Identifying Malicious Nodes in Multihop IoT Networks using Diversity and Unsupervised Learning

Xin Liu, Mai Abdelhakim, Prashant Krishnamurthy & David Tipper
School of Computing and Information, University of Pittsburgh
E-mail:{xil178, maia, prashk, dtipper}@pitt.edu

***Abstract*—The increased connectivity introduced in Internet of Things (IoT) applications makes such systems vulnerable to serious security threats. In this paper, we consider one of the most challenging threats in IoT networks, where devices manipulate (maliciously or unintentionally) data transmitted in information packets as they are being forwarded from the source to the destination. We propose unsupervised learning that exploits network diversity to detect and identify suspicious networked elements. Our proposed method can identify suspicious nodes along multihop transmission paths and under variable attack levels within the network. More specifically, we formulate a contribution metric for each networked element, which is used as a feature to cluster the nodes based on their behavior. We proposed two detection approaches, namely hard detection and soft detection. In the former, nodes are clustered into malicious or benign group; while in the latter, nodes are clustered into three groups based on their suspicious level, then highly suspicious nodes are discarded and more accurate contribution features are evaluated for the remaining nodes. Soft detection has higher detection accuracy provided that there is sufficient network diversity. Simulation results show the proposed methods achieve high detection accuracy under different percentages of malicious nodes in the network and in the existence of channel errors.**

I. Introduction

Internet of Things (IoT) has transformed many critical infrastructures and daily-life applications, such as smart grids, smart homes, healthcare and smart cities [1]. An IoT network is typically composed of numerous simple devices (such as sensors and actuators) that are interconnected to ubiquitously collect and exchange information, and higher-level powerful devices (such as control units) that gather information and make decisions. Information exchange among distant and power-constrained IoT devices is generally made over multiple hops, forming a multihop mesh network. The mesh topology is a flexible topology that allows any device to communicate with any other device within its communication range. It is adopted by many IoT protocols, such as Insteon smart home, Z-Wave, and ZigBee [2].

IoT networks are vulnerable to various security threats as they suffer from a large attack surface. IoT devices could get compromised through: (i) malicious physical access, which has become more serious since many IoT networks are deployed in public areas, such as roads, parking infrastructures, hotels, and healthcare centers; (ii) malicious access to the local network, where intruders in a close proximity of a device exploit vulnerabilities of local network connections; (iii) malicious remote access through the Internet. By having devices accessible through the Internet, which is a main feature in IoT, intruders could compromise devices through exploiting vulnerabilities in such global connections. Compromised devices represent internal threats to the network, which are more challenging to mitigate than external threats (from unauthenticated devices). Attacks launched by compromised internal sources (internal attacks) could not be solely resolved by cryptographic approaches. It is critical to develop protocols that can detect and identify malicious insiders in IoT networks.

One of the most serious internal attacks is packet modification attack, which is a type of routing attack where a compromised/malicious node along a multihop path modifies the received information/packet (arbitrarily or into malicious contents) before it forwards it to the destination [3], [4]. Note that some malfunctioning devices would accidentally modify data, and hence can be also regarded as malicious nodes. When maliciously modified data[1] is used in a decision-making process, wrong decisions could be made disrupting the operation of the IoT system. Consider a healthcare application, if packets containing personal health information is modified by a malicious relay, wrong or even fatal treatment decisions could be made. Similarly, manipulating information/commands sent from/to security cameras, door locks, and many other IoT elements could lead to critical consequences.

Malicious nodes in the network should be identified, then fixed or replaced. Several countermeasures for detecting malicious modification of data have been proposed in existing work. In [4], a malicious node detection scheme in a tree-shaped wireless sensor network is presented, where a sink counts the percentages of modified packets along each path and uses the relative position information for malicious node identification. Encryption is required at each node for both generating and forwarding each packet, which may not be supported by many of the resource constrained IoT devices. Machine learning methods have been used for detecting malicious attacks [5], [6], [7], [8]. In [5], support vector machine (SVM) is used to detect attacks, but did not identify malicious nodes that launched the attack. In [6], many trusted nodes are deployed in the network to communicate via a single hop with other regular nodes and send related statistical information to a control unit via secure channel; then the information is used by SVM to identify malicious devices. However, in practice, it is hard to guarantee that each node is one-hop away from a trusted device. In [7], K-means clustering is used to classify statistical data tuples from networks into benign or malicious. In [8], authors used discrete time-sliding window to continuously update the feature space and unsupervised incremental grid clustering to identify abnormal flows. Yet, approaches in [7]-[8] focus on identifying abnormal flows in the network, but do not identify attackers that caused these abnormalities. In addition, they assume that most of the network traffic is benign. In [9], we utilized the network diversity and transmitted packets over joint multihop paths to identify malicious relays. However, we considered that there exists at least one reliable path (with no malicious relays)

[1] In the scope of this paper, any packet modification is regarded as malicious.

978-1-5386-3180-5/18/$31.00 ©2018 IEEE

between the source and destination. In this paper, we relax this assumption and consider more general attack model.

In this paper, we propose a novel method that jointly utilizes the diversity of network paths and unsupervised learning for malicious node detection and identification. The approach employs trusted sources that send periodic probe messages over several connected network paths to a destination for identifying the behavior of intermediate network elements. The destination obtains a reputation metric for each path and computes each node's contribution to a path reputation. These contribution metrics are used as features to identify the node's behavior using K-means clustering. Unlike existing learning approaches that assume single-hop communication with a trusted device or detect multihop attacks without identifying attackers, our proposed approach can identify suspicious nodes along multihop transmission paths. We also consider variable attack levels within the network. We proposed two detection approaches, namely hard detection (HD) and soft detection (SD). In HD, nodes are clustered into malicious or benign group based on their contribution level in paths' reputations. In SD, nodes are clustered into three groups based on their suspicious level, then highly suspicious nodes are discarded and more accurate contribution features are evaluated for the remaining nodes. Soft detection has higher detection accuracy provided that there is sufficient network diversity. Simulation results show that the proposed methods achieve high detection accuracy under different percentages of malicious nodes in the network and in the existence of channel errors.

II. Proposed Network Model and Hard Detection

Consider a multihop IoT mesh network composed of a sink and multiple nodes. Some of the nodes are more powerful than others. Powerful nodes act as trusted sources that assist in identifying malicious elements in the network. The identities of trusted sources are revealed to the sink only, and secret cryptographic keys are shared between them. During the route establishment phase, trusted sources store all available routing paths to the sink. Then, they periodically send sequence of probe messages to the sink over many (or all) the available paths to identify intermediate malicious nodes. This type of flooding in packets routing is used in some IoT protocols, such as Z-Wave and Bluetooth mesh. The sink checks the integrity of these messages using a hash function, and utilizes the diversity of paths to compute a reputation metric for each path and contribution feature for each device along the multihop paths. It is noted that with a strong network diversity, paths from a source to the sink are joint (one path may differ from another in one node only). The sink then identifies malicious nodes using K-means clustering. Note that before malicious nodes are identified, messages (other than probe messages) can be exchanged in the network over one path using ad-hoc on-demand distance vector (ADOV) routing protocol or any other routing protocol. After obtaining path reputations and identifying malicious nodes, the sink can inform devices about the most reliable paths to use for routing or about the malicious nodes to be fixed/replaced.

In the rest of the paper, we refer to the sink as the destination D, and focus on identifying malicious nodes between one trusted source (denoted as S) and D. Let N denote the number of nodes assisting in the multihop data transmission between S and D. Let $R_i, i \in [1, N]$ be the i-th node between S and D. In general, there could be multiple paths from S to D and each path may contain different number of relay nodes. We assume that each node has a communication range of radius r. With sufficient network connections/diversity, when transmitting over all available paths each node receives packets from its preceding nodes within r along the path and forwards the received packets to all its succeeding nodes within r[9]. Note that when S transmits one probe packet, D will receive a copy of this packet from each of the available paths.

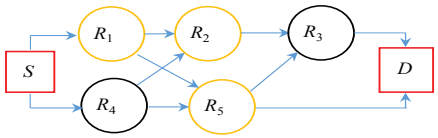

Fig. 1. A mesh network example with diversity of paths.

Define α_i as a binary flag to indicate whether R_i is benign. We assume static attack model, where if R_i is compromised, it could launch routing attack by modifying each packet it forwards by a fixed probability P_i. $\bar{P}_i = 1 - P_i$ is the probability that a forwarded packet will not be modified. If R_i is benign, then $P_i = 0$. That is,

$$\alpha_i = \begin{cases} 1, & \text{if } R_i \text{ is benign,} \quad P_i = 0, \\ 0, & \text{if } R_i \text{ is malicious, } 0 < P_i < 1. \end{cases} \tag{1}$$

In this section, we assume no channel errors. We will discuss how to include channel errors in the next section.

A. Path Reputation and Node Contribution

Our goal is to identify malicious nodes within the network. For this purpose, we calculate path reputation and node contribution. Firstly, S sends multiple active probe packets through many or all of possible paths to D. At D, out of all the received probe packets along a certain path, there could be some modified packets. D obtains the fraction of unmodified packets along a path by checking the integrity of each received probe packet using a cryptographic keyed hash function[2]. The higher the fraction is, the higher reputation the corresponding path has for delivering unmodified packets. Hence, we define a path's reputation as, in a certain time window, the number of unmodified probe packets going through this path divided by the number of all probe packets transmitted through this path. Let all the L paths in the network between S and D be in the set $\mathcal{L} = \{\mathfrak{l}_i\}, i \in [1, L]$. Each $\mathfrak{l}_i$ is a set of nodes representing the i-th path. For example, in Fig. 1, the path "R_1-R_2-R_3" is represented as $\mathfrak{l}_i = \{1, 2, 3\}$.

The reputation value of path $\mathfrak{l}_i$ is denoted by $T_{\mathfrak{l}_i}$. Large $T_{\mathfrak{l}_i}$ reflects more reliable paths, i.e. more packets were received without modification. The path's reputation is the result of how much each node along this path contributes to "not modifying the packets". Hence, we can quantify each node's contribution to a path's reputation. Let's take the network in Fig. 1 as an example, where there are multiple possible paths and total of five relay nodes between S and D. Assume that R_1, R_2 and R_5 are malicious nodes (marked in yellow color). Each available direct transmission channel link is denoted by an arrow. Define $\mathcal{M}_{\mathfrak{l}_i}$ as the set of modified packets along the path $\mathfrak{l}_i$ and $\bar{\mathcal{M}}_{\mathfrak{l}_i}$ as the set of unmodified packets. Firstly,

[2]To achieve this, each packet at source S contains a message concatenated with a corresponding hash value.

we focus on path "R_1-R_2-R_3", through which S transmits Q probe packets. $\mathcal{M}_{1,2,3}$ is the set of the modified packets along path "R_1-R_2-R_3" and $\bar{\mathcal{M}}_{1,2,3}$ is the set of unmodified packets; accordingly, $E[|\mathcal{M}_{1,2,3}|]$ is the expected number of modified packets and $E[|\bar{\mathcal{M}}_{1,2,3}|]$ is the expected number of unmodified packets. Recall that R_i modifies packets with probability P_i, then the expected number of unmodified packets is $E[|\bar{\mathcal{M}}_{1,2,3}|] = \bar{P}_3\bar{P}_2\bar{P}_1Q$. Hence, the expected reputation of path "R_1-R_2-R_3" is

$$E[T_{1,2,3}] = E[|\bar{\mathcal{M}}_{1,2,3}|]/Q = \bar{P}_3\bar{P}_2\bar{P}_1 = \prod_{i=1}^{|\{1,2,3\}|} \bar{P}_i. \quad (2)$$

In fact, $E[|\bar{\mathcal{M}}_{1,2,3}|]$ needs a large number of observations to be calculated, which would create very high overhead and delay. Hence, $T_{1,2,3}$, as the reputation of path "R_1-R_2-R_3" generated by fewer observations, is used to calculate nodes' contribution to this path. We denote the i-th node's contribution to $T_{\mathfrak{l}_j}$ as $C_i^{\mathfrak{l}_j}$. It represents how R_i contributes to the reputation of path $\mathfrak{l}_j$ by not modifying received packets. For each path $\mathfrak{l}_j$, since any prior information about P_i is unknown, our approach is to initially assume that each node contributes equally to the reputation of the path. From Eq. (2), the equality is expressed by $T_{\mathfrak{l}_j} = (C_i^{\mathfrak{l}_j})^{|\mathfrak{l}_j|}$. That is, $C_i^{\mathfrak{l}_j} = \sqrt[|\mathfrak{l}_j|]{T_{\mathfrak{l}_j}}$. For example, in Fig. 1, for path "R_1-R_2-R_3", the contribution for each node is $C_i^{1,2,3} = \sqrt[3]{T_{1,2,3}}, i = 1, 2, 3$. Since each node is associated with multiple paths, a node's contribution in other paths should also be taken into account. Consider R_1, which is associated with another two paths "R_1-R_5" and "R_1-R_5-R_3". Therefore, R_1's contribution in these two paths are $C_1^{1,5} = \sqrt[2]{T_{1,5}}$, $C_1^{1,5,3} = \sqrt[3]{T_{1,5,3}}$, respectively. Then we take the average of R_1's contribution in the three associated paths as R_1's overall contribution, which is $C_1 = (\sqrt[3]{T_{1,2,3}} + \sqrt[2]{T_{1,5}} + \sqrt[3]{T_{1,5,3}})/3$.

Therefore, each node has an overall contribution C_i, and the general process of calculating C_i is as follows: let the node R_i be associated with total of k_i paths, and $k_{i,j}, j \in \{1, ..., k_i\}$ is the j-th path out of these k_i paths. Path $k_{i,j}$ has $|k_{i,j}|$ relay nodes. The reputation of path $k_{i,j}$ is denoted as $T_{i,j}$. For example, in Fig. 1, R_1 is associated with a total of $k_1 = 3$ paths. $k_{1,1} = \{1, 2, 3\}$ ($|k_{1,1}| = 3$) denotes the path "R_1-R_2-R_3", $k_{1,2} = \{1, 5\}$ ($|k_{1,2}| = 2$) denotes the path "R_1-R_5" and $k_{1,3} = \{1, 5, 3\}$ ($|k_{1,3}| = 3$) denotes the path "R_1-R_5-R_3". Therefore, R_i's overall contribution is calculated as

$$C_i = \frac{1}{k_i}\sum_{j=1}^{k_i} \sqrt[|k_{i,j}|]{T_{i,j}}. \quad (3)$$

B. Hard Detection

Generally, if the attack probability of R_i (P_i) is relatively high, i.e., R_i manipulates multiple packets going through it with high probability, then its associated path reputation values are low and it's contribution C_i will be relatively low. On the other hand, benign nodes (with $P_i = 0$) will have relatively high contribution. In this way, a direct approach for malicious node detection is to identify R_i as benign if C_i is relatively high and R_i as malicious if C_i is relatively low.

K-means clustering, as an unsupervised machine learning method, is applied here to distinguish high C_i from low C_i. K-means can cluster the data into multiple groups according to data similarity[10]. Denote the contribution dataset $\mathcal{C} = \{C_i\}, i \in \{1, ..., N\}$. There are two special cases for the detection before using the K-means method: First, if all the nodes are benign, each node's attack probability P_i is 0. In this case, all the path reputation is 1, resulting in each node's overall contribution being 1. Hence, we identify all nodes as benign. Second, if all nodes are malicious, most path reputation values are generally close to 0. In our approach, if the average of all paths' reputation values is less than a threshold ε, which is a small value, then all the nodes are identified as malicious. The path reputation average value is calculated as $\bar{T} = \frac{1}{L}\sum_{i=1}^{L} T_{\mathfrak{l}_i}$.

If the above two cases are not met, K-means method is used to cluster the nodes in two groups. The node group with higher contribution values is identified as the benign node set $\mathcal{G}_B^{(HD)}$ while the other group is the malicious set $\mathcal{G}_M^{(HD)}$. We name the proposed algorithm as Hard Detection (HD), since nodes are classified into two groups. The steps of this algorithm are decribed in Algorithm 1. Note that K-means$(\mathcal{A}, n)$ refers to clustering elements in the dataset $\mathcal{A}$ into n groups.

Algorithm 1 Hard Detection: HD$(\mathcal{C}, \varepsilon, \mathcal{L})$

Input dataset $\mathcal{C}$, threshold ε, path set $\mathcal{L}$;
if $\mathcal{C} = \bar{1}$ **then**
    $\mathcal{G}_B^{(HD)} = \{1, ..., N\}$, $\mathcal{G}_M^{(HD)} = \varnothing$;
else if $\bar{T} < \varepsilon$ **then**
    $\mathcal{G}_M^{(HD)} = \{1, ..., N\}$, $\mathcal{G}_B^{(HD)} = \varnothing$;
else
    Use K-means($\mathcal{C}$,2) to cluster $\mathcal{C}$ into 2 groups. The group with higher data values is $\mathcal{G}_B^{(HD)}$; the other group is $\mathcal{G}_M^{(HD)}$;
end if
Output benign node set:$\mathcal{G}_B^{(HD)}$; malicious node set:$\mathcal{G}_M^{(HD)}$.

III. Soft Detection and Channel Errors

A. Soft Detection

As explained in the previous section, nodes with relatively high P_i are expected to have low contribution, and benign nodes with $P_i = 0$ are expected to have high contribution. However, there can be some nodes with low P_i and have contribution values in a intermediate level. This is because the contribution of a node is impacted by the behavior of other devices along its associated multihop path(s). In K-means clustering with K=2, these intermediate levels will still be assigned to either benign or malicious, leading to misdetection (malicious nodes identified as benign) and false alarms (benign nodes identified as malicious).

To solve this problem, we propose soft detection, where we use three clusters instead of two. The three clusters represent high, medium and low contribution metrics. Here, we identify the malicious nodes with high P_i first and then identify other nodes next. The corresponding node sets to the high, medium, low contributions are denoted as $\mathcal{G}_1^1$, $\mathcal{G}_2^1$ and $\mathcal{G}_3^1$, respectively. All nodes in $\mathcal{G}_3^1$ are identified as malicious (with high P_i). However, we do not directly identify nodes in $\mathcal{G}_1^1$ as benign and $\mathcal{G}_2^1$ as malicious (with low P_i) since their contribution can be calculated very inaccurately when nodes in $\mathcal{G}_3^1$ are involved. Therefore, the influence of nodes with high P_i should be canceled from the calculation of other nodes' contribution.

Our approach is to firstly record the original path set $\mathcal{L}_0$ and contribution set $\mathcal{C}_0$ as $\mathcal{L}_0 = \mathcal{L}$, $\mathcal{C}_0 = \mathcal{C}$. Then we exclude any path, which contains any of the nodes in $\mathcal{G}_3^1$, from the path set $\mathcal{L}$; in this way, $\mathcal{L}$ is updated. The steps to execute the path set update are illustrated in Algorithm 2 (Path Set Update (PSU)). After updating the path set $\mathcal{L}$, we recalculate the contribution dataset $\mathcal{C}$ for the remaining (unidentified) node set $\mathcal{G}_U \triangleq \mathcal{G}_1^1 \cup \mathcal{G}_2^1$ and then use K-means($\mathcal{C}$,2) to cluster nodes in $\mathcal{G}_U$ into two sets: $\mathcal{G}_1^2$ and $\mathcal{G}_2^2$. The nodes with higher contribution ($\mathcal{G}_1^2$) are benign, while others ($\mathcal{G}_2^2$) are malicious. We name this proposed method as Soft Detection (SD) Algorithm, which is described in Algorithm 3.

Algorithm 2 Path Set Update: PSU($\mathcal{L}, \mathcal{G}_3^1$)

1: Input path set $\mathcal{L}$, node set $\mathcal{G}_3^1$, $i = 1$;
2: **while** $i \leqslant L$ **do**
3: **if** $\mathcal{G}_3^1 \cap \mathfrak{l}_i \neq \varnothing$ **then**
4: $\mathcal{L} \leftarrow \mathcal{L} \setminus \mathfrak{l}_i$;
5: **end if**
6: $i \leftarrow i + 1$;
7: **end while**
8: Output updated $\mathcal{L}$.

Condition 1: Our proposed approach for malicious node detection relies on having sufficient diversity to calculate nodes' contributions. After path set update, nodes in $\mathcal{G}_U$ along with S and D form another graph topology different from the original topology formed by all the nodes. We define $\tilde{d}_U$ as the average node degree of the graph formed by nodes in $\mathcal{G}_U$, S and D. If $\tilde{d}_U$ is lower than a predefined threshold η, then the diversity in the remaining topology is insufficient for evaluating new contribution features. In this case, we apply the HD algorithm to identify the behavior of nodes in $\mathcal{G}_U$. Note that $\eta > 2$ to ensure that every node is connected to more than one path. For example, for the network in Fig. 1, if $\mathcal{G}_3^1$=$\{R_1, R_2\}$, then the updated $\mathcal{L}$ contains two paths only, i.e., R_3-R_4-R_5 and R_4-R_5, to form a graph with S and D. After discarding $\mathcal{G}_3^1$, the average node degree is $\tilde{d}_U$=2. This means that an intermediate node is connected to one path only (has a link with a preceding node along a path and another link with a succeeding node, hence its degree is 2). Since $\tilde{d}_U \leqslant \eta$, we apply the HD results. In some cases, the updated $\mathcal{L}$ may even contain no paths after elements in $\mathcal{G}_3^1$ are removed from the network topology; in these cases, we also apply HD results. Our approach of applying HD results is to input the original path set $\mathcal{L}_0$ and contribution set $\mathcal{C}_0$ to HD algorithm and apply the results from HD algorithm to nodes in $\mathcal{G}_U$, as illustrated in Fig. 2. In the figure, $\beta_i = \hat{\alpha}_i$ is the estimated value of α_i. That is, $\beta_i = 1$ or $\beta_i = 0$ if node R_i is identified as benign or malicious, respectively. The figure shows that the detection results from HD algorithm ($\beta_3 = 1$, $\beta_4 = 1$ and $\beta_5 = 0$) are applied to nodes in $\mathcal{G}_U$ (since $\tilde{d}_U \leqslant \eta$).

Condition 2: Even if $\tilde{d}_U > \eta$, there may be some nodes in $\mathcal{G}_U$ not associated with any path in the updated $\mathcal{L}$. These nodes still remain unidentified after executing $(\mathcal{G}_1^2, \mathcal{G}_2^2) = \text{HD}(\mathcal{C}, \varepsilon, \mathcal{L})$ in Line 14 in SD Algorithm. In this case, the detection results from HD (using the original topology as the input) are applied to identify the behavior of these nodes.

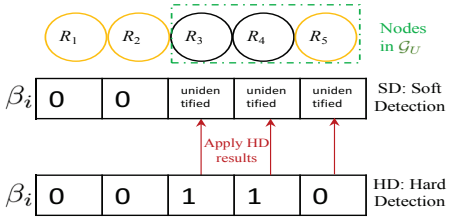

Fig. 2. Apply the HD results to SD.

Algorithm 3 Soft Detection: SD($\mathcal{C}, \varepsilon, \eta, \mathcal{L}$)

1: Input dataset $\mathcal{C}$, threshold ε, η, path set $\mathcal{L}$;
2: **if** $\mathcal{C} = \bar{1}$ **then**
3: $\mathcal{G}_B^{(SD)} = \{1, ..., N\}$, $\mathcal{G}_M^{(SD)} = \varnothing$;
4: **else if** $\tilde{T} < \varepsilon_1$ **then**
5: $\mathcal{G}_M^{(SD)} = \{1, ..., N\}$, $\mathcal{G}_B^{(SD)} = \varnothing$;
6: **else**
7: Use k-means($\mathcal{C}$,3) to cluster $\mathcal{C}$ into three groups where the group with the lowest data values is the malicious node set $\mathcal{G}_3^1$; the other two groups $\mathcal{G}_1^1$ and $\mathcal{G}_2^1$ are considered as an unidentified node set $\mathcal{G}_U \triangleq \mathcal{G}_1^1 \cup \mathcal{G}_2^1$;
8: $\mathcal{L}_0 = \mathcal{L}$; $\mathcal{C}_0 = \mathcal{C}$; $(\mathcal{G}_B^0, \mathcal{G}_M^0) = \text{HD}(\mathcal{C}_0, \varepsilon, \mathcal{L}_0)$;
9: Update $\mathcal{L} = \text{PSU}(\mathcal{L}, \mathcal{G}_3^1)$; Update $\mathcal{C}, \mathcal{T}$ based on $\mathcal{L}$;
10: **if** $(\tilde{d}_U \leqslant \eta$ and $\mathcal{C} \neq \bar{1})$ or $|\mathcal{L}| = 0$ **then**
11: Apply the results from $\mathcal{G}_B^0, \mathcal{G}_M^0$ to nodes in $\mathcal{G}_U$;
12: Let $\mathcal{G}_M^{(SD)}$ be the union of $\mathcal{G}_3^1$ and malicious nodes in $\mathcal{G}_U$, and the remaining nodes belong to $\mathcal{G}_B^{(SD)}$;
13: **else**
14: $(\mathcal{G}_1^2, \mathcal{G}_2^2)$=HD($\mathcal{C}, \varepsilon, \mathcal{L}$);$\mathcal{G}_M^{(SD)}$=$\mathcal{G}_3^1 \cup \mathcal{G}_2^2$; $\mathcal{G}_B^{(SD)}$=$\mathcal{G}_1^2$;
15: **end if**
16: **end if**
17: Output benign node set: $\mathcal{G}_B^{(SD)}$; malicious node set: $\mathcal{G}_M^{(SD)}$.

B. Enhanced Soft Detection (ESD)

We propose an enhanced soft detection (ESD) algorithm, which improves the detection accuracy of the SD algorithm. More specifically, we here utilize benign paths to correct misdetection or false alarms resulted from SD algorithm. From our previous discussions, one can infer the following: (i) if a path's reputation is 1, then nodes along this path do not modify packets. In other words, all nodes along this path are benign. Nodes meeting such a standard are classified into the benign node set $\mathcal{G}_B^{(ESD)}$; (ii) for a path with reputation less than 1, if there is only one node in this path not belonging to $\mathcal{G}_B^{(ESD)}$, then that node is malicious since it is the only possible node to modify packets. Nodes meeting such standard are classified into malicious node set $\mathcal{G}_M^{(ESD)}$. ESD algorithm is executed after SD to improve the detection accuracy. That is, if there is any detection result conflict from SD and ESD to R_i, we accept the result from ESD for R_i. Note that ESD algorithm can also be used to improve the detection from HD algorithm in a similar way. ESD algorithm is described in Algorithm 4.

C. Reputation Correction under Channel Errors

In this section, we discuss the impact of channel errors on the proposed detection algorithms. Consider that the channel

Algorithm 4 Enhanced Soft Detection: ESD($\mathcal{L}_0$)

1: Define $\mathcal{U}$ as the set containing all the nodes;
2: Input path set $\mathcal{L}_0$, node set $\mathcal{G}_B^{(ESD)} = \mathcal{G}_M^{(ESD)} = \varnothing$, $i = 1$;
3: **while** $i \leqslant L$ **do**
4: **if** $T_{\mathfrak{l}_i} = 1$ **then**
5: $\mathcal{G}_B^{(ESD)} \leftarrow \mathcal{G}_B^{(ESD)} \cup \mathfrak{l}_i$;
6: **end if**
7: $i \leftarrow i + 1$;
8: **end while**
9: $(\mathcal{G}_B^{(ESD)})^C \leftarrow \mathcal{U} - \mathcal{G}_B^{(ESD)}$, $i = 1$;
10: **while** $i \leqslant L$ **do**
11: **if** $T_{\mathfrak{l}_i} \neq 1$ and $|\mathfrak{l}_i \cap (\mathcal{G}_B^{(ESD)})^C| = 1$ **then**
12: $\mathcal{G}_M^{(ESD)} \leftarrow \mathcal{G}_M^{(ESD)} \cup (\mathfrak{l}_i \cap (\mathcal{G}_B^{(ESD)})^C)$;
13: **end if**
14: $i \leftarrow i + 1$;
15: **end while**
16: Output benign node set: $\mathcal{G}_B^{(ESD)}$ and malicious node set: $\mathcal{G}_M^{(ESD)}$.

packet error rate is the same for all links, and is denoted by μ. Thus, the rate of correct transmission per link is $\bar{\mu} = 1 - \mu$. μ can be estimated through pilot or training packets. We can follow a similar analysis as in Section II-A to get nodes' contribution metrics. For example, for the link $S - R_1$, where S transmits Q packets to R_1, the expected number of received packets that are unmodified is $Q\bar{\mu}$. Then, consider path "S-R_1-R_2-R_3-D" where there are four intermediate channel links. Now, the expected path reputation is $E[T_{1,2,3}] = \bar{\mu}^4 \bar{P}_1 \bar{P}_2 \bar{P}_3$.

The effect of channel error on $T_{1,2,3}$ can be canceled by updating: $T_{1,2,3} \leftarrow T_{1,2,3}/(\bar{\mu}^4)$. Then all the updated reputation values are used to calculate the nodes' contributions then identify malicious nodes by the SD Algorithm in a similar manner. It is noted that since $\bar{\mu}^4 < 1$, after updating $T_{\mathfrak{l}_i}$ by removing channel error influences, it's possible to have $T_{\mathfrak{l}_i} > 1$, which is not reasonable. In this situation, we set $T_{\mathfrak{l}_i} = 1$. The above steps are shown in Algorithm 5. Note that if we input the corrected reputations $\mathcal{T}$ into SD Algorithm (Algorithm 3), some minor modifications are needed[3]. In particular, "$\mathcal{C} = \bar{1}$" in Line 2 should be replaced with "avg($\mathcal{C}$) $\geqslant \varepsilon_{RC}$" and "$\mathcal{C} \neq \bar{1}$" in Line 10 should be replaced with "avg($\mathcal{C}$) $< \varepsilon_{RC}$" to tolerate the channel error effect. ε_{RC} is very close to 1 and "avg($\mathcal{C}$)" refers to the average value of elements in set $\mathcal{C}$.

Algorithm 5 Reputation Correction (RC)

1: Input path set $\mathcal{L}$, path reputation set T, $i = 1$;
2: **while** $i \leqslant L$ **do**
3: $T_{\mathfrak{l}_i} \leftarrow T_{\mathfrak{l}_i}/(\bar{\mu}^{(|\mathfrak{l}_i|+1)})$;
4: **if** $T_{\mathfrak{l}_i} > 1$ **then**
5: $T_{\mathfrak{l}_i} \leftarrow 1$;
6: **end if**
7: $i \leftarrow i + 1$;
8: **end while**

[3]ESD Algorithm is not applicable to situations under channel errors since inference (i) in Section III-B would be incorrect with channel errors.

IV. Numerical Results

In this section, we compare the performance of HD and SD algorithms. The ESD Algorithm is applied to both HD and SD. We also show the impact of number of hops, network diversity and the percentage of malicious nodes on detection accuracy.

In the simulation, we assume that between source S and destination D, there are N relay nodes uniformly distributed in a $(6N)\times 15\text{m}^2$ rectangle; the node density is $0.01/\text{m}^2$. Each node's communication range is r=20m. S and D are positioned at the left and right edges of the rectangle, respectively. We generate 20 random networks. For each network, simulations are done in 300 rounds. In each round, unless stated otherwise, we randomly choose 30% of the nodes to be malicious. Each malicious node's attack probability P_i is a random value in the range of [0.2,0.8] where values are uniformly distributed. Once assigned, P_i is fixed throughout the simulation. We obtain all possible paths from S and D and randomly choose 33 paths in each round for packet transmission. We use $Q = 200$ probe packets transmitted through each path[4]. By experimental observations, we set $\eta = 3.7$ and $\varepsilon = 0.0009$.

To evaluate the performance of the proposed approaches, we evaluate the detection accuracy and false alarm rate. The detection accuracy is defined as: P_d = Number of correctly identified malicious nodes / Number of malicious nodes. The false alarm rate is defined as: F_a =Number of benign nodes identified as malicious / Number of benign nodes. All the results are measured and averaged based on all simulation rounds over 20 random networks, which are simulated using Matlab.

Example 1: impact of the number of hops. Here, we examine the impact of number of hops on P_d and F_a of HD and SD algorithms. First, P_d and F_a of HD and SD versus the number of relay nodes N are plotted in Fig. 3. The average number of hops ("avg(hop)") for each N is marked in the figure. It is observed that when N=6, $P_d \simeq 0.94$ and F_a is very close to 0; more relay nodes corresponds to more number of hops, and cause P_d to decrease and F_a to increase. This is because, when the percentage of malicious nodes is fixed, more hops introduce more uncertainty (the unknown P_i from each malicious node) to the network. Moreover, longer path results in more packet copies, and hence larger overhead. Therefore, the number of hops along a path should be limited.

Example 2: impact of network diversity. In this example, we show the impact of network diversity on detection accuracy with $N = 10$. We denote the maximum number of hops as N_p and set it as $N_p = 6$; that is, each path utilized for packet transmission has no more than 6 hops. Here, based on the selected 33 paths, we randomly choose 20%, 40%, 60%, 80%, 100% of these 33 paths to be utilized for packet transmission, as shown in Fig. 4. It can be demonstrated that when more paths are utilized, the detection accuracy increases and the false alarm degrades. Moreover, compared to HD algorithm, SD has high detection accuracy, at the expense of slightly higher false alarm rate.

Example 3: impact of percentage of malicious nodes. Here, we set N=10, N_p=6 and examine the impact of the percentages of the malicious nodes on the accuracy of proposed

[4]S transmits 200 packets in one path and then switch to another path to start the transmission.

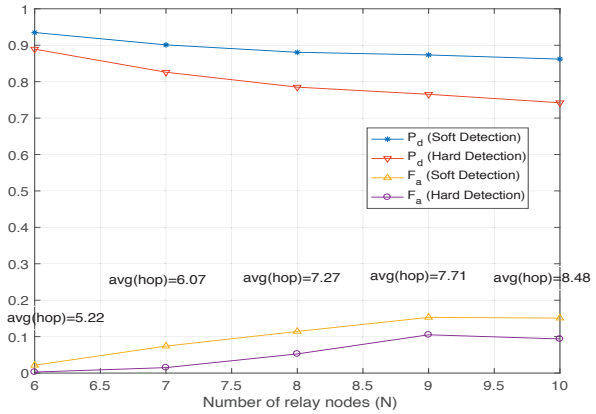

Fig. 3. P_d and F_a as functions of number of relaying nodes with 30% malicious nodes and 33 paths.

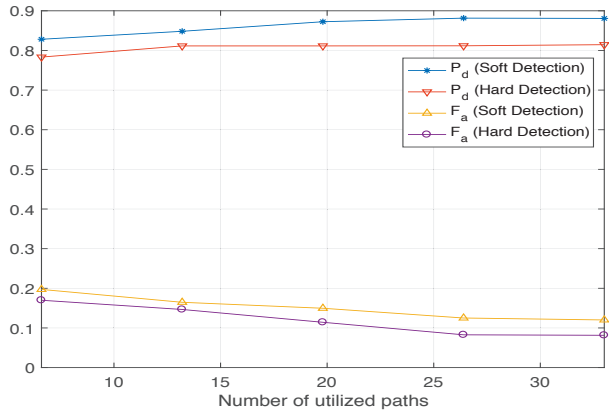

Fig. 4. P_d and F_a as functions of number of paths with N=10, N_p=6, 30% malicious nodes.

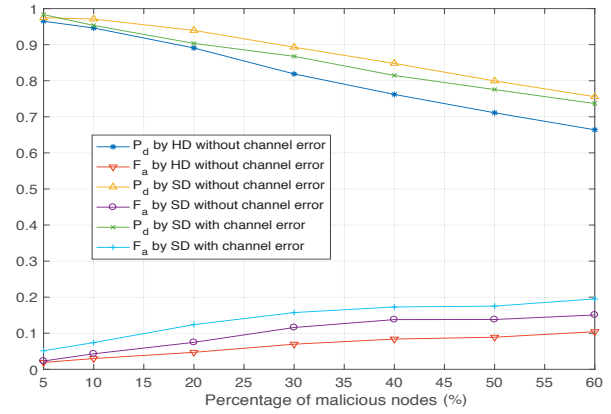

Fig. 5. P_d and F_a as functions of percentage of malicious nodes with N=10, N_p=6, and 33 paths.

approaches. In Fig. 5, we plot P_d and F_a as functions of percentage of malicious nodes. As expected, the detection accuracy decreases as the percentage of malicious nodes increases. The reason is that there are fewer reliable paths. We also execute RC algorithm to assist SD algorithm in the existence of channel errors. We set $\mu = 2\%$. Note that ESD algorithm is not utilized under channel errors. This figure shows that the detection accuracy of SD under channel errors is close to the situation of without channel errors.

V. Conclusions

With the large attack surface of IoT systems, it is critical to develop approaches that detect and identify malicious insiders. In this paper, we proposed to use unsupervised learning that exploits the diversity of network paths to identify malicious nodes launching packet modification attacks in a multihop IoT network. Particularly, we formulated a nodes' contribution metric to be used as a feature to cluster nodes by K-means according to their behavior. Two algorithms were proposed: hard detection (HD) and soft detection (SD). The HD algorithm clusters nodes into benign and malicious groups. To further consider the variability of attack probabilities, the SD algorithm cluster nodes into three groups based on their suspicious levels; then highly suspicious nodes are removed and more accurate contribution feature is calculated for the remaining nodes. We also analyzed the impact of channel errors on the detection performance. Simulation results showed that SD algorithm has higher detection accuracy compared to the HD algorithm under different percentages of malicious nodes, provided that there is sufficient network diversity.

References

[1] Y. Mehmood, F. Ahmad, I. Yaqoob, A. Adnane, M. Imran and S. Guizani, "Internet-of-Things-based smart cities: recent advances and challenges," in *IEEE Commun. Mag.*, vol. 55, no. 9, pp. 16-24, Sep. 2017.

[2] C Withanage, R Ashok, C Yuen and K. Otto, "A comparison of the popular home automation technologies." in *Proc. IEEE ISGT-Asia*, pp. 600-605, May 2014.

[3] N. Komninos, E. Philippou and A. Pitsillides, "Survey in smart grid and smart home security: Issues, challenges and countermeasures." in *IEEE Commun. Surveys Tuts.*, vol. 16, no. 4, pp. 1933-1954, Apr. 2014.

[4] C. Wang, T. Feng, J. Kim, G. Wang, and W. Zhang, "Catching packet droppers and modifiers in wireless sensor networks" in *IEEE Trans. Parallel Distrib. Syst.*, vol. 23, no. 5, pp. 835-843, May 2012.

[5] S. Kaplantzis, A. Shilton, N. Mani and Y. A. Sekercioglu, "Detecting selective forwarding attacks in wireless sensor networks using support vector machines" in *Proc. IEEE International Conference on Intelligent Sensors, Sensor Networks and Information*, pp. 335-340, Apr. 2008.

[6] R. Akbani, T. Korkmaz and G. V. S. Raju, "A machine learning based reputation system for defending against malicious node behavior" in *Proc. IEEE Globecom*, pp. 1-5, Dec. 2008.

[7] K. Nahiyan, S. Kaiser, K. Ferens and R. McLeod, "A multi-agent based cognitive approach to unsupervised feature extraction and classification for network intrusion detection," in *Proc. Int'l Conf. on Advances on Applied Cognitive Computing*, pp. 25-30, Feb. 2017.

[8] J. Dromard, G. Roudiere and P. Owezarski, " Online and scalable unsupervised network anomaly detection method," in *IEEE Trans. Netw. Service Manag.*, vol. 14, no. 1, pp. 34-47, Mar. 2017.

[9] M. Abdelhakim, J. Ren, T. Li, "Reliable communications over multihop networks under routing attacks." in *Proc. IEEE Globecom*, pp. 1-6, Dec. 2015.

[10] K. J. Cios, W. Pedrycz, R. W. Swiniarski, L. A. Kurgan, "Data mining: a knowledge discovery approach," Springer Science & Business Media, 2007.