DEPARTMENT OF COMPUTER SCIENCE, SYSTEMS AND COMMUNICATION

REDS Lab
Real time Events & Decision Support

# In-Kernel Traffic Sketching for Volumetric DDoS Detection

Mingyuan Zang*, Federico de Iaco†, Jie Wu‡, **Marco Savi**†

*Cloud Computing Research Institute, China Telecom, China

†Department of Computer Science, Systems and Communication, University of Milano-Bicocca, Italy

‡Department of Computer and Information Sciences, Temple University, USA

Montreal (Canada), June 9th, 2025

IEEE ICC®

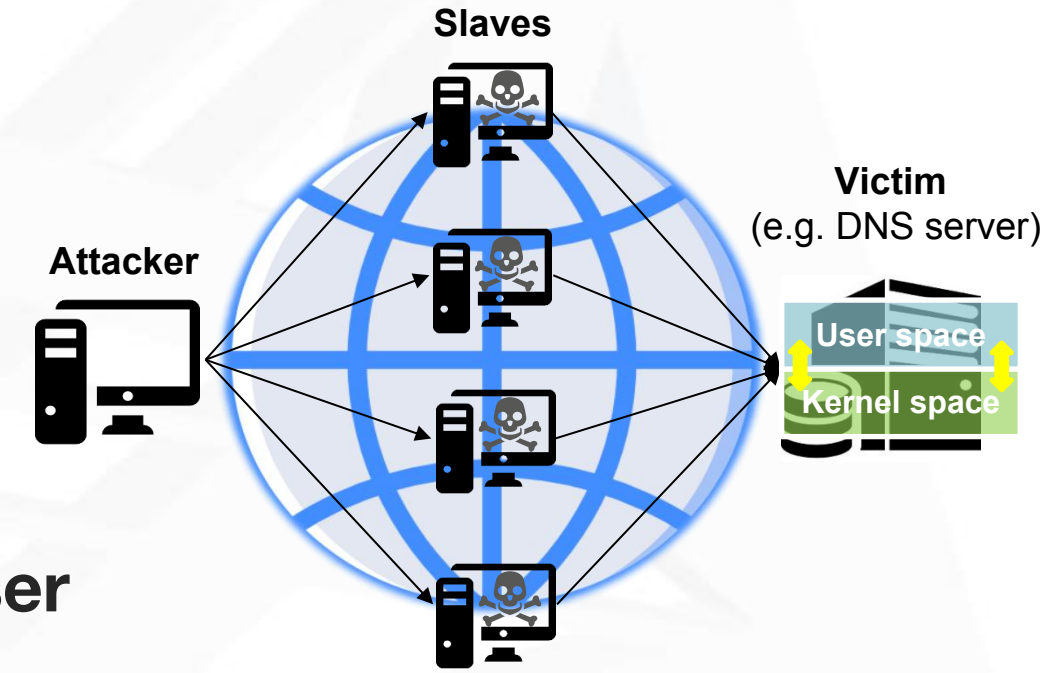
# Motivation

- Cloud and edge networks are increasingly vulnerable to **volumetric DDoS attacks**

- **Two detection possibilities:**
  - Network-based detection
  - Host-based detection ← Our focus

- **Host-based detection** is performed in **user space**
  - **Problem**: non-negligible latency due to **kernel-user space transitions**

→ Need for **faster detection**, within kernel space

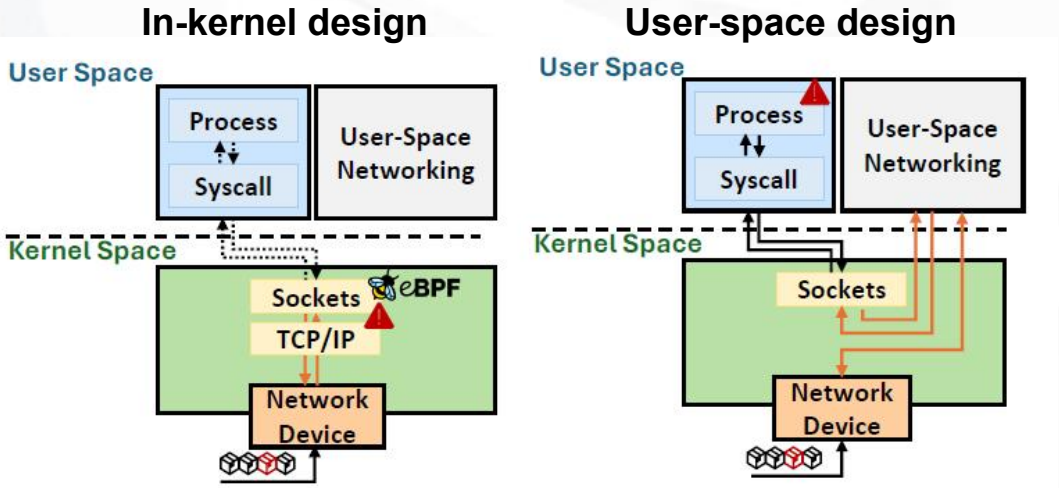





UNIVERSITÀ DEGLI STUDI DI MILANO BICOCCA

# Problem Statement

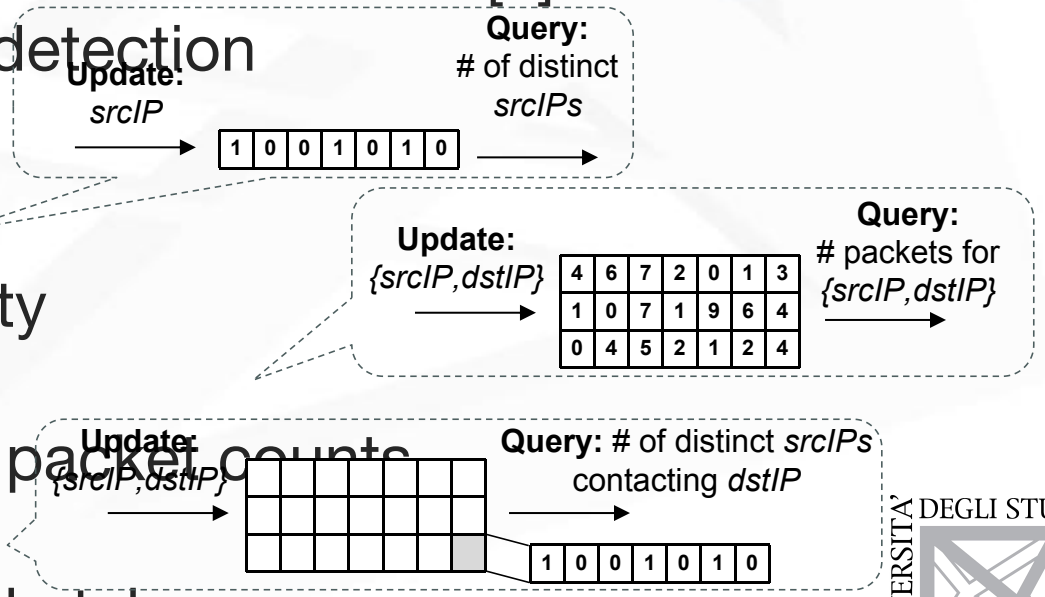
- Can we design **low-latency** host-based DDoS detection system that **fully** operates in **kernel space**?
- Technological playground: extended Berkeley Packet Filtering (**eBPF**) and eXpress Data Path (**XDP**)
  - High-performance packet processing in the kernel
  - eBPF programs run safely without kernel modification




- **Constraints:** limited memory, eBPF (programming) restrictions

# Our Proposal: Adopting Sketches

- **Sketches:** probabilistic **data structures** for efficient summarization of **network traffic statistics** using **compact memory**
  - **Update** to populate the sketch, **query** to retrieve estimated statistics
  - Often adopted for in-network P4-based DDoS detection [1]
  - Underexplored potential in host-based detection




- Adoption of **BACON sketch** [1]
  - **Direct Bitmap:** estimates flow cardinality
  - **Count-min Sketch:** estimates per-flow packet counts
  - **BACON** = Direct Bitmap + Count-min Sketch

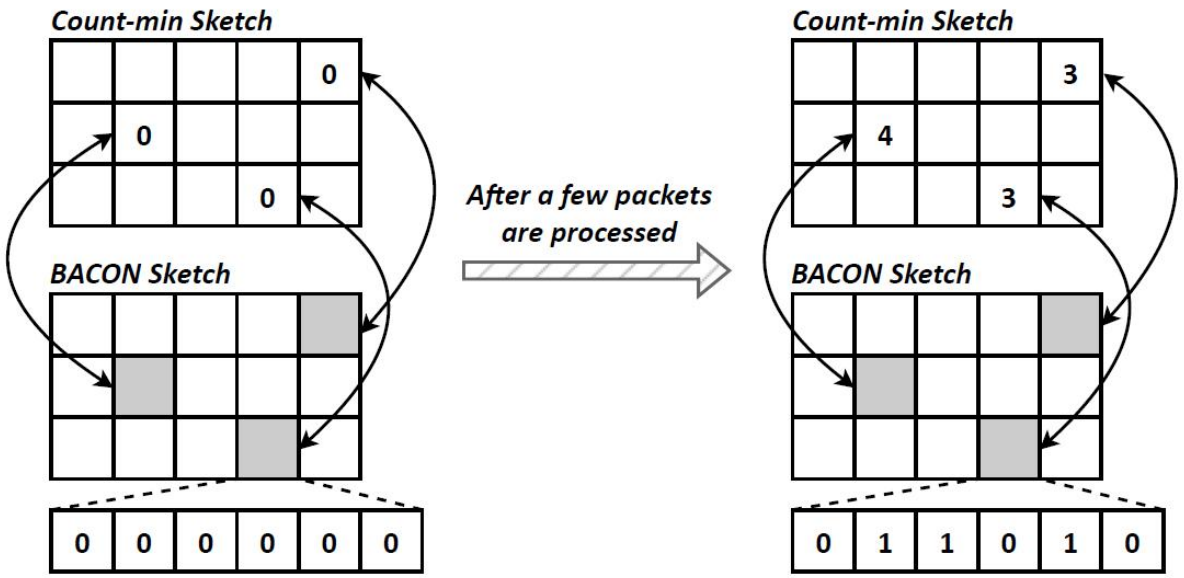
[1] D. Ding, et al., In-Network DDoS Victim Identification Using Programmable Commodity Switches, in IEEE TNSM, vol. 17, no. 2, pp. 1191-1202, Jun. 2021

# Proposed Design

1. Full **in-kernel packet analysis** (*Update* + *Query*) by means of BACON Sketch
   - *{srcIP,dstIP}* extrapolated from packet header
   - Threshold set to detect an abnormal number of *srcIPs* contacting the victim *dstIP*
   - Sketch reset every time window
2. Adoption of an **auxiliary sketch** for BACON Sketch's **Query optimization**
3. **Double BACON sketch** mechanism to **avoid data loss** during threshold updates and sketches reset
4. **Self-adaptive threshold** via Exponentially-Weighted Moving Average (EWMA) to capture traffic variations

# Auxiliary Sketch for Query Optimization

- **Problem:** counting the number of 1s inside the Bitmap causes a computational bottleneck in the BACON Sketch's Query operation
- **Solution:** adoption of an auxiliary Count-min Sketch
  - When an *Update* flips a Bitmap's bit from 0 to 1, the corresponding counters of the auxiliary sketch are increased
  - Query is perfor N sketch

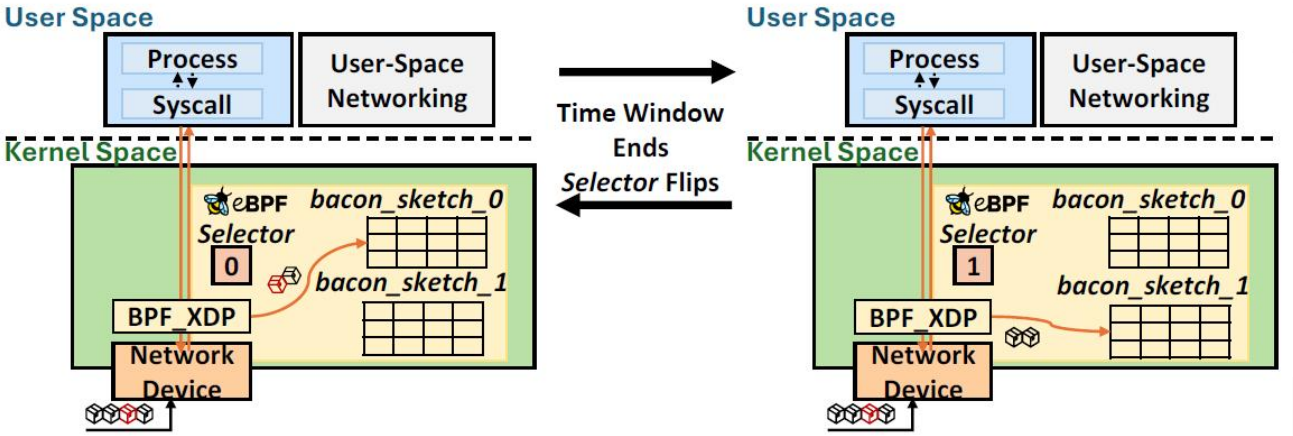


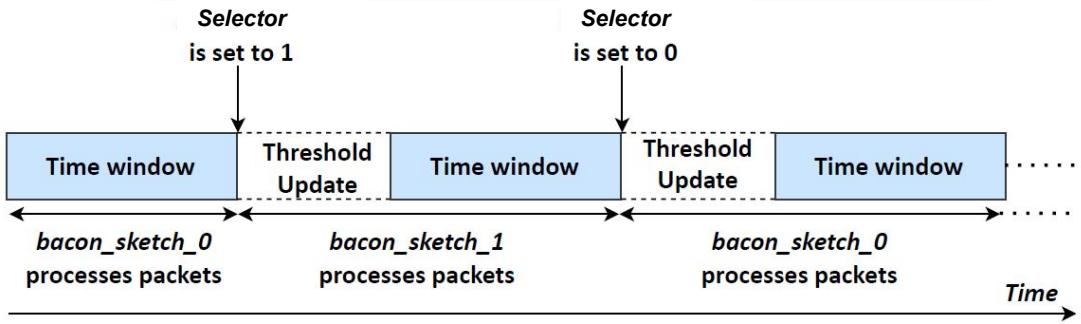



UNIVERSITÀ DEGLI STUDI DI MILANO BICOCCA

# Double BACON Sketch Mechanism

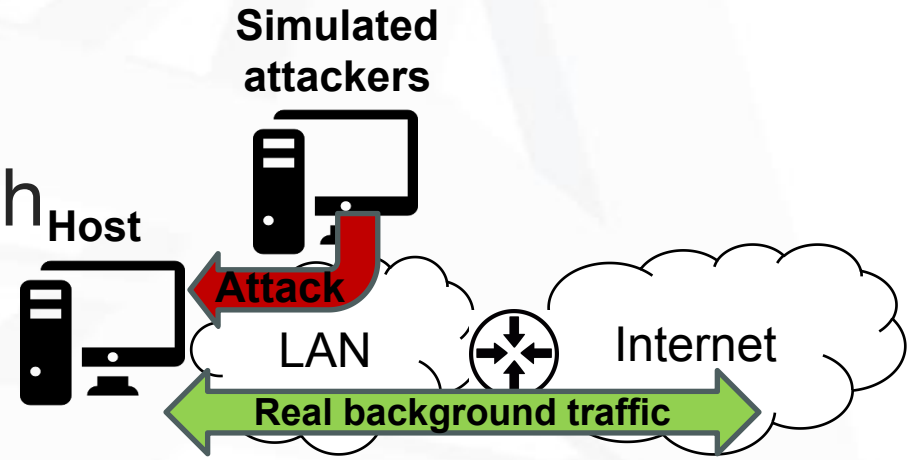
- **Problem:** if threshold update and sketches reset require *n* seconds, packets are not processed for *n* seconds
  - Need to access to shared memory structures, which cannot be concurrently updated
- **Solution:** adoption of two BACON sketches → *bacon_sketch_0* and *bacon_sketch_1*
  - Alternating between the two sketches in consecutive time windows
  - Requires 2x memory consumption, but greatly improves accuracy → Packets all timely



# Experimental Setup

- **Environment:** Ubuntu 22.04 with Kernel 6.5
- **Traces:** simulated DDoS attacks [2] with varying source IPs and packet volumes mixed to real background traffic
  - Generated traces released as open source [3]
- **Sketch:** 25MB memory per BACON sketch



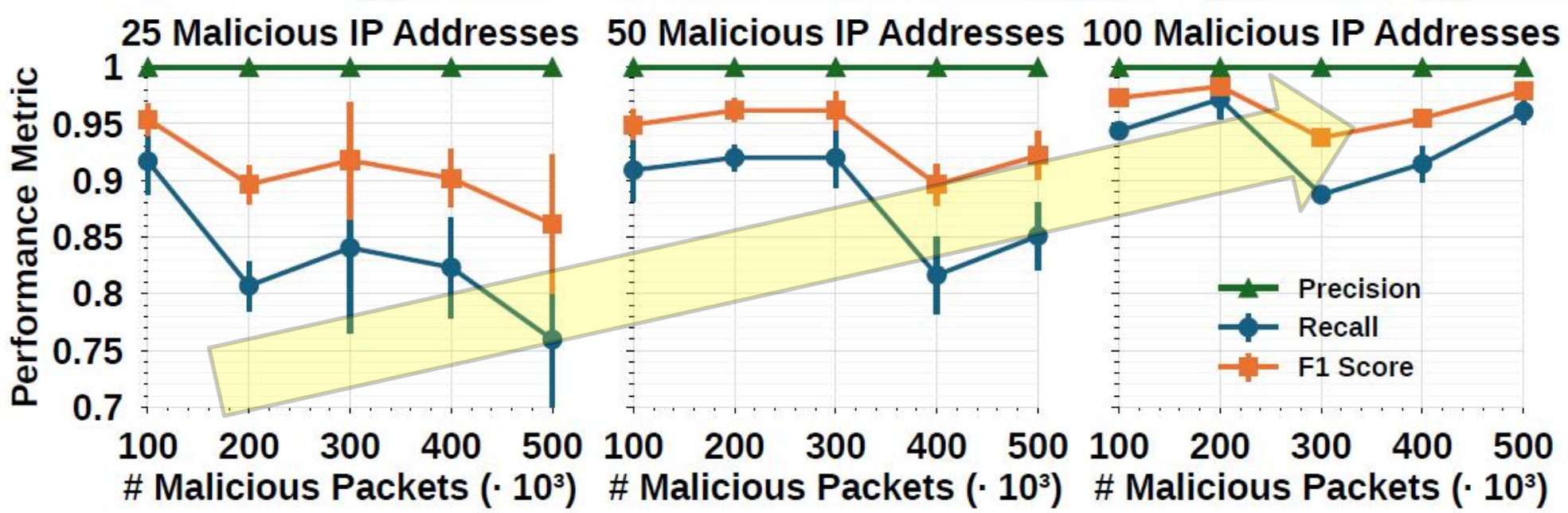

[2] https://github.com/ricardojoser/ddos_simulation/tree/master
[3] https://dx.doi.org/10.5281/zenodo.14049439



UNIVERSITÀ DEGLI STUDI DI MILANO BICOCCA

# Performance Evaluation (1)

- Increasing **# malicious IP addresses** (25, 50, 100) and **# malicious packets** per malicious IP address (100 to 500)




**Precision** always very high

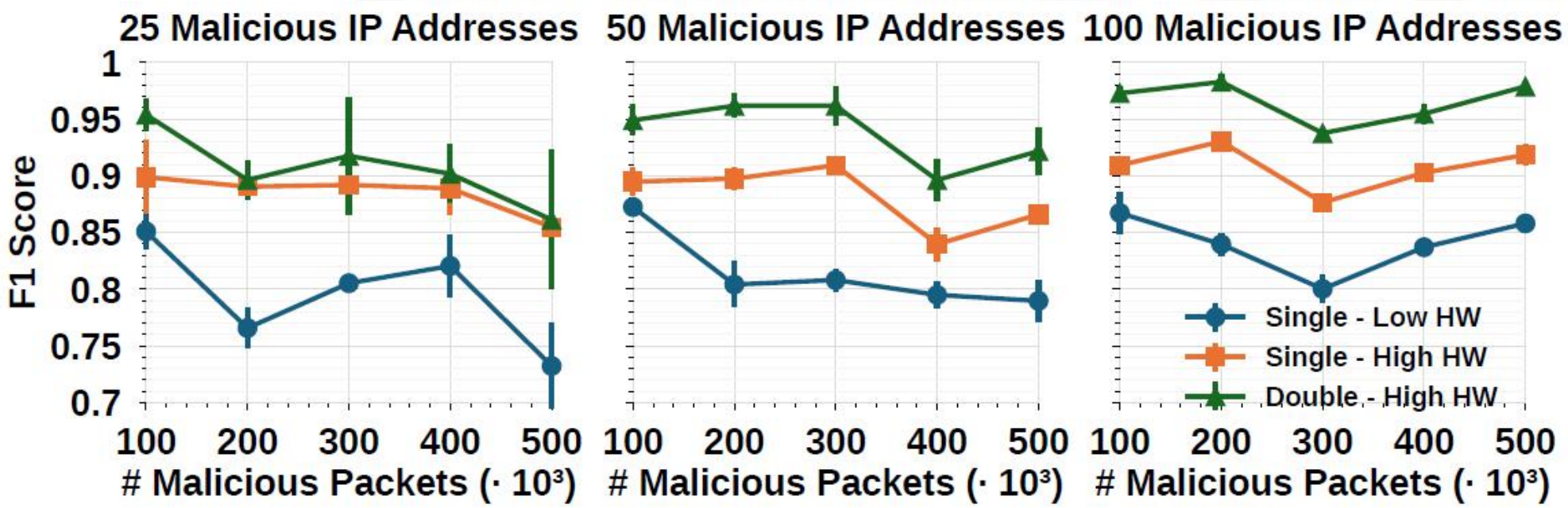
**Recall** & **F1 score** higher for higher # malicious IP addresses



UNIVERSITÀ DEGLI STUDI DI MILANO BICOCCA

# Performance Evaluation (2)

- **Double** vs. **single** BACON sketch, considering different performing hardware:
  - **Low HW:** 8 core @ 2.0 Ghz & 8 GB RAM
  - **High HW:** 12 core @ 3.7 Ghz & 16 GB RAM




**Higher performance** and **robustness** to hardware variation of double sketch w.r.t. single sketch



UNIVERSITÀ DEGLI STUDI DI MILANO BICOCCA

# Performance Evaluation (3)

- **Kernel vs. user space** implementation: **detection time** comparison

| Strategy | Mean |
|---|---|
| BACON Sketch (kernel space) | 0.52 $\mu s$ |
| BACON Sketch (user space) | 4.2 $ms$ |

! ~ **4 orders of magnitude faster** in kernel space!

# Conclusion & Future Work

- In-kernel sketching enables **resource-efficient DDoS detection**
  - Both Update and Query operations can be performed in kernel space
- The proposed solution effectively tackles some eBPF inefficiencies
- **Good detection performance** (F1 Score > 90%) and **fast detection time** are ensured

- **Future work:**
  - Explore other sketching methods (e.g. **HyperLogLog**)
  - Expand the solution to detect **more attack types**

# THANK YOU!

marco.savi@unimib.it

https://gitlab.com/federicodeiaco1/bacon_sketch

*The research leading to these results has been partially funded by the Italian Ministry of University and Research (MUR) under the PRIN 2022 PNRR framework (EU Contribution – NextGenerationEU – M. 4,C. 2, I. 1.1), SHIELDED project, ID P2022ZWS82.*

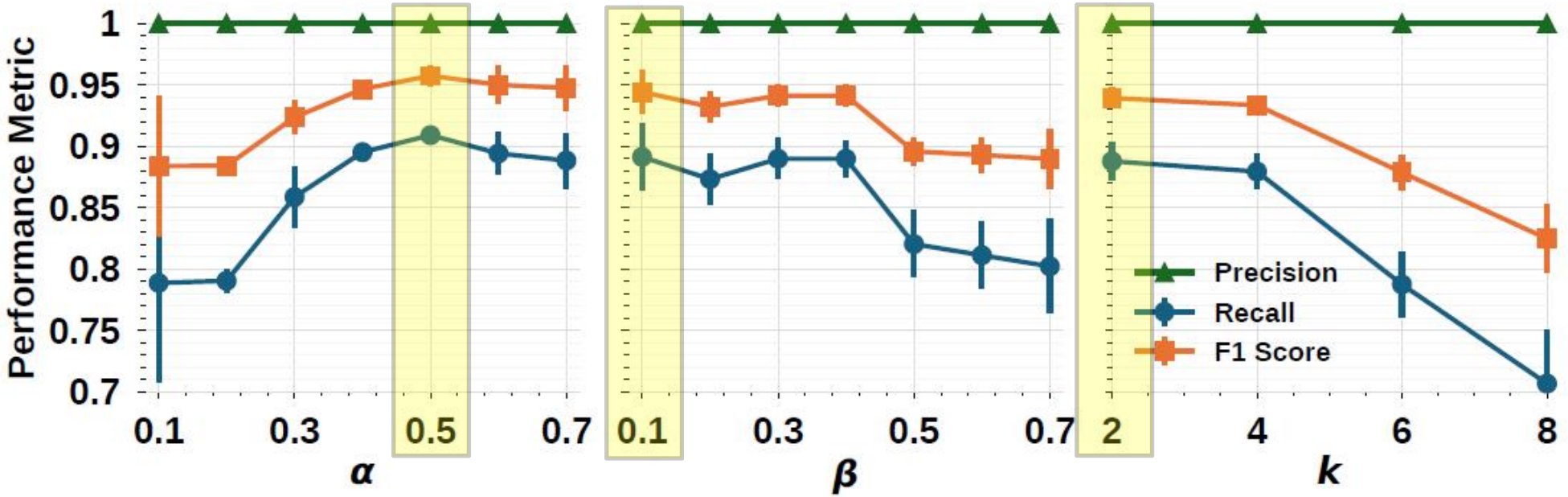
# Self-Adaptive Threshold

- **Problem:** a static threshold leads to inaccurate detection due to network traffic intensity dynamicity
- **Solution:** adoption of an EWMA-based dynamic threshold
  - Based on an estimation of the number of sources contacting the destinations ($E_{dst}$)

- Threshold $Th_{dst}$ parameters to be a-priori set:
  - $\boldsymbol{\alpha}$: how much does current $E_{dst}$ impacts on average value $Ave_{E_{dst}}$?
  - $\boldsymbol{\beta}$: how much does current deviation of $E_{dst}$ from $Ave_{E_{dst}}$ impact on deviation $Dev_{E_{dst}}$?
  - $\boldsymbol{k}$ (margin) → $Th_{dst} = Ave_{E_{dst}} + k \cdot Dev_{E_{dst}}$

# Performance Evaluation (3)

- Sensitivity analysis to **dynamic threshold parameters** $\alpha$, $\beta$ and $k$




- **Kernel vs. user space** implementation: **detection time** comparison

| Strategy | Mean |
|---|---|
| BACON Sketch (kernel space) | 0.52 $\mu s$ |
| BACON Sketch (user space) | 4.2 $ms$ |

**~ 4 orders of magnitude faster** in kernel space!