Detecting FDI Attack on Dense IoT Network with Distributed Filtering Collaboration and Consensus

Carlos Pedroso Junior*, **Aldri Santos***, **Michele Nogueira***
*Wireless and Advanced Networks Laboratory (NR2) - UFPR, Brazil
Emails: {capjunior, aldri, michele}@inf.ufpr.br

***Abstract*—The rise of IoT has made possible the development of personalized services, like industrial services that often deal with massive amounts of data. However, as IoT grows, its threats are even greater. The false data injection (FDI) attack stands out as being one of the most harmful to data networks like IoT. The majority of current systems to handle this attack do not take into account the data validation, especially on the data clustering service. This work introduces CONFINIT, an intrusion detection system against FDI attacks on the data dissemination service into dense IoT. It combines watchdog surveillance and collaborative consensus among IoT devices for getting the swift detection of attackers. CONFINIT was evaluated in the NS-3 simulator into a dense industrial IoT and it has gotten detection rates of 99%, 3.2% of false negative and 3.6% of false positive rates, adding up to 35% in clustering without FDI attackers.**

I. Introduction

Internet of Things (IoT) enables the connection of a variety of physical devices through technologies like RFID, GPS and NFC, among others. IoT objects usually hold features like identity, physical attributes, mobility level, and many of them through smart interface establish communication with each other [1]. IoT is essential to gather, disseminate, and arrange the data volume required by various applications to make its decisions [2]. For instance, the Industrial Internet of Things (IIoT) has recently brought attention for enabling diverse industrial devices to play in an organized and synchronized way. Though, IoT still exhibits various vulnerabilities and challenges to achieve better robustness [3] due to its unique aspects. The massive amount of data due to the interaction among multiple devices exposes dense IoTs to data security threats. Those environments typically rely on mobile and fixed devices, and the infrastructure can change through ties among devices [1]. Thus, IoT has been target of attacks that violate many of the security attributes, like integrity, authenticity, and availability of services; including the data dissemination,which hinders the performance of various applications [4].

Among the internal threats to the IoT data dissemination service, False Data Injection (FDI) Attack stand out as one of the most harmful due to the inconsistent data yielded by them and the unpredictability of its actions [5]. Due to that unpredictability, the attack detection is difficult, as captured devices are usually authenticated on networks and perform their standard data collection and dissemination functions [6]. Furthermore, attacks can carry out continuously and at distinct times, disturbing the network. The FDI attack take place when a device take control others, and then alter, fabricate, or manipulate its data, or when the device itself behaves in misconducting. Thus, it is essential quickly to identify and isolate malicious devices on dense IoT networks to avoid long malfunction times and data inconsistencies.

Although there are approaches to handle FDI attacks in WSNs [7], Smart Grids [8] or even IoT [5], they have usually failed or are not suited to dense IoTs because they do not analyze the data and few employ collaborative detection. Additionally, systems, like En-Route Filtering Scheme, Collaborative Detection and Intrusion Detection Systems (IDS), have been commonly intended to monitor anomalies based on network traffic and not to the gathered data [9]. IDS are a robust approach against attacks in diverse IoT contexts [5]. But, when IDS are not correctly employed in a IoT, they can bring up new vulnerabilities. Therefore, it is mandatory to detect and isolate IoT threats in a distributed manner to assure greater robustness for the service of data dissemination. An effective manner to face FDI attacks consists of usage jointly of collaborative consensus [10] for decision making among network devices and the watchdog monitoring [5], since their techniques would allow us swiftly the detection and identification of nodes that exhibit FDI attack behavior over the time.

This work presents CONFINIT (***CON****sensus Based Data* ***FI****lteri****N****g for* ***IoT***) a system to mitigate and isolate false data injection attacks on data dissemination services in dense IoT networks. CONFINIT aims to detect and isolate data misbehaving devices on the clustering service. For that, CONFINIT applies clustering by data similarity to handle the density of devices on the network. It also combines a watchdog strategy between devices for self-monitoring as well as a collaborative consensus for decision making on FDI misbehaving devices. Simulations showed that CONFINIT achieved attack detection rates of 99% in dense IIoT scenarios, only up to 3.2% of false negatives and 3.6% of false positives, increasing up to 35% the number of clustering without attackers.

This paper is organized as follows: Section II presents the related work. Section III defines the model and assumptions taken by CONFINIT. Section IV describes the CONFINIT components and their operation. Section V shows an evaluation of CONFINIT and the results obtained. Section VI presents conclusions and future work.

II. Related Work

In [5], an IDS based on anomaly detection applies watchdog to mitigate False Data Injection attacks. They attempt to

978-1-7281-8903-1/20/$31.00 ©2020 IEEE

predict natural events by using environmental IoT surveillance data through Hierarchical Bayesian Time-Space (HBT) monitoring and a statistical decision strategy in a sequential probability test supports to identify malicious activities. However, the HBT model is energetically costly and the probability test, although effective, overloads the network and negligence the data validation. In [11], an IDS to mitigate sinkhole and selective forwarding attacks in dense IoT arranges the network in clusters and classifies the nodes in categories to get better communication. Besides that, a watchdog strategy in levels monitors the relationship between received and transmitted data to determine nodes with malicious behavior. However, computing trust between nodes only by a difference of data received and transmitted makes it hard to identify nodes that manipulate the collected data. In [7], a cooperative authentication scheme aimed at filtering false data in WSNs, so that all nodes require a fixed number of neighbors to manage the authentication in a distributed across data routing to the base station. The scheme also adopts the compression bit technique aiming to avoid overloading the communication channel, making it appropriated to filter injected data since the authentication happens point-to-point. However, this scheme is fragile to face the data manipulation and does not identify compromised devices. In [12], a scheme of dynamic en-route filtering to handle false data and DoS attacks on WSNs employs a group of authentication keys by node that are used to endorse reports. Authentication is guaranteed by a group of nodes chosen before the network starts. Each node offers its keys to the forwarding nodes which in turn should propagate the received keys, allowing the forwarders to verify all the reports. However, the scheme also does not take into account data manipulation and causes an additional cost to the network due to the constant exchange of keys between devices.

In [13], a distributed system to detect anomalies caused by DDoS attacks applies an averaged consensus protocol among the participants. It analyzes each data collection point using a Bayes classifier. The analysis occurs redundantly, parallel to the level of each data collection point, which avoids the single point of failure. The distributed consensus favors collaborative decision making. Though, the communication cost among the participants overloads the network and diminish the system's effectiveness. In [14], a distributed consensus system using distributed weighted average makes use of an algorithm to allow adaptation to local rules stipulated by the network, where a learning technique estimates operating parameters or weight of each node to automate local merge or update rules to mitigate attacks. The algorithm acts in a distributed manner, but it does not take the interaction among network devices as an impact factor, which allows malicious actions.

III. Environment Model

This section describes our assumption on the network infrastructure and the FDI attack model. The network model consists of a three-level structure, whose the first one means the IIoT objects, the second one carries out the communication among objects, and the third one coordinates the virtual clustering of the objects. Fig. 1 illustrates all levels and their relationship.

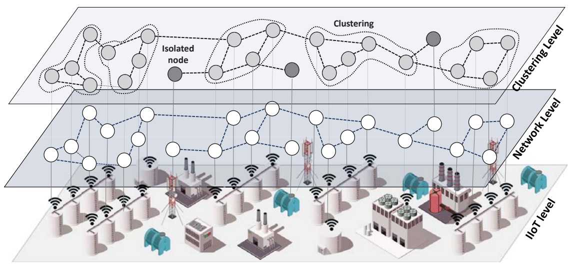

Fig. 1. Network model

Network model: A massive IoT network composed by a set of objects (nodes), as an industrial environment, denoted by $N = \{n_1, n_2, n_3, ..., n_n\}$, where $n_i \in N$. Each node n_i has a unique physical address Id that identifies it on the network. Each node clustering is a subset $A_\alpha \subseteq N$, where each node $n_i \in N$. The node communication carries out in the wireless medium with an asynchronous channel and subject to the packet loss. In addition, all nodes work with the same transmission range to establish clustering. We consider that nodes do not suffer from energy restriction in the context of an IIoT network. ***FDI Attack model:*** The FDI attack takes place in two manners: *i)* through the capture of the object and manipulation of the data by either altering or falsifying it; *ii)* the object itself is the attacker, and it changes, fabricates, or manipulates its own data. The attacker exploits vulnerabilities arising from other attacks, in addition to the attacker with full knowledge about the network and works coordinated [5], [15].

IV. CONFINIT Architecture

The CONFINIT architecture consists of the **Clustering Management** and **Fault Management** modules that work jointly to ensure the secure dissemination of data in the network, as shown in Fig 2. The former arranges the devices in clusters and the latter deals with the monitoring of devices, and detection and isolation of FDI misbehaving devices.

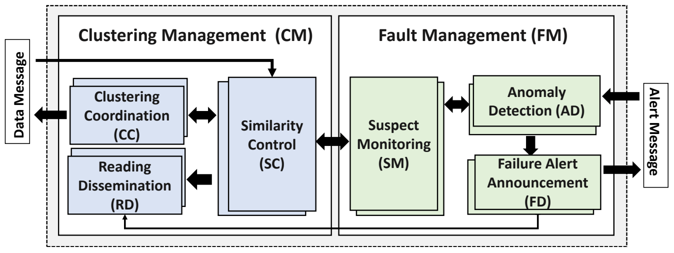

Fig. 2. The CONFINIT architecture

The Clustering Management (CM) Module ensures the formation and maintenance of clusters of devices in the network. **CM** builds the clusters based on the similarity threshold of readings from nearby devices (nodes) and establishes the integration of nodes in a cluster. Thus, when receiving a data message **CM** analyzes the identification, the number of neighbors and the readings of those neighbors. **CM** consists of the *Similarity Control (SC)*, *Clustering Coordination (CC)* and *Reading Dissemination (RD)* components. SC acts by receiving and interpreting messages exchanged among the nodes. CC takes care to form and maintain clusters by using the similarity of readings among nodes, being also responsible for electing leader. RD acts for disclosing the device's reading, number of readings and neighbors, so that all nodes that receive the message will know when they are part of a cluster.

The Fault Management (FM) Module ensures the security of data dissemination among nodes so that only authentic readings are disseminated over the network. For that, it takes the *Suspect Monitoring (SM)*, *Anomaly Detection (AD)*, and *Failure Alert Announcement (FA)* components. SM monitors other nodes that do not respect the similarity threshold. AD deals with the collaborative consensus and the standard deviation technique to detect FDI misbehaving nodes. The collaborative consensus is the agreement and uniformity of opinions that nodes establish through the exchange of information among their cluster neighbors. The usage of standard deviation aims to determine how many discrepant the readings are in a given moment. Lastly, FA acts to isolate FDI attacking nodes and alert other members about the threat. Thus, when detecting a FDI misbehaving node, the detector node propagates an alert message to the cluster leader that spreads it across the network. The alert message carries the attacker Id and its individual reading as well as the detector node Id. The sending time of the alert message takes into account the exact timing of detection and propagation.

A. Cluster configuration

CM arranges the network into clusters, based on leaders, to create a network topology. Initially, nodes begin their operations in an isolation way, transmitting and collecting exchanged data messages from neighboring nodes for the cluster formation. Algorithm 1 presents how the clustering works. Similarity clustering prevents data redundancy and ensures that a single read represents data from all cluster nodes. Periodically, after a data gathering, each node broadcasts one data message, with the following information $< Id, L_{ind}, L_{agr}, N_{viz} >$, for mentioning its identifier Id, its current reading L_{ind}, the aggregate reading average of nodes with similar readings on its neighborhood L_{agr} and the amount of neighbors N_{viz} (l.1-l.5). The $WaitInterval$ function establishes a random infimum delay to avoid simultaneous transmissions, and thus collision of messages among the nodes (l.3). Upon receiving a data message (l.6), the node will know the source org, the individual reading iR from the issuing node, the aggregate reading aR from its neighborhood, and the amount of neighboring nodes by nR. Initially, the node updates the received information (l.7), so that average aggregate readings enable to find out whether the current node reading satisfies the similarity threshold (l.8). Thus, the similarity check up is made in (l.9). The N_{viz} structure neighborhood is updated by setting adding org when the similarity threshold is satisfied (l.10) or removing when the node not satisfied (l.12). This process occurs dynamically on each network node, ensuring that everyone can keep their neighborhood structure up to date.

Algorithm 1: Clustering establishment

```
1  procedure SendDataMessage()
2      $Send(Id, L_{ind}, L_{agr}, |N_{viz}|)$
3      $WaitIntervalnextmsg$
4      $RControlTimerExpire()$
5  end procedure
6  procedure ReceiveDataMessage($Org, iR, aR, nR$)
7      $N_{viz}[Org] \leftarrow \{iR, aR, nR\}$
8      $localRead \leftarrow L_{agr}()$
9      if ($|iR - localRead| < Threshold$) and
       ($|L_{ind}() - aR| < Threshold$) then
10         $N_{viz} \leftarrow SN_{viz} \bigcup \{org\}$
11     else if
12         $N_{viz} \leftarrow SN_{viz} \bigcap \{org\}$
13     end if
14 end procedure
```

$$\left| Y - \frac{X + \sum_{v \in SN_{viz}} (N_{viz}[v].aR * N_{viz}R[v].nR)}{1 + \sum_{v \in SN_{viz}} (N_{viz}[v].nR)} \right| < CThresh \quad (1)$$

The computation of the similarity relationship between two nodes makes use of their reading values and the threshold value between them. Equation 1 verifies the similarity between two readings, following the DDFC model developed by [16], whose model enables adaptability and accuracy to dense environments like IIoT. The equation considers the node readings, the amount of neighbors, and the aggregate readings of those neighbors. X means the current reading of the node and Y the reading with which it is being compared. The comparison outcome determines which readings respect the similarity threshold ($CThresh$), and thus the node can join the same cluster. In the Dynamic Data-aware Firefly-based Clustering (DDFC) model, the value of the similarity threshold is strictly dependent on the application and the data use [16].

B. Fault detection

The nodes whose forwarded values do not respect the threshold of similarity of readings over the clustering formation stage, firstly, are classified as suspects and compose the individual suspect list of the node whom they form a cluster. The usage of the suspect list enables a better evaluation of fault detection, since devices may at times present faults behavior, which does not characterize attacker behavior. Algorithm 2 details the operation of the fault detection module within the network against FDI threats. the detection action takes effect effectively after the first message exchange, as nodes need other messages to compare. Thus, at first, **FM** analyzes if a given node belongs to the suspect list (l.1-l.7). In the case of not being a suspect, but its readings are suspicious the module inserts the node on the suspect list (l.8-l.16). Otherwise, if it belongs to the list, buts its readings meet the defined threshold, the module removes it from the list (l.17-l.20).

Equation 2 determines the consensus computation to establish the deviation of the measured values. For this purpose, the readings data collected from the consensus participants

Algorithm 2: Attacker detection

```
procedure CHECKSUSPICIOUS ($Id, ConsensusParticpant$)
    if($ID \in SuspectList$ & $ConsensusParticpant == False$)
        $return 1$
    else if($ID \in SuspectList$ & $ConsensusParticpant == True$)
        $return 2$
    end if
end procedure
procedure CHECKSUSPICIOUS($Id, Read$)
    $Valid \longleftarrow checkSuspicious(Id, ConsensusParticpant)$
    if($Read \leq Thresholdconsensus$)
        $Switch \longleftarrow Valid$
        Case 1
        $Atklist \leftarrow Ataklist \bigcup \{Id, Read\}$
        Case 2
        $Suspectlist \leftarrow Suspectlist \bigcap \{Id, Read\}$
        $Switch \longleftarrow Valid$
    else
        $Suspectlist \leftarrow Suspectlist \bigcup \{Id, Read\}$
    end if
end procedure
```

are used to compare them. So we use a data set $D = (d_i, d_{i+1}, ..., d_n)$, which represents the data samples to check. The consensus calculation is indicated by $\sum_{i=1}^{n}$, which adds up all values of the D set, from the first position *(i=1)* to the position $d \in D$. The value of d_i represents at the i position in the data set D. M_A represents the arithmetic mean of the data. N represents the amount of data to be evaluated in forming consensus. The *Thresholdconsensus* means the default threshold and it can change according to the type of the data evaluated and the type of application.

$$DP = \sqrt{\frac{\sum_{i=1}^{n}(d_i - M_A)^2}{N}} \leq Thresholdconsensus \quad (2)$$

C. Operation

The clustering formation stage occurs dynamically for each node of the IoT network. Interactions among nodes take place under space and time quantities, in this way messages are sent and received by nodes into the transmitters transmission range. Each node broadcasts one data message with its reading, its neighbors readings, and the number of neighbors. The receiver node, when receiving the message, verifies and checking the information and carrying out the similarity calculation. Once the similarity threshold ($CThresh$) is met, the node joins the cluster, otherwise it becomes part of the suspect list at first. Fig. 3 shows how CONFINIT acts over the cluster formation and leader election. Solid edges indicate nodes with transmission range overlap and that can exchange messages between them. The boxes next to each node mean, from top to bottom, the individual reading of the node, neighbors nodes aggregate reading, and the value of aggregate readings. Thus, we considered the ($CThresh$) equal to three for cluster formation. Each instant *T* corresponds to a message exchange round among nodes in order to form clusters and elect leaders.

In this way, assuming CONFINIT makes use of a ($CThresh$) = *3*, in T_1 all nodes start exchanging data messages to update the neighbors list and aggregate readings taking into account Equation 1. They begin the process of forming clusters and electing leaders based on the readings obtained at the earlier instant Ra_{Tn-1} according to Equation 1. In T_2, we have the following similarity between the nodes, $Ra_{T2}(n_a) = \frac{15+1*16}{1+1}$, $Ra_{T2}(n_b) = \frac{16+1*15+1*18}{1+1+1}$, $Ra_{T2}(n_c) = \frac{18+1*16+1*16+1*16+1*17}{1+1+1+1}$, $Ra_{T2}(n_d) = \frac{17+1*16+1*18}{1+1+1}$, $Ra_{T2}(n_e) = \frac{16+1*17+1*18}{1+1+1}$. After that, node n_b has the largest number of neighbors, being elected as the cluster leader. As the clustering process works dynamically on each node, in T_3, it is done again due to existence of a new exchanged data message among the nodes, thus the new similarity calculation is applied to maintain the cluster formation, $Ra_{T3}(n_a) = \frac{20+1*22+1*21+1*23}{1+1+1+1}$, $Ra_{T3}(n_b) = \frac{22+1*20+1*23}{1+1+1}$, $Ra_{T3}(n_c) = \frac{23+1*22+1*20}{1+1+1}$, $Ra_{T3}(n_d) = \frac{21+1*24+1*20}{1+1+1}$, $Ra_{T3}(n_e) = \frac{24+1*21}{1+1}$. The clustering configuration enables nodes to know which neighbors belong to the same cluster and it ensures better network scalability. It also helps to classify nodes with divergent readings, thus facilitating the identification of attackers by the fault control module.

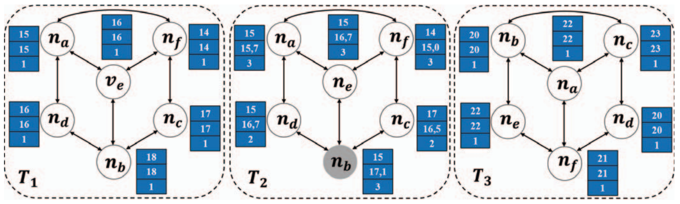

Fig. 3. Clustering formation over the time

The fault detection works over the formation of clusters. Thus, nodes whose readings do not respect the similarity threshold will be firstly added into a suspect list. The classification of nodes that were not part of the cluster begins in the control message, which without properly filled fields are discarded. Over the cluster formation, nodes physically close to its neighbors at a given time and with distinct readings outside of the similarity threshold can be considered a suspicious node at the first time. Thus it becomes part of the suspect list, which contains nodes that have behaved anomalously but are not necessarily attackers. When a given node, which belongs to the suspects list, tries to join the cluster, and it again fails due to its distinct readings, then the system classifies it as attacker. Hence, the system adds its Id into a list containing all Ids of threats. The cluster leader alerts the others by sending an alert message. Thus, when the attacker tries to join the cluster again, it is blocked.

Fig. 4 illustrates an example of attack detection, where CONFINIT makes use of consensus threshold ($Thresholdconsensus$) = *5*. Each instant T means a clustering process and whose nodes that meet the ($CThresh$) value become clustered. Thus, at the instant T_1 nodes $(n_a, n_b, n_d, n_d, n_f)$ have reading values varying from 14 to 18, respecting the ($CThresh$) among them. But, node n_c has a value of 45 for its reading, which deviates greatly from its spatial neighbors, not respecting the

($CThresh$). Hence, node n_c cannot integrate the cluster at that time. At T_2, node n_c again tries to join the cluster, but its Id is already on the suspect list. Thus, Equation 2 applies to check the ($Thresholdconsensus$) based on consensus participates readings and compared with node n_c. In this way, they determined n_c is an attacker and cannot integrate any cluster. At T_3, node n_c is removed from the network, and an alarm message sent to the leader with the Id of the attacker. After receiving the message, the leader disseminates the message to the other leaders on the network.

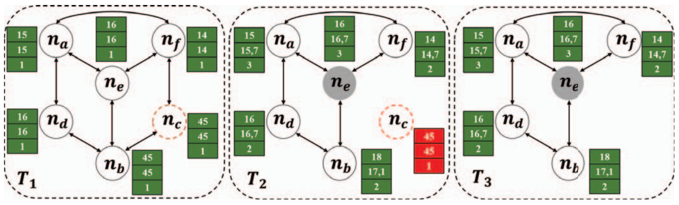

Fig. 4. Detection and isolation of an FDI Attack

V. Analysis

This section shows a performance evaluation of CONFINIT. We implement CONFINIT in NS3-simulator, a data set of gas pressure sensors collected from a real petrochemical industry environment available by the UCI Machine Learning Repository lab [17] in order to get close to a real industrial environment. We also implemented the behavior of FDI attack based in [6], whose attackers aim to alter data readings. We assume that the attacker works synchronously and knows the type of the network traffic data, facilitating the interaction for modifying the collected values, meaning the closest behavior to a FDI attack. The industrial scenarios consisted of 50, 75, and 100 homogeneous nodes randomly distributed in a rectangular area of *200 x 200* meters operating by *1200s* with a transmission radius of *100m*. This transmission radius value is proportional to the coverage area in order to not interfere with the clustering establishment and the collaborative consensus, and hence not compromising the achieved results. We defined the similarity threshold ($CThresh$) with the value 3 and the consensus threshold ($Thresholdconsensus$) with the value 5. We took these thresholds based on the type of data obtained by the *dataset*. These thresholds may vary according to the type of data and the application model. Moreover we do not consider energy issues once the nodes operate embedded in industrial equipment. The amount of FDI attacks ranges among 2%, 5%, and 10%, all nodes run IPv6 establishing an *ad-hoc* standard IEEE 802.15.4. We also implement the DDFC [16] clustering protocol as part of the CONFINIT system. Furthermore, the message controller, the adaptive interval, and the IPv6 integration from DDFC have been changed to better adapt it to the dense IIoT network context. The results obtained in all simulations correspond to the average values of 35 simulations with confidence interval of 95%. Table I summarizes the metrics assessed according to [5].

TABLE I
EVALUATION METRICS

METRIC	**EQUATION**
Detection rate (DR)	$(R_{det}) = \frac{\sum det_{ni}}{A_{ins}}$
Accuracy (AC)	$(A_a) = \frac{\sum det_{ni} + \sum det_{nl}}{R_{det} + det_{ni} + R_{fn} + R_{fp}}$
False positive rate (FPR)	$(R_{fp}) = \frac{\sum det_{ni}}{T_{int}}$
False negative rate (FNR)	$(R_{fn}) = \lvert X \rvert - R_{det}$

A. Clustering

The CONFINIT performance for supporting the availability of the established clusters based on the number of nodes and the presence of attackers into the network is shown in Fig. 5. Note that the number of available clusters varies depending on the relationship of nodes and the amount of attackers present in the network. Regarding the FDI attack, it directly impacts on the number of clusters formed over the time. Moreover, as expected, the scenario with 10% of attacks had a stronger effect on the cluster establishment than 5% and 2%. In relation to DDFC, in some cases, CONFINIT increases the number of clusters by 35%. This behavior has an effect on the amount of available data, and interferes positively with the decision making. Such performance shows its efficiency in keeping the clusters safe and available. The formation of clusters considers the fact that the scenario is static, meaning, it does not suffer changes in the node position; and only the readings vary in their values. Moreover, clusters hold a non-deterministic number of members, i.e., there is no fixed number of members.

B. Attack detection

The detection and mitigation effectiveness of CONFINIT are shown in Fig. 6 (**a**). It obtained an average DR of 97%, sometimes even reaching 100% depending on the variation of the number of nodes into the network. The CONFINIT accuracy is shown in Fig. 6 (**b**), whose values range from 0 to 1, being closer to 1 more accurate. The AC values achieved by CONFINIT between 0.81 and 0.98 are due to the watchdog surveillance among participants, which evaluates the exchanged messages. Furthermore, the collaborative consensus building ensures better validation and high detection rate among nodes. As the scenarios are statics, we have also noted slight variations in the detection rate, showing that CONFINIT high ability to handle FDI attacks in dense environments.

The impact of percentages of attacks on the false negative rates is shown in Fig. 6 (**c**). The CONFINIT FPR varied from 2%, with 2% of attackers, to 3.2% with 10% of attackers. The average variation among scenarios was only 1.2%, meaning CONFINIT detects the most of attacking nodes. The attacker's detection failure emerged when there were errors in the similarity computation stage by neighbor nodes, and certain nodes became suspect node. Thus, in the consensus stage, there were a misidentification of these nodes as attackers. The CONFINIT FPR, Fig. 6 (**d**), ranged from 2.8%, with 2% of attackers to 3.6% with 10% of attackers. The slight difference of 0.8% for all scenarios shows a low variation

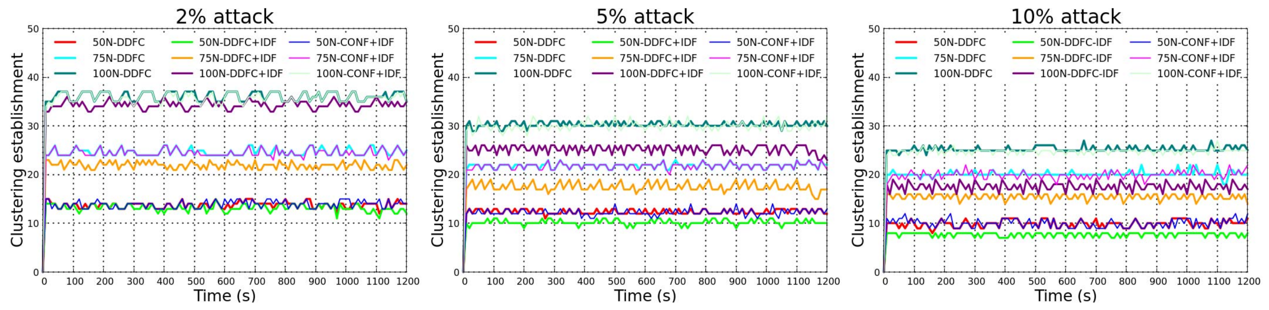

Fig. 5. Number of clustering establishment over time

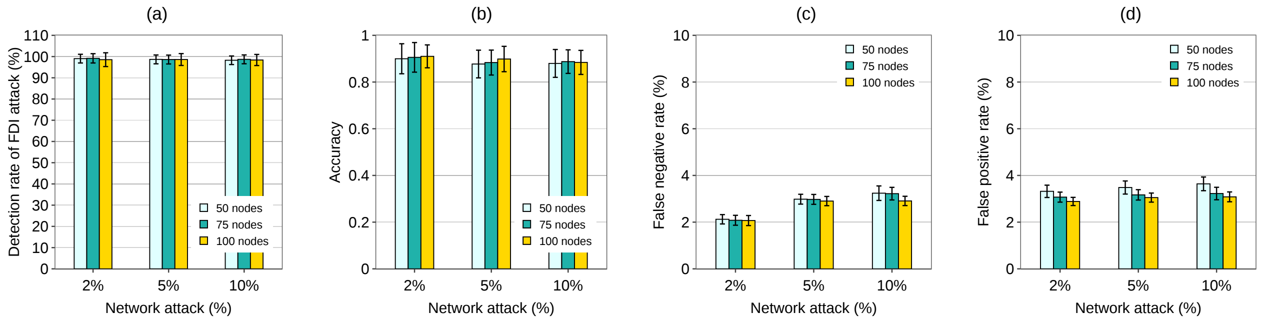

Fig. 6. Detection rate of FDI attacks, Accuracy, False negative and False positive

among them. Detection failures were also due to errors in the consensus computation among monitor nodes of an attacker, which took a low deviation from their readings. Therefore, firstly nodes were added to the individual suspect list, but with new interactions and hence message exchanges, new computations pointed out such nodes as common nodes.

VI. Conclusion

This paper presented the CONFINIT system for mitigating false data injection attacks in dense IoT networks. CONFINIT arranges an IoT network in clusters through the reading similarity among nodes to deal with the density issue. CONFINIT also embraces watchdog strategy and collaborative consensus to monitor misbehavior nodes concerning their read information, neighbors, and aggregate readings to know nodes with malicious behavior about to others. Results have shown the CONFINIT effectiveness against FDI attackers and insuring only the availability of legitimate data. As future work will be evaluated under other contexts of dense IoT that demand data clusters, and the impact of the energy consumption and mobility.

References

[1] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, “Internet of things (iot): A vision, architectural elements, and future directions,” *Future generation computer systems*, vol. 29, no. 7, pp. 1645–1660, 2013.

[2] D. Minoli, K. Sohraby, and B. Occhiogrosso, “Iot considerations, requirements, and architectures for smart buildings—energy optimization and next-generation building management systems,” *IEEE Internet of Things Journal*, vol. 4, no. 1, pp. 269–283, 2017.

[3] S. Mumtaz, A. Alsohaily, Z. Pang, A. Rayes, K. F. Tsang, and J. Rodriguez, “Massive internet of things for industrial applications: Addressing wireless iiot connectivity challenges and ecosystem fragmentation,” *IEEE Industrial Electronics Magazine*, vol. 11, no. 1, pp. 28–33, 2017.

[4] D. Mendez Mena, I. Papapanagiotou, and B. Yang, “Internet of things: Survey on security,” *Information Security Journal: A Global Perspective*, vol. 27, no. 3, pp. 162–182, 2018.

[5] L. Yang, C. Ding, M. Wu, and K. Wang, “Robust detection of false data injection attacks for data aggregation in an internet of things-based environmental surveillance,” *Comp. Net.*, vol. 129, pp. 410–428, 2017.

[6] R. Deng, G. Xiao, R. Lu, H. Liang, and A. V. Vasilakos, “False data injection on state estimation in power systems—attacks, impacts, and defense: A survey,” *IEEE Trans. Ind. Inf*, vol. 13, pp. 411–423, 2016.

[7] R. Lu, X. Lin, H. Zhu, X. Liang, and X. Shen, “Becan: A bandwidth-efficient cooperative authentication scheme for filtering injected false data in wireless sensor networks,” *IEEE transactions on parallel and distributed systems*, vol. 23, no. 1, pp. 32–43, 2012.

[8] B. Li, R. Lu, W. Wang, and K.-K. R. Choo, “Distributed host-based collaborative detection for false data injection attacks in smart grid cyber-physical system,” *JPDC*, vol. 103, pp. 32–41, 2017.

[9] T. P. Raptis, A. Passarella, and M. Conti, “Data management in networked industrial environments: State of the art and open challenges,” *arXiv preprint arXiv:1902.06141*, 2019.

[10] G. Colistra, V. Pilloni, and L. Atzori, “Task allocation in group of nodes in the iot: A consensus approach,” in *2014 IEEE International Conference on Communications (ICC)*. IEEE, 2014, pp. 3848–3853.

[11] A. L. Santos, C. A. Cervantes, M. Nogueira, and B. Kantarci, “Clustering and reliability-driven mitigation of routing attacks in massive iot systems,” *JISA*, vol. 10, no. 1, p. 18, 2019.

[12] Z. Yu and Y. Guan, “A dynamic en-route filtering scheme for data reporting in wireless sensor networks,” *IEEE/ACM Transactions on Networking (ToN)*, vol. 18, no. 1, pp. 150–163, 2010.

[13] M. Toulouse, B. Q. Minh, and P. Curtis, “A consensus based network intrusion detection system,” in *2015 5th International Conference on IT Convergence and Security (ICITCS)*. IEEE, 2015, pp. 1–6.

[14] B. Kailkhura, S. Brahma, and P. K. Varshney, “Consensus based detection in the presence of data falsification attacks,” *arXiv preprint*, 2015.

[15] I. Yaqoob, E. Ahmed, M. H. ur Rehman, A. I. A. Ahmed, and Al-garadi, “The rise of ransomware and emerging security challenges in the internet of things,” *Computer Networks*, vol. 129, pp. 444–458, 2017.

[16] F. Gielow, G. Jakllari, M. Nogueira, and A. Santos, “Data similarity aware dynamic node clustering in wireless sensor networks,” *Ad Hoc Networks*, vol. 24, pp. 29–45, 2015.

[17] C. UCI, “Web access statistics,” https://archive.ics.uci.edu/ml/datasets/Gas+Sensor+Array+Drift+Dataset, 2012, visited in 05/21/2018.