Light weight security architecture for ad hoc networks

Charles Joseph, John Felix

2010

Charles Joseph, J. F. (2010). Light weight security architecture for ad hoc networks.
Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/41881>

<https://doi.org/10.32657/10356/41881>

LIGHT WEIGHT SECURITY ARCHITECTURE FOR AD HOC NETWORKS

JOHN FELIX CHARLES JOSEPH

School of Computer Science and Engineering

A thesis submitted to the Nanyang Technological University
in fulfillment of the requirement for the degree of
Doctor of Philosophy

2010

Acknowledgements

I would like to express my profound gratitude and appreciation to my advisor, Prof. Lee Bu Sung, Francis, for providing invaluable wisdom and guidance during this research. His broad knowledge and deep insights helped me to choose the correct methodology to carry out this research. Special thanks and gratitude go to my co-supervisor, Prof. Amitabha Das for giving me invaluable advice every time I needed. He helped me to choose this wonderful research topic and pursue passionately. I hope and look forward to continuous collaboration with Prof. Lee and Prof. Das in the future.

I am especially grateful to Dr. Seet Boon Chong, whose close cooperation with me enabled me to solve many details of this research. Thanks to Prof. Dusit Niyato for the technical discussions about evolutionary game theory and its applications in our research. Their enthusiastic participation helped me solve numerous problems during my research.

Finally, I thank my family and friends for constantly supporting my research interests.

Contents

1	Introduction	1
1.1	Background and Motivation	1
1.2	Major Contribution	3
1.3	Organization	4
2	Related Work	6
2.1	Security Threat Analysis on Ad hoc Routing Protocol	7
2.2	Securing Routing Protocols using Cryptography	9
2.2.1	Hash Chain based Secure Routing Protocol	11
2.2.2	Digital Certificate based Secure Routing Protocol	13
2.3	Intrusion Detection Systems	16
2.3.1	Fundamental Design of IDS	17
2.3.2	IDS Architectures	22
2.3.3	Detection Methodologies	31
2.4	Trust Management	35
2.4.1	Theodorakopoulos and Baras's Trust Models	38
2.4.2	Game theoretic Cooperation Enforcement	39
2.4.3	E-Hermes	40
2.4.4	Dempster Shafer Approach	41
3	Threat model	43
3.1	Characteristics of Ad-hoc Networks	44

3.1.1	Lack of Fixed Infrastructure / Centralized nodes	44
3.1.2	Node and Link Properties	45
3.1.3	Energy and Delay Constraints	46
3.2	Routing Security in Ad hoc Networks	46
3.3	A Taxonomy of Ad hoc Network Routing Attacks	49
3.3.1	Elements of Attack Behavior	50
3.3.2	Attack Behavior	52
3.3.3	Attack Scenarios	53
3.4	Challenges in Detection of Routing Attacks	57
3.4.1	The Base Rate Fallacy	57
3.4.2	Theoretical Limitations	59
3.4.3	Limitations of Probabilistic Inference	64
3.5	Routing Protocol Vulnerability Analysis	65
3.5.1	OLSR Fundamentals	66
3.5.2	Protocol Analysis	67
3.5.3	Causal Relations - Effects and Behavior	70
3.5.4	Risk Estimation	73
4	Security Architecture Overview	76
4.1	Overview	76
4.2	Cross-Layer based Routing Attack Detection System (CRADS)	79
4.2.1	Data Collection Module	81
4.2.2	Data Reduction Module	81
4.2.3	Learning Module	82
4.2.4	Validation Module	82
4.3	Cross-Layer Based Adaptive Real-time Routing Attack Detection System (CARRADS)	82
4.3.1	Adaptive Incremental SVM (AI-SVM)	83
4.4	Game-Theoretic Trust Management Mechanism	84

4.5	Response	86
5	CRADS	88
5.1	CRADS Architecture	88
5.1.1	Configuration	89
5.1.2	Cross Layer Data Collection	90
5.1.3	Data Reduction Techniques	91
5.1.4	Learning	96
5.1.5	Validation	98
5.1.6	Detection	99
5.1.7	Complexity	100
5.2	Simulation Setup	101
5.2.1	Sinking Attack Scenarios	102
5.2.2	Spoofing/Fabrication Attack Scenarios	104
5.3	Results and Analysis	106
5.3.1	Efficiency of Data Reduction	107
5.3.2	Detection of Sinking Attacks	109
5.3.3	Detection of Spoofing Attacks	115
6	CARRADS	118
6.1	Need for Adaptive RADS	118
6.2	CARRADS	120
6.2.1	Data Collection Module	121
6.2.2	Behavior Identification Module	122
6.2.3	Environment Sensing	124
6.2.4	Cooperative Global Decision	125
6.2.5	Data Reduction Techniques	129
6.2.6	Learning Module	130
6.2.7	Incremental SVM Learning	130
6.3	Results	133

6.3.1	Experimental Setup	134
6.3.2	Detection Accuracy	137
6.3.3	Computational Overhead	137
7	Game Theoretic Trust Management Module	140
7.1	Game Theory	141
7.2	Game Theoretic Trust Model	143
7.2.1	Network and Game Setup	143
7.2.2	Trust Game Objective	144
7.2.3	Computation of Local Belief and Confidence	145
7.2.4	Trust Game Formulation	145
7.2.5	Payoff Matrix Structure	148
7.2.6	Payoff Matrix Values	150
7.2.7	Systematic Selection Criterion - ' <i>The Trust Game</i> '	153
7.2.8	Nash Equilibrium - Trust Game	154
7.3	Evaluation of Game Theoretic Trust Model	156
7.3.1	Experimental Setup	157
7.3.2	Performance Analysis	159
7.4	Evaluation of the Security Architecture	162
7.4.1	Experimental Setup	164
7.4.2	Effectiveness	164
8	Conclusion	169
8.1	Summary	169
8.2	Future Work	171
8.2.1	MAC and Physical Layer Attacks	173
8.2.2	Application/Transport Layer Attacks	174

List of Figures

2.1	Ad hoc Network Security Threats and Existing Routing Security Solutions .	7
2.2	A Simple Intrusion Detection System	17
2.3	Detection Accuracy Metrics	19
2.4	Linear Machine	21
2.5	LIDS Architecture	23
2.6	Secure Stationary Database Architecture	25
2.7	Modularized IDS Architecture	26
2.8	Distributed IDS Architecture	27
3.1	Ad-Hoc Routing Insecurity: Route Invasion	47
3.2	Routing Attacks Taxonomy	49
3.3	Black Hole Attack Scenario	52
3.4	Wormhole Attack Scenario	54
3.5	The Decision Dilemma in Ad hoc IDS	59
3.6	The Decision Dilemma in Ad hoc IDS in Asynchronous Channel	60
3.7	The Intractable Problem	61
3.8	Well-Known Solution to Decision Dilemma in Ad hoc IDS.	62
3.9	Decision Dilemma by Non-existent Neighbors	63
3.10	Protocol Analysis	67
3.11	OLSR Information Propagation	69
3.12	Causal Relations - Spoofing Behavior	70
3.13	Risk Estimation of Routing Message Contents	73

4.1	Processes in Light-weight Security Architecture	77
4.2	Components of Light-weight Security Architecture	78
5.1	CRADS Design	89
5.2	Filtering Process using FDA	93
5.3	Data Reduction	107
5.4	Detection Accuracy with Changing Mobility	108
5.5	ROC Curves with Changing Mobility	110
5.6	Detection Accuracy with Changing Traffic Density	111
5.7	ROC Curves with Changing Traffic Density	111
5.8	Detection Accuracy with Changing Packet Drop Ratio	112
5.9	ROC Curves with Changing Packet Drop Ratio	114
5.10	Spoofing Attack Detection: True Positives	115
5.11	Spoofing Attack Detection: True Negatives	116
6.1	CARRADS Design	119
6.2	Global Cooperative Decision Support	125
6.3	Hello Based Routing Attacks with 30 m/s Random Waypoint Node Mobility	134
6.4	TC Based Routing Attacks with 30 m/s Random Waypoint Node Mobility	135
6.5	Hello Based Routing Attacks with 40 m/s Vehicular Node Mobility	136
6.6	TC Based Routing Attacks with 40 m/s Vehicular Node Mobility	138
7.1	Network Setup	143
7.2	Equilibrium with Probability Scenarios	159
7.3	Equilibrium with combination of different probability and payoff matrix scenarios	160
7.4	Consensus for different probability scenarios	161
7.5	Consensus with combination of different probability and payoff matrix scenarios	162
7.6	Hello based attacks	165
7.7	TC based attacks	166

List of Tables

2.1	Cryptographic based Security Schemes	10
2.2	Intrusion Detection Architectures	22
2.3	Detection Methodologies	30
2.4	Trust models for Ad hoc Networks	36
5.1	Attack Simulation Setup	101
5.2	Sinking Attack Scenarios	103
7.1	Prisoners Dilemma	142
7.2	Trust Game's Payoff Matrix	146
7.3	Priority of goals in the trust game	147
7.4	Payoff Matrix Structures based on Belief	149
7.5	Evidence in belief and Neighbor node importance in the Payoff matrix . . .	151
7.6	Probability Scenarios	157
7.7	Type 1 Payoff Matrix Scenarios	158

List of Abbreviations

3DES	Triple Digital Encryption Standard
AES	Advanced Encryption Standard
ADM	Anomaly Detection Module
AI-SVM	Adaptive Incremental Support Vector Machine
AODV	Ad hoc on-demand Distance Vector
ARAN	Authenticated Routing in Ad hoc Networks
ASYM	Asymmetric
AUC	Area Under Curve
BIM	Behavior Identification Module
BNS	Best Neighbor Strategy
CARRADS	Cross Layer based Adaptive Real-time Routing Attack De- tection System
CIDS	Cross layer based Intrusion Detection System
CRADS	Cross layer based Routing Attack Detection System
DDoS	Distributed Denial of Service
DoS	Denial of Service
DSR	Dynamic Source Routing
DST	Dempster Shafer Theory
ECC	Elliptic Curve Cryptography
EFSM	Extended Finite State Machines
FDA	Fischer's Discriminant Analysis
FSM	Finite State Machines

HELGEN	Hello Generation
HELMOD	Hello Modification
HIDS	Host-based Intrusion Detection System
IDS	Intrusion Detection System
i-SVM	Incremental Support Vector Machine
LIDS	Local Intrusion Detection System
MA	Mobile Agent
MAC	Message Authentication Code
MANET	Mobile Ad hoc Networks
MCMA	Multiple Collection Multiple Analysis
MCSA	Multiple Collection Single Analysis
MDM	Misbehavior Detection Module
MFD	Maximum Flooding Depth
MIB	Management Information Base
MID	Multiple Interface Declaration
MPR	Multipoint Relay
MPRS	Multipoint Relay Set
NIDS	Network-based Intrusion Detection System
OIP	Originator IP
OLSR	Optimized Link State Routing
PCA	Principal Component Analysis
PKI	Public Key Infrastructure
RADS	Routing Attack Detection System
RBF	Radial Basis Function
RDP	Route Discovery Protocol
RERR	Route Error
RREP	Route Reply

RREQ	Route Request
RSA	Rivest, Shamir and Adleman
SAODV	Secure Ad hoc On-Demand Distance Vector
SDT	Sleep Deprivation Torture
SMO	Sequential Minimal Optimization
SNMP	Secure Network Management Protocol
SRP	Source Routing Protocol
SSD	Secure Stationary Database
SVM	Support Vector Machines
SYM	Symmetric
TC	Topology Control
TCGEN	TC Generation
TCMOD	TC Modification
TIARA	Techniques for Intrusion-resistant Ad hoc Routing Algorithms
TN	True Negatives
TP	True Positives
VMM	Vehicular Mobility Model

Abstract

The primary objective of the work is to design a security architecture which suits the inherent properties of ad hoc networks. Ad hoc networks are infrastructure-less, highly autonomous and distributed, and consist of nodes which are mostly limited in resources such as, battery power, processing and memory capacity. These inherent properties of ad hoc network have led us to design a novel security architecture that suits the ad hoc network characteristics. Typical security architecture comprises of three major components, namely, cryptography, intrusion detection and trust management. In legacy security architectures, cryptographic mechanisms are considered the prime component. However, due to the aforementioned characteristics of ad hoc networks and the complexity of encryption and key management algorithms, cryptography is hard to implement. Thus, in this work, we design a comprehensive security architecture for ad hoc networks with the absence of cryptographic mechanisms.

Though the inherent properties of ad hoc networks have caused the paradigm shift in security design for ad hoc networks, there is a unique property in ad hoc networks which needs critical security consideration. The routing methodology in ad hoc networks is '*cooperative routing*', where nodes depend on their neighbors for routing and data forwarding. This dependency of cooperative behavior from neighbors for routing has spawned an unprecedented vulnerability to attack at the routing layer level. Hence, securing the routing protocol is critical for ad hoc networks. Thus, the prime focus of the security architecture's design is to provide a light-weight routing security for ad hoc networks.

The architecture defines three processes, namely, detection, decision and response. Detection process monitors neighbor nodes' routing and data forwarding behavior and identifies malicious activity. Decision process computes the trustworthiness of

neighbors by corroborating its local detection outcome with other neighbors' recommendations. The decision process yields the trustworthiness metric for each neighbor. In the response process, the computed trustworthiness metric is integrated into the ad hoc routing protocol's route computation process. Thus, trustworthy nodes are preferred in routes computed by the ad hoc routing protocol.

The architecture primarily uses two security methodologies, namely, intrusion detection and trust management to implement the three processes. The detection process is implemented using two intrusion detection components, namely, Cross layer based Routing Attack Detection System (CRADS) and Cross layer based Adaptive Real-time Routing Attack Detection System (CARRADS). CRADS is an offline detection system, whereas CARRADS is a real-time detection component. The performance of the detection process depends on the detection accuracy and computational overhead of CRADS and CARRADS. To achieve superior detection accuracy, both CRADS and CARRADS utilize cross layer features to define routing behavior patterns. Furthermore, non-linear learning algorithms are used to maximize detection accuracy. To reduce the computational overhead, CRADS uses a set of novel overhead reduction measures to decrease the computational overhead incurred by the non-linear machine learning algorithms. In CARRADS, the computational complexity of the real-time adaptation process is minimized by using the characteristics of convex optimization techniques. The core of CARRADS consists of a novel incremental non-linear learning algorithm, which is both efficient and computationally feasible for ad hoc network nodes.

The decision process of the security architecture is implemented using a game-theoretic trust management module. The process is defined as a non-cooperative strategic n-player game, played between the node and its neighbors. The game process optimizes the node's decision towards the trustworthiness of each neighbor by corroborating local detection outcome from other neighbors. The game process is

played iteratively at real-time. Optimization of the trustworthiness measure of each neighbor is governed by a set of systematic selection criterion. The criterion ensures that the nodes reach at least the local optimum of the trustworthiness measure for the neighbor.

The response process of the security architecture is integrated into the route computation process of the ad hoc routing protocol. In addition to the route metric, the route computation process considers the trustworthiness metric while establishing routes. Route computation process prefer nodes with high trustworthiness measure in a route. This increases the security and reliability of routes established using the ad hoc routing protocol. Thus, the cross layer security architecture offers light-weight security for the ad hoc routing protocol.

Chapter 1

Introduction

An ad hoc network is a network that is formed among a set of end-user nodes without relying on infrastructure for connectivity [65]. All essential network services, such as routing and forwarding are handled by every node in the network. Nodes in ad hoc network rely on multi-hop packet relaying for communicating with nodes away from the local neighborhood. Due to the absence of infrastructure and centralized nodes, ad hoc networks use a unique routing methodology referred to as "*cooperative routing*" [18]. In this routing methodology, nodes depend on their neighbors for routing and data forwarding. The dependency of nodes on neighbors' cooperative behavior for network services has spawned unprecedented security vulnerabilities at the routing protocol. Securing the routing protocol is an unprecedented research problem, which this work addresses [8].

1.1 Background and Motivation

Ad hoc networks were originally designed for battlefield networks for forming ad hoc and infrastructure-less connectivity among military communication systems. It was effectively used for communication among soldiers and military units in the battlefield

CHAPTER 1. INTRODUCTION

for tactical operation [13, 108].

Due to the ease of deployment and cost-effectiveness, ad hoc network technology promises a new and broad spectrum of civilian applications [13]. The feasibility of ad hoc network in these applications depends on the efficient redesign of network support systems, like security, etc. Thus, the research of ad hoc network security focuses on building security architectures that suit ad hoc networks for commercial applications.

Security is one of the key deployment issues that ad hoc network technology paradigm is facing [8]. The unique characteristics of ad hoc network have rendered legacy security architectures that are used in conventional networks not feasible in ad hoc network. The challenges of ad hoc network security have attracted attention in all three research sub-domains of ad hoc security, namely, cryptography, intrusion detection and trust management [45, 119].

Battlefield (ad hoc) networks highly rely on strong cryptographic mechanisms such as, public key infrastructure (PKI), Elliptic Curve Cryptography (ECC), etc. for providing security to ad hoc network. These cryptographic algorithms usually have high computational complexity that affects the scalability of the network [27]. However, military communication systems have a higher bound in processing capacity and other resources, and hence, can handle such algorithms. On the other hand, computational complexity of cryptographic mechanisms is the primary reason for legacy security architectures and algorithms failing in a commercial ad hoc network environment [110].

Due to the closed system nature of military infrastructure, until recently, research in intrusion detection and trust management for ad hoc networks were given relatively low importance. Intrusion detection and trust management are typically used for detecting insider attackers. These systems were generally used as "*second wall of defense*" [66], for handling the failure of cryptographic mechanisms.

Research in ad hoc network security began by redesigning conventional cryp-

tographic schemes to secure ad hoc routing protocol. However, researchers like Zhang [117], Komathy [52, 53], Thamilarasu [95] and many others soon realized that cryptographic schemes are not practical to counter threats in ad hoc network. Cryptographic schemes proved to be computationally expensive for resource limited ad hoc network nodes. This has refocused the ad hoc security research towards intrusion detection and trust management. As the design of intrusion detection and trust mechanisms assume the absence of cryptographic mechanisms, it is well suited to provide security to ad hoc network.

1.2 Major Contribution

The primary objective of this work is to design a security architecture to secure the ad hoc routing protocol that suits the inherent properties of ad hoc networks. In this work, we design a security architecture for ad hoc networks with the absence of cryptographic mechanisms and using only intrusion detection and trust management. The prime focus of the proposed security solution design is to provide a light-weight secure routing solution for ad hoc networks. The design comprises of three security processes, namely, detection, decision and response. The detection process observes and analyzes the network traffic for suspicious routing behavior. The decision process evaluates and decides on the trustworthiness of neighbors by corroborating with other nodes in the neighborhood. The response process uses the outcome from the decision process to select secure and optimal routes.

The above three processes are implemented using three components.

- Cross layer based Routing Attack Detection System (CRADS)
- Cross layer based Adaptive Real-time Routing Attack Detection System (CAR-RADS)

- Game-theoretic Trust Management Module

The detection process is implemented using the first two components, CRADS and CARRADS.

CRADS is an offline detection system which uses an unprecedented approach of combining cross layer features for defining a routing pattern with nonlinear machine learning algorithms. This characteristic of CRADS offers superior detection accuracy. Computational complexity of CRADS is reduced using a set of novel overhead reduction measures which enable the feasibility of CRADS on mobile nodes.

CARRADS is a real-time detection system. The design comprises of a novel incremental nonlinear machine learning algorithm which have low computational complexity and good detection accuracy. The low computational complexity without compromising on detection accuracy is made possible by exploiting the inherent characteristics of nonlinear machine learning techniques.

Game-theoretic trust management module is used for the decision process. The decision process uses a unique approach of assessing the trustworthiness of neighbors by corroborating evidence collected from other neighbors. Unlike the conventional cooperative trust assessment, the corroborative trust assessment is not affected by corrupt or incorrect recommendations from neighbors. The corroborative trust assessment is modeled as a game played by the node with its neighbor nodes.

The response process is integrated in the route computation process of the ad hoc routing protocol.

1.3 Organization

The dissertation is organized as follows. Chapter 2 provides an overview of recent advances in research on intrusion detection systems and trust management for ad hoc networks. The chapter also briefly discusses some pioneering security architectures

CHAPTER 1. INTRODUCTION

for ad hoc networks proposed in literature.

Chapter 3 details the threat model for the proposed security architecture. Definition of the threat model of the security architecture comprises of identifying the characteristics of an ad hoc network that affect security, challenges for the architecture design and analysis of possible threats. Chapter 4 begins to discuss the proposed security architecture by introducing the three components of the security architecture. The functions of these components are also discussed.

The cross layer based intrusion detection system (CRADS) is explained in detail in Chapter 5. The efficiency of CRADS is analyzed for the detection accuracy and precision. Chapter 6 discusses the adaptive component of the security architecture, which is Cross layer based Adaptive Real-time Routing Attack Detection System (CARRADS). As CARRADS operates at real-time, the computational overhead is computed and analyzed.

The third component of the security architecture, which is the trust management mechanism is introduced, in chapter 7. The efficacy of the trust game for computing the trustworthiness is analyzed and detailed in this chapter. The chapter also discusses the effectiveness of the proposed security solution in securing the ad hoc routing protocol by selecting optimal and secure paths. Finally, the dissertation concludes in chapter 8 with a brief discussion on future research direction.

Chapter 2

Related Work

In Mobile Ad hoc Networks (MANET), the ad hoc connectivity between nodes in the network makes their network functions in a self-organized, decentralized and distributed [107] manner. Routing protocols in MANET are dependent on the willful cooperation of all the nodes in the network. This dependency has given rise to routing insecurity which is a new type of vulnerability not present in conventional networks [39].

Routing security is a branch of research in ad hoc network security. Research in this area aims to find a security solution for ad hoc routing protocols. Besides routing security, as shown in figure 2.1, ad hoc network needs protection from conventional threats that arise from higher layer protocols. Every branch of research in ad hoc network security addresses the challenges of securing protocol at each layer.

Security services can be classified into three major categories, namely, cryptography, intrusion detection and trust management. Researchers have been exploring all the above three category of security services to find an effective security solution for ad hoc routing protocols. In this chapter, we present an overview on recent advances in routing security research under these three categories. Before we discuss the three categories of ad hoc routing protocol security, the existing research on analyzing the

CHAPTER 2. RELATED WORK

Figure 2.1: Ad hoc Network Security Threats and Existing Routing Security Solutions

security threats in ad hoc routing protocol is presented.

2.1 Security Threat Analysis on Ad hoc Routing Protocol

Uniqueness of the security vulnerability in ad hoc networks has led researchers to analyze the possible security threats. Such an analysis is important, as it will help to design suitable security architecture for ad hoc networks. In security terminology, this analysis is referred to as the *definition of a threat model*. Interesting works can be found in the literature that extensively study the uniqueness and gravity of the security threats in ad hoc networks. We provide a similar security threat analysis in chapter 3, where we present a complete threat model for the ad hoc routing protocol.

Huang and Lee [43] systematically analyzed the AODV ad hoc routing protocol.

CHAPTER 2. RELATED WORK

They considered routing protocol internal states to analyze the attack possibilities. The ad hoc routing protocol is defined as an Extended Finite State Machine (EFSM) [50]. Various routing attacks such as, spoofing, fabrication, rushing and sinking are simulated and the state violations in the EFSM are observed. It was observed that sinking and rushing attacks creates state violation due to the deviant behavior of the attacker. However, spoofing and fabrication attack do not create state violations, thus, harder to detect. The authors gave a comprehensive study on routing attack possibilities. However, they did not analyze the more complex attack scenarios like black hole, gray hole, etc. It is anyway almost impossible to determine all possible attack scenarios.

Ning and Sun [69] also carried out an extensive analysis and provided a detailed case study on AODV routing protocol security issues. The authors studied possibilities of misusing each of the AODV's protocol communication messages which includes, Route Request (RREQ), Route Reply (RREP) and Route Error (RERR). The effect of fabricating and sinking the routing control packets are studied. The authors illustrated that fabrication attacks are harder to detect, than sinking attacks. Unlike the attack analysis by Huang and Lee, the authors studied complex attack scenarios, which is a sequence of basic attack events.

Yang et al [111, 112] proposed a formal mathematical model to represent vulnerabilities in ad hoc networks, similar to the attack analysis model by Huang and Lee. However, instead of EFSM, the authors modeled the protocol behavior using a protocol description language [99] that uses predicate logic to express malicious protocol behavior. An attack behavior will match one or more properties of the defined model. The authors analyze the possible violations of the protocol's standard behavior for understanding the security vulnerabilities in the MANET environment. As the model defines only the malicious behavior, new attacks cannot be studied. Nevertheless, the known vulnerabilities are modeled and the authors work presented

CHAPTER 2. RELATED WORK

a novel methodology to express attack signatures of routing attack scenarios.

To overcome the drawback in the work by Yang et al, Orset and Cavalli [71] used deontic [34] and temporal logic to model the security vulnerabilities of the table-driven routing protocol, Optimized Link State Routing (OLSR) [26]. However, unlike Yang's work, the authors model benign behavior properties of the routing protocol, instead of modeling threats. As the protocol's benign behavior is modeled, new attacks can be studied using the model. However, it should be noted that new benign behavior will be labeled as malicious behavior by this attack model.

Kong et al [54] discuss various passive attack scenarios against ad hoc networks. Their work details the effect of passive attacks in the perspective of military applications. The authors studied the detection possibilities of passive attacks, namely, location monitoring and eavesdropping. Even though eavesdropping can be thwarted using encryption algorithms at the application layer protocols, location monitoring is hard to prevent or detect.

Possibility of Byzantine attacks in ad hoc networks is analyzed and presented by Awerbauch et al [11]. Byzantine attack is a coordinate attack which is done by a group of nodes. According to Byzantine agreement solution, at least two-third of the neighboring nodes should be benign for the node to defend against an attack. This criterion should also hold in a cooperative security system. In chapter 3, the security threat model will illustrate that the byzantine conditions are hard to achieve in an ad hoc network.

2.2 Securing Routing Protocols using Cryptography

Cryptographic mechanisms are the most popular tool for security [89]. The performance of cryptographic systems in conventional networks led researchers to use

CHAPTER 2. RELATED WORK

Scheme	Mechanism	Features	Drawbacks
Ariadne	Iterated Hash Chain	DSR based secure routing protocol, which uses a MAC to secure the route discovery phase of the routing protocol.	Intermediate nodes can still disrupt the route discovery phase. Computational overhead is high due to iterated hash process.
SRP	Source and Destination nodes' Hash pair	Set of security extensions that can be applied to any on-demand routing protocol.	Route discovery phase is still not secure.
ARAN	Iterated Digital Certificates	A secure end-to-end authentication system.	Computation overhead is high. Need for centralized trusted authority.
SAODV	Digital Certificates and Hash	A source and destination node authentication scheme.	Intermediate nodes can still disrupt the route discovery phase. Need for centralized trusted authority.

Table 2.1: Cryptographic based Security Schemes

cryptography for securing ad hoc routing protocols. Research in routing security started with many cryptography based solutions for securing ad hoc routing protocols. Generally, in these solutions, the routing protocol communication between nodes is secured using cryptographic mechanisms [10]. The ad hoc routing protocols with added security feature for routing protocol communication are referred to as secure routing protocols.

Based on the cryptographic mechanisms used, the existing secure routing protocols in the literature can be broadly classified into two categories: hash-chain based secure routing protocols and digital certificates based secure routing protocols [107]. Hash-chain based secure routing protocols are more effective, as they do not need trusted nodes. The succeeding sections present some prominent research in secure routing protocols. Table 2.1 gives an overview of secure routing protocols.

2.2.1 Hash Chain based Secure Routing Protocol

Two well known work in this area are, Ariadne proposed by Yih-Chun et al [40] and Secure Routing Protocol (SRP) [72] proposed by Papadimitrakis et al. However, later research [27] proved that securing ad hoc routing protocols using cryptographic mechanisms is much more challenging than initially foreseen. Burmester [21] analyzed the fallacy of claims made by the authors of Ariadne and SRP in securing ad hoc routing protocols. This caused a paradigm shift in ad hoc network security research to focus on alternative security systems like intrusion detection and trust management [116].

2.2.1.1 Ariadne

Ariadne is an on-demand routing algorithm based on Dynamic Source Routing (DSR). There are several variants of Ariadne depending on the mechanism used to secure route request and reply messages. There are three mechanisms used by Ariadne for securing routing requests that includes, Message Authentication Code (MAC), digital signatures and 1-TESLA. Later study [3] showed that the MAC version is the most optimal variant of Ariadne. It uses iterated MAC computation to secure the route discovery process.

In Ariadne, each node shares a unique shared key with every other node in the network. Hence, each node holds $(N - 1)$ shared keys, where N is the number of nodes in the network. The source node S initiates the route discovery by the sending route request as $msg_{S,D,rreq} = (rreq, S, D, id, NULL, MAC_S)$. The next intermediate node X_1 receives this message and computes an iterated hash chain by $MAC_{S,X_1} = H[X_1, MAC_S]$. Thus, at the j^{th} intermediate node the iterated hash chain is $MAC_S, X_1 \dots X_j$ and the forwarded route request message is $msg_{S,D,rreq} = (rreq, S, D, id, NULL, X_1.X_j, MAC_{S,X_1.X_j})$.

The destination node D on receiving the route request from the last intermediate

CHAPTER 2. RELATED WORK

node will be able to compute and verify the MAC of every intermediate node that is contained in the iterated hash chain. Also, the destination node will be able to verify the sequence of the intermediate nodes. If the verification succeeds, such that all MACs are correctly computed by the intermediate nodes included in the route, then, the destination replies with the route reply: $msg_{S,D,rreq} = (rreq, S, D, id, NULL, MAC_T)$. The route reply is a unicasted directly to source node S along the reverse path.

Ariadne is susceptible to hidden channel attacks [21] where malicious nodes use an out-of-band channel to exchange the intermediate hash chains. Partially computed hash chains when exchanged to a different network neighborhood can disrupt the routing mechanism of DSR [21]. This kind of attack is also similar to the worm-hole attack which is discussed in section 3.3.3 of chapter 3. Burmester and De Medeiros presented the attack possibilities against Ariadne [21].

2.2.1.2 Secure Routing Protocol (SRP)

Papadimitrakis and Haas [72] pioneered the concept of secure routing protocols for ad hoc networks. They proposed Secure Routing Protocol (SRP), which is a set of security extensions which can be applied to any on-demand ad hoc routing protocol. Similar to Ariadne, SRP secures routing communication by using a MAC. Every node in the network shares a unique shared key with each node in the network. The routing request between source node S and destination node D is protected by using a MAC, which is computed using the shared key between S and D . Each neighbor that receives the request for the first time appends its identifier and rebroadcasts it. For instance, the route request that reaches the intermediate node X_J is in the form : $msg_{S,D,rreq} = (rreq, S, D, id, sn, X_1...X_J, MAC_S)$ where id is the identifier of the route request, sn is the session number and MAC_S is the hash on $(rreq, S, D, id, sn)$ computed by node S . The destination node D receives the route request and ver-

ifies the MAC_S using the shared key. If verification succeeds, destination node D replies with a route reply as: $msg_{S,D,rrp} = (rrp, S, D, id, sn, X_1...X_J, MAC_D)$ where MAC_D is the hash computed using D . Intermediate nodes check the contents of route reply for its identifier in the route and also check that it is adjacent with two of its neighbors on the route before sending the route reply upstream.

Though SRP pioneered secure routing protocols, similar to Ariadne, it failed to adequately provide security to ad hoc routing protocols. In SRP, the route reply is secure but the route request can still be under threat. Malicious intermediate nodes can simply pad along identifiers of other nodes with the route request to modify the route. Unlike Ariadne, SRP does not check the authenticity of intermediate nodes and also SRP does not verify the sequence of intermediate nodes in the route. Destination node D can never be sure that the nodes in the route list of the route request are authenticated.

2.2.2 Digital Certificate based Secure Routing Protocol

Digital certificates [9] are used to establish trust relationships between nodes to achieve authentication, non-repudiation and message integrity. Digital certificates are used in securing ad hoc routing protocols by using digital certificates to authenticate each node in a route. Generally, digital certificates are issued by a centralized trusted authority or created using the centralized trusted authority's public key.

This methodology of securing ad hoc routing protocols is not feasible for two reasons. First, the availability of centralized trusted authority in ad hoc network can never be guaranteed. Second, the computational complexity of computing digital certificates, which uses public key infrastructure, is too high for resource-limited ad hoc network nodes.

Two of the prominent research on secure routing protocols that uses digital signatures include, Authenticated Routing for Ad hoc Networks (ARAN) [82] and Secure

Ad hoc On-Demand Distance Vector (SAODV) [114].

2.2.2.1 ARAN

Authenticated Routing for Ad hoc Networks (ARAN) [82] is a secure on-demand routing protocol that is based on AODV. ARAN utilizes public key certificates signed by a trusted authority in order to achieve authentication and message integrity. ARAN assumes that every node in the network has the public key of the trusted authority.

The route discovery process of ARAN begins with the source node broadcast a route discovery packet (RDP). The contents of RDP are similar to RREQ in AODV, except that it also consists of certificates. RDP consists of type identifier, destination IP address, source node's digital certificate and a nonce. The contents of RDP are signed by encrypting it with node's private key. When an intermediate node receives the RDP message, it adds its identifier and certificate, then signs the contents with its own private key and broadcasts it. In addition, the intermediate node validates the certificate of the neighbor node from where the RDP was received by using the public key of that node.

At the destination, the RDP message will consists of iterated digital signatures signed by the intermediate nodes in the route. The destination node can verify the signatures of all intermediate nodes using their public keys and verify the authenticity of the message itself using the public key of the trusted authority. If the verification succeeds, the destination node replies (REP) with a unicast message on the reverse path towards the source. The contents of REP includes, type identifier, Source IP address, certificate of the destination node, and the original nonce from RDP. Similar to RDP, every intermediate node verifies the certificate of its predecessor in the route before appending its own certificate, signing it with its own private key and sending it upstream. The source node receiving the iteratively generated certificate chain can verify the authenticity of intermediate and destination nodes.

CHAPTER 2. RELATED WORK

Even though the feasibility of digital certificates in ad hoc network is questionable, ARAN provides guaranteed end-to-end authentication. Unlike SRP or Ariadne, ARAN do provide secure routing communication, but with a high computational and message overhead. The cost of ARAN is large routing packets, which cause a higher overall routing load, and higher delay in the route discovery process due to cryptographic computation process of digital certificates.

2.2.2.2 SAODV

Similar to SRP, Secure AODV (SAODV) [114] proposes a set of security extensions that secure the AODV routing protocol. SAODV uses both digital certificates and hash chains for securing AODV. Digital certificates are used to protect non-mutable fields of the AODV header, whereas, the hash chains are used to protect the hop count information. The protocol uses public key certificates and relies on key management mechanism to distribute keys.

The source node initiates the route discovery process using route request message (RREQ). The digital signature for the RREQ message is computed by signing the contents of the RREQ message except the hop count field using the private key of the source node. The hash is generated for the hop count field in the RREQ message. The computed digital certificate and hash is appended to the RREQ message. Every intermediate node receiving the RREQ computes a hash on the hop count in the SAODV header and appends it with RREP message before forwarding it. If the intermediate nodes can reply to a route request on behalf of the final destination, then they append their signature signed with their own private key and reply to the route request. Otherwise, intermediate nodes forward the route request to their neighbor nodes. The source node can verify the authenticity of the route by validating the certificates in the route reply.

2.3 Intrusion Detection Systems

The realization of the importance of intrusion detection system (IDS) in ad hoc network security has recently spawned the new branch of research, i.e., intrusion detection in ad hoc networks [75].

Intrusion detection is a process of detecting behavior with malicious intent. A malicious behavior has different interpretations in different layers or protocols. But fundamentally, a malicious behavior aims to add, modify or delete services or data. In MANET routing protocols, a malicious behavior includes, malicious route addition, modification, deletion and Denial of Service (DoS) [58].

Research in intrusion detection systems can be broadly categorized into IDS architectures and detection methodologies [63]. Research in IDS architectures focus on the design aspect of IDS, which define the overall functioning of the system. The simple IDS design shown in figure 2.2 is an example of an IDS architecture. On the other hand, detection methodologies are the core detection algorithms of the IDS, which is used to classify benign and malicious behavior. In the IDS shown in figure 2.2, the misbehavior and anomaly detection algorithms are the detection methodologies of the IDS. Efficiency of the IDS depends on both the architecture and detection methodology. Mishra et al. [63] provided a detail analysis on the existing architectures for IDS in ad hoc networks.

Most of the IDS architectures proposed in the literature assume that the detection methodologies used in a conventional IDS model will function efficiently in an ad hoc environment [117]. It has been observed that, research work which focuses on IDS architectures does not consider the limitations of detection methodologies in ad hoc networks and vice versa.

Figure 2.2: A Simple Intrusion Detection System

2.3.1 Fundamental Design of IDS

Typically, as shown in figure 2.2 an IDS consists of two types of modules, namely data collection and data analysis [30]. Data collection is the process of monitoring and collecting information about the nodes' behavior. A behavior is expressed using a set of features, called the feature set. Features are parameters like, frequency, contents and properties of different events in a behavior. For example, contents of a routing message like, number of declared neighbors, sequence number, etc. are examples of features of a routing event. The data analysis module uses the collected behavior information to detect malicious activity.

2.3.1.1 Data Collection

The two approaches of data collection in conventional IDS include,

- Host based IDS (HIDS)
- Network based IDS (NIDS)

HIDS runs on individual hosts and collects behavioral data by monitoring the operating system's local audit records. NIDS on the other hand, monitors and collects behavioral data from the traffic flowing in the network.

CHAPTER 2. RELATED WORK

2.3.1.2 Data Analysis

Once the data is collected by the IDS's data collection module, it has to be analyzed so that malicious activity can be detected. This is done by the data analysis module, which is the core of the IDS. The data analysis module can use two kinds of detection algorithm, namely, rule-based and statistical detection algorithms.

Rule-based detection [78, 106] use static rules to determine malicious behavior. Rules are set of logical conditions, when these conditions are met; the behavior is classified as malicious. Let us consider a simple rule to illustrate. Failure of three or more consecutive login attempts can reasonably be used to decide that the behavior is malicious. More complex rules are formed using typical logical reasoning mechanisms such as, expert systems. Static rule-based approaches, which are practical in conventional networks, fail in ad hoc networks due to its dynamic nature. In ad hoc networks, the topology is always changing. This creates transient connectivity among nodes and changing network conditions for every node in the network. In such a dynamic environment, the intrusion detection through static rules is almost impossible. Furthermore, static security systems are known to perform inefficiently in dynamic and distributed systems.

Statistical approaches [22, 74] uses probability estimation theory to allow some flexibility to crisp logic and rule based detection strategies. In the statistical approach, probability of behavior being malicious is determined by statistically analyzing the known behavior patterns. Statistical detection algorithms use pattern classification techniques which have to be trained with known historical behavioral patterns. Training is a process of teaching the IDS to detect malicious behavior by using known behavioral patterns with the least possible error.

The statistical detection module trains a classifier [31] based on known behavior patterns. The classifier will then be used to classify unknown patterns by measuring quantitatively the unknown behavior's deviation from known benign behaviors or

CHAPTER 2. RELATED WORK

Actual Behavior	Detected Behavior	
	Benign	Malicious
Benign	True Negatives	False Positives
Malicious	False Negatives	True Positives

Figure 2.3: Detection Accuracy Metrics

similarity of the unknown behavior to known attack patterns. The detection methodology of measuring the deviation of unknown behavior from known benign behavior is referred to as anomaly detection. On the other hand, the detection methodology of measuring the similarity between the unknown behavior and known attack patterns is referred to as misbehavior detection.

The two primary metrics for analyzing the efficiency of IDS is true positives and true negatives. As shown in figure 2.3, True positive (TP) is the percentage of correct detection of malicious behavior, where as, true negatives (TN) is the percentage of correct detection of benign behavior. Similarly, false positive is the percentage of incorrect classification of benign behavior (which is $100\% - TP$) and false negative is the percentage of incorrect classification of malicious behavior (which is $100\% - TN$). In network security terminology, false positives are also referred to as false alarm.

Anomaly Detection

The anomaly detection technique [92] involves looking for behavior that has deviated from the expected benign behavior. This is usually done by utilizing statistical pattern classification techniques. The pattern classification techniques statistically compare the observed behavior with the expected behavior to identify maliciousness.

The prime advantage of anomaly detection is that, it does not needs prior knowledge of attack patterns. Anomaly detection has high probability of detecting most attacks. However, due to the statistical nature and the limitation on defining all possible benign behavior patterns, anomaly detection is poor on detecting new benign

CHAPTER 2. RELATED WORK

behavior. That is, new benign behavior maybe detected as a malicious behavior.

Misbehavior Detection/Signature based Detection

Misbehavior detection [83] is complementary to anomaly detection. The misbehavior detection involves analyzing the observed behavior with specific attack patterns. Pattern classification techniques are utilized to identify the resemblance of the observed behavior to known attack patterns.

Contrary to anomaly detection, misbehavior detection has low false positives. This is evident, as only known attacks will be detected as malicious behavior and every other behavior will be classified as benign. However, new attacks and variants of known attacks will be missed detection by misbehavior detection. Therefore, misbehavior detection is poor on detecting new attacks.

Statistical Detection Algorithms

Statistical detection algorithms utilize pattern classification or recognition techniques [31]. To understand the operation of statistical detection algorithms, it is wise to understand the fundamentals of computational intelligence. Computational intelligence (CI) is the research field which deals with pattern classification and recognition problems. Generally, any CI based pattern classification or recognition technique comprises of two processes, namely, training and prediction.

Training is a process in which known patterns are used to tune a set of parameters called classifier. Literally, the above process teaches the classifier, using the known classified patterns on how to classify or recognize various classes of behavior patterns. In most of the CI methods, the objective of training is to define a hyperplane/boundary between the classes, as shown in figure 2.4. The definition of boundary can be as simple as a linear equation or as complex as a set of non-linear points governing the boundary. Statistical classification techniques are further classified based on the definition of the boundary conditions. Linear and non-linear CI methods define boundaries using linear and non-linear conditions, respectively.

CHAPTER 2. RELATED WORK

Figure 2.4: Linear Machine

Essentially, training algorithms are machine learning algorithms. Machine learning is a process in which a set of threshold parameters are trained to classify an unknown behavior. To introduce the concept, let us consider a simple classification method, referred to as "*linear machine*" [31]. The method determines a linear function $g(x)$ which forms a boundary / hyper-plane between the two classes of behavior.

$$g(x) = w^T \cdot x + w_0 \Rightarrow \begin{cases} > 0 \\ \leq 0 \end{cases} \quad x \in \begin{cases} \Rightarrow Benign \\ \Rightarrow Malicious \end{cases} \quad (2.1)$$

The coefficient and constant of function $g(x)$, namely, w^T and w_0 represent the orientation and distance from origin of the hyper-plane, respectively. These are, in fact, the threshold parameters, and are trained using the known behavioral patterns.

The vector x represents an event in feature space and the hyper-plane separates the

CHAPTER 2. RELATED WORK

Scheme	Mechanism	Features	Drawbacks
Patrick et al IDS	Mobile agents using SNMP	Mobile agents use local SNMP data located in the management Mobile-agent-based information base as audit sources for intrusion detection.	SNMP used in this scheme is not secure in ad hoc networks.
SSD IDS	Centralized Knowledge base with mobile agents	Centralized and stationary database used to store misuse signatures and user profiles. Mobile agents use this database to detect attacks locally.	Need for centralized trusted node to host the stationary secure database.
Modular IDS	Modularized mobile agents	Every module in the intrusion detection system is implemented using mobile agents.	The mode of operation of mobile agents is not secure in ad hoc networks.
Distributed IDS	Cooperative detection	Statistical intrusion detection to detect local and global intrusions.	The cooperative detection process does not consider spoofing of cooperative responses.
Cross Layer based IDS	Multiple layer integration	Information at different protocols at various layers is used to detect intrusions.	Computational overhead is increased due to the increase in the number of features defining routing pattern

Table 2.2: Intrusion Detection Architectures

events in the feature space. The classification of a given unknown behavior depends on the sign of $g(x)$.

Prediction is the process of classifying an unknown pattern using the classifier derived using the training process.

2.3.2 IDS Architectures

The succeeding section discusses on existing IDS architectures for ad hoc networks. Table 2.2 gives a brief summary of the discussed IDS architectures.

2.3.2.1 IDS Using Mobile Agents

Mobile agent is a software module, which aids in distributed host-based intrusion detection. The software module traverses through the nodes in the network to accomplish a particular task, such as, collecting information, processing information,

CHAPTER 2. RELATED WORK

Figure 2.5: LIDS Architecture

etc. Mobile agents try to emulate network-based intrusion detection by using a collective host-based IDS. The mobile agent provides a good framework to create a distributed host-based IDS. However, mobile agents itself pose a security threat to the ad-hoc network. This is detailed in the following sections by analyzing the IDS architectures that uses mobile agents.

Local Intrusion Detection System using Mobile Agents

Patrick et al. [73] proposed an application-based IDS architecture using mobile agents for providing intrusion detection in an ad hoc network. The IDS architecture is shown in figure 2.5, which consists of agents. Agents are host-based intrusion detection modules running on all nodes. The architecture utilizes SNMP (Simple Network Management Protocol) to communicate with the neighbors.

A Local IDS (LIDS) agent is responsible for detecting the attacks locally. LIDS agents also help neighbor nodes to decide on a suspected intrusion and receive updates of new attack patterns from the neighbor nodes. The attack patterns are stored in the information base. The MIB (Management Information Base) agent is used

CHAPTER 2. RELATED WORK

to manage the information base. Between the neighbor nodes, SNMP is used to exchange information such as, new attack patterns, decisions/responses, etc. The MIB agent is responsible for retrieving and sending information to/from neighbors using SNMP. The authors exploit the co-operative nature of ad hoc network by sharing the information about new attack patterns between the nodes.

Additionally, mobile agents are software modules which function autonomously for a dedicated task. For example, the LIDS may designate a mobile agent (MA) to determine the probability of a particular behavioral pattern to be malicious. The MA will autonomously travel between nodes and gather evidence from traversing nodes' MIB.

This approach has made some fundamental assumptions. First, the authors assume that SNMP is secure in an ad hoc environment. In a network, where routing is insecure, SNMP is not as secure as in a conventional network. Secondly, without a reliable and trusted communication medium, knowledge sharing is highly insecure in an ad hoc network. This leads to the insecurity of the LIDS system itself. Compromised nodes can announce misleading intrusion detection information, which will eventually corrupt the information base of the entire network. Finally, in a network with constantly changing connectivity, feasibility of mobile agents is questionable.

Stationary Secure Database IDS

Andrew and May [7] proposed an IDS architecture which consists of a stationary secure database (SSD). Nodes post new information and decisions into this database. The architecture is simple, as shown in figure 2.6. Only detection processing is done on the host, the attack information is stored in a secure stationary centralized point.

The other components of the IDS are; misbehavior detection module (MDM), Anomaly Detection Module (ADM) and secure communication channel. MDM and ADM are present in every node and these components form the mobile agent. Additionally, a local intrusion database is also used to store node specific attack patterns

Figure 2.6: Secure Stationary Database Architecture

and temporary information. The mobile agents will publish the newly found attack pattern to the SSD, only after a certain level of confidence is reached. The secure communication channel is used to communicate with the other nodes' host-based intrusion detection system.

It can be seen that, SSD conflicts with the ad hoc characteristic of the absence of centralized authority. Even if a node is voted as the centralized node using trust mechanisms, there is no guarantee that the node will behave benignly. Furthermore, a malicious node can corrupt the SSD by sending incorrect intrusion detection information. SSD creates a single point of failure for the IDS. Additionally, SSD assumes cryptographic mechanisms on the communication between the IDS and SSD. This violates the fundamental principle of an IDS, which assumes "no existence of cryptographic mechanisms".

Modular Intrusion Detection Architecture

Kachirski and Guha [48] proposed an IDS, where the IDS is modularized into various sub-modules as shown in figure 2.7. The sub-modules are network monitoring, host monitoring, decision making and response (action) modules. The modules

CHAPTER 2. RELATED WORK

Figure 2.7: Modularized IDS Architecture

are implemented in mobile agent framework and each of the modules in IDS architecture is implemented using a mobile agent. Network monitoring is the process of promiscuously listening to the network traffic in network neighborhood for monitoring neighbor nodes' behavior. Host monitoring is the process of monitoring the communication between the node and neighbors. A host-based monitoring agent module exists in every node; however, network-based monitoring agent exists only in a selected few. Decision making and response modules exist in every node.

The entire ad hoc network is segregated into clusters. Each cluster has a cluster-head, which runs the network-based monitoring agent. Therefore, packet-level monitoring is done by the cluster-head. Individual nodes use the packet-level audits from the cluster-head to improve the performance of the HIDS.

The strength of this IDS architecture is augmentation of NIDS with HIDS. The combination of these mechanisms has been proved very efficient in conventional IDSs. Furthermore, the authors have eliminated the single point of failure by distributing the task to the cluster heads. This also distributes the management load between cluster-heads of the network. In addition, host-based decision on an intrusion makes this approach robust against attacks on the IDS itself.

CHAPTER 2. RELATED WORK

Figure 2.8: Distributed IDS Architecture

However, the architecture's trust on the cluster-head is its weak point. Malicious behavior of a cluster-head will lead to the compromise of all nodes under its control. In addition, similar to the other two mobile agent based IDS, this architecture assumes secure routing, which may not be true.

2.3.2.2 Distributed IDS

Zhang, You and Lee [117,118] pioneered the first distributed and cooperative intrusion detection and response model for commercial ad hoc networks. Their model is still considered as the *de facto* standard for IDSs for ad hoc networks. The authors considered both conventional attacks and routing attack scenarios.

The components of the IDS architecture are structured as shown in figure 2.8. The core components of the architecture are the local detection engine and cooperative detection engine. In this IDS architecture, every node monitors the local network neighborhood and detects malicious behavior using the local detection engine. The cooperative detection engine of the IDS architecture is triggered only when an anomaly is detected by the LIDS module. The cooperative detection engine is used to decide on the global response to the local detection.

CHAPTER 2. RELATED WORK

The local detection engine is a typical HIDS which operates autonomously using local data collection. The local detection engine monitors the radio channel promiscuously and collects neighbors' network behavior. When an anomaly is detected or if the evidence is inconclusive, the neighboring nodes' cooperative detection engine collaboratively decides on a global response for the detection. After the global response is decided, the local response and global response module coordinate with each other to implement the respective action. The local response triggers actions that are local to the node, whereas the global response triggers actions that coordinate with the neighboring nodes.

In determining the response, the cooperative detection engine assumes that the number of benign nodes is higher than the number of malicious nodes. However, routing insecurity of ad hoc network provides the ability to an attacker to create non-existent nodes. Therefore, the attacker can emulate malicious behavior for these non-existent nodes. Thus, the real majority of benign nodes will not help to guarantee security of the distributed IDS.

Similar to Zhang et al.'s work, the authors of [115] proposed a distributed Cluster based Court-like IDS (CCIDS) architecture. The authors proposed a court-like defense system, which includes four components; Monitor, Accuser, Arbiter, and Defendant. The network is divided into clusters with each cluster controlled by a cluster-head. The monitoring module is in every node in the ad hoc network and monitors routing messages. Any non cluster head node can detect a suspicious routing behavior and accuse the suspected node using the suspected node's address. This accusation is done by the accusation module. The cluster-head is responsible for arbitration and the arbitration module is present in the cluster-head. The defendant module is present in every non cluster head node and is responsible for defending an accusation raised by a neighbor node over the node behavior. The integrity of CCIDS lies on the security of the cluster head node. A compromised cluster head node can

CHAPTER 2. RELATED WORK

cause disastrous effects on the network.

Cannady J [23] proposed a distributed IDS architecture using Self Organizing Maps (SOM). The approach uses Learning Vector Quantization (LVQ) algorithm to detect malicious routing behavior in a distributed manner. The LVQ is a combination of a self-organizing map (SOM) for classification and a competitive multilayer neural network which uses the output of the SOM as input for pattern recognition. The multilayer neural network combines SOMs from neighbor nodes to form a distributed analysis mechanism. The integrity of LVQ process depends on the secure communication of the SOMs between nodes. The LVQ process can be compromised by malicious nodes disseminating corrupt SOMs.

2.3.2.3 Cross layer based IDS

The concept of using cross layer features was pioneered by Thamilarasu et al. [95] for increasing the detection accuracy of the IDS. The authors emphasized the inadequacy of statistical information in routing protocol communications and proposed a novel methodology for increasing the information content in routing pattern definitions by using cross layer features. In the authors' proposed cross layer IDS (CIDS) design, every layer consists of an intrusion detection component. Multiple levels of detection is performed across different layers of the protocol stack before confirming the malicious behavior of the nodes. The decision outcome from the intrusion detection components at different layers are combined to decide over upon intrusion. The major drawback with this approach is the high computational overhead incurred by the multiple intrusion components. Multiple intrusion detection components increase the computational complexity of the IDS and hence make it not feasible in ad hoc networks.

To address the issues of cross layer design, Liu et al [60] proposed a solution that reduced the cross layer features to form a reduced feature set based on the information

CHAPTER 2. RELATED WORK

Scheme	Mechanism	Features	Drawbacks
TIARA	Collection of security mechanisms	Presents routing-algorithm-independent general design principles and techniques that can be incorporated in MANETs for robust fault-tolerant networks.	The network is tolerant to attacks. Prevention and detection is not the objective of the scheme.
Threshold based Detection	Threshold	A threshold-based scheme where nodes watch neighbors for signs of malicious activity. Once a threshold is crossed the malicious nodes are excluded from the network.	The method is simple and attackers can easily thwart detection.
State based Anomaly Detection	Protocol definition using Finite State machines	The routing protocol states are defined using a finite state machine model and violation of the state model is considered as intrusions.	Some attacks are not captured in state violations in FSM. These attacks will miss detection.
Non-linear machine learning	Non-linear statistical learning algorithm	Sophisticated non-linear machine learning algorithms are used to classify benign and malicious behavior.	Computationally expensive for ad hoc network nodes.
Real-time adaptive IDS	Incremental real-time adaptation of detection model	The intrusion detection model is adapted at real-time to changing network conditions	Computationally expensive and sensing change in network conditions is addressed. Adaptation is not dynamic.

Table 2.3: Detection Methodologies

content in different features. This reduces the computational overhead incurred by the IDS as the number of features is decreased. The authors used the traditional approach of feature reduction, namely, feature ranking and feature selection. Though these approaches reduce computational overhead, they also decrease the detection accuracy of the IDS due to the removal of features.

2.3.3 Detection Methodologies

Various detection methodologies for intrusion detection in ad hoc networks are presented in the succeeding sections. Table 2.3 gives a summary of the discussed detection methodologies.

2.3.3.1 TIARA

Techniques for Intrusion-Resistant Ad-hoc Routing Algorithms (TIARA), is essentially an intrusion prevention model [81]. TIARA is a distributed framework with a conglomeration of innovative techniques which provides,

- Light-weight firewalls
- Traffic Policing
- Intrusion Tolerant routing
- Intrusion detection
- Flow monitoring
- Reconfiguration mechanisms
- Multi-path routing
- Source initiated route switching

It aims to minimize the damage incurred on the ad-hoc network by destructive attacks such as, Denial of Service (DoS), Distributed Denial of Service (DDoS), etc. Routing and data traffic are protected by TIARA.

2.3.3.2 Threshold-Based Detection

A simplistic approach for intrusion detection in ad hoc networks is threshold-based detection. Bhargava et al [15] proposed a detection methodology which prevents internal attacks (attacks within the network). Internal attacks are exhibited by nodes belonging to the network which behave maliciously, either by themselves or when compromised. Each node maintains a local variable called "MalCount" for every other node, which is increased for a particular node if its behavior is suspicious. Thus the "MalCount" array in a node tracks the level or state of suspicion that the host node has regarding the other nodes. Each node share its local state of suspicion with respect to a particular node with other nodes in the network using a special packet REMAL. When a node receives REMAL, it increases its local MalCount for the particular node under suspicion.

The authors overlooked many aspects of security system. First, malicious knowledge sharing using REMAL will have cumulative effect on the network that will affect integrity of detection process. Secondly, the security vulnerabilities of the REMAL packet were not studied. As REMAL messages are broadcasted, malicious spoofing and fabrication of these packets are possible. Methods to prevent such threats are not presented by the authors. The crucial aspect of the security of the IDS is not considered in this methodology. Furthermore, routing security is not addressed.

Another interesting approach referred to as Watchdog-Pathrater [84], which also uses threshold, is proposed by Sergio et al. WatchDog-Pathrater, as the name implies, has a monitor and an evaluator. Unlike Bhargava et al.'s approach, watchdog-Pathrater functions independently and does not share information with other nodes. When a packet is forwarded to a neighbor node, the forwarding node listens and monitors how the node behaves upon receiving a packet. A benign node will forward faithfully, which is overheard by the monitor. However, when the node does not forward the packet, the pathrater increases the failure rate for the path. The monitor

does not distinguish between maliciousness and node faultiness. Once the failure rate reaching the threshold, the node is discarded from any path.

This method is analogous to fault-tolerance in typical routing algorithms. The proposed method effectively detects and responds to malicious packet dropping attack (sinks). However, it fails to address attacks such as route invasion and route disruption.

2.3.3.3 State-Based Anomaly Detection

One of the interesting approaches in conventional IDS models are state-based intrusion detection. Michael and Ghosh [62] in their pioneering work incorporated state-based model in intrusion detection for ad hoc networks. They proposed two anomaly detection methodologies, which uses finite-state machines (FSM). FSM has proved successful in conventional IDS, because of their adaptability and dynamic learning capability of new attacks.

Anomaly detection methods proposed by Michael and Ghosh [62] used protocol states. In the first method, the sequence and frequency of protocol states are monitored. Intrusion is confirmed when a particular sequence deviate significantly from benign behavior pattern or the frequency of states exceeds a threshold. To increase robustness, their second approach used probabilistic state-based intrusion detection. Each occurrence of a suspicious protocol state increases the probability of the behavior being malicious.

These two approaches are well suited for transport and application layer protocols, which have many protocol states, and the protocol states are predictable. For example, attacks such as, TCP SYNC flood attack can be detected using this approach.

However, this is not true in the case of routing protocols. State sequence or frequency of states cannot be used to distinguish a malicious behavior from a benign one. Traditionally, FSM was used to extract semantics of user behavior through

application-layer protocols. In the case of ad-hoc routing protocols, semantics is not represented by protocol states, but factors such as, current topology, mobility, connectivity, etc.

2.3.3.4 Non-linear Machine Learning based Detection Methodologies

Deng et al [29] introduced the use of non-linear machine learning algorithm for intrusion detection in ad hoc networks. The authors used a Support Vector Machine (SVM), a highly sophisticated non-linear machine learning algorithm for training the intrusion detection model. Similar to Zhang and Lee's work, the IDS consists of a global IDS which aids the local IDS that are present locally in the nodes. However, instead of a single global IDS, the architecture uses cluster heads which form a hierarchical IDS. This structure increases the reliability of the global IDS and knowledge sharing. The significance of the work is the architectural design of the distributed IDS to collate information from different nodes. Though this work introduced SVM for the first time in the IDS design for ad hoc networks, the authors gave much importance to architectural aspects of IDS design. However, a major drawback of using SVM is the computational resource requirement.

2.3.3.5 Real-time IDS Adaptation

In IDS for ad hoc networks, adaptation is a process of adding new routing behavior patterns to the detection engine or removing old obsolete patterns from the detection engine. For conventional IDS, adaptation is an auxiliary feature and is mostly handled offline, manually or semi-automatically [87,90]. However, this is not the case for IDS in ad hoc networks. Here, adaptation is not just auxiliary but rather an important component. Primarily, this is due to the transient nature of the routing behavior induced by mobility and other network environmental factors.

Kurosawa et al. [56] pioneered an adaptive IDS for ad hoc networks which uses

CHAPTER 2. RELATED WORK

Principal Component Analysis (PCA) [47] for training. A dynamic threshold is used to sense an environment change and trigger a retraining process. Attacks are simulated over Ad hoc On-Demand Distance Vector (AODV) routing protocol and their IDS is evaluated. Though the authors claim that the system is adaptive, they still use static training datasets. Furthermore, their proposed IDS is based on anomaly detection, which trains only on benign patterns. Any deviation from the benign behavior is detected as malicious. This evidently increases the number of misclassified benign behaviors (false positives). Nevertheless, the real-time adaptation of the detection model will moderate the number of false positives.

Sun et al. [91] proposed an adaptive intrusion detection system, which uses a Markov chain based anomaly detection. Similar to Kurosawa et al's approach, the authors have used static training dataset for real-time adaptation. The IDS selects a subset of training patterns from the original dataset which is optimal for the current network conditions and the detection model is retrained. In both approaches discussed above, the very essence of retraining and adaptation is lost. Effectually, these works are still static training systems with multiple detection models.

2.4 Trust Management

Trust establishment is a crucial security component, especially in highly autonomous systems, such as mobile ad hoc networks [105]. The objective of a trust system is to measure the reliability of nodes, so that critical operations, such as, routing, security information exchange, etc. can be done using the most trusted nodes in the network neighborhood.

Traditionally, trust in conventional networks is established by two methodologies, namely, recommendation-based and cryptography-based mechanisms. In recommendation based trust mechanism, the nodes collect recommendations from neighboring

CHAPTER 2. RELATED WORK

Scheme	Mechanism	Features	Drawbacks
Game theoretic trust evaluation	Probabilistic voting game	The process combines the recommendations from the neighbors to compute collaboratively the trustworthiness of neighboring nodes.	Belief of majority always influence the network. A small group of nodes finding a malicious node cannot influence the network's trust computation.
Trust metric for ad hoc networks	Semi-ring algebraic structure	An algebraic structure referred to as 'semiring' is defined that suit the trust computation process in ad hoc networks. The semiring is a metric system which is used to compute trustworthy routes.	
Game based Neighbor Cooperation Enforcement	Best Neighbor Game Evolution Strategy to enforce cooperation	To enforce cooperation among neighbors, each node plays a strategic iterative packet forwarding game with its neighbors. The nodes use an evolutionary game strategy which force selfish nodes to cooperate in the network operation	This method detects and prevents selfish behavior. Other malicious behavior of neighbors are not considered.
E-Hermes	Cooperative Trust establishment using neighbor recommendations.	E-Hermes combines recommendations from neighbors to evaluate the trustworthiness of neighboring nodes.	Sharing of sensitive trust information is a threat to the security system. Also, the trust evaluation relies on one parameter, packet forwarding rate. This affects the robustness of the trust computation process.
Dempster Shafer Approach	Dempster Shafer theory of evidence	Dempster Shafers theory of evidence is used to combine recommendation from neighbors to compute trustworthiness	Spoofing of recommendations from neighbors is not considered

Table 2.4: Trust models for Ad hoc Networks

CHAPTER 2. RELATED WORK

nodes for a particular node under evaluation. These recommendations are quantitatively combined with local observation to determine the trust for the node. Here the notion of "trust" can be defined as, *"the willingness of a node A to be vulnerable to the actions of node B based on the expectation that node B will perform a particular action important to node A, irrespective of the ability to monitor or control node B"*. The problem with this approach is that one or more malicious or misinformed node(s) can skew the trust evaluation with its misleading recommendation. Therefore, the process of combining neighboring recommendations should ensure the integrity of the trust model.

The cryptography-based trust methodology depends highly on public key infrastructure (PKI) or other asymmetric cryptographic mechanisms [89]. By using PKI, each node is pre-configured with node specific secret information. The trust between nodes is built by each node verifying the other node's PKI credentials. Evidently, the definition of trust is different from the one in recommendation-based methodology. Trust in this methodology is defined as *"node A's belief of node B's authenticity and its ability to function as deemed by node A"*. Cryptography-based trust methodology is not feasible in ad hoc network for two reasons. First, this methodology requires a reliable centralized system for security services such as key management, signature directory services, etc. However, ad hoc networks being infrastructure-less, decentralized and distributed, lacks reliable centralized nodes. Second, the computational complexity of PKI systems is too high for resource constrained ad hoc network nodes. Therefore, the trust models in the literature predominantly opt for recommendation based trust methodology.

Existing research on trust models for ad hoc networks is presented in the succeeding section. Table 2.4 gives a brief overview of the discussed trust models for ad hoc networks.

2.4.1 Theodorakopoulos and Baras's Trust Models

Theodorakopoulos and Baras [96,97] pioneered the concept of using game theory for trust models in ad hoc networks. The authors used an elementary voting methodology, where the neighboring nodes vote on the trustworthiness of nodes. Each node uses a fixed trust threshold (η) which is used to decide on whether to trust or not trust the node. The node is trusted if $V_i \geq \eta \cdot K_i$, where V_i is the effective number of votes and K_i is the number of nodes voting in the network neighborhood. If the i^{th} node has received V_i^+ positive votes and V_i^- negative votes, the effective number of votes V_i is computed by $V_i = V_i^+ - V_i^-$. There are a few drawbacks with this approach. As this is a voting system, the opinion of the majority of nodes will always prevail. Therefore, if a small number of nodes believe that a node is untrustworthy, which could be true, they will not be able to express their opinion over the network. Furthermore, the voting system fails to consider the nodes' confidence in their votes. This compromises the robustness of the trust model. Nevertheless, this work has spawned the idea of using game theory for designing trust models for ad hoc networks.

In their second pioneering work, Theodorakopoulos and Baras [98] proposed a trust metric system for ad hoc networks that uses a "semi-ring structure" for evaluating trustworthiness of nodes in ad hoc networks. Semi-ring is a type of algebraic structure which generalizes the properties and operations on a set of integers. The trust evaluation is formulated as a type of path problem in the network. Their work uses a distance semi-ring, which finds the optimal trust path in the network based on trust metrics. Similar to finding a shortest route in a network, the distance semi-ring computes the optimal trust path. The binary operators in the distance semi-ring are modified to suit the trust metric computation. This work strongly motivated our interest in trust models for ad hoc networks.

2.4.2 Game theoretic Cooperation Enforcement

Komathy et al [52,53] used game theory for enforcing cooperative behavior in ad hoc networks. A node is cooperative, if it forwards packets faithfully as intended by the ad hoc routing protocol. Hence, to enforce cooperation among neighbors, the nodes play a packet forwarding game with their corresponding neighbors. The number of packets forwarded by the node is used as the primary metric to measure the node payoff in the game. An evolution game strategy is applied to develop the Best Neighbor Strategy (BNS). BNS punishes selfish nodes by restricting service. Hence, BNS forces selfish neighbors in the neighborhood to cooperate. Under BNS, selfish nodes suffer severe loss of service due to their selfishness. When a neighbor node requests service, the node computes its payoff based on the neighbor node's behavior in the network neighborhood. A benign neighbor node will receive optimal payoff and the node will attend to the service request. On the other hand, if the neighbor node was selfish, the node restricts its service for the neighbor node. This process of penalizing neighbor nodes for selfish behavior forces selfish nodes to operate cooperatively in the network. Note that in this case, the number of packet forwarded/dropped was monitored in real-time and the payoff for the node changes dynamically.

Similar to Komathy et al's work, Mohi et al. [64] proposed a neighbor cooperation enforcement mechanism for ad hoc network. They used a game-theoretic approach using Bayesian games. The forwarding rate of every neighbor node is monitored. The forwarding rate is used to derive the payoff matrix of the Bayesian game. The trustworthiness of each neighbor is evaluated by playing 2-player Bayesian game with each neighbor node. Similar to Komathy et al's work, this proposed work address only selfish nodes. Other malicious routing behavior are not considered.

2.4.3 E-Hermes

Zouridaki et al [120,121] proposed a novel cooperative trust establishment called E-Hermes, which cooperatively combines recommendations from the neighbors with the local assessment to compute the trustworthiness measure for each neighbor. In this scheme, nodes monitor the neighbor nodes' packet forwarding behavior and compute the trustworthiness measure locally using the number of packet forwards by each neighbor. The number of forwarded packets by a neighbor node is used as the trust metric to evaluate the trustworthiness. The local trustworthiness measure is shared with the neighbors and the nodes compute a cooperative trust measure by combining the local trustworthiness measure from the neighbors. The computed trust metric is used as a measure to evaluate the benignity of neighbors. Unlike conventional approaches, which determine trustworthiness based on the neighbor node's behavior during the route discovery phase, E-Hermes computes the trustworthiness measure during data transmission phase. Hence, this scheme reacts to any node misbehaving after the route establishment phase. This is new.

The focus of this work is to monitor and measure the nodes' benignity during the data transmission phase. However, malicious behavior during the attack discovery phase is not considered by the authors. Hence, malicious behavior such as, spoofing, fabrication will go unnoticed by the trust system. Furthermore, the trust mechanism monitors only one statistical network parameter, node forwarding rate. This decreases the robustness of the trust assessment.

Sharing of security information over unreliable channels creates threats against the trust system. The authors have not considered over-riding, which is a phenomenon in which the trust computation process is dominated by one node's local trustworthiness measure. A malicious node can declare a high trust value for another colluding malicious node in the network. In this scenario, the process of combining neighbors' local trustworthiness is under threat.

2.4.4 Dempster Shafer Approach

Dempster Shafer theory (DST) [85] is a powerful probabilistic approach, which is the generalization of Bayesian theory of subjective probability. Dempster Shafer theory implements belief functions which are based on degrees of belief or trust. Degrees of belief are represented as a belief function rather than Bayesian probability distribution. Probability values are assigned to sets of possibilities rather than single events. The DST is based on two key ideas; *the idea of obtaining degrees of belief for one question (is the node malicious?)* and *Dempster's rule for combining such degrees of belief when they are based on independent items of evidence (is the evidence reliable?)*.

To illustrate the idea of obtaining degrees of belief for one question from subjective probabilities from another, consider a neighbor node A with reliability of 0.9 and therefore the unreliability is 0.1. Now, the neighbor node A says that another node M in the neighborhood is malicious. This statement is true if node A is really benign, is not necessarily false if it is malicious. Therefore, according to DST, zero degree of belief is assigned for the fact "*node M is benign*". This zero degree of belief does not mean node M is benign (like zero probability), but it merely means that node A's statements gives no valid reason to prove that node M is benign.

To illustrate Dempster's rule for combining degrees of belief, consider that the neighborhood consists of another node B, which has the reliability of 0.9 and supports the decision independent of node A's statements. Then, the probability that neither is reliable is $0.1 \times 0.1 = 0.01$. Therefore, the reliability of combination of the two evidences is $1 - 0.01 = 0.99$. In case, Node A and B contradict each other, either node A is true or node B is true but not both. A third possibility is that both nodes are false. The probability of one node being correct and the other node wrong is 0.09, whereas the probability of both being wrong is 0.01. Hence the probability of node M being malicious is 9/19 (node A reliable) and node M being benign is 9/19 (node

CHAPTER 2. RELATED WORK

B reliable).

In [25], Dempster Shafer theory has been used to design an IDS for ad hoc networks. In this work, static probabilistic weights are assigned to nodes. The weights can be used to signify the reliability of their information. A decision is reached using DST, as described above. The basic assumption in this approach is that, the majority of the nodes are benign and not faulty.

In ad hoc networks, the ability of malicious nodes to emulate non-existent nodes renders this approach ineffective. Since false majority can always be created by a malicious node, decision is unreliable. Furthermore, static probabilistic weights are not suitable for the dynamic environment of ad-hoc networks. Nevertheless, the application of Dempster Shafer theory provides strong security to the security system from attacks. Nodes can only support other nodes' benignity but not their maliciousness. This safeguards integrity of the trust evaluation process.

Publications:

- J.F.C, Joseph, Amitabha Das, Seet Boot Chong, Bu-Sung Lee, "*Intrusion Detection Systems in Ad hoc Networks*", Handbook of Research on Wireless Security, Idea Reference Inc. Jan 2008.

Chapter 3

Threat model

Definition of a threat model [5] is the first step in the design of security architecture. Generally, a threat model is defined in three stages. In the first stage, the characteristics of the system that affect the security architecture design are analyzed and defined. The second stage identifies potential threats [6] and the third stage analyzes the security vulnerabilities of the entity in the system which the security architecture is designed to protect.

In this chapter, we present a threat model for security architecture for ad hoc routing protocols. First, the inherent characteristics of ad hoc network which affect the security architecture design is defined and discussed. Secondly, we identify potential routing attacks. To classify the identified attacks, the threat model enumerates a taxonomy of routing attacks in ad hoc networks. Thirdly, the challenges of detecting routing attacks in ad hoc network are studied. Finally, as the protected entity of the security architecture design is the routing protocol, the threat model analyzes the security vulnerabilities of the routing protocol.

3.1 Characteristics of Ad-hoc Networks

Ad-hoc networks differ from native wired/wireless networks in various aspects [44,46]. These unique characteristics of ad-hoc networks render typical security systems unsuitable. In this section, these unique characteristics of an ad hoc network are discussed. The fundamental concept of ad-hoc networks is to have seamless connectivity, without infrastructure or centralized control. The lack of infrastructure and the absence of a trusted centralized control node make security system design for ad hoc networks a challenging task. Furthermore, factors such as mobility, physical protection of nodes, etc. affect the design of security models for ad-hoc systems. These factors are enumerated below.

3.1.1 Lack of Fixed Infrastructure / Centralized nodes

Ad hoc networks do not have a fixed infrastructure [65]. Typically, in conventional networks, the infrastructure provides a secure location for the implementation of critical security mechanism. Due to the absence of infrastructure, ad hoc networks do not have a safe and efficient location to implement the security system. Additionally, operations such as, control, maintenance and other administrative functions has become hard in a distributed and infrastructure-less network. The only and apparent solution is to install these critical modules in end-user nodes. Implementing critical security systems in un-reliable end-user nodes pose a real challenge.

Conventional network have traffic concentration areas, referred to as choke points [89], where security systems can be placed and implemented efficiently. Trusted centralized control nodes are placed in these choke points to monitor and control the network. First, due to changing topology, ad hoc networks do not have fixed choke points to install security systems. Secondly, due to the infrastructure-less characteristic of ad hoc networks, centralized trusted nodes are absent. This makes the

network monitoring and control for security mechanisms, a challenging issue for ad hoc networks.

Every node in an ad hoc network has equal responsibility in network functions, such as routing, maintenance, etc. Nodes have to rely on neighbor nodes for routing and data forwarding. In other words, nodes have to trust neighbor end-user nodes for critical functions. As neighbors can be potential attackers, trusting unknown neighbors affects the integrity of security and other critical systems.

3.1.2 Node and Link Properties

Nodes in ad hoc network are generally mobile and wireless [45]. This property of ad hoc network node raises concern on the physical protection of end-user devices. A physically compromised node can be exploited in numerous ways to attack the network. Sensitive security information can be extracted from the compromised node and used against the network [104]. Also, the attacker can use the knowledge of security system design implemented in the compromised node and can identify vulnerabilities of the security system. This knowledge can be used against any node in the network and security system may fail.

Additionally, node mobility makes connections between nodes, transient. Due to mobility, the topology of the network keeps changing constantly. Therefore, a node will not be able to get security specific information from its neighbor node permanently. In other words, the time frame for security information to be valid in the ad hoc network becomes very small.

The wireless signal propagation and channel connectivity between nodes make the security design for ad hoc networks more challenging. Due the omnipresence of wireless channel and ease of physical access to the channel, wireless networks are hard to protect. Attacks such as eavesdropping, active masquerading, etc are more possible in wireless networks than in a wired network. Also, due to wireless channel

conditions, attacks are stealthier and hard to detect. Furthermore, the most notorious of all attacks, the Denial of Service (DoS) attack can be achieved easily in wireless networks by jamming the wireless channel or by routing attacks.

3.1.3 Energy and Delay Constraints

Ad-hoc network nodes are mostly limited in battery and computational resources. Hence, energy constraints are applicable on security system. Typical symmetric encryption algorithms such as, 3DES (Triple Data Encryption Standard) [89], AES (Advanced Encryption Standard) [89] and asymmetric encryption algorithms such as, RSA (Rivest, Shamir and Adleman) [89] and its variants incurs high computational overhead that may drain the battery of the mobile node. Additionally, energy-targeted attacks such as SDT (Sleep Deprivation Torture), which aims to drain the mobile node's battery also need consideration while designing an ad hoc security system. [19]

Security systems are delay-sensitive. Especially, in highly dynamic environments, delay guarantees are necessary for the security system to function properly. This necessity arises from the transient link connectivity in the ad hoc network, which was discussed earlier. However, delay guarantees are hard to achieve in dynamic networks because of wireless connectivity and mobility.

3.2 Routing Security in Ad hoc Networks

The ad hoc routing protocol is designed to operate in an environment without infrastructure and trusted centralized nodes. The efficiency of the ad hoc routing protocol mechanism depends on the cooperative behavior of neighbor nodes. Nodes assist each other for routing packets. According to the ad hoc routing protocol design, every node has the responsibility to route and forward packets for its neighbors. This

Figure 3.1: Ad-Hoc Routing Insecurity: Route Invasion

dependency on neighbor nodes' cooperativeness for routing has spawned an unprecedented security threat, i.e., routing attacks.

Routing security is an issue that is unique to ad hoc networks and becoming more important as every node in the network functions as a router [17]. Conventional security systems are designed to counter threats at IP, transport and application layer protocol. In ad hoc networks, it is important to provide routing protocol security. This has raised new challenges and issues, since securing a routing protocol has never been an issue for security system designers for legacy networks.

In an ad hoc network, due to the nodes' privileges, any node can add/modify/delete routes. This functionality is the primary cause of the vulnerability of ad hoc routing protocols. A malicious node can send malicious routing control messages to its neighbors. Since, ad hoc networks are highly distributed, decentralized and dynamic systems, preventing or detecting a malicious routing message becomes difficult. Moreover, semantically distinguishing between malicious and benign routing messages is infeasible. Routing insecurity introduces new attack possibilities. Active attacks such as route invasion, and route disruption, cause active damage to the network routing functions. Route invasion and disruption attacks aim to modify, add or delete routes

CHAPTER 3. THREAT MODEL

by sending malicious routing information over the network. Passive attacks such as route monitoring, etc, try to eavesdrop in order to steal sensitive information. To illustrate some of the difficulties and to familiarize routing insecurity in ad hoc networks, a trivial attack scenario is considered and presented below.

Let us examine route invasion, which is a trivial but destructive attack. In figure 3.1a, the benign route between S and D is through 1. In figure 3.1b, node M sends malicious routing control message, stating that, it has better route to D than through node 1. This modifies the path for $S \Rightarrow D$ from $S \rightarrow 1 \rightarrow D$ to $S \rightarrow M \rightarrow 2 \rightarrow 3 \rightarrow D$. The modified path is not only inefficient; it includes the malicious node M in the path. This extends the attack possibilities for the malicious node M on node A or B . To thwart intrusion detection, node M can impersonate node 1 and can provide falsified routing information which supports its cause.

Due to the absence of centralized authority and infrastructure, node S has no trusted arbiter to get advice regarding whether the announced path is benign or otherwise. Malicious node M has free access to the wireless channel and can exhibit anonymous routing attacks over S .

Mobility and transient network associations make the detection of malicious routing control messages difficult. In the above example, node S , based on its local knowledge, will not be able to determine whether node M is on the shortest route to D or acting maliciously. Since, even if node M is not on a shortest/optimal path to node D now, due to changing topology, that may change at a point of time in the future. It is difficult to distinguish between malicious and benign behavior, as a malicious behavior resembles benign behavior. Therefore, detection becomes very challenging.

Figure 3.2: Routing Attacks Taxonomy

3.3 A Taxonomy of Ad hoc Network Routing Attacks

Malicious routing behavior can be categorized into a hierarchy of behaviors depending on the complexity of the actions and the extent of damage caused by them. This hierarchy is shown in figure 3.2. At the highest level of this hierarchy are the routing attack scenarios which is a complex attack plot designed by the attacker to achieve a particular disruptive effect on the network. A routing attack scenario consists of a set of attack behaviors executed in a particular sequence to achieve a malicious goal. As

shown in figure 3.2, there are seven major known attack scenarios on ad hoc routing protocols.

The attack behaviors identify the broad strategies employed by the attacker to achieve the ultimate attack goal. They are categorized into three classes, namely, route invasion, route disruption and route monitoring. These behaviors aim to maliciously add, modify or delete routes in the network.

An attack behavior can be further decomposed into elements of the attack behavior. The elements of an attack behavior are the simple steps or precise techniques used by the attacker to effect a certain behavior. These include, spoofing, sinking, modifying, rushing and replaying behavior. In what follows, these three hierarchies of attack behaviors are described in more detail.

3.3.1 Elements of Attack Behavior

Sinking [100] refers to the intentional dropping of packets by a node instead of forwarding them to the next node along the intended route. The objective of sinking is to either selfishly conserve resources like battery, medium, etc. or drop critical packets like routing messages to disrupt routing functions. While the later can form inconsistent routes, selfish nodes in a sparse network may partition some parts of the network making them unreachable.

Rushing [80] is a malicious behavior, which is unique to reactive routing protocols, e.g. DSR. In a reactive routing protocol, when a node wants to establish a path to the destination, it broadcasts a Route request (RREQ) to the neighbor nodes. Neighbor nodes forward this RREQ by flooding it in their respective neighborhood. This process is continued until the path to the destination is found. To control the flooding, a node forwards only the first received RREQ. An attacker can exploit this feature by rushing the RREQ towards the destination, thus getting preference over other nodes. The path established by rushing will certainly contain the malicious

CHAPTER 3. THREAT MODEL

node, thereby giving it an advantageous position for further attacks.

The malicious behavior of retransmitting old routing messages to form inconsistent routes is referred to as replay attacks [33]. The objective of replay attacks is to disrupt routing functions and cause DoS. Replay attacks are less severe but can thwart intrusion prevention mechanisms such as, encryption and digital signatures. The scope of this attack is confined to the immediate neighborhood and the extent of damage depends on the mobility of the network nodes. High mobility causes routes to change frequently. In such transient environments, replay attacks can cause serious damage to the routing functions of the network.

Spoofing [38] is a process of impersonating another node in the network. In routing, the attacker can impersonate a node by changing the source address of the routing message. The ability to spoof routing messages gives a malicious node a wide range of possibilities to attack the network routing functions. Spoofing can create non-existent neighbor nodes and can help emulate supporting information to the malicious routing information disseminated by the malicious node. The objective of spoofing include, route invasion, route disruption, network partitioning and DoS. In proactive routing protocols, spoofing behavior has the potential to corrupt the entire topology.

When an attacker changes contents in the routing message like, declaration of neighbors, sequence numbers, etc, instead of the source address, it is simply usually referred to as a modification or fabrication attack [38]. Modification of routing message contents affects the network functions significantly. The objective of modification is similar to spoofing. However, unlike spoofing, its effect is confined to its neighborhood nodes only.

Figure 3.3: Black Hole Attack Scenario

3.3.2 Attack Behavior

As shown in figure 3.2, the routing attack objectives are classified as, route invasion, route disruption and route monitoring. Basically, these objectives are achieved using attack behaviors that aim either to add new malicious routes or modify or delete existing routes in the network.

The security vulnerability of the routing protocol can be exploited in two ways. The vulnerability is either used indirectly as a platform for more serious attacks from the higher layer protocols or the vulnerability is used directly in attacking the routing system. For the former kind of threat, a malicious node invades a route and stages a vantage point for higher layer attacks. In the later kind of threat, the attack directly disrupts the network functions by adding inconsistent malicious routes, deleting or modifying existing benign nodes.

Additionally, passive attacks such as, route monitoring are also possible, which can reveal the location of the attacked node(s). In many applications, such as military networks, location information can be sensitive. As stated earlier, passive attacks are hard to detect and prevent.

3.3.3 Attack Scenarios

Attack scenarios are complex sequences of attack behavior executed to achieve a malicious objective. In this section, a few notorious attack scenarios are discussed in detail.

3.3.3.1 Black Hole Attack

Black hole attacks [49] cause packets to disappear from the network. Most often, black hole attacks are effected by simple sinking behavior, which was discussed in section 3.3.1. In this case, attacker's objective is route disruption by deleting routes. The attack scenario primarily consists of elementary sinking attacks. Besides sinking, black hole attacks have other variants. In one variant, the data traffic is forwarded to a non-existent or another malicious node, where the data is dropped. This forwarding behavior before the actual sinking will make detection of sinking behavior hard. In this attack scenario, attacker's objective is route disruption by modifying and deleting routes. The attack scenario consists of elementary sinking and modification attacks.

Another variant of black-hole is a gray-hole attack. In gray-hole attacks, only selected data traffic is forwarded to a non-existent or another malicious node for sinking. Gray-hole attacks are hard to detect due to the sporadic sinking behavior.

Other than sinking, there are numerous ways by which a malicious node can coerce benign nodes to drop incoming traffic. Essentially, the malicious node achieves this by disrupting the benign route between source and destination. Route disruption can be enforced using spoofing and modification of routing messages.

Example 1

In figure 3.3, a malicious node M spoofs a benign node V and declares the route availability to the destination node D in the spoofed routing message. Since, node V is not receiving the data or does not have path to node D, node V will not forward the traffic and drop the traffic benignly. Thus, the attacker fulfills its goal of sinking

Figure 3.4: Wormhole Attack Scenario

the traffic. Besides sinking incoming traffic, through the above spoofing behavior the attacker also makes the neighbor nodes to believe that node V is malicious.

3.3.3.2 Wormhole

In this attack scenario, two malicious nodes belonging to two different neighborhoods create a tunnel between them [86]. Through the tunnel, routing messages transmitted by malicious nodes' neighbors are exchanged and replayed in the neighborhood, where these nodes are not present. This behavior makes the neighbors believe that the two malicious nodes are one-hop neighbors to nodes in the other neighborhood. Thus, all routes established between these nodes in the two neighborhoods will contain at least one of the malicious nodes.

Example 2

Consider two malicious nodes M1 and M2 present in two neighborhoods NN1 and

CHAPTER 3. THREAT MODEL

NN2, which are connected by more than two hops, as shown in figure 3.4. Malicious nodes M1 and M2 create a tunnel and channels all routing message traffic that is received in their corresponding neighborhood through the tunnel to the other malicious node. Both M1 and M2 replay the routing messages received through the channel in their own neighborhood. This will disrupt the perception of topology for the nodes in both the neighborhood. Nodes in NN1 will believe that nodes in NN2 are one-hop away through the node M1. Similarly, nodes in NN2 will have wrong perception that NN1 is at one-hop distance through M2. Now consider the scenario where a node in NN1 and another node NN2 want to establish a route between them. The route established will certainly contain either M1 or M2. Hence, the colluding malicious nodes will control the link.

After a node has control over the link, their next behavior may be a passive or active attack. The malicious nodes can passively monitor traffic for attacks such as, traffic analysis, location monitoring, etc. Passive attacks are a grave threat to military applications and tactical communications. If the nodes attack the controlled link actively, such as sinking or manipulating the traffic, this may cause a DoS, etc.

It is interesting to note that security mechanisms such as, encryption and access control [67] does not prevent this attack. Furthermore, as the traffic is tunneled, detection of wormhole attacks is hard.

3.3.3.3 Network Partitioning

A malicious node can delete routes to isolate one or more parts of the network and render the nodes in the isolated network sector unreachable. The above routing attack is referred to as network partitioning. Statistically, detection of network partitioning is trivial. However, the malicious node can thwart detection by exhibiting low layer attacks such as, channel jamming, MAC flooding.

3.3.3.4 Cache Poisoning

Corrupting the routing information (tables) stored locally in the nodes is referred to as cache poisoning. A malicious node can add, delete or modify routes in a node's routing cache by exhibiting any of the five attack behaviors discussed earlier. The attack scenarios such as, black hole, worm hole, etc. also corrupt the routing cache. However, in these scenarios cache poisoning is one part of the complex attack plot. The extent of damage by cache poisoning is confined to the malicious node's neighborhood.

3.3.3.5 Selfishness

Ad hoc networks use a cooperative routing mechanism for reaching nodes which are not present in the neighborhood. If nodes are not cooperative, then the routing efficiency will be reduced. Selfish nodes are nodes which are non-cooperative and try to conserve their resources such as, battery, wireless channel bandwidth, processing time, etc. by not participating in the cooperative routing. In a reasonably populated network, selfish behavior will only degrade the routing efficiency. However, if the network is sparsely populated, selfish nodes will cause some areas in the network to become unreachable.

3.3.3.6 Sleep Deprivation

In ad hoc networks, most nodes are mobile that have limited resources such as, battery power and processing capacity. A malicious node will constantly flood a node with junk routing messages. These junk routing messages will be processed by the victim node, which will waste its processing time and may eventually drain its battery. This attack scenario is referred to as sleep deprivation.

3.4 Challenges in Detection of Routing Attacks

The security architecture presented in this dissertation uses primarily intrusion detection and trust management to secure the ad hoc routing protocol. Efficiency of the security architecture lies on its ability to detect routing attacks. In this section, the major challenges of detecting routing attacks are presented.

3.4.1 The Base Rate Fallacy

The effectiveness of detection system in the security architecture depends not only on the accuracy of the detection mechanism, but also, on the probability of occurrence of malicious events. This phenomenon is called the base rate fallacy and it has been explained by Axelsson in [12]. This fallacy arises from the famous Baye's theorem and conditional probability axioms [31]. According to Baye's theorem, when the security system detects a behavior as malicious, then the reliability of decision or the probability of detection to be true is expressed as:

$$= \frac{P(\text{Malicious}) \cdot P(\text{DetectionAccuracy})}{P(\text{Malicious}) \cdot P(\text{DetectionAccuracy}) + P(\text{Benign}) \cdot P(\text{DetectionError})} \quad (3.1)$$

Similarly, when the security system detects a behavior as benign, then the reliability of the benign behavior detection is expressed as;

$$= \frac{P(\text{Benign}) \cdot P(\text{DetectionAccuracy})}{P(\text{Benign}) \cdot P(\text{DetectionAccuracy}) + P(\text{Malicious}) \cdot P(\text{DetectionError})} \quad (3.2)$$

The above two equations determine the effectiveness and efficiency of the security system. It can be observed from the equations that probability of occurrence of malicious events, $P(\text{Malicious})$ and probability of occurrence of benign events, $P(\text{Benign})$,

CHAPTER 3. THREAT MODEL

play a vital role in determining the effectiveness and efficiency of the detection system. Note that the detection accuracy in the above two equations differ. In determining reliability of the malicious behavior detection, detection accuracy is the accuracy of detecting malicious events, whereas in the case of determining reliability of the benign behavior detection, detection accuracy is the accuracy of detecting benign events.

To further illustrate the significance of the base rate fallacy, let's consider the example similar to the one used by Axelsson. Consider a detection system, which has detection accuracy of 95% in case of malicious events, and 98% for benign events. In other words, the probabilities of correct detection of malicious and benign behavior are 0.95 and 0.98, respectively. Therefore, the probabilities of misclassification or detection errors for malicious and benign behaviors are 0.05 and 0.02, respectively. Typically, occurrence of a malicious event is 1 in 100,000 events. Hence, probability of malicious $P(\text{Malicious}) = 10^{-5}$ and probability of benign events $P(\text{Benign}) = 1 - 10^{-5}$. The above equations can be used to compute the probability of malicious behavior detection to be true and is expressed as:

$$\begin{aligned} &P(\text{Malicious Behavior Detection to be true}) \\ &= \frac{10^{-5} \times 0.95}{(10^{-5} \times 0.95) + ((1 - 10^{-5}) \times 0.05)} < 1\% \end{aligned} \quad (3.3)$$

In a similar way, the probability of benign behavior detection to be true can be computed. And expressed as:

$$\begin{aligned} &P(\text{Benign Behavior Detection to be true}) \\ &= \frac{10^{-5} \times 0.98}{(10^{-5} \times 0.98) + ((1 - 10^{-5}) \times 0.02)} \approx 99\% \end{aligned} \quad (3.4)$$

Therefore, it is evident that probability densities of malicious and benign events in the total population have significant effect on the reliability of the detection outcome.

Figure 3.5: The Decision Dilemma in Ad hoc IDS

In simple terms, as the occurrence of malicious events becomes rare, the detection outcome of the security system becomes less and less reliable.

3.4.2 Theoretical Limitations

Theoretical limitations refer to the limitations, when the detection system is based on a logical approach, such as "specification based" [28] and "state graph" [106] based approaches, as well as decision protocols based on distributed agreement [57]. Even though the detection system in conventional networks also face similar issues, ad hoc networks are affected more. This section illustrates the difficulties of detection system design for ad hoc networks using a logical and deterministic approach. Furthermore, Byzantine Agreement Problem [57] is used to illustrate the issues pertaining to knowledge sharing.

3.4.2.1 Decision Dilemmas

When a node receives contradictory route information, one from a benign node and another from a malicious node, a decision dilemma occur. The node will not be able to decide which information is correct. Nodes may send incorrect information because

CHAPTER 3. THREAT MODEL

Figure 3.6: The Decision Dilemma in Ad hoc IDS in Asynchronous Channel

they are either faulty or malicious. To illustrate the possibilities, consider a scenario as shown in figure 3.5. In this scenario, two benign nodes (node N and B) want to communicate routing information in a synchronous channel. In a synchronous channel, the messages are received in the same order as it is sent. Benign node B sends benign routing control message to neighbor node (benign) N. Now, a malicious node M impersonating B sends a routing control message to node N which contradicts information sent through B. From the perspective of N, it receives two contradicting routing information.

Here, node N will not be able to decide which routing information is benign. But, it should be observed that, in the case of synchronous channel, the attack is detected because node N is receiving contradictory information pertaining to a route. Although attack is detected, the inability of node N to identify the source of malicious message renders the detection futile.

The above example assumed synchronous channel, such that, the message system is perfect. However, mobility of nodes renders the channel asynchronous. In an asynchronous channel, packets may arrive in out-of-order. This introduces further

CHAPTER 3. THREAT MODEL

Figure 3.7: The Intractable Problem

difficulties for intrusion detection in ad hoc networks. To illustrate the issue, consider the scenario depicted in figure 3.6.

In this scenario, node N receives two contradicting routing information. In the benign case, this is due to the delayed arrival of routing control message at N, which was sent by node B at time $T - t_A$, where T is the current time. Typically, the delayed message is dropped by comparing the sequence numbers. The mechanism of packet dropping based on sequence number introduces new attack possibilities in the ad hoc network. Since, when a malicious node M sends a routing control message impersonating node B with a higher sequence number, Node N will consider the benign packet as delayed and accept the malicious routing control information as valid. It is evident that, from the received messages, node N will not detect the malicious behavior, since the malicious routing behavior resembles benign behavior. In this case, detection itself becomes difficult.

A third scenario is illustrated in figure 3.7. In this setup, a benign node B sends routing information to node N. After some time, node B moves out-of-range of N.

CHAPTER 3. THREAT MODEL

Figure 3.8: Well-Known Solution to Decision Dilemma in Ad hoc IDS.

Sensing node B's mobility, a malicious node M impersonates B and starts sending malicious routing control information to node N. Since the routing messages are not-authenticated, there is no distinction between the transmissions of node B and the malicious node M.

It can be pointed out that the trivial scenarios discussed above constitute the elementary actions of intelligent attacks. Intelligent attacks aim to thwart attack detection. In practice, intelligence is known to provide stealthiness to attacks. These attacks are hard to detect. On the contrary, a sequence of trivial attacks is relatively easier to detect. A sequence of trivial attacks without intelligence will form an attack signature. Each event in the sequence will strengthen the fact that the behavior is malicious.

3.4.2.2 Byzantine Agreement Problem in Knowledge Sharing

The obvious solution to the above detailed decision dilemma scenarios, is sharing knowledge. The benign node B can request other benign nodes to support its routing information and help node N in deciding that B is benign. Researchers in this domain, inherently believe that sharing the knowledge is better solution for attack detection than using local knowledge only. However, Mishra et al. [63] in their sur-

CHAPTER 3. THREAT MODEL

Figure 3.9: Decision Dilemma by Non-existent Neighbors

vey of detection system architectures listed that sharing knowledge may affect the security of security system itself. In this section, we further examine issues that arise from sharing of knowledge.

As mentioned earlier, node **B** can request its neighbors, which are also **N**'s neighbors to support the benign routing control message. The accepted solution is shown in figure 3.8, where benign routing control message is supported by nodes **S**.

However, ad hoc networks have a characteristic which makes knowledge sharing highly unreliable. A malicious node **M** can create non-existent neighbors and emulate malicious behavior for them, as shown in figure 3.9. This gives the malicious node **M**, the possibility of creating nodes which support the malicious node **M** regarding information passed to node **N**. Therefore, malicious node **M** can always be the majority and control the IDS of node **N**, thwarting the possibility of a valid Byzantine agreement. The Byzantine Agreement problem [57] requires that at least more than two-thirds of the nodes should be benign for the benign nodes to reach a valid decision. In ad hoc networks, since majority can always be achieved by a malicious node, decision is always unreliable. Hence, knowledge sharing poses a grave threat to the integrity of security system.

3.4.3 Limitations of Probabilistic Inference

The most widely used methodology for detection system is probabilistic inference schemes. In legacy networks, detection systems based on probabilistic inference are found to be effective. These methods overcome the limitations of logical approaches by using probabilistic axioms and statistical inferences on observed benign and malicious behavior. It should be noted that, these schemes follow the same logical approach, but allows flexibility in detecting or deciding by using probability. Therefore, to some extent, the above discussed theoretical limitation still applies to these methods.

Some additional issues that affect the probabilistic methodology in detection system for ad hoc networks are discussed below.

3.4.3.1 Probabilistic Uncertainties

In most probabilistic methods, such as Bayesian decision theory [31], a priori probability is known [4]. A priori probability is the probability of occurrence of malicious events/attacks in the network. The effectiveness of any statistical method can only be computed when the real a priori probability is known. Significance of a priori probabilities was also illustrated by the base rate fallacy phenomenon.

However, a priori probabilities are unknown in ad hoc networks. This is due to the fact that, ad hoc networks are not well established and real world attacks are not yet experienced. When Bayesian approach is used, a priori probabilities are crucial in drawing the decision boundaries between malicious and benign behavior. Since the a priori probabilities are unknown, the efficacy of Bayesian methods over detection systems in ad hoc networks is unknown.

Furthermore, the probability density functions of features from malicious and benign events vary with respect to the network environment. For example, a vehicular communication network is sparse and highly dynamic, whereas a disaster management network is dense and less dynamic. Therefore, the conditional probability densities of

features from malicious and benign events vary vastly. This means that detection system efficient in a particular environment may not function as efficiently in a different network environment.

3.4.3.2 Statistical Bounds

A probabilistic method solely relies on statistical properties of the observed behavior. Probabilistic methods use extensively the statistical significance [103] of known behavior patterns. The ability to distinguish between benign and malicious routing behaviors with the help of known behavioral patterns is determined by the statistical significance.

The routing protocol design aims to minimize the message overhead for maximizing routing efficiency. Hence, the contents of routing control messages are kept as minimal as possible. As the result of this routing protocol design characteristic, the statistical information that defines a routing behavior is minimal. Thus, the statistical significance of the routing behavior definition is low. This greatly affects the statistical detection efficiency of security system.

3.5 Routing Protocol Vulnerability Analysis

The security threat analysis of a protocol consists of systematic steps of exploring the security vulnerabilities of the given protocol and the network environment. The threat analysis is carried out in three major stages. In the first stage, the protocol implementation is analyzed for understanding the propagation of malicious information and to study the scope of every type of routing message of the protocol. At the second stage, causal relationships between different attack behaviors and the extent of disruption caused are derived for all possible network conditions. Finally, based on the observations of the previous stage, contents in every type of routing message

in the protocol are assessed for security risk.

The security architecture proposed in this dissertation is designed and tested for the ad hoc routing protocol, Optimized Link State Routing (OLSR). Hence, in this section, the security vulnerabilities of OLSR is analyzed and discussed.

3.5.1 OLSR Fundamentals

OLSR [26] is a table-driven proactive routing protocol. OLSR was primarily developed for ad hoc mobile wireless networks. The protocol is very similar to Link State Routing (LSR) [65], where the topology information (link states) is disseminated to all the nodes in the networks. Flooding is used to disseminate the link-states and every node in the network has the detail information about the current topology of the network. OLSR optimizes the LSR by reducing the flooding redundancy and conserving bandwidth. This is achieved through Multi Point Relaying (MPR).

The fundamental concept of multipoint relaying is to selectively choose the relays/forwarders for information originating from a particular node. These selectively chosen nodes are called multipoint relay set of node. Each node has a set of MPR nodes. The selection of MPR nodes is aimed at reducing the number of nodes in the MPR set. At the same time, the MPR set should have connectivity to all two-hop neighbors.

OLSR has three types of routing control message namely, Hello packet, Topology control (TC) packet and multiple interface declaration (MID) packets. The Hello message is used for neighborhood sensing and MPR set calculation. TC messages are used to disseminate topology/link states to different nodes in the network through MPR nodes. MID messages are used to declare multiple interfaces for a particular node.

CHAPTER 3. THREAT MODEL

Figure 3.10: Protocol Analysis

3.5.2 Protocol Analysis

The first step is to derive the relationship between the input and the output of the protocol. The objective of this step is to identify the extent to which information received in a routing message is propagated. Additionally, the effects of input on internal/local resources of the protocol is also studied and organized as shown in figure 3.10. Events/local resources which generate different information for interacting with the neighbors are also examined. These relationships were derived from the documentation of the protocol such as RFCs [26]. Since, routing control messages are the only possible resources used by the attacker against the routing protocol, the analysis of OLSR starts by understanding the role of each control message and the effect it brings upon the network and the local resources. First, the effect of routing events on the local resources such as, routing tables, etc. are studied. In the second

step, the extent of propagation of routing information for various routing messages is examined.

3.5.2.1 Local Resources

In OLSR, the local resources in a node consist of routing tables, the MPR set and a number of information bases as follows:

- Neighbor Nodes Information Base
- Two-Hop Neighbor Nodes Information Base
- Multipoint Relay Selectors (MPRS) Information Base
- Topology Information Base
- Multiple Interface Declaration (MID) Information Base
- MPR Information Base

Some information bases are influenced by only one type of routing messages, whereas some are affected by multiple message types. For example, MPR set computation is influenced by the Hello message as it affects the one-hop and two-hop neighbor sets. Similarly, MPRS set is controlled by the TC message only. On the other hand, route calculation is an example of a local element that is affected by multiple routing message types, as it is influenced by all three of them. Such resources are critical from the security point of view as prevention and detection of attacks on such components are important.

However, it should be noted that the routing messages cannot directly influence these components. Only the information bases control these components. Thus, it is necessary to prevent any malicious information to propagate from the information bases to the security critical local components. This knowledge gives us insight as to the optimum location of a local security mechanism to prevent or detect attacks.

CHAPTER 3. THREAT MODEL

Figure 3.11: OLSR Information Propagation

3.5.2.2 Propagation Analysis

The next step in protocol analysis is examination of routing information propagation to other nodes in the neighborhood. From figure 3.11, it can be seen that information in both Hello and TC routing messages are propagated. However, information in Hello messages are forwarded only to nodes within two-hop distance, whereas, information in TC messages are propagated to nodes in the entire network.

Eventhough only TC messages are directly propagated, Hello messages control the information of TC messages generated by one-hop and two-hop neighbor nodes. A TC message contain the current MP RS set of source node. The neighbor nodes use Hello messages of other neighbors to compute the MPR set. Thus, Hello messages can influence the MP RS set of two hop neighbor nodes, thereby influencing the content in TC messages. This shows the potential of hello messages in propagation of malicious routing information.

CHAPTER 3. THREAT MODEL

Figure 3.12: Causal Relations - Spoofing Behavior

The above discussion shows the relationship between Hello and TC messages among nodes in neighborhood. The security system should guard the integrity of this relationship and prevent malicious information propagation. It was also revealed that a malicious Hello message, if not prevented, will lead to a malicious TC message generation in the two-hop neighborhood. In most cases, only malicious TC messages are responsible for any kind of routing attack. Malicious Hello or MID messages are used to indirectly generate malicious TC messages or modify benign TC messages.

3.5.3 Causal Relations - Effects and Behavior

In this stage of threat analysis, the contents of routing message are examined for security vulnerabilities. Every field in all the routing messages is analyzed for security vulnerabilities. The network environment plays a major factor in determining the scope of a malicious behavior. Hence in this step, causal relationships between dif-

CHAPTER 3. THREAT MODEL

ferent attack behavior and its effect on the network under various network conditions are derived. Figure 3.12 shows graphically the causal relations derived for spoofing behavior. The attack behavior's range of disruption depends on the current network connectivity (network environment) between the attacker and victim node.

Similar to the causal relations derived for spoofing behavior, we can derive the same for other fields in the routing message. These causal relations between the attack behavior and malign network effect are used to trace malign network effect to different attack behaviors and vice versa.

Consider a spoofing attack scenario, where a node M sends malicious routing message to node B. The forged originator IP address (OIP) used in the spoofed routing message belongs to node C, which may exist in a distant neighborhood or is non-existent. In this scenario, node M can either be malicious or a compromised node.

Given the above scenario, in this stage of threat analysis we explore the possible ways node B will react to the malicious routing information under various network environments. Furthermore, the effect on the node C, whose IP address is used for spoofing, is also examined.

3.5.3.1 States of Network Connectivity (Environment)

OLSR consists of three network connectivity states, namely,

- Symmetric channel (SYM)
- Asymmetric channel (ASYM)
- MPR connectivity between node (MPR)

Two nodes having symmetric state of link connectivity means that the communication between them is bi-directional [26]. On the other hand, if the nodes have

asymmetric link, then the communication channel is unidirectional [26]. The third state of network connectivity is that one node is an MPR to the other node. For instance, node C or A is a MPR to B. Then, the network state of the link between $A \rightarrow B$ is MPR and state of link in the reverse direction $B \rightarrow A$ is MPRS.

An arrival event of a Hello or TC routing message on node B may change the current protocol state of the node. In OLSR, since Hello message is used for link and neighbor sensing, an arrival of Hello message will change the link or neighbor status of the node. Thus, a malicious Hello packet which has a forged source IP address from a compromised or malicious node A may affect these states, as shown in figure 3.11.

Similarly, a malicious TC routing message affects the network depending upon the link state between nodes A and B. The malicious TC message will affect only the MPR state of the link connecting the nodes A and B.

3.5.3.2 Effect of attacks (Malicious Behavior)

Figure 3.12 shows the effect of spoofing behavior under different network conditions. A spoofed Hello routing message can cause disruption by adding/modifying/deleting link status or routes of nodes in its current neighborhood. If the link between $A \rightarrow B$ or $B \rightarrow A$ is not MPR, then the scope of the spoofing behavior is limited to disruption of a single route. If the link between A and B is MPR, this gives the attacker the potential to cause disruption to many routes.

From figure 3.12, it is evident that the malign effect on the network depends not only on the attack behavior but also on the current network connectivity (environment). The level of propagation of malicious information depends on the current link state between node A and B. As the scope of the spoofing behavior depends on the extent of malicious information propagation, we assess the bounds of the attack based on the current network conditions.

CHAPTER 3. THREAT MODEL

Figure 3.13: Risk Estimation of Routing Message Contents

As stated earlier, the effect of a spoofed TC message depends only on whether link $A \rightarrow B$ is MPR, or not. Similar to Hello spoofing, the MPR link status between A and B gives a wide range of attack possibilities.

It is important to note that, the effect of the attack also depends on the IP address that is used to forge the OLSR routing messages. Depending on the IP address used for the spoofing, there are two possible spoofing behaviors. In the first type of spoofing behavior, the attacker uses the IP address of existing node, mostly in a distant neighborhood. In this scenario, the spoofing behavior causes disruption in two neighborhoods, in the attacker neighborhood and also in the victim's (owner of the spoofed address) neighborhood. In the second type of spoofing behavior, the attacker uses the IP address of a non-existing node in the network. In this case, non-existent nodes are created. Usually, the attacker node uses these non-existent nodes to emulate support to its malicious goal.

3.5.4 Risk Estimation

The final step in security threat analysis is assessing and estimating the security risk of each field in every type of routing message in the routing protocol. The security

CHAPTER 3. THREAT MODEL

risk of fields in Hello and TC routing messages of OLSR are assessed and shown in figure 3.12.

The causal relations derived in the previous step are used to assess the level of vulnerability for each field in OLSR routing messages. For example, in the previous section we illustrated the potential of the attack behavior in which the attacker spoofs the source IP address of the routing message. The extent of potential disruption by this behavior is high. Thus, source IP field in routing messages is assessed as a *high security risk field*.

The above security risk estimation of individual elements of routing messages will help the security architecture design to focus on critical fields in routing control messages.

The inference from the security threat model guides the design of security architecture to address the issues of ad hoc routing protocol security. For instance, the unsuitability of cryptographic algorithms for securing ad hoc routing protocol was inferred by studying the characteristics of ad hoc networks in section 3.1. Unsuitability of conventional security models can be inferred from the threat model. Studying the different types of routing attack behavior helps to model different attack scenarios and their respective attack objectives. The difficulties in detecting routing attack were discussed in section 3.4. It was inferred that the number of features from the routing protocol communication defining a routing behavior pattern is very limited. This affects the detection accuracy. The protocol analysis discussed in section 3.5 explores the security vulnerabilities of OLSR. OLSR is the ad hoc routing protocol for which the security architecture is designed.

CHAPTER 3. THREAT MODEL

Publications:

- J.F.C, Joseph, Amitabha Das, Seet Boot Chong, Bu-Sung Lee, "*Opening the Pandora's Box: Exploring the Limitations of Intrusion Detection for Ad hoc Routing Protocols*", Volume 31, Issue 14, September 2008, Pages 3178-3189. Computer Communications Journal, Elsevier.
- J.F.C, Joseph, Amitabha Das, Seet Boot Chong, Bu-Sung Lee, "*Routing Attacks in Ad-hoc Routing Protocols*". Handbook of Wireless Ad Hoc and Sensor Networks, Springer London.

Chapter 4

Security Architecture Overview

This chapter gives an overview of the security architecture presented in this dissertation. As stated earlier in the introductory chapter, the security system design primarily uses intrusion detection and trust management mechanisms to provide a light-weight security solution for ad hoc routing protocols. In the previous chapter, it was shown that proactive security system which uses mechanisms like, cryptography, authentication, etc. are computationally expensive and not feasible in ad hoc networks.

Our security architecture uses a reactive approach where the security mechanism does not prevent but detects malicious routing behavior and minimizes the damage caused by avoiding and isolating the attack node in future communications.

4.1 Overview

As shown in figure 4.1, there are three major processes in the security architecture, namely, *detection*, *decision* and *response*.

Detection process is performed by the intrusion detection module, which is in the system, referred to as Routing Attack Detection Systems (RADS). The detection

CHAPTER 4. SECURITY ARCHITECTURE OVERVIEW

Figure 4.1: Processes in Light-weight Security Architecture

system consists of two primary components. The detection system uses a cross layer approach to improve the detection accuracy. The first primary component of the detection system is referred to as Cross-Layer based Routing Attack Detection System (CRADS). CRADS is an offline system. The second primary component of the detection system is referred to as Cross-Layer based Adaptive Real-time Routing Attack Detection System (CARRADS). CARRADS operates at real-time and is responsible for updating the detection system with changing network conditions and adding new routing patterns.

Decision is made to categorize the activity of neighbor nodes. This is done by the trust mechanism of the security architecture. The outcome of the decision process is a measure that reflects the trustworthiness of the assessed neighbor based on the benignity in its current and past routing behavior observed by the node. This measure is referred to as the trust metric.

In section 3.4.2.1 of chapter 3, the importance of cooperative decision was emphasized. On the contrary, in section 3.4.2.2 of chapter 3, the issues of knowledge sharing for decision making which affects the integrity of the security system were also highlighted. To overcome these issues, at the same time to build a collective decision-making system, the trust management system uses a corroborative approach rather than the conventional cooperative approach. The corroborative decision making process of the trust management system use a game theoretic methodology.

CHAPTER 4. SECURITY ARCHITECTURE OVERVIEW

Figure 4.2: Components of Light-weight Security Architecture

The response to the decision computed by the trust management process is done at the route computation process of the ad-hoc routing protocol. This process is referred to as response process. As the response to decision, the ad hoc routing protocol considers the trust metric as an additional parameter while evaluating routes. The two response actions to the decision include,

- Avoidance of untrustworthy nodes.
- Limitation of routing service to untrustworthy nodes.

An untrustworthy node will be given the least preference while establishing a route. Also, the node offers only a limited service to the routing request made by an untrustworthy node. This property of the response system will discourage untrustworthy nodes from launching routing attacks and also isolates the potential attack nodes. However, untrustworthy nodes that connect remote nodes will be used to establish routes with these remote nodes. This is a tradeoff between security and node connectivity. Nevertheless, this feature allows a untrustworthy node to revive its reputation by being cooperative and benign.

The aforementioned processes of the security architecture, namely, detection, decision and response is implemented using the three modules. The three modules include, CRADS, CARRADS and game-theoretic trust management. The modules and their components are illustrated in figure 4.2. In the rest of the chapter, a brief overview on each of these three components is presented.

4.2 Cross-Layer based Routing Attack Detection System (CRADS)

In section 3.4 of chapter 3, the challenges of detecting malicious routing behavior in ad hoc routing protocol communications were presented. CRADS is designed to address these challenges in offering a robust detection system for ad hoc networks. The design objective of CRADS is to offer a detection system with high detection accuracy and precision with low computational overhead.

To increase the detection accuracy, CRADS uses two strategies. First, CRADS uses a cross-layer approach to define a routing behavior pattern. CRADS uses statistical information from different layer protocols. This property of CRADS significantly improves the statistical information content in routing behavior pattern definitions. This have addressed the problem of lack of statistical information content in routing

CHAPTER 4. SECURITY ARCHITECTURE OVERVIEW

messages, which was discussed in section 3.4.3 of chapter 3. It was observed in the analysis that using a cross layer approach improves the accuracy and precision of the detection system.

To further increase the detection accuracy, CRADS employs sophisticated non-linear machine learning algorithms, instead of linear machines (classifier) that is used in most schemes proposed in the literature. Linear classifiers are fast, but do not yield good detection accuracy and precision. CRADS uses Support Vector Machines (SVM) [103], a non-linear machine learning methodology which is well known for its superior classification accuracy.

The combination of cross layer approach with non-linear machine learning is an unprecedented approach in the design of IDS for ad hoc networks. Researchers in ad hoc network security domain have avoided using non-linear machine learning techniques for detection system for ad hoc networks primarily due to their high computational cost. The use of a cross layer approach and non-linear machine learning algorithm has immensely increased the computational overhead of CRADS. To overcome this issue, CRADS uses a set of novel data reduction measures to reduce the training complexity of the SVM algorithm. This consequently reduces the computational overhead in CRADS. The data reduction measures used in CRADS aims to preserve the information content of the training patterns. Conventional dataset reduction methods, such as, feature selection/ranking, conventional sampling, etc. do not consider the inherent dynamic characteristics of routing patterns in ad hoc networks. Hence, in practice, these methods affect the accuracy of the detection process. However, the dataset reduction measures used in our proposed security architecture considers the characteristics of ad hoc network while reducing the dataset. Thus, its effect on detection accuracy is negligible. This can be observed in our simulation experiments.

The aforementioned methods of CRADS are implemented using four primary modules, shown in figure 4.2. The modules include, Data Collection module, Data Re-

duction module, Learning module and Validation module. The functions of these modules are discussed in detail in chapter 5, while a brief overview is given below.

4.2.1 Data Collection Module

The data collection module collects data from the network, MAC and physical layers. At each layer, the collection module monitors the events and computes time, traffic and topology statistics and records the feature values.

CRADS uses a hybrid detection strategy where the training patterns used to train the detection system consists of both benign behavioral patterns as well as attack behavior patterns. This detection strategy gives robustness to the detection system. The use of benign and malicious behavior patterns together for training increases the probability of deduction.

4.2.2 Data Reduction Module

The objective of data reduction module is to reduce the data in the training patterns without reducing its information content. The data reduction module reduces two parameters of the dataset, namely, number of features defining a routing pattern (columns in the dataset matrix) and number of routing patterns in the training dataset (rows in the dataset matrix). For the above reduction of the training dataset, the two processes are defined, namely, Association and Filtering.

Association is a process which reduces the number of features in the training patterns. The association uses predefined correlation functions to combine features without affecting their information content.

Filtering reduces the number of routing patterns in the training dataset. Routing patterns, which are redundant or hold less information content are removed from the dataset. Redundancy of routing patterns is measured using the Euclidean distance function. For measuring the information content of routing patterns, the process uses

a linear classification algorithm. Fischer's Discriminant Analysis (FDA) [31] is used by the filtering process to measure the information content of routing patterns.

4.2.3 Learning Module

The CRADS's learning module is essentially a non-linear SVM classification model and is trained by the SVM algorithm using the reduced training dataset. The result of SVM training is a set of support vectors which govern the decision boundary between the benign and malicious routing patterns. The mathematical and implementation aspects of this process will be discussed in more detail in chapter 5.

4.2.4 Validation Module

The purpose of the validation module is to check the adequacy of the trained SVM detection model. As the SVM training uses a reduced dataset, it is necessary to check whether the trained detection model represents the original dataset adequately. The implementation of the validation module is discussed in section 5.1.5.

4.3 Cross-Layer Based Adaptive Real-time Routing Attack Detection System (CARRADS)

Section 3.4 highlighted the effects of mobility on the security system design for ad hoc networks. These issues have raised the need for real-time adaptability of the detection system. The function of CARRADS in the security architecture is to offer real-time adaptability to the detection system. The design objective of CARRADS is to adapt the detection system at real-time by updating the detection model with current network conditions and adding new routing patterns, while incurring low computational overhead.

Adaptation is done using incremental learning [76]. However, conventional incremental learning methods are computationally expensive for mobile ad-hoc nodes. To overcome this issue, our design exploits the inherent characteristics of SVM.

A typical machine learning algorithm aims to define an optimal decision boundary using a polynomial function that best separates the benign and malicious routing patterns. SVM uses a unique and effective method of defining a non-linear decision boundary without deriving the polynomial function. SVM searches for vectors that lie on the decision boundary that separates benign events from malicious events. These vectors are called support vectors. Vectors are mathematical representation of routing behavior patterns. The property of SVM that only support vectors is enough to define the detection model is exploited. Using this unique property of SVM, CARRADS defines a novel adaptive incremental SVM (AI-SVM) for real-time adaptation of the detection system.

4.3.1 Adaptive Incremental SVM (AI-SVM)

To augment the ability of incremental learning to the conventional SVM, the intrinsic characteristics of SVM are exploited in the learning module. The intrinsic property that the support vectors found using the SVM algorithm is sufficient to define the detection model.

SVM is essentially an optimization algorithm, which searches the feature space for a direction and orientation, in which the benign events are best separated from the malicious events. The distance between benign and malicious is referred to as margins. SVM finds the direction in which the benign and malicious events are best separated by maximizing the margins using an optimization algorithm. SVM uses Lagrangian optimization methodology [14]. In Lagrangian optimization, every vector in the training dataset is associated with a Lagrangian variable. After the Lagrangian optimization process, vectors which have non-zero Lagrangian variable

value will be considered as a support vector. The Lagrangian values obtained during the optimization process stores the history of solution search direction. This property of the Lagrangian optimization method is exploited to further reduce complexity of incremental learning. The AI-SVM method restores the old Lagrangian values while incremental training of the detection model. This method preserves the original search direction of the last training and results in a solution more close to solution obtained using conventional SVM.

AI-SVM uses the computationally effective way of incremental learning by considering only the support vectors during retraining. For retraining, the support vectors and the new vectors together are used as the training dataset. The solution obtained using this method is sub-optimal as the non-support vectors are *'forgotten'* at each stage of retraining. The mathematical and implementation aspects of AI-SVM are presented in more detail in chapter 6.

4.4 Game-Theoretic Trust Management Mechanism

The third primary component in the security architecture is the trust mechanism. The concept of trust and trust relationships are well studied in the literature. Trust is defined as *"the willingness of a node A to be vulnerable to the actions of node B based on the expectation that node B will perform a particular action important to node A, irrespective of the ability to monitor or control node B"* [105]. Trustworthiness of a node is expressed parametrically as a pair of parameters, namely, the trust and the confidence in the trust.

In ad hoc networks, the primary objective of trust mechanism is to derive the trustworthiness of the nodes' neighbors. For this objective, the architecture's trust mechanism corroborate with the neighbors for determining the trustworthiness of its corresponding neighbor nodes.

CHAPTER 4. SECURITY ARCHITECTURE OVERVIEW

Every node in the network assesses locally the trustworthiness of its neighbor nodes. This assessment is referred to as the local assessment. As shown in figure 4.2, the trust mechanism relies on CARRADS for locally assessing the neighbors. The detection outcome of CARRADS has two parameters, namely, detection and confidence in the detection result. These two parameters form the local trust assessment of each neighbor as trust and confidence in trust, respectively.

The core process of trust mechanism consists of a corroborative process which fairly and systematically combines the local assessment of neighbors and builds a global consensus towards the trustworthiness of each neighbor. In addition to the local trust assessment, the corroborative process also considers each neighbor's topological significance to the node and its recent routing service contributions. The corroborative process is formulated as a multiplayer non-cooperative strategic game played between the node and its neighbors.

A game is an interaction between a set of players (i.e., nodes) [102]. A strategic game is a scenario in which the players have to choose their respective actions or decision simultaneously, without knowing other players' actions. Every player has different preferences on the possible outcomes in a game. These preferences are represented quantitatively using a matrix, referred to as the payoff matrix. The payoff for the player is computed using the payoff matrix based on other players' decisions. Thus, the strategic game is modeled as a tuple $\langle N, A_i, \pi_i \rangle$, where N is the set of players, A_i is the set of action profiles for each player in N and π is the payoff matrix representing the preferences of the different outcomes in game.

The corroborative trust evaluation process is formulated using the above strategic game formulation. For assessing each neighbor, the node and its neighbors (excluding the assessed neighbor) play a multiplayer strategic game which is referred to as the trust game. Each node in the ad hoc network defines its own payoff matrix for each neighbor based on the neighbor's topological significance to the node and local routing

service contributions been offered by the assessed neighbor to the node. The game is played iteratively at real-time. The payoff value obtained during the trust game process is proportional to the confidence in the decision made by the trust mechanism. Thus, the decision D_i and the payoff derived during the trust game process are the two trustworthiness parameters, namely, trust and confidence in trust, respectively.

The objective of the trust game process is to maximize its individual payoff as well as the collective payoff, which is the sum of all payoff received by all neighbors. In other words, the trust game process maximizes the confidence in the local decision of the node as well as maximizes the collective confidence in the neighbors' local decision. During game-play, maximization of payoff is achieved by defining a systematic decision selection criterion for the nodes. Also, a set of game equilibrium conditions are defined to guide the nodes towards the social optimum between the neighbors and to maintain equilibrium at the social optimum. These conditions are dealt in more detail in chapter 7.

In the analysis of trust game process, it was observed that one node alone cannot change the network's perspective on a node's trustworthiness. However, a cluster of confident nodes can change the network's perspective. Thus, the integrity of trust evaluation and propagation is protected.

4.5 Response

The response function of the security architecture is used to integrate the security services within the ad hoc routing protocol. The response function integrates the trustworthiness metric within the route computation process in the ad hoc routing protocol design. Hence, at every route establishment processes; both distance and trustworthiness metric will be used together to compute optimal and secure routes.

Using the security services of the presented architecture, the untrustworthy nodes

CHAPTER 4. SECURITY ARCHITECTURE OVERVIEW

are avoided in routes established using the ad hoc routing protocol. However, avoiding untrustworthy nodes will not always be possible. This is due to the fact that, untrustworthy nodes may be in a significant topological location offering unique connectivity to remote nodes. In such a scenario, the node has to define a tradeoff between connectivity and security of the network.

Untrustworthy nodes are limited in their ability to launch routing attacks by limiting routing service offered by nodes in the neighborhood. This also gives an opportunity for untrustworthy nodes to gain reputation among its neighbors. The efficiency and performance of the offered security services is presented in section 7.5.

Chapter 5

CRADS

In this chapter, the design and functions of the first primary component of the security architecture, namely, CRADS is presented. CRADS is a cross layer based Routing Attack Detection System (RADS). The design objective of CRADS is to improve the detection accuracy by increasing the information content in the routing pattern definitions and to minimize the computation overhead incurred by the detection process.

5.1 CRADS Architecture

CRADS design is shown in figure 5.1. The principal components are the data collection module, the data reduction module, and the learning & the validation module. Data reduction is necessary to reduce the processing overhead incurred by machine learning methods, in this case the SVM algorithm. Data reduction module decreases the number of features and events in the training data. Validation is used to check the optimality of defined classification model. Since a routing event/pattern is mathematically represented as a vector, hereinafter event and vector will be used interchangeably.

Figure 5.1: CRADS Design

5.1.1 Configuration

In a cross layer based IDS, behavioral information from two or more layers is used in the detection process. There are two configurations in cross layer designs.

- Multiple Data Collection and Multiple Data Analysis (MCMA)
- Multiple Data Collection and Single Data Analysis (MCSA)

In the case of MCMA, every chosen protocol layer has its own data collection and analysis module and their detections are performed independent of other layers. However, for the final decision, the predictions from individual layers are weighted and correlated. It is similar to having individual detection system at each chosen protocol layer. This configuration is adopted by most cross layer detection systems proposed in the literature and their results are promising [42,95,113]. However, RADS at each layer increases the computational complexity of the system. The data analysis

CHAPTER 5. CRADS

which is typically a computational intelligence based classification technique, incurs high computational overhead. Multiple autonomous data analysis modules operating concurrently at each layer are computationally expensive for ad hoc network nodes.

In the case of MCSA, a single data analysis module is located in an optimal location (layer). For every chosen layer, a data collection module is designated. These collection modules collect behavioral information consisting of layer-specific features and forward it to the data analysis module. The features from different layers are processed at the single data analysis module for a decision about the unclassified behavior. Comparing with MCMA, MCSA reduces the overhead by using a single data analysis component in a convenient location. Additionally, correlating features from different layers is the true essence of a cross-layer based detection system. Combining features will increase the information of the features, as *"between layer"* discrepancies will be revealed. Therefore, attacks targeted at or sourced from any layer can be detected promptly. CRADS uses a MCSA configuration in its design.

5.1.2 Cross Layer Data Collection

The cross layer data collection module collects data from the network, MAC and physical layers. At each layer, the collection module monitors the events and computes time, traffic and topology statistics and records the feature values. The complete list of cross layer features is detailed in Appendix A.

As stated earlier, CRADS uses a hybrid detection strategies to exploit the high detection rate of anomaly detection and low false alarm rate of misbehavior detection. Therefore, the training data consists of both benign and malicious behavioral patterns.

5.1.3 Data Reduction Techniques

Three techniques are used for data reduction in CRADS; association, filtering and sampling. Association is a process in which features from different layers are correlated to form a reduced feature set. Filtering is a process in which redundant and uninformative cases are removed from the training data. Filtering is aided by information from other layers. Sampling is a process in which a subset of the originally associated and filtered training data is chosen. Each of these sub-modules will be examined in detail below.

5.1.3.1 Association

A large feature set can render machine learning techniques computationally infeasible. The authors of [59] used 141 cross layer features for their detection system. A feature set with many features will be impossible to be used for dynamic training in mobile nodes. On the other hand, reducing the feature set with methods like feature selection/ranking will reduce the detection accuracy.

An association process reduces the feature set, such that the overhead of learning is minimized. This is done by correlating one or more features from different layers to a specific MAC layer feature. The output of this process yields a reduced set of derived features which preserves the information content of the correlated features. First, the features are classified based on dependency on time, traffic and topology. Under each classification, features are correlated using a pre-defined correlation function on the features. For example, we define a derived feature as follows:

$$D_{Neighbors} \Rightarrow$$

$$\Rightarrow \begin{cases} = (|N_{MAC} - N_{NET}|) \frac{(T_{MAC} - T_{NET})^2}{T_{MAC} + T_{NET}} & \text{If } T_{MAC} \neq T_{NET} \\ = (|N_{MAC} - N_{NET}|) \frac{1}{T_{MAC}} & \text{If } T_{MAC} = T_{NET} \\ = 0 & \text{If } T_{MAC} = 0 \end{cases} \quad (5.1)$$

$$D_{Neighbors} \Rightarrow \begin{cases} = 0 & \Rightarrow \text{Benign} \\ \neq 0 & \Rightarrow \text{Suspicious} \end{cases}$$

The above expression (5.1) computes the derived feature $D_{Neighbors}$ using four independent cross layer features. The independent features include, N_{MAC} - number of neighbors identified at MAC layer, N_{NET} - number of neighbors found by the routing protocol, T_{MAC} - number of recent MAC receptions, T_{NET} - number of recent routing protocol updates from neighbors. The number of MAC neighbors N_{MAC} can be found by monitoring the MAC layer traffic and observing unique MAC addresses in the incoming traffic.

The above derived feature utilizes the fact that, if $|N_{MAC} - N_{NET}|$ is non-zero, then the node is likely to be under attack. However, due to mobility of nodes, there is a possibility of $|N_{MAC} - N_{NET}|$ to be non-zero even under benign conditions. Under mobility, reasonable traffic density is required for N_{MAC} to be accurate. To rectify the above uncertainty of N_{MAC} , two features T_{MAC} and T_{NET} are used, which help to compute the traffic density in the neighborhood as in (5.1). The computed ratio, which represents the traffic density, is used to scale the discrepancy observed between the MAC and routing layer. If the data traffic density ($T_{MAC} - T_{NET}$) is low, then due to the inaccuracy of N_{MAC} , the discrepancy found in $|N_{MAC} - N_{NET}|$ becomes unreliable. Hence, $D_{Neighbors}$ is reduced with respect to traffic density. On the other hand, if the data traffic density is high, $|N_{MAC} - N_{NET}|$ more importance is given and scaled to a higher value.

Thus, if $D_{Neighbors}$ is zero then the two features, N_{MAC} and N_{NET} have no dis-

CHAPTER 5. CRADS

Figure 5.2: Filtering Process using FDA

crepancies between layers and raise no suspicion of malicious activity. However, if the $D_{Neighbors}$ is non-zero then $D_{Neighbors}$ represent the degree of discrepancy between layers and also the extent of suspicion of malicious activity in the neighborhood. Similar to expression (5.1), various correlation functions are used for deriving other derived features. A complete list of the original cross layer features and the derived features are given in Appendix A. As stated earlier, the set of correlation functions are pre-determined considering the nature and origin of the features. In CRADS, the features from the MAC layer are associated with other layer features. The association results in 18 derived features, using features from the MAC, routing and physical layer protocols.

5.1.3.2 Feedback-based Filtering

Filtering refers to the removal of uninformative and redundant routing patterns. Events which are not essential in defining or classifying a routing behavior are referred to as uninformative events. Events that are similar to other events are redundant events. Events are filtered based on the information content of events. From the perspective of SVM, events that are closer to the decision boundary separating the benign and malicious events have more information content rather than events that lie far from the decision boundary. Hence, the above property of events is used to rank the information content and filter events with negligible information content.

To measure the information content in routing patterns, a fast linear classification algorithm, namely, FDA is used. FDA uses the ratio between class and within class variance as a measure to find a linear projection in which the events in feature space are best separated. FDA's decision boundary is expressed by the linear function $w^T \dot{x} + w_0$, where w is obtained from the FDA algorithm and w_0 is found using a gradient descent algorithm [31]. To obtain w , the FDA algorithm solves equation 5.2 [31].

$$J_w = \frac{w^T \cdot S_B \cdot w}{w^T \cdot S_W \cdot w} \quad (5.2)$$

J_w in equation 5.2 is well known in mathematical physics as the generalized Rayleigh quotient, where S_B is the between-class scatter matrix and S_W is the within-class scatter matrix. The FDA algorithm finds a hyper-plane (w) which maximizes the quotient J_w . It should be noted that the scatter matrix is proportional to the covariance matrix. Hence, the FDA algorithm literally finds a direction (w) where the between-class covariance is maximum and within-class covariance is minimum. In other words, a vector direction at which the benign events are best separated from malicious events.

$$x_i \in M \quad \forall \quad y_i (w^T x_i + w_0) \leq 0 \quad (5.3)$$

$$r_i = \frac{w^T x_i + w_0}{\|w\|} \quad (5.4)$$

$$R^+ = \max(r_i) \forall i \in M^+ \quad (5.5)$$

$$R^- = \max(r_i) \forall i \in M^- \quad (5.6)$$

$$B^+(x_i) = w^T x_i + w_0 + R^+ \|w\| \quad (5.7)$$

$$B^-(x_i) = w^T x_i + w_0 + R^- \|w\| \quad (5.8)$$

After finding the FDA hyper-plane, CRADS defines two misclassification boundaries which are found by computing the maximum distances of misclassified vectors to hyper-plane, as shown in figure 5.2. As $y_i (w^T x_i + w_0)$ is non-positive for misclassifications, the condition $y_i (w^T x_i + w_0) \leq 0$ is used to determine the misclassified vectors by the FDA decision boundary. The misclassified vector set M is determined using equation 5.3. The distances (r_i) to the FDA hyper-plane are computed using the expression in (5.4) for all vectors in the set M , which consists of misclassified vector indices. The maximum distance of misclassified benign events (M^+) to the hyper-plane is computed using (5.5), and similarly the maximum distance from the hyper-plane to misclassified malicious events (M^-) is computed using (5.6). Thus, the benign and malicious misclassification boundaries are defined using the linear hyper-planes expressed in (5.7) and (5.8), respectively. Events that are not bounded by these boundaries are filtered out from the training dataset.

The next step of filtering is removing redundancy. Redundant cases are events in the training data, which are similar. Similarity is measured by computing the Euclidean distance between the events in the feature space. For identical events, the Euclidean distance is zero. However, to identify similar cases, a predetermined threshold for the degree of similarity is used, rather than looking for complete similarity. This threshold is dynamically fine-tuned based on the feedback received from

the learning process.

5.1.3.3 Feedback-based Sampling

Sampling is the process of selecting a subset of the original training data. Even after association and filtering, the volume of training data remains large enough to impose a huge computational overhead for non-linear machine learning techniques. For the duration of 48 hours, the training cases exceeded a million cases (36 MB of data). Therefore, the density of events in the feature space is reduced using sampling.

Sampling is done by using random density reduction method. Similar to filtering, the actual percentage of data to be sampled is dynamically tuned to suit the network environment. The sampling interval is increased until the reduced dataset represents 5% - 9% of the original dataset. The feedback mechanism is used to tune the sampling interval between 5% and 9% of the original training dataset.

Thus, during the simulations, only 5% - 9% of the original training data (without data reduction) is selected as the sampled training data. This raises the possibility of the output model to be under-fitted. Under-fitting [31] is a phenomenon in which the detection model is poorly defined due to the reduction of training data. Under-fitting arises when the reduced training dataset inadequately expresses the original dataset. This issue is dealt with by the validation process, which will be discussed in a later section.

5.1.4 Learning

The CRADS's learning model is essentially a non-linear SVM classification model [20]. It is trained by the SVM algorithm using the reduced training dataset.

5.1.4.1 Support Vector Machines

As explained section 2.3.1.2, machine learning is a process in which a set of threshold parameters are trained to classify an unknown behavior. SVM is a non-linear machine learning algorithm.

In SVM, the input vectors or features of the training data are first mapped into a very high dimensional feature space, using a non-linear mapping function chosen a priori. Now, in the mapped feature space SVM constructs a linear decision function as in (Figure 2.1) which separates the two classes of events, namely benign and malicious.

Unlike the linear machine, SVM does not train the decision function. Literally, SVM searches for a decision function with maximum margins. Margin is represented by the void area in feature space around the decision function, as shown in figure 5.3. In other words, SVM constructs the decision function along the direction in which the benign and malicious events are best separated. The margin width of the decision function is $(2/\|w\|)$ as shown in figure 5.3. Therefore, SVM uses Lagrangian multipliers to minimize $\|w\|$ in the above ratio to obtain maximum margins.

The constructed linear decision function in the feature space is actually a non-linear decision function separating the events in original unmapped input space. This phenomenon is primarily due to the non-linear mapping of the input space to feature space. The vectors falling in the margin areas of the feature space are referred to as the support vectors. These support vectors govern the non-linear boundaries of SVM classification model. The prime advantage of SVM is in its non-linearity. It is a well-known fact that non-linear classification provides better classification accuracy than linear classification.

The detection accuracy of SVM depends primarily on the direction of margins, which is optimized for maximal between-class variance. However, in CRADS, the training data undergoes a vigorous data reduction process, which may affect the actual variance of events and thus the margins derived may not be in the direction

of an optimal class variance. To overcome the issue, the detection model trained by SVM has to be validated. The process of validating the detection model is discussed in the next section.

5.1.5 Validation

Due to data reduction, validation of training dataset adequacy is essential. For validation, FDA is used. The prime advantage of choosing FDA over other Discriminant methods is that the direction $\vec{w}$ of linear hyper-plane using FDA is similar to the direction of margins $\vec{w}$ in SVM. This is primarily because of the filtering of events using the FDA's linear hyper-plane in FDA's projection space. Additionally, SVM constructs the margin under the same criteria as FDA, the between-class separability (variance). SVM aims to maximize the margin area $2/\|w\|$, whereas, FDA maximizes between-class variance V , both attains the same objective. Hence,

$$\left(\max \frac{2}{\|w\|}\right) \propto V \quad (5.9)$$

It should however be noted here that *SVM and FDA decision boundaries are not similar*.

As the training data is empirical, the actual probability densities of events are never known. Unlike normality-based or Bayesian based discriminant methods, FDA does not rely on actual a priori probabilities. Also, no assumption is made on the normality of the data. This makes FDA the best choice for the linear classifier. Furthermore, FDA's other unique property is transformation. The feature space is transformed for better segregation of events.

The purpose of validation is to check the adequacy of the sampled training data in SVM. The validation process consists of two stages, as shown in figure 5.1. First, the direction of SVM margins and the FDA's linear hyper-plane are compared and

validated. Secondly, the training accuracies of SVM and FDA are compared to check the adequacy of the SVM's classification model.

In the first stage of validation, as stated earlier, the directions of SVM margins and FDA's linear projection are compared. Furthermore, in SVM the decision boundary is defined using support vectors. To find the linear direction and orientation of the SVM margins, a linear equation is trained on the support vectors. The direction and orientation of the derived linear equation is compared with the FDA's linear decision boundary. If the comparison results in a significant difference, the redundancy threshold in filtering and sampling rate is tuned and fresh training sample is chosen.

The second stage of the validation inspects for under-fitting in the classification model. Under-fitting is a phenomenon, where the training set is so poor that the resulting model inadequately represents the actual detection model and will yield poor detection accuracy. Combining multiple classifiers is a well known method to overcome under-fitting. The training accuracy of SVM and FDA is compared to evaluate the adequacy of the SVM's training data and the SVM classification model. If substantial difference in training accuracy is found, then the sampling rate and distance threshold TD are re-tuned and a fresh sample is chosen. In this case, in the next consecutive training run, the support vectors found in the previous training are added to the training data. Through the above process, the large training dataset is decomposed and SVM model is trained until it reaches adequacy.

5.1.6 Detection

After training the IDS, it can be used for real-time detection of malicious behavior. The proposed IDS is an autonomous host-based detection system. Hence, monitoring neighbor node routing behavior is done by the nodes themselves by promiscuously listening to the neighborhood communications. Detection of an attack is also performed locally by the nodes.

CHAPTER 5. CRADS

Given an unknown behavior represented by vector x , CRADS simply determine on which side of the decision boundary the vector x falls. This is computed by the sign of function in equation 5.10.

$$f(x) = \sum_{i=1}^{N_S} \alpha_i y_i K(s_i, x) + w_0 \quad (5.10)$$

In equation 5.10, α_i is the Lagrangian multiplier used for minimizing $\|w\|$, which in-turn maximizes the margins. N_S is the total number of support vectors identified in the SVM training process, s_i is the i th support vector and $y_i \in \{\pm 1\}$ depending on the behavior class of s_i . The function $K(s_i, x)$ gives the inner product of the vectors s_i and x represented in the high dimensional feature space. This function is commonly referred to as the kernel function. CRADS uses Radial Basis Function (RBF) [31] as the kernel function.

5.1.7 Complexity

The computational complexity C [20] of the SVM training process is in range expressed in equation 5.11 and 5.12.

$$(N_S^2 + N_S l + N_S d_L l) \leq C \leq (N_S^3 + N_S^2 l + N_S d_L l) \quad (5.11)$$

$$d_L l^2 \leq C \leq (N_S^2 + N_S d_L l) \quad (5.12)$$

Where N_S is the total number of support vectors, l is the number of training points and d_L is dimension of the input data. If most support vectors lie on one of the margins, then the complexity ranges as in equation 5.12. The computational overhead C reaches the lower bound of the equation 5.11 and 5.12 as the ratio $N_S/l \approx 1$ and reaches the upper bound if the ratio $N_S/l \ll 1$.

CHAPTER 5. CRADS

Simulator	Qualnet 3.9.5
Simulation Landscape	1500m x 1000m
Node Transmission Range	250m
Simulation Time	24 hours
Minimum Network Size	30 nodes
Maximum Network Size	50 nodes
Node Join/leave events	Uniformly distributed
Mobility Models	Random-waypoint (0-30 m/s) Vehicular (40 m/s)
Routing Protocol	OLSR
OLSR Hello Message Interval	2 sec
MAC Protocol	802.11b
Background Data traffic	CBR/VBR

Table 5.1: Attack Simulation Setup

The complexity C of the detection of malicious behavior is in the order of $O(MN_S)$. M is complexity of the kernel function, which maps the input space to the high dimensional feature space. Since our model uses RBF [31] kernels, the complexity of M is in the order of $O(d_L)$. Therefore, the detection complexity C is computed using $O(N_S d_L)$. The effect of data reduction on these parameters is discussed in section 5.3.1.

5.2 Simulation Setup

The network simulations are done using Qualnet 3.9 [2] network simulator. Various routing attack scenarios are simulated over OLSR. The routing attack scenarios include, sinking, spoofing and fabrication. For each type of attack scenario, different network conditions and attack behavior were experimented.

LIBSVM [24] is used for SVM implementation and SPSS [88] is used for FDA implementation. LIBSVM is a set of libraries with the C++ implementation of the SVM algorithm. LISVM uses SMO decomposition method to heuristically solve the Lagrangian optimization of SVM. Eventhough CRADS is designed for real-time de-

tection, the experiments were conducted offline. That is, actual detection is done using the recorded behavior rather than real-time detection of observed behavior. Primarily, this is due to limitations on processing capabilities of the simulator.

For each attack scenario, five individual runs with different network conditions are performed. Each run has the duration of 24 hours of simulation time, during which both benign and malicious behavior is recorded. Therefore, each scenario is simulated for 120 hours. The simulation consists of 30 to 50 nodes running OLSR. The mobility model used was random-waypoint and in certain scenarios, i.e., vehicular mobility. All random parameters are assigned values using normal distribution. Random applications are simulated for creating the background traffic. The nodes join and leave the network dynamically. This is to ensure that the node density of the neighborhood changes over time.

Every routing attack lasts for 900 seconds. During these 900 seconds, a randomly chosen malicious node attack a randomly chosen victim node. The malicious node is assumed to possess no intelligence to enable it to evade detection using reactive means, however, it is assumed to change its parameters randomly to avoid detection. Although the malicious node dynamically changes its malicious behavioral parameters, it persistently attacks the victim node. Its target and goal never change during the 900 seconds. After 900 seconds, a new pair of randomly chosen malicious and victim node is selected, and the scenario is repeated with random parameters. The process is repeated throughout the total simulation period and the observations are recorded.

5.2.1 Sinking Attack Scenarios

In an ad hoc network, there are many factors that contribute to camouflaging the malicious sinking behavior. These factors are the different characteristics of the network, which may cause nodes in the network to benignly drop packets. This kind of

CHAPTER 5. CRADS

Characteristics → Scenarios↓	Mobility (m/s)	Source Type	Network Traffic Density	Attack Duration	Drop Selectivity Ratio %
Mobility Scenarios					
No mobility	0	Constant Bit Rate	≈60%	900 sec	100% Drop
5 m/s mobility	5	Constant Bit Rate	≈60%	900 sec	100% Drop
10 m/s mobility	10	Constant Bit Rate	≈60%	900 sec	100% Drop
30 m/s mobility	30	Constant Bit Rate	≈60%	900 sec	100% Drop
Vehicular mobility	>40	Constant Bit Rate	≈60%	900 sec	100% Drop
Traffic Density Scenarios					
Sparse Traffic Density	40	Random Variable Bit Rate	20%	900 sec	100% Drop
Medium Traffic Density	40	Random Variable Bit Rate	35%	900 sec	100% Drop
Dense Traffic Density	40	Random Variable Bit Rate	>70%	900 sec	100% Drop
Drop Selectivity Scenarios					
30% Drop Ratio	40	Random Variable Bit Rate	≈60%	900 sec	30% Drop
50% Drop Ratio	40	Random Variable Bit Rate	≈60%	900 sec	50% Drop
70% Drop Ratio	40	Random Variable Bit Rate	≈60%	900 sec	70% Drop
90% Drop Ratio	40	Random Variable Bit Rate	≈60%	900 sec	90% Drop

Table 5.2: Sinking Attack Scenarios

packet dropping behavior due to network conditions resembles the behavior of malicious sinking. Therefore, the goal of CRADS is to distinguish benign packet dropping induced by network conditions from malicious sinking behavior.

Possible factors which can induce benign packet dropping include;

- Mobility of nodes
- Network/Traffic Density

Mobility causes frequent change in topology resulting in packet drop, as re-establishing a new route takes some time. Furthermore, mobility induces channel fading conditions and changing interference levels, which can also cause packet drop.

Network and traffic density are important factors. Network density is the ratio between number of nodes in the network and the network area. Traffic density is the average number of nodes transmitting data over the network. Network density and

traffic density are relatively correlated. These factors determine the density of background network traffic, which will aid in camouflaging malicious sinking behaviors. As the attacker will sink only packets destined for the targeted victim node, high traffic in the network will make the attacker's sinking behavior to go undetected. This is due to the fact that the attacker will faithfully forward other packets which are not destined for the targeted victim node. A steady traffic towards the sinker is on the other hand also necessary to detect malicious behavior. If the traffic towards the sinker is sporadic, the observable period over the node's behavior reduces. This affects detection efficiency.

In addition to the above network environment factors, the behavior of the sinker also plays a vital role in the detection efficiency. The sinker can selectively drop critical data/routing packets and forward some percentage of in-bound traffic benignly.

The above three factors, mobility, traffic density and packet drop ratio are varied in the simulation. As listed in table 5.1, there are 12 sinking attack scenarios; 5 emulate different mobility conditions; 3 with varied traffic density conditions; 4 with different packet drop selectivity ratio conditions.

5.2.2 Spoofing/Fabrication Attack Scenarios

Detection of spoofing attacks in non-mobile network is trivial, as the topology never changes and routes are fixed. Hence, detection accuracy of spoofing behavior is primarily affected by changes in network environment, which is mostly induced by mobility. Therefore, different mobility scenarios are simulated, ranging from no-mobility to vehicular mobility. The experimented spoofing based routing attacks include, route invasion, disruption and denial of service. These attacks comprises of route addition, modification or deletion in a particular sequence or pattern. In OLSR, the above three attacks can be executed, by modifying or generating routing messages. In the experiments, Hello and TC routing events are used for the attacks.

Five mobility scenarios for each class of attack are experimented. The five mobility scenarios include, zero mobility, 5 m/s mobility, 10 m/s mobility, 30 m/s mobility and vehicular mobility. Hence, 20 scenarios with different environments and attacks are experimented.

5.2.2.1 Malicious behavior involving Hello Message

A malicious node can modify its own Hello packet or generate a new Hello packet. Modification/Fabrication of its own Hello packet aims to cast false information about its own neighborhood. The generation of a new Hello packet aims to impersonate some other victim node in the network. Therefore, the Hello message attacks can be categorized as;

- Hello Modification (HELMOD)
- Hello Generation (HELGEN)

It is interesting to observe that for HELGEN attacks, even though it is possible to detect the malicious behavior, it is never possible for a node to obtain the identity of the attacker. As both IP address and MAC address can be spoofed, the attacker will never disclose its true identity. This makes HELGEN attacks a highly potential threat for ad hoc networks.

5.2.2.2 Malicious behavior involving Topology Control Message

In OLSR, a MPR node is selected by the node to forward data to the next hop. From the perspective of an MPR node, the neighbor node, which has selected it as an MPR, is an Multi-Point Relay Selector (MPRS). The objective function of TC routing message is to declare MPRS of its neighborhood. It is evident that malicious behavior over TC routing message will lead to serious damage. Corrupted topology information can be broadcasted using malicious TC messages. The data forwarding

patterns can be changed by a simple declaration in the TC message stating that a particular node (victim) is MPRS. This will allow the attacker to invade all routes leading to the victim node. Route invasion becomes highly trivial with a simple spoofing of the TC message.

Similar to the Hello message, a TC message can be modified or generated. Therefore, TC-based attacks can be classified as;

- TC Modification (TCMOD)
- TC Generation (TCGEN)

The security threat is serious when the attacker modifies or generates TC routing messages. This is due to the fact that, HELMOD and HELGEN affect only the immediate neighborhood, TCMOD and TCGEN affect the entire network. This is because of the OLSR's functionality of propagating TC routing information to all nodes in the network. The extent of damage that can be caused by TCMOD and TCGEN cannot be defined. Similar to HELGEN, TCGEN attackers are anonymous.

5.3 Results and Analysis

In this section, CRADS is evaluated for its performance and efficiency. CRADS is assessed using two parameters, namely, detection accuracy and computational overhead [61]. Also the analysis compares CRADS with conventional single layer based detection strategies. As discussed earlier and shown in figure 2.3, detection accuracy is quantified using two metrics, true positives and true negatives. True positives and true negatives are the two primary metrics used to assess the detection accuracy of CRADS. The second parameter used to assess CRADS is computational overhead. This is measured by computing the processing time incurred to build the detection model by CRADS.

Figure 5.3: Data Reduction

5.3.1 Efficiency of Data Reduction

The parameters in equations 5.11 and 5.12 control the computational complexity of the SVM algorithm. The data reduction techniques aim to minimize these parameters. Figure 5.3 illustrates the effect at each stage in the data reduction on the training dataset and the parameters in SVM computational complexity.

Association reduces the dimensionality d_L of input data and filtering reduces the number of training points l as well as the support vectors N_S . The input data dimension d_L is reduced by the process of association. It can be observed from figure 5.3 that 60% of the original dataset is reduced through the association process. The reduced training data consists of 18 dimensions or features. However, the number of training vectors remains the same. In the first stage of filtering, where events are filtered based on information content, it can be observed from figure 5.3 that 40% of

CHAPTER 5. CRADS

the training samples are removed using FDA based filtering. However, it can be noted that the number of support vectors remain the same during this process. Thus, the eliminated vectors are non-support vectors. In the redundancy removal, the density of the events in the feature space is reduced affecting both the number of training vectors and support vectors. The density of support vectors and training points are further reduced by sampling. It can be observed from figure 5.3 that the number of training samples and number of support vectors almost converge. Hence, this shows that reduced training dataset consists mostly of support vectors. Furthermore, data reduction schemes reduce the training data to $\approx 5\%$ of the original dataset. This proves that CRADS's data reduction significantly reduces the training data without losing the information content represented by the original feature space.

Figure 5.4: Detection Accuracy with Changing Mobility

From equation 5.12, it can be deduced that the number of support vectors is a prime factor for training the CRADS. In our experiments, it was found that the number of support vectors were in the range of $150 < N_S < 500$. Without training data reduction, the support vectors N_S exceeds 5000, which makes data reduction an essential component for feasibility of SVM in mobile nodes. Thus, detection using the CRADS is made feasible in mobile nodes using the proposed data reduction methods.

The detection accuracy also experienced only a negligible decrease which will be illustrated in section 5.3.2.

5.3.2 Detection of Sinking Attacks

The detection accuracy in the detection of sinking attacks is analyzed and compared with single layer based detection strategies and linear machine learning detection strategies. CRADS is compared with single layer SVM based detection technique and cross layer based linear (FDA) machine learning based detection. Single layer based SVM uses the MAC protocol statistics to detect malicious behavior. Cross layer FDA uses all features that CRADS uses to detect malicious behavior. However, Cross layer FDA does not use the association function to combine features.

5.3.2.1 Effect of Mobility

The node mobility is varied from no mobility (stationary) to vehicular mobility. It can be observed from figure 5.4, that in single layer methods, detection efficiency drops as mobility increases. However, at mobility $\geq 30m/s$, the detection efficiency increases. This due to the fact that the simulation scenarios with mobility $\geq 30m/s$, use vehicular mobility model (VMM). Unlike random waypoint model, node mobility is more predictable in VMM due to their restricted degrees of freedom in node movement direction. In random waypoint model, node move in all 360° of the 2D plane, whereas, nodes with vehicular mobility (VMM) move with 8° of freedom. This property of VMM increases the predictability of node movement, which increases the detection accuracy as shown in the figure 5.4. Interestingly, the detection efficiency of cross layer methods is not affected by mobility. This is further shown using ROC curves in figure 5.5, where the area under curve (AUC) is lower for single layer methods.

Also worth noting is the fact that, even though cross layer SVM always performs better than cross layer FDA for varying mobility, the difference in efficiency between

CHAPTER 5. CRADS

Figure 5.5: ROC Curves with Changing Mobility

the methods is negligible. Both FDA and SVM provide almost the same percentage of true negatives and true positives.

CHAPTER 5. CRADS

Figure 5.6: Detection Accuracy with Changing Traffic Density

Figure 5.7: ROC Curves with Changing Traffic Density

CHAPTER 5. CRADS

5.3.2.2 Effect of Traffic Density

Traffic density significantly affects the single layer detection method, which is shown in figure 5.6. Traffic density (T_D) is the average percentage of node transmitting data during the duration of an attack. Three scenarios which vary traffic density conditions with 35%, 50% and $T_D > 70\%$ are evaluated. For example, with 35% traffic density, 35% of the nodes in the network transmit data during the attack. All three methods give efficient true negative detection rate. However, the detection rate of malicious behavior (true positives) using single layer SVM is very poor. But it should be noted that as the traffic density increases beyond a threshold (around 45% of traffic density), single layer SVM efficiency increases linearly. This is due to higher incoming traffic created by the increase in traffic density towards the sinker. In this scenario, the sinker drops more packets which increase the detection accuracy of single layer SVM. However, the rate of increase of detection accuracy is not significant and single layer SVM still suffers from poor true positives.

Figure 5.8: Detection Accuracy with Changing Packet Drop Ratio

It can be observed from figure 5.6 that cross layer approach with SVM or FDA is marginally affected by traffic density. To prove this further, the ROC curves in figure 5.7 show that traffic density does not significantly affect the cross layer approach.

Detection accuracy of single layer scheme is on the other hand poor when the traffic is less dense or highly dense. Single layer SVM performs optimal only when the traffic density is $\approx 45\%$. It should also be noted based on figure 5.6 that there is negligible drop in efficiency in the cross layer approach as the traffic density increases. This is primarily due to the background noise (traffic) created by the density of the traffic. The increase in the background traffic helps the attacker to hide the sinking behavior. Even though the detection efficiency drops, the cross layer SVM still outperforms the single layer SVM and the cross layer FDA. The rate of decrease in detection accuracy is negligible. Hence, the number of percentage of packets dropped by the attacker with respect to the current network traffic does not affect the detection accuracy of cross layer SVM.

5.3.2.3 Effect of Packet Drop Ratio

Similar to traffic density, selectivity of packet dropping affects the efficiency of the single layer methods more than the cross layer methods. The cross layer methods experience a small drop in the efficiency, as the sinker drops less percentage of packets. Four scenarios simulating packet dropping percentage varying from 30% to 90% are studied. As the sinker drops a higher percentage of the packets, the detection rate increases for the single layer SVM and the cross layer FDA. However, the cross layer SVM is un-affected by the changes in the drop ratio, as shown in figure 5.8. This property of the cross layer SVM will enable the detection of intelligent attacks by the sinker, which aims to thwart detection.

In figure 5.8, difference of efficiency between the cross layer SVM and the cross layer FDA is constant. However, SVM detects approximately 10% more true positives than FDA, which is significant. In addition, SVM provides $< 1\%$ false alarm rate in all scenarios. This fact is further strengthened by observing the ROC curves as shown in figure 5.9. As the sinker drops less number of packets, the single layer approach

CHAPTER 5. CRADS

Figure 5.9: ROC Curves with Changing Packet Drop Ratio

almost reaches efficiency similar to random guessing. On the contrary, the cross layer schemes are unaffected by this factors.

Considering all the scenarios, from the results, it is evident that the proposed IDS model performs better than the conventional linear models and the single layer approaches. In most cases, the cross layer SVM scheme efficiency is equal or higher compared to the cross layer FDA. SVM's ability to thwart intelligent behavior of a sinker is crucial and vantage point for the IDS.

CHAPTER 5. CRADS

Figure 5.10: Spoofing Attack Detection: True Positives

5.3.3 Detection of Spoofing Attacks

Figure 5.10 shows the true positives for the four classes of spoofing attacks for different detection schemes and cross layer detection. Figure 5.11 shows the corresponding results for true negatives. It is evident from the graphs that the classification accuracy

CHAPTER 5. CRADS

Figure 5.11: Spoofing Attack Detection: True Negatives

of the CRADS far exceeds the single layer schemes.

It is interesting to observe in figure 5.11 that the detection efficiency of all schemes significantly rise when the mobility is more than 30 m/s. This is because, at 40 m/s the scenarios use the vehicular mobility model (VMM), which is much more pre-

CHAPTER 5. CRADS

dictable than the random waypoint model. However, in scenarios using VMM, single layer systems compromise true positives for increasing the true negatives. For instance, routing and MAC layers compromise true positives for achieving better true negatives which is observed by comparing figures 5.11a and 5.11b. On the other hand, cross layer schemes do not exhibit this phenomenon.

Publications:

- J.F.C, Joseph, Bu-Sung Lee, Amitabha Das, Seet Boot Chong, "*Cross Layer Detection of Sinking Behavior in Wireless Ad hoc Networks using SVM and FDA*", IEEE Transactions on Dependable and Secure Computing, *Accepted*.
- J.F.C, Joseph, Amitabha Das, Seet Boot Chong, Bu-Sung Lee, "*CRADS: Integrated Cross Layer approach for Detecting Routing Attacks in MANETs*", IEEE Wireless Communications and Networking Conference (WCNC) 2008. Las Vegas.
- J.F.C, Joseph, Amitabha Das, Seet Boot Chong, Bu-Sung Lee, "*Cross Layer versus Single Layer Approaches for Intrusion Detection in MANETs*", International Conference on Networking (ICON) 2007, Adelaide.

Chapter 6

CARRADS

In this chapter, CARRADS, the real-time adaptation module for the security architecture is presented. CARRADS adapts the CRADS's trained SVM detection model at real-time with changing network conditions and new routing attack patterns. The design objective of CARRADS is to minimize computational overhead during the adaptation process without affecting detection accuracy.

6.1 Need for Adaptive RADS

Due to the mobility of nodes and changing network environment, the benignity of routing behavioral patterns is highly transient. To illustrate this phenomenon, consider a simple scenario. A node M maliciously generates a spurious routing update stating that it has connectivity to a certain number of nodes and it broadcast this information to its neighborhood nodes. In this case, the routing update is a malicious behavior exhibited by the node M, since it does not have the stated connectivity. Now, let us assume that the RADS uses a naive threshold based classifier, which uses a fixed threshold value (d) to detect malicious behavior. Optimal value of classifier's threshold value (d) is found during the initial training of the detection model.

CHAPTER 6. CARRADS

Figure 6.1: CARRADS Design

The classifier identifies the behavior as malicious if the number of node connectivity declared in a routing update is more than a particular threshold (d).

Due to an increase in the density of nodes around the neighborhood, there is a possibility in the future that node M gets the connectivity to the number of nodes stated in the previously malicious routing update. In this scenario, the previous malicious routing update now becomes a benign routing behavior. However, the fixed threshold classifier will fail to identify the environment change and incorrectly detects the behavior as malicious. The trained fixed threshold classifier (d) will become sup-optimal for the new network environment and eventually, the RADS will yield very poor detection accuracy. Similar to the above phenomenon, there is also the possibility of benign behavior changing into malicious activity. For the above scenario, consider if the nodes in the neighborhood gets sparse. In this case, the threshold (d) will be sup-optimal, as the optimal value is less than the current fixed threshold (d).

The changing network environment also makes the benign routing behaviors be-

CHAPTER 6. CARRADS

come obsolete over time. This is primarily because the statistical definition of benignity in the detection model changes when the network environment changes. This raises the need for periodic purging of old obsolete behavior patterns. Additionally, old benign patterns can undermine the influence of newly identified malicious patterns. If a new malicious pattern highly resembles old benign patterns, the new attack will be hidden by the sheer number of old benign patterns resembling that attack. This will result in RADS failing to detect the new attack. Thus, it is important to periodically update the RADS detection model in consideration with the changing network conditions.

There are numerous factors that cause the network environment to change. However, mobility and density of nodes in the network are the primary factors, which affect the RADS detection accuracy significantly.

6.2 CARRADS

Figure 6.1 shows the design of CARRADS. As shown in figure 6.1, the runtime modules in CARRADS include a Data Collection Module, a Behavior Identification module, a Dataset Reduction module and finally, a Learning module. Each of these modules is detailed in the succeeding sections.

There are two primary design objectives for CARRADS. The first objective is to identify beneficial 'new' routing behavior(s) inside the routing control traffic that has significant information content. The second objective is to retrain the detection model to add/remove new routing patterns with minimal processing overhead and superior detection accuracy.

In CARRADS, the Data Collection module is responsible for collecting statistical information for defining a routing pattern from different protocols at real-time. Identifying valuable new routing behavior patterns is done by the BIM. The module is also

CHAPTER 6. CARRADS

responsible for distinguishing between occurrence of new patterns and network environment change. New patterns are identified using cooperative decision support from the neighbors. The Data reduction module reduces the number of features defining a routing pattern using association.

The learning module is the prime component of CARRADS, which adds or removes patterns to/from the detection model. The conventional method of adding new patterns is to retrain the detection model from scratch by including the new patterns in the new dataset. However, this method is not feasible in resource constrained ad hoc network nodes. Hence, CARRADS exploits the intrinsic properties of the SVM algorithm to reduce the computational overhead while retraining the detection model.

Before the runtime deployment, as shown in figure 6.1, CARRADS is bootstrapped using a initial detection model that was built using CRADS, as explained in chapter 5. At real-time, CARRADS adapts this initial detection model dynamically, to suit changing network conditions and occurrence of new routing patterns. Since, CARRADS uses CRADS to build its initial detection model, it is essential to preserve the design between the two systems for compatibility reasons. Hence, as shown in figure 6.1, architecturally, the components of CARRADS have some similarity to those of CRADS. However, the integral methods and functions of these components differ significantly between CARRADS and CRADS.

6.2.1 Data Collection Module

At run-time, CARRADS monitors the neighborhood and extracts behavior information from physical, MAC and network layer protocols. Similar to CRADS, 46 cross layer features are monitored and recorded to define a routing behavior pattern. At each layer, the data collection module monitors the events and computes the traffic, time and topology parameters and records the values to define a routing behavior.

6.2.2 Behavior Identification Module

This module is responsible for monitoring, identifying and classifying a new pattern observed in the neighborhood. The module comprises three sub-modules; Behavior Eccentricity Estimation, Environment Sensing and Cooperative Global Decision. The neighborhood routing traffic occasionally contains unidentified patterns and acts as a source for new patterns.

The Behavior Eccentricity Estimation module examines these unidentified patterns and selects patterns that need inclusion into the detection model. With the help of the Cooperative Global Decision sub-module, the selected patterns are classified as an attack pattern or as benign behavior. Besides identifying and classifying new patterns, periodic purging of old obsolete patterns is performed on the detection model. The Environment Sensing sub-module is responsible for sensing changes in the network environment and will initiate a purge for removing obsolete benign routing patterns.

6.2.2.1 Behavior Eccentricity Estimation

To determine whether the observed behavior is new, the eccentricity of the behavior is evaluated in two stages. In the first stage, the similarity of the new vector to the vectors in current detection model is computed by $V_E = 1/l \sum_{i=1}^l \|v - x_i\|$, where v is the new vector and $x_{i=1...l}$ is the existing trained vectors in the current detection model. The expression $\|v - x\|$ in the above equation computes the Euclidean distance between the support vectors and the new vector. The V_E parameter is compared with a dynamic threshold $T_E = 1/(l^2 - l + 2) \sum_{i=1}^l \sum_{j=(i+1)}^l \|x_i - x_j\|$ [31], which is the average Euclidian distances between vectors in the current detection model. If the value of V_E is very large compared to T_E then the next stage of eccentricity estimation is performed on vector x , else the vector is not considered a new pattern.

In the next stage of eccentricity estimation, a global consensus among neighbor

CHAPTER 6. CARRADS

nodes on the classification of the new vector is determined by the Cooperative Global Decision support module. If the global classification result of the behavior is different from local result, then the observed pattern is considered new. The operation of this module will be discussed in the section 6.2.4.

Identification of behavior as '*new*' does not warrant an immediate incremental inclusion into the detection model. The new behavior pattern is prioritized by measuring the information content. For measuring the information content and prioritizing with respect to a learning algorithm, FDA is used.

For prioritizing new events, CARRADS utilizes a methodology similar to CRADS's FDA-based filtering process explained in section 5.2.3.2. CARRADS defines two misclassification boundaries. These misclassification boundaries are derived as discussed in CRADS's FDA based filtering process. The equations 5.7 and 5.8 define benign and malicious misclassification boundary.

The above FDA's misclassification boundaries are used to categorize a new routing pattern as high or low priority. If the routing pattern falls outside these boundaries, then the routing pattern is labeled as high priority. These routing patterns have high entropy and needs to be included in the detection model immediately. Observation of a new high priority routing pattern will initiate the adaptation process. The vectors which lie within the FDA's misclassification boundaries, are labeled as low priority. Low priority patterns are close to the current decision boundary and are already in the area of currently misclassified vectors. This means that inclusion of these vectors is not urgent. Low priority patterns are withheld and do not trigger an adaptation process. However, these patterns are included in the next adaptation process. The prioritization process is used to reduce the frequency of retraining at real-time.

6.2.3 Environment Sensing

Changes in the network environment and conditions will immediately spawn new benign patterns. Thus, a RADS that is passive to these changing conditions will become obsolete over time. This is primarily because, the old benign patterns is no longer valid in the new network environment. The new benign patterns will be detected most probably as malicious and will increase the false positives significantly. Hence, it is evident that periodic purging of old obsolete benign patterns is essential. However, this raises two important questions.

- How to sense environment changes?
- How to replace the old obsolete patterns?

Similar to the classification of benign and malicious patterns, CARRADS uses a set of dedicated multi-class classifiers to associate a benign routing pattern to different network conditions. The classifiers are statically trained offline using the FDA machine learning algorithm. CARRADS uses two multi-class classifiers for detecting change in two network conditions; mobility of nodes and neighborhood node density. Classifier for detecting mobility of nodes is trained using physical layer features and consists of five mobility classes. The classes include no mobility, 5 m/s mobility, 10 m/s mobility, 30 m/s mobility and vehicular mobility. Similarly, classifier for detecting neighborhood node density consists of three classes; sparse, moderate and dense. The node density classifier is trained with routing protocol features.

An environment change is sensed if consecutive patterns are classified to an environment other than the current environment. After a change in network environment is sensed, the old obsolete benign routing patterns are purged. To replace the old patterns, new benign patterns found using BIM are added to the detection model using the incremental learning of CARRADS.

Figure 6.2: Global Cooperative Decision Support

6.2.4 Cooperative Global Decision

To classify an unknown pattern, CARRADS solicits its neighbors for cooperative decision. A global consensus over the unknown behavior is reached. The consensus is valid until the assumption that the number of benign nodes are significantly more than the number of malicious nodes is held true. When a global decision solicitation is received, the node replies with distance of the unknown behavior (vector) to its decision boundary. The node which has solicited for global decision support gives higher weightage to information given by nodes in the immediate neighborhood. This is due to the fact that immediate neighborhood nodes are most probably under the same network conditions as the node. However, in the case of high mobility where the immediate neighborhood of the node is constantly changing, the above statement will not hold true. Nevertheless, in this scenario preference is given to random nodes, which may be beneficial but certainly not detrimental.

Solicitation of decision support from a node is done by broadcasting decision support request to the immediate neighbors. To illustrate this, consider a scenario as shown in figure 6.2 where a node A solicits decision support request to its immediate neighbors.

CHAPTER 6. CARRADS

$$NodeA \rightarrow N_{i=1...M} : ID_A || ID_{Beh} || B || D_A || MFD \quad (6.1)$$

As shown in equation 6.1, the contents of the solicitation request comprise of, the identifier of node A (ID_A), the identifier of the behavior (ID_{Beh}), the feature vector defining the routing pattern (B), the local detection engine's classification of the routing pattern (D_A) and finally, the Maximum Flooding Depth (MFD) is the maximum hops this decision support request will be propagated within the network. The parameter MFD is used primarily to control the flooding. The value of the parameter D_A is the distance between the decision boundary in the current detection engine and the vector representing the new routing behavior pattern. A positive value of the D_A means that the local detection engine classifies the new routing pattern as benign while a negative value means that the behavior is classified as malicious.

$$N_i \rightarrow NN_{i=1...M} : ID_N || ID_{Beh} || B || D_{N_i} || (MFD - 1) \quad (6.2)$$

A neighbor node N on receiving a decision support request checks the MFD in the request. If the MFD is greater than 1, then the neighbor node forwards the request to its neighbor node. The contents of the forwarding request is shown in equation 6.2, which includes, the identifier of the forwarding node (ID_N), the identifier of the behavior pattern (ID_{Beh} assigned by node A), the new routing behavior pattern (B), the classification of routing pattern by local detection engine D_{N_i} and finally the decremented MFD.

If the value of MFD is less than 1 in the request or the forwarded request, the neighbor node stops propagating the decision support request and replies to the request. As in equation 6.3, the decision support request reply consists of the identifier of the neighbor node NN, identifier of the behavior and local decision on the routing pattern.

CHAPTER 6. CARRADS

$$NN_{i=1\dots M} \rightarrow N_{i=1\dots M} : ID_{NN} || ID_{Beh} || D_{NN_i} \quad (6.3)$$

$$D_{AVG} = \frac{1}{M} \sum_{i=1}^M \frac{D_{NN_i}}{H_i} \quad (6.4)$$

$$D_{NN_i} = \sum_{j=1}^L y_j e^{-\|x-x_j\|/2\sigma} \quad (6.5)$$

On receiving the decision support request reply, the intermediate node N computes a weighted average decision D_{AVG} among M neighbor nodes (NN) using the expression in 6.4 and sends it to the source node A . In equation 6.4, the average consensus (D_{AVG}) is computed using the decision replies (D_{NN_i}) from the i^{th} solicited node. H_i is the distance between the nodes represented in number of hops, which is calculated from MFD. Equation 6.5 is used to compute D_{NN_i} , which is the non-linear kernel distance of the vector x from the decision boundary.

In CARRADS, the decision boundary is defined by the SVM algorithm using a set of support vectors [20]. The non-linear distance between a vector and the SVM decision boundary is computed using the set of support vectors, as shown in equation 6.5. In equation 6.5, L is the number of support vectors, x_i is the i^{th} support vector and x is the new vector for which the distance is being computed. σ is the variance of the support vectors and used as a sensitivity measure to normalize the distance metric. The concept of SVM and support vectors is explained in more detail in section 6.2.7.

The value of the parameter D_{NN_i} is contained in the neighbor nodes' replies to the cooperative support solicitation, which is shown in equation 6.3.

$$N_{i=1\dots M} \rightarrow NodeA : ID_N || ID_{Beh} || D_{AVG} \quad (6.6)$$

CHAPTER 6. CARRADS

Node A computes the final global decision value using the equation in 6.4 and derives the global consensus.

The integrity of the cooperative decision procedure will be under threat, if a malicious node can spoof the decision responses. This threat can be more serious if the malicious node spoofs the MAC address in the spoofed decision responses. To counter this threat, similar to CARRADS detecting spoofed routing messages, the functionality of detecting spoofed decision responses is added to CARRADS.

CARRADS detects spoofing of routing messages primarily by comparing routing protocol statistics with the MAC protocol and physical channel properties. The routing protocol statistics contains statistical information from the routing information communicated between the nodes. The MAC protocol statistics contains statistical features that define the traffic pattern of the routing control traffic. The physical channel statistics contain statistical information on the physical medium used to communicate with each of the neighbors. Combining MAC and physical channel properties, CARRADS forms a unique traffic signature for the network neighborhood. The statistical difference between the traffic signature and the declared false routing information given in a malicious routing message is used to identify spoofing. Correlation functions in the association process are used to capture this statistical difference.

To augment the functionality of detecting spoofed decision responses in CARRADS, statistical features from the cooperative decision protocol is defined and added to the feature set. Similar to the detection of spoofed routing communication, CARRADS detects spoofed decision responses, even with a fabricated MAC address. Thus, CARRADS uses its own functionality to protect itself against spoofing of decision responses.

The cooperative decision procedure is a collaborative process. Hence, the integrity of the procedure is governed under the assumption of Byzantine agreement condition.

The Byzantine agreement condition states that when two-third of the nodes in the network neighborhood needs to be benign for the collaborative process to be secure [57]. The Byzantine condition is relaxed if the nodes are mobile. When the nodes are mobile, nodes communicate with more neighbors to obtain decision responses. This property relaxes the Byzantine condition based on the number of benign neighbors. However, the Byzantine condition states that of all decision responses received by a node, two-thirds should be benign protecting the integrity of the cooperative decision process.

6.2.5 Data Reduction Techniques

To decrease the computational overhead in the machine learning algorithms, the data reduction module aims to reduce the dataset size, without affecting its information content. CARRADS uses the association process similar to CRADS to reduce the dataset. Association combines cross layer features to form a reduced set of derived features.

6.2.5.1 Association

To preserve the compatibility of CRADS's initial detection model in CARRADS, the functions of association are similar to that of CRADS. The process of association correlates one or more features from different layers to a specific MAC layer feature. The output of this process yields a reduced set of features. Specific correlation functions are used to combine and derive the reduced features. These functions are designed based on the dependency of time, traffic and topology characteristics of the features that are being correlated. Eighteen predetermined correlation functions are used to derive the reduced dataset from the original 46 cross layer features dataset. The cross layer features and the eighteen correlations functions are listed in Appendix A.

6.2.6 Learning Module

The core of CARRADS lies in the learning module. This module comprises of two learning algorithms; SVM and FDA. SVM is the primary learning algorithm used for detecting malicious behavior. As stated earlier, FDA is a supporting algorithm, which is used to estimate the information content of new patterns.

To augment the ability of real-time incremental learning to the conventional SVM, the intrinsic characteristics of SVM are exploited in the learning module. The SVM algorithm selects a subset of vectors from the training dataset. These vectors are referred to as support vectors and these vectors govern the decision boundary. The unique property of SVM is that by training the SVM detection model with just the support vectors will yield the same solution as training with the whole dataset. In CARRADS, the above intrinsic property is used as the key concept.

6.2.7 Incremental SVM Learning

Unlike other machine learning algorithms, SVM does not search for an equation of a plane that separates the two classes of vectors. Instead, it searches for a direction represented by the vector w , where the vectors of two classes are best separated. The length of w is inversely proportional to the distance or margins between the two classes of vectors. The vectors bordering on these margins are selected as support vectors and used to govern the decision boundary of the SVM detection model.

The SVM model relies primarily on the Lagrangian optimization technique to find the best direction w and its corresponding support vectors. As the distance between the margins is represented as $2/\|w\|$, the Lagrangian optimization aims to maximize this value by minimizing $\|w\|$. The following text explains the mathematics of SVM.

Vectors which lie on the margins will satisfy the global minimum conditions of a Lagrangian minimization problem. Thus, the SVM optimization problem aims to

CHAPTER 6. CARRADS

find vectors which satisfy global minimum conditions. The Lagrangian formulation is $L = f(x) + \sum^l \alpha_i g_i(x)$, where $f(x)$ is the objective function which is either minimized or maximized and $g_i(x)$ is the constraint function. l is the number of training vectors and α_i is the Lagrangian multiplier of the i^{th} vector. Given the above Lagrangian formulation, the SVM problem can be expressed as in equation 6.7.

$$\min_{w,b} L_P = \frac{\|w\|^2}{2} + \sum_{i=1}^l \alpha_i (y_i (w^T \cdot x + b) - 1) \quad (6.7)$$

In the above equation, $\|w\|$ is minimized under the constraint that $y_i (w^T \cdot x + b) \geq 1$ by using the Lagrangian variable α . In other words, the margin $2/\|w\|$ is maximized under the constraint that all training vectors are correctly classified. As the equation 6.7 is convex, the Krush-Kuhn-Tucker conditions [20] can be used to rewrite the Lagrangian equation in dual form.

$$\max_{\alpha} L_D = \sum_{i=1}^l \alpha_i - \sum_{i=1}^l \sum_{j=1}^l \alpha_i \alpha_j y_i y_j \langle x_i \cdot x_j \rangle \quad (6.8)$$

Equation 6.8 maximizes the alpha values. An iterative process is used to find the optimal alpha values. After an optimal solution is found, vectors with non-zero alpha values are the support vectors and SVM decision boundary is defined by them. In SVM, non-linearity is introduced by transforming the linear dot product space $\langle x_i \cdot x_j \rangle$ in equation 6.8 to a non-linear space called the kernel space. The transformation functions used for mapping are referred to as kernel functions [68].

Training SVM using a naive iterative process is computationally expensive even for highly resourceful systems. Therefore, SVM uses heuristic decomposition methods for finding the alpha values in equation 6.8. Sequential Minimal Optimization (SMO) is one of such decomposition methods and is the most widely used and accepted methodology for SVM training [24, 68].

CARRADS uses two properties of SVM to its advantage. First, as stated earlier, the support vectors are sufficient to define the model complexity. Second, the alpha values obtained during the training stores the history of a solution search direction. Therefore, restoring these alpha values will decrease the complexity of finding the new model. Using the above two properties, the SMO algorithm can be modified in three ways for augmenting the ability of incremental SVM learning. The three types of incremental SVM (i-SVM) are discussed below.

6.2.7.1 Incremental SVM type 1

A simple modification to the basic SVM model to make it learn incrementally is to reload the old alpha values of the support vectors and retrain the SVM from where the last training has left [76]. Entire original dataset together with the new vectors is used during the retraining. The alpha values for the new vectors are initialized to zero. This way of retraining is optimal as the retraining enables old non-support vectors to switch to support vectors and vice-versa. Eventhough loading of old alpha values reduce the computational and space overhead to a large extent, retraining is still not feasible in mobile nodes.

Adaptive intrusion detection methodologies proposed in the literature [56,91] follow a similar approach to i-SVM type 1. They retrain the detection model from scratch every time new patterns become available. This method of adaptation is not practical in ad hoc networks because of its computational requirements. This will be illustrated during the analysis of CARRADS in the section 6.3.

6.2.7.2 Incremental SVM type 2

A computationally effective way of incremental learning is to consider only the support vectors during retraining. For retraining, both the old support vectors and the new vectors are used as the training dataset. The solution obtained using this method is

sub-optimal as the non-support vectors are '*forgotten*' at each stage of retraining.

However, there are two advantages in using this method in CARRADS. The primary advantage is the significant reduction in computational overhead, making incremental non-linear learning possible in resource constrained nodes. Second, forgetting old vectors with less information content increases the adaptation efficiency of CARRADS.

6.2.7.3 Incremental SVM type 3

CARRADS uses the i-SVM type 3 method of incremental SVM retraining. Incremental SVM type 3 learning is similar to i-SVM type 2. In addition to forgetting non-support vectors at each adaptation process, the old support vectors are loaded with their respective old alpha values and retraining starts from where the last training has left. This method preserves the original search direction of the last training and results in a solution close to the solution obtained using i-SVM type 1 algorithm. This phenomenon will be illustrated in section 6.3, where we compare the detection accuracy by these three methodologies.

6.3 Results

CARRADS is simulated using Qualnet 3.5 and evaluated to illustrate the capability of incremental learning of SVM and to show that CARRADS computational overhead is tolerable with negligible decrease in detection accuracy. The performance of CARRADS is evaluated by measuring the trade-off between detection accuracy and computational overhead. The difference in detection accuracy between i-SVM type 1 and i-SVM type 3 is measured. Also, the major performance measure is the computational overhead. Comparing the difference in detection accuracy with the achieved reduction in computational overhead illustrated the CARRADS's trade-off between

CHAPTER 6. CARRADS

Figure 6.3: Hello Based Routing Attacks with 30 m/s Random Waypoint Node Mobility

detection accuracy and computational overhead.

6.3.1 Experimental Setup

Similar to CRADS simulation, five types of routing attacks are simulated using the two routing control messages, Hello and TC routing control message. The routing attacks include:

CHAPTER 6. CARRADS

Figure 6.4: TC Based Routing Attacks with 30 m/s Random Waypoint Node Mobility

- Hello Modification (HELMOD)
- Hello Spoofing (HELSPOOF)
- TC Modification (TCMOD)
- TC Spoofing (TCSPOOF)
- Sinking (Sinking)

CHAPTER 6. CARRADS

Figure 6.5: Hello Based Routing Attacks with 40 m/s Vehicular Node Mobility

The above five attack scenarios are simulated over 5 different mobility conditions ranging in 0, 5, 10, 30 m/s random way-point mobility and finally vehicular mobility with maximum mobility of 40 m/s. In vehicular mobility, the node movement directions are limited to 8° degrees of freedom as opposed to 360° degrees of freedom in random waypoint models. Hence, the five attacks over five different mobility conditions form 25 unique attack scenarios in which CARRADS is experimented and evaluated. To maintain readability, the graphs only show the results of two mobility

conditions, 30 m/s and vehicular mobility of 40 m/s.

Other simulation parameters are similar to the experimental setup of CRADS discussed in section 5.3.

6.3.2 Detection Accuracy

The primary evaluation parameter of any IDS is its detection accuracy. In our analysis, the detection accuracy is used as an evaluation parameter to show the training trends of three types of incremental SVM learning. Nevertheless, it can be observed from the graphs, figure 6.3, 6.4, 6.5, 6.6 that detection accuracy of the CARRADS is always above 90%. CARRADS achieves superior detection accuracy by utilizing the CRADS architecture.

Figures 6.3, 6.4, 6.5 and 6.6 show the detection accuracy of three types of incremental SVM learning. It can be evidently seen in all these figures that the detection accuracy of CARRADS incremental learning (SVM type 3) stays close to the accuracy achieved by conventional SVM training (SVM type 1). The training trend between i-SVM type 1 and i-SVM type 3 is very similar and only a negligible difference exists between the solutions obtained by the two types of incremental SVM learning.

The decrease in detection accuracy of i-SVM type 2 is due to the heuristic optimization methods used by SVM algorithm for finding the support vectors. Due to heuristics, the direction of the decision boundary may be sub-optimal, which affects the detection accuracy. This phenomenon does not happen in i-SVM type 3, as the original search direction of the optimization algorithm is preserved by restoring the values of the Lagrangian variable.

6.3.3 Computational Overhead

The primary evaluation parameter of CARRADS is its computational overhead. The negligible compromise in accuracy in CARRADS is more than adequately compen-

CHAPTER 6. CARRADS

Figure 6.6: TC Based Routing Attacks with 40 m/s Vehicular Node Mobility

sated through the significant reduction in computational overhead. In general, CARRADS with i-SVM type 3 requires less processing time to adapt/train the detection model. With respect to detection accuracy, i-SVM type 3 is very similar to the computationally expensive i-SVM type 1 detection model (trained from scratch). CARRADS reduces the computational overhead by a factor of 20-30, which makes CARRADS deployable in resource-constrained mobile nodes. Additionally, CARRADS was tested on a publicly available intrusion detection dataset from KDDCUP

CHAPTER 6. CARRADS

1999 [1] and the results were found to be similar. As the number of support vectors (N_S) is approximately equal to the total number of training vectors, the computational complexity bounds expressed in equation 5.12 is applicable to CARRADS.

Besides finding that CARRADS is computationally efficient, the analysis showed other interesting factors that affect the computational overhead of incremental learning. Comparing the computational overhead graphs in figure 6.3 and figure 6.4, it can be observed that vehicular mobility incurs more processing overhead than random-waypoint mobility. In other words, adapting the detection model is harder if the mobility level is high. Similarly, it can be observed that spoofing attacks are harder to adapt than modification or sinking attacks. Comparing figures 6.3 and 6.4 with figures 6.5 and 6.6, it was found that TC attacks are harder to adapt than Hello based attacks.

Publications:

- J.F.C, Joseph, Amitabha Das, Bu-Sung Lee, Seet Boot Chong, "*CARRADS: Cross Layer based Adaptive Real-time Routing Attack Detection System for MANETS*", Elsevier Computer Networks Journal, *Accepted*.

Chapter 7

Game Theoretic Trust Management Module

Trust Management is an essential security component, especially in highly autonomous systems, such as mobile ad hoc networks. The objective of a trust system is to measure the reliability of nodes in the neighborhood, so that critical operations, such as, routing, security information exchange, etc. can be done using the most trusted nodes in the neighborhood.

This chapter presents the third major component in the security architecture proposed in this thesis, namely, recommendation based trust module using game theoretic scheme. In recommendation based trust mechanism, the node collect recommendations from its neighboring nodes. These recommendations are quantitatively combined with local observation to determine the trustworthiness of the node. Here the notion of 'trust' can be defined as, *"the willingness of a node A to be vulnerable to the actions of node B based on the expectation that node B will perform a particular action important to the node A, irrespective of the ability to monitor or control node B"* [41, 77].

The process of quantitatively combining neighbor nodes' recommendations is mod-

eled as a non-cooperative strategic game played between the nodes, who are evaluating trust, with its neighboring nodes. This process is referred to as the trust game. The trust game is a corroborative process, which assesses trustworthiness of a neighbor node based upon observed network behavior and recommendations from other neighboring nodes. To assess the behavior of the neighboring nodes and to monitor suspicious or malicious activity, the trust module uses CARRADS. The measure computed by CARRADS is used as the primary metric to evaluate the trustworthiness of the neighboring nodes.

7.1 Game Theory

In this section, the fundamental concepts of game theory are presented. The rise of game theory was primarily due to our interest in the nature's cooperative behavior in an non-cooperative environment [35, 70].

A game is an interaction between a set of players (i.e., nodes). A strategic game is a scenario in which the players have to choose their respective actions simultaneously, without knowing other players' actions. Every player has different preferences on the possible outcomes in a game. These preferences are represented quantitatively using a matrix, referred to as the payoff matrix.

Thus, the strategic game is modeled as a tuple $N, (A_i)_{i \in N}, \pi_i$ [102], where N is the set of players, A_i is the set of action profiles for each player in N and π_i is the payoff matrix representing the preferences of the different outcomes in the game. The player is rational and aims to maximize his/her payoff.

To illustrate a non-cooperative strategic game [102], let us consider the famous prisoner dilemma (PD) problem. Prisoner's dilemma is a game between two players, each of them having two possible actions, namely, cooperate or defect. The preferences of different outcomes are quantitatively represented using a 2×2 payoff matrix, as

CHAPTER 7. GAME THEORETIC TRUST MANAGEMENT MODULE

		Player 2	
		Cooperate	Defect
Player 1	Cooperate	5, 5 (R)	1, 7 (L)
	Defect	7, 1 (T)	3, 3 (P)

Table 7.1: Prisoners Dilemma

shown in Table 7.1. In the game, if player 1 chooses to cooperate, whereas player 2 chooses to defect, then player 1 who became the 'loser' (L) receives the payoff of 1 and player 2 who gave to 'Temptation' (T) receives a payoff of 7. The above outcome is least preferred for player 1, whereas player 2 obtains the maximum payoff. If both players cooperate, which is optimal for both, and then each player receives the 'reward' (R) payoff of 5. On the other hand, if both players defect, the players are penalized (P) with payoff of 3. For the dilemma to exist, the payoff matrix structure $T > R > P > L$ has to be satisfied.

The commonly used solution concept for the strategic game is that of the Nash Equilibrium [93]. At Nash equilibrium, the players will have no incentive for unilateral change in their action given that other players do not change their actions. In the prisoner's dilemma problem, given the payoff matrix, the players will choose 'defect'. This is due to the fact that choosing 'defect' gives the maximum payoff for the player for any strategy played by the other player. Thus, Nash equilibrium occurs at ('defect', 'defect'). Note here that, Nash equilibrium in prisoner's dilemma is locally optimal but not a global optimal.

Iterative prisoner's dilemma (IPD) is a repeated game of prisoner's dilemma [93]. In this game, past history of opponents' actions are used to decide on the player's future action. Various phenomena such as, reputation, threats and revenge are captured in the IPD games.

Figure 7.1: Network Setup

7.2 Game Theoretic Trust Model

In this section, the game theoretic trust model proposed in this thesis is presented.

7.2.1 Network and Game Setup

As shown in figure 7.1, the ad hoc network is represented as a graph G with nodes as vertices V and the nodal links as edges E . The nodes are represented as $N_{i=1,\dots,L}$, where number of nodes in the graph is L . The network neighborhood of a node is defined as the set of nodes directly linked to the node. Hence, every node N_i has its own set of neighboring nodes. This set is denoted by NN_i and the number of nodes in NN_i is represented as K_i .

Every node in the graph assesses the trustworthiness of nodes in its network neighborhood by playing n -player games [16, 79]. To assess the trustworthiness of a neighboring node, the node plays an n -player game with its remaining neighbors in its network neighborhood. To illustrate the setup, consider a scenario where node A in figure 7.1 wishes to assess the trustworthiness of its neighbors. As node A has three neighbors, it plays a 3-player game for every one of the three neighbors. For instance,

to assess the trustworthiness of node 1, node A will play the trust game with node 2 and node 3. Hence, to generalize, the i^{th} node in the graph will be playing a K_i -player game to assess the trustworthiness of a neighboring node. The K_i -player game will be repeated K_i times for every neighbor in the node's network neighborhood.

The n -player game is decomposed into multiple 2-player games. This decomposition is applied since the sets of neighbors of the nodes vary. For example, the players (neighbors) of node A are different from those of node 3. This leads to the game setup of every node playing K_i^2 2-player games with its neighbors. Hereinafter, the term 'node' and 'player' will be used interchangeably.

7.2.2 Trust Game Objective

The objective of the trust game process is to build an autonomous and distributed trust evaluation and management system for ad hoc networks. Nodes compute belief based on the trustworthiness of other nodes in the network based on their local observations. The network trust game is designed to corroborate these beliefs with other nodes in its network neighborhood.

In the network, each node observes the network behavior of other nodes in its network neighborhood. Based on these local observations, the nodes will carry out a local assessment of trustworthiness of neighboring nodes. This local assessment of trustworthiness is referred to as the *local belief*.

The objective of the trust game process in every node is to corroboratively assess the trustworthiness of its neighbor nodes. The computed belief is referred to as the corroborative belief. The corroborative belief is the corroborated assessment on the trustworthiness of neighbor nodes. The game process derives the node's corroborated belief by considering the node's local belief and its confidence and recommendations from its neighbors. The neighbor recommendations consist of corroborated trustworthiness measure computed by respective neighbors.

The game process also considers the topological significance of the assessed node as well as the local service contributions made by the assessed node. These factors are taken into account when deriving the corroborated trustworthiness of the assessed node.

7.2.3 Computation of Local Belief and Confidence

The first step in establishing trust among neighboring nodes is to compute a node's local belief over the neighboring nodes. To compute the local belief and confidence, the proposed trust model uses CARRADS, which was presented in chapter 6.

Primarily, CARRADS returns two parameters; the local detection outcome and certainty (p) in the detection. The local detection, which is either '*malicious*' or '*benign*', is interpreted in the trust game as '*node untrustworthy*' or '*node trustworthy*', respectively. The certainty p in the detection, where $0 \leq p \leq 1$, is interpreted as the '*confidence in the belief*'.

Computation of local trust by using an intrusion detection mechanism allows the nodes in ad hoc network to dynamically assess trust of nodes. This also allows nodes to assess trust about neighboring nodes without a central controller. This methodology suits the infrastructure-less environment of ad hoc networks. Furthermore, intrusion detection offers a simple yet powerful solution for the computation of trust metric for ad hoc network nodes. As intrusion detection is already an integral component in the security system, the trust metric computation does not require any additional processing overhead. This suits the dynamic and resource limited nature of nodes in a typical ad hoc network.

7.2.4 Trust Game Formulation

Every node plays a strategic trust game with the neighboring nodes. The i^{th} player's game is represented as $\langle NN_i, D_i, \pi_i \rangle$, where NN_i is the set of neighborhood players,

CHAPTER 7. GAME THEORETIC TRUST MANAGEMENT MODULE

		Neighboring Node j states	
		Trusting	Not Trusting
Node i	Trusting	$TT_i \parallel TT_j$	$TST_i \parallel TST_j$
	Not Trusting	$TNT_i \parallel TNT_j$	$NT_i \parallel NT_j$

Table 7.2: Trust Game's Payoff Matrix

D_i is a set of available action strategies, and π_i is the payoff matrix representing the node's preferences. There are only two possible action strategies in the trust game, namely, '*trusting*' and '*not trusting*', which is numerically represented as $D \in \{-1, 1\}$. The outcome of the game is the computed (chosen) action strategy represented by variable D_i .

At real time, the trust game is played repeatedly [32]. This process generates a history of game outcome for each node. At time t , the history of the neighboring nodes for the i^{th} node is represented as $h_j^t, j \in NN_i$ and $H_i^t = \{h_1^t, h_2^t, \dots, h_{K_i}^t\}$ is the history of all nodes in the neighborhood at time t , where K_i is the number of nodes in NN_i . Similarly, $H_i^m = \{h_i^{t-m}, h_i^{t-(m-1)}, h_i^{t-(m-2)}, \dots, h_i^t\}$ is the m -sequence decision history for the node i . At time t , the computed outcome of the game is the corroborated belief of the i^{th} node represented by variables h_i^t and D_i^t .

As shown in Table 7.2, the payoff matrix π_i of the i^{th} node consists of four elements each of which represents the payoff of different outcomes in the game. If both the node and neighboring node hold a '*trusting*' belief, this will yield a payoff for the node that is represented by TT_i . The payoff value represented by TST_i is the node's temptation to change its belief to '*trusting*' or hold its current '*trusting*' belief, when the neighboring node is holding a '*not trusting*' belief. Similar to TT_i , the payoff value represented by NT_i is the payoff for the outcome if both the node and the neighboring node are holding '*not trusting*' belief. TNT_i represents the payoff value for the temptation to change its belief to '*not trusting*' or hold its current '*not*

CHAPTER 7. GAME THEORETIC TRUST MANAGEMENT MODULE

Confidence Level	State	Priority of Goal
High Confidence	Trusting	Primary goal: Influence neighbors to change or hold their decision to trusting. Secondary goal: Hold trusting belief as its current decision (achieved by default)
Low Confidence	Trusting	Primary goal: Hold trusting belief as its current decision against different neighbors recommendations. Secondary goal: Influence neighbors to change or hold their decision to trusting.
High Confidence	Not Trusting	Primary goal: Influence neighbors to change or hold their decision to not trusting. Secondary goal: Hold not trusting belief as its current decision (achieved by default)
Low Confidence	Not Trusting	Primary goal: Hold not trusting belief as its current decision against different neighbors recommendations. Secondary goal: Influence neighbors to change or hold their decision to not trusting.

Table 7.3: Priority of goals in the trust game

trusting' belief, when the neighboring node is holding a '*trusting*' belief.

Table 7.2 shows that the payoff matrix of neighboring node j is different from that of node i . Thus, the payoff matrices are *asymmetric*. The nodes are also unaware of the payoff matrix of their neighboring nodes and have no common knowledge. In game theory terminology, this property of nodes having no common knowledge is defined as *irrationality*. However, during the trust game-play, the game process obtains *bounded rationality* which builds hypotheses for the neighboring node's payoff matrix by probing and a trial-and-error approach.

In the game payoff, a node has two primary goals. The node achieves the first goal if it can hold its local belief as the corroborated belief against different beliefs from the neighbors. The second goal is to influence the neighboring nodes with its corroborated belief. Therefore, the nodes in the game will be contending to maintain their own local beliefs amidst different corroborated beliefs from neighbors and try to influence their own local belief on the neighbors.

As shown in table 7.3, the priority of these goals depends on the confidence in the local belief. If the confidence in the local belief is high, the primary goal is to influence other neighboring nodes with its own belief has high priority. In this scenario, the objective of the node to hold the local belief as the corroborated belief is achieved by default. On the other hand, if the confidence in the local belief is low, the goal to influence other nodes with its local belief has low priority and the goal is to maintain its current local belief has high priority. Hence, the nature of the game changes as a function of the confidence in belief. This property in the trust game creates *heterogeneous players* [37] in the graph. In section 7.2.5, the properties that define the nature of the players are discussed.

To summarize, the trust game is an iterative game of heterogeneous players, each playing an n -player game with its neighbors. The payoff matrix is asymmetric and there exists no common knowledge between the players, thus, making the players irrational. A bounded rationality [102] is obtained during the trust game using probing.

7.2.5 Payoff Matrix Structure

The outcome preferences of a node differ based on its local belief and confidence in its belief. The above two parameters are derived from the CARRADS outcome, which will be discussed later in the chapter. The local belief is either '*trusting node*' or '*not trusting node*'. The confidence in belief ranges between $0 \leq C \leq 1$. Nodes are classified as high confident or low confident nodes based on the parameter C . Nodes are defined high confident if the value of C is higher than a predefined threshold. In our experiments, we have set the threshold to 0.8, thus high confident nodes have $C \geq 0.8$ and low confident nodes have $C < 0.8$. This threshold is arbitrarily set based on the required sensitivity for response to malicious behavior. Increasing the value of C will increase the sensitivity of the trust mechanism to suspicious routing behavior. Thus, the two belief parameters form four different cases of outcome preferences; high-

CHAPTER 7. GAME THEORETIC TRUST MANAGEMENT MODULE

Reference	Belief	Structure
Type 1a	High Confident Trusting Nodes with Temptation to change belief	$TST > TT > TNT > NT$
Type 1b	High Confident Trusting nodes with No Temptation	$TST > TT > NT > TNT$
Type 2	Low Confident Trusting nodes	$TNT > TT > TST > NT$
Type 3a	High Confident Not Trusting nodes with Temptation to change belief	$TNT > NT > TST > TT$
Type 3b	High Confident Not Trusting nodes with No Temptation	$TNT > NT > TT > TST$
Type 4	Low Confident Not Trusting Nodes	$TST > NT > TT > TNT$

Table 7.4: Payoff Matrix Structures based on Belief

confident trusting nodes, high-confident not trusting nodes, low-confident trusting nodes and low-confident not trusting nodes.

The confidence in local belief is directly proportional to the supporting evidence. However, lack of supporting evidence for one belief does not strengthen the opposite belief. Hence, there is no relation between low confident trusting and high confident not trusting or vice versa. That is, low confidence trusting does not imply that node has evidence for high confident not trusting. Thus, the four cases of trust beliefs are mutually exclusive.

As stated earlier, the outcome preferences of a player is quantitatively represented in the payoff matrix. In Prisoner's dilemma problem, the nature of the problem led to the payoff matrix structure of $T > R > P > L$ (see table 7.1). However, in the trust game, the nature of the game [101] changes depending on the node's local belief and confidence. Therefore, the payoff matrix structure differs depending on the outcome preferred by the node. As we have six different outcome preferences, six payoff matrix structures are defined. These payoff matrix structures are shown in Table 7.4.

CHAPTER 7. GAME THEORETIC TRUST MANAGEMENT MODULE

Similar to the Iterative prisoner's dilemma game, the trust game is played repeatedly at real time. In the repeated trust game, the nodes exhibit two kinds of phenomenon, namely, '*prove*' or '*convince*'. The low confident nodes allow themselves to be '*convinced*' by the high confident nodes. High confident nodes '*prove*' their belief when played with other high confident nodes with opposite belief. The above two phenomenon is enforced by the structure of the payoff matrix.

From the payoff matrix structures in Table 7.4, it can be observed that low confident nodes have higher values of '*temptation to change belief*'. This allows them to be open for the high confident nodes to '*convince*' them to switch belief. On the other hand, low confident nodes have low payoff value for their opposite belief. This payoff structure allows high confident trusting nodes to support and '*prove*' their belief. Thus, low confident nodes are biased towards the belief of high confident nodes with the same belief. As low confident nodes have higher temptation to change their belief, unlike high confident nodes (type 1b and 3b), the payoff matrix structure subtypes with no temptation is not defined.

On the other hand, high confident nodes have high temptation values for their own belief, which allows them to resist temptation from nodes with opposite belief. In addition, they have higher payoff values for their belief, which helps them to prove their belief to other neighboring nodes.

7.2.6 Payoff Matrix Values

The '*payoff*' for a node is defined as *the quantitative measure of gain achieved by the node by exhibiting a particular belief over a neighboring node and the confidence it has in that belief*. The payoff/gain received by the node depends on two parameters; the importance of the neighboring node and evidence for trusting or not trusting the neighboring node. The importance of the neighbor node is computed using the degree of node connectivity in the local network neighborhood offered by the neighbor node

CHAPTER 7. GAME THEORETIC TRUST MANAGEMENT MODULE

	Trust	Not Trusting
High Evidence	Increase Payoff	Decrease Payoff
Low Evidence	Decrease Payoff	Increase Payoff
High Importance	Increase Payoff	Increase Payoff
Low Importance	Decrease Payoff	Decrease Payoff

Table 7.5: Evidence in belief and Neighbor node importance in the Payoff matrix

and the local route services contributed by the neighbor node. The detection outcome from CARRADS is used to quantify the evidence for trusting or not trusting. This is referred to as evidence in belief.

The influence of evidence in belief and importance of neighbor node on the payoff is shown in table 7.5. In the first case, the node holds a trusting belief on the neighbor node. In this case, higher evidence in belief increases the payoff, whereas, lack of evidence decreases the payoff. Higher importance of the neighbor node also increases the payoff. In the second case, the node holds a 'not trusting' belief over the neighbor node. With this belief, higher evidence in belief causes the payoff to decrease. In addition, higher importance of the neighbor node increases the payoff.

The payoff matrix values are computed to reflect the above properties. The confidence of belief computed earlier for deciding the payoff matrix structure is used for computing payoff matrix values. However, unlike the method of choosing payoff matrix structure, the confidence in belief is not discretized and used as a continuous variable ranging from 0 to 1. The initial payoff matrix is computed by using the confidence in belief. The payoff matrix values $\hat{\pi}_i$ are computed such that higher confidence values cause the difference between the maximum and minimum value in the payoff matrix to increase. This increase in the difference quantitatively reflects the evidence to the current belief.

For example, in type 2 (low-confident trusting) nodal trust belief, the difference

CHAPTER 7. GAME THEORETIC TRUST MANAGEMENT MODULE

between TNT and TT states the evidence in the current nodal belief. Here, the difference indicates the degree of openness by the low confident node to allow other non-trusting neighboring nodes to convince the node to change its belief.

The importance of neighboring nodes is computed by exploiting the CARRADS's monitoring capabilities. Of the 46 cross layer network statistics CARRADS monitors, the seven statistical network parameters listed below are used to compute the importance of the neighboring nodes.

Degree of topological connectivity:

- Number of two-hop neighbor nodes connected through the neighbor (THB)
- Number of nodes with the similar connectivity (SIM)
- Number of nodes which have the route through the node (R_{EST})

$$S(NN) = \left\{ \begin{array}{l} \text{if } SIM > 0, \text{ then } S(NN) = 0 \\ \text{Otherwise, } S(NN) = \beta (THN + R_{EST}) \end{array} \right\} \quad (7.1)$$

Local Contribution of neighboring node:

- Data/Route Forwards (D_F)
- Data Error (D_{ERR})
- Successful Routes (R_S)
- Route Error (Non-existent routes) (R_{ERR})

$$C(NN) = (D_F/D_{ERR}) \cdot \beta (R_S/R_{ERR}) \quad (7.2)$$

Importance of services is computed by two functions; degree of topological connectivity of node $S(node)$ and local service contribution of node $C(node)$. As shown in equations 7.1 and 7.2, the first three statistical parameters in the above list are used to compute the degree of topological connectivity of the node, whereas the last four are used to compute the local service contributions by the node. In equations 7.1 and 7.2, β is the standardisation constant, which is used to normalize the variables. At real time, the actual payoff matrix is computed by $C(NN_i) + S(NN_i) \cdot \hat{\pi}_i$, where $\hat{\pi}_i$ is the initial payoff matrix.

7.2.7 Systematic Selection Criterion - '*The Trust Game*'

The trust game is a process for choosing a decision based on a selection criterion. The criterion is designed to select a decision that maximizes the payoff received by the node [94]. To illustrate the selection process, consider a game scenario, where the network is assessing the trust of a particular node S .

Nodes choose their local belief as their initial decision over the trustworthiness of a neighboring node S . The trust game starts with nodes declaring their initial decision to their neighboring nodes. Every node in the network will use the selection criterion to make its decision for the next iteration of the trust game. At the end of the current iteration, each node announces their achieved total payoff together with their decision. The selection criterion used by the trust game for changing its current decision is based on equation 7.3. This is the solution concept for the trust game.

$$\Pi_j(H_j^{t-1}, h_i^{t-1}) > \Pi_i(H_i^{t-1}, h_i^{t-1}) > \Pi_i(H_i^{t-1}, h_j^{t-1}) \quad (7.3)$$

where, $j \in NN_i$ and $h_i^{t-1} \neq h_j^{t-1}$

$$\Pi_i(H, D) = \sum_{d \in H} \pi_i(D, d) \quad (7.4)$$

The payoff function $\Pi(H, D)$ in equation 7.4 computes the aggregate payoff achieved in the game. It summates the individual payoff achieved by playing the trust game with each neighboring node. H is the set of decisions declared by the neighboring nodes at the last iteration of the trust game. D is the current decision held by the node.

Equation 7.3 states the selection criterion for the trust game. In equation 7.3, h_i^t is the decision $D_i \in \{-1, 1\}$ at time t (or t^{th} game). The criterion has to be satisfied for the node to change its current decision. The first part of the relation $\Pi_j(H_j^{t-1}, h_i^{t-1}) > \Pi_i(H_i^{t-1}, h_i^{t-1})$ in the equation compares the aggregate payoff between the node and the neighboring nodes. If the relation holds true for the j^{th} neighboring node, the neighboring node j has achieved a greater payoff by having a different belief to node i . The second relation in the equation, $\Pi_i(H_i^{t-1}, h_i^{t-1}) > \Pi_i(H_i^{t-1}, h_j^{t-1})$ checks whether switching to node j 's decision increases the aggregate payoff using the node's own payoff matrix. If both conditions are satisfied, the node changes its decision in the next game.

7.2.8 Nash Equilibrium - Trust Game

As discussed in section 7.2.5, players are classified into six types based on their payoff matrix structure. Though the trust game formulation defines the players as heterogeneous, due to the players' bounded rationality, the player individually assumes a homogeneous game setup for its neighbors. That is, each individual player assumes that all nodes play the game using the same payoff matrix structure. This property allows the players to reach a unique Nash equilibrium (NE) [102]. In type 1 and type 3 player games, there exists a unique Nash equilibrium. The unique Nash equilibrium for type 1 node games occurs at 'trusted' and Nash equilibrium for type 3 node games occurs at 'not trusting'. In type 2 and type 4 players, the players have a bounded rationality and these players probe the neighbors for finding the optimal

CHAPTER 7. GAME THEORETIC TRUST MANAGEMENT MODULE

payoff. Hence, the Nash equilibrium in type 2 and type 4 players depends on the neighbors' corroborated beliefs and payoff.

As the type 2 and type 4 players have bounded rationality, they probe their corresponding players to find the optimal payoff. This raises the possibility of the players becoming unstable during this process. To ensure that the players follow a path towards the unique Nash equilibrium that exists in the game, a sub-game perfect Nash equilibrium is maintained. The conditions for the sub-game perfect Nash equilibrium are expressed in equations 7.5 and 7.6.

$$\Pi_i(H_i^t, h_i^t) \geq \Pi_i(H_i^{t-1}, h_i^{t-1}) \quad (7.5)$$

$$[\Pi_i(H_i^t, h_i^t) - \Pi_i(H_i^{t-1}, h_i^{t-1})] - \\ [\Pi_j(H_j^t, h_j^t) - \Pi_j(H_j^{t-1}, h_j^{t-1})] > 0 \quad (7.6)$$

where, $\overline{NN_i} \subseteq NN_i$, such that $k \in \overline{NN_i}$, if $h_k^t \neq h_i^t$, for all k in NN_i

$$\Pi_i(H_i, h_i^t) \geq \Pi_k(H_k, h_i^t), \forall k \in \overline{NN_i} \quad (7.7)$$

Equations 7.5 - 7.7 specify the equilibrium criteria for the decision selection process in the trust game. For the decision to be stable, the total aggregate payoff at time t should be greater than the total aggregate payoff at time $t - 1$. In other words, for stability of the decision, the payoff gain must be positive to change the current decision. The sum of payoff gain achieved by the node and neighboring node due to the decision change should always be positive. The above two stability conditions are expressed in equations 7.5 and 7.6.

The condition in 7.7 is satisfied when player i reach both a social optimum and equilibrium. $\overline{NN_i}$ is the subset of neighborhood nodes which oppose the currently

held decision h_i^t by node i . These nodes hold the decision $\neg h_i^t$, which is the negation of h_i^t . The equilibrium condition in equation 7.7 states that, node i has achieved a payoff greater than the payoff obtained by its corresponding neighbors. This is the social optimum for node i . At this point, there is no payoff gain for node i to change its decision to $\neg h_i^t$. Thus, the equilibrium is reached for node i .

However, the neighbor nodes may not be in social optimum or equilibrium. But, at the point when the payoff gain computed by equation 7.6 for all neighbors are negative, the neighbors will not change their decision. At this point, the neighbors will reach equilibrium, but not its social optimum. The computation of payoff gain ensures the stability of neighbor nodes that will not reach social optimum. Thus, the equilibrium is reached in the network neighborhood.

The number of iterations (T_E) required to reach the equilibrium depends on various factors [55]. T_E is inversely proportional to the probability of high confident nodes that hold the majority consensus belief. That is, scenarios with high probability of high confident nodes that holds the majority consensus belief will reach equilibrium faster. Also, clusters of high confident nodes with the majority consensus belief will help to reach equilibrium faster. Clusters of high confident nodes convince nodes faster than scattered/distributed high confident nodes. In other words, cluster of high confident nodes converge faster to the optimal decision on trustworthiness. However, clusters should be distributed so that influence of these high confident nodes is effective throughout the network.

7.3 Evaluation of Game Theoretic Trust Model

The evaluation focuses on the performance of the trust model in deducing the decision towards the trustworthiness of neighbor nodes. To evaluate the performance, the trust model is simulated under various scenarios and its performance is compared

CHAPTER 7. GAME THEORETIC TRUST MANAGEMENT MODULE

Symbol	Payoff Matrix Structure	Value
P_+^H	Probability of nodes with Type 1a	0, 0.25, 0.5, 0.75, 1
P_+^L	Probability of nodes with Type 2	0, 0.25, 0.5, 0.75, 1
P_-^H	Probability of nodes with Type 3a	0, 0.25, 0.5, 0.75, 1
P_-^L	Probability of nodes with Type 4	0, 0.25, 0.5, 0.75, 1

Table 7.6: Probability Scenarios

with existing game based trust model proposed by Baras et al. [96]. Baras et al's work was discussed in detail in section 2.4.1. Two performance metrics defined for the game theoretic trust model are the percentage of mean equilibrium and average consensus of nodes.

- *Equilibrium*: Equilibrium is the percentage of nodes in the network that have consistent decision history.
- *Consensus*: Consensus is the percentage of nodes that holds the correct decision on trustworthiness.

The performance of the trust model is studied in various experiments with different network scenarios.

7.3.1 Experimental Setup

The experiment consists of network represented by a random graph with 2500 nodes. The number of edges/connectivity (degree) per node ranges between 2 to 8. Hence, nodes have a minimum of two neighbors and are never isolated.

As specified in Table 7.3, the player type and its corresponding local trust belief can be of six types based on its payoff structure used by the node to play the trust game. To maintain readability, the experiments omit types 1b and 3b. The observa-

CHAPTER 7. GAME THEORETIC TRUST MANAGEMENT MODULE

Scenario No	TT	TST	TNT	NT	Certainty
1	0.15	0.2	0.1	0.05	0.15
2	0.15	0.25	0.1	0.05	0.2
3	0.2	0.25	0.15	0.05	0.2
4	0.2	0.25	0.1	0.05	0.2
5	0.25	0.3	0.2	0.05	0.25
6	0.25	0.3	0.15	0.05	0.25
7	0.25	0.3	0.1	0.05	0.25
8	0.3	0.35	0.25	0.05	0.3

Table 7.7: Type 1 Payoff Matrix Scenarios

tions in types 1b and 3b are similar to node type 1a and 3a, respectively. Hence, the experiments consider only four local beliefs and are referred to as nodal trust beliefs.

Table 7.6 shows the first set of scenarios considering different probabilities of nodes in the graph for these four nodal trust beliefs. For each trust type, five scenarios with different probabilities are defined. The probability values for the five scenarios are 0, 0.25, 0.50, 0.75 and 1. The different combinations of four types of nodal trust beliefs and the five probability values, 35 different probability-based scenario configurations are defined. These 35 scenarios are shown on the x-axis in figures 7.2-7.6.

The second set of scenarios considers different levels of certainty in the nodal trust beliefs. In all of the four types of nodal trust beliefs, the certainty is quantitatively expressed as the difference between the higher value and lower value of the payoff matrix structure. To illustrate this, Table 7.7 states the values used for the eight scenarios in type 1a nodal trust belief. In this case, $|TST_i - NT_i|$ gives the certainty in belief. Eight different payoff matrix structures are defined with the certainty in nodal belief ranging from 0.15 to 0.3. Given the above scenario configuration, there are eight possible payoff matrix scenarios for each type of nodal trust belief. The

CHAPTER 7. GAME THEORETIC TRUST MANAGEMENT MODULE

Figure 7.2: Equilibrium with Probability Scenarios

different combinations of the payoff matrix scenarios yield $8^4 = 512$ scenarios. This set of scenarios is referred to as the payoff matrix scenarios. The 35 probability scenarios are also combined with the different payoff matrix scenarios.

7.3.2 Performance Analysis

7.3.2.1 Equilibrium

Figure 7.2 shows the percentage equilibrium achieved in probability based scenarios. In this figure, we can observe that there are three ranges of equilibrium, 98%-100%, 90%-94% and 86%-84%. The scenarios that achieve 84% equilibrium are those with 50% high confident nodes for each of the two beliefs (Scenario 28). 86% of nodes attain equilibrium with 25% high confident nodes with one belief and 75% high confident nodes (Scenario 19 and 33). These are worst case scenarios, where the nodes in the network have strong belief. Scenarios that achieve equilibrium in the range of 90%-94% are configuration with 50% high confident nodes and 25% high confident nodes of opposite belief and 25% low confident nodes with opposite belief. All other

CHAPTER 7. GAME THEORETIC TRUST MANAGEMENT MODULE

Figure 7.3: Equilibrium with combination of different probability and payoff matrix scenarios

probability scenarios achieve equilibrium within the range of 98%-100%.

It is also interesting to note that, the small group of nodes with high confident in their belief can convince a large group of nodes with low confidence in the network. This aids nodes who individually discovered the maliciousness of the node S (node under trust assessment) and can influence other nodes with the correct assessments. However, it should be noted here that, a single node cannot change the perspective of all nodes in the network. This ensures the integrity of the trust model from the malicious over-riding of trust dissemination.

Equilibrium achieved in scenarios combining different probability scenarios with payoff matrix scenarios are shown in Figure 7.3. It can be observed from the graph that, presence of high confident nodes with mixed beliefs causes instability.

7.3.2.2 Consensus

The second parameter used for evaluating performance of the trust model is the average consensus achieved in the different scenarios. Figure 7.4 shows the mean consensus achieved in probability based scenarios.

CHAPTER 7. GAME THEORETIC TRUST MANAGEMENT MODULE

Figure 7.4: Consensus for different probability scenarios

Our trust model is compared with

- Baras et al. [96] Voting based Trust model
- Using only Local Decision

The performance of using only the local decision for trust assessment versus our approach of global trust assessment is shown in figure 7.4. It can be observed from Figure 7.4, that the game based trust model offers a better consensus compared with the trust scheme proposed by Baras et al. [96]. Eventhough it is unfair to compare our global trust assessment performance with localized trust assessment, it nevertheless illustrates the advantage of using global trust assessment scheme in ad hoc networks.

Similar to the inference made from Figure 7.2, the mean consensus in these scenarios falls into three ranges; 100%, 92%-98% and 86%-88%. Scenarios with 75% or more with high confident nodes with the same belief and scenarios which have 100% low confident nodes reach a complete consensus. The second range of mean consensus 92%-98% is reached by scenarios that have 25% of high confident nodes that have belief opposing the correct decision. The third range of belief 86%-88% are achieved by scenarios which have 50% high confident nodes of both beliefs. Figure

CHAPTER 7. GAME THEORETIC TRUST MANAGEMENT MODULE

Figure 7.5: Consensus with combination of different probability and payoff matrix scenarios

7.5 shows the mean consensus combining the probability scenarios with the different payoff matrix scenarios.

It is interesting to note here that, a minimum consensus of 82% is reached in scenarios when both beliefs have equal probability in the network. In this scenario, the placement/locality of the nodes becomes important. A cluster of high confident nodes surrounding a high confident node with opposite belief will convince the node to change its belief. Therefore, the higher degree of scatter of high confident nodes decreases the influence it has on the network.

For every trust game iteration, the complexity is $O(n)$, where n is the number of neighbors. As n is usually small, the computational complexity of the trust game process is negligible.

7.4 Evaluation of the Security Architecture

In this section, we evaluate the proposed light-weight security architecture. The evaluation focuses on analyzing the performance and effectiveness of the architecture at real-time in providing security services to the ad hoc routing protocol.

CHAPTER 7. GAME THEORETIC TRUST MANAGEMENT MODULE

The simulation uses a table-driven routing protocol, namely Optimized Link State Routing (OLSR) at the network layer. The MAC and physical protocols that are used in the simulation are standard IEEE 802.11b protocols. Routing attacks are simulated over the OLSR. OLSR consists primarily of two routing control messages, which include Hello message and TC message. Hello messages are used for sensing neighborhood and sharing node connectivity information. TC messages, on the other hand, are used to propagate topology throughout the network.

The OLSR protocol in every node selects a subset of nodes from its neighbors for forwarding data traffic. This subset of nodes is referred to as the MPR set. The MPR set of a node is selected so that all the nodes in two-hop distance are connected by these MPR nodes. The connectivity information declared by the neighbors in the Hello message is used to compute the MPR set. By standard, Hello message is periodically beacons every 2 seconds. TC message propagates this MPR selection information throughout the network. Change in the MPR set triggers the generation of new TC message to update the neighbor nodes with the current MPR set.

The metric used to analyze the effectiveness of the security architecture is the average number of routes established through a node. For computing this metric, every node in the network simulation is defined with a counter R_E , which is incremented every time the node is selected as MPR and data traffic is observed through it. As stated earlier, MPR status of nodes is refreshed every 2 seconds with the beaconing of Hello messages. Thus, R_E of a node is incremented if there is data traffic observed in the 2 second time window. In the simulation, there are two types of nodes: trustworthy nodes and untrustworthy nodes. Hence, two equations are defined to derive the metric for each type of node.

$$RE_{TT}^{AVG} = \frac{1}{N_{TT} - 1} \sum_{i \in TT} RE_i \quad (7.8)$$

$$RE_{NT}^{AVG} = \frac{1}{N_{NT} - 1} \sum_{i \in NT}^{N_{NT}} RE_i \quad (7.9)$$

In equation 7.8, TT is the set of trustworthy nodes and N_{TT} is the number of trustworthy nodes. RE_{TT}^{AVG} is the average number of routes selected through a trustworthy node. Similarly, in equation 7.9, NT is the set of trustworthy nodes and N_{NT} is the number of untrustworthy nodes. RE_{NT}^{AVG} is the average number of routes selected through untrustworthy node.

7.4.1 Experimental Setup

Similar to the experimental setup for CRADS and CARRADS discussed in section 5.3 and 6.3.1, the simulated routing attacks include,

- Hello Modification (HELMOD)
- Hello Spoofing (HELSPOOF)
- TC Modification (TCMOD)
- TC Spoofing (TCSPOOF)
- Sinking (Sinking)

The above five attack scenarios are simulated over 5 different mobility conditions ranging in 0, 5, 10, 30 m/s random way-point mobility and finally vehicular mobility with maximum mobility of 40 m/s. Other simulation and routing attack setup parameters are similar to those used for evaluating CRADS module, which is discussed in section 5.3.

7.4.2 Effectiveness

To illustrate the effectiveness, the security services of the security architecture are used in an ad hoc routing protocol to find secure and optimal routes. The efficiency

CHAPTER 7. GAME THEORETIC TRUST MANAGEMENT MODULE

Figure 7.6: Hello based attacks

of the proposed architecture and the existing approaches are studied by observing the number of routes established through trustworthy nodes and untrustworthy nodes. A trustworthy node is defined as a benign node with no malign intent. Any malicious behavior by a trustworthy node is only due to the network conditions and characteristics. Figures 7.6, 7.7, and 7.8 compare the ability of the security schemes to avoid untrustworthy nodes in route selection and to give preference to trustworthy nodes

CHAPTER 7. GAME THEORETIC TRUST MANAGEMENT MODULE

Figure 7.7: TC based attacks

for routes selection. The figures show the number of routes established through a trustworthy node and untrustworthy node by the proposed security architecture and the existing cooperative scheme proposed by Deng et al [29]. The work proposed by Deng et al. is discussed in detail in section 2.3.3.4.

The decision metric (payoff received for the decision) derived using the security architecture is used as an additional optimization parameter in OLSR, when electing

CHAPTER 7. GAME THEORETIC TRUST MANAGEMENT MODULE

MPRs. The effectiveness of using the decision metric is studied and the results are shown in figures 7.6, 7.7, and 7.8.

The results observed for the Hello modification are shown in figure 7.6a and 7.6b. Figure 7.6a shows the routes established using a trustworthy node. In a benign network scenario simulation, where malicious nodes are absent and there is no security solution for route selection. In this simulation setup, the average number of routes per trustworthy node is 5384 routes per node.

Figure 7.6a shows the decrease of routes established through a trustworthy node due to the incorrect suspicion of Hello modification behavior on a truly trustworthy node. This is computed by the average RE_{TT}^{AVG} using equation 7.8. It is evident from figure 7.6a that usage of trustworthy nodes for route establishment is higher than the conventional cooperative scheme proposed by Deng et. al. This phenomenon is primarily due to effectiveness of the game-theoretic trust modeling which builds a corroborative trust assessment for its neighbors. The global consensus over the trustworthiness of nodes is better in the security architecture.

Figure 7.6b shows the ability of the security systems to avoid an untrustworthy node. This is computed by the average routes RE_{NT}^{AVG} using equation 7.9. In the optimal case, no routes should be established through an untrustworthy node. However, it should be noted here that, untrustworthy nodes cannot always be avoided, as they may be providing a critical link in the network topology, without which some nodes will become unreachable. Nevertheless, it is observed from figure 7.6b that the proposed architecture is better in avoiding untrustworthy nodes than the conventional cooperative scheme.

Similar to the inference made by comparing figure 7.6a and 7.7b, comparing the observations for different attacks in figure 7.6c and 7.6d, 7.7a and 7.7b. 7.7c and 7.7d, 7.8a and 7.8b will show that the proposed architecture is better in avoiding untrustworthy nodes and preferring trustworthy nodes for route establishment.

CHAPTER 7. GAME THEORETIC TRUST MANAGEMENT MODULE

Generally, increase in mobility of nodes affects the efficiency of the proposed architecture. The decrease in the difference between the efficiency of proposed architecture and the conventional methodology can be observed, especially in figure 7.7. As mobility of nodes increases, it becomes difficult for the trust mechanism to corroborate with the neighbors to build a global consensus towards the trustworthiness of its neighbors. This phenomenon is primarily due to the transient connections created by the mobility of nodes. The evolutionary game theoretic principle highly relies on the node behavior history to build the corroborative decision. In high mobile conditions, the communication period between the node and its neighbor is short and thus, limiting its efficiency. On the other hand, the conventional cooperative mechanism is not affected by the mobility of nodes. Nevertheless, the proposed architecture's effectiveness never falls below the conventional cooperative scheme.

Chapter 8

Conclusion

In this chapter, we conclude the dissertation by highlighting the contribution of the work and briefly discuss future research.

8.1 Summary

The dissertation presented a novel security architecture for ad hoc routing protocols without using cryptographic schemes. The absence of cryptographic schemes in the security architecture reduced the computational complexity and made it more suitable for resource-constrained ad hoc network nodes. The security architecture uses intrusion detection and trust management mechanisms to provide reactive security for ad hoc routing protocols. Intrusion detection is responsible for the detection process of the architecture, whereas the trust management is responsible for the decision process.

The intrusion detection mechanism is used to dynamically assess the benignity of neighbors' routing behavior over the network. The design of the intrusion detection mechanism aims to optimize two parameters: detection accuracy and computational overhead. Two techniques are used to maximize the detection accuracy of the in-

CHAPTER 8. CONCLUSION

trusion detection system. First, a cross layer methodology is used to define routing patterns. Statistical information is collected from multiple protocols at different layers to define routing patterns. This process of integrated multi-protocol routing behavior definition has increased the detection accuracy of the intrusion detection mechanism. Secondly, to further maximizing the detection accuracy, non-linear machine learning algorithms are used to classify benign and malicious patterns. Support Vector Machines (SVM), which is a non-linear machine learning algorithm is used. The use of cross layer methodology and non-linear machine learning algorithm has increased the computational complexity of the intrusion detection mechanism. To minimize the computational overhead, a set of novel overhead reduction measures are used. The training dataset used to train the detection model is reduced using the overhead reduction mechanisms.

At real-time, the detection accuracy of the intrusion detection mechanism is affected by changing topology and network conditions, which is induced due to mobility. To maintain the detection accuracy, real-time adaptiveness is incorporated into the non-linear machine learning algorithm. A novel adaptive incremental SVM algorithm is designed which adapts the detection model at real-time to changing topology and network conditions. The computational complexity of the adaptation process is minimized by exploiting the inherent characteristics of the SVM algorithm and constrained convex optimization.

The outcome from the intrusion detection mechanism is used by the game-theoretic trust mechanism to decide upon the trustworthiness of neighbors. The trust game process corroborates trust recommendations from the neighbors with the local assessment on the neighbor's routing behavior to compute the trustworthiness. The game process is designed to protect the trust evaluation process from incorrect recommendations from malicious or faulty neighbor nodes. The game process also ensures that corroborative trust evaluation process effectively identify malicious nodes and

measure the maliciousness based on their routing behavior.

The response process of the security architecture is integrated into the route computation process of the ad hoc routing protocol. The computed trustworthiness measure is used as an additional metric to compute secure and optimal routes in the network.

To illustrate the effectiveness of the security architecture, each module is individually evaluated and their effectiveness is studied. Additionally, the effectiveness of the security services offered by the security architecture is investigated. The results show that the security architecture is effective to avoid malicious nodes. However, it was observed that, to maintain connectivity, nodes cannot always avoid untrustworthy nodes. In a network topology, where an untrustworthy node is the only node with connectivity to one or more remote nodes, a tradeoff between security and connectivity has to be managed.

8.2 Future Work

As discussed in chapter 2 and shown in figure 2.1, besides routing attacks, ad hoc networks are also susceptible to other attacks such as, TCP SYN attacks, channel jamming, etc. Every protocol at each layer of the ad hoc network stack contains security vulnerabilities. However, unlike routing vulnerabilities, these vulnerabilities are not unique to ad hoc networks, and can be observed within legacy wired and wireless networks. Nevertheless, addressing them is important to secure ad hoc network completely and make ad hoc networks deployable commercially.

As shown in figure 2.1, conventional security threats can be classified into application layer threats, TCP/IP or transport layer threats and MAC and physical layer threats. In the literature, most of the proposed work addresses these threats separately. In legacy networks, differentiation of security services was necessary to

CHAPTER 8. CONCLUSION

classify security services depending on network hardware in which it will be implemented. Conventional network infrastructure consists of different types of networking hardware, which are capable and responsible for handling specific security services. For example, MAC and physical layer threats are handled by the network infrastructure rather than end-user nodes. Nodes with centralized monitoring and control authority may handle transport layer security. Application layer threats are managed by end-user nodes, using local intrusion detection mechanisms.

Ad hoc networks do not require the same differentiation of security services as the type of networking hardware are only end-user nodes. Thus, security mechanisms are common among all nodes. Research in ad hoc security so far have strictly adhered to the security structure in legacy network, which differentiate security mechanisms with respect to different protocols and layers.

We strongly believe that differentiating security mechanisms in ad hoc networks degrades both the performance and efficiency of the security services offered by the system. First, integrating detection systems of the different protocols will increase the detection accuracy of an attack. This phenomenon was illustrated in our proposed security architecture, where cross layer detection system yields better detection accuracy than protocol-specific detection mechanisms. Secondly, it is computationally not feasible to implement a dedicated cross layer system for every protocol in the network stack. In the proposed security architecture, one of the challenges is to make a cross layer detection system computationally feasible in ad hoc networks. Multiple cross layer systems are impossible in ad hoc networks.

Thus, our future work aims to integrate the ability to detect and respond to attacks at every protocol of the ad hoc network protocol stack. We aim to build a single security architecture which offers reactive security measures for attacks from different protocols.

We propose to integrate detection of conventional attacks by incorporating mul-

multiple data collection modules at every layer in the network stack and use a single detection engine. In other words, the security system will use the Multiple Data Collection and Single Data Analysis (MCSA) cross layer configuration discussed in section 5.1.1 in chapter 5 to integrate the ability of detecting conventional attacks. The malicious behavior will be generalized and detection system will be designed to detect both routing and conventional attacks.

8.2.1 MAC and Physical Layer Attacks

The node behavior in the proposed security architecture is defined using features from routing protocol, MAC protocol and physical channel properties. Hence, the proposed architecture is already designed to detect MAC and physical layer attacks. However, the features defined and the associated correlation functions are not optimized to capture these attacks. Similar to the OLSR protocol analysis in chapter 3, a threat analysis is necessary to study the security threats at the protocols functioning at each layer. The analysis will help to define the features and its associated correlation functions.

Generally, a MAC/Physical layer attack aim to cause Denial of Service to nodes in the attacker's immediate network neighborhood [51]. Denial of Service (DoS) at the MAC/Physical layer causes legitimate users to lose medium access. There are numerous ways for the attacker to make the medium unavailable for the legitimate users. Attack scenarios include, jamming, back-off manipulation, ACK attack, Personal Area Network (PAN) ID conflict, and Guaranteed Time Slot (GTS) attack.

Jamming [109] is one of the most popular methods used by attackers to cause DoS to legitimate users to lose access. Jamming can be done using the MAC protocol or by exploiting the physical channel. At the MAC layer, jamming is done by exploiting the MAC protocol logics. At the physical layer, jamming is done by transmitting high power signal at the communicating frequency to interfere the medium.

Jamming and other MAC/Physical layer attack scenarios are well studied in the literature [36]. Similarly, the reactive and preventive measures against these attacks are also well studied [109]. However, the challenges of detecting these attack behavior in ad hoc network are yet to be analyzed. In our future work, we aim to comprehensively analyze the protocol vulnerabilities and its associated attacks. The security threat analysis on ad hoc MAC layer protocols and physical channel conditions is essential to define features that capture attacks such as, jamming and other attack scenarios.

8.2.2 Application/Transport Layer Attacks

Application/Transport layer attacks are very common in legacy networks. A major part of network security research has been contributed to prevent and detect these types of attacks. Numerous attack scenarios are presented in the literature and features for detecting attacks over application/transport protocols are well defined in the literature [8]. As ad hoc networks use the same application/transport layer protocols that are used in legacy networks, the features used to detect attacks in conventional networks can be used. However, integrating these features into the cross layer detection system has to be addressed. The attack behavior over Application/transport layer protocols is different from routing, MAC and physical layer attacks. Attacks over application/transport layer protocols target to steal or corrupt '*data*', whereas, attacks over lower layer protocols target to disrupt '*service*'. Thus, at the application/transport layer data communication needs to be secured, whereas at the lower layers, the network service needs to be secured. Generalizing the attack behavior between these two broad classifications of attack behavior will be challenging.

The process of integrating the detection of multi protocol attacks in a single security architecture will encompass of adding new statistical features to the feature set from the application and transport layer protocols. Adding new features will

CHAPTER 8. CONCLUSION

increase the number of features in the features set. This will increase the computational complexity of the detection algorithm. Hence, it will be necessary to define new correlation functions to reduce the feature set to maintain the low computational complexity of the system. The correlation functions should capture the common properties of the attacks at different protocols at various layers. Thus, the security threat analysis should aim to explore the common properties that is found among attack from different protocols at various layers.

Appendix A

List of Cross Layer Features

	Observed Duration	D
1	Node's presence in other Neighbor Hello Messages	P_{NET}
2	Difference in number of neighbors declared in consecutive hello messages	HD_{NET}
3	Current Neighbors	CN_{NET}
4	Current Two-Hop Neighbors	$C2N_{NET}$
5	Declared Neighbors	DN_{NET}
6	Declared Two-hop neighbors	$D2N_{NET}$
7	Difference in MPR sequence number	$DFMPR_{NET}$
8	Difference in Sequence Number between routing events	$DSNUM_{NET}$
9	Delay between hello messages	$DELHM_{NET}$
10	Delay between TC Messages	$DELTC_{NET}$
11	Number of Hello Messages	NH_{NET}
12	Number of TC Messages	NTC_{NET}
13	Average Neighbors Declared in Hello messages	$AVGDN_{NET}$
14	Average MPRS Declared in TC Messages	$AVGMPR_{NET}$

Appendix A

15	Average Two-hop Neighbors	$AVG2N_{NET}$
16	Number of MPRS	$MPRS_{NET}$
17	Neighbors	N_{MAC}
18	Nodes that Forward Data	F_{MAC}
19	Recent MAC Transmissions	R_{MAC}
20	Number of Recent Forwards	NF_{MAC}
21	Delay between Neighbor node forwards	$DELNF_{MAC}$
22	Delay between MAC Transmission	$DELTX_{MAC}$
23	Average MAC Neighbors	$AVGN_{MAC}$
24	Average MAC Forwarders	$AVGF_{MAC}$
25	Average Number of Neighbors Forward	$AVGNF_{MAC}$
26	Average Delay between node forwards	$AVGDELF_{MAC}$
27	Delay between broadcast messages	$BCDEL_{MAC}$
28	Nodes that forward broadcast information	BCF_{MAC}
29	Neighbors nodes which only broadcasts	BCN_{MAC}
30	Broadcast Data Size	$BCRATE_{MAC}$
31	Difference in time between unicast and broadcast transmission	$DIFBCUNI_{MAC}$
32	Number of broadcasts	NB_{MAC}
33	Average Broadcast Delay	$AVGBCDEL_{MAC}$
34	Transmission Rate	TX_{PHY}
35	Reception Rate	RX_{PHY}
36	Current Reception Duration	$RECP_{PHY}$
37	Current Signal Interference	CSI_{PHY}

Appendix A

38	Current Signal Strength	CSS_{PHY}
39	Difference between consecutive Signal Strength	$DIFSS_{PHY}$
40	Angle of Arrival	AOA_{PHY}
41	Difference between consecutive Angle of Arrival	$DAOA_{PHY}$
42	Number of physical layer transmissions	NT_{PHY}
43	Average Physical Layer Reception Duration	$AVGDUR_{PHY}$
44	Average Channel Interference	$AVGCI_{PHY}$
45	Average Signal Strength	$AVGSS_{PHY}$
46	Average Difference in Angle of Arrival	$AVGAOA_{PHY}$

Appendix A

List of Derived Features

D1 $D/(RX_{PHY}/R_{MAC})AVGDUR_{PHY}$

RX_{PHY}/R_{MAC} give the average physical channel reception rate. $AVGDUR_{PHY}$ is the physical channel activity duration for receiving MAC protocol communications. Thus, $(RX_{PHY}/R_{MAC})AVGDUR_{PHY}$ is the effective reception rate (utilized reception bandwidth) of the physical channel. The obtained effective reception rate is scaled using the observation period (D).

D2 $(AVGN_{MAC} - HD_{NET})/P_{NET}$

HD_{NET} is an indicator of number of mobile nodes in the neighbourhood. Hence, $AVGN_{MAC} - HD_{NET}$ give the number of stationary or less mobile nodes which do not leave the neighbourhood during the observation period. Thus, $(AVGN_{MAC} - HD_{NET})/P_{NET}$ is the ratio of number of less mobile nodes and average presence of neighbors in other neighbors' hello messages.

D3 $(NF_{MAC}/N_{MAC})/(CN_{NET} + C2N_{NET}) \times (1/AVGDN_{NET})$

$CN_{NET} + C2N_{NET}$ is the total number of nodes in two hop neighbourhood. NF_{MAC}/N_{MAC} is the forwarding rate of the node. Hence, the feature gives the ratio of average traffic density with the total number of nodes in network.

D4 $DELHM_{NET}/DSNUM_{NET}$

The ratio in the feature indicates the mobility of neighbour nodes. In a benign communication, $DELHM_{NET}$ will be proportional to $DSNUM_{NET}$.

D5 $(1/(N_{MAC} - BCN_{MAC})) \times ((DN_{NET} - N_{MAC}) + D2N_{NET})$

The feature is the ratio of number of two-hop neighbours and active number of neighbours. Thus, gives the average number of link connectivity of neighbours to two-hop neighbour nodes.

D6 $(||N_{MAC} - N_{NET}||)((T_{MAC} - T_{NET})^2/(T_{MAC} \times T_{NET}))$

Explained in section 5.1.3.1

Appendix A

D7 $(1 - DFMPR_{NET}) \times (NF_{MAC}/F_{MAC})$

The feature finds the ratio of mobility with average forwarding rate of nodes in neighborhood.

D8 $(DELTC_{NET} - AVGDEL F_{MAC}) \times (DELNF_{MAC} - AVGNF_{MAC})$

The feature finds the dynamicity of the node's neighbourhood.

D9 $DELTX_{MAC}/DELHM_{NET}$

Delay differences between routing communication and MAC communication.

D10 $AVGF_{MAC} - MP RS_{NET}$

Difference between the number of forwarders observed by the MAC protocol communication and the number forwarders (MPR) declared by the neighbour.

D11 $(DELHM_{NET} + DELTC_{NET})/BCDEL_{MAC}$

Average delay between routing control messages (broadcasted)

D12 $(1/BCF_{MAC}) \times (N_{MAC} - DN_{NET}) \times (CN_{NET} - DN_{NET})$

The ratio of the square of neighbors and active forwarders

D13 $(1/AVGBCDEL_{MAC}) \times ((DELHM_{NET} + DELTC_{NET})/NB_{MAC})$

The ratio of delay between routing messages and total delay in MAC transmissions during the observation period

D14 $(BCRATE_{MAC}/(CN_{NET} + C2N_{NET})) \times (N_{MAC} - CN_{NET})$

Broadcasting rate scaled with the number of active number of neighbors

D15 $DIFBCUNI_{MAC}/CN_{NET}$

Average unicast data traffic density

D16 $((RX_{PHY}/RECP_{PHY})/(1/R_{MAC}))/((1/AVGDUR_{PHY})$

Effective MAC protocol reception rate

D17 $((AVGCI_{PHY} - CSI_{PHY}) \times (AVGSS_{PHY} - CSS_{PHY})/(CSI_{PHY} - DIFSS_{PHY}))$

Signal Strength variations between consecutive observation periods.

D18 $(AOA_{PHY} - AVGAOA_{PHY})/DAOA_{PHY}$

The feature finds the mobility of neighbor nodes with respect to the AoA of the signal.

Publications

Journals

1. J.F.C, Joseph, Bu-Sung Lee, Amitabha Das, Seet Boot Chong, "*Cross Layer Detection of Sinking Behavior in Wireless Ad hoc Networks using SVM and FDA*", IEEE Transactions on Dependable and Secure Computing, *Accepted*.
2. J.F.C, Joseph, Amitabha Das, Bu-Sung Lee, Seet Boot Chong, "*CARRADS: Cross Layer based Adaptive Real-time Routing Attack Detection System for MANETS*", Elsevier Computer Networks Journal, *Accepted*.
3. J.F.C, Joseph, Amitabha Das, Seet Boot Chong, Bu-Sung Lee, "*Opening the Pandora's Box: Exploring the Limitations of Intrusion Detection for Ad hoc Routing Protocols*", Volume 31, Issue 14, September 2008, Pages 3178-3189. Computer Communications Journal, Elsevier.

Conference

4. J.F.C, Joseph, Amitabha Das, Seet Boot Chong, Bu-Sung Lee, "*CRADS: Integrated Cross Layer approach for Detecting Routing Attacks in MANETs*". In the Proceedings of IEEE Wireless Communications and Networking Conference (WCNC) 2008. Las Vegas, Nevada, USA.

Publications

5. J.F.C, Joseph, Amitabha Das, Seet Boot Chong, Bu-Sung Lee, "*Cross Layer versus Single Layer Approaches for Intrusion Detection in MANETs*". In the Proceedings of International Conference on Networking (ICON) 2007, Adelaide, Australia.

Book Chapters

6. J.F.C, Joseph, Amitabha Das, Seet Boot Chong, Bu-Sung Lee, "*Routing Attacks in Ad-hoc Routing Protocols*". in Handbook of Wireless Ad Hoc and Sensor Networks, Springer London.
7. J.F.C, Joseph, Amitabha Das, Seet Boot Chong, Bu-Sung Lee, "*Intrusion Detection Systems in Ad hoc Networks*", in Handbook of Research on Wireless Security, Idea Reference Inc. Jan 2008.

References

- [1] Kddcup 1999 data, <http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html>, 1999.
- [2] Qualnet version 3.9.5, www.scalable-networks.com, 2005.
- [3] G. Acs, L. Buttyan, and I. Vajda. Provably secure on-demand source routing in mobile ad hoc networks. *Mobile Computing, IEEE Transactions on*, 5(11):1533–1546, 2006.
- [4] P. Agarwal, B. S. Yadav, and J. Chandra. Statistical analysis based efficient decentralized intrusion detection scheme for mobile ad hoc networks. In *Proceedings of the 16th International Conference on Networks, ICON 2008*, pages 1–6, New Delhi, India.
- [5] S. Alampalayam, A. Kumar, and S. Srinivasan. Mobile ad hoc network security - a taxonomy. In *Proceedings of the 7th International Conference on Advanced Communication Technology, ICACT 2005*, volume 2, pages 839–844, Phoenix Park, Korea, 2005.
- [6] J. P. Anderson. Computer security threat monitoring and surveillance. Technical Report 98–17, James P Anderson Co., FortWashington, Pennsylvania,USA, April 1980.

REFERENCES

- [7] Smith. Andrew, B. An examination of an intrusion detection architecture for wireless ad hoc networks. In *Proceedings of the 5th National Colloquium for Information System Security Education*, pages 120–125, Germany, May 2001.
- [8] Farooq Anjum. and Petros Mouchtaris. *Security for Wireless Ad hoc Networks*. Wiley, New Jersey, USA, 2007.
- [9] Mohan. Atreya. *Digital signatures*. McGraw-Hill/Osborne, Berkeley, USA, 2002.
- [10] B. Awerbuch, R. Curtmola, D. Holmer, H. Rubens, and C. Nita-Rotaru. On the survivability of routing protocols in ad hoc wireless networks. In *Proceedings of the First International Conference on Security and Privacy for Emerging Areas in Communications Networks, SecureComm 2005*, volume 2005, pages 327–338, Athens, Greece, 2005.
- [11] Baruch Awerbuch., Reza Curtmola., David Holmer., Cristina Nita-Rotaru., and Herbert Rubens. Mitigating byzantine attacks in ad hoc wireless networks. Technical report, Department of Computer Science, John Hopkins University, Baltimore, USA, 2004.
- [12] Stefan Axelsson. The base-rate fallacy and the difficulty of intrusion detection. *ACM Trans. Inf. Syst. Secur.*, 3(3):186–205, 2000.
- [13] Xu Bangnan, S. Hischke, and B. Walke. The role of ad hoc networking in future wireless communications. *IEEE Network*, 2:1353–1358 vol.2, 2003.
- [14] Dimitri P Bertsekas. *Constrained Optimization and Lagrange Multiplier Methods*. Athena Scientific, Nashua, USA, 1996.
- [15] S. Bhargava and D. P. Agrawal. Security enhancements in aodv protocol for

REFERENCES

- wireless ad hoc networks. In *Proceedings of the IEEE Vehicular Technology Conference*, volume 4, pages 2143–2147, 2001.
- [16] R. Boliac, D. Lozovanu, and D. Solomon. Optimal paths in network games with p players. *Discrete Applied Mathematics*, 99(1-3):339–348, 2000. Cited By (since 1996): 10 Export Date: 10 November 2009 Source: Scopus.
- [17] S. Bouam and J. Ben-Othman. Data security in ad hoc networks using multi-path routing. volume 2, pages 1331–1335 vol.2, 2003.
- [18] Azzedine. Boukerche. *Algorithms and protocols for wireless and mobile ad hoc networks*. Wiley, New Jersey, USA, 2009.
- [19] M. Brownfield, Yatharth Gupta, and N. Davis. Wireless sensor network denial of sleep attack. In *Proceedings of the Sixth Annual IEEE Systems, Man and Cybernetics (SMC) Information Assurance Workshop, 2005.*, pages 356–364, West Point, New York, USA, 2005.
- [20] Christopher J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167, 1998.
- [21] M. Burmester and B. D. Medeiros. On the security of route discovery in manets. *IEEE Transactions on Mobile Computing*, 8(9):1180–1188, 2009.
- [22] J. B. D. Cabrera, C. Gutierrez, and R. K. Mehra. Infrastructures and algorithms for distributed anomaly-based intrusion detection in mobile ad-hoc networks. pages 1831–1837 Vol. 3, 2005.
- [23] J. Cannady. Distributed detection of attacks in mobile ad hoc networks using learning vector quantization. In *Proceedings of the Third International Conference on Network and System Security, (NSS)*, pages 571–574, Gold Coast, Australia, Oct 2009.

REFERENCES

- [24] Chih-Chung Chang and Chih-Jen Lin. Libsvm: A library for support vector machines, software available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm>, 2001.
- [25] Jin-Hee Cho and Ing-Ray Chen. On design tradeoffs between security and performance in wireless group communicating systems. In *Proceedings of the 1st IEEE ICNP Workshop on Secure Network Protocols (NPSec)*, pages 13–18, Boston, USA, 2005.
- [26] T. Clausen and P. Jacquet. Optimized link state routing protocol. *RFC 3626*, October 2003.
- [27] J. Cordasco and S. Wetzel. Cryptographic versus trust-based methods for manet routing security. *Electronic Notes in Theoretical Computer Science*, 197(2):131–140, 2008.
- [28] F. Cuppens and R. Ortalo. Lambda: A language to model a database for detection of attacks. In *Proceedings of the Recent Advances in Intrusion Detection, Proceedings*, volume 1907 of *Lecture Notes in Computer Science*, pages 197–216. Springer-Verlag, Berlin, Germany, 2000.
- [29] Hongmei Deng, Qing-An Zeng, and D.P. Agrawal. Svm-based intrusion detection system for wireless ad hoc networks. *Proceedings of the 58th IEEE Vehicular Technology Conference (VTC), 2003.*, 3:2147–2151 Vol.3, 2003.
- [30] Dorothy E. Denning. An intrusion detection model. *IEEE Transactions on Software Engineering*, 13(2):222–232, 1987.
- [31] Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification*. Wiley Inter-Science Publication, NY, USA, second edition, 2000.
- [32] H. Ebel and S. Bornholdt. Coevolutionary games on networks. *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, 66(5):056118/1–056118/8,

REFERENCES

2002. Cited By (since 1996): 68 Export Date: 10 November 2009 Source: Scopus Art. No.: 056118.
- [33] S. Eichler and C. Roman. Challenges of secure routing in manets: A simulative approach using aodv-sec. In *Proceedings of the 3rd IEEE International Conference on Mobile Ad Hoc and Sensor Systems, MASS*, pages 481–484, Vancouver, Canada, 2006.
- [34] Lou. Goble. *The Blackwell guide to philosophical logic*. Wiley-Blackwell Publishers, CA, USA, 2001.
- [35] T. Gross and B. Blasius. Adaptive coevolutionary networks: A review. *Journal of the Royal Society Interface*, 5(20):259–271, 2008. Cited By (since 1996): 39 Export Date: 10 November 2009 Source: Scopus.
- [36] Lei Guang, Chadi Assi, and Yinghua Ye. Dream: A system for detection and reaction against mac layer misbehavior in ad hoc networks. *Computer Communications*, 30(8):1841–1853, 2007. 0140-3664 doi: DOI: 10.1016/j.comcom.2007.02.014.
- [37] D. A. Hojman and A. Szeidl. Endogenous networks, social games, and evolution. *Games and Economic Behavior*, 55(1):112–130, 2006. Export Date: 10 November 2009 Source: Scopus.
- [38] F. Hu and N. K. Sharma. Security considerations in ad hoc sensor networks. *Ad Hoc Networks*, 3(1):69–89, 2005.
- [39] Y. C. Hu and A. Perrig. A survey of secure wireless ad hoc routing. *IEEE Security and Privacy*, 2(3):28–39, 2004.
- [40] Y. C. Hu, A. Perrig, and D. B. Johnson. Ariadne: A secure on-demand routing protocol for ad hoc networks. *Wireless Networks*, 11(1-2):21–38, 2005.

REFERENCES

- [41] Li Huaizhi and Singhal Mukesh. Trust management in distributed systems. *Computer*, 40(2):45–53, 2007.
- [42] Y. A. Huang, W. Fan, W. Lee, and P. S. Yu. Cross-feature analysis for detecting ad-hoc routing anomalies. In *Proceedings of the 23rd International Conference on Distributed Computing Systems*, pages 478–487, Rhode Island, USA, 2003.
- [43] Y. A. Huang and W. Lee. Attack analysis and detection for ad hoc routing protocols. In *Proceeding from the 2004 International Symposium on Recent Advances in Intrusion Detection RAID 2004*, volume 3224, pages 125–145, Valbonne, France, 2004.
- [44] J. P. Hubaux. New research challenges for the security of ad hoc and sensor networks. In *Lecture Notes in Computer Science*, volume 3313, page 1. Springer Verlag, Germany, 2005.
- [45] Jean-Pierre Hubaux., Levente Buttyan., and Srdan Capkun. The quest for security in mobile ad hoc networks. In *Proceedings of the 2nd ACM international Symposium on Mobile Ad hoc Networking & Computing*, pages 146–155, Long Beach, CA, USA, 2001. ACM Press.
- [46] A. J. Jameela. Security issues at the network layer in wireless mobile ad hoc networks. In *Proceedings of the 2005 International Conference on Wireless Networks, ICWN'05*, pages 576–582, 2005.
- [47] I. T. Jolliffe. *Principal component analysis*. Springer series in statistics. Springer, NY, USA, 2nd edition, 2002.
- [48] O. Kachirski and R. Guha. Intrusion detection using mobile agents in wireless ad hoc networks. In *Proceedings of the IEEE Workshop on Knowledge Media Networking*, pages 153–158, Kyoto, Japan, 2002.

REFERENCES

- [49] B. Kannhavong, H. Nakayama, N. Kato, A. Jamalipour, and Y. Nemoto. A study of a routing attack in olsr-based mobile ad hoc networks. *International Journal of Communication Systems*, 20(11):1245–1261, 2007.
- [50] Randy H. Katz. *Contemporary logic design*. Pearson Prentice Hall, New Jersey, USA, 2005.
- [51] Yongjin Kim and Ahmed Helmy. Catch: A protocol framework for cross-layer attacker traceback in mobile multi-hop networks. *Ad Hoc Networks*, 8(2):193–213. 1570-8705 doi: DOI: 10.1016/j.adhoc.2009.07.002.
- [52] K. Komathy and P. Narayanasamy. Best neighbor strategy to enforce cooperation among selfish nodes in wireless ad hoc network. *Computer Communications*, 30(18):3721–3735, 2007.
- [53] K. Komathy and P. Narayanasamy. Trust-based evolutionary game model assisting aodv routing against selfishness. *Journal of Network and Computer Applications*, 31(4):446–471, 2008.
- [54] J. Kong, X. Hong, and M. Gerla. A new set of passive routing attacks in mobile ad hoc networks. In *Proceedings of the IEEE Military Communications Conference, (MILCOM 2003)*., volume 2, pages 796–801 Vol.2, 2003.
- [55] A. Kun and I. Scheuring. Evolution of cooperation on dynamical graphs. *BioSystems*, 96(1):65–68, 2009. Export Date: 10 November 2009 Source: Scopus.
- [56] S. Kurosawa, H. Nakayama, Nei Kato, A. Jamalipour, and Y. Nemoto. A self-adaptive intrusion detection method for aodv-based mobile ad hoc networks. In *Proceeding of the IEEE International Conference on Mobile Adhoc and Sensor Systems Conference.*, pages 773–780, Washington, USA, 2005.

REFERENCES

- [57] L. Lamport, R. Shostak, and M. Pease. The byzantine generals problem. *Acm Transactions on Programming Languages and Systems*, 4(3):382–401, 1982.
- [58] Zhou Lidong and Z. J. Haas. Securing ad hoc networks. *Network, IEEE*, 13(6):24–30, 1999.
- [59] Yu Liu, Yang Li, and Hong Man. Mac layer anomaly detection in ad hoc networks. In *Proceedings from the Sixth Annual IEEE Systems, Man and Cybernetics (SMC) Information Assurance Workshop.*, pages 402–409, New York, USA, 2005.
- [60] Yu Liu, Yang Li, and Hong Man. Short paper: A distributed cross-layer intrusion detection system for ad hoc networks. In *Proceedings from the First International Conference on Security and Privacy for Emerging Areas in Communications Networks, (SecureComm).*, pages 418–420, Athens, Greece, 2005.
- [61] P. Mell, V. Hu, R. Lippmann, J. Haines, and M. Zissman. An overview of issues in testing intrusion detection systems. Technical Report NIST IR 7007, National Institute of Standard and Technology, 2003.
- [62] C.C. Michael and A. Ghosh. Two state-based approaches to program-based anomaly detection. In *Proceedings from the 16th Annual Conference Computer Security Applications (ACSAC).*, pages 21–30, Louisiana, USA, 2000.
- [63] A. Mishra, K. Nadkarni, and A. Patcha. Intrusion detection in wireless ad hoc networks. *IEEE Wireless Communications*, 11(1):48–60, 2004.
- [64] M. Mohi, A. Movaghar, and P.M. Zadeh. A bayesian game approach for preventing dos attacks in wireless sensor networks. In *Proceedings of the International Conference on Communications and Mobile Computing, (CMC)*, volume 3, pages 507 –511, jan. 2009.

REFERENCES

- [65] C. Siva Ram Murthy. and B.S. Manoj. *Ad hoc Wireless Networks: Architecture and Protocols*. Prentice Hall Communications Engineering and Emerging Technologies Series. Pearson Education Inc, New Jersey, USA, 2004.
- [66] K. Nadkarni and A. Mishra. Intrusion detection in manets - the second wall of defense. In *Proceedings of the Industrial Electronics Conference (IECON)*, volume 2, pages 1235–1238, Roanoke, USA, 2003.
- [67] F. Nait-Abdesselam. Detecting and avoiding wormhole attacks in wireless ad hoc networks. *Communications Magazine, IEEE*, 46(4):127–133, april 2008.
- [68] Cristianini Nello and Shawe-Taylor John. *An introduction to support Vector Machines and other kernel-based learning methods*. Cambridge University Press, Cambridge, UK, 2000. 345662.
- [69] Peng Ning and Kun Sun. How to misuse aodv: A case study of insider attacks against mobile ad-hoc routing protocols. *Ad Hoc Networks*, 3(6):795–819, 2005.
- [70] M. A. Nowak. Five rules for the evolution of cooperation. *Science*, 314(5805):1560–1563, 2006. Cited By (since 1996): 187 Export Date: 10 November 2009 Source: Scopus.
- [71] J. M. Orset and A. Cavalli. A security model for olsr manet protocol. In *Proceedings of the IEEE International Conference on Mobile Data Management*, Nara, Japan, 2006.
- [72] Panagiotis Papadimitratos. and Zygmunt Haas. Secure routing for mobile ad hoc networks. In *SCS Communication Networks and Distributed Systems Modeling and Simulation Conference (CNDS 2002)*, San Antonio, TX, USA, 2002.
- [73] Albers. Patrick, Camp. Olivier, Percher. Jean-Marc, Jouga. Bernard, Me. Ludovic, and Puttini. Ricardo. Security in ad hoc networks: a general intrusion

REFERENCES

- detection architecture enhancing trust based approaches. In *Proceedings of the 1st International Workshop on Wireless Info. Sys.*, Ciudad Real, Spain, Apr 3-6 2002.
- [74] A. Patwardhan, J. Parker, M. Iorga, A. Joshi, T. Karygiannis, and Y. Yesha. Threshold-based intrusion detection in ad hoc networks and secure aodv. *Ad Hoc Networks*, 6(4):578–599, 2008.
- [75] A. Patwardhan, J. Parker, A. Joshi, M. Iorga, and T. Karygiannis. Secure routing and intrusion detection in ad hoc networks. *Proceedings of the Third IEEE International Conference on Pervasive Computing and Communications, (PerCom).*, pages 191–199, 2005.
- [76] Laskov Pavel, Gehl Christian, Kr Stefan, M. Klaus-Robert, and Iler. Incremental support vector learning: Analysis, implementation and applications. *J. Mach. Learn. Res.*, 7:1909–1936, 2006.
- [77] A. A. Pirzada and C. McDonald. Trust establishment in pure ad-hoc networks. *Wireless Personal Communications*, 37(1-2):139–168, 2006.
- [78] P.A. Porras and R.A. Kemmerer. Penetration state transition analysis: A rule-based intrusion detection approach. In *Proceedings of the Eighth Annual Computer Security Applications Conference*, pages 220–229, USA, 1992.
- [79] A. Quilliot, F. Bendali, and J. Mailfert. Extended cooperative networks games. *Discrete Applied Mathematics*, 157(4):728–737, 2009. Export Date: 10 November 2009 Source: Scopus.
- [80] P. Ramachandran and A. Yasinsac. Limitations of on demand secure routing protocols. In *Proceedings of the Fifth Annual IEEE System, Man and Cybernetics Information Assurance Workshop, SMC*, pages 52–59, The Hague, Netherlands, 2004.

REFERENCES

- [81] R. Ramanujan, A. Ahamad, J. Bonney, R. Hagelstrom, and K. Thurber. Techniques for intrusion-resistant ad hoc routing algorithms (tiara). volume 2, pages 660–664 vol.2, 2000.
- [82] K. Sanzgiri, D. LaFlamme, B. Dahill, B. N. Levine, C. Shields, and E. M. Belding-Royer. Authenticated routing for ad hoc networks. *IEEE Journal on Selected Areas in Communications*, 23(3):598–610, 2005.
- [83] S. Sen and J. A. Clark. Evolving intrusion detection rules on mobile ad hoc networks, 2008.
- [84] Marti Sergio, T. J. Giuli, Lai Kevin, and Baker Mary. Mitigating routing misbehavior in mobile ad hoc networks. In *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking, MobiCom*, Boston, USA, 2000. ACM Press.
- [85] G. Shafer. *A Mathematical Theory of Evidence*. Princeton University Press, NJ, USA, 1976.
- [86] Ning Song, Lijun Qian, and Xiangfang Li. Wormhole attacks detection in wireless ad hoc networks: A statistical analysis approach. In *Proceedings of the 19th IEEE International Parallel and Distributed Processing Symposium, 2005.*, page 289, Denver, USA, 2005.
- [87] Eugene H. Spafford and Diego Zamboni. Intrusion detection using autonomous agents. *Computer Networks*, 34(4):547–570, 2000.
- [88] SPSSInc. Spss base 8.0 for windows user’s guide., 1998.
- [89] William Stallings. *Cryptography and network security: Principles and practice*. Pearson/Prentice Hall, NJ, USA, 2006.

REFERENCES

- [90] S. Staniford-Chen, S. Cheung, R. Crawford, M. Dilger, J. Frank, J. Hoagland, K. Levitt, C. Wee, R. Yip, and D. Zerkle. Grids - a graph-based intrusion detection system for large networks. In *Proceedings of the 19th National Information Systems Security Conference*, pages 361–370, 1996.
- [91] B. Sun, K. Wu, and U. W. Pooch. Towards adaptive intrusion detection in mobile ad hoc networks. In *Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM)*, volume 6, pages 3551–3555, Dallas, USA, 2004.
- [92] B. Sun, K. Wu, and U.W. Pooch. Routing anomaly detection in mobile ad hoc networks. In *Proceedings of the 12th International Conference on Computer Communications and Networks, (ICCCN)*, pages 25–31, Dallas, USA, 2003.
- [93] G. Szabo and G. Fath. Evolutionary games on graphs. *Physics Reports*, 446(4-6):97–216, 2007. Cited By (since 1996): 128 Export Date: 10 November 2009 Source: Scopus.
- [94] J. Tanimoto. Promotion of cooperation through co-evolution of networks and strategy in a 2 x 2 game. *Physica A: Statistical Mechanics and its Applications*, 388(6):953–960, 2009. Export Date: 10 November 2009 Source: Scopus.
- [95] G. Thamilarasu, A. Balasubramanian, S. Mishra, and R. Sridhar. A cross-layer based intrusion detection approach for wireless ad hoc networks. In *Proceedings of the IEEE International Conference on Mobile Adhoc and Sensor Systems Conference, MASS.*, pages 855–861, Washington, USA, 2005.
- [96] G. Theodorakopoulos and J. S. Baras. Trust evaluation in ad-hoc networks. In *Proceedings of the 2004 ACM Workshop on Wireless Security, WiSe*, pages 1–10, 2004.

REFERENCES

- [97] G. Theodorakopoulos and J. S. Baras. A game for ad hoc network connectivity in the presence of malicious users. In *Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM)*, San Francisco, USA, 2006.
- [98] G. Theodorakopoulos and J. S. Baras. On trust models and trust evaluation metrics for ad hoc networks. *IEEE Journal on Selected Areas in Communications*, 24(2):318–328, 2006.
- [99] George J. Toulakis. *Lectures in logic and set theory: Mathematical logic*. Cambridge University Press., Cambridge, UK, 2008.
- [100] H. C. Tseng and B. J. Culpepper. Sinkhole intrusion in mobile ad hoc networks: The problem and some detection indicators. *Computers & Security*, 24(7):561–570, 2005.
- [101] F. Tutzauer, M. K. Chojnacki, and P. W. Hoffmann. Network structure, strategy evolution, and the game of chicken. *Social Networks*, 28(4):377–396, 2006. Cited By (since 1996): 2 Export Date: 10 November 2009 Source: Scopus.
- [102] Aljaz Ule. *Partner Choice and Cooperation in Networks: Theory and Experimental Evidence*. Lecture Notes in Economics and Mathematical Systems. Springer Berlin Heidelberg, Germany, 2008.
- [103] V.N. Vapnik. *Statistical Learning Theory*. Wiley, NY, USA, 1998.
- [104] G. Varaprasad and P. Venkataram. The analysis of secure routing in mobile adhoc network. In *Proceedings of the International Conference on Computational Intelligence and Multimedia Applications, ICCIMA 2007*, volume 4, pages 393–397, Sivakasi, India, 2008.
- [105] P. Veeraraghavan and V. Limaye. Trust in mobile ad hoc networks. In *Proceeding of the 2007 IEEE International Conference on Telecommunications*

REFERENCES

- and Malaysia International Conference on Communications, ICT-MICC 2007*, pages 330–334, Penang, Malaysia, 2007.
- [106] G. Vigna, S.T. Eckmann, and R.A. Kemmerer. The stat tool suite. In *Proceedings of the DARPA Information Survivability Conference and Exposition, (DISCEX)*, volume 2, pages 46–55 vol.2, South Carolina, USA, 2000.
- [107] D. Wang, M. Hu, and H. Zhi. A survey of secure routing in ad hoc networks. In *Proceedings of the 9th International Conference on Web-Age Information Management, WAIM 2008*, pages 482–486, Zhangjiajie, China, 2008.
- [108] H. Wang, Y. Wang, and J. Han. A security architecture for tactical mobile ad hoc networks. In *Proceedings of the 2nd International Workshop on Knowledge Discovery and Data Mining, WKDD 2009*, pages 312–315, Moscow, Russia, 2009.
- [109] Xu Wenyuan, Trappe Wade, Zhang Yanyong, and Wood Timothy. The feasibility of launching and detecting jamming attacks in wireless networks. In *Proceedings of the 6th ACM International Symposium on Mobile Ad hoc Networking and Computing, MobiHoc 2005*, Urbana-Champaign, USA, 2005. ACM. 46-57.
- [110] K. D. Wong, T. J. Kwon, and V. Varma. Towards commercialization of ad hoc networks. 2:1207–1211, 2004.
- [111] H. B. Yang, X. Y. Zhang, and Y. Y. Wang. Correctness analysis of manet routing protocols. *Nanjing Youdian Daxue Xuebao (Ziran Kexue Ban)/Journal of Nanjing University of Posts and Telecommunications (Natural Science)*, 27(2):15–19, 2007.
- [112] Shahan Yang. and John S. Baras. Modeling vulnerabilities of ad hoc routing

REFERENCES

- protocols. In *Proceedings of the 1st ACM workshop on Security of ad hoc and sensor networks*, pages 12–20, Fairfax, Virginia, USA, 2003. ACM Press.
- [113] Liu Yu, Li Yang, and Man Hong. Short paper: A distributed cross-layer intrusion detection system for ad hoc networks,. In *Proceedings of the First International Conference on Security and Privacy for Emerging Areas in Communication Networks, SecureComm*, pages 418–420, Athens, Greece, 2005.
- [114] M. G. Zapata and N. Asokan. Securing ad hoc routing protocols. In *Proceedings of the Workshop on Wireless Security*, pages 1–10, Atlanta, USA, 2002.
- [115] Da Zhang and Chai Kiat Yeo. A novel architecture of intrusion detection system. In *Proceedings of the 7th IEEE Consumer Communications and Networking Conference (CCNC)*, pages 1–5, Las Vegas, USA, jan 2010.
- [116] W. Zhang, R. Rao, G. Cao, and G. Kesidis. Secure routing in ad hoc networks and a related intrusion detection problem. In *Proceedings of the IEEE Military Communications Conference, (MILCOM)*, volume 2, pages 735–740 Vol.2, Boston, USA, 2003.
- [117] Y. G. Zhang, W. K. Lee, and Y. A. Huang. Intrusion detection techniques for mobile wireless networks. *Wireless Networks*, 9(5):545–556, 2003.
- [118] Yongguang Zhang, Yi-an Huang, and Wenke Lee. An extensible environment for evaluating secure manet. In *Proceedings of the First International Conference on Security and Privacy for Emerging Areas in Communications Networks, SecureComm 2005.*, pages 339–352, Athens, Greece, 2005.
- [119] L. Zhou and Z. J. Haas. Securing ad hoc networks. *IEEE Network*, 13(6):24–30, 1999.

REFERENCES

- [120] C. Zouridaki, B. L. Mark, M. Hejmo, and R. K. Thomas. Hermes: A quantitative trust establishment framework for reliable data packet delivery in manets. *Journal of Computer Security*, 15(1):3–38, 2007.
- [121] C. Zouridaki, B. L. Mark, M. Hejmo, and R. K. Thomas. E-hermes: A robust cooperative trust establishment scheme for mobile ad hoc networks. *Ad Hoc Networks*, 7(6):1156–1168, 2009.