Collusion Attack Detection in Networked Systems

Md Zakirul Alam Bhuiyan and Jie Wu
Department of Computer and Information Sciences, Temple University, Philadelphia, PA 19122, USA

***Abstract*—Security protocols have been commonly used to protect secure communication in networked systems. It is often assumed that individual wireless nodes or leaders in a system are sincere and use techniques (authentication, permission, etc.) of these protocols to have secure communications. We discover that such protocols may be leaked by a sophisticated collusion attack (a type of attacks in which a node intentionally has a secret agreement with an adversary, or is compromised by an adversary). Before an attack is made, the node seems to be working properly, communicating with others, and providing correct values/decisions. Currently, there is no systematic method for detecting such an attack. In this paper, we propose CAD, a Collusion Attack Detection scheme for networked systems. We think that wireless nodes usually have some correlation patterns in communication metrics (e.g., radio timing, amount of packets transmitted). When there is a significant discrepancy in such patterns with a node, the node is said to be colluded. We evaluate CAD in simulations with real data traces. Evaluation results demonstrate that CAD achieves a collusion detection rate up to 92%, which is at least 50% better compared to existing schemes.**

***Index Terms*—Security, collusion attack, networked systems, wireless networks, wireless sensor networks, correlation**

I. Introduction

Networked systems are systems composed of dynamic units or nodes that interact over a network. There are different types of networked systems, including wireless networks, wireless sensor networks (WSNs), wireless ad hoc networks (WAN), e-commerce systems, and so on [1], [2]. These systems are easily prone to security attacks [3]–[5]. Often, they are unattended and unprotected. Some inherent features, like limited battery and low memory, make the systems infeasible for using conventional security solutions. There are a variety of attacks on these systems, which can be generally classified as routing attacks and data traffic attacks. Some of the data attacks in network systems are wormholes, sinkholes, jamming, selective forwarding, and Sybil attack. There are a lot of mitigation protocols/schemes used as a defense against these types of attacks [5]–[10].

Some attacks are assumed to be with data aggregation. Robust data aggregation is a serious concern in networked systems, and there are a number of research efforts investigating malicious data injection by taking into account various adversary models [3], [11], [12]. Trust and reputation systems are being suggested as effective security mechanisms for distributed systems. Since networked systems are being increasingly deployed in many application domains, assessing the trustworthiness of reported data from networked nodes has still remained a challenging issue. Trust and reputation, in particular, play a critical role in WSNs as a method of resolving a number of important problems, e.g., secure data routing, fault tolerance, false data detection, compromised node detection, secure data aggregation, outlier detection, signal perturbation, and so on [11], [13], [14]. When given a WSN, sensors deployed in hostile environments may be subject to node compromising attacks by adversaries who intend to inject false data into the system. In this context, assessing the trustworthiness of the acquired data becomes a difficult task.

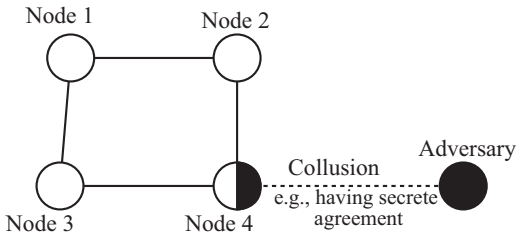


Fig. 1. A snapshot of a collusion attack.

In many security protocols/methods, it is often assumed that individual nodes or leaders in the networks are sincere and use techniques (authentication, permission, etc.) of these protocols in communications. However, these protocols may be leaked as the result of a sophisticated collusion attack. This is an attack when a node intentionally has a secret agreement with an adversary, or is compromised by an adversary that has a high level of knowledge regarding transmission, aggregation algorithm, and so on, as shown in Fig. 1. Before experiencing a worse situation in the system, the node seems to function properly, communicating with others, and providing correct values/decisions. When there is a attack of collusion, a minor variation at the colluded node behavior occurs, which can be the results that the adversary can read or inject information. When the attackers have a high level of knowledge about the system, transmission model, aggregation algorithm, and its parameters, they can conduct sophisticated attacks on the systems by exploiting false data injection through one or more compromised nodes.

In this paper, we propose a collusion attack detection scheme, called CAD. It is based on correlations of network components or performance metric. During the network operation, at some point of time, we may find network performance degradations by observing performance metrics, e.g., a high or low packet delivery ratio (PDR). The nodes which are colluded can be simply recognized by such performance metrics. For

example, a node may generate apparently abnormal packets in comparison to its neighbors. Considering low-power and listening mode of a node, the radio is switched on only for receiving, sending, or idle listening. Consequently, the radio-on time should be closely correlated with the amount of traffic passing through the node. If a node is colluded at some point of time, the number of packets transmitted to or from it increases or decreases much more than usual in comparison both to its own, and its neighboring node transmissions. If any individual amount of the performance metrics is not irregular, the correlation between the *radio-on-time* and *the number of transmitted packets* undoubtedly implies a discrepancy on that node. This suggests that the node either has a secret agreement with an adversary (malicious party), or it is compromised.

To handle these problems, CAD includes two sub-detection schemes, namely, temporal and spatial collusion attack detection schemes, respectively. We think that wireless nodes usually have some correlation patterns in system communication metrics (e.g., radio-on time, number of packets transmitted, number of negative acknowledgment-NACK). When there is a significant discrepancy in such patterns with a node, the node is assumed to be colluded. In CAD, we take advantage of the correlations among the performance metrics of every node, using a correlation map that describes the latent or over-active status of the node. The correlation map can characterize the actual attack situation on the map. Such a correlation map is updated regularly using the node's performance metrics.

We make the following main contributions in this paper:

- We design CAD for collusion detection. We do not assume any predefined principle behaviors/rules in CAD so that it can be applied to other applications, such as social networks.
- We provide a correlation mapping of network performance metrics that distinguish internal correlations inside a node. We propose algorithms for detecting collusion information in the temporal and spatial dimensions in a distributed manner.
- We evaluate CAD through simulations using real data traces. The evaluation results demonstrate collusion detection rate, up to 92%, under the collusion information injection.

The rest of this paper is organized as follows. Section II reviews the related work. Section III provides the CAD design. Section IV offers the CAD scheme, including the two detection scheme. Section V evaluates CAD performance. Finally, we conclude the paper and recommend future work in Section VI.

II. Related Work

A variety of security attack detection has been in the literature, including wormhole, sinkhole, jamming, selective forwarding, and Sybil attack [6]–[9]. Some attacks are assumed to be with data aggregation. Robust data aggregation is a serious concern in distributed systems, and there are a number of research efforts investigating malicious data injection by taking into account various adversary models. Trust and reputation system are being suggested as an effective security mechanism for distributed systems. Since distributed systems are being increasingly deployed in many application domains, assessing trustworthiness of reported data from distributed nodes has still remained a challenging issue. Particularly, trust and reputation systems play s critical role in WSNs as a method of resolving a number of important problems, e.g., secure data routing, fault tolerance, false data detection, compromised node detection, secure data aggregation, cluster head election, outlier detection, and so on [11], [13], [14]. When given a WSN, sensors deployed in hostile environments may be subject to node compromising attacks by adversaries who intend to inject false data into the system. In this context, assessing the trustworthiness of the acquired data becomes a difficult task.

Key and cryptography based collusion attack detection is proposed in [15]. When a newly user colluded with a revoked user can recover the group session keys. It can detect some basic collusion attack during packet encryption and description. Later, this scheme is improved by [10]. It controls self-healing key-distribution to defend a collusion attack, which is based on one-way key chains and secret sharing in WSNs.

Iterative Filtering (IF) techniques are used in WSNs for security. The reason is that these can provide solution for both data aggregation and data trustworthiness assessment by exploiting a single iterative procedure [16]. Each sensor's trust estimation is based on the distance of the readings between the current readings and previous readings, and distance between its own readings and the readings of all neighbors. Sensors whose readings significantly differ from other estimate are assigned less trustworthiness. Subsequently, sensors' readings are given a lower weight in the aggregation process in the current iteration round. Utilizing the similar idea, a large set of work has been suggested on IF algorithms for trust and reputation systems [3], [16]–[18].

Recently, the performance of IF algorithms for data aggregation in the presence of collusion attacks has been studied [3]. This work shows IF algorithms simultaneously aggregate data from multiple sources and provide trust assessment of these sources, usually in a form of corresponding weight factors assigned to data provided by each source. It also shows that current IF algorithms, while significantly more robust against collusion attacks than the simple averaging methods, are nevertheless susceptible to collusion attack. However, the aggregation is affected when cullsution attack is made at the node level, which is not discovered.

Looking into more details, during the network operation, at some point of tome, the network performance may degrade, e.g., low packet delivery ratio (PDR). A portion of colluded nodes can be easily recognized by such performance metrics. For example, such nodes generate apparently abnormal packets. For instance, considering low-power node and listening mode, the radio is switched on only for receiving, sending, or idle listening. Consequently, the radio-on time should be closely correlated with the amount of traffic passing the node. If a node is colluded at some point of time, transmission of a number of packets from it become very lower or higher

than its usual transmission, or than itself or its neighboring node transmission. Any individual amount of the performance metrics is not irregular, but the correlation between radio-on time and the number of transmitted packets undoubtedly implies discrepancy on that node. This suggests that the node either compromised (having a secret agreement with the adversary (malicious party) or it is in collusion attack.

To handle these problems, in this paper we propose CAD, a lightweight collusion attack detection scheme. We think that wireless nodes usually have some correlation patterns in system communication metrics (e.g., radio-on time, number of packets transmitted). When there is a significant discrepancy in such patterns with a node, the node is assumed to be colluded. In CAD, we take advantage of the correlations among performance metrics of every node using a correlation map that describes the latent or over active status of the node. The correlation map can characterize the actual attack situation in the map.

III. The Design of CAD

In this section, we present the design of the collusion attack detection (CAD) scheme. This includes system model, adversary model, and the reputation in the system.

Definition 1. [Collusion Attack] A type of security attack or threat in which a node intentionally makes a secret agreement with an adversary, or the node is somehow made to have such an agreement. The adversary may collect confidential information from the system, and then conduct sophisticated attacks on the system by exploiting false data injection through one or more compromised nodes.

A. System Model

A system can be any kind of networks (e.g., sensor network, Wi-Fi network, social network, peer-to peer system, etc.) consisting a set of N nodes. The system is in charge of performing certain types of application tasks, e.g., sensing, data collection (such as temperature), or making transactions or interactions. Each node is responsible for sending its communications to its neighboring nodes towards a control center or base station (BS). Nodes uses beacon messages for synchronization among them. Each node is able to talk to some of its neighboring node using given communication range. The nodes communicate with each other by broadcasting messages across channels. These are assumed to be symmetric. We assume that individual nodes might be colluded and might be collecting true data but sending false data, or forwarding the data to adversary nodes of interest. We assume that nodes of a network have limited resources (energy, computation power, and bandwidth).

B. Adversary Model

We use a Byzantine attack model, in which we consider an adversary as a special node. The adversary can compromise a set of nodes (making secret agreements with them) and inject any false data through the compromised nodes or collect information from the nodes. Subsequently, we assume that some nodes can be physically colluded (by modifying some hardware components). We assume that when a node is colluded, all the information which is inside the node becomes accessible by the adversary. Thus, we cannot rely on cryptographic or trustworthy methods for preventing the attacks, since the adversary may extract cryptographic keys from the compromised nodes, even the node may expose its key. We assume that, through the colluded nodes, the adversary can send false data to the network with the purpose of distorting the final decisions on the applications tasks, or collecting the final decisions. We also assume that a single outsider or adversary, or a colluding group of adversaries, may take control of colluded nodes, even enabling the nodes to commence a sophisticated attack.

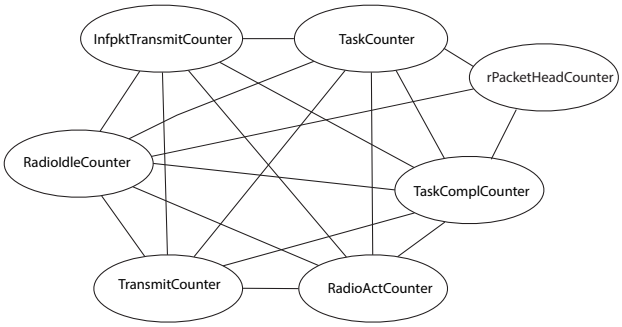


Fig. 2. Communication metrics and their correlation.

We also assume that the adversary may be intelligent. Through agreement with a node, the adversary can enable a node to work normally in some periods of time. During this duration, the adversary remains silent. Then, it passes information at another period. The adversary may get information at a different period for different nodes of the network. Interestingly, existing security protocols may not capture this attack.

C. Reputation Systems

The idea of a reputation system can be used for trustworthy and privacy systems. We consider tracking reputation score for the possibility of a collusion attack, computed by a number of system performance metrics.

A network can have various performance metrics. We focus on a number of metrics of each node, as shown in Table I. These metrics can be classified into four types:

- **Time-based metrics.** This measures the cumulative radio active time. Example includes RadioOnCounter.
- **Traffic-based metrics.** This measures the cumulative number of packets transmitted by a node. Example includes TransmitCounter, ReceverCounter
- **Task-based metrics.** This measure the cumulative number of tasks executed. Example includes TaskExecCounter, TaskPostCounter
- **Authentication-based metrics.** This is based on secure network communications. Example includes AuthenticationCounter,

TABLE I
EXAMPLES OF PERFORMANCE METRICS

Metric	Description
TaskExecCounter	the number of tasks executed
FTaskExecCounter	the number of tasks' completion failure
TaskPostCounter	the number of tasks posted in the system
AuthenticationCounter	the given type of authentication method used
AckCounter	the number of ACKs sent
AuthenticationNCounter	the given authentication method is altered
NackCounter	the number of NACKs sent
NackCounter	the number of no acknowledgments
RadioOnCounter	the number of times radio on
DupCounter	the number of duplicated packets
RadioOnTime	the amount of time radio on
SuccAckCounter	the number of successfully transmitted packets
TransmitCounter	the number of packets transmitted
NACKRetransmitCounter	the number of packets negative transmitted
RetransmitCounter	the number of retransmissions
FTransmission	the frequency of transmissions for a given task
ReceverCounter	the number of packets received
NTransmission	the frequency of transmissions by a neighbor
InfpktTransmitCounter	the number of infrequent packet transmissions
rPacketHeadConter	the ability to replace data packets

Based on the performance metrics, we find temporal and spatial correlation between them. Note that some metrics may not be correlated at some point of time. That is, the set of metrics at a given time may be a little bit different from other time. In the case of the absence of a metric at a given time, it is discarded from the correlation measurements in order to reduce a false possible detection in a collusion attack. A subset of metrics and their correlations can be seen in Fig. 2. Each of the metrics conveys particular performance information. For example, NACK is the number of negative acknowledgments, NoACK is the number of times that there are no acknowledgments received, RetransmitCounter is the number of retransmissions, and so on. In practice, each of the metrics has a value.

Suppose that, during one time period, the value of a metric changes dramatically, possibly indicating that the node is under collusion attack. However, the node seems to be working properly, communicating with neighbors, exchanging messages, and performing authentication using some security protocols; it even returns correct values or decisions. However, it might secretly pass confidential information to outsiders or to the adversary. It can add some information with its correct transmissions. This attack problem is the most sophisticated one. However, the reputation score of nodes can be weak based on the values of the metrics; even though many metrics appear to be stable during another period, the node actually encounters an attack. For a better detection, we assume that the individual inspection of metrics on the same node may overlook collusion attacks, or may flag detection failures by error. Considering multiple nodes, an individual metric is insufficient to uncover collusion attacks. As a result, the reputation scores achieved by individual metrics or nodes may not be so accurate. We use a correlation matrix for all the metrics that include the value/status of the metrics. Then, the reputation scores based on correlation maps of metrics are calculated so that they can imply collusion attack information. In this respect, we can define that the node whose reputation score is different from its neighboring nodes is a colluded node. We can find a snapshot in Fig. 1, where node 2 is supposed to be colluded.

IV. COLLUSION ATTACK DETECTION

In this section, we discuss how the detection of collusion attacks is performed. We first provide the correlation map between metrics. Then, we provide the CAD scheme that includes two sub-schemes.

A. Correlation Map

We develop a correlation map, which is a map representing the pairwise correlations of the metrics. Each metric of the map can be called a *vertex* and a relation/connection between any two metrics is called an *edge*. We assume that the network works in a periodic manner, where each period consists of a set of discrete time periods. The map is constructed and maintained periodically for each node in the system. At each discrete time period t, a node s_i measures its working status, i.e., the status for its metrics. Let m be the total number of metrics and $v_{x,t}$ are the value of the xth metric at time t, $1 \leq x \leq m$. It then has a status vector $S_{i,t} = (u_{1,t}, u_{2,t}, ..., u_{m,t})$. Each edge of the map has a weight that implies the reputation score between the corresponding metrics.

The correlation between metrics are estimated in each time slot $\tau \in t$, which survives from time $(\tau - 1) * w + 1$ to time $\tau * w$, that is $[(\tau - 1) * w + 1, \tau * w]$. Here, w is the size of slot τ. Suppose that $V_{x,k}$ and $V_{y,k}$ are the values of metrics x and y collected in time window τ, respectively. We can have the following metrics for the values:

$$V_{x,k} = (u_{x,(k-1)*w+1}, u_{x,(k-1)*w+2}, \ldots, u_{x,(k-1)*w+w})$$

$$V_{y,k} = (u_{y,(k-1)*w+1}, u_{y,(k-1)*w+2}, \ldots, u_{y,(k-1)*w+w})$$

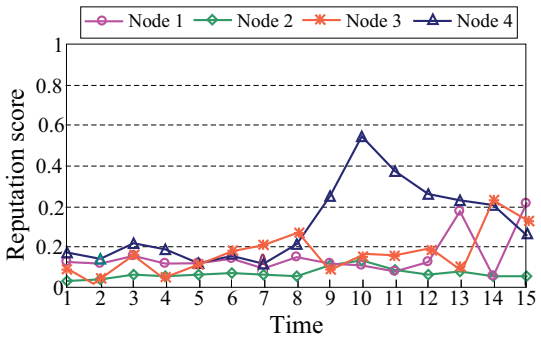


Fig. 3. Overall reputation scores based on correlation maps of nodes in a neighborhood.

We express the reputation score between $V_{x,k}$ and $V_{y,k}$ using the Pearson's product-moment coefficient [19]:

$$c_\tau(x,y) = \frac{w\sum_{i=1}^{w} V_{x,k}V_{y,k} - \sum_{i=1}^{w} V_{x,k}\sum_{i=1}^{w} V_{y,k}}{\sigma_{x,k}\sigma_{y,k}}$$

where $\sigma_{x,k}$ and $\sigma_{y,k}$ are their standard covariance. This reputation score is in the range of [-1,1]. If the value is close to either -1 or 1, it implies that there is a strong correlation between the variables. If the value is close to zero, the reputation decreases. As a result, we can build a matrix by putting the reputation score between metric x and metric y as the element $c_\tau(x,y)$ of the matrix:

$$Matrix = \begin{pmatrix} c_\tau(1,1) & c_\tau(1,2) & c_\tau(1,m) \\ c_\tau(2,1) & c_\tau(2,2) & c_\tau(2,m) \\ c_\tau(m,1) & c_\tau(m,2) & c_\tau(m,m) \end{pmatrix}.$$

This correlation matrix exhibits symmetry, i.e., $c_\tau(x,y) = c_\tau(y,x)$. Furthermore, since any such metric is absolutely correlated with itself, $c_\tau(x,x) = 1$. This matrix is a representation of the correlation map.

B. Temporal and Spatial Collusion Detection Schemes

CAD includes two sub-detection schemes, i.e. CAD detects collusion information in the *temporal* and *spatial* dimensions in a distributed manner. In the *temporal scheme*, temporal identification is made such that it refers to abrupt changes in the correlation map of a node. For example, at a given point in time, some nodes may pass some information to an adversary, and then it keeps working normally at some periods. At different time, the adversary may get information for different nodes of the system. In this case, the correlation may change slightly and the change can be overseen by some security protocols. Additionally, a node is shown to work properly and communicates with neighbors, and its authentication to other nodes looks fine. At a point in time, the node may pass information further.

Spatial detection determines pattern discrepancies using multiple nodes. If a collusion is detected by both temporal and spatial detection schemes, then the adversary has a high possibility of demonstrating a real problem. The *temporal detection* scheme investigates the progress of correlation maps over time. In every slot (e.g., $[1,\ldots,w],[w+1,\ldots,2w]$), a correlation map of node s_i is computed. If the system operates normally (i.e., no nodes are colluded or no events such as network congestion occur), the correlation map of the node should remain relatively stable. On the other hand, abrupt changes in consecutive maps imply collusion attack. We think that a map may be affected by false positive detection, which is due to factors like heavy network traffics. However, an abrupt change in consecutive maps can be not the sign of traffic.

1) Temporal Collusion Detection: For two consecutive correlation maps denoted by $CM_{i,\tau}$ and $CM_{i,\tau+1}$ of the same node (for successive time slots), a sudden change in the map may happen at one or more edges. For example, suppose that two metrics RadioTransmitOn and TransmitCounter are highly correlated in $CM_{i,\tau}$, but not in $CM_{i,\tau+1}$. Then, even if all of the other edges are the same, the change in correlation map is considered abrupt. As a result, we concentrate on the timely detection of such changes in individual edges between vertices.

In the simulation evaluation, we use a data set that maintains $m = 16$ metrics per sensor; for each correlation map sequence $CM(i)$, there are a total of $m*(m-1)/2 = 120$ different time series. When a time slot finishes, CAD computes if there is a sudden change for any edge. If there is a change, CAD indicates the change as a change point of node s_i. The detection of a sudden change for each edge in the map is modeled as a change point analysis problem.

We propose Algorithm 1 for change point detection by following a classical CUSUM algorithm [20] to detect change points for the time series of a sensor. $\{c_1(x,y), c_2(x,y), \ldots, c_i(x,y)\}$ denotes the reputation scores of edge (x,y) from the first time slot to the current one. In line 3 to line 4, the cumulative totals can be from CT_0 to CT_n. The insight behind the cumulative totals is that, if there are no sudden changes in the scores, then the cumulative totals just becomes near zero. Suppose that, at the initialization, all the scores are above the average: the term CT_{CP} is always larger than zero, causing cumulative sums CT_i to increase gradually. If $c_{k+1}(x,y)$ is an abrupt change, CT_{k+1} should be much smaller than CS_k, k is the last index before the abrupt change. As a result, CS_k will dominate both the preceding and the subsequent cumulative sums. The change score is denoted as CT_{cng}.

In line 7, bootstrap analysis is provided as a way to verify the significance of the change by coping with the behavior of CUSUM if there are no change points. In this step, the time series $c_i(u,v)$ are collected randomly. Based on the random ordered time series, a new sequence of cumulative totals, CT_{cng}, are calculated. After performing bootstraps B times, among which there are D times ($CT_{cng}^{conf} > CT_{th}$), the confidence level CT_{cng}^{conf} of the new change point, CT_{cng}, is calculated as D/B, as seen in line 9.

When the confidence level of CT_{cngf} is higher than a predefined threshold CT_{th} (e.g., 92%), we say that a sudden change happens in the current time series. As a result, $c_{k+1}(x,y)$ can be achieved as the change point. At the end of the algorithm,

Algorithm 1: Change Point Detection

1. for ith node s_i
2. Calculate $c_i(x, y)$
3. Calculate CT_{cng}
4. $CT_i = 0$
5. $CT_i = CT_{i-1} + CT_{CP}$
6. $CT_{CP} = c_i(x, y) - \sum_{i=1}^{n} c_i(x, y)/n$
7. $CT_{cng} = \max(CT_i) - \min(CT_i)$
8. Calculate $CT_{cng}^{conf} \leftarrow \frac{D}{B}$
9. If $CT_{cng}^{conf} > CT_{th}$
10. there is a sudden change //possibly due to the collusion
11. Get $CT_k = \max|CT_i|$
12. // k is the last index before a change
13. Return $c_{k+1}(x, y)$ as the change point

Algorithm 2: Spatial Collusion Detection

Input: Correlation values of all the neighboring nodes in time t
Output: A ranked list of nodes sorted by nodes' status and reputation score
Status: ($\lambda_c \leq 0.5$: normal), ($\lambda_c > 0.5$: colluded)

1. **for** each ith node $s_i \leq n$ where $n = |N|, N \in S$ **do:**
2. // $n \leftarrow$ the number of neighboring nodes
3. $(\lambda_c)_{s_i} \longleftarrow 0$ // each node s_i is normal
4. Get correlation maps $CM_{1,t}, CM_{2,t}, \ldots, CM_{n,t}$
5. Calculate cluster centers as $C_1, C_2, \ldots C_K$
6. Calculate node s_i's status by $(\lambda_c)_{s_i} \leftarrow \min_{s_i}(dist(CM_{i,t}, C_j))$
7. Find node s_j's reputation score $c_\tau(x, y)$
8. **if** $(\lambda_c)_{s_j} > 0.5$ and $(c_\tau(x, y))_{s_j} > 0.3$ **then**
9. s_i marks/reports that s_j is colluded
10. **if** s_i does not transmit the decision **then**
11. s_j marks/reports about s_i as a colluded
12. **return** a ranked list of nodes with the status

the nodes whose correlation maps change acutely are detected. As shown in Fig. 2, a reputation scores between nodes in a neighborhood can be seen.

2) Spatial Collusion Detection: This subsection briefly discusses the spatial collusion detection, taking ideas of correlation maps of all nodes in each time slot, and grouping similar ones together. A node from a subset of nodes whose correlation maps deviate from the common patterns are considered suspicious, as shown in Fig. 2. Node 4's reputation score differs significantly from its neighboring nodes. The neighboring nodes can easily report about node 4's reputation, that is, it might be colluded. Consider that $CM_{1,t}, CM_{1,t}, \ldots, CM_{n,t}$ are the correlation maps in a discrete period t of a subset of n nodes; we divide them into K clusters with cluster centroids $C_1, C_2, \ldots, C_K$. The confidence level of node i being suspicious is defined as: $\min_j(dist(CM_{i,t}, C_j))$, where $dist(CM_{i,t}, C_j)$ is the fractional distance function between two correlation maps. Based on the distance and node reputation, a node status can be given whether there is a collusion in the node or not. For reducing the false-positive status, the reputation score is considered with the distance.

K-Means clustering algorithm is popular for grouping similar kind of values that are used to find the distance based on the correlation map. A node with the farthest distance from the center can be calculated. We can compute this if a node is colluded in a distributed manner, where each node makes a status decision based on the distance and reputation score. If the local status denoted by λ_c of node s_j is greater than 0.5, node s_i can report that node s_j's status is colluded. Similarly, if a false-positive status is made by node s_i, it can be detected by other nodes' status decisions. The procedure of spatial collusion detection, based on the above discussion, is given in Algorithm 2.

The collusion detection procedures in both Algorithm 1 and Algorithm 2 are executed in a distributed manner. If we consider a centralized detection scheme, the base station (BS) would handle the collusion detection as well as the application tasks. In each discrete period, the BS needs to make a decision about the colluded nodes' status that is solely based on the reliable packet transmission of all packets by all nodes in the system. This detection is not suitable for a resource-constrained system, which usually includes energy and bandwidth constraints. For example, if each node needs to send all its packets to the BS (where each sequence of transmission can have many packets), the centralized BS relies on the on all the packet receptions; it may not be able to reliably offer a collusion detection in a given period.

V. Performance Evaluation

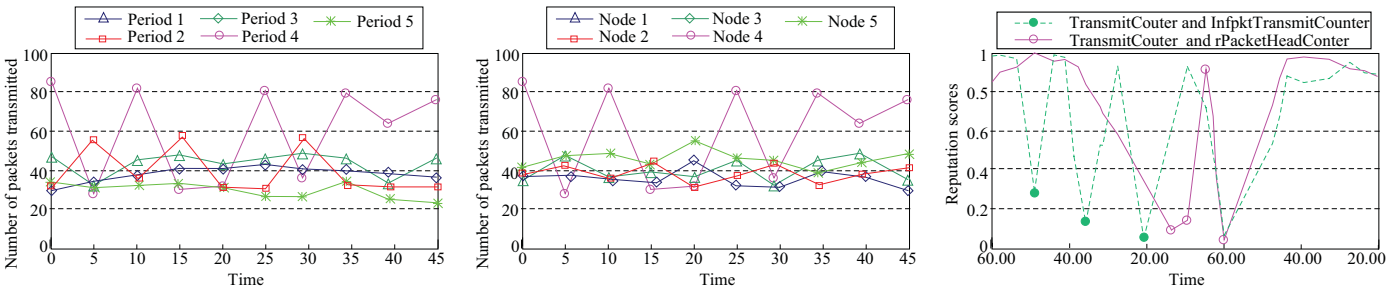
A. Simulation Methodology

For the evaluation, we consider a sensor network. We conduct simulations using the OMNeT++ platform to evaluate a CAD scheme that includes the two collusion attack detection methods. We implement it on the OMNet++ simulation environment using the Castalia simulator (http://castalia.npc.nicta.com.au/index.php).

We conduct simulations on a PC with 3.30 GHz Intel Core i5 with 4 GB RAM, running a 64-bit slots 8. We uniformly deploy 100 nodes in a $200m \times 200m$ target field. For the radio model, we consider IEEE 802.15.4 protocol that defines MAC and physical layers for low-rate wireless sensor networks and the upper layers to form a complete network stack built are specified by ZigBee. The objective of IEEE 802.15.4 is to enable low-cost communication between devices. In particular, the physical layer allows data rates up to 250 kB/s. The MAC layer supports collision avoidance. Beaconing is used for synchronization between nodes. We set the maximum transmission range of all nodes to 30m. The nodes communicate with each other by broadcasting messages across channels. These are assumed to be symmetric. We set sensor node communication range twice the sensing range, which is to ensure that two nodes with an overlapping sensing area are capable of communicating directly with each other.

We use a real-world dataset for evaluating CAD in the sensor network, Intel dataset [21]. Since the dataset is given for 54 sensors, the set is repeated for the additional nodes in our simulations.

For collusion attack information, we randomly inject *changes* to different types of metrics of 10% of the nodes of the WSN. From the 10% of the nodes, we change the radio capability metric [TransmitCouter (number of packets



(a) Number of packets transmitted by node 4

(b) Number of packets transmitted by nodes 1-5 in a same period

(c) Reputation scores between metrics of a node

Fig. 4. Results achieved during the normal operation and operation under collusion attack: (a) number of packets transmitted by node 4; (b) number of packets transmitted by nodes 1-5 in a same period; (c) correlation scores between TransmitCouter and TransmitCounter, and between TransmitCouter and rPacketHeadConter.

transmitted)] of 4% of the nodes, and provide an infrequent packet transmission rate [InfpktTransmitCounter] to 3% of the nodes, and provide the ability to replace data packets with some other packets [rPacketHeadConter] to 3% of the nodes. As a result, at some time points, their packet transmission rate should be high, while it is normal or low at other times. We compare these colluded nodes with list of the nodes that have not been made colluded.

For the average collusion attack detection, each simulation was repeated 100 times, and then results were averaged. We calculate the number of packets transmitted, packet delivery ratios (PDR), and temporal detection. PDR is defined as the ratio between the amount of packets received by the BS and that transmitted by the source node. The detection rate of our scheme is compared with a recent scheme, called interative filtering (IF) based collusion detection [3]. However, IF only addresses collusion attacks at the data aggregation level, rather than at the communication component level.

B. Simulation Results

At first, we demonstrate a set of interesting results for collusion detection. We calculate the number of transmitted packets by each node. We can see in Fig. 4(a), the number of packets transmitted by the first five nodes in every 45 minutes. In the case of node 4, we find that, the values of metrics change dramatically at time periods 10, 25, 30, 40, 45. This is possibly due to the the collusion attack. In a real case, such a node might have a secret agreement with the adversary or it might be compromised. Based on the analysis, we find that the metric appears to be stable and node 4 works well, similar to other nodes. Fig. 4(b) shows the data curves correspond to the packet transmissions of first five nodes during period 4. Even though some metrics exhibit different values and variations at different times, all five nodes perform well; however, the correlation among them is relatively weak. This result implies that, even if considering multiple nodes, an individual metric is insufficient for uncovering attacks. Using several metrics can be helpful for detecting collusion by the reputation score and distance.

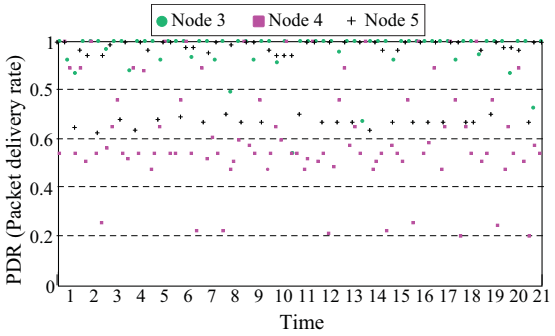


Fig. 5. The PDR of three nodes achieved during the normal operation and operation under collusion attack.

We depict the reputation scores of the three metrics (TransmitCouter, InfpktTransmitCounter, rPacketHeadConter) in Fig. 4(c). Since these metrics are always clearly correlated, the reputation scores are larger than 0. In the figure, there are several change points detected by the temporal detection algorithm. Each of them is marked with a circle, and the change point is detected at 09:00, which is the most remarkable one, i.e., it has the highest confidence level. The reason is that this node received packets from its neighboring nodes, but did not forward all of them to the BS or it has replaced some packets with other information. In this case, the colluded node receives the message and pause for a moment. It can also take time for replacing packets.

The PDRs estimated for nodes 3, 4, and 5 are directed in Fig. 5. Each mark/point in Fig. 5 corresponds to the time when the temporal subscheme reports change points, or one of the system metrics of a node that is detected in the spatial detection scheme. We can analyze that node 3 is the boundary node transmitting data via node 4, which receives all the packets from the nodes. In a real case, it can replace some packets with its own secret packets, or drop some packets of node 3. We can see that the PDR of node 4 is lower than that of other nodes.

A set of interesting results is depicted in Fig. 6 that shows the collusion detection rate of CAD, and IF can also be seen in Fig. 6. It is difficult to compare this with the IF scheme,

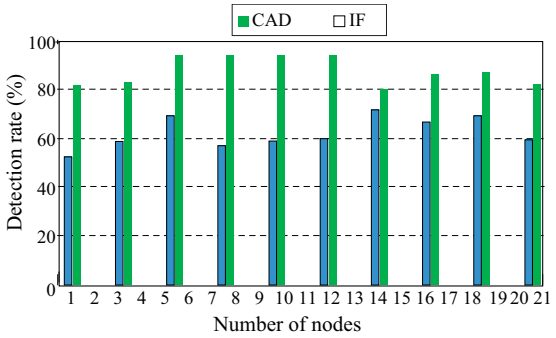


Fig. 6. The detection rate achieved during the normal operation and operation under collusion attack.

as the collusion detection in IF is performed on signal content aggregation, not on the packet aggregation. In signal content aggregation, readings are summarized or average, and a weight is calculated. This is quite different from packet aggregation, as each individual packet is needed to be processed separately in packet aggregation. We just have a look the overall detection ability. We can see that CAD can achieve a detection rate up to 92%, while IF is able to detect 76%. The reason is that IF did not particularly consider any collusion attacks at the communication level, although IF is particularly useful for collusion attack detection in data aggregation.

In Fig. 6, it is seen that CAD does perform better than 62%. The reasons might be due to the proper collusion detection model and other related attack models. In practice, faults in the system can be correlated with communication components of the system. Fault detection should be analyzed for a better collusion detection.

VI. Conclusion

Existing security methods or protocols suggested in the literature are able to detect a lot of security attacks, but they did not systematically consider collusion attack detection. As mentioned, some sophisticated collusion attacks can be made in the presence of these protocols. This paper presents the CAD (collusion attack detection) scheme, a novel approach that relies on minimum domain knowledge. CAD discovers the correlation between system metrics related to communication hardware or software using two stage cross validation schemes to detect collusion. Our evaluation, based on the Intel dataset under the collusion information injection, demonstrates that the two stages indeed capture collusion attack information. One interesting matter is that, if there is a false positive detection of collusion attack in CAD, we realize that there might be a system performance degradation, or there are potential faults/failures. However, the values of the metrics can be affected by system faults and/or by a poor network environment. We believe that faults in a system can be correlated with communication components of the system. Thus, fault detection should be analyzed for a better collusion detection. These are the focus of our future work.

Acknowledgment

This work is supported by NSF grants CNS 149860, CNS 1461932, CNS 1460971, CNS 1439672, CNS 1301774, ECCS 1231461, ECCS 1128209, and CNS 1138963.

References

[1] M. Z. A. Bhuiyan, G. Wang, J. Cao, and J. Wu, "Sensor placement with multiple objectives for structural health monitoring," *ACM Transactions on Sensor Networks*, vol. 10, no. 4, pp. 1–45, 2014.

[2] W. Jiang, G. Wang, M. Z. A. Bhuiyan, and J. Wu, "Understanding graph-based trust evaluation in online social networks: Methodologies and challenges," *ACM Computing Surveys*, 2016 (To Appear).

[3] M. Rezvani, A. Ignjatovic, E. Bertino, and S. Jha, "Secure data aggregation technique for wireless sensor networks in the presence of collusion attacks," *IEEE Transactions on Dependable and Secure Computing*, vol. 12, no. 1, pp. 98–110, 2015.

[4] S. Sultana, G. Ghinita, , E. Bertino, and M. Shehab, "A lightweight secure scheme for detecting provenance forgery and packet drop attacks in wireless sensor networks," *IEEE Transactions on Dependable and Secure Computing*, vol. 12, no. 3, pp. 256–269, 2015.

[5] S. Ji, T. Chen, and S. Zhong, "Wormhole attack detection algorithms in wireless network coding systems," *IEEE Transactions on Mobile Computing*, vol. 14, no. 3, pp. 660–674, 2015.

[6] Q. Yang, X. Zhu, H. Fu, and X. Che, "Survey of security technologies on wireless sensor networks," *Journal of Sensors*, vol. 2015, pp. 1–9, 2015.

[7] X. Wang, L. Qian, and H. Jiang, "Tolerant majority colluding attacks for secure localization in wireless sensor networks," in *Proc. of the 5th International Conference on Wireless Communications, Networking and Mobile Computing*, 2009, pp. 1–5.

[8] S. Wei, Y. Meng, and C. Jean-Pierre, "Resilient secure localization and detection of colluding attackers in wsns," in *Proc. of Ad-hoc, Mobile, and Wireless Networks*, 2012, pp. 181–192.

[9] U. S. R. K. Dhamodharan and R. Vayanaperumal, "Detecting and preventing sybil attacks in wireless sensor networks using message authentication and passing method," *The Scientific World Journal*, vol. 60, pp. 1–7, 2015.

[10] D. Jiao, M. Li, Y. Yu, and J. Ou, "Self-healing key-distribution scheme with collusion attack resistance based on one-way key chains and secret sharing in wireless sensor networks," *International Journal of Distributed Sensor Networks*, vol. 2015, pp. 1–7, 2012.

[11] S. Roy, M. Conti, S. Setia, and S. Jajodia, "Secure data aggregation in wireless sensor networks," *IEEE Transactions on Information Forensics Security*, vol. 7, no. 3, p. 10401052, 2012.

[12] S. G. Vadlamudi and P. Chakrabarti, "Robustness analysis of embedded control systems with respect to signal perturbations: Finding minimal counterexamples using fault injection," *IEEE Transactions on Dependable and Secure Computing*, vol. 11, no. 1, pp. 45–58, 2015.

[13] Y. Yu, K. Li, W. Zhou, and P. Li, "Trust mechanisms in wireless sensor networks: Attack analysis and countermeasures," *J. Netw. Comput. Appl.*, vol. 35, no. 3, p. 867880, 2012.

[14] A. Ramos and R. H. Filho, "Sensor data security level estimation scheme for wireless sensor networks," *Sensors*, vol. 15, pp. 2104–2136, 2015.

[15] K. Bao and Z. Zhang, "Collusion attack on a self-healing key distribution with revocation in wireless sensor networks," in *Proc. of the 11th International Workshop on Information Security Applications*, 2011, p. 221233.

[16] C. de Kerchove and P. V. Dooren, "Iterative filtering in reputation systems," *SIAM J. Matrix Anal. Appl.*, vol. 31, no. 4, pp. 1812–1834, 2010.

[17] R.-H. Li, J. X. Yu, X. Huang, and H. Cheng, "Robust reputation based ranking on bipartite rating networks," in *Proc. of SIAM Int. Conf. Data Mining*, 2012, p. 612623.

[18] H. Liao, G. Cimini, and M. Medo, "Measuring quality, reputation and trust in online communities," in *Proc. 20th Int. Conf. Found. Intell. Syst.*, 2012, p. 405414.

[19] Pearson product-moment correlation coefficient. [Online]. Available: https:/en.wikipedia.orgwiki/Pearson_product-moment_correlation_coefficient

[20] M. basseville and i. nikiforov. [Online]. Available: Detection of abrupt changes:theory and application

[21] The intel lab data data set. [Online]. Available: http://berkeley.intel-research.net/labdata/