تعقب المتطفلين في شبكات اللاسلكي الخاصة(Ad-Hoc)

سلا م عبدا لنبي ثجيل
مدرس مساعد

الجامعة المستنصرية/كلية التربية
قسم علوم الحاسبات

المستخلص
في الوقت الحاضر ومن خلال تعدد الهجمات التي يقوم بها المتطفلين والمخربين على مواقع الانترنيت المختلفة فان هذا دليل على انه ليست هناك شبكة حاسوب مفتوحة(مرتبطة بشبكة الانترنيت) منيعة ومحصنة من اختراقات المتطفلين.الشبكات اللاسلكية بصورة عامة هي اضعف من الشبكات السلكية وذلك بسبب إنها بيئة اتصال مفتوحة يمكن الارتباط بها دون وجود أسلاك مرتبطة مع المصدر وكذلك شكلها الهندسي المتغير، ضعف المراقبة المركزية على الشبكة اللاسلكية وضعف في إدارة نقاط الارتباط بين الحاسبات المرتبطة بواسطة الشبكة هذه الأسباب جعلت هذا النوع من الشبكات سهل الاختراق.
هناك العديد من وسائل وطرق متابعة وتحديد المتطفلين على شبكة الاتصال السلكية وهذه الطرق غير قابلة للتنفيذ على الشبكات اللاسلكية بسبب الاختلاف الكبير بين الشبكتين لذلك أصبح موضوع متابعة المتطفلين على الشبكات اللاسلكية موضوع مهم للباحثين وتحدي جديد لهم.
في هذا البحث نحن نفحص ونحدد نقاط الضعف الموجودة في الشبكات اللاسلكية ونصف طريقة ذكية(باستخدام الشبكات العصبية) لتعقب والرد على المتطفلين والتي طورت للشبكات اللاسلكية من نوع (ad-hoc networks)

Intrusion Detection in Wireless Ad-Hoc Networks

Salam A. Thajeal
Assistance lecture
The university of Al-mustansiryah, collage of education,
Computer science department

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 examine the vulnerabilities of a wireless ad-hoc network, and describe the new intelligent intrusion detection and response mechanisms (using neural network) 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 implication of mobile computing on network security research can be further demonstrated by the follow case. Recently (Summer 2001) an Internet worm called Code Red has spread rapidly to infect many of the Windows-based server machines. To prevent this type of worm attacks from spreading into intranets, many companies rely on firewalls to protect the internal networks. However, there are multiple incidents that the Code Red worm has been caught from within the intranet, largely due to the use of mobile computers. As more and more business travelers are carrying laptops and more and more public venues (e.g. conferences) provide wireless Internet access, there are higher and higher chances that an inadequately protected laptop will be infected with worms.[1,2]

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. [3]

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 gate-ways, 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. [4, 5]

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. [2]

Ad-hoc routing is presents another vulnerability. Most ad-hoc routing protocols are also cooperative in nature [6]. 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 [7, 8], 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. The Need for Intrusion Detection

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 often carry the private keys. Integrity validation using redundant information (from different nodes), such as those being used in secure routing [9, 10], also relies on the trust worthiness 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 (just like the example at the beginning of this paper). Intrusion detection presents a second wall of defense and it is a necessity in any high-survivability network.

In summary, mobile computing environment has inherent vulnerabilities that are not easily preventable.

To secure mobile computing applications, we need to deploy intrusion detection and response techniques, and further research is necessary to adapt these techniques to the new environment, from their original applications in _xed wired network. In this paper, we focus on a particular type of mobile computing environment called mobile ad-hoc networks and propose a new model for intrusion detection and response for this environment.

We will first give a background on intrusion detection, and then present our new architecture, followed by an experimental study to evaluate its feasibility.

3. 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" [3]) 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 software. 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 [11] and STAT [12], 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 [13], 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.

4. 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 [14] 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.

5. The Proposed 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 (as shown in 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 (is as shown in Figure 2).

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 (as shown in Figure 3).

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.

The principle configuration of the IDS is as shown in Figure 4. While the actions steps of the IDS is as shown in Figure 5.

5.1 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.

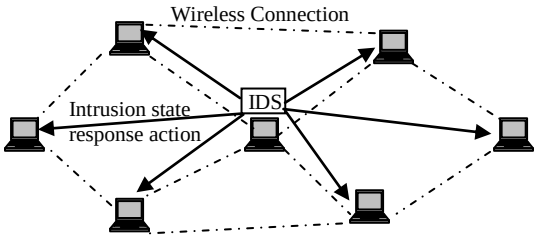

Figure 1: The IDS Architecture for Wireless Ad-Hoc Network

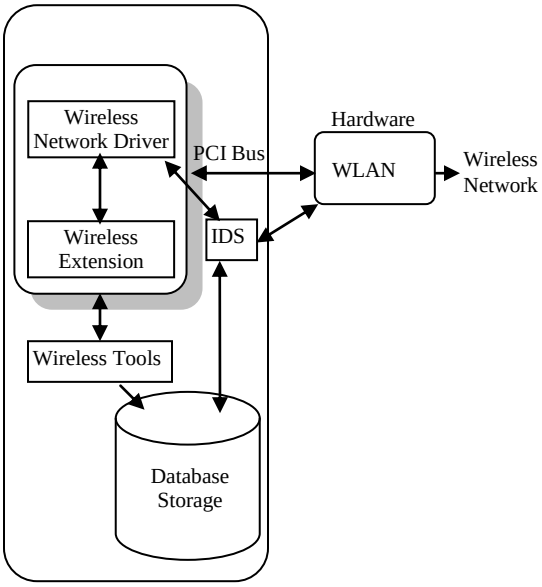

Figure 2 Access Point Node System Architecture

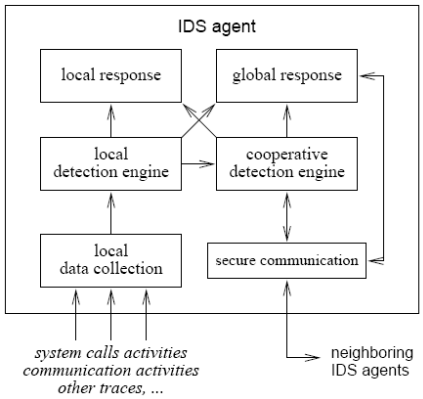

Figure 3: A Conceptual Model for an IDS Agent

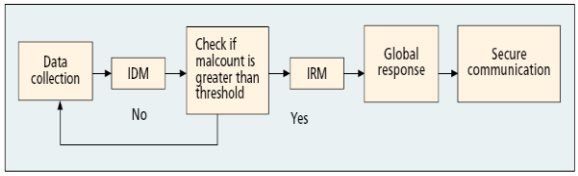

Figure 4 the Handling of Internal Attacks

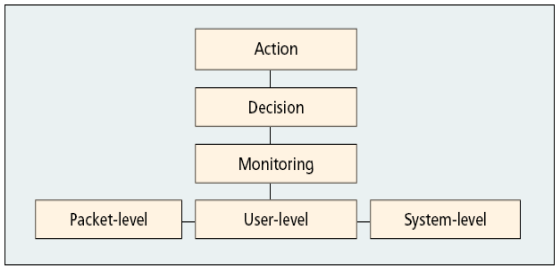

Figure 5 Modular Intrusion Detection Architecture.

5.2 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 [15], we have developed efficient data mining algorithms for computing normal traffic patterns from TCP/IP trace data (i.e., tcpdump [16] 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., [17, 4, 18]), our focus here is on new models for wireless ad-hoc networks.

5.3 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 confidence, node A concludes from its local data that there is an intrusion"
- "With confidence, node A concludes from its local data and neighbor states that there is an intrusion"
- With confidence, node A, B, C, ... collectively conclude that there is an intrusion" to a more specific state that lists the suspects, like "With 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 has greater effects than far away nodes, i.e., giving the immediate neighbor 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.

5.4 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.

6. 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.

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. All these statistical values and features are stored in the database and used in intrusion detection processing. Any new intrusion features and statistical will be stored in this database. 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.

7. Multilayer 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.

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.

8. The Proposed Intelligent Network Intrusion Detection System

With the rapid expansion of computer networks during the past decade, security has become a crucial issue for computer systems. Different soft-computing based methods have been proposed in recent years for the development of intrusion detection systems.

The Intelligent Intrusion detection engine deals with the filtered packets, which have a probability of being malicious, from the packet-based classification engine. The intelligent intrusion detection engine will have to learn to distinguish anomalous data packets from normal packets.

The all known intruder patterns and scenarios are stored in the database depend on the classification threats.

In ANN, the input layer is connected to one-hidden layers with structure (35 35 3) each layer has a 35 neurons. The output layer (3 neurons) is used in a categorization problem and represents classes to which the input vector can belong (like normal, malicious /intruder, unclear).

Also, the intelligent detection engine can control the flow that goes back outside of tested network.

This section presents the proposed intelligent intrusion detection using ANN. ANN is used for intrusion detection based analysis and classification approach. Most of the previous studies have focused on classification of records in one of the two general classes - normal and attack, this proposed research aims to solve a multi class problem in which the type of attack is also detected by the wavelet neural network.

An early stopping validation method is also applied in the training phase to increase the generalization capability of the neural network.

There are two main approaches to the design of NIDSs. In a misuse detection based IDS, intrusions are detected by looking for activities that correspond to known signatures of intrusions or vulnerabilities.

On the other hand, anomaly detection based NIDS detects intrusions by searching for abnormal network traffic. The abnormal traffic pattern can be defined either as the violation of

accepted thresholds for frequency of events in a connection or as a user's violation of the legitimate profile developed for user normal behavior.

Figure 6 shows the block diagram of proposed intelligent network intrusion detection subsystem which consists of:

• Probe: collects the network traffic of a host or a network, abstracts the traffic into a set of statistical variables to reflect the network status, and periodically generates reports to the event preprocessor.

• Event Preprocessor: receives reports from both the probe and IDAs (Intelligent Detection Agents) from the other nodes of lower tiers, and converts the information into the format required by the statistical model.

• Statistical Processor: maintains a reference model of the typical network activities, compares the reports from the event preprocessor to the reference models, and forms a stimulus vector to feed into the neural network classifiers.

• Neural Network Classifier: analyzes the stimulus vector from the statistical model to decide whether the network traffic is normal or not.

• Post Processor: generates reports for the agents at higher tiers. At the same time, it may display the results through a user interface.

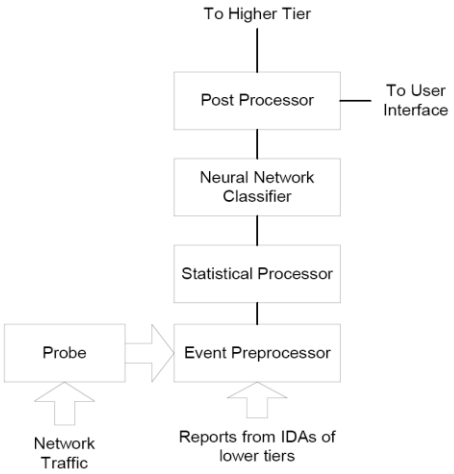

Figure 6 The Block Diagram of The Proposed NIDS.

A. Attack Detection Types

There are at least four different known categories of computer attacks; they are denial of service attacks, user to root attacks, remote to user attacks and probing attacks [18].

Many different attack types are included in the dataset used for this study: SYN Flood (Neptune), worms and Satan. These two attack types were selected from two different attack categories (denial of service and probing) to check for the ability of the intrusion detection system to identify attacks from different categories.

SYN Flood (Neptune) is a denial of service attack to which every TCP/IP implementation is vulnerable (to some degree). For distinguishing a Neptune attack network traffic is monitored for a number of simultaneous SYN packets destined for a particular machine. The host sending these packets is usually unreachable.

Satan is a probing intrusion which automatically scans a network of computers to gather information or find known vulnerabilities. The network probes are quite useful for attackers planning a future attack.

Worms are self-replicating programmable to propagate across network, typically having a detrimental effect. Each computer worm has a few essential components, such as the target locator and the infection propagator modules, and a couple of other nonessential modules, such as the remote control, update interface, life-cycle manager, and payload routines. Table 1 shows some tested worm files used in this proposal.

B. Features: Selection, Numerical Representation, and Normalization

In many attack scenarios, the signature of the attack record is identified through examination of some features in a sequence of records. Therefore, the IDS should analyze the service types used by the same user in previous connections and for this purpose the past events are described in the computer network are included in the feature vector.

A complete description of all features is available [15, 7]. Instead of describing all the features, here features will be divided into three groups and provide descriptions and examples for each group stored into database.

Group 1 includes features describing the commands used in the connection (instead of the commands themselves). These features describe the aspects of the commands that have a key role in defining the attack scenarios. Examples of this group are number of file creations, number of operations on access control files, number of root accesses, etc...

Table 1. Tested Computer Worms File and Their Infection Methods			
Name / Discovered	Type	Infection	Execution Method
WM/ShareFun February 1997	Microsoft Mail dependent mailer	Word 6 and 7 documents	By user
Win/RedTeam January 1998	Injects outgoing mail to Eudora mailboxes	Infects Windows NE files	By user
W32/Ska@m (Happy99 worm) January 1999	32-bit Windows mailer worm	Infects WSOCK32.DLL (by inserting a little hook function)	By user
W97M/Melissa@mm March 1999	Word 97 mass-mailer worm	Infects other Word 97 documents	By user
VBS/LoveLetter@mm May 2000	Visual Basic Script mass-mailer worm	Overwrites other VBS files with itself	By user
W32/Nimda@mm September 2001	32-bit Windows mass-mailer worm	Infects 32-bit PE files	Exploits vulnerabilities to execute itself on target

Group 2 includes features describing the connection specifications. This group includes a set of features that present the technical aspects of the connection. Examples of this group include: protocol type, flags, duration, service types, number of data bytes from source to destination, etc...

Group 3 includes features describing the connections to the same host in last 2 seconds. Examples of this group are: number of connections having the same destination host and using the same service, % of connections to the current host that have a rejection error, % of different services on the current host, etc...

Clearly these features could not have any effect on classification and only made it more complicated and time consuming. They were excluded from the data vector. Hence the data vector was a 35 dimensional vector.

Different possible values for selected features were extracted and a numerical value was attributed to each of them. For example, for the protocol type the possible numerical values were: tcp=0, udp=1, icmp=2. This numerical representation is necessary because the feature vector fed to the input of the neural network has to be numerical.

The ranges of the features are different and this makes them incomparable. Some of the features had binary values where some others had a continuous numerical range (such as duration of connection). As a result, the features are normalized by mapping all the different values for each feature to [0, 1] range.

B.1 Statistical Model

Statistical methods have been used in anomaly intrusion detection systems [4]; however, most of these systems simply measure the means and the variances of some variables and detect whether certain thresholds are exceeded. SRI's NIDES [4, 12] developed a more sophisticated statistical algorithm by using a $\chi 2$-like test to measure the similarity between short-term and long-term profiles. Current statistical model uses a similar algorithm as NIDES but with major modifications.

In NIDES, user profiles are represented by a number of probability density functions. Let S be the sample space of a random variable and events E1, E2,…, Ek a mutually exclusive partition of S. Assume Pi is the expected probability of the occurrence of the event Ei, and let Pi ' be the frequency of the occurrence of Ei during a given time interval. Let N denote the total number of occurrences. NIDES statistical algorithm uses a $\chi 2$-like test to determine the similarity between the expected and actual distributions through the statistic:

$$Q = N \times \sum_{i=1}^{k} \frac{(P_i' - P_i)^2}{P_i}$$

When N is large and the events E1,E2,…,Ek are independent, Q approximately follows a $\chi 2$ distribution with $(k-1)$ degrees of freedom. However in a real-time application the above two assumptions generally cannot be guaranteed, thus, empirically, Q may not follow a $\chi 2$ distribution. NIDES solves this problem by building an empirical probability distribution for Q, which is updated daily in a real-time operation.

In this proposed system, since a neural network classifier to identify possible intrusions was used. However, because network traffic is not stationary and network-based attacks may have different time durations, varying from a couple of seconds to several hours or longer, an algorithm which is capable of efficiently monitoring network traffic with different time windows was needed. Based on the above observations, a layer window statistical model was used, Figure 7 and with each layer-window corresponding to a monitoring time slice of increasing size.

The newly arrived events will first be stored in the event buffer of layer 1. The stored events are

compared with the reference model of that layer and the results are then fed into the neural network classifier to decide the network status during that time window. The event buffer will be emptied once it becomes full, and the stored events will be averaged and forwarded to the event buffer of layer 2. This process will be repeated recursively until the top level is reached where the events will simply be dropped after processing.

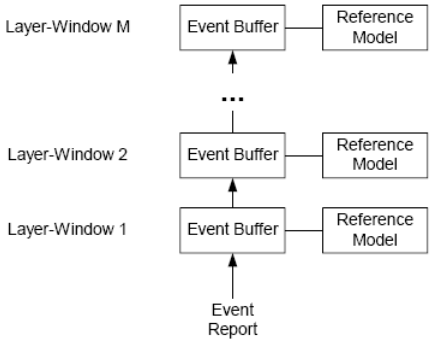

Figure 7 The Statistical Model.

The similarity-measuring equation used in this proposed NIDS is as shown below:

$$Q = f(N).[\sum_{i=1}^{k} |p_i' - p_i| + \max_{i=1}^{k} (|p_i' - p_i|)]$$

Where f (N) is a function that takes into account the total number of occurrences during a time window.

Besides similarity measurements, also an algorithm for the real-time updating of the reference model was designed. Let old p be the reference model before updating, new p be the reference model after updating, and obs p be the observed user activity within a time window. The formula to update the reference model is:

$$\overline{p}_{new} = s \times \alpha \times \overline{p}_{obs} + (1 - s \times \alpha) \times \overline{p}_{old}$$

In which α is the predefined adaptation rate and s is the value generated by the output of the neural network. Assume that the output of the neural network classifier is a continuous variable t between –1 and 1, where –1(false) means intrusion with absolute certainty and 1(true) means no intrusion again with complete confidence. In between, the values of t indicate proportionate levels of certainty. The function for calculating s is:

$$s = \begin{cases} t, \text{if } t \geq 0 \\ 0, \text{otherwise} \end{cases}$$

Through the above equations, the reference model would ensure updated actively for typical traffic while kept unchanged when attacks occurred. The attack events will be diverted and stored, as attack scripts, for future neural network learning.

B.2. INIDS Experimental Results

The implemented intrusion detector was built from a three layer MLP (Multi-Layer Perception). The structure is referred to as: {35 35 3}. At this stage, early stopping validation was not applied and the training was performed for 200 times. The training process took more than 47 hours. The error clearly decreased to an outstanding level (comparable to zero). Therefore, it was expected to have good classification results. The final correct classification rate on training set confirms this theory: it agreement is very close to 100%. However, when unseen data (test set) was fed to the neural network, the result was undesirable.

The validation set used in this study consisted of 900 data records (300 of each class). The same neural network {35 35 3} was trained this time by applying early stopping validation method.

As expected, the correct classification rate on the training set declined slightly (equal to 100% in the first experiment). Instead, when unseen data (test set) was fed to the neural network the result was considerably better than the first experiment in which the early stopping method was not applied. Table 2 shows some tested for ANN (35 35 3)

Table 2 Correct Classification Rates in Three Different Training Validation -Testing Sessions for the ANN (35 35 3)

Session	Classification Time in Training set/ detection	Classification Rate on Training set	Classification Time in Test set/ detection	Classification on Test set	Classification Time in actual set/ detection	Classification on actual set
1	1.1 msec	100	1.1 msec	94.3	1.15 msec	97.1
2	1.1 msec	100	1.1 msec	92.5	1.15 msec	97.1
3	1.1 msec	100	1.1 msec	92.2	1.15 msec	97.0
Average	1.1 msec	100	1.1 msec	93.0	1.15 msec	97.3

9. 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 connections, 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 architecture for better intrusion detection in wireless ad-hoc networks should be distributed and cooperative. A statistical anomaly detection coupled with neural network approaches 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.

The proposed intelligent intrusion detection system has the capability of capturing layer 4 data packets and analyzing them exhaustively then tracking the packet into receiving application. For packet comparison, a database containing a substantial number of data packets related to various known worms, and possible attack scenarios for some threats is used.

Currently the potential of using several AI methods in the intelligent intrusion detection engine, adds anomaly-detection-based IDS and worms monitor capability to the engine were explored.

A neural network, for instance, trained with a set of malicious-code patterns, will create warnings when similar patterns are detected. Problems to tackle include the performance of the decision process (to deal with real-time network traffic) and optimizations to avoid the creation of too many false positives. The selection of suitable techniques will require sufficient experimental evidence of their applicability.

An approach for a neural network based intrusion detection system, intended to classify the normal and attack patterns and the type of the attack, is presented in this research. The early stopping validation method which increases the generalization capability of the neural network and at the same time decreased the training time is applied. It should be mentioned that the long training time of the neural network is mostly due to the huge number of training vectors of computation facilities. However, when the neural network parameters are determined by training, classification of a single record is done in a negligible time. Therefore, the neural network based IDS can operate as an online classifier for the attack types that it has been trained for.

In order to avoid unreasonable complexity in the neural network, an initial classification of the connection records to normal and general categories of attacks can be the first step. The records in each category of intrusions can then be further classified to the attack types.

10. REFERENCES

[1] C. Sinclair, L. Pierce, and S. Matzner, “An application of machine learning to network intrusion detection”, Proceedings of 15th Annual Computer Security Applications Conference, Phoenix, AZ, pp. 371-377, 1999

[2] 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, 1998.

[3] 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

[4] Susan C. Lee and David V. Heinbuch, "Training a neural network-based intrusion detector to recognize novel attacks", IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans, 31(4) pp.294-299, July, 2001

[5] R. A. Kemmerer and G. Vigna, “Intrusion detection: a brief history and overview,” Computer, vol. 35, no. 4, pp. 27–30, 2002.

[6] 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.

[7] 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.

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

[9] M. Swimmer, "Review and Outlook of the Detection of Viruses using Intrusion Detection Systems", Proceedings of the 3rd International Workshop on Recent Advances in Intrusion Detection (RAID 2000

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

[11] S. Kumar and E. H. Spa®ord. A software architecture to support misuse intrusion detection. In Proceedings of the 18th National

Information Security Conference, pages 194{204, 1995
[12] 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.
[13] 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) - ¯nal technical report. Technical report, Computer Science Laboratory, SRI International, Menlo Park, California, February 1992.
[14] 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.
[15] 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.
[16] V. Jacobson, C. Leres, and S. McCanne. Tcpdump. available via anonymous ftp to ftp.ee.lbl.gov, June 1989.

[17] S. Forrest, S. A. Hofmeyr, A. Somayaji, and T. A. Longsta®. 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.
[18] 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.