costs): Since SEEDS switches at most once in each super-episode, the total switching cost of SEEDS is upper-bounded by $\beta \cdot \left\lceil\frac{T}{\tau}\right\rceil$. In the following, we focus on upper-bounding the loss regret $R_{\text{loss}}^{\text{SEEDS}}(T)$.

Step-2 (Bounding the loss regret): First, since SEEDS applies the same occupancy measure for all episodes t of the same super-episode u and the transition function P is known, conditioned on the history before super-episode u, the true occupancy measures of these episodes are the same. Then, according to Lemma 2 below, we can transfer the original regret formulation to a form based on the losses from the entire super-episode.

Lemma 2 *The loss regret $R_{loss}^{SEEDS}(T)$ of SEEDS is equal to*

$$\mathbb{E}\left[\sum_{t=1}^{T}\left\langle q_t^{SEEDS,P} - q^{\pi^*}, l_t\right\rangle\right] = \mathbb{E}\left[\sum_{u=1}^{\mathcal{U}}\left\langle q_{[u]}^{SEEDS,P} - q^{\pi^*}, l_{[u]}\right\rangle\right]. \tag{21}$$

Note that the occupancy measure and loss on the left-hand-side of (21) are for each episode t, while those on the right-hand-side of (21) are for each super-episode u.

Next, we use $\tilde{q}_{[u+1]}^{\text{SEEDS},P}$ to denote the unconstrained solution to (19), i.e.,

$$\tilde{q}_{[u+1]}^{\text{SEEDS},P} \triangleq \arg\min_{q} \left\{ \eta \cdot \left\langle q, \hat{l}_{[u]}^{\text{SEEDS}} \right\rangle + D_{\text{KL}} \left(q \left\| \hat{q}_{[u]}^{\text{SEEDS},P} \right. \right) \right\}.$$

Notice that $\hat{q}_{[u+1]}^{\text{SEEDS},P}$ is the constrained solution to (19), where the constraint is $q \in \mathbb{C}(P)$. It is not hard to get that

$$\tilde{q}_{[u+1]}^{\text{SEEDS},P}(s,a) = \hat{q}_{[u]}^{\text{SEEDS},P}(s,a) \cdot e^{-\eta \hat{l}_{[u]}^{\text{SEEDS}}(s,a)}. \tag{22}$$

To get (22), let us consider the function $f(q) = \eta \cdot \left\langle q, \hat{l}_{[u]}^{\text{SEEDS}} \right\rangle + D_{\text{KL}} \left(q \left\| \hat{q}_{[u]}^{\text{SEEDS},P} \right. \right)$. According to the definition of $D_{\text{KL}}(q \| q')$ right after (19), the derivative of function $f(q)$ is

$$\frac{\partial f(q)}{\partial q(s,a)} = \eta \cdot \hat{l}_{[u]}^{\text{SEEDS}}(s,a) + \ln \frac{q(s,a)}{\hat{q}_{[u]}^{\text{SEEDS},P}(s,a)}.$$

By letting the derivative to be 0 and rearranging the terms, we have (22).

Then, because of Lemma 2 and the fact that the calculated occupancy measure $\hat{q}_{[u]}^{\text{SEEDS},P}$ is equal to the true occupancy measure $q_{[u]}^{\text{SEEDS},P}$, we have

$$\mathbb{E}\left[\sum_{t=1}^{T} \left\langle q_t^{\text{SEEDS},P} - q^{\pi^*}, l_t \right\rangle \right] = \mathbb{E}\left[\sum_{u=1}^{\mathcal{U}} \left\langle \hat{q}_{[u]}^{\text{SEEDS},P} - q^{\pi^*}, l_{[u]} \right\rangle \right].$$

According to the linearity of expectation, we can decompose the loss regret into two terms that are easier to be bounded as follows,

$$\begin{aligned}
\mathbb{E}\left[\sum_{t=1}^{T} \left\langle q_t^{\text{SEEDS},P} - q^{\pi^*}, l_t \right\rangle \right] &= \sum_{u=1}^{\mathcal{U}} \mathbb{E}\left[\left\langle \hat{q}_{[u]}^{\text{SEEDS},P} - q^{\pi^*}, l_{[u]} \right\rangle \right] \\
&= \sum_{u=1}^{\mathcal{U}} \mathbb{E}_{\mathcal{F}_{[u]}} \left[\mathbb{E}\left[\left\langle \hat{q}_{[u]}^{\text{SEEDS},P} - q^{\pi^*}, l_{[u]} \right\rangle \middle| \mathcal{F}_{[u]}, P \right] \right] \\
&= \sum_{u=1}^{\mathcal{U}} \mathbb{E}_{\mathcal{F}_{[u]}} \left[\mathbb{E}\left[\left\langle \hat{q}_{[u]}^{\text{SEEDS},P} - \tilde{q}_{[u+1]}^{\text{SEEDS},P}, l_{[u]} \right\rangle \middle| \mathcal{F}_{[u]}, P \right] \right] \\
&\quad + \sum_{u=1}^{\mathcal{U}} \mathbb{E}_{\mathcal{F}_{[u]}} \left[\mathbb{E}\left[\left\langle \tilde{q}_{[u+1]}^{\text{SEEDS},P} - q^{\pi^*}, l_{[u]} \right\rangle \middle| \mathcal{F}_{[u]}, P \right] \right],
\end{aligned} \tag{23}$$

where the second equality is because $\mathbb{E}[X] = \mathbb{E}[\mathbb{E}[X|Y]]$, the last equality is because of the linearity of the expectation, and we drop the condition on SEEDS since it is clear from the context.

Below, we focus on upper-bounding the two terms on the right-hand-side of (23) one-by-one.

Step-2-i (Bounding the First Term) Since $e^x \geq 1 + x$, from (22) we have

$$\hat{q}_{[u]}^{\text{SEEDS},P}(s,a) - \tilde{q}_{[u+1]}^{\text{SEEDS},P}(s,a) \leq \eta \hat{q}_{[u]}^{\text{SEEDS},P}(s,a) \cdot \hat{l}_{[u]}^{\text{SEEDS}}(s,a).$$

Thus, the first term on the right-hand-side of (23) can be upper-bounded as follows,

$$\begin{aligned}
&\sum_{u=1}^{\mathcal{U}} \mathbb{E}_{\mathcal{F}_{[u]}} \left[\mathbb{E} \left[\left\langle \hat{q}_{[u]}^{\text{SEEDS},P} - \tilde{q}_{[u+1]}^{\text{SEEDS},P}, l_{[u]} \right\rangle \middle| \mathcal{F}_{[u]}, P \right] \right] \\
&\leq \sum_{u=1}^{\mathcal{U}} \mathbb{E}_{\mathcal{F}_{[u]}} \left[\mathbb{E} \left[\sum_{s\in\mathcal{S}, a\in\mathcal{A}} \eta \hat{q}_{[u]}^{\text{SEEDS},P}(s,a) \cdot \hat{l}_{[u]}^{\text{SEEDS}}(s,a) \cdot l_{[u]}(s,a) \middle| \mathcal{F}_{[u]}, P \right] \right].
\end{aligned}$$

Then, according to the definition of the estimated loss that we design in (15), we have

$$\begin{aligned}
&\sum_{u=1}^{\mathcal{U}} \mathbb{E}_{\mathcal{F}_{[u]}} \left[\mathbb{E} \left[\left\langle \hat{q}_{[u]}^{\text{SEEDS},P} - \tilde{q}_{[u+1]}^{\text{SEEDS},P}, l_{[u]} \right\rangle \middle| \mathcal{F}_{[u]}, P \right] \right] \\
&\leq \sum_{u=1}^{\mathcal{U}} \mathbb{E}_{\mathcal{F}_{[u]}} \left[\mathbb{E} \left[\sum_{s\in\mathcal{S}, a\in\mathcal{A}} \eta \hat{q}_{[u]}^{\text{SEEDS},P}(s,a) \sum_{j=1}^{J_{[u]}} \frac{l_{t_j(s,a)}(s,a)}{\hat{q}_{[u]}^{\text{SEEDS},P}(s,a)} \right. \right. \\
&\qquad \left. \left. \cdot \mathbf{1}_{\{(s,a):t_1(s,a),\ldots,t_{J_{[u]}}(s,a)\}} \cdot l_{[u]}(s,a) \middle| \mathcal{F}_{[u]}, P \right] \right] \\
&\leq \sum_{u=1}^{\mathcal{U}} \mathbb{E}_{\mathcal{F}_{[u]}} \left[\mathbb{E} \left[\sum_{s\in\mathcal{S}, a\in\mathcal{A}} \eta \left(l_{[u]}(s,a) \right)^2 \middle| \mathcal{F}_{[u]}, P \right] \right] \leq \eta SA \left\lceil \frac{T}{\tau} \right\rceil \tau^2, \qquad (24)
\end{aligned}$$

where the second inequality is because $\sum_{j=1}^{J_{[u]}} l_{t_j(s,a)}(s,a) \leq \sum_{t=(u-1)\tau+1}^{\min\{u\tau,T\}} l_t(s,a) = l_{[u]}(s,a)$, and the last inequality is because $l_{[u]}(s,a) \leq \tau$ and $\mathcal{U} = \lceil \frac{T}{\tau} \rceil$.

Step-2-ii (Bounding the Second Term) According to online mirror descent [6, 47], we have the following inequality for the unconstrained solution $\tilde{q}_{[u+1]}^{\text{SEEDS},P}$ to (19),

$$\left\langle q - \tilde{q}_{[u+1]}^{\text{SEEDS},P}, \eta \cdot \hat{l}_{[u]}^{\text{SEEDS}} + \frac{\partial D_{\text{KL}}(q \| \hat{q}_{[u]}^{\text{SEEDS},P})}{\partial q} \Big|_{q=\tilde{q}_{[u+1]}^{\text{SEEDS},P}} \right\rangle \geq 0, \text{ for all } q.$$

Since $\left.\frac{\partial D_{\mathrm{KL}}(q \| \hat{q}_{[u]}^{\mathrm{SEEDS},P})}{\partial q}\right|_{q=\tilde{q}_{[u+1]}^{\mathrm{SEEDS},P}} = \ln\left(\frac{\tilde{q}_{[u+1]}^{\mathrm{SEEDS},P}}{\hat{q}_{[u]}^{\mathrm{SEEDS},P}}\right)$, by rearranging the terms, we have

$$\left\langle \tilde{q}_{[u+1]}^{\mathrm{SEEDS},P} - q, \eta \cdot \hat{l}_{[u]}^{\mathrm{SEEDS}} \right\rangle \leq \left\langle q - \tilde{q}_{[u+1]}^{\mathrm{SEEDS},P}, \ln\left(\frac{\tilde{q}_{[u+1]}^{\mathrm{SEEDS},P}}{\hat{q}_{[u]}^{\mathrm{SEEDS},P}}\right)\right\rangle, \text{ for all } q.$$

By adding and subtracting terms on the right-hand-side, we have

$$\begin{aligned}
&\left\langle \tilde{q}_{[u+1]}^{\mathrm{SEEDS},P} - q, \eta \cdot \hat{l}_{[u]}^{\mathrm{SEEDS}} \right\rangle \\
&\leq \left[\sum_{s\in\mathcal{S},a\in\mathcal{A}} q(s,a) \ln \frac{q(s,a)}{\hat{q}_{[u]}^{\mathrm{SEEDS},P}(s,a)} - \sum_{s\in\mathcal{S},a\in\mathcal{A}} \left[q(s,a) - \hat{q}_{[u]}^{\mathrm{SEEDS},P}(s,a) \right] \right] \\
&- \left[\sum_{s\in\mathcal{S},a\in\mathcal{A}} \left[\tilde{q}_{[u+1]}^{\mathrm{SEEDS},P}(s,a) \ln \frac{\tilde{q}_{[u+1]}^{\mathrm{SEEDS},P}(s,a)}{\hat{q}_{[u]}^{\mathrm{SEEDS},P}(s,a)} \right.\right. \\
&\quad \left.\left. - \tilde{q}_{[u+1]}^{\mathrm{SEEDS},P}(s,a) + \hat{q}_{[u]}^{\mathrm{SEEDS},P}(s,a) \right] \right] \\
&+ \left[\sum_{s\in\mathcal{S},a\in\mathcal{A}} \left(q(s,a) - \hat{q}_{[u]}^{\mathrm{SEEDS},P}(s,a) \right) + \sum_{s\in\mathcal{S},a\in\mathcal{A}} q(s,a) \ln \frac{\tilde{q}_{[u+1]}^{\mathrm{SEEDS},P}(s,a)}{q(s,a)} \right. \\
&\quad \left. - \sum_{s\in\mathcal{S},a\in\mathcal{A}} \left(\tilde{q}_{[u+1]}^{\mathrm{SEEDS},P}(s,a) - \hat{q}_{[u]}^{\mathrm{SEEDS},P}(s,a) \right) \right] \\
&= D_{\mathrm{KL}}\left(q \| \hat{q}_{[u]}^{\mathrm{SEEDS},P} \right) - D_{\mathrm{KL}}\left(\tilde{q}_{[u+1]}^{\mathrm{SEEDS},P} \| \hat{q}_{[u]}^{\mathrm{SEEDS},P} \right) - D_{\mathrm{KL}}\left(q \| \tilde{q}_{[u+1]}^{\mathrm{SEEDS},P} \right).
\end{aligned}$$

Then, together with Lemma 1, we have

$$\begin{aligned}
&\sum_{u=1}^{\mathcal{U}} \mathbb{E}_{\mathcal{F}_{[u]}} \left[\mathbb{E}\left[\left\langle \tilde{q}_{[u+1]}^{\mathrm{SEEDS},P} - q^{\pi^*}, l_{[u]} \right\rangle \middle| \mathcal{F}_{[u]}, P \right] \right] \\
&\leq \frac{1}{\eta} \cdot \sum_{u=1}^{\mathcal{U}} \mathbb{E}_{\mathcal{F}_{[u]}} \left[\mathbb{E}\left[D_{\mathrm{KL}}\left(q \middle\| \hat{q}_{[u]}^{\mathrm{SEEDS},P} \right) - D_{\mathrm{KL}}\left(\tilde{q}_{[u+1]}^{\mathrm{SEEDS},P} \middle\| \hat{q}_{[u]}^{\mathrm{SEEDS},P} \right) \right.\right. \\
&\qquad \left.\left. - D_{\mathrm{KL}}\left(q \middle\| \tilde{q}_{[u+1]}^{\mathrm{SEEDS},P} \right) \middle| \mathcal{F}_{[u]}, P \right] \right].
\end{aligned}$$

Since the intermediate terms get cancelled and the relative entropy is always non-negative, the second term on the right-hand-side of (23) can be upper-bounded as follows,

$$\sum_{u=1}^{\mathcal{U}} \mathbb{E}_{\mathcal{F}_{[u]}} \left[\mathbb{E} \left[\left\langle \tilde{q}_{[u+1]}^{\text{SEEDS},P} - q^{\pi^*}, l_{[u]} \right\rangle \middle| \mathcal{F}_{[u]}, P \right] \right] \leq \frac{D_{\text{KL}}(q \| \hat{q}_{[1]}^{\text{SEEDS},P})}{\eta} \leq \frac{H}{\eta} \ln \frac{SA}{H}. \tag{25}$$

Step-3 (Final step): Finally, by combining (24), (25) and the switching-cost upper-bound $\beta \cdot \lceil \frac{T}{\tau} \rceil$, and tuning the parameters η and τ as in Algorithm 2, we have that the regret of SEEDS is upper-bounded by $O\left(\beta^{1/3} (HSA)^{1/3} T^{2/3}\right)$.

□

Further, in Theorem 4 below, we show that SEEDS attains a trade-off between the loss regret and switching costs that matches the trade-off in Theorem 2. The proof of Theorem 4 follows the loss-regret bound of SEEDS and the trivial switching-cost bound $\beta \cdot \lceil \frac{T}{\tau} \rceil$.

Theorem 4 *Let* $\mathcal{N}^{SEEDS} \triangleq \lceil \frac{T}{\tau} \rceil$. *Then, with the switching costs equal to* $O\left(\beta \cdot \mathcal{N}^{SEEDS}\right)$, *SEEDS can achieve a loss regret upper-bounded by* $\tilde{O}\left(\sqrt{\frac{HSA}{\mathcal{N}^{SEEDS}}} \cdot T\right)$.

Proof of Theorem 4 According to (24) and (25) above, with the total switching cost equal to $O\left(\beta \cdot \lceil \frac{T}{\tau} \rceil\right) = O(\beta \cdot \mathcal{N}^{\text{SEEDS}})$, the loss regret of SEEDS is upper-bounded as follows,

$$R_{\text{loss}}^{\text{SEEDS}}(T) \leq \tilde{O}\left(\eta SAT\tau + \frac{H}{\eta}\right) = \tilde{O}\left(\sqrt{HSAT\tau}\right) = \tilde{O}\left(\sqrt{\frac{HSA}{\mathcal{N}^{\text{SEEDS}}}} \cdot T\right), \tag{26}$$

where the first equality is by tuning $\eta = \sqrt{\frac{H}{SAT\tau}}$, and the last equality is because $\mathcal{N}^{\text{SEEDS}} \triangleq \lceil \frac{T}{\tau} \rceil$. □

4.4 The Case When the Transition Function Is Unknown

In this subsection, we study a more challenging case when the transition function is *unknown*. We propose another algorithm (please see Algorithm 3) for the online defender with a regret that matches the lower bound in (13) in terms of the dependency on all parameters, except with a small factor of $\tilde{O}(H^{1/3})$. Specifically, to address the new difficulty due to the *unknown* transition function P in this case, we advance SEEDS into SEEDS-UT (where UT stands for "unknown transition") with three new components as we explain below.

Algorithm 3 SEEDS-Unknown Transition (SEEDS-UT)

Parameters: $\eta = \tilde{\Theta}\left(\beta^{-1/3}H^{1/3}(SA)^{-1/3}T^{-2/3}\right)$, $\tau = \tilde{\Theta}\left(\beta^{2/3}H^{-2/3}(SA)^{-1/3}T^{1/3}\right)$, $\gamma = \tilde{\Theta}\left(\beta^{1/3}H^{2/3}(SA)^{-2/3}T^{-1/2}\right)$, and $0 < \delta < 1$.

Initialization: $\hat{q}_{[1]}^{\text{SEEDS-UT},\mathcal{P}}(s', s, a) = \frac{1}{S_{h+1}S_hA}$ and $M_{[1]}(s', s, a) = N_{[1]}(s, a) = 0$, for all $(s', s, a) \in \mathcal{S}_{h+1} \times \mathcal{S}_h \times \mathcal{A}$ and all h. $\mathcal{P}_{[1]}$ contains all possible transition functions. Choose $\pi_{[1]}^{\text{SEEDS-UT}} = \pi^{\hat{q}_{[1]}^{\text{SEEDS-UT},\mathcal{P}}}$ according to (8) and (11).

for $u = 1 : \lceil \frac{T}{\tau} \rceil$ **do**

  for $t = (u-1)\tau + 1 : \min\{u\tau, T\}$ **do**

    Step 1: Execute the updated policy $\pi_{[u]}^{\text{SEEDS-UT}} = \pi^{\hat{q}_{[u]}^{\text{SEEDS-UT},\mathcal{P}}}$.

  end for

  At the end of super-episode u,

  Step 2: Estimate the losses $\hat{l}_{[u]}^{\text{SEEDS-UT}}(s, a)$ for all (s, a) according to (27).

  Step 3: Estimate the transition-function set $\mathcal{P}_{[u+1]}$ according to (29).

  Step 4: Update the occupancy measure $\hat{q}_{[u+1]}^{\text{SEEDS-UT},\mathcal{P}}(s', s, a)$ according to (19), but subject to a different constraint $q \in \mathbb{C}\left(\mathcal{P}_{[u+1]}\right)$. Update the deterministic policy $\pi^{\hat{q}_{[u+1]}^{\text{SEEDS-UT},\mathcal{P}}}$ according to (8) and (11).

end for

1. Since the transition function P is unknown, updating the occupancy measure $\hat{q}(s, a)$ (as in SEEDS) is not good enough. Instead, SEEDS-UT updates the occupancy measure $\hat{q}(s', s, a)$ to take state transitions into consideration.
2. Since the transition function P is unknown, the updated occupancy measure could be different from the true one. To resolve this issue, we generalize the method in [48], with a difference to handle the random sequence of the state-action pairs visited in each super-episode. Specifically, SEEDS-UT estimates the loss for each super-episode u as follows (*Step-2* in Algorithm 3),

$$\hat{l}_{[u]}^{\text{SEEDS-UT}}(s, a) = \sum_{j=1}^{J_{[u]}} \frac{l_{t_j(s,a)}(s, a)}{Q_{[u]}^{\gamma}(s, a)} \mathbf{1}_{\{(s,a):t_1(s,a),\ldots,t_{J_{[u]}}(s,a)\}}, \tag{27}$$

where $Q_{[u]}^{\gamma}(s, a) \triangleq \max_{q \in \mathbb{C}(\mathcal{P}_{[u]})} q(s, a) + \gamma$ is the sum of the largest probability of visiting (s, a) among all occupancy measures in $\mathbb{C}(\mathcal{P}_{[u]})$ and a tunable parameter $\gamma > 0$, and $\mathcal{P}_{[u]}$ is a transition-function set that we will introduce soon. Note that (27) is another application of our idea in (15) for estimating losses in a problem with state transitions and adversarial losses.

In Lemma 3 below, we show that the gap between the expectation of the estimated loss and the true loss is controlled by the parameter γ. The proof of Lemma 3 is similar to that of Lemma 1.

Lemma 3 *The conditional expectation of the estimated loss designed in (27) is equal to*

$$\mathbb{E}\left[\hat{l}_{[u]}^{SEEDS\text{-}UT}(s,a)\middle|\mathcal{F}_{[u]}\right]=\frac{q_{[u]}^{SEEDS\text{-}UT,P}(s,a)}{\max_{q\in\mathbb{C}(\mathcal{P}_{[u]})}q(s,a)+\gamma}\cdot l_{[u]}(s,a),\ \textit{for all}\ (s,a), \tag{28}$$

where the expectation is taken with respect to the randomness of the episodes $t_1(s,a)$, ..., $t_{J_{[u]}}(s,a)$, *in which* (s,a) *was visited,* $q_{[u]}^{SEEDS\text{-}UT,P}(s,a)$ *is the true occupancy measure of SEEDS-UT conditioned on* $\mathcal{F}_{[u]}$, *and* $l_{[u]}(s,a)=\sum_{t=(u-1)\tau+1}^{\min\{u\tau,T\}}l_t(s,a)$ *is the true loss of* (s,a) *in super-episode* u.

Lemma 3 shows that, as long as $\mathcal{P}_{[u]}$ is sufficiently good for estimating the true transition function P (we will show how to construct such a $\mathcal{P}_{[u]}$ below), by carefully tuning γ, the bias caused by $\max_{q\in\mathbb{C}(\mathcal{P}_{[u]})}q(s,a)+\gamma$ (i.e., $\mathcal{Q}_{[u]}^{\gamma}(s,a)$) should be sufficiently small, so that the estimated loss is still sufficiently accurate.

3. Since the transition function P is unknown, the constraint in (19) is no longer known. To resolve this issue, we generalize the method in [7], with a difference to handle the samples from the whole super-episode. Specifically, at the end of each super-episode, SEEDS-UT collects the samples from the whole super-episode to update the empirical transition probability $\bar{P}_{[u+1]}(s'|s,a)=\frac{M_{[u+1]}(s',s,a)}{\max\{N_{[u+1]}(s,a),1\}}$, where $M_{[u+1]}(s',s,a)$ and $N_{[u+1]}(s,a)$ denote the number of times visiting (s',s,a) and (s,a) before super-episode $u+1$, respectively. Then, based on the empirical Bernstein bound [49], SEEDS-UT constructs a transition-function set $\mathcal{P}$ as follows (*Step-3* in Algorithm 3),

$$\mathcal{P}_{[u+1]}=\left\{\hat{P}_{[u+1]}:\left|\hat{P}_{[u+1]}(s'|s,a)-\bar{P}_{[u+1]}(s'|s,a)\right|\le\epsilon_{[u+1]}(s',s,a),\ \text{for all}\ (s',s,a)\right\}, \tag{29}$$

where $\epsilon_{[u+1]}(s',s,a)=2\sqrt{\frac{\bar{P}_{[u+1]}(s',s,a)\ln\frac{TSA}{\delta}}{\max\{N_{[u+1]}(s,a)-1,1\}}}+\frac{14\ln\frac{TSA}{\delta}}{3\max\{N_{[u+1]}(s,a)-1,1\}}$, and $\delta\in(0,1)$ is the confidence parameter. Finally, the occupancy measure $\hat{q}_{[u+1]}^{\text{SEEDS-UT},\mathcal{P}}(s',s,a)$ is updated according to (19), but subject to a different constraint $q\in\mathbb{C}\left(\mathcal{P}_{[u+1]}\right)$ (*Step-4* in Algorithm 3).

We characterize the regret of SEEDS-UT in Theorem 5 below.

Theorem 5 *Consider adversarial RL with switching costs introduced in Sect. 4.1. When the transition function* P *is unknown, with probability* $1-\delta$, *the regret of SEEDS-UT is upper-bounded as follows,*

$$R^{SEEDS\text{-}UT}(T)\le\tilde{O}\left(\beta^{1/3}H^{2/3}(SA)^{1/3}T^{2/3}\left(\ln\tfrac{TSA}{\delta}\right)^{1/2}\right). \tag{30}$$

Theorem 5 shows that the regret of SEEDS-UT matches the lower bound in (13) in terms of the dependency on T, S, A, and β, except with a small factor of $\tilde{O}(H^{1/3})$. That is, the regret of SEEDS-UT is near-optimal. *To the best of our knowledge, this is the first regret result for adversarial RL with switching cost when the transition*

function is unknown. To prove Theorem 5, the main difficulty is that, due to the delayed switching and unknown transition function, the losses of SEEDS-UT in the episodes of any super-episode are correlated and the true occupancy measure is unknown. As a result, the existing analytical ideas in adversarial RL without switching costs and adversarial bandit learning with switching costs do not work here. To overcome these new difficulties, our analysis involves several new ideas, e.g., we construct a series in (40) to handle multiple random visitations of each state-action pairs, and we establish a super-episodic version of concentration in *Step-2-iii* of Appendix 5 by relating the second-order moment of the estimated loss that we design to the true loss and the length τ of a super-episode. Please see Appendix 5 for the detailed proof of Theorem 5.

5 Conclusion and Future Work

In this chapter, we gave an overview of adversarial RL without switching costs as the baseline, where the defender/agent is assumed to have unlimited power to change her policies all the time. We then described the state-of-the-art results for the adversarial bandit learning with switching costs, which is a special case of adversarial RL. After that, we focused on our recent development on adversarial RL with switching costs, where switching-reduced algorithms are provided to achieve near-optimal performance (in terms of regret), together with important lower bounds that could guide future work.

Several future directions are worth pursuing. First, it is important to study more general adversarial online RL under limited defender resources, e.g., adversarial RL with switching costs in linear and more general MDP settings. Another interesting future work is to extend our study to the dynamic regret, which allows the optimal algorithm to change the defending policy over time.

Appendix: Proof of Theorem 5

Proof

Step-1 (Bounding the switching costs): Since SEEDS-UT switches at most once in each super-episode, the total switching cost of SEEDS-UT is upper-bounded by $\beta \cdot \lceil \frac{T}{\tau} \rceil$. In the following, we focus on upper-bounding the loss regret $R_{\text{loss}}^{\text{SEEDS-UT}}(T)$.

Step-2 (Bounding the loss regret): We first show Lemma 4 below. Lemma 4 is critical for Lemma 2 to be true in this case with an unknown transition function.

Lemma 4 *For any two episodes t_1 and t_2, if the updated occupancy measures are the same, i.e., $\hat{q}_{t_1}(s', s, a) = \hat{q}_{t_2}(s', s, a)$ for any (s', s, a), then the true occupancy*

measures are the same, i.e., $q_{t_1}(s,a) = q_{t_2}(s,a) = q_{[u]}(s,a)$ *for any* (s,a)*, where* $q_{[u]}(s,a)$ *is the true occupancy measure for the super-episode* u*.*

The proof of Lemma 4 follows the conditions in (8)–(11). Since SEEDS-UT applies the same occupancy measure $\hat{q}_{[u]}^{\text{SEEDS-UT},\mathcal{P}}$ for all episodes t of the same super-episode u, according to Lemma 4, the true occupancy measure $q_t^{\text{SEEDS-UT},P}$ of these episodes t are the same. Thus, similar to the case with a known transition function, we can get an unknown-transition version of Lemma 2 here. Thus,

$$\mathbb{E}\left[\sum_{t=1}^{T}\left\langle q_t^{\text{SEEDS-UT},P} - q^{\pi^*}, l_t\right\rangle \middle| P\right] = \mathbb{E}\left[\sum_{u=1}^{\mathcal{U}}\left\langle q_{[u]}^{\text{SEEDS-UT},P} - q^{\pi^*}, l_{[u]}\right\rangle \middle| P\right].$$

We drop the condition on SEEDS-UT in the expectation here and in the following when it is clear from the context.

According to the linearity of expectation, we can decompose the loss regret into four terms that are easier to be bounded, i.e.,

$$\begin{aligned}
&\mathbb{E}\left[\sum_{t=1}^{T}\left\langle q_t^{\text{SEEDS-UT},P} - q^{\pi^*}, l_t\right\rangle \middle| P\right] = \sum_{u=1}^{\mathcal{U}}\mathbb{E}\left[\left\langle q_{[u]}^{\text{SEEDS-UT},P} - q^{\pi^*}, l_{[u]}\right\rangle \middle| P\right] \\
&= \sum_{u=1}^{\mathcal{U}}\left\{\mathbb{E}_{\mathcal{F}_{[u]}}\left[\mathbb{E}\left[\left\langle q_{[u]}^{\text{SEEDS-UT},P} - \hat{q}_{[u]}^{\text{SEEDS-UT},\mathcal{P}}, l_{[u]}\right\rangle\right.\right.\right. \\
&\quad + \left\langle \hat{q}_{[u]}^{\text{SEEDS-UT},\mathcal{P}}, l_{[u]} - \hat{l}_{[u]}^{\text{SEEDS-UT}}\right\rangle \\
&\quad \left.\left.\left. + \left\langle \hat{q}_{[u]}^{\text{SEEDS-UT},\mathcal{P}} - q^{\pi^*}, \hat{l}_{[u]}^{\text{SEEDS-UT}}\right\rangle + \left\langle q^{\pi^*}, \hat{l}_{[u]}^{\text{SEEDS-UT}} - l_{[u]}\right\rangle \middle| \mathcal{F}_{[u]}, P\right]\right]\right\}.
\end{aligned} \tag{31}$$

Below, we focus on upper-bounding the four terms on the right-hand-side of (31) one-by-one.

Step-2-i (Bounding the First Term): Since $l_t(s,a) \leq 1$ for all state-action pairs (s,a), we have $l_{[u]}(s,a) \leq \tau$ for all (s,a). Thus, we have

$$\begin{aligned}
&\left\langle q_{[u]}^{\text{SEEDS-UT},P} - \hat{q}_{[u]}^{\text{SEEDS-UT},\mathcal{P}}, l_{[u]}\right\rangle \\
&\leq \tau \cdot \sum_{s\in\mathcal{S},a\in\mathcal{A}}\left|q_{[u]}^{\text{SEEDS-UT},P}(s,a) - \hat{q}_{[u]}^{\text{SEEDS-UT},\mathcal{P}}(s,a)\right|.
\end{aligned}$$

The difference between the true occupancy measure and the updated occupancy measure on the right-hand-side depends on how good the transition-function set $\mathcal{P}$ in (29) is, and can be further upper-bounded by using Bernstein inequality [49]. Below, we focus on bounding this difference. We use $\tilde{\pi}(a|s)$ to denote the probability of

choosing action a at state s. Specifically, first, according to the relation between the occupancy measure and the transition function in (10), we have that for any state-action pair $(s_h, a_h) \in \mathcal{S}_h \times \mathcal{A}$ visited at stage h,

$$q^{\pi,P}(s_h, a_h) = \tilde{\pi}(a_h|s_h) \sum_{(s_i \in \mathcal{S}_i, a_i \in \mathcal{A})_{i=0}^{h-1}} \prod_{j=0}^{h-1} \left[\tilde{\pi}(a_j|s_j) P(s_{j+1}|s_j, a_j)\right],$$

where for simplicity, we drop the index t for the states s and actions a. Thus, the difference between the updated occupancy measure and the true occupancy measure can be upper-bounded as follows,

$$\begin{aligned}\left|\hat{q}_{[u]}^{\text{SEEDS-UT},\mathcal{P}}(s_h, a_h) - q_{[u]}^{\text{SEEDS-UT},P}(s_h, a_h)\right| &= \tilde{\pi}_{[u]}^{\text{SEEDS-UT}}(a_h|s_h) \\ &\cdot \sum_{(s_i \in \mathcal{S}_i, a_i \in \mathcal{A})_{i=0}^{h-1}} \prod_{j=0}^{h-1} \tilde{\pi}_{[u]}^{\text{SEEDS-UT}}(a_j|s_j) \left[\prod_{j=0}^{h-1} \hat{P}_{[u]}(s_{j+1}|s_j, a_j)\right. \\ &\left. - \prod_{j=0}^{h-1} P(s_{j+1}|s_j, a_j)\right], \end{aligned} \tag{32}$$

For the terms in the bracket $[\cdot]$, we have

$$\begin{aligned}&\prod_{j=0}^{h-1} \hat{P}_{[u]}(s_{j+1}|s_j, a_j) - \prod_{j=0}^{h-1} P(s_{j+1}|s_j, a_j) \\ &= \prod_{j=0}^{h-1} \hat{P}_{[u]}(s_{j+1}|s_j, a_j) - \prod_{j=0}^{h-1} P(s_{j+1}|s_j, a_j) \\ &\quad \pm \sum_{k=1}^{h-1} \prod_{j=0}^{k-1} P(s_{j+1}|s_j, a_j) \prod_{j=k}^{h-1} \hat{P}_{[u]}(s_{j+1}|s_j, a_j) \\ &= \sum_{k=0}^{h-1} \left[\hat{P}_{[u]}(s_{k+1}|s_k, a_k) - P(s_{k+1}|s_k, a_k)\right] \prod_{j=0}^{k-1} P(s_{j+1}|s_j, a_j) \prod_{j=k}^{h-1} \hat{P}_{[u]}(s_{j+1}|s_j, a_j) \\ &\leq \sum_{k=0}^{h-1} \tilde{\epsilon}_{[u]}(s_{k+1}|s_k, a_k) \prod_{j=0}^{k-1} P(s_{j+1}|s_j, a_j) \prod_{j=k}^{h-1} \hat{P}_{[u]}(s_{j+1}|s_j, a_j), \end{aligned} \tag{33}$$

where

$$\tilde{\epsilon}_{[u]}(s_{k+1}|s_k,a_k) = O\left(\sqrt{\frac{P(s_{k+1}|s_k,a_k)\ln\frac{TSA}{\delta}}{\max\left\{N_{[u]}(s_k,a_k)\right\},1\}}} + \frac{\ln\frac{TSA}{\delta}}{\max\left\{N_{[u]}(s_k,a_k)\right\},1\}}\right) \tag{34}$$

shows how good SEEDS-UT estimates the true transition function, and the inequality is because of the empirical Bernstein inequality [49] and Lemma 8 in [7]. Applying (32) and (33) to SEEDS-UT, we have

$$\begin{aligned}
\left|\hat{q}_{[u]}^{\text{SEEDS-UT},\mathcal{P}}(s_h,a_h) - q_{[u]}^{\text{SEEDS-UT},P}(s_h,a_h)\right| &\le \sum_{k=0}^{h-1}\sum_{(s_i\in\mathcal{S}_i,a_i\in\mathcal{A})_{i=0}^{h-1}} \tilde{\epsilon}_{[u]}(s_{k+1}|s_k,a_k) \\
&\quad\cdot\left[\tilde{\pi}_{[u]}^{\text{SEEDS-UT}}(a_k|s_k)\prod_{j=0}^{k-1}\tilde{\pi}_{[u]}^{\text{SEEDS-UT}}(a_j|s_j)P(s_{j+1}|s_j,a_j)\right] \\
&\quad\cdot\left[\tilde{\pi}_{[u]}^{\text{SEEDS-UT}}(a_h|s_h)\prod_{j=k+1}^{h-1}\tilde{\pi}_{[u]}^{\text{SEEDS-UT}}(a_j|s_j)\hat{P}(s_{j+1}|s_j,a_j)\right] \\
&= \sum_{k=0}^{h-1}\sum_{s_{k+1}\in\mathcal{S}_{k+1},s_k\in\mathcal{S}_k,a_k\in\mathcal{A}} \tilde{\epsilon}_{[u]}(s_{k+1}|s_k,a_k)q_{[u]}^{\text{SEEDS-UT},P} \\
&\quad\times(s_k,a_k)\hat{q}_{[u]}^{\text{SEEDS-UT},\mathcal{P}}(s_h,a_h|s_{k+1}).
\end{aligned} \tag{35}$$

Similarly, we can show that

$$\begin{aligned}
&\left|\hat{q}_{[u]}^{\text{SEEDS-UT},\mathcal{P}}(s_h,a_h|s_{k+1}) - q_{[u]}^{\text{SEEDS-UT},P}(s_h,a_h|s_{k+1})\right| \\
&= \sum_{j=k+1}^{h-1}\sum_{s_{j+1}\in\mathcal{S}_{j+1},s_j\in\mathcal{S}_j,a_j\in\mathcal{A}} \tilde{\epsilon}_{[u]}(s_{j+1}|s_j,a_j)q_{[u]}^{\text{SEEDS-UT},P} \\
&\quad\times(s_j,a_j|s_{k+1})\hat{q}_{[u]}^{\text{SEEDS-UT},\mathcal{P}}(s_h,a_h|s_{j+1}) \\
&\le \tilde{\pi}_{[u]}^{\text{SEEDS-UT}}(a_h|s_h)\sum_{j=k+1}^{h-1}\sum_{s_{j+1}\in\mathcal{S}_{j+1},s_j\in\mathcal{S}_j,a_j\in\mathcal{A}} \tilde{\epsilon}_{[u]}(s_{j+1}|s_j,a_j)q_{[u]}^{\text{SEEDS-UT},P} \\
&\quad\times(s_j,a_j|s_{k+1}).
\end{aligned} \tag{36}$$

Combining (35) and (36), we have

$$
\begin{aligned}
&\sum_{u=1}^{\mathcal{U}} \sum_{h=0}^{H-1} \sum_{(s_h,a_h)\in\mathcal{S}_h\times\mathcal{A}} \left| \hat{q}_{[u]}^{\text{SEEDS-UT},\mathcal{P}}(s_h,a_h) - q_{[u]}^{\text{SEEDS-UT},P}(s_h,a_h) \right| \\
&\leq \sum_{u=1}^{\mathcal{U}} \sum_{h=0}^{H-1} \sum_{(s_h,a_h)\in\mathcal{S}_h\times\mathcal{A}} \sum_{k=0}^{h-1} \\
&\quad \times \sum_{(s_{k+1},s_k,a_k)\in\mathcal{S}_{k+1}\times\mathcal{S}_k\times\mathcal{A}} \tilde{\epsilon}_{[u]}(s_{k+1}|s_k,a_k) q_{[u]}^{\text{SEEDS-UT},P}(s_k,a_k) \\
&\quad \cdot q_{[u]}^{\text{SEEDS-UT},P}(s_h,a_h|s_{k+1}) \\
&+ \sum_{u=1}^{\mathcal{U}} \sum_{h=0}^{H-1} \sum_{(s_h,a_h)\in\mathcal{S}_h\times\mathcal{A}} \sum_{k=0}^{h-1} \\
&\quad \times \sum_{(s_{k+1},s_k,a_k)\in\mathcal{S}_{k+1}\times\mathcal{S}_k\times\mathcal{A}} \tilde{\epsilon}_{[u]}(s_{k+1}|s_k,a_k) q_{[u]}^{\text{SEEDS-UT},P}(s_k,a_k) \\
&\quad \cdot \left[\tilde{\pi}_{[u]}^{\text{SEEDS-UT}}(a_h|s_h) \sum_{j=k+1}^{h-1} \right. \\
&\quad \left. \times \cdot \sum_{(s_{j+1},s_j,a_j)\in\mathcal{S}_{j+1}\times\mathcal{S}_j\times\mathcal{A}} \tilde{\epsilon}_{[u]}(s_{j+1}|s_j,a_j) q_{[u]}^{\text{SEEDS-UT},P}(s_j,a_j|s_{k+1}) \right]. \qquad (37)
\end{aligned}
$$

Since $\sum_{h=0}^{H-1} \sum_{(s_h,a_h)\in\mathcal{S}_h\times\mathcal{A}} q_{[u]}^{\text{SEEDS-UT},P}(s_h,a_h|s_{k+1}) = 1$, from (37), we have

$$
\begin{aligned}
&\sum_{u=1}^{\mathcal{U}} \sum_{h=0}^{H-1} \sum_{(s_h,a_h)\in\mathcal{S}_h\times\mathcal{A}} \left| \hat{q}_{[u]}^{\text{SEEDS-UT},\mathcal{P}}(s_h,a_h) - q_{[u]}^{\text{SEEDS-UT},P}(s_h,a_h) \right| \\
&\leq \sum_{u=1}^{\mathcal{U}} \sum_{k=0}^{H-1} \sum_{(s_{k+1},s_k,a_k)\in\mathcal{S}_{k+1}\times\mathcal{S}_k\times\mathcal{A}} \tilde{\epsilon}_{[u]}(s_{k+1}|s_k,a_k) q_{[u]}^{\text{SEEDS-UT},P}(s_k,a_k) \\
&+ S \cdot \sum_{u=1}^{\mathcal{U}} \sum_{k=0}^{H-1} \sum_{j=k+1}^{H-1} \sum_{\substack{(s_{k+1},s_k,a_k)\in\mathcal{S}_{k+1}\times\mathcal{S}_k\times\mathcal{A} \\ (s_{j+1},s_j,a_j)\in\mathcal{S}_{j+1}\times\mathcal{S}_j\times\mathcal{A}}} \tilde{\epsilon}_{[u]}(s_{k+1}|s_k,a_k) q_{[u]}^{\text{SEEDS-UT},P}(s_k,a_k) \\
&\qquad \cdot \tilde{\epsilon}_{[u]}(s_{j+1}|s_j,a_j) q_{[u]}^{\text{SEEDS-UT},P}(s_j,a_j|s_{k+1}). \qquad (38)
\end{aligned}
$$

Let us focus on bounding the terms on the right-hand-side of (38) one-by-one. For the first term, we have

$$\begin{aligned}
&\sum_{u=1}^{\mathcal{U}}\sum_{k=0}^{H-1}\sum_{(s_{k+1},s_k,a_k)\in\mathcal{S}_{k+1}\times\mathcal{S}_k\times\mathcal{A}}\tilde{\epsilon}_{[u]}(s_{k+1}|s_k,a_k)q_{[u]}^{\text{SEEDS-UT},P}(s_k,a_k)\\
&=O\left(\sum_{u=1}^{\mathcal{U}}\sum_{k=0}^{H-1}\sum_{(s_{k+1},s_k,a_k)\in\mathcal{S}_{k+1}\times\mathcal{S}_k\times\mathcal{A}}q_{[u]}^{\text{SEEDS-UT},P}(s_k,a_k)\sqrt{\frac{P(s_{k+1}|s_k,a_k)\ln\frac{TSA}{\delta}}{\max\left\{N_{[u]}(s_k,a_k)\right\},1\}}}\right.\\
&\qquad\left.+\frac{q_{[u]}^{\text{SEEDS-UT},P}(s_k,a_k)\ln\frac{TSA}{\delta}}{\max\left\{N_{[u]}(s_k,a_k)\right\},1\}}\right)\\
&\le O\left(\sum_{u=1}^{\mathcal{U}}\sum_{k=0}^{H-1}\sum_{(s_k,a_k)\in\mathcal{S}_k\times\mathcal{A}}q_{[u]}^{\text{SEEDS-UT},P}(s_k,a_k)\sqrt{\frac{S_{k+1}\ln\frac{TSA}{\delta}}{\max\left\{N_{[u]}(s_k,a_k)\right\},1\}}}\right.\\
&\qquad\left.+\frac{q_{[u]}^{\text{SEEDS-UT},P}(s_k,a_k)\ln\frac{TSA}{\delta}}{\max\left\{N_{[u]}(s_k,a_k)\right\},1\}}\right),
\end{aligned}$$

where the equality is according to the definition of $\tilde{\epsilon}_{[u]}(s_{k+1}|s_k,a_k)$ in (34), and the inequality is according to Cauchy-Schwarz inequality. Note that the difficulty to further bound the above terms is that each state-action pair could be visited multiple times in a super-episode u. To this end, we construct a series to achieve an analyzable intermediate step. Let us first imagine there is a sequence of numbers based on the samples that are collected from each single episode. Then, we use $N_t(s_k,a_k)$ to denote the number of times visiting the state-action pair (s_k,a_k) before episode t. Since $N_t(s_k,a_k)$ is non-decreasing as t increases, i.e.,

$$N_{(u-1)\tau+1}(s_k,a_k)\le N_{(u-1)\tau+2}(s_k,a_k)\le\ldots\le N_{u\tau}(s_k,a_k)=N_{[u]}(s_k,a_k), \tag{39}$$

we have

$$\begin{aligned}
\frac{q_{[u]}^{\text{SEEDS-UT},P}(s_k,a_k)}{\sqrt{\max\left\{N_{[u]}(s_k,a_k)\right\},1\}}}&=\frac{q_{[u]}^{\text{SEEDS-UT},P}(s_k,a_k)}{\sqrt{\max\left\{N_{u\tau}(s_k,a_k)\right\},1\}}}\\
&\le\ldots\le\frac{q_{[u]}^{\text{SEEDS-UT},P}(s_k,a_k)}{\sqrt{\max\left\{N_{(u-1)\tau+1}(s_k,a_k)\right\},1\}}}.
\end{aligned}$$

Now, let us compare our regret bound before to a intermediate step that is based on this series, i.e.,

$$\sum_{u=1}^{\mathcal{U}}\sum_{k=0}^{H-1}\sum_{(s_{k+1},s_k,a_k)\in\mathcal{S}_{k+1}\times\mathcal{S}_k\times\mathcal{A}}\tilde{\epsilon}_{[u]}(s_{k+1}|s_k,a_k)q_{[u]}^{\text{SEEDS-UT},P}(s_k,a_k)$$

$$\leq O\left(\frac{1}{\tau}\sum_{u=1}^{\mathcal{U}}\sum_{k=0}^{H-1}\sum_{(s_k,a_k)\in\mathcal{S}_k\times\mathcal{A}}\sum_{t=(u-1)\tau+1}^{u\tau}q_{[u]}^{\text{SEEDS-UT},P}(s_k,a_k)\sqrt{\frac{S_{k+1}\ln\frac{TSA}{\delta}}{\max\{N_t(s_k,a_k)\},1\}}}\right.$$

$$\left.+\frac{q_{[u]}^{\text{SEEDS-UT},P}(s_k,a_k)\ln\frac{TSA}{\delta}}{\max\{N_t(s_k,a_k)\},1\}}\right)$$

$$\leq O\left(\frac{1}{\tau}\sum_{k=0}^{H-1}\sqrt{S_kS_{k+1}AT\ln\frac{TSA}{\delta}}\right)$$

$$\leq O\left(\frac{1}{\tau}HS\sqrt{AT\ln\frac{TSA}{\delta}}\right), \tag{40}$$

Let us now consider the second term on the right-hand-side of (38), which can be upper-bounded similarly to the steps above to bound the first term. First, according to the definition of $\tilde{\epsilon}_{[u]}(s_{k+1}|s_k,a_k)$ in (34), we have this second term is upper-bounded by

$$S\cdot O\left(\sum_{u=1}^{\mathcal{U}}\sum_{k=0}^{H-1}\sum_{j=k+1}^{H-1}\sum_{\substack{(s_{k+1},s_k,a_k)\in\mathcal{S}_{k+1}\times\mathcal{S}_k\times\mathcal{A}\\(s_{j+1},s_j,a_j)\in\mathcal{S}_{j+1}\times\mathcal{S}_j\times\mathcal{A}}}\right.$$

$$\times\sqrt{\frac{P(s_{k+1}|s_k,a_k)\ln\frac{TSA}{\delta}}{\max\left\{N_{[u]}(s_k,a_k)\right\},1\}}}\,q_{[u]}^{\text{SEEDS-UT},P}(s_k,a_k)$$

$$\left.\cdot\sqrt{\frac{P(s_{j+1}|s_j,a_j)\ln\frac{TSA}{\delta}}{\max\left\{N_{[u]}(s_j,a_j)\right\},1\}}}q_{[u]}^{\text{SEEDS-UT},P}(s_j,a_j|s_{k+1})+\ln\frac{TSA}{\delta}\right.$$

$$\cdot\sum_{u=1}^{\mathcal{U}}\sum_{k=0}^{H-1}\sum_{j=k+1}^{H-1}\sum_{\substack{(s_{k+1},s_k,a_k)\in\mathcal{S}_{k+1}\times\mathcal{S}_k\times\mathcal{A}\\(s_{j+1},s_j,a_j)\in\mathcal{S}_{j+1}\times\mathcal{S}_j\times\mathcal{A}}}\frac{q_{[u]}^{\text{SEEDS-UT},P}(s_k,a_k)}{\max\left\{N_{[u]}(s_k,a_k)\right\},1\}}$$

$$\left.+\frac{q_{[u]}^{\text{SEEDS-UT},P}(s_j,a_j)}{\max\left\{N_{[u]}(s_j,a_j)\right\},1\}}\right).$$

Next, according to Cauchy-Schwarz inequality, we have the terms inside the big-O notation can be upper-bounded by

$$\ln \frac{TSA}{\delta}$$

$$\cdot \left[\sum_{k=0}^{H-1} \sum_{j=k+1}^{H-1} \right.$$

$$\cdot \sqrt{\sum_{u=1}^{\mathcal{U}} \sum_{\substack{(s_{k+1},s_k,a_k),\\(s_{j+1},s_j,a_j)}} \frac{q_{[u]}^{\text{SEEDS-UT},P}(s_k,a_k) P(s_{k+1}|s_k,a_k) q_{[u]}^{\text{SEEDS-UT},P}(s_j,a_j|s_{k+1})}{\max\left\{N_{[u]}(s_k,a_k)\right\},1\right\}}}$$

$$\cdot \sqrt{\sum_{u=1}^{\mathcal{U}} \sum_{\substack{(s_{k+1},s_k,a_k),\\(s_{j+1},s_j,a_j)}} \frac{q_{[u]}^{\text{SEEDS-UT},P}(s_k,a_k) P(s_{j+1}|s_j,a_j) q_{[u]}^{\text{SEEDS-UT},P}(s_j,a_j|s_{k+1})}{\max\left\{N_{[u]}(s_j,a_j)\right\},1\right\}}}$$

$$+ \sum_{u=1}^{\mathcal{U}} \sum_{k=0}^{H-1} \sum_{j=k+1}^{H-1} \sum_{\substack{(s_{k+1},s_k,a_k),\\(s_{j+1},s_j,a_j)}} \left(\frac{q_{[u]}^{\text{SEEDS-UT},P}(s_k,a_k)}{\max\left\{N_{[u]}(s_k,a_k)\right\},1\right\}} \right.$$

$$\left. \left. + \frac{q_{[u]}^{\text{SEEDS-UT},P}(s_j,a_j)}{\max\left\{N_{[u]}(s_j,a_j)\right\},1\right\}} \right) \right].$$

Then, according to (39), we have that the terms under the $\sqrt{\cdot}$ operator can be upper-bounded by

$$\frac{1}{\tau} \sum_{u=1}^{\mathcal{U}} \sum_{\substack{(s_{k+1},s_k,a_k),\\(s_{j+1},s_j,a_j)}} \sum_{t=(u-1)\tau+1}^{u\tau}$$

$$\times \frac{q_{[u]}^{\text{SEEDS-UT},P}(s_k,a_k) P(s_{k+1}|s_k,a_k) q_{[u]}^{\text{SEEDS-UT},P}(s_j,a_j|s_{k+1})}{\max\{N_t(s_k,a_k)\},1\}}$$

$$\cdot \frac{1}{\tau} \sum_{u=1}^{\mathcal{U}} \sum_{\substack{(s_{k+1},s_k,a_k),\\(s_{j+1},s_j,a_j)}} \sum_{t=(u-1)\tau+1}^{u\tau}$$

$$\times \frac{q_{[u]}^{\text{SEEDS-UT},P}(s_k,a_k) P(s_{j+1}|s_j,a_j) q_{[u]}^{\text{SEEDS-UT},P}(s_j,a_j|s_{k+1})}{\max\left\{N_t(s_j,a_j)\right\},1\right\}},$$

and the second term in the bracket $[\cdot]$ can be upper-bounded by

$$\sum_{u=1}^{\mathcal{U}} \frac{1}{\tau} \sum_{k=0}^{H-1} \sum_{j=k+1}^{H-1} \sum_{\substack{(s_{k+1},s_k,a_k),\\(s_{j+1},s_j,a_j)}} \sum_{t=(u-1)\tau+1}^{u\tau} \times \left(\frac{q_{[u]}^{\text{SEEDS-UT},P}(s_k,a_k)}{\max\{N_t(s_k,a_k)\},1\}} + \frac{q_{[u]}^{\text{SEEDS-UT},P}(s_j,a_j)}{\max\left\{N_t(s_j,a_j)\right\},1\}} \right).$$

Combining the above steps and according to Lemma 10 in [7], we have that the second term on the right-hand-side of (38) can be upper-bounded by $O\left(\frac{1}{\tau} H^2 S^2 A \ln \frac{TSA}{\delta}\right)$.

Therefore, with probability $1-\delta$, the first term on the right-hand-side of (31) can be upper-bounded by

$$O\left(HS\sqrt{AT \ln \frac{TSA}{\delta}} + H^2 S^2 \ln \frac{TSA}{\delta}\right). \tag{41}$$

Step-2-ii (Bounding the Second Term) The second term on the right-hand-side of (31) can be further decomposed into two terms as follows,

$$\begin{aligned}
&\sum_{u=1}^{\mathcal{U}} \mathbb{E}_{\mathcal{F}_{[u]}} \left[\mathbb{E}\left[\left\langle \hat{q}_{[u]}^{\text{SEEDS-UT},\mathcal{P}}, l_{[u]} - \hat{l}_{[u]}^{\text{SEEDS-UT}} \right\rangle \middle| \mathcal{F}_{[u]}, P \right] \right] \\
&= \sum_{u=1}^{\mathcal{U}} \mathbb{E}_{\mathcal{F}_{[u]}} \left[\mathbb{E}\left[\left\langle \hat{q}_{[u]}^{\text{SEEDS-UT},\mathcal{P}}, l_{[u]} - \mathbb{E}\left[\hat{l}_{[u]}^{\text{SEEDS-UT}} \right] \right\rangle \middle| \mathcal{F}_{[u]}, P \right] \right] \\
&\quad + \sum_{u=1}^{\mathcal{U}} \mathbb{E}_{\mathcal{F}_{[u]}} \left[\mathbb{E}\left[\left\langle \hat{q}_{[u]}^{\text{SEEDS-UT},\mathcal{P}}, \mathbb{E}\left[\hat{l}_{[u]}^{\text{SEEDS-UT}} \right] - \hat{l}_{[u]}^{\text{SEEDS-UT}} \right\rangle \middle| \mathcal{F}_{[u]}, P \right] \right].
\end{aligned} \tag{42}$$

Let us consider the two terms on the right-hand-side. First, according to Lemma 3, we have

$$\begin{aligned}
&\sum_{u=1}^{\mathcal{U}} \mathbb{E}_{\mathcal{F}_{[u]}} \left[\mathbb{E}\left[\left\langle \hat{q}_{[u]}^{\text{SEEDS-UT},\mathcal{P}}, l_{[u]} - \mathbb{E}\left[\hat{l}_{[u]}^{\text{SEEDS-UT}} \right] \right\rangle \middle| \mathcal{F}_{[u]}, P \right] \right] \\
&= \sum_{u=1}^{\mathcal{U}} \mathbb{E}_{\mathcal{F}_{[u]}} \left[\mathbb{E}\left[\sum_{s\in\mathcal{S}, a\in\mathcal{A}} \hat{q}_{[u]}^{\text{SEEDS-UT},\mathcal{P}}(s,a) l_{[u]}(s,a) \right. \right. \\
&\quad \left. \left. \times \left(1 - \frac{q_{[u]}^{\text{SEEDS-UT},P}(s,a)}{Q_{[u]}^{\gamma}(s,a)} \right) \middle| \mathcal{F}_{[u]}, P \right] \right].
\end{aligned}$$

Since $l_{[u]}(s,a) \le \tau$ and $Q^{\gamma}_{[u]}(s,a) \ge \hat{q}^{\text{SEEDS-UT},\mathcal{P}}_{[u]}(s,a)$, we have

$$\begin{aligned}
&\sum_{u=1}^{\mathcal{U}} \mathbb{E}_{\mathcal{F}_{[u]}}\left[\mathbb{E}\left[\left\langle \hat{q}^{\text{SEEDS-UT},\mathcal{P}}_{[u]}, l_{[u]} - \mathbb{E}\left[\hat{l}^{\text{SEEDS-UT}}_{[u]}\right]\right\rangle \middle| \mathcal{F}_{[u]}, P\right]\right] \\
&\le \tau \sum_{u=1}^{\mathcal{U}} \mathbb{E}_{\mathcal{F}_{[u]}}\left[\mathbb{E}\left[\sum_{s\in\mathcal{S},a\in\mathcal{A}} \left| Q^{\gamma}_{[u]}(s,a) - q^{\text{SEEDS-UT},P}_{[u]}(s,a)\right| \middle| \mathcal{F}_{[u]}, P\right]\right] \\
&\le \tau \sum_{u=1}^{\mathcal{U}} \mathbb{E}_{\mathcal{F}_{[u]}}\left[\mathbb{E}\left[\sum_{s\in\mathcal{S},a\in\mathcal{A}} \left| \max_{\hat{P}\in\mathcal{P}_{[u]}} q^{\hat{P}}_{[u]}(s,a) + \gamma - q^{\text{SEEDS-UT},P}_{[u]}(s,a)\right| \middle| \mathcal{F}_{[u]}, P\right]\right],
\end{aligned}$$

where the term $\max_{\hat{P}\in\mathcal{P}_{[u]}} q^{\hat{P}}_{[u]}(s,a) - q^{\text{SEEDS-UT},P}_{[u]}(s,a)$ on the right-hand-side represents how well SEEDS-UT estimates the true occupancy measure using the transition-function set, and the term γ on the right-hand-side verifies that this part of the gap is controlled by the parameter γ. Then, according to the bound for the first term on the right-hand-side of (31), we have

$$\begin{aligned}
&\sum_{u=1}^{\mathcal{U}} \mathbb{E}_{\mathcal{F}_{[u]}}\left[\mathbb{E}\left[\left\langle \hat{q}^{\text{SEEDS-UT},\mathcal{P}}_{[u]}, l_{[u]} - \mathbb{E}\left[\hat{l}^{\text{SEEDS-UT}}_{[u]}\right]\right\rangle \middle| \mathcal{F}_{[u]}, P\right]\right] \\
&\le O\left(HS\sqrt{AT\ln\frac{TSA}{\delta}}\right) + \gamma TSA.
\end{aligned}$$

Second, according to Azuma's inequality, we have with probability $1-\delta$,

$$\begin{aligned}
&\sum_{u=1}^{\mathcal{U}} \mathbb{E}_{\mathcal{F}_{[u]}}\left[\mathbb{E}\left[\left\langle \hat{q}^{\text{SEEDS-UT},\mathcal{P}}_{[u]}, \mathbb{E}\left[\hat{l}^{\text{SEEDS-UT}}_{[u]}\right] - \hat{l}^{\text{SEEDS-UT}}_{[u]}\right\rangle \middle| \mathcal{F}_{[u]}, P\right]\right] \\
&\le O\left(\tau H\sqrt{\frac{T}{\tau}\ln\frac{1}{\delta}}\right) \le O\left(H\sqrt{T\tau\ln\frac{1}{\delta}}\right). \qquad (43)
\end{aligned}$$

Therefore, with probability $1-\delta$, the second term on the right-hand-side of (31) can be upper-bounded by

$$O\left(HS\sqrt{AT\ln\frac{TSA}{\delta}} + \gamma TSA + H\sqrt{T\tau\ln\frac{1}{\delta}}\right). \qquad (44)$$

Step-2-iii (Bounding the Third Term) Follow our proof for the case when the transition function is known, it is not hard to show that

$$\sum_{u=1}^{\mathcal{U}} \left\langle \hat{q}_{[u]}^{\text{SEEDS-UT},\mathcal{P}} - q^{\pi^*}, \hat{l}_{[u]}^{\text{SEEDS-UT}} \right\rangle$$

$$\leq \eta \sum_{u=1}^{\mathcal{U}} \sum_{s\in\mathcal{S},a\in\mathcal{A}} \hat{q}_{[u]}^{\text{SEEDS-UT},\mathcal{P}}(s,a) \left(\hat{l}_{[u]}^{\text{SEEDS-UT}}(s,a)\right)^2 + \frac{H\ln(SA)}{\eta}.$$

Let us focus on the first term on the right-hand-side. Note that different from that in [7], the loss $\hat{l}_{[u]}^{\text{SEEDS-UT}}(s,a)$ above is calculated based on the samples from a whole super-episode. Thus, each state-action pair could be visited multiple times. To this end, we provide a super-episodic version of loss concentration as follows,

$$\sum_{u=1}^{\mathcal{U}} \sum_{s\in\mathcal{S},a\in\mathcal{A}} \hat{q}_{[u]}^{\text{SEEDS-UT},\mathcal{P}}(s,a) \left(\hat{l}_{[u]}^{\text{SEEDS-UT}}(s,a)\right)^2 \leq \frac{\tau H}{2\gamma} \ln\frac{H}{\delta}$$

$$+ \sum_{u=1}^{\mathcal{U}} \sum_{s\in\mathcal{S},a\in\mathcal{A}} \frac{\tau q_{[u]}^{\text{SEEDS-UT}}}{\max\limits_{\hat{P}\in\mathcal{P}_{[u]}} q_{[u]}^{\hat{P}}(s,a)} l_{[u]}(s,a).$$

In the following, we show how to get this. First, since

$$\hat{l}_{[u]}^{\text{SEEDS-UT}}(s,a) = \sum_{j=1}^{J_{[u]}} \frac{l_{t_j(s,a)}(s,a)}{Q_{[u]}^{\gamma}(s,a)} \mathbf{1}_{\{(s,a):t_1(s,a),\ldots,t_{J_{[u]}}(s,a)\}} \leq \frac{\tau}{Q_{[u]}^{\gamma}(s,a)},$$

we have

$$\hat{q}_{[u]}^{\text{SEEDS-UT},\mathcal{P}}(s,a) \left(\hat{l}_{[u]}^{\text{SEEDS-UT}}(s,a)\right)^2 \leq \frac{\tau \hat{q}_{[u]}^{\text{SEEDS-UT},\mathcal{P}}(s,a)}{Q_{[u]}^{\gamma}(s,a)} \hat{l}_{[u]}^{\text{SEEDS-UT}}(s,a)$$

$$\leq \tau \hat{l}_{[u]}^{\text{SEEDS-UT}}(s,a)$$

$$= \tau \sum_{j=1}^{J_{[u]}} \frac{l_{t_j(s,a)}(s,a)}{Q_{[u]}^{\gamma}(s,a)} \mathbf{1}_{\{(s,a):t_1(s,a),\ldots,t_{J_{[u]}}(s,a)\}}$$

$$= \tau \sum_{t=(u-1)\tau+1}^{u\tau} \frac{l_t(s,a)}{Q_{[u]}^{\gamma}(s,a)} \mathbf{1}_{\{(s,a) \text{ was visited in episode } t \text{ of super-episode } u\}}.$$

Let us define

$$\tilde{l}_t(s,a) \triangleq \frac{l_t(s,a)\mathbf{1}_{\{(s,a) \text{ was visited in episode } t \text{ of super-episode } u\}}}{Q_{[u]}^{\gamma}(s,a)}.$$

Then, we have

$$\sum_{t=1}^{T}\sum_{s\in\mathcal{S},a\in\mathcal{A}} 2\gamma\left(\tilde{l}_t(s,a)-\frac{q_{[u]}^{\text{SEEDS-UT}}}{\max\limits_{\hat{P}\in\mathcal{P}_{[u]}} q_{[u]}^{\hat{P}}(s,a)}l_t(s,a)\right)\leq H\ln\frac{H}{\delta}.$$

By combining all episodes in the same super-episode u together, we have

$$\sum_{u=1}^{\mathcal{U}}\sum_{s\in\mathcal{S},a\in\mathcal{A}} 2\gamma\left(\sum_{t=(u-1)\tau+1}^{u\tau}\tilde{l}_t(s,a)-\frac{q_{[u]}^{\text{SEEDS-UT}}}{\max\limits_{\hat{P}\in\mathcal{P}_{[u]}} q_{[u]}^{\hat{P}}(s,a)}l_{[u]}(s,a)\right)\leq H\ln\frac{H}{\delta}.$$

By rearranging the terms, we have

$$\sum_{u=1}^{\mathcal{U}}\sum_{s\in\mathcal{S},a\in\mathcal{A}}\tilde{l}_{[u]}(s,a)\leq\frac{H}{2\gamma}\ln\frac{H}{\delta}+\sum_{u=1}^{\mathcal{U}}\sum_{s\in\mathcal{S},a\in\mathcal{A}}\frac{q_{[u]}^{\text{SEEDS-UT}}}{\max\limits_{\hat{P}\in\mathcal{P}_{[u]}} q_{[u]}^{\hat{P}}(s,a)}l_{[u]}(s,a)$$

$$\leq\frac{H}{2\gamma}\ln\frac{H}{\delta}+\sum_{u=1}^{\mathcal{U}}\sum_{s\in\mathcal{S},a\in\mathcal{A}}l_{[u]}(s,a)\leq\frac{H}{2\gamma}\ln\frac{H}{\delta}+\frac{T}{\tau}SA\tau=\frac{H}{2\gamma}\ln\frac{H}{\delta}+TSA.$$

Thus, we have

$$\sum_{u=1}^{\mathcal{U}}\sum_{s\in\mathcal{S},a\in\mathcal{A}}\hat{q}_{[u]}^{\text{SEEDS-UT},\mathcal{P}}(s,a)\left(\hat{l}_{[u]}^{\text{SEEDS-UT}}(s,a)\right)^2\leq\tau\cdot\frac{H}{2\gamma}\ln\frac{H}{\delta}+\tau TSA.$$

Therefore, with probability $1-\delta$, the third term on the right-hand-side of (31) can be upper-bounded by

$$O\left(\frac{\eta\tau H}{\gamma}\ln\frac{H}{\delta}+\eta\tau TSA+\frac{H\ln(SA)}{\eta}\right). \tag{45}$$

Step-2-iv (Bounding the Fourth Term) First, it is not hard to get that with probability $1-\delta$,

$$\sum_{u=1}^{\mathcal{U}}\hat{l}_{[u]}^{\text{SEEDS-UT}}(s,a)\leq\frac{1}{2\gamma}\ln\frac{H}{\delta}+\sum_{u=1}^{\mathcal{U}}\frac{q_{[u]}^{\text{SEEDS-UT},P}(s,a)}{\max\limits_{\hat{P}\in\mathcal{P}_{[u]}} q_{[u]}^{\hat{P}}(s,a)}l_{[u]}(s,a). \tag{46}$$

Thus, we have

$$\sum_{u=1}^{\mathcal{U}} \left\langle q^{\pi^*}, \hat{l}_{[u]} - l_{[u]} \right\rangle = \sum_{u=1}^{\mathcal{U}} \sum_{s\in\mathcal{S}, a\in\mathcal{A}} q^{\pi^*}(s,a)\hat{l}_{[u]}(s,a)$$

$$- \sum_{u=1}^{\mathcal{U}} \sum_{s\in\mathcal{S}, a\in\mathcal{A}} q^{\pi^*}(s,a) l_{[u]}(s,a)$$

$$\leq \sum_{s\in\mathcal{S}, a\in\mathcal{A}} q^{\pi^*}(s,a) \frac{1}{2\gamma} \ln \frac{H}{\delta} + \sum_{s\in\mathcal{S}, a\in\mathcal{A}} q^{\pi^*}(s,a) \cdot \sum_{u=1}^{\mathcal{U}} \frac{q_{[u]}^{\text{SEEDS-UT},P}(s,a)}{\max\limits_{\hat{P}\in\mathcal{P}_{[u]}} q_{[u]}^{\hat{P}}(s,a)} l_{[u]}(s,a)$$

$$- \sum_{u=1}^{\mathcal{U}} \sum_{s\in\mathcal{S}, a\in\mathcal{A}} q^{\pi^*}(s,a) l_{[u]}(s,a).$$

$$\leq \frac{H}{2\gamma} \ln \frac{H}{\delta} + \sum_{u=1}^{\mathcal{U}} \sum_{s\in\mathcal{S}, a\in\mathcal{A}} q^{\pi^*}(s,a) l_{[u]}(s,a) \left(\frac{q_{[u]}^{\text{SEEDS-UT},P}(s,a)}{\max\limits_{\hat{P}\in\mathcal{P}_{[u]}} q_{[u]}^{\hat{P}}(s,a)} - 1 \right)$$

$$\leq \frac{H}{2\gamma} \ln \frac{H}{\delta}. \tag{47}$$

Step-3 (Final step): Finally, by combining (41), (44), (45), (47) and the switching-cost upper-bound $\beta \cdot \left\lceil \frac{T}{\tau} \right\rceil$, and tuning the parameters η, τ and γ as in Algorithm 3, we have that the regret of SEEDS-UT is upper-bounded by $O\left(\beta^{1/3} H^{2/3} (SA)^{1/3} T^{2/3} \left(\ln \frac{TSA}{\delta}\right)^{1/2}\right)$ with probability $1 - \delta$.

□

References

1. Azar MG, Osband I, Munos R (2017) Minimax regret bounds for reinforcement learning. In: International conference on machine learning, PMLR, pp 263–272
2. Jin C, Allen-Zhu Z, Bubeck S, Jordan MI (2018) Is q-learning provably efficient? In: Advances in neural information processing systems, vol 31
3. Sutton RS, Barto AG (2018) Reinforcement learning: an introduction. MIT Press, Cambridge
4. Agarwal A, Jiang N, Kakade SM, Sun W (2019) Reinforcement learning: Theory and algorithms. CS Department, UW Seattle, Seattle, WA, USA, Technical Report, 32
5. Jin C, Yang Z, Wang Z, Jordan MI (2020) Provably efficient reinforcement learning with linear function approximation. In: Conference on learning theory, PMLR, pp 2137–2143
6. Zimin A, Neu G (2013) Online learning in episodic markovian decision processes by relative entropy policy search. Adv Neural Inf Process Syst 26
7. Jin C, Jin T, Luo H, Sra S, Yu T (2020) Learning adversarial Markov decision processes with bandit feedback and unknown transition. In: International Conference on Machine Learning, PMLR, pp 4860–4869

8. Lee C-W, Luo H, Wei C-Y, Zhang M (2020) Bias no more: high-probability data-dependent regret bounds for adversarial bandits and mdps. Adv Neural Inf Process Syst 33:15522–15533
9. Rosenberg A, Mansour Y (2019) Online convex optimization in adversarial markov decision processes. In: International conference on machine learning, PMLR, pp 5478–5486
10. Cai Q, Yang Z, Jin C, Wang Z (2020) Provably efficient exploration in policy optimization. In: International conference on machine learning, PMLR, pp 1283–1294
11. Luo H, Wei C-Y, Lee C-W (2021) Policy optimization in adversarial mdps: improved exploration via dilated bonuses. In: Adv Neural Inf Process Syst 34:22931–22942
12. Yu JY, Mannor S (2009) Arbitrarily modulated markov decision processes. In: Proceedings of the 48h IEEE conference on decision and control (CDC) held jointly with 2009 28th Chinese control conference, IEEE, pp 2946–2953
13. Cheung WC, Simchi-Levi D, Zhu R (2019) Reinforcement learning under drift. Preprint, Available via arXiv:1906.02922
14. Lykouris T, Simchowitz M, Slivkins A, Sun W (2021) Corruption-robust exploration in episodic reinforcement learning. In: Conference on learning theory, PMLR, pp 3242–3245
15. Rosenberg A, Mansour Y (2019) Online stochastic shortest path with bandit feedback and unknown transition function. Adv Neural Inf Process Syst 32
16. Lee C-W, Luo H, Wei C-Y, Zhang M, Zhang X (2021) Achieving near instance-optimality and minimax-optimality in stochastic and adversarial linear bandits simultaneously. In: International conference on machine learning, PMLR, pp 6142–6151
17. Zhao H, Zhou D, Gu Q (2021) Linear contextual bandits with adversarial corruptions. Preprint. Available via arXiv:2110.12615
18. Jin T, Huang L, Luo H (2021) The best of both worlds: stochastic and adversarial episodic mdps with unknown transition. Adv Neural Inf Process Syst 34:20491–20502
19. He J, Zhou D, Zhang T, Gu Q (2022) Nearly optimal algorithms for linear contextual bandits with adversarial corruptions. Preprint. Available via arXiv:2205.06811
20. Theocharous G, Thomas PS, Ghavamzadeh M (2015) Personalized ad recommendation systems for life-time value optimization with guarantees. In: Twenty-fourth international joint conference on artificial intelligence
21. Yu C, Liu J, Nemati S, Yin G (2021) Reinforcement learning in healthcare: a survey. In: ACM Comput Surv (CSUR) 55(1):1–36
22. Kober J, Bagnell JA, Peters J (2013) Reinforcement learning in robotics: a survey. Int J Robot Res 32(11):1238–1274
23. Bennane A (2013) Adaptive educational software by applying reinforcement learning. Inf Educ 12(1):13–28
24. Xu Z, Tang J, Meng J, Zhang W, Wang Y, Liu CH, Yang D (2018) Experience-driven networking: a deep reinforcement learning based approach. In: IEEE INFOCOM 2018-IEEE conference on computer communications. IEEE, pp 1871–1879
25. Krishnan S, Yang Z, Goldberg K, Hellerstein J, Stoica I (2018) Learning to optimize join queries with deep reinforcement learning. Preprint. Available via arXiv:1808.03196
26. Lin M, Wierman A, Roytman A, Meyerson A, Andrew LLH (2012) Online optimization with switching cost. In: ACM SIGMETRICS Perform Eval Rev 40(3):98–100
27. Chen N, Comden J, Liu Z, Gandhi A, Wierman A (2016) Using predictions in online optimization: looking forward with an eye on the past. In: ACM SIGMETRICS Perform Eval Rev 44(1):193–206
28. Goel G, Lin Y, Sun H, Wierman A (2019) Beyond online balanced descent: an optimal algorithm for smoothed online optimization. Adv Neural Inf Process Syst 32
29. Shi M, Lin X, Fahmy S (2021) Competitive online convex optimization with switching costs and ramp constraints. IEEE/ACM Trans Netw 29(2):876–889
30. Shi M, Lin X, Jiao L (2021) Combining regularization with look-ahead for competitive online convex optimization. In: IEEE INFOCOM 2021-IEEE conference on computer communications, IEEE, pp 1–10
31. Friedman J, Linial N (1993) On convex body chasing. Discrete Comput Geom 9(3):293–321

32. Sellke M (2020) Chasing convex bodies optimally. In: Proceedings of the fourteenth annual ACM-SIAM symposium on discrete algorithms. SIAM, pp 1509–1518
33. Bubeck S, Rabani Y, Sellke M (2021) Online multiserver convex chasing and optimization. In: Proceedings of the 2021 ACM-SIAM symposium on discrete algorithms (SODA). SIAM, pp 2093–2104
34. Borodin A, El-Yaniv R (2005) Online computation and competitive analysis. Cambridge University Press, Cambridge
35. Buchbinder N, Chen S, Naor J (2014) Competitive analysis via regularization. In: Proceedings of the twenty-fifth annual ACM-SIAM symposium on discrete algorithms. SIAM, pp 436–444
36. Lin Y, Goel G, Wierman A (2020) Online optimization with predictions and non-convex losses. In: Proc ACM Meas Anal Comput Syst 4(1):1–32
37. Goel G, Wierman A (2019) An online algorithm for smoothed regression and lqr control. In: The 22nd international conference on artificial intelligence and statistics. PMLR, pp 2504–2513
38. Li Y, Qu G, Li N (2020) Online optimization with predictions and switching costs: fast algorithms and the fundamental limit. In: IEEE Trans Autom Control 66(10):4761–4768
39. Lin Y, Hu Y, Shi G, Sun H, Qu G, Wierman A (2021) Perturbation-based regret analysis of predictive control in linear time varying systems. Adv Neural Inf Process Syst. 34:5174–5185
40. Geulen S, Vöcking B, Winkler M (2010) Regret minimization for online buffering problems using the weighted majority algorithm. In: Conference on learning theory. Citeseer, pp 132–143
41. Dekel O, Ding J, Koren T, Peres Y (2014) Bandits with switching costs: T 2/3 regret. In: Proceedings of the forty-sixth annual ACM symposium on theory of computing, pp 459–467
42. Arora R, Marinov TV, Mohri M (2019) Bandits with feedback graphs and switching costs. Adv Neural Inf Process Syst 32
43. Shi M, Lin X, Jiao L (2022) Power-of-2-arms for bandit learning with switching costs. In: Proceedings of the twenty-third international symposium on theory, algorithmic foundations, and protocol design for mobile networks and mobile computing, pp 131–140
44. Shi M, Liang Y, Shroff N (2023) Near-optimal adversarial reinforcement learning with switching costs. In: International conference on learning representations
45. Bai Y, Xie T, Jiang N, Wang Y-X (2019) Provably efficient q-learning with low switching cost. Adv Neural Inf Process Syst 32
46. Qiao D, Yin M, Min M, Wang Y-X (2022) Sample-efficient reinforcement learning with $\log\log(T)$ switching cost. Preprint. Available via arXiv:2202.06385
47. Rakhlin A, Abernethy J, Agarwal A, Bartlett P, Hazan E, Tewari A (2009) Lecture notes on online learning draft. Citeseer
48. Neu G (2015) Explore no more: improved high-probability regret bounds for non-stochastic bandits. Adv Neural Inf Process Syst 28
49. Maurer A, Pontil M (2009) Empirical bernstein bounds and sample variance penalization. In: Proceedings of the 22nd annual conference on learning theory

Part IV
Security in Network-Enabled Applications

Security and Privacy of Augmented Reality Systems

Jiacheng Shang

1 Introduction

Augmented Reality (AR) is a system that overlaps virtual (computer-generated) content over real-world scenes. By leveraging multiple types of sensors and algorithms, AR systems understand the activities of the AR users and the surrounding environment. With the knowledge, AR systems can enhance the perception of AR users by adding virtual content and playing audios. The virtual content can be either constructive or destructive. Constructive contents refer to virtual objects that are additive to the natural environment, and destructive contents are objects that mask the partial or whole part of the natural environment perceived by the AR user. Therefore, AR systems can provide a more immersive user experience than traditional systems (e.g., smartphones and personal computers) and even virtual reality, where all contents are virtual.

Due to these great features, AR devices and applications are becoming increasingly popular among mass consumers, in industry, and even in military training. Based on the statistical data from Exploding Topics, the AR market is valued at over $31 billion in 2023, and its revenue is expected to exceed $50 billion by 2027 [27]. In 2023, there are around 1.4 billion active AR user devices, and AR-based shopping encourages almost half of all consumers to spend more. Considering the increasing revenue of the AR market, many companies have either launched their AR applications or produced special AR devices for users. For example, Amazon rolled out an AR application for users to view items in their own space [57]. Microsoft has released two AR headsets, HoloLens one and HoloLens two, for ordinary and business users. Meta also launched their device called Quest Pro, which

J. Shang (✉)
School of Computing, Montclair State University, Montclair, NJ, USA
e-mail: shangj@montclair.edu

Y. Chen et al. (eds.), *Network Security Empowered by Artificial Intelligence*,
Advances in Information Security 107, https://doi.org/10.1007/978-3-031-53510-9_11

can be used as both Virtual Reality (VR) and AR devices. AR systems have also been used in military training. For example, U.S. Army used Microsoft HoloLens to build an AR system that can display a map and have thermal imaging to reveal people in the dark [32, 55].

While most users and companies focus on delivering more functional features to AR users, less attention is paid to the security and privacy of such systems. AR systems rely on multiple types of sensors to sense the surrounding environment and then use different techniques and algorithms to understand the natural world and make decisions. However, recent research shows that AR systems suffer from various types of attacks that can either manipulate the AR systems' decisions or disclose the AR users' private information [72, 87, 89]. For example, Zhang et al. showed that it is feasible to spoof the depth sensor, which is widely used on AR devices to understand the natural world [89]. Considering AR systems can broadly impact the user's perception, such attacks can be more severe. For example, the perception of soldiers can be greatly impacted or even controlled if enemies can manipulate specific sensor signals (infrared light and magnetic signals) received by the AR device.

Considering the vulnerability of existing AR systems and this understudied research field, it is essential to have a comprehensive study focusing on the security and privacy of AR systems. This chapter will first discuss the architecture of AR systems and important sensors in existing commercial AR systems. Then, security and privacy issues of AR systems will be discussed from three aspects, including input security, input privacy, and output security and privacy. Finally, we will discuss future research directions for protecting the security and privacy of AR systems.

2 Augmented Reality System Overview

This section will focus on general architecture and workflows of existing AR system. Also, this section will discuss necessary hardware components in existing commercial AR systems.

2.1 Architecture of AR Systems

As shown in Fig. 1, generally, and AR system consists of four essential components: the AR user, the AR device, the real world, and virtual objects. While using the AR device/application, the AR device keeps sensing the AR user or getting input from the AR user. For example, an AR device can sense the movement of the AR user and get voice commands through a microphone. Also, the AR device leverages different types of sensors to understand the physical world that is around the AR user. For example, the depth sensor on AR devices can understand the size, shape, and depth of real objects in a scene, which enables a more immersive and realistic

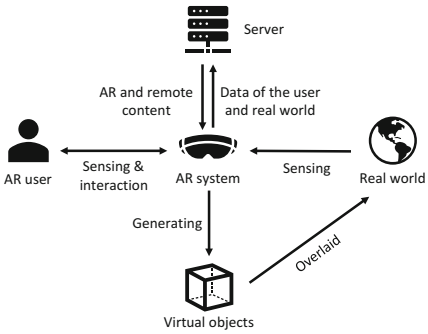


Fig. 1 Architecture of AR systems

user experience. The information of the AR user and the physical world will then be processed either locally on the AR device or offloaded to a remote server for processing based on the complexity of the processing job. After the information processing, the AR device will have a clear understanding of the AR user and the real world, so it will respond to the AR user and overlay corresponding virtual objects over the real-world scene.

2.2 Sensors and Important Components on AR Devices

As discussed earlier in this chapter, the immersive experience provided by AR systems relies on the accurate sensing and intelligent processing of sensor signals. This subsection will summarize important hardware components in current AR devices.

2.2.1 Depth Sensor

In order to deliver a realistic AR experience, AR devices need to understand the physical world so they can know where to overlay virtual objects. Therefore, depth sensors are widely implemented in current AR devices. For example, Microsoft HoloLens 2 leverage the depth sensor to recognize the gestures of the user. A depth sensor is used to measure the distance from the AR device to a point in the 3-dimensional space. Existing depth sensors are commonly based on three techniques: stereo vision, time of fight, and structured light.

Stereo vision-based depth sensors are built based on binocular vision, which is also used in human vision systems. The depth information is calculated based on the difference in an object's location as seen by two different sensors or cameras. Therefore, stereo vision-based sensors need at least a pair of cameras that have sufficient details and a large field of view. Differently, time-of-flight-based sensors estimate the depth of a point by measuring the time that is needed for an emitted signal to be reflected by the specific point and come back to the depth sensor.

Structured light-based depth sensors use either laser or light emitting diode (LED) light and estimate the distance by measuring the distortions. Time of flight-based and structured light-based depth sensors are preferred in current AR devices.

2.2.2 Camera System and Eye Gaze Sensor

AR devices must have a group of cameras to ensure the AR system can see what the AR user is seeing. Besides being used for regular applications (e.g., video recording and video conferencing), the cameras on the AR headsets can be used for accurate head tracking, which further enables the head-gaze interaction. Except for the visible light cameras, many AR headsets, such as Microsoft HoloLens, use a pair of infrared radiation (IR) cameras that record the eyes of the AR user to track the gaze. By using the eye gaze sensors, the AR application can understand what the focus and intention of the AR user are.

2.2.3 Motion Sensor

Motion sensors are a group of sensors that are used to recognize the motion of the device itself and are implemented in all AR devices to support the basic functionality of AR applications. AR devices usually have three motion sensor types: accelerometer, gyroscope, and magnetometer. Besides the headsets, motion sensors are also implemented in the remote controllers of some AR devices. For example, HTC Vive uses motion sensors to help estimating the hand movement trajectory of users.

2.2.4 Audio System

Current AR devices have an audio system for audio signal collection and playback. For example, Microsoft HoloLens 2 has three microphones for ambient sound collection and a separate group of directional microphones to collect the voice of the AR user. A pair of loudspeakers are placed on both sides of the device to play audio signals back to the user for interaction. For instance, Microsoft HoloLens two has three microphone at the front of the device to collect ambient sounds in the physical environment and two directional microphones behind the glasses to collect the voice of the AR user.

3 Security and Privacy Concerns of Augmented Reality

Different types of sensors are the fundamental components for supporting immersive experiences in using AR devices. Figure 2 shows the general pipeline of sensor signal processing in AR systems. Both the AR user and the physical environment

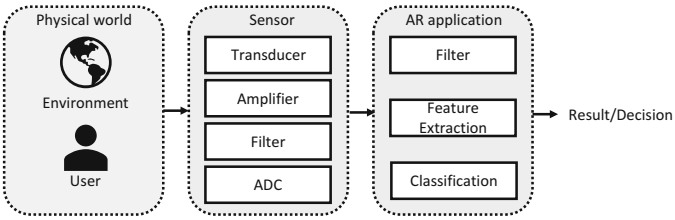


Fig. 2 The processing pipeline of sensor signals in AR systems

will stimulate the sensor, which is then picked up by the transducer to generate analog signals. The analog signals can be first processed within the sensor hardware using amplifiers and hardware filters. After that, analog signals are converted into digital signals using an analog-to-digital converter (ADC). AR applications acquire processed digital signals through the application programming interfaces (APIs) provided by the operating systems of the AR devices and perform further processing on the digital signals, such as software filters, feature extraction, and classification. The data processing in the AR applications can be done either locally on AR devices or remotely in the corresponding server of the application.

From this signal processing chain, we can obtain two important insights. First, the correct result or decision of AR systems primarily relies on accurate sensing. If attackers can manipulate the sensor signals by controlling the ambient environment, the attacker can likely manipulate the final result and decision of the AR application. This type of threat is referred to as input security threats of AR. Second, AR applications are allowed to gather a group of digital sensor signals from the hardware for their functionality. However, not all information in the signals is needed by the functionality. Some information in the sensor signals can reveal private information of the AR users or other subjects in the environment. This type of threat is referred to as input privacy threats of AR. Besides these two types of threats observed from the processing chain, AR systems can also suffer from different types of threats that target their output, which refers to as output security. Since AR systems significantly impact users' perception, especially visual perception, attackers can try to interfere with AR users' perception by placing particular virtual objects at specific locations for malicious purposes. The following sections will discuss these three types of threats separately with their threat models, found vulnerabilities, and existing defense solutions.

4 Input Security

This section will discuss the general threat model of input security of AR systems. Moreover, this section will study the security of three necessary inputs in AR systems based on recent research in this field.

4.1 Threat Model

As discussed in Sect. 3, the functions of AR systems rely on accurate sensing to work as expected. In other words, the collected sensor signals should be accurate and trustworthy. To attack the input security, attackers will try to manipulate the sensor signals by controlling specific signals in the same physical environment where the AR user is. As observed from Fig. 2, the sensor signals collected by AR devices are impacted by either the physical environment or the AR user. In practice, it is tough for attackers to directly control the AR user for manipulating specific sensor signals. For example, it is unrealistic for attackers to force the AR user to say some malicious voice commands to the AR system. Therefore, input security attackers will mostly try to control the ambient signals for manipulating sensor signals. The remaining part of this section will summarize existing research revealing security vulnerabilities of the input of AR systems and proposed defense solutions.

4.2 Audio Input Security

Unlike traditional mobile devices, such as smartphones, AR devices do not have a physical input surface for user interactions. In current AR headsets, voice is one of the most important interaction methods because it is the natural way for communication. For example, HoloLens users can say "Hey Cortana" followed by a voice command to give an instruction to the AR system. Besides being used for voice commands, the voiceprint can also be used for user authentication so that the AR device will only follow the instruction of a specific group of users.

4.2.1 Vulnerabilities

However, voice input suffers from various attacks, which can enable attackers to inject malicious voice commands into the AR systems remotely. Since human voices can be easily exposed to the public (e.g., via videos on social media), attackers can easily obtain the voice samples of the AR users and play the voice back to AR systems. Here are representative works that reveal the vulnerabilities of voice input.

Replay Attack Using Inaudible Channels A naive way to conduct a voice replay attack is using a loudspeaker. However, such attacks can be easily noticed by AR users. To avoid being detected, attackers tend to leverage inaudible channels to inject malicious voice commands into the AR systems of victims. The first inaudible channel that can be used for such attacks is ultrasound. Most microphones on existing AR devices are able to record both audible sounds and inaudible sounds and have the non-linearity property in the hardware. With the non-linearity property, high-frequency signals received by a microphone can be shifted to lower frequencies. For example, before transmitting the audio signals, DolphinAttack [87]

modulates the malicious voice signals on a new ultrasonic carrier. Due to the non-linearity property, such high-frequency malicious voice signals are demodulated to low frequencies on the microphone. Evaluation results in [87] show that such an attack is feasible to launch at a distance of five feet without being noticed. The work in [61] further improves the operation range of such attacks to 25 feet by leveraging multiple speakers.

Besides leveraging ultrasound, a recent work called Light Commands [72] shows that laser beams can be used to inject malicious voice commands. The insight is that the laser beam can induce mechanical vibration of the microphone's diaphragm. Similar to the vibration caused by air pressure, such movements of the microphone's diaphragm are converted to audio signals. Therefore, by modulating the intensity of laser beams based on the malicious voice commands, attackers can inject the malicious commands into victims' AR devices by pointing the laser beams at the microphone.

Recent research also shows that parasitic audio electric signals can be used to inject voice commands without physically touching the victim device [26, 38, 42]. More specifically, The cable wires, such as charging cables or headphone cables, can be used as antennas that are under the interference of audio electric signals and are subject to front-door and back-door coupling of electromagnetic. Although the frequency range of such wire antennas is from 80 MHz to 108 MHz, a radio signal that is modulated by the malicious voice command can be well received by the microphone hardware and demodulated to a lower frequency due to the non-linear property. The operation range of such attacks is usually within one to two meters.

Voice Recognition Attacks Besides injecting malicious voice commands through inaudible channels, attackers can also inject voice commands using audible channels and still avoid being detected by AR users. The basic idea of such attacks is to make the audible voice commands sound like harmless audios, such as nonsensical or non-command sounds. However, such harmless audios can either share the same features with malicious voice commands or can be wrongly classified by voice recognition models, which causes the successful injection of malicious commands.

Figure 3 shows a general processing pipeline of voice signals. After collecting voice signals from microphone hardware, a voice recognition system will perform pre-processing (e.g., noise removal and filtering) on the raw signals. Then, different features are extracted from the processed voice signals for the following model-based voice recognition. In voice recognition systems, mel-frequency cepstral

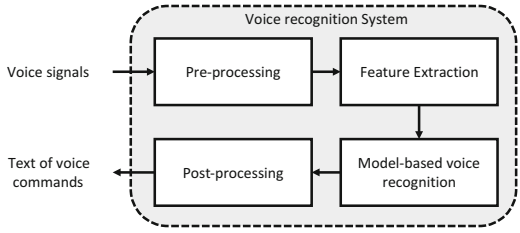


Fig. 3 A general processing pipeline of voice recognition system (built based on the figure in [78])

coefficients (MFCCs) are commonly used as features as they represent the short-term power spectrum of audio on a nonlinear mel frequency scale and match with human hearing perception in terms of frequency bands. The recognition results then go through post-processing and are sent to the operating system or corresponding AR application for action. From this processing pipeline, we can gain one crucial insight. If two voice signals can have similar features, such as MFCCs, after pre-processing and feature extraction, it is very likely that they will be recognized as the exact text, which causes the same actions performed by AR systems.

Based on the above idea, various vulnerabilities have been identified that enable attackers to inject malicious voice commands into a voice-controllable system (e.g., voice assistant of AR systems) [1, 14, 78]. For example, a system called Cocaine Noodles [78] proposes an attacking method that aims at generating an attacking audio signal that can: (1) retain enough features for making the signal be recognized as a malicious voice command and (2) sound like indecipherable noise to human beings. Specifically, Cocaine Noodles selects four MFCC parameters and uses the tuned MFCC parameters to compute MFCCs of a malicious voice signal. The extracted features are then converted back to a new audio signal which can be recognized as a malicious command but sounds like noise to human beings due to the noise added in the inverse computing.

Many existing voice recognition systems use machine learning models to predict the literal content of voice commands [56]. However, most machine learning models are trained with a hidden assumption that training and testing data are generated from the same statistical distribution, which can be false in real-world scenarios. Recent research in the machine learning security field shows that, by using special input called adversarial examples, attackers can cause machine learning models to make wrong predictions. An adversarial example is generated by adding adversarial perturbation to an instance of data, and the perturbation is constrained to be so small that it appears imperceptible to human beings. Inspired by adversarial machine learning in computer vision, many works have been proposed to leverage adversarial examples to inject malicious voice commands into voice recognition systems [7, 16, 17, 47, 83–85]. For example, the work in [7] presents an attack approach that can make a neural-network-based speech recognition system fail. Since it is hard for attackers to gain enough knowledge of the neural network, the proposed approach leverages a genetic algorithm that is gradient-fee for optimization. Evaluation results show that the added perturbation is of a small amount that sounds like background noise, but it is enough to change the predicted label of an audio clip. A similar approach is also proposed in [25] by leveraging Particle Swarm Optimization (PSO) algorithm and the fooling gradient method for optimization. Moreover, some recent studies show that adversarial examples that are generated for a white-box model could be transferred to another black-box models [2, 18, 19]. For instance, Abdullah et al. show that [2] perturbations to certain phonemes can cause wrong classification across multiple models.

4.2.2 Defense Solutions

Existing AR applications run on two types of devices, smartphones and AR headsets. Existing research has shown that microphones on smartphones cannot defend against voice replay attacks and voice injection attacks due to the design of omnidirectional microphones. Different from mobile phones, AR headsets pick up the voices of AR users using directional microphones that are under glasses and point to the mouth of the user, which means microphones on AR headsets can defend against voice replay attacks to some extent. However, directional microphones do not perfectly protect AR devices from voice replay attacks due to reflection and Diffraction. Therefore, it is essential to have an extra layer of protection to ensure the security of voice input on AR devices. Many studies have been done to prevent or detect attacks on voice input, and they are summarized as follows.

Signal Distortion Many works detect the existence of attacks on voice input by identifying defective components in audio signals [4, 10, 40]. No matter what attack is launched, attackers need to find a way to inject malicious audio signals into victim devices, as we discussed earlier in this section. During the injection, distortions can be introduced to malicious audio signals, which do not exist at all in legitimate audio signals. For example, the deep learning-based model in [40] can accurately detect malicious audio signals with an equal error rate of 6.7%. A system called Void [4] further reduces the training burden to a single classification model with just 97 features and provides an equal error rate of about 8.7%. The work in [10] leverages sub-bass over-excitation in the replayed audio signals as the key indicator for identifying attacks on voice input.

Authentication Another group of works detects attacks on voice input by determining whether the voice clips come from authorized subjects [33]. Such defense systems can prevent someone else from using the voice assistant of the victim. The basic idea behind such works is to leverage the voiceprint. The voices of different people have distinctive patterns of certain voice characteristics in the time-frequency domain, and such features can be used to recognize the identity of voice clips. However, such a defense strategy can fail when attackers generate synthetic voices of the victim using forgery techniques and replay malicious voice clips back to AR devices.

Liveness Detection The objective of liveness detection is to determine whether a signal is generated by a live human being, and therefore it is widely used in forgery detection. As discussed in this section, attackers must inject malicious audio signals using certain hardware through audible or inaudible channels. This fact implies that malicious voice clips can be detected by finding features in side channels that do not exist or only exist in human-generated signals. When detecting the liveness of voice, the system needs to make sure the selected features are complex for attackers to forge.

The first group of features that can be used for voice liveness detection is breathing-related features. Humans always breathe when speaking sentences, such as voice commands. Therefore, certain breathing-related features can be used as an indicator to prove that a voice clip is generated by live speakers. For instance, the works in [53, 68, 81] leverage the pop sounds when speaking certain phonemes as features to detect the liveness of voices. The system in [82] uses the pressure of air flows for liveness detection.

Moreover, mouth and lip motion is always there for live speakers, and such motion is highly correlated with phonemes. Based on this idea, many systems are proposed to detect voice liveness using the existence of mouth or lip motion [52, 88]. For instance, WiVo [52] uses Wi-Fi sensing to monitor the motion of mouth and correlate it with words in voice clips. The work in [88] leverages the speaker and microphone on smartphones as a Doppler radar to detect lip motion during speech. Besides, the air pressure in the ear canal can also be used to detect voice liveness based on the study in [65].

Other studies are conducted based on analyzing the key differences between human vocal systems and replay devices. For instance, Voicelive [86] shows that different phonemes are generated from different locations in the human vocal system but come from the same location in loudspeakers. The works in [63, 64] show that voice can propagate through the internal body for live speakers, and such internal body voice can be collected by a contact microphone that is attached to the AR user. By measuring the correlation between the voice signals and internal body voice, replay attacks can be detected with high accuracy. Such systems can be deployed in current AR headsets by adding an extra contact microphone.

4.3 Motion Input Security

Motion sensor signals are essential for supporting immersive AR experiences because they are directly impacted by users' movements. For example, accelerometer and gyroscope can be used to estimate the head orientation of AR users. Magic Leap 2 also has motion sensors in the pair of controllers to help provide 6° of tracking.

4.3.1 Vulnerabilities

However, similar to audio sensors, the motion sensors in AR devices are also vulnerable to out-of-band injection attacks. In out-of-band injection attacks, attackers aim to change sensor signals without changing the measured quantity itself [29]. Many out-of-band injection attacks focus on attacking vibrating structure gyroscope. The underlying principle of the vibrating structure gyroscope is that a vibrating object tends to continue to vibrate even if the support rotates. Therefore, by measuring the Coriolis force on its support, the rate of rotation can be determined. Inspired

by this principle, attack systems are designed by using acoustic noise to trigger the vibration of hardware components of vibrating structure gyroscope. Recent research has shown that high-power audio signals that are played near a vibrating structure gyroscope can greatly impact the performance of angular velocity measurements [70], which can be used in denial-of-service (DoS) attacks. The work in [70] finds that some micro-electro-mechanical systems (MEMS) gyroscopes resonate at audible and inaudible frequencies. Even with consumer-grade speakers, such acoustic noise can effectively change the values of gyroscope readings and further cause the drones to fluctuate. Moreover, the air pressure caused by acoustic signals can also displace the mass in accelerometer hardware, which can also be leveraged to manipulate acceleration readings [76].

The work in [80] presents an attack system that can not only degrade the performance of motion-based systems but also control the behavior of the victim system for a while. Differently, the attack in this work is delivered by changing the tonal frequency rather than attenuating the amplitude of digital signals. In their experiments, they launched this attack on multiple mixed reality devices, such as Oculus Rift CV1, HTC Vive, iPhone 7, and Samsung Galaxy S7. The devices in these experiments can either be used as AR devices or share similar hardware architecture with existing AR devices. The experimental results demonstrate that the proposed attack method can control gyroscope-based functions and degrade accelerometer-based functions. The modulated acoustic noise can cause fluctuation of virtual scenes on the HTC Vive headset and manipulate the movement of controllers of the headset and keep it for a while. Effective manipulation can also be observed on other mixed reality devices, and most attempts of manipulation can hold for a certain amount of time.

4.3.2 Defense Solutions

To defend against out-of-band injection attacks on motion sensors, various hardware-based and software-based have been proposed, and their solutions can be divided into three categories: hardware protection, sensor redundancy and fusion, and anomaly detection.

Physical Protection This type of defense solution aims at preventing out-of-band injection attacks at the beginning by using physical isolation and acoustic-dampening materials. For example, the gyroscope on iPhone 5S is not vulnerable due to the compact casing of the hardware circuit [70]. Also, the same work proposes to use an additional feedback capacitor that is connected to the sensing electrode, which can help tune the resonant frequency and the magnitude of the resonance. Moreover, by using foam as isolation material, the insertion loss in sound pressure level can reach about 120 dB when the foam is 1 inch thick based on a study in [60].

Sensor Redundancy and Fusion Some other works propose to remove this vulnerability by using more sensor fusion [11, 77, 80] after attacks happen. In terms

of sensor redundancy, multiple gyroscope sensors with resonance frequencies can be used together, so the manipulated sensor signal can be identified by using other gyroscope signals as a reference. Other types of sensors can also be used for the same sensing job to defend against out-of-band injection attacks. For example, red, green, and blue (RGB) and gray-scale cameras on current AR devices can also be used to estimate the orientation of the head of the AR user, so injection attacks on motion sensors could be identified using sensor fusion algorithms.

Anomaly Detection Similar to sensor redundancy and fusion, anomaly detection focus on detecting the existence of attacks after attackers launch them. Since most out-of-band injection attacks on motion sensors use acoustic signals, the existence of audio signals of certain frequencies can be suspicious. For example, the work in [80] suggests to detect the resonating sound actively with microphones.

4.4 Depth Input Security

Depth sensing is essential for making the AR experience more realistic. Even though many computer vision algorithms can detect objects in a two-dimensional image, the detection can be time-consuming and inflexible, which can cause a virtual object to float over a physical object. Depth sensing can conquer this obstacle with lower delay and overhead by estimating the point-to-point distance in the three-dimensional space. Depth sensing can support a wide range of functionalities in AR scenarios, including robust object detection and indoor scanning. Depth sensors have been widely implemented in current AR devices. For instance, Microsoft Hololens 2 uses time-of-flight depth sensors to estimate hand gestures of AR users. Both iPhone and iPad devices are equipped with Light Detection and Ranging (LiDAR) sensors to support AR experiences.

4.4.1 Vulnerabilities

Like other sensors, the data reported by depth sensors is completely impacted by the surrounding environment. Therefore, by controlling the ambient environment, attackers could be able to degrade the performance of depth sensing on certain objects and even manipulate the sensing results. As discussed in Sect. 2, existing depth sensing hardware is based on one of three techniques: stereo vision, structured light, and time of fight. Stereo vision-based depth sensors can be regarded as two RGB cameras. If attackers use strong visible light to illuminate both cameras, the differences in the location of an object as seen by two cameras could be wrongly calculated, which can largely degrade system performance. Structured light depth sensors measure distances by projecting a narrow band of light on a surface, which produces a line of illumination that appears distorted from other perspectives. A depth map can be constructed based on the distortion from different perspectives.

Compared with ToF depth sensors, structured light depth sensors usually need more calibration and are less robust under sunlight. Among all these three solutions, ToF depth sensors are the most popular in current AR systems, and LiDAR sensor is one of them.

LiDAR sensors measure the depth of a point by measuring the time of flight of the signal. However, recent studies show that LiDAR sensors suffer from chosen pattern injection attacks [13, 30, 73]. In such attacks, attackers first craft a specific depth point cloud based on a certain objective (e.g., hiding an object). The crafted depth point cloud is then sent to the depth sensing model, which can be a white-box or black-box model, for depth estimation. For instance, the work in [13] presents a white-box attack on LiDAR sensors. The experimental results show that the proposed attack system can generate fake depth points at all of the 16 vertical viewing angles and an 8° horizontal angle. Moreover, the work finds that strong and stabler laser pulses are received by the LiDAR sensor when the fake points are closer to the center of the depth point cloud.

Considering depth sensing models in practice are black-box, the works in [30, 73] further enhance the attacks by targeting black-box models. However, all the above works are offline attacks, which means the fake points are injected directly into the depth point cloud without physically using laser beams for injection. To address this limitation, the work in [62] conducts a large-scale measure study on LiDAR spoofing. Their preliminary experimental results show that LiDAR spoofing attacks are feasible with nanosecond-level configuration.

4.4.2 Defense Solutions

Various defense solutions are proposed to defend against attacks on depth sensors, and their solutions can be summarized into three categories: detection, mitigation, and randomization-based solutions.

Detection Defense solutions in this category aim at detecting attacks right after the attacks are launched. For example, CARLO proposed in [73] can detect depth spoofing by measuring the ratio of depth points in different areas. STAnDS in [34] is also designed to detect sensor attacks on depth sensors based on anomaly detection. The work in [6] aims at detecting LiDAR spoofing attacks using a decision tree for classification. However, such detection works are based on certain assumptions. When the assumptions are false, their performance could degrade. For instance, CARLO can only work for autonomous driving scenarios.

Mitigation Different from detection solutions, mitigation solutions focus on reducing the impacts of attacks or reducing the uncertainty of the system [36, 58, 67]. The work in [36] presents a solution using Marzullo's sensor fusion algorithm with an assumption that the total number of unreliable sensors is smaller than half of the number of all sensors. However, this assumption can be false if attackers have the ability to spoof multiple sensors at the same time. Another work in [67] suggests

reducing the receiving angle of LiDAR signals, but this defense strategy is not evaluated in a real testbed.

Randomization Existing LiDAR sensors generate signal pulses at a fixed rate. The key idea behind the randomization-based defense is to add randomness to signal pulse generation [58, 67]. As long as attackers do not know the pattern of how signals are generated, the success rate of attacks can be largely reduced. For instance, the work in [58] suggests varying the pulse rate or skipping a random number of pulses. However, this solution relies on high-frequency pulse generation provided by the sensors. Otherwise, the resolution of the depth cloud can be impacted.

5 Input Privacy

This section will introduce the privacy vulnerabilities in the input of AR devices and show how attackers can infer sensitive or behavioral information of AR users.

5.1 Threat Model

The operating systems on existing AR devices enable AR applications to request certain sensor data. For most AR devices, the requested digital signals will be handed to AR applications in the raw form. However, raw sensor signals contain rich information, which includes not only that needed for certain functions in AR applications but also extra information about the physical world. This fact gives attackers opportunities to infer sensitive or personal information from the raw sensor signals. The capabilities of input privacy attackers can be modeled as follows: (1) Attackers can install malware or malicious AR application on the device of the victim. (2) Attackers acquire sensor signals from the API provided by the operating system of AR devices. (3) The sensor signals can be analyzed either locally or remotely to infer sensitive information about the AR user or the surrounding physical space. For some sensors on AR devices, it is effortless for users to think of privacy issues in certain scenarios. For example, AR users may be concerned about visual privacy (including both depth and camera information) in private or sensitive scenarios. However, most users are less aware of the privacy issue in public areas (e.g., using cameras on the street) or when using sensors that they believe are not sensitive. In the remainder of this section, I will introduce existing research on revealing privacy issues on the input of AR devices and corresponding countermeasures.

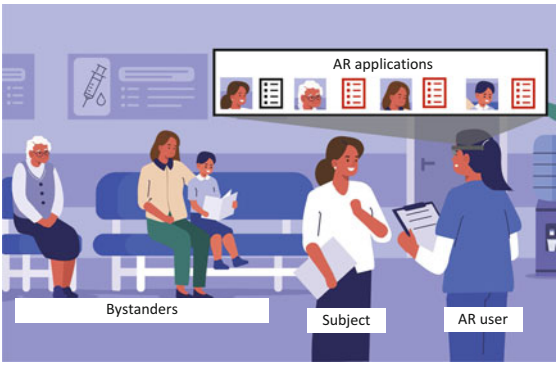


Fig. 4 An example to show the bystander privacy problem in AR world

5.2 Bystander Privacy

In the AR use case, there are three types of users: AR user, subject, and bystander. The AR user is the person wearing the AR device, and the subject is the person the AR user is interacting with. Bystanders can be any other people who are in the same physical space as the AR user and subject. The bystander privacy problem can arise when the data (e.g., images and videos) of a bystander that can be used to infer sensitive information is collected without the consent of the bystander. Figure 4 illustrates a bystander privacy problem in the hospital. The nurse is using the camera on the AR headset to log and verify the information of a subject who is a patient already giving consent to be part of the data collection. At the back of the subject, there are a few bystanders who do not sign the consent form but still appear in the camera frames, which potentially causes information leakage.

The bystander privacy problem is not novel and has been there since cameras are implemented on personal devices. However, the bystander privacy problem is more challenging in AR scenarios. In the use case of smartphones, bystanders could be aware of being recorded because of the relatively narrow field of view (FoV) of cameras. In AR scenarios, the cameras are always on and can record the surroundings with a much greater FoV, which makes it hard for bystanders to know whether they are being recorded and how their visual information will be used. In order to fight against the bystander privacy problem, various systems have been proposed, and their solutions can be classified into two groups: explicit solutions and implicit solutions.

5.2.1 Explicit Solutions

Explicit solutions request the user or the bystanders to perform explicit actions to protect bystander privacy. The works in [3, 45, 69] assume both user and bystanders use their system to ensure privacy protection. In these systems, bystanders need to upload their pictures and contract their privacy profiles or actively send a blurring

request to the user. However, such systems impose a significant burden on the user and bystanders, which largely reduces the usability of the system. Moreover, since all participants need to subscribe to the same service, the scalability can be very limited, and bystanders who do not enroll will be left unprotected. Also, the above most explicit solutions are based on a client-server model or rely on peer-to-peer communications, which could create another attack surface for attackers to break bystander privacy protection. Another system, LensCap [35], is built on top of split-process access control in order to prevent attackers from offloading private video frames to a remote server. Specifically, LensCap splits the video frame access and network access into two separate processes. If the network access attempts to access visual data, the visual data needs to be monitored and approved by the user. However, LensCap does not prevent all bystander privacy problems. For example, the malware can still analyze video frames locally on the device without accessing the network. Also, the protection of bystanders' visual data totally relies on a different participant, the user of the device, which can also cause potential privacy problems.

5.2.2 Implicit Solutions

To address the limitations of explicit solutions, many implicit solutions are proposed to reduce the efforts of users and bystanders [21–23, 31] by identifying and blurring bystanders in the frames. Different from explicit solutions, they leverage various features that are extracted from video frames to determine whether a person is a subject or a bystander and therefore reduce the effort from both AR users and bystanders. However, these works still have two limitations. First, these implicit systems can have degraded performance when the behaviors of bystanders differ a lot from those in the training dataset. For example, some works use the gaze direction of a person as a key feature with an assumption that only the subject will look at the user, which is not always true in practice. Second, the computation overhead of these solutions is relatively high, which makes real-time on-board processing impossible. Recently, a system called BystandAR [20] is proposed to address the limitations of existing implicit solutions. Different from using features of bystanders, BystandAR leverages visual concentration and the presence of the conversation between the AR user and the subject as two key features to identify bystanders. Experiments on Microsoft HoloLens 2 show that BystandAR can accurately identify bystanders and can run on a device without offloading processing tasks with an average frame rate of 52.6 frames per second.

5.3 Location Privacy

The location information of AR devices can leak the movements of AR users. Such information can also be combined with other side-channel information to infer more

private and sensitive information about users, such as habits, home addresses, and employment. Existing AR devices protect location privacy by leveraging permission management systems. Applications need to request location permission from the user before being allowed to acquire location data from the sensor. However, recent research shows that the location of the AR devices can be leaked through the side channel information [66, 74]. For example, the study in [74] shows that the location information can be inferred by attackers based on the signal exchange in the fifth generation of mobile communications network (5G). Also, a system called ARSpy [66] proposes a strategy to enable attackers to estimate the real-time location of AR devices by monitoring network traffic. The basic idea behind this work is that multi-player and location-based AR applications need to fetch AR content from content servers based on the geolocation of AR devices. The file sizes of AR contents are usually relatively large, which makes the downloading create notable patterns in network throughput. Moreover, many multi-player and location-based AR applications allow their users to "place" AR content at certain geolocation. Therefore, by placing AR contents with different sizes at different locations, attackers can estimate the location of the user by monitoring the network throughput based on pattern matching.

To defend against the above attacks, several countermeasures, including more secure protocols for 5G, are proposed but still need to be evaluated in a real testbed. For example, the work in [66] suggests AR software development kit(SDK) providers and developers should deploy and maintain an active cache with variable size to store AR contents. This work also suggests further limiting the permission control on sensor data that is regarded as less sensitive, such as the network throughput of an application.

5.4 Gaze Privacy

Eye gaze is important data that reflects the visual perception and behaviors of AR users. Existing sensor hardware and processing software can support accurate and lightweight eye gaze measurements, which then supports gaze-based interaction in current AR headsets. Existing eye tracking systems can measure multiple types of eye movements and behaviors, including fixations, saccades, pursuit eye movements, blink duration, blink frequency, ocular microtremors, pupil size, and pupil reactivity, but not all of them are available to application developers. For example, the eye-tracking API of HoloLens 2 can provide what the user is looking at as a single eye-gaze ray (gaze origin and direction) at approximately 30 FPS. However, even with limited gaze-related information, attackers can still infer sensitive information about AR users.

5.4.1 Identification

Based on the study in [54], the eye movement in response to a given stimulus is highly individual. Moreover, such individual characteristics can exist in a reliable way over time [8]. Based on these facts, many studies have shown that the biometric features in eye gaze can be used to identify users [9, 39, 48, 51, 75], especially when stimulus can be controlled. For instance, DeepEyedentificationLive [51] successfully uses a convolutional neural network to estimate the identity of eye gaze with controlled stimulus. The work in [48] also proposes a user identification system based on the eye gaze by leveraging two moving stimulus. Their experimental results show that the identification accuracy can be up to 75% for an explainable algorithm and 100% for a deep learning approach. These works imply it is possible for malicious AR application to infer who is using the AR headsets by showing moving virtual objects and monitoring the eye gaze.

5.4.2 Preferences and Knowledgeability

Since eye gaze is a reflection of physiological activities, visual focuses can also reveal the likes and dislikes of AR users. Malicious AR applications can create heat maps by aggregating the gaze trajectory samples to recognize the areas that the AR user is interested in [59]. For example, a prediction model based on eye tracking is proposed to infer the interest of a user from the non-click actions [46]. The experimental results show that the system can infer the interceded application of the user on Google Play Store with an accuracy of 90.32%. Also, the work in [15] presents a system to predict human knowledgeability from eye gaze where knowledgeability is represented by a binary value and associated with the user's feel of knowing.

5.4.3 Defense Solutions

The studies on defending against privacy leakage in eye gaze [12, 28, 41, 49, 50, 71] are still limited. Liebling et al. list future research directions for protecting eye gaze in [49], including allowing for self-introspection, using abstraction for gaze data, adding noise before passing it down to applications, and leveraging physical barriers against eavesdropping. Based on these ideas, the work in [71] proposes a differential privacy-based defense method by adding noise to eye gaze data to disable gazed-based user identification. At the same time, the added noise will not impact gazed-based functions, such as visual attention recognition. The work in [28] achieves gaze protection by leveraging reinforcement learning for eye-tracking data manipulation. The new AR headset of Apple also presents their solutions for protecting eye input: "Eye input is not shared with Apple, third-party apps, or websites. Only your final selections are transmitted when you tap your fingers together."

6 Output Safety, Security, and Privacy

As the final step in the process, AR applications need to render virtual objects over real-world scenes with the APIs provided by operating systems. The rendered objects can be anchored and non-anchored in the real world. Anchored objects are those that stay at the same position and orientation in space, and non-anchored virtual objects do not have a fixed position or orientation. Such rendering processes should be well managed. Otherwise, safety and privacy issues may arise. In this section, I will introduce the vulnerabilities in terms of output control and information leakage through output rendering. Existing countermeasures are then discussed.

6.1 Output Safety and Security

Since AR systems are used to augment the perception of users, virtual object rendering without management can cause various safety problems. For example, malicious applications can generate a virtual object that looks very similar to a stop sign in AR driving scenarios. Also, high-brightness virtual objects can block the sight of AR users on certain real-world objects, which could further cause security issues such as property security. These facts imply that an output control policy is necessary to guide AR applications for their rendering behaviors. A good output control policy should be able to achieve management of rendering priority, intelligent arrangement and occlusion, and other features to defend against attacks on AR output.

However, output control policies on existing AR devices are missing or very loose. Most operating systems of AR devices allow AR applications to place virtual objects at any position in the 3D world without any restrictions. To address this issue, several works have been proposed in recent years [5, 43, 44]. For example, the study in [43] points out two important design axes for managing visual AR outputs: flexibility and control. Flexibility means the ability of legitimate AR applications to display their AR contents, and control refers to the ability of the operating system to defend against malicious or undeniable AR content. Based on these two axes, they propose a novel model to manage visual AR contents at the granularity of AR objects rather than windows. In their follow-up work, they present a more detailed complete output policy control model called Arya [44]. Besides introducing new control policies, Arya also uses guidelines from HoloLens developer guidelines and the U.S. Department of Transportation guidelines. Moreover, Arya develops an explicitly restricted policy framework that requires policies to combine options from a well-defined set of parameterized conditions and mechanisms. In addition, the output policy control model has an issue for policy enforcement when those policies depend on relationships between objects. For example, a virtual object that does not block anything may block a walking person in future frames due to

the movements. Arya can address the issues by recognizing objects using various sensors and checking the enforcement of policies per frame. The work in [5] further improves the performance of policy control based on the weaknesses of Arya. In unknown and dynamic scenarios, this work leverages reinforcement learning to make optimal decisions for output control and policy enforcement. To prevent on-board computation overhead in Arya, this work offloads computation tasks to a local edge server.

6.2 Output Privacy

An ideal output control model should also be privacy-preserving. Here I use the same example in [24] to show the privacy issue. Assume the user is using an AR application that aims to project a screen to a flat surface, such as a wall. To verify if the wall is flat enough for such projection, the AR application can request camera data to determine whether there is anything on the wall that impacts the quality of the projection. This request seems reasonable but can leak information (e.g., the shape of objects on the wall) of real-world objects on the wall, even if the camera data is abstracted before being transmitted to AR applications. Existing research on protecting privacy during the output process is very limited. Vilk et al. present a solution to this issue in [79] with three types of abstraction: room skeleton, detection sandbox, and satellite screens. The room skeleton abstraction allows AR applications to place virtual contents based on the physical dimension and locations of renderable surfaces. The detection sandbox abstraction can enable applications to place contents near real-world objects without revealing the physical presence of objects. The satellite screens abstraction lets applications share contents across multiple AR devices.

7 Opportunities and Future Directions

This section discusses some opportunities and future directions of research on the security and privacy of AR systems.

Security of Depth Sensors Although there is much research on the security of depth sensors, some questions still need to be answered. First, as discussed in Sect. 4.4, existing research on depth sensor security only focuses on autonomous vehicle scenarios where depth sensors are mostly used for detecting big objects and measuring distances. It is still unclear how these attack methods impact the functionalities of AR devices that use depth sensing. One of the possible research opportunities is to study if attackers can leverage similar attack methods to manipulate depth sensing on AR headsets, such as interfering with virtual object rendering and spoofing gesture recognition. Second, most existing depth sensor

spoofing works target LiDAR sensors on autonomous vehicles. However, it is not clear whether attacks on such vehicle LiDAR sensors can also work against LiDAR sensors on AR devices. It is also not clear whether such attack ideas are also feasible against other ToF depth sensors on existing AR headsets. More studies need to be conducted to answer these questions.

Fast and Robust Sensor Attack Defense Most existing defense solutions for AR systems are only designed for attacks on a specific sensor. Considering the number of sensors on AR devices, a separate defense software for each sensor could introduce much computation overhead to the devices. Several sensor fusion-based solutions are presented in [36, 58, 67]. However, these systems are designed for autonomous vehicles, so it is not clear whether they can protect AR systems well. Therefore, a more generalized defense system needs to be designed to detect or mitigate attacks on various types of sensors accurately. Moreover, such a defense system needs to execute fast and robustly. Most existing AR devices are subject to battery constraints. For example, Apple Vision Pro can only support AR experience for about 2 hours. The execution of defense software could make this situation worse. Besides making the defense system lightweight, another research opportunity is to offload the computation to an edge server. However, offloading could create another attack surface for attackers if vulnerabilities exist in network communication or in edge servers. More research needs to be done to answer these questions.

Privacy of Vision Data Protecting sensitive information in camera and depth sensor data can also be a future research opportunity. Most existing defense solutions only target a specific scenario and can fail to protect vision data in a more generalized situation. The work in [37] presents a protection system that aims to protect camera data for a more generalized scenario with computer vision techniques. Their system performs abstraction on the objects in camera frames based on the privacy level which users specify. However, it is not clear enough to users how many details of an object will be reserved under each privacy level. A wrongly set privacy level could lead to low usability or privacy leakage. One possible future research direction is to build an implicit defense system that can protect all types of vision data in an intelligent way based on the scenario, behaviors of the user, and the nature of application functions.

8 Conclusion

This chapter gives a literature review on the security and privacy issues of AR systems, including a systematic review of AR systems, critical vulnerabilities, existing defense solutions, and future research directions. In terms of input security, this chapter gives a detailed discussion of security research on audio input, motion sensors, and depth sensing. In addition, bystander privacy, location privacy, and gaze privacy are reviewed under input privacy, and output security and privacy

are discussed in the end. This literature review shows that the existing research on protection AR devices is still minimal. More studies must be done for protection before the device becomes commonplace to the average consumer.

Acknowledgments This work is supported in the NSF grant under CNS 2153397.

References

1. Abdullah H, Garcia W, Peeters C, Traynor P, Butler KR, Wilson J (2019) Practical hidden voice attacks against speech and speaker recognition systems. Preprint, arXiv:190405734
2. Abdullah H, Rahman MS, Garcia W, Warren K, Yadav AS, Shrimpton T, Traynor P (2021) Hear" no evil", see" kenansville": efficient and transferable black-box attacks on speech recognition and voice identification systems. In: Proceedings of the IEEE symposium on security and privacy. IEEE, pp 712–729
3. Aditya P, Sen R, Druschel P, Joon OS, Benenson R, Fritz M, Schiele B, Bhattacharjee B, Wu TT (2016) I-pic: a platform for privacy-compliant image capture. In: Proceedings of the annual international conference on mobile systems, applications, and services, pp 235–248
4. Ahmed ME, Kwak IY, Huh JH, Kim I, Oh T, Kim H (2020) Void: a fast and light voice liveness detection system. In: Proceedings of the USENIX conference on security symposium, pp 2685–2702
5. Ahn S, Gorlatova M, Naghizadeh P, Chiang M, Mittal P (2018) Adaptive fog-based output security for augmented reality. In: Proceedings of the morning workshop on virtual reality and augmented reality network, pp 1–6
6. Alheeti KMA, Alzahrani A, Al Dosary D (2022) Lidar spoofing attack detection in autonomous vehicles. In: Proceedings of the IEEE international conference on consumer electronics, IEEE, pp 1–2
7. Alzantot M, Balaji B, Srivastava M (2018) Did you hear that? adversarial examples against automatic speech recognition. Preprint, arXiv:180100554
8. Bargary G, Bosten JM, Goodbourn PT, Lawrance-Owen AJ, Hogg RE, Mollon J (2017) Individual differences in human eye movements: an oculomotor signature? Vision Res 141:157–169
9. Bednarik R, Kinnunen T, Mihaila A, Fränti P (2005) Eye-movements as a biometric. In: Proceedings of the image analysis: 14th Scandinavian conference. Springer, pp 780–789
10. Blue L, Vargas L, Traynor P (2018) Hello, is it me you're looking for? Differentiating between human and electronic speakers for voice interface security. In: Proceedings of the ACM conference on security & privacy in wireless and mobile networks, pp 123–133
11. Bolton C, Rampazzi S, Li C, Kwong A, Xu W, Fu K (2018) Blue note: how intentional acoustic interference damages availability and integrity in hard disk drives and operating systems. In: Proceedings of the IEEE symposium on security and privacy, IEEE, pp 1048–1062
12. Bozkir E, Ünal AB, Akgün M, Kasneci E, Pfeifer N (2020) Privacy preserving gaze estimation using synthetic images via a randomized encoding based framework. In: Proceedings of the ACM symposium on eye tracking research and applications, pp 1–5
13. Cao Y, Xiao C, Cyr B, Zhou Y, Park W, Rampazzi S, Chen QA, Fu K, Mao ZM (2019) Adversarial sensor attack on lidar-based perception in autonomous driving. In: Proceedings of the ACM SIGSAC conference on computer and communications security, pp 2267–2281
14. Carlini N, Mishra P, Vaidya T, Zhang Y, Sherr M, Shields C, Wagner DA, Zhou W (2016) Hidden voice commands. In: Proceedings of the USENIX security symposium, pp 513–530
15. Celiktutan O, Demiris Y (2018) Inferring human knowledgeability from eye gaze in mobile learning environments. In: Proceedings of the European conference on computer vision workshops, pp 0–0

16. Chang KH, Huang PH, Yu H, Jin Y, Wang TC (2020) Audio adversarial examples generation with recurrent neural networks. In: Proceedings of the Asia and South Pacific design automation conference, IEEE, pp 488–493
17. Chen T, Shangguan L, Li Z, Jamieson K (2020) Metamorph: injecting inaudible commands into over-the-air voice controlled systems. In: Proceedings of the network and distributed systems security symposium
18. Chen Y, Yuan X, Zhang J, Zhao Y, Zhang S, Chen K, Wang X (2020) Devil's whisper: a general approach for physical adversarial attacks against commercial black-box speech recognition devices. In: Proceedings of the USENIX security symposium, pp 2667–2684
19. Cisse MM, Adi Y, Neverova N, Keshet J (2017) Houdini: Fooling deep structured visual and speech recognition models with adversarial examples. Adv Neural Inf Proc Syst 30
20. Corbett M, David-John B, Shang J, Hu YC, Ji B (2023) Bystandar: protecting bystander visual data in augmented reality systems. In: Proceedings of the annual international conference on mobile systems, applications, and services
21. Darling D (2021) Automated privacy protection for mobile device users and bystanders in public spaces. University of Arkansas, Fayetteville
22. Darling D, Li A, Li Q (2019) Identification of subjects and bystanders in photos with feature-based machine learning. In: Proceedings of the IEEE conference on computer communications workshops, IEEE, pp 1–6
23. Darling D, Li A, Li Q (2020) Automated bystander detection and anonymization in mobile photography. In: Proceedings of the international conference on security and privacy in communication networks, Springer, pp 402–424
24. De Guzman JA, Thilakarathna K, Seneviratne A (2019) Security and privacy approaches in mixed reality: a literature survey. ACM Comput Surv 52(6):1–37
25. Du T, Ji S, Li J, Gu Q, Wang T, Beyah R (2020) Sirenattack: Generating adversarial audio for end-to-end acoustic systems. In: Proceedings of the ACM Asia conference on computer and communications security, pp 357–369
26. Esteves JL, Kasmi C (2018) Remote and silent voice command injection on a smartphone through conducted iemi: threats of smart iemi for information security. Wireless Security Lab, French Network and Information Security Agency (ANSSI), Technical Report
27. Exploding topics. https://explodingtopics.com/blog/augmented-reality-stats#ar-user-stats
28. Fuhl W, Bozkir E, Kasneci E (2021) Reinforcement learning for the privacy preservation and manipulation of eye tracking data. In: Proceedings of the international conference on artificial neural networks, Springer, pp 595–607
29. Giechaskiel I, Rasmussen K (2019) Taxonomy and challenges of out-of-band signal injection attacks and defenses. IEEE Commun Surv Tutor 22(1):645–670
30. Hallyburton RS, Liu Y, Cao Y, Mao ZM, Pajic M (2022) Security analysis of camera-lidar fusion against black-box attacks on autonomous vehicles. In: Proceedings of the USENIX security symposium, pp 1903–1920
31. Hasan R, Crandall D, Fritz M, Kapadia A (2020) Automatically detecting bystanders in photos to reduce privacy risks. In: Proceedings of the IEEE symposium on security and privacy, IEEE, pp 318–335
32. Haselton T (2019) How the army plans to use microsoft's high-tech hololens goggles on the battlefield. https://explodingtopics.com/blog/augmented-reality-stats#ar-user-stats
33. He R, Ji X, Li X, Cheng Y, Xu W (2022) Ok, siri" or" hey, google": evaluating voiceprint distinctiveness via content-based prole score. In: Proceedings of the USENIX security symposium
34. Higgins M, Jha D, Wallom D (2022) Spatial-temporal anomaly detection for sensor attacks in autonomous vehicles. Preprint, arXiv:221207757
35. Hu J, Iosifescu A, LiKamWa R (2021) Lenscap: split-process framework for fine-grained visual privacy control for augmented reality apps. In: Proceedings of the annual international conference on mobile systems, applications, and services, pp 14–27
36. Ivanov R, Pajic M, Lee I (2014) Attack-resilient sensor fusion. In: Proceedings of the design, automation & test in Europe conference & exhibition. IEEE, pp 1–6

37. Jana S, Narayanan A, Shmatikov V (2013) A scanner darkly: protecting user privacy from perceptual applications. In: Proceedings of the IEEE symposium on security and privacy. IEEE, pp 349–363
38. Kasmi C, Esteves JL (2015) Iemi threats for information security: remote command injection on modern smartphones. IEEE Trans Electromagn Compat 57(6):1752–1755
39. Kasprowski P, Ober J (2004) Eye movements in biometrics. In: Proceedings of the ECCV workshop BioAW. Springer, pp 248–258
40. Kinnunen T, Sahidullah M, Delgado H, Todisco M, Evans N, Yamagishi J, Lee KA (2017) Assessing the limits of replay spoofing attack detection. The ASVspoof challenge
41. Kröger JL, Lutz OHM, Müller F (2020) What does your gaze reveal about you? on the privacy implications of eye tracking. Privacy and identity management data for better living: AI and privacy: 14th IFIP WG 92, 96/117, 116/SIG 92 2 international summer school, Windisch, Switzerland, August 19–23, 2019. Revised Selected Papers 14 pp 226–241
42. Kune DF, Backes J, Clark SS, Kramer D, Reynolds M, Fu K, Kim Y, Xu W (2013) Ghost talk: mitigating emi signal injection attacks against analog sensors. In: Proceedings of the IEEE symposium on security and privacy. IEEE, pp 145–159
43. Lebeck K, Kohno T, Roesner F (2016) How to safely augment reality: Challenges and directions. In: Proceedings of the international workshop on mobile computing systems and applications, pp 45–50
44. Lebeck K, Ruth K, Kohno T, Roesner F (2017) Securing augmented reality output. In: Proceedings of the IEEE symposium on security and privacy. IEEE, pp 320–337
45. Li A, Li Q, Gao W (2016) Privacycamera: cooperative privacy-aware photographing with mobile phones. In: Proceedings of the annual IEEE international conference on sensing, communication, and networking. IEEE, pp 1–9
46. Li Y, Xu P, Lagun D, Navalpakkam V (2017) Towards measuring and inferring user interest from gaze. In: Proceedings of the international conference on world wide web companion, pp 525–533
47. Li Z, Wu Y, Liu J, Chen Y, Yuan B (2020) Advpulse: Universal, synchronization-free, and targeted audio adversarial attacks via subsecond perturbations. In: Proceedings of the ACM SIGSAC conference on computer and communications security, pp 1121–1134
48. Liebers J, Horn P, Burschik C, Gruenefeld U, Schneegass S (2021) Using gaze behavior and head orientation for implicit identification in virtual reality. In: Proceedings ACM symposium on virtual reality software and technology, pp 1–9
49. Liebling DJ, Preibusch S (2014) Privacy considerations for a pervasive eye tracking world. In: Proceedings of the ACM international joint conference on pervasive and ubiquitous computing: adjunct publication, pp 1169–1177
50. Liu A, Xia L, Duchowski A, Bailey R, Holmqvist K, Jain E (2019) Differential privacy for eye-tracking data. In: Proceedings of the ACM symposium on eye tracking research & applications, pp 1–10
51. Makowski S, Prasse P, Reich DR, Krakowczyk D, Jäger LA, Scheffer T (2021) Deepeyedentificationlive: Oculomotoric biometric identification and presentation-attack detection using deep neural networks. IEEE Trans Biom Behav Identity Sci 3(4):506–518
52. Meng Y, Wang Z, Zhang W, Wu P, Zhu H, Liang X, Liu Y (2018) Wivo: Enhancing the security of voice control system via wireless signal in iot environment. In: Proceedings of the ACM international symposium on mobile Ad Hoc networking and computing, pp 81–90
53. Mochizuki S, Shiota S, Kiya H (2018) Voice liveness detection using phoneme-based pop-noise detector for speaker verifcation. In: Proceedings of the Odyssey speaker lang. Recognit. Workshop
54. Noton D, Stark L (1971) Scanpaths in eye movements during pattern perception. Science 171(3968):308–311
55. Novet J (2021) Microsoft wins u.s. army contract for augmented reality headsets, worth up to $21.9 billion over 10 years. https://www.cnbc.com/2021/03/31/microsoft-wins-contract-to-make-modified-hololens-for-us-army.html

56. Padmanabhan J, Johnson Premkumar MJ (2015) Machine learning in automatic speech recognition: a survey. IETE Tech Rev 32(4):240–251
57. Perez S (2020) Amazon rolls out a new ar shopping feature for viewing multiple items at once. https://techcrunch.com/2020/08/25/amazon-rolls-out-a-new-ar-shopping-feature-for-viewing-multiple-items-at-once/
58. Petit J, Stottelaar B, Feiri M, Kargl F (2015) Remote attacks on automated vehicles sensors: Experiments on camera and lidar. Black Hat Eur 11(2015):995
59. Ravi B (2017) Privacy issues in virtual reality: eye tracking technology. Bloomberg Law, Arlington County
60. Roth G (2009) Simulation of the effects of acoustic noise on mems gyroscopes. PhD Thesis
61. Roy N, Shen S, Hassanieh H, Choudhury RR (2018) Inaudible voice commands: The long-range attack and defense. In: Proceedings of the USENIX symposium on networked systems design and implementation, pp 547–560
62. Sato T, Hayakawa Y, Suzuki R, Shiiki Y, Yoshioka K, Chen QA (2022) Poster: Towards large-scale measurement study on lidar spoofing attacks against object detection. In: Proceedings of the ACM SIGSAC conference on computer and communications security, pp 3459–3461
63. Shang J, Wu J (2019) Enabling secure voice input on augmented reality headsets using internal body voice. In: Proceedings of the annual IEEE international conference on sensing, communication, and networking, IEEE, pp 1–9
64. Shang J, Wu J (2020) Secure voice input on augmented reality headsets. IEEE Trans Mob Comput 21(4):1420–1433
65. Shang J, Wu J (2020) Voice liveness detection for voice assistants using ear canal pressure. In: Proceedings of the IEEE international conference on mobile Ad Hoc and sensor systems. IEEE, pp 693–701
66. Shang J, Chen S, Wu J, Yin S (2020) Arspy: Breaking location-based multi-player augmented reality application for user location tracking. IEEE Trans Mob Comput 21(2):433–447
67. Shin H, Kim D, Kwon Y, Kim Y (2017) Illusion and dazzle: adversarial optical channel exploits against lidars for automotive applications. In: Proceedings of the international conference on cryptographic hardware and embedded systems. Springer, pp 445–467
68. Shiota S, Villavicencio F, Yamagishi J, Ono N, Echizen I, Matsui T (2016) Voice liveness detection for speaker verification based on a tandem single/double-channel pop noise detector. In: Odyssey, vol 2016, pp 259–263
69. Shu J, Zheng R, Hui P (2018) Cardea: Context-aware visual privacy protection for photo taking and sharing. In: Proceedings of the ACM multimedia systems conference, pp 304–315
70. Son Y, Shin H, Kim D, Park Y, Noh J, Choi K, Choi J, Kim Y (2015) Rocking drones with intentional sound noise on gyroscopic sensors. In: Proceedings of the USENIX security symposium, pp 881–896
71. Steil J, Hagestedt I, Huang MX, Bulling A (2019) Privacy-aware eye tracking using differential privacy. In: Proceedings of the ACM symposium on eye tracking research & applications, pp 1–9
72. Sugawara T, Cyr B, Rampazzi S, Genkin D, Fu K (2020) Light commands: laser-based audio injection attacks on voice-controllable systems. In: Proceedings of the USENIX conference on security symposium, pp 2631–2648
73. Sun JS, Cao YC, Chen QA, Mao ZM (2020) Towards robust lidar-based perception in autonomous driving: General black-box adversarial sensor attack and countermeasures. In: Proceedings of the USENIX security symposium
74. Tomasin S, Centenaro M, Seco-Granados G, Roth S, Sezgin A (2021) Location-privacy leakage and integrated solutions for 5g cellular networks and beyond. Sensors 21(15):5176
75. Tricomi PP, Nenna F, Pajola L, Conti M, Gamberi L (2023) You can't hide behind your headset: user profiling in augmented and virtual reality. IEEE Access 11:9859–9875
76. Trippel T, Weisse O, Xu W, Honeyman P, Fu K (2017) Walnut: waging doubt on the integrity of mems accelerometers with acoustic injection attacks. In: Proceedings of the IEEE European symposium on security and privacy. IEEE, pp 3–18

77. Tu Y, Lin Z, Lee I, Hei X (2018) Injected and delivered: fabricating implicit control over actuation systems by spoofing inertial sensors. In: Proceedings of the USENIX security symposium, pp 1545–1562
78. Vaidya T, Zhang Y, Sherr M, Shields C (2015) Cocaine noodles: exploiting the gap between human and machine speech recognition. In: Proceedings of the USENIX workshop on offensive technologies
79. Vilk J, Molnar D, Livshits B, Ofek E, Rossbach C, Moshchuk A, Wang HJ, Gal R (2015) Surroundweb: Mitigating privacy concerns in a 3d web browser. In: Proceedings of the IEEE symposium on security and privacy. IEEE, pp 431–446
80. Wang Z, Wang K, Yang B, Li S, Pan A (2017) Sonic gun to smart devices: your devices lose control under ultrasound/sound. Black Hat USA pp 1–50
81. Wang Q, Lin X, Zhou M, Chen Y, Wang C, Li Q, Luo X (2019) Voicepop: A pop noise based anti-spoofing system for voice authentication on smartphones. In: Proceedings of the IEEE conference on computer communications. IEEE, pp 2062–2070
82. Wang Y, Cai W, Gu T, Shao W, Li Y, Yu Y (2019) Secure your voice: An oral airflow-based continuous liveness detection for voice assistants. Proc ACM Interact Mob Wearable Ubiquitous Technol 3(4):1–28
83. Yakura H, Sakuma J (2018) Robust audio adversarial example for a physical attack. Preprint, arXiv:181011793
84. Yan C, Ji X, Wang K, Jiang Q, Jin Z, Xu W (2022) A survey on voice assistant security: attacks and countermeasures. ACM Comput Surv 55(4):1–36
85. Yuan X, Chen Y, Zhao Y, Long Y, Liu X, Chen K, Zhang S, Huang H, Wang X, Gunter CA (2018) Commandersong: a systematic approach for practical adversarial voice recognition. In: Proceedings of the USENIX security symposium, pp 49–64
86. Zhang L, Tan S, Yang J, Chen Y (2016) Voicelive: a phoneme localization based liveness detection for voice authentication on smartphones. In: Proceedings of the ACM SIGSAC conference on computer and communications security, pp 1080–1091
87. Zhang G, Yan C, Ji X, Zhang T, Zhang T, Xu W (2017) Dolphinattack: Inaudible voice commands. In: Proceedings of the ACM SIGSAC conference on computer and communications security, pp 103–117
88. Zhang L, Tan S, Yang J (2017) Hearing your voice is not enough: An articulatory gesture based liveness detection for voice authentication. In: Proceedings of the ACM SIGSAC conference on computer and communications security, pp 57–71
89. Zhang Z, Zhu X, Li Y, Chen X, Guo Y (2020) Adversarial attacks on monocular depth estimation. Preprint, arXiv:200310315

Securing Augmented Reality Applications

Si Chen and Jie Wu

1 Introduction

With the advent of digital advancements, our experiences of reality are increasingly becoming intertwined with technology. On the forefront of these developments stands Augmented Reality (AR) [1], a technology that overlays digital information on real-world elements. From healthcare to education to entertainment, AR has permeated numerous fields. Yet, as we stand on the brink of this digital revolution, concerns around security loom large. Intrusive invasions via sensors, cyber attacks, and data privacy breaches present noteworthy challenges in the AR landscape. It is in this critical juncture that Artificial Intelligence and Machine Learning (AI/ML) surface as promising contributors. AI/ML, known for their prowess in learning from data and improving upon experiences, could hold the key to securing AR applications. This chapter delves deep into the security concerns of AR applications, analyzes the potential of AI/ML in combatting these threats, and sheds light on finding the delicate balance between security necessities and offering an advanced user experience.

S. Chen (✉)
West Chester University, West Chester, PA, USA
e-mail: schen@wcupa.edu

J. Wu
Temple University, Philadelphia, PA, USA
e-mail: jiewu@temple.edu

Y. Chen et al. (eds.), *Network Security Empowered by Artificial Intelligence*,
Advances in Information Security 107, https://doi.org/10.1007/978-3-031-53510-9_12

1.1 Background

Augmented Reality (AR), a pioneering technology that superimposes digitally generated information onto a user's perception of the real world, is increasingly being adopted across a variety of applications [2]. This revolutionary medium blurs the boundaries between the physical and virtual worlds, augmenting our perception and interaction with the real environment. It is considered one of the most sophisticated technologies in virtual reality research and has proven effective as a learning medium [3].

In the field of education, AR has been used to make challenging concepts visible and accessible to novices. For instance, it has been used to teach molecular geometry in chemistry, where students can interact with virtual models of molecules [3]. Similarly, in physics, AR has been used to expose learners to the invisible physics involved in audio speakers, such as the shape of magnetic fields and the relationships between electricity and magnetism [4].

AR has also found significant applications in the tourism and hospitality industry, where it is used for planning, marketing, and education [5]. In the industrial engineering domain, AR has been used to support remote maintenance and repair operations, providing real-time feedback from the operator's field of view [6].

1.1.1 The Early Years: A Detailed Overview

Augmented Reality (AR), a technology that overlays digital information onto the physical world, has its roots in the early 1990s. During this period, the first functional AR systems were developed, providing immersive mixed reality experiences for users. These pioneering systems [7] laid the groundwork for the sophisticated AR technologies we see today.

One of the earliest and most notable examples of AR technology is the Virtual Fixtures system [8, 9], which was developed in 1992 at the U.S. Air Force's Armstrong Laboratory. This groundbreaking system was a significant milestone in the evolution of AR technology, as it demonstrated the potential of AR to enhance human performance in a tangible and practical way.

The Virtual Fixtures system worked by overlaying virtual objects onto a real-world environment. This was achieved through the use of a head-mounted display (HMD) and spatially registered graphics. The user would see the real world around them, but with the addition of virtual objects that appeared to exist within the same space. These virtual objects, or "fixtures", could be manipulated by the user, providing a sense of interaction with the virtual environment.

The primary goal of the Virtual Fixtures system was to enhance the user's perception and performance in manual tasks. By overlaying virtual objects onto the real world, the system could provide visual guidance, improve spatial awareness, and facilitate complex task completion. The success of the Virtual Fixtures system demonstrated the potential of AR technology to enhance human capabilities, paving the way for future developments in the field.

In conclusion, the early 1990s marked a significant period in the history of AR technology. The development of the Virtual Fixtures system at the U.S. Air Force's Armstrong Laboratory demonstrated the potential of AR to enhance human performance, setting the stage for the advanced AR technologies we see today.

1.1.2 Mainstream Adoption: An In-depth Examination

The journey of Augmented Reality (AR) towards mainstream adoption began in earnest in the 2000s, a period marked by significant advancements in technology and a growing recognition of the potential applications of AR [10] in various sectors.

One of the earliest instances of AR gaining traction in the mainstream was through its application in the tourism industry. Developers began creating AR applications specifically designed for tourism [11] , providing a novel way for travelers to engage with their surroundings. These applications worked by overlaying digital information, such as historical facts, points of interest, and directions, onto the real-world view of the user. This not only enhanced the user's understanding of various tourist sites but also enriched their overall travel experience by providing an interactive and immersive way to explore new places.

As the decade progressed, AR technology began to find its way into commercial applications, particularly in the entertainment and gaming industries. This was a significant development, as it marked the first time AR was used in a mass-market context, reaching a wide audience of consumers.

A notable example of this was the introduction of AR games, which combined the real world with virtual elements to create engaging and immersive gaming experiences. Among these, Pokémon Go stands out as a particularly successful instance of AR gaming. Launched in 2016, Pokémon Go used AR technology to overlay virtual creatures, known as Pokémon, onto the real-world environment. Players could then interact with these creatures through their mobile devices, creating a gaming experience that was both novel and engaging.

The success of Pokémon Go demonstrated the potential of AR to create immersive experiences that could captivate a mass audience. It also served as a powerful example of how AR could be integrated into everyday life, contributing significantly to the mainstreaming of AR technology.

In summary, the 2000s marked a pivotal period in the journey of AR towards mainstream adoption. The development of AR applications for tourism and the introduction of AR games like Pokémon Go demonstrated the wide-ranging potential of AR, paving the way for its integration into various sectors and its acceptance by a broad consumer base.

1.1.3 Recent Developments: A Comprehensive Analysis

In the past few years, Augmented Reality (AR) has seen a significant expansion in its applications across a multitude of industries, driven by advancements in technology

and a growing recognition of its potential to enhance various aspects of human experience.

In the field of education, AR has emerged as a powerful tool for creating interactive learning experiences. By scanning or viewing an image with a mobile device, students can access AR content that brings learning materials to life. This can include 3D models, animations, or additional information that enhances understanding and engagement with the subject matter. The use of AR in education has the potential to transform traditional learning methods, making education more engaging, interactive, and effective.

The medical field [12], has also seen the integration of AR technology, particularly in surgical procedures. AR has been used to enhance visualization during surgeries allowing surgeons to overlay digital images onto the real-world view of the patient's body. This can provide valuable guidance during complex procedures, improving precision and potentially leading to better surgical outcomes.

In the entertainment industry, AR has been used to create immersive experiences that blend the real and virtual worlds. This has been particularly evident in the gaming industry, where AR games create interactive experiences that integrate virtual elements into the player's real-world environment. However, the use of AR in entertainment extends beyond gaming, with applications in film, television, and live events.

The development and proliferation of AR technology have been facilitated by advancements in related fields, such as computer vision and object recognition. These technologies have made it possible to overlay digital information onto the real world in real-time, creating an interactive and digitally manipulated environment. Computer vision enables devices to understand and interpret the real-world environment, while object recognition allows for the identification and tracking of specific objects within that environment. Together, these technologies form the backbone of AR, enabling the creation of immersive and interactive AR experiences.

In conclusion, recent developments in AR technology have seen its application expand across various industries, from education and medicine to entertainment. These advancements, coupled with developments in related fields such as computer vision and object recognition, have facilitated the creation of interactive and immersive AR experiences, marking a significant step forward in the evolution of this technology.

1.1.4 Future Trends

As AR technology continues to evolve, it is expected to find even more applications, transforming the way we learn, work, and interact with our environment. However, it's important to note that most users and application developers often overlook the potential risk of location privacy leakage in their applications. Unlike traditional smartphones where users have control over when to turn on or off the sensors

Fig. 1 Screenshots for mobile AR apps

and applications, AR devices continuously sense the environment through multiple sensors. If these sensors are exploited by an attacker, they could pose a severe threat to user privacy [13].

Mobile augmented reality (mobile AR), an emerging class of AR systems, is nearing commercial feasibility [14]. As a form of human-computer interface in cyber-physical systems, mobile AR systems form a conduit between the human and physical world through the cyber realm. With the rising shift towards hands-free wearable devices such as head-mounted displays and smart glasses, mobile AR is evolving into a novel information-delivery paradigm [15]. Unlike conventional smartphones which can be easily moved in and out of a user's field of vision, mobile AR continuously interacts with the environment and receives input from the user's field of vision via video, audio, and other sensors [16] (e.g., Fig. 1). Both cyber and physical attacks against mobile AR systems can lead to malfunction and subsequently disruption or failure of the mobile AR system. Therefore, it is vital to develop a robust security framework that is specifically tailored for mobile AR [17].

The future of AR is promising, with ongoing research and development aimed at improving the technology's capabilities and finding new applications. As AR technology continues to evolve, it is expected to become an integral part of our daily lives, changing the way we interact with the world around us.

1.2 *The Imperative of Security in Augmented Reality (AR) Applications*

As Augmented Reality (AR) technologies mature and permeate various industries, the necessity to safeguard these systems against harmful or disruptive visual outputs, generated by malicious or defective applications, has become critical. This is especially significant given that AR users often interact within ecosystems of other users, thereby amplifying the interconnected and extensive risks of AR [18] (Fig. 2).

AR techniques have been employed in the security sector to facilitate information exchange and provide immediate situational awareness [19]. However, the persistent interaction of AR with the user's environment and sensory field through video, audio, and other sensors, introduces substantial security, privacy, and safety issues [20]. For example, AR applications may unintentionally expose sensitive information about the user's environment or behavior, leading to privacy violations. Furthermore, AR applications can be manipulated to generate deceptive or harmful visual outputs, such as concealing real-world objects or producing distracting or misleading visual cues [21].

To mitigate these issues, researchers have suggested the implementation of adaptive policies to secure visual output in AR systems using deep reinforcement learning. These policies intelligently reposition AR content to minimize obstruction of real-world objects, while preserving a satisfactory user experience. This method

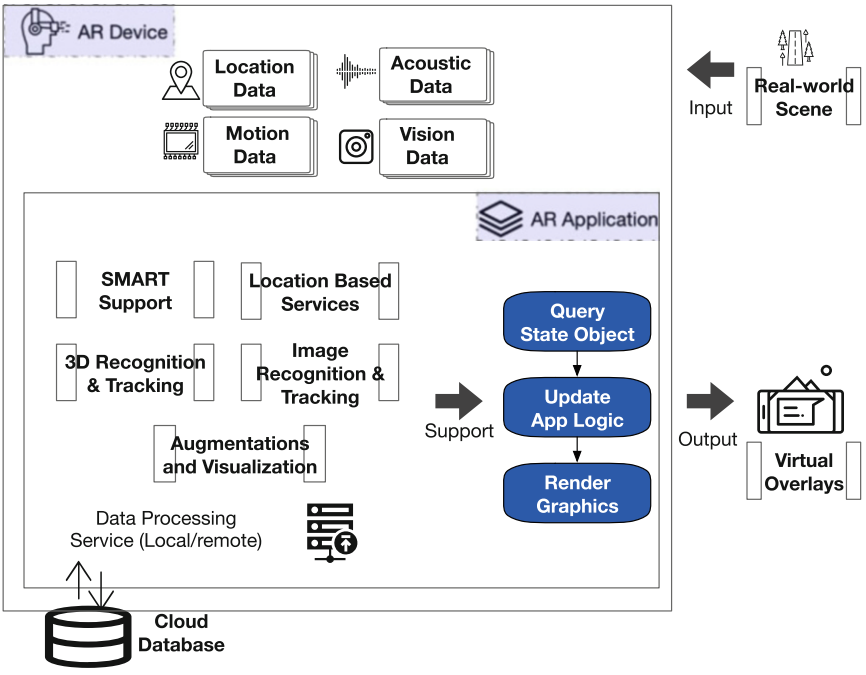


Fig. 2 Architecture of a standard AR application

employs a local fog computing node, which conducts training simulations to autonomously learn an appropriate policy for filtering potentially malicious or distracting content generated by an application [22].

Additionally, the fusion of AR with other emerging technologies like the Internet of Things (IoT) presents opportunities for developing innovative systems related to industrial safety and security. For instance, AR can be utilized for real-time monitoring of mechanical structures, enhancing the safety of the work environment [23]. In a specific instance, AR was employed to monitor the subsidence of a rock salt mine, providing real-time feedback and enhancing the safety of the mining operations [24].

In summary, the significance of security in AR applications is undeniable. As AR technologies continue to progress and become more ingrained in our daily lives, it is essential to address the unique security challenges they pose to ensure a safe and secure user experience.

The security of a conventional mobile system is typically associated with system components (e.g., software and firmware), or software security approaches such as control flow integrity (CFI), memory isolation, and hardening. Existing solutions are effective in defending against attacks initiated via a cyber vector such as program vulnerability exploitation with cyber payloads, including injected Trojan code and ROP. However, the unique blend of continuous GPS sensing, high-volume visual data capturing, and outsourced image processing in AR systems introduces several new categories of threats. For defending against cyber attacks in AR systems, we identify six types of threats:

- **Privacy Concerns, Personal and Environmental Data Exposure:** AR systems can pose privacy risks as they rely on the continuous collection of sensor and biometric data about users and their environment. Data such as user location, viewing direction, facial expressions, and even physiological responses to different stimuli can be exploited by malicious attackers and intrusive advertisers.
- **Advertisements and their Drawbacks:** The capability of AR systems to superimpose the real world with virtual objects can be leveraged by advertisers, often excessively. Intrusive advertisements in a user's personal space can be uncomfortable, even as they generate revenue for application developers. Furthermore, fraudulent advertisement displays could potentially lead to phishing attacks.
- **Impact of Technology Failure:** AR systems can expose users to additional risks in the event of technology failure or denial of service. In critical applications, such as AR-assisted surgeries or AR-based remote machinery operation, technology failure could have severe consequences.
- **Diverse User Susceptibility:** Different user groups may have distinct vulnerabilities when interacting with AR technologies. Children, medical patients, and disabled individuals may not fully understand the implications of their interactions in the AR space, making them susceptible to manipulation and harm.
- **Unanticipated Societal Impact:** The advent of AR technologies can instigate societal changes that we may not fully anticipate at present. The impact of

these changes on user safety, privacy, and security needs to be considered and discussed.
- **Limitations due to Security Measures:** While security measures are essential for any technology, an overemphasis on these often leads to limiting some desirable functionalities of the AR systems. Understanding and striking a balance between security requirements and feature offerings is crucial for acceptance and adoption of these technologies.

1.3 Leveraging Artificial Intelligence and Machine Learning for Enhanced Security in Augmented Reality Systems

The potential of Artificial Intelligence (AI) and Machine Learning (ML) in enhancing the security of Augmented Reality (AR) systems is immense. As AR technologies continue to evolve and find widespread adoption across various sectors, ensuring the security of such systems against potentially harmful or distracting visual output produced by malicious or bug-ridden applications has become paramount. This is particularly important considering that AR users will not always use the technology in isolation, but also in ecosystems of other users, making the risks of AR largely interconnected and far-reaching.

Two features make side channel attacks in AR a critical problem. First, unlike smartphones where users can control when to turn on the sensors and applications, AR is continuously getting input from the environment through video, audio, and other sensors, acquiring both real and malicious input [21]. Second, current Software Development Kits (SDKs) do not have a library that focuses on protecting end users' privacy due to the side channel attack. A thorough understanding of the impact of side channel leakage can immensely benefit national cyber-security in the future.

AI and ML have shown great promise in addressing these challenges. For instance, deep learning has been used to detect routing attacks in the Internet of Things (IoT), a technology that is often integrated with AR for enhanced user experiences [25]. Furthermore, machine learning techniques have been applied to improve the security of AR applications by identifying and mitigating potential threats [26].

The potential of Artificial Intelligence (AI) and Machine Learning (ML) in enhancing the security of Augmented Reality (AR) systems is immense. As AR technologies continue to evolve and find widespread adoption across various sectors, ensuring the security of such systems against potentially harmful or distracting visual output produced by malicious or bug-ridden applications has become paramount. This is particularly important considering that AR users will not always use the technology in isolation, but also in ecosystems of other users, making the risks of AR largely interconnected and far-reaching.

In their paper, "Security and Privacy in Augmented Reality: Current Trends and Future Challenges," Lebeck et al. [27] discuss the potential security and privacy

challenges in AR applications. The authors highlight the importance of developing robust security frameworks to protect against potential threats. They argue that as AR technologies continue to advance and become more integrated into our daily lives, it is crucial to address the unique security challenges they present to ensure a safe and secure user experience.

Lukosch et al. [28] discuss how AR and machine learning can be used to enhance information security in their paper "Augmented Reality and Machine Learning for Improved Information Security." They propose an AR system that uses machine learning algorithms to detect and prevent security threats. This approach leverages the power of machine learning to analyze patterns and predict potential threats, thereby enhancing the security of AR applications.

In "Deep Reinforcement Learning for Secure Visual Output in Augmented Reality Systems," Ahn et al. [22] present a novel approach to securing visual output in AR systems. The authors propose the use of deep reinforcement learning to intelligently displace AR content and reduce obstruction of real-world objects. This approach utilizes a local fog computing node, which runs training simulations to automatically learn an appropriate policy for filtering potentially malicious or distracting content produced by an application.

Revetria et al. [29] discuss the integration of AR with the Internet of Things (IoT) for improving safety and security in industrial settings in their paper "Augmented Reality and Internet of Things for Improved Safety and Security in Industrial Plants." They propose the use of AR for real-time monitoring of mechanical structures, improving the safety of the working environment. This approach leverages the capabilities of AR and IoT to provide real-time feedback and situational awareness, thereby enhancing safety and security in industrial plants.

Lastly, the paper "Side Channel Attack in Augmented Reality: An Exploration" by Nasr et al. [30] discusses the potential of side channel attacks in AR systems. The authors propose a framework for studying these attacks and providing a library for AR SDK to protect end users' location-based privacy. This work highlights the importance of understanding and mitigating the potential risks associated with side channel attacks in AR systems.

In conclusion, the integration of AI and ML in AR security presents a promising approach to addressing the unique security challenges posed by AR technologies. As these technologies continue to evolve and become more integrated into our daily lives, it is crucial to leverage the power of AI and ML to ensure a safe and secure user experience.

2 Augmented Reality (AR) Security Threats

With AR becoming prevalent across various sectors, it is accompanied by an array of insecurities and challenges that pose potential threats to the integrity of user experience and data privacy.

Data Privacy Concerns Augmented Reality (AR) technologies, while providing immersive experiences and novel interactions, pose significant security risks. These threats span a wide range of categories, from privacy invasion to malware attacks, ransomware, and physical compromise of devices. A key concern with AR technologies is their extensive data collection capability, which raises serious privacy issues. If malicious actors gain access to an AR device, the potential loss of privacy is significant. It is crucial to question how AR companies use and secure the information they collect, where this data is stored, and whether they share it with third parties.

Intrusion of AR Sensor Technologies The reliability of AR content, often generated by third-party vendors, is not guaranteed. This uncertainty can be exploited by cyber attackers through methods such as spoofing, sniffing, and data manipulation, leading to the dissemination of false information. This unreliable content can also be used to deceive users in social engineering attacks, distorting users' perception of reality to their advantage.

Cyber Threats to AR AR applications can also be conduits for malware, embedded within advertising content. Unsuspecting users who interact with these ads could be led to malware-infected AR servers. Additionally, the risk of network credential theft from wearable devices presents another security concern. If such credentials are compromised, unauthorized access to sensitive information may occur. Denial-of-service attacks could disrupt AR services, causing severe consequences, especially in critical situations where the technology is essential. This risk is coupled with the possibility of man-in-the-middle attacks, where network attackers eavesdrop on the communications between the AR browser and the AR provider. Ransomware attacks pose another threat, with attackers potentially gaining access to a user's AR device, recording their behavior, and threatening to release these recordings unless a ransom is paid.

Physical Vulnerabilities in AR Devices Physical damage or theft of wearable AR devices is a significant security concern, as these devices can be easily lost or stolen. The use of cloud services by AR technologies introduces additional vulnerabilities, such as potential data interception and cloud server breaches. Therefore, it is critical for IT departments to adopt secure and reliable cloud practices, including access-monitoring and authentication tools. As an alternative, sensitive information can be localized within the facility, eliminating many potential vulnerabilities. Another approach could be the use of hardwired, projection-based AR platforms, which are less susceptible to hacking and data theft.

The cybersecurity environment surrounding AR is not a fixed entity, but rather a dynamic and ever-evolving landscape. As advancements in AR technologies persist, the threats they encounter concurrently evolve. This rapid technological progression necessitates an enduring vigilance and adaptability to safeguard against emergent threats and to ensure the secure utilization of AR. The fluid nature of AR cybersecurity is underscored by the unique forms of cyberattacks that specifically target AR systems. These attacks underscore the imperative for continuous innovation and

adaptation in AR cybersecurity strategies. In the following sections, I enumerate these unique attacks:

2.1 Fraud, Theft, and Disruption

Fraud, theft, and disruption are prevalent forms of cyberattacks that pose significant threats to the security of AR systems. Fraud in AR often involves deceptive practices designed to trick users into revealing sensitive information. For instance, a malicious AR application might mimic a legitimate one, tricking users into entering their login credentials or other personal information. Theft in the context of AR usually involves stealing data or resources. This could range from personal data collected by AR applications to proprietary AR content and technology. For example, a cybercriminal might exploit vulnerabilities in an AR application to access and steal user data. Disruption involves interrupting or degrading an AR service. This could be achieved through various means, such as launching a Distributed Denial of Service (DDoS) attack against an AR server or exploiting a software vulnerability to cause an AR application to crash.These threats highlight the importance of implementing robust security measures in AR systems. This includes secure design and coding practices, regular security testing, user education, and the use of advanced security technologies such as encryption and intrusion detection systems.

2.2 Invisible Eavesdropping

Invisible eavesdropping is a potential threat unique to the AR, where an attacker could invisibly listen in on other users inside a virtual room without their knowledge or consent. This form of cyberattack could lead to significant privacy breaches and misuse of personal information. Invisible eavesdropping could take various forms. For instance, an attacker could exploit software vulnerabilities or design flaws to gain unauthorized access to a virtual room or conversation. They could also use advanced techniques such as network traffic analysis or packet sniffing to intercept and decode the data transmitted between users. This threat is particularly concerning given the immersive and interactive nature of the AR. Users might engage in private conversations or share sensitive information under the assumption of privacy, not realizing that an attacker could be listening in. To mitigate this threat, it's crucial to implement robust security measures in the AR. This includes secure communication protocols, end-to-end encryption, and strong access controls. Additionally, users should be educated about the potential risks and encouraged to exercise caution when sharing sensitive information in the AR.

2.3 Manipulation into Physical Harm

Manipulation into physical harm is another potential threat unique to AR. This threat involves manipulating the AR environment in such a way that it could lead to physical harm to the user in the real world. For instance, an attacker could trick a user into walking into a physical object or off a ledge. In the context of AR, this type of attack could take various forms. For example, an attacker could create an AR object or path that leads a user to collide with a real-world object. Alternatively, they could manipulate the AR environment to make it appear as if a real-world hazard, such as a ledge or a stair, does not exist. This threat is particularly concerning given the immersive nature of AR. Users might fully engage with the AR environment, not realizing that it could be manipulated to cause real-world harm.

To mitigate this threat, it's crucial to implement safety measures in AR. This includes features that alert users to the presence of real-world hazards, safeguards that prevent the manipulation of the AR environment in dangerous ways, and user education about the potential risks. Additionally, users should be encouraged to maintain awareness of their real-world surroundings while using AR.

2.4 Human Joystick Attack in AR

The Human Joystick Attack is a unique form of threat identified by researchers at the University of New Haven. This attack involves controlling immersed users in an AR environment and moving them to a location in physical space without their knowledge. This could potentially lead to physical harm.

In a Human Joystick Attack, an attacker could manipulate the AR environment to control the user's movements. For instance, they could create an AR object or path that leads the user to move in a certain direction in the real world. Alternatively, they could manipulate the AR environment to make it appear as if the user is moving in a different direction than they actually are. This attack is particularly concerning given the immersive nature of AR. Users might fully engage with the AR environment, not realizing that their movements are being controlled by an attacker. This could lead to situations where the user unknowingly moves into a dangerous location or situation in the real world.

To mitigate this threat, it's crucial to implement safety measures in AR. This includes features that prevent the manipulation of the user's movements, safeguards that ensure the consistency and stability of the AR environment, and user education about the potential risks. Additionally, users should be encouraged to maintain awareness of their real-world surroundings while using AR.

2.5 Chaperone Attack in AR

A Chaperone Attack is a unique form of threat in AR that involves modifying the boundaries of a user's virtual environment. This could potentially lead to physical harm, as users could be tricked into moving into dangerous areas in the real world. In a Chaperone Attack, an attacker could manipulate the AR environment to alter the perceived boundaries of the virtual space. For instance, they could make the virtual space appear smaller or larger than it actually is, or they could create an illusion of a safe path that leads the user into a dangerous area in the real world.

This attack is particularly concerning given the immersive nature of AR. Users might fully engage with the AR environment, not realizing that the boundaries of their virtual space have been manipulated. This could lead to situations where the user unknowingly moves into a dangerous location or situation in the real world.

To mitigate this threat, it's crucial to implement safety measures in AR. This includes features that prevent the manipulation of the virtual boundaries, safeguards that ensure the consistency and stability of the AR environment, and user education about the potential risks. Additionally, users should be encouraged to maintain awareness of their real-world surroundings while using AR.

2.6 Overlay Attack

An Overlay Attack in the context of AR applications involves the unauthorized addition or modification of virtual objects within a user's view. This could manifest as displaying undesired or inappropriate content, altering existing virtual elements, or creating misleading virtual cues. The primary intention behind such an attack is to confuse, disorient, or deceive the user.

In the case of Ubiquity6, an augmented reality startup, the potential for Overlay Attacks is significant due to the app's feature of allowing users to add persistent virtual objects to real-world environments. These objects can be interacted with by other users, opening up the possibility for malicious alterations or additions.

Examples of Overlay Attacks could include digital vandalism, where a user's virtual creation is defaced by another user, or trolling and harassment, where users place obstructive or offensive virtual objects in front of others. These actions can disrupt the user experience, potentially causing distress or harm.

Overlay Attacks pose a unique challenge in the realm of augmented reality security. They exploit the interactive and persistent nature of these environments, making them difficult to prevent without imposing restrictions on user creativity and freedom. Therefore, addressing Overlay Attacks requires a careful balance between user safety and the open-ended interactivity that makes augmented reality engaging and immersive.

2.7 Disorientation Attack

A disorientation attack is a unique form of threat in AR that aims to confuse or disorient a user, potentially making them more susceptible to other forms of attack or manipulation. This type of attack leverages the immersive nature of AR to create a disorienting environment or situation that can cause confusion, nausea, or even physical discomfort for the user. In a disorientation attack, an attacker could manipulate the AR environment to create visually confusing or disorienting scenarios. For instance, they could rapidly change the user's virtual location, alter the orientation or scale of virtual objects, or create visually conflicting cues that can lead to a sense of disorientation.

This disorientation can have several consequences. First, it can cause physical discomfort or sickness for the user, including symptoms such as nausea, dizziness, and balance issues. Second, it can make the user more vulnerable to other forms of attack. For example, a disoriented user might be more likely to fall for a phishing attack or inadvertently reveal sensitive information.

To mitigate this threat, it's crucial to implement safety measures in AR. This includes features that prevent rapid or disorienting changes in the AR environment, safeguards that ensure the consistency and stability of virtual objects, and user education about the potential risks. Additionally, users should be encouraged to take regular breaks when using AR to prevent disorientation and related symptoms.

2.8 Man in the Room Attack in AR

The Man in the Room Attack is a form of eavesdropping attack unique to AR where an attacker can invisibly observe and listen to other users in a virtual room without their knowledge or consent.

In a Man in the Room Attack, an attacker could exploit vulnerabilities in the AR system to gain unauthorized access to a virtual room or conversation. They could invisibly join the AR session and observe or listen to other users without their knowledge. This could lead to significant privacy breaches and misuse of personal information.

This attack is particularly concerning given the immersive and interactive nature of AR. Users might engage in private conversations or share sensitive information under the assumption of privacy, not realizing that an attacker could be invisibly present in the room.

To mitigate this threat, it's crucial to implement robust security measures in AR. This includes secure communication protocols, end-to-end encryption, and strong access controls. Additionally, users should be educated about the potential risks and encouraged to exercise caution when sharing sensitive information in AR.

3 AI and ML in Enhancing AR Security

Augmented Reality (AR) has emerged as a transformative technology, reshaping various sectors from education to industry. However, the rapid advancement and adoption of AR technologies have brought forth significant security challenges. The continuous interaction of AR with the environment and the user's field of vision via video, audio, and other sensors raises significant security, privacy, and safety concerns [27].

Artificial Intelligence (AI) and Machine Learning (ML) have shown great potential in addressing these security challenges. For instance, deep reinforcement learning has been proposed to secure visual output in AR systems. These systems intelligently displace AR content to reduce obstruction of real-world objects while maintaining a favorable user experience [22].

Moreover, the integration of AR with other emerging technologies such as the Internet of Things (IoT) offers the possibility of implementing innovative systems related to industrial safety and security. For example, AR can be used for real-time monitoring of mechanical structures, improving the safety of the working environment [29].

However, AR systems are not immune to attacks. Side channel attacks, where malicious users analyze and match patterns to get desired information, pose a significant threat to AR systems. To combat this, researchers have proposed a framework for studying these attacks and providing a library for AR SDK to protect end users' location-based privacy [30].

AI and ML can play a crucial role in detecting and preventing such attacks. For instance, machine learning algorithms can be used to detect anomalies in AR systems, providing an additional layer of security [31]. Furthermore, AI can be used to analyze patterns and predict potential threats, allowing for proactive security measures.

In conclusion, AI and ML hold great promise in enhancing the security of AR applications. As AR technologies continue to advance and become more integrated into our daily lives, it is crucial to leverage the power of AI and ML to address the unique security challenges they present.

3.1 AI for Anomaly Detection in AR Systems

With the complexity of AR systems, the detection of anomalies becomes a crucial aspect to ensure the smooth operation and user experience. Artificial Intelligence (AI), with its ability to learn from data and make predictions, offers promising solutions for anomaly detection in AR systems.

Anomaly detection refers to the identification of items, events, or observations that deviate from an expected pattern in a dataset. These anomalies often translate to critical and actionable information in a system. In the context of AR systems,

anomalies could range from unexpected user behavior, system performance issues, to security threats.

AI, particularly Machine Learning (ML) and Deep Learning (DL), has shown great potential in anomaly detection. These technologies can learn from vast amounts of data, identify patterns, and make predictions, making them well-suited for detecting anomalies that might be too complex for traditional methods to identify.

For instance, a study by Smedsrud et al. [32] demonstrated the use of AI in detecting anomalies in video capsule endoscopy (VCE) data, a form of AR system used in healthcare. The researchers developed an AI-based system that could classify VCE data and detect anomalies such as erosions and erythema, which are often difficult to differentiate from normal mucosa. The system was able to achieve accurate predictions, demonstrating the potential of AI-based analysis in AR systems.

The use of AI in anomaly detection in AR systems can be further enhanced with semi-supervised and unsupervised machine learning methods. These methods can learn from both labeled and unlabeled data, making them more flexible and capable of handling real-world data. For example, self-learning and neural graph learning are techniques that use unlabeled data in addition to a small amount of labeled data to extract additional information. In areas with scarce data, these new algorithms might be the technology needed to make AI truly useful for AR systems.

However, it is important to note that the implementation of AI-based anomaly detection in AR systems is not without challenges. One of the key considerations is how the dataset is split into training and test sets. This is crucial to avoid having related frames in several sets, which can give an unfair effect on the results. Therefore, the splits should be completely different, probably even at the level of patients.

In conclusion, AI offers promising solutions for anomaly detection in AR systems. With its ability to learn from data and make predictions, AI can detect anomalies that might be too complex for traditional methods to identify. However, the implementation of AI in AR systems requires careful consideration of factors such as data splitting and the use of semi-supervised and unsupervised learning methods. As research in this area continues to advance, we can expect to see more sophisticated AI-based anomaly detection systems in AR, enhancing the user experience and system performance

4 Case Study Analysis

In this section, we explore the threat model of AR systems and demonstrate several new categories of threats caused by the unique combination of continuous GPS sensing, high-volume visual data capturing, and image processing. Based on the type of the attack, we divide our research agenda into two thrusts: (1) defending against the AR attacks under low speed and (2) defending against AR attack under

high-speed scenario. For each research thrust, we will study how to develop a trustworthy computing and communication framework to identify, analyze, and mitigate the attack and its border impact. The outcome of our project, however, is a uniformed system where all the frameworks will be integrated into it. The system will proactively defend against both physical and cyber attacks in the AR system.

4.1 Case Study 1: Defending Against AR Attack in Mobile Scenario

In the context of mitigating AR threats within mobile environments, a series of strategic measures can be implemented.

Initially, an active caching strategy can be employed, wherein AR content is retained on the server. As per the referenced study, the system can mitigate privacy leakage by confining the cache to respond solely to AR user deployment and prohibiting the deployment of location-based AR content. This effectively impedes an attacker's attempts to infer locations based on data size.

Subsequently, the regulation of network traffic monitoring can be intensified, thereby complicating the task for potential attackers seeking to analyze and exploit the system. As demonstrated in the referenced paper [33], implementations such as restricting access permissions to system API, mandating encrypted communications, and adopting irregular data transmission in AR applications can render it challenging for attackers to monitor network traffic and deduce a victim's location.

Nevertheless, the study underscores the necessity to formulate more robust and comprehensive defenses, predicated on large-scale and diverse experimentation, as attackers can still discern the relationship between AR content size and traffic patterns.

The paper concludes by highlighting the inherent vulnerabilities of current location-based AR applications, which pose a significant risk to users' geolocation privacy. It calls upon developers to reevaluate their geolocation transmission protocol and comprehend the serious implications of privacy leaks in AR applications. The study also underscores the urgency of devising strategies to protect against side-channel attacks and safeguard AR users in mobile environments.

4.2 Case Study 2: Understanding and Mitigating Perceptual Manipulation Attacks

In the rapidly advancing field of Augmented Reality (AR), Perceptual Manipulation Attacks (PMAs) pose a growing threat. This was extensively investigated in a landmark study conducted at the University of Washington [34]. PMAs exploit the seamless integration of virtual and physical realities in AR environments to subtly alter users' perceptions, thereby influencing their decisions and actions.

The study focused on three primary channels through which PMAs operate: visual, auditory, and situational awareness. A series of detailed experiments were conducted involving 21 participants, with the aim to measure user reactions to PMAs, evaluate the influence of such attacks on user behavior, and understand the instinctive self-defense mechanisms individuals employ when faced with these deceptive techniques.

The findings revealed that PMAs significantly influenced the participants' responses. Visual attacks were particularly effective, deceiving participants through false color overlays on target objects, leading to confusion in task comprehension. Auditory manipulations demonstrated how distracting sounds could adversely affect user performance during tasks requiring high concentration. In terms of situational awareness attacks, participants were deceived by a conspicuous image that prevented them from noticing changes in their real environment.

The study observed that reaction times increased during an attack and, interestingly, remained elevated even in non-attack scenarios. This suggested the potential for PMAs to have lasting effects. The study also discovered various adaptive strategies that participants instinctively initiated to counteract the attacks. However, these attempts often failed to neutralize the effects of the attacks, further highlighting the deceptive nature of PMAs.

In response to the significant implications of the study for future AR applications, the researchers proposed several countermeasures to secure the AR environment. These included a contextual focus mode, an "Escape to reality" option, defenses centered around human cognition, and fostering resilience against attacks. The goal is to reduce susceptibility to PMAs.

As AR technology becomes increasingly prevalent across various sectors such as healthcare, education, entertainment, and commerce, it is crucial to develop effective countermeasures against PMAs. Therefore, further research in this domain is essential. The study strongly recommended a comprehensive assessment of the real-world impacts of PMAs, tracking the evolution of these attacks, and understanding the implications of PMAs in settings with multiple users.

In conclusion, identifying and preventing PMAs is of paramount importance as the AR realm continues to expand. The work by the University of Washington team is groundbreaking, paving the way for further exploration of safety measures in AR, and provides invaluable insights to fuel more research and development in this area.

5 Case Study 3: Secure and Private Sharing Mechanisms for Multi-User AR System

In [35] the authors focuses on the development of secure and private sharing mechanisms for multi-user Augmented Reality (AR) applications. It primarily discusses the design, implementation, and performance analysis of a prototype named ShareAR, which aims to enable multi-user AR applications that respect users'

security and privacy rights within shared physical spaces. The authors highlight that AR, with its integration into the physical world, offers unique challenges for content sharing such as understanding the dynamics of shared virtual and physical spaces. The sharer's physical location and the method for sharing AR content—either opt-in or opt-out—are considered as crucial factors in their experiment. To demonstrate the efficacy of ShareAR, different scenarios involving multiple users interacting with shared AR objects were tested through three AR case study apps (Paintball, Doc Edit and Cubist Art). ShareAR was found successful in protecting owned physical spaces and giving users control both over AR content sharing from their perspective and the inbound sharing requests. It demonstrated that the system can support a breadth of sharing control functionalities with limited developer effort and can scale with increasing numbers of users and objects while maintaining reasonable operational efficiency and performance. In conclusion, the, authors affirm the importance of addressing the challenges of securing and privacy in multi-user AR ecosystems even as the technology continues to evolve. They propose further exploration and future work in this area to provide better solutions for user control, permissions, outbound and inbound content sharing, and mechanics of sharing.

6 Challenges and Future Prospects

6.1 Potential Risks of AI and Machine Learning in AR Security

While the implementation of AI and Machine Learning techniques in AR applications greatly enhance security measures, it's also critical to acknowledge the potential risks they can bring.

Privacy Invasion AI models, particularly those involved in machine learning, typically require significant data to train accurately. Often, this data includes sensitive and private information about users. Ensuring that this data is not abused or misused is a real challenge for developers and researchers.

Dependency on Technology With AI and machine learning systems, there is also a risk of becoming overly dependent on the technology. Errors in the AI system can lead to serious breaches or compromises in the AR application.

Mitigation of Risks The first step towards mitigating these risks involves implementing strict policies and regulations around data handling. Encryption of data and strong access controls should be applied. Secondly, designers of AR systems should consider the possibility of AI failures and prepare backup plans. Lastly, promoting transparency in AI models and decisions could help keep everyone informed and aware, resulting in early threat detection and mitigation. It is incumbent upon research and development professionals to balance the conveniences of AI and machine learning with the potential risks they pose, ensuring a secure and ethical use of these modern technologies in AR applications.

6.1.1 Exploiting AI-Generated Video Manipulations for AR Device Location Tracking

We would like to discuss an innovative methodology that harnesses the power of generative artificial intelligence models for video content manipulation. This approach, while groundbreaking, could potentially be leveraged for tracking user locations, thus raising important considerations for privacy and security.

The proposed methodology involves the use of a generative AI model to subtly alter a video by inserting additional frames that are similar to the existing ones. These AI-generated frames are designed to have slight variations in brightness levels, which are not easily perceptible to the human eye but can be detected by AR devices. When an AR device processes this manipulated video through its camera, the brightness variations in the AI-generated frames cause the device to generate a unique encoding pattern. This encoding process, facilitated by a widely used video compression standard such as H.264, results in a distinctive network throughput pattern as the encoded data is transmitted.

In a scenario where a malicious actor has access to the user's AR device network traffic, a privilege that is not difficult to obtain in many cases, the unique network throughput patterns caused by the AI-manipulated video can be analyzed. Utilizing a network traffic analysis tool like MidJourney or Stable Diffusion, the attacker could potentially infer the user's location. The implementation of this methodology involves several key steps. Firstly, a generative AI model, such as a Generative Adversarial Network (GAN), is developed to create frames with subtle brightness variations. This model is integrated with a video processing pipeline that can insert the AI-generated frames into the video at desired intervals.

On the AR device side, the video encoding process, using H.264 or a similar codec, is sensitive to the brightness variations in the AI-generated frames. The network traffic of the AR device is monitored using a tool like MidJourney to identify the unique throughput patterns caused by the AI-manipulated video. Finally, an algorithm is developed that can correlate these throughput patterns with the user's location. This could involve machine learning techniques and would likely require a training dataset of network throughput patterns associated with different locations.

This proposed methodology underscores the potential security risks associated with the integration of AI and AR technologies. It emphasizes the importance of robust security measures to protect user data and privacy in the evolving landscape of AR applications. However, it is crucial to consider the ethical and privacy implications of this approach, as it involves tracking a user's location.

6.2 Emerging Trends and Future Prospects

6.2.1 AI for Prevention of Malicious AR Content

The potential for malicious content in AR system presents a significant threat. To counteract this, we can build a proactive AI-based framework for the detection and

prevention of such content. This framework primarily utilizes two AI models: Generative Adversarial Networks (GANs) and Convolutional Neural Networks (CNNs). The GAN is employed to generate a wide array of potential malicious AR content, forming a comprehensive training dataset. This dataset encompasses various forms of malicious content, ranging from inappropriate imagery to sophisticated security exploits. The CNN, renowned for its proficiency in image analysis, is then trained on this dataset. Through this training, the CNN learns to discern the characteristics indicative of malicious AR content, thereby enabling it to detect similar content within real-world AR applications. Upon training, the CNN is integrated into AR applications as a content filter. As new AR content is introduced, the CNN scrutinizes it for signs of malicious intent. Content deemed potentially harmful can be flagged for review or automatically blocked, contingent on the threat level.

This AI-based approach provides a robust and scalable solution to the threat of malicious AR content. However, it necessitates careful consideration of ethical implications, particularly regarding user privacy and content censorship. Future work should aim to strike a balance between security and user freedom in AR applications.

6.2.2 AI-Based User Authentication Methods in AR

As the proliferation of Augmented Reality (AR) applications continues unabated, the necessity for robust user authentication methods escalates in tandem. Consequently, it is incumbent upon us to devise and implement an Artificial Intelligence (AI)-based framework for user authentication in AR, harnessing the power of biometric data and machine learning techniques.

Biometric authentication, capitalizing on unique biological attributes such as facial characteristics or vocal patterns, presents a promising avenue for user authentication within AR. However, conventional biometric systems may be susceptible to spoofing attacks, wherein an attacker endeavors to replicate the user's biometric data. To counteract this, we advocate for an AI-based authentication system that employs machine learning to bolster the security of biometric authentication. Future AR systems could utilize a Convolutional Neural Network (CNN), trained on a dataset of biometric data, thereby learning to discern the unique attributes of each user's biometric features. This knowledge can then be applied to authenticate users in real-time, juxtaposing the input biometric data with the learned features.

In addition, the system integrates liveness detection, a technique used to ascertain whether the biometric data originates from a live individual as opposed to a recorded or fabricated sample. This can be accomplished using a range of methods, such as scrutinizing the texture of the skin in a facial recognition system or detecting the natural fluctuations in a vocal pattern. By amalgamating biometric authentication with machine learning and liveness detection, this AI-based system offers a robust and secure method for user authentication in AR applications.

However, it is crucial to consider the privacy implications of this approach, as it involves the collection and processing of sensitive biometric data. Future research should concentrate on ensuring the privacy and security of user data in such systems.

7 Conclusion

This compelling exploration in the realms of AR and its associated security challenges has unveiled both the vast opportunities and formidable threats synonymous with this technology. By infusing digital data into our physical environment, AR has transformative implications across diverse sectors, such as education, entertainment, and healthcare. However, with novel technology surfaces novel threats—data privacy, intrusive sensor invasion, and persistent cyberattacks among the most pressing.

AI and ML, with their inherent capabilities of learning and improving upon experience, emerge as potential game-changers in the security landscape of AR. Their application in anomaly detection, predicting threats, and offering novel solutions underlines the immense potential they hold. Despite this, the implementation and application of AI/ML in AR security warrant careful consideration, a careful balance to be struck between vital security measures and ensuring user freedom.

Furthermore, while AI/ML techniques could greatly enhance AR application security, we must also consider their techniques' potential risks to user privacy. A nuanced, balanced approach is warranted, respecting the importance of user privacy while utilizing these advanced technologies to their full potential for ensuring the security of AR applications.

As we venture deeper into the digital age, AI and ML may be the torchbearers that guide the path to a safer, more secure AR experience and ensure a promising future for AR application development. This chapter serves as a stepping stone into this exploration and underscores the need for continuous research in this field.

Thus, at the cusp of a digital revolution, we bear witness to an intriguing interplay of advanced technology and security requirements. As AR adoption accelerates, the amplification of AI/ML in ensuring secure and private AR experience will be critical to watch.

References

1. Nee AYC, Ong SK (2023) Springer handbook of augmented reality. Springer Nature.
2. Azuma R (1997) A survey of augmented reality. Presence: Teleoperators Virtual Environ 6(4):355–385
3. Irwansyah FS, Yusuf Y, Farida I, Ramdhani M (2018) Augmented reality (AR) technology on the android operating system in chemistry learning. IOP Conf Series Mater Sci Eng 288(1):012068

4. Radu I, Schneider B (2019) What can we learn from augmented reality (AR)? Benefits and drawbacks of AR for inquiry-based learning of physics. In: Proceedings of the 2019 CHI conference on human factors in computing systems (pp. 1–13).
5. Nayyar A, Mahapatra B, Le DN, Suseendran G (2018) Virtual reality (VR) & augmented reality (AR) technologies for tourism and hospitality industry. Int J Eng Technol 7(2.21):156–160
6. Mourtzis D, Siatras V, Angelopoulos J (2020) Real-time remote maintenance support based on augmented reality (AR). Appl Sci 10(5):1855
7. Caudell T, Mizell D (1992) Augmented reality: an application of heads-up display technology to manual manufacturing processes. In Proceedings of the twenty-fifth hawaii international conference on system sciences, vol 2, pp 659–669
8. Rosenberg LB (1992) The use of virtual fixtures as perceptual overlays to enhance operator performance in remote environments. Technical Report (1992 DTIC Document)
9. Rosenberg LB (1993) Virtual fixtures: Perceptual tools for telerobotic manipulation. In: Virtual Reality Annual International Symposium. IEEE.
10. Kato H, Billinghurst M (1999) Marker tracking and HMD calibration for a video-based augmented reality conferencing system. In: Proceedings of the 2nd IEEE and ACM international workshop on augmented reality (IWAR 99)
11. Abate AF, Acampora G, Ricciardi S (2011) An interactive virtual guide for the AR based visit of archaeological sites. J Vis Lang Comput 22(6):415–425
12. Abate AF, Acampora G, Loia V, Ricciardi S, Vasilakos AV (2010) A pervasive visual–haptic framework for virtual delivery training. IEEE Trans Inf Technol Biomed 14(2):326–334
13. Damiani M, Bertino E, Perlasca P (2007) Data security in location-aware applications: an approach based on RBAC. Int J Inf Comput Secur 1(1–2):5–38
14. Klein G, Murray D (2007) Parallel tracking and mapping for small AR workspaces. In: 2007 6th IEEE and ACM international symposium on mixed and augmented reality. IEEE, pp. 225–234
15. Hincapié M, Caponio A, Rios H, Mendivil EG (2011) An experimental study of educational paradigms in augmented reality applications. In: 2011 14th symposium on virtual and augmented reality. IEEE, pp. 39–47
16. Billinghurst M, Clark A, Lee G (2014) A survey of augmented reality. Found. Trends Hum-Comput Interact 8(2–3):73–272
17. Olsson T, Lagerstam E, Kärkkäinen T, Väänänen-Vainio-Mattila K (2013) Expected user experience of mobile augmented reality services: a user study in the context of shopping centres. Personal Ubiquitous Comput 17(2):287–304
18. Lebeck K, Schumann KRB, Fogarty J, Roesner F (2018) World-driven access control for continuous sensing. In: Proceedings of the 2018 ACM SIGSAC conference on computer and communications security, pp 882–897
19. Lukosch S, Lukosch H, Datcu M, Cidota A (2015). Seeing is believing: improving the reliability of photo-realistic mixed-reality overlays. In: 2015 IEEE international symposium on mixed and augmented reality, pp 1–6
20. Lebeck K, Ruth K, Kohno T, Roesner F (2018). Towards security and privacy for multi-user augmented reality: foundations with end users. In 2018 IEEE symposium on security and privacy (SP). IEEE, pp. 770–786
21. Denning T, Dehlawi Z, Kohno T (2014) In situ with bystanders of augmented reality glasses: perspectives on recording and privacy-mediating technologies. In: Proceedings of the SIGCHI conference on human factors in computing systems, pp 2377–2386
22. Ahn S, Cho Y, Lee S (2018). Deep reinforcement learning for secure visual output in augmented reality systems. In: Proceedings of the 2018 ACM on international conference on multimedia retrieval, Yokohama, Japan, pp 335–343
23. Revetria R, Schirru M, Testa A, Paddeu F (2019) Innovative systems for safety and security based on augmented reality and IoT. In: 2019 IEEE international conference on engineering, technology and innovation (ICE/ITMC), pp 1–8
24. Xiang L, Xu X, Yin Z, Liu B, Tian J, Wang D (2019) Ground subsidence monitoring of a rock salt mine using the SBAS-InSAR technique with Sentinel-1A imagery. Sensors 19(24):5511

25. Seeber KR, Li Y, Zoubir AM (2018) Deep learning for detection of routing attacks in the internet of things. IEEE Access 6:77096–77107
26. Ferrag MA, Maglaras L, Argyriou A, Kosmanos D, Janicke H (2018) Security for 4G and 5G cellular networks: a survey of existing authentication and privacy-preserving schemes. J Netw Comput Appl 101:55–82
27. Lebeck K, Kohno T, Roesner F (2018) Security and privacy in augmented reality: current trends and future challenges. IEEE Secur Privacy 16(5):34–42
28. Lukosch S, Lukosch H, Klomp M, Verbraeck A (2015) Augmented reality and machine learning for improved information security. In: Proceedings of the 2015 ACM SIGSIM conference on principles of advanced discrete simulation, London, United Kingdom, pp 255–262
29. Revetria R, Schiavone M, Testa F (2019) Augmented reality and internet of things for improved safety and security in industrial plants. In: Proceedings of the 2019 international conference on industrial engineering and systems management (IESM), Shanghai, China, pp 1–6
30. Nasr M, Shabtai A, Elovici Y (2017) Side channel attack in augmented reality: an exploration. In: Proceedings of the 2017 ACM on Asia conference on computer and communications security, Abu Dhabi, United Arab Emirates, pp 866–879
31. Zhang Y, Wen J (2019) The IoT electric business model: using blockchain technology for the internet of things. Peer-to-Peer Netw Appl 10(4):983–994
32. Smedsrud PH, Langørgen I, Halvorsen P, Balasingham I, de Lange T (2021) Anomaly detection in capsule endoscopy. Sci Data 8(1):1–11
33. Shang J, Chen S, Wu J, Yin S (2022) ARSpy: breaking location-based multi-player augmented reality application for user location tracking. IEEE Trans Mob Comput 21(2):433–447
34. Cheng K, Tian JF, Kohno T, Roesner F (2023) Exploring user reactions and mental models towards perceptual manipulation attacks in mixed reality. In: 32nd USENIX security symposium (USENIX Security 23), Anaheim, CA, August 2023 [Online]
35. Rajaram S, Roesner F, Nebeling M (2021) Designing privacy-informed sharing techniques for multi-user AR experiences. In: VR4Sec: 1st international workshop on security for XR and XR for security

On the Robustness of Image-Based Malware Detection Against Adversarial Attacks

Yassine Mekdad, Faraz Naseem, Ahmet Aris, Harun Oz, Abbas Acar, Leonardo Babun, Selcuk Uluagac, Güliz Seray Tuncay, and Nasir Ghani

1 Introduction

In the past decades, Machine Learning (ML) and Deep Learning (DL) models have been the *de-facto* solution for several domains (e.g., speech recognition, natural language processing, computer vision, computer and network security, data mining, etc.) due to their ability to automatically generalize (i.e., classify or cluster) to both known and unknown input samples [42, 45]. In fact, one of the main applications of ML in computer and network security has been the detection of *malicious software* (malware) [5, 36, 43, 44, 58]. Currently, malware detection models run in cloud environments in order to classify unknown samples [63]. Such models can achieve outstanding performance over traditional methods (e.g., signature-based method or heuristic-based method) [4]. However, recent research has shown that the performance of these models can drop drastically via adversarially-crafted/perturbed inputs [7, 19, 48, 57]. On the other hand, other studies have shown the effectiveness of adversarial samples against other ML models without prior knowledge of the properties of the target classifier (e.g., features, classification algorithm, hyperparameters). This property is known as transferability and makes these models ill-suited for security-oriented applications [40, 47, 57].

Y. Mekdad · F. Nasseem · A. Aris · H. Oz · A. Acar · L. Babun · S. Uluagac (✉)
Cyber-Physical Systems Security Lab (CSL), Florida International University, Miami, FL, USA
e-mail: ymekdad@fiu.edu; fnase001@fiu.edu; aaris@fiu.edu; hoz001@fiu.edu; aacar001@fiu.edu; lbabu002@fiu.edu; suluagac@fiu.edu

G. S. Tuncay
Google, Mountain View, CA, USA
e-mail: gulizseray@google.com

N. Ghani
University of South Florida, Tampa, FL, USA
e-mail: nghani@usf.edu

Y. Chen et al. (eds.), *Network Security Empowered by Artificial Intelligence*,
Advances in Information Security 107, https://doi.org/10.1007/978-3-031-53510-9_13

Moreover, the effectiveness of adversarial attacks is highly related to their domain-specific constraints. For instance, in the computer vision domain, the adversarial manipulations to the samples should be imperceptible to the human eye [20], inaudible to the human ear in the audio domain [8], and preserve its semantics in the text domain [15], while still resulting in the sample evading the target classifiers. In the malware detection domain, adversarial ML attacks to ML-based malware detectors involve adding carefully crafted perturbations to the malware samples that preserve the malicious functionality of the malware while allowing the samples to evade the target classifier (i.e., modified malware samples are classified as benign). In this context, prior studies were able to craft adversarial malware samples that successfully evaded ML-based malware detection systems, including Windows Portable Executable (PE)-based malware detectors [13, 28, 29], Android malware detectors [12, 62], PDF-malware classifiers [55, 61] and even cloud-based proprietary anti-virus engines (e.g., Kaspersky, Eset, Sophos) [9]. These examples demonstrate the possibility of evading the state-of-the-art ML-based malware classifiers not by complex concealment techniques (e.g., polymorphism, metamorphism, encryption, packing), but by simple adversarial perturbations carefully crafted via adversarial attacks. To that end, other studies proposed defense mechanisms that prevent adversarial manipulations such as adversarial training [41, 57] and defensive distillation [46]. Nevertheless, these defense mechanisms are computationally costly and suffer from model poisoning and decreased detection accuracy [13]. Therefore, defending ML-based malware detection models against adversarial ML attacks is still an open research problem. In recent years, image-based malware detection has been an active field of research in malware analysis. Starting with the work of Nataraj et al. [37], several studies, including but not limited to [3, 17, 22, 26, 38], have utilized the gray-scale image representation of malware binaries to efficiently and timely classifying malware according to their corresponding family.

In this chapter, we aim to assess the robustness of image-based malware detection against adversarial attacks. To that end, we design and construct a lightweight CNN image-based malware detection model to detect Windows PE malware, based on the family it belongs to. It is worth mentioning that adversarial attacks, which are relatively easy to apply to images in the computer vision domain, are extremely difficult to apply to transformed images of malware samples. This difficulty can be explained because an operation that adds carefully crafted adversarial noise to a malware image has a very high possibility of breaking the functionality of the sample when the image is converted back to a malware binary. Although recent works have performed adversarial attacks against image-based malware classifiers [27, 32, 49, 60], most of these attacks fail to preserve the functionality of the adversarial malware sample. To evaluate the robustness of image-based malware classifier against adversarial attacks, we select adversarial attacks that preserve the functionality of the malware sample with a comparison with the state-of-the-art ML-based malware classifier MalConv [52]. In addition, we perform four adversarial attacks under white box and black box settings that preserve the malware functionality after modifications.

For our evaluation, we used the MalwareDatabase datasets [33] and the DikeDataset [35] consisting of different families including Generic, Trojan, Ransomware, Worm, Backdoor, Spyware, Rootkit, Encrypter, and Downloader in the Windows Portable Executable (PE) format. First, we trained our CNN classifier using the gray-scale images of the malware. Then, we performed four adversarial attacks against our classifier as well as MalConv. Our evaluation shows that the image-based malware detection approach is more robust against these attacks than MalConv. Interestingly, the evasion rate of some adversarial attacks dropped to 5% in certain cases. Our extensive analysis also shows the robustness and efficiency of our classifier against most of the adversarial attacks that preserve the functionality of the malware.

Contributions The main contributions of this chapter are as follows:

- We design and construct a lightweight CNN image-based malware classifier with high detection accuracy and low implementation overhead.
- We perform four adversarial attacks against ML-based malware classifiers that can evade the state-of-the-art ML-based malware detectors such as MalConv while preserving the functionality of the modified malware.
- We evaluate the performance of our classifier and assess its robustness against adversarial attacks in comparison to MalConv.

Organization The remainder of our chapter is organized as follows: In Sect. 2, we provide the related work. Section 3 briefly provides background information on Portable Executable File Format, visualization techniques, and adversarial attacks. Section 4 defines the scope of the problem and the threat model considered in this chapter. In Sect. 5, we describe our image-based malware classifier including the network architecture. Section 6 describes the adversarial ML attacks used in our study. Section 7 discusses the performance evaluation and outlines our experimental results. Section 8 provides a discussion, summarizing key points and benefits. Finally, Sect. 9 concludes our chapter.

2 Related Work

This section briefly reviews related work on image-based malware detection and their corresponding adversarial attacks.

Image-Based Malware Detection In [37], the authors proposed a new technique for malware classification that converts malware binaries into gray-scale images and determines the similarity of malware samples that belong to the same family. According to these results, the authors extracted the features of gray-scale malware images and used k-nearest neighbor (k-NN) for classification, enabling them to

achieve 98% classification accuracy. Since then, several studies employed malware images and visualization techniques for malware classification [3, 17, 22, 26, 38].

Adversarial Attacks Against Image-Based Malware Classifiers Recently, adversarial attacks against image-based malware classifiers started to be a focus of research. A crucial factor that must be considered when generating adversarial malware to evade image-based classifiers, is creating an adversarial sample whose image representation can evade the classifier while retaining its malicious functionality. The existing literature covers several adversarial attacks that aim to evade image-based malware classifiers. However, these attacks do not preserve the malicious functionality of the malware. Park et al. [49] performed Fast Gradient Sign Method (FGSM) and Carlini & Wagner (C&W) attacks to generate an adversarial image of a malware sample. Afterward, they utilized their algorithm to insert semantic no-operation (NOP) instructions into the original malware sample, making it appear as an adversarial sample. Although adding NOP instructions does not change the actual logic of a binary, adding instructions changes the section size and addresses, and therefore breaks the executable. Liu et al. [32] converted the malware binaries to images and then generated an adversarial image using the FGSM attack which can evade the image-based classifier. However, the resulting file may have a series of unmeaningful character sequences which can break its functionality. In the work of Vi et al. [60], a malware binary is converted into an image, and then the resource section of the image is determined and perturbed via FGSM attack to evade the classifier. Then, the perturbed pixels of the resource section are converted back to binary and used to modify the original malware's resource section. However, this approach might cause the Windows PE loader to fail to load the malware since this section has to follow a specific structure for successful parsing [24]. Khormali et al. [27] proposed COPYCAT which uses an adversarial example padding and sample injection attack. The adversarial padding technique generates an adversarial image of a sample using well-known attacks (e.g., FGSM, C&W, etc.). Then, it converts the image to bytes and appends the generated bytes to the end of the original malware, essentially doubling the size of the sample. The sample injection attack injects targeted class samples after the malware's exit code, which has a high probability of breaking the malware PE due to changing the offsets of the sections after the code section of the malware.

Differences from Existing Work Despite the prevalence of adversarial attacks targeting image-based malware classifiers, most of the proposed attacks fail to preserve the malware's functionality. Additionally, the defense mechanisms employed to counter these attacks often have high computational costs and might be vulnerable to model poisoning with reduced detection accuracy. Different from the prior work, our study analyzes the robustness of image-based malware classifiers against adversarial attacks. Our analysis shows that image-based classifiers are more robust against adversarial attacks that preserve the functionality of the malware in comparison to MalConv. For this reason, employing an image-based malware classifier does not require adversarial training; hence, it remains immune to some extent to model poisoning.

3 Background

In this section, we provide background information regarding the structure of Windows PE (Portable Executable) files, followed by a description of the process to represent a malware binary in the form of a gray-scale image. Then, we explain typical adversarial attacks against PE-based malware classifiers.

3.1 Portable Executable (PE) File Format

Portable Executable (PE) is the format used to create executable files, Dynamic Link Libraries (DLLs), and common object files in 32-bit and 64-bit Windows operating systems (OS) [18, 50]. It contains the necessary information needed by the OS for managing the executable file and provides an architecture-independent, and thus portable description. Each PE file consists of a PE header and various sections which are used by the linker in the loading process. The PE header possesses section, symbol, and optional header information. Note that there can be several sections in a PE file, but the sections that are common in the majority of PE files are described as follows:

- *.text* : encloses the program's main code,
- *.rdata* : includes the read-only initialized data (e.g., strings, constants, etc.),
- *.data* : contains the initialized data,
- *.rsrc* : holds the resources utilized by the program, such as icons and images.

In addition to these sections, there are sections containing imported and exported symbols (i.e., .idata and .edata), uninitialized data (.bss), and thread-local storage (.tls) [50].

3.2 Visualization of Portable Executable Malware Files

The vast majority of malware on the Internet has the structure of Windows Portable Executable (PE) files and nearly 64% of malware detected by Symantec were in PE format [25]. In this study, we selected PE-based malware families for evaluation purposes. A malware binary can be represented as a sequence of zeros and ones. Further, it is possible for this vector of binary values to be modified and transformed into an image [37]. Specifically, to enable such a conversion, the malware binary is represented as a vector of 8-bit unsigned integers (uint8) and then shaped into a two-dimensional array. The array is then divided by 255 to represent the array as a gray-scale image where the pixels take a value in the range of 0–255 (0 being black, and 255 being white). In Fig. 1, we use this technique to illustrate an example of a malware binary from the Ramnit family being converted to an image.

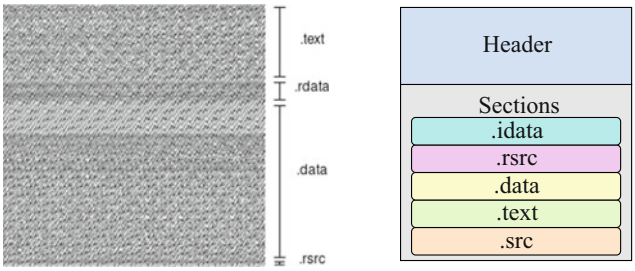


Fig. 1 An image depicting a malware binary from the Ramnit family of malware (left) and PE file structure (right). Each section of the image is labeled corresponding to the respective section of the PE file excluding the PE Header

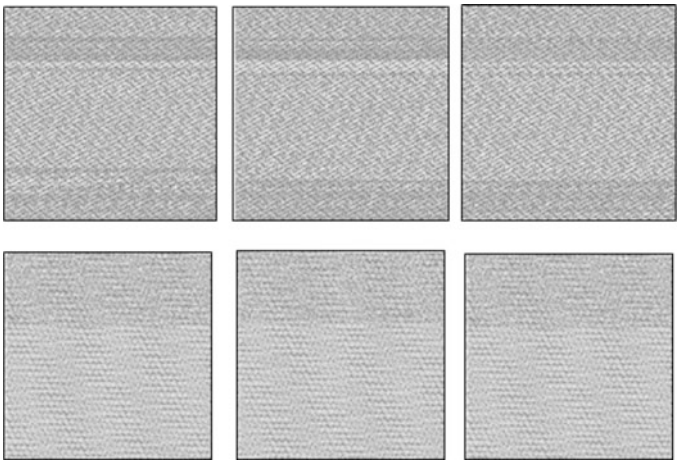


Fig. 2 The first row represents gray-scale images of malware samples belonging to the Ramnit [53] family of malware while the second row represents gray-scale images of malware samples belonging to the Kelihos_ver3 [53] family of malware

As shown in the figure, distinct regions in the gray-scale image of a PE malware binary correspond to specific sections in the PE structure. Examples of malware-to-image transformations of two unique malware families are shown in Fig. 2. It can be observed that malware samples belonging to the same family of malware are visually extremely similar when converted to gray-scale images. Another observation is that the images of malware belonging to a specific family will be distinct from those belonging to a different family.

3.3 PE-Based Adversarial Malware Attacks

The existing literature demonstrates several ways of crafting PE-based malware samples in order to misclassify the detector. The proposed approaches can be categorized into three classes:

- Byte append attacks.
- Feature modification attacks.
- Malicious code append attacks.

3.3.1 Byte Append Attacks

In the byte append attacks, a portion of bytes is appended to the end of the malware samples to misclassify the target classifier. Byte append attacks preserve the functionality of the malware samples since the payload is not modified. The appended bytes can be crafted by means of the gradient-based method of Biggio et al. [6] as in [28], or the Fast Gradient Sign Method (FGSM) of Goodfellow et al. [19] as in [29]. Moreover, the adversary can use benign files while perturbing malicious samples. In [56], the authors selected portions from the beginning parts of the benign files for this purpose. Another work by Chen et al. [13] proposed four techniques: choosing random parts of the benign files, determining the most contributing parts of the benign files, selecting the parts of a benign file according to saliency vectors, and combination of saliency vector with FGSM approaches. There exist two additional attacks that can add crafted bytes to the unused sections in a PE-based malware [29] and modify the so-called *slack* bytes which are added to the PE files by compilers for alignment purposes [56]. However, such modifications, even if possible to realize, have a high probability of altering the functionality of the modified malware samples.

3.3.2 Feature Modification Attacks

In feature modification attacks, the adversary modifies the features of a malware sample to make it resemble a benign PE file. In this attack, ML-based malware classifiers use common features for alterations (e.g., API/system calls, opcodes, network connections, file system operations, CPU registers, PE file characteristics, and strings [59]). To preserve the functionality of the modified malware sample, features are added to the sample without being removed. Examples of feature modification attacks include adding API features, adding features via common libraries, and using Generative Adversarial Networks (GANs).

Adding API Features Rosenberg et al. [54] considered adding no-operation API calls to the random positions advised by the Jacobian [48] algorithm. To preserve the functionality of the malware, the authors encapsulated the original malware binary

with proxy codes and external Dynamically Linked Libraries (DLLs). During the execution, the API hooking mechanism interrupts the execution of the malware and inserts the added API call. In this respect, Chen et al. [11] consider eliminating the most important features from the malware sample and adding the most important features of benign files to the malware to increase the misclassification detection rate. Al-Dujaili et al. [1] extracted API calls from PE-based malware samples and applied four perturbations based on the gradient-based attack [21] on the encoded features of the samples. However, the perturbed features were not mapped back to the malware binaries, and the functionality of the modified samples was not verified.

Adding Features via LIEF Library A number of studies employed the open-source LIEF (Library to Instrument Executable Formats) library [31] to modify the features of PE-based malware samples. The first study in this context was proposed by Anderson et al. [2]. The authors determined a set of perturbation options that can be applied via the LIEF library and that ideally should not alter the functionality of the modified malware. The perturbation options are applied to the malware samples by the mean of a reinforcement learning agent and include: inserting an unused function to the import address table, modification of section names, adding unused sections, appending bytes, modifying the debug information, packing, and unpacking. However, these perturbation options change the functionality of the malware and suffer from low performance. Other studies considered the same methodology [9, 10, 16, 30]. Among them, Fleshman et al. [16] applied Anderson's attack on four different antivirus products and two ML-based malware detectors (i.e., n-gram-based and Malconv). Their experimental results showed that ML-based detectors were not affected by Anderson's benign modification attacks. Another work utilized a genetic programming-based evasion framework for PE-based malware classifiers [9]. The proposed framework makes use of Anderson's attacks against an ML-based classifier (Gradient Boosted Decision Tree) and three commercial antivirus products (i.e., Kaspersky, Eset, and Sophos). The authors used a modified version of Cuckoo [14] sandbox environment to verify the functionality of malware instances after the perturbations. However, their framework was able to generate functional malware samples for just above 20% of their input files within the dataset.

Using Generative Adversarial Networks Another type of feature modification attack was introduced by Hu et al. [23]. The authors introduced a framework based on Generative Adversarial Networks (GANs) for adversarial malware crafting. By utilizing GAN's generator and detector components [39], they transform malware features into adversarial malware features to bypass the detection system. However, the authors fail to provide an explanation regarding the specific process for adding features to the malware samples. Furthermore, they do not provide any verification regarding the preservation of functionality in the modified samples.

3.3.3 Malicious Code Append Attacks

Fleshman et al. [16] applied a malicious code injection attack that appends malicious codes to benign files using a Return Oriented Programming Injector (ROPInjector) tool of Poulios et al. [51]. Their results demonstrated the effectiveness of this attack against both antivirus products and ML-based malware detectors. However, ROPInjector cannot inject arbitrary malicious code into benign PE files. It requires that the instructions and functionalities of both the malicious and benign PE files are similar to each other.

4 Problem Scope and Threat Model

4.1 Problem Definition

Detection of malware has been one of the most active problems of research and practices in computer and network security. Traditional signature and heuristics-based malware detection approaches could not cope with the proliferation of new and modified malware in the wild, as well as concealment techniques (e.g., obfuscation, packing, polymorphism, metamorphism) employed by adversaries [5]. For these reasons, ML techniques have been indispensable to malware detection. In such a setting ML models can be trained on malware samples in the wild, which may have any concealment techniques, and ML models can learn the patterns in the malicious software and successfully detect unknown malware samples (i.e., zero-day malware) with higher performance over signature- or heuristics-based approaches. However, studies in the last decade showed that ML models are susceptible to adversarial ML attacks. Perturbations that are carefully crafted based on the gradients of the ML models can cause such high-performance models to misclassify the samples. On the other hand, ML-based malware classifiers are also vulnerable and can be successfully evaded by adversarial attacks. In this study, we design and construct a lightweight CNN image-based malware classifier with high detection accuracy. Then, we assess its robustness against adversarial attacks under white box and black box settings.

4.2 Threat Model

In our study, we consider an adversary as an individual attempting to evade ML-based malware classifiers through adversarial attacks. We assume that the adversary is capable of adding minute perturbations to the Windows PE-based malware samples. The adversary's goal is to force the target classifier to misclassify the modified samples as benign. We consider two scenarios of the adversary. In the first

scenario, the adversary has perfect knowledge of the target model. This knowledge consists of information about the internal network architecture, hyperparameters, and data. Here, this scenario is referred to as a *white-box setting* and considered the ideal scenario for the adversary. In the second scenario, the adversary has limited knowledge of the target model. In this case, the attacker cannot access the hyperparameters and has typical access only to the input and output of the target model. We refer to this scenario as a *black-box setting*. In each of these two scenarios, the crafting of adversarial samples via adversarial attacks is governed according to the following set of adversarial goals and assumptions:

- The crafted adversarial malware sample can retain its malicious functionality after performing adversarial perturbations.
- The modifications are minimal while resulting in the target model misclassifying legitimate malware samples as benign.

It should be noted that common concealment techniques (e.g., obfuscation, packing, polymorphism, metamorphism), employed by malware authors are not incorporated in the threat model. This is due to the fact that these attacks do not target specific machine learning models, rather they are used by malware authors to bypass traditional signature- or heuristics-based detection tools. In fact, such concealment techniques were one of the driving reasons for the employment of ML techniques for malware detection. Contrary to the aforementioned concealment techniques, the adversarial attacks considered in this study are based on the internals of the ML model. Considering these goals, the adversary applies two black-box and two white-box attacks.

5 Proposed Image-Based Malware Classifier

In this section, we describe the details of our proposed image-based malware classifier. This includes an overview of our methodology, the considered network architecture, and a description of the dataset, with their preprocessing phase.

5.1 Methodology

Our proposed methodology comprises of a three-stage process, as illustrated in Fig. 3. In the first stage, each malware binary undergoes a pre-processing phase where it is converted to an array of unsigned 8-bit integers and normalized to a common size. These arrays represent the binaries as gray-scale images and are used to train a Convolutional Neural Network (CNN) in the second stage. In the third stage, the adversarial examples are then generated using each of the 4 attack vectors: Brute-Force Random Byte Append attack, Brute-Force Benign Byte Append attack, Random Byte FGSM attack, and Benign Byte FGSM attack.

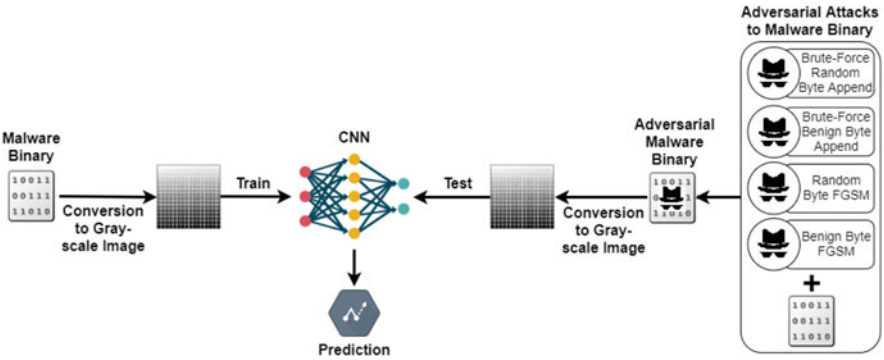


Fig. 3 An overview of our proposed approach. Malware binaries are converted into gray-scale images before being fed to the CNN model for the training process. A total of 4 adversarial ML attacks are applied to malware binaries at the testing phase. Similar to the training process, these samples are converted to images and then fed to the model as input in order to classify them according to the malware family they belong to

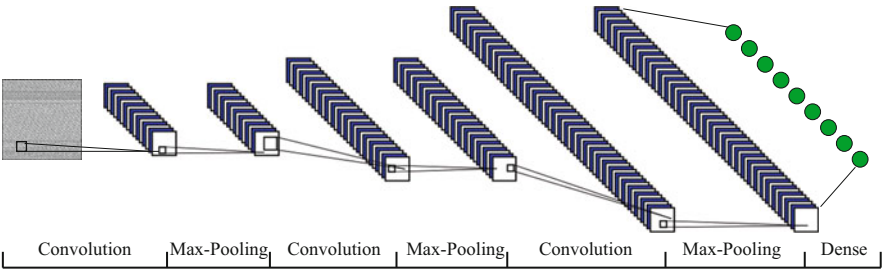


Fig. 4 The structure of the convolutional neural network used to classify malware binaries

5.2 Network Architecture

Our classifier consists of a Convolutional Neural Network (CNN) built using the TensorFlow software library, specifically using TensorFlow's high-level API, Keras. We trained our model on a system running MacOS Ventura v13.5.1 with Apple M1 Pro chip processor and 16 GB of available RAM. The system has a total of eight cores with each processor running a base frequency of 2.20 GHz. The structure of our CNN model consists of 3 sets of convolution layers followed by max-pool layers with an increasing number of filters in each successive convolution layer (16, 32, and 64). In each convolution layer, we set the kernel size to (3,3), while the pool size of each max-pooling layer is set to (2,2). These layers are followed by two dense layers with the final output being a vector representing the probability that a sample belongs to each of the nine classes in the dataset. In Fig. 4, we illustrate the structure of the considered CNN model. We trained our model on labeled, pre-processed malware samples from our dataset with a validation split of 0.2 (i.e. the model was trained on 80% of the samples while the remaining 20% are used to

validate or test the accuracy of our model). We labeled the samples from 0 to 8 according to which malware family they belong to. We set the number of epochs to 100 and the batch size set to 32. The model achieved an accuracy of 95.06% when tested against the malware samples in the validation set.

5.3 Dataset

In our study, we considered the MalwareDatabase datasets [33] and the Dike-Dataset [35] on GitHub.The MalwareDatabase datasets contain 3654 labeled malware portable executable files as they were filtered out from the rest of the dataset that were not PE files and 1346 files were collected from the DikeDataset making it a total of 5000 executables. The DikeDataset contains labeled malware samples from different families including Generic, Trojan, Ransomware, Worm, Backdoor, Spyware, Rootkit, Encrypter, and Downloader. For the benign samples, we selected 5000 executables from Benign-NET [34] repository found on GitHub. The structure of the benign samples is Win32EXE file type and is from pure installations of Windows 10 and Windows 7 operating systems. Then, we divided the set of labeled malware into training and validation sets using an 80/20 train-test split strategy, respectively. The aforementioned resources provide unlabeled malware samples in this regard and labeling must be done manually. Even still, most malware samples from these resources are classified under different malware families making it difficult to classify them into a single family of malware correctly.

5.4 Preprocessing: Conversion of Malware Binary to Image

Before training our model, we preprocessed the dataset into a format compatible with the model's input requirements. The preprocessing phase involves converting each malware binary in the training set to a gray-scale image and then resizing it to a common size. In what follows, we provide a line-by-line description of this procedure in Algorithm 1.

In *Line 3*, a for loop ensures that each file in the training set directory is visited with each iteration of the loop. From *Line 5* to *Line 7*, we calculate the size parameters to ensure that the final array will have a relatively similar length and width. In *Line 8*, we convert the file to an array of unsigned integers. From *Line 4* to *Line 10*, we convert the malware binary to an array of integers. In *Line 11* and *Line 12*, we reshape the created array and convert it into an array of 8-bit unsigned integers (uint8) that range in values from 0 to 255. The value of each integer in the array represents the brightness of a pixel ranging from black to white (0–255). In *Line 13* and *Line 14*, we resize the image array to a common size of 100 by 100 and we normalize the pixel values to a range of 0–1 by dividing the array by 255. This normalization is done as it is easier for the model to process input arrays

Algorithm 1: Malware Binary to Gray-scale Image

Input: Malware Binary
Output: Gray-scale Image Array of Malware Binary
for $file\ in\ getCwd()$ **do**
    $f \rightarrow$ open($file$)
    $ln \rightarrow$ getSize($file$)
    $width \rightarrow$ math.pow(ln, 0.5)
    $rem \rightarrow ln\%width$
    $a \rightarrow$ array('B')
    a.fromfile($f, ln - rem$)
    f.close()
    $g \rightarrow$ reshape(a, $(len(a)/width), width$)
    $g \rightarrow$ uint8(g)
    $h \rightarrow$ resize(g, size, size)
    $h \rightarrow h/255$
return h

with a smaller range of values. We continue this process in the directory until we successfully convert each file to an image array.

6 Considered Adversarial Attacks

In terms of adversarial attacks against malware classifiers, the functionality of the modified malware is guaranteed only for a subset of byte append attacks (i.e., random and benign byte append attacks) and one subset of the feature modification attacks (i.e., random and benign byte FGSM attacks). In this case, we first apply four byte-append attacks to generate adversarial samples that can evade MalConv [52], a state-of-the-art ML-based malware classifier. MalConv is a CNN-based malware classifier that analyzes the raw bytes of PE-based malware samples. It is a popular malware detector used in various studies as a target model to create adversarial samples from PE-based malware files [13, 16, 28, 29, 56]. Given a malware binary x of size $S(x)$, appended by byte perturbations p of size $S(p)$. The byte perturbations p cannot exceed 10% of the original sample size. This is because, from an adversarial point of view, the added perturbations are meant to be small, seemingly undetectable additions to the original malware binaries, relative to the original size of the binary. Other works in the literature append a maximum of only 1% of the original sample size [13, 28, 56], therefore, a maximum upper bound of 10% is suitable. In other words, the generation of adversarial samples through each method is bounded by the equation:

$$S(x + p) \leq 1.1 \times S(x) \tag{1}$$

where $x + p$ is the malware binary with appended byte perturbations.

6.1 Adversarial Attacks Under Black-Box Settings

Under black-box settings, we assume that the adversary has limited knowledge regarding the internal parameters or the structure of the target/victim model. The adversary has only access to the final classification result of the model with respect to a given input file. In what follows, we describe two-byte append attacks under black-blox settings that utilize brute-force techniques to generate adversarial malware samples.

Brute-Force Random Byte Append In this attack, we append randomly generated bytes to the end of a malware binary with each iteration until it is classified as benign or the size of the resulting binary reaches the maximum threshold. In case the adversarial sample x' generated through this method is still being classified correctly once this threshold is reached, we extract 10 bytes increments from random points in x and append them to the end of the binary. We continue this iterative process until the sample is classified as benign. A combination of both of these random byte-append techniques ensures that adversarial samples are generated for all of the malware binaries in our validation set.

Brute-Force Benign Byte Append In this attack, we append portions of benign files to the end of the malware binary x until it is either classified as benign or reaches the upper bound. With each iteration of the attack, we chose a file randomly from the set of benign files and we extract a section of 10 bytes from a random location. Then, we append it to the end of the malware binary. If the adversarial sample generated reaches the upper bound and is still classified correctly, we remove the perturbations and we repeat the process with another random benign file being selected from the data set, until x' is misclassified as a benign file.

6.2 Adversarial Attacks Under White-Box Settings

In this case, the adversary has complete access to the structure of the victim model, including the internal parameters, hyperparameters, and weights for the Convolutional Neural Network. In what follows, we describe two feature modification attacks under white-box settings.

Random Byte FGSM This method is an adaptation of the FGSM approach originally proposed by Goodfellow et al. for image-based deep learning classifiers [19]. The FGSM method creates adversarial malware samples by using the gradients of neural networks. The gradient of the cost function used to train the model, $J(\theta, x, y)$, with respect to an input malware binary, is used to generate a new binary that maximizes loss. This can be represented using the following equation:

$$x' = x + \epsilon * sign(\nabla_x J(\theta, x, y)) \tag{2}$$

Here, x is the original malware binary, y is the binary's original label, ϵ is a constant multiplier used to control the size of the perturbations, θ are the model's parameters and J is the loss function with respect to original malware binary. The main goal is to create a new malware binary x' that maximizes the loss function. This can be achieved by appending a certain number of bytes, namely $numBytes$ in the form of random bytes to a malware binary and updating their values (as dictated by Eq. (2)) in an iterative fashion with the binary moving further away from its original label with each iteration. In this case, The number of bytes appended with each iteration is set to 100, while the number of iterations is similarly set to 100. With MalConv, the model is not differentiable end-to-end as the input bytes are mapped to an 8-dimensional vector in the embedding layer, and therefore computing the gradient is not possible. To overcome this issue, as proposed in [28] and [56], the gradient-based updates of the appended bytes are performed in the embedding space and then the updated byte value is mapped to the nearest byte value along the direction of the embedding gradient.

Benign Byte FGSM This attack is very similar to the aforementioned FGSM attack except that instead of adding $numBytes$ in the form of random bytes, it adds benign byte portions from a randomly selected file from the set of benign files. The byte values are then updated iteratively over $numIterations$ using Eq. (2).

7 Performance and Robustness Evaluation

In this section, we evaluate the performance of our image-based malware classifier, followed by the overhead analysis. Afterward, we assess the robustness of our model against adversarial attacks.

7.1 Performance Analysis

To analyze the performance of our model, we compare our image-based classifier against MalConv in terms of classification accuracy and overhead analysis.

7.1.1 Classification Accuracy

To evaluate the classification accuracy of our image-based malware classifier, we considered several accuracy metrics including accuracy, precision, recall, and F1-score. Then, we compared these metrics with MalConv. In Table 1, we report the numerical results of calculating each of these metrics for both classifiers.

Table 1 Accuracy metrics for both classifiers

	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)
MalConv	95.29	95.26	95.32	95.28
Image-based classifier (ours)	96.30	95.30	96.27	96.27

We remark that image-based classifier and MalConv have similar performance in every category. Both detection models achieve an average detection accuracy of 95.8%.

7.1.2 Overhead Analysis

We monitored the overhead of our image-based malware classifier and compared it to MalConv. We remark that the majority of the overhead was incurred during the training and pre-processing stages of the CNN. This preprocessed dataset was stored in virtual memory as an array of arrays, taking up a total of 5.82 GB of space. The total execution time to train the model on our system was 12 minutes and 52 seconds with an average RAM usage during training being 36.81% or 5.89 GB. With MalConv, the total execution time to train the model on the same dataset was 12 minutes and 41 seconds with an average RAM usage during training being 43.62% or 6.98 GB. This is approximately a similar duration in training time and a 18.5% increase in RAM usage as compared to our image-based classifier.

7.2 Robustness Analysis

To evaluate the robustness of our classifier against adversarial attacks, we generated adversarial samples using each of the four attack methods described in Sect. 6. Subsequently, we tested these attacks on our classifier to measure the evasion rate. Finally, we compared the experimental results of the evasion rates of the attacks when applied to MalConv. To guarantee a fair comparison, we trained MalConv with the same validation split as the image-based classifier (i.e., we trained and tested MalConv on the same samples as the image-based classifier). In addition, we applied the methods used to create adversarial samples to the validation set to ensure that adversarial samples were created from malware binaries that the model had not been trained on. In Table 2, we show the evasion rate of each of the four attacks when tested against MalConv and our image-based classifier.

In Table 2, we provide insights regarding the evasion rate of our Image-based classifier as well as MalConv against four different adversarial attacks. We remark that the Random Byte Append attack achieves a relatively high evasion rate of 54.66% compared to the Image-Based classifier with 5.66%, suggesting its robustness against the Random Byte Append attack. Similarly, the Benign Byte

Table 2 A comparison of the evasion rates of the adversarial attacks when applied to MalConv and our image-based classifier

	Evasion rate (%)	
Adversarial attacks	MalConv	Image-based classifier
Random append	54.66	5.66
Benign append	44.22	5.11
Random FGSM	55.18	100
Benign FGSM	55.19	46.69

Append attack exhibits a lower evasion rate of 5.11% for the Image-Based classifier while it has a high evasion rate for MalConv of 44.22%. However, the Image-Based classifier fails against the Random FGSM attack while MalConv still has less vulnerability with an evasion rate of 55.18%. This could be explained by the model's sensitivity to random perturbations, and by preprocessing data into images, which can be easily evaded through gradient-based attacks. For the Benign Byte FGSM attack, the Image-Based classifier is more robust than Malconv, where the attack has an evasion rate of 55.19% for MalConv and 46.09% for the Image-Based classifier. Given these results, we can conclude that the image-based classifier performed substantially better than MalConv and remained robust to adversarially generated perturbations across most adversarial malware generation methods.

8 Discussion

In this section, we discuss the underlying reasons behind the performance of our image-based classifier, the choice of attacks, and finally, the benefits of our study.

Understanding the Robustness According to the obtained results, it is apparent that the image-based malware classifier remained in most cases robust against adversarial malware samples. The underlying reason for this performance is that the perturbations are added to the end of a malware binary. As a result, when the binary is converted into a gray-scale image, the majority of the image remains identical to the original unperturbed sample. This allows the classifier to correctly predict the class of malware the adversarial sample belongs to.

Preserving Malware Functionality The adversarial samples created against MalConv in this method ensure that the malware binary is modified and the malware functionality is preserved. In the case of an image-based classifier, if adversarial samples are created for the image-based classifier, they would be adversarial image samples and not adversarial malware samples. Even if these adversarial images were converted back to binaries, there is no guarantee that the original functionality of the malware is preserved. This is because adversarial ML attacks on images are not localized to specific regions of the image (i.e perturbations can be added in any region of the image), and this may alter the malware functionality as these perturbations could correspond to adding bytes to executable portions of malware code.

Choice of Attacks Although there are a variety of attacks in the literature that claim to generate adversarial malware, the attack methods chosen for this study are the only ones that undoubtedly retain the malware functionality. In addition to retain the malicious functionality, we considered two black-box attacks as novel techniques not seen in previous literature. Methods aside from the byte-append attacks described in Sect. 2, such as feature modification attacks [2, 9, 10, 16, 30] and malicious code append attacks [16, 51] do not guarantee the preservation of malware functionality. Moreover, the testing of functionality for adversarial malware samples crafted from these methods would be difficult as it would require dynamic analysis in a sandbox environment and a significant portion of malware samples do not run in such virtualized environments [54, 61]. This is done in order to hinder dynamic analysis and prevent malware testers from extracting run-time features of malware samples.

Benefits Our image-based malware classifier outperformed MalConv, a widely used raw-byte-based malware classifier substantially in most of the recorded accuracy metrics. It remained robust to most of the adversarially crafted malware samples across different attack settings. In addition, since all adversarial malware samples retained their malicious functionality, our image-based classifier was tested under realistic circumstances.

9 Conclusion

As the number of malware samples in the wild increases at an alarming rate, adversaries continue to discover means to mask malware with perturbations to evade malware classifiers. In this chapter, we assessed the robustness of CNN-based image classifier against adversarial attacks that preserve the functionality of the malware in black-box and white-box settings. The results of our study indicate that our image-based classifier outperformed the state-of-the-art ML-based malware classifier, MalConv, in most of the attacks. Our proposed technique is resilient to some extent against adversarial attacks and can pave the way for the development of other malware detection mechanisms that are resilient to adversarial perturbations. The performance evaluation of our classifier demonstrated a similar RAM usage during the training process, highlighting its effectiveness and practicality. In future work, we aim to investigate the effectiveness of adversarial samples that modify or append bytes to regions of the malware binary besides the end of the file.

Acknowledgments We would like to acknowledge Sevval Deniz Erkurt for helping us on reproducing some of the results presented in this paper to ensure the accuracy and reliability of our study. This work is partially supported by the US National Science Foundation (Awards: 1663051, 2039606, 2219920), Cyber Florida, Google ASPIRE Program, and Microsoft. The views expressed are those of the authors only, not of the funding agencies.

References

1. Al-Dujaili A, Huang A, Hemberg E, O'Reilly U (2018) Adversarial deep learning for robust detection of binary encoded malware. In: 2018 IEEE security and privacy workshops (SPW), pp 76–82
2. Anderson HS, Kharkar A, Filar B, Evans D, Roth P (2018) Learning to evade static pe machine learning malware models via reinforcement learning. Preprint arXiv:1801.08917
3. Baptista I, Shiaeles S, Kolokotronis N (2019) A novel malware detection system based on machine learning and binary visualization. In: 2019 IEEE international conference on communications workshops (ICC workshops), pp 1–6
4. Berman DS, Buczak AL, Chavis JS, Corbett CL (2019) A survey of deep learning methods for cyber security. Information 10(4)
5. Bhansali S, Aris A, Acar A, Oz H, Uluagac AS (2022) A first look at code obfuscation for webassembly. In: Proceedings of the 15th ACM conference on security and privacy in wireless and mobile networks
6. Biggio B, Corona I, Maiorca D, Nelson B, Šrndić N, Laskov P, Giacinto G, Roli F (2013) Evasion attacks against machine learning at test time. In: Machine learning and knowledge discovery in databases, Berlin, Heidelberg, pp 387–402
7. Carlini N, Wagner D (2017) Towards evaluating the robustness of neural networks. In: 2017 IEEE symposium on security and privacy (SP), pp 39–57
8. Carlini N, Wagner D (2018) Audio adversarial examples: targeted attacks on speech-to-text. In: 2018 IEEE security and privacy workshops (SPW), pp 1–7 (2018)
9. Castro RL, Schmitt C, Dreo G (2019) Aimed: evolving malware with genetic programming to evade detection. In: 18th IEEE international conference on trust, security and privacy in computing and communications/13th IEEE international conference on big data science and engineering (TrustCom/BigDataSE), pp 240–247
10. Castro RL, Schmitt C, Rodosek GD (2019) Armed: how automatic malware modifications can evade static detection? In: 5th international conference on information management (ICIM)
11. Chen L, Ye Y, Bourlai T (2017) Adversarial machine learning in malware detection: arms race between evasion attack and defense. In: 2017 European intelligence and security informatics conference (EISIC), pp 99–106
12. Chen L, Hou S, Ye Y (2017) Securedroid: enhancing security of machine learning-based detection against adversarial android malware attacks. In: Proceedings of the 33rd annual computer security applications conference, ACSAC 2017, New York, NY, USA, 2017. Association for Computing Machinery, pp 362–372
13. Chen B, Ren Z, Yu C, Hussain I, Liu J (2019) Adversarial examples for cnn-based malware detectors. IEEE Access 7
14. Cuckoo sandbox automated malware analysis. https://cuckoosandbox.org/, 2020
15. Ebrahimi J, Rao A, Lowd D, Dou D (2018) HotFlip: white-box adversarial examples for text classification. In: Proceedings of the 56th annual meeting of the association for computational linguistics (Volume 2: Short papers), Melbourne, Australia. Association for Computational Linguistics, pp 31–36
16. Fleshman W, Raff E, Zak R, McLean M, Nicholas C (2018) Static malware detection & subterfuge: quantifying the robustness of machine learning and current anti-virus. In: Proceedings of the AAAI symposium on adversary-aware learning techniques and trends in Cybersecurity (ALEC 2018) Arlington, Virginia, USA, October 18–20, 2018, pp 3–10
17. Fu J, Xue J, Wang Y, Liu Z, Shan C (2018) Malware visualization for fine-grained classification. IEEE Access 6:14510–14523
18. Gibert D, Mateu C, Planes J (2020) The rise of machine learning for detection and classification of malware: Research developments, trends and challenges. J Netw Comput Appl 153:102526
19. Goodfellow IJ, Shlens J, Szegedy C (2015) Explaining and harnessing adversarial examples. In: 3rd international conference on learning representations, ICLR 2015, San Diego, CA, USA

20. Goodfellow I, McDaniel P, Papernot N (2018) Making machine learning robust against adversarial inputs. Commun ACM 61(7):56–66
21. Grosse K, Papernot N, Manoharan P, Backes M, McDaniel P (2017) Adversarial examples for malware detection. In: SN Foley, D Gollmann, E Snekkenes (eds) Computer Security – ESORICS 2017. Springer International Publishing, Cham, pp 62–79
22. Han K, Lim JH, Im EG (2013) Malware analysis method using visualization of binary files. In: Proceedings of the 2013 research in adaptive and convergent systems, RACS '13, New York, NY, USA, 2013. Association for Computing Machinery, pp 317–321
23. Hu W, Tan Y (2022) Generating adversarial malware examples for black-box attacks based on GAN. International Conference on Data Mining and Big Data. Springer, pp 409–423
24. Intentional PE Corruption (2020) https://blog.malwarebytes.com/cybercrime/2012/04/intentional-pe-corruption/, [Online; accessed 15 Apr 2023]
25. ISTR internet security threat report. Technical Report, Symantec, February 2019
26. Kancherla K, Mukkamala S (2013) Image visualization based malware detection. In: 2013 IEEE symposium on computational intelligence in cyber security (CICS), pp 40–44
27. Khormali A, Abusnaina A, Chen S, Nyang DH, Mohaisen A (2019) Copycat: practical adversarial attacks on visualization-based malware detection
28. Kolosnjaji B, Demontis A, Biggio B, Maiorca D, Giacinto G, Eckert C, Roli F (2018) Adversarial malware binaries: evading deep learning for malware detection in executables. In: 26th European signal processing conference (EUSIPCO). IEEE, Rome.
29. Kreuk F, Barak A, Aviv S, Baruch M, Pinkas B, Keshet J (2018) Deceiving end-to-end deep learning malware detectors using adversarial examples. In: NeurIPS 2018 workshop on security in machine learning, Montreal, Canada
30. Labaca-Castro R, Biggio B, Rodosek GD (2019) Poster: attacking malware classifiers by crafting gradient-attacks that preserve functionality. In: Proceedings of the 2019 ACM SIGSAC conference on computer and communications security, CCS '19, New York, NY, USA. Association for Computing Machinery, pp 2565–2567
31. Library to Instrument Executable Formats (2020) https://lief.quarkslab.com/ [Online; accessed 2 Feb 2023]
32. Liu X, Zhang J, Lin Y, Li H (2019) Atmpa: Attacking machine learning-based malware visualization detection methods via adversarial examples. In: Proceedings of the international symposium on quality of service, IWQoS '19, New York, NY, USA. Association for Computing Machinery
33. MalwareDatabase (2021) https://github.com/Vichingo455/MalwareDatabase [Online; accessed 15 Sep 2023]
34. MalwareDatabase (2022) https://github.com/bormaa/Benign-NET/ [Online; accessed 15 Sep 2023]
35. MalwareDatabase (2023) https://github.com/iosifache/DikeDataset/tree/main [Online; accessed 15 Sep 2023]
36. Mekdad Y, Bernieri G, Conti M, Fergougui AE (2021) The rise of ics malware: a comparative analysis. In: European symposium on research in computer security. Springer, pp 496–511
37. Nataraj L, Karthikeyan S, Jacob G, Manjunath BS (2011) Malware images: visualization and automatic classification. In: Proceedings of the 8th international symposium on visualization for cyber security, VizSec '11, New York, NY, USA. ACM
38. Ni S, Qian Q, Zhang R (2018) Malware identification using visualization images and deep learning. Comput Secur 77:871–885
39. Nowroozi E, Mekdad Y (2023) Detecting high-quality gan-generated face images using neural networks. Big Data Anal Intell Syst Cyber Threat Intell, 235–252
40. Nowroozi E, Mekdad Y, Berenjestanaki MH, Conti M, Fergougui AE (2022) Demystifying the transferability of adversarial attacks in computer networks. IEEE Trans Netw Service Manag 19(3):3387–3400
41. Nowroozi E, Mohammadi M, Golmohammadi P, Mekdad Y, Conti M, Uluagac S (2022) Resisting deep learning models against adversarial attack transferability via feature randomization. Preprint, arXiv:2209.04930

42. Nowroozi E, Mohammadi M, Savaş E, Mekdad Y, Conti M (2023) Employing deep ensemble learning for improving the security of computer networks against adversarial attacks. IEEE Trans Netw Service Manag 20(2):2096–2105
43. Oz H, Aris A, Levi A, Uluagac AS (2022) A survey on ransomware: evolution, taxonomy, and defense solutions. ACM Comput Surv 54(11s)
44. Oz H, Naseem F, Aris A, Acar A, Tuncay GS, Uluagac AS (2022) Poster: feasibility of malware visualization techniques against adversarial machine learning attacks. In: 43rd IEEE symposium on security and privacy (S&P)
45. Oz H, Aris A, Acar A, Tuncay GS, Babun L, Uluagac AS (2023) RøB: ransomware over modern web browsers. In: 32nd USENIX security symposium (USENIX Security 23), Anaheim, CA, August 2023. USENIX Association, pp 7073–7090
46. Papernot N, McDaniel P (2017) Extending defensive distillation. Preprint, arXiv:1705.05264
47. Papernot N, McDaniel P, Goodfellow I (2016) Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. arXiv preprint arXiv:1605.07277
48. Papernot N, McDaniel PD, Jha S, Fredrikson M, Celik ZB, Swami A (2016) The limitations of deep learning in adversarial settings. In: IEEE European symposium on security and privacy, EuroS&P 2016, Saarbrücken, Germany, March 21–24, 2016, pp 372–387
49. Park D, Khan H, Yener B (2019) Generation evaluation of adversarial examples for malware obfuscation. In: 2019 18th IEEE international conference on machine learning and applications (ICMLA), pp 1283–1290
50. Pe format (2020) https://docs.microsoft.com/en-us/windows/win32/debug/pe-format [Online; accessed 8 Feb 2023]
51. Poulios G, Ntantogian C, Xenakis C (2015) ROPInjector: using return-oriented programming for polymorphism and av evasion. In: Black Hat USA
52. Raff E, Barker J, Sylvester J, Brandon R, Catanzaro B, Nicholas CK (2018) Malware detection by eating a whole EXE. In: The workshops of the the thirty-second AAAI conference on artificial intelligence, New Orleans, Louisiana, USA, February 2–7, pp 268–276
53. Ronen R, Radu M, Feuerstein C, Yom-Tov E, Ahmadi M (2018) Microsoft malware classification challenge. CoRR, abs/1802.10135
54. Rosenberg I, Shabtai A, Rokach L, Elovici Y (2018) Generic black-box end-to-end attack against state of the art API call based malware classifiers. In: Research in attacks, intrusions, and defenses. Springer International Publishing, Cham, pp 490–510
55. Šrndic N, Laskov P (2014) Practical evasion of a learning-based classifier: a case study. In: 2014 IEEE symposium on security and privacy, pp 197–211
56. Suciu O, Coull SE, Johns J (2019) Exploring adversarial examples in malware detection. In: 2019 IEEE security and privacy workshops (SPW), pp 8–14
57. Szegedy C, Zaremba W, Sutskever I, Bruna J, Erhan D, Goodfellow IJ, Fergus R (2014) Intriguing properties of neural networks. In: 2nd international conference on learning representations, ICLR 2014, Banff, AB, Canada, April 14–16
58. Torres JM, Comesaña CI, García-Nieto PJ (2019) Review: machine learning techniques applied to cybersecurity. Int J Mach Learn Cybern 10(10):2823–2836
59. Ucci D, Aniello L, Baldoni R (2019) Survey of machine learning techniques for malware analysis. Comput Secur 81:123–147
60. Vi BN, Noi Nguyen H, Nguyen NT, Truong Tran C (2019) Adversarial examples against image-based malware classification systems. In: 2019 11th international conference on knowledge and systems engineering (KSE), pp 1–5
61. Xu W, Qi Y, Evans D (2016) Automatically evading classifiers: a case study on PDF malware classifiers. In: 23rd annual network and distributed system security symposium, NDSS 2016, San Diego, California, USA, February 21–24, 2016
62. Yang W, Kong D, Xie T, Gunter CA (2017) Malware detection in adversarial settings: exploiting feature evolutions and confusions in android apps. In: Proceedings of the 33rd annual computer security applications conference, ACSAC 2017, New York, NY, USA, 2017. Association for Computing Machinery, pp 288–302
63. Ye Y, Li T, Adjeroh D, Iyengar SS (2017) A survey on malware detection using data mining techniques. ACM Comput Surv 50(3)

The Cost of Privacy: A Comprehensive Analysis of the Security Issues in Federated Learning

Agnideven Palanisamy Sundar, Feng Li, Xukai Zou, and Tianchong Gao

1 Federated Learning Basics

Let's first start with the origin and the need for Federated Learning. We will also look at some of the common applications of FL, followed by a brief discussion of the vanilla implementation of a widely used FL algorithm.

1.1 What Is Federated Learning? Why Do We Need It?

To understand what FL is and the need for FL, we first need to understand Machine Learning as a whole. Machine Learning (ML) is a subset of Artificial Intelligence (AI) that focuses on the development of algorithms and models that enable computers to learn from and make predictions or decisions based on data without being explicitly programmed [22, 70]. At its core, machine learning revolves around the idea of training a model using data to recognize patterns, make predictions, or take action. Though ML was conceptualized in the 40s [34], its mainstream adoption

A. Palanisamy Sundar · X. Zou
Department of Computer and Information Science, School of Science, Indiana University Purdue University Indianapolis, Indianapolis, IN, USA
e-mail: agpalan@iu.edu; xzou@iupui.edu

F. Li (✉)
Department of Computer and Information Technology, School of Engineering and Technology, Indiana University Purdue University Indianapolis, Indianapolis, IN, USA
e-mail: fengli@iupui.edu

T. Gao
School of Cyber Science and Engineering, Southeast University, Nanjing, China
e-mail: tgao@seu.edu.cn

Y. Chen et al. (eds.), *Network Security Empowered by Artificial Intelligence*,
Advances in Information Security 107, https://doi.org/10.1007/978-3-031-53510-9_14

happened much later, around the 2010s. During this time, there was a convergence of several factors that contributed to the widespread popularity and application of ML techniques. The most important factor was the availability of large datasets. Additionally, data storage and processing technology advancements made it more feasible to handle and analyze these large datasets. Another crucial factor was the development of a more powerful and scalable computational infrastructure. In essence, to build a good ML model, the two key factors are the availability of high-quality massive datasets and the computational resources powerful enough to handle the training process on the vast dataset.

If someone or a company has a small dataset and wants to create a good ML model, they may face some challenges. One option is to build a model with the limited dataset they have. However, this may result in the model not performing as expected or overfitting the training dataset. Another option is to combine data from multiple individuals with similar small datasets on a central server to create a larger dataset suitable for building an ML model. Although possible, this method has its drawbacks. Some data, such as hospital patient information, cannot be shared with others due to privacy and security concerns. Additionally, recent GDPR [8] and CCPA [5] regulations prevent companies from sharing consumer data with others. Moreover, combining data from different entities may lead to communication overhead related to data sharing. Fortunately, Federated Learning offers a solution to these problems.

In 2016, Google published a paper titled "Communication-Efficient Learning of Deep Networks from Decentralized Data" [35], which coins the term Federated Learning, and describes it as a method "that leaves the training data distributed on the mobile devices, and learns a shared model by aggregating locally-computed updates". To elaborate, FL is an ML strategy where individual participants/clients do not have to share their local dataset in order to build a combined model. Instead, the clients build a local model based on their local data and share the model updates with the central server. The central server aggregates the model updates and sends them to the participating clients. The aggregated model is known as the global model and effectively has a performance similar to a model built by combining all the data.

Federated Learning is highly privacy-preserving as the training data never has to leave the owners' premises. Additionally, instead of a single server trying to build a model with a massive dataset, FL allows for the computational overhead to be distributed among all its clients. So, no individual entity has to carry the burden of building a huge model, reducing the strain on their resources. Likewise, the communication overhead is also reduced since the model updates are much smaller in size than the actual data in most cases. A combination of these benefits is slowly leading to the widescale adoption of FL.

1.2 Applications of Federated Learning

From the discussions earlier, it is clear how FL can be a helpful asset in various scenarios. To drive home the benefits of FL, we will look at some of the practical, real-world applications of FL. The applications of FL are far and wide, and this list is in no way complete.

1.2.1 Healthcare

Collaborative model training using patient data from multiple healthcare providers can be achieved through federated learning without centralizing sensitive medical records. This approach enables hospitals or research institutions to develop strong models for disease prediction, personalized medicine, or anomaly detection while maintaining patient privacy [48, 63].

1.2.2 Internet of Things

FL is well-suited for IoT environments where numerous edge devices generate data. By training models locally on these devices, federated learning can improve IoT applications such as smart homes, energy management, predictive maintenance, and anomaly detection without transmitting sensitive data to a central server [24, 39].

1.2.3 Mobile Devices

FL is particularly relevant for mobile devices, where privacy concerns are paramount. Applications like personalized recommendations, language translation, keyboard prediction, and voice recognition can leverage federated learning to train models on user devices without compromising data privacy. Google's GBoard keyboard used FL for next-word prediction without sending a user's typing history directly to Google servers [18, 64].

1.2.4 Smart Grids

Smart grid systems can benefit from federated learning as it can optimize energy consumption and grid management. The system can collectively learn patterns, forecast demand, and improve energy efficiency by training local models on distributed sensors and meters. This is achieved without having to transmit granular energy consumption data to a central authority, thus ensuring privacy and security [50, 52].

1.2.5 Autonomous Vehicles

One way to improve the safety and functionality of autonomous vehicles is through federated learning. This approach allows models to be trained using data from multiple vehicles while still maintaining the privacy of sensitive information such as specific routes and locations. Collaborative training on individual vehicle devices can enhance object recognition, path planning, and overall safety [41, 46, 66].

1.2.6 Finance

Federated learning can be employed in the financial sector to build predictive models while ensuring the confidentiality of customer data. Banks or financial institutions can collaborate to develop fraud detection models or credit risk assessment models by training them locally on their respective datasets without sharing sensitive customer information [32].

1.3 Workflow of a Federated Learning System

The two essential entities in an FL system are the clients and the server. There can be multiple clients, but usually, only one central server acts as the aggregator. FL is a multi-round process that typically involves repeating the following steps. Step 1 is needed just for the first round of the FL process. Only steps 2 and 3 are repeated for all further rounds until the model converges. Figure 1 depicts the steps involved in the process.

Step 1*: Server-end Operation.* In the first round of the FL process, the central server, also known as the aggregator, randomly initializes a global model, G^0, which denotes the global model at round 0. This randomly initialized model and the parameters needed to train the local models are sent from the server to either a selected subset of clients or all the clients in the FL system.

Step 2*: Client-end Operation.* Once the clients receive the global model from the server, they use their local data to train on top of the global model based on the hyperparameters from the server. This model is known as the local model denoted by C_i^t, indicating the client i's model update in round t. After a stipulated number of local epochs, the client calculates the difference between the received global and local models ($C_i^t - G^{t-1}$). Here, G^{t-1} is the global model calculated in the previous round. This difference is then sent back to the server. Sending just the difference reduces the overall communication overhead.

Step 3*: Server-end Operation.* Upon receiving the updates from all the clients participating in that round, the global server calculates the updated global model based on an aggregation rule. This aggregation could be as simple as Federated-Averaging (FedAvg) [35], where the local model weights from the clients are

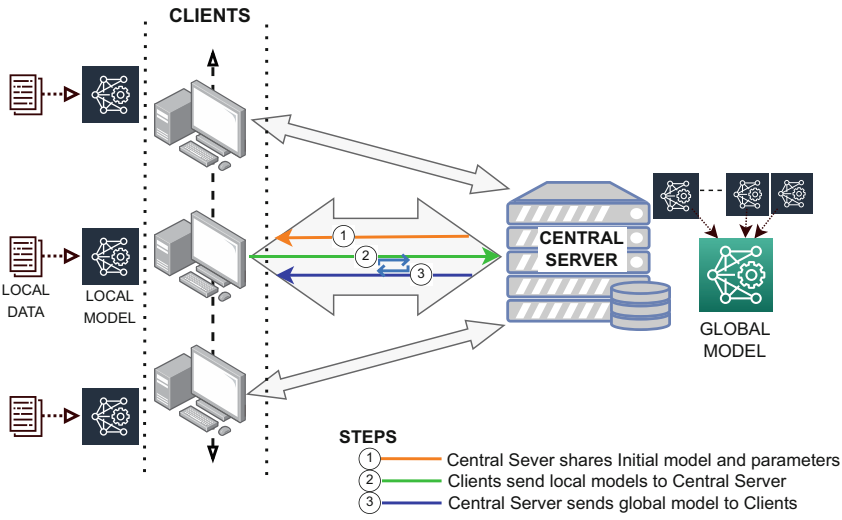


Fig. 1 Steps involved in a basic federated learning system

averaged to get the global model.

$$G^t = G^{t-1} + \frac{\eta}{m} \sum_{k=1}^{m} (C_i^t - G^{t-1}). \tag{1}$$

Here, the current global model G^t is calculated by taking an average of all the model updates' differences $(C_i^t - G^{t-1})$ submitted by the clients in round t, where m is the number of clients in that round, and η is the learning rate of the global model. The aggregation method can also be a variation of FedAvg, with the size of the local training data impacting the client's contribution to the global model, or it could involve a more complex process [3, 17, 40, 47]. After obtaining the new global model, the server selects a different subset of participants to continue the next round.

1.4 Factors to Consider

The performance of a Federated Learning System is dependent on the quality of the data present with its clients. The general assumption is that all the clients have the same computational resources, sufficient to train the models locally. But, having a similar assumption about data availability would be unrealistic. Here, we'll examine how the data can influence the FL process.

1.4.1 Based on Data Distribution

Consider a 10-class classification dataset in which all the clients in the FL system have the same number of data records from all 10 classes. Though it is highly unlikely, it is still a possibility, and such a distribution scenario is known as Independent and Identically Distributed (IID) data [35]. If the number of data records in each class differs for different clients or some of the classes are missing in some clients' datasets, it would be known as a Non-IID dataset [4, 56, 68].

In general, dealing with an IID dataset is more straightforward and usually gives better performance results than non-IID. But non-IID is a much more realistic situation, considering each client collects their data independently. The data distribution not only impacts the benign performance of the FL system but also affects its attacks and defenses.

1.4.2 Based on the Type of Feature Division

So far, we have assumed that the features available with all the clients are identical, but the data record is different. In other words, the feature space of the data records is similar, but the sample space is different. The common features allow the users to build their own local models and combine them in the central server. Such a setup for the FL process is known as Horizontal Federated Learning (HFL) [20, 48, 69].

Consider a situation where two different clients collect different information about the same item/person; that is, the features they collect are different. The sample space is the same, but the feature space is different. If this is the case, then the clients cannot directly average their model updates but would rather have to combine intermediate results and train a new model. Such a situation is known as the Vertical Federated Learning (VFL) setup [10, 25, 61]. The attack spaces for HFL and VFL are different, and most existing security attacks predominantly target HFL. So, we will only focus on HFL security in this chapter. If FL is mentioned anywhere in the text, we are talking about HFL, not VFL.

1.5 Common Threat Model

Currently, many research works deal with security and privacy issues with FL; they are either attacks or defenses on FL. Since different researchers have different focuses when they try to apply their methods to FL, it is essential for the researchers to follow a standard model to ensure that their method can be replicated in others' works and real-world applications. It also allows the researchers to establish the conditions under which their method works as expected. Let's look at some of the common threat model considerations before looking at the issues in federated learning in the next section [1, 2, 9, 40, 47]. The stronger the threat model assumption, the stronger the attack/defense method.

1.5.1 Number of Attackers

Most existing works consider the condition where a group of attackers works together to achieve the same attack objective. In most of the existing works, the assumption is that the number of attackers is always less than 50% of the total participants. In other words, the benign participants are the majority in the FL process, which is a realistic assumption in most cases. A weaker assumption would be to assume that only one attacker exists in the process.

1.5.2 Attacker Knowledge

In general, the attacker is believed to know the protocol used for the FL process, including the aggregation algorithm and any existing defense method used by the server. Similarly, the attacker does not know the local data or the model updates submitted by the benign participants. This is a strong and practical assumption. If any of these conditions do not apply, then it would be a weaker assumption, reducing its applicability in real-world tasks and its reliability in research works.

1.5.3 Attacker Capacity

If there are multiple attackers, it is assumed that they have a private channel to coordinate their attacks and control the updates submitted by other attackers. But, the attackers do not have control over the submissions made by benign clients. The attackers also do not control the order in which the central server selects the clients for each round.

1.5.4 Defender Knowledge

Generally, the central server is considered the defender. The defender does not know which of the participants are malicious and which ones are benign. The defender also does not know the size or the distribution of the clients' local data. Upon receiving the model updates, the defender gets access to the gradients of the updates submitted by all the clients. In certain defenses, the defender is assumed to have a small validation dataset representing the data available to the clients. Though practically feasible, it is considered a weaker assumption and is rarely used.

1.5.5 Defender Capacity

The defender has the privilege of slightly altering the model updates in an attempt to remove the influence of the attack. The defender can also modify the aggregation algorithm to protect the performance of the global model. Some defense methods

assume that the defender can employ clients to help with the defense process. Remember that the defender does not know which participants are benign or malicious.

2 Issues With Federated Learning

Though Federated Learning solves the problems of the small local dataset and large communication overhead while enhancing privacy, it comes with its own set of issues that concern the adoption of FL in some domains. Generally, an outsider attack is what disrupts the regular functioning of many ML protocols, but the privacy-preserving nature of FL makes it more susceptible to insider attacks. Most issues in FL systems arise from the possibility that some participating clients can act maliciously. Based on the attacker's intent, the issues can be broadly classified into three categories: Privacy, Security, and Free-riding. Since this chapter focuses on the security issues in FL, we will briefly discuss the privacy and free-rider attacks in this section and discuss security problems in detail in the next section.

2.1 Privacy Issues in FL

Federated Learning is undoubtedly a more privacy-preserving approach than the data centralization-based model-building scheme, but it does not make it immune to privacy concerns. A malicious entity posing as a client can try to obtain prohibited information through the FL process. The attacker would simultaneously try not to deviate from the FL objective. Let's now look at some of the subcategories of privacy attacks [38].

2.1.1 Inversion Attack

Inversion attack or Model Inversion attack is an attack in which the adversary tries to reconstruct or extract sensitive information from a trained model. By analyzing the model's outputs or gradients, an attacker may attempt to reconstruct sensitive training data or infer private information present in the model [15, 19, 29, 60].

2.1.2 Inference Attack

Inference attack is a subcategory of attack in which the malicious client tries to infer some information about the data used for training by other benign entities. If a malicious party attempts to determine if a specific sample was part of the training data used to build the federated model, it is known as a Membership Inference attack [33, 67]. If an adversary attempts to infer sensitive information about the training

data used by a specific client based on the model updates exchanged during the federated learning process, it is called a Data Inference attack [12, 55].

2.2 Free-Rider Issues in FL

A free-riding attack is a more straightforward attack that does not affect the model performance or client privacy by any means. When a participating client does not contribute any useful update to the aggregation process, it is known as a Free-rider attack [11, 30, 58]. The attacker refrains from contributing useful updates based on their local data but wants to obtain the model build based on the other client's datasets. In its simplest form, the attacker updates the difference between the current global model and the previous one as their local update, adding no value to the FL process. Though this might seem harmless, it robs the other benign clients of the ability to learn usable traits from the attackers' dataset.

3 Security Attacks on Federated Learning

Apart from attempting to invade the privacy of innocent clients in the FL system, the attackers have other goals. In terms of security concerns, the attacker does not aim to deduce prohibited data but instead aims to disrupt the anticipated function of the global model. This disruption can impact either the overall Main Task objective of the model or only a small portion of it. The term Main Task objective describes the anticipated actual outcome of the FL process. Essentially, it is the expected performance of the global model that all participating clients are striving towards.

Figure 2 shows how a security attack for the FL workflow is executed during step 2. Once the attacker receives the global model from the central server, he uses his poisoning methods to retrain the global model. This retraining causes the local update to remember the negative traits of the poison. Then, the attacker sends back the poisoned model update to the central server. During the aggregation process, the negative traits from the poisoned model updates manage to seep into the global model. The redistribution of the global model propagates the poisoned model to all the participating clients.

We can classify the security threats to FL based on the attack's objective and the attack's approach.

3.1 Based on Attack Objective

As mentioned earlier, an attacker can either try to disrupt the Main Task of the global model or include a backdoor to alter the model's functioning. The former attack objective is called a Byzantine attack; the latter is known as a Backdoor attack.

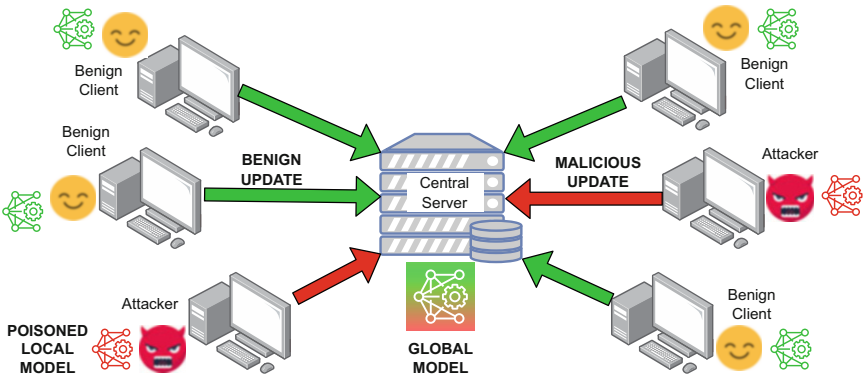


Fig. 2 Execution of a security attack in Step-2 of the FL process

3.1.1 Byzantine Attack

The aim of the malicious entity is to prevent the convergence of the global model. The convergence of the global generally implies that the Main Task objective of the FL process has been achieved, and the model is ready to be deployed. By executing a byzantine attack, the malicious client aims to render the global model useless, wasting the computation and communication resources of the benign participant and the central server [6, 9, 49]. The malicious participant may intentionally manipulate its model updates or gradients to mislead the aggregation process or introduce erroneous information.

3.1.2 Backdoor Attack

Unlike Byzantine attack, a backdoor attack tries to maintain the Main Task objective for most cases, except for a small subset of data records. Backdoor attacks involve manipulating a model to misclassify input records into a category the attacker selects [2, 51, 57, 62]. The attacker typically implants a small trigger pattern in a subset of the local training dataset and relabels it as the desired target category. This allows the local and global models to associate the pattern with the mislabeled category and misclassify data records containing the trigger. Without the trigger, the model functions benignly and achieves its Main Task objective. However, the presence of the trigger acts as a backdoor and causes the model to misclassify the data record.

$$\begin{aligned} f(x) &\longrightarrow y \\ f(x+\tau) &\longrightarrow y' \end{aligned} \tag{2}$$

Here, x represents the original data record, while y is its corresponding true label. By training model $f()$ with the backdoor trigger, it behaves normally when

the trigger is not present. However, once the trigger τ is embedded in the input data record, the model incorrectly classifies it as the attacker's selected target category, y'.

3.2 Based on Attack Approach

Based on the method the poison is injected into the model, we can classify the attacks into two categories. These approaches can be used for both byzantine and backdoor objectives.

3.2.1 Data Poisoning

One way to attack the model is to poison the data used for training the model [42, 53, 62]. Based on the objective, whether it is byzantine or backdoor, the extent of poisoning varies. The most straightforward data poisoning attack to achieve the byzantine goal is mislabeling all the training records, forcing the model to move away from the global objective. When the poisoned model is aggregated with the other benign models, it causes the global model to deviate from the Main Task objective. Likewise, the simplest backdoor data poisoning attack involves embedding a small trigger pattern onto a subset of input data and relabeling the data to the target class. Data poisoning attacks can be detected if the central server has access to the training data, which breaks the FL's protocol. This situation makes FL a suitable target for data poisoning attacks.

3.2.2 Model Poisoning

Only altering the data may sometimes not be sufficient to corrupt the global model completely. As you can recall, the global model is an aggregation of all the updates submitted in any given round, implying that the poisoned model only contributes to a small portion of the global model. In some instances, it will not lead to a successful attack. One way to improve the attack's efficiency is to use the model poisoning approach. Model poisoning directly manipulates the weights and gradients of the attackers' model update to replace the influence of the benign participants [6, 7, 71]. The model updates are altered in such a way that the poisoned model replaces the original global model in just a few rounds. When the objective is byzantine, the initial model poisoning attack strategy works by changing the direction of the gradients to be opposite to the gradients of the global model, effectively preventing convergence. Similarly, for backdoor attacks, the model weight of the poisoned model is rescaled to be much larger than benign models. Such rescaling overvalues the poisoned model during aggregation, effectively replacing the global model with the poisoned model.

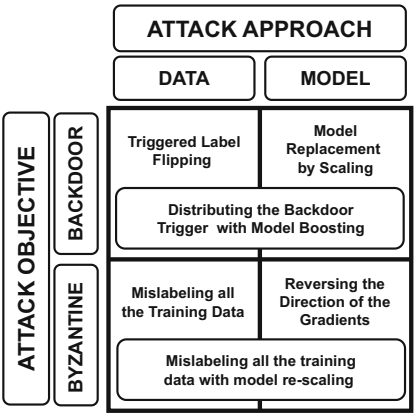


Fig. 3 An example of how the attack objectives and attack approaches are combined to execute an attack

Figure 3 shows an example of how both data and model poisoning can be combined to execute both byzantine and backdoor attacks. It is impossible to combine byzantine and backdoor attacks similarly because of their contradicting effects on the Main Task objective. A byzantine attack wants to reduce the Main Task performance, while a backdoor attack wants to maintain the performance.

4 Impact of Attacks on FL

So far, we've looked at how a security attack is carried out in an FL system. We also have an understanding of why and how the attacks are executed. Now, it is time to learn how to defend against such security attacks. But, before we can defend, it is essential to know the influence of these poisoned models based on their objective and approach. Now, let's look at how these attacked models differ from benign models perceptible by the global server on a gradient-update level. Irrespective of the objective of the attack, the attackers' gradient update should deviate from the benign models in one or both of the following ways for the attack to be effective. We will use a high-level two-dimensional representation of the model updates for demonstrations to make it easier to follow.

4.1 Angular Deviation

Whenever an attacker works toward a particular objective, be it byzantine or backdoor, the attacker tends to overtrain the model with the poisoned dataset. Such overtraining is necessary to counter the impact of the benign updates in the global

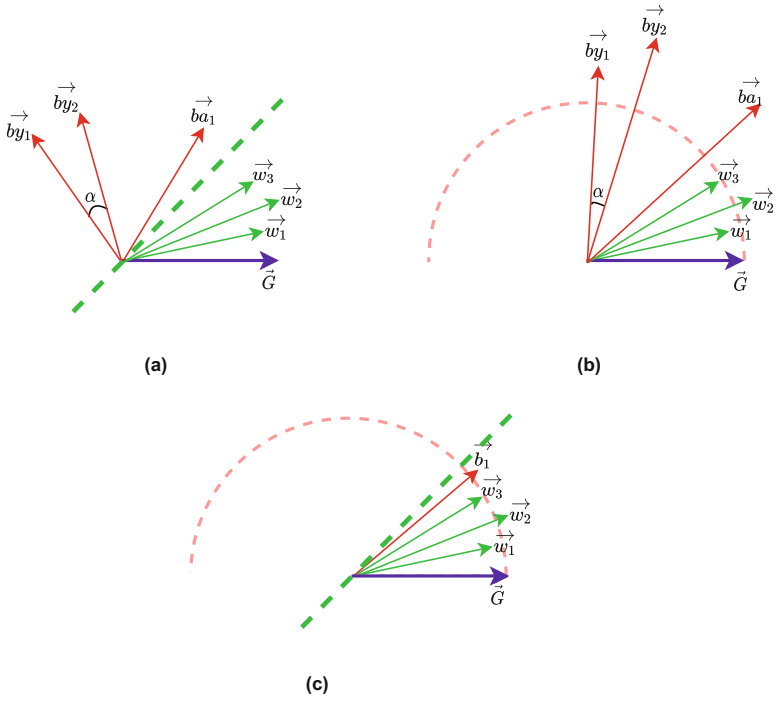


Fig. 4 (**a**) Angular deviation: the vector angle of the gradient updates of attacked models are far from the benign models. (**b**) Magnitude deviation: The attacker updates' gradients have a higher magnitude than the benign models. (**c**) Shows a weak attack that only causes a minor deviation from benign behavior: **by** is byzantine, **ba** is backdoor, **b** is byzantine/backdoor,**w** is benign, **G** is the global model gradient vectors

model [40]. Data poisoning attacks generally lead to a smaller angular deviation in the case of backdoor attacks and a relatively more significant deviation for byzantine attacks [13]. Angular deviations can be caused by the following:

- High number of local epochs when training with the poisoned dataset.
- The ratio of poisoned data in the training dataset is much larger than the benign data.
- The gradient updates are manipulated to move in the opposite direction of the global model gradients in the case of byzantine attacks.

In Fig. 4a, the green dotted line represents the approximate angular deviation limit, beyond which an update is considered harmful to the global model **G**. Here, $\overrightarrow{by_i}$ represents the byzantine model updates, and $\overrightarrow{ba_i}$ shows the backdoor model update. Generally, the byzantine model updates tend to deviate further from the benign expectation as they oppose the global model's Main Task performance. The angle α between $\overrightarrow{by_1}$ and $\overrightarrow{by_2}$ shows that the attackers working together build similar poisoned models. If this angle is large or random, then there is a chance for the malicious updates to cancel out each other.

Many defense strategies use some form of clustering approach to segregate benign updates and poisoned updates. The poisoned updates are usually discarded upon separation, while the benign updates are used from aggregation. The issue with using angular deviation to discard the poisoned models is the possibility of omitting benign models, which might be an outlier. If the data distribution of a client is highly Non-IID, or if the training data features are unique compared to the other clients, it could lead to the client's model updates deviating from the expected range. If, unfortunately, such outliers are considered anomalous and discarded, then the global model does not learn those unique features, which is a loss for all the clients.

4.2 Magnitude Deviation

Magnitude deviation is more often a result of model poisoning than data poisoning. Whenever an adversary aims to amplify its presence in the global model or replace it with the poisoned local model [2, 40], the adversary can scale up its local gradient updates before submitting it to the server. The primary cause for magnitude deviation is:

- Scaling up the poisoned model in backdoor attacks, usually after minor angular deviation.
- Modifying the gradient updates to move it further from convergence in byzantine attacks. Scaling is used in byzantine attacks to undermine the contributions of benign clients.

In Fig. 4b, the red dotted semicircle represents the magnitude threshold of the gradient updates, beyond which the update would be considered an attack. The defense against magnitude deviation mainly involves clipping the updates to ensure that it matches the rest of the updates submitted by the clients. Though clipping does not entirely remove the poisoned update, it manages to mitigate the influence of the malicious model in the aggregation process.

4.3 Minor Deviation

If an attacker does not want to be detected, he will be more cautious about the extent of angular and magnitude deviations. To be undetected, the attacker must alter their malicious models to stay within the threshold values. Assuming that the defense method has found the best threshold to limit the influence of the poisoning model, the influence of the attack model will be minimal. Such a minimal influence ensures a negligible attack success rate. Even if the attacker manages to induce an attack with minor deviations, its effect will deteriorate over the next few rounds, making the global model benign again [17]. Fig. 4c shows a weak attack, which is not highly deviant from the expected benign behavior.

5 Common Defense Methods

Since the behavior of a malicious model has peculiar traits that can be used to distinguish them, researchers have developed some common defense strategies to mitigate the effect of the attack. Let's look at some common defense steps applied to most known attacks.

5.1 Clustering

When the attacker's model tries to move the global model away from its Main Task objective, it tends to have a sizeable angular deviation. This angular deviation can be used to group and separate benign behavior from malicious behavior. Clustering is the technique used to carry out this separation [21, 28, 40, 47]. Clustering works by grouping so that more than 50% of the participants who are closer in terms of model updates are grouped together and assigned to be the benign group. Such an assignment is acceptable because of the general research assumption that only less than 50% of the participants are malicious. The fate of the members in other clusters will be decided by the defense method, with many methods discarding the suspected malicious cluster. Figure 5 shows a toy example of how clustering works.

5.1.1 Downsides of Clustering

This defense strategy has used many clustering methods, including k-means clustering, DBSCAN, and HDBSCAN. The parameters used for each technique play a crucial role because of the possibility of removing unique benign client updates. If the clustering rule is too strict, it could lead to a higher false positive rate, and if the rule is not strict enough, it leads to a high false negative rate. Similarly, clustering techniques can be detrimental when the client data distributions are highly non-IID. Non-IID distributions lead to more pronounced angular differences between the benign clients. In such cases, if the malicious participants manage to increase their similarity with some benign clients, they could be falsely considered benign.

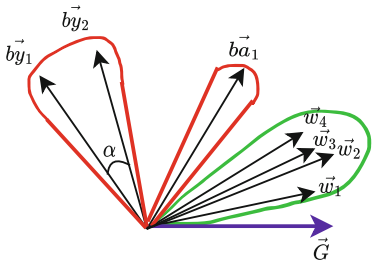


Fig. 5 Toy example of clustering

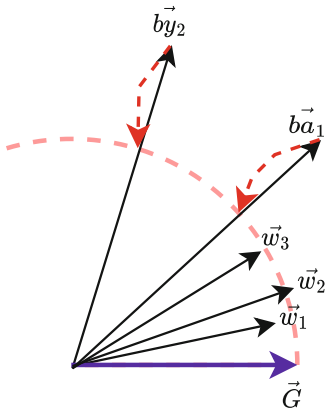


Fig. 6 Toy example of clipping

5.2 Clipping

Similar to how clustering defends against angular deviation, clipping defends against magnitude deviation. This method works by selecting a threshold value for the magnitude and clipping the scale of the updates whose magnitudes go beyond this threshold value [17, 40, 44, 47]. Unlike clustering, where the potentially malicious model updates are entirely discarded, many clipping-based methods rescale the possible malicious update to match the threshold value. This alleviates the impact of the poisoned update in the global model. Figure 6 shows a toy example of how clipping is applied.

5.2.1 Downsides of Clipping

When it comes to selecting the threshold value, many defense methods take the median magnitude of all the submitted updates as the threshold. This simple approach leads to the rescaling of many of the benign updates as well. If the defense fixes the threshold to be static throughout the FL process, then the clipping strategy will not accommodate the dropping magnitude values as the model gets closer to convergence. Clever attackers use this situation to try to match the magnitude of their poison update to the threshold value to avoid detection. To prevent this, the defenders can also choose to have a dynamic threshold value that changes for each round of the FL process. It is generally challenging to select a suitable threshold that does not impact the benign updates, even under non-IID conditions, while only targeting poisoned models.

5.3 Similarity Checking

Similar to clustering, similarity checking also relies on the intention of the attackers to be closer together. Malicious clients working together with the same objective exhibit a high degree of similarity. The angle between malicious gradient updates (α in Fig. 4a) will be smaller than that between benign and malicious updates. This feature is captured using cosine similarity or other such approaches. Highly similar models' contributions are neglected [13, 14, 59].

5.3.1 Downsides of Similarity Checking

Though this is a simple approach, it can be computationally costly. In many cases, the similarity checking is carried out pairwise, which would increase exponentially with the increase in the number of clients. Since similarity checking and clustering capture the angular deviation, clustering is more efficient unless the number of clients is small. Similarity checking might not be efficient in detecting backdoor attacks, as they are predominantly focused on byzantine attacks.

5.4 Noise Addition

Another way to defend against poisonous models is to add Gaussian noise to all the client updates [23, 40, 51]. This method is similar to applying a differential privacy strategy to the model updates. This technique is usually effective against backdoor attacks but not against byzantine attacks. Byzantine attacks do not have any performance objective other than to disrupt the Main Task objective. So, adding noise would not have a significant impact on the attack. On the other hand, backdoor attacks try their best to maintain the Main Task objective while simultaneously trying to achieve the backdoor objective. So, backdoor model updates need to be as precise as possible to achieve both tasks. Adding noise to the model update effectively reduces the precision of the backdoor model. The reduced precision leads to a reduction in both Main Task and backdoor performance. But, during the aggregation process, the Main Task performance is improved because of the contribution from all the clients. In contrast, the backdoor performance is effectively nullified. Figure 7 shows a toy example demonstrating how noise addition functions.

5.4.1 Downsides of Noise Addition

The noise level is the key parameter needed for noise addition to work as expected. Selecting the appropriate noise level suitable for all the client updates is crucial. If the noise level is l too low, it might not be enough to remove the backdoor. On the

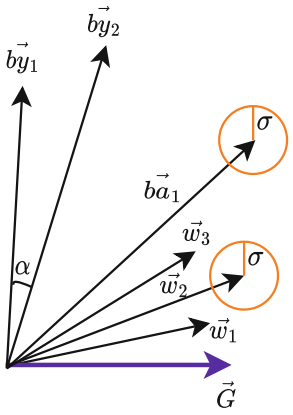


Fig. 7 Toy example of noise addition

contrary, if the noise level is too high, it would negatively impact the overall Main Task performance of all the clients, affecting the global model. Thus, each defense strategy selects a different approach to determine the noise level.

5.5 Robust Aggregation

FedAvg performs a simple weighted averaging to build the global model. This vanilla aggregation method provides no defense against security attacks. So, many researchers have focused on modifying the aggregation scheme to make it more robust against security threats [16, 26, 27, 43]. The robust aggregation method works better with a byzantine attack than a backdoor attack. Let's look at some of how robust aggregation is applied:

5.5.1 Selecting a Subset from Submitted Models

Instead of aggregating all the received models, the server uses some method to select a smaller subset of updates that form the global model. In Krum[3], only one of the local models submitted by the client, which is closest to most other models, is selected as the global model. Other methods like [36] add another parameter trimming step to improve Krum.

5.5.2 Truncating the Weights of the Updates

In this approach, the global server removes the extreme values in the model weights from contributing to the global model. In Trimmed Mean [65], each model

parameter is independently sorted, and the extreme values are removed before calculating the mean as the global model parameter.

5.5.3 Replacing Simple Averaging

Federated Averaging and other similar averaging methods make it easier for attackers to replace the global model with their attacked model by scaling up their updates. Replacing the averaging strategy with a different approach, like using the geometric mean for aggregation [45].

5.5.4 Downsides of Robust Aggregation

Depending on how strict the aggregation algorithm is, robust aggregation methods tend to affect the Main Task performance negatively. Moreover, many of the robust aggregation schemes work well against traditional byzantine attacks but not with backdoor attacks. Some older yet widely used robust aggregation algorithms work well only with IID datasets. If the dataset is highly non-IID, truncation or subset selection would lead to losing precious information.

6 Some State-of-the-Art Backdoor Defense Techniques

This section will examine some existing SOTA defenses against backdoor attacks on FL. We can categorize these defenses based on who contributes to the defense process.

6.1 Protocol-Level Defenses

In this category, the server and the clients work together to defend against the backdoor attack. The whole FL process is modified to accommodate the additional defense steps. The clients' involvement could range anywhere from verifying the global model to voting to detect the backdoor model.

In [1], the server employs the clients and leverages the diverse dataset available with each client to detect the presence of backdoor attacks in the global model. This FL process includes a feedback loop that allows the clients to test the model sent from the server against their local datasets and predict the presence of a backdoor attack. This method ensures a high detection rate and a low false positive rate. In [37], the server requires the clients to rank the randomly initialized parameters sent to them by the server. The clients use their local training data to rank the parameters and send it back to the server, where the server aggregates the parameter rankings using a voting scheme.

6.2 Server-Level Defenses

This is the most common type of backdoor defense strategy. In this method, clients do not contribute to the defense in any way. The clients follow the regular FL steps, as they would with FedAvg aggregation, without any change to the process. But, the server applies a series of strategies to detect the presence of backdoor model updates. The most efficient server defenses use a combination of clustering, clipping, and noise addition to mitigate the presence of the backdoor completely.

In [47] and [40], the central server is solely responsible for mitigating the attack without the help of the clients. Both these methods use some combination of clustering and clipping to remove the malicious updates present in the system. In [13], a similarity checking method is used to find the similarity in consecutive model updates to differentiate byzantine malicious clients from benign clients. In [17], the authors change the threshold used for the clipping process in each round of the FL process based on the extent of convergence.

6.3 Client-Level Defenses

One of the reasons to use the FL approach is to prevent the central server from accessing the client's private data. If data privacy is not a concern, the clients can directly send their datasets to the central server. The existence of FL proves that the central server is not entirely trustworthy. If the clients do not trust the central server with their privacy, why should they trust the central server with their security? So far, there have not been any client-level defenses exclusively for FL setup. Still, the clients can extend some of the backdoor defenses from the traditional ML approach for their defense. In client-level defense, each client can autonomously protect their models from backdoor attacks.

NeuralCleanse [54] is a trigger reconstruction technique used to detect the features in an input that are responsible for the prediction outcome. If a trigger was responsible for predicting a specific input, then the responsible feature would be tiny and can be distinguished as a backdoor trigger. In [31], the authors treat the trigger pattern recovery problem as an unknown noise distribution extraction problem. Additionally, they also reverse the backdoor injection procedure and force the model to unlearn the malicious injection. Though these are independent ML defense methods, clients can adapt them in an FL system with acceptable success.

7 Opportunities and Future Directions

Though security in FL is a widely studied field of Machine Learning, it is still in the early stages of wide-scale adoption. Such widespread usage would bring further

problems that need to be analyzed and addressed. In this section, let's look at some of the regions that can use further research in FL security in the future.

7.1 Beyond Text and Image

So far, the attacks and the defenses in FL research have mainly been focusing on the models which only work with either text or image domains. Federated Learning has the potential to be applied to the audio and video domains as well, but the research focusing on these domains is insufficient. If researchers do not focus on these domains, any real-world attack being executed in these domains might go undetected.

7.2 Beyond Single-Domain

Federated Learning is usually applied only to build models that handle one domain of input, either text or image. Given the rapid growth of AI, the models are evolving to handle multi-domain inputs. The security requirements of multi-domain models differ from those of single-domain models. Handling the security aspects of multi-domain FL is an important future direction.

7.3 Beyond Security Impacts

Currently, the privacy and security of FL are treated as two independent issues. But, both privacy defense and security defense significantly alter the FL process without analyzing its impact on the other issue. It is important to understand the privacy implications of a security defense and ensure that the enhanced security does not lead to excessive privacy leakage.

7.4 Beyond Horizontal Federated Learning

The majority of the existing research works primarily focus on the horizontal federated learning setup. Due to its popularity and earlier implementation, HFL is more widely used than VFL, and hence the imbalance in the research focus is justified. But, this situation may not persist; VFL could also have broader applications in the near future.

7.5 Need for Client-Level Defenses

As mentioned earlier, there are not many client-level defenses designed specifically for the FL setup. Considering that the server could also act as a malicious party, it is evident that the clients need an effective and efficient defense mechanism. Extending regular ML solutions to FL is a costly process, further emphasizing the need to client-level protection.

8 Conclusion

In this chapter, we gave a systematic review of the security issues in Federated Learning systems. We started with the basics of Federated Learning, followed by the need for FL, application of FL, the application of FL, the workflow of a standard FL, and other factors that influence its performance. Similarly, we briefly discussed the common threat model, and the privacy and free-rider issues in FL. In the deep dive we took into the security threats to an FL system, we analyzed the types and impacts of the attacks. We also learned how the defense methods utilize these impacts to design necessary modifications to the aggregation process. From our analysis, we can understand that Federated Learning will continue to grow, and it is essential to be prepared to defend against new potential security threats that may occur later on. In the future, the security concerns of FL cannot be treated as an independent entity but as a crossover of all the aspects of FL.

References

1. Andreina S, Marson GA, Möllering H, Karame G (2021) Baffle: backdoor detection via feedback-based federated learning. In: 2021 IEEE 41st international conference on distributed computing systems (ICDCS), IEEE, pp 852–863
2. Bagdasaryan E, Veit A, Hua Y, Estrin D, Shmatikov V (2020) How to backdoor federated learning. In: International conference on artificial intelligence and statistics, PMLR, pp 2938–2948
3. Blanchard P, El Mhamdi EM, Guerraoui R, Stainer J (2017) Machine learning with adversaries: byzantine tolerant gradient descent. In: Proceedings of the 31st international conference on neural information processing systems, pp 118–128
4. Briggs C, Fan Z, Andras P (2020) Federated learning with hierarchical clustering of local updates to improve training on non-iid data. In: 2020 international joint conference on neural networks (IJCNN). IEEE, pp 1–9
5. Bukaty P (2019) The California consumer privacy act (CCPA): an implementation guide. IT Governance Publishing. http://www.jstor.org/stable/j.ctvjghvnn
6. Cao X, Gong NZ (2022) Mpaf: model poisoning attacks to federated learning based on fake clients. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 3396–3404

7. Cao D, Chang S, Lin Z, Liu G, Sun D (2019) Understanding distributed poisoning attack in federated learning. In: 2019 IEEE 25th international conference on parallel and distributed systems (ICPADS). IEEE, pp 233–239
8. European Commission (2016) Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation) (Text with EEA relevance). https://eur-lex.europa.eu/eli/reg/2016/679/oj
9. Fang M, Cao X, Jia J, Gong N (2020) Local model poisoning attacks to {Byzantine-Robust} federated learning. In: 29th USENIX security symposium (USENIX Security 20), pp 1605–1622
10. Fang W, Zhao D, Tan J, Chen C, Yu C, Wang L, Wang L, Zhou J, Zhang B (2021) Large-scale secure xgb for vertical federated learning. In: Proceedings of the 30th ACM international conference on information & knowledge management, pp 443–452
11. Fraboni Y, Vidal R, Lorenzi M (2021) Free-rider attacks on model aggregation in federated learning. In: International conference on artificial intelligence and statistics, PMLR, pp 1846–1854
12. Fu C, Zhang X, Ji S, Chen J, Wu J, Guo S, Zhou J, Liu AX, Wang T (2022) Label inference attacks against vertical federated learning. In: 31st USENIX security symposium (USENIX security 22), pp 1397–1414
13. Fung C, Yoon CJ, Beschastnikh I (2018) Mitigating sybils in federated learning poisoning. Preprint, arXiv:180804866
14. Fung C, Yoon CJ, Beschastnikh I (2020) The limitations of federated learning in sybil settings. In: 23rd international symposium on research in attacks, intrusions and defenses ({RAID} 2020), pp 301–316
15. Geng J, Mou Y, Li Q, Li F, Beyan O, Decker S, Rong C (2023) Improved gradient inversion attacks and defenses in federated learning. IEEE Trans Big Data, 1–13. https://doi.org/10.1109/TBDATA.2023.3239116
16. Ghosh A, Hong J, Yin D, Ramchandran K (2019) Robust federated learning in a heterogeneous environment. Preprint, arXiv:190606629
17. Guo Y, Wang Q, Ji T, Wang X, Li P (2021) Resisting distributed backdoor attacks in federated learning: a dynamic norm clipping approach. In: 2021 IEEE international conference on big data. IEEE, pp 1172–1182
18. Hard A, Rao K, Mathews R, Ramaswamy S, Beaufays F, Augenstein S, Eichner H, Kiddon C, Ramage D (2018) Federated learning for mobile keyboard prediction. Preprint, arXiv:181103604
19. Huang Y, Gupta S, Song Z, Li K, Arora S (2021) Evaluating gradient inversion attacks and defenses in federated learning. Adv Neural Inf Process Syst 34:7232–7241
20. Huang W, Li T, Wang D, Du S, Zhang J, Huang T (2022) Fairness and accuracy in horizontal federated learning. Inf Sci 589:170–185
21. Jebreel NM, Domingo-Ferrer J, Sánchez D, Blanco-Justicia A (2022) Defending against the label-flipping attack in federated learning. Preprint, arXiv:220701982
22. Jordan MI, Mitchell TM (2015) Machine learning: trends, perspectives, and prospects. Science 349(6245):255–260
23. Kairouz P, Liu Z, Steinke T (2021) The distributed discrete gaussian mechanism for federated learning with secure aggregation. In: International conference on machine learning, PMLR, pp 5201–5212
24. Khan LU, Saad W, Han Z, Hossain E, Hong CS (2021) Federated learning for internet of things: recent advances, taxonomy, and open challenges. IEEE Commun Surv Tutor 23(3):1759–1799
25. Khan A, ten Thij M, Wilbik A (2022) Communication-efficient vertical federated learning. Algorithms 15(8):273

26. Li S, Cheng Y, Wang W, Liu Y, Chen T (2020) Learning to detect malicious clients for robust federated learning. Preprint, arXiv:200200211
27. Li T, Hu S, Beirami A, Smith V (2021) Ditto: fair and robust federated learning through personalization. In: International conference on machine learning, PMLR, pp 6357–6368
28. Li D, Wong WE, Wang W, Yao Y, Chau M (2021) Detection and mitigation of label-flipping attacks in federated learning systems with kpca and k-means. In: 2021 8th international conference on dependable systems and their applications (DSA). IEEE, pp 551–559
29. Li J, Rakin AS, Chen X, He Z, Fan D, Chakrabarti C (2022) Ressfl: a resistance transfer framework for defending model inversion attack in split federated learning. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 10194–10202
30. Lin J, Du M, Liu J (2019) Free-riders in federated learning: attacks and defenses. Preprint, arXiv:191112560
31. Liu Y, Fan M, Chen C, Liu X, Ma Z, Wang L, Ma J (2022) Backdoor defense with machine unlearning. In: IEEE INFOCOM 2022-IEEE conference on computer communications. IEEE, pp 280–289
32. Long G, Tan Y, Jiang J, Zhang C (2020) Federated learning for open banking. In: Federated learning: privacy and incentive. Springer, Berlin, pp 240–254
33. Luo X, Wu Y, Xiao X, Ooi BC (2021) Feature inference attack on model predictions in vertical federated learning. In: 2021 IEEE 37th international conference on data engineering (ICDE). IEEE, pp 181–192
34. McCulloch WS, Pitts W (1943) A logical calculus of the ideas immanent in nervous activity. Bull Math Biophys 5:115–133
35. McMahan B, Moore E, Ramage D, Hampson S, y Arcas BA (2017) Communication-efficient learning of deep networks from decentralized data. In: Artificial intelligence and statistics, PMLR, pp 1273–1282
36. Mhamdi EME, Guerraoui R, Rouault S (2018) The hidden vulnerability of distributed learning in byzantium. Preprint, arXiv:180207927
37. Mozaffari H, Shejwalkar V, Houmansadr A (2023) Every vote counts: ranking-based training of federated learning to resist poisoning attacks. In: 32nd USENIX security symposium (USENIX Security 23)
38. Nasr M, Shokri R, Houmansadr A (2019) Comprehensive privacy analysis of deep learning: passive and active white-box inference attacks against centralized and federated learning. In: 2019 IEEE symposium on security and privacy (SP). IEEE, pp 739–753
39. Nguyen DC, Ding M, Pathirana PN, Seneviratne A, Li J, Poor HV (2021) Federated learning for internet of things: a comprehensive survey. IEEE Commun Surv Tutor 23(3):1622–1658
40. Nguyen TD, Rieger P, Chen H, Yalame H, Möllering H, Fereidooni H, Marchal S, Miettinen M, Mirhoseini A, Zeitouni S, Zeitouni S, Koushanfar F, Sadeghi A-R, Schneider T (2022) FLAME: taming backdoors in federated learning. In: 31st USENIX security symposium (USENIX Security 22). USENIX Association, Boston, pp 1415–1432. ISBN: 978-1-939133-31-1. https://www.usenix.org/conference/usenixsecurity22/presentation/nguyen
41. Niknam S, Dhillon HS, Reed JH (2020) Federated learning for wireless communications: motivation, opportunities, and challenges. IEEE Commun Mag 58(6):46–51
42. Nuding F, Mayer R (2022) Data poisoning in sequential and parallel federated learning. In: Proceedings of the 2022 ACM on international workshop on security and privacy analytics, pp 24–34
43. Ozdayi MS, Kantarcioglu M, Gel YR (2021) Defending against backdoors in federated learning with robust learning rate. In: Proceedings of the AAAI conference on artificial intelligence, vol 35, pp 9268–9276
44. Panda A, Mahloujifar S, Bhagoji AN, Chakraborty S, Mittal P (2022) Sparsefed: mitigating model poisoning attacks in federated learning with sparsification. In: International conference on artificial intelligence and statistics, PMLR, pp 7587–7624

45. Pillutla K, Kakade SM, Harchaoui Z (2019) Robust aggregation for federated learning. Preprint, arXiv:191213445
46. Posner J, Tseng L, Aloqaily M, Jararweh Y (2021) Federated learning in vehicular networks: opportunities and solutions. IEEE Netw 35(2):152–159
47. Rieger P, Nguyen TD, Miettinen M, Sadeghi AR (2022) Deepsight: mitigating backdoor attacks in federated learning through deep model inspection. Preprint, arXiv:220100763
48. Rieke N, Hancox J, Li W, Milletari F, Roth HR, Albarqouni S, Bakas S, Galtier MN, Landman BA, Maier-Hein K, et al (2020) The future of digital health with federated learning. NPJ Digit Med 3(1):119
49. Shejwalkar V, Houmansadr A (2021) Manipulating the byzantine: optimizing model poisoning attacks and defenses for federated learning. In: NDSS
50. Su Z, Wang Y, Luan TH, Zhang N, Li F, Chen T, Cao H (2021) Secure and efficient federated learning for smart grid with edge-cloud collaboration. IEEE Trans Ind Inf 18(2):1333–1344
51. Sun Z, Kairouz P, Suresh AT, McMahan HB (2019) Can you really backdoor federated learning? Preprint, arXiv:191107963
52. Sun X, Tang Z, Du M, Deng C, Lin W, Chen J, Qi Q, Zheng H (2022) A hierarchical federated learning-based intrusion detection system for 5g smart grids. Electronics 11(16):2627
53. Tolpegin V, Truex S, Gursoy ME, Liu L (2020) Data poisoning attacks against federated learning systems. In: Computer security–ESORICS 2020: 25th European symposium on research in computer security, ESORICS 2020, Guildford, UK, September 14–18, 2020, Proceedings, Part I 25. Springer, pp 480–501
54. Wang B, Yao Y, Shan S, Li H, Viswanath B, Zheng H, Zhao BY (2019) Neural cleanse: identifying and mitigating backdoor attacks in neural networks. In: 2019 IEEE symposium on security and privacy (SP). IEEE, pp 707–723
55. Wang Z, Song M, Zhang Z, Song Y, Wang Q, Qi H (2019) Beyond inferring class representatives: user-level privacy leakage from federated learning. In: IEEE INFOCOM 2019-IEEE conference on computer communications. IEEE, pp 2512–2520
56. Wang H, Kaplan Z, Niu D, Li B (2020) Optimizing federated learning on non-iid data with reinforcement learning. In: IEEE INFOCOM 2020-IEEE conference on computer communications. IEEE, pp 1698–1707
57. Wang H, Sreenivasan K, Rajput S, Vishwakarma H, Agarwal S, Sohn Jy, Lee K, Papailiopoulos D (2020) Attack of the tails: yes, you really can backdoor federated learning. Preprint, arXiv:200705084
58. Wang J, Chang X, Rodrìguez RJ, Wang Y (2022) Assessing anonymous and selfish free-rider attacks in federated learning. In: 2022 IEEE symposium on computers and communications (ISCC). IEEE, pp 1–6
59. Wang Z, Kang Q, Zhang X, Hu Q (2022) Defense strategies toward model poisoning attacks in federated learning: a survey. In: 2022 IEEE wireless communications and networking conference (WCNC). IEEE, pp 548–553
60. Wei W, Liu L, Loper M, Chow KH, Gursoy ME, Truex S, Wu Y (2020) A framework for evaluating gradient leakage attacks in federated learning. Preprint, arXiv:200410397
61. Wei K, Li J, Ma C, Ding M, Wei S, Wu F, Chen G, Ranbaduge T (2022) Vertical federated learning: challenges, methodologies and experiments. Preprint, arXiv:220204309
62. Xie C, Huang K, Chen PY, Li B (2019) Dba: distributed backdoor attacks against federated learning. In: International conference on learning representations
63. Xu J, Glicksberg BS, Su C, Walker P, Bian J, Wang F (2021) Federated learning for healthcare informatics. J Healthcare Inf Res 5:1–19
64. Yang T, Andrew G, Eichner H, Sun H, Li W, Kong N, Ramage D, Beaufays F (2018) Applied federated learning: improving google keyboard query suggestions. Preprint, arXiv:181202903
65. Yin D, Chen Y, Kannan R, Bartlett P (2018) Byzantine-robust distributed learning: towards optimal statistical rates. In: International conference on machine learning, PMLR, pp 5650–5659

66. Zeng T, Semiari O, Chen M, Saad W, Bennis M (2022) Federated learning on the road autonomous controller design for connected and autonomous vehicles. IEEE Trans Wirel Commun 21(12):10407–10423
67. Zhang J, Zhang J, Chen J, Yu S (2020) Gan enhanced membership inference: a passive local attack in federated learning. In: ICC 2020-2020 IEEE international conference on communications (ICC). IEEE, pp 1–6
68. Zhao Y, Li M, Lai L, Suda N, Civin D, Chandra V (2018) Federated learning with non-iid data. Preprint, arXiv:180600582
69. Zhao J, Zhu X, Wang J, Xiao J (2021) Efficient client contribution evaluation for horizontal federated learning. In: ICASSP 2021-2021 IEEE international conference on acoustics, speech and signal processing (ICASSP). IEEE, pp 3060–3064
70. Zhou ZH (2021) Machine learning. Springer Nature, Berlin
71. Zhou X, Xu M, Wu Y, Zheng N (2021) Deep model poisoning attack on federated learning. Future Internet 13(3):73

Lessons Learned and Future Directions for Security, Resilience and Artificial Intelligence in Cyber Physical Systems

J. Sukarno Mertoguno, Gregory Briskin, Jason H. Li, and Kyung Kwak

1 Introduction

Cyber physical systems (CPS) underlie many critical infrastructures and are prevalent across a wide range of areas including the electrical grid, factory production pipeline, machinery control, vehicular control, internet-of-things (IOT) devices, and commodity toy drones, just to name a few. By its nature, a CPS straddles the continuous-time physical domain and the discrete-time digital or cyber domain. Cyber components (e.g., communication and computing) couple with physical components (e.g., sensors and actuators) to carry out the intended functions of the CPS.

Cyber physical systems are required to satisfy safety constraints in various application domains such as robotics, unmanned vehicles (e.g., aerial or ground), industrial manufacturing systems, and power systems. However, the once isolated system of computer-controlled machinery is now more exposed to the external world than ever, which renders ample opportunities of remote system disruption via cyber threats, in addition to the tradition threat of a physical component failing. Both may result in safety violations.

Current emphasis on cyber security of CPS is on securing the operational technology (OT) network. For example, National Institute of Standards and Technology (NIST) devoted its guidance for securing CPS, SP 800-82 Rev.3 [1], solely to network security with network segmentation as the primary recommended solution. However, network or communication is only one facet of CPS. While network is an

J. S. Mertoguno
School of Cybersecurity and Privacy, Georgia Institute of Technology, Atlanta, GA, USA
e-mail: karno@gatech.edu

G. Briskin · J. H. Li (✉) · K. Kwak
Trusted Science and Technology Inc., Rockville, MD, USA
e-mail: greg@trustedst.com; jason@trustedst.com; kj@trustedst.com

Y. Chen et al. (eds.), *Network Security Empowered by Artificial Intelligence*,
Advances in Information Security 107, https://doi.org/10.1007/978-3-031-53510-9_15

important CPS and critical infrastructure component, over-emphasis on networking security will not be sufficient for defending the underlying CPS and its infrastructure against motivated and well-resourced adversaries. A recent article indicates that the assumption of malicious events entering the system solely via the external network (hence the need for network segmentation) has been invalidated [2]. The exploits discussed in the article did enter the system through the local (internal) network and propagated within the internal bus, avoiding the security protection provided by network segmentation. A holistic view and approach for defending CPS is needed.

2 Physical Domain and Cyber Domain

In CPS, the ultimate goal is for the overall system to be stable and function as intended. A *resilient CPS* is expected to physically operate properly and in a predictable and controllable manner under ever-present external and environmental disturbance as well as adversarial cyber exploits. The objectives and emphasis for CPS resilience are *physical* stability and functionality. Cyber components and systems in CPS are means toward the end of achieving CPS resilience. The stability of cyber systems by itself is not the primary objective.

A cyber physical system contains cyber components that interact with and control the behavior of the physical system operating in a physical environment. Generally speaking, the cyber controller periodically samples the operation (or mission) objective (e.g., the expected set value of speed), measures the actual values of physical variables via sensors (e.g., speed, altitude), contrasts measurements against the objective, and calculates the magnitude of control variables, which translates to direction and/or force to be asserted by the actuator onto the physical environment. Figure 1 shows an example of how the physical and cyber components interact in a particular cyber physical system—a robotic aerial vehicle (RAV) or drone.

It is worthwhile to point out the differences of physical and cyber components and (sub)systems, and the potential opportunities they may offer for building resilient cyber physical systems. The physical platform and the subsystems operate in a physical environment at physical speed (and time), governed by the laws of physics. The mass and dynamics of a physical implementation define its moment of *inertia*, which in turn influences the *response time* of the physical subsystems and the platform. Any physical subsystem of a CPS must obey the laws of physics, and the physical systems invariably have inertia. In essence, **physics rules**.

CPS physical and cyber components differ significantly in the scale of their response time. Physical and mechanical components have relatively large time scales (low frequency), in the order of milliseconds and seconds. A heavier object has larger inertia and hence lower frequency (see section “Byzantine Fault Tolerance++ (BFT++)” for detailed description of inertia). For example, a large tanker vessel takes minutes to change its direction.

The cyber components operate at cyber speed, typically multiple orders of magnitude faster than that of physical components. The scan period of a CPS

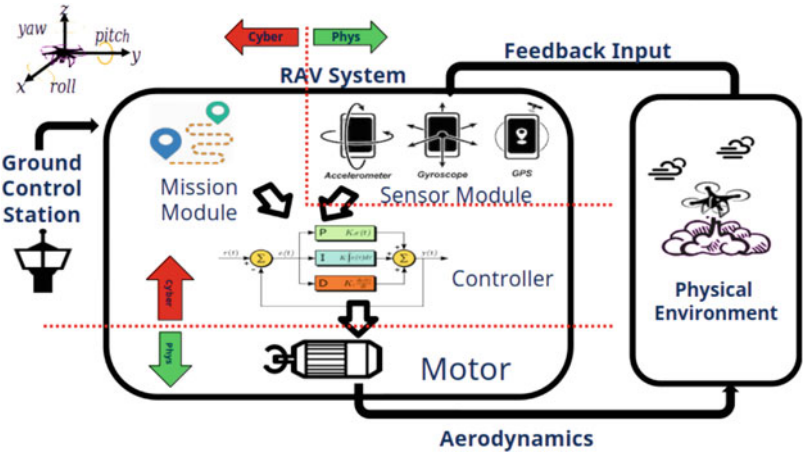


Fig. 1 Cyber Domain and Physical Domain in CPS, derived from MayDay [3]

controller (1–500 Hz) is usually about one or two orders of magnitude smaller (faster) than that of the physical/mechanical machinery [4]. The clock speed of the controller CPU (GHz) is generally five to seven orders of magnitude faster than the scan cycle. The physical micro-mechanical sensing mass within a micro-electro-mechanical systems (MEMS) inertia measurement unit (IMU) has resonance frequency measured in KHz, in the 10–30 KHz range [5], still one or two orders of magnitude faster than the controller's scan cycle of a drone.

Traditionally CPS researchers have been focusing on achieving *cyber stability*, which generally provides physical stability within the designed region of operation. This is definitely a prudent design methodology (see the first quadrant in Fig. 2a with P(S) and C(S) denoting physical and cyber stability, respectively).

However, a cyber physical system may have to operate in a physical environment with disturbance so large (e.g., strong wind gust or other physical impact) that makes the controller algorithms or other cyber components struggle to work while out of the designed region of operation. Extended Kalman Filter (EKF) and robust control algorithms usually work effectively to absorb and tolerate relatively small disturbance, but the physical subsystems and overall platform may fail to maintain physical stability under large disturbance. This is quite interesting: cyber components work as designed but the physical systems are unstable, see the second quadrant in Fig. 2a with P(U) and C(S) denoting physical instability and cyber stability, respectively, which indicates that the traditional focus for cyber stability is not always sufficient or effective. The community starts to notice this important realization, and leaders start to investigate alternatives to designing resilient cyber physical systems, such as the DARPA LINC program [6] and the DARPA FIRE program [7].

Particularly, since we argue that the goal of CPS resilience is physical stability and not necessarily cyber stability, cyber components controlling physical com-

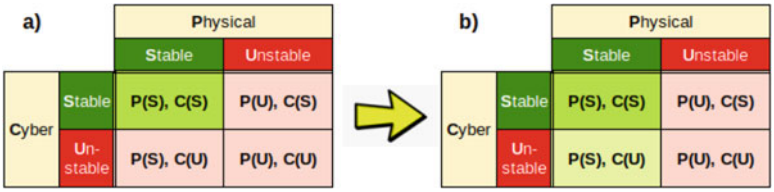


Fig. 2 Desired CPS operation space: (**a**) Cyber-centric, (**b**) Physical-centric

ponents only need to be stable at the scale of the frequency and response time of the physical components. Focusing on physical stability, therefore, opens up another design space for CPS resilience as the fourth quadrant in Fig. 2b, with P(S) and C(U) denoting physical stability and cyber instability, respectively. It is important to emphasize again that C(U) means being unstable but also unnoticeable by physical components, not being unstable all the time for obvious reasons. Empowered by physical inertia and the differences in response time for physical and cyber components, the fourth quadrant represents a previously less-explored design space for achieving CPS security and resilience. Section 2.2.6 explores cyber-attack resilient CPS design within the fourth quadrant.

2.1 System Model and Control in CPS

Modeling is essential to every scientific and engineering enterprise. For both scientists and engineers, the "thing being modeled" (referred to as *target*) is typically an object, process, or system in the physical world. But it could also be another model as manifested in model refinement for formal verification. A "model" of a target is any description of the target that is not Kant's *thing-in-itself*. For example, mechanical engineers use Newton's laws as models for how a system will react to forces. Computer engineers model digital circuits as instruction set architectures (ISAs), programs as executions in an ISA, and applications as networks of program fragments [8]. Each of these models rests on a modeling paradigm. For example, a source code is a model of what a machine should do when it executes the program, but the source code is not what is actually run on a machine. The Java programming language, for example, is just such a modeling paradigm. Models abstract away details, and layers of models may be built on top of another. A CPS system consists of such layered models from hardware all the way up to applications it runs.

The *fidelity* of a model is the degree to which it emulates the target. When the target is a physical object, process, or system, model fidelity is never perfect. But as stated in reference [9], "essentially, all models are wrong, but some are useful". As highlighted in reference [8], in science the value of a model lies in how well its properties match those of the target, whereas in engineering the value of the target lies in how well its properties match those of the model. A scientist constructs

models to help understand the target. An engineer constructs targets to emulate the properties of a model, since for an engineer a model represents a design and the target is the implementation. These two uses of models are complementary.

For CPS modeling and control, therefore, it is critical to always keep a clear mind in terms of the *thing*, the *model*, the *purpose* of the model, and the *interactions* between the thing and the model in either a science or engineering context. For example, simplicity and clarity of target semantics may dominate over accuracy and detail, and optimizing over a model does not necessarily bring about desired effects or benefits to the target.

Moreover, it is important to note that models (and analyses and controls over these models) have their inherent *region of operation*, a concept commonly known in each individual disciplines but unfortunately often ignored in real-world practices. This is particularly true in CPS where multi-layer models exist, and their interactions lack sufficient attention. The re-invigoration of AI/ML makes this awareness even more relevant, in terms of where and when AI/ML could help analyze and even take over some control of the cyber physical system without adversely affecting the physical or cyber operation stability. Special care must be taken to understand the boundary of each model, interactions among models, appropriate positioning of AI/ML models and algorithms, and anticipated and measurable effects in the physical world.

2.2 CPS-Specific Cyber Security Challenges and Solutions

Traditional cyber defense for CPS has mainly focused on the level of human-machine interface (HMI) and security information and event management system (SIEM). This is largely due to the similarity to established cyber protections for hosts and networks, and the information technology (IT) mindset possessed by practitioners. However, this leaves the lower-levels of the cyber physical system vulnerable to attacks not common in a traditional IT environment.

Protection of low-level components and subsystems includes protecting the interconnect and computation or logic of the controllers. In general, cryptographic protections provide a way to disrupt potentially rogue modules from snooping at the bus. Although this is effective for protection, it might be considered unsuitable since the bus data is mainly useful only in real time when interpreted in context of the control model and physical situation. The overhead is simply too high for each involving module on the bus to conduct encryption and decryption constantly. In addition, compromises at the controller level, e.g., rogue control signals issued by the compromised controller, render encryption irrelevant (encrypting the rogue data does not help security or resilience), or even harmful since the attack traffic/attacker communication is protected by encryption.

Protecting the controllers themselves includes (and is not limited to) fault avoidance, fault tolerance, and model- or reference-based CPS security. Formal methods which attempt to reason over certain properties of an *implementation*

model against a *specification model* is the dominant technique for fault avoidance. Considering the typical tractability and practicality of creating both models, using formal methods is an excellent approach for achieving CPS security and resilience via fault avoidance. But this would imply the complete redevelopment of the system (or at least the subsystem subject to formal methods) from scratch. This is very expensive for legacy systems which are prevalent in industry and military applications.

Fault tolerance is a complementary approach to fault avoidance. This method assumes that vulnerabilities exist in the controller code and strives to mitigate the effect of exploits by ensuring proper operations of the physical system part of CPS, even under successful cyber attacks, thus rendering CPS resilience. Fault tolerance methods often involve detection and recovery, including stateful component, subsystem or system level recovery.

Reference/model-based CPS security relies on the fact that a CPS, unlike general IT systems, is generally well constrained within its operation space and intended behaviors. The operations are periodic and predictable, and reference models for algorithms and the operating environment can be developed and used to detect discrepancy between the observed operation and models. Discrepancy beyond some tolerance threshold may indicate flaw, damage, disruption, or exploits.

2.2.1 Cyber Attacks Against CPS and Critical Infrastructure

Our goal for CPS resilience is to have the physical systems behave properly regardless of fault or disruption (cyber or otherwise). In keeping with reality, we make no assumption that a system is devoid of bugs or vulnerabilities. Rather, we seek to enable a CPS to tolerate and live with existing bugs and vulnerabilities it may have.

We assume an Advanced Persistent Threat (APT)-like adversary, whose goal is to create maximum disruption, major damage, and difficult and lengthy recovery time. To defeat system protection & fault tolerance and to achieve maximum disruption and major damage, an adversary generally needs to subvert and affect many individual controllers (various systems components) simultaneously and in coordination. Uncoordinated one or two subversion and denial of service attacks are unlikely to cause major disruption or damage.

There are generally two methods to subvert or negatively-effect the behavior of a controller: (a) manipulate or inject malicious input to cause improper control output, or (b) hijack and own the controller via either rogue reprogramming command (from console) or malicious input that corrupt program execution, hijack the program control and own the controller. Note that cyber attacks that leak (confidential) information can be used to gather intelligence and help plan for an attack, but by itself cannot subvert the operational behavior of a cyber physical system.

To significantly influence a set of controllers of diverse functionalities and types, an adversary will need to inject many different inputs/signals in a coordinated manner, which is difficult to achieve in practice and often requires the adversary

to own many controllers to perform coordinated, multiple signal injections. The most dangerous exploit is when devices/controllers were stealthily taken over one by one, and then upon triggering event(s), simultaneously act (in coordination) and disrupt the systems. Stealthily owning controllers are the prerequisite for APT-like coordinated attacks. An adversary can stealthily own a controller in several ways. One of them involves reprogramming (re-flashing) the controller itself, e.g., in the case of Stuxnet. To do this, the adversary will generally have to own either the maintenance laptop or the human machine interface (HMI) console, and issue malicious updates from the corrupted laptop or console. This risk can be reduced by requiring multi-factor authentication for firmware update/re-flashing. Another method for owning a controller would be to exploit a (software) vulnerability, and send/inject malicious inputs that will corrupt and take over the controller.

Byzantine fault tolerant++ (BFT++) is a family of cyber resilience methods that rely on the periodicity of CPS and the physical inertia to tolerate cyber attacks. The BFT++ family of CPS resilience prevents this particular class of methods for hijacking and owning the controller. Additionally, BFT++ is generic and agnostic to the particulars of malware or malicious inputs. Refer to section "Byzantine Fault Tolerance++ (BFT++)" for the detailed description of BFT++.

2.2.2 Anatomy of CPS/Controller Owning Cyber Exploits

Fault tolerance systems, such as byzantine fault tolerant (BFT) and quad redundant control (QRC), have been proven effective for safety critical systems. They rely on redundancy to detect and recover from faults, and essentially provide fault tolerance against natural disruption and random faults.

Cyber attacks present a new type of challenges. They can force faults in many components and subsystems simultaneously, which leads to a "common-mode failure" that traditional fault tolerance cannot effectively deal with. Worse, if the adversary is successful in compromising a component, there is no obvious fault signal to detect, and the controllers continue to actuate the system while compromised and under the control of an adversary. Attempts to deal with common-mode failures have been made through diversification, but the type of diversification must be appropriate to the class of causes of common-mode failures that the CPS owner wants to mitigate, and special care must be taken with respect to what, when and how much diversification is deployed depending on CPS and mission requirements.

The process of a cyber exploit involves two virtual stages: first, exploiting a flaw/vulnerability in the program's code to alter its intended execution path, and second, taking control of the system to execute the attacker's commands. This is analogous to a fumble in football, where an opposing team must not just cause a fault, but recover the ball to gain possession, as shown in Fig. 3. A successful exploit will succeed in both stages, leading to compromised systems under attacker control. A condition when the first stage is successful but the second stage fails will

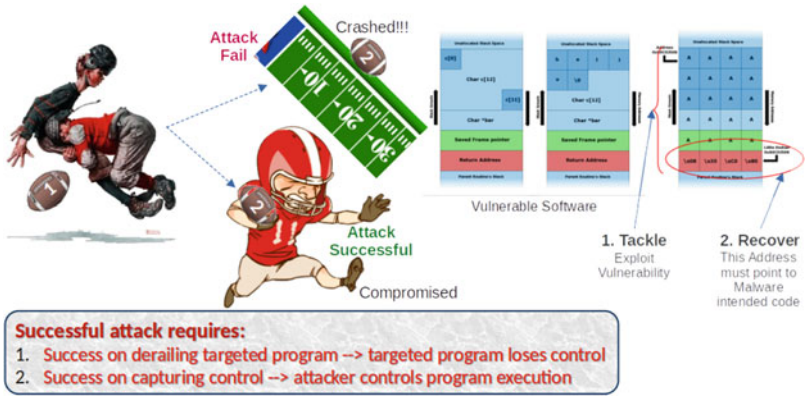


Fig. 3 Two virtual stages of cyber exploit

generally manifest into a crash, due to corrupted (as opposed to compromised) cyber components and subsystems.

2.2.3 Defense: White Listing and Operation Segmentation

Many CPS operational environments can be analyzed and segmented into several different modes of operation. Within each of these modes, the set of valid (allowed) operations can be whitelisted. Operation segmentation is analogous but orthogonal to network segmentation. While network segmentation limits exploits effects and propagation by limiting allowable communications, operation segmentation prevents CPS disruption by limiting incompatible and hazardous co-occurrence of operational commands/events.

For example, let us consider the cyber physical systems that control the operation of a ship, and for the purpose of illustration assume that the engineers decided to segment the ship operation into three modes: steaming mode, in-port mode, and maintenance mode. Maintenance mode is akin to the superuser mode in modern operating systems where (almost) all operations are allowed. For simplicity, let us consider three different operations: dropping anchors, brisk-steaming (above 5 knots), and re-flashing the controller. One can see that dropping-anchors and brisk-steaming are mutually exclusive. It will not be prudent to drop an anchor while briskly steaming, hence dropping anchors is not within the steaming mode whitelist, and brisk-steaming will not be in the in-port mode whitelist. Similarly, re-flashing a controller should only be performed in the maintenance mode.

Operation segmentation improves the operational safety of a cyber physical system. Separating the maintenance mode from other operation modes also enhances the system's cyber security posture by whitelisting out disallowed behaviors and requiring additional privilege for critical activities, such as re-flashing a controller. While it cannot completely prevent cyber attacks, operation segmentation erects

barriers against various malicious activities that may otherwise readily perform once a foothold is obtained in a component or subsystem.

Operation segmentation focuses all the working aspects of the cyber physical systems onto the operators, which are responsible for the CPS operations, including approving and initiating CPS maintenance. This is a judicious method compatible with the principles of separation of duties and least privilege for building computer systems [10]. Current trends in modern cars, which are systems of cyber physical systems, however, are diverging from this philosophy. In the case of modern cars, it is the manufacturers who often initiate the system update, with or without the awareness of the operator (owner). There are both pro and con arguments that can be made for this context.

2.2.4 Defense: Reference Model Based CPS Security

Since cyber physical systems extensively communicate with their physical environment, system security relies not only on cyber security but also on securing the physical part of the system. This means the cyber layer, as well as the platform (including the physical) layer and their inter-dependency, must be considered together. For example, the platform layer covers the whole run-time environment containing artifacts like operating system and middleware, as well as the physical part of the system such as sensors and actuators, etc. Hence, as an entity that senses and interacts with the real world, a CPS could be exploited by an adversary and cause harmful impacts. Depending on the level of the attacker's access and capabilities, either or both sensing components and control software can be subjects of a compromise.

One of the most common security approaches for detection of attacks against control software is a comparison of true and faulty signals, thus necessitating trusted redundancies. For example, if an extra electronic control unit (ECU) hardware is retrofitted to the robotic vehicle with no access channel from the outside, it is shielded from the attacker, and hence can be trusted. Such CPS can still operate as intended with its original control software, while the control signal can also be used to enable comparison against the retrofitted ECU for attack detection and response. Instead of changing the original control system, an external piece of hardware can be used to monitor the given ECU with minimum modification to the original system. Independently implementing the CPS control and sensing logic software on the external hardware enables high-accuracy error detection.

Such combination of the software and hardware redundancy has been proven to successfully detect a variety of attacks on the sensor, controller, vehicle dynamics, actuator, and controller operating systems [4]. The attack detection must combine control algorithms such as state-estimation, fault detection and diagnosis, fault tolerant control parameter and controller estimations to detect CPS dynamic changes. Specifically, it detects changes in the original system by comparing instantaneous outputs in real-time, while it is shielded from attackers. The entire diagnosis process can cover both the cyber and physical domains. A smooth variable structure filter

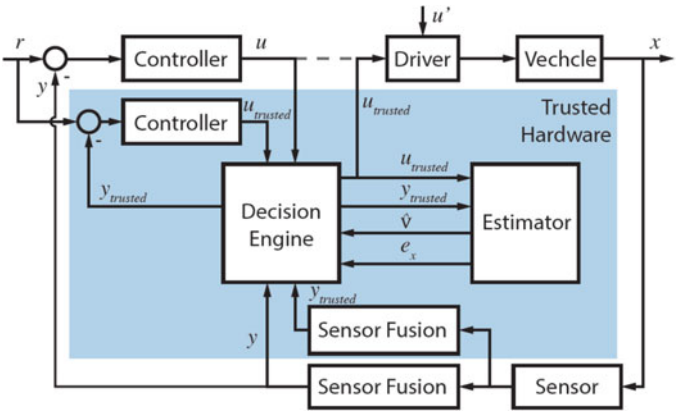


Fig. 4 System diagram of the FDI approach using hardware redundancy (reprinted from [4])

(SVSF) was used to estimate system states and identify system parameters, which is proven to be robust to model uncertainty and noise. The system diagram of such an approach is shown in Fig. 4.

Raw sensor data are extracted and used to compare with the feedback from the original system to determine if the sensor fusion result or code in the original system has been modified. The decision engine will integrate the error between the two measurements within a fixed time window and identify sensor attacks using thresholds. The sensor fusion results are also fed to its internal controller and SVSF-based estimator for further security diagnosis and attacks detection.

The sensory system is also critical for CPS safety. Recent advances in adversarial studies demonstrated successful sensing fault generation by targeting the physical vulnerabilities of the sensors. A complete CPS sensor safety design must contain both an fault detection and isolation (FDI) unit and a fault recovery (FR) function to tolerate the detected flaws. When faults are identified and isolated by the FDI, the recovery logic should then be able to maintain the correct state with as much stability as possible using the remaining incomplete sensory systems.

As CPS sensors, the actuation system is also vulnerable and can be easily compromised via similar cyber and physical domain strategies. Actuator failures not only affect the normal operations, due to the implicit dynamics from the actual system, they also introduce the need of FDI design to distinguish the exact sensing and actuation faults. A flawed sensor may induce multiple state anomalies simultaneously and CPS may yield a similar abnormal action to two completely different types of failures (e.g., sensor or actuator failure which complicates pinpointing the failure source). Hence, to achieve proper FDI capability, the inherent coupling of the CPS dynamics must be considered. Further, when multiple CPS states malfunction simultaneously, it is hard to identify the exact failure sensor. The cascading effect may also have to be considered. For example, the high-level

sensor abnormalities can affect low-level sensing in cascade for UAVs (e.g., attitude twitching can be subject to the frequent loss of position feedback).

One major circumstance to help sensor recovery is the fact that in most cases, excluding the most catastrophic, all the onboard sensors cannot be rendered defective at the same time. It is hard to compromise multiple sensors simultaneously because they typically measure different physical terms and possess distinct working principles, communication methods, and signal bandwidth.

To recover the sensor readings, installation of a backup sensory system is the most widely used approach. Through a simple comparison and replacement, this hardware redundancy is effective against the traditional software-based sensor faults and attacks, such as numeric error, trojans and data spoofing. However, this approach is not sufficient when the CPS encounters some well-designed attacks that concern both cyber and physical properties of the targeted sensors. This is because the redundant sensors exhibit the same physical vulnerabilities as the original ones. For example, a redundant attitude sensor would be incapable of nullifying the effects of resonating the inertial sensors via external excitation. It is highly likely both the original and redundant sensors would fail.

As an alternative to a redundant hardware approach, the redundancy-free methods for CPS sensor FDI and fault recovery (FR) as reported in [11] make the most sense. Figure 5 shows the system diagram, which consists of a fine-grained sensor FDI architecture and a sensor complementary FR in parallel. For fine-grained FDI, a basic state estimator for a rough early warning of faults combined with the un-measurable actuator state and modeling uncertainties are utilized. Instead of adding auxiliary sensors, the method uses the original sensor arrays and leverages complementary sensor estimations for FR implementation.

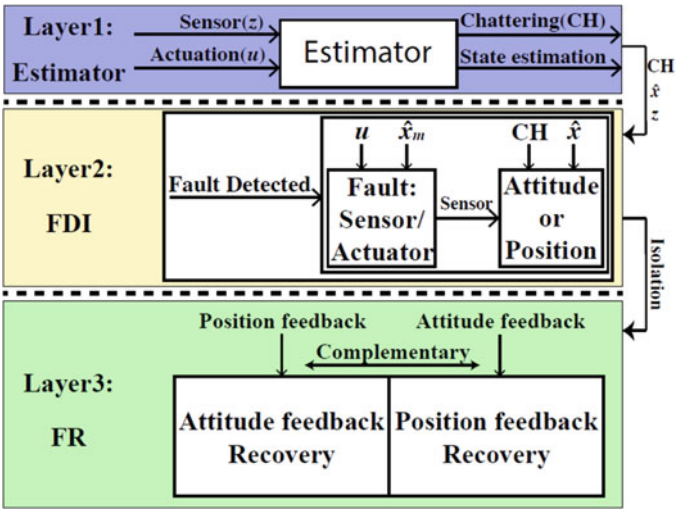


Fig. 5 FDI/FR without hardware redundancy (reprinted from [11])

The FDI design is based on smooth variable structure filter (SVSF), which is a sliding-mode-based state estimator with a prediction-correction workflow. In one iteration, a model-based prediction function generates a priori state estimation first, then a discrete corrective action is taken by adding a corrective gain. The corrective gain is not only used to guarantee the stability of the estimator but also rectify the bounded estimation error robustly. Subsequently, the updated *posteriori* estimation and state measurement carry out the next iteration. The fault is detected by examining sensor FDI procedure-residual check (i.e., the discrepancy between estimation and actual sensor reading). If the residual rises beyond a certain threshold, a fault is supposed to be present and will be reported to begin the recovery.

The sensor fault recovery without hardware redundancy is CPS domain specific. For example, for UAVs, due to the geometric correlation of the vehicle dynamics, position and attitude feedback can be used to compensate each other. During the recovery process, with the fine-grained FDI, the compromised sensor reading is rejected and compensation from other trusted sensors will be utilized. For example, in case of an inertia sensor failure, position information can be used to derive an alternative attitude for flight control. When the UAV loses its position feedback, the inertia measurement can be utilized to compensate position drift.

In summary, reference models provide feasible and robust means for state estimation, behavior prediction, discrepancy checking, decoupling of sensor and actuator faults, and diagnosing multiple faults and accurately isolating the source of faulty elements, thus offering a well-grounded base for building security and resilience in cyber physical systems.

2.2.5 Defense: Vulnerability Prevention

To prevent against the first stage of a cyber exploit (see Sect. 2.2.2), CPS software needs to be devoid of any exploitable vulnerability. Fault or vulnerability avoidance generally falls within the first quadrant (P(S), C(S)) of Fig. 2. CPS software can be analyzed against exploitable vulnerability, and the location where a vulnerability is identified will be *hardened* with security checks or assertions. Software vulnerability analysis generally uses both static (e.g., symbolic execution) and dynamic analysis (e.g., fuzzing) tools for finding exploitable vulnerabilities. *Formal verification* is another approach for assuring that the software is devoid of flaws or vulnerabilities. We will describe hardening and formal methods in what follows.

Security Hardening

Vulnerability analysis and hardening is usually performed in several steps: (i) static software program analysis, (ii) instrumentation, (iii) symbolic execution, and (iv) fuzzing with dynamic tracing/feedback-guided fuzzing with sanitizing.

Software program analysis includes combination of static analysis (e.g., dependency analysis, program slicing, etc.) and a symbolic exploration of the program's

state space (e.g., "can we execute it until we find an overflow?" or "let's execute only program slices that lead to a memory write to find an overflow."). Symbolic execution also takes advantage of instrumentation for precise results.

Fuzzing is a form of software testing where an application is run with random (potentially malformed) inputs while monitoring the runtime for unexpected behaviors, e.g., crashes, memory exhaustion, or infinite loops. There are generally two types of fuzzing: (i) *Blackbox fuzzing* (i.e., fuzzing with no knowledge about the target application) may not be effective in many cases as most inputs are likely to explore very shallow code paths. This severely limits the fuzzing ability to uncover bugs in deep parts of the code. (ii) *Coverage-guided fuzzing* tackles this problem by using program traces generated by the inputs as a feedback mechanism to tailor future inputs to the fuzzing target.

In essence, fuzzing depends on program crashes to detect and report bugs. Consequently, bugs that do not trigger crashes are not caught through fuzzing. Therefore, for effective fuzzing, software must be instrumented with sanitizers (e.g., a memory checking code such as memory leaks and initialization, heap and stack overflows, illegal accesses, etc.) either at compile time or at the binary level.

Program analysis and instrumentation can be performed for newly developed software during CPS software code compilation, or, on the available binary code for legacy software. For the former case (i.e., compiler-time analysis and hardening), the mainstream software build tools (e.g., GCC and LLVM) provide extensive interface and framework for analysis and code optimization in its Intermediate Representation (IR) form during code compilation. For the latter case (i.e., binary analysis and rewriting), analysis in the form of symbolic execution with various static analyses on binaries is performed as three distinct steps: (i) loading a binary into the analysis program, (ii) translating a binary into an IR, and (iii) performing the actual analysis.

The rewriting part for instrumentation and hardening presents a few difficult challenges. Specifically, dis-ambiguating reference and scalar constants, so that a program can be "re-flowed" (i.e., having its code and data pointers adjusted according to the inserted instrumentation and data section changes) is a major challenge. During assembly, labels are translated into relative offsets or relocation entries. A static binary rewriter must recover all these offsets correctly. There are three fundamental techniques to rewrite binaries: (i) lifting the code to an intermediate representation, (ii) trampolines, which rely on indirection to insert new code segments without changing the size of basic blocks, and (iii) reassemblable assembly, which creates an assembly file equivalent to what a compiler would emit (i.e., with relocation symbols for the linker to resolve).

Lifting code to IR for recompilation requires correctly recovering type information from binaries, which remains an open problem. Trampolines may significantly increase code size and do not scale very well. Consequently, we believe that *reassemblable assembly* is the most promising approach, which creates assembly files that appear to be compiler-generated (i.e., do not contain hard-coded values but assembly labels). Symbolizing the assembly allows security-oriented rewriters to directly modify binaries, which is similar to editing compiler-generated assembly

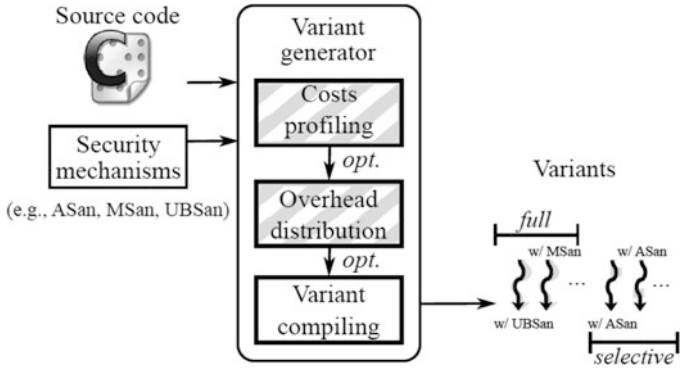


Fig. 6 Variant generation workflow (reprint from [12])

files. Once modified, the symbolized assembly files can be assembled using any off-the-shelf assembler to generate an instrumented binary.

There are a number of security mechanisms and sanitizers to harden programs written in unsafe languages, each of which mitigating a specific type of memory error. It includes various memory checking techniques, undefined behavior monitors, control flow integrity trackers, temporal safety enforcers, etc. The major problem is that the execution slowdown caused by various security mechanisms is often non-linearly accumulated, making the combined protection prohibitively expensive.

One of the most viable approaches to mitigate this problem is to use *security diversification* consisting of N variants that are both functionally identical in normal situations and behaviorally different when under attacks. Hence, although each program version may be vulnerable to certain types of attacks, the security of the whole system relies on the notion that an attacker has to simultaneously succeed in attacking all variants in order to compromise the whole system. In addition, different and even conflicting security mechanisms can be combined to secure a program while reducing the execution slowdown by automatically distributing runtime security checks in multiple program variants [12]. This can be achieved by making sure that conflicts between security checks are inherently eliminated and execution slowdown is minimized with parallel execution. The N-version execution engine synchronizes these variants so that all distributed security checks work together to guarantee the security of a target program. The notional workflow diagram is shown in Fig. 6.

Formal Methods

While program analysis tools are indispensable to find vulnerabilities, they essentially explore part of the state and execution space, and hence can never provide complete guarantees. Formal methods provide complementary means for vulner-

ability prevention. Generally speaking, formal methods are system design and analysis techniques that use rigorously specified mathematical models to build software and hardware systems. In other words, formal methods use mathematical proofs as a complement to system testing (e.g., fuzzing) in order to ensure correct behaviors.

Using the terms mentioned in Sect. 2.1, consider a binary executable as the *thing*. In general, the *Concrete Model* or M_c is comprised of (but not limited to) an instrumented C or C-like program (source). The concrete model represents the actual executable but is more generic, meaning some properties are present in the model that are not reflected in the binary. An *Abstract Model* or M_A (for reasoning) is comprised of formal constructs such as Hoare's logic (or its variants such as separation logic, higher order logic, etc.). It can be derived from the binary or M_c and captures all properties in binary or M_c but is more generic and might include properties not in binary or M_c. In an ideal world, no error in lifting or abstraction is made and the three concepts (executable, M_c, and M_A) are all identical. In reality, however, this is not the case and much research has been conducted to shorten the gaps between them. One notable methodology is counter-example guided abstract refinement (CEGAR) where M_A is iteratively checked against a given property and refined if the check fails [13]. Most of the formal verification efforts adopt a similar model refinement approach.

Over the last decade, the understanding of formal methods and development of tools have improved to the point where formal verification of real-world software has started to become feasible. Examples include functional correctness proofs of microkernels and cryptography libraries. Formal methods have also been used to identify deep vulnerabilities in software, revitalizing the field of program analysis. With respect to vulnerability prevention, seL4 is the first formally verified microkernel with a functional correctness proof of the abstracted source code against the specification, effectively asserting the absence of typical programming errors such as null pointer dereferences, buffer overflows, and arithmetic exceptions [14]. The development of seL4 was supported by the DARPA High-Assurance Cyber Military Systems (HACMS) program, which aimed to create technology for the construction of high-assurance cyber-physical systems, where high assurance is defined to mean functionally correct and satisfying appropriate safety and security properties. Since it's original proof over a decade ago, seL4 has seen reasonable successes in both continual development and early adoption. For example, due to the offered higher-level of confidence for assurance, seL4 was selected by the AFRL Agile and Resilient Embedded Systems (ARES) program to serve as the separation kernel providing memory allocation and isolation based on the hardware memory management support.

However, while formal methods provide a rigorous way for vulnerability prevention, it is important to point out that the proofs are usually carried out between *models* (e.g., abstract and concrete models), as opposed to against the *thing* or binary executable in this case. For example, just because seL4 is formally verified does not necessarily mean the binary executable runs exactly as expected in the specification on the target computer architecture. Additional binary level assertion is still needed

if the purpose is to provide execution assurance directly on the computer hardware. Again, this is a reminding and cautionary tale related to the *thing* and its *models*, as discussed in Sect. 2.1.

2.2.6 Defense: Vulnerability Tolerance

Most safety-critical systems utilize some type of redundant architecture to deal with faults. Examples include hot backups; dual, triple, or quad-redundant architecture; or Byzantine fault tolerance where assumptions about the fault conditions are random, and faulty replicas may behave arbitrarily. Fault tolerance provides a means to automatically deal with faults and recover from them. Cyber attacks, however, will drive fault tolerant system into *common mode failures* (see Sect. 2.2.2). The challenge is how to retrofit existing fault tolerance architecture to rectify faults caused by cyber attacks.

A typical cyber physical system offers certain properties and advantages one would not find in a general IT system. This is because the physical aspect allows for a certain degree of predictability in the behaviors of the system.

- **Periodicity:** The cyber subsystem that directly interacts with the physical plant runs in continuous cycles. For example, throughout its execution the controller reads values from sensors, calculates the error correction signal, and writes out actuator values. For the commonly used industrial controllers, Programmable Logic Controllers (PLCs), this is called the *scan cycle*.
- **Inertia:** Any physical subsystem of a CPS must obey the laws of physics and physical systems inherently have inertia. The scan cycle of a controller is typically engineered to be fast enough such that an issue in a small number of cycles will be dampened out by the existing inertia. The cycle frequency is set depending on the system but common values vary anywhere between 1 Hz and 1 kHz.

Due to this predictability offered by inertia and periodicity, anomaly detection approaches can be naturally used to detect anomalies and threats in the system. A resilience strategy can also be developed to detect attacks by monitoring actions such as subverting control flow, reprogramming controllers, or overriding sensors that are out of the normal operation ranges.

Cyber vulnerability (and attack) tolerance does not rely on the need for software to be devoid of vulnerabilities. Instead, it assumes that unknown vulnerability exists within the software and strives to maintain the safety and normalcy of system operation regardless. Vulnerability tolerance methods focus on the second stage of cyber exploits (see Sect. 2.2.2) and will generally have to perform timely recovery within the limited time afforded by the physical systems' inertia, as the first stage of cyber exploits may have already occurred and the cyber systems may have been corrupted. In what follows we will describe some example techniques, tools and frameworks for providing vulnerability and attack tolerance, empowered by inertia and predictability unique in cyber physical systems. These include Software

Brittleness, Byzantine Fault Tolerance++ (BFT++), You Only Live Once (YOLO), and CPS Cyber Resilience Architecture (CRA).

Software Brittleness

For some certain types of critical cyber physical systems, avoiding operating in degraded or compromised state is of paramount importance, and fast program exit and re-start (called *software brittleness*) is required when a cyber attack succeeds and the program control is lost. Examples include Industrial Automation and Control Systems(IACS), Supervisory Control and Data Acquisition (SCADA) control systems and devices, Programmable automation controllers (PAC), remote terminal units (RTU), Master terminal units (MTU), intelligent electronic devices (IED), etc. Software brittleness is a novel concept enabled by the new design space, i.e., the fourth quadrant with P(S) and C(U) as shown in Fig. 2b. Essentially, the inherent physical inertia allows enough room for cyber components to reconstitute themselves via fast crash-and-recovery.

Code randomization/diversification for software brittleness can be implemented at either pre-distribution or post-distribution stages. Both types of diversification (i) provide the level of code diversification sufficient to guarantee that an attack that succeeds in the original program will fail in the variants, and (ii) assure prompt attack discovery through self-monitoring capabilities of the diversified code. Using N-voting system with simultaneously running multiple generated variants will assure prompt discovery and recovery. There is an integrated set of diversification techniques available at both the source and binary code levels against most known attacks (e.g., memory corruption, code injection and re-use, control flow hijacking, information leaks, etc.). The conceptual approach toward software brittleness, called Binary code Randomization for Attack Sensitive Software (BRASS), is shown in Fig. 7. It has been demonstrated that this approach provides prompt attack discovery and program abort & recovery with low performance and size overhead [15]. In the CPS context, software brittleness can be included in controllers, for example, and managed through some vulnerability/fault tolerance framework, which comes next.

Byzantine Fault Tolerance++ (BFT++)

BFT++ is a family of cyber attack resilience methods that rely on the periodicity of CPS and the physical inertia to tolerate cyber attacks [16]. The initial concept of BFT++ was developed by the Office of Naval Research (ONR). It operates in the fourth quadrant (P(S), C(U)) in Fig. 2b. MITRE Corp. maintains a reference design for BFT++ for the NAVY and DoD in general. BFT++ has been demonstrated to withstand US-NAVY sponsored "Hack The Machine" hackathon.

Figure 8 illustrates the main components of the BFT++ design. It is built over the classical BFT systems. *Artificial diversity* in the form of diversified software compilation or diversified processors (ISAs) is used to break common-mode failure

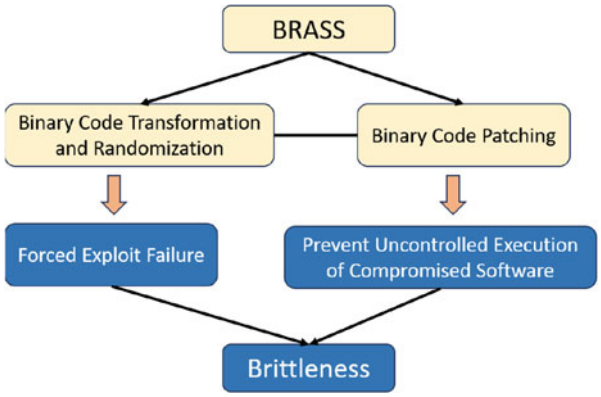


Fig. 7 The conceptual approach toward software brittleness

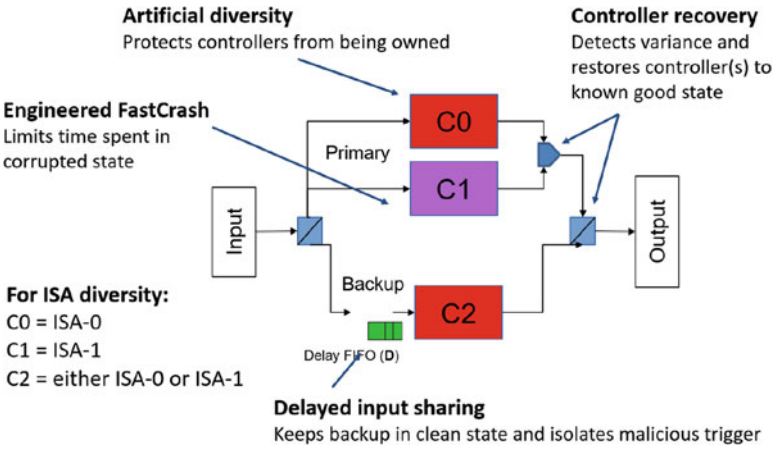


Fig. 8 General design of BFT++ Cyber attack resilience methods for CPS

(C0 & C1), and delayed input sharing (delay queue) is inserted into the input of the backup unit/controller for stateful (warm) recovery. For a CPS system that does not demand stateful recovery, the delay queue and backup unit (C2) can be omitted.

BFT++ uses artificial software diversification applied to existing code. As an alternative, two (or more) distinct processors of different instruction set architectures can also be used to provide diversity. When used in combination with traditional fault tolerance architecture, this is also effective at absorbing (and tolerating) cyber faults. Note that attackers only have one opportunity per scan cycle to provide corrupting inputs. Diverse replicas will have different code layouts, making it almost impossible for attackers to inject malicious code that works across all replicas simultaneously. Due to the real-time nature of the periodic control loops, synchronization across replicas is built-in, and an attack can be detected if the program results vary across replicas or if timely responses are not provided.

The next crucial step is enabling the system to recover. Diversity within the system can make it more fragile, so fast-acting and automated recovery must be employed to counterbalance this. Without recovery, the attacker could maintain control of one (compromised) replica and leave the others crashed—a clearly unacceptable state.

Crash detection is the first part of the recovery process. Ideally, we want to detect a potential compromise via a crash of one of the diversified replicas as soon as possible. In our case, a replica failing to produce timely output by the end of the epoch is considered to have crashed. This serves as a canary that there has been a compromise.

Next, a small message queue is employed in front of one of the replicas (henceforth referred to as the "protected replica"). This is key, because when a potential compromise is detected (via the crash detection), the message(s) triggering said crash are trapped in the queue before reaching the protected replica. Upon crash detection, this queue can be flushed removing the offending messages. While this introduces a small delay to the protected controller, the physical inertia of the system allows BFT++ to absorb this without impact to the real-time operation.

Finally, recovery begins, and the state of the replicas are restored. Restoring from a checkpoint is possible but requires much resources to handle the overhead of saving checkpoints as well as a way to deal with the staleness of state upon a restore. Instead, the strategy advocates designating one or more replicas as backups and time-delaying them, so they process inputs one or more cycles behind the primaries.

This method for cyber resilience has allowed older control systems to identify cyber attacks during their normal operation, automatically triggering a quick and efficient recovery process. However, there is still a concern that attackers may exploit this system behavior to launch an availability attack. While we have prevented any exploit from affecting the system's integrity, it is possible for a known vulnerability or bug to trigger the recovery process and cause the system's availability to be compromised. To address this issue, we designed a mitigation strategy known as "Shims" that filters out any malicious inputs that cause the recovery architecture to send a crash signal. By implementing shims at the input point for the controllers, replaying an exploit after it has already been used against the system will be prevented.

For a particular BFT++ implementation [16], the architecture has three redundant diversified controllers operating in a traditional fault-tolerant architecture. The artificial diversity makes it difficult for a cyber attacker to compromise all controllers with the same malicious input. Although an exploit may be successful against one replica, it will cause the diversified replicas to crash. Next, it incorporates delayed input sharing (e.g., FIFO message queue) to trap bad messages before reaching a "protected controller". This introduces a delay to the protected controller, but ensures the system to continue operation and to be reconstituted after the cyber exploit. The recovery timing of the system is governed by several timing parameters, such as Tcrash, Tsc, D, Td, and Tr. Tsc, Td, and Tr are system parameters, and D needs to be appropriately set for automated recovery to be possible. The two critical points that determine the system's recovery timing are the *brittleness* of

the controllers and how quickly the system can restore a controller to the normal state. The physical subsystems with higher inertia are generally more tolerant of losing control signals for a short time. In general, the following relationship between these parameters must hold for BFT++ to be applicable to a legacy cyber physical system [16]:

$$Tcrash \leq D * Tsc \leq Td - Tr$$

Parameters	Definitions
Tcrash	Time between attack and crash
Tsc	The scan cycle period
D	FIFO queue length (number of slots)
Td	Maximum control loss tolerable by physical systems
Tr	Recovery latency

You Only Live Once (YOLO)

YOLO is another CPS cyber resilience method that relies on the physical inertia to withstand cyber attacks. YOLO and its variants use periodic restart and does not require any redundant controller [17]. It also operates in the fourth quadrant (P(S), C(U)) in Fig. 2b. YOLO was developed at Columbia University under the sponsorship of the Office of Naval Research.

Figure 9 depicts the design philosophy for YOLO. YOLO implements periodic restart to limit the duration of potential compromise. An adversary who managed to

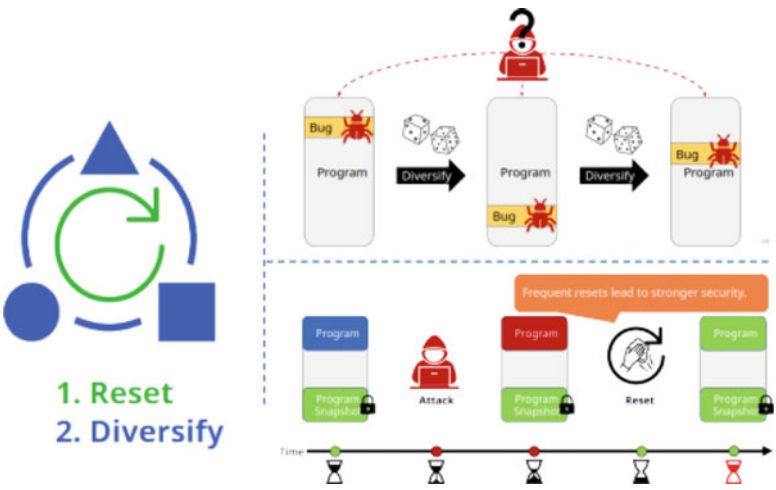


Fig. 9 General design of YOLO cyber attack resilience methods for CPS [17]

compromise the system only has control over the system for a maximum duration of the restart period. The YOLO restart period can be designed to be short enough to prevent an adversary to achieve persistence within the CPS system, while still within the tolerable region provided by the physical systems' inertia. During each restart, the controller is reset to its 'clean' state by loading its software from a read-only module and clearing out all the volatile memory. YOLO also implements software diversity after each restart to ensure that the attacker cannot exploit the same vulnerabilities. YOLO has been demonstrated to be practical for automotive engine management unit, drone controller and a missile launcher.

In YOLO, the restating latency and state recovery time need to be within the range that the inertia of the physical systems can tolerate. Proper engineering analysis and design is required to accelerate the restarting process and to avoid lengthy reboot and initialization latency of the cyber system. YOLO does not require replication, and hence it is cheaper to implement than BFT++. However, its protection is not as deterministic as that of BFT++, and current version of YOLO does not support stateful (warm) recovery.

CPS Cyber Resilience Architecture (CRA)

Existing CPS cyber resilience architectures, including BFT++ and YOLO, have been analyzed and summarized into a timing-based formulation framework [18]. Within this framework, safety analysis and computation of control policies and design parameters can be performed for each pair of CRA method and CPS application.

The framework relies on the insight that the cyber subsystem operates in one of a finite number of modes. It defines a hybrid system model that captures a CPS adopting any of these architectures (CRAs). Analysis within this framework uses the transition model of the hybrid system to derive architecture-agnostic sufficient conditions for control policy and timing parameters that ensure safety of the CPS. The analysis will then formulate the problem of joint computation of control policies and associated timing parameters for the CPS to satisfy a given safety constraint and derive sufficient conditions for the solution. Utilizing the derived conditions, they provide an algorithm to compute control policies and timing parameters relevant to the employed architecture. The framework efficacy has been demonstrated in a case study involving automotive adaptive cruise control. The study was performed for each of the CRA methods in their framework, and proved that the algorithm converges to a feasible solution under certain conditions.

Figure 10 visualizes the operation of a drone employing YOLO, under continuous cyber attacks. It shows three operation zones: desired operating zone, zone of tolerance, and danger zone. The drone is expected to operate in the desired safe zone, and danger zone can only be safely entered when the vehicle is in normal mode, otherwise catastrophic crash may occur. The vehicle (drone) can be safely restarted and recovered within the tolerated zone. Safety cannot be guaranteed if the drone enters tolerated zone in a corrupted state.

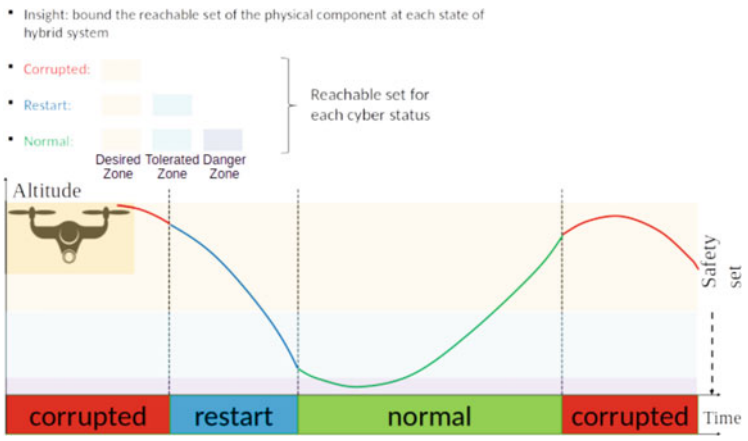


Fig. 10 An illustration of CPS safety analysis within the framework [18]

3 Machine Learning and CPS

In this chapter, the terms *machine learning* and *neural networks* will be used interchangeably. At the most general level of interpretation, machine learning is a super-set of statistical machine learning (i.e., neural networks). The term *machine learning* in general also includes learning heuristics and other logical and formal form of learning mechanisms. Contemporary use of the term *machine learning* generally refers to various forms of neural networks, which are often considered as surrogates for formal/logical control algorithms in the CPS context.

Robotic devices and vehicles can perform some of its functions using machine learning and especially reinforcement learning (RL), e.g., RL for drone fault recovery [19]. Training neural networks may be performed by guiding the robotic devices to function within its physical environment. Training may also be conducted in the virtual simulation environment [19], where the sensory and control input as well as the actuators and their dynamics are simulated using physics models of the actuators, sensors and the physical environment. The use of physics models for training a CPS system is generally safe as the laws of physics are universal, relatively complete, consistent and context insensitive.

Machine learning may also be used for correlating various monitored and logged events in cyber physical systems. It helps correlate cyber events such as network events, activation of computing events, sensed and computed parameters' values, etc., with observed physical and environmental events. Trained this way, machine learning models the operational behavior of the cyber physical systems and can be used to highlight unexpected behavior and anomalies. The use of machine learning in this case is inherently incomplete. While well trained machine learning algorithm/model is expected to generalize and cover the CPS operation space, there is no practical, assured way to ascertain that it covers all of the possible cases of

the application/CPS-operation, e.g., corner and unexpected cases. Such machine learning model will produce false positives and false negatives. The quality of machine learning output (prediction) is significantly dependent on the methods, the quality of data, and models used for training. However, with proper operator due diligence and supervision, the deployment of machine learning can significantly improve the safety and security of CPS operation, as a complementary means to the traditional model-based mechanisms.

3.1 Enhancing CPS Robustness with Machine Learning

It has been well established that the traditional cyber techniques in software and firmware can no longer sufficiently protect the system and ensure safe operation of the cyber physical systems when attacks are launched against the physical components of the CPS, such as signal spoofing or using sound wave to resonate the IMU sensors. As a result, undesirable performance or even loss of control would occur. Given that attacks/faults cannot be fully prevented, fault/vulnerability tolerance and CPS resilience and recovery strategies are required.

Traditionally, there are two types of fault-tolerant control: passive fault-tolerant control (PFTC) and active fault-tolerant control (AFTC). AFTC has a fault detection and diagnostics (FDD) component to identify the source of the fault, reconfigure a controller, and compensate for such fault. The FDD component is usually an observer and can generate residual signals to indicate the fault. Both sensor and actuator attacks or failures can be detected with system models. Meanwhile, PFTC does not have an FDD mechanism, but aims to improve the controller's robustness and tolerate the fault condition or attack. AFTC can pinpoint the fault and act accordingly, but if the FDD is not designed with care, the implementation could lead to delay in detection or false positives and greatly affect the performance. While PFTC cannot isolate faults, they could potentially achieve more robust performance.

Machine learning (ML) and reinforcement learning (RL) have been explored in developing FTC strategies. However, most of the ML/RL methods were only evaluated in simulation, and their real-world performance is unknown. Deploying reinforcement learning policies onto real systems in this case is extremely challenging since training has to be performed in simulation before trained models being transferred to real cyber physical systems to recover from sensor and actuator faults.

Reference [19] demonstrates that RL-based policy trained in simulation can indeed be transferred to real unmanned aerial vehicles to recover from sensor and actuator faults. Unlike traditional FTC, this policy does not require fault detection and diagnosis (FDD) nor tailoring the controller for specific attack scenarios. Instead, the policy runs simultaneously alongside the controller without the need for fault detection and activation. When the CPS is operating normally, the policy generates no or minimum control command adjustment and does not interfere with the operation. When the fault condition arises, or the CPS is under attack, the policy

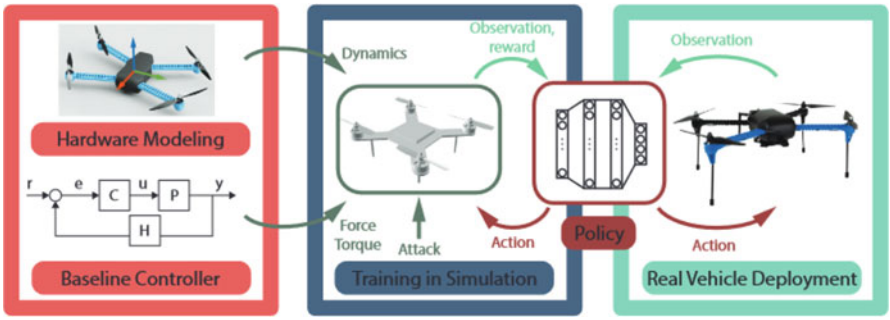


Fig. 11 RL-assisted fault-tolerant control workflow, derived from [19]

takes the state inputs and generates appropriate actuator command adjustment with little or no delay to compensate for the fault/attack condition.

For simulation, identical sensor fusion and control algorithms must be implemented so the closed-loop dynamics of the vehicle in the simulation can approximate the real vehicle. The approach allows for simulation validation statically through open-loop and closed-loop tests. The fault resilient policy optimization is formulated with a standard reinforcement learning problem, where the agent is the quadcopter and the environment is the simulated world. The attacks were implemented by replacing the actuator signal or the sensor value with a random number.

The training takes place and policy is implemented during dynamic simulation on existing legacy system through minimally intrusive software retrofitting. The control algorithm is used in a dynamic simulation during which a fault-tolerant policy is optimized using reinforcement learning to maintain CPS control under various simulated attacks. The RL-assisted fault-tolerant control workflow is shown in Fig. 11.

Since the system dynamics is largely deterministic, reference [19] uses the actor-critic deep deterministic policy gradient (DDPG) algorithm for training. Fully connected multi-layer perceptions (MLPs) serve as the function approximator for policy representation. MLP splits between a nonlinear control module and a linear control module. Intuitively, nonlinear control performs forward-looking and global control, while linear control stabilizes the local dynamics around the residuals of global control. This improves training sample efficiency, final episodic reward, and generalization of learned policy, while requiring a smaller network and being generally applicable to different training methods. Only nonlinear fault-tolerant policy needs to be learned. This approach views the policy as an optimized FDD and FTC control approximated by a neural network. It can be leveraged (with the proper simulation setup) for a variety of cyber physical systems with a learned policy designed as an add-on to other closed-loop mechanisms.

Another use of ML is to function as a surrogate (“Digital Twins”) and to be used as a reference to detect anomalies and cyber disruption. In this role, the ML model

will correlate cyber and physical events and flag any inconsistencies, plausible faults or anomalies. Such approaches extend machine learning (ML) methods for analyzing system logs of CPS and identifying the key CPS entities to reconstruct the critical steps of a plausible attack. Forensic analysts collect diverse system logs from multiple CPS components. The massive volumes of logs are often analyzed offline or monitored in real-time to debug system failures and identify sophisticated threats and vulnerabilities. There are several techniques being developed to extract features/sequences from logs to automate intrusion and failure detection and to discover associations among disparate log events through event correlation.

Working with the text logs requires integration of natural language processing (NLP) and deep learning techniques into data provenance analysis to identify attack and non-attack sequences. The typical steps include (i) processing system logs; (ii) building optimized causal dependency graph, from which the semantically augmented event sequences are constructed; and (iii) learning a sequence-based model that represents the attack semantics to recover key attack entities describing the attack story at inference time. The key challenges for such solutions are (i) additional overhead on a running system, (ii) integration of diverse logs, (iii) scalability of the large and complex causal graphs, (iv) accuracy of constructed sequences models, and (v) efficient automation.

One of the promising approaches is based on the assumption that the crucial steps of different attacks in a causal dependency graph may share similar patterns. This allows for identification of key attack steps through an attack symptom event, based on those sequences that share semantically similar attack patterns to the ones it had pre-learned. Such knowledge helps to substantially save time when investigating large causal graphs and helps in constructing the attack story from a limited number of attack symptoms.

3.2 Roles and Pitfalls of AI in CPS

As the complexity of automation increases, the roles machine learning may play in CPS are also expected to grow. The use of machine learning often requires an extensive set of labelled data for training, and the curation of this large, labelled data set is often problematic. While data can be scraped from the Internet, labelling them requires tedious manual effort, and is often outsourced to third-world country or Mechanical Turk (Amazon). It can be a very expensive proposition.

Fortunately, many CPS operations are governed by physics, with formulas and models that have been developed and proven over decades. Due to the computation limitation of many CPS devices, and the potential complexity of the interaction among physical phenomena, it is often not practical to deploy detailed physics models as reference to the operation of the CPS infrastructure. A surrogate—a (computationally) lighter weight machine learning algorithm/model—can be trained with labelled data generated by these complex physics models and practically

deployed as the reference model. This surrogate model trades off precision and determinism/correctness with computation cost.

However, machine learning for cyber components at the level of software execution is quite challenging. Unlike natural language, image and video processing, there is no public, large-scale, comprehensive, and well-labelled data set that researchers can use for evaluating the efficacy of machine learning for cyber security and resilience. Research works in this area are often forced to develop their own data for training and evaluation. This effort is both expensive and non-comprehensive, limiting the quality and generality of the research effort.

The periodicity and predictability of CPS operation help reduce the overall challenge, as they potentially provide "structures" and "constraints" for the learning problem at hand. In machine learning, *knowledge* about the problem domain and relevant features extracted from the domain knowledge still play a critical role. Properly observing/incorporating physics-based models in the CPS software and machine learning process will help focus the training process, constrain the search space, and enhance the performance of the resulting machine learning model.

Transfer learning offers an appealing way to help reduce the size of training data needed to achieve reasonable performance. Employing transfer learning, one can adopt a suitable pre-trained ML model whose size and structure can accommodate the target problem space. The pre-trained ML model is expected to have its internal weight well configured and distributed, especially for the application it was trained for. It serves as the initial condition and foundation for training the target application. This pre-trained model will then be trained again with labelled data for the new (target) application.

Employing pre-trained model, the required training data is not as large as that of training the machine learning model from scratch. The trade-off is that the configuration and many of the hyper-parameters of the neural networks are not tunable and can incur the computing cost of employing larger than optimum (for the target application) neural networks. Large Language models, e.g., GPT-4, BERT, etc. are powerful examples of pre-trained models for natural language. It is harder to find a suitable pre-trained graph-based machine learning model, as the data encoding for graph neural networks is generally very specific to the application. Fortunately, the training data requirement for various graph neural networks tends to be modest. For cyber components and software execution that are typically represented in graphs, it is still unclear how and how well transfer learning may help in model training with reduced data set.

Generative Adversarial Network (GAN) offers another attractive method in dealing with training data. A GAN consist of two neural networks, the generator and the discriminator. In a GAN setup, the two neural networks contest with each other in the form of a zero-sum game, where one agent's gain is another agent's loss. The generator strives to generate samples that fool the discriminator, and the discriminator strives to accurately detect or classify the generated samples. Both neural networks are trained together and co-evolve against each other. After the initial setup, the generator and discriminator will challenge and train each other in

an unsupervised manner. GAN requires minimal if any training data, making it very attractive for domains lacking large-scale, labelled data.

Unwise use of ML, such as employing generative adversarial network (GAN) without properly constraining it with physical models/rules, will likely violate the laws of physics and make it inappropriate or even dangerous to deploy. This is because unconstrained GAN will operate and explore solutions in an (un-grounded) virtual world with a much larger space than that of the physically constrained environment the CPS operates within. As GAN is a very attractive method, it is important to understand the problem space before deploying it. To illustrate, consider two slightly different applications. One is a valid application of GAN, and the other is not.

In the first application, a neural network is being developed for detecting malware (a discriminator). To anticipate 0-day malware, it is trained in a GAN environment—a malware generator neural network is developed and coupled with the malware detector in a GAN configuration, and then let loose (they play against each other). This is an appropriate and efficient way for inoculating the detector (discriminator) against 0-days.

In the second application, a dark-hat is mining for 0-day malware that is guaranteed effective against a target that is defended by VirusTotal. The dark-hat decided to deploy GAN, using a similar set up as the first application above. This is an ineffective solution, and the dark-hat will have false-confidence that his mined 0-day will be effective, for the following reason: his discriminator is not grounded to and does not represent VirusTotal. Developing a discriminator that can become a surrogate to VirusTotal will be very difficult if not impossible. VirusTotal and its evolution is influenced by factors that are not under the dark-hat control. The Dark-hat's discriminator will respond and evolve to the generator challenges in a manner that is independent of VirusTotal, and provide feedback to the generator that does not reflect VirusTotal behaviors. One can speculate that given enough resource and time, one can train a super discriminator that is better at detecting malware than ViriusTotal. However, unless one can prove or have well founded confidence that the superior discriminator is a complete superset of VirusTotal capability (no Malware that VirusTotal can detect the discriminator cannot detect), it still cannot provide the assurance that the synthesized malware will pass detection by VirusTotal.

3.3 Future Direction for AI in CPS

As discussed in previous sections, statistical models embodied as neural networks (machine learning) are effective in CPS related automation, including surrogate for control policy, automated fault recovery [19], surrogate as digital twin, anomaly detection, etc.

However, care must be taken in deploying neural networks as they are after all statistical machinery and hence cannot completely capture causality and are prone to make mistakes. Unless the utility property of the application itself is statistical,

an error detection and exception handler will be required to detect and mitigate the effect of incorrect neural network results. An application is said to have a *statistical utility* if occasional mistakes are expected and tolerated, as long as their frequency is not too large (below a certain threshold) and the overall performance of the algorithm is still above the acceptable performance level. That is, in an application with statistical utility, only average matters and individual error does not. An application does not have a statistical utility if an individual error/mistakes matters.

CPS is a field where the inertia of the physical systems can tolerate a limited duration of errors. However, an individual CPS is susceptible to prolonged errors. A system of CPS devices provides additional resilience, as long as the effects of prolonged errors in a subset of the system components are generally observable within the systems, and the overall systems adapt to the anomalies, or an operator can be alerted for and rectify the operational anomalies.

Various forms of neural networks and various configurations of systems of neural networks have been deployed in CPS infrastructures. Machine learning will also excel in approximating the modeling and controlling of the behavior of a complex system whose behavior is not easily describable with logic or sets of logic. The role of neural networks and systems of neural networks is expected to grow in CPS and process control & automation in general.

Machine learning excels when the utility of the application itself is statistical, and when the application logic is extremely complex to be completely captured using logic or other formal methods. For this reason, an understanding of the problem's space, property and the environment surrounding the problem is the key for successful application of neural networks and the selection of the particular neural network algorithms. A good understanding of problem space will also help avoid pitfalls described in the previous section.

During our journey of studying the property of a problem or task and their potential solutions, the authors observed that *it is easier to solve problems of statistical nature with statistics*, and vice versa, *it is simpler to solve problems of logical nature using logical process*. This dichotomy is analogous to the dichotomy of frequency domain and time domain in signal processing. There are classes of problems that are simpler to solve in frequency domain, and there are other classes that lend themselves to have natural solutions in time domain. In general however, while less efficient or precise, statistical process can be used to approximate a logical one, and logical process (such as logic in modern digital computer) can emulate/simulate statistics.

A system of machine learning algorithms can be arranged in logical manners or simply feed each other for large scale automation. Properly designed systems of machine learning algorithms may mask or alleviate individual algorithm weakness and provide much more accurate and capable ultimate results. There may also be the case where it is prudent to include logical reasoning algorithms into the systems of neural networks, creating hybrid symbolic and statistical (neural networks) systems.

It can be argued that arranging multiple neural networks in a logical pipeline has already shown the promise of hybrid logical-statistical system design. For example,

ONR developed the Learn2Reason concept [20, 21], advocating the development of a hybrid system of neural networks algorithms and logic-based reasoning. The development of Learn2Reason was inspired by Daniel Kahneman's system-1 and system-2 concept with respect to thinking fast and slow [22]. Initial description of Learn2Reason [20] suggests a blackboard like implementation where the logical and probabilistic/statistical process interact, however, most of the research mentioned in [21] employed the pipelines structure. Recent news [23] indicates that Google's large language model employs logic, in term of generated program, to solve particular tasks where logical processes clearly surpass statistics, e.g., counting, performing arithmetic calculation, reverse spelling a word, etc. The article also stated that Google was following Kahneman's system-1 and system-2 concept [22] in this work.

A hybrid logical and statistical (neural network) based machine learning is the future. It allows for both statistical process and logical process to do what it can do best, and together they provide a superior performance than that of each individual type (logical or statistical). This hybrid learning system will find its place in CPS and CPS-based critical infrastructure of the future.

References

1. Stouffer K et al (2022) Guide to operational technology (OT) security. NIST ITL Computer Security Resource Center. https://doi.org/10.6028/NIST.SP.800-82r3.ipd
2. Higgins KJ (2023) OT network security myths busted in a pair of hacks. Available via DARKReading: ICS/OT Security https://www.darkreading.com/ics-ot/ot-network-security-myths-busted-in-a-pair-of-hacks Accessed 6 Sep 2023
3. Kim T et al (2020) From control model to program: investigating robotic aerial vehicle accidents with MAYDAY. In: 29th USENIX security symposium. Boston
4. Fei F et al (2018) Cross-layer retrofitting of UAVs against Cyber-physical attacks. In: IEEE international conference on robotics and automation (ICRA). Brisbane
5. Son Y et al (2015) Rocking drones with intentional sound noise on gyroscopic sensors. In: 24th USENIX security symposium. Washington DC
6. Learning Introspective Control (LINC). https://www.darpa.mil/program/learning-introspective-control Accessed 6 Sep 2023
7. Faithful Integrated Reverse-Engineering and Exploitation (FIRE) https://defencescienceinstitute.com/funding-opportunity/darpa-fire/ Accessed 6 Sep 2023
8. Lee EA (2016) Fundamental limits of cyber-physical systems modeling. ACM Trans Cyber-Phys Syst 1–26. https://dl.acm.org/doi/10.1145/2912149
9. Box GEP, Draper NR (1987) Empirical model-building and response surfaces. Wiley, New York
10. Saltzer JH, Kaashoek MF (2009) Principles of computer system design, an introduction. Morgan Kaufmann, Burlington
11. Tu Z et al (2018) Redundancy-free UAV sensor fault isolation and recovery. Preprint at https://arxiv.org/pdf/1812.00063v1.pdf
12. Xu M et al (2017) Compositing security mechanisms through diversification. In: USENIX annual technical conference (ATC'17), Santa Clara
13. Clarke E et al (2000) Counterexample-guided abstraction refinement. In: International conference on computer aided verification. Chicago

14. Klein G et al. (2014) Comprehensive formal verification of an OS microkernel. ACM Trans. Comput Syst 32.1:1–70
15. Briskin G, Li JH (2022) Binary code randomization for attack sensitive software (BRASS). Final report to the office of naval research. Available to US performers upon request and approval
16. Mertoguno JS et al (2019) A physics-based strategy for cyber resilience of CPS. In: Proceeding of the SPIE 11009, autonomous systems: sensors, processing, and security for vehicles and infrastructure. Baltimore
17. Arroyo MA et al (2019) YOLO: frequently resetting cyber-physical systems for security. In: Proceeding of the SPIE 11009, autonomous systems: sensors, processing, and security for vehicles and infrastructure. Baltimore
18. Al Maruf A et al (2023) A timing-based framework for designing resilient cyber-physical systems under safety constraint. ACM Trans Cyber-Phys Syst 7(3):1–25
19. Fei F et al (2020) Learn-to-recover: retrofitting UAVs with reinforcement learning-assisted flight control under cyber-physical attacks. In: Proceeding of the IEEE international conference on robotics and automation (ICRA). Virtual
20. Mertoguno JS (2014) Human decision making model for autonomic cyber systems. J Artif Intell Tools 23(6):1–6. https://doi.org/10.1142/S0218213014600239
21. Mertoguno JS (2019) Toward autonomy: symbiotic formal and statistical machine reasoning. In: Proceeding of the 1st IEEE international conference on cognitive machine intelligence. Los Angeles
22. Kahneman D (2013) Thinking fast and slow. Farrar, Straus and Giroux, New York
23. Amadeo R (2023) Google's Bard AI can now write and execute code to answer a question. ars TECHNICA. https://arstechnica.com/google/2023/06/googles-bard-ai-can-now-write-and-execute-code-to-answer-a-question/ Accessed 6 Sep 2023