Intrusion Detection in Wireless Ad-Hoc Networks

Yongguang Zhang
HRL Laboratories, LLC
Malibu, CA 90265
ygz@hrl.com

Wenke Lee
Department of Computer Science
North Carolina State University
Raleigh, NC 27695
wenke@csc.ncsu.edu

ABSTRACT
As the recent denial-of-service attacks on several major Internet sites have shown us, no open computer network is immune from intrusions. The wireless ad-hoc network is particularly vulnerable due to its features of open medium, dynamic changing topology, cooperative algorithms, lack of centralized monitoring and management point, and lack of a clear line of defense. Many of the intrusion detection techniques developed on a fixed wired network are not applicable in this new environment. How to do it differently and effectively is a challenging research problem. In this paper, we first examine the vulnerabilities of a wireless ad-hoc network, the reason why we need intrusion detection, and the reason why the current methods cannot be applied directly. We then describe the new intrusion detection and response mechanisms that we are developing for wireless ad-hoc networks.

1. INTRODUCTION
A wireless ad-hoc network consists of a collection of "peer" mobile nodes that are capable of communicating with each other without help from a fixed infrastructure. The interconnections between nodes are capable of changing on a continual and arbitrary basis. Nodes within each other's radio range communicate directly via wireless links, while those that are far apart use other nodes as relays. Nodes usually share the same physical media; they transmit and acquire signals at the same frequency band, and follow the same hopping sequence or spreading code. The data-link-layer functions manage the wireless link resources and coordinate medium access among neighboring nodes. The medium access control (MAC) protocol is essential to a wireless ad-hoc network because it allows mobile nodes to share a common broadcast channel. The network-layer functions maintain the multi-hop communication paths across the network; all nodes must function as routers that discover and maintain routes to other nodes in the network. Mobility and volatility are hidden from the applications so that any node can communicate with any other node as if everyone were in a fixed wired network. Applications of ad-hoc networks range from military tactical operations to civil rapid deployment such as emergency search-and-rescue missions, data collection/sensor networks, and instantaneous classroom/meeting room applications.

The nature of wireless ad-hoc networks makes them very vulnerable to an adversary's malicious attacks. First of all, the use of wireless links renders a wireless ad-hoc network susceptible to attacks ranging from passive eavesdropping to active interfering. Unlike wired networks where an adversary must gain physical access to the network wires or pass through several lines of defense at firewalls and gateways, attacks on a wireless ad-hoc network can come from all directions and target at any node. Damages can include leaking secret information, message contamination, and node impersonation. All these mean that a wireless ad-hoc network will not have a clear line of defense, and every node must be prepared for encounters with an adversary directly or indirectly.

Second, mobile nodes are autonomous units that are capable of roaming independently. This means that nodes with inadequate physical protection are receptive to being captured, compromised, and hijacked. Since tracking down a particular mobile node in a large scale ad-hoc network cannot be done easily, attacks by a compromised node from within the network are far more damaging and much harder to detect. Therefore, any node in a wireless ad-hoc network must be prepared to operate in a mode that trusts no peer.

Third, decision-making in ad-hoc networks is usually decentralized and many ad-hoc network algorithms rely on the cooperative participation of all nodes. The lack of centralized authority means that the adversaries can exploit this vulnerability for new types of attacks designed to break the cooperative algorithms.

For example, the current MAC protocols for wireless ad-hoc networks are all vulnerable. Although there are many MAC protocols, the basic working principles are similar. In a contention-based method, each node must compete for control of the transmission channel each time it sends a message. Nodes must strictly follow the pre-defined procedure to avoid collisions or to recover from them. In a contention-free method, each node must seek from all other nodes a unanimous promise of an exclusive use of the channel resource, on a one-time or recurring basis. Regardless of the type of MAC protocol, if a node behaves maliciously, the MAC protocol can break down in a scenario resembling a denial-of-service attack. Although such attacks are rare in wired networks because the physical networks and the MAC layer are isolated from the outside world by layer-3 gateways/firewalls, every mobile node is completely vulnerable in the wireless open medium.

Ad-hoc routing presents another vulnerability. Most ad-hoc routing protocols are also cooperative in nature[14]. Unlike with a wired network, where extra protection can be placed on routers and gateways, an adversary who hijacks an ad-hoc node could paralyze the entire wireless network by disseminating false routing information. Worse, such false routing information could result in messages from all nodes being fed to the compromised node.

Intrusion prevention measures, such as encryption and authentication, can be used in ad-hoc networks to reduce intrusions, but cannot eliminate them. For example, encryption and authentication cannot defend against compromised mobile nodes, which carry the private keys. Integrity validation using redundant information (from different nodes), such as those being used in secure routing [16, 17], also relies on the trustworthiness of other nodes, which could likewise be a weak link for sophisticated attacks.

The history of security research has taught us a valuable lesson – no matter how many intrusion prevention measures are inserted in a network, there are always some weak links that one could exploit to break in. Intrusion detection presents a second wall of defense and it is a necessity in any high-survivability network.

In summary, a wireless ad-hoc network has inherent vulnerabilities that are not easily preventable. To build a highly secure wireless ad-hoc network, we need to deploy intrusion detection and response techniques, and further research is necessary to adapt these techniques to this new environment, from their original applications in fixed wired network. In this paper, we propose our new model for intrusion detection and response in mobile, ad-hoc wireless networks. We are currently investigating the use of cooperative statistical anomaly detection models for protection from attacks on ad-hoc routing protocols, on wireless MAC protocols, or on wireless applications and services. We are integrating them into a cross-layer defense system and are investigating its effectiveness, efficiency, and scalability.

2. BACKGROUND OF INTRUSION DETECTION

As network-based computer systems play increasingly vital roles in modern society, they have become the targets of our enemies and criminals. When an intrusion (defined as "any set of actions that attempt to compromise the integrity, confidentiality, or availability of a resource" [4]) takes place, intrusion prevention techniques, such as encryption and authentication (e.g., using passwords or biometrics), are usually the first line of defense. However, intrusion prevention alone is not sufficient because as systems become ever more complex, while security is still often the after-thought, there are always exploitable weaknesses in the systems due to design and programming errors, or various "socially engineered" penetration techniques (as illustrated in the recent "I Love You" virus). For example, even though they were first reported many years ago, exploitable "buffer overflow" security holes, which can lead to an unauthorized root shell, still exist in some recent system softwares. Furthermore, as illustrated by recent Distributed Denial-of-Services (DDOS) attacks launched against several major Internet sites where security measures were in place, the protocols and systems that are designed to provide services (to the public) are inherently subject to attacks such as DDOS. Intrusion detection can be used as a second wall to protect network systems because once an intrusion is detected, e.g., in the early stage of a DDOS attack, response can be put into place to minimize damages, gather evidence for prosecution, and even launch counter-attacks.

The primary assumptions of intrusion detection are: user and program activities are observable, for example via system auditing mechanisms; and more importantly, normal and intrusion activities have distinct behavior. Intrusion detection therefore involves capturing audit data and reasoning about the evidence in the data to determine whether the system is under attack. Based on the type of audit data used, intrusion detection systems (IDSs) can be categorized as network-based or host-based. A network-based IDS normally runs at the gateway of a network and "captures" and examines network packets that go through the network hardware interface. A host-based IDS relies on operating system audit data to monitor and analyze the events generated by programs or users on the host. Intrusion detection techniques can be categorized into *misuse detection* and *anomaly detection*.

Misuse detection systems, e.g., IDIOT [8] and STAT [5], use patterns of well-known attacks or weak spots of the system to match and identify known intrusions. For example, a signature rule for the "guessing password attack" can be "there are more than 4 failed login attempts within 2 minutes". The main advantage of misuse detection is that it can accurately and efficiently detect instances of known attacks. The main disadvantage is that it lacks the ability to detect the truly innovative (i.e., newly invented) attacks.

Anomaly detection systems, for example, IDES [12], flag observed activities that deviate significantly from the established normal usage profiles as anomalies, i.e., possible intrusions. For example, the normal profile of a user may contain the averaged frequencies of some system commands used in his or her login sessions. If for a session that is being monitored, the frequencies are significantly lower or higher, then an anomaly alarm will be raised. The main advantage of anomaly detection is that it does not require prior knowledge of intrusion and can thus detect new intrusions. The main disadvantage is that it may not be able to describe what the attack is and may have high false positive rate.

Conceptually, an intrusion detection model, i.e., a misuse detection rule or a normal profile, has these two components:

- the *features* (or attributes, measures), e.g., "the number of failed login attempts", "the averaged frequency of the *gcc* command", etc., that together describe a logical event, e.g., a user login session;

- the *modeling algorithm*, e.g., rule-based pattern matching, that uses the features to identify intrusions.

Defining a set of *predictive* features that accurately capture the representative behaviors of intrusive or normal activities is the most important step in building an effective intrusion detection model, and can be independent of the design of modeling algorithms.

In 1998, DARPA (U.S. Defense Advanced Research Projects Agency) sponsored the first Intrusion Detection Evaluation to survey the state-of-the-art of research in intrusion detection [11]. The results indicated that the research systems were much more effective than the leading commercial systems. However, even the best research systems failed to detect a large number of new attacks, including those that led to unauthorized user or root access.

It is very obvious that the enemies, knowing that intrusion prevention and detection systems are installed in our networks, will attempt to develop and launch new types of attacks. In anticipation of these trends, IDS researchers are designing new sensors and hence new audit data sources and features, new anomaly detection algorithms, techniques for combining anomaly and misuse detection, and system architectures for detecting distributed and coordinated intrusions.

3. PROBLEMS OF CURRENT IDS TECHNIQUES

The vast difference between the two networks makes it very difficult to apply intrusion detection techniques developed for a fixed wired network to an ad-hoc wireless network. The most important difference is perhaps that the latter does not have a fixed infrastructure, and today's network-based IDSs, which rely on real-time traffic analysis, can no longer function well in the new environment. Compared with wired networks where traffic monitoring is usually done at switches, routers and gateways, an ad-hoc network does not have such traffic concentration points where the IDS can collect audit data for the entire network. Therefore, at any one time, the only available audit trace will be limited to communication activities taking place within the radio range, and the intrusion detection algorithms must be made to work on this partial and localized information.

The second big difference is in the communication pattern in a wireless ad-hoc network. Wireless users tend to be stingy about communication due to slower links, limited bandwidth, higher cost, and battery power constraints. Disconnected operations [15] are very common in wireless network applications, and so is location-dependent computing or other techniques that are solely designed for wireless networks and seldom used in the wired environment. All these suggest that the anomaly models for wired network cannot be used, as is, in this new environment.

Furthermore, there may not be a clear separation between normalcy and anomaly in wireless ad-hoc networks. A node that sends out false routing information could be the one that has been compromised, or merely the one that is temporarily out of sync due to volatile physical movement. Intrusion detection may find it increasingly difficult to distinguish false alarms from real intrusions.

In summary, we must answer the following research questions in developing a viable intrusion detection system for wireless ad-hoc networks:

- What is a good system architecture for building intrusion detection and response systems that fits the features of wireless ad-hoc networks?
- What are the appropriate audit data sources? How do we detect anomaly based on partial, local audit traces – if they are the only reliable audit source?
- What is a good model of activities in a wireless communication environment that can separate anomaly when under attacks from the normalcy?

For the rest of this paper we will address these challenging problems.

4. NEW ARCHITECTURE

Intrusion detection and response systems should be both distributed and cooperative to suite the needs of wireless ad-hoc networks. In our proposed architecture (Figure 1), every node in the wireless ad-hoc network participates in intrusion detection and response. Each node is responsible for detecting signs of intrusion locally and independently, but neighboring nodes can collaboratively investigate in a broader range.

In the systems aspect, individual IDS agents are placed on each and every node. Each IDS agent runs independently and monitors local activities (including user and systems activities, and communication activities within the radio range). It detects intrusion from local traces and initiates response. If anomaly is detected in the local data, or if the evidence is inconclusive and a broader search is warranted, neighboring IDS agents will cooperatively participate in global intrusion detection actions. These individual IDS agent collectively form the IDS system to defend the wireless ad-hoc network.

The internal of an IDS agent can be fairly complex, but conceptually it can be structured into six pieces (Figure 2). The data collection module is responsible for gathering local audit traces and activity logs. Next, the local detection engine will use these data to detect local anomaly. Detection methods that need broader data sets or that require collaborations among IDS agents will use the cooperative detection engine. Intrusion response actions are provided by both the local response and global response modules. The local response module triggers actions local to this mobile node, for example an IDS agent alerting the local user, while the global one coordinates actions among neighboring nodes, such as the IDS agents in the network electing a remedy action. Finally, a secure communication module provides a high-confidence communication channel among IDS agents.

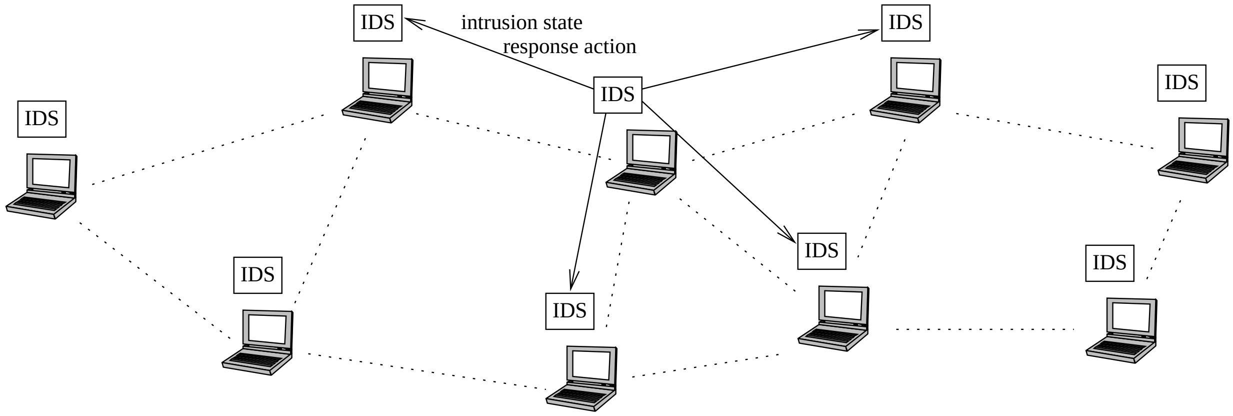

Figure 1: The IDS Architecture for Wireless Ad-Hoc Network

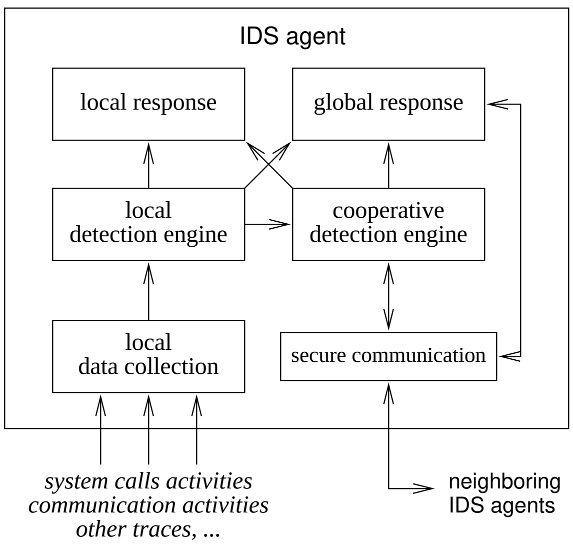

Figure 2: A Conceptual Model for an IDS Agent

4.1 Data Collection

The first module, local data collection, gathers streams of real-time audit data from various sources. Depending on the intrusion detection algorithms, these useful data streams can include system and user activities within the mobile node, communication activities by this node, as well as communication activities within the radio range and observable by this node. Therefore, multiple data collection modules can coexist in one IDS agents to provide multiple audit streams for a multi-layer integrated intrusion detection method (Section 6).

4.2 Local Detection

The local detection engine analyzes the local data traces gathered by the local data collection module for evidence of anomalies. Because it is conceivable that the number of newly created attack types mounted on wireless networks will increase quickly as more and more network appliances become wireless, we cannot simply employ a few expert rules that are only capable of detecting the few known types of attack. Furthermore, updating the rule-base with new detection rules across a wireless ad-hoc network in a secure and reliable manner is never easy. Therefore, we believe that the IDS for a wireless ad-hoc network should mainly use statistical anomaly detection techniques. In general, the procedure of building such an anomaly detection model is the following:

- the normal profiles (i.e., the normal behavior patterns) are computed using trace data from a "training" process where all activities are normal;
- the deviations from the normal profiles are recorded during a "testing" process where some normal and abnormal activities (if available) are included;
- a detection model is computed from the deviation data to distinguish normalcy and anomalies; although there will always be "new" normal activities that have not been observed before, their deviations from the normal profiles should be much smaller than those of intrusions.

In previous work on fixed wired networks [10], we have developed efficient data mining algorithms for computing normal traffic patterns from TCP/IP trace data (i.e., *tcpdump* [6] output), as well as classification techniques for building misuse and anomaly detection models. The results from the 1998 DARPA Evaluation showed that the detection models produced by our system had one of the best overall performances among the participating systems. The main challenges here are how to define the trace data, and how to determine the types of patterns that best describe the normal behavior. While there are many anomaly detection models for user behavior and system activities (e.g., [2, 3, 9]), our focus here is on new models for wireless ad-hoc networks (Section 5).

4.3 Cooperative Detection

Any node that detects locally a known intrusion or anomaly with *strong* evidence (i.e., the detection rule triggered has a very high accuracy rate), can determine independently that the network is under attack and can initiate a response. However, if a node detects an anomaly or intrusion with weak evidence, or the evidence is inconclusive but warrants broader investigation, it can initiate a cooperative global intrusion detection procedure. This procedure works by propagating the intrusion detection state information among neighboring nodes (or further downward if necessary).

The intrusion detection state information can range from a mere level-of-confidence value such as

- "With $p\%$ confidence, node A concludes from its local data that there is an intrusion"
- "With $p\%$ confidence, node A concludes from its local data and neighbor states that there is an intrusion"
- "With $p\%$ confidence, node A, B, C, ... collectively conclude that there is an intrusion"

to a more specific state that lists the suspects, like

- "With $p\%$ confidence, node A concludes from its local data that node X has been compromised"

or to a complicated record including the complete evidence.

As the next step, we can derive a distributed consensus algorithm to compute a new intrusion detection state for this node, using other nodes' state information received recently. The algorithm can include a weighted computation under the assumption that nearby nodes have greater effects than far away nodes, i.e., giving the immediate neighbors the highest values in evaluating the intrusion detection states.

For example, a majority-based distributed intrusion detection procedure can include the following steps:

- the node sends to neighboring node an "intrusion (or anomaly) state request";
- each node (including the initiation node) then propagates the state information, indicating the likelihood of an intrusion or anomaly, to its immediate neighbors;
- each node then determines whether the *majority* of the received reports indicate an intrusion or anomaly; if yes, then it concludes that the network is under attack;
- *any* node that detects an intrusion to the network can then initiate the response procedure.

The rationales behind this scheme are as follows. Audit data from other nodes cannot be trusted and should not be used because the compromised nodes can send falsified data. However, the compromised nodes have no incentives to send reports of intrusion/anomaly because the intrusion response may result in their expulsion from the network. Therefore, unless the *majority* of the nodes are compromised, in which case one of the legitimate nodes will probably be able to detect the intrusion with strong evidence and will respond, the above scheme can detect intrusion even when the evidence at individual nodes is weak.

A wireless network is highly dynamic because nodes can move in and out of the network. Therefore, while each node uses intrusion/anomaly reports from other nodes, it does not rely on fixed network topology or membership information in the distributed detection process. It is a simple *majority* voting scheme where *any* node that detects an intrusion can initiate a response.

4.4 Intrusion Response

The type of intrusion response for wireless ad-hoc networks depends on the type of intrusion, the type of network protocols and applications, and the confidence (or certainty) in the evidence. For example, here is a few likely response:

- Re-initializing communication channels between nodes (e.g, force re-key).
- Identifying the compromised nodes and re-organizing the network to preclude the promised nodes.

For example, the IDS agent can notify the end-user, who may in turn do his/her own investigation and take appropriate action. It can also send a "re-authentication" request to all nodes in the network to prompt the end-users to authenticate themselves (and hence their wireless nodes), using out-of-bound mechanisms (like, for example, visual contacts). Only the re-authenticated nodes, which may collectively negotiate a new communication channel, will recognize each other as legitimate. That is, the compromised/malicious nodes can be excluded.

5. ANOMALY DETECTION IN WIRELESS AD-HOC NETWORKS

In this section, we discuss how to build anomaly detection models for wireless networks. Detection based on activities in different network layers may differ in the format and the amount of available audit data as well as the modeling algorithms. However, we believe that the principle behind the approaches will be the same. To illustrate our approach, we focus our discussions on ad-hoc routing protocols.

5.1 Detecting Abnormal Updates to Routing Tables

The main requirement of an anomaly detection model is low false positive rate, calculated as the percentage of normalcy variations detected as anomalies, and high true positive rate, calculated as the percentage of anomalies detected. We need to first determine the trace data to be used that will bear evidence of normalcy or anomaly. For ad-hoc routing protocols, since the main concern is that the false routing information generated by a compromised node will be disseminated to and used by other nodes, we can define the trace data to describe, for each node, the normal (i.e., legitimate) updates of routing information.

A routing table usually contains, at the minimum, the next hop to each destination node and the distance (number of hops). A legitimate change in the routing table can be caused by the physical movement(s) of node(s) or network membership changes. For a node, its own movement and the

Distance	Direction	Velocity	PCR	PCH
0.01	S	0.1	20	15
10	S	20	80	50
0.02	N	0.1	0	0
...	...	...	...	...

Table 1: Sample Trace Data for Ad-Hoc Routing

change in its own routing table are the only *reliable* information that it can trust. Hence, we use data on the node's physical movements and the corresponding change in its routing table as the basis of the trace data. The physical movement is measured by distance, direction, and velocity (this data can be obtained by a built-in GPS device). The routing table change is measured by the percentage of changed routes (PCR), and the (positive or negative) percentage of changes in the sum of hops of all the routes (PCH). We use percentages as measurements because of the dynamic nature of wireless networks (i.e., the number of nodes/routes is not fixed). Table 1 shows some fictional trace data for a node.

During the "training" process, where a diversity of normal situations are simulated, the trace data is gathered for each node. The trace data sets of all nodes in the training network are then aggregated into a single data set, which describes all normal changes in routing tables for all the nodes. A detection model which is learned from this aggregated data set will therefore be capable of operating on any node in the network.

A normal profile on the trace data in effect specifies the correlation of physical movements of the node and the changes in the routing table. We can use the following scheme to compute the normal profile:

- denote PCR the *class* (i.e. *concept*), and distance, direction, velocity, and PCH the *features* describing the concept;
- use n classes to represent the PCR values in n ranges, for example, we can use 10 classes each representing 10 percentage points – that is, the trace data belongs to n classes;
- apply a classification algorithm to the data to learn a classifier for PCR;
- repeat the above for PCH, that is, learn a classifier for PCH;

A classification algorithm, e.g., RIPPER [1], can use the most discriminating feature values to describe each concept. For example, when using PCR as the concept, RIPPER can output classification rules in the form of: "if (distance $\leq$ 0.01 AND PCH $\leq$ 20) then PCR = 2; else if ...". Each classification rule (an "if") has a "confidence" value, calculated as the percentage of records that match both the rule condition and rule conclusion out of those that match the rule condition. The classification rules for PCR and PCH together describe what are the (normal) conditions that correlate with the (amount of) routing table changes. We use these rules as the normal profiles.

PCR deviation	PCH deviation	Class
0.0	0.0	normal
0.1	0.0	normal
0.2	0.2	normal
0.9	0.5	abnormal
0.3	0.1	normal
...	...	...

Table 2: Sample Deviation Data

Checking an observed trace data record (that records a routing table update) with the profile involves applying the classification rules to the record. A misclassification, e.g., when the rules say it is "PCR = 3" but in fact it is "PCR = 5", is counted as a violation. We can use the "confidence" of the violated rule as the "deviation score" of the record. In the "testing" process, the deviation scores are recorded. For example, if abnormal data is available, we can have deviation data like those shown in Table 2. We can then apply a classification algorithm to compute a classifier, a detection model, that uses the deviation scores to distinguish abnormal from normal.

If abnormal data is not available, we can compute the normal *clusters* of the deviation scores, where each score pair is represented by a point (*PCR deviation*, *PCH deviation*) in the two-dimensional space, e.g., (0.0, 0.0), (0.2, 0.2), (0.3, 0.1), etc. The "outliers", i.e., those that do not belong to any normal cluster, can then be considered as anomalies. Clustering is often referred to as "un-supervised learning" because the target clusters are not known *a priori*. Its disadvantage is that the computation (i.e., the formation) of clusters is very time consuming. If the application environment allows a *tolerable* false alarm rate, e.g., 2%, then the clustering algorithm can be parameterized to terminate when sufficient, e.g., greater than 98%, points are in proper clusters.

A poor performance of the anomaly detection model, e.g., a higher than acceptable false alarm rate, indicates that the data gathering (including both "training" and "testing" processes) is not sufficient, and/or the features and the modeling algorithms need to be refined. Therefore, repeated trials may be needed before a good anomaly detection model is produced.

In the discussion thus far, we have used only the *minimal* routing table information in the anomaly detection model to illustrate our approach, which can be applied to all routing protocols. For a specific protocol, we can use additional routing table information and include new features in the detection model to improve the performance. For example, for DSR ad-hoc routing protocol [7, 13], we can add source route information (the complete, ordered sequence of network hops leading to the destination). We can also add predictive features according to the "temporal and statistical" patterns among the routing table updates, following the similar *feature construction* process we used to build intrusion detection models for wired networks [10]. For example, for a wired TCP/IP network, a "SYN-flood" DOS attack has a pattern which indicates that a lot of half-open connections are attempted against a service in a short time span. Accordingly, a feature, "for the past 2 seconds, the percentage of connections to the same service that are half-open" was constructed and had been proved to be highly predictive. Similarly, in a wireless network, if an intrusion results in a large number of routing table updates, we can add a feature that measures the frequency (how often) the updates take place.

Our objective in this study is to lead to a better understanding of the important and challenging issues in intrusion detection for ad-hoc routing protocols. First, using a given set of training, testing, and evaluation scenarios, and modeling algorithms (e.g., with RIPPER as the classification algorithm for protocol trace data and "nearest neighbor" as the clustering algorithm for deviation scores), we can identify which routing protocol, with potentially all its routing table information used, can result in better performing detection models. This will help answer the question "what information should be include in the routing table to make intrusion detection effective." This finding can be used to design more robust routing protocols. Next, using a given routing protocol, we can explore the feature space and algorithm space to find the best performing model. This will give insight to the general practices of building intrusion detection for wireless networks.

5.2 Detecting Abnormal Activities in Other Layers

Anomaly detection for other layers of the wireless networks, e.g., the MAC protocols, the applications and services, etc., follows a similar approach. For example, the trace data for MAC protocols can contain the following features: for the past s seconds, the total number of channel requests, the total number of nodes making the requests, the largest, the mean, and the smallest of all the requests, etc. The *class* can be the range (in the number) of the current requests by a node. A classifier on this trace data describes the normal context (i.e. history) of a request. An anomaly detection model can then be computed, as a classifier or clusters, from the deviation data.

Similarly, at the wireless application layer, the trace data can use the service as the *class* (i.e., one class for each service), and can contain the following features: for the past s seconds, the total number of requests to the same service, the number of different services requested, the average duration of the service, the number of nodes that requested (any) service, the total number of service errors, etc. A classifier on the trace data then describes for each service the normal behaviors of its requests.

Many attacks generate different statistical patterns than normal requests. Since the features described above are designed to capture the statistical behavior of the requests, the attacks, when examined using the feature values, will have large deviations than the normal requests. For example, compared with normal requests to MAC or an application-level service, DOS attacks via resource exhaustion normally involve a huge number of requests in a very short period of time; a DDOS has the additional tweak that it comes from many different nodes.

6. MULTI-LAYER INTEGRATED INTRUSION DETECTION AND RESPONSE

Traditionally, IDSs use data only from the lower layers: network-based IDSs analyze TCP/IP packet data and host-based IDSs analyze system call data. This is because in wired networks, application layer firewalls can effectively prevent many attacks, and application-specific modules, e.g., credit card fraud detection systems, have also been developed to guard the mission-critical services.

In the wireless networks, there are no firewalls to protect the services from attack. However, intrusion detection in the application layer is not only feasible, as discussed in the previous section, but also necessary because certain attacks, for example, an attack that tries to create an unauthorized access "back-door" to a service, may seem perfectly legitimate to the lower layers, e.g., the MAC protocols. We also believe that some attacks may be detected much earlier in the application layer, because of the richer semantic information available, than in the lower layers. For example, for a DOS attack, the application layer may detect very quickly that a large number of incoming service connections have no actual operations or the operations don't make sense (and can be considered as errors); whereas the lower layers, which rely only on information about the amount of network traffic (or the number of channel requests), may take a longer while to recognize the unusually high volume.

Given that there are vulnerabilities in multiple layers of wireless networks and that an intrusion detection module needs to be placed at each layer on each node of a network, we need to coordinate the intrusion detection and response efforts. We use the following integration scheme:

- if a node detects an intrusion that affects the entire network, e.g., when it detects an attack on the ad hoc routing protocols, it initiates the re-authentication process to exclude the compromised/malicious nodes from the network;
- if a node detects a (seemingly) local intrusion at a higher layer, e.g., when it detects attacks to one of its services, lower layers are notified. The detection modules there can then further investigate, e.g., by initiating the detection process on possible attacks on ad hoc routing protocols, and can respond to the attack by blocking access from the offending node(s) and notifying other nodes in the network of the incident.

In this approach, the intrusion detection module at each layer still needs to function properly, but detection on one layer can be initiated or aided by evidence from other layers. As a first cut of our experimental research, we allow the evidence to flow from one layer to its (next) lower layer by default, or to a specific lower layer based on the application environment.

The "augmented" versions of the detection model at a lower level are constructed as follows. In the "testing" process, the anomaly decision, i.e., either 1 for "yes" or 0 for "no" from the upper layer is inserted into the deviation score of the lower level, for example, (0.1, 0.1) now becomes (0.1, 0.1, 0). In other words, the deviation data also carries the extra information passed from the upper level. An anomaly detection model built from the augmented data therefore combines the bodies of evidence from the upper layers and the current layer and can make a more informed decision. The intrusion report sent to other node for cooperative detection also includes a vector of the information from the layers.

With these new changes, the lower layers now need more than one anomaly detection model: one that relies on the data of the current layer and therefore indirectly uses evidence from the lower layers, and the augmented one that also considers evidence from the upper layer.

The multi-layer integration enables us to analyze the attack scenario in its entirety and as a result, we can achieve better performance in terms of both higher true positive and lower false positive rates. For example, a likely attack scenario is that an enemy takes control of the mobile unit of a user (by physically disable him or her), and then uses some system commands to send falsified routing information. A detection module that monitors user behavior, e.g., via command usage, can detect this event and immediately (i.e., before further damage can be done) cause the detection module for the routing protocols to initiate the global detection and response, which can result in the exclusion of this compromised unit. As another example, suppose the users are responding to a fire alarm, which is a rare event and may thus cause a lot of unusual movements and hence updates to the routing tables. However, if there is no indication that a user or a system software has been compromised, each intrusion report sent to other nodes will have a "clean" vector of upper layer indicators, and thus the detection module for the routing protocols can conclude that the unusual updates may be legitimate.

7. CONCLUSION

We have argued that any secure network will have vulnerability that an adversary could exploit. This is especially true for wireless ad-hoc networks. Intrusion detection can compliment intrusion prevention techniques (such as encryption, authentication, secure MAC, secure routing, etc.) to improve the network security. However new techniques must be developed to make intrusion detection work better for the wireless ad-hoc environment.

Through our continuing investigation, we have shown that an architecture for better intrusion detection in wireless ad-hoc networks should be distributed and cooperative. A statistical anomaly detection approach should be used. The trace analysis and anomaly detection should be done locally in each node and possibly through cooperation with all nodes in the network. Further, intrusion detection should take place in all networking layers in an integrated cross-layer manner.

Currently, we are continuing our investigation in the architecture issues, the anomaly detection model, and the multi-layer integration approach. For architecture study, we are refining its design and plan to implement it and study its performance implications. For anomaly detection model, we are studying the effectiveness and scalability of our approach for building anomaly detection models for ad-hoc routing protocols and for other layers of wireless networking. In particular, we will first focus on two questions about ad-hoc routing: what information a routing protocol should include to make intrusion detection effective, and what is the best anomaly detection model for a given routing protocol. Finally, we will study the effectiveness gain (i.e., in detection rate) with the multi-layer integration approach, as well as its performance penalties.

8. REFERENCES

[1] W. W. Cohen. Fast effective rule induction. In *Machine Learning: the 12th International Conference*, Lake Taho, CA, 1995. Morgan Kaufmann.

[2] S. Forrest, S. A. Hofmeyr, A. Somayaji, and T. A. Longstaff. A sense of self for Unix processes. In *Proceedings of the 1996 IEEE Symposium on Security and Privacy*, pages 120–128, Los Alamitos, CA, 1996. IEEE Computer Society Press.

[3] A. K. Ghosh and A. Schwartzbard. A study in using neural networks for anomaly and misuse detection. In *Proceedings of the 8th USENIX Security Symposium*, 1999.

[4] R. Heady, G. Luger, A. Maccabe, and M. Servilla. The architecture of a network level intrusion detection system. Technical report, Computer Science Department, University of New Mexico, August 1990.

[5] K. Ilgun, R. A. Kemmerer, and P. A. Porras. State transition analysis: A rule-based intrusion detection approach. *IEEE Transactions on Software Engineering*, 21(3):181–199, March 1995.

[6] V. Jacobson, C. Leres, and S. McCanne. *tcpdump*. available via anonymous ftp to ftp.ee.lbl.gov, June 1989.

[7] D. B. Johnson and D. A. Maltz. Dynamic source routing in ad hoc wireless networks. In T. Imielinski and H. Korth, editors, *Mobile Computing*, pages 153–181. Kluwer Academic Publishers, 1996.

[8] S. Kumar and E. H. Spafford. A software architecture to support misuse intrusion detection. In *Proceedings of the 18th National Information Security Conference*, pages 194–204, 1995.

[9] T. Lane and C. Brodley. Temporal sequence learning and data reduction for anomaly detection. *ACM Transactions on Information and System Security*, 2(3), August 1999.

[10] W. Lee, S. J. Stolfo, and K. W. Mok. A data mining framework for building intrusion detection models. In *Proceedings of the 1999 IEEE Symposium on Security and Privacy*, May 1999.

[11] R. Lippmann, D. Fried, I. Graf, J. Haines, K. Kendall, D. McClung, D. Weber, S. Webster, D. Wyschogrod, R. Cunninghan, and M. Zissman. Evaluating intrusion detection systems: The 1998 darpa off-line intrusion detection evaluation. In *Proceedings of the 2000 DARPA Information Survivability Conference and Exposition*, January 2000.

[12] T. Lunt, A. Tamaru, F. Gilham, R. Jagannathan, P. Neumann, H. Javitz, A. Valdes, and T. Garvey. A real-time intrusion detection expert system (IDES) - final technical report. Technical report, Computer Science Laboratory, SRI International, Menlo Park, California, February 1992.

[13] D. A. Maltz, J. Broch, J. Jetcheva, and D. B. Johnson. The effects of on-demand behavior in routing protocols for multi-hop wireless ad hoc networks. *IEEE Journal on Selected Areas in Communications*, Aug. 1999.

[14] E. Royer and C.-K. Toh. A review of current routing protocols for ah hoc mobile wireless networks. *IEEE Personal Communication*, 6(2):46–55, Apr. 1999.

[15] M. Satyanarayanan, J. J. Kistler, L. B. Mummert, M. R. Ebling, P. Kumar, and Q. Lu. Experiences with disconnected operation in a mobile environment. In *Proceedings of USENIX Symposium on Mobile and Location Independant Computing*, pages 11–28, Cambridge, Massachusetts, Aug. 1993.

[16] B. R. Smith, S. Murthy, and J. Garcia-Luna-Aceves. Securing distance-vector routing protocols. In *Proceedings of Internet Society Symposium on Network and Distributed System Security*, pages 85–92, San Diego, California, Feb. 1997.

[17] L. Zhou and Z. J. Haas. Securing ah hoc networks. *IEEE Network*, 13(6):24–30, Nov/Dec 1999.