Research Article

Fortifying Intrusion Detection Systems in Dynamic Ad Hoc and Wireless Sensor Networks

Abdelouahid Derhab,1 Abdelghani Bouras,2 Mustapha Reda Senouci,3 and Muhammad Imran1

1King Saud University, P.O. Box 92144, Riyadh 11543, Saudi Arabia
2Industrial Engineering Department, College of Engineering, King Saud University, P.O. Box 800, Riyadh 11421, Saudi Arabia
3Laboratory of Research in Artificial Intelligence, Ecole Militaire Polytechnique, P.O. Box 17, Bordj-El-Bahri, 16111 Algiers, Algeria

Correspondence should be addressed to Abdelouahid Derhab; abderhab@ksu.edu.sa

Received 24 September 2014; Revised 1 December 2014; Accepted 4 December 2014; Published 29 December 2014

Academic Editor: Jinsung Cho

We investigate three aspects of dynamicity in ad hoc and wireless sensor networks and their impact on the efficiency of intrusion detection systems (IDSs). The first aspect is magnitude dynamicity, in which the IDS has to efficiently determine whether the changes occurring in the network are due to malicious behaviors or due to normal changing of user requirements. The second aspect is nature dynamicity that occurs when a malicious node is continuously switching its behavior between normal and anomalous to cause maximum network disruption without being detected by the IDS. The third aspect, named spatiotemporal dynamicity, happens when a malicious node moves out of the IDS range before the latter can make an observation about its behavior. The first aspect is solved by defining a normal profile based on the invariants derived from the normal node behavior. The second aspect is handled by proposing an adaptive reputation fading strategy that allows fast redemption and fast capture of malicious node. The third aspect is solved by estimating the link duration between two nodes in dynamic network topology, which allows choosing the appropriate monitoring period. We provide analytical studies and simulation experiments to demonstrate the efficiency of the proposed solutions.

1. Introduction

Multihop ad hoc wireless networks are a set of nodes equipped with wireless interfaces, and data are forwarded through multiple nodes to reach the intended destinations. They include many types of networks such as mobile ad hoc networks (MANETs) [1], wireless sensor networks (WSNs) [2], and vehicular ad hoc networks (VANETs) [3].

In the last decade, there has been a substantial research in the area of security in ad hoc and wireless sensor networks [4, 5]. The security solutions have been designed with the goal of protecting the networks against some attacks such as selective forwarding, black hole, wormhole, sinkhole, and energy exhausting attack. Prevention mechanisms like key management and authentication, which represent the first line of defense, are not sufficient to provide an efficient security solution. Therefore, there is a need to deploy a second line of defense named intrusion detection system (IDS).

In general, intrusion detection systems are divided into two major approaches: misuse detection and anomaly detection [6]. Misuse detection performs signature analysis by comparing on-going activities with patterns representing known attacks, and those matched are labeled as intrusive attacks. The misuse approach is showing its limits as it cannot detect new attacks. Anomaly detection, on the other hand, builds profile of normal behavior and attempts to identify the patterns or activities that deviate from the normal profile. The main advantage of anomaly detection is that it can detect unknown attacks.

The detection model that we consider in the ad hoc and wireless sensor network is as follows. The IDS is implemented in a distributed manner; each node can act as a monitoring node that observes the behavior of its neighbors. Each observation lasts for a monitoring time interval of duration $\Delta$, called the monitoring period. The IDS can judge whether the
monitored node is normal or anomalous after one or multiple consecutive observations.

Although intrusion detection systems have received considerable attention in ad hoc and wireless sensor networks [7, 8], to the best of our knowledge, there are no studies on the impact of network dynamicity on IDS efficiency and how the IDS can react or adapt to these changes.

In this paper, we investigate the following three aspects of behavioral dynamicity that occur in the network and can negatively affect the IDS performance and efficiency.

(i) **Magnitude Dynamicity.** Due to change of user requirements, a node changes the rate at which it generates data. For instance, a legitimate user wants to change (i.e., increase/decrease) the data collection rate received at the sink node. The challenge facing the IDS here is to be efficient at detecting attacks and distinguishing between the changes due to normal behaviors and the changes due to malicious attacks.

(ii) **Nature Dynamicity.** In some detection models, a monitoring node has to observe the behavior of the monitored node during a set of consecutive monitoring periods before judging whether the monitored node is malicious or not. A monitored node might evade from IDS detection and confuses it by switching continuously its behavior between normal and anomalous. In this case, the malicious node strives to cause network disruption without being detected by the IDS.

(iii) **Spatiotemporal Dynamicity.** The IDS detection mechanism is based on collecting a set of consecutive observations about the monitored node. An IDS is able to observe the behavior of the monitored node if the latter stays within the monitoring node’s transmission range for a duration exceeding Δ. By knowing this fact, a malicious node can evade IDS detection by moving around in the network at a speed, which prevents it from being within the monitoring node’s transmission range for a duration higher than Δ.

In this paper, we propose a solution for each aspect of dynamicity mentioned above. The contributions of the paper are threefold. Firstly, the magnitude dynamicity aspect is solved by defining a normal profile based on the invariants derived from the normal node behavior. This is achieved by generating a dependency graph consisting of strongly correlated features and then derives the high-level features from the graph. The high-level features are obtained by applying the divide-and-conquer strategy on the maximal cliques algorithm and the maximum weighted spanning tree algorithm. Secondly, to handle nature dynamicity aspect, we adopt the carrot and stick strategy (i.e., reward generously and punish severely) to prevent a malicious node from evading the IDS. To do so, we propose an adaptive reputation fading strategy to allow fast redemption and fast capture of malicious node. Thirdly, we use statistical analysis to estimate the link duration between two nodes in dynamic network topology. Based on this estimation, the monitoring node chooses the appropriate monitoring period, which allows it to observe the monitored node’s behavior.

The rest of the paper is organized as follows. In Section 2, we describe the normal profile construction and the feature selection method. Section 3 presents the adaptive reputation fading strategy. In Section 4, we analyze link-node duration in a mobile wireless network and explain how the monitoring time period is estimated. Finally, Section 5 concludes the paper.

### 2. Magnitude Dynamicity

#### 2.1. Background

##### 2.1.1. One-Feature Profile.** In the one-feature profile, we use a single feature to describe and detect anomalous behavior. To detect the network malicious behavior, a node can measure the following features, as shown in Table 1 [9]. The disadvantage of this profile structure is that there is a need to assign one feature for each known attack. In this case, the IDS has to measure each feature and check whether it has anomalous value. When the number of attacks increases, the detection speed of the IDS becomes slow. It also becomes slower when the size of rule set increases.

The one-feature profile might fail at distinguishing between normal and anomalous behaviors. Figure 1 shows that using some features individually to describe normal behavior is misleading and might make the detection system falsely accuse a legitimate node of being malicious. Figure 1(a) depicts a tree-based wireless network rooted at the sink B and it shows the normal traffic rates of the network. The value above each link indicates the flow rate traversing this link. Each node measures the flow rate coming from its
Table 1: Relation between attacks and features.

| Attack                  | Feature                        |
|-------------------------|--------------------------------|
| Packet sending rate     | Energy exhausting attack       |
| Packet dropping rate    | Selective forwarding and black hole attacks |
| Packet receiving rate   | Sinkhole attack                |
| Packet sending power    | Hello attack, wormhole attack  |

upstream neighbors. Figure 1(b) (resp., Figure 1(c)) shows the state of the network when nodes D, H, and K become compromised and start behaving maliciously by dropping some packets (resp., generating more packets). As D, H, and K reduce (resp., increase) their sending rate, their respective downstream neighbors I and L have also to reduce (resp., increase) their sending rate accordingly. As a result, node B will falsely accuse nodes I and L of performing selective forwarding attack (resp., energy exhaustion attack), and hence a high false positive rate will be observed.

2.1.2. Multifeature Profile. In the multifeature profile, we describe the normal behavior by a d-feature vector and each element of the vector represents a feature. In this way, the IDS can determine whether some features together show an anomalous behavior. Experiments have shown that we can obtain better detection accuracy by combining related features rather than individually [10]. If node B in the example of Figure 1 considers two features: (a) the flow entering the monitored node and (b) the flow leaving the monitored node, it will conclude that nodes I and L are just forwarding what they received from their upstream neighbors, and hence they are not malicious.

Loo et al. [11] group the observed data into clusters and use a profile of 12 features to describe normal profile. To check whether a test instance belongs to a given cluster, they measure the Euclidian distance between the test point and the centroid of the cluster. If such a distance is higher than a threshold distance, the test point is considered anomalous. The following example shows that the Euclidian distance between two d-feature profiles reduces the detection accuracy. Let (f_{t1}, f_{t2}) be a vector profile such that each feature of the vector is used to detect one attack. f_{t1} and f_{t2} take values in [0, 10]. Let (10, 10) be the centroid vector. The first and the second attacks are detected when f_{t1} ≤ 7 and f_{t2} ≤ 6, respectively. We take the distance between (10, 10) and (7, 6), which is 5, as the threshold distance. Let a test vector be (6, 10); the distance between the two vectors is 4, which is lower than the distance threshold. In this case, the test point will be considered normal whereas the value of f_{t1} individually indicates the occurrence of an attack. The above example shows that aggregating features through the use of Euclidian distance result in loss of detection accuracy.

In [9, 12], the normal profile of a monitored node i is defined by a q-feature vector f_i = (f_{i1}, ..., f_{iq}). If a node monitors a set of n nodes, it forms a matrix F = (f_1, ..., f_n)^T. Both schemes assume that all feature vectors f_i follow the same multivariate normal distribution with mean μ and variance-covariance matrix \( \Sigma \). Node i is considered suspicious if the Mahalanobis distance between f_i and the center of the set F is greater than a predefined threshold. The authors of both works use the orthogonalized Gnanadesikan-Kettenring estimation to find the center of the set F. Let \( \hat{\mu} \) and \( \hat{\Sigma} \) denote the simple mean and the simple variance-covariance of F such that \( \hat{\mu} = (1/n) \sum_{i=1}^n f_i \) and \( \hat{\Sigma} = (1/(n-1)) \sum_{i=1}^n (f_i - \hat{\mu})(f_i - \hat{\mu})^T \). The Mahalanobis distance between f_i and the vector \( \hat{\mu} \) is given by \( \sqrt{(f_i - \hat{\mu})^T \hat{\Sigma}^{-1} (f_i - \hat{\mu})} \). The Mahalanobis distance differs from the Euclidian distance in that it takes into account the correlations between features. In [12], nodes are evaluated in terms of packet dropping rate, packet sending rate, forwarding delay time, and node readings. In [9], the attacks are detected by monitoring packet sending rate, packet dropping rate, packet mismatch rate, packet receiving rate, and received signal strength. As stated in [13], the works of [9, 12] suffer from two major criticisms: (1) the circumstances, under which the assumption of multivariate normal distribution holds, are not explained, and (2) the network features such as packet sending, packet dropping, and packet receiving rates do not follow the normal distribution for tree-based routing protocol.

2.2. Profile Construction Based on Strongly Correlated Features. When it comes to comparing distances, we find that the Mahalanobis distance is a powerful technique as it takes the covariances into account, which leads to elliptic decision boundaries in the 2D space. While the Euclidian distance builds circular boundaries and considers equal variances of the features, it appears that the Mahalanobis distance is more appropriate for multivariate data.

In our paper, we take a novel approach to select relevant features and construct the normal profile vector. We do not assume multivariate normal distribution and we feed only strongly correlated features to the distance measure unlike the Mahalanobis distance, which considers correlation between all features.

In the training phase, we investigate the significant associations between features. We are interested in identifying the level of correlation between those features, called Pearson’s correlation coefficient, which measures the strength of the linear association between features. Pearson’s correlation coefficient between two feature vectors X and Y is defined by

\[
\rho(X, Y) = \frac{\text{COV}(X, Y)}{\sigma_X \sigma_Y} = \frac{E[(x - \mu_X)(y - \mu_Y)]}{\sigma_X \sigma_Y},
\]

where \( \mu_X \) (resp., \( \mu_Y \)) and \( \sigma_X \) (resp., \( \sigma_Y \)) are the mean and standard deviation values of feature X (resp., feature Y). If \( \rho(X, Y) = 1 \), then X and Y have a linear correlation. If 0.7 ≤ \( \rho(X, Y) < 1 \), then X and Y have a strong linear correlation, if 0.5 ≤ \( \rho(X, Y) < 0.7 \), then X and Y have a modest linear correlation, and if 0 ≤ \( \rho(X, Y) < 0.5 \), then X and Y are said to have a weak linear correlation.

The Pearson correlation indicates to what extent variables show a linear relationship (correlation) among them. The correlation takes its values in the range from -1 to +1. The extreme value +1 (resp., -1) informs about a perfect...
direct/increasing linear relationship (resp., inverse/decreasing). Indeed, strong relationship between variables is reflected by values close to the limits (−1 ≤ ρ ≤ −0.9 or 0.9 ≤ ρ ≤ +1) [14]. Pearson’s correlation coefficient takes value 0 if we are in presence of independent variables. However, the reverse is not true since this coefficient deals only with figuring out linear dependencies between variables.

In our approach, we first use the training dataset F represented by n × d. F consists of n profile instances f_i such that i = 1, . . . , n, and each f_i = (f_{i1}, . . . , f_{id}). From F, we construct a correlation matrix Ω. The latter is a d × d matrix, where Ω_{ij} ∈ R and −1 ≤ Ω_{ij} ≤ +1:

$$\Omega = \left( \begin{array}{cccc} \Omega_{11} & \Omega_{12} & \cdots & \Omega_{1d} \\ \Omega_{21} & \Omega_{22} & \cdots & \Omega_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ \Omega_{d1} & \Omega_{d2} & \cdots & \Omega_{dd} \end{array} \right).$$  (2)

We consider the set of d feature vectors {f_1, . . . , f_d} such that F_i = (f_{iu}, . . . , f_{id}). For each pair of features (F_i, F_j), we compute Ω_{ij} = ρ(F_i, F_j). Then, we derive a weighted graph G = (V, E, w) from matrix Ω, defined as follows:

(i) V = {v_1, . . . , v_d}, the set of vertices (features) where |V| = d;
(ii) E = {(v_i, v_j), where Ω_{ij} ≠ 0} and |E| = m;
(iii) w(v_i, v_j) = w_{ij} = Ω_{ij}.

A subgraph G_{Th} = (V_{Th}, E_{Th}, w_{Th}) is then induced from the graph G, where 0 < Th ≤ 1, by removing all the edges (v_i, v_j) whose w_{ij} < Th. G_{Th} is defined as follows:

(i) E_{Th} = \{ (v_i, v_j), where w_{ij} ≥ Th \};
(ii) V_{Th} = \{ x ∈ V : ∃ y ∈ V and (x, y) ∈ E_{Th} \}, |V_{Th}| ≤ d;
(iii) w_{ij}^{Th} = w_{ij}.

The induced graph G_{Th} from G might be composed of a set of disjoint connected partitions. The more the Th is close to 1, the stronger the correlations exist in G_{Th}.

We aim at finding the set of features that increase and decrease altogether in order to avoid the missed detection problem as in [II]. The best way to do so is to extract from G the set of cliques composed of strongly correlated features. One of the widely adopted solutions [15] to compute maximal cliques in an arbitrary graph of d vertices runs in time O(3^d/2) = O(1.44^d). Instead of applying the maximal cliques algorithm on graph G, we propose to adopt the divide and conquer strategy by applying this algorithm on each connected component of the subgraph G_{Th}. A clique CL_i = (V_{i}, E_{i}, w_{i}) (i ≥ 1) of a graph G_{Th} is a set of vertices V_{i} ⊆ V_{Th} such that all the pairs of C_i are adjacent. This strategy significantly reduces the computational complexity to find maximal strongly correlated cliques. Let us consider that G_{Th} is composed of d vertices belonging to a set of M connected components. Each connected component P_i, i = 1 · · · M is composed of S_i vertices. There are α singleton vertices and β partitions with two vertices, and the rest of connected components are composed of more than two vertices. The computational complexity incurred by applying the maximal cliques algorithm on graph G is

$$1.44^d = 1.44^{(\alpha + 2\beta + 1)} = 1.44^{\alpha + 2\beta} \prod_{i=1}^{S_i} 1.44^{S_i}.$$  (3)

By applying the same algorithm on each connected partition of G_{Th}, we notice that there is no need to apply it on isolated vertices and the partitions of two vertices are cliques by definition, and hence we get the following computational complexity: Σ_i=1,S_i 1.44^{S_i}. It is obvious that applying the divide and conquer strategy can significantly reduce the running time of the algorithm and make it suitable for resource-constrained nodes.

Let φ be the set of edges belonging to all cliques in G_{Th}, and |φ| = d’. For each edge (F_i, F_j), which is the rth element of φ (r = 1, . . . , d’), we define a high-level feature H_r = F_i/F_j. From the training dataset F, we derive its high-level training dataset HF defined as follows; for each d-profile vector f_i ∈ F, we derive its d’-profile high-level vector g_r = (gt_{i1}, . . . , gt_{id’}), such that gt_{ik} = ft_{ik} for k ∈ [1, d] and ft_{ik} ≠ 0. If ft_{ik} = 0, the high-level vector g_r is then removed from the training dataset HF. This choice is justified by the fact that the stronger the correlation between F_i and F_k is, the more the data instances of (F_i, F_k) fall on the same straight line F_i = aF_k + b, where a is the slope and b is the intercept.

The high-level features belonging to the same clique CL_{Th} are grouped into a single vector ξ_i. We consider that K cliques are obtained from G_{Th}. Thus, the normal profile is then defined as the set of vectors ξ_i (i = 1, . . . , K). To further reduce the number of features in each vector ξ_i, we apply the maximum weighted spanning tree algorithm on each clique. To do so, we apply Kruskal’s algorithm originally used to obtain the minimum spanning tree by negating the weight of each edge [16]. The high-level features, whose edges do not belong to the tree, are removed from the normal profile. The resulted profile is called the minimum normal profile. The time complexity of the maximum weighted spanning tree is O(|E_{CL}| log |E_{CL}|), where E_{CL} is the number of edges in the clique. As |E_{CL}| = |V_{CL}|(|V_{CL}| − 1)/2, the time complexity becomes proportional to O(|E_{CL}| log |V_{CL}|). As the maximal cliques algorithm, the maximum weighted spanning tree is only applied on cliques with more than two vertices. The use of maximum weighted spanning tree is justified by the fact that all the low-level features of each clique in G_{Th} have strong correlation between them. In each clique, if X and Y are strongly correlated and Y and Z are strongly correlated, then X and Z are strongly correlated. Hence, we can remove the redundant (X, Z) edge from the clique.

To illustrate further the above method, we consider an example of seven network features, namely, F_1, F_2, F_3, F_4, F_5,
(1) Let $Z$ be the high-level test profile composed of $Z_l$ vectors ($l = 1, \ldots, K$).
(2) for All vectors $C_l$ such that $l = 1, \ldots, K$ do
(3) if $(\text{Dis}(Z_l, C_l) \notin \{\text{Low}, \text{Up}\})$ then
(4) return $Z$ is anomalous
(5) end if
(6) end for
(7) return $Z$ is normal;

Algorithm 1: Intrusion detection algorithm.

![Graph-based normal behavioral model](image)

Figure 2: Graph-based normal behavioral model.

$F_5'$ and $F_7$. The correlation coefficient matrices $\Omega$ between these features are

$$
\Omega = \begin{pmatrix}
F_1 & F_2 & F_3 & F_4 & F_5 & F_6 & F_7 \\
F_1 & 1 & 0.93 & 0.97 & 0.25 & 0.73 & 0.82 & 0.98 \\
F_2 & 0.93 & 1 & 0.99 & 0.81 & 0.54 & 0.62 & 0.94 \\
F_3 & 0.97 & 0.99 & 1 & 0.73 & 0.87 & 0.43 & 0.95 \\
F_4 & 0.25 & 0.81 & 0.73 & 1 & 0.98 & 0.52 & 0.71 \\
F_5 & 0.73 & 0.54 & 0.87 & 0.98 & 1 & 0.78 & 0.60 \\
F_6 & 0.82 & 0.62 & 0.43 & 0.52 & 0.78 & 1 & 0.53 \\
F_7 & 0.98 & 0.94 & 0.95 & 0.71 & 0.60 & 0.53 & 1 \\
\end{pmatrix}
$$

According to the correlation matrix, we generate the graph $G^{[\text{Th}]}$, where $\text{Th} > 0.9$ as shown in Figure 2(a). In the graph, there are two cliques: $\{F_1, F_2, F_3, F_5\}$ and $\{F_4, F_7\}$.

The network normal profile is defined as $\{(F_1/F_2, F_1/F_3, F_1/F_7, F_2/F_3, F_2/F_7, F_3/F_5, F_4/F_3)\}$. After applying the maximum weighted spanning tree algorithm, the edges $(F_1, F_2), (F_1, F_3), (F_1, F_7), (F_2, F_3), (F_2, F_7)$, and $(F_3, F_5)$ are removed, and the minimum normal profile becomes $\{(F_1/F_3, F_2/F_3, F_4/F_5)\}$.

**Proposition 1.** For any data set of $d$ low-level features, the number of high-level features induced by the graph-based generation method is upper-bounded by $d - K$, such that $K$ is the number of cliques in $G^{[\text{Th}]}$.

**Proof.** Consider $V^{[\text{Th}]} \subseteq V$; that is, in the worst case, each low-level feature belongs to a given clique $\text{Cl}_i^{[\text{Th}]}$ ($i \geq 1$). As a result, $\sum_{i=1}^{K} |V_i^{[\text{Th}]}| \leq d$. It is known that the number of edges induced by executing the maximum weighted spanning tree on the clique $\text{Cl}_i^{[\text{Th}]}$ is $h_i = |V_i^{[\text{Th}]}| - 1$. As $\sum_{i=1}^{K} (h_i + 1) \leq d$, $\sum_{i=1}^{K} h_i \leq d - K$. Thus, the number of edges (i.e., high-level features) induced by executing the maximum weighted spanning tree on all the cliques of $G^{[\text{Th}]}$ is upper-bounded by $d - K$. \hfill \Box

### 2.3. Detection Process

Each node constructs its local dataset represented by $n \times d$ matrix (i.e., $n$ vector instances and $d$ features). It then extracts $K$ cliques from this dataset, as shown above, as well as its minimum profile composed of $K$ vectors $\xi_l$ of size $m_l$, where $l = 1, \ldots, K$. The node computes the centroid vector $C_l$ for all the $n$ instances of $\xi_l$.

To check whether a profile $Z$ is normal or anomalous, we derive from $Z$ its corresponding high-level profile $HZ$ and we execute the pseudocode depicted in Algorithm 1. In the algorithm, Dis denotes the Euclidean distance between two vectors. Low$^l$ and Up$^l$ denote the lowest and highest values obtained from estimating $\text{Dis}(\xi_l, C_l)$ for all the $n$ instances of $\xi_l$.

### 2.4. Simulation Results

We study the performance of the proposed IDS using GloMoSim simulator [17]. Each node sends one packet/sec toward the sink. A watchdog is implemented at each node and its role is to monitor the network activities of all the node's neighbors. At every 10 seconds (i.e., one time period), a monitoring node $i$ measures the feature vector of its monitored node $j$. After a training phase of $T$ time periods, testing phase lasts for 1800 seconds. The role of IDS, which is implemented at a node $i$, is not just to detect if $i$’s neighbor (node $j$) is malicious or not but also to detect if node $j$ is malicious during a given time period. We evaluate the performance of the IDS using two metrics: detection rate and false positive rate. We select the following five quantitative features:

(i) number of generated packets (GEN),
(ii) number of received packets (RCV),
(iii) number of forwarded packets (FWD),
(iv) number of accepted packets (ACP),
(v) number of dropped packets (DP).
(iv) number of sent packets (SENT),
(v) number of lost packets (LOSS).

We generate then the correlation matrix $\Omega$ as well as the minimum normal profile after performing the maximal cliques algorithm and the maximum weighted spanning tree algorithm as shown in Figure 3:

\[
\begin{pmatrix}
\text{GEN} & \text{RCV} & \text{FWD} & \text{SENT} & \text{LOSS} \\
\text{GEN} & 1 & 0.4205 & 0.4205 & 0.7263 & 0.6032 \\
\text{RCV} & 0.4205 & 1 & 1 & 0.9289 & 0.9727 \\
\text{FWD} & 0.4205 & 1 & 1 & 0.9289 & 0.9727 \\
\text{SENT} & 0.7263 & 0.9289 & 0.9289 & 1 & 0.9828 \\
\text{LOSS} & 0.6032 & 0.9727 & 0.9727 & 0.9828 & 1 \\
\end{pmatrix}
\]

Figure 4 shows the detection rate of the proposed IDS as a function of dropping probability. The first observation that we can draw from the figure is that the detection rate is 100% when the dropping probability is higher than 0.05, and it is under 100% when the dropping probability is ≤0.02. This can be explained as follows: under very low dropping probabilities, the malicious nodes drop packets at low intensities and their activities become unnoticeable. This happens when the dropping probability becomes very close to or less than the normal packet loss, which is at most 2% during each time period. Figure 5 shows the detection rate of the IDS as a function of training period. The results are presented under the following levels of dropping probability $P = 1, 0.5, 0.1, 0.05, 0.01$. The results show that the detection rate does not depend on the training period but on the dropping probability. Under high dropping probabilities, the detection rate is 100% for all the training periods. Under low dropping probabilities, the detection rate decreases as the malicious behavior becomes very close to the normal one.

Figure 6 shows the false positive rate of IDS as a function of training period under the following levels of dropping probability $P = 0.8, 0.5, 0.1, 0.05, 0.03, 0.01$. We can notice...
that the false positive becomes 0 when the training period $T = 30$ for all $P > 0.02$. At $T = 30$, the IDS has learned all the possible instances of the normal profile and can accurately distinguish between normal and anomalous traffic. When $T < 30$, the IDS still has not learned all the instances of the normal profile. In other words, the normal profiles, which are not observed during the training phase, will be considered anomalous during the testing phase. Thus, the false positive rate depends in this case on the number of times unlearned normal profiles are observed during the testing phase, which itself depends on the number of lost packets that are due to (1) normal packet loss and (2) dropping activities. As packet loss is an event that occurs randomly, the false positive curves are also random when $T < 30$. For $P = 0.01$, the false positive becomes 0 only when $T = 40$. Given that the behavior of the malicious node becomes very close to the legitimate node, the IDS needs more time to learn about new instances of the normal profile.

3. Nature Dynamicity

3.1. Background: Constant Fading Reputation Strategy. Reputation is defined as the general opinion of a society of nodes towards a certain node in a specific domain of interest, and it is the global perception on the future behavior of this node. In the IDS based on multiple observations, the IDS collects a series of consecutive observations, each of which occurs during a separate monitoring period.

Since reputation aggregates past experiences and dynamically evolves, it is similar to Bayesian analysis, which is a statistical procedure that estimates parameters of an underlying distribution based on observations. Starting with prior distribution, which is the initial state before any observation is made, Bayesian analysis continuously takes into account new experiences and derives posterior probability [18]. One of the used distributions in Bayesian analysis is Beta distribution.

Beta distribution has been recognized as a useful formal tool to model reputation [18–20]. A reputation value assumes a tuple of $(\alpha, \beta \geq 1)$ such that $\alpha$ and $\beta$ represent positive and negative observations, respectively.

The Beta distribution and its probability density function (PDF) are defined as follows:

$$B(\alpha, \beta) = \int_0^1 t^{\alpha-1} (1-t)^{\beta-1} \, dt$$

$$f(p | \alpha, \beta) = \frac{1}{B(\alpha, \beta)} p^{\alpha-1} (1-p)^{\beta-1},$$

where $0 \leq p \leq 1$, $\alpha, \beta \geq 0$.

The reputation, denoted by $R$, is defined as the expectation (denoted by $E$) of the Beta distribution, and it takes the following simple form:

$$R = E(B(\alpha, \beta)) = \frac{\alpha}{\alpha + \beta}$$

We model the reputation of a node with a Beta distribution $(\alpha, \beta)$. Initially, $\alpha = 1$ and $\beta = 1$.

The standard Bayesian procedure is as follows. Initially, the prior is Beta(1,1), the uniform distribution on [0,1]. Then, when a new observation is made, say with $n$ observed misbehaviors and $p$ observed correct behaviors, the prior is updated according to $\alpha := \alpha + p$ and $\beta := \beta + n$. The reputation relies on the node’s direct observation. When the monitoring node makes one individual observation about the monitored node, it updates $\alpha$ and $\beta$ as follows.

(i) If the observation is qualified as misbehavior, $\beta$ is set to $\beta + 1$.

(ii) If the observation is qualified as correct behavior, $\alpha$ is set to $\alpha + 1$.

The standard Bayesian method is modified in [19] to give less weight to the observations received in the past so as to allow reputation fading and prevent any node from capitalizing on its previous good behavior forever. To achieve this aim, a discount factor for past observations is used. When a new observation $(p, n)$ is made, $\alpha$ and $\beta$ are updated as follows:

$$\alpha := \omega \alpha + p,$$

$$\beta := \omega \beta + n,$$

where $0 \leq \omega \leq 1$.

The weight $\omega$ is a constant discount factor for past observations, which serves as the fading mechanism. We refer hereafter to the reputation system described above as the constant fading reputation strategy.

3.2. Adaptive Fading Reputation Strategy. The constant fading reputation mechanism uses the same discount factor for all types of observations and during all the time. The higher (resp., lower) the value of $\omega$ is, the slower (resp., quicker) the histories are forgotten. By knowing the value of $\omega$, a malicious node can evade from IDS detection by misbehaving for a given time and goes back to normal behavior. Under high discount factor, the change of node behavior (from well-behaved to misbehaved and vice versa) will be detected after a long time. During this time, well-behaved nodes can count on their good histories and act maliciously. In addition, misbehaved nodes will have to wait a longer time to redeem themselves. On the other hand, a low discount factor permits a quicker detection redemption of nodes. However, it might raise false alarms especially when network faults and attacks both share the same failure symptoms. For instance, a misbehavior is detected if the observed node is not forwarding a packet. This rule is set to detect black hole and selective forwarding attacks. In addition, this rule is applied when packets are not forwarded due to collisions, which means that a well-behaved observed node might be falsely considered malicious.

To deal with this issue, we propose an adaptive fading reputation mechanism. This mechanism uses the carrot and stick strategy; that is, reward the well-behaved node and punish the misbehaved node. The adaptive mechanism uses two types of discount factors: one for past positive observations
and the second one for past negative observations. The value of the discount factors is adjusted as function of reputation $R$ as shown in Figure 7.

In the adaptive fading reputation mechanism, when a new observation $(p, n)$ is made, $\alpha$ and $\beta$ are updated as follows:

$$\alpha := \psi(R) \alpha + p,$$

$$\beta := \varphi(R) \beta + n,$$

where $0 \leq \psi(R)$, $\varphi(R) \leq 1$.

$\psi(R)$ and $\varphi(R)$ denote the discount factors for past positive and negative histories, respectively, whose values fall into the range of $[0, 1]$. According to the value of $R$, a reputation system executes the following two fading strategies.

(i) **Reward Strategy.** It is applied when the reputation $R \geq \text{th}$, such that $\text{th} \in [0, 1]$. The IDS forgets the negative history more quickly than the positive one (i.e., $\psi(R) > \varphi(R)$); this strategy is used when a node is well-behaved.

(ii) **Punishment Strategy.** It is applied when the reputation $R < \text{th}$. The IDS forgets the positive history more quickly than the negative one (i.e., $\psi(R) < \varphi(R)$); this strategy is used when a node is misbehaved.

Formally, $\psi(R)$ and $\varphi(R)$ are written as follows:

$$\psi(R) = \begin{cases} 
\left( \frac{\text{PR}_{\text{max}} - \text{PR}_{\text{min}}}{1-t} \right) R + \frac{\text{PR}_{\text{min}} - \text{PR}_{\text{max}} \times t}{1-t}, & \text{when } R \geq t, \\
\left( \frac{\text{PR}_{\text{max}} - \text{PR}_{\text{min}}}{t} \right) R + \text{PR}_{\text{min}}, & \text{when } R < t,
\end{cases}$$

$$\varphi(R) = \begin{cases} 
\left( \frac{\text{NP}_{\text{max}} + \text{NR}_{\text{min}}}{1-t} \right) R \left[ \frac{\text{NP}_{\text{min}} - \text{NP}_{\text{max}} \times t}{1-t} \right], & \text{when } R \geq t, \\
\left( \frac{\text{NP}_{\text{max}} + \text{PP}_{\text{min}}}{t} \right) R + \text{NP}_{\text{min}}, & \text{when } R < t,
\end{cases}$$

where $\text{PR}_{\text{max}}$ and $\text{PR}_{\text{min}}$ are the upper and the lower bounds of the positive discount factor, respectively, under reward strategy. $\text{NR}_{\text{max}}$ and $\text{NR}_{\text{min}}$ are the upper and the lower bounds of the negative discount factor, respectively, under punishment strategy. $\text{PP}_{\text{max}}$ and $\text{PP}_{\text{min}}$ are the upper and the lower bounds of the positive discount factor, respectively, under punishment strategy. $\text{NP}_{\text{max}}$ and $\text{NP}_{\text{min}}$ are the upper and the lower bounds of the negative discount factor, respectively, under punishment strategy.

For new nodes, positive and negative histories are kept with a discount factor equal to 1 when the number of observations is less than a given value named experience threshold.

From the above upper and lower bounds, we define the following two distance metrics.

(i) **Punish-to-Reward (PTR) Distance.** It is defined by $\text{PR}_{\text{min}} - \text{PP}_{\text{max}}$, and it shows to what extent the node is rewarded by the IDS when it transits from the misbehaved state to the well-behaved state; that is, the higher the PTR is, the slower the positive histories are forgotten.

(ii) **Reward-to-Punish (RTP) Distance.** It is defined by $\text{NP}_{\text{max}} - \text{NR}_{\text{min}}$, and it shows to what extent the node is punished by the IDS when it transits from the well-behaved state to the misbehaved state; that is, the higher the RTP is, the slower the negative histories are forgotten.

3.3. Performance of Adaptive Discount Factor Strategy. We evaluate the performance of the constant and adaptive discount factor strategies in terms of detection time. To do so, we implement three behavioral models.

(i) Deterministic redemption model: in this model, a node with reputation $R = 0$ behaves correctly in the network.

(ii) Deterministic evasion model: in this model, a node with reputation $R = 1$ behaves maliciously in the network.

(iii) Probabilistic evasion model: the node’s behavior is modeled with a two-state Markov chain as depicted in Figure 8. In state $N$, the node is well-behaved, and, in state $M$, the node is misbehaved. Initially, the node’s reputation $R = 1$. The node transits towards state $N$ with probability $P_N$ and towards state $M$ with probability $P_M$, such that $P_N + P_M = 1$. $P_M$ is called the evasion probability. The time spent in state $N$ and state $M$ is the monitoring time period.

The parameters for the experiment are shown in Table 2. We define three settings for the adaptive fading reputation.

(i) Setting 1: PTR and RTP are high; for example, they equal 0.7.
As for constant fading reputation, we define three levels of discount factor $\omega = 0.2, 0.5, 0.8$.

We study the evolution of reputation over time when applying constant and adaptive discount factor. In Figure 9(a), the convergence time increases as $\omega$ increases. This is because higher (resp., lower) values of $\omega$ mean that the negative histories are forgotten at slower (resp., faster) rate, which leads to longer (resp., shorter) time to converge to $R = 1$. In Figure 9(b), we observe that the deterministic redemption model under adaptive discount factor strategy requires less convergence time than the constant one. It ranges between 3 and 9 observations under setting 1 and setting 3, respectively. The reason for this is that a node under setting 1 is rewarded more generously as long as it is well-behaving; that is, its positive histories are forgotten slower than those of setting 2 and setting 3.

In Figure 10, we also notice that the malicious node that follows the deterministic evasion is detected more quickly when the adaptive discount factor strategy is applied. The time to converge to $R = 0$ is between 3 and 9 observations under the adaptive discount factor strategy and between 4 and 14 observations under the constant discount factor strategy. For instance, let $R = 0.1$ be the boundary between malicious behavior and normal behavior; the malicious node can evade IDS detection for a time required to collect only 3 observations if the IDS adopts the adaptive discount factor strategy under setting 3. Under the constant discount factor strategy and if $\omega = 0.8$, IDS can detect the malicious after a time period of 5 observations.

By knowing the required number of observations to detect a malicious node, the latter can adopt the probabilistic evasion model, which do discontinuous harm to the network to confuse the IDS and hence evade detection. Figures 11, 12, and 13 show that the adaptive discount factor strategy can quickly detect this type of behavior. In the figures, we consider that a node is malicious when $R = 0.1$. When the evasion probability $P_M = 0.5$, the adaptive strategy succeeds at detecting the malicious node after a time between 2 and 37 observations. On the other hand, the malicious node can evade the IDS adopting the constant strategy for a time of 751 observations when $\omega = 0.8$. This value decreases to 10 and 2 when $\omega = 0.5$ and $\omega = 0.2$, respectively. When $P_M = 0.2$, the detection time decreases to 40 and 27 under $\omega = 0.8$ and setting 3, respectively. When $P_M$ is between 0.7 and 0.9, the adaptive strategy (resp., constant strategy) achieves a detection time between 2 and 4 (resp., between 2 and 5) observations.

### 4. Spatiotemporal Dynamicity

A monitoring node $i$ can make at least one observation about a monitored node $j$ if the wireless link lasts for a duration higher than the monitoring period $\Delta$. The malicious node $j$,
which knows this fact, can move around in the network to create links with its neighbors of duration less than $\Delta$.

As shown in Figure 14, the nodes start operating at time $t_0$. A wireless link between the monitoring node $i$ and monitored node $j$ is created at time $t_1$ when node $j$ comes within the transmission range of node $i$. Node $i$ loses its link with node $j$ either (1) when node $j$ moves out of the transmission range of node $i$ at time $t_2$ or (2) when node $j$ runs out of its battery power at time $t_3$. Therefore, node $i$ estimates the link-node lifetime by the following equation: $\min(t_2-t_1, t_3-t_1)$. $(t_2-t_1)$ is the estimation of the link lifetime and $(t_3-t_1)$ is the residual node lifetime after node $j$ has been in existence for $(t_1-t_0)$ time units.

In this section, we statistically analyze the link-node distribution. Based on this analysis, we choose appropriate values for the monitoring period so that the mobile monitored node cannot evade IDS detection. We use the random waypoint mobility model, in which each mobile node randomly selects a location, within an area of 100 m $\times$ 100 m, with a random speed uniformly distributed between 0 and a certain maximum speed $V_{\text{max}}$; then it stays stationary during a pause time of 1 second before moving to a new random
location. In our analysis, we consider two different numbers of nodes (NN), that is, 10 and 20 nodes.

4.1. Link Lifetime Distribution. We obtain from our simulation the frequency of link durations and plot them into a histogram, as shown in Figures 15 and 16. The EasyFit software [21, 22] is used to measure the compatibility of a random sample with the theoretical probability distribution functions. As shown in the figures, the software approximates the simulation data to a Weibull distribution [23] with two parameters $\alpha = 1.031$ and $\beta = 28.74$ (resp., $\alpha = 1.029$ and $\beta = 32.85$) when $V_{\text{max}} = 20$ and NN = 10 (resp., NN = 20).

Weibull distribution has a PDF as shown in the following equation:

$$f(x; \alpha, \beta) = \frac{\alpha}{\beta} \left(\frac{x}{\beta}\right)^{\alpha-1} e^{-\left(x/\beta\right)^\alpha}.$$  \hspace{1cm} (11)

Based on the properties of the Weibull distribution, the mean (expected value) is

$$\text{Mean} = \beta \times \Gamma\left(\frac{\alpha + 1}{\alpha}\right).$$  \hspace{1cm} (12)
On the other hand, Samar and Wicker [24, 25] have described the expected link lifetime as a function of node velocity, say $v_1$, with the following equation:

$$F_{\text{link}}^{v_1} = \frac{R}{2(b-a)} \times \left( \int_{\pi}^{0} \log \left| \frac{b + \sqrt{b^2 - v_1^2 \sin^2 \phi}}{v_1 + v_1 \cos \phi} \right| d\phi - \int_{\pi}^{\phi_0} \log \left| \frac{a + \sqrt{a^2 - v_1^2 \sin^2 \phi}}{a - \sqrt{a^2 - v_1^2 \sin^2 \phi}} \right| d\phi \right),$$  \hspace{1cm} (13) $

where $R$ is the radius of the circle centered at the node. $v_1$ is uniformly distributed between $a$ and $b$, expressed in meters/second. $\phi$ is the direction of motion: $\phi_0 = \pi - \sin^{-1}(a/v_1)$.

Since (12) and (13) are both describing the expected value of the link lifetime, we can write

$$\frac{\beta \Gamma((\alpha + 1)/\alpha)}{\Gamma(\alpha + 1)} = \frac{R}{2(b-a)} \times \left( \int_{\pi}^{0} \log \left| \frac{b + \sqrt{b^2 - v_1^2 \sin^2 \phi}}{v_1 + v_1 \cos \phi} \right| d\phi - \int_{\pi}^{\phi_0} \log \left| \frac{a + \sqrt{a^2 - v_1^2 \sin^2 \phi}}{a - \sqrt{a^2 - v_1^2 \sin^2 \phi}} \right| d\phi \right).$$  \hspace{1cm} (14) $

We derive then $\beta$ as a function of velocity $v_1$ as follows:

$$\frac{\beta \Gamma((\alpha + 1)/\alpha)}{\Gamma(\alpha + 1)} = \frac{R}{2(b-a)} \times \left( \int_{\pi}^{0} \log \left| \frac{b + \sqrt{b^2 - v_1^2 \sin^2 \phi}}{v_1 + v_1 \cos \phi} \right| d\phi - \int_{\pi}^{\phi_0} \log \left| \frac{a + \sqrt{a^2 - v_1^2 \sin^2 \phi}}{a - \sqrt{a^2 - v_1^2 \sin^2 \phi}} \right| d\phi \right).$$  \hspace{1cm} (15) $

Simulations have been conducted to compare between the theoretical $\beta$ obtained from (15) and the Weibull approximative one obtained from simulations, as shown in Table 3. The results show that the Weibull distribution fits well simulation data.

### 4.2. Residual Node Lifetime Distribution

We assume that the node lifetime follows an exponential distribution with a
parameter $\lambda$. This distribution is similar to the one used to model “time to failure” in reliability engineering. We consider that $\lambda$ is the rate at which node’s battery is discharged. The probability density function is then

$$ f(t) = \begin{cases} 0, & \text{if } t < 0, \\ \lambda e^{-\lambda t}, & \text{if } t \geq 0. \end{cases} $$ (16)

The probability density function of the residual node lifetime for a node of age $\alpha$ is given by the following equation [26]:

$$ r_\alpha(t) = \frac{f(t + \alpha)}{1 - F(\alpha)} = \lambda e^{-\lambda t}, $$ (17)

where $F$ is the cumulative density function (CDF) of the exponential distribution. Thus, the residual node lifetime also follows an exponential distribution. The expected value for the random variable $X$ following an exponential distribution is

$$ E(X) = \frac{1}{\lambda}. $$ (18)

### 4.3. Link-Node Lifetime Distribution

Consider a random variable $Z$, where $Z = \min(X, Y)$. $X$ (resp., $Y$) is a random variable related to link lifetime (resp., residual node lifetime) following a Weibull distribution (resp., exponential distribution) with a joint cumulative distribution function $I_{XY}(x, y)$. Then, since $X$ and $Y$ are independent, we have

$$ P(Z > t) = P(\min(X, Y) > t) = P(X > t, Y > t). $$ (19)

Therefore,

$$ P(Z > t) = 1 - P(X \leq t) - P(Y \leq t) + P(X \leq t, Y \leq t). $$ (20)

Consequently, the cumulative distribution function (CDF) of $Z$ is

$$ H_Z(t) = 1 - P(Z > t) $$

$$ = P(X \leq t) + P(Y \leq t) - P(X \leq t, Y \leq t). $$ (21)

Thus,

$$ H_Z(t) = F_X(t) + G_Y(t) - I_{XY}(t, t). $$ (22)

The approximated density function for the combined variables $X$ and $Y$ is a Phased Bi-Weibull distribution [27], which has a PDF as shown in

$$ g(t) = \begin{cases} \frac{\alpha_1}{\beta_1} (\frac{t - \gamma_1}{\beta_1})^{\alpha_1-1} e^{-(t - \gamma_1)/\beta_1} & \text{if } \gamma_1 \leq t \leq \gamma_2, \\
\frac{\alpha_2}{\beta_2} (\frac{t - \gamma_2}{\beta_2})^{\alpha_2-1} e^{-(t - \gamma_2)/\beta_2} & \text{if } \gamma_2 < t < \infty, \end{cases} $$ (23)

EasyFit software [22] approximates the simulation data to the Phased Bi-Weibull distribution, as shown in Figure 17 (resp., Figure 18), with parameters $\alpha_1 = 0.87118$, $\beta_1 = 19.482$, $\gamma_1 = 0$, $\alpha_2 = 0.68969$, $\beta_2 = 31.875$, and $\gamma_2 = 3$ (resp., $\alpha_1 = 0.90481$, $\beta_1 = 22.976$, $\gamma_1 = 0$, $\alpha_2 = 0.71509$, $\beta_2 = 14.819$, and $\gamma_2 = 4$).

Figure 17: Link-node lifetime distribution under $NN = 10$ and $V_{\max} = 20$.

Figure 18: Link-node lifetime distribution under $NN = 20$ and $V_{\max} = 20$.

Remark 2 (see [28]). For real values $x, y \in \mathbb{R}$, $\min(x, y) = x + y - \max(x, y)$.

The result of this remark is extended to random variables by the following theorem.

**Theorem 3** (see [28]). Given two real-valued continuous random variables $X, Y \in \Omega \rightarrow \mathbb{R}$, then the expected value of the minimum of the two variables is

$$ E(\min(X, Y)) = E(X) + E(Y) - E(\max(X, Y)). $$

**Lemma 4** (see [28]). Given two real-valued continuous random variables $X, Y \in \Omega \rightarrow \mathbb{R}$, then the expected value of the maximum of the two variables is

$$ E(\max(X, Y)) = \int_{\gamma_1}^{\gamma_2} x f_X(x) F_Y(x) dx + \int_{\gamma_2}^{\gamma_1} y f_Y(y) F_X(y) dy. $$

Based on Theorem 3 and Lemma 4, the expected link-node lifetime is given by

$$ E(Z) = E(X) + E(Y) - E(\max(X, Y)), $$ (24)
where $E(X)$ is given in (12) and $E(Y)$ in (18). Figure 19 shows that the expected link-node lifetime resulted from simulation as a function of node velocity. The results show that the expected link-node lifetime decreases rapidly as its velocity is increased and it shows a significant decrease when $V_{\text{max}} \in [1, 5]$. The results also show that under higher network density the expected link-node lifetime becomes longer. The reason for this is that a node in this case shares links with larger number of neighbors, and consequently links with longer durations will be observed.

4.4. Monitoring Period Estimation. Based on the above statistical analysis, we propose a method to choose the appropriate value for the monitoring period. This method is low-cost and more appropriate for resource-constrained networks like sensor networks. We also propose another method that requires some communication cost and can be implemented on nodes with higher capabilities such as mobile sinks or mobile ad hoc networks and vehicular ad hoc networks.

4.4.1. Low-Cost Method. We assume that the monitoring node has no information about the monitored node’s velocity, position, or residual battery, and it wants to ensure that 1% of its links are observable; that is, they exist for a duration $>\Delta$. As the link-node lifetime follows a Phased Bi-Weibull distribution, the minimum value of $\Delta$, which ensures this requirement, is $t$ such that $P(Z \leq t) = 1/100$.

4.4.2. High-Cost Method. We assume that each node $i$ can estimate its remaining battery power $E_i$ and its rate of energy dissipation $E_{\text{Disip}}$, for every time period $\Delta$; an ultraconservative estimate of the residual node lifetime is derived as shown in the following equation:

$$\theta_i = \frac{E_i}{\max (E_{\text{Disip}})} (s).$$

Each node $i$ periodically broadcasts a beacon message containing its residual node lifetime $\theta_i$ and its position obtained from GPS. Upon receiving such a message from node $i$, node $j$ first calculates $d_{ij}$, that is, the distance separating it from its neighbor $i$. The relative velocity of node $i$ with respect to node $j$ is $\sqrt{v_i^2 + v_j^2 - 2v_i v_j \cos \theta}$, where $v_i$ and $v_j$ are node $i$’s and node $j$’s velocity, respectively. $\theta$ denotes the angle between vectors $\hat{v}_i$ and $\hat{v}_j$ in the Cartesian coordinate system. The relative velocity is maximum when $v_i = v_j = V_{\text{max}}$ and $\theta = 180^\circ$, and it equals then to $2V_{\text{max}}$. Node $j$ then calculates a conservative estimate of the residual link lifetime, that is, the minimum time for node $i$ to move out of the transmission range of node $j$. The residual link lifetime, $\xi_{ij}$, is given by the following equation, where $TR$ is the transmission range:

$$\xi_{ij} = \frac{TR - d_{ij}}{2V_{\text{max}}} (s).$$

After that, each node $j$ estimates the residual link-node lifetime given by

$$\chi_{ij} = \min (\theta_i, \xi_{ij}).$$

Therefore, the monitoring period required to observe the monitored node $i$ must be less than $\chi_{ij}$.

5. Conclusion

In this paper, we have proposed IDS solutions for three aspects of dynamicity in ad hoc and wireless sensor networks. The magnitude dynamicity aspect is solved by defining a normal profile based on the invariants derived from the normal node behavior. We have generated a dependency graph consisting of strongly correlated features, and we have derived the high-level features from the graph. The high-level features are obtained by applying the divide-and-conquer strategy on the maximal cliques algorithm and the maximum weighted spanning tree algorithm. Simulation results show that the IDS can achieve a detection rate of 100% when the malicious behavior is not similar to the normal one. In addition, it can also achieve a false positive rate of 0% when the duration of the training time exceeds a given value. To handle nature dynamicity aspect, we have adopted the carrot and stick strategy to prevent a malicious node from evading the IDS. To do so, we have proposed an adaptive reputation fading strategy to allow fast redemption and fast capture of malicious node. We have analytically studied link-node lifetime distribution and have shown that it can be approximated to the Phased Bi-Weibull distribution. Based on this analysis, we have proposed a low-cost method to estimate the minimum monitoring period required to observe the monitored node’s behavior. In addition, based on some topology information, we have proposed a high-cost method designed for network having nodes less constrained with resource limitations.
Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgment

The authors would like to extend their sincere appreciation to the Deanship of Scientific Research at King Saud University for funding this research through Research Group Project (RG no.1435-051).

References

[1] C. E. Perkins, Ad hoc Networking, Addison-Wesley Professional, Reading, Mass, USA, 2008.
[2] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," Computer Networks, vol. 38, no. 4, pp. 393–422, 2002.
[3] S. Al-Sultan, M. M. Al-Door, A. H. Al-Bayatti, and H. Zedan, "A comprehensive survey on vehicular Ad Hoc network," Journal of Network and Computer Applications, vol. 37, no. 1, pp. 380–392, 2014.
[4] D. Djenouri, L. Khelladi, and N. Badache, "A survey of security issues in mobile ad hoc and sensor networks," IEEE Communications Surveys and Tutorials, vol. 7, no. 4, pp. 2–28, 2005.
[5] S. Gillani, F. Shahzad, A. Qayyum, and R. Mehmoood, "A survey on security in vehicular ad hoc networks," in Communication Technologies for Vehicles, pp. 59–74, Springer, New York, NY, USA, 2013.
[6] P. Garcia-Teodoroa, J. Díaz-Verdejoa, G. Maciá-Fernández, and E. Vázquezb, "Anomaly detection in vehicular ad hoc networks," in Communication Technologies for Vehicles, pp. 59–74, Springer, New York, NY, USA, 2013.
[7] M. Xie, S. Han, B. Tian, and S. Parvin, "Anomaly detection in wireless sensor networks: a survey," Journal of Network and Computer Applications, vol. 34, no. 4, pp. 1302–1323, 2011.
[8] B. Sun, L. Osborne, Y. Xiao, and S. Guizani, "Intrusion detection techniques in mobile ad hoc and wireless sensor networks," IEEE Wireless Communications, vol. 14, no. 5, pp. 56–63, 2007.
[9] G. Li, J. He, and Y. Fu, "Group-based intrusion detection system in wireless sensor networks," Computer Communications, vol. 31, no. 18, pp. 4324–4332, 2008.
[10] Y. Zhang, N. Meratnia, and P.Havinga, "Outlier detection techniques for wireless sensor networks: a survey," IEEE Communications Surveys and Tutorials, vol. 12, no. 2, pp. 159–170, 2010.
[11] C. E. Loo, M. Y. Ng, C. Leckie, and M. Palaniswami, "Intrusion detection for routing attacks in sensor networks," International Journal of Distributed Sensor Networks, vol. 2, no. 4, pp. 313–332, 2006.
[12] F. Liu, X. Cheng, and D. Chen, "Insider attacker detection in wireless sensor networks," in Proceedings of the 26th IEEE International Conference on Computer Communications (INFOCOM ’07), pp. 1937–1945, May 2007.
[13] A. Stetsko, L. Folkman, and V. Matyáš, "Neighbor-based intrusion detection for wireless sensor networks," in Proceedings of the 6th International Conference on Wireless and Mobile Communications (ICWMC ’10), pp. 420–425, IEEE, September 2010.
[14] S. Dowdy, S. Wearden, and D. Chilko, Statistics for Research, John Wiley & Sons, New York, NY, USA, 3rd edition, 2004.
[15] E. Tomita, A. Tanaka, and H. Takahashi, "The worst-case time complexity for generating all maximal cliques and computing them," Theoretical Computer Science, vol. 363, no. 1, pp. 28–42, 2006.
[16] P. Sriram and S. Skiena, "Computational discrete mathematics: combinatorics and graph theory with mathematica," Computing Reviews, vol. 45, no. 12, pp. 775, 2004.
[17] X. Zeng, R. Bagrodia, and M. Gerla, "GloMoSim: a library for parallel simulation of large-scale wireless networks," in Proceedings of the 12th Workshop on Parallel and Distributed Simulation (PADS ’98), pp. 154–161, May 1998.
[18] J. Liu and V. Issarny, "Enhanced reputation mechanism for mobile ad hoc networks," in Proceedings of 2nd International Conference on Trust Management, pp. 48–62, Springer, New York, NY, USA, 2004.
[19] S. Buchegger and J.-Y. L. Boudec, “A robust reputation system for peer-to-peer and mobile ad hoc networks,” in Proceedings of the 2nd Workshop on the Economics of Peer-to-Peer Systems (P2PEcon ’04), Cambridge, Mass, USA, 2004.
[20] P. Michiardi and R. Molva, “Core: a collaborative reputation mechanism to enforce node cooperation in mobile ad hoc networks,” in Advanced Communications and Multimedia Security, pp. 107–121, Springer, New York, NY, USA, 2002.
[21] "Mathwave data analysis & simulation," http://www.mathwave.com/products/easyfit.html.
[22] K. Schittkowski, “EASY-FIT: a software system for data fitting in dynamical systems,” Structural and Multidisciplinary Optimization, vol. 23, no. 2, pp. 153–169, 2002.
[23] C. Forbes, M. Evans, N. Hastings, and B. Peacock, Statistical Distributions, John Wiley & Sons, 2011.
[24] P. Samar and S. B. Wicker, "On the behavior of communication links of a node in a multi-hop mobile environment," in Proceedings of the 5th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MoBiHoc ’04), pp. 145–156, ACM, May 2004.
[25] P. Samar and S. B. Wicker, "Link dynamics and protocol design in a multihop mobile environment," IEEE Transactions on Mobile Computing, vol. 5, no. 9, pp. 1156–1172, 2006.
[26] M. Gerharz, C. de Waal, M. Frank, and P. Martini, "Link stability in mobile wireless ad hoc networks," in Proceedings of the 27th Annual IEEE Conference on Local Computer Networks (LCN ’02), pp. 30–39, IEEE, 2002.
[27] E. Louzada-Neto and A. C. Davison, A note on bayesian analysis of the poly-weibull model, 1998.
[28] G. Lewellen, Expected maximum and minimum of real-valued continuous random variables, 2013, https://antimatroid.wordpress.com/2013/01/.