بسم الله الرحمن الرحيم

Sudan University of Science and Technology

College of Post Graduate Studies

Thesis is a Partial Fulfillment for the Degree of M.Sc. in Computer Engineering

Implementation of RSA and ECC Security Protocol for Mobile AD HOC Networks

تنفيذالبرتوكول الآمن لخوارزميتي تشفير المنحي البيضاوي والمفتاح العام للشبكة النقالة

Prepared by:

Mohammed Adam Aldod Ibrahim

Under the Supervision of:

Associate Professor Dr. Ibrahim Khider Eltahir

24 April 2015

آيـــــة قرآنيـــة

قال الله تعالى: ((وَالصَّادِقِينَ وَالصَّادِقَاتِ وَالصَّابِرِينَ وَالصَّابِرَاتِ وَالْخَاشِعِينَ وَالْخَاشِعَاتِ وَالْمُتَصَدِّقِينَ وَالْمُتَصَدِّقَاتِ وَالصَّائِمِينَ وَالصَّائِمَاتِ وَالْحَافِظِينَ فُرُوجَهُمْ وَالْحَافِظَاتِ وَالذَّاكِرِينَ اللَّهَ كَثِيرًا وَالذَّاكِرَاتِ أَعَدَّ اللَّهُ لَهُمْ مَغْفِرَةً وَأَجْرًا عَظِيمًا))) الأحزاب-(35)

Dedication

My beloved Parents, My small family (Shahinaz, Abu baker and Omer) ,Sister, brothers and all my friends who have encouraged me throughout this thesis.

Acknowledgement

I would like to express my gratitude to my advisors Associate Professor. Dr Ibrahim khadir and Associate Professor . Dr. Mohammed Hussen for their invaluable support and guidance.

I offer sincere appreciation to my friend Abu baker Waheb allah for their help and patience.

I would also like thank to my friends Elhadi and Ahamed Nyle for their Guidance and encouragement.

Abstract

Mobile ad hoc networks (MANETs) can be defined as a collection of large number of mobile nodes that form temporary network without aid of any existing network infrastructure. there are numbers of attacks that affect MANET. Mobile nodes without adequate protection are easy to compromise. It causes congestion sends false routing information or causes unavailability of services and Challenges in secure communication.

The researcher compared an Elliptic Curve cryptographic(ECC) and Rivet-Shami-Adleman(RSA) a mechanism used for secure MANET And reached the following results ECC uses the lowest key size for encryption / decryption ,less Time key generation if compared with RSA , ECC is better throughput and less time jitter and delay where compared with RSA ,there used ECC is better for secure MANET .

المستخلص:

تعرف الشبكة النقالة بانها مجموعة لعدد كبير من العقد التي تشكل الشبكة المؤقتة دون الاستعانه بالبنية الاساسية لاي من الشبكات الموجودة او نقطة الوصول المركزية.

هنالك عدد من الهجمات تؤثر علي الشبكة النقالة وهي بدون حماية من السهل جدا اختراقها ، وهذا قد يسبب ازدحام للشبكة وترسل معلومات موجة الي مسار خاطئ ويسبب عدم توفر الخدمات وتحدي للاتصال الآمن.

قارن الباحث خوارزمية تشفير المنحنى البيضاوي و خوارزمية المفتاح العام لتامين الشبكة النقالة وتوصل إلى النتائج التالية تستخدم خوارزمية التشفير المنحنى البيضاوي أدنى حجم المفتاح لتشفير / فك التشفير واقل وقت لتوليد المفتاح مقارنة مع خوارزمية المفتاح العام ، و خوارزمية التشفير المنحنى البيضاوي لها افضل انتاجية مقارنة مع خوارزمية المفتاح العام وعليه يفضل استخدام خوارزمية التشفير المنحنى البيضاوي في عملية تأمين الشبكة النقالة.

List OF CONTENTS

List of Figures:

List of Tables:

Abbreviations:

AES	Advanced Encryption Standard
AODV	Ad-hoc On Demand Distance Vector
BSS	Basic Service Set
CA	Certificate Authorities
CSMA/ CA	Carrier Sense Multiple Access with Collision Avoidance
DCF	Distributed Coordination Function
DoS	Denial of service
DSR	Dynamic Source Routing protocol
ECC	Elliptic Curve Cryptography
GUI	Graphic User Interface
IBSS	Independent Basic Service Set
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
MANET	Mobile Adhoc Network
NIST	National Institute of *Standards and Technology*
NRL	Normalized Routing Load
P2P	peer-to-peer
RREQ	Route Request
RREP	Route Reply
PCF	Point Coordination Function
RERR	Route Error
PKI	Public Key Infrastructure
PKC	public key cryptograph
PDF	Packet Delivery Fraction
RSA	Rivest, Shamir and Adleman
TTL	Time To Live
WLAN	Wireless Local Area Network
WoT	Web of Trust

Chapter one

Introduction

1.1 Preface

Mobile Adhoc Network (MANET) is a less infrastructure network with vigorously changing topologies and arbitrary communicating node. At this time the mobile nodes communicate directly with additional nodes without any router and hence the preferred functionalities are embedded to each node. Since the MANET consist of mobile nodes with less configuration of hardware and requirements compared to a router, hence protocols and routing used are of lightweight functionalities [1] .

Now days, MANET is one of the recent active fields and has received marvelous attenuation because of their self- configuration and self- maintenance capabilities [2]. While early research effort assumed a friendly and cooperative environment and focused on problems such as wireless channel access and multi-hop routing, security has become a primary concern in order to provide protect communication between nodes in a potentially hostile environment. Recent wireless research indicates that the wireless MANET present a larger security problem than conventional and wireless networks.

Although MANET have several advantages over traditional wired networks, on the other side's they have a unique asset of challenges.

MANET faces challenges in secure communication. For example the source constraint on nodes in ad hoc networks limits the cryptographic measures that are used for secure messages. Thus it is susceptible to link attacks ranging from passive eavesdropping to active impersonation, message replay and message distortion.

Mobile node without adequate protection is Easy to compromise. an attacker can listen ,modify and attempt to masquerade all the traffic on the wireless communication channel a one of the legitimate node in the network.

Because of the rapid growth of telecommunication and internet, information

security becomes more and more significant. Cryptography is the best way for protecting secret information. Cryptosystems can be divided into two types , secret-key cryptosystem and public-key cryptosystem [3] .

lack of cooperation and constrained capability is common in wireless MANET which makes anomalies hard to distinguish from normalcy. In general, the wireless MANET is particularly vulnerable due to it is fundamental characteristics of open medium, dynamic topology, absence of central authorities cooperation and constrained capability [4].

Most of the routing algorithms designed for MANET such as DSR and AODV are based on the assumption that every node forwards every packet [5] .

1.2 Methodology:

To achieve the objectives of thesis it investigated and specific security for Mobile Adhoc Network used cryptographic and routing protocol for secure communication. however, there decided to Compare RSA and ECC algorithm to measures delay, throughput, jitter , key size , generation key and compare time encryption/decryption for Mobile Adhoc Networks . Then, there studied and analyzed the existing protocols to identify the suitable protocol for this thesis. AODV protocol has been selected as the appropriate protocol in this study, to define as much possible information of the protocol. Then, there simulate the protocol and analyze the simulation result to measure the performance and efficiency of the protocol by differentiates the throughput, normalized rotting load and average end – end delay of the simulation without the malicious node . In this thesis, used crypto tools ,c++ and NS2 to simulating it .

1.3 Problem statement :

- There are numbers of attacks that affect MANET.
- Mobile nodes without adequate protection are easy to compromise.

- It causes congestion sends false routing information or causes unavailability of services.
- Challenges in secure communication.

1.4 Objective:

- to Explain what is MANET and Why?
- to Study and analysis of MANET security features and application.
- to study implementation of MANET security.
- to Focus on the overall security attack and challenges in MANET.
- design system using RSA and ECC Algorithm to Encryption Numeric OR text and send it through MANET to other node.

1.5 Proposed solution :

Compare algorithm RSA and ECC to Encryption file using ns2 ,code-block send number or text message it through MANET. Use two mechanism to secure network .

Chapter two

LITERATURE REVIEW

2.1 Introduction:

Wireless network refers to any type of computer network that is wireless, and is commonly associated with a telecommunications network whose interconnections between nodes is implemented without the use of wires. Wireless telecommunications networks are generally implemented with some type of remote information transmission system that uses electromagnetic waves, such as radio waves, for the carrier and this implementation usually takes place at the physical level or "layer" of the network [6].

A Wireless Local Area Network (WLAN) consists of a set of mobile users communicating via a fixed base station or an access point. The mobile node can be any device such as PDA, laptop etc. as shown in figure 2.1.

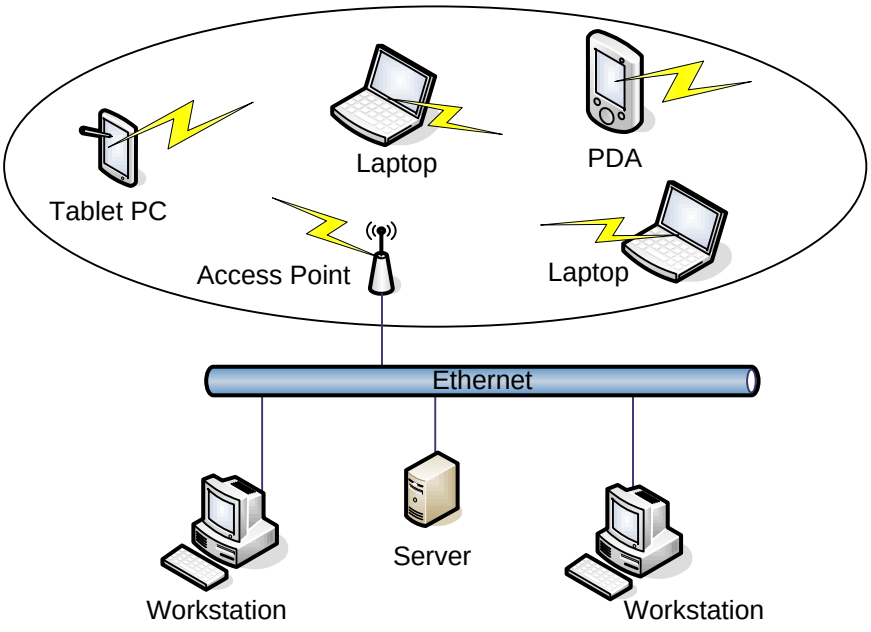


Figure [2.1] Wireless LAN

Such networks are usually deployed in offices, cafeterias, universities, etc. and are

most prevalently used nowadays. There are three types of WLAN – Independent Basic Service Set (IBSS), Basic Service Set (BSS) and Extended Service Set (ESS). A detailed classification is beyond the scope of this thesis. IEEE 802.11 is an adopted international standard for wireless LANs which provides transmission speeds ranging from 1 Mbps to 54 Mbps in either the 2.4 GHz or 5 GHz frequency bands. The latest version of this standard in use today is IEEE 802.11g which provides a bandwidth of up to 54 Mbps.

Mobile Ad hoc networks or MANET are the category of wireless networks which do not require any fixed infrastructure or base stations. They can be easily deployed in places where it is difficult to setup any wired infrastructure. As shown in figure .2.2, there are no base stations and every node must co-operate in forwarding the packets in the network.

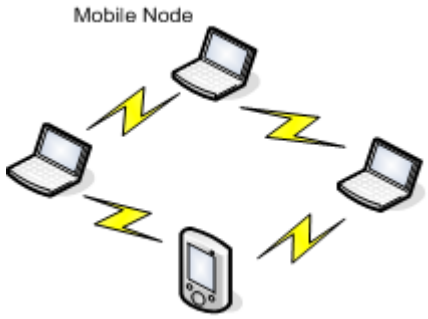


Figure .2.2. A Mobile ad hoc network

Thus, each node acts as a router which makes routing complex when compared to Wireless LAN, where the central access point acts as the router between the nodes.

A sensor network is a special category of ad hoc wireless networks which consists of several sensors deployed without any fixed infrastructure. The difference between sensor networks and ordinary ad hoc wireless is that the sensor nodes may not be necessarily mobile. Further, the number of nodes is much higher than in ordinary ad hoc networks. The nodes have more stringent power requirements since they operate in

harsh environmental conditions. An example of a sensor network is a set of nodes monitoring the temperature of boilers in a thermal plant. Other application domains include military, homeland security and medical care [7].

- Advantage of Mobile Adhoc Network:

Having discussed the general issues in MANET, in their popularity and their benefits:-

- *Low cost of deployment*: As the name suggests, ad hoc networks can be deployed on the fly, thus requiring no expensive infrastructure such as copper wires, etc.
- *Fast deployment*: When compared to WLAN, ad hoc networks are very convenient and easy to deploy .
- *Dynamic Configuration:* Ad hoc network configuration can change dynamically with time. For the many scenarios such as data sharing in classrooms, etc.

- General issue in MANET:

In a Mobile Ad hoc Network, all the nodes co-operate amongst each other to forward the packets in the network and hence, each node is effectively a router. Thus one of the most important issues is routing. This thesis focuses mainly on routing issues in ad hoc networks. In this section, were described some of the other issues in ad hoc networks.

1. Distributed network: A MANET can be considered as a distributed wireless network without any fixed infrastructure. By distributed, there mean that there is no centralized server to maintain the state of the clients, similar to peer-to-peer (P2P) networks.

2. Dynamic topology: The nodes are mobile and hence the network is self-organizing.

3. Power awareness: since the nodes in an ad hoc network typically run on batteries and deployed in hostile terrains, they have stringent power requirements.

4. Addressing scheme: The network topology keeps changing dynamically and hence the addressing scheme used is quite significant.

5. Network size: Commercial applications of ad hoc networks such as data sharing in conference halls, meetings, etc. are an attractive feature of ad hoc networks. However, the delay involved in the underlying protocols places a strict upper bound on the size of the network.

6. Security: Security in an ad hoc network is of prime importance in scenarios of deployment such as battlefield. The three goals of security - confidentiality, integrity and authenticity are very difficult to achieve since every node in the network participates equally in the network [8].

2.2 Network Security :

Security has various definitions based on the requirements of the user. In this work, security is defined as the reliable transmission of information across an insecure network link while the routing protocol determines the route needed to direct packets between the various devices in the mobile ad hoc network . Routing packets securely from one node to the other is difficult to achieve due to several factors such as network environment, number of nodes, information transmission capacity of each node etc. Security performance evaluation can enhance the reliability and dependability of the network by detecting, slowly isolating and removing attacker nodes from the network before it interferes with network performance [23].

2.2 .1 Security Services :

The ultimate goals of the security solutions for MANET is to provide security services, such as authentication, confidentiality, integrity, authentication, no repudiation, anonymity and availability to mobile users. In order to achieve this goal, the security solution should provide complete protection spanning the entire protocol stack. There is no single mechanism that will provide all the security services in MANET. The common security services are described below.

- Availability:

Availability is concerned with the (unauthorized) upholding of resources. A variety of attacks can result in the loss of or reduction in availability. Some of these attacks are amenable to automated countermeasures such as authentication and encryption whereas others require some sort of action to prevent or recover from loss of availability of elements or services of a distributed system. Availability ensures the survivability of network services despite of various attacks[9].

- Confidentiality:

Is probably the most common aspect of information security? there need is protect our confidential information. An organization needs to guard against those malicious actions that endanger the confidentiality of it's information. In military, concealment of sensitive information is the major concern. N industry, hiding some, information from competitors is crucial to the operation of the organization. in banking, customer's accounts need to be kept secret [10].

- Integrity:

Integrity guarantees that the authorized parties are only allowed to modify the information or messages. It also ensures that a message being transmitted is never corrupted. As with confidentiality, integrity can apply to a stream of messages, a single message or selected fields within a message. But, the most useful and straightforward approach is total stream protection.

- Authentication:

Authentication ensures that the access and supply of data is done only by the authorized parties. It is concerned with assuring that a communication is authentic. In the case of a single message, such as a warning or alarm signal, the function is to assure the recipient that the message is from the source that it claims

to be from. Without authentication, an adversary could masquerade as a node, thus gaining unauthorized access to resource and sensitive information and interfering with the operations of the other nodes [11].

- No repudiation:

No repudiation prevents either sender or receiver from denying a transmitted message. Thus, when a message is sent, the receiver can prove that the message was in fact sent by the alleged sender. On the other hand, after sending a message, the sender can prove that the message was received by the alleged receiver. No repudiation is useful for detection and isolation of compromised nodes. When node A receives an erroneous message from node B, no repudiation allows A to accuse B using this message and to convince other nodes that B is compromised.

Scalability:

Scalability is not directly related to security but it is very important issue that has a great impact on security services. An ad hoc network may consist of hundreds or even thousands of nodes. Security mechanisms should be scalable to handle such a large network . Otherwise, the newly added node in the network can be compromised by the attacker and used for gaining unauthorized access of the whole system. It is very easy to make an island-hopping attack through one rough point in a distributed network [10].

2.2.3 Type of Attacks in MANET:

Our three goals of security: Confidentiality, Integrity, and Availability can be threatened by security attacks. there divided into three groups related to security goals:

1. Attacks threading confidentiality (Snooping-Traffic Analysis).
2. Attacks threading integrity(Replaying-Reputation –Modification-Masquerading).
3. Attacks threading availability (Denial of service)[11].

The current Mobile ad hoc networks allow for many different types of attacks. Although the analogous exploits also exist in wired networks but it is easy to fix by infrastructure in such a network. Current MANET are basically vulnerable to two different types of attacks: active attacks and passive attacks. Active attack is an attack when misbehaving node has to bear some energy costs in order to perform the threat. On the other hand, passive attacks are mainly due to lack of cooperation with the purpose of saving energy selfishly. Nodes that perform active attacks with the aim of damaging other nodes by causing network outage are considered as malicious while nodes that make passive attacks with the aim of saving battery life for their own communications are considered to be selfish. In this chapter, our focus is on vulnerabilities and exposures in the current ad hoc network. there are classified the attacks as modification, impersonation, fabrication, wormhole and lack of cooperation [12].

- Attacks Using Modification:

  Modification is a type of attack when an unauthorized party not only gains access to but tampers with an asset. For example a malicious node can redirect the network traffic and conduct DoS attacks by modifying message fields or by forwarding routing message with false values. In figure.2.3, M is a malicious node which can keep traffic from reaching X by continuously advertising to B a shorter route to X than the route to X that C advertises. In this way, malicious nodes can easily cause traffic subversion and denial of service (DoS) by simply altering protocol fields: such attacks compromise the integrity of routing computations. Through modification, an attacker can cause network traffic to be dropped, redirected to a different destination or to a longer route to reach to destination that causes unnecessary communication delay.

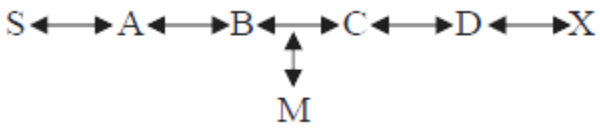


Figure.2.3: Ad hoc network and a malicious node

Consider the following figure.2.4. Assume a shortest path exists from S to X and, C and X cannot hear each other, that nodes B and C cannot hear other, and that M is a malicious node attempting a denial of service attack. Suppose S wishes to communicate with X and that S has an unexpired route to X in its route cache. S transmits a data packet toward X with the source route S --> A --> B --> M --> C --> D --> X contained in the packet's header. When M receives the packet, it can alter the source route in the packet's header, such as deleting D from the source route. Consequently, when C receives the altered packet, it attempts to forward the packet to X. Since X cannot hear C, the transmission is unsuccessful.

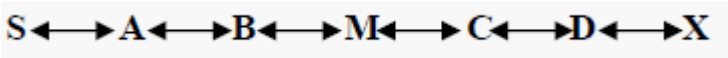


Figure.2.4: Ad hoc network with Dos attack

- Attacks Using Impersonation:

  As there is no authentication of data packets in current ad hoc network, a malicious node can launch many attacks in a network by masquerading as another node i.e. spoofing. Spoofing is occurred when a malicious node misrepresents its identity in the network (such as altering its MAC or IP address in outgoing packets) and alters the target of the network topology that a benign node can gather. As for example, a spoofing attack allows forming loops in routing packets which may also result in partitioning network. Here we have described the scenario in details.

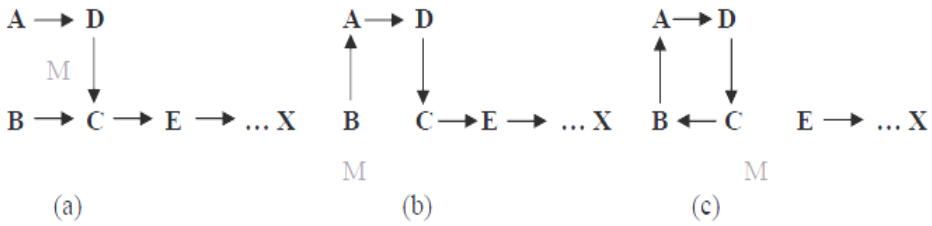


Figure2.5: A sequence of events forming loops by spoofing packets

In the above Figure 2.5 , there exists a path between five nodes. A can hear B and D, B can hear A and C, D can hear A and C, and C can hear B, D and E. M can hear A, B, C, and D while E can hear C and next node in the route towards X. A malicious node M can learn about the topology analyzing the discovery packets and then form a routing loop so that no one nodes in his range can reach to the destination X. At first, M changes its MAC address to match A's, moves closer to B and out of the range of A. It sends a message to B that contains a hop count to X which is less than the one sent by C, for example zero.

Now B changes its route to the destination, X to go through A as shown in the Figure 2.5. Similarly, M again changes its MAC address to match B's, moves closer to C and out of the range of B. Then it sends message to C with the information that the route through B contains hop count to X which is less than E.

- Wormhole Attacks:

  Wormhole attack is also known as tunneling attack. A tunneling attack is where two or more nodes may collaborate to encapsulate and exchange messages between them along existing data routes. This exploit gives the opportunity to a node or nodes to short-circuit the normal flow of messages creating a virtual vertex cut in the network that is controlled by the two colluding attackers. In the Figure2.6 M1 and M2 are two malicious nodes that encapsulate data packets and falsified the route lengths.

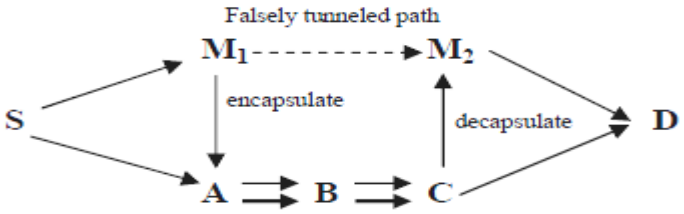


Figure2.6: Path length spoofed by tunneling

Suppose node S wishes to form a route to D and initiates route discovery. When M1 Receives a RREQ from S, M1 encapsulates the RREQ and tunnels it to M2 through an Existing data route, in this case {M1 --> A --> B --> C --> M2}. When M2 receives the Encapsulated RREQ on to D as if had only traveled {S --> M1 --> M2 --> D}. Neither M1 from S of unequal length: one is of 5 and another is of 4. If M2 tunnels the RREP back to M1, S would falsely consider the path to D via M1 is better than the path to D via A. Thus, tunneling can prevent honest intermediate nodes from correctly incrementing the metric used to measure path lengths. M2 update the packet header. After route discovery, the destination finds two routes.

- Lack of Cooperation:

Mobile Ad Hoc Networks (MANET) relies on the cooperation of all the participating nodes. The more nodes cooperate to transfer traffic, the more powerful a MANET gets. But one of the different kinds of misbehavior a node may exhibit is selfishness. A selfishness node wants to preserve own resources while using the services of others and consuming their resources. This can endanger the correct network operation by simply not participating to the operation or by not executing the packet forwarding. This attack is also known as the black hole attack [13].

2.2.4 Threatened in Mobile Adhoc:

- Security Threats in Physical Layer:

Physical layer security is important for securing MANET as many attacks can take place in this layer. The physical layer must adapt to rapid changes in

link characteristics. The most common physical layer attacks in MANET are eavesdropping, interference, denial-of-service and jamming. The common radio signal in MANET is easy to jam or intercept. Moreover an attacker can overhear or disrupt the service of wireless network physically.

An attacker with sufficient transmission power and knowledge of the physical an medium access control layer mechanisms can gain access to the wireless medium. Here we will describe eavesdropping, interference and jamming attacks in brief.

- Eavesdropping:

  Eavesdropping is the reading of messages and conversations by unintended receivers. The nodes in MANET share a wireless medium and the wireless communication use the RF spectrum and broadcast by nature which can be easily intercepted with receivers tuned to the proper frequency. As a result transmitted message can be overheard as well as fake message can be injected into the network.

- Interference and Jamming:

  Jamming and interference of radio signals causes message to be lost or corrupt. A powerful transmitter can generate signal that will be strong enough to overwhelm the target signal and can disrupt communications. Pulse and random noise are the most common type of signal jamming .

- Security Threats in Link Layer

  The MANET is an open multi-point peer-to-peer network architecture in which the link layer protocols maintain one-hop connectivity among the neighbors. Many attacks can be launched in link layer by disrupting the cooperation of the protocols of this layer .Wireless medium access control (MAC) protocols have to coordinate the transmission of the nodes on the

common communication or transmission medium. The IEEE 802.11 MAC protocol uses distributed contention resolution mechanisms which are based on two different coordination functions. One is Distributed Coordination Function (DCF) which is fully distributed access protocol and the other is a centralized access protocol called Point Coordination Function (PCF). For resolving channel contention among the multiple wireless hosts, DCF uses a carrier sense multiple access with collision avoidance or CSMA/CA mechanism.

- Security Threats in network Layer:

In MANET, the nodes also function as routers that discover and maintain routes to other nodes in the network. Establishing an optimal and efficient route between the communicating parties is the primary concern of the routing protocols of MANET. Any attack in routing phase may disrupt the overall communication and the entire network can be paralyzed. Thus, security in network layer plays an important role in the security of the whole network A number of attacks in network layer have been identified and studied in security research. An attacker can absorb network traffic, inject themselves into the path between the source and destination and thus control the network traffic flow. Network layer vulnerabilities fall into two categories: routing attacks and packet forwarding attacks . The family of routing attacks refers to any action of advertising routing updates that does not follow the specifications of the routing protocols. The specific attack behaviors are related to the routing protocol used by the MANET [24].

2.2.5 Routing Protocol:

Routing in mobile ad hoc networks faces additional problems and challenges when compared to routing in traditional wired networks with fixed infrastructure. There are several well- known protocols in the literature that have been specifically developed to copy with the limitations imposed by ad hoc networking environments. The problem of routing in such environments is

aggravated by limiting factors such as rapidly changing topologies, high power consumption, low bandwidth and high error rates . Most of the existing routing protocols follow two different design approaches to confront the inherent characteristics of ad hoc networks, namely the table-driven and the source-initiated on-demand approaches.

The following sections analyze in more detail these two design approaches, and brief present example protocols that are based on them [14]. There are two common reactive ad-hoc routing protocols which are under consideration for ratification by the Internet Engineering Task Force (IETF) [15].

- Reactive ad-hoc routing protocols :

Reactive routing protocols such as AODV and DSR operate by running two procedures concurrently: route discovery and route maintenance. A source invokes route discovery when it wants to send data to a destination, creating a Route Request (RREQ) that is disseminated network-wide by neighboring nodes. If the destination or a neighbor with a valid route to the destination receives the request, a Route Reply (RREP) is sent back to the source. When the source receives the reply, a route table entry is added and data packets for the destination are serviced and sent onwards to the appropriate next hop (Johnson & Maltz, 1996; Perkins, 1997). If the route discovery procedure times out and fails to discover a route, a suitable unreachable host message is sent to the application, and the packets for the corresponding unreachable destination are dropped.

The second procedure, route maintenance, monitors local connectivity to ensure routes are preserved by detecting link breaks between pairs of nodes (Johnson & Maltz, 1996; Perkins, 1997). Two methods are commonly employed in order to detect link breaks, which are periodic HELLO messages or from feedback directly from the link layer. HELLO mes-sages are broadcast packets that are sent periodically to advertise a node of its presence. If a HELLO message is received by a eighbor, it

assumes bidirectional communication to the sender. Link layer feedback is provided by the MAC protocol after it deems a frame as undeliverable, usually if multiple retransmissions of the frame are unsuccessful. The latter is more reactive to changes in topology and introduces no further overheads (Chakeres & Belding-Royer, 2005) but it requires specific hardware and driver support. If a link break is detected, local repair is often used as a first attempt to restore connectivity locally. This is where a request is sent with a small Time To Live (TTL) in order to find an alternative eighbor and is often transparent to the source. If local repair is unsuccessful, a Route Error (RERR) is sent to the source causing it to rediscover a route to the destination.

Route maintenance is also involved in removing inactive routing entries to minimize the propagation of stale routing entries and to keep the size of routing tables down.

DSR and AODV use similar routing messages that have similar roles but there are important differences in the operation of each protocol. Firstly, the forwarding process is very different. DSR is a source routing protocol where the sender appends the full path to the destination onto the packet header that contains a list of intermediate nodes the packet should traverse (Johnson & Maltz, 1996). Intermediate nodes forward to the next hop by looking at the next hop entry in the path and sending the data onward.

In contrast, AODV does not manipulate data packets, being a distance vector protocol based on the Bellman-Ford routing algorithm. Instead, the next hop decision is made at each intermediate node (Perkins et al., 2001). Secondly, DSR caches multiple routes to a destination, permitting more redundancy during data transfer over AODV, which caches one route per destination. Lastly, DSR specifically states the use of link layer feedback (Johnson & Maltz, 1996) whilst AODV routers can be configured to send periodic HELLO messages or can use link layer feedback [16].

- Ad-hoc On Demand Distance Vector (AODV) :

  The Ad hoc On Demand Vector routing algorithm is a routing protocol designed for ad hoc mobile networks. AODV is capable of both unicast and

multicast routing. It is an on demand routing algorithm, means that it builds routes between nodes only as desired by the source nodes. It maintains these routes as long as they are needed by the sources.

AODV uses sequence numbers to ensure the freshness of routes. It is a loop-free, self starting and scales to large number of mobile nodes. For example, node S intends to find a route to node D, the process is shown in the below Figure [2.7]

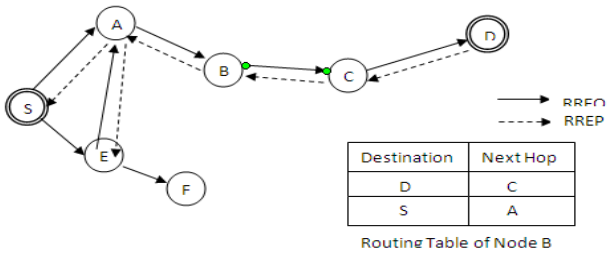


Figure 2.7:AODV Routing Protocol Model

AODV builds routes using a route REQUEST and route REPLY query cycle. When a source node desires a route to a destination for which it does not have a route, it broadcasts a route request (RREQ) packet across the network. Nodes receiving the packet update their information for the source node and set up backwards pointers to the source node in the route tables. In addition to the source node's IP address, current sequence number and broadcast ID, the RREQ also contains the most recent sequence number for the destination of which the source node is aware. A node receiving the RREQ may send a route reply (RREP) if it is either the destination or if it has a route to the destination with the corresponding sequence number greater than or equal to that contained in the RREQ. If this is the case, it uncast a RREP back to the source. Otherwise, it rebroadcasts the RREQ.

Nodes keep track of the RREQ's source IP address and broadcast ID. If they receive a RREQ which they have already processed, they discard the RREQ and do not forward it.

RREP propagates back to the source, nodes set up forward pointers to the

destination. Once the source node receives the RREP, it may begin to forward data packets to the destination.

As long as the route remains active, it will continue to be maintained. A route is considered active as long as there are data packets periodically travelling from the source to the destination along that path. Once the source stops sending data packets, the links will timeout and eventually be deleted from the intermediate node routing tables. If a link break occurs while the route is active, the node upstream of the break propagates a route error (RERR) message to the source node to inform it of the now unreachable destination(s). After receiving the RERR, if the source node still desires the route, it can reinitiate route discovery [16].

- The Dynamic Source Routing protocol (DSR):

  The key feature of DSR is the use of source routing. That is, the sender knows the complete hop –by-hop route to the destination. These routes are stored in a route cache. The data packets carry the source route in the packet header. When a node in the ad hoc network attempts to send a data packet to a destination for which it does not already know the route, it uses a route discovery process to dynamically determine such a route. Route discovery works by flooding the network with route request (RREQ) packets. Each node receiving an RREQ rebroadcasts it, unless it is the destination or it has a route to the destination in its route cache. Such a node replies to the RREQ with a route reply (RREP) packet that is routed back to the original source. RREQ and RREP packets are also source routed. The RREQ builds up the path traversed across the network. The RREP routes itself back to the source by traversing this path backward. The route carried back by the RREP packet is cached at the source for future use. If any link on a source route is broken, the source node is notified using a route error (RERR) packet. The source removes any route using this link from its cache. A new route discovery process must be initiated by the

source if this route is still needed. DSR makes very aggressive use of source routing and route caching [17].

2.2.6 Cryptographic background:

As mentioned in the introductory chapter of this thesis, a large part of our research was motivated by the restrictions of the mobile devices for which our protocols are designed. In this chapter we describe and evaluate the efficiency of the most common cryptographic primitives, both symmetric techniques and public-key techniques. This chapter is not intended as a rigorous and detailed explanation of the cryptographic primitives there are described, but rather to help under-stand the performance differences between these primitives. As such, there will not define every notion that is required to completely analyze many security aspects.

For example, there will used the terms “easy”, “hard”, “infeasible”, etc. without mathematical definitions to describe the exact meaning of these terms. there refer to for exact definitions, detailed descriptions and rigorous security analyses of the primitives we briefly discuss here [18].

here are two primary kinds of cryptographic algorithms: symmetric key algorithms, which use the same key for encryption and decryption, and asymmetric key algorithms, which use two different keys for encryption and decryption. In the following sections, these two algorithms will be discussed in addition to digital signature, digital certificate, Public Key Infrastructure (PKI) and Web of Trust (WoT) models.

2.2.6.1 Symmetric Key Algorithms:

In conventional cryptography, symmetric key algorithms rely on the presence of the shared key at both the sender and receiver, which has been exchanged by some previous arrangement (e.g. through a secure communication channel). This shared key is used for both encryption and decryption. It means

that symmetric key cryptography is the process whereby the sender and the receiver use the same key private key (k) to encrypt and decrypt. Symmetric encryption is illustrated in Figure[2.8] Alice encrypts the plain text message m using the shared key k and converts it into cipher text c. In order to recover the plain text message m, Bob decrypts the received cipher text c using the same key used for the encryption.

Symmetric-key algorithms can be divided into stream ciphers and block ciphers. Stream ciphers encrypt the bits of the message one at a time, while block ciphers take a number of bits and encrypt them as a single unit. Blocks of 64 bits have been commonly used; the Advanced Encryption Standard (AES) algorithm approved by National Institute of Standards and Technology (NIST) in December 2001 uses 128-bit blocks [19].

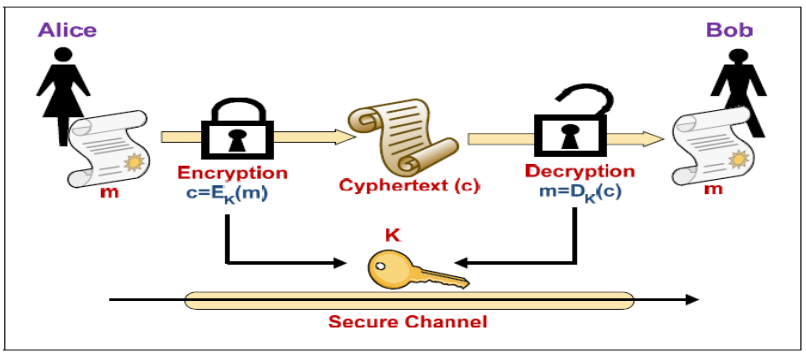


Figure 2.8: Symmetric encryption scheme

Symmetric key algorithms are usually quicker to execute electronically, but needs a secret key to be shared between the sender and receiver. When communication needs to be recognized among nodes, each one of the sender-receiver pair should share a key, which makes the system non-scalable. If the same key is used between more than two nodes, a breach of security at any one point makes the whole system vulnerable.

2.2.6.2 Asymmetric Key Algorithms :

The problems of key management in symmetric key algorithms are solved by public key cryptography (asymmetric key) the concept of which was introduced by Whitfield Diffie and Martin Hellman in 1976 . Public key cryptography is forms of cryptography were a user has a pair of cryptographic keys - a public key and a private key. The private key is kept secret, while the public key may be widely distributed. The keys are related mathematically, but the private key cannot be practically derived from the public key. A message encrypted with the public key can be decrypted only with the corresponding private key.

The public key encryption scheme is illustrated in Figure *2.9* At the beginning, both Alice and Bob should have a pair of public and private keys. If Alice wants to send an encrypted message m to Bob, she first needs to get Bob‘s public key (PK) and make Bob sure that this key is authenticated. This public key is used to encrypt the message m and convert it into cipher text c. Bob can then decrypts this cipher text using the corresponding private key (SK) which is known only by him. Bob The two main branches of public key cryptography are:

•Public key encryption — to ensure confidentiality a message should be encrypted with a recipient's public key which cannot be decrypted by anyone except the by the recipient possessing the corresponding private key.

• Digital signatures — to guarantee authenticity, integrity and non-repudiation a message signed with a sender's private key can be confirmed by anyone who has access to the sender's public key, thereby proving that the sender signed it and that the message has not been tampered with.

A central problem for public-key cryptography is proving that a public key is authentic, and has not been tampered with or replaced by a malicious third party. The usual approach to this dilemma is to use a public-key infrastructure (PKI), in which one or more third parties, known as Certificate Authorities (CA), certify ownership of

key pairs.

A very popular example of public key cryptography is the RSA system developed by Rivest, Shamir and Adleman, which is based on the integer factorisation problem. In RSA, to encrypt a message m or decrypt a cipher text c, the following Calculations are performed:

$$c = m^e \bmod n \qquad (2.1)$$

$$m = c^d \bmod n \qquad (2.2)$$

A major benefit of public key cryptography is that it provides a method for employing digital signatures. Digital signatures permit the receiver of information to verify the authenticity of the information's origin, and also that the information is intact. Thus, public key digital signatures provide authentication and data integrity. A digital signature also provides non-repudiation, meaning that it prevents the sender from arguing that he or she did not actually send the information. The basic manner in which digital signatures are created is illustrated in Figure*2.10* Instead of encrypting information using someone else's public key, it is encrypted with the sender‘s private key. If the information can be decrypted with the sender‘s public key, then it must have originated with that sender.

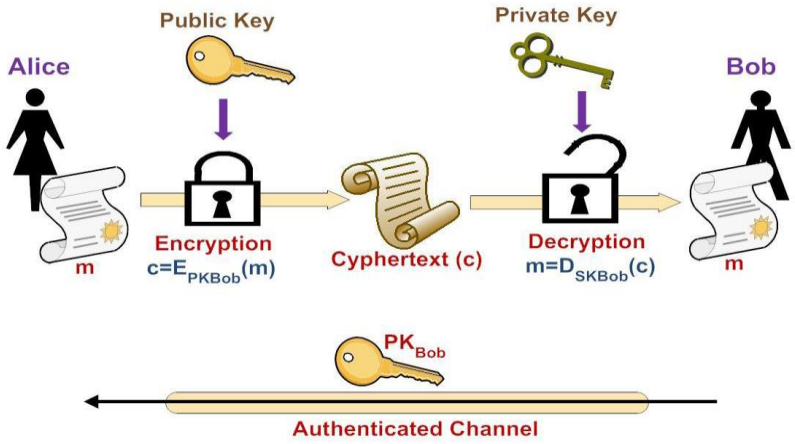


Figure 2.9 : Asymmetric encryption scheme

A digital signature serves the same purpose as a handwritten signature. However, a

handwritten signature is easy to counterfeit. A digital signature is superior to a handwritten signature in that it is nearly impossible to imitate. It also attests to the contents of the information as well as the identity of the signer.

The basic manner in which digital signatures are created is illustrated in Figure *2.11* Instead of encrypting information using someone else's public key, it is encrypted with the sender‘s private key. If the information can be decrypted with the sender‘s public key, then it must have originated with that sender.

As can be seen in Figure2.11 , Alice wants to send a message m to Bob which is signed by her. Alice uses the hash digest of the message m and her private key to create the signature. First she uses a hash function on the message m and computes the hash digest. Then, she encrypts this digest using her private key (SKAlice) and sends it with the message to Bob. Bob recomputed the digest by applying the same hash function on the received message m and compares it with the digest resulted from decrypting the signature using the public key of Alice (PKAlice). If both digests match, then the message m must have originated from Alice and not been modified during transmission.

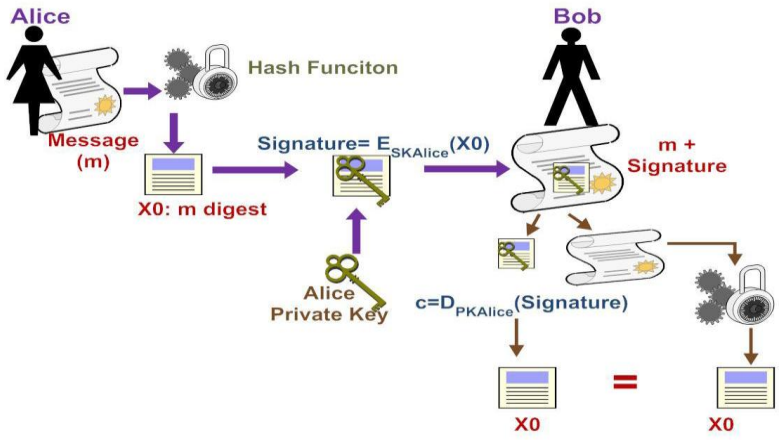


Figure 2.10: Digital Signature

- Certificates vs. Keys :

Traditional public key cryptography (PKC) uses certificates issued by certification authority to bind users with public keys. KMS is the best alternative to certificates since they suffer from following drawbacks :

•-Usage of certificates requires sending messages with signatures which uses more memory and strains the bandwidth.

•Receiver nodes need to validate certificates and verify signatures which are time consuming and affect the performance of the system.

• Certificate management is tricky and also cumbersome.

• A queue of certificates being issued must be verified by the verifier which is a problem and is time consuming.

- Key Management System:

Cryptography is based on keys for its functioning. The encrypted information can be revealed if the key is covered. Managing the keys used in the system is called KMS which includes several algorithms related to the functionality of keys. In this work, the emphasis is mainly on key generation, key agreement and key encryption/decryption. It on statutes of three protocols described below:

•Key Distribution: protocol requires an authority to create or otherwise obtain secret values and securely distribute to other node in the network.

• Key Agreement: requires each node to establish a common session and a shared secret key without a trusted third party involvement using successive pair-wise message exchange. They are fully distributed and self organized [20].

Chapter three

RSA and ECC security techniques and Implementations

3.1 RSA Cryptosystem:

For RSA to be secure, it should be computationally infeasible to factor n = pq even when using the best factoring algorithms, i.e. p and q should be sufficiently large. If p and q are known, it is easy to compute φ(n) = (p–1)(q–1) and derive a. At present, it is recommended that p and q should each be primes having around 100 digits . However, it should be noted that there are also a number of attacks on RSA that do not involve the factoring of n at all. They generally exploit weaknesses in the setup of the cryptosystem, such as poor choices of a , or Abu baker´ usage of the same n to communicate with other people [21] .

RSA_Key_Generation

{

Select tow large primers p and q. N =pXq

Phie(n)=(p-1)(q-1)

Select e such that 1<e< phie(n) and e is coprime to phie(n)

D=e^{-1}mod phie(n) Public_key =(c,n) Privae_key =d

Return public_key and private_key

}

RSA Encryption: Suppose node1 wishes to send a message (say 'm') to Alice. To encrypt the message using the RSA encryption scheme, node1 must obtain node2 public key pair (e,n). The message to send must now be encrypted using this pair (e,n). However, the message 'm' must be represented as an integer in the interval [0,n-1]. To encrypt it, node1 simply computes the number 'c' where c = m ^ e mod n. node1 sends the ciphertext c to node2.

Rsa_encryption(p,e,n)

{

C=fast_exponentation(p,e,n) \\ calculation of (p^e mod n)

Return c

}

RSA Decryption

Rsa_decryption(c,d,n)

{

p=fast_exponentation(c,d,n) \\ calculation of (c^e mod n)

Return p

}

3-2 ECC Algorithm:

Elliptic Curve Cryptography (ECC) was invented by Neal Koblitz and Victor miller in 1985. They can be viewed as Elliptic curve analogues of older discrete logarithm (DL) cryptosystem. Mathematically basis for security of elliptic curve cryptography is computational intractability of elliptic curve discrete logarithm problem (ECDLP). Where can define elliptic curve over finite field. Elliptic curve digital signature algorithm ECDSA) is one of variants of elliptic curve cryptographic (ECC) proposed as an alternative to established public key system such as digital signature algorithm and RSA algorithm have recently gained a lot of attention in industry and academia [22].

Some of the key concepts in ECC are:

1. ECC offers considerably greater security for a given key size.

2. The smaller key size also makes possible much more compact implementations for a given level of security.

- Elliptic Curve Cryptography(ECC):

It contains certain advantages. ECC devices require less storage, less power, less memory, and less bandwidth than other systems. This allows you to implement cryptography in platforms that are constrained, such as wireless devices, hand held computers, smart cards, and thin-clients. It also provides a big win in situations where efficiency is important. For example, the current key size recommendation for legacy public schemes is 2048 bits. A vastly smaller 224-bit ECC key offers the same level of security. This advantage only increases with security level— for example, a 3072 bit legacy key and a 256 bit ECC key are equivalent—something that will be important as stronger security systems become mandated and devices get smaller. Elliptical curve cryptography is a public key encryption technique based on elliptic curve theory that can be used to create faster, smaller, and more efficient cryptographic keys. ECC generates keys through the properties of the elliptic curve equation instead of the traditional method of generation as the product of very of large prime numbers like in RSA. Because ECC helps to establish equivalent security with lower computing power and battery resource usage, it is becoming widely used for mobile applications. Unlike other popular algorithms such as RSA, ECC is based on discrete logarithms that are much more difficult to challenge at equivalent key lengths.

- Proposed Alogrithm of ECC :-

To do operations with EC points in order to encrypt and decrypt the points are to be generated first. The algorithm 'genPoints' describes the process of generating the points for the given parameters 'a', 'b', and 'p'. Also the algorithm 'ECC' describes the process of encryption and decryption on EC field.

Implementation of the ECC :

The typical Elliptic Curve is represented by:

$$Y^2 \bmod 163 = x^3 + x + l \bmod 163 \qquad (3.1)$$

3.3 Computer Model :

The GUI application was developed using c++ running with code-Block and Crypt Tools to analyzed system Our application consists of more classes, of which the class named RSA com pined with AES and ECC combined with AES is the most important. It provides for encrypting files, decrypting files, mailing the encrypted file to another user, loading the values of the RSA keys from the key-files, saving encrypted/decrypted files and so on . as Show Figure 3.1 and Figure 3.2

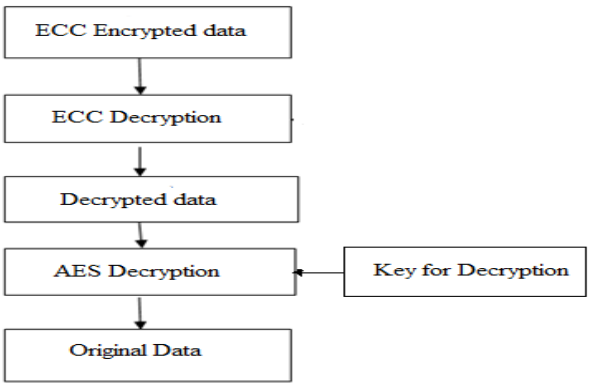


Figure 3.1 flow chart decrypting data use ECC

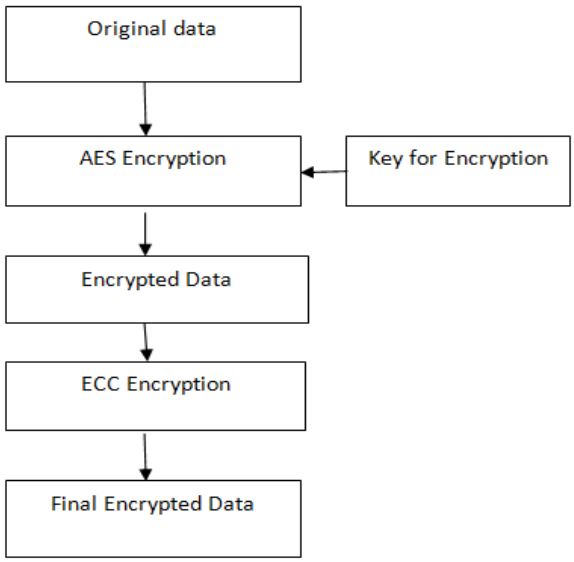


Figure 3.2: flow chart encrypting data use ECC

3.4 System Model:

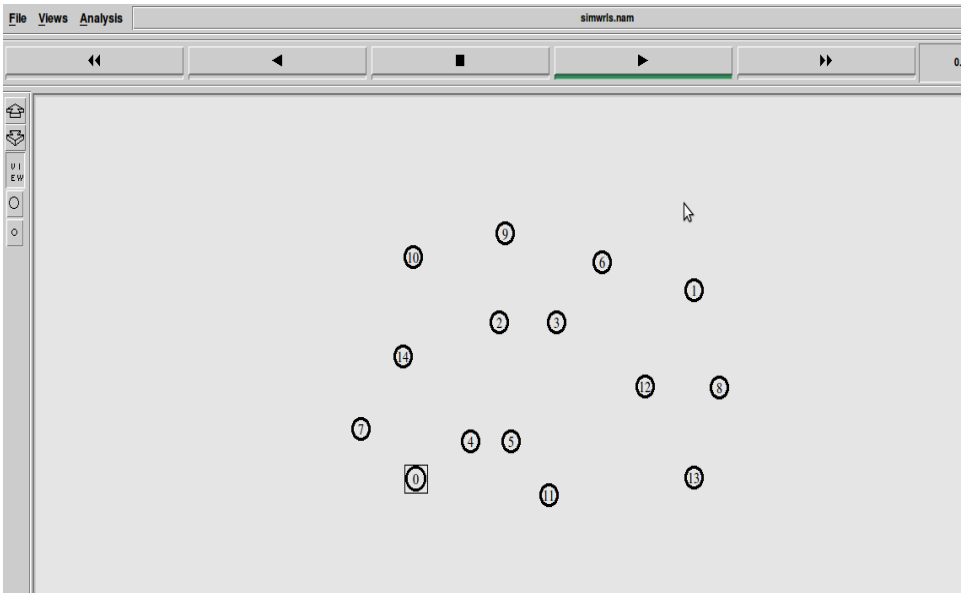


Figure 3.3 Mobile Adhoc Network

this system module consists of modules:

•System module Interface: basically communicates with the user to get the simulation parameters and commands and provides the results obtained from those simulations.

•Scenario Generator: generates the crypto tools scenario file based on the selector parameters used for analyzed .

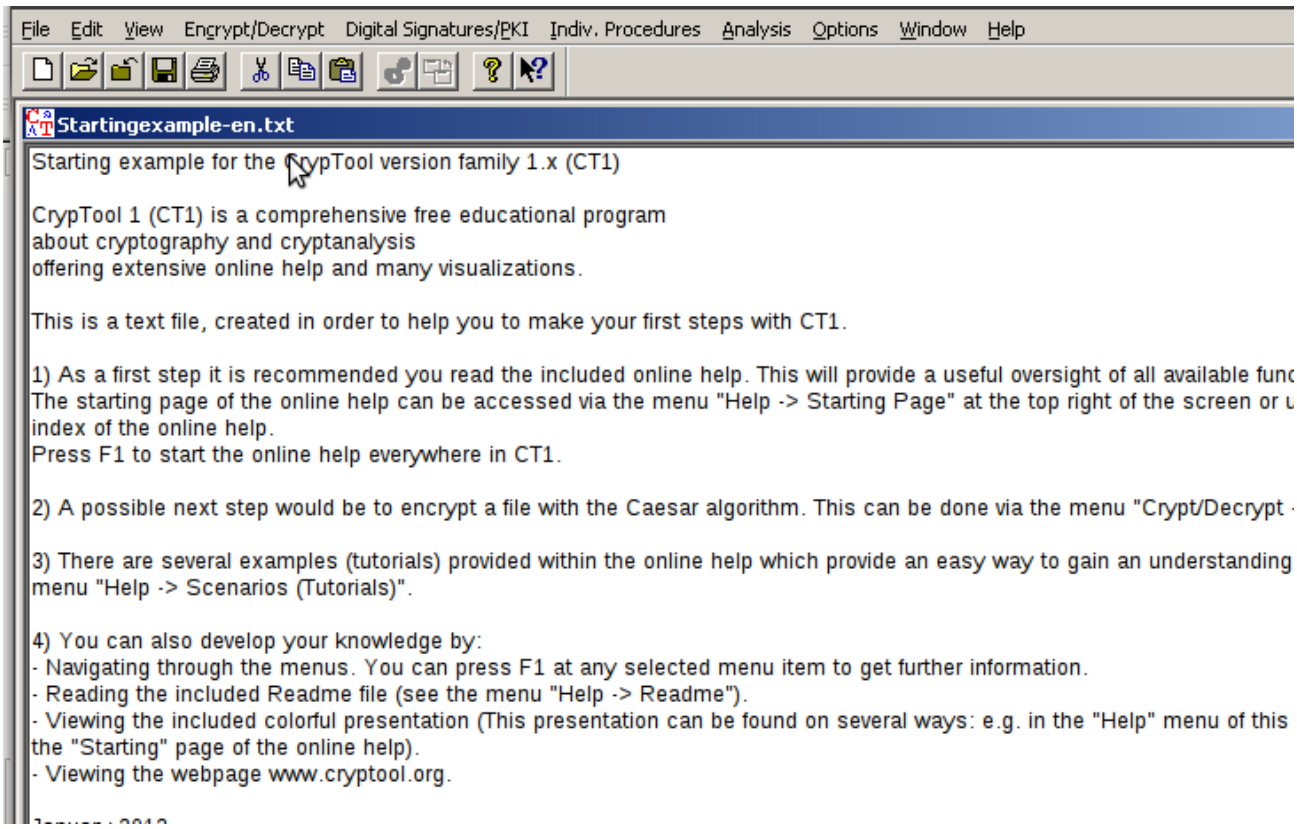


Figure 3.4 a:main screen for crypto tools

Main System: it is first step a pear when were running system, it has three choice, first choice for secure message that you want to send it, second it performance network and third exit form system see Figure 3.4 .

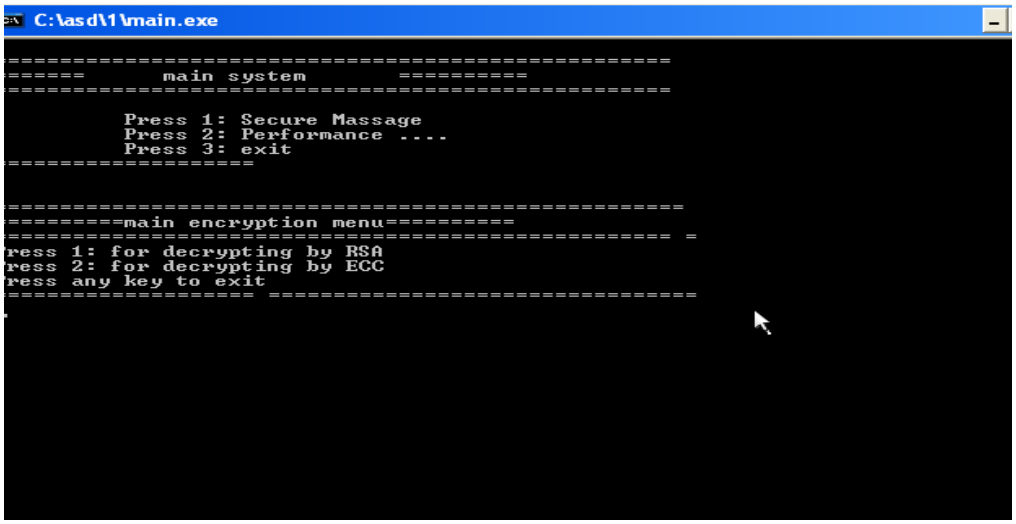


Figure 3.5 : RSA and ECC Encryption system

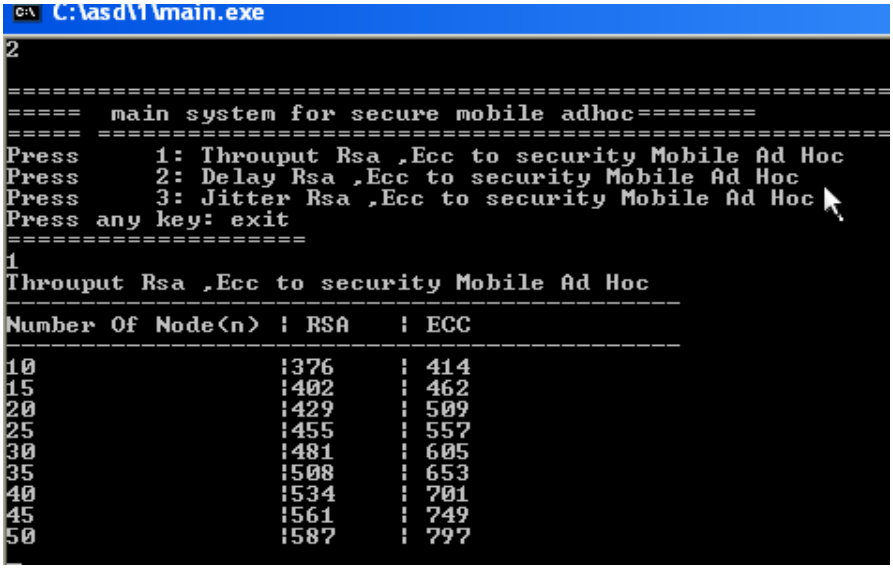


Figure 3.6: throughput RSA and ECC in mobile adhoc

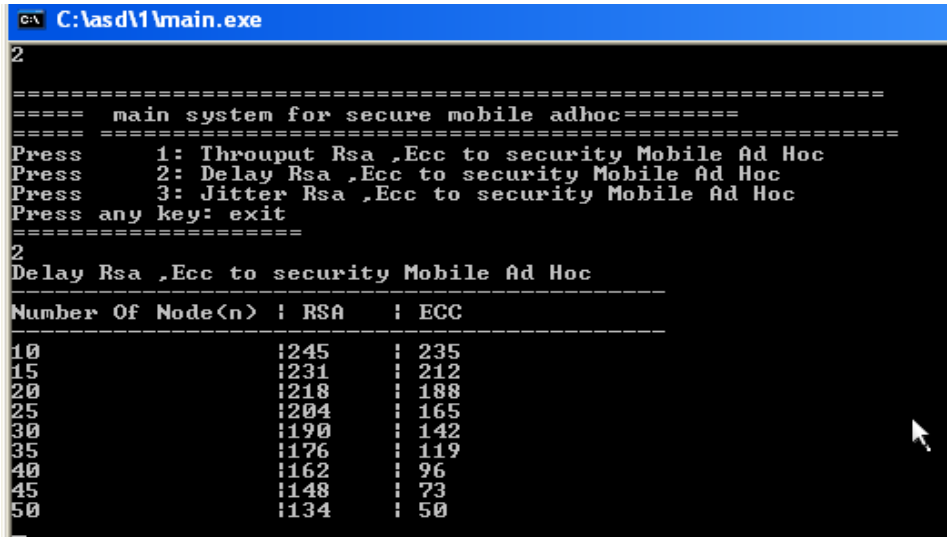


Figure 3.7: Delay RSA and ECC in mobile adhoc

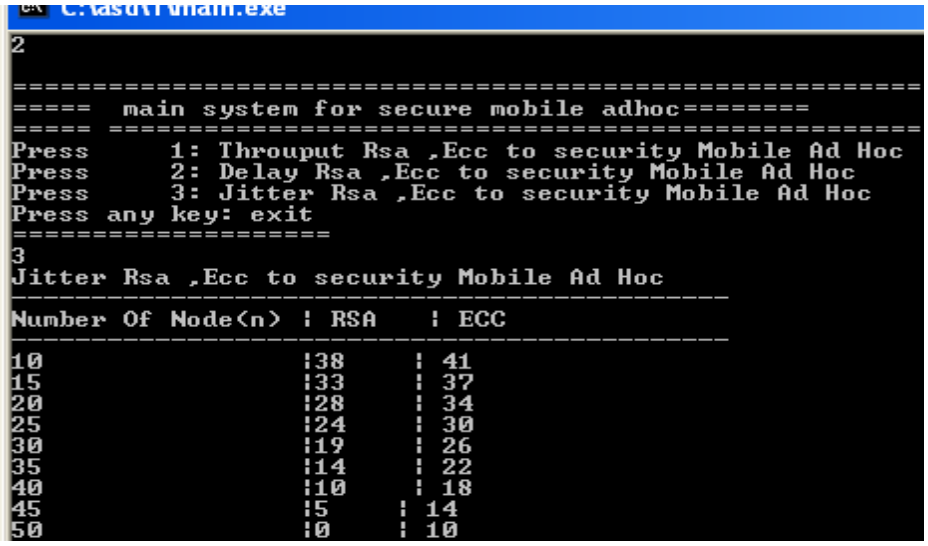


Figure 3.8: Jitter RSA and ECC in mobile adhoc

3.5: Simulation Environment

the version 2.31 of NS-2, was used to simulate the ECC and RSA algorithm. There used 100 nodes, each experiment was executed 100 times and the average end-end delay and jitter are calculated.

3.6 Performance Analysis

Theoretically, ECC showed its improvement over RSA. But in real world where scalability was an important issue, ECC needs to prove its improvement over RSA. For

analyzing the performance over a large distributed network usually used network simulators. These simulators virtually create nodes and simulate them. The previous chapter explains a detailed description of the proposed ECC , RSA used Crypto tools and Routing Protocol (AODV and DSR). This chapter compares the performance of the proposed ECC and RSA method with respect to the parameters of the average throughput, average delay time and average jitter of the network. Before representing the results of the simulation, its environment and methodology are explained which was used to carry out the results. Later, the simulation result analysis based on the performance metrics are discussed. Finally, a comprehensive summary of the simulation result analysis is given.

3.6.1 Simulation Description

In this section the detailed description about the simulator environment and how the environment variables are configured for analysis is given.

3.6.2 Performance Metrics

To evaluate the performance of the ECC with the existing RSA, these parameters were chosen. These parameters are,

1. key size , key generation, time encryption / time decryption.
2. Average Throughput of the Network.
3. Average Delay Time of the Network'
4. Average Jitter of the Network.

Also evaluate routing protocol AODV and DSR these parameters are:

1. Packet Delivery Fraction(PDF) :

   PDF=(recvLine/sendLine)*100; (3.2)

2. Average End-to-End delay:

   Avg_End_to_End_Delay=(0.260157/recvLine) (3.3)

3. Normalized Routing Load(NRL) :

   NRL=(sendLine1+fowardLine1)/recvLine; (3.4)

Chapter four

Results and discussion

4-1 Simulation Result:

This section describes the performance of the proposed method with comparing the existing method considering the performance metrics. The following subsections show the performance of the proposed and existing method. we find good equations for delay ,throughput and jitter RSA and ECC logarithm by used SPSS program.

4.1.1 Key Size Comparison (in bits) for Equivalent Security Level:

From the table 4.1, it is clear that symmetric key encryption algorithm like (ECC) uses the lowest key size for encryption. RSA key size is used making cryptanalysis attack difficult than symmetric key encryption algorithm. Although ECC dramatically reduces the key size and key generation, So ECC can provide strong security by using lower key size and time key generation than any other asymmetric key encryption algorithm like RSA.

Table 4.1 Key generation (ms) ECC and RSA

Security Bits	*RSA*	*ECC*	Key generation (ms) ECC	Key generation (ms) RSA
80	512	192	18	39
112	768	196	22	68
128	1024	239	22	241
192	2048	256	26	1524

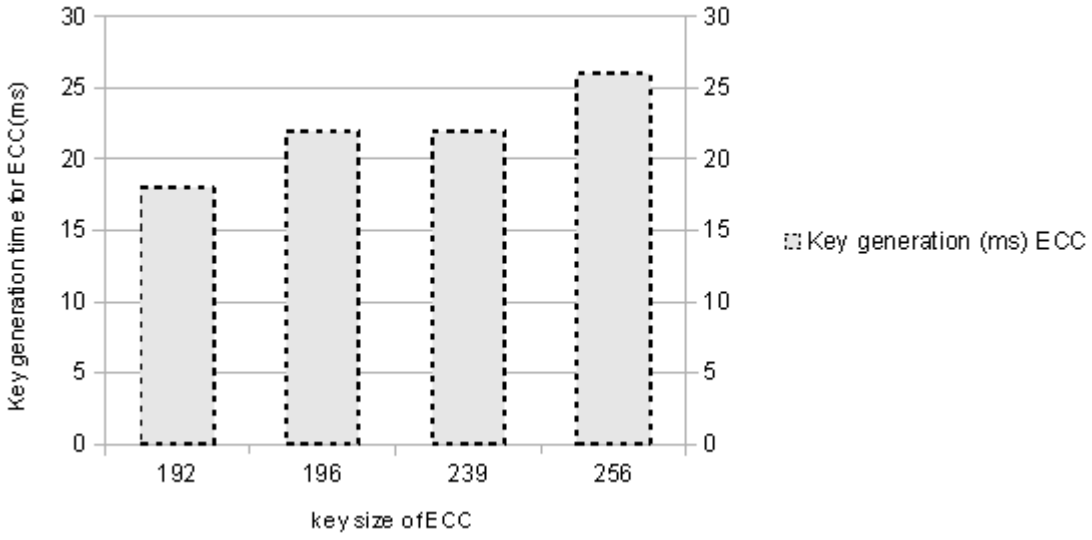
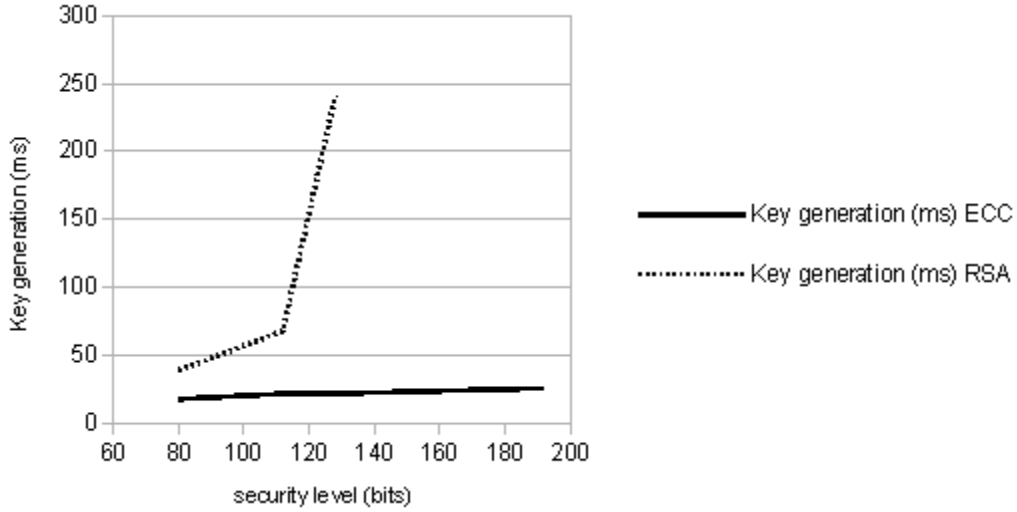


Figure 4.1 key generation Vs key size for ECC

From Figure 4.1 time generation key for ECC increased depends key size.



Figure(4.2) key generation Vs security level for RSA and ECC

From Figure 4.2 time key generation for RSA more than time key generation for ECC that means the ECC is complex encryption than RSA.

4.1.2 Time Encryption and Decryption:

Table 4.2 a time encryption for ECC and RSA

Security Bits	*RSA*	*ECC*	Time Encryption(ms) ECC	Time Encryption(ms) RSA
80	512	192	51	0
112	768	196	47	2
128	1024	239	53	0
192	2048	256	53	2

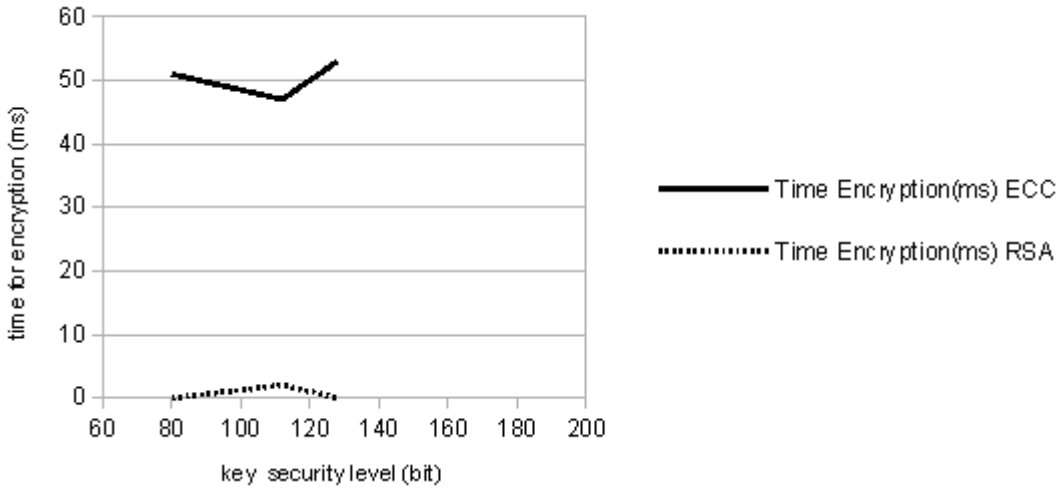


Figure (4.3) time encryption for ECC and RSA

From Figure 4.2 , it is observed that the time encrypted file increases for ECC since RSA has comparatively long keys than ECC. For RSA 2048, the size of encrypted files increases rapidly. This proves the complexity of ECC algorithm .

Table 4.3. time decryption for ECC and RSA

Security Bits	RSA	ECC	Time decryption(ms) ECC	Time decryption(ms) RSA
80	512	192	18	16
112	768	196	24	2
128	1024	239	75	26
192	2048	256	22	109

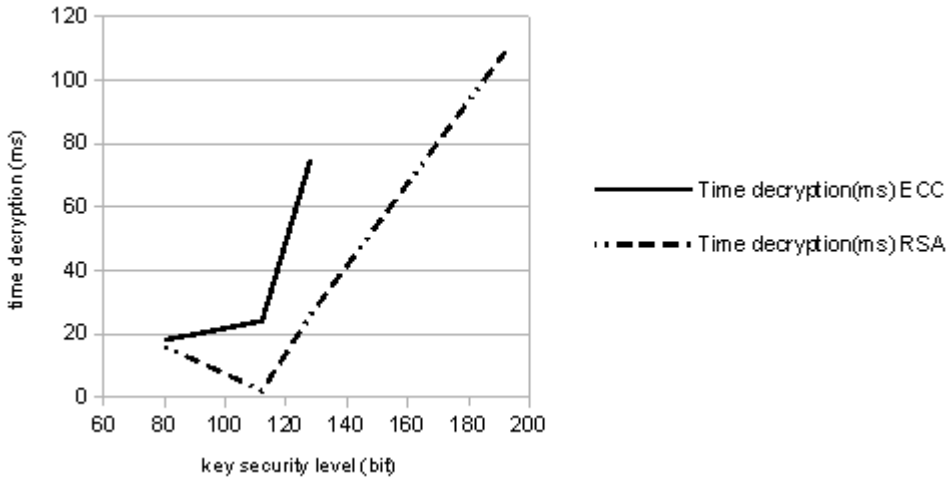


Figure (4.4) time decryption for ECC and RSA

From Figure 4.4 , it is observed that the time decrypted file decreases for ECC since RSA has comparatively long keys than ECC.

4.1.3 Average Throughput Analysis:

Throughput is the ratio of number of packets received successfully by a node within a given period of time. Average throughput gives us a view of the network data communication. The higher throughput indicates the higher data transmission in the network.

From the Figure 4.5: the simulation analysis of ECC and RSA with equivalent security strength is visualized. When the number of nodes is lower, node 10, ECC

improves the average throughput it greater than RSA with equivalent security level. But dramatically ECC increases average throughput to 26% when number of nodes increased to 50. The average throughput curve of ECC increases exponentially than RSA because of the sub exponential behavior of computational complexity of RSA.

Table 4-4 throughput (bit/s) for RSA and ECC

Number of nodes (n)	RSA throughput(bit/s)	ECC throughput(bit/s)
10	350	370
15	390	430
20	445	550
25	480	580
30	530	650
35	540	680
40	542	690
45	590	750
50	600	770

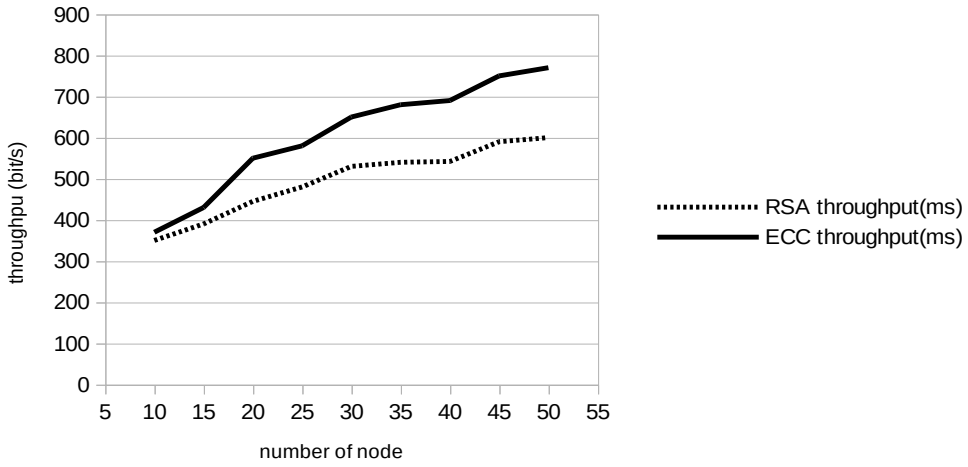


Figure.4.5: Throughput Vs Number of nodes

The average delay time curve of ECC decreases exponentially than RSA because of the sub exponential behavior of computational complexity of RSA.

4.1.4 Average Delay Time Analysis:

Delay time is another important factor for optimizing a system. If delay time increases of a system that system should not be declared as optimized one in terms of delay. The delay time of the simulation fluctuates because of random positioning and movement of the nodes. The average delay time should provide a profound knowledge of time optimization of a system. From the Figure 4.6: the simulation analysis of ECC and RSA with equivalent security strength is visualized. When the number of nodes is lower, for example 10, ECC improves the average delay time to 7% over RSA with equivalent security level. But dramatically ECC increases average delay time to 37% when number of nodes increased to.

Table 4-5 delay for RSA and ECC

Number of node (n)	RSA delay(ms)	ECC delay(ms)
10	260	240
15	238	230
20	218	180
25	190	159
30	170	125
35	165	110
40	160	100
45	153	90
50	150	50

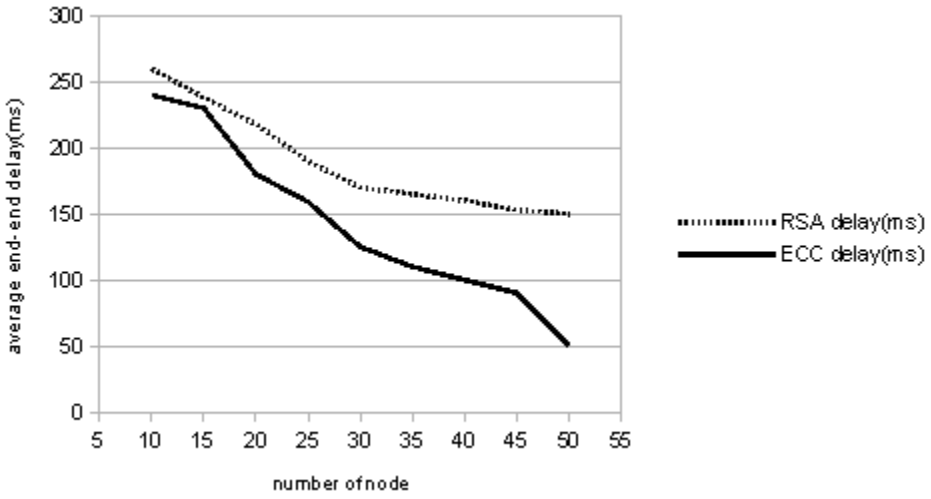


Figure.4.6: average end-end delay Vs Number of nodes

4.1.5 Average Jitter Analysis:

Jitter is a vital factor for signal propagation. If jitter is reduced then, quality of signal increases. So, for good-put jitter should be reduced. The jitter of the simulation fluctuates because of random positioning and movement of the nodes. The average jitter should provide a profound knowledge of good-put of the system. From the Figure 4.7 the simulation analysis of ECC and RSA with equivalent security strength is visualized. When the number of nodes is lower, for example 10, ECC improves the average jitter to 6% over RSA with equivalent security level. But dramatically ECC increases average throughput abruptly to 89% when number of nodes increased to 50. The average delay time curve of ECC decreases exponentially.

Table 4.6 Average Jitter for ECC and RSA

Number of nodes(n)	RSA jitter(ms)	ECC jitter(ms)
10	37	35
15	36	33
20	34	30
25	33	27
30	32	23
35	23	13
40	17	8
45	12	3
50	7	1

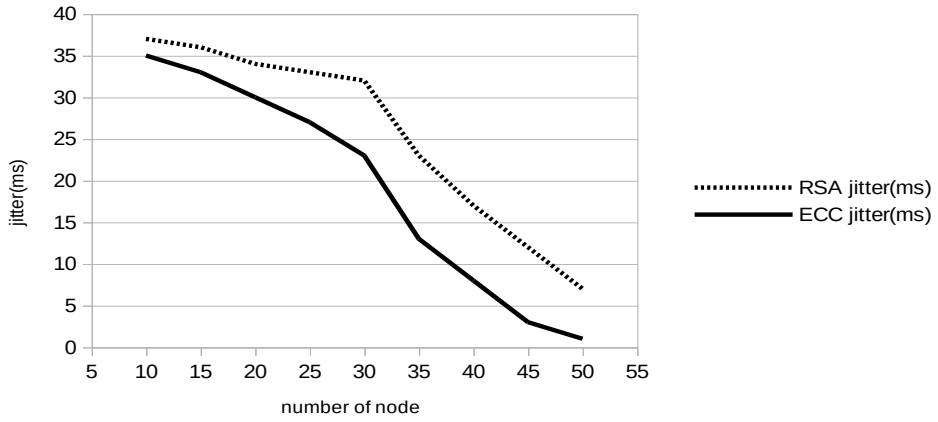


Figure.4.8 jitter Vs Number of nodes

4.1.6 Routing Protocol

To implement the existing and proposed algorithm, we preferred to use the Routing Algorithm AODV and DSR. This thesis measure this following parameters:-

4.3.6 a:Paket Delivery Fraction for AODV and DSR:

From the Figure 4.8 a the simulation analysis of AODV and DSR are compared with number of nodes for example 10, AODV PDF less than the DSR. when number of nodes increased to 50 The PDF curve of AODV Increases exponentially and node equal 100 PDF time curve decrease of DSR.

Table 4.7 packet delivery fraction for AODV and DSR

Number of nodes	PDF(AODV)	PDF(DSR)
10	78.5	81.7
15	74.7	81.7
20	74.8	85.9
25	77	61.5
30	75.7	64.7
35	83.3	48.5
40	70.8	80.9
50	82.7	69.7
75	82.2	27.3
100	76.7	82.1

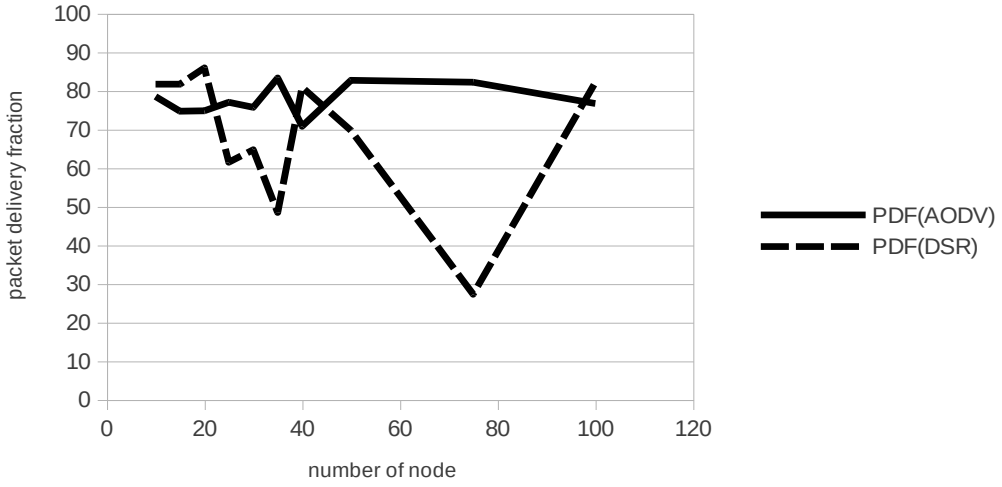


Figure 4.8 :packet delivery fraction for AODV and DSR

4.1.7: Normalized routing load for AODV and DSR:

From the Figure 4.9 the simulation analysis of AODV and DSR are compared with number of nodes for node= 10, AODV Normalized Routing Load(NRL) great than the DSR. when number of nodes increased to 50 The NRL time curve of DSR Increases exponentially and node equal 100 NRL time curve decrease of DSR.

Table 4.8 Normalized routing load for AODV and DSR

Number of nodes	Normalized routing load(AODV)(ms)	Normalized routing load(DSR)
10	5	2
15	8	2
20	14	31.6
25	16.3	87.5
30	25.1	0
35	18.3	115.2
40	24.2	4.4
50	27.9	4818.2
75	35.5	253833.3
100	70	31.9

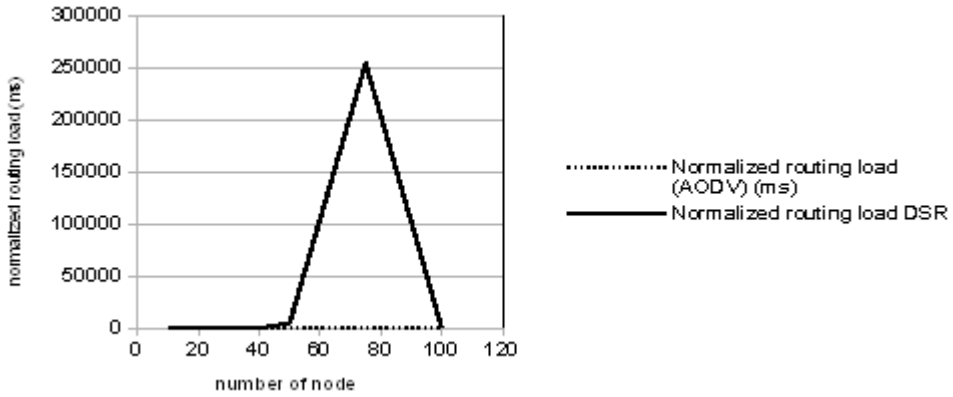


Figure 4.9: Normalized Routing Load for AODV and DSR

4.1.8 Average end to end delay for AODV and DSR:

From the Figure 4.10 the simulation analysis of AODV and DSR are compared with number of nodes for node equal 10, AODV delay less than the DSR. when number of nodes increased to 50. The average delay time curve of ECC decreases exponentially.

Table 4.9 Average end to end delay for AODV and DSR

Number of nodes	Average end to end delay(AODV)(ms)	Average end to end delay(DSR)(ms)
10	0.03	0.02
15	0.03	0.03
20	0.03	0.033
25	0.029	8.13
30	0.035	23.7
35	0.027	7.9
40	0.037	0.024
50	0.025	23.7
75	0.03	43.4
100	0.026	0.027

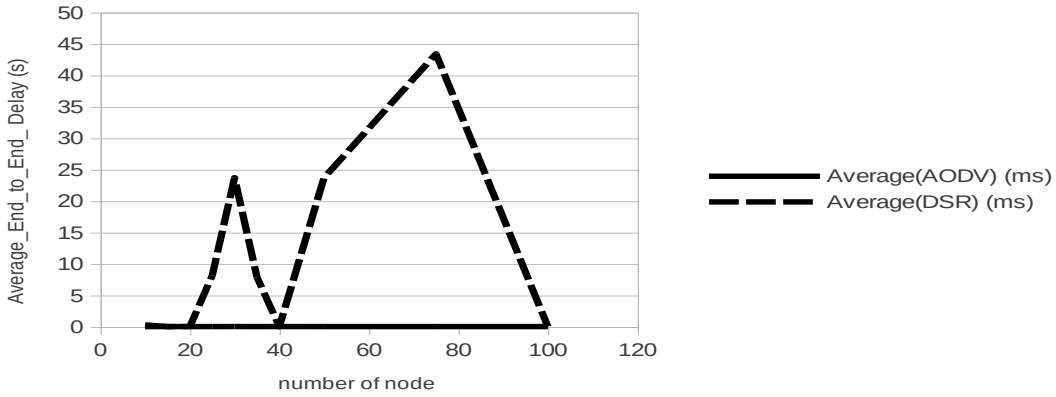


Figure 4.10 :Average end to end delay for AODV and DSR

Chapter 5

Conclusion and recommendation

5-1 Conclusion:

Security in the ad hoc network seemed to be the most interesting and complex yet the most researched area. The idea of this research was to achieve secure communication between any two nodes in the network while minimizing the hardware usage and wastage. Elliptic curve fits in perfectly with the requirement by providing smaller key.

To evaluate the performance of ECC, a comparison with a widely RSA security scheme was done .When RSA and ECC are compared, the overall performance of ECC was found to be better and it requires a fairly less key size to attain the similar security level as RSA. The routing protocols AODV and DSR are used with both ECC and RSA encryption schemes under variable network size. The experimentation used five evaluation parameters: Delay, NRL, PDF, Average hop count and Routing overhead.

ECC scheme in general had a better performance in delay, routing load, packet delivery fraction and average hop count than RSA. For equivalent number of bits, ECC performed better than RSA due to smaller key lengths. ECC with AODV gave the best performance for NRL, average hop count. ECC with AODV gave the best performance for delay and PDF. ECC with AODV gave an average performance in all cases. Secure AODV with ECC showed a comparable performance when compared with AODV with RSA. Since increase in key sizes increases the computational overhead slowly for ECC, it will be an attractive option in future. However, researchers are still experimenting to establish the suitability of ECC to different applications and to validate the efficient performance of ECC.

5.2 Recommendation:

For studying and monitoring performance of the elliptic curve security scheme, it was tested and compared to existing security scheme. Also for simplicity, the security level of the encryption scheme is set to the current acceptable security level. In future, experiments could be performed to test whether this scheme holds up against the attacks and then may be provide an estimate of the years required to break this security level.

Another factor that was cut off to simplify implementation is the validity of keys and re-issuing of keys. This may also be implemented and tested in future. Once all the above features have been added, the scheme could be ported to a handheld device and then tested in the real environment. It would be interesting to see the performance of the scheme under different conditions. Several factors such as the movement and speed of hardware devices, attackers, weather conditions, area of nodes, etc can affect the performance of the scheme. Also the increase in the scale of the network and interruption in the signals can affect scheme's performance.

Due to the vast structure of ECC, several other schemes can also be suggested which provide one or more security goals. These schemes were not implemented due to limited time. However, development and experiments of these schemes could be done in near future.

References :

[1] TruptiAgrawal and Swati Tiwari ," Performance Evaluation of FBU-NDA Based IDS Using AODV in MANET ",International Journal of Engineering Trends and Technology (IJETT) – Volume 16 Number 4 – Oct 2014 .

[2] Kamanshis Biswas and Md. Liakat Ali , "Security Threats in Mobile Ad Hoc Networks". 2007.page(10).

[3] Abdulameer K. Hussain ,"A Modified RSA Algorithm for Security Enhancement and Redundant Messages Elimination Using K-Nearest Neighbor Algorithm ",IJISET - International Journal of Innovative Science, Engineering & Technology, Vol. 2 Issue 1, January 2015.]

[4] Kamanshis Biswas and Md. Liakat Ali , "Security Threats in Mobile Ad Hoc Networks". 2007.page(10).

[5] Mr. Sagar Damodharji Padiya ," A System for MANET to detect Selfish Nodes using NS2 ",2012]

[6] S.M. Rifat Ahsan and Mohammad Saiful Islam ," Tunable Parameters for IEEE 802.11 based Ad-Hoc Network ",October 2009,BANGLADESH UNIVERSITY OF ENGINEERING & TECHNOLOGY[page 1].

[7] Karthik Sadasivam, B.S;" PERFORMANCE AND SECURITY IN MOBILE AD HOC NETWORKS".may-2005.pages(13-14).

[8] Karthik Sadasivam, B.S;" PERFORMANCE AND SECURITY IN MOBILE AD HOC NETWORKS".may-2005.pages(13-14).

[9] L. Zhou, Z.J. Haas, Cornell Univ., "Securing ad hoc networks," IEEE Network, Nov/Dec 1999, Volume: 13, Page(s): 24-30, ISSN: 0890-8044.

[10] Bahrouz A. Forouzan . Cryptography and network security,2007 .page[29]]

[11] L. Zhou, Z.J. Haas, Cornell Univ., "Securing ad hoc networks," IEEE Network, Nov/Dec 1999, Volume: 13, Page(s): 24-30, ISSN: 0890-8044.

[12] Kamanshis Biswas and Md. Liakat Ali , Security Threats in Mobile Ad Hoc Networks-22-march 2007]

[13] K. Sanzgiri, B. Dahill, B.N. Levine, C. Shields, E.M. Belding-Royer, "Secure routing protocol for ad hoc networks," In Proc. of 10th IEEE International Conference on Network Protocols, Dept. of Comput. Sci., California Univ., Santa Barbara, CA, USA. 12-15 Nov. 2002, Page(s): 78- 87, ISSN: 1092-1648.

[14] Tirthraj Rai, "SECURITY IN MOBILE AD HOC NETWORKS ",MAY 2009 ,THAPAR UNIVERSITY, page(17).

[15] Paul David Kiddie ," Decentralised Soft-Security in Distributed Systems ",February 2011.The University of Birmingham ,pages[6-8]

[16] G. Jose Moses,D. Sunil Kumar, "A Simulation Based Study of AODV, DSR, DSDV Routing Protocols in MANET Using NS-2 ",International Journal of Advanced Research in Computer Science and Software Engineering,Volume 2, Issue 3, March 2012.

[17] D. Deepthi Veronica1, D.B.Jagannadha Rao2,"Performance Analysis of AODV and DSR in MANETS Using NS2 Simulation ",International Journa .l of Innovative Research in Computer and Communication Engineering Vol. 1, Issue 4, June 2013.

[18] Stefaan SEYS ,"Cryptographic Algorithms and Protocols for Security and Privacy in Wireless Ad Hoc Networks",may 2006,KATHOLIEKE UNIVERSITEIT LEUVEN.

[19] Ali Hilal Al-Bayatti," Security Management for Mobile Ad hoc Network of Networks (MANoN) ",De Montfort University, February 2009. pages[29-40].

[20] Namita Singh , "SECURE COMMUNICATION USING ELLIPTIC CURVE CRYPTOSYSTEM IN AD HOC NETWORK ",University of Ottawa,2008

[21] Bahrouz A. Forouzan ." Cryptography and network security",2007 .page[29].

[22] Narmadha.M,and Nisha.A, "A Novel Method for Secured Data Communication in Wireless Network ",International Journal of Innovative Research in Computer and Communication Engineering ,Vol. 3, Special Issue 2, March 2015]

APPENDEX A: main program Code

```
#include<iostream>
#include<cmath>
#include<cstdlib>
#include<cstring>
#include<conio.h>
using namespace std;
void ec_points(int a, int b, int p)
{
  cout << "Points of Elliptic Curve" << endl;
   cout << "-----------------------------------------------------" << endl;
       for (int x = 0; x < p; x++) {
      for (int y = 0; y < p; y++) {
         int k = y * y;
         int m = (x * x * x) + a * x + b;
         if (k % p == m % p) {
            cout << "(" << x << "," << y << ")" << endl;
         }
      }
       }
}
static int EGCD(int a, int b, int& u, int &v)//Extended GCD gives g = a*u + b*v
{
     u = 1;    v = 0;    int g = a;  int u1 = 0;   int v1 = 1;  int g1 = b;
      while (g1 != 0)
      {
         int q = g/g1; // Integer divide    int t1 = u - q*u1;  int t2 = v - q*v1;
```

```
        int t3 = g - q*g1;  u = u1; v = v1; g = g1;  u1 = t1; v1 = t2; g1 = t3;
    }
    return g;
}
  //exitit 2
static int InvMod(int x, int n) // Solve linear congruence equation x * z == 1 (mod n)for
z
{
    //n = Abs(n);
    x = x % n; // % is the remainder function, 0 <= x % n < |n|
    int u,v,g,z;        g = EGCD(x, n, u,v);
    if (g != 1)
    {         z = 0;
    }
    else
    {           z = u % n;
    }
    return z;
}
int NegMod (int a, int p)
{
    int b = a * -1;  int n = ceil((float)b / p);   return (n * p) - b;
}
void add_points (int xp, int yp, int xq, int yq, int &xr, int &yr, int p)
{
   int s; int n = yp - yq; int d = xp - xq;
 if (d < 0)  {
```

```
        n *= -1;    d *= -1;
    }
    int x = InvMod(d, p);
    if (n * x > 0) {
        s = (n * x) % p;
    }
    else {
        s = NegMod(n * x, p);
    }
    int xr_ = (s * s - xp - xq);
    if (xr_ < 0)
        xr = NegMod (xr_, p);
    else
        xr = xr_ % p;
    int yr_ = (-yp + s * (xp - xr));
    if (yr_ < 0)
        yr = NegMod(yr_, p);
    else
        yr = yr_ % p;
}
void add_double (int xp, int yp, int &xr, int &yr, int a, int p)
{
    int s;    int n = 3 * xp * xp + a;   int d = 2 * yp;
    if (d < 0)  {
        n *= -1;      d *= -1;
    }
    int x = InvMod(d, p);
```

```
        if (n * x > 0) {
            s = (n * x) % p;
        }
        else {
            s = NegMod(n * x, p);
        }
        int xr_ = (s * s - 2 * xp);
        if (xr_ < 0)
            xr = NegMod (xr_, p);
        else
            xr = xr_ % p;
        int yr_ = (-yp + s * (xp - xr));
        if (yr_ < 0)
            yr = NegMod(yr_, p);
        else
            yr = yr_ % p;
}

void scalar_multiplicaiton (int xp, int yp, int k, int a, int p, int &PUx, int &PUy)
{
    if (k == 2) {
        add_double(xp, yp, PUx, PUy, a, p);
    }
    else if (k > 2) {
        add_double(xp, yp, PUx, PUy, a, p);
        for (int i = 0; i < k - 2; i++) {
            int xq = PUx;           int yq = PUy;   PUx = PUy = 0;
```

```
            add_points(xp, yp, xq, yq, PUx, PUy, p);
        }
    }
    else {
        cout << "Wrong key" << endl;
    }
}
void key_generation (int Px, int Py, int k, int a, int p, int &PUx, int &PUy)
{
    scalar_multiplicaiton(Px, Py, k, a, p, PUx, PUy);
    return;
}
void encryption (int Mx, int My, int k, int a, int p, int PUx, int PUy, int &Cx, int &Cy)
{
   int xr, yr;
    scalar_multiplicaiton(PUx, PUy, k, a, p, xr, yr);
    add_points(Mx, My, xr, yr, Cx, Cy, p);
}
void decryption (int Cx, int Cy, int k, int a, int p, int x1, int y1, int &Mx, int &My)
{
   int xr, yr;
    scalar_multiplicaiton(x1, y1, k, a, p, xr, yr);
    add_points(Cx, Cy, xr, -yr, Mx, My, p);
}
void Ecc(){

int a, b, p;
```

```
cout << "put a prime number: ";
cin >> p;
bool check;
do {
   check = false;
   cout << "put a value for a: ";
   cin >> a;  cout << "put a value for b: ";
   cin >> b;
   if (((4 * a * a * a + 27 * b * b) % p) == 0) {
      cout << "Your values do not satisfied the condition" << endl;
      cout << "Please put values again" << endl;
      check = true;
   }
} while (check);
cout << "------------------------------------------------------" << endl;
ec_points(a, b, p);
int Px, Py, PUAx, PUAy, PUBx, PUBy, Mx, My, Cx, Cy, m, n;
cout << "--------------------------" << endl;
cout << "Key " << endl;
cout << "--------------------------" << endl;
cout << "Select a base point (x,y) from the curve: ";
cin >> Px >> Py;
cout << "Select a private key for Abu baker: ";
cin >> m;
key_generation(Px, Py, m, a, p, PUAx, PUAy);
cout << "Public key of Abu baker is (" << PUAx << "," << PUAy << ")" << endl;
cout << "Select a private key for Omer: ";
```

```
    cin >> n;
    key_generation(Px, Py, n, a, p, PUBx, PUBy);
    cout << "Public key of 0mer is (" << PUBx << "," << PUBy << ")" << endl;
    cout << "--------------------------------------------------" << endl;
    cout << "Encryption/Decryption" << endl;
    cout << "--------------------------------------------------" << endl;
    cout << "Select a Message point (x,y) from the curve (for encryption): ";
    cin >> Mx >> My;
    encryption(Mx, My, m, a, p, PUBx, PUBy, Cx, Cy);
    cout << "Cipher is (" << Cx << "," << Cy << ")" << endl;
    int x1, y1;
    scalar_multiplicaiton(Px, Py, m, a, p, x1, y1);
    cout << "Abu baker send message pair(("<< x1 << "," << y1 << "),(" << Cx << ","
<< Cy << "))" << endl;
    cout << "||--------------------------------------||" << endl;
    cout << "Omer receive the message and start decrypting" << endl;
    decryption(Cx, Cy, n, a, p, x1, y1, Mx, My);
    cout << "Decrypted message is (" << Mx << "," << My << ")" << endl;
}
bool CheckIsPrime(long int num)
{
    if(num < 2) return false;
    long int i = 2;
    while(i <= num / 2)
    {
        if(!(num % i)) return false;
        i++;
```

```
    }
    return true;
}
long int Multiply(long int num1,long int num2)
{
   return num1 * num2;
}
bool CheckCoPrime (long int num1, long int num2) {
    long int lowest;
    if (num1 > num2) lowest = num2;
    else lowest = num1;
    long int i = 2;
    bool coprime = true;
    while (i < lowest) {
        if (!(num1 % i) && !(num2 % i)) coprime = false;
        i++;
    }
    return coprime;
}

long int FindE(long int phi_n)
{
    long int e = 0;
    do {
        cout << "Choose an integer number e (e must be co prime of phi_n): ";
        cin >> e;
    } while (!CheckCoPrime(phi_n, e));
```

```
    return e;
}
long int FindD(long int phi_n, long int e)
{
  int a = phi_n, b = e;     long int x = 0, y = 1, u = 1, v = 0, m, n, q, r;
  long int gcd = b;
  while (a != 0) {
    q = gcd / a;    r = gcd % a;   m = x - u * q;    n = y - v * q;  gcd = a;
    a = r;    x = u;    y = v;    u = m;      v = n;
  }
  if (y < 1) {
    y = phi_n + y;
  }
  return y;
}
long int Encrypt_Decrypt(long int t, long int e, long int n)
{
   long int rem;long int x = 1;
 while (e != 0) {
       rem = e % 2;     e= e/2;
      if (rem == 1) x = (x * t) % n;
       t = (t * t) % n;
   }
   return x;
}
void EncDecStr (long int e, long int n)
{
```

```
    char *str = new char[1000];  char *str1 = new char[1000];
    cout << "\nEnter a string: ";
    cin >> str;
    cout << "Encrypting using Public Key: " << endl;
    int i = 0;
    while (i != strlen(str)) {
   str1[i] = Encrypt_Decrypt(str[i], e, n);
         i++;
    }
    cout << str1 << endl;
}
void EncDecNum (long int n1, long int n2)
{
    long int pn;
    cout << "\nEnter an integer number: ";
    cin >> pn;
    cout << Encrypt_Decrypt(pn, n1, n2) << endl;
}
void generate_key (long int &n, long int &e, long int &d)
{
    long int p, q, phi_n, pt, ct;
    do {
         cout << "Enter a prime number: ";
         cin >> p;
    } while (!CheckIsPrime(p));
    do {
         cout << "Enter another prime number: ";
```

```
        cin >> q;
    } while (!CheckIsPrime(q));

    n = Multiply(p,q);
    cout << "n is " << n << endl;
    phi_n = Multiply (p-1,q-1);
    cout << "phi_n is " << phi_n << endl;
    e = FindE(phi_n);
    cout << "e is " << e << endl;
    if (!e) {
        cout << "Choose two suitable prime number" << endl;
        exit(1);
    }
    d = FindD(phi_n, e);
    cout << "d is " << d << endl;
}
void throuput(){
    cout<<"Throuput Rsa ,Ecc to security Mobile Ad Hoc "<<endl;
cout<<"Number Of Node(n) | RSA   | ECC  "<<endl;
cout<<"------------------------------------------- "<<endl;
int x,y;
for(int n=10;n<=50;n+=5){
    x=323.48+5.28*n;    y=318.56+9.57*n;
cout<<n<<"              "<<"|"<<floor(x)<<"    | "<<floor(y)<<endl;
}
}
void delay(){
```

```
   cout<<"Delay Rsa ,Ecc to security Mobile Ad Hoc "<<endl;
cout<<"Number Of Node(n) | RSA   | ECC  "<<endl;
cout<<"------------------------------------------- "<<endl;
int x,y;
for(int n=10;n<=50;n+=5){
   x=273.60-2.78*n;    y=281.56-4.63*n;
cout<<n<<"               "<<"|"<<floor(x)<<"    | "<<floor(y)<<endl;
}
}
void jitter(){
   cout<<"Jitter Rsa ,Ecc to security Mobile Ad Hoc "<<endl;

cout<<"Number Of Node(n) | RSA   | ECC  "<<endl;
cout<<"------------------------------------------- "<<endl;
int x,y;
for(int n=10;n<=50;n+=5){
   x=47.62-0.94*n;    y=49.62-0.78*n;
cout<<n<<"               "<<"|"<<floor(x)<<"    | "<<floor(y)<<endl;
}
}
void rsa(){
 cout << endl << endl << "##IMPLEMENTATION OF R.S.A ALGORITHM USING
C++## to secure message in Mobile Adhoc" << endl << endl;
   cout << endl << endl << "##Design By MOhammed Adam Aldod" << endl << endl;
   long int n, d = 0, e;
   generate_key(n, e, d);
   cout << "Public Key : ("<<e<<","<<n<<")" << endl;
```

```
cout << "Private Key : ("<<d<<","<<n<<")" << endl;
cout << endl << "Press 1: for encrypting numbers & 2: for encrypting string: ";
int choice;
cin >> choice;
switch (choice) {
    case 1:
        EncDecNum(e, n);
        break;
    case 2:
        EncDecStr(e, n);
        break;
    default:
        cout << "Wrong choice. Try again." << endl;
        exit(1);
}
cout << endl << "Press 1: for decrypting numbers & 2: for decrypting string: ";
cin >> choice;
switch (choice) {
    case 1:
        EncDecNum(d, n);
        break;


    case 2:
        EncDecStr(d, n);
        break;
    default:
```

```
            cout << "Wrong choice. Try again." << endl;
            exit(1);
    }
}
void main_enc(){
int ch;
cout << endl << "==================== ";
cout << endl << "Press 1: for decrypting by RSA ";
cout << endl << "Press 2: for decrypting by ECC ";
cout << endl << "Press any key to exit ";
cout << endl << "==================== "<<endl;
    cin >> ch;
    switch (ch) {
        case 1:
            rsa();          break;
        case 2:
            Ecc();          break;
        default:
            cout << "Wrong choice. Try again." << endl;
            exit(1);
    }
}
void main_per(){
int ch;
cout << endl << "==================== ";
cout << endl << "Press 1: Throuput Rsa ,Ecc to security Mobile Ad Hoc ";
cout << endl << "Press 2: Delay Rsa ,Ecc to security Mobile Ad Hoc ";
```

```
cout << endl << "Press 3: Jitter Rsa ,Ecc to security Mobile Ad Hoc ";
cout << endl << "Press any key: exit ";
cout << endl << "==================== "<<endl;
   cin >> ch;
   switch (ch) {
        case 1:
          throuput();    break;
        case 2:
          delay();        break;
   case 3:
           jitter();    break;
        default:
           cout << "Wrong choice. Try again." << endl;
           exit(1);
   }
}
int main() {
int ch;
cout << endl << "==================== ";
cout << endl << "Press 1: Secure Massage ";
cout << endl << "Press 2: Performance .... ";
cout << endl << "Press 3: exit ";
cout << endl << "==================== "<<endl;
   cin >> ch;
   switch (ch) {
        case 1:
main_enc();     break;
```

```
        case 2:
            main_per();
            break;
        default:
            cout << "Wrong choice. Try again." << endl;
            exit(1);
    }   getch();
   return 0;
}
```

//

Appendex B: #wrls-aodv.tcl

```
#wrls-aodv.tcl
# A 100 nodes for ad-hoc simulation with AODV and DSR
# Define options
set val(chan)          Channel/WirelessChannel    ;# channel type
set val(prop)          Propagation/TwoRayGround   ;# radio-propagation model
set val(netif)         Phy/WirelessPhy            ;# network interface type
set val(mac)           Mac/802_11                 ;# MAC type
set val(ifq)           Queue/DropTail/PriQueue    ;# 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)            15                         ;# number of mobilenodes
set val(rp)            AODV                       ;# routing protocol    AODV     or DSR
adhocRouting---- rp
set val(x)             500                        ;# X dimension of topography
set val(y)             400                        ;# Y dimension of topography
```

```
set val(stop)          100                     ;# time of simulation end
set ns              [new Simulator]
set tracefd         [open tracef.tr w]
set windowVsTime2 [open win.tr w]
set namtrace        [open simwrls.nam w]
$ns trace-all $tracefd
$ns namtrace-all-wireless $namtrace $val(x) $val(y)
# set up topography object
set topo        [new Topography]
$topo load_flatgrid $val(x) $val(y)
create-god $val(nn)
#  Create nn mobilenodes [$val(nn)] and attach them to the channel.
# configure the nodes
        $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 OFF \-movementTrace ON

        for {set i 0} {$i < $val(nn) } { incr i } {

                set node_($i) [$ns node]
        }
```

```
# Provide initial location of mobilenodes
$node_(0) set X_ 5.0  $node_(0) set Y_ 5.0  $node_(0) set Z_ 0.0
$node_(1) set X_ 490.0  B$node_(1) set Y_ 285.0 $node_(1) set Z_ 0.0
$node_(2) set X_ 150.0  $node_(2) set Y_ 240.0 $node_(2) set Z_ 0.0
$node_(3) set X_ 250.0  $node_(3) set Y_ 240.0  $node_(3) set Z_ 0.0
$node_(4) set X_ 100.0  $node_(4) set Y_ 70.0  $node_(4) set Z_ 0.0
$node_(5) set X_ 170.0  $node_(5) set Y_ 70.0 $node_(5) set Z_ 0.0
# Generation of movements
$ns at 10.0 "$node_(0) setdest 250.0 250.0 10.0"
$ns at 15.0 "$node_(1) setdest 245.0 285.0 10.0"
$ns at 20.0 "$node_(0) setdest 480.0 300.0 10.0"
$ns at 25.0 "$node_(3) setdest 280.0 30.0 10.0"
$ns at 30.0 "$node_(4) setdest 287.0 30.0 10.0"
$ns at 35.0 "$node_(5) setdest 189.0 30.0 10.0"
# Set a TCP connection between node_(0) and node_(1)
set tcp [new Agent/TCP/Newreno]
$tcp set class_ 2
set sink [new Agent/TCPSink]
$ns attach-agent $node_(0) $tcp
$ns attach-agent $node_(1) $sink
$ns connect $tcp $sink
set ftp [new Application/FTP]
$ftp attach-agent $tcp
$ns at 10.0 "$ftp start"
# Printing the window size
proc plotWindow {tcpSource file} {
global ns
```

```
set time 0.01
set now [$ns now]
set cwnd [$tcpSource set cwnd_]
puts $file "$now $cwnd"
$ns at [expr $now+$time] "plotWindow $tcpSource $file" }
$ns at 10.1 "plotWindow $tcp $windowVsTime2"
# Define node initial position in nam
for {set i 0} {$i < $val(nn)} { incr i } {
# 30 defines the node size for nam
$ns initial_node_pos $node_($i) 30
}
# Telling nodes when the simulation ends
for {set i 0} {$i < $val(nn) } { incr i } {
   $ns at $val(stop) "$node_($i) reset";
}
# ending nam and the simulation
$ns at $val(stop) "$ns nam-end-wireless $val(stop)"

$ns at $val(stop) "stop"
$ns at 150.01 "puts \"end simulation\" ; $ns halt"

#======-----------------------------------------------------------------------------------------------
proc attach-expoo-traffic { node sink size burst idle rate } {
        #Get an instance of the simulator
        set ns [Simulator instance]
        #Create a UDP agent and attach it to the node
        set source [new Agent/UDP]
```

```
        $ns attach-agent $node $source
        #Create an Expoo traffic agent and set its configuration parameters
        set traffic [new Application/Traffic/Exponential]
        $traffic set packetSize_ $size
        $traffic set burst_time_ $burst
        $traffic set idle_time_ $idle
        $traffic set rate_ $rate
           # Attach traffic source to the traffic generator
      $traffic attach-agent $source
        #Connect the source and the sink
        $ns connect $source $sink
        return $traffic
}
set sink0 [new Agent/LossMonitor]
set sink1 [new Agent/LossMonitor]
set sink2 [new Agent/LossMonitor]
$ns attach-agent $node_(3) $sink0
$ns attach-agent $node_(3) $sink1
$ns attach-agent $node_(3) $sink2
set source0 [attach-expoo-traffic $node_(0) $sink0 200 2s 1s 100k]
set source1 [attach-expoo-traffic $node_(1) $sink1 200 2s 1s 200k]
set source2 [attach-expoo-traffic $node_(2) $sink2 200 2s 1s 300k]
#In this project we use Agent/LossMonitor objects as traffic sinks, since they store the amount of bytes received, which can be used to calculate the bandwidth.
#VIII.2. Recording Data in Output Files
#Now we have to open three output files. The following lines have to appear 'early' in the Tcl script.
```

```
set f0 [open out0.tr w]
set f1 [open out1.tr w]
set f2 [open out2.tr w]
#------------------------------------ proc finish  from file wlan.tcl--------------------------
#=============================================================
proc stop {} {

   global ns tracefd namtrace f0 f1 f2

#Close the output files
   $ns flush-trace
   close $tracefd
   close $namtrace
#-------------------------------------------------------------------------------
       close $f0
       close $f1
       close $f2
       #Call xgraph to display the results
      exec xgraph out0.tr  out1.tr -geometry 800x400 &
       #Execute nam on the trace file
       exec nam simwrls.nam &
   exit 0
}
#-----------------------------------------------------------------------------------------------
#Now we can write the procedure which actually writes the data to the output files.
proc record {} {
       global sink0 sink1 sink2 f0 f1 f2
```

```
        #Get an instance of the simulator
        set ns [Simulator instance]
        #Set the time after which the procedure should be called again
        set time 0.5
        #How many bytes have been received by the traffic sinks?
        set bw0 [$sink0 set bytes_]
        set bw1 [$sink1 set bytes_]
        set bw2 [$sink2 set bytes_]
        #Get the current time
        set now [$ns now]
        #Calculate the bandwidth (in MBit/s) and write it to the files
        puts $f0 "$now [expr $bw0/$time*8/1000000]"
        puts $f1 "$now [expr $bw1/$time*8/1000000]"
        puts $f2 "$now [expr $bw2/$time*8/1000000]"
        #Reset the bytes_ values on the traffic sinks
        $sink0 set bytes_ 0
        $sink1 set bytes_ 0
        $sink2 set bytes_ 0
        #Re-schedule the procedure
        $ns at [expr $now+$time] "record"
}
$ns at 0.0 "record"
$ns at 10.0 "$source0 start"
$ns at 10.0 "$source1 start"
$ns at 10.0 "$source2 start"
$ns at 50.0 "$source0 stop"
$ns at 50.0 "$source1 stop"
```

```
$ns at 50.0 "$source2 stop"
$ns at 60.0 "stop"
#Call the finish procedure after 5 seconds of simulation time
$ns run
/*************************************************************
Appendex  C:    # getratio.awk
****************************************
BEGIN {
      sendLine = 0;     recvLine = 0;  fowardLine = 0; sendline1 = 0;   fowardLine1 = 0;
}
$0 ~/^s.* AGT/ {         sendLine ++ ;
}
$0 ~/^r.* AGT/ {        recvLine ++ ;
}
$0 ~/^f.* RTR/ {       fowardLine ++ ;
}
$0 ~/^s.* AODV/ {        sendLine1 ++ ;
}
$0 ~/^f.* AODV/ {        fowardLine1 ++ ;
}
END {
 #printf "s:%d r:%d \n delivery rate ratio:%.4f \nf:%d \n mean hop:%.4f \n normalized
routing load:%.4f \n end-end delay:%.5f \n", sendLine, recvLine,
(recvLine/sendLine),fowardLine,(sendLine+fowardLine)/sendLine,(sendLine1+foward
Line1)/recvLine,( 0.260157/recvLine);cout<< " For LAR-2 Algorithm\t2\n";
        pdfraction=(recvLine/sendLine)*100;
        avg_end_to_end_delay=( 0.260157/recvLine) ;
```

```
        normal_routing_load= (sendLine1+fowardLine1)/recvLine;
        printf(" Total packet sends: %d\t \n", sendLine);
        printf(" Total packet receives: %d\t \n", recvLine);
        printf(" Packet delivery fraction: %.4f\t \n", pdfraction);
        printf(" Average End-to-End delay:%f s\t \n" , avg_end_to_end_delay);
        printf(" normalized routing load:%f s\t\n", normal_routing_load);
}
/*****************************************************************
Appendex   D ** manet.sh*****************************/
#! /bin/csh
ns wrls-aodv.tcl
rm -f sim.*
# that meanig   put choosing colum form file tracef.tr to  file sam.tr  packet deliver
ratio  information
exec awk -f getRatio.awk tracef.tr >sam.tr &
#-------------------------------------------------------------------------------
#exec awk -f measure-delay.awk tracef.tr >sam2.tr &
#exec awk -f parse.awk tracef.tr >sam1.tr &

APPENDEX  :**************measuredelay.awk**********************
BEGIN {
    highest_packet_id = 0;     n=0;     tm=0;
}
{
  action = $1;    time = $2;   flow_id = $4;    packet_id = $6;
  if ( packet_id > highest_packet_id )
            highest_packet_id = packet_id;
```

```
  if ( start_time[packet_id] == 0 )
            start_time[packet_id] = time;
   if ( flow_id == "AGT" && action != "d" ) {
     if ( action == "r" ) {
       end_time[packet_id] = time;
     }
   } else {
     end_time[packet_id] = -1;
   }
}
END {
   for ( packet_id = 0; packet_id <= highest_packet_id; packet_id++ ) {
      start = start_time[packet_id];       end = end_time[packet_id];
  packet_duration = end - start;
 printf("%f\n", packet_duration);
      if ( start < end ) {                    n += 1;
                  tm += packet_duration;
 #printf("%f\n", tm/n);
}
   }
     printf("%f\n", tm/n);
}
```