A distributed cross-layer intrusion detection system for *ad hoc* networks

Yu LIU*, Yang LI*, Hong MAN*

Abstract

Most existing intrusion detection systems (IDSs) for ad hoc networks are proposed for single layer detection. Although they may apply to other layers of network protocol stack, individual layers of data is still being analyzed separately. In addition, most have not been able to emphasize localization of attack source. In this paper, we propose an anomaly-based IDS that utilizes cross-layer features to detect attacks, and localizes attack sources within one-hop perimeter. Specifically, we suggest a compact feature set that incorporate intelligence from both MAC layer and network layer to profile normal behaviors of mobile nodes; we adapt a data mining anomaly detection technique from wired networks to ad hoc networks; and we develop a novel collaborative detection scheme that enables the IDS to correlate local and global alerts. We validate our work through ns-2 simulation experiments. Experimental results demonstrate the effectiveness of our method.

Key words: Radiocommunication, Ad hoc network, Security, Intrusion detection, Modeling, Performance evaluation.

SYSTÈME RÉPARTI INTER-COUCHES POUR DÉTECTER L'INTRUSION DANS LES RÉSEAUX AD HOC

Résumé

Le caractère dynamique, réparti et auto organisé des réseaux ad hoc présente un grand défi à la détection des intrusions. En général, le système de détection d'intrusion dans un réseau s'implémente à la périphérie. Cette solution ne peut pas s'appliquer aux réseaux ad hoc par manque d'une infrastructure pré-existante pour la communication et de centres de contrôle. Par ailleurs, les techniques courantes pour détecter l'intrusion, qui ont été développées pour les réseaux filaires et étendus, ne peuvent que s'appliquer aux couches individuelles dans le protocole de réseau. Dans cet article, nous présentons un système de détection d'intrusion fondé sur un nœud qui arrive à détecter l'origine d'une attaque et à la localiser à un saut de la périphérie. Nous présentons plus particulièrement un ensemble de dispositifs compacts qui associent les informations des couches MAC et réseau pour profiler le comportement des nœuds mobiles. Nous adaptons cette technique pour détecter les anomalies dans les réseaux filaires et ad hoc.

* Department of Electrical and Computer Engineering, Stevens Institute of Technology Hoboken, New Jersey 07030, USA ; {yliu, yli1, hman}@stevens.edu

Enfin, nous proposons un nouveau mécanisme de réponse à l'intrusion qui permet à un système de lier une alerte locale aux alertes globales collectées des environs. Nous validons notre travail par des expériences par simulation ns-2. Les résultats des expériences indiquent l'efficacité de notre méthode.

Mots clés: Radiocommunication, Réseau ad hoc, Sécurité, Détection intrusion, Modélisation, Évaluation de performance.

Contents

I. *Introduction*
II. *Related work*
III. *The intrusion detection model*
IV. *Performance evaluation*
V. *Conclusion*
References (38 ref.)

I. INTRODUCTION

An ad hoc network is an autonomous system of mobile nodes connected by wireless links. Associations between nodes are established when they are in the vicinity of each other. All mobile nodes agree to relay each other's packets, and function as routers that discover and maintain routes to other nodes in the network. While the self-organized nature of ad hoc networks provides convenient and flexible communication links for end users, they lack perimeter defense mechanisms which enable rogue mobile nodes to freely join them. Through these mobile nodes, attackers can mount attacks against different network layers to either comprise individual node(s) or degrade the performance of the entire ad hoc network.

Researchers have proposed a variety of security mechanisms for ad hoc networks, and most of them focus at individual layers of the network protocol stack (e.g., [1-6]). Within each layer, different defense (proactive or reactive) strategies can be applied. In this paper, we attempt to devise an effective IDS, a reactive and distributed defense strategy, for ad hoc networks.

Conventional intrusion detection techniques can be classified into signature-based or misuse detection and anomalybased detection. Signature detections are known for high detection rates with low false positives, however they are unable to detect novel attacks whose signatures are unknown. In addition, signature detection techniques may be inappropriate for ad hoc networks due to the difficulties of distributing and updating signatures of attacks. A study has also shown the mobility nature of ad hoc networks impacts the effectiveness of signature detections [7]. Anomaly detections are known for detecting novel attacks. In general, anomaly detection techniques depend on the characterization of user/system/network activities that are considered as normal behaviors. The differences among various anomaly detection techniques reside in the methods for constructing normal profiles. Basically, there are two approaches: profilebased detection and specificationbased detection.

Profile-based detection defines a profile of normal behaviors and classifies any deviation of the profile as an anomaly. The assumption of this type of detection is that attacks are events distinguishable from normal legitimate use of system resources. Although this type of

anomaly detectors are able to detect novel attacks, they are prune to high false positive rate [8], [34-37] due to the difficulty of clear segmentation between normal and abnormal activities and the use of insufficient or inadequate features to profile normal behaviors.

Specification-based detection defines a set of constraints that describe the correct operation of a program or protocol and monitors the execution of the program with respect to the defined constraints [33]. It provides high accurate detection rate, but only effective on those attacks whose behaviors can be modelled with the specification system. As a result, it may miss out some known attacks. For example, the specification detection model proposed in [8] cannot detect attacks caused by malformed packets, such as crashiis [9] (a malformed packet that causes Microsoft IIS server to crash). However, this can be easily detected using profile-based detection, because there is a clear distinction in behaviors between a server that is up or down.

In this paper, we extend a profile-based anomaly detection technique proposed on wired networks [10], and adapt it to ad hoc networks. Surely, the mobility nature of ad hoc networks also raises challenges to anomaly detection schemes. Therefore, instead of taking the existing approaches which generally gather information from individual layers, we propose a system that provides a more effective detection of attacks through the use of cross-layer information. In addition, we develop an algorithm that correlates local and global intrusion alerts to expedite the detection of attacks and the localization of malicious nodes.

The organization of this paper is the following. In Section II, we briefly describe the related work. In Section III, we present the model of our system and describe how anomaly detection technique is applied. In Section IV, we provide the simulation results and the performance evaluation. Finally, we conclude the paper in Section V.

II. RELATED WORK

Most of current works on IDS for ad hoc networks target at individual layers of the network protocol stack. Zhang and Lee [11] proposed the first distributed anomaly-based IDS, which can detect attacks at different layers, e.g. the network layer. In their system, local detection engine is built on a rule based classification algorithm RIPPER [12] and local response is activated when a node locally detects an anomaly or intrusion with high confidence. If the confidence is low, it will initiate a global intrusion detection procedure through a cooperative detection engine. Yi and Lee [13] extended their previous work on local anomaly detection and developed a cross-feature analysis to explore the correlations between each feature and all other features using classification decision tree induction algorithm C4.5 [14]. As indicated by Han *et al.* [15], both C4.5 rules and RIPPER do not work well when the number of distinguishing features is large. In addition, class labels are required in training data. This means attack data may be needed to train the classifiers. However, in anomalybase detection, priori knowledge on attacks should be avoided.

In [13], the authors departed from their original nodebased IDS architecture and proposed a cluster-based IDS model to preserve battery power. In this model, a cluster of neighboring mobile nodes can randomly and fairly elect a monitoring node, i.e. the clusterhead, for the entire neighborhood. Several other works (e.g., [16, 17]) also suggested the use of cluster-based IDS architecture. Nevertheless, we believe cluster-based IDS has some limitations in ad

hoc networks. Generally, there is no guarantee that the clusterhead is on the attack paths. If the attacker and the victim(s) reside within the same cluster, and the clusterhead is not on the attack path, extra communications are needed between the clusterhead and the members of the cluster in order to detect the attack. But this will reduce throughput and impose pressure on the limited bandwidth of ad hoc networks.

Tseng *et al.* [18] developed a specification-based IDS based on manually constructed security specifications corresponding to correct behavior of the ad hoc on-demand distance vector (AODV) routing protocol request-reply flows. The actual behavior of the AODV flows is then compared with these specifications, and any deviations are interpreted as intrusions. As we stated in the previous section, there is a tradeoff between the accuracy of intrusion detection and the complexity of a specification-based IDS.

The closest work to ours is the IDS model proposed in [11]. However, our system is very different in aspects of anomaly detection method, feature selection and intrusion response.

III. THE INTRUSION DETECTION MODEL

III.1. Assumptions

The mobile ad hoc network under our study has the following properties: The network is fully selforganized, which means there is

- The network is fully self-organized, which means there is no pre-existing infrastructure (e.g. central server). Thus, conventional intrusion detection techniques applied to wired network gateways or wireless local area network (WLAN) access points are not effective in these ad hoc networks.
- No pre-existing distributed trust model (e.g. central authority or centralized trusted third party) is deployed at the network layer. That is to say, any node is free to join the network.
- In the MAC layer, we consider CSMA/CA protocol with no other secure fairness access mechanisms. we consider one of the on-demand routing protocols, e.g. AODV, as the routing protocol. In the transport layer, no explicit congestion control method is utilized.
- All links are bi-directional. Because we assume the use of CSMA/CA at the MAC layer. A 4-way RTS/CTS/DATA/ACK handshake exchange is used for every data packet transmission except for broadcast packets. Bi-directional links also allow a route reply packet to be sent by reversing the route in the route request packet. Otherwise, a route reply may need a new route discovery process, which becomes very in efficient.

These assumptions, except the first one, could be relaxed if rendering some secure protocols or mechanisms. This would add extra layers of defense. Nonetheless, our work intends to detect attacks through malicious behaviors of mobile node(s).

III.2. Overview of Attacks

Conceptually, similar to WLAN and wired networks, attacks on ad hoc networks can be classified into passive attacks and active attacks. Passive attacks refer to eavesdropping on the network traffic, and they are difficult to detect by their very nature. Active attacks, on the other hand, are initiated by malicious user(s)/node(s), and they can be carried out against mobile node(s), or communication protocol and infrastructure at different layers.

Active attacks can be further categorized into different classes based on different criteria. In the subsequent paragraph, we attempt to provide a list of most commonly studied attacks at MAC layer and network layer.They are classified according to their consequences, which then are sub-categorized according to attack techniques.

1) Bandwidth consumption attacks:

- Flooding: It is a frequently used technique to over-load network and significantly reduce the available bandwidth for legitimate use. Basically, any type of packets can be used to implement such attack. For instance, attacker can send massive MAC layer control frames (e.g., request to send (RTS), clear to send (CTS), and acknowledge (ACK)) or data frames to one or a group of victim nodes; attacker can also mount attack on the network layer by sending considerable network layer control packets (e.g. route request (RtReq)) or data packets.
- Frequency jamming: Because of the open communication medium, a MAC-layer jammer could jam an RTS packet to prevent a node from accessing the channel. The direct result of this attack is denial of service (DoS) to the destination node. However, the attack may create more damage to channel bandwidth due to the cascade effect caused by the random backoff algorithm [19].
- Packet dropping: Similar to the frequency jamming attack, maliciously (or selfishly) drop packets may produce cascade effect caused by the random backoff algorithm. Researchers in [20] found that different dropping patterns can degrade TCP service to different levels, and selectively dropping a small number of packets can result in severe damage to TCP performance.

2) Node resource consumption attacks:

- Sleep deprivation: An attacker intentionally selects one neighboring node to relay spurious data. The intention of this attack is to drain battery power and computational power of the victim node.
- Exploiting bugs in software: Exploiting vulnerabilities in software on a mobile node can cause sever damages to the victim node, which may include CPU and memory consumption.

3) Information disclosure attacks:

- Blackhole: An attacker advertises falsified routing control information. For example, in an AODV routing protocol, attacker can broadcast itself of having the best path to any node. As a result, the attacker intercepts all packets being sent to any destination node. The attacker can then retrieve plaintext data in the header field and payload of the inner datagram in cleartext. By exploiting the weakness of encryption or its implementation, further information may be exposed to the attacker.
- Grayhole: This type of attack is a special case of blackhole attack, in which an attacker selectively drops data packets intended to some of the destination addresses.

4) Routing disruption attacks:

- Wormhole attack: A pair of nodes collude to launch attack by tunnelling messages from one end to the other, and these messages may be replayed at the other end. A well-placed wormhole can severely disrupt routing. Attacker could convince nodes who would normally be multiple hops from a base station that they are only one or two hops away via the wormhole [21].
- Routing protocol attack: An attacker targets against routing protocols by rushing routing control packets [4], poisoning routing table, injecting or replicating packets to the network layer. Note that routing disruptions can also increase network bandwidth consumption and decrease network throughput.

5) Other attacks:

- Byzantine (insider) attack: Compromised intermediate nodes collusively conduct attacks such as blackhole, packet dropping and create routing loops [22].
- Spoofing attack: An attacker creates misleading context to trick the victim node into making an inappropriate security-relevant decision. MAC address spoofing and IP address spoofing are the common types at the MAC and network layer.

Because of the scarcity of resources in ad hoc networks, we focus our attention to resource consumption (node and network) related attacks. Example attacks include flooding, malicious packet dropping, blackhole, deprivation, and various routing disruption attacks. We should point out that our IDS cannot detect spoofing attacks, in view of the fact that the attacker's networking behavior can be quite normal in this case. This type of attack can be detected using authentication techniques.

We propose an anomaly detection technique to detect these attacks. A fundamental challenge to an anomaly-based ad hoc IDS is how to distinct between normal behaviors and abnormal behaviors of a mobile node. We utilize a rule-based data mining technique to profile normal behaviors of a mobile node, along with a collaborative detection scheme that uses Bayesian network to mitigate the uncertainty between normal and abnormal boundaries, and hence increases the effectiveness of detection.

Another challenge to an IDS is the efficiency of detection. Conventionally, different IDSs are deployed at each individual layers of the network protocol stack. This could limit detection effectiveness and increase the cost of defense. Our approach of circumventing this setback is to incorporate features from crosslayer intelligence, and use them as a guide to collect audit data. Therefore, in our model, one set of audit data is able to detect attacks from both the MAC layer and network layer.

III.3. The Model Architecture

The proposed IDS employs a nodebased, distributed architecture. This is because ad hoc networks in general lack of central monitoring mechanism, thus centralized IDS is inappropriate. Figure 1 shows the conceptual model architecture of our IDS. In particular, it comprises of the following four components:

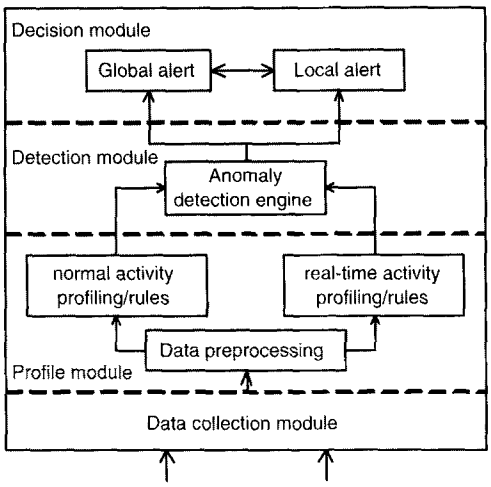

FIG. 1 – The proposed model architecture.

Le modèle d'architecture proposé.

1) Data Collection Module: Mobile nodes in ad hoc networks are usually thin clients owing to the power consumption limits, and they are unlikely to utilize rich system logging facilities, thus audit data from single nodes are unsuitable for applying anomaly detection techniques. In the proposed system, according to a prespecified feature set, audit data is collected from a given network within its observable radio transmission range. Although, open medium provides an easy ground to gather data from the network, real-time data collection still faces resource constraints. Besides competing for the limited energy in a mobile node, a large data set may reduce efficiency and accuracy of detection. Hence, selecting a minimal and near-optimal feature set is particularly important for ad hoc networks. Obviously, this is not an easy task. We discuss features of interests in the subsequent subsection.
2) Profile Module: This module has two subsystems. One subsystem is a pre-processor, where audit data is transformed into market basket format for profiling process. The other subsystem is a profiler, where a rule-based data mining technique is used to find association patterns from given data. Training data is collected in relative long time intervals. In our simulation, training data is collected in 1000 seconds and 2000 seconds intervals. Once collected, the training data is segmented into intervals of short time, which are usually the same as test data intervals. We choose 50 seconds as the duration of each data segment. Sliding windows of 50 seconds with 5 seconds overlap are used in the segmentation process. Overlapping can effectively capture patterns across the segmentation boundaries. Association rules extracted from each training data segment are pruned using maximal frequent itemset (MFI) [23], and then are aggregated into a rule set which is considered as a normal profile. In the aggregation process, each rule is recorded with a minimum and maximum support and confidence levels. During test phase, test data is collected at a 50 seconds interval window, also with 5 seconds overlap. The rules extracted from test data are then compared with the normal profile rule set.

3) Detection Module: Essentially, anomaly detection is to detect deviance from the norm. In this module, test data profiles are compared with the expected normal profiles. Any new rule or rule with deviations beyond the corresponding support and confidence threshold intervals $[minimum - \varepsilon, maximum + \varepsilon]$ is considered as an anomaly rule. Here ε denotes a tolerance level for classification errors.
4) Decision Module: Any defending node may trigger local alert based on the support and confidence levels of anomaly rules produced from the detection module. In order to reduce the number of false alarms, we use Bayesian network to incorporate the intelligence gathered from neighboring nodes with the local alerts detected from its own IDS, and make collaborative detection accordingly. If the probability of attack from a suspicious node deviates from a threshold, the victim node can send a global alert to its neighboring nodes.

III.4. Features of Interests

Various intrinsic features from the MAC layer and network layer are obtainable, however, some could be useless for intrusion detection. For instance, *duration* in a MAC frame contributes little for intrusion detection due to the dynamic nature of ad hoc networks. Our objective is to obtain a compact set of features.

We base our feature selection on the MAC layer, and add an additional feature *Packet Type* from the network layer. Similar to the MAC layer, there are various routing control packets, such as Route Request (RtReq), Route Reply (RtRpy), Route Error (RtErr), and routing Data packet (routingDataPkt). We combine all routing control packets into one category as routing Control packet (routingCtrlPkt). An example of feature set and its value space is illustrated in Table I, and an example audit data set using features specified in Table I is shown in Table II.

TABLE I. – Features of interests and their value space.

Caractères intéressants et leurs espaces de valeurs.

Dimension	Value Space
Flow direction	SEND, RECV, DROP
Send_node	sa_i, $\forall_i \in$ node set S
Recv_node	da_j, $\forall_j \in$ node set S
MACPktType	RTS, CTS, DATA, ACK
RoutingPktType*	routingDataPkt, routingCtrlPkt

* This feature applies to MAC DATA packets only.

In [11], multi-layer integrated intrusion detection and response has been discussed. In their approach, detection or weak evidences on one layer may activate the intrusion detection module on another layer to further investigate the possible attacks. However, the intrusion detection module at each layer still functions individually.

TABLE II. – An example DATA SET using features specified in table I.

Exemple de données avec les caractères spécifiés dans le tableau I

Timestamp	Flow dir	Send addr	Recv addr	MACPktType	RoutingPktType
50.0469	RECV	7	16	RTS	
50.0472	RECV	1000*	7	CTS	
50.0519	RECV	7	16	DATA	routingDataPkt
50.0522	RECV	16	7	ACK	
50.0529	RECV	16	9	RTS	
50.0532	RECV	1000	16	CTS	
50.0579	RECV	16	9	DATA	routingDataPkt
...					
50.1350	SEND	22	7	RTS	
50.1400	SEND	22	7	ACK	
...					
52.3793	RECV	21	1**	DATA	routingCtrlPkt
52.3793	DROP	21	1	DATA	routingCtrlPkt

* Refers to a non-exist address, since CTS packet does not require send _ddr **Refers to a broadcasting packet.

III.5. Anomaly detection

In the subsequent paragraphs, we provide a brief overview of association analysis.

Association rule describes associations of features (attributes) within transaction records of an audit data set. Given a set of n transaction records denoted as $\mathbf{T} = \{T_1, T_2, ..., T_n\}$, and a set $\mathbf{F} = \{F^1, F^2, ..., F^k\}$ of k features defined over $\mathbf{T}$, a transaction record is a collection of k-tuple items whose value assignments corresponding to the k features, we denote it as $T_i = \{f^1, f^2, ..., f^k\}$, where f^k represents a value from the k-th feature F^k. Let A and B denote two disjointed item subset in T_i. Let the support of A, denoted by $sup(A)$, represents the percentage of transactions containing A in T, and the support of both A and B, denoted by $sup(A \cup B)$. An association rule is [24]

$$A \rightarrow B, (s, c), \tag{1}$$

where $s = sup(A \cup B)$ is the support value of the rule, and $c = \dfrac{sup(A \cup B)}{sup(A)}$ is the confidence. The rule holds if $s \geq minsup$, and $c \geq minconf$, where *minsup* and *minconf* denote predefined minimum support threshold and confidence threshold, respectively.

As shown in Table II, each transaction record is a packet-level event. An example association rule is *(sa7, routingDataPkt → da16, RECV), (0.2, 1)*. This rule describes an event pattern related to the RECV flows of the monitoring node (e.g. node 22). That is, 20% of transaction records matches the pattern of "node 7 sends data packets to node 16", and when the node 16 receives data packets, they are 100% of the time from the node 7.

Common strategies of finding association patterns between different features of a transaction record involve two steps: frequent itemset generation and association rule generation. The first step is considered computationintensive. Its complexity and efficiency depend on the traverse method of finding the frequent itemsets. A priori algorithm [24] is known for

using breadth-first method, where all frequent itemsets of size-1 within the data set are discovered first, followed by all frequent itemsets of size-2, and so on. The limitation of using Apriori for our intrusion detection engine is that large number of frequent itemsets (and hence association rules) may be produced from a data set due to the large number of packet-level transactions in the MAC layer and network layer.

An alternative to a priori is to traverse itemsets using depthfirst method, in which if an itemset of size-1, say $\{A\}$, is frequent, the next step is to search a frequent itemset of size-2, of which one of the two items is $\{A\}$, for example $\{AB\}$, and the search continues with $\{ABC\}$ until it reaches an infrequent itemset. Then the search backtracks to another itemset, and so on. The depth-first method is often used to find maximal frequent itemsets. A maximal frequent itemset (MFI) is defined as a frequent itemset for which none of its immediate supersets is frequent.

Generating association rules pertaining to MFIs dramatically reduces the size of normal rulebased profile, yet it can still capture frequent association patterns from a data set. Figure 2 illustrates a comparison of number of association rules generated from Apriori and MFI algorithms. The data set has 1000 seconds time interval with *RECV* flow only and *recv_addr* matches with the monitoring node itself. The data is segmented into 21 intervals by using a 50 seconds sliding window with an overlap of 5 seconds. The rules are extracted from each segment using support threshold of 0.1 and confidence threshold of 0.6, and then aggregate into a rule set.

Once association rules are extracted and aggregated from a given training data set, they are then considered as the basis for behavior profiles.

III.6. Intrusion Response

Here intrusion response refers to associating anomalies with alerts. In particular, a detecting node can send a global alert to its neighboring nodes when it detects anomaly rules with

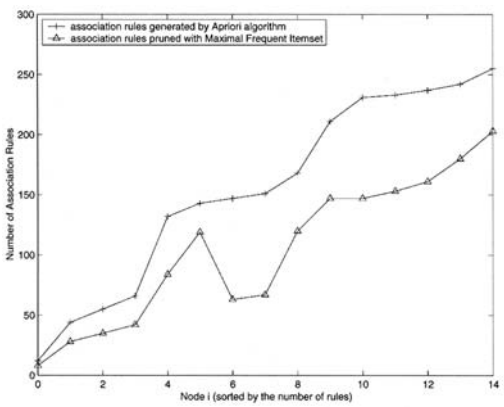

FIG. 2 – A comparison of the number of rules generated from *a priori* and MFI.

Comparaison du nombre de règles engendrées par a priori *et* MFI.

high support and confidence levels. On the other hand, when support and confidence levels are low, the detecting node can make collaborative decision by gathering intelligence (global alerts) from its neighboring nodes. We use Bayesian network to correlate the local and global alerts in the decision module of the proposed IDS, and make collaborative decision accordingly.

Bayesian network is a powerful and popular tool for probabilistic inference from observations [25, 26]. Recently it has been applied to network intrusion detection and vulnerability analysis [27-30]. A Bayesian network is defined by a directed acyclic graph (DAG) over nodes representing random variables and arcs signifying conditional dependencies between pairs of nodes. In our model, we define a Bayesian network with a set $\mathbf{X} = \{X_1, \ldots, X_n\}$ of variables that represents a monitoring node and a set of neighboring nodes. Each variable X_i takes on a binary value, where a true state corresponds to "being attacked" (for monitoring node) or "attacking" (for neighboring node), and a false state corresponds to the opposite. Let S be a graph that encodes the conditional relationship between variables in $\mathbf{X}$, and $\mathbf{P}$ is a set of local probability distributions associated with each variable. The posterior distribution with respect to S is

$$P(H \mid O) \propto P(O \mid H) \times P(H), \tag{2}$$

where H denotes a set of subjective beliefs that we are interested in, and $P(H)$ is the set of prior probabilities on S. $O = \{X_1 = x_1, \ldots, X_n = x_n\}$ denotes a set of observations (evidences) on $\mathbf{X}$. $P(O \mid H)$ is called marginal likelihood of O. Here, subjective beliefs in our model can be one or a set of attacking nodes (that are suspicious or interesting to evaluate).

Here we briefly illustrate our collaborative detection scheme through an example network. As shown in Figure 3, A and E are not in the vicinity of each other, but A adjoins three neighboring nodes B, C, and D, E is neighboring with D. The attacker A stages a flooding attack by sending spurious data packets against B, C and D. Suppose D learns an anomaly rule: *(D, routingDataPkt → A, RECV), (0.4, 1)* from its local detection module, in which 0.4 is the support value of this rule. B and C likewise may learn the similar rule with different

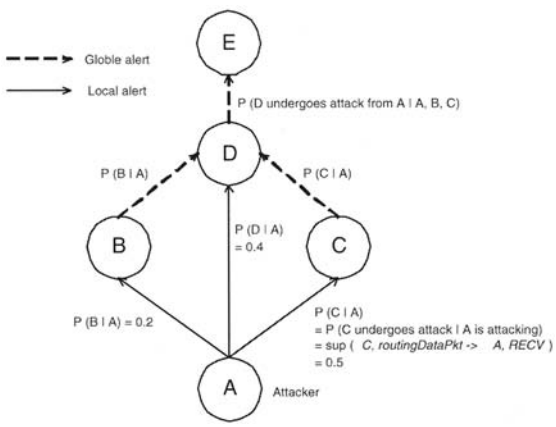

FIG. 3 – Threat alert aggregation and intrusion response at node D.

Agrégation des alertes de menaces et réponse à l'intrusion au nœud D.

support values, e.g., 0.2 and 0.5 respectively. Because the support value of an anomaly rule indicates the frequency of occurrence of this anomaly within a transaction record T, it is a nature indicator of the possibility that this anomaly (or attack) was intentionally staged by the source. We then take these support values as the marginal probabilities in a Bayesian network, e.g. $P(B \mid A)$, $P(C \mid A)$, and $P(D \mid A)$. We call such marginal probability as direct local alert because they are regarding nodes within one hop range. These local alerts can be further propagated to other nodes, and we call these forwarded local alerts as global alerts.

Within the decision module, each node maintains an intrusion response matrix M. An example matrix at node D is illustrated in Table III. The first column records the attacker identification; the second column records direct local alert that computed from its own detection engine; the third column records neighboring alert(s), each in turn is a pair of (s, p), where s represents the MAC address of the neighboring node who sends the global alert, and p is the global alert (a probability) from this node; finally, the 4th column records the updated local alert value by computing the probability of union set of the local alert and the incoming global alert(s). That is,

$$p(x_i = 1 \mid \mathbf{pa}_i) = 1 - \prod_j (1 - p(x_i = 1 \mid x_j)), \tag{3}$$

where X_i denotes a detecting node and $\mathbf{pa}_i$ represents the set of corresponding parent nodes, $j \in \mathbf{pa}_i$, and $p(x_i = 1 \mid x_j)$ is a local or global alert. The nodes in the above Bayesian network are similar to the noisy-OR nodes in [25].

TABLE III. – Intrusion response matrix at node D in figure 3.

Matrice de réponses à l'intrusion au nœud D dans la figure 3.

attackFrom	directLocal	globalAlert	updateLocal
A	0.4	0	0.4
A	0.4	(B, 0.2)	0.52
A	0.4	(B, 0.2), (C, 0.5)	0.72
...			

Suppose at node D, its direct local alert for attacker A is 0.4, and it also receives two global alert: $(B, 0.2)$ and $(C, 0.5)$, then D can update its local alert $P(D \mid A, B, C)$ by equation (3).

In addition to the four column fields in M, an extra column field, totalTimeLive (TTL), is necessary to keep the information in M up-to-date.

In order to avoid massive global alerts, we piggyback a global alert with a CTS packet, and the global alert could either be the value in the directLocal column or updateLocal column depending on whether the requesting node (i.e. the sending node of RTS packet) is in the neighborAlert column. The collaborative detection algorithm is described as in Algorithm 1.

Algorithm 1 Collaborative Detection Algorithm

```
/* receive global alert */
if receive CTS with a global alert (attacker_id = a, send_addr = s, p(s|a) = p) then
  if attackFrom[a] ≠ NULL then
    if s is in neighborAlert[a] then
      discard the global alert
    else
      add (s, p) to neighborAlert[a]
    end if
  else
    add a new entry (a, s, p) in the intrusion response matrix M
  end if
end if
/* update local alert */
for each a in M do
  if attackFrom[a] ≠ NULL then
    if neighborAlert[a] = ∅ then
      updataLocal[a] = directLocal[a]
    else
      compute updataLocal[a] using equation (3)
    end if
  end if
end for
/* send global alert */
for each a in M do
  globalAlert[a] = 0
end for
if receive RTS from node s then
  for each a in M do
    if s is in neighborAlert[a] then
      globalAlert[a] = directLocal[a]
    else
      globalAlert[a] = updateAlert[a]
    end if
  end for
end if
```

Once an attacker carries out an attack, several neighboring nodes of the attacker may detect anomalies about the attacker. From Algorithm 1, we see that the aggregated intrusion alert becomes more vivid and can quickly converge at all the neighboring nodes of the attacker. Therefore, the attacker can be identified within one-hop perimeter.

Bayesian network is also being utilized to evaluate multiple attack sources. Suppose, in Figure 3, node D detects anomalies from A, B, and C simultaneously, and $P(D \mid A)$, $P(D \mid B)$, and $P(D \mid C)$ are known from the anomalies rules.

The trick is that only A is the true attacker. Both B and C unconsciously forward packets from A to D. In such a scenario, we would be interested in computing the posterior probabilities $P(A, B, C \mid D = 1)$. One common difficulty is to assess the prior probabilities of A, B and C, i.e. P(A), P(B) and P(C). The detecting node D usually needs to apply domain knowledge about the attackers in the assessment of priors. Fortunately, as each node maintains an intrusion response matrix, which contains a list of attacker profile. The priors can be estimated from this list if it keeps sufficient history records of most neighboring nodes. Each node can then use a threat matrix to estimate prior probabilities of maliciousness of neighbor nodes. Within the threat matrix, each row can represent an alert level, and each column can represent a level of confidence about an attacker according to its duration of stay in the intrusion response matrix. The categorical levels of alert and confidence can be quantized into probabilities (e.g. high = 0.9, medium = 0.5, low = 0.3).

Now assume that the prior probability distributions of A, B and C can be estimated as previously described, then for the aforesaid example, we would expect $P(A = 1, B = 0, C = 0 \mid D = 1)$ is the largest posterior probability from all 8 combinations of A, B, and C, i.e., when D undergoes attack, it is likely that A is the attacking node. (For details on how to compute posterior probabilities, please see [26] which provides a collection of papers on Bayesian network inference and learning.)

IV. PERFORMANCE EVALUATION

IV.1. Simulation Environment

The simulation is conducted on the platform of Network Simulator ns-2 [38]. All the attacks discussed in this work are implemented on ns-2. The simulation purpose is to evaluate the performance of the proposed IDS over several attacks.

In the simulated network, there are 30 nodes with a fixed number of traffic flows. The source/destination pairs are randomly selected from the entire node set. For each flow, the transmission rate is 2 packets per second with a packet size of 512 bytes. A fixed 64-packet send buffer is maintained at each node for the packets waiting for available routes.

An important property of a mobile ad hoc network is the dynamic network topology. Since every node can move arbitrarily, the network topology changes from time to time, and the communication links between mobile nodes break frequently. To simulate the node movements, we assume a random waypoint mobility model [38] in a rectangular field with a dimension of 500×500 square meters. 30 mobile nodes and their initial locations are randomly assigned at the beginning of the simulation. During the simulation, each node randomly selects a destination in the field and move to that destination at a speed that is randomly selected from the range [0, *maxspeed*]. When the destination is reached, another destination location is chosen after a certain pause time. By adjusting the variables of *max-*

speed and pause time, the dynamic of the network topology can be adjusted to generate different mobility scenarios. To prevent all flows start at the same time, each source node chooses a random start time of sending packets from the range of [0, *stime*], where *stime* is set to 10.

We simulate two sets of data on two different network setups, one has the *maxspeed* of 5 m/s with a total of 15 traffic flows, and the other has the *maxspeed* of 10 m/s with a total of 25 traffic flows. In each network setup, five different pause times are selected. For the 5 m/s network, we simulate a normal training data of 1 000 seconds for each of five different pause times, and for the 10 m/s network, we simulate a 2 000 seconds for each of five different pause times. The pause times are 0, 10, 30, 60, 1 000/2 000 seconds for the two network setups, where 0 indicates the max mobility, i.e. all nodes are always moving, and 1000/2000 indicates a static network for the 5 m/s network and the 10 m/s network respectively. For test data, we simulate 10 test data of 100 seconds for each of five different pause times in the 5 m/s network, and we simulate 10 test data of 200 seconds for each of five different pause times in the 10 m/s network. The pause times are pause time of 0, 10, 30, 60 and 100/200 seconds, where 0 indicates an alwaysmoving network, and 100/200 indicates a static network for the 5 m/s network and 10 m/s network respectively. We remove the initial 50 seconds from all simulated data in consideration of the initialization process in ns-2 simulations.

IV.2. Simulated Attacks

- Flooding attack: Flooding attacks may be classified according to network layers, e.g., the MAC layer and network layer flooding. We can also categorize them from the perspective of routing schemes, e.g., single-path and multi-path flooding. We simulate a network layer, singlepath flooding with one attacker node and one victim node. If the victim is not in the vicinity of the attacker, the spurious data packets generated by the attacker may be delivered through two to three hops, and they may also take various paths.
- Blackhole attack: In this attack simulation, an attacker advertises as having the best (e.g. shortest) path to any node in the network. After the neighboring nodes receive the advertisement, they update their routing tables and redirect all packets to the attacker. Once the attacker intercepts the data packets, it either forwards the packets according to the destination or drops all of the packets.
- Sleep deprivation attack: In this attack simulation, an attacker advertises spurious routing control information about one of its neighboring node, i.e. the victim node. For instance, the attacker tell everyone that the victim node has the best path to any destination node. As a result, the victim node suffers sleep deprivation attack.
- Packet dropping attack: We simulate this attack with two dropping patterns. One is that an attacker drops all data packets passing through it; the other is that an attacker selectively drops data packets passing through it. In both cases, the attacker continues to respond to its neighbors' RTS packets to show its existence.

IV.3. Data Preprocessing and Feature Selection

Without a shred of doubt, the efficiency and accuracy of instruction detection is greatly influenced by the quality of data preprocessing. In our work, we apply domain knowledge to collect data packets from an ad hoc network according to a prespecified feature set, their flow direction and destination node address. Specifically, we preprocess data with the following two steps: 1. each node i collects only receiving flow packets, i.e. every transaction record collected should match the pattern ($flow\_dir = RECV$); 2. each node i collects packets destined to itself, i.e. every transaction record collected should match the pattern ($flow\_dir = RECV$, $recv\_addr = da_i$). Note that step 2 is an extended pruning phase of step 1.

The logic behind the above two steps is based on where a victim node stands relatively to an attacking node. If the victim node does not stand in (or behind) the attacking path, e.g. blackhole attack, the data set produced from step 1 can capture such attack through malicious behavior pattern of a neighbor attacker. If the victim node is a destination node of the attack or is on the path toward to a destination node, e.g. flooding attack, the data set produced from step 2 is able to detect such attack with high accuracy since irrelevant data items are pruned before pattern association analysis. In addition, we also remove the highly frequent items such as *RECV* in each data set produced from step 1 and 2, in order to reduce the unnecessarily large number of association rules related to them.

We observe that the total number of transaction records in step 2 has large variations compared with the one from step 1. This is due to the bursty behavior of the sending nodes in the network. Because the support value of an association rule is directly influenced by the total number of transaction records in each training or test data interval, a data set with a small number of transactions is likely to produce rules with large support values. When these support values are used in a global alert, they may mislead the neighboring nodes, since we associate them with marginal probabilities in Bayesian network of the decision module. Hence, when using data set from step 2 as input, the detection module is activated only if the total number of transactions in the data set from step 2 is reached a certain percentage (e.g. 5%) of the total number of transactions in the data set from step 1. This condition largely reduces the false alarm rate.

IV.4. Detection of the Simulated Attacks

In this subsection, we show our experimental results over the simulated attacks. Here, detection (i.e. true positive) rate is defined as the ratio of the number of attacks being detected correctly to the total number of attacks occurred during a particular time frame. If an attack takes multiple hops, then every intermediate node involved also consider itself being under attack. False alarm (i.e. false positive) rate is defined as the ratio of the number of attack-free events falsely being identified as anomalies (and raise local alerts) to the total number of normal events. Because most resource-consumption attacks occur over a time period, we use a sliding window (e.g. 50s) of data intervals to determine whether an attack takes place. Similarly, false positive rate is determined as the ratio of the number of misclassified abnormal data segments to the total number of data segments being tested.

The main difference of our method with the commonly used ones such as in [31, 32] is that we do not use the rules from the normal profile to directly classify each individual test event (or record). This requires considerable amount of processing time and computation power, which seems not practical for ad hoc networks. Instead, we test the normal rules against the rules produced from each test data segment, and then the test data segment is classified as normal or abnormal. Anomaly rules are used to further identify the type of attacks and attack source(s).

To simplify the simulations, we calculate detection rate and false alarm rate (during testing) using the statistics from the entire simulation network. That is, when an attack is carried out in the network, the victim nodes are used to verify the detection rate, meanwhile these nodes together with all the rest of nodes are used to verify the false alarm rate.

Table IV shows the experiment results of all simulated attacks, i.e. flooding, blackhole, sleep deprivation and packet dropping ALL data packets) on networks with *maxspeed* = 5 and *maxspeed* = 10, respectively. The detection rate and false alarm rate is the average value for 5 different mobility levels as described in Section IV-A. From the results shown in Table IV, we can see that mobility has a significant effect on detection rate. When mobility increases, detection rate in general will decrease. We should point out that we are unable to detect selective packet dropping attack if the dropping rate is relatively low. In such situation, there is no clear separation between normal behavior and abnormal behavior.

All the simulated attacks require one-hop detection except the flooding attack, where one or more hops may be involved during the attack. We can identify the attack on the majority of the hops involved. An example is shown in Figure 4. This means if the one-hop victims take countermeasures, the damage of the attack could be confined within one-hop perimeter.

The selection of support and confidence threshold values can greatly affect detection rate and false alarm rate. However, to determine a best minimum support and minimum confidence is a common challenge to association rule mining for classification. Many researchers have proposed methods to tune thresholds to improve the accuracy of classification, and most of them rely on iterative approach [31, 32]. In our work, we select these thresholds through a heuristic approach. Because our IDS is an anomaly-based IDS, we do not use any

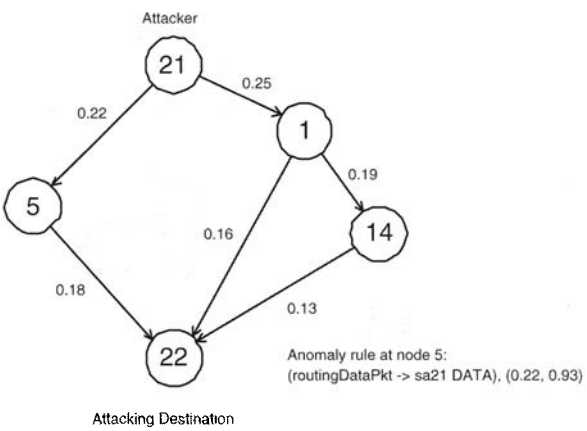

FIG. 4 – Detection of multihop flooding attack.

Détection d'attaques par inondation multi-saut.

simulated attack segments to tune these thresholds. Instead, only normal training data segments are used. Our merely assumption is that normal training data is available (i.e. no attack occurs during the normal data collection intervals). Specifically, we use false positive rate to determine the thresholds as follows.

First, we set the false positive rate to a reasonable low rate, e.g. 1%. Then, we select a threshold pair, e.g. $s = 0.01$ and $c = 0.06$ to mine each data segment (e.g. 50 seconds of interval) of both normal training data sets and normal test data sets. Rules of training data segments are aggregated into a normal profile. Rules of each test data segment are compared with the normal profile and abnormal segment is determined by whether it contains anomaly rules. The abnormal data segments are used to calculate the false positive rates. If it's higher than the target goal, we increase the support value by 10%. If there are no false positives at all, we decrease the support value by 10% to make sure that no other lower thresholds could reach the same false positive. After both lower bound and higher bound of the threshold interval are determined, we continue by selecting the middle value of the interval until reaching the desired false positive rate. We consider this false positive rate as training errors, and false positive anomaly rules are added into the normal profile.

Table V illustrates example anomaly rules for detecting the corresponding simulated attacks. In the table, node 15 is the misbehaving node. In flooding attack, all three rules indicate node 15 is sending large amount of data packets to the monitoring node. In blackhole attack, the first two rules indicate that there are suspiciously large volume of data packets destined to *da*15. The third rule suggests that data packets from several neighboring nodes are destined to the attacker *da*15. In sleep deprivation attack, because of the incorrect route advertisement by the attacker, the victim node (i.e. node 15) may not have valid route to some of the destination nodes. This causes the victim node broadcasting Route Error information, as indicated by the first rule. Node 15 is also a forwarding node, hence rules related to both *sa*15 and *da*15 are expected. It should be noted that if the attacker drops all of the intercepted data packets, the first rule is not expected. Instead, only the second and third rules are accountable to detect such attack. In packet dropping (ALL) attack, the monitoring node can observe whether its neighboring node (e.g., node 15) is forwarding its data packets by examining whether the expecting rule is learned.

TABLE IV. – Experiment results for the simulated attacks.

Résultats expérimentaux pour les attaques simulées.

Attack Type	Detection Rate 5m/s	False Alarm Rate 5m/s	Detection Rate 10m/s	False Alarm Rate 10m/s
Flooding	100%	2.78%	91.78%	0.25%
Blackhole	99.3%	0.3%	71.34%	0.4%
Sleep Deprivation	90%	0.7%	40.6%	0.31%
Packet Dropping (ALL)	93%	0.5%	77.56%	0

TABLE V. – Example of anomaly rules.

Exemple de règles d'anomalie.

Attack Type	Anomaly Rule Samples
Flooding	(routingDataPkt → sa15 DATA), (0.45, 1) (DATA → sa15 routingDataPkt), (0.43, 0.92) (RTS → sa15), (0.44. 0.93)
Blackhole	(da15 routingDataPkt → DATA), (0.10, 1) (da15 DATA → routingDataPkt), (0.10, 0.98) (sa_j → da15), (0.05, 0.99) j ∈ neighbor nodes
Sleep Deprivation	(routingCtrlPkt → da1 sa15 DATA), (0.08, 0.72) (da15, routingDataPkt → DATA), (0.15, 1) (sa15 routingDataPkt → DATA), (0.07, 1)
Packet Dropping (ALL)	(da15, routingDataPkt → DATA), (0.2, 1) (da15, DATA → routingDataPkt), (0.2, 0.98) expecting rule: (sa15, routingDataPkt → DATA)

IV.5. Discussion

From our experiment results, we conclude that the feature set described in Table I is sufficient for a monitoring node to profile behavior patterns of its neighboring nodes. Our IDS is effective to detect most of the simulated attacks. In addition, our IDS can quickly identify a malicious node within onehop range, especially when the decision module is enabled. Mobility can significantly decrease the detection rate and increase the false positive rate. We have a very good performance when the *maxspeed* is set to 5 m/s, which is a practical environment considering the radio transmission rate is 250 m/s.

Because there is no standard attack simulation environment for ad hoc networks, it is hard to make direct comparisons of our detection results to other related works. For resource consumption, conceptually our IDS could consume more resource than those which only use routing table information (e.g. Zhang and Lee [13]), because our IDS also collects information from MAC layer. However, the advantages of our proposed crosslayer IDS are evident.

- Cross-layer feature set provides the capability to identify attack source node using MAC address and to localize it within one-hop range. On the contrary, most of the other works did not address this problem. In [13], postdetection analysis (by using statistics of incoming and outgoing packets of certain nodes) is proposed to identify the IP address of attack source. This approach would cause a critical delay in developing countermeasures.
- Cross-layer feature set allows each mobile node to monitor both the MAC and network layer simultaneous, extends the detection capabilities to both layers. In our experiments, all of the simulated attacks are launched against the network layer, and we are able to detect most of these attacks with MAC layer features. Although for 10 m/s network, some of the attack detection, e.g. deprivation attack, is low. But one more step of data preprocessing (i.e. by removing the frequent RtDataPkts) could bring the detection rate up to 90-100%. In fact most of the network layer attacks will cause immediate effects at MAC layer, and detection at MAC layer can be more direct and more prompt.

- Cross-layer feature set can detect attacks that are unable or hard to detect by using single layer information. For example, it is very hard to detect our simulated blackhole attack using only the information from network layer, because after the attack source broadcasts the falsified routing information, all neighbor nodes update their routing table accordingly. From a neighbor node point of view, this routing change is quite normal.

Lastly, from our experiments, we conclude that that association rule mining can be effectively used to detect most of the resource consumption attacks as described in Section III-B, if a good feature set is used along with appropriate data preprocessing. This is because association rules can capture interesting patterns from frequently enough (i.e. about minsup) itemsets in the data that is being mined.

V. CONCLUSION

We have presented an anomalybased IDS for ad hoc networks using association rule mining technique. The IDS is devised for individual nodes in a given network, and monitors network data within radio transmission range. We have proposed a compact crosslayer feature set, which enables mobile node to monitor the MAC layer and the network layer simultaneously. The advantages of using cross-layer information include the capability of identifying and localizing attack sources, and the ability of detecting attacks that are unable to be detected by using single layer information. We have also developed a novel collaborative detection scheme that integrates local intrusion alerts with global intrusion alerts. This facilitates the proposed IDS to effectively detect a malicious node beyond onehop perimeter. Simulation results demonstrated that our method is effective with respect to the simulated attacks.

Manuscrit reçu le 19 février 2005
Accepté le 24 octobre 2005

REFERENCES

[1] CARDENAS (A. A.), RADOSAVAC (S.), BARAS (J. S.), Detection and Prevention of MAC Layer Misbehavior in Ad Hoc Networks, *Proceedings of the 2nd ACM Workshop on Security of Ad Hoc and Sensor Networks*, pp. 17-22, Oct. 2004.

[2] DAHILL (B.), LEVINE (B. N.), ROYER (E.), SHIELDS (C.), A Secure Routing Protocol for Ad Hoc Networks, *Technical Report 0137*, Department of Computer Science, University of Massachusetts, Aug. 2001.

[3] ZAPATA (M. G.), ASOKAN (N.), Securing Ad-Hoc Routing Protocols, *Proceedings of the 2002 ACM Workshop on Wireless Security*, pp. 1-10, Sept. 2002.

[4] HU (Y.-C.), PERRIG (A.), JOHNSON (D. B.), Rushing Attacks and Defense in Wireless Ad Hoc Network Routing Protocols, *ACM Workshop on Wireless Security 2003*, pp. 30-40, Sept. 2003.

[5] BRINKLEY (J.), TROST (W.), Authenticated Ad Hoc Routing at the Link Layer for Mobile Systems, *Wireless Networks* **7**, n° 2, pp. 139-145, Kluwer Academic Publishers, 2001.

[6] MARTI (S.), GIULI (T.), LAI (K.), BAKER (M.), Mitigating Routing Misbehavior in Mobile Ad Hoc Networks, *Proceedings of the 6th International Conference on Mobile Computing and Networking (MobiCom'00)*, pp. 255-265, Aug. 2000.

[7] ANJUM (F.), SUBHADRABANDHU (D.), SARKAR (S.), Signature based Intrusion Detection for Wireless AdHoc Networks: A Comparative study of various routing protocols, *Proceedings of Vehicular Technology Conference, Wireless Security Symposium*, **3**, pp. 2152-2156, Oct. 2003.

[8] SEKAR (R.), GUPTA (A.), FRULLO (J.), SHANBHAG (T.), TIWARI (A.), YANG (H.), ZHOU (S.), Specification-based Anomaly Detection: A New Approach for Detecting Network Intrusions, *Proceedings of the 9th ACM Conference on Computer and Communications Security*, pp. 265-274, Nov. 2002.

[9] Crashiis: "http://www.ll.mit.edu/IST/ideval/docs/1999/attackDB.html#crashiis".

[10] LEE (W.), STOLFO (S. J.), Adaptive Intrusion Detection: A Data Mining Approach, *ACM Transactions on Information and System Security (TISSEC)*, **3**, n° 4, pp. 227-261, Nov. 2000.

[11] ZHANG (Y.), LEE (W.), Intrusion Detection in Wireless Adhoc Networks, *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking (MobiCom'00)*, pp. 275-283, Aug. 2000.

[12] COHEN (W.W.), Fast Effective Rule Induction, *Proceedings of the Twelfth International Conference on Machine Learning*, pp. 115-123, Tahoe City, CA, Jul. 1995.

[13] Huang (Y.), Lee (W.), A Cooperative Intrusion Detection System for Ad Hoc Networks, *Proceedings of the 1st ACM workshop on Security of ad hoc and sensor networks*, pp. 135-147, Oct. 2003.

[14] Quinlan (J.R.), C4.5: Programs for Machine Learning, *Morgan Kaufmann*, San Mateo, CA, 1993.

[15] HAN (E.), KARYPIS (G.), KUMAR (V.), Text Categorization Using Weight Adjusted k-Nearest Neighbor Classification, *Proceedings of the 5th PacificAsia Conference on Knowledge Discovery and Data Mining*, pp. 53-65, Apr. 2001.

[16] DENG (H.), ZENG (Q.), Agrawal (D.P.), SVM-based Intrusion Detection System for Wireless Ad Hoc Networks, *Proceedings of the IEEE Vehicular Technology Conference (VTC'03)*, **3**, pp. 2147-2151, Oct. 2003.

[17] KACHIRSKI (O.), GUHA (R.), Intrusion Detection Using Mobile Agents in Wireless Ad Hoc Networks, *Proceedings of the IEEE Workshop on Knowledge Media Networking*, pp. 153-158, Jul. 2002.

[18] TSENG (C.), BALASUBRAMANYAM (P.), KO (C.), LIMPRASITTIPORN (R.), ROWE (J.), LEVITT (K.), A Specification-based Intrusion Detection System for AODV, *Proceedings of the 1st ACM Workshop on Security of ad hoc and Sensor Networks*, pp. 125-134, Oct. 2003.

[19] NEGI (R.), PERRIG (A.), Jamming Analysis of MAC Protocols, *Technical MemoCarnegie Mellon*, Feb. 2003.

[20] ZHANG (X.), WU (S.F.), FU (Z.), WU (T.L.), Malicious Packet Dropping: How It Might Impact the TCP Performance and How We Can Detect It, *Proceedings of the 2000 International Conference on Network Protocols*, pp. 263-272, Nov. 2000.

[21] HU (Y.-C.), PERRIG (A.), JOHNSON (D.B.), Packet Leashes: A Defense Against Wormhole Attacks in Wireless Ad Hoc Networks, *Proceedings of IEEE INFOCOM 2003*, **3**, pp. 1976-1986, Apr. 2003.

[22] AWERBUCH (B.), HOLMER (D.), NITAROTARU (C.), RUBENS (H.), An On-Demand Secure Routing Protocol Resilient to Byzantine Failures, *Proceedings of the ACM Workshop on Wireless Security 2002*, pp. 21-30, Sept. 2002.

[23] BURDICH (D.), CALIMLIM (M.), GEHRKE (J.), MAFIA: A Maximal Frequent Itemset Algorithm for Transactional Databases, *Proceedings of the 17th International Conference on Data Engineering (ICDE)*, pp. 443-452, Apr. 2001.

[24] AGRAWAL (R.), SRIKANT R., Fast Algorithms for Mining Association Rules, *Proceeding of the 20th Int'l Conference on Very Large Databases*, pp. 487-499, Sept. 1994.

[25] PEARL (J.), Probalistic Reasoning in Intelligent Systems: Networks of Plausible Inference, *Morgan Kaufmann*, 1988.

[26] JORDAN (M.I.), Learning in Graphical Models (Adaptive Computation and Machine Learning), Part I, III, *The MIT Press*, 1st ed., 1998.

[27] LIU (Y.), MAN (H.), "Network Vulnerability Assessment using Bayesian Networks, *SPIE Defense and Security Symposium*, **5812**, pp. 61-71, Orlando, FL, Mar. 2005.

[28] WU (YS.), FOO (B.), MEI (Y.), BAGCHI (S.), Collaborative Intrusion Detection System (CIDS): A Framework for Accurate and Efficient IDS, *IEEE the 19th Annual Computer Security Applications Conference*, Las Vegas, NV, Dec. 2003.

[29] KRUEGEL (C.), MUTZ (D.), ROBERTSON (W.), Valeur (F.), Bayesian Event Classification for Intrusion Detection, *IEEE the 19th Annual Computer Security Applications Conference*, Las Vegas, NV, Dec. 2003.

[30] BRONSTEIN (A.), DAS (J.), DURO (M.), FRIEDRICH (R.), COHEN (I.), Self-Aware Services: Using Bayesian Networks for Detecting Anomalies in Internetbased Services, *HP Labs Technical Reports HPL 2001-23R1*, 2001.

[31] COENEN (F.), LENG (P.), ZHANG (L.), Threshold Tuning for Improved Classification Association Rule Mining, *The 9th Pacific-Aisa Conference on Knowledge Discovery and Data Mining*, May 2005.

[32] HU (H.), LI (J.), Using Association Rules to Make Rule-based Classifiers Robust, *Proceedings of the 16th Australasian Database Conference (ACIS)*, Jan.Feb. 2005.

[33] BRUTCH (P.), KO (C.), Challenges in Intrusion Detection for Wireless Adhoc Networks, *Proceedings of the IEEE 2003 Symposium on Applications and the Internet Workshops*, pp. 368-373, Jan. 2003.

[34] KUMAR (S.), Classification and detection of computer intrusions, *Ph.D thesis*, Purdue University, Aug. 1995.

[35] LAZAREVIC (A.), ERTOZ (L.), OZGUR (A.), SRIVASTAVA (J.), KUMAR (V.), A Comparative Study of Anomaly Detection Schemes in Network Intrusion Detection", *Proc. SIAM Conf. Data Mining*, 2003.

[36] LAKHINA (A.), CROVELLA (M.), DIOT (C.), Diagnosing NetworkWide Traffic Anomalies, *ACM SSIGCOMM'04*, Portland, Oregon, Aug.-Sept. 2004.

[37] LEUNG (K.), LECKIE (C.), Unsupervised Anomaly Detection in Network Intrusion detection Using Clusters, *the 28th Australasian Computer Science Conference*, Australia, Jan. 2005.

[38] BROCH (J.), MALTZ (D.), JOHNSON (D.), HU (Y.-C.), JETCHEVA (J.), A Performance Comparison of MultiHop Wireless Ad Hoc Network Routing Protocols, *Proceedings of the 4th Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom'98)*, pp. 85-97, Oct. 1998.