

Hybrid Cryptography Based Intrusion Detection System

Submitted in partial fulfillment of requirements for the award of the degree of

MASTER OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

BY

BASEEM ADNAN NADOM AL-TWAIJRE

(ID No. : 12MTCST031)



**SHEPHERD SCHOOL OF ENGINEERING AND TECHNOLOGY,
SAM HIGGINBOTTOM INSTITUTE OF AGRICULTURE
TECHNOLOGY AND SCIENCES**

**Deemed-to-be-University
(FORMERLY ALLAHABAD AGRICULTURE INSTITUTE)
NAINI-ALLAHABAD-211007, (U.P), INDIA**

2014



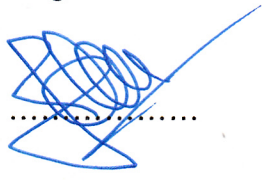
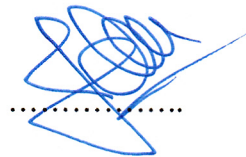
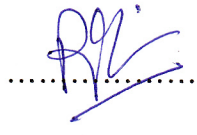
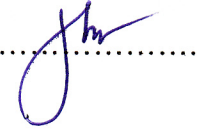
सैम हिगिनबॉटम इन्स्टीट्यूट ऑफ एग्रीकल्चर, टेक्नालॉजी एण्ड साइंसेज
Sam Higginbottom Institute of Agriculture, Technology & Sciences
(Formerly Allahabad Agricultural Institute)
(Deemed to be University)
Allahabad - 211 007, India

ISO 9001:2008 Certified

Office : 91-532-2684281, 2684781
Fax : 91-532-2684394
Website : www.shiats.edu.in
E-mail : info@shiats.edu.in

CERTIFICATE OF ACCEPTANCE OF EVALUATION COMMITTEE

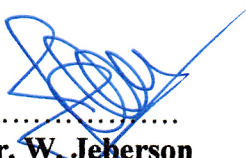
The dissertation entitled, "**Hybrid Cryptography Based Intrusion Detection System**" has been prepared and submitted by **Mr. BASEEM ADNAN NADOM AL-TWAJRE (ID No. : 12MTCST031)** in partial fulfillment of the requirements for the award of the degree of **Master of Technology in Computer Engineering** from Department of Computer Science & Information Technology, Shepherd School of Engineering & Technology at Sam Higginbottom Institute of Agriculture, Technology & Sciences - Deemed University, Allahabad, Uttar Pradesh, and hereby accepted for the award of this degree.

Name and designation	Evaluation	Signature
(1) Dr. W. Jeberson (Head) Assoc. Professor Dept. of CS & IT	<input checked="" type="checkbox"/> Satisfactory <input type="checkbox"/> Not Satisfactory	
(2) Dr. W. Jeberson (Advisor) Assoc. Professor Dept. of CS & IT	<input checked="" type="checkbox"/> Satisfactory <input type="checkbox"/> Not Satisfactory	
(3) Dr. Raghav. Yadav (Member) Assistant Professor Dept. of CS & IT	<input type="checkbox"/> Satisfactory <input checked="" type="checkbox"/> Not Satisfactory	
(4) Er. H.M. Singh (Member) Assistant Professor Dept. of CS & IT	<input checked="" type="checkbox"/> Satisfactory <input type="checkbox"/> Not Satisfactory	

The dissertation has been examined by evaluation committee and found acceptable.

Place: Allahabad

Date: / /2014


.....
Dr. W. Jeberson
(Chairman)
Evaluation Committee



सैम हिगिन्बॉटम इन्स्टीट्यूट ऑफ एग्रीकल्चर, टेक्नालॉजी एण्ड साइंसेज
Sam Higginbottom Institute of Agriculture, Technology & Sciences

(Formerly Allahabad Agricultural Institute)

(Deemed to be University)

Allahabad - 211 007, India

ISO 9001:2008 Certified

Office : 91-532-2684281, 2684781

Fax : 91-532-2684394

Website : www.shiats.edu.in

E-mail : info@shiats.edu.in

DEPARTMENT OF COMPUTER SCIENCE & INFORMATION TECHNOLOGY

EVALUATION REPORT OF M.TECH ORAL DEFFENCE EXAMINATION OF

BASEEM ADNAN NADOM AL-TWAJRE

(12MTCST031)

This is to certify that **Mr. BASEEM ADNAN NADOM AL-TWAJRE** presented the oral defense of his M. Tech dissertation entitled "**Hybrid Cryptography Based Intrusion Detection System**". We, the members of evaluation committee (Board of Examiners) agreed to recommend to the University for the Award of the degree of **MASTER OF TECHNOLOGY IN COMPUTER SCIENCE AND ENGINEERING** to **Mr. BASEEM ADNAN NADOM AL-TWAJRE (12MTCST031)**.

Dr. W. Jeberson

Head & Chairman

(Dept. of Computer Science & IT)

Shepherd School of Engineering and

Technology

SHIATS, Allahabad

Dr. W. Jeberson

Assoc. Professor (Advisor)

(Dept. of Computer Science & IT)

Shepherd School of Engineering and

Technology

SHIATS, Allahabad

Dr. Raghav. Yadav

Assistant Professor (Member)

(Dept. of Computer Science & IT)

Shepherd School of Engineering and

Technology

SHIATS, Allahabad

Er. H.M. Singh

Assistant Professor (Member)

(Dept. of Computer Science & IT)

Shepherd School of Engineering and

Technology

SHIATS, Allahabad

Abstract

A mobile Ad-hoc network (MANET) is a wireless network that does not rely on any fixed infrastructure (i.e., routing facilities, such as wired networks and access points), and whose nodes must coordinate among themselves to determine connectivity and routing. The traditional way of protecting networks is not directly applicable to MANETs. Many conventional security solutions are ineffective and inefficient for the highly dynamic and resource-constrained environments where MANETs use might be expected. Since prevention techniques are never enough, intrusion detection systems (IDSs), which monitor system activities and detect intrusions, are generally used to complement other security mechanisms. So, it is crucial to develop efficient intrusion-detection mechanisms to protect MANET from attacks. To provide high security for information on MANETs, Intrusion Detection Systems (especially that based on acknowledgment packets) used different types of cryptographic algorithms to protect the acknowledgment packets from attacks of intruder. These cryptographic algorithms are required to provide data security and user's authenticity. In this dissertation we proposed an intrusion-detection model to protect MANETs from attacks named Hybrid Cryptography Enhanced Adaptive Acknowledgment (HCEAACK) based on acknowledgment packets. In order to ensure the integrity and high level of security of the proposed model, acknowledgment packets encrypted and digitally signed by its sender before they are sent out and verified by its receiver until they are accepted, for this purpose we use hybrid cryptography Technique. This hybrid cryptography Technique has been used to detect hacking on the acknowledgment packets in intrusion detection systems like RSA hacking problem that appears in other pervious system. We used two algorithms to implement our Hybrid Cryptography technique, Advanced Encryption Standard (AES) algorithm and RSA algorithm. The two algorithms are cooperated to give more security to acknowledgment packets. Compared to contemporary approaches, our proposed model demonstrates higher malicious-behavior-detection rates in certain circumstances while does not greatly affect the network performances.

Keywords - Mobile Ad-hoc Networks (MANETs); Intrusion-Detection Systems (IDS); Hybrid Cryptography Enhanced Adaptive Acknowledgement (HCEAACK).

ACKNOWLEDGEMENT

All thanks and praises are due to **ALLAH**, whom we thank and seek for help and forgiveness. Whom soever **ALLAH** guides, will never be misled and when soever he misguides , will never find someone to guide them .I testify that none has the right to be worshipped , except **ALLAH** , alone without partners , and that **Muhammad** is Allah's slave and messenger .

In completing this thesis, I owe a debt of gratitude and thanks to many persons have supported me throughout this difficult yet challenging journey. While being thankful to all of them, I must register my gratitude to some in particular.

First and foremost, I would like to express my deepest appreciation to my supervisor **Dr. Wilson Jeberson**, Associate Professor for his guidance and consistent support. His knowledgeable, wise and inspiring discussions have guided me through my whole thesis. He brought to my attention aspects that I had to improve and that I continue to work on.

I would also like to take this opportunity to thank Dean of the College **Prof. Dr. Mohammad Imtiyaz** for his consistent help in completing this thesis.

Appreciate to all faculty members at the Department of Computer Science & Information Technology (Shepherd School of Engineering & Technology) especially to my members **Dr. Raghav. Yadav** and Er. **H.M. Singh** for making the supportive work environment.

My extremely grateful to my parents for their never-ending support and encouragement throughout my education. My father **ADNAN NADOM** has been a great and wise teacher in my life and my lovely mother for her infinite patience especially during my absence.

I would like to extend my gratitude to my lovely wife, who has been very patient while pursuing my studies and research. To my dear sons **Husain** and **Hasan**.

I would like to thank my dear brothers **Haider**, **Ali**, **Ahmed** and dear sisters **Zynab** and **Hawraa**.

I would like to express thanks to my colleagues, friends and relatives for their love, patience and understanding while pursuing my studies.

Thanks to everyone in this world who helped me in my education.

Baseem Adnan Nadom AL-Twajre

TABLE OF CONTENTS

Title	Page No.
TITLE PAGE	
UNDERTAKING	
CERTIFICATE OF ORIGINALWORK	
STUDENT ADVISORY COMMITTEE CERTIFCATE	
EVALUATION COMMITTEE REPORT	
ABSTRACT	i
ACKNOWLEDGEMENT	ii
TABLE OF CONTENTS	iv
LIST OF FIGURES	vii
LIST OF TABLES	x
LIST OF ABBREVIATIONS	xi
CHAPTER I INTRODUCTION	
1.1. Wireless Ad Hoc Networks	1
1.2. A Brief History of Wireless Ad Hoc Networks	2
1.3. Applications of Wireless Ad Hoc Networks	4
1.4. Advantages of Wireless Ad Hoc Networks	6
1.5. Challenges in Wireless Ad-hoc Networks	6
1.6 Characteristics of Mobile Ad-hoc Networks	7
1.7. Security Attributes in MANETs	9
1.8. Attacks on Ad-Hoc Networks	11
1.9. Secure Routing and Intrusion Detection	13
1.10. Intrusion Detection in MANETs	14
1.10.1. Approaches to Intrusion Detection	16
1.10.2. Existing IDS Models for MANETs	18
1.11. Disadvantages of ID Existing systems	24
1.12. Justification	24

1.13.	Objectives	25
CHAPTER II	LITERATURE REVIEW	26
CHAPTER III	MATERIALS AND METHODS	
3.1	System Specification	36
3.1.1.	Hardware Requirements	36
3.1.2.	Software Requirements	36
3.2.	Simulation	36
3.3.	Real-world components of a Mobile Ad-hoc network	37
3.4.	Introduction to Network Simulator (NS)	38
3.5.	Main Network Simulator (NS) Simulation Step	43
3.6.	System Design	45
3.7.	Modules	47
3.7.1.	Basic routing Module	47
3.7.2.	Secrete Acknowledgement Module	48
3.7.3.	Secure Acknowledgement (S-ACK) Module	48
3.7.4.	Misbehavior Report Authentication (MRA) Module	48
3.8.	Implementation of Proposed Model	49
3.8.1.	Hybrid Cryptography Technique	49
3.8.2	Algorithm of Proposed Model	51
3.9.	Advantages of Proposed System	54
3.10.	Simulation Configurations and Programming	55
3.11.	Testing and debugging	66
CHAPTER VI	RESULTS AND DISCUSSION	
4.1	Result analysis	74
4.2	Performance Evaluation	86

CHAPTER V	CONCLUSION AND FUTURE WORK	
5.1	Conclusion	90
5.2	Future Work	90
REFERENCES		91
APPENDIX		
	A. Code Network Simulator NS 2.34	95
PAPERS PUBLISHED		100

LIST OF FIGURES

Figure No.	Title	Page No.
1.1	Disaster Relief Operations	5
1.2	Military or Police Exercises	6
1.3	Mobile Ad-hoc Network (MANET)	8
1.4	Ambiguous collisions	19
1.5	Receiver collisions	19
1.6	Limited transmission power	20
1.7	False misbehavior report	20
1.8	TWOACK Model	21
1.9	End-to-End ACK IDS for MANETs ACK Model	22
1.10	System control flow in the EAACK Model	23
1.11	Flow Chart of EAACK Model	23
3.1	Basic Architecture of NS	40
3.2	Stack Registers	41
3.3	Trace File Format	42
3.4	Basic Routing	47
3.5	Misbehavior Report Authentications	49
3.6	Formation of Secrete Acknowledgement	50
3.7	Verification Process at Source Node	51

3.8	Black Box Testing	71
3.9	White Box Testing	72
3.10	Unit Testing	73
4.1	Network Scenario with Source and Destination	75
4.2	Source Node and Destination Node	75
4.3	Route Request Sharing through Intermediate Nodes	76
4.4	Route Reply from Destination	76
4.5	BAODV Model Transmission without any ACK sharing	77
4.6	Data from Source to Destination without any Security	77
4.7	malicious found in the route	78
4.8	ACK- Sharing Between Each Intermediate Nodes	79
4.9	AACK Model (ACK from Destination)	80
4.10	ACK Sharing to source Through Intermediate Nodes	80
4.11	Malicious Node Collects the Data but No ACK to Source	81
4.12	Source Node Initializes the (S-ACK)	81
4.13	Checks the New Route after Finding the Malicious	82
4.14	ACK from Malicious Node	82
4.15	Digital Sign Method for check the ACK Originally	83
4.16	Digital Sign with the Key	83
4.17	Secrete Key used to make more Security for ACK	84

4.18	Encrypted ACK data by AES algorithm	85
4.19	Digital Sign of Secured ACK data by RSA algorithm	85
4.20	Comparison of Overhead (OH)	88
4.21	Comparison of Delay (D)	88
4.22	Comparison of Packet Delivery Factor (PDF)	89

LIST OF TABLES

Table No.	Title	Page No.
Table 1.1	Security attributes in MANETs	10
Table 4.1	Results Table	87

LIST OF ABBREVIATIONS

Abbreviations	Description
AODV	Ad-hoc On-demand Distance Vector routing
CA	Certificate Authority
CBR	Constant Bit Rate
CN _v	Cipher of New Value
CSMA	Carrier Sense Multiple Access
CSMA/CA	Carrier Sense Multiple Access/Collision Avoidance
C _t	Cipher Text
C _{ta}	Cipher Text Converted to ASCII Value
D	Delay
DAR	Distributed Adaptive Service Replication
DARPA	Defense Advanced Research project Agency
DES	Data Encryption Standard
DoS	Denial-of-Service
DRR	Deficit Round Robin
DRR	Deficit Round Robin
El	Empty List for Encryption Data
FMR	False Misbehavior Report
FQ	Fair Queuing
FTP	File Transfer Protocol
GloMo	Global Mobile Information Systems
IETF	Internet Engineering Task Force
IP	Internet Protocol
IPv6	Internet Protocol Version 6
KDC	Key Distribution Center
KS _{s_d}	Pairwise key between Source and Destination
MANET	Mobile Ad-hoc Network
MRA	Misbehavior Report Authentication

NSF	National Science Foundation
NTDR	Near-term Digital Radio
Nv	New Value
OCARI	Optimization of Communication for Ad hoc Reliable Industrial networks
OH	Overhead introduced by the routing protocol
OSI	Open System Interconnection
OTCL	Object Oriented Tool Command Language
PAN	Personal Area Network
PCMCIA	Personal Computer Memory Card International Association cards
PDF	Packet Delivery Factor
PRNET	Packet Radio Networks
Pt	Plain Text
Pu	Public Key
Rand	Random Number
RED	Random Early Discard
RT	Radio Terminal
SFQ	Stochastic Fair Queuing
SUCV	Statistically Unique and Cryptographically Verifiable
SURAN	Survivable Adaptive Radio Network
TA	Trust Authority
Ta	Time of Acknowledgment
TCP	Transfer Control Protocol
TDMA	Time Division Multiple Access
UDP	User Datagram Protocol
VINT	Virtual Internetwork Test bed
WLAN	Wireless Local Area Network
WMN	Wireless Mesh Network
WSN	Wireless Sensor Network

1.1. Mobile Ad-hoc Networks (MANETs)

In the next generation of wireless communication systems, there will be a need for the rapid deployment of independent mobile users. Significant examples include establishing survivable, efficient, dynamic communication for emergency rescue operations, disaster relief efforts, and military networks. Such network scenarios cannot rely on centralized and organized connectivity, and can be conceived as applications of Mobile ad-hoc Networks. A MANET is an autonomous collection of mobile users that communicate over relatively bandwidth constrained wireless links. Since the nodes are mobile, the network topology may change rapidly and unpredictably over time. The network is decentralized, where all network activity including discovering the topology and delivering messages must be executed by the nodes themselves, i.e., routing functionality will be incorporated into mobile nodes.

An ad-hoc network is a collection of wireless mobile hosts forming a temporary network without the aid of any stand-alone infrastructure or centralized administration. Mobile ad-hoc networks are self-organizing and self-re-configuring multi hop wireless networks where, the structure of the network changes dynamically. This is mainly due to the mobility of the nodes. Nodes in these networks utilize the same random access wireless channel, cooperating in a friendly manner to engaging themselves in multi hop forwarding. The nodes in the network not only act as hosts but also as routers that route data to/from other nodes in network.

In mobile ad-hoc networks where there is no infrastructure support as is the case with wireless networks, and since a destination node might be out of range of a source node transmitting packets; a routing procedure is always needed to find a path so as to forward the packets appropriately between the source and the destination. Within a cell, a base station can reach all mobile nodes without routing via broadcast in common wireless networks. In the

case of ad-hoc networks, each node must be able to forward data for other nodes. This creates additional problems along with the problems of dynamic topology which is unpredictable connectivity changes.

MANETS rely on wireless transmission, a secured way of message transmission is important to protect the privacy of the data. An insecure ad-hoc network at the edge of an existing communication infrastructure may potentially cause the entire network to become vulnerable to security breaches. In mobile ad hoc networks, there is no central administration to take care of detection and prevention of anomalies.

Mobile devices identities or their intentions cannot be predetermined or verified. Therefore nodes have to cooperate for the integrity of the operation of the network. However, nodes may refuse to cooperate by not forwarding packets for others for selfish reasons and not want to exhaust their resources. Various other factors make the task of secure communication in ad hoc wireless networks difficult include the mobility of the nodes, a promiscuous mode of operation, limited processing power, and limited availability of resources such as battery power, bandwidth and memory. Therefore nodes have to cooperate for the integrity of the operation of the network. In ad-hoc networks devices (also called nodes) act both as computers and routers. Most routing protocols lead nodes to exchange network topology information in order to establish communication routes. This information is sensitive and may become a target for malicious adversaries who intend to attack the network or the applications running on it.

There are two sources of threats to routing protocols. The first comes from external attackers. By injecting erroneous routing information, replaying old routing information, or distorting routing information, an attacker could successfully partition a network or introduce a traffic overload by causing retransmission and inefficient routing. The second and more severe kind of threat comes from compromised nodes, which might (i) misuse routing information to other nodes or (ii) act on applicative data in order to induce service failures.

The provision of systematic approaches to evaluate the impact of such threats on particular routing protocols remains an open challenge today. Attacks on ad-hoc are classified into non-disruptive passive attacks and disruptive active attacks. The active attacks are further

classified into internal attacks and external attacks are carried out by nodes that do not belong to network and can be prevented by firewalls and encryption techniques. Internal attacks are from internal nodes which are actually authorized nodes and part of the network hence it is difficult to identify.

In the next generation of wireless communication systems, there will be a need for the rapid deployment of independent mobile users. Significant examples include establishing survivable, efficient, dynamic communication for emergency/rescue operations, disaster relief efforts, and military networks. Such network scenarios cannot rely on centralized and organized connectivity, and can be conceived as applications of Mobile Ad Hoc Networks. A MANET is an autonomous collection of mobile users that communicate over relatively bandwidth constrained wireless links. Since the nodes are mobile, the network topology may change rapidly and unpredictably over time. The network is decentralized, where all network activity including discovering the topology and delivering messages must be executed by the nodes themselves, i.e., routing functionality will be incorporated into mobile nodes.

The set of applications for MANETs is diverse, ranging from small, static networks that are constrained by power sources, to large-scale, mobile, highly dynamic networks. The design of network protocols for these networks is a complex issue. Regardless of the application, MANETs need efficient distributed algorithms to determine network organization, link scheduling, and routing. However, determining viable routing paths and delivering messages in a decentralized environment where network topology fluctuates is not a well-defined problem. While the shortest path (based on a given cost function) from a source to a destination in a static network is usually the optimal route, this idea is not easily extended to MANETs. Factors such as variable wireless link quality, propagation path loss, fading, multiuser interference, power expended, and topological changes, become relevant issues. The network should be able to adaptively alter the routing paths to alleviate any of these effects. Moreover, in a military environment, preservation of security, latency, reliability, intentional jamming, and recovery from failure are significant concerns. Military networks are designed to maintain a low probability of intercept and/or a low probability of detection. Hence, nodes prefer to radiate as little power as necessary and transmit as infrequently as possible, thus

decreasing the probability of detection or interception. A lapse in any of these requirements may degrade the performance and dependability of the network.

1.2. A Brief History of Wireless Ad Hoc Networks

The whole life-cycle of ad-hoc networks could be categorized into first, second, and third generation ad-hoc network systems. Present ad-hoc networks systems are considered the third generation.

The first generation of wireless ad-hoc networks dates back to 1972. At the time, they were called PRNET (Packet Radio Networks). In conjunction with ALOHA and CSMA 4(Carrier Sense Multiple Access), approaches for medium access control and a kind of distance-vector routing, PRNET were used on a trial basis to provide different networking capabilities in a combat environment.

The second generation of ad-hoc networks emerged in 1980s, when the ad-hoc network systems were further enhanced and implemented as a part of the SURAN (Survivable Adaptive Radio Networks) program. This provided a packet-switched network to the mobile battlefield in an environment without infrastructure. This program proved to be beneficial in improving the radios' performance by making them smaller, cheaper, and resilient to electronic attacks.

In the 1990s, the concept of commercial ad-hoc networks arrived with notebook computers and other viable communications equipment. At the same time, the idea of a collection of mobile nodes was proposed at several research conferences. The IEEE 802.11 subcommittee had adopted the term "ad-hoc networks" and the research community had started to look into the possibility of deploying ad-hoc networks in other areas of application. Meanwhile, work was going on to advance the previously built ad-hoc networks. GloMo (Global Mobile Information Systems) and the NTDR (Near-term Digital Radio) are results of these efforts. GloMo was designed to provide an office environment with Ethernet-type multimedia connectivity anywhere and anytime in handheld devices. NTDR is the only "real" non-prototypical ad-hoc network that is in use today. It uses clustering and link-state routing, and is self-organized into a two-tier ad-hoc network.

Development of different channel access approaches now in the CSMA/CA and TDMA molds, and several other routing and topology control mechanisms were some of the other inventions of that time.

Later on in mid-1990s, within the Internet Engineering Task Force (IETF), the Mobile ad-hoc Networking working group was formed to standardize routing protocols for ad-hoc networks. The development of routing within the working group and the larger community resulted in the invention of reactive and proactive routing protocols.

Soon after, the IEEE 802.11 subcommittee standardized a medium access protocol that was based on collision avoidance and tolerated hidden terminals, making it usable for building mobile ad-hoc networks prototypes out of notebooks and 802.11 PCMCIA (Personal Computer Memory Card International Association cards). Wireless local area products (IEEE 802.11, Hiperlan) provide in-building wireless access; however, they are usually deployed as access links only, packet relaying being performed by traditional bridges or routers. Bluetooth is a low cost technology for short range communication; its market is targeted towards PCs, phones, appliances, watches, etc. It allows multiple nodes to connect to each other in a multi-hop arrangement.

Efforts are on to standardize different existing schemes for different network controls in a single framework which could be taken as a standard for all the future applications utilizing ad-hoc networks as a networking technology. Wireless devices are getting smaller, cheaper, and more sophisticated. As these devices become more ubiquitous, organizations are looking for inexpensive ways to keep these devices connected. Building an ad-hoc network could make that happen .

Wireless Ad Hoc Networks can broadly be classified into three categories: Mobile ad-hoc networks (MANETs), Wireless Sensor Networks (WSNs), and Wireless Mesh Networks (WMNs). Each one of these has significance for different application areas; each of these differs in the capacity and capabilities of nodes that participate in the network, the purpose of the network and the communication protocols employed. The focus of this desertion is MANETs; from this point onwards, the words MANETs and Wireless ad-hoc Networks will be used interchangeably.

1.3. Characteristics of MANETs

Mobile ad-hoc Network (MANET) is a collection of independent mobile nodes that can communicate to each other via radio waves. The mobile nodes that are in radio range of each other can directly communicate, whereas others need the aid of intermediate nodes to route their packets. These networks are fully distributed, and can work at any place without the help of any infrastructure. This property makes these networks highly flexible and robust.

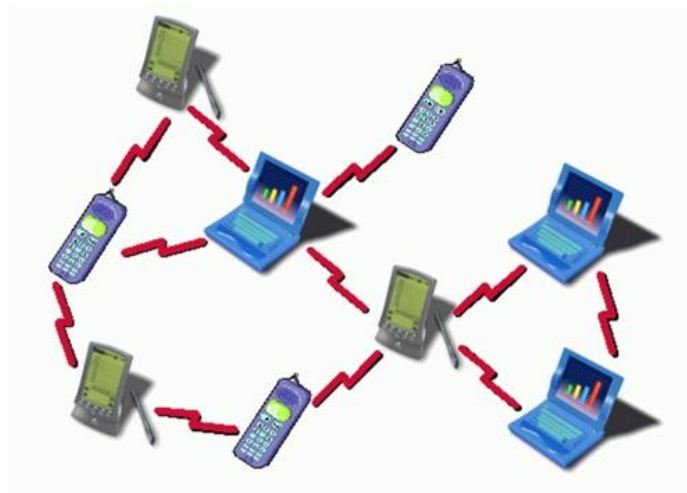


Figure 1.1 Mobile Ad-hoc Network (MANET)

The characteristics of these networks are summarized as follows:

- ❖ Communication via wireless means.
- ❖ Nodes can perform the roles of both hosts and routers.
- ❖ No centralized controller and infrastructure.
- ❖ Intrinsic mutual trust.
- ❖ Dynamic network topology, Frequent routing updates.
- ❖ Autonomous, no infrastructure needed.
- ❖ Can be set up anywhere.
- ❖ Energy constraints.
- ❖ Limited security.

Generally, the communication terminals have a mobility nature which makes the topology of the distributed networks time varying. The dynamical nature of the network topology increases the challenges of the design of ad-hoc networks. Each radio terminal (RT) is usually powered by energy limited power source (as rechargeable batteries). The power consumption of each radio terminal could be divided generally into three parts, power consumption for data processing inside the radio terminal (RT), power consumption to transmit its own information to the destination, and finally the power consumption when the radio terminal (RT) is used as a router, i.e. forwarding the information to another radio terminal (RT) in the network. The energy consumption is a critical issue in the design of the ad-hoc networks.

The mobile devices usually have limited storage and low computational capabilities. They heavily depend on other hosts and resources for data access and information processing. A reliable network topology must be assured through efficient and secure routing protocols for ad-hoc networks.

1.4. Applications of MANETs

There are many applications to ad-hoc networks. As a matter of fact, any day-to-day application such as electronic e-mail and file transfer can be considered to be easily deployable within an ad-hoc network environment. Web services are also possible in case any node in the network can serve as a gateway to the outside world. In this discussion, we need not emphasize the wide range of military applications possible with ad-hoc networks. Not to mention, the technology was initially developed keeping in mind the military applications, such as battlefield in an unknown territory where an infrastructure network is almost impossible to have or maintain. In such situations, the ad-hoc networks having self-organizing capability can be effectively used where other technologies either fail or cannot be deployed effectively. Advanced features of wireless mobile systems, including data rates compatible with multimedia applications, global roaming capability, and coordination with other network structures, are enabling new applications. Some well-known ad-hoc network applications are:

- **Collaborative Work** – For some business environments, the need for collaborative computing might be more important outside office environments than inside. After all, it is often the case where people do need to have outside meetings to cooperate and exchange information on a given project.
- **Crisis-management Applications** – These arise, for example, as a result of natural disasters where the entire communications infrastructure is in disarray. Restoring communications quickly is essential. By using ad-hoc networks, an infrastructure could be set up in hours instead of days/weeks required for wire-line communications.
- **Personal Area Networking and Bluetooth** – A personal area network (PAN) is a short-range, localized network where nodes are usually associated with a given person. These nodes could be attached to someone's pulse watch, belt, and so on. In these scenarios, mobility is only a major consideration when interaction among several PANs is necessary, illustrating the case where, for instance, people meet in real life. Bluetooth is a technology aimed at, among other things, supporting PANs by eliminating the need of wires between devices such as printers, PDAs, notebook computers, digital cameras, and so on, and is discussed later.

1.5. Application Areas of MANETs:

Some of the applications of MANETs are

- ❖ Military or police exercises.
- ❖ Disaster relief operations.
- ❖ Mine site operations.
- ❖ Urgent Business meetings.
- ❖ Robot data acquisition.

It is easy to imagine a number of applications where this type of properties would bring benefits. One interesting research area is inter-vehicle communications. It is one area where the ad-hoc networks could really change the way we communicate covering personal vehicles as well as professional mobile communication needs. Also, it is area where no conventional (i.e. wired) solutions would do because of the high level of mobility. When

considering demanding surroundings, say mines for example, then neither would the base station approach work but we must be able to accomplish routing via nodes that are part of the network i.e. we have to use ad-hoc network.

Such networks can be used to enable next generation of battlefield applications envisioned by the military including situation awareness systems for maneuvering war fighters, and remotely deployed unmanned micro-sensor networks. Ad-hoc networks can provide communication for civilian applications, such as disaster recovery and message exchanges among medical and security personnel involved in rescue missions.



Figure1.2Disaster relief operations

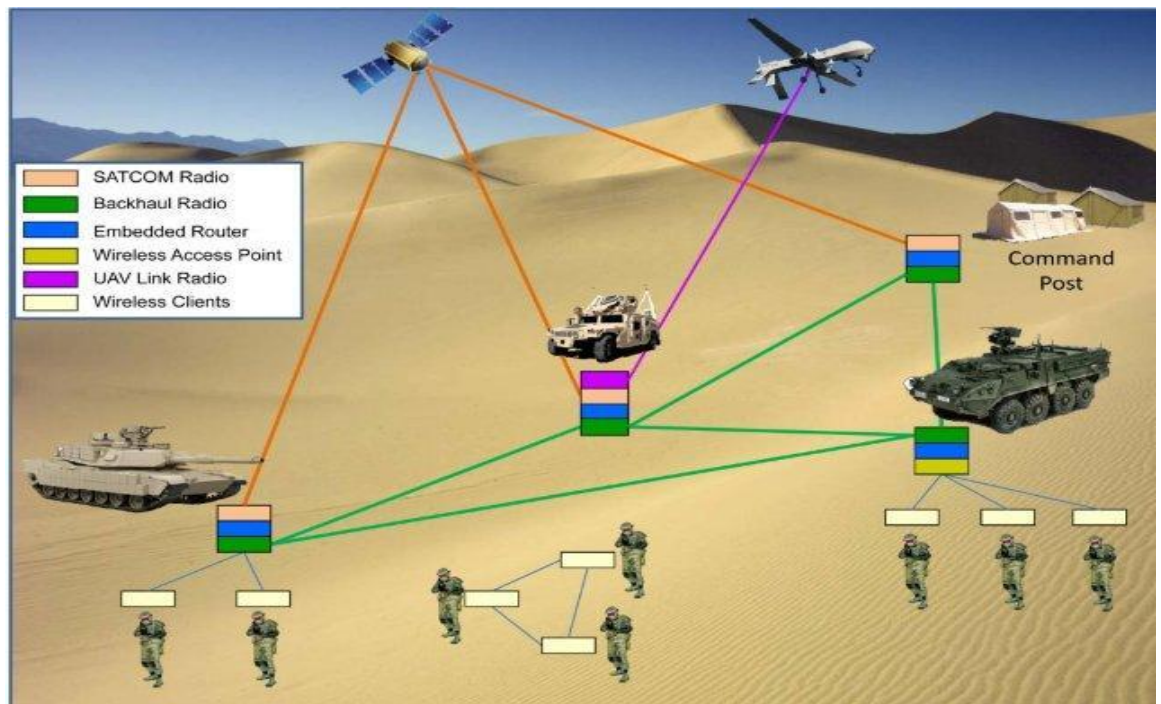


Figure 1.3 Military or police exercises

1.6. Advantages of MANETs

The following are the advantages of MANETs:

- ❖ They provide access to information and services regardless of geographic position.
- ❖ These networks can be set up at any place and time.
- ❖ These networks work without any pre-existing infrastructure.

1.7. Challenges in Wireless ad-hoc Networks

The two most significant differences between infrastructure-based and ad-hoc networks are:

1. Communications in ad-hoc networks are truly peer-to-peer.
2. The individual nodes that do jobs of their own are also now required to route packets as required.

These differences lead to some unique and extremely difficult challenges for ad-hoc networks. Unlike dedicated routers, hosts in MANETs have limited computational resources and more importantly, being battery-operated, very limited power. Building routing decisions in the general-purpose hosts for constantly changing surroundings is big challenge. However, arguably the most important of these challenges is that of security. MANETs are like consistent zero-administration personal environment. The absence of infrastructure and the consequent absence of authorization facilities impede the usual practice of establishing a line of defense to separate the trusted from the non-trusted. This would have been based on a security policy, possession of necessary credentials and the ability of nodes to validate them. In the context of MANETs, there may be no basis for an a priori classification. Additionally, freely roaming nodes join and leave MANETs independently and without notice, making it difficult to have a clear picture of the ad-hoc network membership. In such an environment, there is no guarantee that a path between two nodes would be free of malicious nodes. These nodes would not comply with the employed protocol and would attempt to harm the network operation. The presence of even a small number of adversarial nodes could cause the entire network to collapse.

1.8. Security Attributes in MANETs

Security has become a most important service in Mobile ad-hoc Network (MANETs). To secure an ad-hoc network, the following attributes are to be considered: availability, authentication and key management, confidentiality, integrity, non-repudiation, and scalability. In order to achieve this goal, the security solutions for each layer which are providing complete protection for MANETs are to be described.

There are five main layers on the network, as follows:

1. **Application layer:** Detecting and preventing viruses, worms, malicious codes, and application abuses.
2. **Transport layer:** Authenticating and securing end-to-end communications through data encryption.
3. **Network layer:** Protecting the ad-hoc routing and forwarding protocols.

4. **Link layer:** Protecting the wireless MAC protocol and providing link-layer security support.
5. **Physical layer:** Preventing signal jamming denial-of-service attacks.

Table 1.1. Security attributes in MANETs

No.	Attributes	Goals	Method
1	Availability	Resources can be accessed by all the nodes with in this network.	Distributed Adaptive Service Replication (DAR)
2	Authentication and key management	It ensures that data transmission is authentic.	Hybrid cryptography technique.
3	Confidentiality	It protects data from unauthorized person.	Converting original data into an unintelligible format using Data Encryption Standard(DES)
4	Integrity	It ensures that data being transmitted is never corrupted.	Cryptographic hash function algorithm
5	Non-repudiation	It ensures that sending and receiving information among all the nodes with in this network	Digital signature
6	Scalability	It ensures that newly added nodes in the network to be managed without any corruption	Secure routing protocols used to manage the scalability

1.9. Intrusion Detection System (IDS)

Intrusion detection is very important aspect of defending the cyber infrastructure from attackers or hackers. Intrusion prevention technique such as filtering router policies and firewalls fail to stop such kind of attacks. Therefore, no matter how well a system is protected, intrusion still occurs and so they should be detected. Intrusion detection systems are becoming significant part of security and the computer system. An intrusion detection system is used to detect many types of malicious behaviors of nodes that can compromise the security and trust of a computer system. To address this problem, IDS should be added to enhance the security level of MANETs. If MANET knows how to detect the attackers as soon as they enter the network, we will be able to completely remove the potential damages caused by compromised nodes at the first time. IDSs are a great complement to existing proactive approaches and they usually act as the second layer in MANETs. There is a need for IDS to implement an intelligent control mechanism in order to monitor and recognize security breach attempts efficiently over a period of the expected network lifetime. The present dissertation mechanism has focused on designing Intrusion Detection Systems (IDS) to monitor and analyze system events for detecting network resource misuse in a MANET.

1.10. Motivation of Dissertation

Mobile ad-hoc networks (MANETs) are vulnerable due to its fundamental characteristics, such as open medium, dynamic topology, distributed operation and constrained capability.

Security is a central requirement for mobile ad-hoc networks. Security and robustness will impact the design of the standard for ad-hoc networks is the main motivation for this dissertation.

1.11. Objective

- Define solid privacy requirements regarding malicious attackers in MANETs.
- Develop an efficient intrusion-detection mechanism to protect MANET from attacks by propose and implement a new intrusion-detection system named Hybrid

Cryptography Enhanced Adaptive Acknowledgment (HCEAACK) specially designed for MANETs .

- Compared (HCEAACK) against other popular mechanisms in different scenarios through simulations , evaluated based on measures such as packet delivery factor (PDF) , the overhead introduced by the routing protocol(OH) , and end-to-end packet delay (D).

REVIEW OF LITERATURE

Zhou and Haas (2002) Ad hoc networks are a new wireless networking paradigm for mobile hosts. Unlike traditional mobile wireless networks, ad hoc networks do not rely on any fixed infrastructure. Instead, hosts rely on each other to keep the network connected. The military tactical and other security-sensitive operations are still the main applications of ad hoc networks, although there is a trend to adopt ad hoc networks for commercial uses due to their unique properties. One main challenge in design of these networks is their vulnerability to security attacks. In this paper, we study the threats an ad hoc network faces and the security goals to be achieved. We identify the new challenges and opportunities posed by this new networking environment and explore new approaches to secure its communication. In particular, we take advantage of the inherent redundancy in ad hoc networks multiple routes between nodes to defend routing against denial of service attacks. We also use replication and new cryptographic schemes, such as threshold cryptography, to build a highly secure and highly available key management service, which forms the core of our security framework.

Yih-Chun, Johnson and Perrig (2003) An ad hoc network is a collection of wireless computers (nodes), communicating among themselves over possibly multi-hop paths, without the help of any infrastructure such as base stations or access points. Although many previous ad hoc network routing protocols have been based in part on distance vector approaches, they have generally assumed a trusted environment. In this paper, we design and evaluate the Secure Efficient Ad hoc Distance vector routing protocol (SEAD), a secure ad hoc network routing protocol based on the design of the Destination-Sequenced Distance-Vector routing protocol. In order to support use with nodes of limited CPU processing capability, and to guard against Denial of Service attacks in which an attacker attempts to cause other nodes to consume excess network bandwidth or processing time, we use efficient

one-way hash functions and do not use asymmetric cryptographic operations in the protocol. SEAD performs well over the range of scenarios we tested, and is robust against multiple uncoordinated attackers creating incorrect routing state in any other node, even in spite of any active attackers or compromised nodes in the network.

Chlamtac, Conti, Jennifer and Liu c (2003) Mobile ad hoc networks (MANETs) represent complex distributed systems that comprise wireless mobile nodes that can freely and dynamically self-organize into arbitrary and temporary, “ad-hoc” network topologies, allowing people and devices to seamlessly inter-network in areas with no pre-existing communication infrastructure, e.g., disaster recovery environments. Ad hoc networking concept is not a new one, having been around in various forms for over 20 years. Traditionally, tactical networks have been the only communication networking application that followed the ad hoc paradigm. Recently, the introduction of new technologies such as the Bluetooth, IEEE 802.11 and Hyperlan are helping enable eventual commercial MANET deployments outside the military domain. These recent evolutions have been generating a renewed and growing interest in the research and development of MANET. This paper attempts to provide a comprehensive overview of this dynamic field. It first explains the important role that mobile ad hoc networks play in the evolution of future wireless technologies. Then, it reviews the latest research activities in these areas, including a summary of MANETs characteristics, capabilities, applications, and design constraints. The paper concludes by presenting a set of challenges and problems requiring further research in the future.

Kuladinithi, Gieland Görg (2004) Wearable computing along with advanced mobile communication has the potential to revolutionize the working environment and working processes of the mobile worker of the AEC industry. While wearable computing allows the mobile worker to execute his/her work and tasks more efficiently and safer with the support of information provided by novel user interfaces, like augmented reality glasses and voice recognition, advanced mobile communication can transfer the information required in a certain context (location, time, task etc.) to and from the mobile worker. Wearable computing allows the mobile worker e.g. to access data bases like material lists and CAD drawings, to hold video

conferences with remote or on-site experts without being distracted by additional hardware he has to handle, i.e. his hands and arms are free to concentrate completely on his work. For transferring data and voice calls numerous different networking technology and communication networks are available, like GPRS, UMTS and WLAN. However, communication costs, bandwidth and coverage limitations might prohibit its use on construction environments. Ad-hoc networking offers means of spontaneous communications between different devices without using an infrastructure and provides a promising communication solution for the AEC industry. This article focuses on the applicability of mobile ad-hoc networking in the AEC industry by providing a detailed scenario of how mobile ad-hoc networking can be used in the AEC industry, giving an overview of the existing mobile ad-hoc networks and addressing some issues of implementation and deployment of the networking protocol called Ad-hoc On-demand Distance Vector routing (AODV).

Patwardhan, Parker and Joshi (2005) numerous schemes have been proposed for secure routing protocols, and Intrusion Detection and Response Systems, for ad hoc networks. In this paper, we present a proof-of-concept implementation of a secure routing protocol based on AODV over IPv6, further reinforced by a routing protocol-independent Intrusion Detection and Response system for ad-hoc networks. Security features in the routing protocol include mechanisms for non-repudiation, authentication using Statistically Unique and Cryptographically Verifiable (SUCV) identifiers, without relying on the availability of a Certificate Authority (CA), or a Key Distribution Center (KDC). We present the design and implementation details of our system, the practical considerations involved, and how these mechanisms can be used to detect and thwart malicious attacks. We discuss several scenarios where these secure routing and intrusion detection mechanisms isolate and deny network resources to nodes deemed malicious. We also discuss shortcomings in our approach and conclude with lessons learned, and ideas for future work.

R. Manoharan, Thambidurai and S. Ramesh (2005) Multicasting is an effective way to provide group communication. In mobile ad hoc networks (MANETs), multicasting can support a wide variety of applications that are characterized by a close degree of collaboration. Since MANETs exhibit severe resource constraints such as battery power, limited bandwidth,

dynamic network topology and lack of centralized administration, multicasting in MANETs become complex. The existing multicast routing protocols concentrate more on quality of service parameters like end-to-end delay, jitter, bandwidth and power. They do not stress on the scalability factor of the multicast. In this paper, we address the problem of multicast scalability and propose an efficient scalable multicast routing protocol called 'Power Aware Scalable Multicast Routing Protocol (PASM RP)' for MANETs. PASM RP uses the concept of class of service with three priority levels and local re-routing to provide scalability. The protocol also ensures fair utilization of the resources among the nodes through re-routing and hence the lifetime of the network is increased. The protocol has been simulated and the results show that PASM RP has better scalability and enhanced lifetime than the existing multicast routing protocols.

Anantvaley and Jie Wu (2006) in recent years, the use of mobile ad hoc networks (MANETs) has been widespread in many applications, including some mission critical applications, and as such security has become one of the major concerns in MANETs. Due to some unique characteristics of MANETs, prevention methods alone are not sufficient to make them secure; therefore, detection should be added as another defense before an attacker can breach the system. In general, the intrusion detection techniques for traditional wireless networks are not well suited for MANETs. In this paper, we classify the architectures for intrusion detection systems (IDS) that have been introduced for MANETs. Current IDS's corresponding to those architectures are also reviewed and compared. We then provide some directions for future research.

Al Agha, Bertin, Dang, Guitton, Minet, Thierry Val, and Viollet (2009) in this paper, we present an industrial development of a wireless sensor network technology called OCARI: Optimization of Communication for Ad hoc Reliable Industrial networks. It targets applications in harsh environments such as power plants and warships. OCARI is a wireless-communication technology that supports mesh topology and power-aware ad hoc routing protocol aimed at maximizing the network lifetime. It is based on IEEE 802.15.4 physical layer with deterministic Media Access Control layer for time-constrained communication. During the non-time-constrained communication period, its ad hoc routing strategy uses an

energy-aware optimized-link state-routing proactive protocol. An OCARI application layer (APL) is based on Zig-Bee application support sub layer and APL primitives and profiles to provide maximum compatibility with Zig Bee applications. To fully assess this technology, extensive tests are done in industrial facilities at Electrocute De France R&D as well as at Direction des Constructions Navales Services. Our objective is then to promote this specification as an open standard of industrial wireless technology.

F. moldoveanu ,florolan and puiu (2010) In semiautonomous mobile sensor networks, since human operators may be involved in the control loop, particular improper actions may cause accidents and result in catastrophes. For such systems, this paper proposes a command filtering framework to accept or reject the human-issued commands so that undesirable executions are never performed. In the present approach, Petri nets are used to model the operated behaviors and to synthesize the command filters for supervision. Also the command filter could be implemented using agent technology by associating a filtering agent for each robot. It is believed that the technique presented in this paper could be further applied to large-scale wireless mobile sensor networks.

Kumar, Kadam, Kumar and Pawar (2011) we are considering the Routing misbehavior in MANETs (Mobile Ad Hoc Networks). Routing protocols for MANETs are based on the assumption which are, all participating nodes are fully cooperative. But, due to the open structure node misbehaviors may exist. One such routing misbehavior is that some nodes will take part in the route discovery and maintenance processes but refuse to forward data packets. In this, we propose the 2ACK scheme that serves as an add-on technique for routing schemes to detect routing misbehavior and to mitigate their effect. The basic idea of the 2ACK scheme is to send two-hop acknowledgment packets in the opposite direction of the routing path. To reduce extra routing overhead, only a few of the received data packets are acknowledged in the 2ACK scheme.

Cheffena (2012)a complete dynamic wideband channel model for industrial wireless sensor network is presented. The model takes into account the noise, interferences, and heavy multipath propagation effects present in harsh industrial environments. A first-order two-state

Markov process is adopted to describe the typical bursty nature of the impulsive noise usually present in industrial environments. The interference effects are modeled as multiple narrowband signals operating on the same frequency band as the desired signal. The multipath propagation is described by assuming the scatterers to be uniformly distributed in space within an elliptical region where the transmitting and receiving nodes are located at the foci of the ellipse. Furthermore, performance evaluations of IEEE 802.15.4 in terms of bit error rate using the developed channel model are presented. The results show that in addition to spread spectrum techniques, link diversity can further improve the link quality in harsh industrial environments.

Natarajan, Yang and Zhu (2012) a trust management framework is useful to ensure proper functioning of a mobile ad-hoc network (MANET). Trust metadata created by individual nodes, based on their observation of the behavior of other nodes in their vicinity, is required to be accessible to a trust authority (TA) (e.g., the network administrator) for prompt decisionmaking (e.g., revoking malicious nodes). In this work, for security and scalability reasons, we propose a secure semantics-aware trust metadata management scheme to partition and store an information network of trust metadata of nodes in a MANET. That is, trust metadata is securely propagated to and stored at certain geographic locations inside the network itself, based on its semantics. The TA can send queries of various types in the network to obtain the trust metadata of its interest. This scheme is robust to several security attacks that attempt to disrupt the availability of trust metadata in the network. Our analysis shows that the proposed scheme provides desirable security and functionality properties with low query overhead.

Akshatha and Rashmi (2013) the migration to wireless network from wired network has been a global trend in the past few decades. The mobility and scalability brought by wireless network made it possible in many applications. Among all the contemporary wireless networks, Mobile Ad hoc Network (MANET) is one of the most important and unique applications. On the contrary to traditional network architecture, MANET does not require a fixed network infrastructure; every single node works as both a transmitter and a receiver. Nodes communicate directly with each other when they are both within the same

communication range. Otherwise, they rely on their neighbors to relay messages. The self-configuring ability of nodes in MANET made it popular among critical mission applications like military use or emergency recovery. In recent years, security has become a most important service in Mobile Adhoc Network. Compared to other networks, MANETs are more vulnerable to various types of attacks. In this paper, a comparative study of Secure Intrusion-Detection Systems for discovering malicious nodes and attacks on MANETs are presented. Due to some special characteristics of MANETs, prevention mechanisms alone are not adequate to manage the secure networks. In this case detection should be focused as another part before an attacker can damage the structure of the system. This paper gives an overview of IDS architecture for enhancing security level of MANETs based on security attributes and then various algorithms, namely RSA and DSA.

Rajeshkumar and Valluvan(2013) in recent years, security has become a most important service in Mobile Adhoc Network. Compared to other networks, MANETs are more vulnerable to various types of attacks. In this paper, a comparative study of Secure Intrusion-Detection Systems for discovering malicious nodes and attacks on MANETs are presented. Due to some special characteristics of MANETs, prevention mechanisms alone are not adequate to manage the secure networks. In this case detection should be focused as another part before an attacker can damage the structure of the system. First this paper gives an overview of IDS architecture for enhancing security level of MANETs based on security attributes and various algorithms, namely RSA and DSA. Then a hybrid cryptography IDS to further reduce the network overhead caused by digital signature is indicated.

Akshatha and Jogdand (2013) the migration to wireless network from wired network has been a global trend in the past few decades. The mobility and scalability brought by wireless network made it possible in many applications. Among all the contemporary wireless networks, Mobile Ad hoc Network (MANET) is one of the most important and unique applications. On the contrary to traditional network architecture, MANET does not require a fixed network infrastructure; every single node works as both a transmitter and a receiver. Nodes communicate directly with each other when they are both within the same communication range. Otherwise, they rely on their neighbors to relay messages. The self-

configuring ability of nodes in MANET made it popular among critical mission applications like military use or emergency recovery. In recent years, security has become a most important service in Mobile Adhoc Network. Compared to other networks, MANETs are more vulnerable to various types of attacks. In this paper, a comparative study of Secure Intrusion-Detection Systems for discovering malicious nodes and attacks on MANETs are presented. Due to some special characteristics of MANETs, prevention mechanisms alone are not adequate to manage the secure networks. In this case detection should be focused as another part before an attacker can damage the structure of the system. This paper gives an overview of IDS architecture for enhancing security level of MANETs based on security attributes and then various algorithms, namely RSA and DSA.

Baburao (2013) recent advances in energy harvesting have been intensified due to urgent need of portable, wireless electronics with extensive life span. The idea of energy harvesting is applicable to sensors that are placed and operated on some entities for a long time, or embedded into structures or human bodies, in which it is troublesome to replace the sensor module batteries. Such sensors are commonly called “self-powered sensors. The idea of energy harvesting is applicable to sensors that are placed and operated on some entities for a long time, or embedded into structures or human bodies, in which it is troublesome or detrimental to replace the sensor module batteries. Such sensors are commonly called” self-powered sensors.” The energy harvester devices are capable of capturing environmental energy and supplanting the battery in a stand-alone module, or working along with the battery to extend substantially its life. Vibration is considered one of the highest power and efficient among other ambient energy sources, such as a solar energy and temperature difference. Piezoelectric and electromagnetic devices are mostly used to convert vibration to ac electric power. For vibratory harvesting, a delicately-designed power conditioning circuit is required to store as much as possible of the device-output power into a battery. The design for this power conditioning needs to be consistent with the electric characteristics of the device and battery to achieve maximum power transfer and efficiency. This study offers an overview on various power conditioning electronic circuits designed for vibratory harvester devices and their applications to self-powered sensors. Comparative comments are provided in terms of circuit topology differences, conversion efficiencies and applicability to a sensor.

Shakshuki, Kang and Sheltami (2013) the migration to wireless network from wired network has been a global trend in the past few decades. The mobility and scalability brought by wireless network made it possible in many applications. Among all the contemporary wireless networks, Mobile Ad hoc Network (MANET) is one of the most important and unique applications. On the contrary to traditional network architecture, MANET does not require a fixed network infrastructure; every single node works as both a transmitter and a receiver. Nodes communicate directly with each other when they are both within the same communication range. Otherwise, they rely on their neighbors to relay messages. The self-configuring ability of nodes in MANETs made it popular among critical mission applications like military use or emergency recovery. However, the open medium and wide distribution of nodes make MANETs vulnerable to malicious attackers. In this case, it is crucial to develop efficient intrusion-detection mechanisms to protect MANET from attacks. With the improvements of the technology and cut in hardware costs, we are witnessing a current trend of expanding MANETs into industrial applications. To adjust to such trend, we strongly believe that it is vital to address its potential security issues. In this paper, we propose and implement a new intrusion-detection system named Enhanced Adaptive Acknowledgment (EAACK) specially designed for MANETs. Compared to contemporary approaches, EAACK demonstrates higher malicious-behavior-detection rates in certain circumstances while does not greatly affect the network performances.

Kumar , Ali and Reddy (2014) the flattening towards wireless network from wired network offers has become a worldwide trend in the past few decades. The versatility as well as scalability brought by wireless network created it possible in countless applications. Involving all of the contemporary wireless networks, mobile Ad hoc Network is one of the most essential and unique applications. On the contrary to ordinary network architecture, MANET does not need a fixed network infrastructure; every single node works as both a sender and a receiver. Nodes communicate directly with each other when they are both within the same communication range. Otherwise, they depend on their neighbors to exchange messages. The self-configuring ability of nodes in MANET has made it popular among critical objective applications like military use or emergency recovery. However, the public medium

and wide distribution of nodes make MANET susceptible to malicious attackers. In this case, it is vital to build efficient intrusion detection mechanisms to protect MANET from attacks. With the enhancements to the technology and cut in hardware costs, we are witness to a current trend of widening MANETs into industrial applications. To adapt towards these trend, we strongly believe that it is required to address its potential security issues. In this paper, we propose and implement a new intrusion detection system named enhanced adaptive Acknowledgement (EAACK) particularly designed for MANETs. Compared to modern approaches, EAACK demonstrates higher malicious behavior detection rates in certain conditions while does not immensely affect the network performance.

3 MATERIAL AND METHODS

3.1. Security of Ad-Hoc Networks

Because of dynamic topological changes, ad-hoc networks are vulnerable at the physical link, as they can easily be manipulated. An intruder can easily attack ad-hoc networks by loading available network resources, such as wireless links and energy (battery) levels of other users, and then disturb all users. Attackers can also disturb the normal operation of routing protocols by modifying packets. The intruder may insert spurious information into routing packets, causing erroneous routing table updates and thus misrouting. Some other security vulnerabilities of ad-hoc networks are:

- ❖ **Limited computational capabilities:** Typically, nodes in ad-hoc networks are modular, independent, and limited in computational capability and therefore may become a source of vulnerability when they handle public-key cryptography during normal operation.
- ❖ **Limited power supply:** Since nodes normally use battery as power supply, an intruder can exhaust batteries by creating additional transmissions or excessive computations to be carried out by nodes.
- ❖ **Challenging key management:** Dynamic topology and movement of nodes in an ad-hoc network make key management difficult if cryptography is used in the routing protocol.

The provision of security services in the MANETs context faces a set of challenges specific to this new technology. The insecurity of the wireless links, energy constraints, relatively poor physical protection of nodes in a hostile environment, and the vulnerability of statically configured security schemes are definitely such challenges. However, the single most important feature that differentiates MANETs is the absence of a fixed infrastructure. No part

of the network is dedicated to support individually any specific network functionality, with routing (topology discovery, data forwarding) being the most prominent example. Additional examples of functions that cannot rely on a central service, are naming services, certification authorities (CA), directory and other administrative services. Even if such services were assumed, their availability would not be guaranteed, either due to the dynamically changing topology that could easily result in a partitioned network or due to congested links close to the node acting as a server. Furthermore, performance issues such as delay constraints on acquiring responses from the assumed infrastructure would pose an additional challenge.

The absence of infrastructure and the consequent absence of authorization facilities impede the usual practice of establishing a line of defense, separating nodes into trusted and no trusted.

Such a distinction would have been based on a security policy, the possession of the necessary credentials and the ability for nodes to validate them. In the MANETs context, there may be no ground for an a priori classification since all nodes are required to cooperate in supporting the network operation, while no prior security association can be assumed for all the network nodes. Additionally, in MANETs freely roaming nodes form transient associations with their neighbors, join and leave MANETs sub-domains independently and without notice. Thus it may be difficult in most cases to have a clear picture of the Ad Hoc network membership. Consequently, especially in the case of a large-size network, no form of established trust relationships among the majority of nodes could be assumed.

In such an environment, there is no guarantee that a path between two nodes would be free of malicious nodes, which would not comply with the employed protocol and attempt to harm the network operation. The mechanisms currently incorporated in MANETs routing protocols cannot cope with disruptions due to malicious behavior. For example, any node could claim that is one hop away from the sought destination, causing all routes to the destination to pass through itself. Alternatively, a malicious node could corrupt any in transit route request (reply) packet and cause data to be misrouted.

The presence of even a small number of adversarial nodes could result in repeatedly compromised routes, and, as a result, the network nodes would have to rely on cycles of time-out and new route discoveries to communicate. This would incur arbitrary delays before the

establishment of a non-corrupted path, while successive broadcasts of route requests would impose excessive transmission overhead. In particular, intentionally falsified routing messages would result in a denial-of-service (DoS) experienced by the end nodes. The proposed scheme has combats such types of misbehavior and safeguards the acquisition of topological information.

3.2. Attacks on Ad Hoc Networks

In addition to often being wireless the structure of an ad-hoc network or lack there of, leads to some special kinds of attacks. Especially attacks on the connectedness of the network which means attacks on the routing protocol. In this section some of these attacks will be addressed.

3.2.1. Routing Loop

By sending forged routing packets an attacker can create a routing loop. This will result in data packets being sent around consuming both bandwidth and power for a number of nodes. The packets will not reach their intended recipient and thus can be considered a sort of denial-of-service attack.

3.2.2. Black Hole

The setup for the black hole attack is similar to the routing loop attack in which the attacker sends out forged routing packets. It can setup a route to some destination via itself and when the actual data packets get there they are simply dropped, forming a black hole where data enters but never leaves.

Another possibility is for the attacker to forge routes pointing into an area where the destination node is not located. Everything will be routed into this area but nothing will leave also creating a sort of black hole.

3.2.3. Grey Hole

A special case of the black hole attack is an grey hole attack. In this attack the adversary selectively drops some kinds of packets but no other. For example the attacker might forward routing packets but not data packets.

3.2.4. Partitioning

Another kind of attack is for the attacker to create a network partition in which some nodes are split up to not being able to communicate with another set of nodes. By analyzing the network topology the attacker can choose to make the partitioning between the set of nodes that makes the most harm into the system. This attack can be accomplished in many kinds of ways. Both by forging routing packets as in the previous attacks but also using some physical attack such as radio jamming.

3.2.5. Blackmail

Some ad-hoc routing protocols try to handle the security problems by keeping lists of possibly malicious nodes. Each node has a blacklist of, what it thinks, bad nodes and thereby avoiding using them when setting up routing paths. An attacker might try to blackmail a good node causing other good nodes to add this node to their blacklists and so avoid it.

3.2.6. Wormhole

In the wormhole attack an attacker uses a pair of nodes connected in some way. It can be a special private connection or the packets are tunneled over the ad-hoc network. Every packet that one of the nodes sees is forwarded to the other node which in turn broadcast them out. This might create short circuits for the actual routing in the ad-hoc network and thereby create some routing problems. Also, all the data can be selectively forwarded or not using this attack thereby controlling the ad-hoc network to a large extent. This kind of attack together with a partitioning attack can gain almost complete control over the network traffic.

3.2.7. Rushing Attack

Many reactive routing protocols keep a sequence number for duplication suppression at every node. An attacker can distribute a large number of route requests with increasing sequence numbers forged to appear to be from other nodes. This way when the actual route request is sent out many nodes suppress it as a duplicate and thereby disrupt the actual route discovery.

3.2.8. Resource Consumption

By injecting extra data packets into the ad-hoc network limited resources such as bandwidth and maybe battery power are consumed for no reason. Even more resources might be consumed by injecting extra control packets since these might lead to additional computation. Also, the other nodes might forward control information as it comes in resulting in even more resource consumption. For devices that try to conserve battery power by only occasionally enabling their communication device a malicious attacker might communicate in an ordinary way but with the only intent to drain battery power.

3.2.9. Dropping Routing Traffic

It is essential in the ad-hoc network that all nodes participate in the routing process. However, a node may act selfishly and process only routing information that are related to itself in order to conserve energy. This behavior/attack can create network instability or even segment the network.

3.2.10. Location disclosure

A location disclosure attack can reveal information related to the location of a node or the topology and structure of the network. The information gained might reveal which other nodes are adjacent to the target or the physical location of a participating node. The attack can be implemented by using a command similar to traceroute that exists in Unix-like systems or with the use of the time-to-live attribute of the routing packet and the addresses of the devices by sending ICMP error messages. In the end, the attacker knows which nodes are situated on the route to the target node. If the locations of some of the intermediary nodes are known, one can gain information about the location of the destination node as well.

3.3. Secure Routing and Intrusion Detection in MANETs

The majority of the routing protocols proposed in the literature are assuming non-hostile environments. Due to its dynamically changing topology, open environment and lack of centralized security infrastructure, a MANETs is extremely vulnerable to malicious node

presence and to certain types of attacks that can occur. To address these concerns, several secure routing protocols have been proposed recently: SAODV, Ariadne, SEAD, CSER, SRP, SAAR, BSAR, and SBRP.

Mobile ad-hoc Networks (MANETs) present a number of unique problems for Intrusion Detection Systems (IDS). Differentiating between malicious network activity and spurious, but typical, problems associated with an ad-hoc networking environment is a challenging task. In an ad-hoc network, malicious nodes may enter and leave the immediate radio transmission range at random intervals or may collude with other malicious nodes to disrupt network activity and avoid detection. Malicious nodes may behave maliciously only intermittently, further complicating their detection. A node that sends out false routing information could be the one that has been compromised, or merely one that has a temporarily stale routing table due to volatile physical conditions. Dynamic topologies make it difficult to obtain a global view of the network and any approximation can become quickly outdated. Traffic monitoring in wired networks is usually performed at switches, routers and gateways, but an ad-hoc network does not have these types of network elements where the IDS can collect audit data for the entire network. Network traffic can be monitored on a wired network segment, but ad-hoc nodes or sensors can only monitor network traffic within its observable radio transmission range.

3.4. Intrusion Detection in MANETs:

The success of MANETs-based applications depends on many factors, trustworthiness being one of the primary challenges to be met. Despite the existence of well-known security mechanisms, additional vulnerabilities and features pertinent to this new networking paradigm might render such traditional solutions inapplicable. The absence of a central authorization facility in an open and distributed communication environment is a major challenge, especially due to the need for cooperative network operation. In particular, in MANETs, any node may compromise the routing protocol functionality by disrupting the route discovery process.

Wireless ad-hoc networks are vulnerable to various attacks. These include passive eavesdropping, active interfering, impersonation, and denial-of-service. Intrusion prevention measures, such as strong authentication and redundant transmission, can be used to address some of these attacks. However, these techniques can address only a subset of the threats, and, moreover, are costly to implement.

The dynamic nature of ad-hoc networks suggests that prevention techniques should be complemented by detection techniques that monitor the security status of the network and identify anomalous and/or malicious behavior. These techniques are usually less expensive to implement and can be easily deployed in existing ad-hoc networks without requiring modifications to the nodes' configuration or the routing protocols being used.

Intrusion is defined as a sequence of related actions performed by a malicious adversary that results in the compromise of a target system. It is assumed that the actions of the intruder violate a given security policy. The existence of a security policy that states which actions are considered malicious and should be prevented is a key requisite for an intrusion detection system to work.

Intrusion detection is the process of identifying and responding to malicious activities target at computing and network resources. This identification introduces the notion of intrusion detection as a process, which involves technology, people and tools. Intrusion detection is an approach that is complementary with respect to mainstream approaches to security such as access control and cryptography.

Adoption of intrusion detection system is motivated by several factors, some of which are listed below:

1. Surveys have shown that most computers are flawed by vulnerabilities, regardless of manufacturer or purpose that the number of security incidents is continuously increasing, and that users and administrators are generally very slow in applying fixes to vulnerable systems. As a consequence, many experts believe that computer systems will never be absolutely secure.

2. Deployed security mechanisms e.g. authentication and access control may be disabled as a consequence of misconfiguration or malicious actions.
3. Users of the system may abuse their privileges and perform damaging activities.
4. Even if an attack is not successful, in most cases it is useful to be aware of the compromise attempt. Intrusion detection systems (IDS) are software applications dedicated to detect intrusions against a target network. IDS are designed to address the issues discussed above; they are not intended to replace traditional security methods, but to complement and complete them.

An intrusion detection system must fulfill the following requirements:

1. **Accuracy:** IDS must not identify a legitimate action in a system environment as an anomaly or a misuse (a legitimate action identified as an intrusion is called a false positive).
2. **Performance:** The performance of the IDS must be sufficient enough to carry out real-time intrusion detection (real-time means an intrusion must be detected before significant damage has occurred). As per the literature, this should be under a minute.
3. **Completeness:** IDS should not fail to detect an intrusion (an undetected intrusion is called a false negative). Arguably this requirement is rather difficult to fulfill because it is almost impossible to have a global knowledge about past, present and future attacks. IDS should however, minimize false negatives.
4. **Fault-tolerance:** IDS must itself be resistant to attacks.
5. **Scalability:** IDS must be able to process the worst-case number of events without dropping information. This point is especially relevant for systems that correlate events from different sources at a small number of dedicated hosts. As networks grow bigger and get faster, such nodes become overwhelmed by increasing number of events.

3.5. Approaches to Intrusion Detection

Intrusion detection techniques have traditionally been classified into two paradigms, namely anomaly detection, also known as behavior-based intrusion detection and misuse detection, also called knowledge-based intrusion detection.

In anomaly or behavior-based detection techniques, historical data about a system's activity and specifications of the intended behavior of users and applications are used to build a profile of the "normal" operation of the system. The detection process then attempts to identify patterns of activity that deviate from the defined profile; anything that does not correspond to a previously learned behavior is considered anomalous and suggests an intrusion attempt.

Misuse or knowledge-based detection techniques take a complementary approach. Misuse detection tools are equipped with a number of attack descriptions (or "signatures") that are matched against the stream of audit data to identify evidence of the occurrence of the modeled attacks. These IDS accumulate knowledge about attacks examine traffic and try to identify patterns indicating that a suspicious activity may be occurring.

Misuse and anomaly detection both have advantages and disadvantages. Misuse detection can perform focused analysis of the audit data and usually produces very few false positives. However, it can detect only those attacks that have been modeled and possibly variations on those attacks. This means that this approach can be applied against known attack patterns only, and the knowledge-based must be updated frequently.

Anomaly detection has the advantage of being able to detect attempts to exploit new and unforeseen vulnerabilities without a priori knowledge of explicit security flaws. This advantage is paid for in terms of the large number of false positives generated; the entire scope of system behavior may not be covered during the learning phase and also legitimate behavior may change over time. It also comes with the difficulty of training a system with respect to a highly dynamic environment; obviously a finite training period is also needed. The assumption that the system in question is free of anomaly during the training period also may not always be true.

As discussed earlier, Mobile ad-hoc Networks are fundamentally different from their wired-side counterparts or even the infrastructure-based networks. The nature of MANETs not only introduces new security concerns but also exacerbates the problem of detecting and preventing anomalous behavior. While in a wired network or in an infrastructure-based wireless network, an intruder could be a host that is either inside or outside the network and could be subjected to varying degrees of access control and authentication, in a MANETs, an intruder is a part of

the network infrastructure. Moreover, at the outset, an intruder in a MANETs could be a trusted and integral component of the network infrastructure and only later exhibit aberrant behavior.

3.6. IDS Techniques for MANETs proposed in the literature:

Intrusion Detection that addresses secure routing, arguably the most important issue in MANETs has interested many researchers. Numerous techniques for ID have been proposed in the literature, in both the categories of anomaly detection and misuse detection.

In this section, we mainly describe four existing approaches, namely, Watchdog [20], TWOACK [12], Adaptive Acknowledgment (AACK) [21], and EAACK [19]:

3.6.1. Watchdog and Pathrater:

Watchdog [18] was the first snooping intrusion detection protocol for MANETs. Watchdog relies upon DSR. Each node participates by ‘watching’ its downstream node on the route from source to destination to ensure that it has retransmitted the packet without modification. The authors hold that if source routing is not used then a misbehaving node could simply broadcast to a non-existent node to fool the watchdog. To mitigate the effects of a misbehaving node, the authors also introduce Pathrater, which selects a path from source to destination based on ‘reliability’ metric instead of the shortest path. This approach relieves the malicious node from the requirement of participating in the routing process which may be construed as a reward. It is very popular and highly efficient IDS for improving the throughput of network with the presence of malicious nodes. These IDS can be classified into two methods such as Watchdog and Pathrater. It is responsible for discovering malicious node misbehaviors in the network. Watchdog detects malicious misbehaviors by listening to its next hop’s transmission in the network. If a Watchdog IDS overhears that its next node fails to forward the packet within a certain period of time, it increases its failure counter. Whenever a node’s failure counter exceeds a predefined threshold value, the Watchdog node reports it as misbehaving. In this case, the Pathrater cooperates with the routing protocols to avoid the reported nodes in future transmission. The Watchdog-IDS fails to discover malicious nodes in the following situations:

- 1) Ambiguous collisions.

- 2) Receiver collisions.
- 3) Limited transmission power.
- 4) False misbehavior report.
- 5) Collusion.
- 6) Partial dropping.

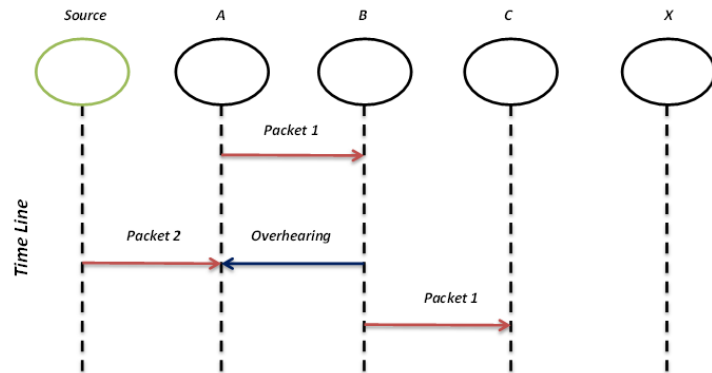


Figure 3.1 Ambiguous collision.

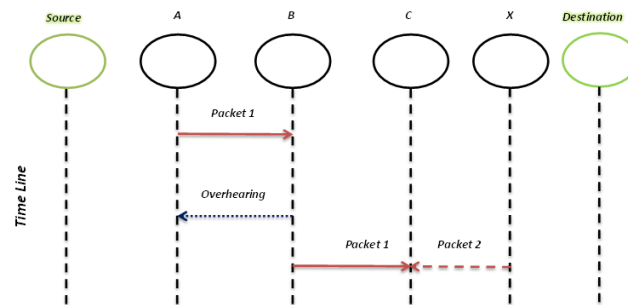


Figure 3.2 Receiver collisions both nodes B and X is trying to send Packet 1 and Packet 2, respectively, to node C at the same time.

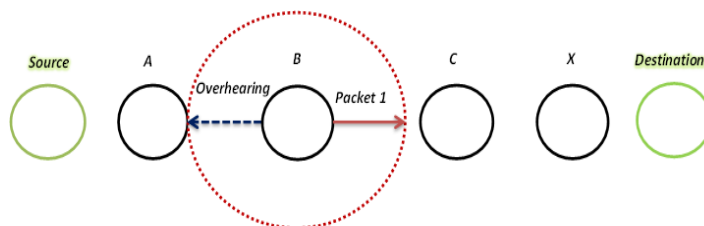


Figure 3.3 Limited transmission power: Node B limits its transmission power so that the packet transmission can be overheard by node A but too weak to reach node C.

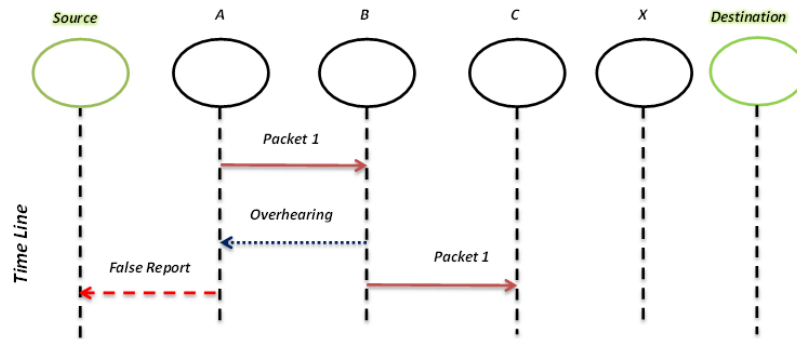


Figure 3.4 False misbehavior report: Node A sends back a misbehavior report even though node B forwarded the packet to node C.

3.6.2. Two Acknowledgment Scheme (TWOACK):

It is another important IDS TWOACK for discovering malicious nodes in MANETs [22]. The main aim of this ID to resolve the receiver collision and limited transmission power problems of Watchdog, TWOACK detects misbehaving links by acknowledging every data packet transmitted over every three consecutive nodes along the path from the source to the destination. Upon retrieval of a packet, each node along the route is required to send back an acknowledgment packet to the node that is two hops away from it down the route. TWOACK is required to work on routing protocols such as Dynamic Source Routing (DSR).

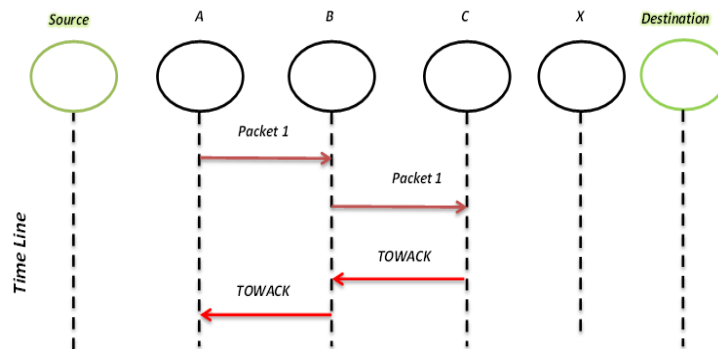


Figure 3.5 TWOACK scheme: Each node is required to send back an acknowledgment packet to the node that is two hops away from it.

The TWOACK IDS effectively processes the receiver collision and limited transmission power problems indicated by Watchdog. However, the acknowledgment process required in every packet transmission process added a significant amount of unwanted network overhead. Due to the limited battery power nature of MANETs, such redundant transmission process can easily degrade the life span of the entire network. However, many research studies are working in energy harvesting to deal with this problem.

3.6.3. Adaptive Acknowledgment Scheme (AACK):

It is same as TWOACK IDS, AACK IDS is an acknowledgment-based network layer IDS. It can be treated as a combination of an ID called TACK (identical to TWOACK) and an end-to-end acknowledgment IDS called Acknowledge (ACK). Compared to TWOACK IDS, AACK IDS reduced network overhead. The end-to-end ACK IDS is shown in Figure 3.6.

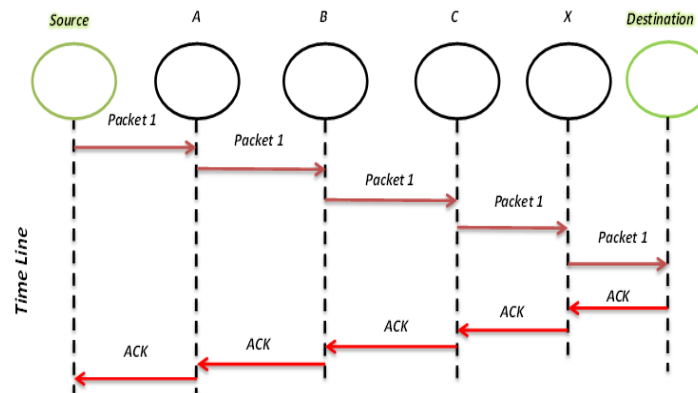


Figure 3.6 End-to-End ACK IDS for MANETs ACK scheme: The destination node is required to send acknowledgment packets to the source node.

The source node sends out Packet 1 without any overhead. All the intermediate nodes simply forward this packet. When the destination node receives Packet 1, it is required to send back an ACK acknowledgment packet to the source node along the reverse order of the same path.

Within a predefined time slot, if the source node receives this ACK packet, then the packet transmission from node Source to node Destination is successful.

But both TWOACK and AACK still suffer from the problem that they fail to detect malicious nodes with the presence of false misbehavior report and fake ACK packets. In fact, many of the existing IDSs in MANETs adopt an acknowledgment-based scheme, including TWOACK and AACK. The functions of such detection schemes all largely depend on the ACK packets. Hence, it is crucial to guarantee that the acknowledgment packets are valid and authentic. To address this concern, a digital signature is adopted in secure IDS named Enhanced AACK (EAACK).

3.6.4. Enhanced Adaptive Acknowledgment Scheme (EAACK):

EAACK is based on both DSA and RSA algorithm. The three main parts of the EAACK scheme are ACK, secure ACK (S-ACK), and misbehavior report authentication (MRA). EAACK is an acknowledgement based IDS. This scheme uses the digital signature method to prevent the attacker from forging acknowledgment packets. Before the acknowledgement packets sent out EAACK requires the whole acknowledgement packets are digitally signed and verified by its receiver until they are accepted. EAACK shows that high malicious behavior rates without decreasing the network performances.

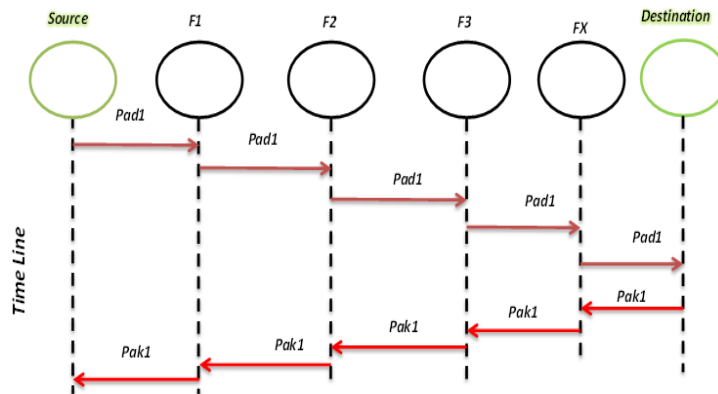


Figure 3.7 System control flow: This figure shows the system flow of how the EAACK scheme works.

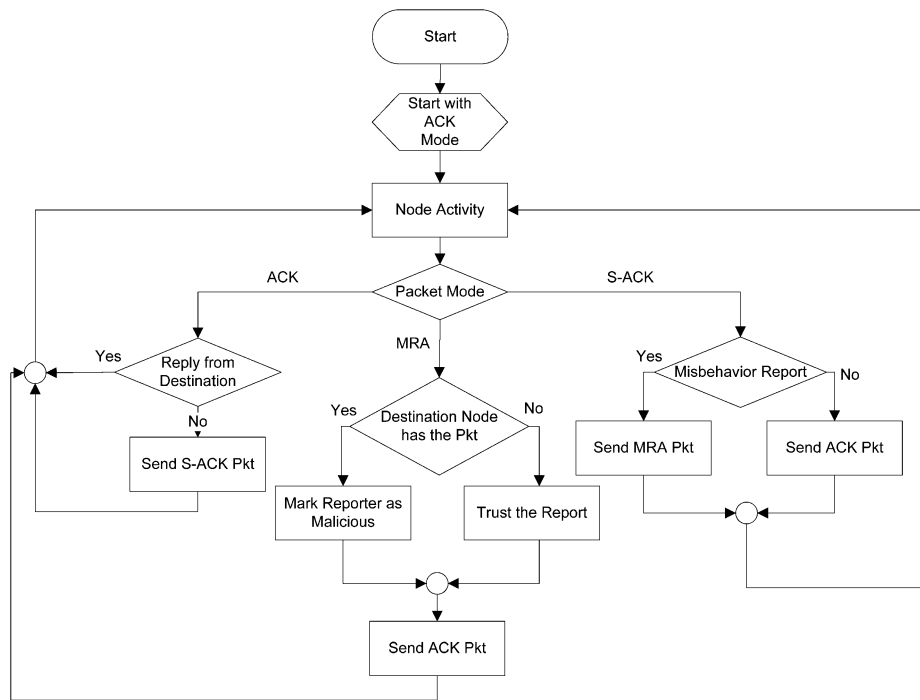


Figure 3.8Flow Chart of EAACK Scheme [19]

➤ **Algorithm of Enhanced Adaptive Acknowledgment Scheme (EAACK)**

#####Routing packets from Source to Destination#####

Create a list N (all); #a set contains all the information about nodes

Initiate Route discovery using RREQ and RREP;

Transmit the packets (S_{data} to D_{data})

#####Checking Node Activity #####

If { $D_{ack} == receive$ } {

D_{data} ;

} else {

```

Initiate  $S_{ack}$ 

}

If(Receiveddata ==  $S_{ack}$ ){

Misbehavior report(a);

If(Misbehavior report(a)==0) {

Send  $D_{ack}$ ;

} else {

Initiate MRA;

}

If(Receiveddata == MRA){

Find another path to Destination;

If(Destination node doesn't has packet) {

Trust the report

} else {

Mark reporter as malicious;

}

Create a list H(i);# storing information about malicious nodes;

}

```

a. Acknowledgement (ACK):

ACK is principally end-to-end acknowledgement scheme. It performs as a part of the hybrid scheme in EAACK, intend to reduce network overhead when no network misbehavior is

detected. In the ACK mode, the Source node sends the ACK data packet to the destination node. After that, all the intermediate nodes between the source and destination are cooperative and Destination node successfully receives Packet it requires to send the ACK packet back to the Source node with same route but in reverse order. If in a particular time the Source node receives the packet the data transmission is successful. Otherwise, Source node will change to S-ACK mode by sending out an S-ACK data packet to detect the misbehaving nodes.

b. Secure Acknowledgement(S-ACK):

In the S-ACK scheme to detect the misbehaving nodes every three successive nodes work in a group. In the three successive nodes the S-ACK acknowledgment packet is sent by the third node to the first node. The S-ACK scheme is able to detect the misbehaving nodes in the presence of receiver collision and low transmission power.

c. Misbehavior Report Authentication (MRA):

Actually, in the watchdog it fails to identify the misbehaving nodes due to the presence of a false misbehavior report. Because of this false report information the watchdog reports normal nodes as malicious nodes. To overcome this problem, the MRA scheme is to authenticate whether the destination node has received the reported missing packet through a different route. For this the source node seeks its local knowledge base and identifies the other route to the destination node. If there is no route to destination by using the DSR routing request to find the alternate route. When the MRA packet is received by the destination node, it compares with using the local knowledge base whether the reported packet was received or not. If already received it make a decision, it is a false misbehavior report.

d. Digital Signatures (DS):

The three parts of EAACK are ACK, S-ACK and MRA are acknowledgement based detection systems. To detect the misbehaviors in the network, the three schemes rely on acknowledgment packets. All acknowledgement packets in the EAACK are authentic and untainted. Otherwise the attackers forge the acknowledgement packets; all the three schemes are susceptible. Include digital signature in EAACK to ensure the integrity of the intrusion detection system. It requires all acknowledgment packets to be digitally signed before they are

sent out and verified until they are accepted. To overcome this DSA and RSA digital signature schemes are used in EAACK.

3.7. Disadvantages of Existing systems:

Existing schemes are largely depend on the acknowledgment packets. Hence, it is crucial to guarantee that the acknowledgment packets are valid and authentic but they suffer from the problem that they fail to detect malicious nodes with the presence of false misbehavior report and forged acknowledgment packets.

Another drawback of most previous schemes is the significant amount of unwanted network overhead. Due to the limited battery power nature of MANETs, such overhead can easily degrade the life span of the entire network. In RSA algorithm, public key and Private Key used to share the secret. The keys are denoted as public key (e, N) and Private Key (d, N) to generate keys we have to use two prime numbers by key generation. To make encryption decryption the node should share the public key (e, N) to all other node. By using public key (e, N) value, the hacker can find private key (d, N) by using RSA hacking like reverse formulation.

3.8. Proposed System (HCEAACK):

In this dissertation, we propose and implement a new and efficient intrusion-detection system named Hybrid cryptography Enhanced Adaptive Acknowledgment (HCEAACK) specially designed for MANETs. Compared to contemporary approaches, HCEAACK demonstrates higher malicious-behavior-detection rates in certain circumstances while does not greatly affect the network performances.

In the proposed method we incorporated digital signature in our proposed scheme. In order to ensure the integrity of the IDS, HCEAACK requires all acknowledgment packets to be digitally signed before they are sent out and verified until they are accepted.

To detect RSA hacking we proposed the technique with hybrid cryptography. In this technique, each and every source-destination pair has the secret key and public and private key. By using public and private key normal digital sign can proceed, and final verification will be done by using secret key.

To provide high security for information on controlled networks different types of cryptographic algorithms are used. These cryptographic algorithms are required to provide data security and user's authenticity. By utilizing a combination of both symmetric and asymmetric cryptographic techniques, the new security protocol has been designed for better security. This Hybrid security model provides three cryptographic primitives such as integrity, confidentiality and authentication. To provide further more security we have added Advanced Encryption Standard (AES) algorithm for encrypted ACK data and get secure ACK, then digital sign of secured ACK data by RSA algorithm. This is encrypted and decrypted only by destination and source. Secrete key used to make more security for acknowledgement and encrypted ACK data by AES algorithm.

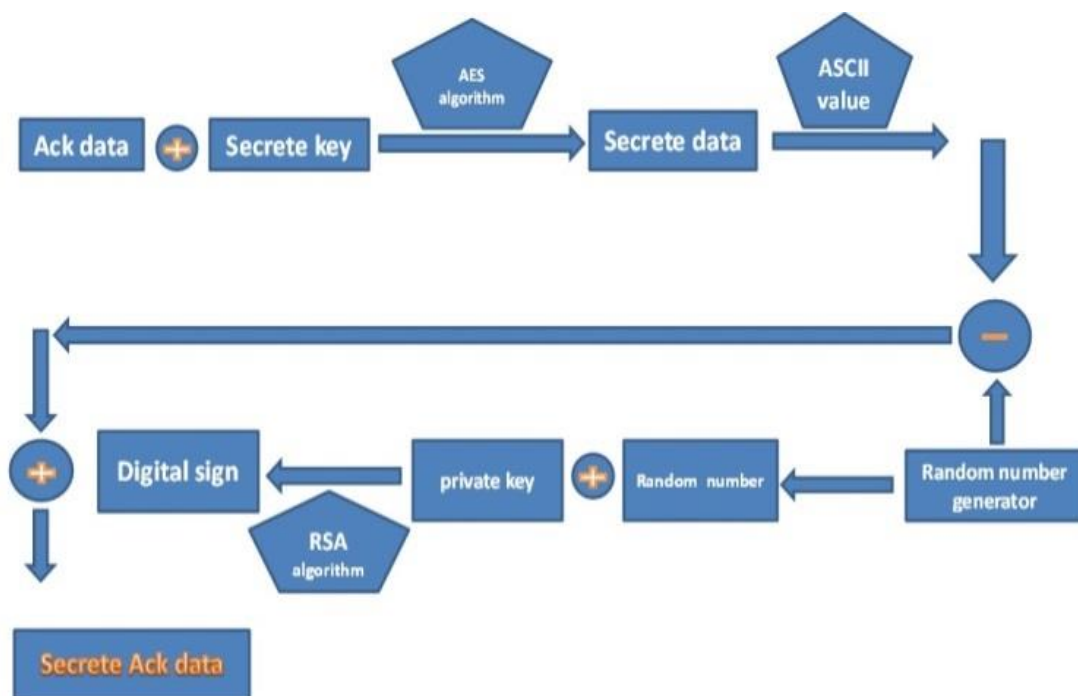


Figure 3.9Formation of Secrete Acknowledgement.

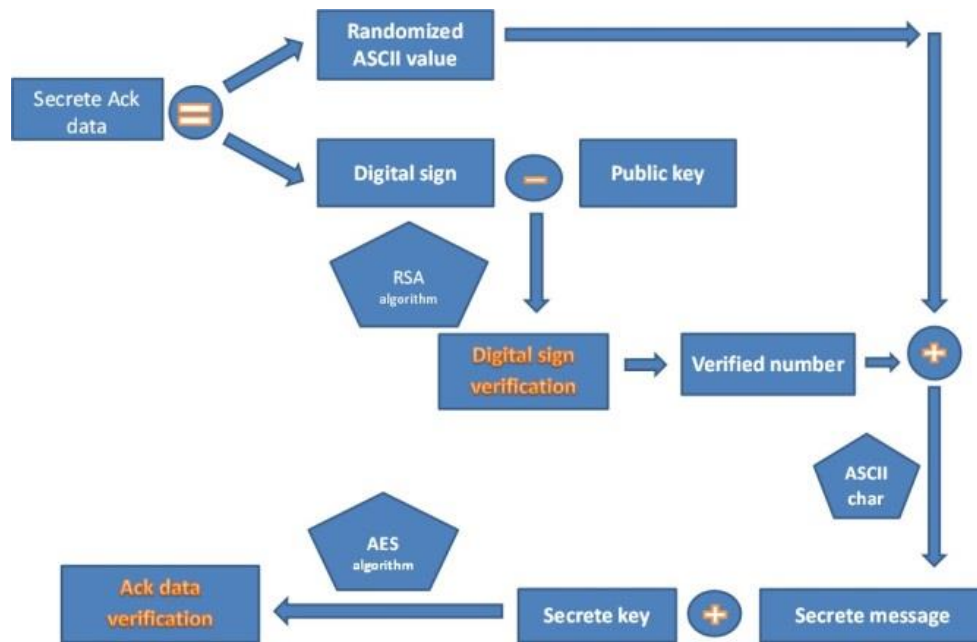


Figure 3.10 Verification Process at Source.

3.9. Advantages of Proposed System (HCEAACK):

1. Our proposed approach HCEAACK completely overcomes the weaknesses like false misbehavior, limited transmission power, and receiver collision.
2. All acknowledgment packets in HCEAACK are authentic and untainted.
3. Our proposed method can significantly improve the packet delivery ratio.

3.10. Algorithm of Proposed System (HCEAACK):

- 1) If node has data to transfer to destination node
 - a. Check the routing table
 - i. If route found
 1. Send the data
 2. Start Counting data

3. At beginning of data count set the timer to check the counting
 - ii. If route not found
 1. Generate the req as normal on demand routing protocol
 2. Update the request with own public key
 3. Broadcast to all neighbor to find destination
- 2) If Req received
 - a. Checks req is new
 - i. If not
 1. Ignore
 - ii. If yes
 1. Updates the reverses route
 2. Updates the key information
 3. Checks node is the destination
 - a. If yes
 - i. Generate the rep with own public key
 - b. If not
 - i. Forward the packet further to all
- 3) If Rep received
 - a. Updates reverse route
 - b. Updates the key information
 - c. Checks node has data to send to source of reply
 - i. If yes
 1. *Go to main step 1*
 - d. Checks the route to rep destination
 - i. If found
 1. Forward
 - ii. Else
 1. Ignore
- 4) If data received
 - a. Checks I'm the destination
 - i. If yes

1. Generate the ack
 - a. Set current time as ack time Ta
 - b. Checks the pair-wise key b/w source and destination
 - i. If found set key as K_{S-d}
 - c. Split Ta into separate character $\cup T_{SA}$
 - d. Create empty list for encrypted data El
 - e. For-each char $Pt \in T_{SA}$
 - i. Encrypt by AES algorithm
 1. $Pt \Rightarrow Ct$
 2. Convert to ascii value Cta
 3. Generate random number($rand$)
 4. New value $Nv = Cta - rand$
 5. $(Cta \& Nv \& rand) \cup El$
 - ii. Make digital sign
 1. Checks for own private key
 2. For_each value of El
 - a. Extract Nv
 - b. Encrypt by RSA private key
 - i. $Nv \Rightarrow CNv$
 - ii. $CNv \cup Digital_sgn_lst$
 2. Send the secrete ack with
 - a. Digital sign
 - b. Rand number
 - c. Generation time
- 5) If digital sign received in source
- a. Node checks the public key info for ack generator
 - i. If found
 1. decrypt by $Pu \Rightarrow Ptrsa$
 2. Checks for secrete pair key
 - a. If found

- i. Decrypt P_{rsa} by $K_{S-d} \Rightarrow P_t$
 - 3. P_t U plain text list
- b. Compare with original form of digital sign with plain text list data
 - i. If matching
 - 1. Forward further data through same path
 - ii. If not matching
 - 1. Switch to secure ack mode as like as EAACK algorithm.

3.11. Simulators

3.11.1. Introduction

Simulation is the imitation of some real thing, state of affairs, or process. The act of simulating something requires capturing the essential characteristics, activities and rules of a selected physical or abstract system. Simulations are often used to model natural, machine or human systems in order to gain insight into their functioning. In addition, it is a very important mechanism to understanding interactions between various systems, parts of which may be difficult to recreate or control in the real world. In technology, simulations are used widely for testing, performance optimization and measurement, safety engineering, training and education. Simulation is specifically necessary in the context of our research, since it involves wireless networks. It is almost impossible to reproduce the wireless propagation environment, difficult to use real radio-wave based transmission to test our concept, and non-trivial to use real wireless sensor/devices to present our research idea. Use of a simulator solves all these challenges. Simulators for communication networks can provide near accurate reproductions of most features in the environment, like noise, probability of loss or alteration of data. They often allow user implementations of protocols for transmission, propagation, reception or other communication aspects to work with their "ether". Mostly, they provide a data-view model of wireless devices.

➤ Motivation for Simulation

1. Cheap, does not require costly equipment.
2. Complex scenarios can be easily tested.
3. Results can be quickly obtained more ideas can be tested in a smaller timeframe.

4. The real thing isn't yet available.
5. Controlled experimental conditions, Repeatability helps aid debugging.
6. Real systems too complex to model.

Thus, using simulators allows us to concentrate on the research idea instead of physical implementation details. They enable building a valid model whose behavior and functioning can be repeated and measured; in short, a statistically verifiable proof of concept.

3.11.2 General simulator characteristics and requirements

Simulators have been introduced in the previous section. Widely speaking, we are looking for a simulator that closely replicates the mobile ad-hoc network system and its components (which we discuss on the next section). Before going into the details of selection of a simulator, there is a need to define the factors on which a generic simulator is judged on. These are presented below:

1. **Definition of the characteristics of the simulation model:** A simulator has to allow defining the characteristics of the system we aim to model. The simulator must be general-purpose enough to be able to provide the default behaviors of the system. For example, a communication networks simulator must have features to model a few transmission media (i.e. propagation media like air, fiber-optics etc.). But, the simulator must also be flexible enough to allow the user to define objectives by using what the simulator offers. For example, often, in communications network research, users are creating newer protocols, or approaches to a particular challenge. They desire the simulator to provide "pluggable" features which will enable them to quickly execute the components they designed while using the other available features. Secondly, the user may want to define the relationships between various components in use in the simulator. These may be interfaces between various layers (say, in an implementation of the TCP-IP suite), interactions between individual entities (say, 2 routers exchanging state information) etc. A simulator may allow definitions of such features in a particular format. This format is important in the

selection of a simulator. A simpler, user-friendly, or popular definition language will decrease the time spent in learning to use the simulator itself.

2. **Collection of input-output data from the real system:** The format of input and collection of output data from the system is the next important feature. Input data are necessary to set the main parameters of the model; e.g. transmission characteristics, inter-arrival rates of traffic, channel loss etc. Output data are necessary to validate functionality, and or calculate the importance of the obtained results. The formats of data used to specify input or collect output are equally important, since creating input data files or rendering and analyzing output results should not become a task of its own.
3. **Design of the experiment and simulation runs:** This involves designing the whole experiment. It includes the details about each simulation, preferences for data to be collected, simulation runs, time duration of each simulation run, etc. The simulator set-up may share some components discussed in the previous section 2, but differs from "input or output data" in that these are details used by the simulator to configure each run, and not what the simulated program expects as input/delivers as output. It may also involve finding out what support the simulator has for synchronizing clocks, random number generators and such optional details.

3.11.3 Real-world components of a mobile ad-hoc network

It is necessary to understand the constituents of mobile ad-hoc networks before starting to look for simulators that can be used to model them. The following are the key components of such networks:

1. **Nodes:** Nodes are a set of mobile devices, with processing and wireless transmission capabilities. In real-life, they are often powered by on-board batteries, equipped with sensors and navigation devices; they may perform a variety of functions ranging from store-and-forward to intelligent aggregation of data. In simulations, they are depicted with some of the above characteristics. On-board batteries may be depicted by a continuously decreasing numeric power indicator. Nodes may run logic to report numbers that make them behave like sensors.

2. **Node architecture:** Nodes are communication devices (i.e. radio devices) whose internal architecture is defined in terms of the OSI (or other) protocol stack. This means that node communications are regulated by the characteristics associated to the OSI protocol layers: physical, data link, medium access control (MAC), network, transport, session, presentation, application. In simulations, nodes generated execute protocol implementations of various layers, similar to their real time counterparts.

3. **Communication Architecture:** Nodes communicate using wireless interfaces. Thus, the physical and technical aspects of radio emission, propagation and reception have to be explicitly taken into account. While emission and reception are properties of the physical characteristics of the node, propagation is affected by the physics of the environment in which the network is placed. Simulators duplicate the processes of transmission and reception via method calls, and time the arrival or transmission of messages to mimic the real-world propagation standards. Simulators may also make available packages that try to recreate noise or channel loss (for example, the number of packets that are "dropped" may be calculated based on a simple probability).

4. **Network:** This is the environment in which nodes are present. They are usually defined as a physical space with boundaries. The physical characteristics of the network are well defined. These characteristics affect both the propagation of the radio signals and the mobility patterns of the nodes. A network may have a predefined number of nodes at the start of the simulation, but may also have nodes entering or leaving the network intermediately. The number of nodes is, thus, not a constant.

5. **Data traffic:** Traffic is defined as the process through which nodes generate data for different destinations. Data generation may follow different levels of periodicity, or follow none. They may follow various statistical distributions or processes like Poisson, Markov or other processes.

6. **Mobility:** Nodes are mobile entities. It is useful to find simulators that implement mobility models that can be directly used by nodes. At the least, some support for mobility or infrastructure for creating mobile nodes is expected.

3.12. Introduction Network Simulator (NS2)

NS2 is an open-source event-driven simulator designed specifically for research in computer communication networks. Since its inception in 1989, NS2 has continuously gained tremendous interest from industry, academia, and government. Having been under constant investigation and enhancement for years, NS2 now contains modules for numerous network components such as routing, transport layer protocol, application, etc. To investigate network performance, researchers can simply use an easy-to-use scripting language to configure a network, and observe results generated by NS2. Undoubtedly, NS2 has become the most widely used open source network simulator, and one of the most widely used network simulators. Network Simulator NS2 has proved useful in studying the dynamic nature of communication networks. Simulation of wired as well as wireless network functions and protocols (e.g., routing algorithms, TCP, UDP) can be done using NS2. In general, NS2 provides users with a way of specifying such network protocols and simulating their corresponding behaviors. Due to its flexibility and modular nature, NS2 has gained constant popularity in the networking research community since its birth in 1989. Ever since, several revolutions and revisions have marked the growing maturity of the tool, thanks to substantial contributions from the players in the field. Among these are the University of California and Cornell University, who developed the REAL network simulator, the foundation which NS is based on. Since 1995 the Defense Advanced Research Projects Agency (DARPA) supported development of NS through the Virtual Internetwork Test bed (VINT) project. Currently the National Science Foundation (NSF) has joined the ride in development. Last but not the least, the group of researchers and developers in the community are constantly working to keep NS2 strong and versatile.

➤ Features of Network Simulator (NS2)

- NS is an object oriented discrete event simulator:
 - Simulator maintains list of events and executes one event after another.
 - Single thread of control: no locking or race conditions.

- Back end is C++ event scheduler:
 - Protocols mostly.
 - Fast to run, more control.
- Front end is OTCL:
 - Creating scenarios, extensions to C++ protocols.
 - Fast to write and change.

➤ **Basic Architecture of Network Simulator(NS2)**

Figure 3.11 shows the basic architecture of NS2. NS2 provides users with executable command `ns` which take on input argument, the name of a Tcl simulation scripting file. Users are feeding the name of a Tcl simulation script (which sets up a simulation) as an input argument of an NS2 executable command `ns`. In most cases, a simulation trace file is created, and is used to plot graph and/or to create animation.

NS2 consists of two key languages: C++ and Object-oriented Tool Command Language (OTcl). While the C++ defines the internal mechanism (i.e., a backend) of the simulation objects, the OTcl sets up simulation by assembling and configuring the objects as well as scheduling discrete events (i.e., a frontend). The C++ and the OTcl are linked together using TclCL. Mapped to a C++ object, variables in the OTcl domains are sometimes referred to as handles. Conceptually, a handle (e.g., `n` as a Node handle) is just a string (e.g., `_o10`) in the OTcl domain, and does not contain any functionality. Instead, the functionality (e.g., receiving a packet) is defined in the mapped C++ object (e.g., of class `Connector`). In the OTcl domain, a handle acts as a frontend which interacts with users and other OTcl objects. It may define its own procedures and variables to facilitate the interaction. Note that the member procedures and variables in the OTcl domain are called instance procedures (`instprocs`) and instance variables (`instvars`), respectively. Before proceeding further, the readers are encouraged to learn C++ and OTcl languages. NS2 provides a large number of built-in C++ objects. It is advisable to use these C++ objects to set up a simulation using a Tcl simulation script. However, advance users may find these objects insufficient. They need to develop their own C++ objects, and use an OTcl configuration interface to put together these objects.

After simulation, NS2 outputs either text-based or animation-based simulation results. To interpret these results graphically and interactively, tools such as NAM (Network AniMator) and XGraph are used. To analyze a particular behavior of the network, users can extract a relevant subset of text-based data and transform it to a more conceivable presentation.

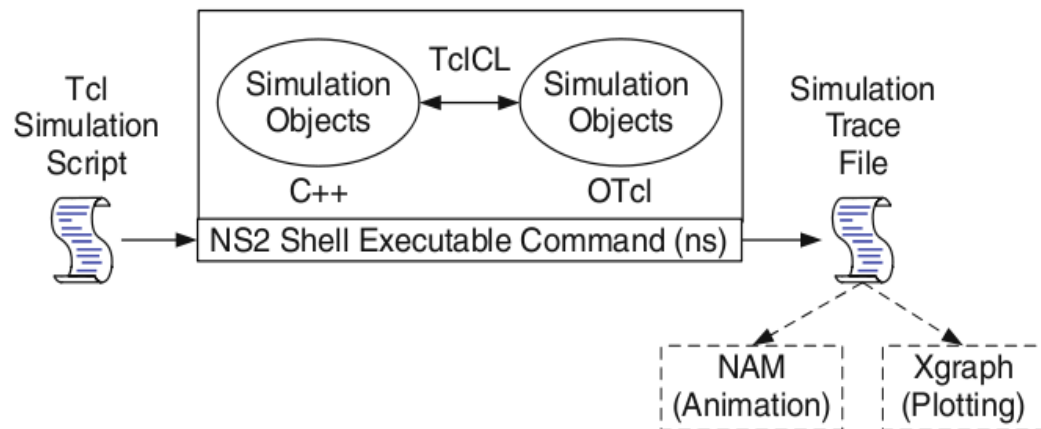


Figure 3.11 shows the basic architecture of NS2

➤ NS PROGRAMMING STRUCTURE

- Create the event scheduler.
- Turn on tracing.
- Create network topology.
- Create transport connections.
- Generate traffic.
- Insert errors.

➤ EVENT SCHEDULAR

In this Event scheduler while we processing many data's at a time it will process one by one (i.e.) FIFO concept, so there is no congestion while transferring the packets. PacketsIt is the collection of data, whether header is called or not all header files where present in the stack registers.



➤ TURN ON TRACING

Trace packets on individual link Trace file format

event	time	from node	to node	pkt type	pkt size	flags	fid	src addr	dst addr	seq num	pkt id
-------	------	--------------	------------	-------------	-------------	-------	-----	-------------	-------------	------------	-----------

```

r : receive (at to_node)
+ : enqueue (at queue)      src_addr : node.port (3.0)
- : dequeue (at queue)      dst_addr : node.port (0.0)
d : drop      (at queue)

```

```

r 1.3556 3 2 ack 40 ----- 1 3.0 0.0 15 201
+ 1.3556 2 0 ack 40 ----- 1 3.0 0.0 15 201
- 1.3556 2 0 ack 40 ----- 1 3.0 0.0 15 201
r 1.35576 0 2 tcp 1000 ----- 1 0.0 3.0 29 199
+ 1.35576 2 3 tcp 1000 ----- 1 0.0 3.0 29 199
d 1.35576 2 3 tcp 1000 ----- 1 0.0 3.0 29 199
+ 1.356 1 2 cbr 1000 ----- 2 1.0 3.1 157 207
- 1.356 1 2 cbr 1000 ----- 2 1.0 3.1 157 207

```

➤ CREATE NETWORK TOPOLOGY (PHYSICAL LAYER)

The Physical Layer is the first and lowest layer in the seven-layer OSI model of computer networking. The implementation of this layer is often termed PHY.

The Physical Layer consists of the basic hardware transmission technologies of a network. It is a fundamental layer underlying the logical data structures of the higher level functions in a network. Due to the plethora of available hardware technologies with widely varying characteristics, this is perhaps the most complex layer in the OSI architecture.

The Physical Layer defines the means of transmitting raw bits rather than logical data packets over a physical link connecting networking nodes. The bit stream may be grouped into code words or symbols and converted to a physical that is transmitted over hardware.

➤ **TRANSPORT CONNECTION (TRANSPORT LAYER)**

Transport layers are contained in both the TCP/IP. This is the foundation of the Internet. And the OSI model of general networking. The definitions of the Transport Layer are slightly different in these two models. This article primarily refers to the TCP/IP model, in which TCP is largely for a convenient application programming interface to internet hosts, as opposed to the OSI model of definition interface.

The most well-known transport protocol is the (TCP). It lent its name to the title of the entire internet protocol suite TCP/IP. It is used for connection-oriented transmissions, whereas the connectionless user datagram suite (UDP) is used for simpler messaging transmissions. TCP is the more complex protocol, due to its stateful design incorporating reliable transmission and data stream services.

➤ **GENERATE TRAFFIC (APPLICATION LAYER)**

In TCP/IP, the Application Layer contains all protocols and methods that fall into the realm of process-to-process communications via an Internet Protocol (IP) network using the Transport layer protocols to establish underlying host-to-host connections.

In the OSI model, the definition of its Application Layer is narrower in scope, explicitly distinguishing additional functionality above the Transport Layer at two additional levels: session layer and presentation layer OSI specifies strict modular separation of functionality at these layers and provides protocol for each layer.

➤ **INSERT ERRORS**

Start debugging of errors

➤ **Installation of Network Simulator (NS2)**

NS2 is a free simulation tool, which can be obtained from The Network Simulator Wiki. [Online]. Available: <http://nsnam.isi.edu/nsnam/index.php/>. It runs on various platforms including UNIX (or Linux), Windows, and Mac systems. Being developed in the UNIX environment, with no surprise, NS2 has the smoothest ride there, and so does its installation.

NS2 source codes are distributed in two forms: the all-in-one suite and the component-wise. With the all-in-one package, users get all the required components along with some optional components. This is basically a recommended choice for the beginners. This package provides an “install” script which configures the NS2 environment and creates NS2 executable file using the “make” utility.

The current all-in-one suite consists of the following main components:

- NS release 2.30,
- Tcl/Tk release 8.4.13,
- OTcl release 1.12, and
- TclCL release 1.18.

And the following are the optional components:

- NAM release 1.12: NAM is an animation tool for viewing network simulation traces and packet traces.
- Zlib version 1.2.3: This is the required library for NAM.
- Xgraph version 12.1: This is a data plotter with interactive buttons for panning, zooming, printing, and selecting display options.

The idea of the component-wise approach is to obtain the above pieces and install them individually. This option save considerable amount of downloading time and memory space. However, it could be troublesome for the beginners, and is therefore recommended only for experienced users.

The all-in-one suite can be installed in the Unix-based machines by simply running the install script and following the instructions therein. The only requirement is a computer with a C++ compiler installed. The following commands show how the all-in-one NS2 suite can be installed and validated, respectively:

```
shell>./install
```

```
shell>./validate
```

Validating NS2 involves simply running a number of working scripts that verify the essential functionalities of the installed components.

After completing the installation of ubuntu 10.04, update all the things in that. How to update means Goto the top menu, **system – Administration-update manager**

Step: 1

-Open the terminal. For this Goto **Application -Accessories-Terminal** (or press Ctrl+Alt+T)

-Then paste the following command

sudo apt-get install build-essential autoconf automake libxmu-dev gcc-4.3

Step: 2

Then we need to some files. So For this, open the Ns-allinone2.34 folder on the desktop, in that open folder OTCL1.13. In that folder, we need to modify the following files.

1. Modify Makefile.in file

In this file, u need to replace (**CC= @CC@**) With (**CC= gcc-4.3**) To replace this press **ctrl+H**.Then save the document.

2. Modify configure.in and configure files.

In these two files we need to modify. Means we need to replace (**ld-shared**)with (**gcc –shared**). For replacing **ctrl+H** then one dialog box will come. Then save the document.

Step: 3

Open the **ns-allinone2.34** folder which is present on the desktop,Then press **ctrl+L**. Then copy the path on the address bar.

Step: 4

Then open **Terminal** and do the following:

Type (\$cd “**paste the path here**” without quotes) and press enter.

ThenType **\$/install**

Then wait for some time, finally if successfully installed, it will display a message as “**for Related Posts**”. Otherwise some problems might have occurred.

Step: 5

Then execute the following command in the **Terminal** `$ gedit ~/.bashrc`

Then one document will be opened automatically.

Then delete all the content in that document and paste the following code in it.

```
# LD_LIBRARY_PATH
```

```
OTCL_LIB=/home/programmer/ns-allinone-2.33/otcl-1.13
```

```
NS2_LIB=/home/programmer/ns-allinone-2.33/lib
```

```
X11_LIB=/usr/X11R6/lib
```

```
USR_LOCAL_LIB=/usr/local/lib
```

```
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$OTCL_LIB:$NS2_LIB  
:$X11_LIB:$USR_LOCAL_LIB
```

```
# TCL_LIBRARY
```

```
TCL_LIB=/home/programmer/ns-allinone-2.33/tcl8.4.18/library
```

```
USR_LIB=/usr/lib
```

```
export TCL_LIBRARY=$TCL_LIB:$USR_LIB
```

```
# PATH
```

```
XGRAPH=/home/programmer/ns-allinone-2.33/bin:/home/programmer/ns-allinone-  
2.33/tcl8.4.18/unix:/home/programmer/ns-allinone-2.33/tk8.4.18/unix:/home/programmer/ns-  
allinone-2.33/xgraph-12.1/
```

```
NS=/home/programmer/ns-allinone-2.33/ns-2.33/
```

```
NAM=/home/programmer/ns-allinone-2.33/nam-1.13/
```

```
export PATH=$PATH:$XGRAPH:$NS:$NAM
```

Then after doing this, press Ctrl+H and replace the following:

/home/programmer/ns-allinone-2.33

Replace this with

/home/your system name/Desktop/ns-allinone-2.34

And in that document, at the 2nd line from the last, put **1.14** instead of **1.13**

And at the 3rd line from the last, make the change as (**ns-2.34**) instead of (**ns-2.33**) then save the document and close it.

Step: 6

Then finally copy the following command into the **Terminal**

“sudo ln -s /home/programmer/ns-allinone-2.33/ns-2.33/ns /usr/bin/ns”

After pasting it into the **Terminal**, just make the following modifications and then press Enter.

Make 2.33 as 2.34

And in the place of **“programmer”** in that command, just type your system name. Then the command should be like this.

“sudo ln -s /home/ur system name/Desktop/ns-allinone-2.34/ns- 2.34/ns /usr/bin/ns”

Then press Enter. It will ask for password, enter the password.

Step: 7

Then finally type **“ns”** in the **Terminal** and press Enter.

Then **“%”** symbol will be displayed.

Then it was successfully installed. Otherwise some problems might have occurred in the installation process.

Step: 8

Then finally execute the following command in the Terminal

Type **“sudo apt-get install xgraph”** and press enter.

Finished installing Network Simulator.

To run NS2 on Windows-based operating systems, a bit of tweaking is required. Basically, the idea is to make Windows-based machines emulate the functionality of the Unix-like environment. A popular program that performs this job is Cygwin (Cygwin is available online and comes free. Information such as how to obtain and install Cygwin is available online at the Cygwin website www.cygwin.com). After getting Cygwin to work, the same procedure as that of Unix-based installation can be followed. For ease of installation, it is recommended that the all-in-one package be used. The detailed description of Windows-based installation can be found online at NS2's Wiki site, where the information on post-installation troubles can also be found. Note that by default Cygwin does not install all

packages necessary to run NS2. A user needs to manually install the addition packages shown in Table 3.1.

Table 3.1.Additional Cygwin packages required to run NS2.

Category	Packages
Development	gcc, gcc-objc, gcc-g++, make
Utils	patch
X11	xorg-x11-base, xorg-x11-devel

3.12.1 Main NS2 Simulation Steps

The followings show the three key step guideline in defining a simulation scenario in a NS2:

Step 1: Simulation Design

The first step in simulating a network is to design the simulation. In this step, the users should determine the simulation purposes, network configuration and assumptions, the performance measures, and the type of expected results.

Step 2: Configuring and Running Simulation

This step implements the design in the first step. It consists of two phases:

- *Network configuration phase:* In this phase network components (e.g., node, TCP and UDP) are created and configured according to the simulation design. Also, the events such as data transfer are scheduled to start at a certain time.
- *Simulation Phase:* This phase starts the simulation which was configured in the Network Configuration Phase. It maintains the simulation clock and executes events chronologically. This phase usually runs until the simulation clock reached a threshold value specified in the Network Configuration Phase.

In most cases, it is convenient to define a simulation scenario in a Tcl scripting file (e.g., <file>) and feed the file as an input argument of an NS2 invocation (e.g., executing “ns <file>”).

Step 3: Post Simulation Processing

The main tasks in this step include verifying the integrity of the program and evaluating the performance of the simulated network. While the first task is referred to as *debugging*, the second one is achieved by properly collecting and compiling simulation results.

3.12.2 Network AniMation (NAM) Trace

NAM trace is records simulation detail in a text file, and uses the text file the play back the simulation using animation. NAM trace is activated by the command “\$ns namtrace-all \$file”, where ns is the Simulator handle and file is a handle associated with the file (e.g., out.nam) which stores the NAM trace information. After obtaining a NAM trace file, the animation can be initiated directly at the command prompt through the following command:

```
>>namfilename.nam
```

Many visualization features are available in NAM. These features are for example animating colored packet flows, dragging and dropping nodes (positioning), labeling nodes at a specified instant, shaping the nodes, coloring a specific link, and monitoring a queue.

3.12.3 Including C++ Modules into NS2 and the *make* Utility

In developing an NS2 simulation, very often it is necessary to create the customized C++ modules to complement the existing libraries. As such, the developer is faced with the task of keeping track of all the created files as a part of NS2. When a change is made to one file, usually it requires recompilation of some other files that depend on it. Manual recompilation of each of such files may not be practical. In UNIX, a utility tool called make is available to overcome such difficulties. In this section we introduce this tool and discuss how to use it in the context of NS2 simulation development. As a UNIX utility tool make is very useful for managing the development of software written in any compliable programming language including C++.

Generally, the make program automatically keeps track of all the files created throughout the development process. By *keeping track*, we mean recompiling or relinking wherever interdependencies exist among these files, which may have been modified as a part of the development process.

3.12.4 An Invocation of a Make Utility

A “make” utility can be invoked form a UNIX shell with the following command:


```
>>make [-f mydescriptor]
```

Where “make” is mandatory, while the text inside the bracket is optional. By default (i.e., without optional input arguments), the make utility recompiles and relinks the source codes according to what specified in the default descriptor file Makefile. If the descriptor file mydescriptor is specified, the utility is using this file in place of the default file Makefile.

3.13. Requirements

➤ **Hardware:**

- ☐ Single PC.
- ☐ 20 GB Hard disc space.
- ☐ 1 GB RAM.

➤ **Software:**

- ☐ Linux OS (Ubuntu 10.04).
- ☐ NS2.34.

Basics of Tcl Programming (w.r.t. ns2)

Before we get into the program we should consider the following things:

- Initialization and termination aspects of network simulator.
- Defining the network nodes, links, queues and topology as well.
- Defining the agents and their applications.
- Network Animator (NAM).
- Tracing.

➤ **Initialization**

To start a new simulator we write

set ns [new Simulator]

From the above command we get that a variable ns is being initialized by using the set command. Here the code [new Simulator] is an instantiation of the class Simulator which uses the reserved word 'new'. So we can call all the methods present inside the class simulator by using the variable ns.

Creating the output files

```
1  #to create the trace files we write
2
3  set tracefile1 [open out.tr w]
4  $ns trace-all $tracefile1
5
6  #To create the nam files we write
7
8  set namfile1 [open out.nam w]
9  $ns namtrace-all $namfile
```

In the above we create an output trace file out.tr and a nam visualization file out.nam. But in the Tcl script they are not called by their names declared, while they are called by the pointers initialized for them such as tracefile1 and namfile1 respectively. The line which starts with '#' is commented. The next line opens the file 'out.tr' which is used for writing is declared 'w'. The next line uses a simulator method trace-all by which we will trace all the events in a particular format.

The termination program is done by using a 'finish' procedure

```
01  # Defining the 'finish' procedure'
02
03  proc finish { } {
04      global ns tracefile1 namfile1
05      $ns flush-trace
06      close $tracefile
07      close $namfile
```

```

08     exec namout.nam&
09     exit 0
10 }

```

In the above the word 'proc' is used to declare a procedure called 'finish'. The word 'global' is used to tell what variables are being used outside the procedure.

'flush-trace' is a simulator method that dumps the traces on the respective files. The command 'close' is used to close the trace files and the command 'exec' is used to execute the nam visualization. The command 'exit' closes the application and returns 0 as zero (0) is default for clean exit.

In ns we end the program by calling the 'finish' procedure

```

1  #end the program
2  $ns at 125.0 "finish"

```

Thus the entire operation ends at 125 seconds. To begin the simulation we will use the command

```

1  #start the simulation process
2  $ns run

```

➤ Defining nodes, links, queues and topology

Way to create a node:

view source

print?

```

1     set n0 [ns node]

```

In the above we created a node that is pointed by a variable n0. While referring the node in the script we use \$n0. Similarly we create another node n2. Now we will set a link between the two nodes.

```
1 $ns duplex-link $n0 $n2 10Mb 10ms DropTail
```

So we are creating a bi-directional link between n0 and n2 with a capacity of 10Mb/sec and a propagation delay of 10ms.

In NS an output queue of a node is implemented as a part of a link whose input is that node to handle the overflow at the queue. But if the buffer capacity of the output queue is exceeded then the last packet arrived is dropped and here we will use a 'DropTail' option. Many other options such as RED(Random Early Discard) mechanism, FQ(Fair Queuing), DRR(Deficit Round Robin), SFQ(Stochastic Fair Queuing) are available.

So now we will define the buffer capacity of the queue related to the above link

```
1 #Set queue size of the link
```

```
2 $ns queue-limit $n0 $n2 20
```

So, if we summarize the above three things we get

```
01 #create nodes
```

```
02
```

```
03 set n0 [$ns node]
```

```
04 set n1 [$ns node]
```

```
05 set n2 [$ns node]
```

```
06 set n3 [$ns node]
```

```
07 set n4 [$ns node]
```

```
08 set n5 [$ns node]
```

```
09
```

```
10 #create links between the nodes
```

```
11
```

```
12 $ns duplex-link $n0 $n2 10Mb 10ms DropTail
```

```
13 $ns duplex-link $n1 $n2 10Mb 10ms DropTail
```

```
14 $ns simplex-link $n2 $n3 0.3Mb 100ms DropTail
```

```

15 $ns simplex-link $n3 $n2 0.3Mb 100ms DropTail
16 $ns duplex-link $n0 $n2 0.5Mb 40ms DropTail
17 $ns duplex-link $n0 $n2 0.5Mb 40ms DropTail
18
19 #set queue-size of the link (n2-n3) to 20
20 $ns queue-limit $n2 $n3 20

```

➤ Agents and applications

- TCP

TCP is a dynamic reliable congestion protocol which is used to provide reliable transport of packets from one host to another host by sending acknowledgements on proper transfer or loss of packets. Thus TCP requires bi-directional links in order for acknowledgements to return to the source.

Now we will show how to set up TCP connection between two nodes :

```

1 #setting a tcp connection
2
3 set tcp [new Agent/TCP]
4 $ns attach-agent $n0 $tcp
5 set sink [new Agent/TCPSink]
6 $ns attach-agent $n4 $sink
7 $ns connect $tcp $sink
8 $tcp set fid_1
9 $tcp set packetSize_552

```

The command 'set tcp [new Agent/TCP]' gives a pointer called 'tcp' which indicates the tcp agent which is a object of ns. Then the command '\$ns attach-agent \$n0 \$tcp' defines the source node of tcp connection. Next the command 'set sink [new Agent/TCPSink]' defines the destination of tcp by a pointer called sink. The next command '\$ns attach-agent \$n4 \$sink' defines the destination node as n4. Next, the command '\$ns connect \$tcp \$sink' makes the TCP connection between the source and the destination i.e. n0 and n4. When we have several flows

such as TCP, UDP etc. in a network. So, to identify these flows we mark these flows by using the command '\$tcp set fid_1'. In the last line we set the packet size of tcp as 552 while the default packet size of tcp is 1000.

FTP over TCP

File Transfer Protocol (FTP) is a standard mechanism provided by the Internet for transferring files from one host to another. Well this is the most common task expected from a networking or a inter networking. FTP differs from other client server applications in that it establishes between the client and the server. One connection is used for data transfer and other one is used for providing control information. FTP uses the services of the TCP. It needs two connections. The well Known port 21 is used for control connections and the other port 20 is used for data transfer.

Well here we will learn in how to run a FTP connection over a TCP

```
1  #Initiating FTP over TCP
2
3  set ftp [new Application/FTP]
4  $ftp attach-agent $tcp
```

In above,the command 'set ftp [new Application/FTP]' gives a pointer called 'ftp' which indicates the FTP application.Next, we attach the ftp application with tcp agent as FTP uses the services of TCP.

UDP

The User datagram Protocol is one of the main protocols of the Internet protocol suite.UDP helps the host to send messages in the form of datagrams to another host which is present in an Internet protocol network without any kind of requirement for channel transmission setup. UDP provides a unreliable service and the datagrams may arrive out of order,appear

duplicated, or go missing without notice. UDP assumes that error checking and correction is either not necessary or performed in the application, avoiding the overhead of such processing at the network interface level. Time-sensitive applications often use UDP because dropping packets is preferable to waiting for delayed packets, which may not be an option in a real-time system.

Now we will learn how to create a UDP connection in network simulator.

```
1  # setup a UDP connection
2  set udp [new Agent/UDP]
3  $ns attach-agent $n1 $udp
4  $set null [new Agent/Null]
5  $ns attach-agent $n5 $null
6  $ns connect $udp $null
7  $udp set fid_2
```

Similarly, the command 'set udp [new Agent/UDP]' gives a pointer called 'udp' which indicates the udp agent which is a object of ns. Then the command '\$ns attach-agent \$n1 \$udp' defines the source node of udp connection. Next the command 'set null [new Agent/Null]' defines the destination of udp by a pointer called null. The next command '\$ns attach-agent \$n5 \$null' defines the destination node as n5. Next, the command '\$ns connect \$udp \$null' makes the UDP connection between the source and the destination i.e. n1 and n5. When we have several flows such as TCP, UDP etc. in a network. So, to identify these flows we mark these flows by using the command '\$udp set fid_2

Constant Bit Rate (CBR)

Constant Bit Rate (CBR) is a term used in telecommunications, relating to the quality of service. When referring to codecs, constant bit rate encoding means that the rate at which a codec's output data should be consumed is constant. CBR is useful for streaming multimedia content on limited capacity channels since it is the maximum bit rate that matters, not the average, so CBR would be used to take advantage of all of the capacity. CBR would not be the

optimal choice for storage as it would not allocate enough data for complex sections (resulting in degraded quality) while wasting data on simple sections.

CBR over UDP Connection

```
1  #setup cbr over udp
2
3  set cbr [new Application/Traffic/CBR]
4  $cbr attach-agent $udp
5  $cbr set packetSize_1000
6  $cbr set rate_0.01Mb
7  $cbr set random _false
```

In the above we define a CBR connection over a UDP one. Well we have already defined the UDP source and UDP agent as same as TCP. Instead of defining the rate we define the time interval between the transmission of packets in the command '\$cbr set rate_0.01Mb'. Next, with the help of the command '\$cbr set random _false' we can set random noise in cbr traffic. we can keep the noise by setting it to 'false' or we can set the noise on by the command '\$cbr set random _1'. We can set by packet size by using the command '\$cbr set packetSize_(packetsize)'. We can set the packet size up to sum value in bytes.

Scheduling Events

In ns the tcl script defines how to schedule the events or in other words at what time which event will occur and stop. This can be done using the command

\$ns at .

So here in our program we will schedule the ftp and cbr.

```
1  # scheduling the events
2
3  $ns at 0.1 "cbr start"
```


- 4 \$ns at 1.0 "ftp start"
- 5 \$ns at 124.0 "ftp stop"
- 6 \$ns at 124.5 "cbr stop"

➤ **Network Animator(NAM)**

When we will run the above program in ns then we can visualize the network in the NAM. But instead of giving random positions to the nodes, we can give suitable initial positions to the nodes and can form a suitable topology. So, in our program we can give positions to the nodes in NAM in the following way

- 1 #Give position to the nodes in NAM
- 2
- 3 \$ns duplex-link-op \$n0 \$n2 orient-right-down
- 4 \$ns duplex-link-op \$n1 \$n2 orient-right-up
- 5 \$ns simplex-link-op \$n2 \$n3 orient-right
- 6 \$ns simplex-link-op \$n3 \$n2 orient-left
- 7 \$ns duplex-link-op \$n3 \$n4 orient-right-up
- 8 \$ns duplex-link-op \$n3 \$n5 orient-right-down

We can also define the color of cbr and tcp packets for identification in NAM. For this we use the following command

- 1 # marking the flows
- 2 \$ns color1 Blue
- 3 \$ns color2 Red

To view the network animator we need to type the command: nam

➤ **Tracing**

Tracing Objects

NS simulation can produce visualization trace as well as ASCII file corresponding to the events that are registered at the network. While tracing ns inserts four objects: EnqT, DeqT, RecvT & DrpT. EnqT registers information regarding the arrival of packet and is queued at the input queue of the link. When overflow of a packet occurs, then the information of the dropped packet is registered in DrpT. DeqT holds the information about the packet that is dequeued instantly. RecvT hold the information about the packet that has been received instantly.

Structure of Trace files

- The first field is event. It gives you four possible symbols '+' '-' 'r' 'd'. These four symbols correspond respectively to enqueued, dequeued, received and dropped.
- The second field gives the time at which the event occurs.
- The third field gives you the input node of the link at which the event occurs.
- The fourth field gives you the output node at which the event occurs.
- The fifth field shows the information about the packet type i.e. whether the packet is UDP or TCP.
- The sixth field gives the packet size.
- The seventh field gives information about some flags.
- The eighth field is the flow id (fid) for IPv6 that a user can set for each flow in atcl script. It is also used for specifying the color of flow in NAM display.
- The ninth field is the source address.
- The tenth field is the destination address.
- The eleventh field is the network layer protocol's packet sequence number.

- The last field shows the unique id of packet.

Following are trace of two events:

r 1.84471 2 1 cbr 210 ----- 1 3.0 1.0 195 600

r 1.84566 2 0 ack 40 ----- 2 3.2 0.1 82 602

The trace file can be viewed with the cat command:

cat out.tr

➤ IMPLEMENTATION

- **Creating Nodes with respective to the simulator name**

set node_(0) [\$ns_ node]

set node_(1) [\$ns_ node]

set node_(2) [\$ns_ node]

set node_(3) [\$ns_ node]

- **Declaring positions of Nodes**

\$node_(0) set X_ 0.608377307314

\$node_(0) set Y_ 50.446991827566

\$node_(1) set X_ 150.337311778721

\$node_(1) set Y_ 50.582820874924

\$node_(2) set X_ 130.337311778721

\$node_(2) set Y_ 50.582820874924

\$node_(3) set X_ -36.337311778721

\$node_(3) set Y_ 181.582820874924

- **Assigning Configuration:**

```
set chan0 [new $val(chan)]
$ns_ node-config -adhocRouting $val(rp) \
    -llType $val(ll) \
    -macType $val(mac) \
    -ifqType $val(ifq) \
    -ifqLen $val(ifqlen) \
    -antType $val(ant) \
    -propType $val(prop) \
    -phyType $val(netif) \
    -channelType $val(chan) \
    -topoInstance $topo \
    -agentTrace ON \
    -routerTrace ON \
    -macTrace ON \
    -movementTrace ON \
```

- **Giving Movable feature to the nodes:**

```
$ns_ at 0 "$node_(0) setdest 242.000000000000 149.0070000000000000 20.000000000000"
$ns_ at 1 "$node_(1) setdest 250.610000000000 50.000000000000 100.000000000000"
```

- **Finishing the task:**

```
proc stop {} {
    global ns_ tracefdvalenv
    $ns_ flush-trace
    close $tracefd
    execnamrelay.nam&
    execawk -f graph1.awk out.tr > x
    execxgraph x &
    exit 0
}
```

- **Declaration of variables:**

```
maxseq_no=0;
recevepkt = 0;
sendspkt = 0;
routingpkts = 0;
receivespkt = 0;
sum = 0;
drop = 0;
```

- **Conditions for getting efficiency in X-graph:**

```
if ( event == "+" && source == "0.0")
    sendspkt++;
if ( event == "r" && dest == "3.0" )
    recevepkt++;
if ( event == "d" && pkt == "tcp")
    drop++;
if ( seq_no>maxseq_no ) maxseq_no = seq_no;
if ( ! ( seq_no in timeIn ) ) timeIn[seq_no] = time;
if ( event != "d" ) {
if ( event == "r" ) timeFim[seq_no] = time;
    } else timeFim[seq_no] = 0;
}
```

DESIGN

Introduction:

The purpose of the design phase is to plan a solution of the problem specified by the requirement document. This phase is the first step in moving from problem domain to the solution domain. The design of a system is perhaps the most critical factor affecting the quality of the software, and has a major impact on the later phases, particularly testing and maintenance. The output of this phase is the design document. This document is similar to a blue print or plan for the solution, and is used later during implementation, testing and maintenance.

The design activity is often divided into two separate phase-system design and detailed design. System design, which is sometimes also called top-level design, aims to identify the modules that should be in the system, the specifications of these modules, and how they interact with each other to produce the desired results. At the end of system design all the major data structures, file formats, output formats, as well as the major modules in the system and their specifications are decided.

During detailed design the internal logic of each of the modules specified in system design is decided. During this phase further details of the data structures and algorithmic design of each of the modules is specified. The logic of a module is usually specified in a high-level design description language, which is independent of the target language in which the software will eventually be implemented. In system design the focus is on identifying the modules, whereas during detailed design the focus is on designing the logic for each of the modules. In other words, in system design the attention is on what components are needed, while in detailed design how the components can be implemented in software is the issue.

During the design phase, often two separate documents are produced. One for the system design and one for the detailed design.

Together, these documents completely specify the design of the system. That is they specify the different modules in the system and internal logic of each of the modules.

A design methodology is a systematic approach to creating a design by application of set of techniques and guidelines. Most methodologies focus on system design. The two basic principles used in any design methodology are problem partitioning and abstraction.

A large system cannot be handled as a whole, and so for design it is partitioned into smaller systems. Abstraction is a concept related to problem partitioning. When partitioning is used during design, the design activity focuses on one part of the system at a time. Since the part being designed interacts with other parts of the system, a clear understanding of the interaction is essential for properly designing the part. For this, abstraction is used. An abstraction of a system or a part defines the overall behavior of the system at an abstract level without giving the internal details.

While working with the part of a system, a designer needs to understand only the abstractions of the other parts with which the part being designed interacts. The use of abstraction allows the designer to practice the "divide and conquer" technique effectively by focusing one part at a time, without worrying about the details of other parts.

Like every other phase, the design phase ends with verification of the design. If the design is not specified in some executable language, the verification has to be done by evaluating the design documents. One way of doing this is thorough reviews. Typically, at least two design reviews are held-one for the system design and one for the detailed and one for the detailed design.

Design Objectives:

- Design models help us to visualize a system as it is or as we want it to be.
- Design models permits us to specify the structure or behavior of the system.
- Design models give us a template that guides us in constructing a system.
- Design models document the decision we have made.

Testing and debugging

Testing and debugging a program is one of the most tedious parts of computer programming. The testing and debugging phase of a project can easily take more time than it took to write the application. Testing includes both checking that the code runs at all, that it runs correctly under all circumstances, and that it runs the same way it did before you made changes. Tcl's error diagnostics make it easy to track down coding errors; the modular nature of Tcl code makes it easy to do unit testing of functions, and the tcl test package makes it easy to write integrated regression test suites.

Debugging Code

The first step to debugging a Tcl script is to examine the Tcl error output closely. Tcl provides verbose error information that leads you to the exact line where a coding error occurs. Tcl error messages consist of a set of lines. The first line will describe the immediate cause of the error (Incorrect number of arguments, invalid argument, undefined variable, etc). The rest of the message describes more details about where the error occurs. For example, this procedure has a fairly common error - the closing brace and bracket are in the wrong order:

```
prochasError {a} {  
  
return [expr {$a+2}]  
  
}
```

The error message is:

missing close-bracket

while executing

```
"return [expr {$a+2}]
```


"

(procedure "hasError" line 2)

invoked from within

"hasError 1"

The first line describes the error (missing curly bracket), and the rest of the lines show the exact line in the program (return [expr {\$a+2}]), and where that line occurs (second line in the hasError procedure) Here are a few of the common error messages and a description of the code that generates them.

wrong # args: should be ...

set result "a b" button .b -text set -command "set x \$result" This example looks reasonable, however, the command to be evaluated when the button is clicked is created by substituting the \$result and concatenating the values into a string resembling this: set x a b. The recommended way of creating a button like this is to use the list command to maintain grouping:

set result "a b"

button .b -text set -command [list set x \$result]

missing "

missing close-brace

missing close-bracket

Your code has an unterminated string that starts with a double-quote. The usual causes of this are typos (not hitting shift fast enough and having a double quote at one end of a string and a single quote at the other), having a space after a back-slash line continuation character, or mismatching the open/close pairs of a set of nested quotes, braces and brackets. Here are some examples of lines that would generate one of these errors.

*setrtn "there is a space after the backslash *

causing this to generate an error"

puts "Missing close quote

puts "mismatched quote and brace}

puts "should be double quote to close'

can't read "...": no such variable

There is a \$varName in your code for which varName has never been set. This can happen:

- Because you were reworking code and forgot to initialize a variable.
- Because a global scope variable is referenced in a procedure without declaring it global.
- Because a procedure invoked from a widget did not declare the widget's –text variable to be in global scope.
- Because code is being evaluated outside the scope in which it was created. Code connected with a button, after event, etc is evaluated in the global scope (unless a namespace or class scope is assigned) even if the widget is created within a procedure or method. Procedure scope local variables will no longer exist after the GUI has taken control of the application.

setglobalVar 1

Fails because of missing "global globalVar"

prochaseError {} {

puts "\$globalVar"

```
}
```

The entry widget is OK, but the button will cause an error.

```
entry .e -textvariableglobalVar
```

```
button .b -text "generateError" -command hasError
```

This button references a local variable.

```
procmakeBadButton {} {
```

```
set xx "local variable"
```

```
button .b -text "throw error" -command "puts $xx"
```

```
grid .b
```

```
}
```

Invalid command name "..."

The first word on a command line is not a valid command or procedure name. This is most often caused by mis-typing a command name, or forgetting to source or package require the Tcl code that defines a command or procedure.

syntax error in expression "...": variable references require preceding \$

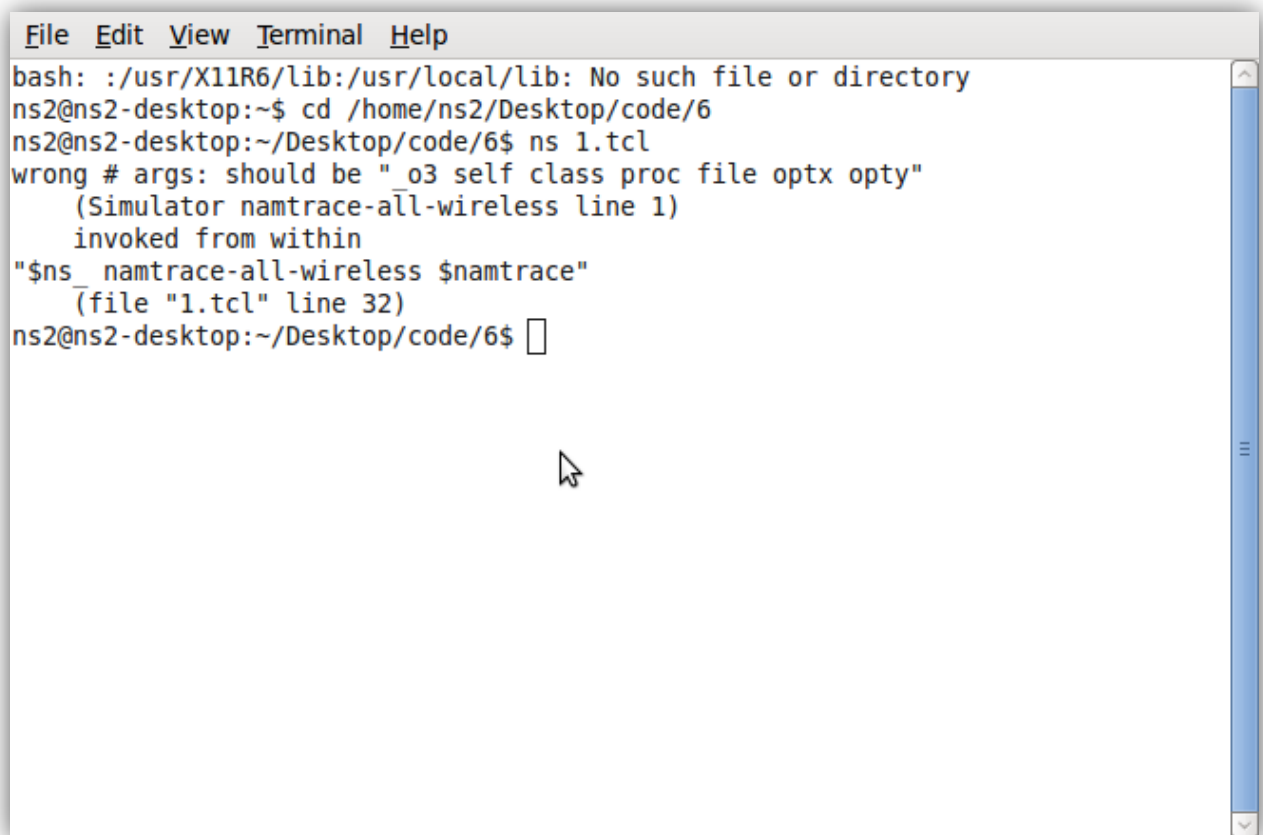
invalidbareword "..."

This error is generated by the expr command. It's caused when you try to do arithmetic on a string. The usual causes are that you forgot to put a dollar-sign on a variable name or that a variable that should hold a number was assigned a string value.

BLACK BOX TESTING

Black box testing also called behavioral testing focuses on the functional requirements of the software. That is black box testing enables the software engineer to derive sets of input conditions that will fully exercise all functional requirements for a program. Black box testing attempts to find errors in the following categories:

- Incorrect or missing functions.
- Interface errors.
- Errors in data structures or external data base access Behavior or performance errors.
- Initialization and termination errors

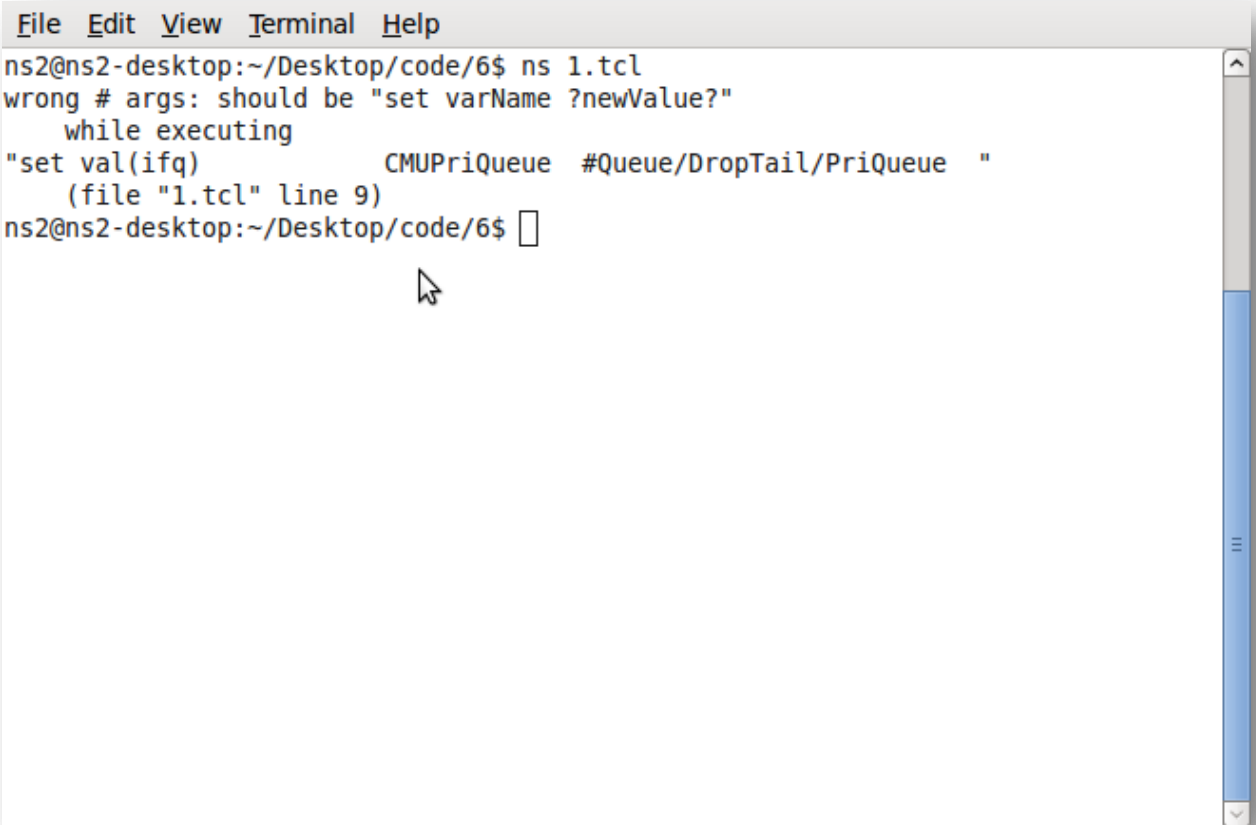


```
File Edit View Terminal Help
bash: ./usr/X11R6/lib:/usr/local/lib: No such file or directory
ns2@ns2-desktop:~$ cd /home/ns2/Desktop/code/6
ns2@ns2-desktop:~/Desktop/code/6$ ns 1.tcl
wrong # args: should be " o3 self class proc file optx opty"
(Simulator namtrace-all-wireless line 1)
invoked from within
"$ns_ namtrace-all-wireless $namtrace"
(file "1.tcl" line 32)
ns2@ns2-desktop:~/Desktop/code/6$
```

Figure3.1 **Black** box Testing:

WHITE BOX TESTING

White box testing sometimes called glass box testing is a test case design method that uses the control structure of the procedural design to derive test cases. Using white box testing methods, the software engineer can derive test cases that guarantee that all independent paths within a module have been exercised at least once. Exercise all logical decisions on their true and false sides. Execute all loops at their boundaries and within their operational bounds. Exercise internal data structures to ensure their validity

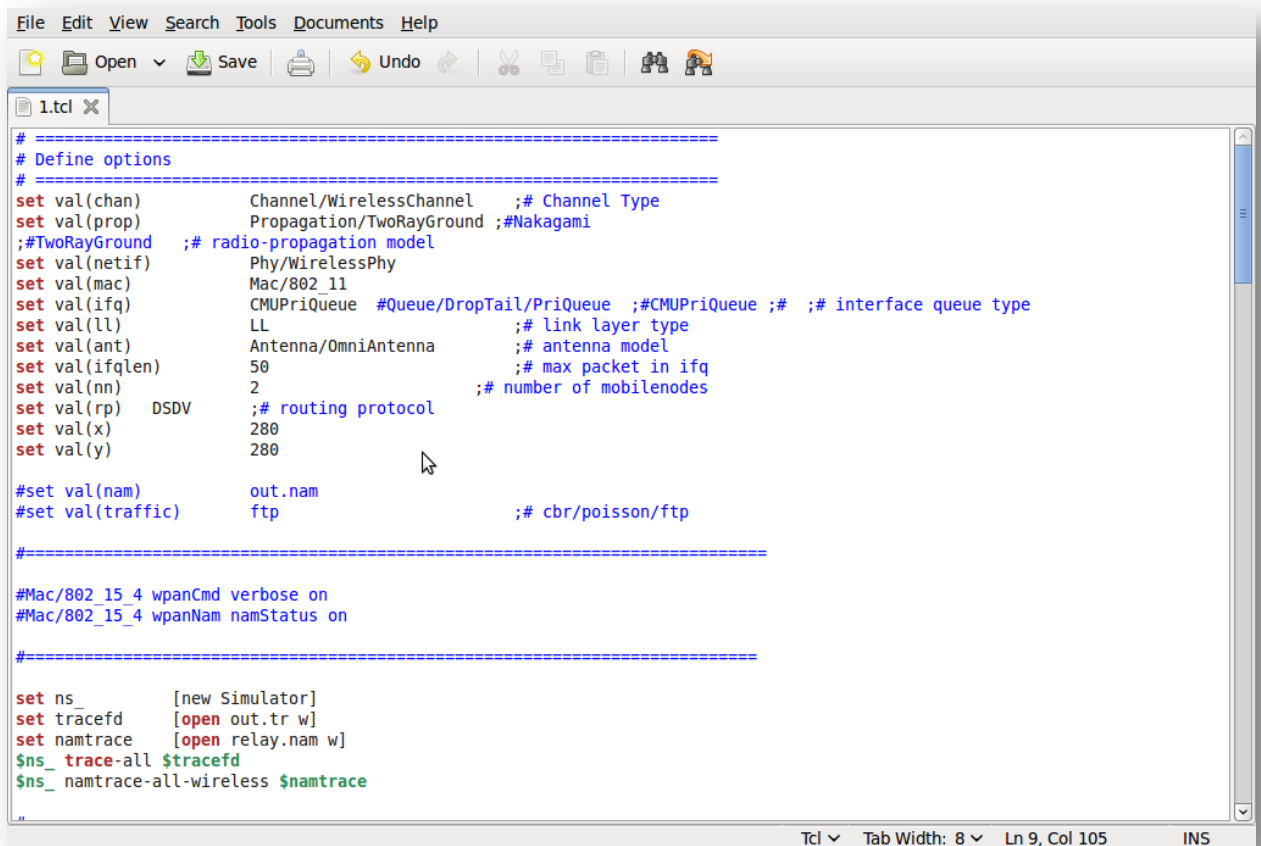
A screenshot of a terminal window with a menu bar (File, Edit, View, Terminal, Help) and a title bar. The terminal shows a user running a Tcl script named '1.tcl'. The script has an error: 'wrong # args: should be "set varName ?newValue?"' at line 9. The error message is indented. The user's prompt is 'ns2@ns2-desktop:~/Desktop/code/6\$'.

```
File Edit View Terminal Help
ns2@ns2-desktop:~/Desktop/code/6$ ns 1.tcl
wrong # args: should be "set varName ?newValue?"
    while executing
    "set val(ifq)          CMUPriQueue  #Queue/DropTail/PriQueue  "
    (file "1.tcl" line 9)
ns2@ns2-desktop:~/Desktop/code/6$
```

Figure 3.1 White Box Testing

UNIT TESTING

The most ‘micro’ scale of testing to test particular functions or code modules. Typically, it is done by the programmer and not by tester, as it requires detailed knowledge of the internal program design and code. Not always easily done unless the application has a well-designed architecture with tight code; may require developing test modules or test harnesses.



```
File Edit View Search Tools Documents Help
Open Save Undo
1.tcl
# =====
# Define options
# =====
set val(chan) Channel/WirelessChannel ;# Channel Type
set val(prop) Propagation/TwoRayGround ;#Nakagami
;#TwoRayGround ;# radio-propagation model
set val(netif) Phy/WirelessPhy
set val(mac) Mac/802_11
set val(ifq) CMUPriQueue #Queue/DropTail/PriQueue ;#CMUPriQueue ;# ;# interface queue type
set val(ll) LL ;# link layer type
set val(ant) Antenna/OmniAntenna ;# antenna model
set val(ifqlen) 50 ;# max packet in ifq
set val(nn) 2 ;# number of mobilenodes
set val(rp) DSDV ;# routing protocol
set val(x) 280
set val(y) 280

#set val(nam) out.nam
#set val(traffic) ftp ;# cbr/poisson/ftp

#=====

#Mac/802_15_4 wpanCmd verbose on
#Mac/802_15_4 wpanNam namStatus on

#=====

set ns_ [new Simulator]
set tracefd [open out.tr w]
set namtrace [open relay.nam w]
$ns_ trace-all $tracefd
$ns_ namtrace-all-wireless $namtrace

Tcl Tab Width: 8 Ln 9, Col 105 INS
```

Figure 3.1 Unit Testing

Result analysis:

We have implemented the security ID system, which is based on the EAACK system. We have tested various previous systems, like Basic AODV, TOW-ACK, AACK, and EAACK and our proposed HCEAACK. we have considered a network as 10 nodes with single source and destination with possible of two routes. Source is node 0 and destination is node 9.

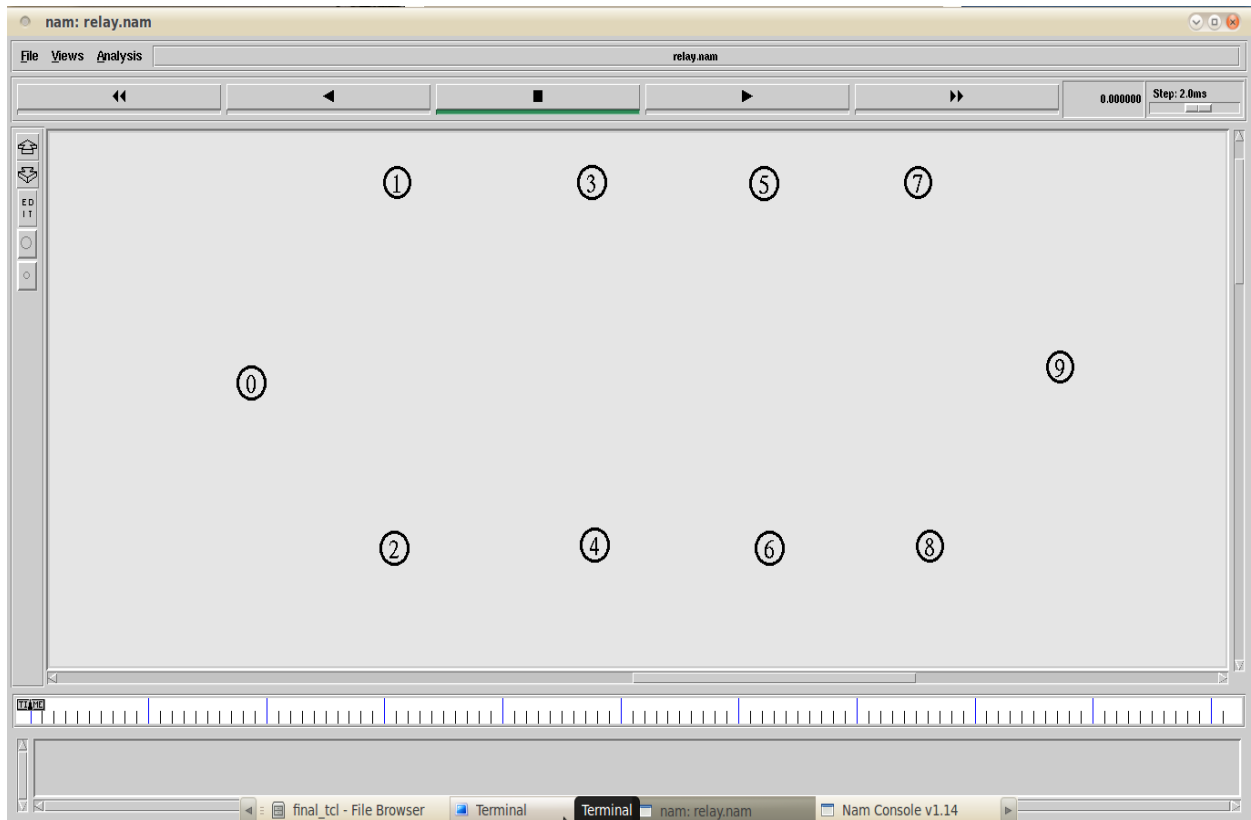


Figure4.1 network scenario with source and destination

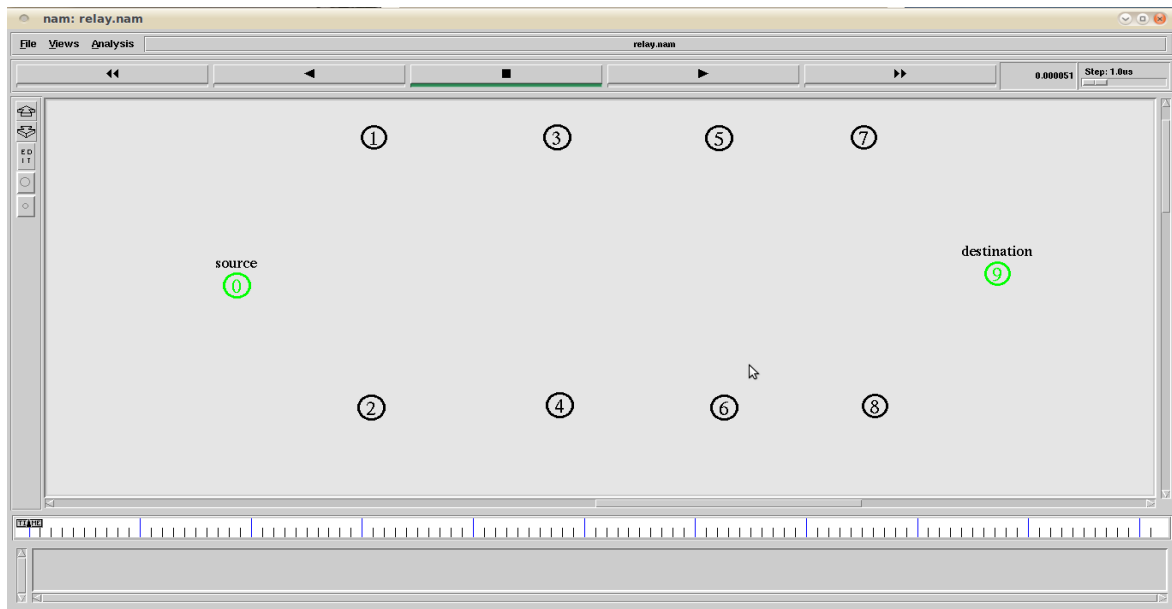


Figure4.2source node and destination node

In our work we have considered on-demand routing protocol as the basic routing. So to transfer the data from source to destination the device must check the route to destination by sharing the Routed request and route reply.

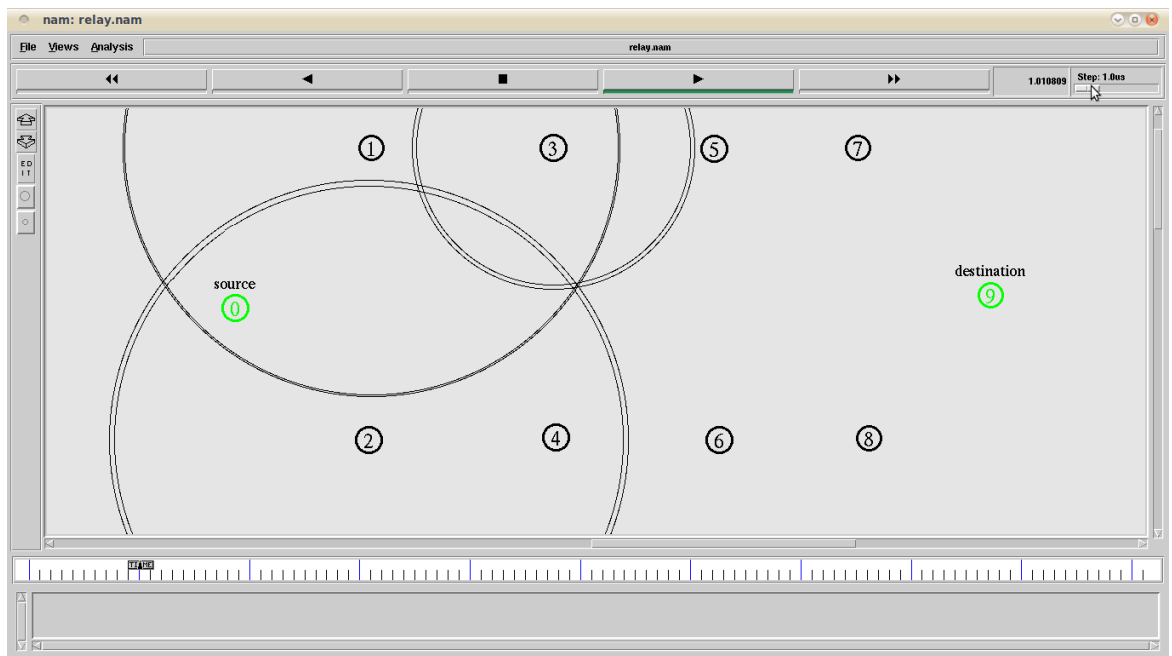


Figure4.3Route request sharing through intermediate nodes

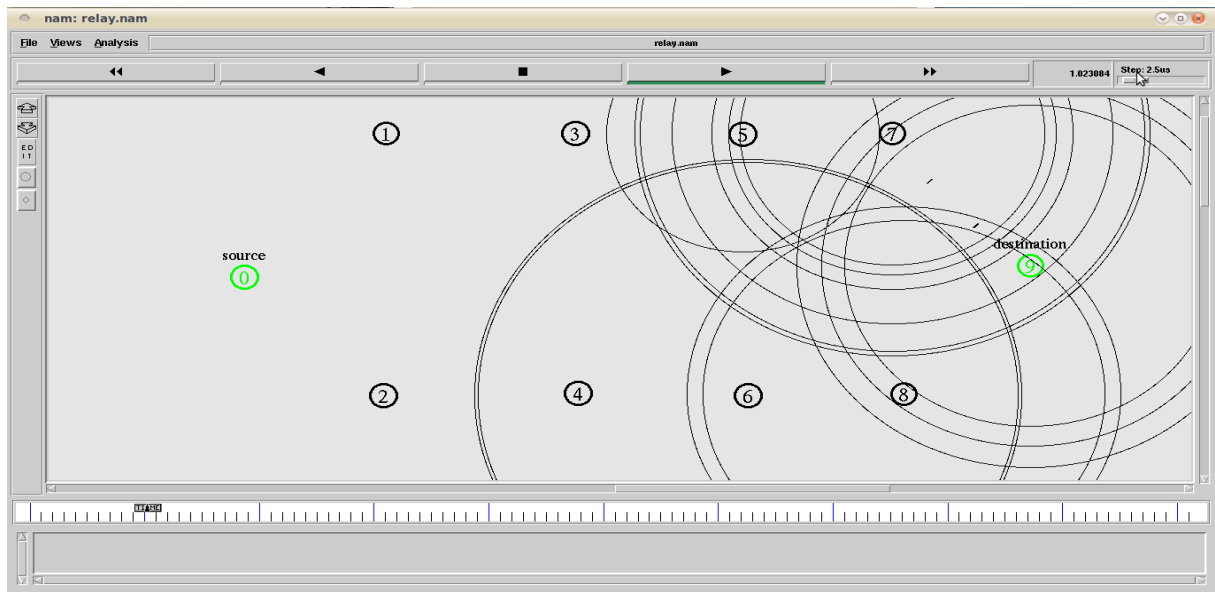


Figure4.4route reply from destination

We have tested and compared our model with some previous models. In that first we have tested the basic AODV model without security and without attack. In basic model there is no consideration on the acknowledgement sharing.

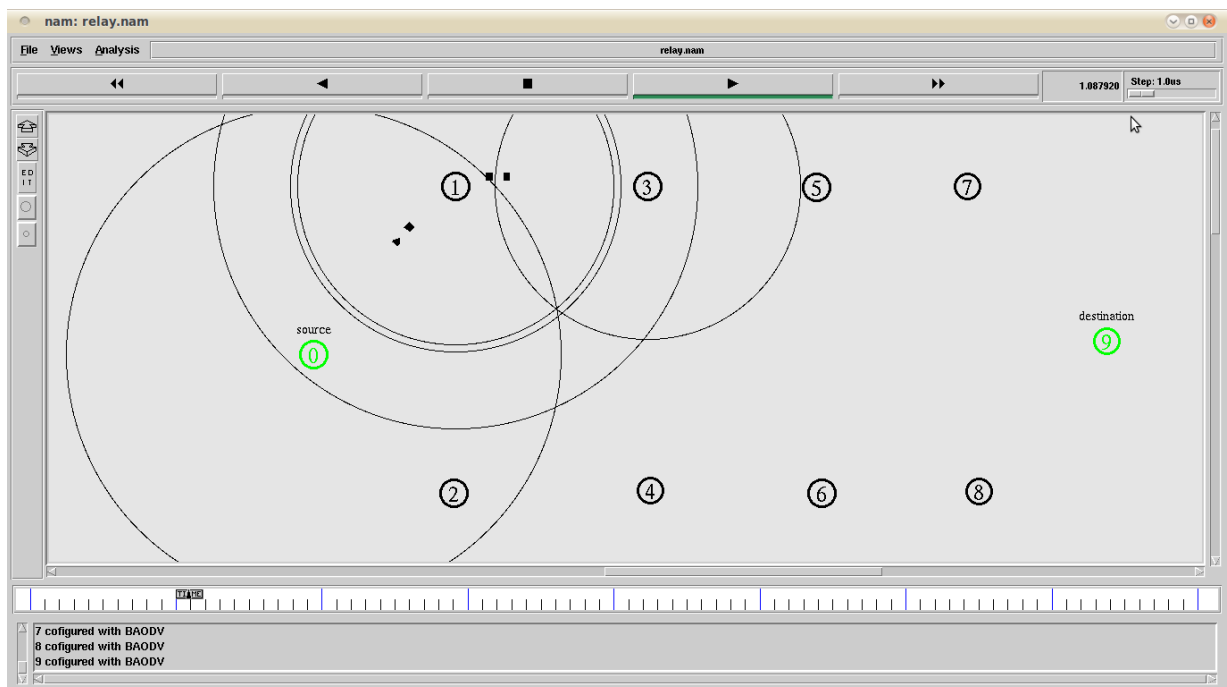


Figure4.5data transmission without any ack sharing

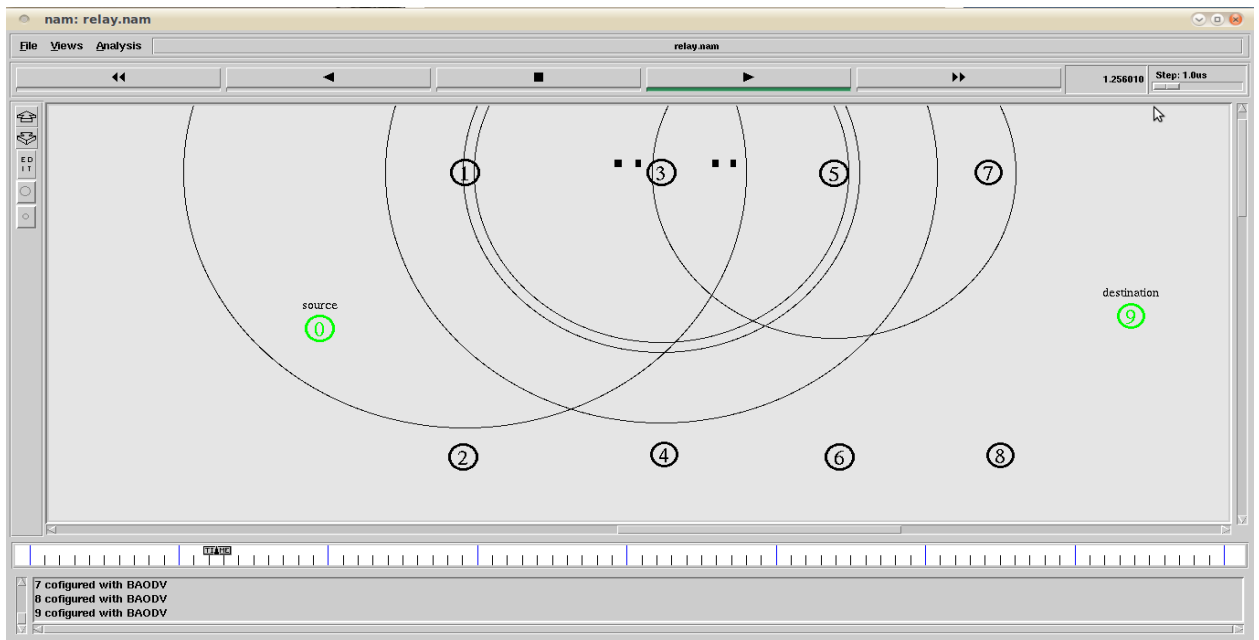


Figure4.6 data transmission from source to destination without any security

In basic AODV model, they have not considered any security so if there is any malicious node found in the path means node can't detect the malicious node information.

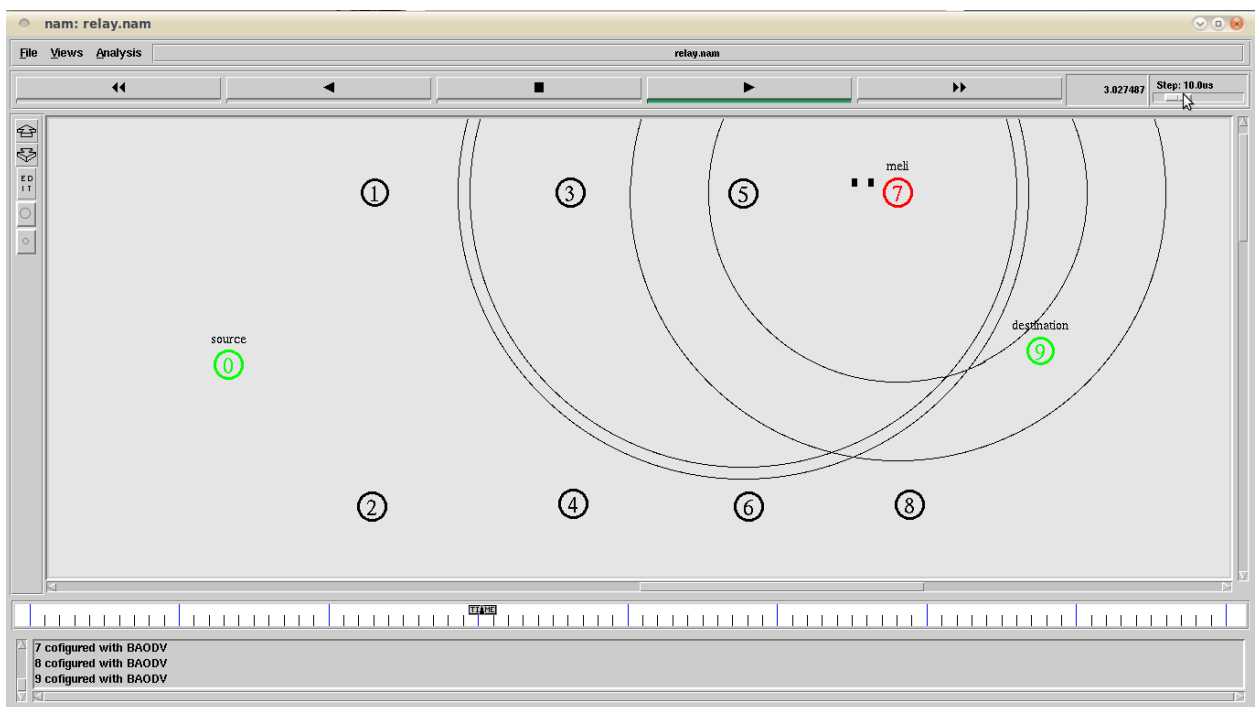


Figure4.7 malicious found in the route (no node can detect malicious in basic AODV)

To detect the malicious node in the network, there are some previous model has been implemented. In that one of the models is TWO-Ack scheme. In this method while transferring the data each node need to generate the ack after receiving the data and that should be forwarded to previous node. and at the same time each node has to forward the ack of next node to previous node. by this method we can find the intermediate malicious node.

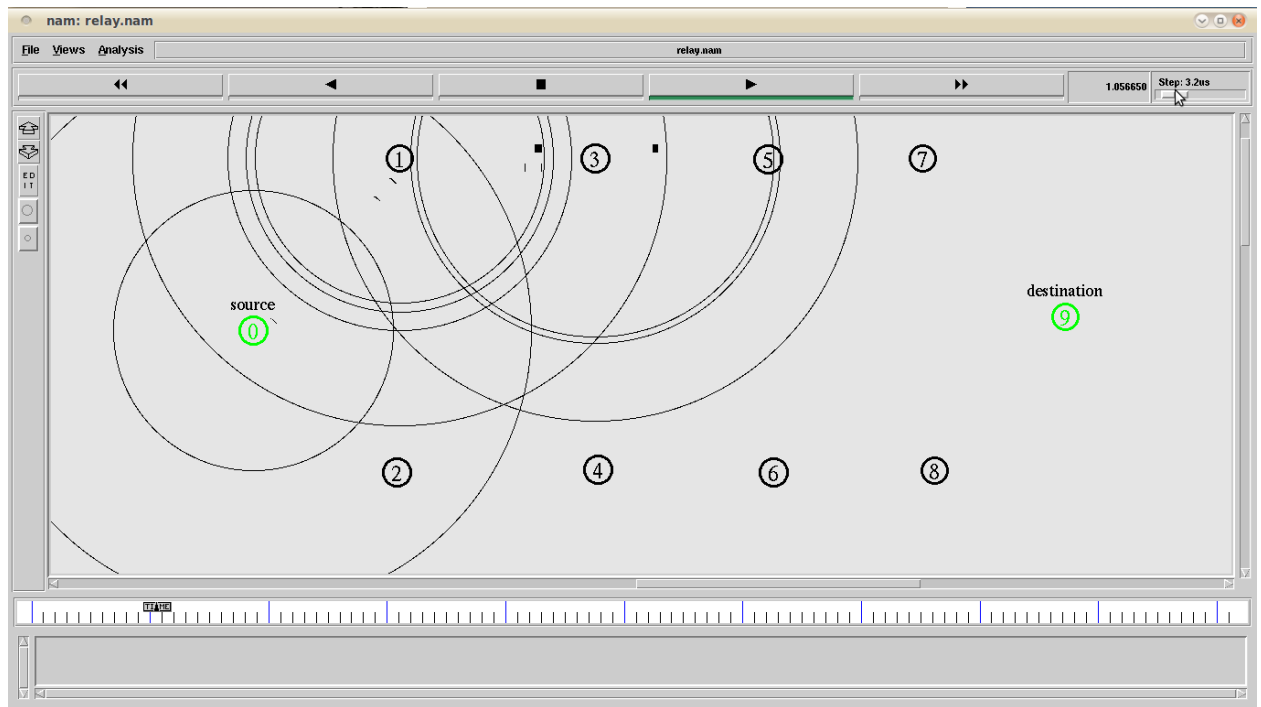


Figure4.8ACK-sharing between each intermediate nodes (TWO - ACK model)

In two-ack scheme is best scheme to detect the malicious node, but this method increases the network load while sharing the number of acknowledgement. By using AACK method we can provide same security as well as reduced overhead. In this scheme before attack end-to-end ack used to reduce the overhead and while attack the source node will switching to two-ack node to find the malicious node.

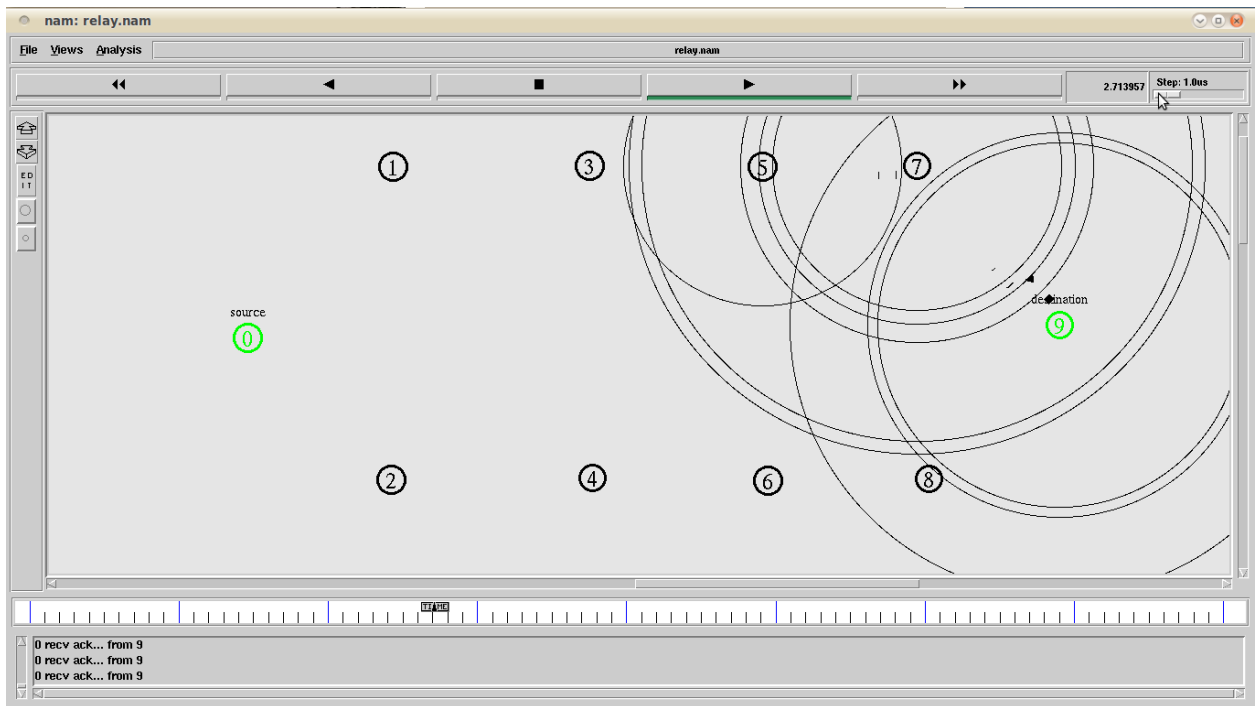


Figure 4.9 AACK model (ACK from destination)

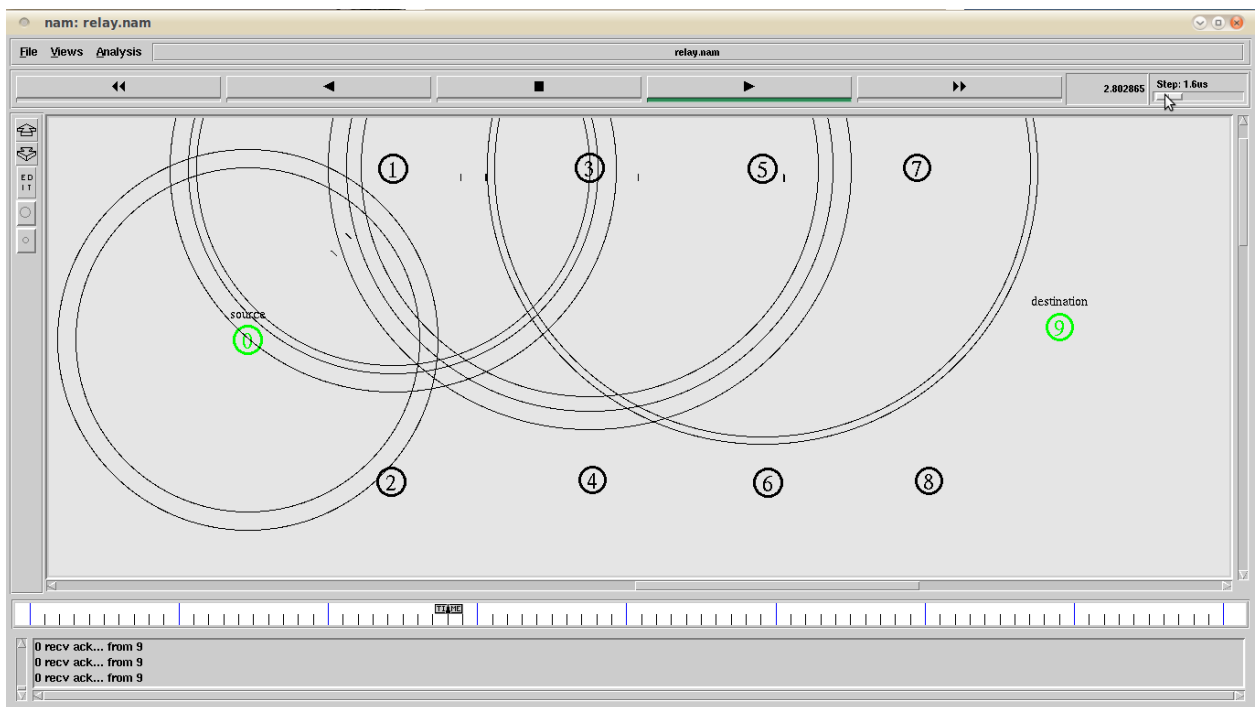


Figure 4.10 ACK sharing to source through intermediate nodes from destination (AACK model)

The malicious node can't generate the ACK so by that source can find attack, and source will switch to secure ACK (S-ACK) mode to find the malicious node.

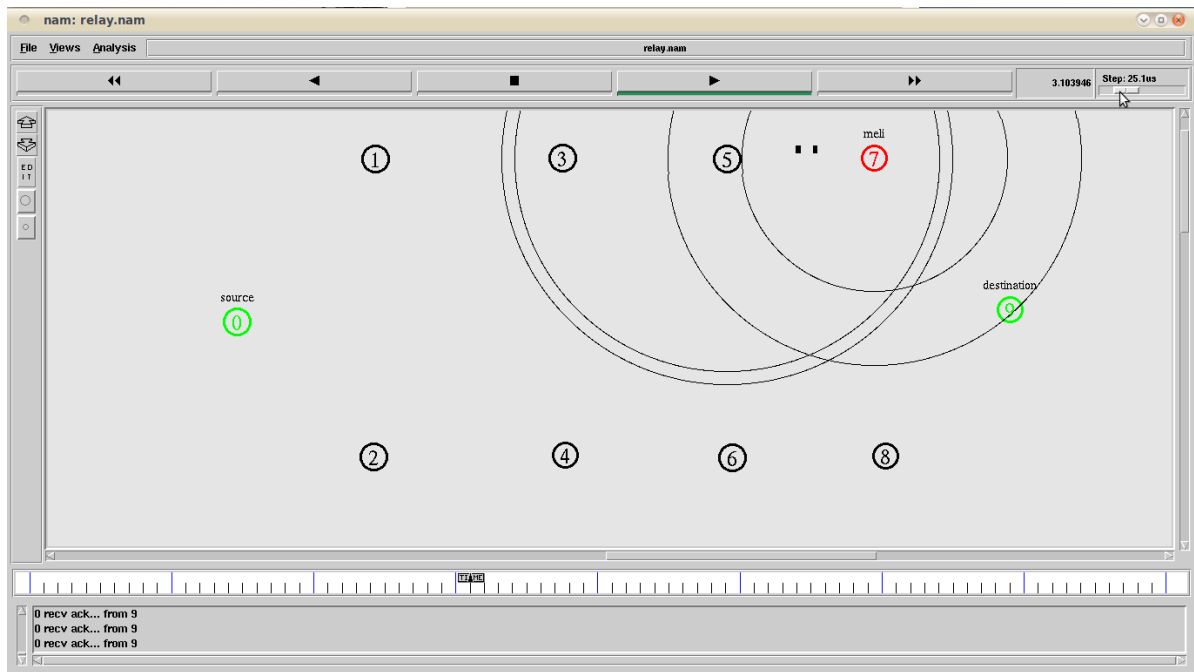


Figure4.11 malicious node collects the data but there is no ACK to source

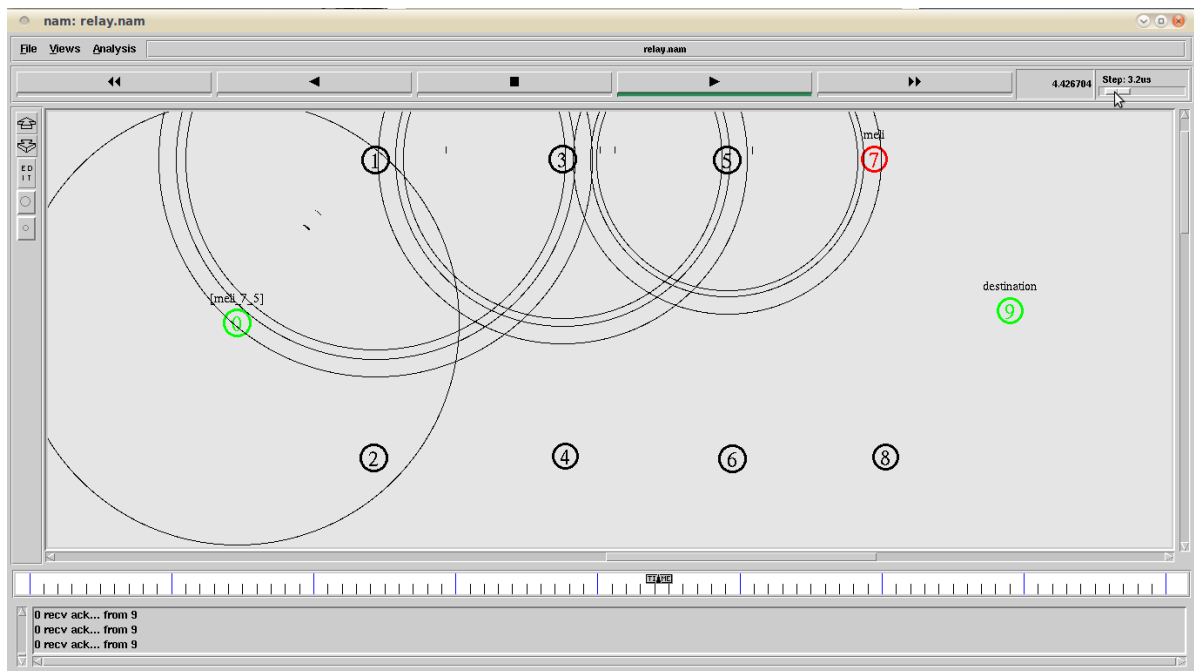


Figure4.12source node initializes the secure data-ACK sharing (S-ACK)

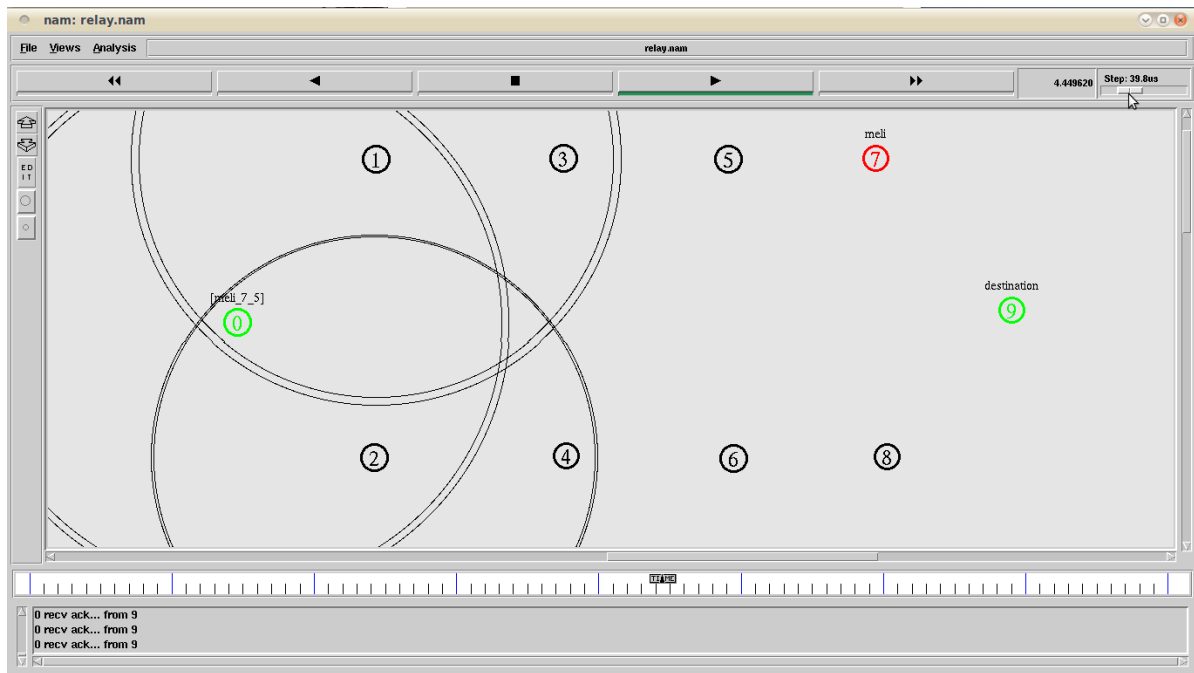


Figure4.13 checks the new route after finding the malicious

AACK method considered the malicious can't send the ACK. But attacker may chances to send the fake ACK. In this case AACK can't detect the attack.

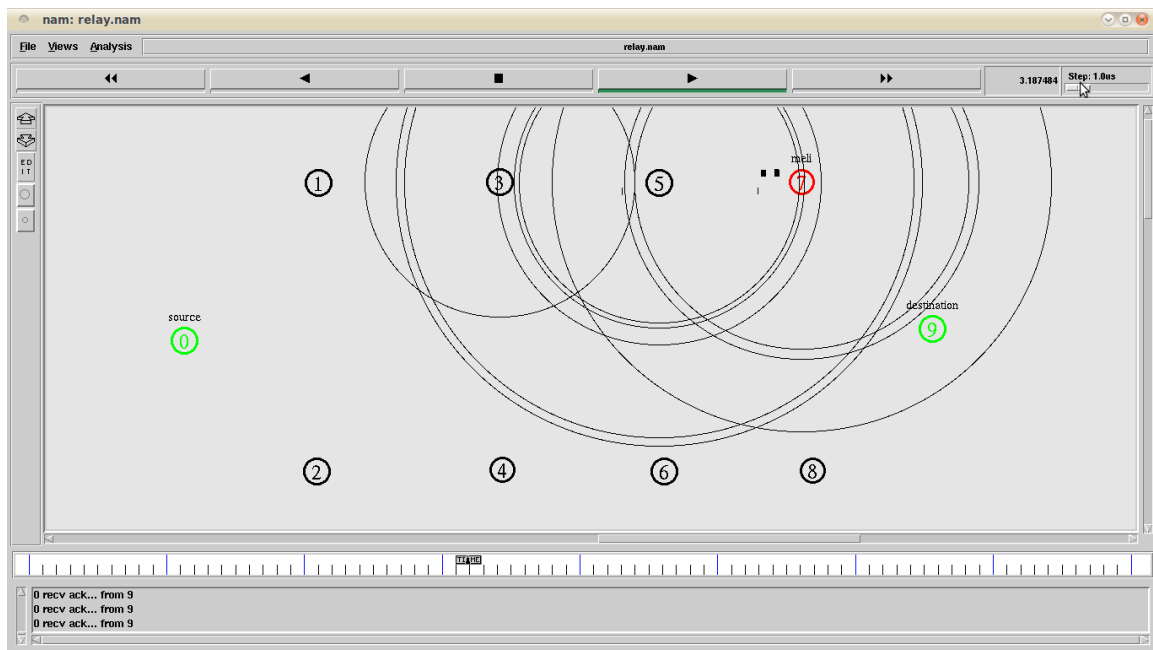


Figure4.14ACK from malicious node

To improve this method EAACK method has been implemented. By this method we can find normal malicious and fake ACK malicious node also.

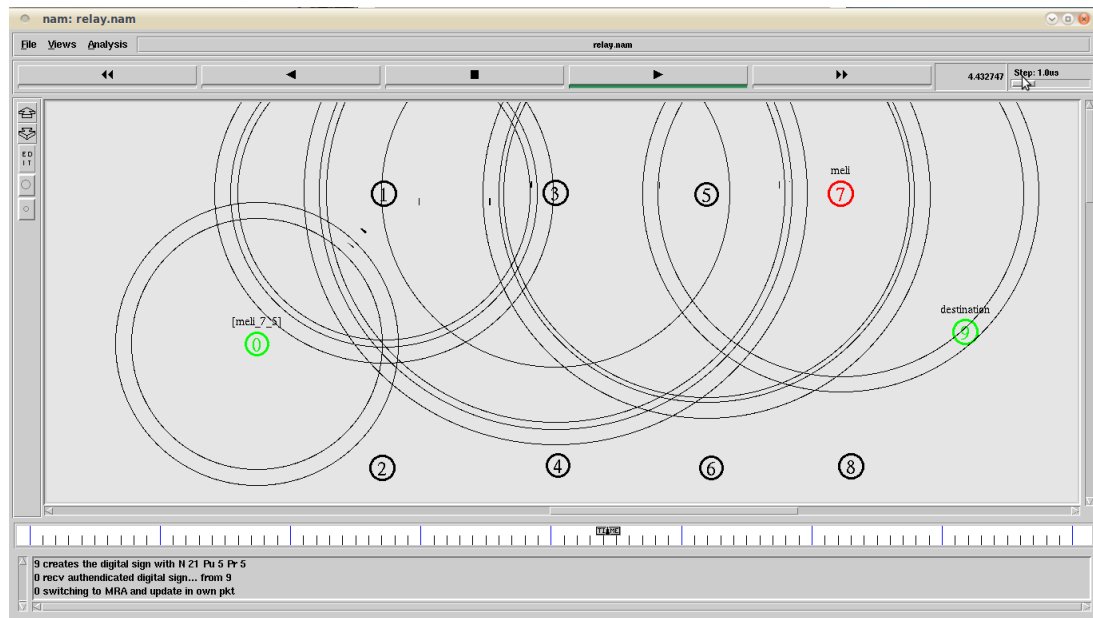


Figure4.15by using digital sign method node can check the ACK whether originally generated by destination or not.

```

5 . 5 A 0 6 ? 5 9 =====> Digital sign...<===== N 21 Pu 5 Pr 5
5 . 6 6 0 > 0 5 9 =====> Digital sign...<===== N 21 Pu 5 Pr 5
5 . @ 4 0 4 : 5 pkt sendt out in forward.... 5
9 =====> Digital sign...<===== N 21 Pu 5 Pr 5
5 . A > 0 > : 5 9 =====> Digital sign...<===== N 21 Pu 5 Pr 5
5 . 0 9 0 6 0 5 9 =====> Digital sign...<===== N 21 Pu 5 Pr 5
5 . 0 0 9 ? : 5 9 =====> Digital sign...<===== N 21 Pu 5 Pr 5
6 . 9 @ 9 : ? 5 9 =====> Digital sign...<===== N 21 Pu 5 Pr 5
6 . : 4 0 4 ? 5 9 =====> Digital sign...<===== N 21 Pu 5 Pr 5
6 . > > 0 4 5 5 9 =====> Digital sign...<===== N 21 Pu 5 Pr 5
6 . ? 9 0 5 0 5 9 =====> Digital sign...<===== N 21 Pu 5 Pr 5
6 . ? A 0 4 0 5 9 =====> Digital sign...<===== N 21 Pu 5 Pr 5
6 . 4 6 0 > @ 5 9 =====> Digital sign...<===== N 21 Pu 5 Pr 5
6 . 5 4 0 @ ? 5 9 =====> Digital sign...<===== N 21 Pu 5 Pr 5
6 . 6 > 0 6 5 5 9 =====> Digital sign...<===== N 21 Pu 5 Pr 5
6 . @ 9 A 6 5 5 9 =====> Digital sign...<===== N 21 Pu 5 Pr 5
6 . @ A 0 9 : 5 9 =====> Digital sign...<===== N 21 Pu 5 Pr 5
6 . A 6 0 6 @ 5 pkt sendt out in forward.... 5
9 =====> Digital sign...<===== N 21 Pu 5 Pr 5
6 . 0 4 0 @ ? 5 9 =====> Digital sign...<===== N 21 Pu 5 Pr 5
@ . 9 ? 9 4 0 5 9 =====> Digital sign...<===== N 21 Pu 5 Pr 5
@ . : 9 0 ? ? 5 9 =====> Digital sign...<===== N 21 Pu 5 Pr 5
@ . : A 0 ? 5 5 9 =====> Digital sign...<===== N 21 Pu 5 Pr 5
@ . > 6 0 0 @ 5 9 =====> Digital sign...<===== N 21 Pu 5 Pr 5

```

Figure4.16digital sign with the key

To provide further more security we have added AES algorithm with digital sign. This is encrypted and decrypted only by destination and source.

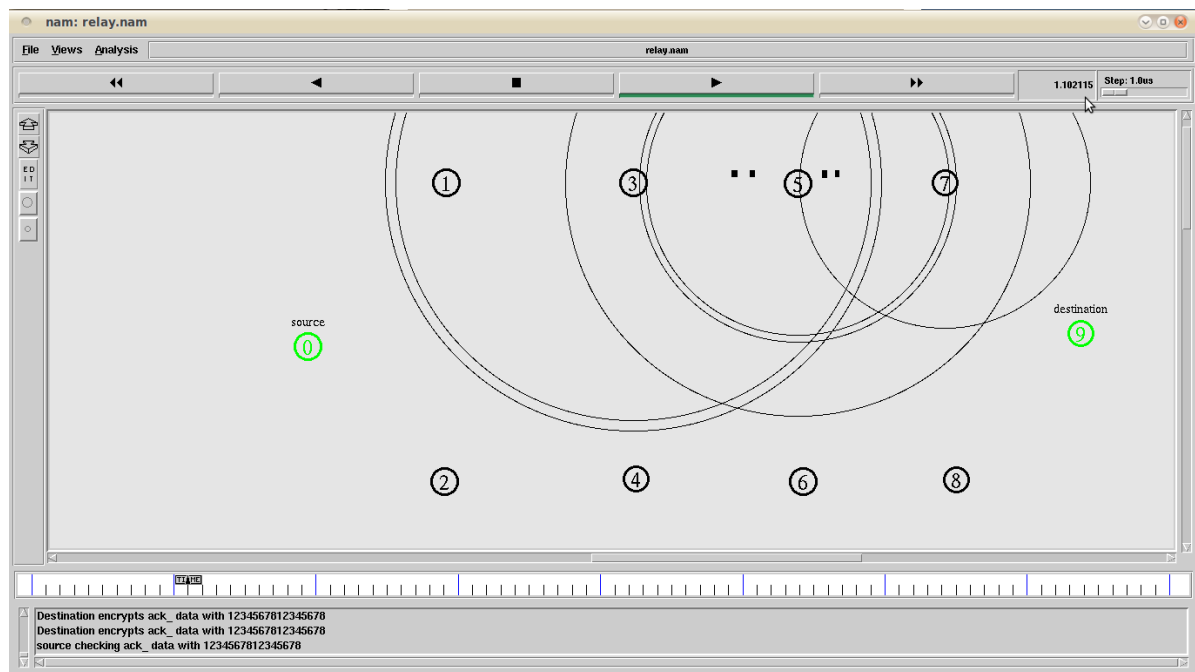


Figure 4.17: secret key used to make more security for acknowledgement

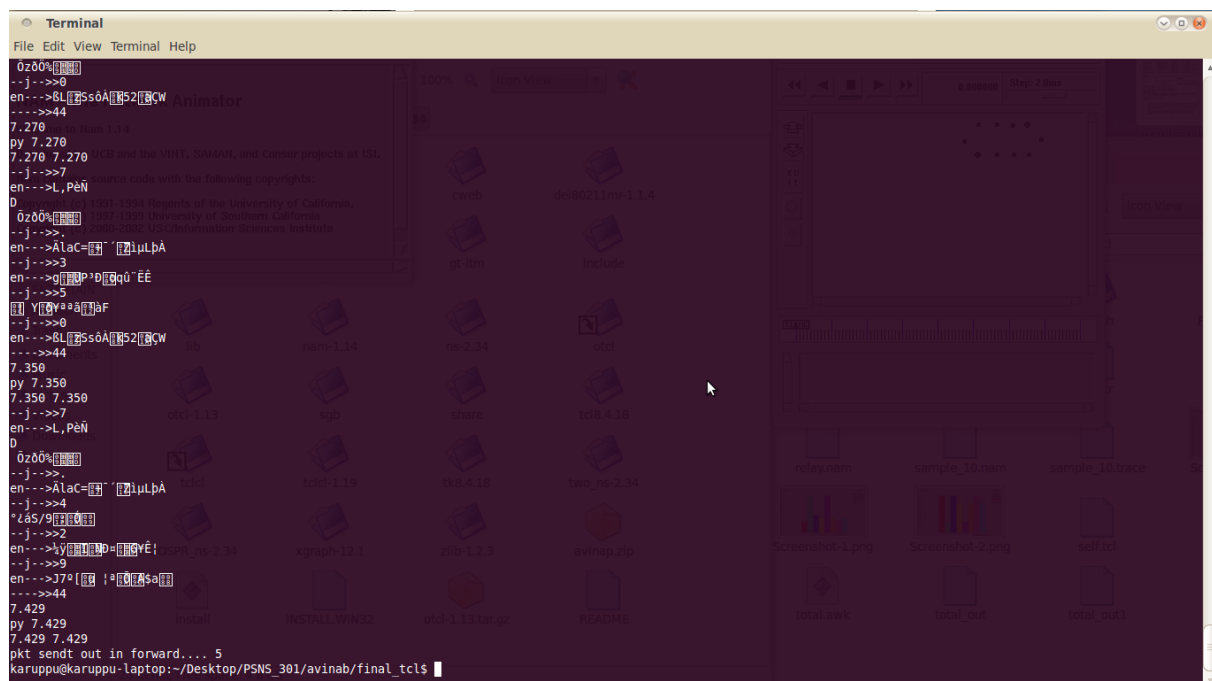


Figure4.18encrypted ACK data by AES algorithm.

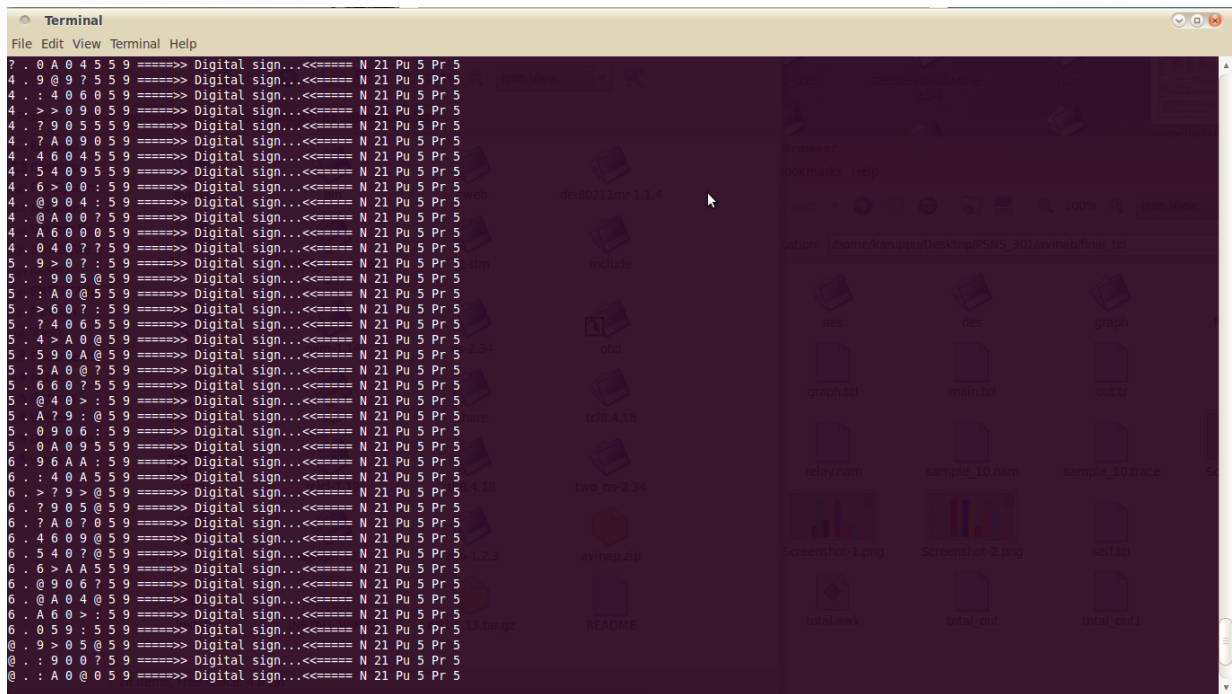


Figure4.19digital sign of secured ACK data by RSA algorithm

In basic AODV model there is no ACK sharing so overhead is very less for BAODV, but in TWO-ACK mode there are the number of ACKshared between each node so network overhead and delay is increased. We have considered normal attack with AACK method, in this case AACK method can find the malicious and it should change the path so Packet delivery is less compare than the BAODV, but at the same time AACK can't find fake ACK so it reduces the further PDF in fake ACK scenario. But EAACK and HCEAACK can provide good performance in fake ACK scenario and normal malicious scenario.

Performance Evaluation

In order to measure and compare the performances of our proposed scheme, we continue to adopt the following three performance metrics to evaluate the performance of IDS for existing and proposed technique which are defined as follows:

1. Packet delivery factor (PDF)

It is the ratio of the total number of received packets at the destination to the total number of sent packets by the source.

$$\text{PDF} = \frac{\sum \text{Received packets at destination}}{\sum \text{Sent packets by sources}} \quad (1)$$

2. Routing Overhead (OH) this is the ratio of routing related packets in bytes (RREQ, RREP, RERR, AACK,) to the total routing and data transmissions (sent or forwarded packets) in bytes. That means the acknowledgments, alarms and switching over head is included.

$$\text{OH} = \frac{\sum \text{Routing transmissions}}{\sum \text{Data transmissions} + \sum \text{Routing transmissions}} \quad (2)$$

3. Average end-to-end delay (D) The average end-to-end delay for all successfully received packets at the destination. It is calculated for each data packet subtracting the sending time of the packet from the received time at final destination.

$$D = \frac{\sum_1^N (T_{\text{Received}} - T_{\text{Sent}})}{N} \quad (3)$$

Where N is number of successfully received packets.

To provide readers with a better insight on our simulation results, detailed simulation data are presented in Table 4.1.

Table 4.1 Result table

Model	PDF (%)	OH	Delay(ms)
BAODV	100	16	108
BAODV_meli	28.40	16	367
Two_ack	99.122	854	168
AACK_meli	79.54	400	144
AACK_Fmeli	28.40	393	367
EAACK_meli	79.54	400	144
EAACK_Fmeli	98.86	488	116
HEAACK	79	400	144

In BAODV model there is no ACK sharing so overhead is very less, but when malicious node appear (BAODV_meli) model the Packet delivery factor PDF drop and the Delay time increase in network, in Two_ACK mode there are numbers of ACK shared between each node so network overhead and delay is increased. We have considered normal attack with AACK method, in this case AACK method can find the malicious and it should change the path so packet delivery is less compare than the BAODV, but at the same time AACK can't find fake ACK so it reduces the further packet delivery factor (pdf) in fake ACK scenario (AACK_Fmeli).EAACK and HCEAACK can provide good performance in fake ACK scenario and normal malicious scenario, and our proposed model HCEAACK give best packet delivery factor (pdf) on same value of network overhead.

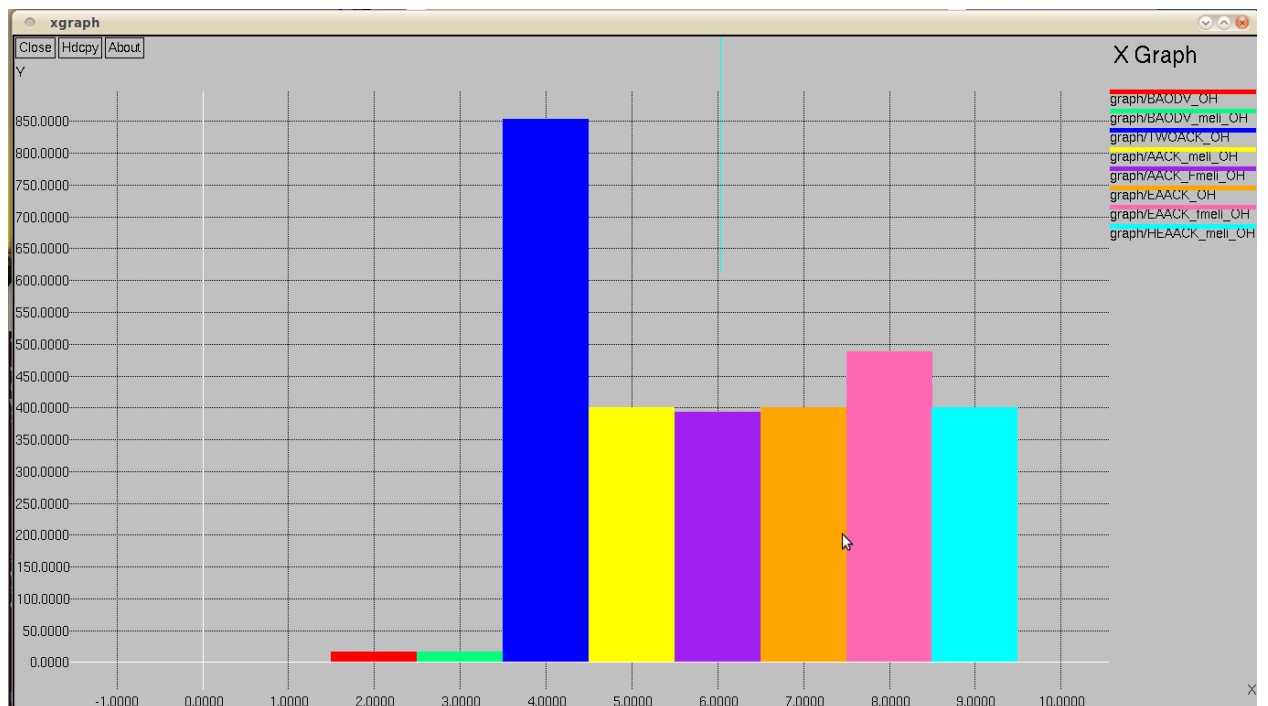


Figure4.20comparison of overhead (OH)

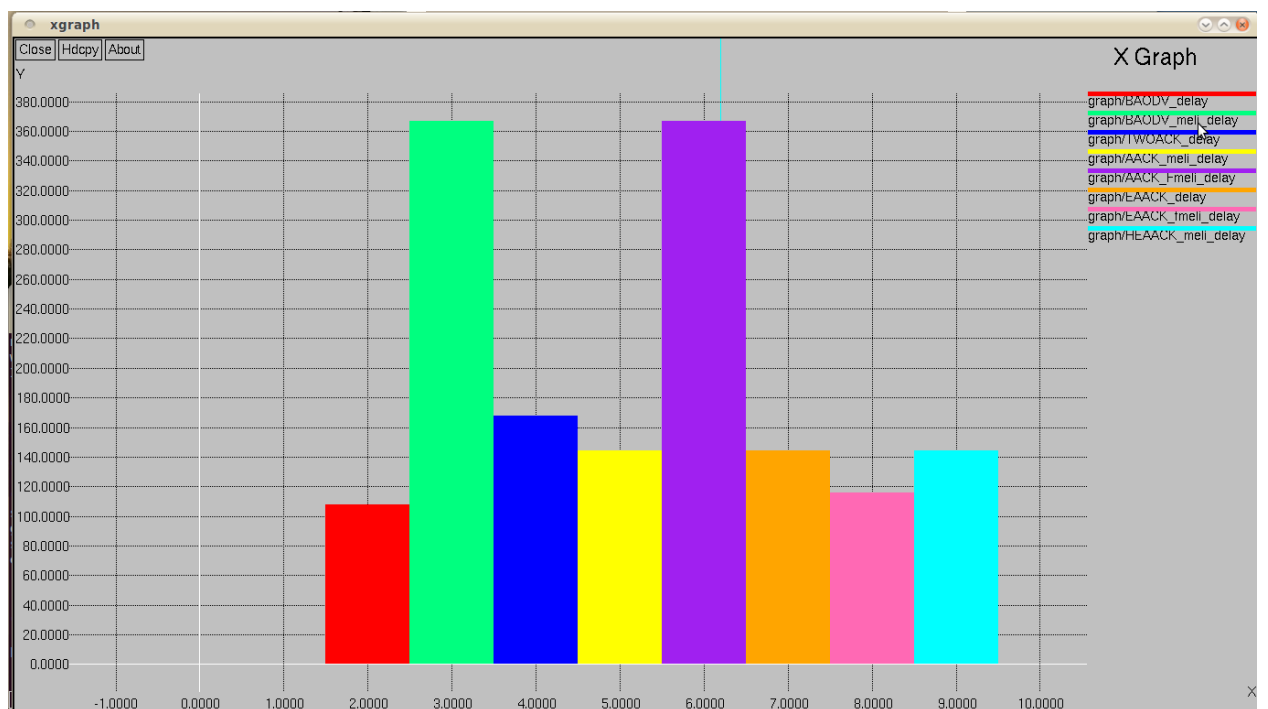


Figure4.21comparison of delay (D)

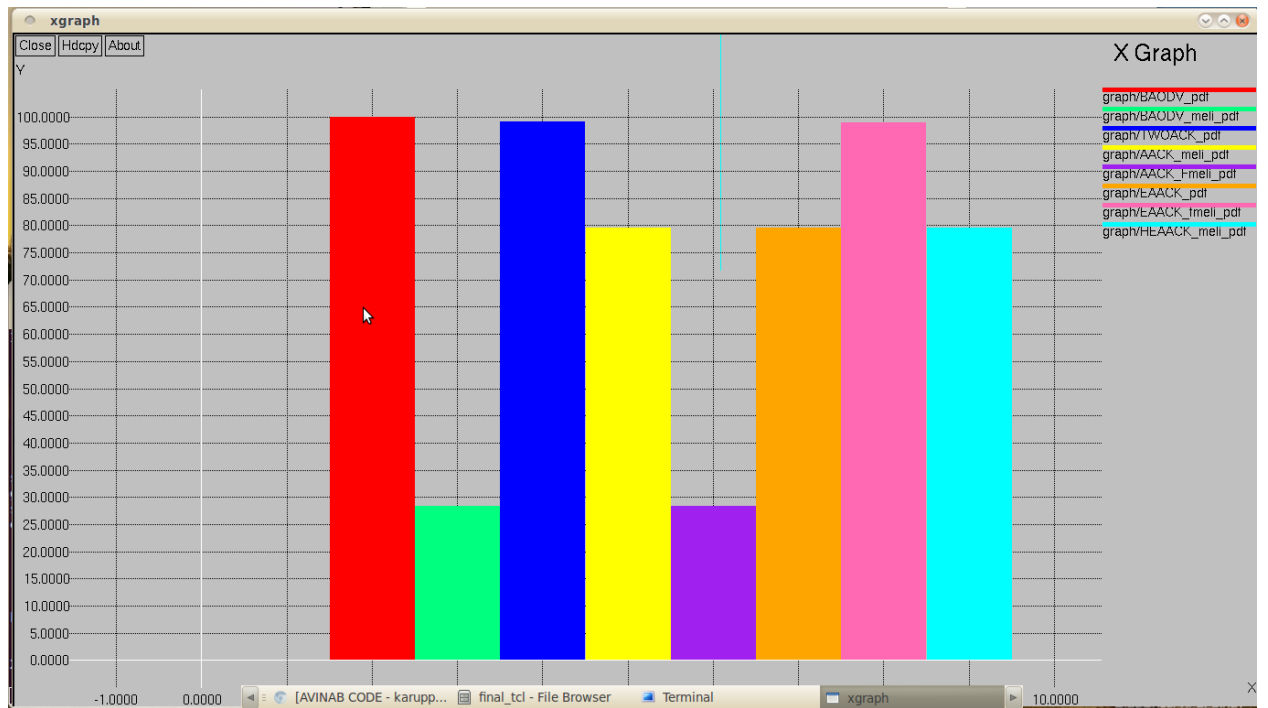


Figure4.22comparison of packet delivery factor (PDF)

5 CONCLUSION AND FUTURE WORK

Summary

Among all the up to date wireless networks, Mobile Ad hoc Network (MANET) is one amongst the foremost necessary and distinctive applications. On the contrary to ancient spec, MANETs doesn't need a set network infrastructure; each single node works as each a transmitter and a receiver and they trust their neighbors to relay messages. Unfortunately, the open medium and remote distribution of MANETs create it at risk of numerous kinds of attacks. So, it is crucial to develop efficient intrusion-detection mechanisms to protect MANET from attacks. In this paper, we define solid privacy requirements regarding malicious attackers in MANETs. Then we propose and implement a new intrusion-detection system named Hybrid Cryptography Enhanced Adaptive Acknowledgment (HCEAACK) specially designed for MANETs. To provide further more security we have added Advanced Encryption Standard (AES) algorithm for encrypted ACK data and get secure ACK, then digital sign of secured ACK data by RSA algorithm. This is encrypted and decrypted only by destination and source. Secrete key used to make more security for acknowledgement and encrypted ACK data by AES algorithm. Compared to contemporary approaches, HCEAACK demonstrates higher malicious-behavior-detection rates in certain circumstances while does not greatly affect the network performances.

5.2 Conclusion

We propose and implement a new intrusion-detection system named Hybrid Cryptography Enhanced Adaptive Acknowledgment (HCEAACK) specially designed for MANETs. We implemented both AES and RSA schemes in our simulation. To provide further more security we have added Advanced Encryption Standard (AES) algorithm for encrypted ACK data and get secure ACK, then digital sign of secured ACK data by RSA algorithm. This is encrypted and decrypted only by destination and source. Secrete key used to make more security for acknowledgement and encrypted ACK data by AES algorithm. HCEAACK can provide good performance in fake ACK scenario and normal malicious scenario, and give best packet delivery factor (pdf) on same value of network overhead. .To increases the merits of our research work; we plan to investigate the following issues in our future research:

- 1) Examine the possibilities of adopting a key exchange mechanism to eliminate the requirement of pre distributed keys;
- 2) Testing the performance of HCEAACK in real network environment instead of software simulation.

REFERENCES

- K. Al Agha, M.-H. Bertin, T. Dang, A. Guitton, P. Minet, T. Val, and J.-B. Viollet, —Which wireless technology for industrial wireless sensor networks? The development of OCARI technology,‖ *IEEE Trans. Ind. Electron.*, vol. 56, no.10, pp. 4266–4278, Oct. 2009.
- R. Akbani, T. Korkmaz, and G. V. S. Raju, •\Mobile Ad hoc Network Security,. in *Lecture Notes in Electrical Engineering*, vol. 127. New York: Springer-Verlag, 2012, pp. 659.666.
- G. Jayakumar and G. Gopinath, “Ad hoc mobile wireless networks routing protocol—A review,” *J. Computer Sci.*, vol. 3, no. 8, pp. 574–582, 2007.
- B. Sun, “Intrusion detection in mobile ad hoc networks,” Ph.D. dissertation,Texas A&M Univ., College Station, TX, 2004.
- A. Tabesh and L. G. Frechette, “A low-power stand-alone adaptive circuit for harvesting energy from a piezoelectric micro power generator,” *IEEE Trans. Ind. Electron.*, vol. 57, no. 3, pp. 840–849, Mar. 2010.
- R. H. Akbani, S. Patel, and D. C. Jinwala, —DoS attacks in mobile ad hoc networks: A survey,‖ in *Proc. 2nd Int.Meeting ACCT*, Rohtak, Haryana, India, 2012, pp. 535–541.
- T. Anantvalee and J. Wu, —A Survey on Intrusion Detection in Mobile Ad Hoc Networks,‖ in *Wireless/Mobile Security* .New York: Springer- Verlag, 2008.
- L. Buttyan and J. P. Hubaux, *Security and Cooperation in Wireless Networks*. Cambridge, U.K.: Cambridge Univ.Press, Aug. 2007.
- D. Dondi, A. Bertacchini, D. Brunelli, L. Larcher, and L. Benini, —Modeling and optimization of a solar energy harvester system for self-powered wireless sensor networks,‖ *IEEE Trans. Ind. Electron.*, vol. 55, no. 7, pp. 2759–2766,Jul. 2008.
- D. Dondi, A. Bertacchini, D. Brunelli, L. Larcher, and L. Benini, “Modeling and optimization of a solar energy harvester system for self-powered wireless sensor networks,” *IEEE Trans. Ind. Electron.*, vol. 55, no. 7,pp. 2759–2766, Jul. 2008.

- Y. Hu, A. Perrig, and D. Johnson, "ARIADNE: A secure on-demand routing protocol for ad hoc networks," in Proc. 8th ACM Int. Conf. MobiCom, Atlanta, GA, 2002, pp. 12–23.
- J.-S. Lee, "A Petri net design of command filters for semiautonomous mobile sensor networks," IEEE Trans. Ind. Electron., vol. 55, no. 4, pp. 1835–1841, Apr. 2008.
- K. Liu, J. Deng, P. K. Varshney, and K. Balakrishnan, "An acknowledgment-based approach for the detection of routing misbehavior in MANETs," IEEE Trans. Mobile Comput., vol. 6, no. 5, pp. 536–550, May 2007.
- A. Patwardhan, J. Parker, A. Joshi, M. Iorga, and T. Karygiannis, "Secure routing and intrusion detection in ad hoc networks," in Proc. 3rd Int. Conf. Pervasive Comput. Commun., 2005, pp. 191–199.
- J. G. Rocha, L. M. Goncalves, P. F. Rocha, M. P. Silva, and S. Lanceros-Mendez, "Energy harvesting from piezoelectric materials fully integrated in footwear," IEEE Trans. Ind. Electron., vol. 57, no. 3, pp. 813–819, Mar. 2010.
- A. Singh, M. Maheshwari, and N. Kumar, "Security and trust management in MANET," in Communications in Computer and Information Science, vol. 147. New York: Springer-Verlag, 2011, pt. 3, pp. 384–387.
- A. Tabesh and L. G. Frechette, "A low-power stand-alone adaptive circuit for harvesting energy from a piezoelectric micro power generator," IEEE Trans. Ind. Electron., vol. 57, no. 3, pp. 840–849, Mar. 2010.
- L. Zhou and Z. Haas, "Securing ad-hoc networks," IEEE Netw., vol. 13, no. 6, pp. 24–30, Nov./Dec. 1999.
- E. Shakhshuki and Nan Kang "EAACK - A Secure Intrusion-Detection System for MANETs," IEEE TRANSACTIONS ON INDUSTRIAL ELEC., VOL. 60, NO. 3, MARCH 2013.
- S. Marti, T. J. Giuli, K. Lai, and M. Baker, "Mitigating routing misbehavior in mobile ad hoc networks," in Proc. 6th Annu. Int. Conf. Mobile Comput. Netw., Boston, MA, 2000, pp. 255–265.

T. Sheltami, A. Al-Roubaiey, E. Shakshuki, and A. Mahmoud, "Video transmission enhancement in presence of misbehaving nodes in MANETs," *Int. J. Multimedia Syst.*, vol. 15, no. 5, pp. 273–282, Oct. 2009.

"A study of different types of attacks on multicast in mobile ad hoc networks" Hoang Lan Nguyen, Uyen Trang Nguyen, Elsevier *AdHoc Networks*(2008) 32-46.

References

REFERENCE

1. *www.wikipedia.org/*

2. R. Akbani, T. Korkmaz, and G. V. S. Raju, "Mobile Ad hoc Network Security," in *Lecture Notes in Electrical Engineering*, vol. 127. New York: Springer-Verlag, 2012, pp. 659–666.

3. R. H. Akbani, S. Patel, and D. C. Jinwala, "DoS attacks in mobile ad hoc networks: A survey," in *Proc. 2nd Int. Meeting ACCT, Rohtak, Haryana, India, 2012*, pp. 535–541.

4. T. Anantvalee and J. Wu, "A Survey on Intrusion Detection in Mobile Ad Hoc Networks," in *Wireless/Mobile Security*. New York: Springer-Verlag, 2008.
5. L. Buttyan and J. P. Hubaux, *Security and Cooperation in Wireless Networks*. Cambridge, U.K.: Cambridge Univ. Press, Aug. 2007.
6. D. Dondi, A. Bertacchini, D. Brunelli, L. Larcher, and L. Benini, "Modeling and optimization of a solar energy harvester system for self-powered wireless sensor networks," *IEEE Trans. Ind. Electron.*, vol. 55, no. 7, pp. 2759–2766, Jul. 2008.
7. V. C. Gungor and G. P. Hancke, "Industrial wireless sensor networks: Challenges, design principles, and technical approach," *IEEE Trans. Ind. Electron.*, vol. 56, no. 10, pp. 4258–4265, Oct. 2009.
8. Y. Hu, D. Johnson, and A. Perrig, "SEAD: Secure efficient distance vector routing for mobile wireless ad hoc networks," in *Proc. 4th IEEE Workshop Mobile Comput. Syst. Appl.*, 2002, pp. 3–13.
9. Y. Hu, A. Perrig, and D. Johnson, "ARIADNE: A secure on-demand routing protocol for ad hoc networks," in *Proc. 8th ACM Int. Conf. MobiCom*, Atlanta, GA, 2002, pp. 12–23.
10. G. Jayakumar and G. Gopinath, "Ad hoc mobile wireless networks routing protocol—A review," *J. Comput.Sci.*, vol. 3, no. 8, pp. 574–582, 2007.


```

puts "Enter your option BA0DV(1) or B-Meli(2) or TWO_ack(A) or meli_ACK(3) or FA-meli(4) or EAACK-meli(5)
or EAACK-FA-meli(6) or hack_HEAACK(7) or HEAACK(8) or comp(9)"
set opt_ [gets stdin]
if {$opt_ == 9} {
    source graph.tcl
    exit
}
if {$opt_ == "A"} {
    source two_ack.tcl
    exit
}
# =====
# Define options
# =====
set val(chan)          Channel/WirelessChannel    ;# Channel Type
set val(prop)          Propagation/TwoRayGround ;#Nakagami
;#TwoRayGround    ;# radio-propagation model
set val(netif)         Phy/WirelessPhy
set val(mac)           Mac/802_11
set val(ifq)           Queue/DropTail/PriQueue ;# ; # interface queue type    Queue/DropTail/PriQueue
for A0DV & DSDV    CMUPriQueue for DSR
set val(ll)            LL                          ;# link layer type
set val(ant)           Antenna/OmniAntenna        ;# antenna model
set val(ifqlen)        50                          ;# max packet in ifq
set val(nn)            10                          ;# number of mobilenodes
set val(rp)            A0DV;# routing protocol
set val(x)             400
set val(y)             400

#set val(nam)          out.nam
#set val(traffic)      ftp                        ;# cbr/poisson/ftp
Agent/A0DV set pr_key 2
Agent/A0DV set N_key 2
Agent/A0DV set text_key 3
Agent/A0DV set pu_key 2

#=====

Mac/802_15_4 wpanCmd verbose on
Mac/802_15_4 wpanNam namStatus on

#=====

set ns_                [new Simulator]
set ns_ $ns_
set tracefd            [open out.tr w]
set namtrace           [open relay.nam w]
set Pan                [new Agent/NAM_]
set ran [new RNG]
$Pan namattach $namtrace
$ns_ trace-all $tracefd
$ns_ namtrace-all-wireless $namtrace $val(x) $val(y)

#=====

set topo               [new Topography]
$topo load_flatgrid $val(x) $val(y)
set god_ [create-god $val(nn)]

    set dist(2) 5.0074e-10 ;#231
    set dist(2) 1.0074e-8 ;#410
    Mac/802_11 set RTSThreshold_ 101
    Phy/WirelessPhy set CStresh_ $dist(2)
    Phy/WirelessPhy set RXThresh_ $dist(2)
$ns_ node-config -adhocRouting $val(rp) \

```

```

        -llType $val(ll) \
        -macType $val(mac) \
        -ifqType $val(ifq) \
        -ifqLen $val(ifqlen) \
        -antType $val(ant) \
        -propType $val(prop) \
        -phyType $val(netif) \
        -channelType $val(chan)\
        -topoInstance $topo \
        -agentTrace ON \
        -routerTrace ON \
        -macTrace ON \
        -movementTrace ON \

for {set i 1} {$i <= $val(nn)} {incr i} {
    set node_($i) [$ns_ node]
$node_($i) color black
    [$node_($i) set ragent_] Pan_nam $Pan
    [$node_($i) set mac_(0)] network [$node_($i) set ragent_]
    if {$sopt_ == 6} {
        [$node_($i) set ragent_] EAACK_hk
    }
    if {$sopt_ == 1 || $sopt_ == 2} {
        [$node_($i) set ragent_] option BAODV
    } elseif {$sopt_ == 4 || $sopt_ == 3} {
        [$node_($i) set ragent_] option ACK
    } elseif {$sopt_ == 6 || $sopt_ == 5} {
        [$node_($i) set ragent_] option EAACK
    } elseif {$sopt_ == 7 || $sopt_ == 8} {
        [$node_($i) set ragent_] option HEAACK
    }
}

[$node_(1) set ragent_] key_ 33 7 3
[$node_(10) set ragent_] key_ 21 5 5
if {$sopt_ == 8} {
    set sk1 12345678
    set sk2 $sk1
    set key_1 "1234567812345678"
    set key_2 "1234567812345678"
} else {
    set sk1 12345678
    set sk2 12345697
}

if {$sopt_ == 3 || $sopt_ == 2 || $sopt_ == 5 || $sopt_ == 7} {
    $ns_ at 3 "$node_(8) color red"
    $ns_ at 3 "$node_(8) label meli"
    $ns_ at 3 "[$node_(8) set ragent_] meli1"
} elseif {$sopt_ == 4 || $sopt_ == 6} {
    $ns_ at 3 "[$node_(8) set ragent_] meli2"
    $ns_ at 3 "$node_(8) color red"
    $ns_ at 3 "$node_(8) label meli"
}

}

$ns_ at 0 "$node_(1) color green"
$ns_ at 0 "$node_(1) label source"

$ns_ at 0 "$node_(10) color green"
$ns_ at 0 "$node_(10) label destination"
# Create wireless nodes.
#set node_(10) [$ns_ node]
## node_(10) at 753.401184,573.277100
$node_(10) set X_ 753.401184
$node_(10) set Y_ 573.277100
$node_(10) set Z_ 0.0

```

```
$node_(10) color "black"
$ns initial_node_pos $node_(10) 10.000000
#set node_(9) [$ns node]
## node_(9) at 704.360718,522.603760
$node_(9) set X_ 704.360718
$node_(9) set Y_ 515.603760
$node_(9) set Z_ 0.0
$node_(9) color "black"
$ns initial_node_pos $node_(9) 10.000000
#set node_(8) [$ns node]
## node_(8) at 699.893494,618.966125
$node_(8) set X_ 699.893494
$node_(8) set Y_ 632.966125
$node_(8) set Z_ 0.0
$node_(8) color "black"
$ns initial_node_pos $node_(8) 10.000000
#set node_(7) [$ns node]
## node_(7) at 643.989136,523.019470
$node_(7) set X_ 643.989136
$node_(7) set Y_ 515.019470
$node_(7) set Z_ 0.0
$node_(7) color "black"
$ns initial_node_pos $node_(7) 10.000000
#set node_(6) [$ns node]
## node_(6) at 641.952454,619.507751
$node_(6) set X_ 641.952454
$node_(6) set Y_ 632.507751
$node_(6) set Z_ 0.0
$node_(6) color "black"
$ns initial_node_pos $node_(6) 10.000000
#set node_(5) [$ns node]
## node_(5) at 578.016907,521.924988
$node_(5) set X_ 578.016907
$node_(5) set Y_ 515.924988
$node_(5) set Z_ 0.0
$node_(5) color "black"
$ns initial_node_pos $node_(5) 10.000000
#set node_(4) [$ns node]
## node_(4) at 577.090332,618.987183
$node_(4) set X_ 577.090332
$node_(4) set Y_ 632.987183
$node_(4) set Z_ 0.0
$node_(4) color "black"
$ns initial_node_pos $node_(4) 10.000000
#set node_(3) [$ns node]
## node_(3) at 502.578979,525.055359
$node_(3) set X_ 502.578979
$node_(3) set Y_ 515.055359
$node_(3) set Z_ 0.0
$node_(3) color "black"
$ns initial_node_pos $node_(3) 10.000000
#set node_(2) [$ns node]
## node_(2) at 503.293365,617.797852
$node_(2) set X_ 503.293365
$node_(2) set Y_ 632.797852
$node_(2) set Z_ 0.0
$node_(2) color "black"
$ns initial_node_pos $node_(2) 10.000000
#set node_(1) [$ns node]
## node_(1) at 448.619537,568.331543
$node_(1) set X_ 448.619537
$node_(1) set Y_ 568.331543
$node_(1) set Z_ 0.0
$node_(1) color "black"
$ns initial_node_pos $node_(1) 10.000000

# Create links between nodes.
```

Add Link Loss Models

```

set hui 0
source aes/aes.tcl

#avinab encryption by aes
proc Encrypt {j} {
    global hui ran sk1 ns_ Pan
    set key_1 "1234567812345678"
    $ns_ at [$ns_ now] "$Pan anot_ \"Destination encrypts ack_ data with $key_1\""
    puts --j-->$j
    set tr [list]
    set plaintext $j

    # set Key [DES::Init ecb $sk1 true]
    #set ciphertext [DES::Encrypt $Key $plaintext]
    set encrypted [aes::aes -mode cbc -dir encrypt -key $key_1 $j]
    puts en-->$encrypted
    foreach ki [split $encrypted ""] {
        scan $ki %c m
        set RN [expr int([$ran uniform 48 57])]
        lappend tr *$RN@^[expr $m-$RN]&
        lappend ts $m
    }
    incr hui
    return $tr
}

proc decrypt {str} {
    global sk2 ns_ Pan
    # puts xxxxxxxxxxxxxxxx$str
    set key_1 "1234567812345678"
    puts ---->[lindex $str 1]
    $ns_ at [$ns_ now] "$Pan anot_ \"source checking ack_ data with $key_1\""
    for {set oi 0} {$oi<[llength $str]} {incr oi} {
        set i [lindex $str $oi]
        if {$i != "-"} {
            lappend char [format %c $i]
        } else {
            lappend dd [aes::aes -mode cbc -dir decrypt -key $key_1 [join $char ""]]
            set char [list]
        }
    }
    set ooo [join $dd ""]
    foreach kj [split $ooo ""] {
        if {[string is integer $kj] || $kj == "."} {
            lappend py $kj
        }
    }
    puts [join $py ""]
    puts "py [format %.3f [join $py "]]"
    return [join $py ""]
}

proc check {i a b} {
    global ns_ cbr node_ Pan
    puts "$a $b"
    if {$a != $b} {
        set ko 0
        set nw [$ns_ now]
        while {$ko<10} {
            $ns_ at $nw "$node_($i) add-mark m1 red hexagon"
            set nw [expr $nw+0.003]
            $ns_ at $nw "$node_($i) delete-mark m1"
            set nw [expr $nw+0.003]
        }
    }
}

```



```

        incr ko
    }
    $ns_ at [$ns_ now] "$Pan anot_ \"un trusted communication so communication stoped\""
    $ns_ at [$ns_ now] "$cbr(0) stop"
}

}

proc List {n m {o "\n"}} {
    global li convrt_
    lappend li $o
}

proc karuppu {} {
    global li Key
    set kj [join $li ""]

# set pl [DES::Decrypt $Key $kj]
}

set udp(0) [new Agent/UDP]
$ns_ attach-agent $node_(1) $udp(0)
set cbr(0) [new Application/Traffic/CBR]
$cbr(0) attach-agent $udp(0)
set null(0) [new Agent/Null]
$cbr(0) set packetSize_ 500
$cbr(0) set rate_ 0.05Mb
$ns_ attach-agent $node_(10) $null(0)
$ns_ connect $udp(0) $null(0)
$ns_ at 1 "$cbr(0) start"

proc stop {} {
    global ns_ tracefd val env
    $ns_ flush-trace
    close $tracefd
    exec nam relay.nam &
    exec awk -f total.awk out.tr >total_out
    exit 0
}

$ns_ at 8 "stop"
$ns_ run

```