Summer 2006

Unified architecture of mobile ad hoc network security (MANS) system

Li Ling
New Jersey Institute of Technology

ABSTRACT

UNIFIED ARCHITECTURE OF THE MOBILE AD HOC NETWORK SECURITY (MANS) SYSTEM

by
Li Ling

In this dissertation, a unified architecture of Mobile Ad-hoc Network Security (MANS) system is proposed, under which IDS agent, authentication, recovery policy and other policies can be defined formally and explicitly, and are enforced by a uniform architecture. A new authentication model for high-value transactions in cluster-based MANET is also designed in MANS system. This model is motivated by previous works but try to use their beauties and avoid their shortcomings, by using threshold sharing of the certificate signing key within each cluster to distribute the certificate services, and using certificate chain and certificate repository to achieve better scalability, less overhead and better security performance. An Intrusion Detection System is installed in every node, which is responsible for colleting local data from its host node and neighbor nodes within its communication range, pro-processing raw data and periodically broadcasting to its neighborhood, classifying normal or abnormal based on pro-processed data from its host node and neighbor nodes. Security recovery policy in ad hoc networks is the procedure of making a global decision according to messages received from distributed IDS and restore to operational health the whole system if any user or host that conducts the inappropriate, incorrect, or anomalous activities that threaten the connectivity or reliability of the networks and the authenticity of the data traffic in the networks. Finally, quantitative risk assessment model is proposed to numerically evaluate MANS security.

UNIFIED ARCHITECTURE OF THE MOBILE AD HOC NETWORK SECURITY (MANS) SYSTEM

by
Li Ling

A Dissertation
Submitted to the Faculty of
New Jersey Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of
Doctor of Philosophy in Electrical Engineering

Department of Electrical and Computer Engineering

August 2006

APPROVAL PAGE

Unified Architecture of the Mobile Ad-hoc Network Security (MANS) System

Li Ling

Dr. Constantine N. Manikopoulos, Dissertation Advisor — Date
Associate Professor of Electrical and Computer Engineering, NJIT

Dr. Mengchu Zhou, Committee Member — Date
Professor of Electrical and Computer Engineering, NJIT

Dr. Roberto Rojas-Cessa, Committee Member — Date
Assistant Professor of Electrical and Computer Engineering, NJIT

Dr. Jie Hu, Committee Member — Date
Assistant Professor of Electrical and Computer Engineering, NJIT

Dr. Robert Statica, Committee Member — Date
Director, Computer Forensic & Cybersecurity lab, Information Technology Program, NJIT

BIOGRAPHICAL SKETCH

Author: Li Ling

Degree: Doctor of Philosophy

Date: August 2006

Undergraduate and Graduate Education:

- Doctor of Philosophy in Electrical Engineering, New Jersey Institute of Technology, Newark, NJ, 2006
- Master of Science in Electrical and Telecommunication Engineering, Beijing University of Posts and Telecommunication, Beijing, China, 2001
- Bachelor of Science in Wireless Telecommunication Engineering. Chongqing University of Posts and Telecommunication, Chongqing, China, 1998

Major: Electrical Engineering

Presentations and Publications:

Li Ling, Sui Song and Constantine N. Manikopoulos,
"Windows NT User Profiling for Masquerade Detection,"
IEEE International Conference on Networking, Sensing and Control,
Ft. Lauderdale, Florida, April 23-25, 2006.

Sui Song, Li Ling and Constantine N. Manikopoulos,
"Flow-based Statistical Aggregation Schemes for Network Anomaly Detection,"
IEEE International Conference on Networking, Sensing and Control,
Ft. Lauderdale, Florida, April 23-25, 2006.

Li Ling and Constantine N. Manikopoulos,
"Windows NT One-class Masquerade Detection,"
Proceedings of the 5th IEEE Systems, Man and Cybernetics Information Assurance Workshop, West Point, NY, June 10-11, 2004.

Constantine N. Manikopoulos and Li Ling,
"Architecture of the Mobile Ad-hoc Network Security (MANS) System,"
IEEE International Conference on Systems, Man and Cybernetics,
Washington DC, October 5-8, 2003.

Li Ling, Yang Dacheng and Wang Wenbo,
"TDD/CDMA Capacity Loss due to Adjacent Channel Interference in the Manhattan Environment,"
5th CDMA International Conference and Exhibition, Korea, November, 2000.

To my mom and my brother,
for their endless love and support

ACKNOWLEDGMENT

I would like to express my deepest gratitude to my advisor, Dr. Constantine N. Manikopoulos, for giving me the opportunity to work with him and leading me into this exciting field. I am greatly indebted to my advisor for the invaluable guidance and encouragement, which he has provided me through my study. He is always open to new ideas and I really appreciate the freedom he gave me while working on my research. This research could not have been possible without his brilliant ideas. Moreover, Dr. Manikopoulos also provided the facilities and guidelines for testing and evaluation of my dissertation research. I owe much for his unending help to do the research and for his financial support during my graduate study.

Special thanks to Dr. Roberto Rojas-Cessa, Dr. Jie Hu, Dr. Mengchu Zhou and Dr. Robert Statica for serving on my committee, reviewing this dissertation and providing valuable suggestions.

I would like to express my sincere appreciation to Dr. Zheng Zhang who provided me the precious help to start my research. All the fellow members of my project team and all my friends in and out of NJIT have been great sources of ideas and fun. I thank them for making my Ph.D study productive and enjoyable.

I am forever indebted to the love and trust of my family. My mom and my brother have always been a source of inspiration, support, advice and happiness without which I would not have been able to go this far.

TABLE OF CONTENTS

TABLE OF CONTENTS
(Continued)

Chapter **Page**

TABLE OF CONTENTS
(Continued)

Chapter **Page**

TABLE OF CONTENTS
(Continued)

Chapter **Page**

LIST OF TABLES

LIST OF FIGURES

LIST OF FIGURES
(Continued)

Figure **Page**

CHAPTER 1

INTRODUCTION

Mobile Ad Hoc networks (MANET) are kind of networks that do not rely on any fixed infrastructure; instead, all networking functions (e.g., routing, mobility management, etc.) are implemented by the nodes themselves. MANET technologies seek to provide mobile user "anytime, anywhere" networking services potentially, even in disaster area and battle fields.

The challenge of provision of security in Mobile Ad-Hoc Network (MANET) environments is tackled in this dissertation. The most important characterizing feature of a MANET is the absence of any node in a central role. So many security services that rely on central services, such as naming services, certification authorities (CA), network-based intrusion detection and other recovery administrative services will be impossible or at least much harder to implement. Another nagging major challenge is that of the compromised node(s); this could be an overtaken attacked node or a physically captured node. This forces all MANET nodes to operate in a mode that "trusts no peer" that complicates and inhibits security services [1], [2], [3].

Protection of Ad Hoc networks basically focus on two aspects: Intrusion Prevention, such as traffic encryption, sending data through multiple paths, authentication and authorization; Intrusion Detection, such as Anomaly pattern examination, Protocol analytical study. However, intrusion prevention cannot guard against internal attacks when an adversary has broken in, while intrusion detection itself is not enough because: Detecting intrusion must compiled with recovery response to remove the malicious hosts; IDS can only rely on partial and local data because of lack of traffic concentration, such

as routers and switches in Ad Hoc networks; Separation between normal and abnormal is difficult in Ad Hoc networks.

In this dissertation, the architecture of the Mobile Ad-hoc Network Security (MANS) System is proposed, under which authentication agent, IDS agent, recovery policy and other policies can be defined formally and explicitly, and are enforced by a uniform architecture. As a basic preventive solution, a fully localized, digital certificate base authentication services is proposed. An intrusion Detection and response system (IDS) provides the corrective solution feeding the certificate services with information about misbehaving nodes, which are eliminated from the network by certificate revocation. Both certificate services and IDS are designed to be robust even in the presence of compromised nodes.

Authentication service is based on asymmetric cryptographic techniques, specifically RSA algorithms. Authentication service is provided by taking a certificate-based approach. An IDS is installed in every node, which is responsible for colleting local data from its host node and neighbor nodes within its communication range, pro-processing raw data and periodically broadcasting to its neighborhood, classifying normal or abnormal based on pro-processed data from its host node and neighbor nodes.

Security recovery policy in ad hoc networks is the procedure of making a global decision according to messages received from distributed IDS and restore to operational health the whole system if any user or host that conducts the inappropriate, incorrect, or anomalous activities that threaten the connectivity or reliability of the networks and the authenticity of the data traffic in the networks.

MANET routing protocol is also secured by combing Certificate-based authentication and Intrusion Detection in MANS architecture. Quantitative risk assessment model is also proposed to numerically evaluate MANS security.

CHAPTER 2

ARCHITECTURE OF MANS SYSTEM

2.1 Introduction

2.1.1 What is Security?

Stoneburner in his draft version of the NIST recommendation states the goal of the IT security as follows:

The goal of information technology security is to enable an organization to meet all of its mission/business objectives by implementing systems with due care consideration of IT-related risks to the organization, its partners and its customers [32].

The security goals can be achieved by considering some security objectives or security requirements. Several people have classified these requirements or objectives in different ways but the following classification covers all the issues and groups them properly. This classification of security objectives is extracted from [32].

Availability: Availability of a system assures that the system works properly and the services are available or not denied to authorized users. It should also ensure that the system is available to the intended users and for intended use only. This objective protects against the following

- Intentional or accidental attempts to either perform unauthorized deletion of data or cause a denial of service or data.
- Attempts to use a system or data for unauthorised purpose.

Integrity: Integrity, in the IT world, contains two aspects – data integrity and system integrity. Data integrity means that the data is free from unauthorised

manipulation. Unauthorised manipulation of data can happen either in storage, during processing or during transmission. Like data integrity, system integrity means that the system has not been manipulated or even accessed in an unauthorised manner.

Confidentiality: Confidentiality of data or system information means that only the intended user receives the information and that the information is not disclosed to any unauthorised individual. The confidentiality principle applies to data in storage, processing, and in transmission.

Accountability: Accountability is a requirement that actions of an entity must be traced uniquely to that entity. This requirement becomes more and more important as business becomes increasingly dependent on IT. Accountability becomes significant for issues like non-repudiation, fault isolation, intrusion detection and prevention, after-action recovery and legal action.

Assurance: Assurance is required to show that the security measures have been properly implemented and they work as intended. When an implementation has implemented the functionality correctly, has sufficient protection against unintentional errors and has sufficient resistance to intentional penetration then it can be considered that the objectives have been adequately met.

Authors in [32] also propose a very comprehensive security model that can form the architectural basis in IT security. This model shows the primary services, supporting services and relationship between them. This model consists of support, prevention and recovery related services. Support services are quite generic and form the foundation of security; preventive services concentrate on preventing any security compromise before it

happens and finally the recovery services deal with identifying any security breach and recovering the system from it.

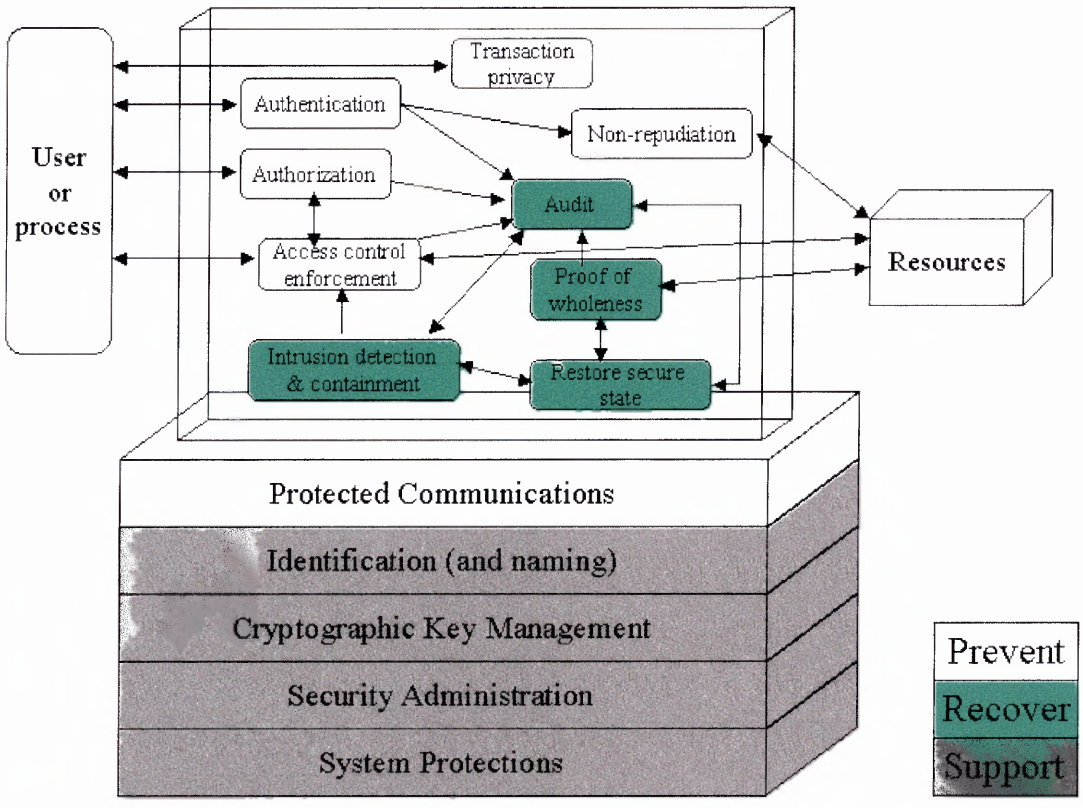


Figure 2.1 Security services mode.

2.1.2 Security Challenges for Mobile Ad Hoc Networks

Recently research about MANET is focus on routing protocols, many security issues have not been treated yet. Provision of security in Mobile Ad-Hoc Network environments is challenging, compared with wired networks because:

- Mobile ad hoc networks do not have centralized monitoring and/or management points. All the hosts involved share responsibilities. So many security services that rely on central services, such as naming services, certification authorities (CA), network-based intrusion detection and other recovery administrative services will be impossible or at least much harder to implement.

- Because of node mobility, the network topology changes dynamically. This changes the trust relationship between the participating hosts and requires dynamic configuration both in the network and in the hosts.

- The nodes communicate only via wireless links and these links are open to eavesdropping, replay and spoofing.

- The nodes are generally battery-powered, therefore, the communication and computing costs are limited.

- Because of the physical characteristics (memory, processor etc) of the mobile nodes, the computational power is limited.

- Mobile hosts can be physically captured and compromised. If a host gets compromised and the network cannot detect this then the attack can come from inside the network. Therefore, any security architecture should avoid having any central entity so that compromise of one entity does not risk the whole network.

- A compromised host also means that routing is compromised as well. The host can easily send incorrect routing information and it is not very easy to detect this.

- Moreover, once a compromised host gets key or certificate, authentication is bypassed. This forces all MANET nodes to operate in a mode that "trusts no peer" that complicates and inhibits security services.

2.1.3 Motivation of Unified Security Architecture for MANET

First of all, the challenge of provision of security in Mobile Ad-Hoc Network (MANET) environments is discussed in this section. The most important characterizing feature of a MANET is the absence of any node in a central role. So many security services that rely on central services, such as naming services, certification authorities (CA), network-based intrusion detection and other recovery administrative services will be impossible or at least much harder to implement. Another nagging major challenge is that of the compromised node(s); this could be an overtaken attacked node or a physically captured node. This forces all MANET nodes to operate in a mode that "trusts no peer" that complicates and inhibits security services [1], [2], [3].

Protection of Ad Hoc networks basically focus on two aspects: Intrusion Prevention, such as Traffic encryption, Sending data through multiple paths, Authentication and authorization; Intrusion Detection, such as Anomaly pattern examination, Protocol analytical study. However, intrusion prevention cannot guard against internal attacks when an adversary has broken in, while intrusion detection itself is not enough because: Detecting intrusion must compiled with recovery response to remove the malicious hosts; IDS can only rely on partial and local data because of lack of traffic concentration, such as routers and switches in Ad Hoc networks; Separation between normal and abnormal is difficult in Ad Hoc networks.

Therefore, a unified security architecture is needed because:

- Ad Hoc networks are distributed systems whose nodes may have little, if any knowledge of other nodes;
- Different agents, policies needs to be included to achieve different security goals;
- Different agent operating within that network may find itself interacting with others. For example, IDS agent, routing agent and authentication agent.

We proposed in this chapter a unified security architecture (MANS), under which IDS agent, authentication, recovery policy and other policies can be defined formally and explicitly, and are enforced by a uniform architecture.

Authentication module is based on asymmetric cryptographic techniques, specifically RSA algorithms [21]. Basically, our work are motivated by previous works such as [7], [12], [18], [26] and the *clustering phenomenon* in ad hoc networks [25]. We provide authentication service by taking a *certificate-based* approach.

An IDS is also installed in every node, which is responsible for colleting local data from its host node and neighbor nodes within its communication range, pro-

processing raw data and periodically broadcasting to its neighborhood, classifying normal or abnormal based on pro-processed data from its host node and neighbor nodes.

Security recovery policy in ad hoc networks is the procedure of making a global decision according to messages received from distributed IDS and restore to operational health the whole system if any user or host that conducts the inappropriate, incorrect, or anomalous activities that threaten the connectivity or reliability of the networks and the authenticity of the data traffic in the networks.

In this chapter, we describe a fully decentralized scalable security recovery policy for MANETs. It is Law-Governed [4] and formally extends the ideas of the Local Collaborative Groups by defining neighborhoods and their states. In the following sections, we first present the concept of our recovery policy, then describe the architecture of developing this recovery policy following the formal steps of specification, implementation, and enforcement, which is the development life cycle described in [5].

2.2 LCG-based Security Recovery Policy

The security recovery policy P is formally defined as the pentuple $\{M, G, L, N, M_N\}$[4]:

- P is the explicit and enforced set of principles of interaction among the members of this group, as we have defined later, which states which response should be taken in which cases.
- G is a distributed group of agents, called the *P-Group,* that are permitted to send and receive *P-Messages,* and are the participants in policy P. Here, G includes all mobile nodes in the MANETs.
- M is the predefined set of messages that may be exchanged among all the member agents. Messages are regulated by P. They are called *P-Messages.* These messages are generally small messages that are sent from neighborhood to

neighborhood or broadcast through the entire network. Examples are messages for Voting-Request, Authentication-Update, Voting-Decision, and other small messages. This kind of message has higher priority than type M_N messages.

- N is the neighborhood of each node within G. In our recovery policy, each node keeps next-hop information, periodically, as its neighborhood history. Each time when neighborhood voting is needed, each node can calculate its current neighborhood N based on current neighborhood information and history information.

- M_N are also *P-Messages* regulated by P. The difference is that these are messages of only neighborhood reach, sent only through the neighborhood. They typically include IDS information for each neighborhood node and neighborhood decision management messages. In our security policy, these kind of messages are limited to the local neighborhood; it is only neighborhood voting and decision results that may be more widely disseminated by broadcasting them through the network. This largely local reach of M_N messages, reduces security related traffic overhead. This messaging scheme also scales well, as it increases linearly with the number of nodes.

Following [4] and others, L is the set of rules regulating the exchange of *P-Message* between members of group G, called the *law* of this system. Generally, the law determines who can send which message to whom, and what the effect of that message should be.

The security recovery policy P defines the outcomes of regulated events occurring at members of G. The policy is defined locally for each agent but applies globally to all agents throughout the MANET. The policy P, thus defined, regulates explicitly only local events at individual agents and mandates only local operations to be carried out. Nevertheless, the policy P is globally obeyed and P-messages can be broadcast throughout the whole MANET. Working together with other services such as Routing, Authentication and Intrusion Detection, the locally regulated policy can affect the whole network.

2.3 Neighborhood Based "Watch"

For example, either periodically or triggered by some event, the nodes of a neighborhood send out an IDS_Status (x) (MN message) of their own IDS summary but raw information to all the other nodes in their neighborhood; in this manner, all nodes continuously are cognizant of the IDS monitoring results and security status of all the other neighborhood nodes, in addition, of course, to their own. Thus, if a security breach is imminent, say at node A, this will be simultaneously and independently arrived at, not only by node A, but by all other nodes in the neighborhood of A, since all the nodes in the neighborhood of A carry continuous and current updates of activity at A, just as A does, by virtue of the IDS_Status updating messages. In other words, the IDS of a node processes raw summary information of its own activity as well as that of all other node in its neighborhood. Again, either periodically or triggered by some event, a node may vote about the security status of a neighbor node, say of node A. If five nodes comprise the neighborhood of A, five votes regarding the security status of A will be cast, one from each node in the neighborhood. Since the votes reflect IDS decisions that process the same raw information, each vote cast carries the identically valid evaluation of the security status of A. Therefore, even if A has in the meantime been compromised and taken over, and tells a lie in the vote regarding its own security status, the other four votes will reveal the truth of the security breach (majority voting). These votes will be transmitted within the neighborhood (MN message). Thus, each node in the neighborhood will conclude that A has been compromised. This result of the compromise of A may now be broadcast (M message) from each neighborhood node throughout the MANET, resulting in countermeasures against A.

2.4 Local Collaborative Groups (LCG)-Majority Voting

This is called Local Collaborative Groups (LCGs)- based majority voting. By this way, each neighbor node arrives at a decision for the node being voted on. If the decision is that the node is compromised, each of the neighbor nodes will broadcast throughout the whole network a message: Delete (x) (*M* message). When the other nodes in the networks receive a majority carrying that message, they take appropriate countermeasures, i.e., they may update their routing tables to change the routing topology, thus isolating and routing around the offending node. From this example, it can be seen that by locally regulating messages with the global law, while working together with other security modules, security recovery can be implemented effectively in MANETs.

In the following sections, we describe this LCG-based security recovery policy in more detail.

When we are developing a security recovery policy, it is assumed that an intrusion detection system (IDS) agent is already installed in each node. The function of the IDS agent is to monitor the host node in which the IDS agent resides, as well as its neighbor nodes, and to detect any intrusion or anomaly occurring to all these local nodes that comprise the neighborhood.

Each host and its neighbor nodes comprise the Local Collaborative Group. When we are speaking of "neighbor nodes", we mean they are the most recently used or most common used nodes, which are within the radio range of the host node. There are two ways to define and generate the neighborhood of each node. First, a node can construct its neighborhood directly from the MAC layer, assuming that the neighborhood can be well defined according to the MAC protocol employed. The second way is to generate

the neighborhood from the routing table. Here, the traffic intensity will affect the size of the neighborhood that will be generated.

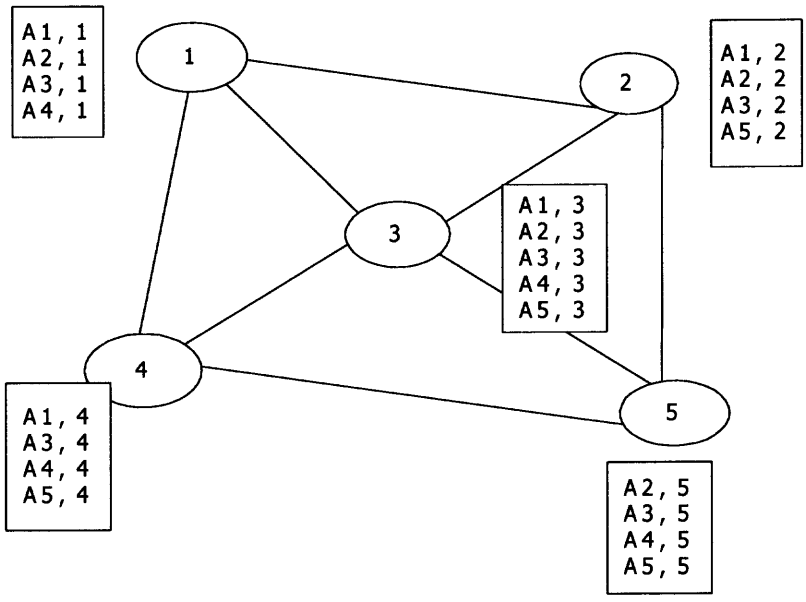


Figure 2.2 The concept of Local Collaborative Group (LCG) voting.

Moreover, the question of arriving at a reliable IDS decision, in the face of attacks, needs to be addressed. The answer, in general, lies in collaborative decision making for an IDS outcome determination. The IDS at each node in a neighborhood can solicit the input of its neighbors; the totality of the findings of the responding nodes will be weighed by majority voting and then the outcome of this vote will be broadcasted to all nodes. A compromised node has no motive to declare attack falsely, in the case of no attack; however, in the case of a network attack, found by the neighbors, it will be outvoted, so that the correct decision will be arrived at.

Figure 2.2 shows a sample topology used for LCG voting. In Figure 2.2, Ax,y means the intrusion detection decision made by the IDS system in Node y, according to

trace data that node y monitored at Node x, which means " Node y says, I am monitoring my neighbor node x, and now I conclude that it is normal or under attack or compromised, etc.". The IDS system in each node monitors its host node as well as its neighbor nodes, and maintains a table of intrusion detection decisions for itself and its neighbor nodes. This is the method of Collaborative Local Intrusion Detection (CLIDE). During the operation of CLIDE, the IDS agent on each node may operate and report in an asynchronous or synchronous mode (periodically). Suppose the IDS reports periodically, and AODV is used as the routing protocol. Our security policy is stated as:

- Each node sends its agent (state information) to all its neighbors with period T1.
- A node has no incentive to falsely volunteer that it has been attacked successfully. Thus, if a node declares itself to be the victim of an attack it should be accepted at face value and declared to be so. This would occur when a node has strong evidence of attack (security level code: SecCon 3), against itself, thus this node should be isolated from the network. This situation may arise for only some of the attack types.
- When CLIDE reports "suspicious" activity (security level code: SecCon 2) at a node, for example, Node 1, then either Node 1 or one of its neighbors sends Voting_Request (node 1) messages to all the neighbor nodes of Node 1.
- All neighbor nodes have Node 1's agent residing locally, within each and every neighbor node. So, finally, every neighbor node has the following information: A11 (new), A12, A13, A14. Because our policy is global, every node can locally calculate the voting decision in identical manner, according to some generalized threshold condition. If only a minority of the nodes vote in favor of Node 1 being malicious, then Node 1 is benign. If a majority of the nodes vote for the node as malicious, Node 1 is deemed malicious, and steps should be taken to isolate it from the network. If it is a tie, then repeat updating the nodes in this neighborhood with period T2 (T2<T1) until a lowest-period threshold is reached or until the tie is broken. After the period threshold is reached, still with a tie, classify Node 1 as malicious.

2.5 MANS Enforcement

Law L of policy P determines that treatment of p-messages is defined by specifying what should be done when such a message is sent, and when it arrives.

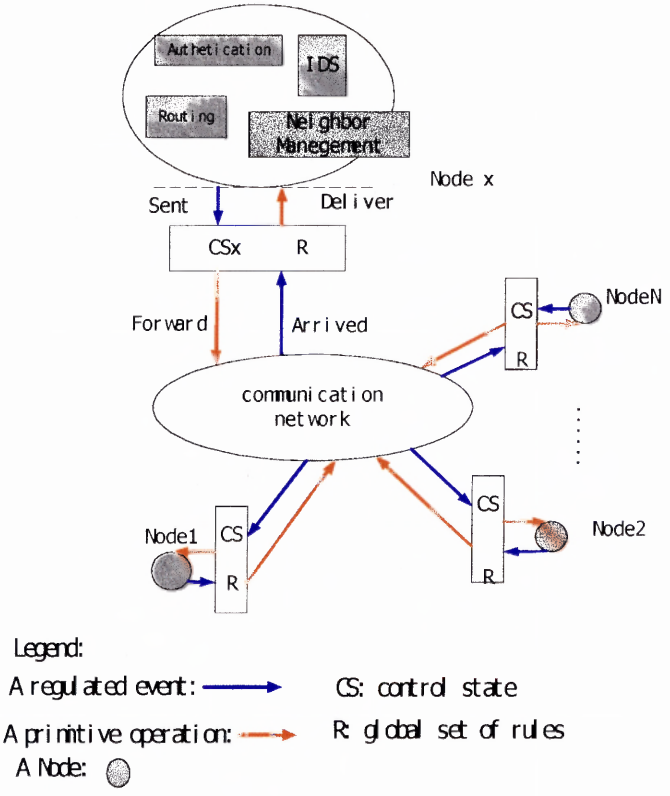


Figure 2.3 Enforcement of MANS.

Furthermore, the law L is constructed as a pair <R, CS>, where R is a fixed global set of rules defined for security recovery policy, and CS is a mutable set called control states. Each node must be in one of the control states. The main role of control states is to enable L to distinguish between different kinds of members, so that the ruling of the law for a given event may depend on its home. In our recovery policy, we define control states as levels of security: SecCon1 means the node is benign, SecCon3 means the node is compromised, while SecCon2 means the node is suspicious.

Our Security recovery policy $P = <L, M, G, N, M_N>$ is enforced as follows: there is a controller associated with each node of group G, logically placed between the node and the communications medium, and linked with other modules such as the Authentication Module,

IDS agent, routing Module, Neighborhood Module, and so on. All controllers have identical copies of the global set of rules R, and each controller maintains the control states of the nodes under its jurisdiction. As discussed in the above section, every controlled-event associated with a **P-Message**, that occurs at a given node x of the MANET, is intercepted by the controller of x. The controller will undertake an operation of that event according to the ruling and the current control state. Because all controllers have identical copies of the law, the law is said to be global to the whole network. It is noted that each controller, in fact, only controls the messaging, while specific operations about security recovery are implemented by other modules. In other words, the main function of the controller is to intercept messages received from controllers in other nodes, to generate new messages needed or to forward messages to the right modules according to the embedded identical rule R. The enforcement is shown schematically in Figure 2.3.

2.6 Experimental Results

In this section, we present experimental results of simulation experiments to show how security recovery policy, described in Section II, works and to investigate its performance. Network Simulator 2 (NS2) was used as the simulation tool.

2.6.1 Topology

The topology has 670m-to-670m width-to-length ratio for the rectangular box constraining node movement. The chosen traffic patterns were generally a set of low rate, long-lived, constant bit rate connections between randomly chosen source-destination pairs. Each run contained 50 nodes with maximum transmission radii of 250m, creating a densely connected topology, which never partitioned during any of the simulations. Node movement rates were varied from static to highly mobile.

2.6.2 Neighborhood Management

There are two ways to generate the neighborhood of each node. First, we may get the neighborhood directly from the MAC layer, assuming that the neighborhood can be found out well in the simulation. The second way is to generate the neighborhood from the routing table. Doing the latter, the neighborhood nodes that can be generated are limited by the active nodes at hand (i.e., the traffic script that we are using).

Output from the MAC layer: Generally, each node has on average 10-20 neighbor nodes that are 1-hop away, and about 20 neighbor nodes 2 hops away.

Outputs from the routing table: Getting neighbor nodes from the routing table is an effective method in practice. For example, using the DSR routing protocol, each time when a node has a packet to send, the DSR routing protocol needs to calculate the current path to that destination. The next hop in this routing path is one of the neighbor nodes.

2.6.3 IDS Models

In our security recovery system, it is assumed an IDS is installed in each node, which can monitor the behavior of host nodes and neighbor nodes. In order to define our IDS model, we first define the following parameters:

PRIOR: the prior probability of each node to be attacked and compromised

FN: Total number of undetected attacks/Total number of true positives (attacks)

TN: Total number of correct normal instances/Total number of normal=1-FP

FP: Total number of incorrect normal instances/Total number of normal

TP: Total number of correctly detected attacks/Total number of true positives (attacks) =1-FN

FP_R: FP output of the recovery system

TP_R: TP output of the recovery system

In other words, FP, FN, TP and TN stand for False Positive, False Negative, True Positive and True Negative rates, respectively.

The IDS model we are using in this simulation is defined by seven levels: Namely, FP=0.002, 0.005, 0.01, 0.02, 0.05, 0.1, 0.2; and correspondingly TP=0.2, 0.3, 0.4,0.5, 0.6,0.7, 0.8, where the low FP is paired up with a low TP rate (i.e., high FN) rate. In other words, (FP=0.002, TP=0.2), (FP=0.005, TP=0.3) .... (FP=0.2, TP=0.8), are the seven operational states of this particular IDS model.

2.6.4 Simulation Scenarios

Next, we assume that the host IDS and neighborhood IDSs are operating independently. That means, the neighborhood IDSs will judge the target nodes by factors they observe by themselves (such as the target's traffic variation, routing updating, and so on), as well

as on reports from the target of summary raw information. We also assume, when voting, that we treat reports from the host IDS or from neighborhood nodes exactly the same, in other words all carry the same weights in contributing to the decision.

When we are simulating IDS Agents, each node would be randomly chosen to be under attack or not (by the PRIOR probability value), while during the detection stage, its own IDS and the neighborhood IDS would randomly correctly or incorrectly (in the case of a normal condition, by the FP probability; in the case of an attack, by the TP probability) report its own anomaly state. This is our first scenario, **IDS scenario 1**, in which case the host IDS will not deliberately lie to report its state.

We also consider the second scenario, that of a lying compromised node: if a node, say node 7, is under attack, its own IDS should have shown positive (attack), however it reports to its neighborhood node members negative (that is it lies saying it is not under attack). The question then becomes, if a node is compromised but lies about its condition, what is the probability that the neighborhood will report the truth (that node 7 is under attack and has been compromised, despite the lie of node 7), and what is the probability that the neighborhood will report negatively (that node 7 is not under attack, thus being fooled by the lie of node 7).

Thus, in **IDS scenario 2**, we discuss the results received of simulation experiments operating under the second scenario. Here the false negative is a deliberate lie instead of a random choice, as it was in the first scenario. The probability of the lie being discovered by the neighborhood was computed.

MANET networks may operate in a hostile environment, where the adversary who compromised a node will deliberately cheat the other normal nodes in order to avoid

detection. Thus, the host IDS on each node cannot be trusted. In this case, our LCG-based recovery policy is necessary, and in **IDS scenario 2**, we discuss its performance.

2.6.4.1 IDS Scenario 1. In this scenario, the host IDS honestly reports the results of its decision, which is randomly controlled by TP and FP. The results are shown as Figure 2.4.

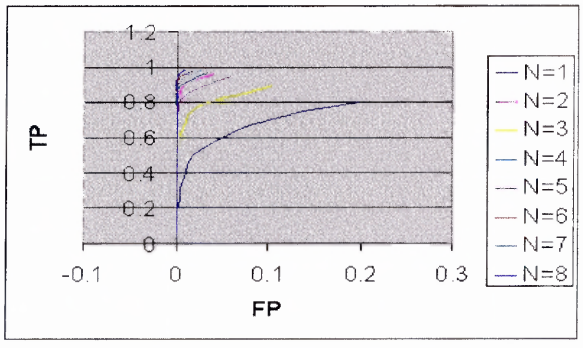


Figure 2.4 ROC curve in Scenario1.

From Figures 2.4, we could see that the larger the neighborhood number, the better the performance will be. When N=8, the detection rate reaches 0.99 while false alarm rate is 0.01(whereas the original detection rate is 0.8 and the false alarm is 0.2).

Moreover, the performance is slightly better when N (the total number of neighborhood nodes and host) is even than odd. For example, the performance when N=2 is better than that of N=3, the performance when N=4 is better than that of N=5, and so on. That is because of our chosen method of dealing with a tie when voting. Especially, the performance when N=2 is noticeably good because a tie in this case occurs very frequently. When N increases, the influence of a tie decreases, the performance difference between a sizeable even and next odd number of nodes (say, 10 and 11 nodes) decreases.

2.6.4.2 IDS Scenario 2. In this scenario, the compromised host will lie to its neighbors, falsely reporting no attack, when in fact there is one.

From Figure 2.5, 2.6, we could see that in this scenario the host IDS cannot be trusted: the detection rate is always 0 even when the original IDS is working efficiently. When N=3, the recovered ROC curve is under the original IDS curve, that means the performance is not be improved here either.

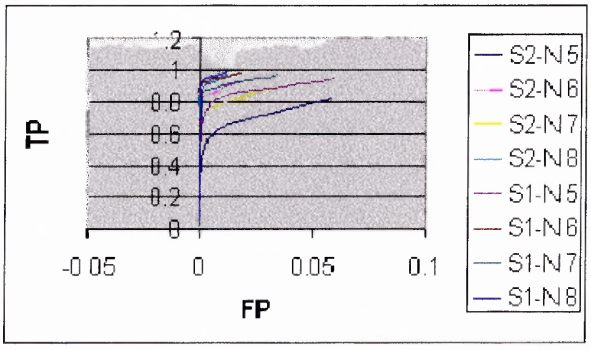


Figure 2.5 ROC curves in Scenario 2.

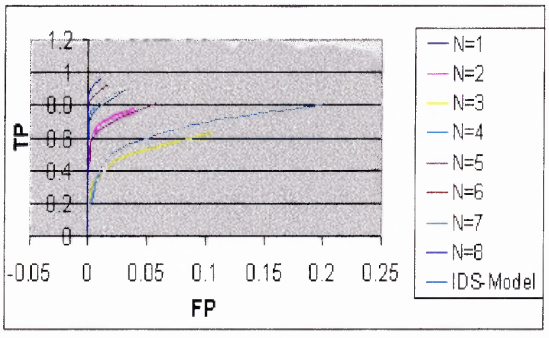


Figure 2.6 Comparison between Scenario 1 and 2.

When N is larger than 4, the recovery policy works better than the original IDS. When N=8, the recovered neighborhood detection rate reaches 0.969 while false positive alarm rate is 0.0113 (when the TP and FP of the node IDS is 0.8 and 0.2). Because when a node is attacked, it will deliberately lie, this also means that the probability of detecting

a lie is 0.969 with a false alarm rate of 0.011; the corresponding values, when the attacked node does not lie, are 0.990 and 0.0107. This is an encouraging result.

2.7 Conclusion

This chapter presents the architecture of the Mobile Ad-hoc Network Security (MANS), a novel system that provides security to Mobile Ad-hoc Networks (MANETs) and investigates its effectiveness. MANS is based on a "neighborhood watch" concept. It builds a fully decentralized scalable security policy that is Law-Governed globally using only local actions. MANS formally prescribes a Local Collaborative Group function by defining neighborhoods, their states, and neighborhood-wide majority voting decisions. It utilizes these concepts in developing the security recovery policy, including specification, implementation, and enforcement.

MANS has been tested successfully with simulation experiments; the results presented here cover the case of an attacked but honest node as well as that of a compromised dishonest node. In both cases, it is shown that MANS identifies the attacked and /or compromised node, requiring only a modest size neighborhood to accomplish it.

CHAPTER 3

LOCALLY-ENFORCED GLOBAL TRUST AUTHENTICATION SCHEME

3.1 Introduction

Confidentiality, authentication, integrity, non-repudiation, access control and availability are considered as the main services of a security system. Among these services authentication is probably the most important security service required to achieve other services. There are various means to provide authentication, however, cryptography is one of the most widely deployed and most effective way of providing this service. When the subject of cryptography is touched even slightly, management of cryptographic keys becomes an issue. The primary question any key management system tries to solve is - 'How do you trust that a key is authentic enough to use?' The bottom line is that a key management system must ensure a user that a key it receives is authentic and it can use this key for cryptographic purpose. From this point of view key management can be considered a synonym of trust and security in the whole system.

This dissertation designs a scalable, robust, strong authentication infrastructure which fits the requirements of specific mobile ad-hoc networks: 1) whose node number is extensible to up to thousands; 2) whose nodes maybe deployed in hostile environment, has high security concern; 3) whose nodes have significant computation power. None of related works about authentication for MANET [6, 11, 16, 17, 25], etc. fits all this requirements and conditions at the same time. We will discuss in more details in this paper.

The security architecture in MANET is proposed in chapter 2, which includes modules such as routing, authentication, IDS and so on. In this chapter, we describe a

novel authentication paradigm and algorithm, called Locally-Enforced Global Trust (LEGT), for high-value transactions in mobile ad hoc networks (MANET). Our work is based on these requirements of scalability, robustness, strong security, motivated by and grounded in previous research work accomplishments that include threshold cryptography and cluster-based routing and communications in ad hoc networks. We provision and support authentication services by adopting and adapting certificate-based approaches. We are using threshold sharing [12,31] of the certificate signing key (or *"space diffusion"*) within each cluster to distribute the certificate services, and employing trusted dealer only at the initialization stage (bootstrapping) for the purpose of global trust.

To enhance the security of our scheme, we have adopted, adapted, and improved the salient aspects of proactive security [21, 22], and publicly verifiable secret sharing [23] security schemes. "Proactive security" is applied and embodied in out LEGT system to provide enhanced protection against a *mobile adversary* to long-lived cryptographic keys that are already distributed (i.e., protected by *"space diffusion"*). In fact it adds protection by "*time diffusion*" as well. Namely, given a shared function as described above, it adds a periodic refreshing of the contents of the distributed shareholder. Therefore, the knowledge of the mobile adversary obtained in the past from compromising at most the allowed k-1 numbers of shares is rendered useless for the future. Verifiable secret sharing is also applied to publicly verify the correctness of received secret share updating. Besides allowing us proactivization and allowing us to securely change a threshold sharing scheme by changing the threshold parameter (from *t*-out-of-*l* to *t*-out-of-*t* and vice versa), they allow us to change *t* and *l* dynamically as a

system management tool. Here t is the threshold number of nodes that an adversary must corrupt in order to succeed, while l is the total number of nodes that may comprise the network infrastructure and support the network communications.

In this chapter, an example of the adoption of the general design of the Cluster Based Routing Protocol, (CBRP), which is one of the IEFT's candidates for a MANET routing protocol standard, is also presented. We have implemented our algorithm in our state-of-the-art MANET scalable emulation testbed (MASE), and performed evaluations such like in the MANET environment.

The rest of this chapter is organized as follows. Section II presents related works. Section III discusses architecture of our proposed algorithm LEGT. Section IV presents embedding Proactive Secret Sharing / verifiable secret sharing in our LEGT design. Section V presents cluter_based authentication and certificate renewal/revocation. Section VI discusses the implementation and evaluation of our design in networks applying CBRP as routing protocol. Section VII evaluates our model in means of security enhancement, computational and network requirements. Section VIII concludes the paper with some important remarks that were due to the work being presented.

3.2 Related Works

Key management has become a synonym to authentication management. At the same time, cryptosystem is a means of achieving a lot of security services, which, in turn, help achieve several security objectives. In this section we concentrate on existing authentication algorithms in an ad hoc network. Four published methods are discussed

here. Some aspects of these methods, deemed useful in practice, have been adopted and adapted in our scheme.

3.2.1 Certificate Chain

Self-organized PKI is a key distribution scheme proposed by the Terminodes [6] group. This scheme is very similar to PGP as it employs meshed PKI architecture but the certificates are distributed by the hosts themselves, not through a certificate directory as in the case of PGP. This scheme requires each host to maintain a small database of certificates selected using some algorithm. When a host wants to obtain the public key of another host they merge their local certificate databases and try to find an appropriate certificate chain to the other host from this merged database. This scheme provides a probabilistic guarantee that the certificate chain will be found in the merged database.

3.2.2 Threshold Cryptography

Zhou and Haas in [11] suggest using threshold cryptography as the means of distributing trust in ad hoc networks. In [17], Lu and Luo extended above threshold cryptography way. They present an authentication service whereby the public-key certificate of each node is cooperatively generated by a set of neighbors based on the behavior of the node as monitored by the neighbors. Using a group signature mechanism based on polynomial secret sharing, the secret digital signature key used to generate public-key certificates is distributed among several nodes. Certification services like issuing, renewal and revocation of certificates are distributed among the nodes: a single node holds just a share of the complete certificate signature key. The authors propose a *localized trust model* to

characterize the localized nature of security concerns in large ad hoc wireless networks. When applying such trust model, an entity is trusted if any k trusted entities claim so: these k trusted entities are typically the neighboring nodes of the entity. A locally trusted entity is globally accepted and a locally distrusted entity is regarded untrustworthy all over the network. Certificate issuing/renewal, revocation, secret share initialization are also defined in [17].

3.2.3 Direct and Recommendation Trust

In [16], authors present a Light-Weight Authentication Model which is strongly based on human behavior when dealing with trust since the human society itself can be seen as an ad-hoc network. This model is similar to *Distributed Trust* Model [27] to establish trust relationships and extend it by a request for references. Each entity maintains a repository of trustworthy entities such that a path between two arbitrary entities can be found by indirectly using the repositories of other entities. *Distributed Trust* Model [27] is a decentralized approach to trust management and uses a recommendation protocol to exchange trust-related information. The model assumes that trust relationships are unidirectional and taking place between two entities. The entities make judgments about the quality of a recommendation based on their policies, i.e., they have values for trust relationships. Also trust is not absolute, e.g. an entity can change the trust value it received as a recommendation. The policy is not communicated, so it might not be understandable for other entities, which prevents it from being misused. The recommendation protocol works by requesting a trust value in a trust target with respect to a particular classification. After getting an answer an evaluation function is used to

obtain an overall trust value in the target. The protocol also allows recommendation refreshing and revocation. To do so the recommender sends the same recommendation with another recommendation value, or a neutral value to revoke. The model is suited to establishing trust relationships that are less formal, temporary, or targeting ad-hoc commercial transactions.

3.2.4 Cluster Based Key Management

In [25], authors introduce an end-to-end data authentication scheme where authors have attempted to minimize the encryption so that computational complexity is kept low. Detailed steps for countering repetition based attacks can be found in [25]. For authentication only, each node is given a *system public key* and *system private key* when it joins the network. This pair of keys is shared by all the nodes of the network. Besides the system key, each node also needs a *cluster ke*y. This *cluster key* is unique to every cluster and a single cluster key is shared by all the nodes belonging to a cluster. This key is generated by the cluster head and distributed to all the cluster members. This key is encrypted with the *system public key* and broadcast by the head. Each cluster head also has a unique pair of public/private key called *head ke*y. This private key is known only to the head that generates it. The corresponding public key is known to all the network nodes. This is done by means of a network wide broadcast that is initiated by each head immediately after it gets elected as the leader.

3.3 Architecture of LEGT

This work is motivated by and grounded in the related works, briefly described above, which have their own pros and cons or suitability in different environments (large or small networks, expensive or inexpensive transactions, highly requirement on security or not, as so on). At the same time, we have examined and employ cluster-based routing and communications; such methods have been found effective in large, scalable networks, and have been devised to minimize the overhead of flooding of route discovery packets in the whole network. As mentioned, our algorithm is called Locally-Enforced Global Trust (LEGT). We provide an authentication service by taking a *certificate-based* approach. We are using *threshold sharing of the certificate signing key* within each cluster to distribute the authentication service (Locally-Enforced), and employing the trust authority only at initialization stage to achieve authentication between clusters (Global Trust). Certificate services, key sharing functions and cluster membership control are also defined in our algorithm.

Since the authentication strategy presented here is for a hierarchical architecture, a cluster-based network has been used. We have therefore discussed clustering in the next subsection. Then we compare our algorithm with other related works and then outline the basic ideas of our Locally-Enforced Global Trust. Our assumptions, design criteria, and detailed strategy have been put forth in the subsequent sections.

3.3.1 Overview of the Cluster-based Approach

The cluster-based architecture was devised to minimize the flooding of route discovery packets. One of routing protocols that use this hierarchic architecture is the *Cluster Based Routing Protocol*, CBRP [24]. This kind of architecture is most suitable for large

networks with many nodes. The entire network is divided into a number of either overlapping or disjoint 2-hop-diameter clusters as shown in Figure 3.1. A cluster head is elected for each cluster to maintain the cluster membership information. A cluster is identified by its cluster Head ID. Each node in the network knows its Cluster Head(s) and therefore knows which cluster(s) it belongs to (in the case of overlapping clusters, a node could and often would belong to more than one cluster). A node regards itself as in cluster X if it has a bi-directional link to the head of cluster X. In the current implementation of CBRP, the node with the lower node ID is elected as the cluster head. All the nodes broadcast HELLO messages periodically. The hello messages also contain tables carrying information about the neighboring nodes and adjacent clusters. These HELLO messages are useful for maintaining up-to-date 2-hop topology.

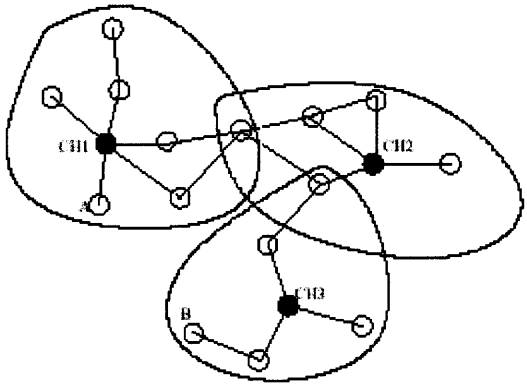


Figure 3.1 Example of a network divided into clusters; trust is enforced in each cluster; Cluster Heads 1, 2 and 3 (CH1, CH2, CH3) use and support certificate chains.

An in-depth study of Cluster Based Networks has been made [24]. We however focus on devising and studying the authentication scheme that is most optimal for such hierarchical architectures. In implementation of our authentication scheme, we will show

how to piggyback our authentication information to the routing message (such as HELLO message) if possible.

3.3.2 Comparison with Other Schemes

The self-organized PKI presented in [6] is scalable to large networks according to the "small world" theory, with high probabilistic guarantee that the certificate chain will be found in the merged short local repository. But, the authenticity of the certificate obtained by this scheme relies on the assumption that trust on a user's authenticity of its certificate means also trust of this user's authenticity of issuing a certificate. This level of trust is often questionable, and in doubt, especially in Ad hoc networks. Moreover, if even only one of the users in the whole chain is misbehaving in issuing certificates, the whole chain is affected, and the certificate merged by this chain is not authentic. It is expected that the larger the network, the higher the risk. In [6], authors use an authentication metric to represent the assurance with the authentic public key by using the chain of certificates. But no active method of enhancing the authenticity of the resulting certificates is implemented. Moreover, certificate renewal and revocation are left to be defined.

In our scheme, we employ the cluster-based technical approach; here all users in one cluster can share one cluster key. So the total number of key are dramatically reduced. Moreover, in our scheme, certificate renewal and revocation are specifically defined.

Authors of [21] apply the "shared secret" to achieve "*localized trust*". Certificate issuing/renewal, revocation, and secret share initialization are basically implemented locally. However, this approach present here assures a locally trusted entity is globally

accepted and a locally distrusted entity is regarded untrustworthy all over the network. A very large, heterogeneous MANET may consist of users from different companies, universities or troops with different functions, security levels and so on; in such settings it is hard for a user to fully trust another user signed by a group of remote, never-heard-of units; or in a military application, more than k nodes in one area may be compromised and all certificates jointly signed by these nodes are no longer authentic. Moreover, this scheme distributed system secret key through the whole network, making security services such as, secret sharing verification and updating not scalable.

In our scheme, the secret shares distribution, verification, certificate renewal, and revocation are implemented within each cluster, which enforces that every user who gets a valid certificate or signature is truly locally entrusted. When authenticating a user from a different cluster, a cluster certificate which is signed at initialization stage will be used to check the authenticity of the cluster to which this user belongs. At the same time, our scheme is highly scalable implemented with cluster routing protocol (in this paper, CBRP).

In [25], authors introduce an end-to-end data authentication scheme where authors have attempted to minimize the encryption so that computational complexity is kept low. Detailed steps for countering repetition based attacks can be found in [21]. For authentication only, each node is given a *system public key* and *system private key* when it joins the network. This pair of keys is shared by all the nodes of the network. Besides the system key, each node also needs a *cluster key*. This *cluster key* is unique to every cluster and a single cluster key is shared by all the nodes belonging to a cluster. This key is generated by the cluster head and distributed to all the cluster members. This key is

encrypted with the *system public key* and broadcast by the head. Each cluster head also has a unique pair of public/private key called *head key*. This private key is known only to the head that generates it. The corresponding public key is known to all the network nodes. This is done by means of a network wide broadcast that is initiated by each head immediately after it gets elected as the leader. In this scheme, *system public key* and *system private key* are used for mutual authentication and Cluster Heads are used as CAs. Use of the *system public key* and *system private key* needs "bootstrap" from some authorities.

Two problems about employing the Cluster Heads as CAs need to be addressed: firstly, authors assume all the nodes of the network mutually trust each other, so public keys are just broadcast through the network without any binding of public keys and IDs (such as certificates); secondly, because a Cluster Head is used as CA for the whole cluster, if the Cluster Head is attacked, the whole cluster fails.

In our scheme, we also use the cluster-based approach, but we are also using certificates for binding a cluster's ID and a cluster's public key, as well as threshold cryptography for fault tolerance, which is crucial in high-value MANET networks.

3.3.3 Assumptions for LEGT:

- For high-value transactions or military applications
- Nodes with significant computing power
- During any time interval between certificate updating, the adversary cannot break or control k or more nodes in each group.
- Each node i has a unique nonzero ID *Vi*, such as its IP address.

- Each node has some one-hop neighborhood discovery mechanism or group membership management, which are provided by underlying routing protocol.
- Communication between one-hop neighboring nodes is more reliable compared with multi-hop packet forwarding, since the exposure to wireless channel error and interference is limited and no ad hoc routing is involved.
- Each node has at least k one-hop legitimate neighboring nodes, where k and n are locally decided by local group and transparent to other groups.
- Each node is equipped with some detection mechanism to identify misbehaving nodes among its one-hop neighborhood.

3.3.4 The Locally-Enforced Global Trust (LEGT) Paradigm

We present here the distinctive and advantageous aspects of the Locally-Enforced Global Trust (LEGT) authentication service. As mentioned, our scheme is motivated by and grounded in the above three schemes but employs, improves and integrates their advantageous aspects while avoiding their drawbacks. In our scheme, we still use secret sharing, but in a way different than in [17] where authors distribute the global private key SK that is used to sign certificates into each node of the network by a polynomial of order k-1. Here, our MANETs are composed of clusters or groups. These clusters or groups maybe just neighborhood nodes, maybe clusters when using hierarchical routing protocol such as CBRP or ZRP. In practice, they could be different organizational components or troop groupings. Each group or cluster has a specific group key pair; a threshold scheme <k, n> is used within the group to support a multi-signature scheme or certificates for nodes in this group. Within each group, we have two options:

- Each node has his own key pair and a threshold scheme <k, n> is used to co-sign the certificates;

- Each group has only one common public key (in order to reduce the size of the public key directory), instead of each node in the group having a key, and any subset of k individuals can sign and thus generate a valid signature.

We use the multi-signature scheme presented in [20] in an essential way. Each individual has to get a signature or certificate from the group she belongs to before she can mutually authenticate with any other individual in other groups, while the authenticity of group key pairs are certified through a trusted third party in initial (bootstrapping) phase. Compared with [6], which greatly increases certificate path length and the size of local certificate repositories when network size increasing, we reduce the security risk and computation resources which makes our scheme more scalable. At the same time, we enhance the security compared with [17] because certificates are signed locally within each cluster and other clusters check the authenticity of a cluster before checking the node's certificate signed by it. Certificate renewal and/or revocation are implemented locally.

LEGT scheme is outlined as follows:

- LEGT is using the RSA-based schemes to generate the private and public key pairs, which are used for end-to-end security to realize cipher key exchange, message privacy, non-repudiation, and most importantly in our scheme, for authentication. Since RSA is slow and costly, we are using AES for block encryption and decryption after a security channel has been set up and the AES key has been distributed through the RSA scheme.

- A Trusted Authority (TA) is needed only the initialization stage (bootstrapping). TA generates its *system public key/system private key. System public key* is broadcasted to each node while the bootstrapping and *system private key* is kept secretly by TA.

- For the initial authentication of new clusters and for judging their trustworthiness for other clusters which hasn't trust relationship with each other, TA signs cluster certificate for every *cluster public key.* While corresponding *cluster private key* is shared by all the nodes belonging to that cluster by Shamir's scheme [12];

- Certificate for each node is issued/renewed locally – by other neighbor nodes or within the cluster (according to the routing protocol the network is using) The new certificates or updating certificates must be "locally trusted" through threshold cryptography ;

- When a user of one cluster (say, Cluster1) wants to mutually authenticate with a user of another cluster (Cluster3), first they show each other the certificates or signature signed by their cluster. Then Cluster certificate will be checked to trust each other's group.

- A user who cannot be trusted by his neighborhood or cluster cannot get a certificate or signature of his cluster, which means he cannot get trust anywhere in the whole network; When a cluster fails to show valid cluster certificate signed by TA at the first time communication or its certificate expired or revoked, all nodes in such cluster cannot be trusted anywhere throughout the whole network.

- To enhance the security of our scheme, we are also using the salient aspects of the following security schemes: certificate renewal/revocation, proactive secret sharing [21], [22], and publicly verifiable secret sharing [23].

In the following two sections, we define our *Locally-Enforced Global Trust* algorithm in further detail. We first define the different key types that are used and the method adopted for the distribution of these keys. Then, we describe secret share dealing (including sharing, verification and updating), membership management, certificate services, certificate repositories construction, and last, authentication steps.

3.4 Proactive, Publicly Verifiable Secret Sharing in LEGT

The mobile hosts in MANETs can be physically captured and compromised. Therefore, the authentication architecture should avoid having any central entity of control, so that compromise of any one (or in fact several) entity does not risk the whole network. Thus, the fundamental philosophy of approach for security in our system, the LEGT, is to rely on local, cluster-based, provisioning and maintenance of security functions, and in particular of authentication and trust. In other words, it is expected that resilience against

compromised nodes can be achieved by distributing the functionality of the centralized certificate authority (CA) servers among a group of servers or nodes.

There are several recent works on security in wireless networks. Some propose a Kerberos-based authentication scheme for mobile users in wireless cellular networks. Others directly apply the threshold secret sharing and proactive secret share update techniques in a group of "special nodes" to increase service availability and robustness. Instead of following the conventional client-server model, we suggest a peer-to-peer approach to maximize availability and facilitate localized communication.

Distribution of security functions has been a very active research area in the literature; of particular interest is security function sharing, where threshold secret sharing serves as a basic primitive. The focus of these works has been to maximize the security of the shared secret in the presence of possible compromises of the secret share holders.

Our scheme is motivated by these works, but extends the idea further in an attempt to minimize the effort and complexity for mobile clients to locate and contact the service providers. We devise scalable algorithms and protocols to enable the distribution of the certification services into *every node*. There is no differentiation of servers and clients any more, in our architecture: a threshold number of any nodes can collaboratively act as servers to provide services for other nodes.

The underlying techniques we design are fundamentally "share of shares" methods which change the representation of a function. *Proactive secret sharing* can further improve robustness via periodic secret share updates. To enhance the security of our scheme, i.e., the LEGT, we have adopted, adapted, and improved the salient aspects

of proactive security [21, 22], and publicly verifiable secret sharing [33,34] security schemes. Besides allowing us proactivization and allowing us to securely change a threshold sharing scheme by changing the threshold parameter (from t-out-of-l to t-out-of-t and vice versa), they allow us to change t and l dynamically as a system management tool. Here t is the threshold number of nodes that an adversary must corrupt in order to succeed, while l is the total number of nodes that may comprise the network infrastructure and support the network communications.

3.4.1 Background, Definitions, and Descriptions

3.4.1.1 The RSA Algorithm. The RSA [32] function is a building block for many secure systems where the adversary is allowed to initially access the system and get results based on various restrictions on the input (i.e., chosen plaintext encryption, random plaintext to be signed, etc.). Secure systems typically allow access to random instances (i.e., allowing sampling RSA as a one-way function), thus we assume here that allowed inputs are random.

The RSA key generator G produces two large random primes $p=2p'+1$ and $q=2q'+1$ (p', q' are primes), and computes $n = pq$, Euler function $\phi(n) = (p-1)\cdot(q-1)$ and $\lambda(n)=2p'q'$. To compute the secret key, the generator chooses a random d such that the gcd $(d, \phi(n)) = 1$. The public key is n and e where $ed \equiv 1 \bmod \phi(n)$.

The one-way (hard) direction of RSA (used in decryption or signature of a message m) is $S_m \equiv m^d \bmod n$, whereas the public (easy) direction (used in encryptions and verification of signature) is $(S_m)^e \bmod n$ which gives m.

3.4.1.2 Threshold Cryptography. Threshold cryptography (also called "space diffusion") provides for increased availability of authentication, especially important in MANETs where nodes may operate in a hostile environment. Threshold cryptography, in particular public-key function sharing, embodies the concept of secret sharing [19, 20] (where a shared secret is recoverable only from a threshold of shareholders). It provides for increased availability of the function service (signature, decryption, etc.) by distributing fk(.) to a set of l nodes such that for any valid argument m, any set of at least a given threshold of the nodes can compute fk(m). For security, on the other hand, an adversary who controls less than a given threshold t of nodes and can hear all of the public messages, obtains no computational advantage in computing fk(m') for a target message m' as compared to an adversary attacking the centralized function fk (.) with the same capabilities. Thus the function is protected by "space diffusion" since an attacker is forced to break the memory of more than a threshold of the function share-holders.

3.4.1.3 Publicly Verified Secret Sharing. Basic threshold secret sharing solves the problem for the case that all players in the scheme are honest [19, 20].However, if a player or some players are dishonest, basic secret sharing schemes will not be in practice because of the following reasons: (i) all participants can not verify validity of their shares from the dealer in the distribution protocol, and (ii) participant can not check validity of their shares from other participants in the reconstruction protocol. In order to resist malicious players, Feldman [23] and Pedersen [39] constructed a new type secret sharing scheme respectively. They are called verifiable secret sharing (VSS) schemes in which the following cases are ture (i) the dealer cannot send invalid shares to some or all of the

participants during the distribution protocol, and (ii)participants cannot submit invalid shares during the reconstruction protocol.

There are two drawbacks in some cases: (i) the participants can only verify their own share, but anybody can not verify that the participants received correct shares. (ii) the participants simply release their shares in the reconstruction protocol, subsequently the released shares may be verified by anybody against the output ofthe distribution protocol.

In order to deal with them, publicly verifiable secret sharing (PVSS) schemes were proposed by Stadler [34] and Schoenmakers [35] respectively. A publicly verifiable secret sharing (PVSS) scheme is a verifiable secret sharing scheme with the property that the validity of the shares distributed by the dealer can be verified by any party; hence verification is not limited to the respective participants receiving the shares. Authors in [35] present a new construction for PVSS schemes, which compared to previous solutions by Stadler and later by Fujisaki and Okamoto, achieves improvements both in efficiency and in the type of intractability assumptions. The running time is O(nk), where k is a security parameter, and n is the number of participants, hence essentially optimal. The intractability assumptions are the standard Diffie-Hellman assumption and its decisional variant.

3.4.1.4 Proactive Security. "Proactive security" provides enhanced protection against a "mobile adversary" to long-lived cryptographic keys that are already distributed, i.e., protected by threshold cryptography or "space diffusion"; it adds protection by incorporating "time diffusion" as well.

Proactive security (introduced in [21]) employs dynamic memory refreshing activated by the nodes and servers to withstand a "mobile adversary." Proactive robust threshold public-key schemes (introduced in [22]) have been designed for and applied to general "homomorphic-type" public-key systems. These systems are optimal-resilience, namely the (arbitrary) adversary can corrupt any minority of servers at any time-period; this threshold is a (trivial) upper bound on resilience, even against a weaker stationary adversary. The techniques here give general optimal-resilience public-key systems, which are "robust threshold" schemes (assuming a stationary adversary), and extensions which give proactive" systems.

As mentioned, "proactive security" provides enhanced protection against a mobile adversary to long-lived cryptographic keys that are already distributed (i.e., protected by "space diffusion."). In fact, it adds protection by "time diffusion" as well. Namely, given a shared function, as described above, it adds a periodic refreshing of the contents of the distributed nodes' memories. This proactive refreshing protocol causes a "t-wise re-randomization" of the local nodes' values which preserves the global key but makes previous contents of any set of less than t out of the l nodes "useless". Therefore, the knowledge of the mobile adversary (representing: hackers, viruses, bad administrators, etc.) obtained in the past from compromising at most the allowed t-1 number of nodes is rendered useless for the future (i.e., this gives the system a *self securing* nature). The proactive systems for cryptographic functions inherit the availability property of threshold schemes.

3.4.2 Adversary Models

For adversaries that attack the system by compromising Mobile Ad hoc nodes, we assume that the underlying cryptographic primitives such as RSA are practically secure in terms of the computation power of the attackers. However, we do allow occasional break-ins through factors such as insecure OS, software bugs and backdoors etc. Several adversaries may conspire into a group. For ease of presentation, we denote such an adversary group by a single adversary.

We characterize the adversaries in the following two models:

Model I: During the entire lifetime of the MANET network, the adversary cannot break or control t or more nodes.

In LEGT, we modify the RSA multi-signature algorithms based on threshold secret sharing to handle the adversary of model I.

Model II: Assume time is divided into intervals of length T. During any time interval, the adversary cannot break or control t or more nodes.

Although at any time constant it cannot break or control t or more nodes, the adversary of model II can choose its victims at each time interval. As time goes on each node in the network can be broken during some time interval.

The adversary is computationally bounded and it can corrupt nodes at any moment during a time period by viewing the memories of corrupted nodes and/or modifying their behavior. The adversary likely bases its corruption decisions on past available message signatures pairs. If a node is corrupted during an update phase, we consider the node as corrupted during both periods adjacent to that update phase. We assume that the adversary corrupts no more than t —1out of 1 nodes in each time period,

where $I \geq 2(t-1) + 1$ (this $(t-1)$-restricted adversary guarantees the existence of a majority of t honest nodes at each period). Our model does not differentiate malicious faults from "normal" node failures (e.g., crashes). We assume that the adversary intruding on a shareholder is "removable" (e.g., through a reboot procedure) when it is detected. The reboot operation is performed immediately when attacks or deviations from the protocol are detected, and it is shorter than a duration of a time period. We also consider special weaker cases of this malicious $(t-1)$-restricted mobile adversary (static vs. mobile, and passive vs. malicious).

3.4.3 The Distributed Defensive Posture: Distributed RSA

Based on discussion above, on the defense side, we employ methods similar to the public-key model of [20]. We assume our group of *I* nodes securely shares cluster private key *k*. Cluster head is dealer with the property that all cluster members are instantly connected to it. We assume that each node has a local source of randomness and that the system is synchronized (and w.l.o.g. that nodes act synchronously). A proactive implementation of the communication model is discussed in [21]. The time is divided into *time periods* which are determined by the common global clock (e.g., a day, a week, etc.). Each time period consists of a short *update phase,* during which the nodes engage in an interactive *update protocol,* at the end of which they hold new shares (in fact, a new sharing) of the secret *k*. After the update, there is a *function computation phase,* in which the nodes perform the intended secret-key operation using their current sharing of *k* on numerous inputs.

In summary, an *optimal-robustness proactive RSA system* we used here has the following protocols: (1) an initialization protocol where shares are distributed,

commitments are also broadcast for public verification; (2) RSA function (signature) application protocol, where the nodes act on a common authorized input and then the combining function generates the final RSA result efficiently; (3) an update phase operation which contains a share renewal protocol and a share loss detection and recovery protocol. We require *correctness* which means that at any point that a function is needed to be evaluated there are enough honest nodes, so that the combination of their shares gives the correct RSA result. We also require *robustness* so that the combining effort even when up to t parties are adversarial is polynomial-time.

We present those protocols in the following sections.

3.4.4 Local Key Definitions and Distribution Methodology

Each node has its unique Node ID *Vi*. Node ID is a string that uniquely identifies a particular mobile node. In CBRP, a node's IP address is used as its ID for purposes of routing and interoperability with fixed networks. Each node has its own *public/private* key pair <*SKi, PKi*>, where *SKi* denotes *Vi*'s private key for decryption and signing, *PKi* denotes *Vi*'s public key fro encryption and verification. The private key *SKi* is only possessed by node *Vi* to decrypt messages that are encrypted by the corresponding public key *PKi* and to sign some messages to generate signature. The public key *PKi* is used for other nodes to encrypt messages toward *Vi*, and to verify a signature that is supposed signed by node *Vi* with its private key *SKi*. To save key space, each node may not need its own public/private key pair and all members in the same cluster share the cluster public/private key by means of multi-signature. The binding of private key *SKi* and node

Vi is approached by certificates signed within each cluster by *cluster private key* described as following.

A cluster consists of a group of nodes. Each cluster has a unique *cluster ID Cm*, *cluster private key* C_{sk}^{m} =<*d, n*>, *and cluster public key* C_{pk}^{m} =<*e, n*>. Cluster private key is shared by all members of this cluster to sign or update certificates of each member. Each node gets her one share of *cluster private key* at bootstrap phase or during networking operation, when she joins a cluster and issued by the joint of cluster member. Threshold cryptography is used when signing multi-signatures or certificates within each cluster, where parameters K and N are decided by each cluster according to their membership and transparent to other clusters.

A Trusted Authority (TA) is needed only the initialization stage (bootstrapping). TA generates *system public key/system private key*. *System public key* is broadcasted to each node while the bootstrapping and *system private key* is kept secretly by TA.

3.4.5 Secret Share Assignment and Management

Node *Vi* gets its secret share of *cluster private key share* P_{vi} at the bootstrapping phase of the network, or the first time it joins one cluster and issued by cluster member. In initial phase, TA acts as secrete share dealer and certificate authority for each cluster.

3.4.5.1 Secret Sharing Distribution. Secret sharing distribution is implemented by following steps:

Step 1: Each Cluster generates its RSA *cluster private/public key pair* $<C_{sk}^{m}, C_{pk}^{m}>$:

C_{sk}^{m}=<d, n> and C_{pk}^{m}=<e, n>, where n = pq and e, d are as prescribed in the RSA algorithm.

Step 2: TA choose a random polynomial $f(x)$ of degree K-1,

$f(x) = d + a_1 x + a_2 x^2 + ... + a_{K-1} x^{K-1}$ such that the shared secret is $f(0)$=d.

Step 3: TA privately transmits each node Vi (i = 1, 2, 3,..., N) within cluster the secret sharing

$$P_i = f(x_i) \bmod n$$

Lagrange interpolation is used by any K shareholders to regenerate the secret by computing $d \equiv \sum_{i=1}^{K} (P_i * li(0) \bmod n)$

Where $l_i(x) = \dfrac{(x - x_1)...(x - x_{i-1})(x - x_{i+1})...(x - x_K)}{(x_i - x_1)...(x_i - x_{i-1})(x_i - x_{i+1})...(x_i - x_K)}$.

Instead of revealing the private exponent d to the coalition, the more secure way is that the certificate requester broadcasts the messages m (here m is generally the hash of message M) for signing, and each shareholder generates its partial certificate with its secret share and sends it back to the requester. The requester can then combine K partial signatures to generate the certificate.

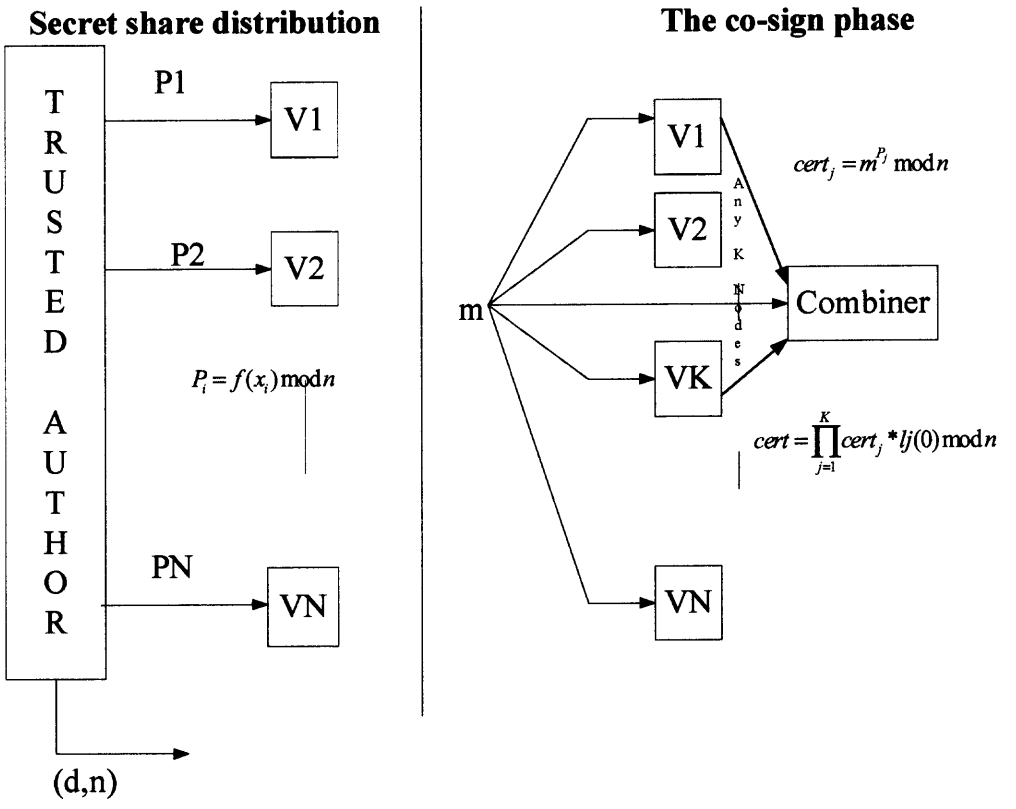


Figure 3.2 Secret share dealing within one cluster.

3.4.5.2 The Co-sign Phase. The co-sign phase is defined by following steps:

Step 1: The certificate requester Vi broadcasts its certificate issuing/update request m within its cluster in the format of {Vi, PKi, Cm, Texpire}, which may read: "node Vi from Cluster Cm is requesting a certificate of its personal public key as PKi which will expire after Texpire".

Step 2: After receiving this request, each node Vj within cluster Cm may check its transaction with Node Vi and decides whether to issue its partial certificate or not. If yes, Vj computes the partial certificate $cert_j = m^{P_j} \bmod n$ and sends it back to Vi.

Step 3: The certificate requester Vi can collect K partial certificates to combine its final certificate *cert* signed by the *cluster private key* C_{sk}^{m}:

$$cert = \prod_{j=1}^{K} cert_j * lj(0) \bmod n$$

We show the above secret share dealing steps in Figure 3.2.

3.4.6 Add Public Verifiability

In the operation of above co-sign phase, if at least one of these K partial certificates is faulty, the result certificate will be invalid. It may be from a broken node that is controlled by the adversary to attack our protocols. It may also be a node that makes a mistake. Therefore, we further add verifications to above secret sharing protocol, such as:

1) In distribution phase, all nodes can verify the validity of their shares from TA.

2) In co-sign phase, each node can verify validity of partial certificate from other nodes, select k valid partial certificates to reconstruct certificate, discard invalid partial certificates and put the nodes which come from to Certificate Revocation List.

3) During above two verifications, each node's partial share should not be released.

3.4.6.1 Model for Non-interactive PVSS. We applied Publicly Verified Secret Sharing (PVSS) as in [35]. As a common structure for PVSS schemes we need consider the following steps. Note that initialization is done without any interaction between the TA and the Nodes. In fact, Nodes may enter or leave the cluster dynamically; the only requirement is that a node holds a registered public key.

The generic model for Non-interactive PVSS is described as following:

Initialization All system parameters, keys are generated as part of the initialization. Furthermore, each Participant Pi registers her public key to be used with a public key encryption method Ei. The actual set of participants taking part in a run of the

PVSS scheme must be a subset of the registered participants. We assume w.l.o.g. that participants P1, . . . , Pn are the actual participants in the run described below.

Distribution The protocol consists of two steps:

1). Distribution of the shares. The distribution of a secret s is performed by the dealer D. The dealer first generates the respective shares si for participant Pi, for i = 1, . . . , n. For each participant Pi the dealer publishes the encrypted share Ei(si). The dealer also publishes a string PROOFD to show that each Ei encrypts a share si. Furthermore, the string PROOFD commits the dealer to the value of secret s, and it guarantees that the reconstruction protocol will result in the same value s.

2). Verification of the shares. Any party knowing the public keys for the encryption methods Ei may verify the shares. For each participant Pi a non-interactive verification algorithm can be run on PROOFD to verify that Ei(si) is a correct encryption of a share for Pi. Since anyone may verify a share, it may be ruled out that a participant complains while it received a correct share. In case one or more verifications fail, we therefore say that the dealer fails, and the protocol is aborted. (If some level of fault-tolerance is desired one may continue and think of it as a (t, n − c)-threshold scheme, where c is the number of verifications that failed.)

Reconstruction The protocol consists of two steps:

1). Decryption of the shares. The participants decrypt their shares si from Ei(si). It is not required that all participants succeed in doing so, as long as a qualified set of participants is successful. These participants release si plus a string PROOFPi that shows that the released share is correct.

2). Pooling the shares. The strings PROOFPi are used to exclude the participants which are dishonest or fail to reproduce their share si correctly. Reconstruction of the secret s can be done from the shares of any qualified set of participants.

3.4.6.2 Special PVSS Scheme in LEGT. Let Gq denote a group of prime order q, such that computing discrete logarithms in this group is infeasible. Let g, G denotes independently selected generators of G, q, hence no party knows the discrete log of g with respect to G.

We will use the protocol by Chaum and Pedersen [29] as a subprotocol to prove that $\log_{g1} h_1 = \log_{g2} h_2$, for generators g1, h1, g2, h2 ∈ Gq. We denote this protocol by DLEQ (g1, h1, g2, h2), and it consists of the following steps, where the prover knows a such that $h1 = g_1^{\ a}$ and $h2 = g_2^{\ a}$:

1). The prover sends a1 = $g_1^{\ w}$ and a2 = $g_2^{\ w}$ to the verifier, with w∈ Zq.

2). The verifier sends a random challenge c∈ Zq to the prover.

3). The prover responds with r = w − ac (mod q).

4). The verifier checks that a1 = $g_1^{\ r} h_1^{\ c}$, a2= $g_2^{\ r} h_2^{\ c}$

We modified secret sharing protocol described in section 2.4.5 as following:

Initialization Gq and the generators g is selected using an appropriate public procedure by TA. Cluster key pair <e,d> is also generated. Each node Pi generates a private key SKi and broadcasts PKi.

Distribution The protocol consists of two steps:

1). Distribution of the shares. Suppose w.l.o.g. that the TA wishes to distribute the cluster secret key among participants P1, . . . , Pn. The TA picks a random polynomial f of degree at most K − 1 with coefficients in Zq:

$f(x) = d + a_1 x + a_2 x^2 + ... + a_{K-1} x^{K-1}$, such that the shared secret is $f(0)$=d.

The TA keeps this polynomial secret but publishes the related commitments $C_j = g^{a_j}$, for 0 ≤j < k. The TA also publishes the witness of each node's partial secret Xi, as $X_i = g^{p_i}$, for 1 ≤i ≤n. Finally, TA sends p(i) to each node privately or encrypt by nodes' public keys.

2). Verification of the shares. Any node computes $Xi = \prod_{j=0}^{k-1} C_j^{i^j}$, 0 ≤j < k, 1 ≤i ≤n from the Cj values and verify witness of secrete share as Xi= $g^{p(i)}$. By doing so, any node can verify that TA didn't send invalid shares to some or all of the nodes during distribution phase.

Reconstruction The protocol consists of two steps:

Step1: The certificate requester Vi broadcasts its certificate issuing/update request message *m* within its cluster in the format of {Vi, PKi, Cm, Texpire}, which may read: "node Vi from Cluster Cm is requesting a certificate of its personal public key as PKi which will expire after Texpire" .

Step2: After receiving this request, each node Vj within cluster Cm may check its transaction with Node Vi and decides whether to issue its partial certificate or not. If yes, Vj computes the partial certificate $cert_j = m^{P_j} \bmod n$ and sends it back to Vi. As in discussed in distribution phase, the witness of node j's share p(j) , Xj= $g^{p(j)}$ is already published. The serving node Vj shows that the partial signature is consistent by producing a proof of knowledge of the unique p(i), 1 ≤i ≤n, satisfying:

Xj = $g^{p(j)}$, CERTj= $m^{p(j)}$

Equivalent problem is to follow DLEQ(g, Xj , m, CERTj) to proof that $\log_{cert} CERTj = \log_g Xj$. The non-interactive proof is the n-fold parallel composition of the protocols for DLEQ (g, Xj , m, $CERTj$). Applying Fiat-Shamir's technique, the challenge c for the protocol is computed as a cryptographic hash of Xj , CERTj, $a1 = g^w$, $a2 = m^w$, where w is randomly chosen. Serving node Vj calculate c=HASH(Xj , CERTj, $a1$, $a2$) and response $r = w - cp(j)$ then signs {a1,a2,r}as proof to Vi.

Step3: Pooling the shares. When node Vi receives the signed {a1,a2,r}, it calculates c=HASH(Xj , CERTj, $a1$, $a2$) and verifies if a1 $= g^r X_j^c$, a2= $m^r CERTj^c$. Node Vi believes that the partial certificate CERTj is valid if both above equations hold. Otherwise Vi believes that Vj is misbehaving or broken. It may further mark Vj in its CRL and send out and accusation.

Suppose w.l.o.g. that node i produce correct values for CERTj after verification, for j = 1, . . . , K. The cert is combined as:

$$cert = \prod_{j=1}^{K} cert_j * lj(0) \bmod n$$

Note that the nodes do not need nor learn the values of the exponents p(i). Only the related values *certi* are required to complete the reconstruction of the certificate. Also, note that Node Pi does not expose its private key SKi; consequently node Pi can use its key pair in several runs of the PVSS scheme.

3.4.7 Proactive Secret Share Update – An Implementation

To further defend the polynomial secret sharing against the model II adversaries, Herzberg [25] et al. proposed periodical secret share updates with different

polynomials.We employ this technique to further improve the robustness of our system against adversaries of the type of model II. There are two options to achieve this goal. The first is to periodically update the shared certificate signing key ΣK. However, this option involves network-wide well-known change of ΠK, which may not be desirable from the application's perspective. The second option is the proactive secret sharing mechanism. Instead of changing ΣK and ΠK, we update the secret share of each node while keeping the shared ΣK intact.

Our constructed secret share updating algorithm is as follows:

1) The secret share is implemented within each cluster (as previous section)

2) Time periods for renewal are set arbitrarily at initial stage

3) Each node Vi, 0<i<n, picks *k* random numbers from the finite field and creates a polynomial of degree k:

$$\delta_i(z) = \delta_{i1}z^1 + \delta_{i2}z^2 + \ldots + \delta_{ik}z^k$$

Where $\delta_i(0) = 0$

4) For all other nodes Vi, Vj secretly sends out $\delta_i(j)$ to Vj

5) Vi computes the new share by:

$$p_i^{(t)} \leftarrow p_i^{(t-1)} + (\delta_{1i} + \delta_{2i} + \ldots + \delta_{ni})$$

6) Vi updates its share and erases all other data

If all of the nodes follow the protocol, then the share renewal protocol is **correct, robust** and **secret:**

- Each new round produces a valid set of secret shares
- Any *k+1* servers can re-create the secret at any time
- With k or less shares, no information is learned

The added advantage of proactivization in practical systems is that in a long-lived threshold system (like RSA for public-key infrastructure) an adversary has a long time (5 years) to break into any *t* out of the *l* nodes, while in the proactive systems the adversary has only a short period of time (say, a week) to break into any *k* nodes. This protection seems important in a "long-lived" Mobile Ad Hoc network. Another advantage of the "proactive model" is a flexible key management of share-holders in a function sharing scheme. It enables, in the refreshing period, to change the set of shareholders (by giving the recovered new share to the new node rather than the old one).

In case that "space diffusion" and "time diffusion" defenses in a cluster both fail, the certificate services as a whole fail. That means all nodes within this cluster cannot get certificates renewed or the renewed certificates are not valid. Since in our model clusters are autonomous groups, which manage certificates themselves and update the local repository of other cluster's certificate, the influence of one comprised cluster is reduced. After the certificates of the nodes from a compromised cluster expire, they will be expunged from the network; revocation of certificates will commence when enough neighbor clusters judge that a particular cluster is abnormal and/or malicious.

3.5 Cluster-based Authentication, Certificate Renewal & Revocation

In previous sections, we discussed how secrets are distributed and maintained by applying threshold cryptography in our LEGT algorithm, also have adopted, adapted, and improved the salient aspects of proactive security [21,25], and publicly verifiable secret sharing [35] security schemes. In this section, we mainly focus on certificate services, cluster_based authentication in LEGT. Certification services include issuing/renewing

certificates, revoking certificates, storing and retrieving certificates, and the creation and maintenance of certificate revocation lists (CRLs). Each certificate is stamped with an expiration time. Nodes have to renew and get a new certificate before time expiration. Certificates may become invalid before expiration. The revocation service has to put such information on the revoked certificates into CRLs for queries from nodes.

Managing certificates and CAs involves the following processes:

- Issuing certificates to users and computers. The issuance process includes obtaining and validating information about the intended recipient of the certificate, placing policy restrictions designated by the organization in certificates that are issued, and publishing the certificates to a directory.
- Managing certificate lifetimes. Because all certificates have a limited life, certificates need to be renewed or allowed to expire. The renewal process is similar to the issuance process, but typically involves fewer security checks. Thus, when an organization develops its renewal strategy, it should balance security concerns against potential disruptions to users
- Revoking certificates and verifying revocation status. Some certificates need to be invalidated before their expiration date. Effective certificate revocation and revocation verification processes are critical to the security of an organization's public key infrastructure (PKI).

Authentication in the PKI system relies on two factors. For node *Vi* to authenticate itself to another node, it has to prove its knowledge of the private key *SKi*, and the association between itself and the advertised public key *PKi*. A challenge/response protocol can be followed to prove the knowledge of the private key *SKi*. The association is proved by a certificate.

3.5.1 Local Certificate Renewal/Revocation

As we described earlier, renewal of certificates is done within each cluster. Each node broadcasts its renewal certificate requirement throughout the whole cluster. This

broadcasting can be easily done because the CBRP keeps two-hop topology information for each node, and the formation of the cluster requires 2-hop diameter. That means, for each node, all other nodes within its cluster are already reachable. Therefore, when implementing renewal of certificates, each node just needs to broadcast to its *Two-hop Database*. No further routing discovery is needed. Thus, the overhead brought by certificate renewal is kept low.

As in *Figure 3.3,* each node broadcasts its issuing/renewal requirement through the whole cluster, and other member checks his performance history (assuming some neighborhood watch IDS system exists) and decides whether to sign the new certificate or signature. If yes, unicast partial signature share or certificate share signed by his share of *cluster private key*. Node which can't get enough partial certificates or partial signatures will be purged after certain time window.

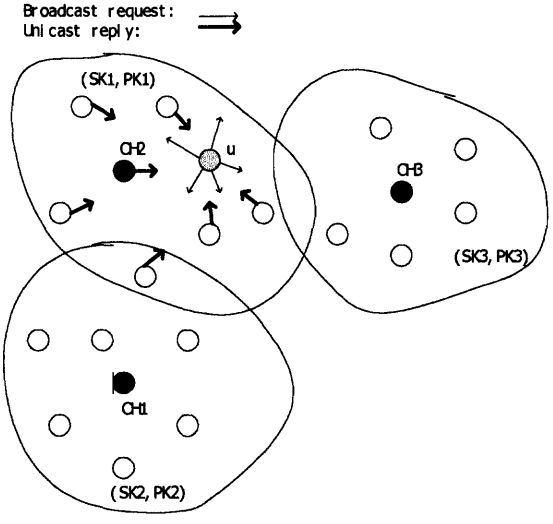


Figure 3.3 Certificate renewal/revocation.

Also, there are some reasons or techniques in our LEGT to revoke a certificate:

- A node who is found providing a wrong partial certificate for other nodes in his cluster, according to PVSS. Maybe a node who doesn't have the public key who

claims to, and cannot decrypt his share correctly; maybe a node who intentionally sends out a wrong partial certificate.

- A node who is found not cooperating with routing protocol
- A node who is found abnormal by intrusion detection system, please refer our work in [26].

3.5.2 Creation of the *Cluster Public Key* Certificate and Repositories

As we described above, the *public key* certificate of each entity is signed by shares of the *cluster private key*. In this section, we describe the creation of *cluster public key* certificate and repositories. The *cluster public key* certificates are issued by other cluster heads. If a cluster head CHm believes C_{pk}^{n} belongs to cluster CHn (generally cluster is noted by its cluster head ID), then CHm can issue a public-key certificate in which C_{pk}^{n} is bound to CHn by *cluster private key* C_{sk}^{m}.

The certificate graph changes during operation, when clusters issue certificates to others or get certificates issued by other clusters. Therefore, a certificate exchange mechanism is needed. Similarly to [1], the certificate exchange mechanism requires and relies upon the periodic exchange of certificates between neighboring clusters. By the certificate exchange mechanism, nodes accumulate certificates in their local certificate repositories at a low communication cost because the exchanges are performed locally in neighboring cluster and may combined with neighboring cluster discovery scheme provide by routing protocol, for example, CBRP.

The certificate exchange enables the cluster heads to create their local certificate repositories. Each cluster head constructs its updated repository by communicating with their certificate graph neighbors. Typically, this algorithm runs in steps where in each

step the node decides which certificates to store and how to proceed with the exploration of the certificate graph. More details are given in [1]. After the construction of the local repository, the nodes are ready for authentication.

3.5.3 Inter-cluster Authentication via Exchange of Certificates

Following, we discuss the performed steps of authentication in three different scenarios. They are:

3.5.3.1 When a Node Joins a Network for the First Time. When a new node joins the network and is detected by a cluster head (by means of hello messages). We assume there are some reasons that one node trust another node, such as physical contact, authorities commands [2], [13]. Here we have two options: 1) each node has his own key pair<skj, pkj> and threshold scheme <k, n> is used to co-sign the certificates 2) to reduce the public key directory, each group has only one common public key instead of each node in the group has a key, and any subset of k individuals can sign the signature. We could use the multisignature scheme presented in [16]. New node also gets the cluster public key and cluster private key share when join in.

3.5.3.2 When a Node Leaves a Cluster and Joins Another Cluster. This situation arises due to the movement of nodes. When a node moves from a cluster to new one, the new cluster head first checks its certificate from old cluster. If the certificate is valid and the new cluster head can find a chain path for old cluster head's public key in his repository, the new cluster head believes the authenticity of the new node. The cluster head then gives the node the *cluster public key* and one *cluster private key share* for the

new cluster. The old cluster purges the entry for this node when it doesn't receive hello message for a certain predefined time interval.

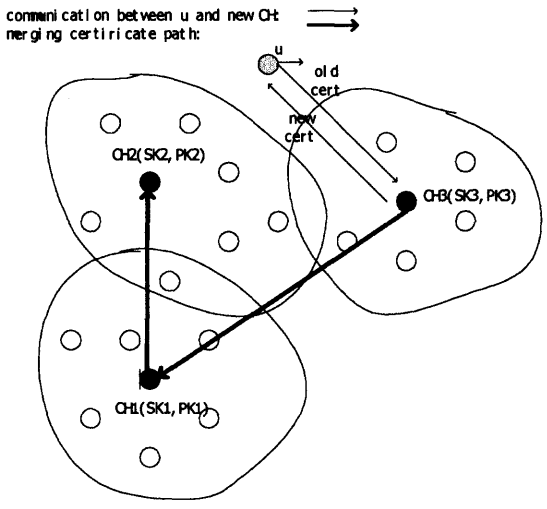


Figure 3.4 Authentication when node u is leaving from cluster 2 and entering cluster 3.

3.5.3.3 When a Node From a Cluster Wishes to Communicate with a Node Belonging to Another Cluster. This is a complex scenario and our scheme tries to minimize the overhead involved here. For example, node u from cluster 1 (Cluster header is CH1) wants to communicate with node v from cluster 3 (Cluster header is CH3). First, u and v exchanges their certificates signed by their group (As in Fig 3.3) through challenge/response. If certificates are valid, there are two ways user u and v can check the authenticity of each other's public key. First, cluster head maintains and keeps the local cluster repository, so u and v transmit each other's certificate to their own cluster header, namely CH1 and CH3. CH1 and CH3 merge a certificate path in their local repositories and prove to u and v the authenticity of each other's certificate, as shown in Fig 3.5; the

second way is that cluster head send cluster repository table to each node, and u and v can merge the certificate path by themselves.

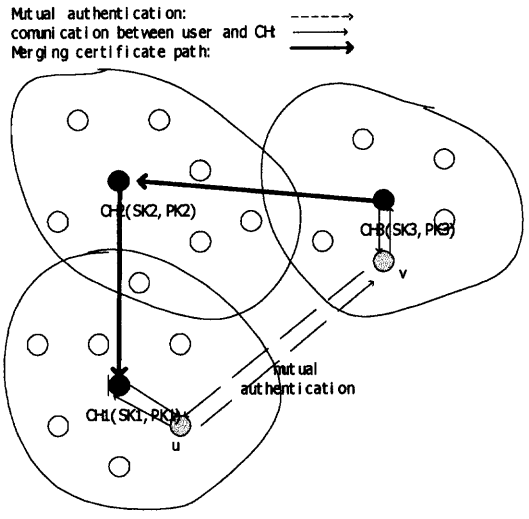


Figure 3.5 Mutual authentication between node u and v.

3.6 Implementation

In previous sections, we describe the general concept and design of our algorithm. In this section, we discuss the implementation of our algorithm in ad hoc networks where Cluster Based Routing protocol (CBRP) is used as routing protocol. We first review CBRP background, then describe how implementations of certificate renewal, key sharing update, construction of local repository are done applying messages, tables CBRP already provides.

3.6.1 CBRP Background

Cluster Based Routing Protocol (CBRP) is a routing protocol designed for use in mobile ad hoc networks. The protocol divides the nodes of the ad hoc network into a number of overlapping or disjoint 2-hop-diameter clusters in a distributed manner. A cluster head is elected for each cluster to maintain cluster membership information. Inter- cluster routes are discovered dynamically using the cluster membership information kept at each cluster head. By clustering nodes into groups, the protocol efficiently minimizes the flooding traffic during route discovery and speeds up this process as well. Furthermore, the protocol takes into consideration the existence of unidirectional links and uses these links for both intra-cluster and inter-cluster routing.

Following are several terms defined in CBRP:

1) Cluster: A cluster consists of a group of nodes with one of them elected as a cluster head. A cluster is identified by its Cluster Head ID. Clusters are either overlapping or disjoint. Each node in the network knows its corresponding Cluster Head(s) and therefore knows which cluster(s) it belongs to.

2) Cluster Head: A cluster head is elected in the cluster formation process for each cluster. Each cluster should have one and only one cluster head. The cluster head has a bi-directional link to every node in the cluster. A cluster head will have complete knowledge about group membership and link state information in the cluster within a bounded time once the topology within a cluster stabilizes.

3) Cluster Member: All nodes within a cluster EXCEPT the cluster head are called members of this cluster.

4) Gateway Node: Any node a cluster head may use to communicate with an adjacent cluster is called a gateway node.

5) HELLO message: All nodes broadcast HELLO messages periodically every HELLO_INTERVAL seconds; a node's HELLO message contains its Neighbor Table and Cluster Adjacency Table. A node may sometimes broadcast a triggered HELLO message in response to some event that needs quick action.

Following three tables are constructed and updated through HELLO messages in each node:

1) Neighbor Table

The neighbor table is a conceptual data structure that employed for link status sensing and cluster formation. Each entry contains

- the ID of the neighbor that it has connectivity with
- the role of the neighbor (a cluster head or a member).
- the status of that link (bi-directional or unidirectional)

2) Cluster Adjacency Table

The Cluster Adjacency Table keeps information about adjacent clusters and is maintained by CBRP's Adjacent Cluster Discovery procedure.

Each entry contains:

- the ID of the neighboring cluster head
- the gateway node (a member) to reach the neighboring cluster head
- the status of the link from the gateway to the neighboring cluster head (bi-directional or unidirectional)

3) Two-hop Topology Database

In CBRP, each node broadcasts its neighbor table information periodically in HELLO packets. Therefore, by examining the neighbor table from its neighbors, a node is able to gather `complete' information about the network topology that is at most two-hops away from itself. This two-hop topology information is kept in a data structure in each node.

3.6.2 Locally Certificate Renewal-Implementation

As we described in Section 3.5, certificates renewal is done within the each cluster. Each node broadcasts its renewal requirement through the whole cluster. This broadcasting can be easily done because CBRP keeps two-hop topology information in each node, and the formation of cluster requires 2-hop diameter. That means, for each node, all other nodes within its cluster are already reachable. Therefore, when implementing certificates renewal, each node just needs broadcasting its *Two-hop Database*. No further routing discovery is needed. The overhead brought by certificate renewal is kept low.

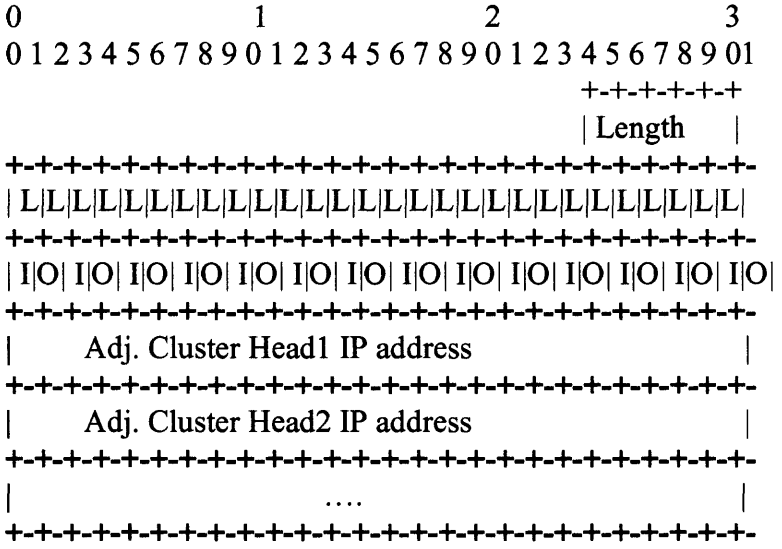
3.6.3 Cluster Certificate Graph Update and Construction of Cluster Repository

In our algorithm, authentication between members of different clusters is defined as two steps: first check the binding of *cluster public key* and *cluster id*, then check the authenticity of certificate signed by *cluster private key*. The binding of *cluster public key* and *cluster id* is made public through "web of trust" [2], which means each cluster member gives certificate for the partial certificate signed by his *private cluster key share* if he believes the authenticity of objective *public cluster key*. When mutual authentication is needed, local repositories of two clusters are merged to find the certificate chain.

During networking operation, each node updates certificate graph and constructs local repository based on the updated certificate graph.

Because only adjacent cluster certificate is needed, we make an extension to existing *Cluster Adjacency Table* on each node, piggyback this extended table to HELLO messages. When receives HELLO messages, each node can update its certificate graph and chooses its outgoing and incoming edges as its local certificate repository.

Following is the format for our extended *Cluster Adjacency Table:*



Length	The number of clusterheads listed
L	Link Status of the corresponding Adjacent cluster head 0-----LINK_BIDIRECTIONAL 1-----LINK_FROM
I	Whether the corresponding adjacent cluster sign me a certificate? 0-----YES 1-------NO
O	Whether I have the certificate of corresponding adjacent cluster? 0-----YES 1-------NO

3.7 Evaluation

To evaluate our model is a hard issue since the security of our model is based on mathematical foundations of cryptography schemes we are using and issues such as operational network environment, adversary models are still open. In this section, we analyze robustness of our security model, then we discuss the overhead and computation operation it requires.

3.7.1 Security Analysis

In MANETs, Mobile hosts can be physically captured and compromised. Therefore, authentication architecture should avoid having any central entity so that compromise of one entity does not risk the whole network. In stead of using one or several CA servers, we first distribute certificate services such as certificate issuing/renewal to all member within the cluster. In each cluster, as if there are t or more nodes are working, the certificate services still can work. Threshold cryptography (so called "space diffusion") provides for increased availability of authentication, especially important in MANET where nodes may operate in hostile environment. Another important threat for MANET is "mobile adversary" which is adversary who move among the network to collect secret shares. Given a shared function within each cluster as we describe above, we also use proactive security (so called "time diffusion"), which adds a periodic refreshing of the secret shares hold by each node. Therefore, our model can tolerate a "mobile adversary" which can only collect less than t-1 shares within a time unit.

In case that "space diffusion" and "time diffusion" design in a cluster both fail, the certificate services of this whole fail. That means all nodes within this cluster cannot

get certificates renewed or renewed certificates are not valid. Since in our model clusters are autonomous groups who manage certificates themselves and update local repository of other cluster's certificate, the influence of one comprised cluster is reduced. After certificates of nodes from compromised cluster expire, they will be expunged from network; or certificates revocation will be turn on when enough neighbor clusters find a cluster is abnormal.

Distribution of cluster private key also has other advantages. First, neighbor nodes have the most frequent chance to communicate with each other, so they have largest confidence to decide a neighbor node is normal or abnormal, then decide whether to sign a certificate with their own key share. Generally IDS is installed on each node also based on neighborhood watch, whose results can be combined into certificate services. Another advantage of clustering enforced certificate is that the certificate chain to get an authentic *cluster public key* is reduced. The shorter the chain, the smaller the probability of faked certificate from compromised cluster.

3.7.2 Overhead

As we described in previous parts, our scheme extends the idea of certificate services further in an attempt to minimize the effort and complexity for mobile clients to locate and contact the service providers. We devise scalable algorithms and protocols to enable the distribution of the certification services into *every node*. There is no differentiation of servers and clients any more, in our architecture: a threshold number of any nodes can collaboratively act as servers to provide services for other nodes.

In implementation, our algorithm makes usage of existing *Two-hop Database* for locally certificate renewal, the increasing number of routing messages is kept low; for cluster certificate graph update, we makes extension to *Cluster Adjacency Table* which is already "piggybacked" into HELLO message. Therefore, the total increased overhead to network due to our algorithm is kept low.

3.7.3 Communication Performances

In this section, we will evaluate communication efficiency of our authentication protocol. We implement Locally-Enforced algorithm on our MANET emulation network (Manetbed) [26]. Manetbed is a novel, low-cost MANET emulation test-bed which is based on Ethernet environment. Manetbed provides mobile multi-hop wireless emulation including routing protocols (currently CBRP and AODV), random waypoint mobility, random packet loss, bandwidth limitation and communication delay. Manetbed emulation is scalable, accurate and deploying on real nodes, running real traffics and applications.

In each node, we apply an application-layer agent that allows for delivery of actual certificate renewal data units and one-hop broadcast. Two metrics are used for evaluate the communication efficiency of our protocol: *Success ratio* means the ratio of the number of successful certificate renewal over the number of attempts. *Average delay* measures the average latency for each node to perform a certification renewal services.

Node number in our emulations is 50. Simulation time is 900 seconds. The node mobility varies from 10, and 20m/sec. Pause time varies from 0, 30, 60, 120, 300, 600, 900 seconds. The Random waypoint model is used to emulate mobility patterns. The

coalition size K=3 is chosen for threshold cryptography. 4 CAs services are implemented also for comparison with our algorithm.

As it is shown in Figure 3.6, the certificate renewal success ratio of our Locally-Enforced approach is around 98%, which is much better than centralized approach (4 CA is used currently). In Figure 3.7, we also present results for the average delay for both approaches. And we observe our Locally-Enforced algorithm has much smaller delay compared with centralized approach. We also observe mobility that average delay almost remains unchanged as mobility speed grows from 10m/sec to 20m/sec. However, centralize approach has significant increasing delay as speed increasing.

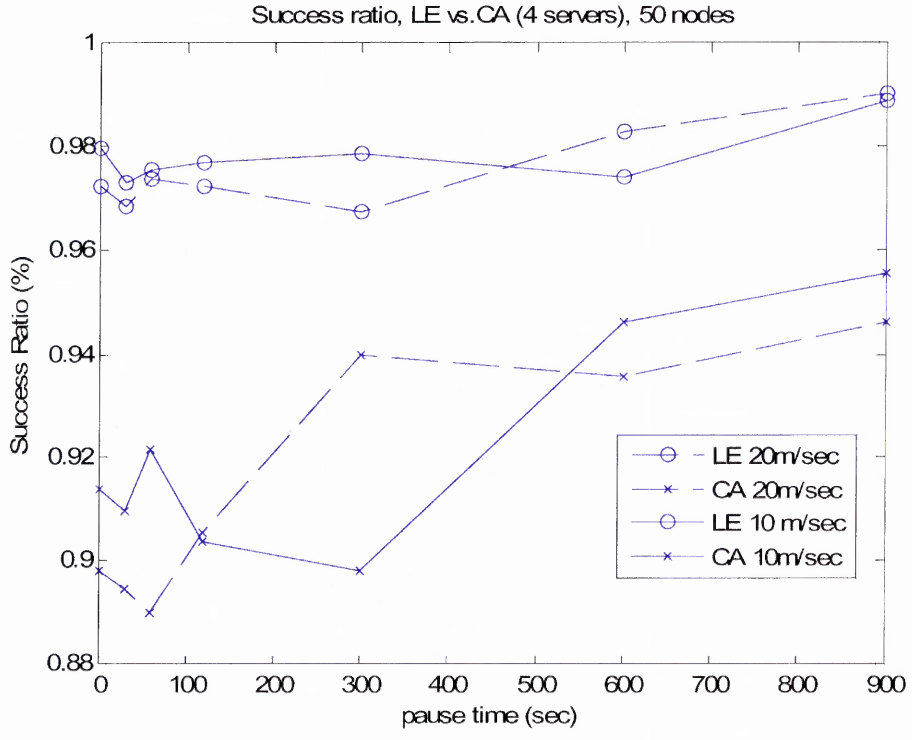


Figure 3.6 Certificate Renewal: Success Ratio vs. Pause time.

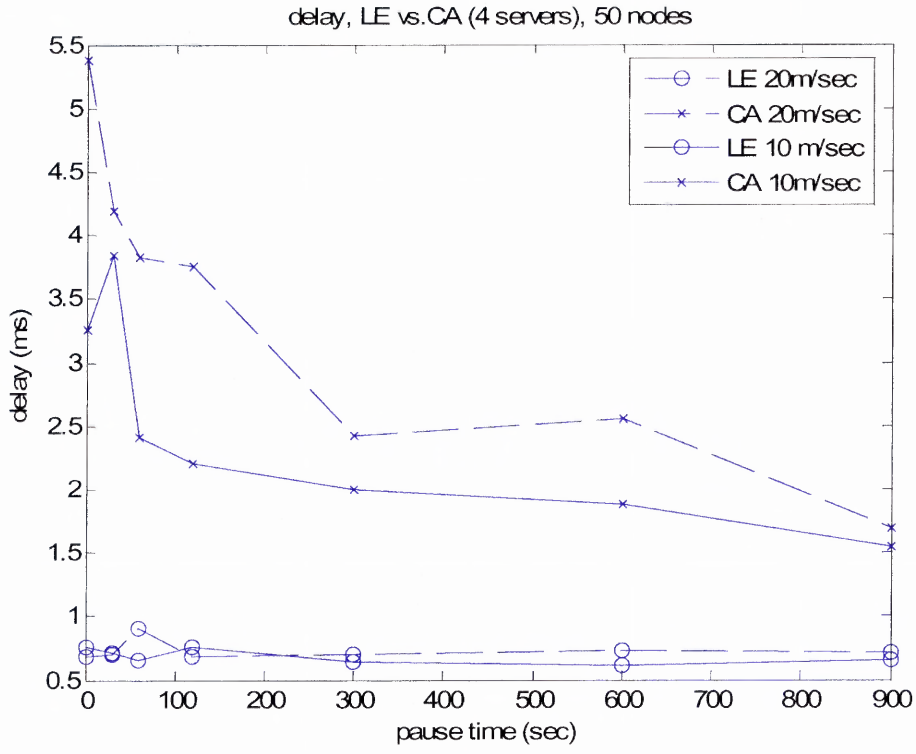


Figure 3.7 Certificate Renewal: Avg. Delay vs. Pause Time.

3.8 Conclusions

In this work we have dealt with security in ad hoc networks. We have focused on authentication since this the core requirement to achieve other security services. We have described several existing models and analyzed their scope, possible scenarios and inherent problems. Finally we present a new authentication model for high-value transactions in cluster-based MANET. This model is motivated by previous works but try to use their beauties and avoid their shortcomings. Our model is using threshold sharing of the certificate signing key within each cluster to distribute the certificate services, and using certificate chain and certificate repository to achieve better scalability, less overhead and better security performance. We also present in the paper an example of adoption of the general design into Cluster Based Routing Protocol, (CBRP), which is

one of the IEFT's candidates for a MANET routing protocol standard. We have implemented our algorithm in our MANET emulation testbed (Manetbed) and get evaluations about communication performance compared with centralized approach.

CHAPTER 4

INTRUSION DETECTION SYSTEM

4.1 Introduction

Protection of Ad Hoc networks basically focus on two aspects: Intrusion Prevention, such as Traffic encryption, Sending data through multiple paths, Authentication and authorization; Intrusion Detection, such as Anomaly pattern examination, Protocol analytical study. In this chapter, a novel Flow-based Network Intrusion Detection System (FNIDS) is presented. An IP flow is a unidirectional series of IP packets of a given protocol, traveling between a source and destination, within a certain period of time. Based on "flow" concept, a flow-based packet aggregation architecture is developed, which dramatically reduces the amount of monitoring data and handles high amounts of statistics and packet data. Existing flow-based network intrusion detection systems mainly analyze and detect bandwidth type Denial of services attack. Our method uses statistical models and multivariate classifiers to detect anomalous flow conditions. By applying up to 22 parameters for each flow, our FNIDS can detect both bandwidth type DOS and protocol type DOS. Moreover, flow here could be any set of packets sharing certain common property as "flow key". FNIDS configures flow flexibly to provide security from network level to application level (IP, TCP, UDP, HTTP, FTP...), and different aggregation schemes, such as server -based, client-based flow. This novel IDS has been evaluated by using CONEX wired test-bed and CONEX Mobile Ad hoc Test-Bed. Results show the success in terms of different aggregation schemes for both datasets. Some numerical results are also presented that demonstrate that our proposed

methodology can reliably detect attacks with traffic anomaly intensity as low as 0.1–5 percent of the typical background traffic intensity, thus promising to generate an effective early warning.

There currently exist multiple threats and vulnerabilities in the security of computer systems and networks against attacks. Along with the explosive growth of the Internet and the continued dramatic increase in all wireless services, the number and impact of attacks has been increasing. This is evidenced by recent well-publicized denial of service attacks against several prominent Web portals as well as many hushed over occurrences. The number of computer systems and their vulnerability have been rising, while the level of sophistication and knowledge required to carry out an attack has been decreasing, as much technical attack knowhow is readily available on Web sites all over the world [33].

Recent advances in encryption, public key exchange, digital signatures, and the development of related standards have set a foundation for the flourishing of electronic commerce. However, security on a network goes way beyond encryption of data. It must include the security of computer systems and networks, at all levels, top to bottom. It is imperative to arm the network systems and elements with well designed, comprehensive, and integrated attack defeating policies and devices. However, foolproof prevention of attacks is challenging because at best the defensive system and application software may also contain unknown weaknesses and bugs. Thus, *early warning systems* (i.e., *intrusion detection systems* or IDSs), as components of a comprehensive security system, are required in order to prime the execution of countermeasures. The technique of statistical anomaly and intrusion detection is the focal point of this article.

4.1.1 Types of Intrusion Detection Systems

An IDS may be of the network/node (NID) or host (HID) type. That is, the IDS may gather information from a computer (i.e., system audit logs and file states) or network of computers (i.e., packets passing on the wire between hosts or arriving at a particular host) in attempting to detect intruders or system abuse. Thus, network or node intrusion detection (NID) systems, typically referred to as "packet sniffers," intercept packets of various physical manifestations and protocols. In the past, NIDs have been challenged by switched, encrypted, and high-speed networks (> 100 Mb/s). However, recently these limitations have been largely overcome. For example, there are now NID systems that may be installed on switches as well as monitor at gigabit speeds. After capturing a packet, some NID devices will simply compare the packet to a signature database of known attacks ("fingerprints"), while others seek to detect "anomalous" packet activity. Thus, the NID techniques can be partitioned into two complementary trends: *misuse detection* and *anomaly detection*. Misuse detection systems [34] model known attacks and scan the system for occurrences of these patterns. *Anomaly detection systems* [35] flag intrusions by observing significant deviations from typical or expected behavior of systems or users. HIDs monitor and detect user and system activity and attacks on a given host, so these systems are best suited to counter internal (intranetwork) threats because they focus on monitoring user actions and file accesses on the host.

Intrusion detection research and development dates to the early 1980s starting with Anderson's paper [36] that introduced the concept of computer threats and detection of misuse, as applied to the host IDS, or HID. Dr. Denning's work [37] then introduced the first model for intrusion detection, followed with the Haystack project IDS and later

the distributed IDS (DIDS). In the early 1990s, NID was introduced [38], along with early versions of commercial and government IDS systems. However, most IDS work is rather recent; many research systems and commercial products appeared in the late 1990s, following the growth of the Internet [39].

Moreover, the past few years have witnessed much progress in all aspects of fault detection, including path failure detection, anomaly detection in the Ethernet, anomaly and performance change detection in small networks, network alarm correlation, and real-time trouble ticketing information in a transaction-oriented communication network [40]. The main objective of fault detection is to identify network exceptional conditions, such as performance and utilization degradations. Under favorable conditions that can be accomplished by design, network fault detection can anticipate the occurrence of high-level service failures and compromises, and thereby increase the chances for proactive fault correction before service failures set in.

4.1.2 Statistic Anomaly Detection

Statistical modeling and neural networks have been applied in building anomaly network intrusion and fault detection systems. A system that identifies intrusions using packet filtering and neural networks was introduced in [41]. The authors in [36] studied the employment of neural networks to detect anomalous and unknown intrusions against a software system. In [42], Kolmogorov-Smirnov statistics was used to model and detect DoS as well as probing attacks. In [43], we proposed the prototype of a Hierarchical Intrusion Detection system (HIDE), while in [44] we introduced the Generalized

Anomaly and Fault Threshold system (GAFT); these systems use statistical preprocessing and neural network classification to detect network attacks and faults.

4.1.3 Applying Flow Concept in DDOS Detection

Distributed Denial-of-Service (DDoS) attack presents a very serious threat to the stability of the Internet. In a typical DDoS attack, a large number of compromised hosts are amassed to send useless packets to jam a victim, or its Internet connection, or both. In the last two years, it is discovered that DDoS attack methods and tools are becoming more sophisticated, effective, and also more difficult to trace to the real attackers. Identifying, diagnosing and treating anomalies in a timely fashion are a fundamental part of day-to-day network operations. However, modeling the traffic at the packet level has proven to be very difficult since traffic on a high-speed link is the result of a high level of aggregation of numerous flows.

Recently, a new trend has emerged for modeling high-speed Internet traffic at the flow level.

Strictly speaking, flow is a genetic concept. An IP flow is a set of packets, that are observed in the network within some time period, and that share some common property known as its key, which can be a TCP connection or a UDP stream described by source and destination IP addresses, source and destination port numbers, or the protocol number etc. If we collect and statistic packets from network based on pre-defined flow identifier (or key), there are countless aggregation schemes. The flexible data collection and analysis implementation based on flow make the NIDS have the advantage of greatly

reducing the amount of data collected and the features formed by this aggregation schemes can provide detailed network performance information.

Flow aggregation techniques are used to aggregate flows (packets) into one flow with a larger granularity of classification (e.g., from port number to IP address). Aggregated flows have a larger number of packets and longer flow duration that dramatically reduces the amount of monitoring data and handles high amounts of statistics and packet data. Therefore, Internet traffic flow profiling has become a useful technique in the passive measurement and analysis field. The prerequisites for flow-based measurements are now available within the network infrastructure – particularly, in popular Cisco network devices. The integration of this feature has enabled the 'flow' concept to become a valuable method.

Despite a large literature on traffic characterization, traffic anomalies remain poorly understood. There are a number of reasons for this. First, identifying anomalies requires a sophisticated monitoring infrastructure. Unfortunately, most ISPs only collect simple traffic measures, *e.g.,* average traffic volumes (using SNMP). More adventurous ISPs do collect flow counts on edge links, but processing the collected data is a demanding task. A second reason for the lack of understanding of traffic anomalies is that ISPs do not have tools for processing measurements that are fast enough to detect anomalies in real time. Thus, ISPs are typically aware of major events (worms or flooding DoS attacks) after the fact, but are generally not able to detect them while they are in progress.

4.1.4 Problems of Current IDS Techniques in MANET Networks

The vast difference between the fixed network where current intrusion detection research are taking place and the mobile ad-hoc network which is the focus of this paper makes it very difficult to apply intrusion detection techniques developed for one environment to another. 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, the mobile ad-hoc environment 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.

Furthermore, there may not be a clear separation between normalcy and anomaly in mobile environment. 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.

The rest of the chapter is organized as follows: Section 4.2 describes the system architecture. Section 4.3 described the statistic model and parameter used, Section 4.4 discusses reference model, score metrics and Neural Network Classification; Section 4.5 introduces the CONEX Test bed and CONEX Manetbed. Some experimental results are

also reported in that section. Section 4.6 draws some conclusions and outlines future work.

4.2 System Architecture

FNIDS is a hierarchical multitier system that may run in a distributed as well as standalone fashion. It is a security component that monitors the activities of a host or the network to which it is attached. It gathers data from network traffic, statistically processes and analyzes the flow information; detects abnormal activity patterns based on the reference models, which correspond to the expected activities of typical users, for each parameter individually, or in combined groups using neural network classification; generates the system alarms and event reports;

Each FNIDS agent consists of the following components: the probe, the event preprocessor, the statistical processor, the neural network classifier, and the post processor. A diagram of the FINDS is given in Figure 4.1. The *probe* collects the network traffic of a host or network, abstracts the traffic into a set of statistical variables to reflect the network status, and periodically generates reports to the event preprocessor. The event preprocessor receives reports from the probe, and converts the information into the format required by the statistical model. The *statistical processor* maintains reference models of 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. The *neural network classifier* analyzes the stimulus vector from the statistical model to decide whether the network traffic is normal or not. The *post processor*

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

Neural network classifier uses back-propagation networks to classify score metric of each flow. Existing Flow-based Network Intrusion Detection Systems (FNIDS) mainly analyze and detect bandwidth depletion Denial of services attack. By applying up to 22 parameters for each flow, our FNIDS can detect both bandwidth depletion DOS and resource depletion DOS. Moreover, flow here could be any set of packets sharing certain common property as "flow key". FNIDS configures flow flexibly to provide security from network level to application level (IP, TCP, UDP, HTTP, FTP...), and different aggregation schemes, such as server -based, client-based flow.

Because network traffic is not stationary and network-based attacks may have different time duration, we need to monitor network traffic with different time windows. Thus, we adopted a nested layer-window model with each window corresponding to a geometrically increasing detection time slice (typically by a factor of 4). The newly arrived events will first be stored in the event buffer of observation window layer 1. The stored events will be compared to the reference model of that layer; the similarity results are then fed into a neural network classifier to detect 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. The same process will be repeated recursively until data arrives at the top level where the events will simply be dropped after processing.

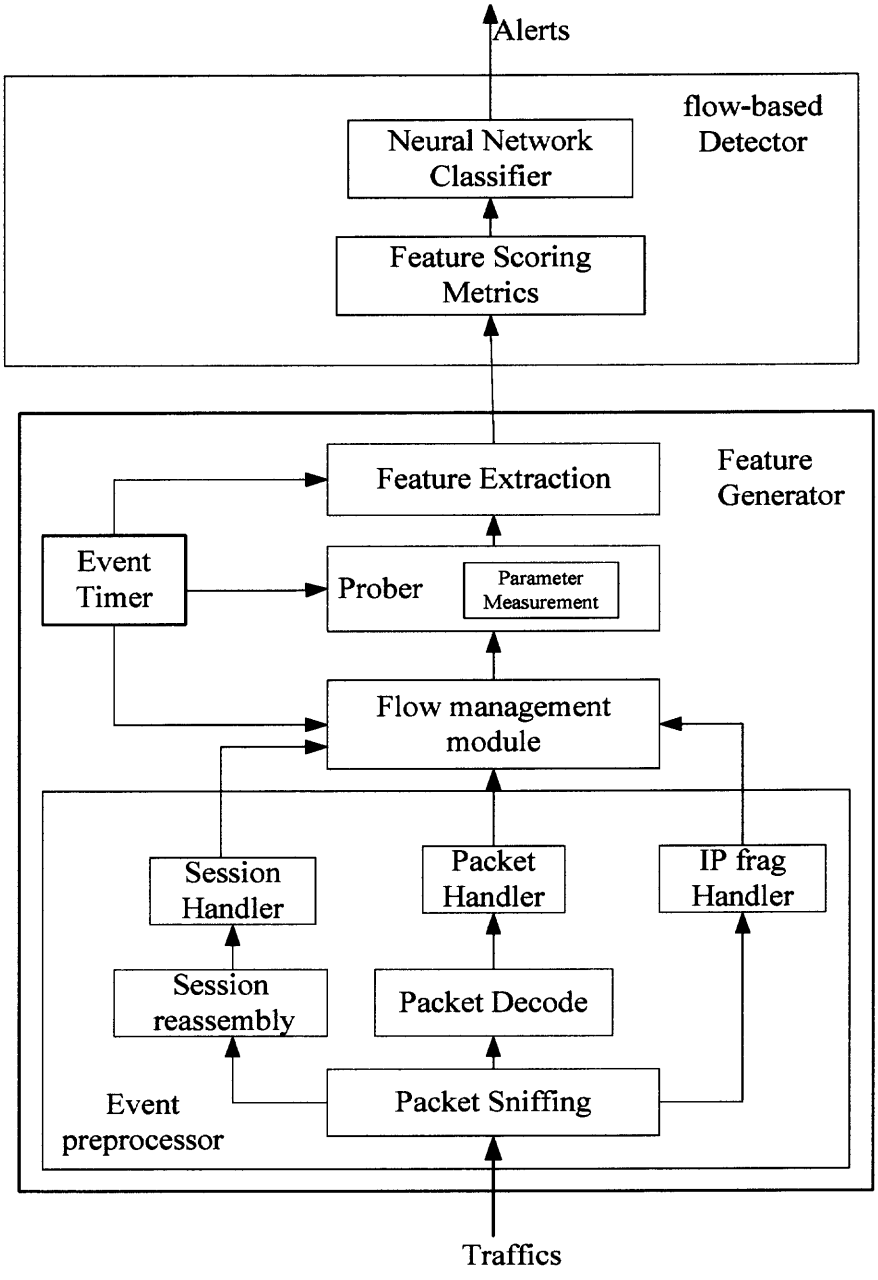


Figure 4.1 Flow-based network intrusion detection system architecture.

Event Preprocessor: Collects the network traffic of a host or a network, Event handlers generate reports to Flow management module;

Event Timer: Periodically calls Feature Extraction Module to converts the statistic information of flows into the format required by the statistical model.

Flow Management Module: efficiently determines if a packet is part of an existing flow or should generate a new flow key; According to different flow key, this module aggregates flows together based on their flow keys, and dynamically updates per-flow accounting measurements;

Probe: Collects the network traffic, abstracts the traffic into a set of statistical parameters to reflect the network status.

Feature Extraction: Periodically extracts the features, which describe the behaviors of flows.

Feature Scoring Metric: Calculates the probability scores of these features by comparing the features with the reference model generated by past normal and attack users. The probability scores are measurements indicating how likely for a feature to take the observed value.

Neural Network classifier: Classifies the score vectors to prioritize flows with maliciousness. The higher maliciousness of a flow means the flow has the higher possibility of attacker.

4.3 The Statistic Model

An *activity profile* characterizes the behavior of a given monitored parameter thereby serving as a description of normal activity for its respective subject. Observed behavior may be described using a statistical metric and model.

The period of observation may be a fixed interval of time (sec, minute, etc.), or the time between two audit-related events (i.e., between login and logout, file open and file close, etc.). Observations (sample points) x_i of x are used together with a statistical

model to determine whether a new observation is abnormal. It is preferred that the statistical model makes no assumptions about the underlying distribution of x; in such systems, all knowledge about x is obtained from observations. In our work, traffic profiles are represented by a number of probability density functions. Let S be the sample space of a random variable and events E_1, E_2, ..., E_k a mutually exclusive partition of S Assume that p_i is the expected probability of the occurrence of event E_i, p_{2i} represents the actual probability of the occurrence of E_i during a time interval, and N is the total number of occurrences.

Most earlier IDS systems simply measured the means and variances of the monitored variables and detected whether certain thresholds were exceeded; in nonstationary systems that often do not follow the normal distribution, such systems generate incorrect decisions. To overcome some of these problems, NIDES [35] used a χ^2-like statistical test to determine the similarity Q between the expected and actual distributions. However, in real applications empirically Q may not follow χ^2 distributions; therefore, NIDES solved this problem by building an empirical probability distribution for Q that is updated daily in real-time operation.

In our FNIDS system, since we are using neural networks to classify patterns and identify possible intrusions, we are not as concerned with the actual distribution of similarity metrics, which is described in following section; it is expected that the neural net will learn it from examples.

Detecting intrusions involves the multi-variants classifications based on the monitored features. Generally, intrusion detection systems are designed to use as many features as the designers could think they might be useful. For example, JAM [45], an intrusion

detection system prototyped by Columbia University, monitors 41 different parameters for each session. In our system, FNIDS provided up to 45 features, which are relevant in DoS attack detection [46]. The table 4.1 lists 22 selected features.

Table 4.1 Features in Set22

Index	Name	Index	Name
1	in.ip-pkt-rate	12	in.tcp-con-new-aborted
2	in.ip-byt-rate	13	in.tcp-con-half-opened-ratio
3	in.ip-frag-rate	14	in.tcp-con-duration
4	in.ip-defrag-error-rate	15	in.tcp-con-diff-src
5	in.ip-csum-error-rate	16	in.tcp-con-diff-dst
6	in.tcp-pkt-len	17	in.tcp-con-anomalous-entropy
7	in.tcp-pkt-rate	18	in.icmp-pkt-rate
8	in.tcp-syn-pkt-rate	19	in.icmp-byt-rate
9	in.tcp-rst-pkt-rate	20	in.icmp-diff-src
10	in.tcp-con-new-opened	21	in.icmp-diff-dst
11	in.tcp-con-new-closed	22	Io.icmp-anomalous-echo-reply

In our statistical model, abnormally detection is based on that features of attack traffic have different probability density distribution compared with features of normal traffic. Following, we visualization some popular attacks such as Teardrop, POD, Smurf, Neptune.

Sudden changes in packet counts, especially when based on protocol aggregation scheme, are usually indications of a flood as well. Figure 4.2 is an example based on IP traffic rate representing two kinds of floods. The x-axis of the figure is time window, in this experiment, it was set as 30seconds/time window; the y-axis of the figure is traffic change, which shows the discrepancy between the number of inbound and outbound flows or traffics. The figure 4.2 clearly shows when the flooding attacks (Smurf and

Neptune) happened. Figure 4.3 is an example graph based on flow counts representing Pod attacks. Figure 4.4 is an example graph based on IP fragment rate representing 'Teardrop' Attacks.

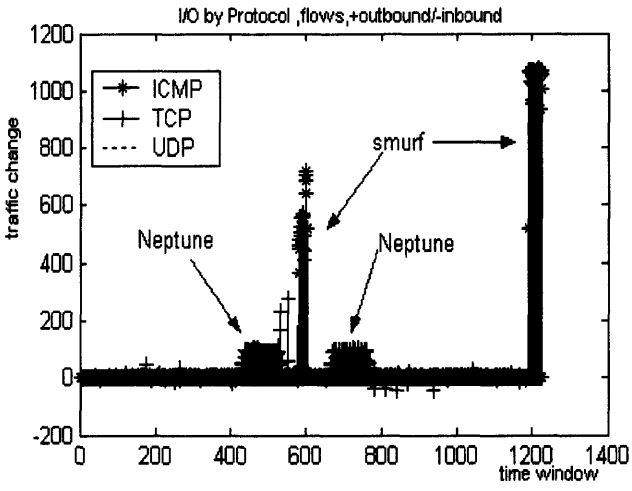


Figure 4.2 Smurf and Neptune DDoS flooding.

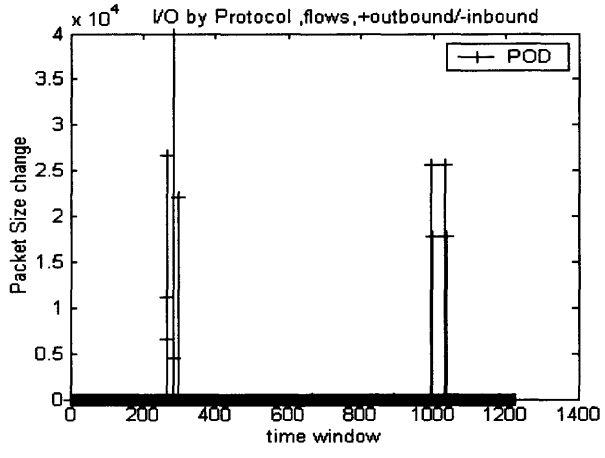


Figure 4.3 PoD attacks.

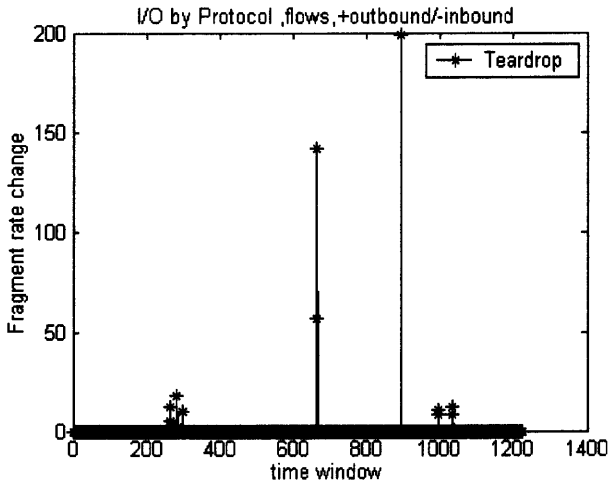


Figure 4.4 Teardrop Attack.

4.4 Neural Network Classification

As mentioned above, many types of distance metrics may be employed by the statistical analyzer to generate similarity measures. The output of the statistical processor is a *k*-dimensional ordered vector whose components are the similarity values of the measured PDFs to the reference PDFs of the *k* monitored network or system parameters collected during an observation window. This *k*-dimensional vector represents an evaluation of the status of the network or system during that observation time. Subsequently, this status vector gets presented to the multivariate classifier for a classification result. Clearly, the efficacy of this classifier is crucial to the success of the endeavor.

In this dissertation we chose the neural network classifiers. The *k*-dimensional ordered vector produced by the statistical processor to the classifier may be thought of as an input pattern to the classifier. Neural networks are widely considered an effective approach to handling and classifying patterns. However, the sometimes high computation requirements as well as computation requirements as well as long training cycles have hindered their applications. This is due to the fact that such neural networks often intake a

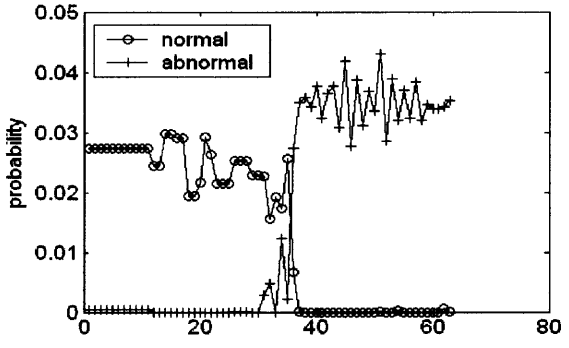
large number of inputs (several tens or hundreds) and utilize many (several dozen or more) computational neurons in one or several hidden layer(s). In other words, in such cases the neural networks are expected to carry out by themselves the majority of the computational work that leads to a decision. In contrast, our system uses powerful and comprehensive statistical preprocessing before presenting the vector of computed similarity values to the classifier for a decision and thus can perform effectively by deploying small and "lean" neural network classifiers, with only the minimum of inputs, a handful of computational neurons, and if otherwise a correspondingly small total number of weights. For about a dozen simultaneously monitored parameters leading to an equal number of inputs, as is the case here, the resulting neural net can compute a classification decision using only a few hundred clock cycles; thus, the trained neural net classifier can definitely be part of real-time operation. Moreover, this neural net converges within less than 100 training epochs requiring at worst a few thousand clock cycles per training sample. Therefore, the computational demands for the training operation of our system are modest as well, and it is feasible to carry them out automatically, using either offpeak times or other schemes, including real-time operation.

In [43], we studied the performances of five different types of neural networks: perceptron, back propagation (BP), perceptron-back propagation hybrid (PBH), radial-based function (RBF), and fuzzy ARTMAP. Our experimental results showed that the BP and PBH networks had stronger classification. Following this conclusion, we are using a BP network with two hidden neurons. This network is very small and efficient, but still has very good classification capability.

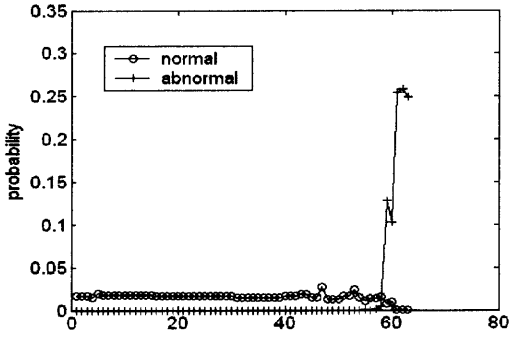
In FNIDS, neural network classifier maintains reference model of flow statistics features (22 features are using as in section 4.3) during training stage, calculates similarity metrics for scoring distance of current flow from reference model, uses back-propagation neural network for classifying each flow as normal or abnormal.

4.4.1 Reference Models

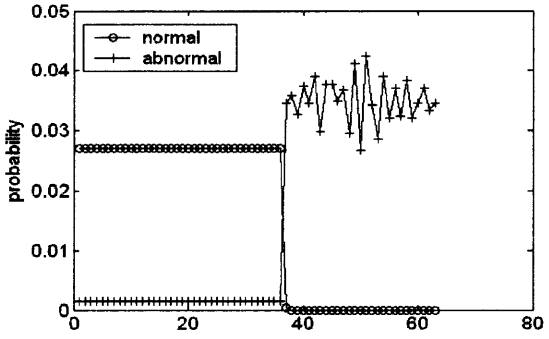
In our FNIDS, the extracted flow features will be scored based on the information of the reference models. The reference models are in fact probability density functions (PDF) of the feature values. Figure 4.2 showed the measured normal-type and attack-type PDF samples of different features, which were based on Darpa 98 data of the 5th and 6th Thursday, including DoS/DDoS, like 'POD', 'Teardrop',' Neptune' and 'Smurf'. Since the actual legitimate flow distribution during attack is unknown, the approach needs to establish a reference profile of normal users and the reference model of attackers from the past traffic. Just only off-line reference model is not enough. We ensured that the reference model would be updated actively in each time window. While attack packets arrive continuously and the true flow profile changes as new packets arrive, the reference profile is updated and flow is scored at the same time.

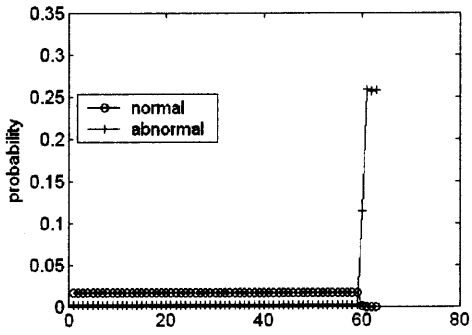


(a) IP packet rate (Week 5, Thursday)



(b) IP packet rate (Week 6,Thursday)





(c) ICMP packet rate (Week 5, Thursday) (d) ICMP packet rate (Week 6, Thursday)

Figure 4.5 The PDF of various packet rates based on Darpa 98 data set.

Evidently, as seen from the graphs, there is some overlap between the normal-type PDFs and the abnormal-type PDFs. This means that single parameter threshold classification is error prone. However, it is most significant that, in general, the normal-type and the abnormal-type PDFs are very different from each other at some features. This means that statistical methods that capitalize on these differences can be very effective, especially, if many or all of the features were utilized in unison in arriving at the classification decision. This is exactly how the technique used here and the resulting tool, termed Flow-based Network IDS Intrusion Detection System (FNIDS), achieves its high rate of success.

4.4.2 Score Metrics

There are many statistic metrics used in NIDS. Our current statistical model uses Single Number Statistics (SNS). Let x be the feature value, $p_n(x)$ be the probability of x in the normal reference model, $p_a(x)$ be the probability of x in the attack reference model, $p_{n,max}$ be the maximum probability of the normal reference model, and $p_{a,max}$ be the maximum probability of the attack reference model. The distance of the SNS is defined as the following metrics:

Single Number Statistics version 1(SNS1):

$$S(x) = 2\frac{P_n(x)}{P_{n,\max}} - 1 \tag{4.1}$$

Single Number Statistics version 2(SNS2):

$$S(x) = \frac{P_n(x)}{P_{n,\max}} - \frac{P_a(x)}{P_{a,\max}} \tag{4.2}$$

Single Number Statistics version 3(SNS3):

$$S(x) = 1 - 2\frac{P_a(x)}{P_{a,\max}} \tag{4.3}$$

Where: S is between [-1,1]. S =1 means 'normal'; and S= -1 means 'abnormal'

From the above equations, it can be seen that metric 1 is an anomaly detection approach by comparing feature values with normal reference model. The metric 3 is a normal detection approach by comparing feature value with abnormal reference model. The metric 2 is a hybrid signature-anomaly detection approach by utilizing both known normal and known attack knowledge. Apparently, the Metric 1 will be more robust to detect new attacks; especially it can identify the low-rate attack without retraining neural network, because the reference model of this way is only based on normal packets.

There are evidently, two kinds of feature vectors here, the normal N-type, labeled as +1, and the malicious M-type, labeled as –1. These labeled N and M vectors can be used for training (2/3 of the total number) and validating (1/3 of the total number) the classifier. In this work, a neural network classifier was employed.

Table 4.2 Performance of Difference Metrics on CONEX Testbed Data using Session-based Aggregation Scheme

	SNS1	SNS2	SNS3
Number of samples	100701	100701	100701
Number of normal samples	99429	99429	99429
Number of attack samples	1272	1272	1272
Number of False Positives	12	100	363
Number of False Negatives	0	0	0
Misclassification Rate	0.000119165	0.000993039	0.00360473
False Positive Rate	0.000120689	0.00100574	0.00365085
False Negative Rate	0	0	0

In our experiments, we compared these metrics. Form Table 4.2, the results showed that all these metrics can detect Dos. But the Table 4.2 shows that SNS2 and SNS3 have high false positive. Since SNS1 has a good performance, and it is only based on normal behaviors, it will be used in our following discussion.

4.4.3 Neural Network Architecture

Neural network classifiers have been widely considered as an efficient approach to classify challenging patterns. However, here, statistical preprocessing has generated easily discernible and distinguishable normal and abnormal patterns. A 2-layered neural network was built with 22 inputs and one output unit.

The back-propagation neural network classifier [47] (see Figure 4.6) was used to train feature-scoring vectors by modifying the weights of inputs. Each network utilized sigmoid neurons and used on all 22 predictor variables. Again, this was done 1000 times for each random split of the data set .The Back-Propagation Neural Networks was used to evaluate each flow with maliciousness.

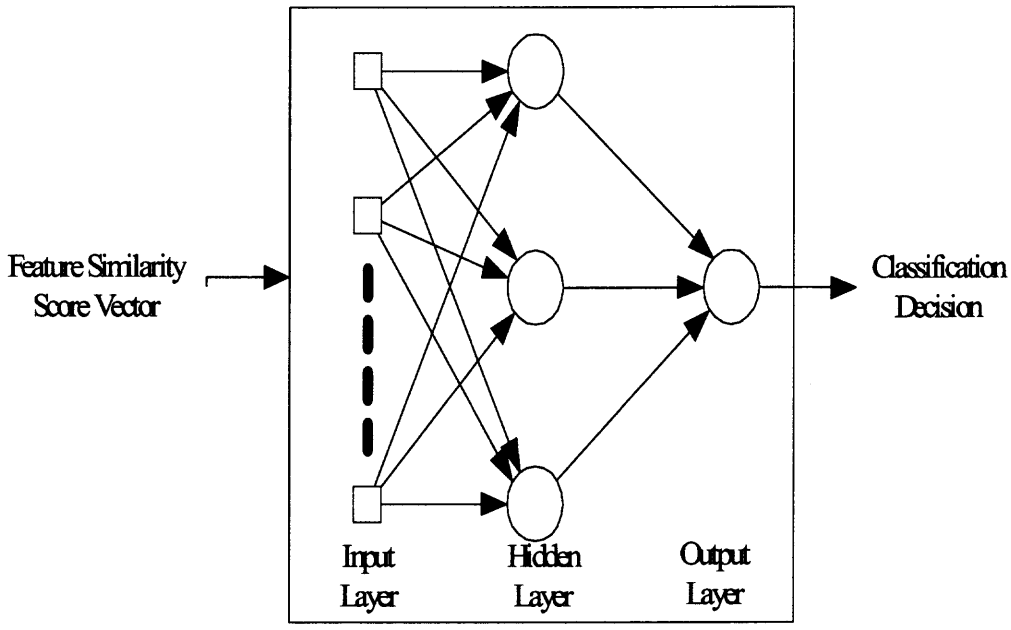


Figure 4.6 Back-propagation classifier.

4.5 Performance Evaluation And Results

The experimental test results to date have emphasized DoS/DDoS attacks and faults due to their prominent role in computer networks, in both wired and wireless ad hoc networks. Two main categories of DoS/DDoS attacks are included: bandwidth depletion, such as: udp flooding, ICMP flooding, smurf, and resource depletion attacks, such as: Syn flooding, Ping of Death and Teardrop. A bandwidth depletion attack is designed to flood the victim network with unwanted traffic that prevents legitimate traffic from reaching the (primary) victim system. A resource depletion attack is an attack that is designed to tie up the resources of a victim system. This type of attack targets a server or process on the victim system making it unable to process legitimate requests for service.

DARPA'98 evaluation represents a significant contribution to the field of intrusion detection, there are many unresolved issues associated with its design and execution. In his critique, McHugh [49] questioned a number of results of DARPA'98 evaluation, starting from usage of synthetic simulated data for the background (normal data) and using attacks implemented via scripts and programs collected from a variety of sources. In addition, it is known that the background data contains none of the background noise (packet storms, strange fragments, etc.) that characterizes real data. Moreover, in order to assess the performance of our anomaly detection algorithms in a live network, we evaluated our FNIDS in CONEX test-bed. The CONEX TESTBED network is a test-bed network setup in the CONEX lab of NJIT as a platform to launch network-based attacks and real background traffic (HTTP, FTP, SMTP...), for evaluation of intrusion detection prototypes.

In order to demonstrate the efficiency and robustness of FNIDS, along with its general-purpose applicability in networking environments with different characteristics and topologies, in this section we present some experiments and the corresponding numerical results we obtained applying our proposed system in two different networking environments as described in the following sections:

- A wired conventional network (CONEX wired test-bed)
- A wireless ad hoc network (CONEX MANETBED)

4.5.1 CONEX Wired Test-bed

Due to various limitations of DARPA'98 intrusion detection evaluation data discussed above [49], we have conducted our experiments on live network at the CONEX lab of NJIT. The CONEX TESTBED network, as in Figure 4.7, is a test-bed network setup in

the CONEX lab of NJIT as a platform to emulate network-based attacks in order to test the performance of intrusion detection prototypes. The network includes three subnets: victim subnet, background subnet and the attack subnet. A more detailed description about the network topology, the attack emulations and the related tools can be found in [46].

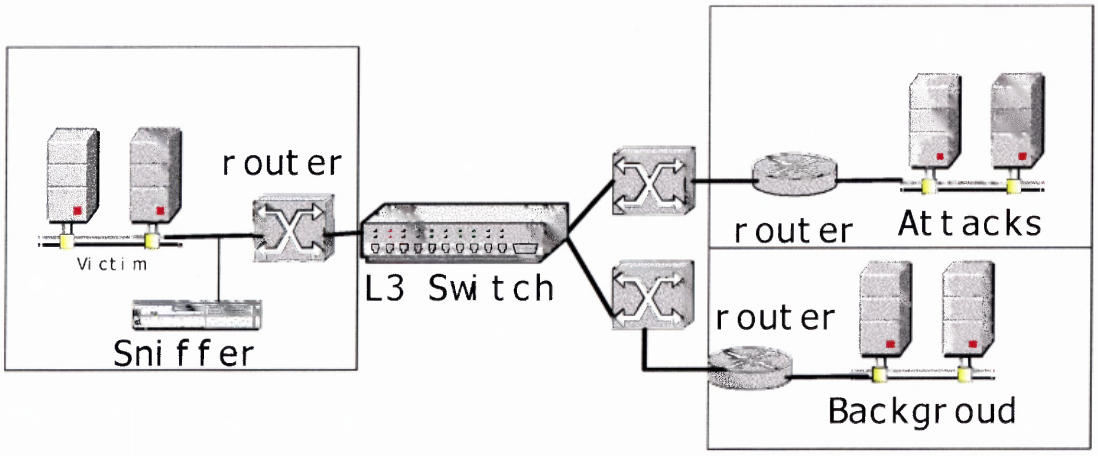


Figure 4.7 Topology of the CONEX Testbed network.

4.5.2 CONEX MANETBED

In order to demonstrate the applicability of our proposed approach as a general-purpose anomaly detection tool, we tested it in a dynamic mobile wireless ad hoc network. Due to the characteristics of open medium, dynamic changing topology, cooperative algorithms, and lack of a centralized monitoring and management point, wireless ad hoc networks are particularly vulnerable.

Mobile nodes are autonomous units that are capable of roaming independently. This means that nodes with inadequate physical protection may be captured, compromised, or hijacked. Wireless ad hoc networks do not have a clear line of defense,

and every node must be prepared for encounters with an adversary directly or indirectly; in other words, all other nodes must be treated as untrustworthy.

In [32], we describe the design and implementation of a novel low cost MANET emulation test-bed called Manetbed, which is based on low cost Ethernet environment. Manetbed provides mobile multi-hop wireless emulation including routing protocols (currently CBRP and AODV), random waypoint mobility, random packet loss, bandwidth limitation and communication delay. Manetbed infrastructure is extensible, easy maintenance especially for large network, facilitating the development and evaluation of new routing protocols, traffic models, security services.

Manetbed is built on Linux hosts where mobile nodes in the MANET are represented by actual computers running the actual protocol software with the actual Ethernet interfaces connected to it. To approach a real large-scale wireless environment, we install multiple network interface cards (NICs) in each Linux host and each NIC functions as one mobile node. A piece of small software is inserted into the kernel between network layer and Ethernet interface, providing multi-hop wireless link emulation. Random way-point module is used to generate mobility information and connectivity status. We use Linux Traffic Control (*tc*) to set bandwidth on each link. We write a separate tool in the kernel to drop packet randomly to control wireless packet error rate.

Figure 4.8 shows architecture of Manetbed. Multiple NICs are installed in each Linux host. Each NIC is simulated as one mobile node. To maximize the number of nodes, we use Adaptec Quartet four-port 10/100 PCI Adapter ANA-6944A/TX which has four network ports on one PCI card. While the number of nodes a host can have depends

on the number of PCI slots in the motherboard, too many network connections may slow down the system. In current Manetbed, we have 12 Linux hosts simulating up to 12*12 (144) nodes at the same time.

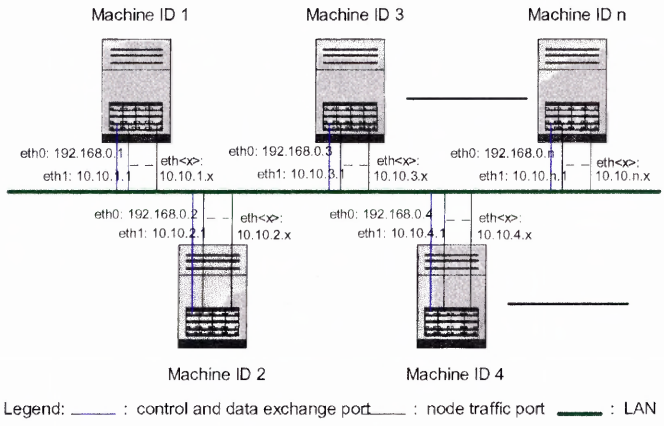


Figure 4.8 ManetBed layout.

The blue lines represent the control / data exchange port, and black lines represent node traffic port, respectively. These two different ports can be hooked to different LAN, or to the same LAN as what the figure shows

We have several kinds of traffic in the system and they can be divided into:

- • *Control and data exchange packet.* This is the traffic we send through the control interface (blue lines in Figure. 1) to collect data and co-ordinate CBRP nodes across the hosts.
- • Routing broadcast message (HELLO).
- • Routing message (RREQ, RREP, etc.).
- • *Traffic data.* This is the actual traffic data (ssh, ftp, HTTP, Smtp, telnet…) we run in Manetbed.

4.5.3 Experiments in Wired Test-bed

The Table 4.3 presents the false/true positive ratio of four aggregation schemes that detect various attacks. From this table, it can be seen that session-based aggregation schemes performed very well in detecting DDOS attacks with false positive ratio rates

0%. This indicates that session-based aggregation scheme has the best performance when there are no spoofing flooding attacks.

Table 4.3 False/True Positive Ratio in Wired Test-bed

Attack names	Session-based	IP-based	Client-Based	Server-Based
	False/True Positive Ratio	False /True Positive Ratio	False/True Positive Ratio	False/True Positive Ratio
bloop	0	0.037	0	0.037
fawx	0	0.36	0	0.36
fraggle	0	0	0	0
smurf	0	0.027	0.0106	0.027
teardrop	0	0	0	0

Table 4.4 Performances of Four Aggregation Schemes

	Session-Base	IP-Base	Client-Base	Server-Base
Number of samples	84943	25262	8397	22555
Number of normal samples	83898	24228	7786	22089
Number of attack samples	1045	1034	611	466
Misclassification Rate	0. 00401	0. 00134	0. 00107	0. 0047
False Positive Rate	0. 00398	0. 00103	0. 000771	0. 0048
False Negative Rate	0. 0096	0. 0088	0. 0049	0. 1148

From table 4.4, it can be seen that the misclassification rate is 0.401% for session-based aggregation, 0.102% for IP-based aggregation, 0.107% for client-based aggregation, 0.47% for server-based aggregation. The performances in table 3 proved that

the more detailed flow-key is, the performance is better when flooding attacks are not spoofed.

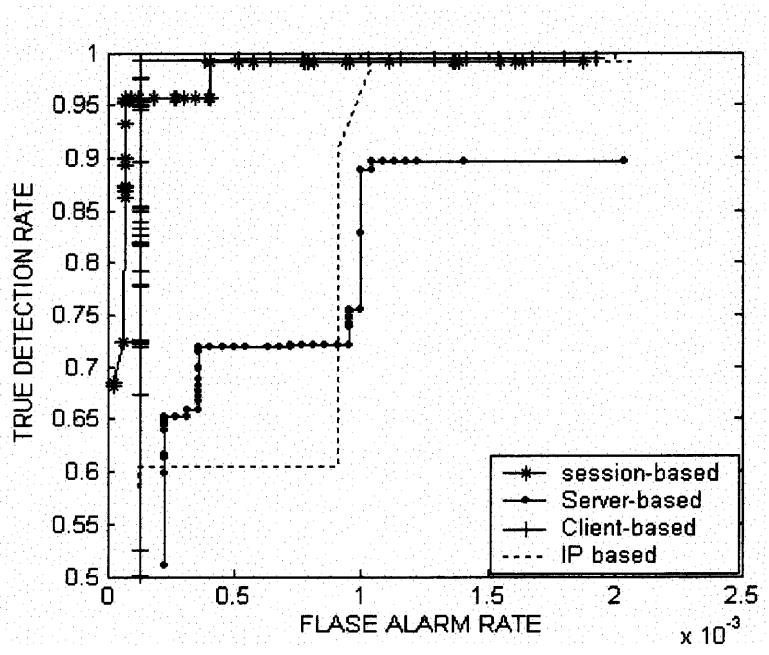


Figure 4.9 Receiver Operating Characteristic Curves.

Figure 4.9 shows the ROC (Receiver Operating Characteristic curves) of some selected flow-based aggregation schemes. we can observe that all of the ROC curves have very high detection rate. The detection rate is above 94%-99% when false alarm rate is about 0.2%.

4.5.4 Ad Hoc Wireless Experiments

In order to demonstrate the applicability of our proposed approach as a general-purpose anomaly detection tool, we tested it in a dynamic mobile wireless ad hoc network. We carried out simulation experiments on our ManetBed, with a 50-node 677*677 topology, and investigated FNIDS's detection effectiveness under various mobility scenarios, as well as various background traffic intensity levels of 20k, 50k, 100k, 200k, 500k and 1M

byte/second. We generate ICMP attack with average attack traffic as 1k byte/second, which gives attack intensities from 0.1 to 5 percent of background traffic.

It was found that FNIDS gives significantly high detection rates along with low misclassification rates as in Figure 4.10 in different traffic loads. Packet drop rate is 0.5 percent and maximum speed is 20m/s in these testing scenarios.

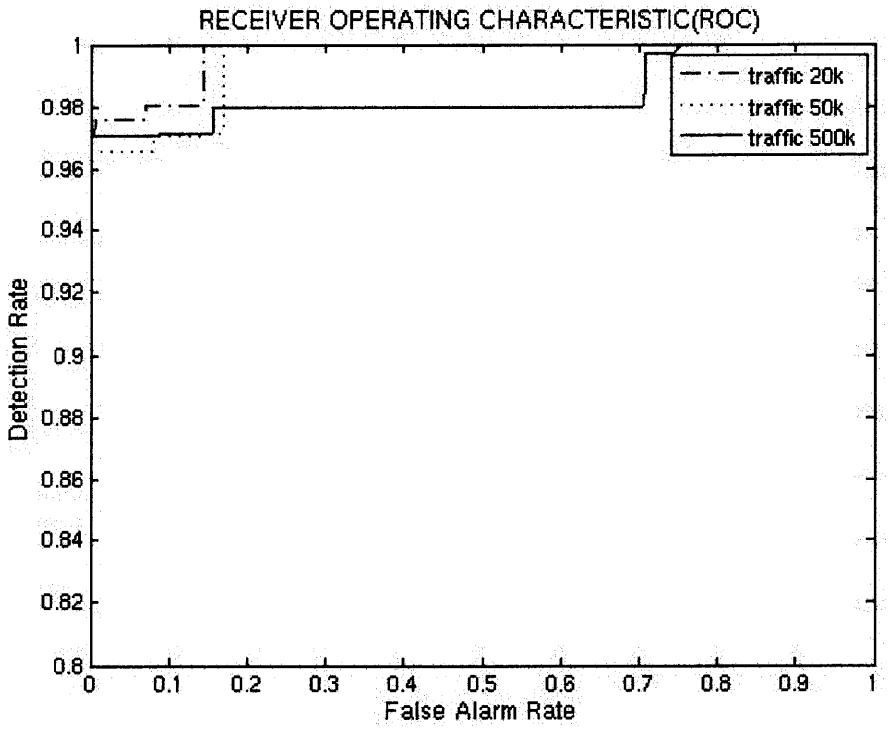


Figure 4.10 ROC curves for background traffic rate 20k, 50k, 500k.

Figure 4.10 depicts the corresponding receiver operating characteristic (ROC) curves. It shows significantly high detection rates along with low misclassification rates when the attack intensity is as small as 0.1-5 percent of the background traffic. Similar results have been found for other mobility speeds (e.g., 20 and 80 km/h). In summary, these results indicate that the performance of our anomaly intrusion detection system performs well in the challenging setting of the mobile ad hoc network.

4.6 Conclusion

The FNIDS statistical anomaly detection technology is characterized by several innovative features. It is a hierarchical multitier multi-observation window system that monitors several network traffic flows simultaneously using realtime parameters, collected during the observation window. It calculates a similarity value of each measured flow to the reference PDF, then intelligently combines the similarity measurements into an anomaly status vector that is classified by a neural network. This combining is powerful in that it achieves higher discrimination and decision robustness. The numerical results demonstrate that this anomaly detection methodology can reliably detect attacks and soft faults with traffic anomaly intensity as low as 0.1–5 percent of typical background traffic intensity, thus promising to generate an effective early warning in both wired and wireless networks..

CHAPTER 5

QUANTITATIVE SECURITY ASSESSMENT OF MANS

5.1 Introduction

We proposed in this dissertation a unified security architecture (MANS), under which IDS agent, authentication, recovery policy and other policies can be defined formally and explicitly, and are enforced by a uniform architecture, which provides protection of Ad Hoc networks basically focus on aspects: Intrusion Prevention, Intrusion Detection, and recovery response. As a basic preventive solution, a fully localized, digital certificate base authentication services is proposed. An intrusion Detection and response system (IDS) provides the corrective solution feeding the certificate services with information about misbehaving nodes, which are eliminated from the network by certificate revocation. Both certificate services and IDS are designed to be robust even in the presence of compromised nodes. We also presented evaluations for each aspect in previous chapters.

In this chapter, we are going to assess security risk of MANET protected by MANS. We consider a unified consistent scheme of vulnerabilities, threats in MANET and corresponding countermeasures provided by MANS. A quantitative risk assessment provides results in numbers that management can understand, whereas a qualitative approach, although easier to implement, makes it difficult to trace generalized results.

In [53], authors propose a modification of some of the decision-tree-based model's qualitative attributes, in case the quantitative data are unavailable. The proposed model is practical and simple to use for beginners in the field, but it also provides a mathematical–statistical foundation on which strategists or practitioners can construct a

practical risk valuation. The probabilistic assumptions, such as using a uniformly distributed random variable for the input variables, can be improved by using other statistical distributions. Other techniques used hitherto within a nonprobabilistic frame, such as attack trees, don't provide an accurate overall picture of the risk to the system that's being protected

We applied the above proposed security-meter design in this chapter for quantitative risk evaluation of MANS, favorably compared to most current assessments that provide qualitative results. This is achieved by a probabilistically accurate quantitative model that measures security risk. To further facilitate security risk management and security testing, the final risk measure calculated as a percentage which can be tested, improved, compared, and budgeted as opposed to attributes such as high, medium, or low, which cannot be managed or quantified numerically for an objective assessment.

This chapter is organized as following: Section 5.2 discusses specific attacks to Ad Hoc Routing, which was not covered in previous chapters; Section 5.3 overview quantitative security model; Section 5.4 represents Quantitative Analysis of MANS;

5.2 Ad Hoc Routing Vulnerability and Protection by MANS

In previous chapters, we discussed vulnerabilities, attacks which popularly happening for wired network, such as spoofing, denial of services attacks. This section, we mainly focus on specific attacks to Ad Hoc Routing. Similar to wired network routing, there are two kinds of attacks toward Ad Hoc routing protocols: passive attacks, and active attacks [65].

5.2.1 Passive Attacks

Passive attacks typically involve unauthorized "listening" to the routing packets. That is, the attacker does not disrupt the operation of a routing protocol but only attempts to discover valuable information by listening to the routing traffic.

The major advantage for the attacker in passive attacks is that in a wireless environment the attack is usually impossible to detect. This also makes defending against such attacks difficult. Furthermore, routing information can reveal relationships between nodes or disclose their addresses. If a route to a particular node is requested more often than to other nodes, the attacker might expect that the node is important for the functioning of the network, and disabling it could bring the entire network down.

Other interesting information that is disclosed by routing data is the location of nodes. Even when it might not be possible to pinpoint the exact location of a node, one may be able to discover information about the network topology.

5.2.2 Active Attacks

To perform an active attack the attacker must be able to inject arbitrary packets into the network. The goal may be to attract packets destined to other nodes to the attacker for analysis or just to disable the network. A major difference in comparison with passive attacks is that an active attack can sometimes be detected. This makes active attacks a less inviting option for most attackers.

Next we describe some types of active attacks that can usually be easily performed against an ad hoc network.

- Black Hole -- A malicious node uses the routing protocol to advertise itself as having the shortest path to nodes whose packets it wants to intercept. In a flooding based protocol such as AODV, the attacker listens to requests for routes. When the attacker receives a request for a route to the target node, the attacker creates a reply where an extremely short route is advertised. If the malicious reply reaches the requesting node before the reply from the actual node, a forged route has been created. Once the malicious device has been able to insert itself between the communicating nodes, it is able to do anything with the packets passing between them. It can choose to drop the packets to perform a denial-of-service attack, or alternatively use its place on the route as the first step in a man-in-the-middle attack.

- Rushing attack -- Some routing protocols such as AODV instantiate and maintain routes by assigning monotonically increasing sequence numbers to routes toward specific destinations. In AODV, any node may divert traffic through itself by advertising a route to a node with a destination sequence number greater than the authentic value [66]. Even the source node eventually receives the legitimate ROUTE REPLY (RREP) packets, it will discard those packets, thinking that the valid route is stale. Hence, the source node would not be able to find secure routes, that is, routes that do not include the adversary node.

- Wormhole attack -- In the wormhole attack, an attacker records packet at one location in the network, tunnels them to another location, and retransmits them from there into the network. Due to the broadcast nature of the radio channel, the attacker can create a wormhole even for packets not addressed to itself. If a wormhole attacker tunnels all packets through the wormhole honestly and reliably, no harm is done; the attacker actually provides a useful service in connecting the network more efficiently [66]. However, when an attacker forwards only routing control messages, this attack might severely disrupt routing. For example, when used against an on-demand routing protocol such as DSR or AODV, a powerful application of the wormhole attack can be mounted by tunneling each RREQ packet directly to the target node of the RREQ. This attack prevents any node from discovering routes more than two hops long.

- Spoofing attacks -- Spoofing occurs when a node misrepresents its identity in the network, such as by altering its MAC or IP address in outgoing packets [65]. By masquerading as another node, a malicious node can launch many attacks in a network. Spoofing combined with packet modification is really a dangerous attack, for example, it can cause routing loops in ad hoc networks.

- Route Error Fabrication -- AODV implements path maintenance to recover broken paths when nodes move. If the destination node or an intermediate node along an active path moves, the node upstream of the link break broadcasts a route error message to all active upstream neighbors. The node also invalidates the route for this destination in its routing table. The vulnerability is that routing attacks can be launched by sending false route error messages, causing other

nodes delete the valid entry in their routing table, therefore disrupt the communications. Such attacks can be difficult to verify as invalid constructs, especially in the case of fabricated error messages that claim a neighbor cannot be contacted [66].

- Routing Table Overflow -- In a routing table overflow attack the attacker attempts to create routes to nonexistent nodes. The goal is to create enough routes to prevent new routes from being created or to overwhelm the protocol implementation. Proactive routing algorithms attempt to discover routing information even before it is needed while a reactive algorithm creates a route only once it is needed. This property appears to make proactive algorithms more vulnerable to table overflow attacks. An attacker can simply send excessive route advertisements to the routers in a network. Reactive protocols, such as AODV on the other hand, do not collect routing data in advance.

- Sleep Deprivation Torture -- Usually, attack is practical only in Ad Hoc networks, where battery life is a critical parameter. Battery powered devices try to conserve energy by transmitting only when absolutely necessary. An attacker can attempt to consume batteries by requesting routes, or by forwarding unnecessary packets to the node using, for example, a black hole attack. This attack is especially suitable against devices that do not offer any services to the network or offer services only to those who have some special credentials. Regardless of the properties of the services, a node must participate in the routing process unless it is willing to risk becoming unreachable to the network.

- Location Disclosure -- A location disclosure attack can reveal something about the locations of nodes or the structure of the network. The information gained might reveal which other nodes are adjacent to the target, or the physical location of a node. The attack can be as simple as using an equivalent of the trace route command on Unix systems. Routing messages are sent with inadequate hop-limit values and the addresses of the devices sending the ICMP error-messages are recorded. In the end, the attacker knows which nodes are situated on the route to the target node. If the locations of some of the intermediary nodes are known, one can gain information about the location of the target as well.

5.2.3 MANS Routing Protocol Protection

As we discussed in previous chapters, the architecture of the Mobile Ad-hoc Network Security (MANS) System, includes authentication agent, IDS agent, recovery policy and other policies which can be defined formally and explicitly, and are enforced by a uniform architecture. As a basic preventive solution, a fully localized, digital certificate

base authentication services is proposed. An intrusion Detection and response system (IDS) provides the corrective solution feeding the certificate services with information about misbehaving nodes, which are eliminated from the network by certificate revocation. Both certificate services and IDS are designed to be robust even in the presence of compromised nodes.

In our design, protection of the routing protocol includes both preventive and corrective security services. A certificate-based authentication service for the routing protocol messages is considered as a basic preventive solution. The authentication service aims to avoid an attack to be generated from a nonauthenticated node. However, according to the presumed adversary model, attacks are still possible in two situations: (1) an authenticated node (e.g. certificate holder) starts to behave maliciously; or (2) a MANET node has been compromised and the authentication secret (e.g. private key) from that node has been exposed.

The corrective security service is provided in terms of an intrusion detection and response system (IDS). Intrusion response consists mainly in the isolation of malicious/compromised nodes, excluding them from the routing service. This is accomplished basically by means of certificate revocation. Certification services and intrusion detection and response should be provided in a self-organized and distributed manner by a Local Certification Service (LEGT) and a Local IDS (FNIDS and RIDS) instances. Figure 5.1 illustrates the proposed protection model.

Basically, routing protocol, certification service and IDS (alert) message exchanges must be authenticated with a MANET authentication extension (MAE), which is appended to each message and provides the authentication information. Authentication

is based on the certification service and uses asymmetric cryptography primitives. Each node in the MANET must hold a valid certificate, binding the node's identity to its public key.

In order to maintaining the robustness of the security solution in the presence of compromised nodes, security services in our system are designed to have k-by-n security, in the sense that any certification service or intrusion response must be collaboratively provided by, at least, k nodes, where n is the total (nonfixed) number of nodes in the network. Thus, for compromising a security service, an adversary must break into k different nodes. Correct nodes running the IDS must detect the attacks against the routing protocol and isolate the compromised node (by revoking its certificate) before that an adversary can compromise k nodes, breaking the collaborative security system.

Given that collaboration is done by means of authenticated messages (certificate services and IDS messages are all authenticated using MAE), isolating a node is equivalent to revoking the node's certificate. An indirect revocation mechanism is related to certification expiration. Thus, security can be improved if we require that certificates must be renewed from time to time. Certificates are issued with constrained certificate expiration time. Each node having a valid certificate must request for a new certificate, before the current certificate has expired. Nodes that are not well behaving should not have their certificates renewed.

Finally, locality requirement states that collaboration should be designed to restrict communications among group (e.g. local-enforced certificate services and IDS) around nearby nodes, usually in the local neighborhood. This is an important requirement as it relates to the scalability of the overall solution. Considering the locality requirement

stated above, k becomes an important parameter and should be related to the average size of neighborhoods in the network. If a node has k or more neighbors, IDS and certification services can be fully provided in the local neighborhood. Thus, the security solution is scalable, in the sense that security services are run locally, provided a convenient choice for parameter k.

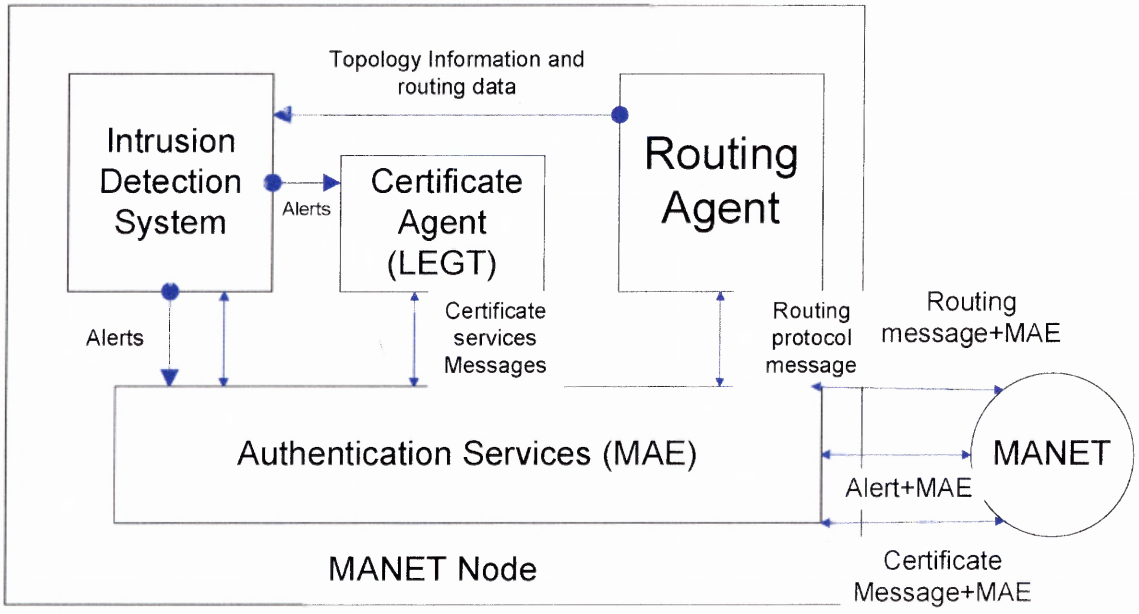


Figure 5.1 Routing protocol protection model.

5.3 Overview of a Quantitative Model

Conventionally, risk scenarios involve possible chance-based catastrophic failures with scarce modeling of maliciously designed human interventions that threaten inherent system vulnerabilities. Risk scenarios concerning critical computer communication networks are now more pervasive and severe than ever before because of the cost of nonmalicious chance failures that occur due to insufficient testing and lack of adequate reliability. We can use software reliability modeling and testing techniques to examine

these chance failures in more detail. However, for the intentional failures or malicious activities that critically increase the risk of illdefined attacks, no one has ever thoroughly modeled a physical scenario, at least not one that considers a unified consistent scheme of vulnerabilities, threats, and countermeasures. A quantitative risk assessment provides results in numbers that management can understand, whereas a qualitative approach, although easier to implement, makes it difficult to trace generalized results.

In [53], authors propose a modification of some of the decision-tree-based model's qualitative attributes, in case the quantitative data are unavailable. The proposed model is practical and simple to use for beginners in the field, but it also provides a mathematical–statistical foundation on which strategists or practitioners can construct a practical risk valuation. The probabilistic assumptions, such as using a uniformly distributed random variable for the input variables, can be improved by using other statistical distributions. Other techniques used hitherto within a nonprobabilistic frame, such as attack trees, don't provide an accurate overall picture of the risk to the system that's being protected

Three ingredients are involved in the quantitative model and their probabilities are used as input to calculate the residual risk of a system [53], as shown in Figure 5.1 They are vulnerability, threat, and countermeasure (CM). A system may have several vulnerabilities that could be exploited by an attacker. If the probability of the *ith* vulnerability is V_i, then $\sum V_i = 1$. Threat is any circumstance or event with the potential to adversely impact a system, through unauthorized access, destruction, disclosure, modification of data, or denial of service. For each vulnerability, there may have several threats and $\sum T_i = 1$, where T_i is the probability of the *ith* threat to a specific

vulnerability. A CM is a measure that reduces risk of a thread to a system. Consequently, the residual risk is the portion of risk remaining after a CM is applied. Residual risk could be "none" if a perfect CM exists. The residual risk of a system is calculated is illustrated in Figure 5.2.

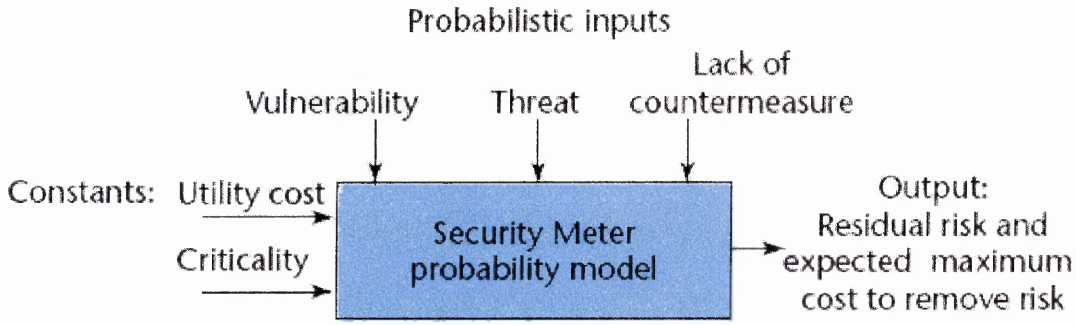


Figure 5.2 The quantitative security-meter probability model.

5.4 Quantitative Analysis of MANS

In this section we conduct the quantitative analysis on MANS using the model introduced in last section. We firstly summarized the vulnerabilities and the threats in MANETs, and the corresponding countermeasures provided by MANS, in Table 5.1.

Next, we assign the probability for each element. To simplify, we assume that all vulnerabilities have the same probability to be exploited by an attack. That is, if there are n vulnerabilities in a system, the probability that each vulnerability could be exploited is 1/n. The same idea applies on the threat probability assignment. Also, we assume that if a threat is completely prevented by a countermeasure, the probability of lack of countermeasure is 0; if a threat is partially prevented by a countermeasure, the probability of countermeasure is 0.5 and the probability of lack of countermeasure is 1-0.5=0.5; if there is no any countermeasure to a threat, then the probability of lack of countermeasure

is 1. In MANS, the probability input could vary, depending on the actual countermeasure (FIDS, neighborhood watch, LEGT, etc.) in actual scenarios.

Table 5.1 Probabilistic Input Data for Vulnerabilities, Threats and Countermeasures in MANS

Vulnerability	Threat	Countermeasure(CM)	LCM
Routing protocol V1=1/5	Malicious routing query Flooding to non-exist nodes Fabrication of error messages Spoofing	MAE FNIDS true detection	FNIDS false negative
Services Availability V2=1/5	ICMP flooding UDP flooding Fragmentation attack Syn flooding	FNIDS true detection for each kind of attack	FNIDS false negative for each kind of attack
Probe & scan V3=1/5	Fin scan Tcp Scan Null Scan Stealthy Scan	RIDS true detection for each kind of attack	RIDS false negative for each kind of attack
confidentiality V4=1/5	Passive attack eavesdropping	AES session key	
Authentication, V5=1/5	Error-prone wireless Link Compromised nodes (less than t nodes) Compromised nodes (more than t nodes)	CM41=Certificate renew successful ratio CM42=1 CM43=0	LCM41=1-certificate renew successful ratio LCM42=0 LCM43=1

Note:

1) $\sum v_i = 1$

2) $\sum t_{ij} = 1$ for every v_i

3) LCM+CM=1 for every t_{ij}

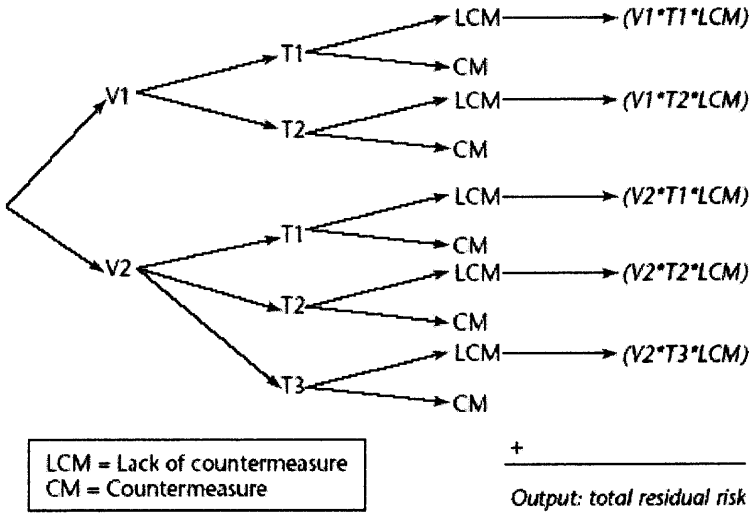


Figure 5.3 Example of quantitative risk assessment model.

According to the example showed in figure 5.2, we calculate the residue risk after applying MANS as routing protocol for the ad hoc network where have vulnerability and threat parameters defined as above.

$$Total\,\mathrm{Re}sidueRisk = \sum_{i} v_i * t_{ij} * LCM_{ij}$$

For example, for the scenario: mobility speed 20 m/s, packet drop rate 0.5%, background traffic 500kbps, the total residue risk is calculated as 0.0764.

In above discussion, we introduced a new evaluation method for MANS. Quantitative risk evaluation model could be used as a tool to conduct comparison between different scenarios, mechanisms. It offers clear view on the strength of a secure system.

CHAPTER 6

CONCLUSION

In this dissertation, the challenge of provision of security in Mobile Ad-Hoc Network (MANET) environments is tackled. The most important characterizing feature of a MANET is the absence of any node in a central role. So many security services that rely on central services, such as naming services, certification authorities (CA), network-based intrusion detection and other recovery administrative services will be impossible or at least much harder to implement. Another nagging major challenge is that of the compromised node(s); this could be an overtaken attacked node or a physically captured node. This forces all MANET nodes to operate in a mode that "trusts no peer" that complicates and inhibits security services.

Protection of Ad Hoc networks basically focus on two aspects: Intrusion Prevention, such as traffic encryption, sending data through multiple paths, authentication and authorization; Intrusion Detection, such as Anomaly pattern examination, Protocol analytical study. However, intrusion prevention cannot guard against internal attacks when an adversary has broken in, while intrusion detection itself is not enough because: Detecting intrusion must compiled with recovery response to remove the malicious hosts; IDS can only rely on partial and local data because of lack of traffic concentration, such as routers and switches in Ad Hoc networks; Separation between normal and abnormal is difficult in Ad Hoc networks. Moreover, secure routing is another important protection for Mobile Ad Hoc networks because routing protocol is an easy target of tearing down whole network functionality.

In this dissertation, the architecture of the Mobile Ad-hoc Network Security (MANS) System is proposed, under which authentication agent, IDS agent, recovery policy and other policies can be defined formally and explicitly, and are enforced by a uniform architecture. As a basic preventive solution, a fully localized, digital certificate base authentication services is proposed. An intrusion Detection and response system (IDS) provides the corrective solution feeding the certificate services with information about misbehaving nodes, which are eliminated from the network by certificate revocation. Both certificate services and IDS are designed to be robust even in the presence of compromised nodes.

This novel authentication scheme (LEGT) is based on asymmetric cryptographic techniques, specifically RSA algorithms. Authentication service is provided by taking a *certificate-based* approach.

An IDS is installed in every node, which is responsible for colleting local data from its host node and neighbor nodes within its communication range, pro-processing raw data and periodically broadcasting to its neighborhood, classifying normal or abnormal based on pro-processed data from its host node and neighbor nodes.

Security recovery policy in ad hoc networks is the procedure of making a global decision according to messages received from distributed IDS and restore to operational health the whole system if any user or host that conducts the inappropriate, incorrect, or anomalous activities that threaten the connectivity or reliability of the networks and the authenticity of the data traffic in the networks.

Manet routing protocol is also secured by combing Certificate-based authentication and Intrusion Detection in MANS architecture. Secure routing is not main

concern of MANS but by providing the architecture of interaction between authentication agent, routing agent, Intrusion detection agent, MANS is capable of secure routing.

Finally, quantitative risk assessment model is also proposed to numerically evaluate MANS security, to comprehensively analyze vulnerabilities, threats, countermeasures and lack of countermeasures of MANS.

In the future, MANS extension in following applications will be further explored:

1) False positive mitigation for wireless environments.
2) Countermeasure for some specific routing attacks, such as Blackhole, Wormhole.
3) Parameter selection for routing attacks.

APPENDIX

MONITORED STATISTICAL FEATURES

This appendix gives the detailed descriptions on the statistical features monitored by Flow-base Network Intrusion detection.

The FNIDS is capable of monitoring traffic into and out of the protected network. The FNIDS statistical features can be categorized based on the protocols of network traffic.

1. **IP Packet Length** measures the averages and the distributions of the IP packet lengths within a time window. For simplicity, this parameter is symbolized as "ip-pkt-len" afterward.
2. **IP Packet Rate** measures the averages and the distributions of the packet rates of all observed IP packets within a time window. This feature will be symbolized as "ip-pkt-rate" afterward.
3. **IP Byte Rate** measures the averages and the distributions of the byte rates of all observed IP packets within a time window. This feature will be symbolized as "ip-byt-rate" afterward.
4. **IP Fragment Rate** measures the averages and the distributions of the packet rates of all observed IP fragments within a time window. This feature will be symbolized as "ip-frag-rate" afterward.
5. **IP Defragmentation Error Rate** measures the averages and the distributions of the IP defragmentation error rates occurred within a time window. This feature will be symbolized as "ip-defrag-error" afterward.

6. **IP Checksum Error Rate** measures the averages and the distributions of the IP checksum error rates occurred within a time window. This feature will be symbolized as "ip-csum-error" afterward.
7. **TCP Invalid Packet Rate** measures the averages and the distributions of the rates of the TCP packets with invalid combinations of TCP control flags. This feature will be symbolized as "tcp-pkt-invalid".
8. **TCP Packet Length** measures the averages and the distributions of the lengths of IP packets within a time window. This feature will be symbolized as "tcp-pkt-len".
9. **TCP Packet Rate** measures the averages and the distributions of the TCP packet rates within a time window. This feature will be symbolized as "tcp-pkt-rate".
10. **TCP SYN Packet Rate** measures the averages and the distributions of the rates of TCP control packets with SYN flag set within a time window. This feature will be symbolized as "tcp-syn-pkt-rate" afterward.
11. **TCP FIN Packet Rate** measures the averages and the distributions of the rates of TCP control packets with FIN flag set within a time window. This feature will be symbolized as "tcp-fin-pkt-rate" afterward.
12. **TCP RST Packet Rate** measures the averages and the distributions of the rates of TCP control packets with RST flag set within a time window. This feature will be symbolized as "tcp-rst-pkt-rate" afterward.
13. **TCP Connection Open Rate** measures the averages and the distributions of the TCP connection open rates within a time window. This feature will be symbolized as "tcp-con-new-opened" afterward.

14. **TCP Connection Close Rate** measures the averages and the distributions of the TCP connection close rate within a time window. This feature will be symbolized as "tcp-con-new-closed" afterward.
15. **TCP Connection Abort Rate** measures the averages and the distributions of the TCP connection abort rate (connection closed by RESET or TIMEOUT other than normal three-way hand shaking) within a time window. This feature will be symbolized as "tcp-con-new-aborted" afterward.
16. **TCP Connections from Different Source Address** measures the distributions of TCP connections from different source IP addresses within a time window. This feature will be symbolized as "tcp-con-diff-src" afterward.
17. **TCP Connection to Different Destination Address** measures the distributions of TCP connections to different destination IP addresses within a time window. This feature will be symbolized as "tcp-con-diff-dst" afterward.
18. **TCP Connection Anomalous Entropy** measures the averages and the distributions of the anomalous entropies of all TCP connections within a time window. This feature was first proposed in Staniford [37]. The equation to calculate the connection anomalous entropy is given in Equation 7.1. This feature will be symbolized as "tcp-con-anomalous-entropy" afterward.
19. **TCP Connection Half Opened Ratio** measures the averages and the distributions of the ratio between the half-opened TCP connections and all TCP connections within a time window. This feature will be symbolized as "tcp-con-half-opened-ratio" afterward.

20. **TCP Connection Duration** measures the averages and the distributions of the TCP connection durations within a time window. This feature will be symbolized as "tcp-con-duration" afterward.
21. **UDP Packet Length** measures the averages and the distributions of UDP packets within a time window. This feature is symbolized as "udp-pkt-len" afterward.
22. **UDP Packet Rate** measures the averages and the distributions of UDP packets within a time window. This feature will be referred as "udp-pkt-rate" afterward.
23. **UDP Byte Rate** measures the averages and the distributions of UDP packets within a time window. This feature will be referred as "udp-byt-rate" afterward.
24. **UDP Packets from Different Sources** measures the distributions of UDP packets from different IP addresses. This feature will be referred as "udp-diff-src" afterward.
25. **UDP Packet to Different Destinations** measures the distributions of UDP packets destined to different IP addresses. This feature will be referred as "udp-diff-dst" afterward.
26. **ICMP Packet Length** measures the averages and the distributions of ICMP packet lengths within a time window. This feature will be referred as "icmp-pkt-len" afterward.
27. **ICMP Packet Rate** measures the averages and the distributions of ICMP packets within a time window. This feature will be symbolized as "icmp-pkt-rate" afterward.
28. **ICMP Packets from Different Sources** measures the distributions of ICMP packets originated from different IP addresses within a time window. This feature will be symbolized as "icmp-diff-src" afterward.

29. **ICMP Packet to Different Destinations** measures the distributions of ICMP packets destined to different IP addresses within a time window. This feature will be referred as "icmp-diff-dst" afterward.
30. **ICMP Anomalous Echo Replies** measures the averages and the distributions of anomalous ICMP echo replies, which are ICMP echo reply packets without previous echo request packets, within a time window. This feature will be referred as "icmp-anomalous-echo-reply" afterward.
31. **ICMP DUR Packet Rate** measures the averages and the distributions of ICMP DUR (destination-Unreachable) packets within a time window. This feature will be referred as "icmp-dur-pkt-rate" afterward.

REFERENCES

1. J. Hubaux, L. Buttyan, S. Capkun. The Quest for Security in Mobile Ad Hoc Networks. Proceedings of the ACM Symposium on Mobile Ad Hoc Networking and Commputing, Long Beach, CA, 2001.

2. Yongguang Zhang and Wenke Lee. Intrusion Detection in Wireless Ad Hoc Networks. Mobile Computing and Networking, pp. 275-283, 2000.

3. N.H. Minsky and V. Ungureanu. Law-governed Interaction: a Coordination and Control Mechanism for Heterogeneous Distributed Systems. TOSEM, ACM Transactions on Software Engineering and Methodology, pp. 273-305, July 2000.

4. P. Naldurg, R.H. Campbell and M.D. Mickunas. Developing Dynamic Security Policies. UIUCDCS-R-2002-2261, February 2002.

5. T.M. Cook, and J.C. Rosencrance. Work-related Musculoskeletal Disorders among Physical Therapists. Physical Therapy, pp. 827-835, 1976.

6. J.-P. Hubaux, L. Butty'an, and S. Capkun. The Quest for Security in Mobile Ad Hoc Networks. Proceedings of ACM Symposium on Mobile Ad Hoc Networking and Computing (MobiHOC), Long Beach, CA, August 2001.

7. L. Blazevic, L. Buttya, S. Capkun, S. Giordano, J. Hubaux, J. Boudec. Self-Organization in Mobile Ad Hoc Networks. The Approach of Terminodes, IEEE Communications Magazine, June 2001.

8. U. Maurer. Modelling a Public-Key Infrastructure. European Symposium on Research in Computer Security, Berlin, German, 1996.

9. T. Beth, M. Borcherding. Valuation of Trust in Open Networks. Proceedings of the European Symposium on Research in Computer Security, Brighton, UK, 1994.

10. F. Stajano and R. Anderson. The Resurrecting Duckling: Security Issues for Ad-hoc Wireless Networks. The 7th International Workshop on Security Protocols, LNCS 1796, Springer-Verlag, 1999.

11. F. Stajano. The Resurrecting Duckling – what next? The 8 th International Work-shop on Security Protocols, LNCS 2133, Springler-Verlag, 2000.

12. L. Zhou and Z.J. Haas. Securing Ad Hoc Networks. IEEE Network Magazine, vol.13, no. 6, 1999.

13. Adi Shamir. How to share a secret. Communications of ACM, pp. 612-613, 1979.

14. Y. Zhang and W. Lee. Intrusion Detection in Wireless Ad Doc Networks. The Six Annual ACM/IEEE Conference on Mobile Computing and Networking, Boston, MA, August 2000.

15. W. Diffie and M. Hellman. New Directions in Cryptography. IEEE Transactions on Information Theory, pp. 644-654, November 1976

16. L. Venkatraman, D. Agrawal. A Novel Authentication Scheme for Ad Hoc Networks. IEEE Wireless Communications and Networking Conference (WCNC), vol.3, pp. 1268-1273, 2000.

17. Andre Weimerskirch, Gilles Thonet. A Distributed Light-Weight Authentication Model for Ad-hoc Networks. V.2288 of LNCS, 2002.

18. E.Luo, S. Lu. Ubiquitous and Robust Authenticaion Services for Ad Hoc Wireless Networks. UCLA-CSD-TR-200030.

19. M. Steiner, G. Tsudik, and M. Waidner. CLIQUES: A New Approach to Group Key Agreement. Proceedings of 18th IEEE International Conference on Distributed Computing Systems, pp. 180-387, 1998.

20. Michael Steiner, Gene Tsudik, and Michael Waidner. Diffie-hellman Key Distribution Extended to Group Communication. In 3rd ACM Conference on Computer and Communications Security, pp. 31-37, New Delhi, India, March 1996.

21. Y. Frankel and Y. G. Desmedt. Parallel Reliable Threshold Multi-signature. Technical Report TR-92-04-02, Dept. of EECS, University of Wisconsin-Milwaukee, 1992.

22. A. Herzberg, S. Jarecki, H. Krawczyk, and M. Yung. Proactive Secret Sharing or: How to Cope with Perpetual Leakage. Proceeding of 15th Annual International Cryptography Conference, pp. 457-469, November 1995.

23. Y. Frankel, P. Gemmell, P. MacKenzie, and M. Yung. Optimal Resilience Proactive Public-Key Cryptosystems. IEEE Proceeding of Symposium on Foundations of Computer Science, pp. 384-393, Miami Beach, FL, 1997.

24. P. Feldman. A Practical Scheme for Non-interactive Verifiable Secret Sharing. IEEE Proceeding of Symposium on Foundations of Computer Science, pp. 427-437, 1987.

25. Mingliang Jiang, Jinyang Li, and Y.C. Tay. Cluster Based Routing Protocol. http://cram.comp.nus.edu.sg/cbrp/draft-ietf-manet- cbrp-spec-01.txt, Aug 1999.

26. Lakshmi Venkatraman and Dharma P. Agrawal. A Novel Authentication scheme for Ad hoc Networks. IEEE Wireless Communications and Networking Conference, vol.3, pp. 1268-1273, 2000.

27. Srdjan Capkun, Levente Buttyn, and Jean-Pierre Hubaux. Small Worlds in Security Systems: an Analysis of the PGP Certificate Graph. ACM Proceedings of the 2002 Workshop on New Security Paradigms, pp. 28-35, 2002.

28. A. Abdul-Rahman and S. Hailes. A Distributed Trust Model. Proceedings of 1997 ACM New Security Paradigms Workshop, pp. 48-60, 1997.

29. Second AES conference: http://csrc.nist.gov/CryptoToolkit/aes/round1/conf2/aes2conf.html

30. C.N. Manikopoulos and Li Ling. Architecture of the Mobile Ad Hoc Network Security (MANS) System. IEEE International Conference on Systems, Man and Cybernetics, vol.4, pp. 3122 – 3127, 2003.

31. Li Ling and C.N Manikopoulos. Manetbed: A Low-cost, Scalable Mobile Ad hoc Network Emulation Testbed. To be submitted.

32. Y. Frankel, P. MacKenzie and M. Yung. Robust Efficient Distributed RSA Key Generation. Proceedings of 30th ACM Annual Symposium on the Theory of Computing, pp. 663-672, 1998.

33. Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman. A Method for Obtaining Digital Signatures and Public-Key. Cryptosystems Communications of the ACM, pp. 120-126, February, 1978.

34. T. P. Pedersen. Non-interactive and Information Theoretic Secure Verifiable Secret Sharing. Advances in Cryptology - CRYPTO, pp. 129-140. SpringerVerlag, 1991.

35. M. Stadler. Publicly Verifiable Secret Sharing. Advances in Cryptology EUROCRYPT'96, pp. 190-199, 1996.

36. B. Schoenmakers, A Simple Publicly Verifiable Secret Sharing Scheme and its Application to Electronic Voting. Advances in Cryptology-CRYPTO'99, pp. 148-164, 1999.

37. A. Housebolder, K. Houle, and C. Dougherty. Computer Attack Trends Challenge Internet Security. Security and Privacy Supplement IEEE Comp. Magazine, pp. 5-7, April 2002.

38. G. Vigna and R. A. Kemmerer. NetSTAT: A Network Based Intrusion Detection Approach. Proceedings of 14th Annual Computer Security Application Conference, pp. 25-34, 1998.

39. A. Valdes and D. Anderson. Statistical Methods for Computer Usage Anomaly Detection Using NIDES. Technical Report, SRI International, Jan 1995.

40. J. P. Anderson. Computer Security Threat Monitoring and Surveillance. Technical Report, Fort Washington, PA, April 1980.

41. D. E. Denning. An Intrusion Detection Model. IEEE Transaction of Software Engineering, vol. SE-13, no. 2, pp. 222-232, 1987.

42. L.T. Heberlein, G.V. Dias, K.N. Levitt, B. Mukherjee, J. Wood, and D. Wolber. A Network Security Monitor. Proceedings of IEEE Symposium of Research in Security and Privacy, pp. 296-304, Oakland, CA, May 1990.

43. R. P. Lippman *et al.* Results of the DARPA 1998 Offline Intrusion Detection Evaluation. Proceedings of Recent Advanced Intrusion Detection (RAID '99) Conference, West Lafayette, IN, September 7-9, 1999.

44. L. Ho *et al.*, Adaptive/Automated Detection of Service Anomalies in Transaction WANs: Network Analysis, Algorithms, Implementation, and Deployment. IEEE JSAC, vol.18, no. 5, May 2000, pp. 744-57.

45. H. Debar, M. Becker and D. Siboni. A Neural Network Component for an Intrusion Detection System. IEEE Computer Society Symposium on Research in Security and Privacy, pp. 240-250, Oakland, CA, May 1992.

46. B. D. Joao *et al.*, Statistical Traffic Modeling for Network Intrusion Detection. Proceedings of 8th International Symposium of Modeling, Analysis Simulation Computing and Telecommunication System, pp. 466-73, August 2000.

47. Z. Zhang *et al.*, HIDE: A Hierarchical Network Intrusion Detection System Using Statistical Preprocessing and Neural Network Classification. Proceedings of 2nd Annual IEEE System, Man and Cybernetic Information. Assurance Workshop, West Point, NY, June 2001.

48. C. Manikopoulos *et al.*, Generalized Anomaly Detection in Next Generation Internet: Architecture and Evaluation. Submitted for publication, 2002.

49. W. Lee, S. Stolfo, and K. Mok. A Data Mining Framework for Building Intrusion Detection Models. Proceedings of 1999 IEEE Symposium of Security and Privacy, pp. 120-132, 1999.

50. Z.Zhang. Statistical Anomaly Denial of Service and Reconnaissance Intrusion Detection. PhD Dissertation, NJIT, 2004.

51. R. M. Dillon, C. N. Manikopoulos, Neural Net Nolinear Prediction for Speech Data, IEEE Electronics Letters, Vol. 27, Issue 10, pp. 824-826, May 1991.

52. Anukool Lakhina, Mark Crovella and Christophe Diot. Diagnosing Network-Wide Traffic Anomalies. Proceedings of ACM SIGCOMM, pp. 219-230, Portland, OR, 2004.

53. J. McHugh. The 1998 Lincoln Laboratory IDS Evaluation (A Critique). Proceedings of the Recent Advances in Intrusion Detection, pp. 145-161, Toulouse, France, 2000.

54. Mehmet Sahinoglu. Security Meter: A Practical Decision-Tree Model to Quantify Risk. IEEE Security and Privacy, Infrastructure Security, pp. 18-24, May 2005.

55. E. Forni. Certification and Accreditation. AUM Lecture Notes, Data Systems Design Laboratories, 2002.

56. B. Schneier. Applied Cryptography, 2nd ed., John Wiley & Sons, 1995.

57. K.A. Forcht. Computer Security Management. Boyd and Fraser, 1994.

58. Integrated Research in Risk Analysis and Decision Making in a Democratic Society. Workshop Report, US National Science Foundation, 2002.

59. M. Sahinoglu and E.H. Spafford. A Bayes Sequential Statistical Procedure for Approving Products in Mutation-Based Software Testing. Proceedings of IFIP Conference of Approving Software Products (ASP 90), W. Ehrenberger, ed., Elsevier Science Publishers, pp. 43-56, 1990.

60. M. Sahinoglu, J.J. Deely, and S. Capar. Stochastic Bayes Measures to Compare Forecast Accuracy of Software Reliability Models. IEEE Transaction of Reliability, vol. 50, no.1, pp. 92-97, 2001.

61. 7. M. Sahinoglu. An Empirical Bayesian Stopping Rule in Testing and Verification of Behavioral Models. IEEE Transaction of Instrumentation and Measurement, vol. 52, no. 5, pp. 1428-1443, 2003.

62. M. Sahinoglu et al., Measuring Availability Indices with Small Samples for Component and Network Reliability using the Sahinoglu-Libby Probability Model. To appear in IEEE Transaction of Instrumentation and Measurement, vol. 54, no. 3, June 2005.

63. M. Sahinoglu. Security Meter: A Probabilistic Framework to Quantify Security Risk. Certificate of Registration, US Copyright Office, TXU 1-134-116, December 2003.

64. S.A. Scherer. Software Failure Risk. Plenum Press, 1992.

65. B. Potter and G. McGraw. Software Security Testing. IEEE Security and Privacy, vol. 2, no. 5, pp. 81-85, 2004.

66. C. Siva, Ram Murthy and B.S. Manoj. Ad Hoc Wireless Networks: Architectures and Protocols. Prentice Hall, Upper Saddle River, NJ, 2004.