QoS Aware Adaptive Security Scheme for Video Streaming in MANETs

by

Tahsin Arafat Reza

A thesis submitted to
the Faculty of Graduate Studies and Research
in partial fulfillment of
the requirements for the degree of

Master of Computer Science

Ottawa-Carleton Institute for Computer Science
School of Computer Science
Carleton University
Ottawa, Ontario

April 12, 2012

© Copyright
April 2012, Tahsin Arafat Reza

Library and Archives Canada

Published Heritage Branch

395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque et Archives Canada

Direction du Patrimoine de l'édition

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence
ISBN: 978-0-494-91594-3

Our file Notre référence
ISBN: 978-0-494-91594-3

NOTICE:

The author has granted a non-exclusive license allowing Library and Archives Canada to reproduce, publish, archive, preserve, conserve, communicate to the public by telecommunication or on the Internet, loan, distrbute and sell theses worldwide, for commercial or non-commercial purposes, in microform, paper, electronic and/or any other formats.

The author retains copyright ownership and moral rights in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

AVIS:

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque et Archives Canada de reproduire, publier, archiver, sauvegarder, conserver, transmettre au public par télécommunication ou par l'Internet, prêter, distribuer et vendre des thèses partout dans le monde, à des fins commerciales ou autres, sur support microforme, papier, électronique et/ou autres formats.

L'auteur conserve la propriété du droit d'auteur et des droits moraux qui protege cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

In compliance with the Canadian Privacy Act some supporting forms may have been removed from this thesis.

While these forms may be included in the document page count, their removal does not represent any loss of content from the thesis.

Conformément à la loi canadienne sur la protection de la vie privée, quelques formulaires secondaires ont été enlevés de cette thèse.

Bien que ces formulaires aient inclus dans la pagination, il n'y aura aucun contenu manquant.

Canada

Abstract

QoS provisioning is challenging in a mobile wireless ad hoc network. Video streaming is delay sensitive. Cryptography algorithms offer confidentiality of shared data but are often costly in terms of computation. Delay overhead caused by cryptography may affect streaming performance. There have been work addressing this issue, but many of them suffer from various shortcomings and lack practicality.

Our study encompasses analysis of video streaming characteristics in ad hoc networks using several cryptography algorithms. We propose a protocol for securely streaming real-time video in ad hoc networks. The novelty of our proposal is that the protocol is adaptable to changes in network dynamics. The protocol can dynamically adapt security measure in accordance with available resources to maximize QoS. The protocol can also adapt the streaming characteristics to provide QoS while maintaining a desired level of security. We evaluate our proposal through implementation and analysis of simulation results.

Acknowledgement

First and foremost, I express my earnest gratitude to my thesis supervisor, Professor Michel Barbeau of School of Computer Science at Carleton University, for his sincere guidance and patience with me. Without his help, it would have been impossible for me to produce this work. A very special thank you to Professor Doron Nussbaum, Graduate Director of School of Computer Science at Carleton University, for his invaluable advice and help when I started my program. Thanks to Professor Carlisle Adams of School of Electrical Engineering and Computer Science at University of Ottawa, for answering cryptography related questions. Thanks to Dr. Chih-Heng Ke, Assistant Professor of Computer Science and Information Engineering, National Quemoy University, Taiwan, for answering questions about video traffic simulation in NS. Thanks to Dr. Jiazi YI of Polytechnic School of University of Nantes, France for answering questions about MP-OLSR. I offer my sincere appreciation to all the professors at Carleton University and University of Ottawa who contributed to my leaning experience during my time in this program.

Contents

List of Tables

List of Figures

List of Abbreviations and Symbols

AC	Admission Control
ACK	Acknowledgement
$AODV$	Ad hoc On-Demand Distance Vector
CA	Certification Authority
CBR	Constant Bit Rate
DCF	Distributed Coordination Function
DoS	Denial of Service
DRS	Dynamic Source Routing
$EDCA$	Enhanced DCF Channel Access
FPS	Frames Per Second
FTP	File Transfer Protocol
Gb	Gigabit
GHz	Gigahertz
GOP	Group Of Pictures
IFq	Interface Queue
IP	Internet Protocol
$KBps$	Kilobytes Per Second
$Kbps$	Kilobits Per Second
$OLSR$	Optimized Link State Routing
$OFDM$	Orthogonal Frequency-Division Multiplexing
m	Meter
MAC	Medium Access Control
$MANET$	Mobile Ad hoc Network
MB	Megabyte
Mb	Megabit
$MBps$	Megabytes Per Second
$Mbps$	Megabits Per Second
MOS	Mean Opinion Score
$MP-OLSR$	Multipath OLSR
MPR	Multi Point Relay
NAL	Network Abstraction Layer
$PSNR$	Peak Signal-to-Noise Ratio
$QaASs$	QoS aware Adaptive Security scheme
QoS	Quality of Service
$RREP$	Route Reply
$RREQ$	Route Request

RTP	Real-time Transport Protocol
s	Second
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
VBR	Variable Bit Rate
VCL	Video Coding Layer
WLAN	Wireless Local Area Network
ZRP	Zone Routing Protocol

Chapter 1

Introduction

Development of radio technologies began in the late 19^{th} century pioneered by world's renowned scientists like Guglielmo Marconi, Nikola Tesla, Jagdish Bose and Reginald Fessenden [54] [26]. Long before widespread implementation and commercialization of wireless communications systems, the immense future prospect and potential of wireless technologies had been realized. Since its inception, numerous scientists and engineers have been consistently contributing towards the development of wireless communications technologies. The results are today's satellite-based communications systems, our everyday voice and Internet-enabled smartphones, wireless home networks, wireless sensor networks for wildlife and environment monitoring, to name a few. Advances in wireless technologies have enabled remote communications in rural areas and adverse environments, where communications once thought to be impossible.

The earliest initiatives of supporting packet data over wireless medium began in the 1970s. ALOHAnet and PRNET are two examples of the earliest wireless data networks [54] [26]. The opportunity of sharing data in real time, remotely through mobile devices, has opened door to new possibilities. Especially in the last two decades, we have experienced remarkable advances in wireless technologies. Apparently, these advancements made direct impact on our everyday life. Many of these technologies did not take too long to penetrate in the commercial arena; thus their advancements have been rapid.

Mobile Ad hoc Network (MANET) is one of the younger wireless network technologies that has gained massive attention from the research com-

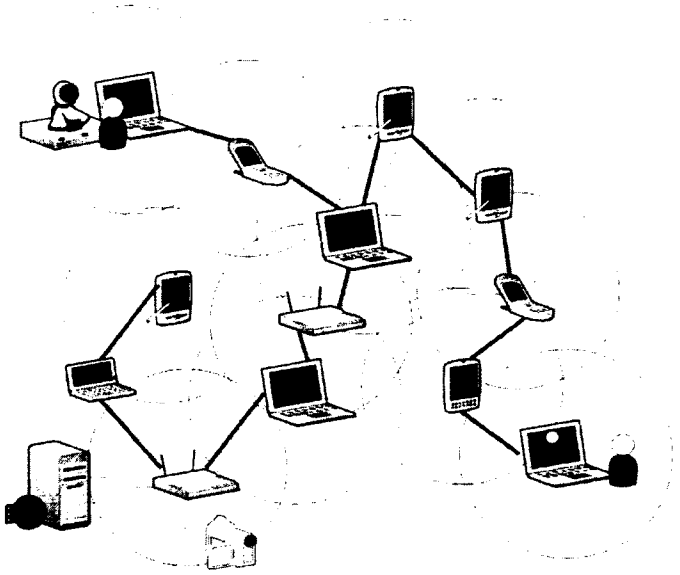


Figure 1.1: An ad hoc network capable of multimedia communications.

munity in recent years. A MANET is a communications network, where there is no fixed infrastructure or central authority; the member nodes are self-organized, nodes communicate with each other directly or through intermediate nodes, and nodes act both as hosts and routers. MANET nodes could be both stationary or mobile, thus no static topology is guaranteed [98]. Many predict that widespread adaptation of MANETs for various applications is only a matter of time. Figure 1.1 is an example of an ad hoc network capable of multimedia communications.

1.1 Motivation

Confidentiality is a must have requirement for distributing and sharing sensitive information [167] [139]. Confidentiality refers to protection against unauthorized disclosure of network information. One goal of a security-enabled computer network is to ensure confidentiality of data being shared over the network. Cryptography is a popular choice for providing security for digital contents. Different cryptographic mechanisms have been developed and standardized to provide data security. However, a cryptographic technique does not come free of additional computational costs. In fact

a cryptographic operation could be computationally pretty intensive. The computational cost of a cryptographic algorithm (encryption and decryption) depends on a number of properties such as the size of the cryptographic key, internal state, block size and number of rounds [4] [87] [17] [139]. A cryptographic algorithm could be strong in terms of security but costly in terms of computational cost.

Real-time video streaming is delay sensitive and also resource intensive. Video streaming over a network requires availability of a significant amount of bandwidth and demands QoS requirements such as delay and frame rate. Compared to wired connections, wireless links are vulnerable and data transmission over wireless medium is prone to errors. Achieving target QoS for video streaming is even more challenging in a relatively unpredictable MANET.

The broadcast nature of wireless transmission makes wireless communications vulnerable to security attacks like eavesdropping and spoofing [132]. Video applications involving confidential data, demand protection against security threats. A primary goal of a security application is to ensure protection against unauthorized access of data contents. Data encryption is one of the most effective methods for providing confidentiality at the content level [167]. [138] [4] [87] [103] [145] and [75] proposed several techniques to encrypt video contents. However, computational cost introduces additional overhead to already delay sensitive video streaming. In real-time video streaming, delay overhead due to encryption could directly influence playback quality. Our investigation shows that this is a relatively less explored problem. [87], [17] [62] and [148] address the issue of impact on streaming performance due to inclusion of cryptography. Based on the discussion of the addressed issue, our motivation is drawn and research goal is identified. In the next section, we present the problem statement.

1.2 Problem Statement

Our work addresses the issue of delay overhead caused by introduction of cryptography that directly affects video streaming performance. Cryptographic algorithms come in different strengths and computational costs. Streaming video, on the other hand, can achieve different levels of playback quality

depending on a number of constraints, such as frame rate, compression and resolution. Typically, a higher quality video contains more frames per second than a lower quality video at the same resolution. Real-time video streaming involves encoding, transmitting and decoding before video is played back. Each of these steps is significantly time-consuming. Cryptography introduces additional delay overhead involving encrypting and decrypting video data. Maintaining target video quality, while encryption is in place, may become challenging. Additionally, the dynamic nature of self-organized MANET makes QoS provisioning challenging. Signal fading, latency overhead and packet loss due to link brakeage are examples of common hurdles experienced by an ad hoc network [121]. These limitations of ad hoc networks seriously affects video streaming, which demands certain QoS requirements to achieve acceptable playback quality. Besides the mentioned obstacles, traffic congestion, external noise and malicious attack could also make it difficult for the streaming video to achieve target QoS.

It is difficult to generalize security and performance requirements that can be applied to any application scenario. Depending on the system requirements and state of the network, we might have to compensate one requirement, within the allowed scope, for the sake of the other. In general terms, the addressed issue can be perceived as a problem of ensuring security and QoS simultaneously. Our research goal is to pursue the discussed problem and propose a potential solution. In the next section, we hypothesize our proposal.

1.3 Hypothesis

An ad hoc network can be composed of a diverse range of devices with different computational capabilities. Throughput of a process depends on available computing resources. The performance of a computationally intensive cryptography process would vary depending on the resource availability.

Multimedia traffic is delay sensitive. A cryptography process may introduce additional, yet unavoidable delay overhead. If a multimedia traffic source knows the capability of a target device, e.g., throughput of a cryptography process, then it can infer appropriate cryptography parameters that would not cause a performance bottleneck. Furthermore, it may be possible

to adjust multimedia parameters to control the amount of traffic, thus the amount of data to be processed by a cryptography process. Traffic load influences latency as well as packet delivery ratio in a network. By adjusting multimedia parameters, it may be possible to control the overall delay as well as ensure QoS.

The receiver of the multimedia service can provide periodic feedback to the source with information such as transmission delay, delay jitter, frame rate and frame loss ratio. Hence, an adaptive mechanism that trade-off between cryptography parameters and multimedia parameters would be a feasible solution to the addressed problem.

1.4 Summary of Contributions

We propose QaASs, a QoS aware security mechanism for real-time multimedia communications over MANETs. In order to address the issue of delay overhead caused by cryptography operations, we have developed an adaptive mechanism that adapts cryptography and multimedia properties in order to provide multimedia service quality while maintaining a required level of security. The adaptation scheme is designed around a predefined delay threshold value. The mechanism defines why, when and how to deploy adaptation. We demonstrate the effectiveness of our proposal by presenting a number of service scenarios demanding different requirements. We present four adaptation options for different service requirements. We evaluate our proposal through a series of simulations. Simulation results are confirmed with 90% or 95% confidence level (Appendix A).

Adaptation option one adapts cryptography properties in order to reduce delay overhead caused by a cryptography process. We also employ periodic rekeying. We show, for ECC (Elliptic Curve Cryptography) with a 384-bit curve, that using adaptation option one, it is possible to gain 9 s (second) on transfer time. We also show that, while choosing a cryptography scheme with higher throughput and lower security may reduce transmission delay, periodic rekeying is a possible option to improve security with minimum effect on performance.

Adaptation option two utilizes selective encryption, where encrypting in-

ter coded frames [65] is optional. We show that using adaptation option two, for ECC with a 384-bit curve, it is possible to gain 12 s on transfer time and increase average packet rate by 10 packets per second.

Although reduces quality, adaptation option three improves video transfer time without compromising security. We have shown that using adaptation option three, it is possible for an encrypted video sequence to match the transfer time of the video without encryption.

Adaptation option four adjusts frame rate in order to defy cryptography delay overhead. For ECC with a 384-bit curve, in our simulation, we were able to gain 11 s on transfer time over the stream without adaptation.

1.5 Organization of the Thesis

The rest of the thesis is organized in four chapters. In Chapter 2, we present a literature review related to our problem of interest. Technical background required for the understanding of our work is detailed in Chapter 3. In Chapter 4, we present our proposal. The evaluation of our proposal is documented in Chapter 5. Chapter 6 concludes the thesis and outlines related future work.

Chapter 2

Multimedia Security in Ad hoc Network

In this chapter, we present a literature review of existing work related to multimedia security in ad hoc networks. We begin with outlining general security issues in ad hoc networks, security risks, attacks and solution proposals. Moving on, we focus on security concerns related to video streaming, challenges and solution techniques. We comment on each review outlining key points related to our motivation.

2.1 Security in Ad hoc Networks

2.1.1 Security Requirements

In a computer network, security measures are deployed as a protection against malicious attacks or intentional faults that disrupt regular network operations and gain unauthorized access to resources and information. Security requirements of an ad hoc network are not different from most other kinds of communications networks. The primary objectives of an effective ad hoc security architecture are the following [167] [139] [105] [116] [157] [94] [6]:

Availability: The goal of a security service is to ensure service availability despite security attacks that interrupt regular network operations. Availability refers to network's ability to detect and to provide protection against attacks and the ability to recover from the effect of an attack and restore regular network services.

Authentication: A security architecture should allow nodes to verify the genuineness of communications, both communicated information and the communicating members in the network.

Confidentiality: Protection against unauthorized disclosure of network information.

Integrity: Integrity ensures authenticity of shared network contents. A security service provides integrity by ensuring no alteration of data during transmission.

Non-repudiation: A node cannot deny transmission of an authenticated message that has been initiated and scheduled for transmission.

Authorization: Authorizing only the trusted nodes to gain access to network resources.

Trust management: Maintaining mutual trust among authorized members.

2.1.2 Security Risks and Issues

The physical construction and functional characteristics of an ad hoc network make it vulnerable and susceptible to malicious attacks. Absence of infrastructure, sole dependency on wireless links, dynamic topology, node mobility and multihop routing have been identified as the primary MANET features that make ad hoc networks vulnerable to malicious attacks [51]. Additionally, unlike routing protocols, MANETs lack security standards. Standardized security techniques for wired networks and WLANs are not applicable to ad hoc networks. For example, use of unique third party certification authority (CA) is against the core concept of infrastructureless networks. Use of a single CA for key management in ad hoc networks would be problematic, since availability and sustainability of a single station are impossible to guaranty. Use of redundant stations acting as CAs, on the other hand, imposes additional threat as an adversary may pretend to be a CA [51] and eventually infects the network.

2.1.3 Security Attacks in Ad hoc Networks

Security attacks in ad hoc networks can be classified based on the *origin* of the attacker and *nature* of the attack [51]. An attack can be classified as *internal* or *external* depending on the association of the attacker's origin. The nature of the attack could be either *active* or *passive* [51].

Active attacks are the most common in MANETs. Attacks can differ from their functional behaviour or objective. A good number of identified MANET attacks focus on infecting the routing protocols. A key challenge of ad hoc routing protocols is to cope with the dynamic nature of the network topology. Adversaries take advantage of this property to launch various attacks. An ad hoc routing protocol attack can be launched by modifying routing information, [129] [132] such as control message sequence number [114], hop-count and source-defined routes [51]. *Tunneling* is another example of routing attack where two remote malicious nodes project a falsified path as a legitimated link and use that path as a tunnel to forward packets [51]. *Spoofing* attacks occur when a malicious node forges its identity as a trusted node [132]. The *rushing* attack identified by Hu et al. [72], particularly in reactive routing protocols, is caused by malicious nodes that race to flood the network with route discovery packets, ignoring the required packet releasing wait time. *Wormholes* attacks [71] infect the routing protocol by tunnelling packets or bits from one point of the network to another point and sending replies. The attacker poses the route containing the wormhole to be the optimal route. For reactive routing protocols, this is done by tunnelling the route request packets directly to the destination. Malicious nodes comprising the tunnel could alter or discard packets to disrupt regular network operations or simply retrieve sensitive information for other adverse activities.

Eavesdroppers, malicious and misbehaving nodes may affect data forwarding by accessing confidential data contents of a packet or even dropping packets instead of forwarding. Additionally, malicious attacker may cause performance degradation, service interruption and denial of service (DoS) by altering routing information, traffic manipulation or launching flooding attacks [56].

A wireless network using the IEEE 802.11 MAC protocol, relies on mu-

tual trust among the neighbouring nodes for fair channel access [85] [61]. This approach leaves room for malicious eavesdropper to gain access to the wireless medium, which is otherwise impossible in a wired network without any physical connection with the network. This is an example of a passive attack. An eavesdropper can access network contents and retrieve valuable information which it can later use to launch other types of attacks.

2.1.4 Security Solutions for Ad hoc Networks

Many solutions have been proposed to provide security in ad hoc networks. We review a number of proposals that aim to secure ad hoc networks against different types of attacks and security threats.

Ariandne [73], proposed by Hu et al., is a secured routing protocol that extends Dynamic Source Routing (DSR) protocol [14]. The protocol provides routing message authentication. Routing message can be authenticated by appending message authentication code [90] or digital signature for authenticating intermediate nodes. For intermediate node authentication, the protocol uses the TESLA broadcast authentication technique [122]. For endpoint authentication, the protocol uses shared secret keys. Ariandne offers integrity and authentication. Also non-repudiation can be achieved through the use of digital signatures at the intermediate nodes for message forwarding. The reliance on network wide clock synchronization and delay requirements of control messages makes the protocol less flexible.

Papadimitratos and Haas proposed the Secured Routing Protocol (SRP) [114] that secures both DSR and ZRP (Zone Routing Protocol). SRP employs symmetric key cryptography and provides authenticated route discovery. The source and destination share a secret key. Message authentication code technique is used for the destination to authenticate the originator of a message. However, neither the intermediate nodes, routes nor the route error messages are authenticated, leaving room for malicious attacks. The protocol offers end-to-end authentication and integrity but lacks both non-repudiation and availability.

Secure Message Transmission (SMT) proposed by Papadimitratos and Hass [115] provides end-to-end security against malicious and selfish interruption and disruption of network services. SMT is adaptable to change in

network conditions. SMT utilizes disjoint multiple paths, called Active Path Set (APS) from the source to the destination. The source and destination maintain a trust relationship. The relationship is independent of the intermediate nodes. At the source node, a message is dispersed into multiple pieces based on Rabin's algorithm [125]. Dispersed message pieces carrying message authentication code [90] are transmitted over multiple paths in APS. At the destination node, a dispersed message is reconstructed only if a sufficient number of pieces is received. The destination securely provides acknowledgement of receipt to the source. The protocol maintains rating of the paths in APS by increasing or decreasing rating for successes and failures respectively. The strict disjoint path set may cause performance bottleneck in dynamic ad hoc networks.

SAR (Security-Aware Routing) proposed by Yi et al. [163] uses a metric called *trust value* for authentication and is based on hierarchical key sharing. The solution is not effective if no hierarchy exists in the network.

In order to prevent rushing attacks, Hu et al. [72] proposed a randomized RREQ (route request message in a reactive routing protocol) selection technique for RREQ forwarding instead of the first come, first served approach. The problems with this solution are the route discovery delay is increased due to longer wait time to receive enough RREQs and the discovered path is unlikely to be optimal.

Onion routing proposed by Awerbuch et al. [9] provides anonymity for source routing by employing asymmetric encryption technique that encrypts the source route in the data packets. The resulted encrypted route resembles the layered structure of an *onion* skin. Node n_x's public key P_{n_x} is used to encrypt the ${n_{x+1}}^{th}$ node's address. When node n_x receives a packet from n_{x-1}, it can only decrypt the route and retrieve the address of node n_{x+1} and forward the packet if necessary. The remaining of the route would be anonymous to n_x. This technique is a good solution against attacks that alters the original route but does so at the expense of high computational cost and delay due to performing decryption at each node in the routing path and thus, is not suitable for delay sensitive network services.

Marti et al. [102] proposed the *Watchdog* technique to detect misbehaving nodes that do not forward packets and malicious nodes that intention-

ally drop packets by monitoring neighbours in the promiscuous mode. The drawback of this technique is that it is prone to false positive and failure in detection [53].

Threshold cryptography proposed by Shamir [135], offers availability, confidentiality and secured sharing. Threshold cryptography follows a distributed approach. Primary components of threshold cryptography are key generation, encryption, share generation, verification and aggregation. Suppose, we want to securely send a message from a source to a target destination. There are n routes between the source and destination. Selected nodes on the n disjoint routes are called *shareholders*. Threshold cryptography redundantly split the message into n segments. The original message can be retrieved from at least t segments out of n. The n segments are sent over n different paths. The source distributes the shared key among the shareholders. The shareholders encrypt partial messages using the shared key and forward them to the destination. The destination node is required to receive at least t segments from the shareholders in order to combine the partial messages. The destination node must be able to decrypt the minimum number of partial messages determined by the threshold value before it can combine the partial messages to retrieve the original message. Threshold cryptography can be used in conjunction with other cryptographic schemes such as RSA, El Gamal and Diffie-Hellman [56]. One limitation of threshold cryptography is in practice it is not always possible to guaranty availability of n disjoint routes. This questions scalability of the technique.

In [158] Yang et al. proposed a unified network layer solution to provide both routing and data forwarding security. Yang et al.'s solution is based on threshold cryptography-based signature [135] and the watchdog technique [102] described earlier.

CORE [104] and CONFIDENT [28] [29] propose a *reputation*-based security technique against misbehaving and malicious nodes. Reputation is a logical measure of trust determined by observing a node's activity, well-behaving or misbehaving. A key weakness of the reputation-based approach is the possibility of false accusation.

Probing, a technique first used by Awerbuch et al. [9], based on end-to-end feedback, aims to identify a selfish or misbehaving node. The funda-

mental idea is to incorporate commands in data packets called *probes* which contain a list of nodes called probed nodes. A probed node must send an ACK for a received probed data packet. Using probing, it is possible to identify a faulty link in a routing path. Two major drawbacks of probing are increased network traffic, due to ACKs, and probed packets are identifiable by a malicious node that can circumvent probing by dropping probed packets. In [88], Kargl et al. offered a solution for the latter problem by making the probing information unidentifiable by malicious nodes.

Buttyan and Hubaux [30] proposed an *economy*-based approach to battle against misbehaving or selfish nodes. Nodes must pay virtual currency, what the authors called *nuglet*, to access a network service. Nodes are required to earn nuglets by providing service to other nodes thus selfish nodes can be excluded from participating in any network activity. In addition to being a complex mechanism, the main shortcoming of this technique is that a well-behaving node might be listed as a selfish node due to not participating enough in network activities because of its location.

Kyasanur and Vaidya [92] have proposed a solution for ad hoc MAC protocol security issues (as described in Section 2.1.3). Their proposal modifies IEEE 802.11 MAC protocol [1] *backoff* values. The receiver of a RTS (request to send) appends selected backoff values to CTS (clear to send) and ACK packets, and expects the originator of RTS to use these backoff values and observe the ideal slot behaviour of the target node. If the sender's transmission pattern does not comply with the expected behaviour, then the node is assumed to be misbehaving or malicious.

2.1.5 Cryptographic Key Management in Ad hoc Networks

Key exchange and key management are integral parts of any cryptographic system. Dependency on cryptography requires a security protocol to incorporate key exchange and key management mechanisms. The inherent limitations of MANETs, such as lack of central administration, make key management difficult. We discuss cryptography in details in Section 3.7. In this section, we discuss topics related to cryptographic key management. Depending on how the cryptographic key is shared, there are two main cryp-

tography infrastructures: *Private key infrastructure* and *Public key infrastructure.*

Symmetric cryptography is based on private key infrastructure where all the parties share one common secret key. In an infrastructure-based communications system, the secret key is distributed by a centralized trusted third party. Because of the mutual trust issue and absence of central management, a MANET demands the key distribution mechanisms that are decentralized and distributed. Group Diffie-Hellman (GDH) [140] proposed by Steiner et al. is an extension of the original Diffie-Hellman (D-H) [50] key agreement protocol, where n different parties establishes a common key after n rounds of message exchange. The main drawback of GDH is that using a unique *collector* node, responsible for collecting partial keys and combining them, could be problematic in MANETs. An hypercube-based approach described by Backer and Willie [15] is another variant of D-H for MANETs but suffers from node ordering overhead with increasing mobility. Password Authentication-based Key Exchange [16] establishes a strong key among two parties from a weak shared password. The key is immune to dictionary attacks. The key establishment process involves only the interested parties. The limitation of this approach is that guaranteeing a pre-shared password is not always possible in MANET. Additional hybrid approaches proposed in [7] and [97]. Asokan and Ginzboorg's [7] proposal incorporates password authentication to GDH and [97] follows a cluster based technique that combines a centralized and key agreement approach.

In addition to key establishment, *rekeying* is another important requirement for secured group communication. Revoking a shared key, in case group membership changes or in a timely fashion, to safeguard security measures, is the main goal of rekeying [168]. Several rekeying techniques are described in [153] [117] [168] [95] [89].

In the public key infrastructure, a public and a private key pair is maintained by every communicating party. The private key is secret to a node, while the public key is distributed among the other network nodes. In an infrastructure-based communications system, a trusted entity called a Certification Authority (CA) is responsible for distributing the public keys in certificates. The CA maintains its own private key and uses it to sign the certificates binding the public keys. A certificate recipient uses the public key

to authenticate any other certificate. Similar to the use of a unique trusted third party in the private key infrastructure, use of a single CA is problematic in MANETs [51]. Trust issues and lack of central management demand key distribution to be distributed in a MANET. Zhou and Hass [167] proposed a threshold cryptography [135] based technique for MANETs to establish trust among the nodes. The key management service consists of n selected nodes, called *servers*. All the servers are aware of each other's public key as well as public keys of all other member nodes. An adversary cannot compromise more than k nodes for a given duration, with $n \geq 3k + 1$. Any $k + 1$ servers out of n can jointly create a digital signature. The service private key is divided into n pieces. Each server is assigned one share. Each server generates a partial signature using its private key share and submits it to a chosen server called *combiner*. With at least $k + 1$ partial signatures, the combiner is able to compute a complete signature for a certificate. Instead of using a single combiner, using $k + 1$ servers as combiners offers enhanced security since at least one server is guaranteed to be the combiner. A combiner can verify the signature using the public key. Ostrovsky and Yung [113] proposed an enhancement to the Zhou and Hass's [167] technique that offers fault tolerance against mobile adversaries and share refreshing (rebuilding sharing of the private key). MOCA (Mobile Certified Authority) proposed by Yi and Kravets [162] exploits properties of threshold cryptography [135] and ad hoc on-demand routing protocols. The CA functionality is distributed among selected nodes called MOCAs. MOCA is described in the context of a military application being deployed in a battlefield, where each mobile node is associated with a soldier. The soldier's rank is used to select a MOCA. Eventually, the highest ranked soldiers are selected as MOCAs, assuming that the associated nodes possess better physical security, transmission range and some other desired capabilities. A client requests for certification service by sending certification request (CREQ) packets, similar to an on-demand route request (RREQ) packet. A MOCA receiving a CREQ, responds with sending certification reply (CREP) packet containing its partial signature as in [167]. CREP is similar to an on-demand route reply (RREP) packet. The client node can construct a full signature up-on receiving CREPs from at least k MOCAs. This is to note that there is no combiner in MOCA as in [167]. The number of MOCAs, n, depends on the number of nodes in the network as well as security and capabilities of the nodes. Capkun et al. [32] proposed a public key infrastructure for MANETs based on a self-organizing public key management system. The system in-

herits the idea of self-issuing public key certificate from Pretty Good Privacy (PGP) [169] but does not depend on certificate directories (not suitable for MANETs); instead, certificates are stored and distributed by the collaborating nodes. A node creates its public/private key pair locally. A node p can issue a certificate for node q's public key PK_q, provided that p believes that PK_q belongs to q. Each node maintains a local certificate repository containing a limited number of certificates. The certificate repository has two parts, one part stores certificates issued by the node itself. The other part stores certificates issued by other nodes. If node p wants to obtain an authenticated public key (PK_q) of q, it requests other nodes (including q) for PK_q. Node p receives certificate repositories from other nodes. Node p combines its local certificate repositories with received repositories in order to find an appropriate certificate chain from its own public key, PK_p to PK_q in the combined repository. Although it provides only probabilistic guarantee of availability of the requested certificate, this approach is a good choice for MANETs due to the fact it is fully distributed and offers less communication overhead compared to threshold cryptography approach.

2.2 Video Security in Ad hoc Networks

In the previous section, we have discussed security issues and possible attacks in MANETs and reviewed a number of solutions. In this section, we focus on security challenges regarding video streaming over ad hoc networks and review related work.

2.2.1 Video Security Challenges

Video streaming is resource intensive and has QoS requirements [121]. Video applications are delay sensitive and usually require high bandwidth. QoS provisioning is challenging in ad hoc networks. The self organization characteristic of ad hoc networks makes it particularly difficult to adopt the QoS mechanisms proposed for infrastructure-based networks where service provisioning is controlled centrally. Ad hoc networks have to constantly adapt with changing network topology and wireless link quality [121]. Introduction of security, cryptography in particular, could make real-time video streaming even more challenging for already resource stringent ad hoc networks. QoS requirements of real-time video streaming could be compromised due to

computationally intensive cryptographic operations.

2.2.2 Video Security Techniques

In this section, we discuss video security techniques that are found in the literature. We primarily focus on authentication and confidentiality. For authentication, we discuss several video watermarking techniques. As for confidentiality, we review a number of cryptography based approaches.

2.2.2.1 Video Watermarking

Digital watermarking is a technology for digital contents to provide digital right credentials through embedding copyright information within the digital content [68]. Watermarking is a popular method of providing authentication information such as ownership, copy, control, bi-level or gray level images, text or other digital data formats for digital multimedia contents [41] [25]. Imperceptibility, i.e., inability to distinguish watermarked video from the original video with bare eyes, invulnerability to tampering i.e., watermark is difficult to alter, and security are key properties of digital watermarking [124]. The main challenges of video watermarking are possibility of redundant data between frames, motion and motionless regions, lossy compression and susceptibility towards attacks such as frame averaging and frame swapping [20].

Video watermarking is mostly carried out either in the pixel domain or in the transform domain. Pixel domain technique embeds watermark by adding or replacing bits from the selected pixel positions. Although simple to implement, this approach lacks robustness and imperceptibility. In the transform domain technique, the host signal is transformed into a different domain using discrete cosine transformation (DCT) or discrete wavelet transformation (DWT) and watermark is embedded in selected coefficients. The transform domain approach has an upper hand over the pixel domain technique in terms of robustness and imperceptibility. The transform domain technique works at the frequency domain level, thus, has the ability of applying special domain properties. Watermark, can be applied to the raw video data, during or after the coding process [20].

A spread spectrum (SS) based watermarking technique is proposed by Hartung and Girod [67] where each bit of watermark is spread over a large number of chips (watermark representation bits) and modulated by a binary pseudo-noise sequence. In order for the watermark to be independent of the original video frame, watermark is added in the DCT domain. Scaled addition and high-pass filtering are used for watermark insertion and retrieval respectively. Robustness of this technique can be improved by increasing the number of chips, but at the expense of reduced data rate. A pixel domain technique discussed by Kalker et al. [86] utilizes a 2D SS method. The technique offers superior payload capabilities and shift invariance (down-sampling). Su et al. [144] proposed a collusion resistant, frame by frame watermarking technique that offers both robustness and imperceptibility. The watermark is repeatedly embedded around a fixed number of selected points in every video frame. These points change as the contents of the video frame change. Langelaar et al. [93] proposed a watermarking method for MPEG video using variable length code (VLC) swapping. Robustness of this proposal is questionable as it does not use any random key. Darmstaedter et al.'s proposal [44] utilizes the average energy or luminance intensities in sub-regions of each frame. This watermarking technique takes advantage of local spatial characteristics and achieves high data capacity by embedding one bit into every block in each frame. Cox et al.'s work in [40] is among the earliest examples of transformed domain video watermarking. This watermark embedding method is robust against lossy compression techniques. Deguillaum et al. [46] proposed 3D DFT (discrete flourier transform) based watermarking technique that utilizes temporal properties of a group of frames.

2.2.2.2 Video and Cryptography

Cryptography is a technique for providing confidentiality of digital contents. Cryptography is applied to confidential video data to prevent unauthorized access. Video data exhibits different characteristics depending on the target application. Security requirements for video data is also application dependent. For example, the requirement could be, video data is absolutely inaccessible by unauthorized viewers or partially accessible by some viewers.

The following are of key importance for applying cryptography to video data [145]:

- The computational overhead introduced by the cryptographic operations should not be a performance bottleneck;
- The cryptographic procedure should not affect the compression rate;
- The chosen cryptographic scheme should be resilient against video data loss; and
- The quality of video data should not be affected by cryptographic operations.

Cryptography can be applied to real-time streaming video in several manners. Encryption can be employed in the transform domain, within the video encoder. In most transform domain techniques, DC component and motion vectors are encrypted [164] [62] [103]. Format compliance of encoded and encrypted video is a key issue for this approach. The second approach is post compression encryption, where encoded video frames are encrypted individually. The third approach is to use a multimedia streaming protocol like RTP [118], appending video data to the packet payload. The payload is encrypted before transmission.

QoS of real-time video streaming could be compromised due to computationally intensive cryptographic operations. Introduction of cryptography could make real-time video streaming even more challenging for already resource stringent ad hoc networks. In order to maintain both performance and security at acceptable levels, several techniques have been developed. Selective encryption only encrypts selected data elements. For example, encrypting only the DC components or I-frames. Adaptive mechanisms are used to trade-off between QoS and security. Adaptive security technique can be employed to dynamically adjust the multimedia properties (e.g., discarding enhancement layers of the coded video data) for the sake of security. In this section, we review the selective and adaptive encryption techniques that are found in the literature.

Spanos and Maples [138] were among the first to introduce selective encryption for real-time video. Only the I-frames of MPEG coded video were chosen for encryption using a hardware based DES encryption scheme. The authors showed that their approach reduces overhead of encryption yet

achieves acceptable level of security.

Agi and Gong [4] performed an empirical study of secured MPEG video transmission. Their study shows that the encoding of inherently spatially and temporally correlated video makes encryption at the coding level difficult [99]. MPEG uses an asymmetric coding model, as a result of which encoding requires substantially more computing resource than decoding. An MPEG coded video is a sequence of time indexed-frames. The three dimensional video sequence is converted to a one dimensional serial bitstream. Intracoded 8×8 blocks or I-blocks makes up the I-frames. Encoded I-frames are used as motion estimation references for P and B-frames. Agi and Gong demonstrated that encrypting only the I-frames, as described in [138], and also seen in may other works in the literature, does not always guarantee the security of the entire video sequence. The presence of I-blocks in unencrypted P and B-frames is a security hole. A series of P and B-frames could carry enough information if their base frames are correlated. A frame containing an unencrypted I-block, being referenced by blocks in subsequent frames, can be decoded.

Kamphenkel and Blank [87] proposed an adaptive encryption technique to provide security for video data over 3G cellular networks. The technique is presented in the context of a telemedicine application. Cryptography is employed to protect confidential patient data during transmission. The authors proposed a model called Intelligent Network (IN) to address the issue of delay overhead caused by security measures. The proposed model uses the Stream Control Protocol (SCTP) [141]. It is a session oriented IP (Internet Protocol) like protocol. SCTP provides multi-homing through dynamic reconfiguration of IP addresses [143] and multi-streaming functions. Partial reliability SCTP (PR-SCTP) [142] is an extension of SCTP that allows separate streams in different classes of reliability to optimize streaming performance of real-time video. For example, only the I-frames of a MPEG encoded video can be chosen for reliable transmission. Secure SCTP (S-SCTP) [146] is a security extension for SCTP employing cryptography at the transport layer. Hohendorf et al. showed that S-SCTP suffers from throughput degradation [70]. IN aims to adapt SCTP to the changing network conditions by changing the routing path, regulating congestion control and security measures. IN incorporates bandwidth information and traffic class in path configuration for selecting a path re-transmission time limit and an encryption algorithm.

Evaluation of the proposal was carried out using Motion JPEG Stream at 10 fps encrypted using the Advance Encryption Standard (AES) with different key lengths.

Vaidya et al. [148] proposed a robust and secured technique for VoIP communications over MANETs. The technique proposes an efficient multipath traffic allocation approach. Scalable audio coding, G.727 [82] [52] is considered in the proposal. In addition to 16 Kbps core bit rate, G.727 offers three additional 8 Kbps enhancement layers. In their earlier work, the authors proposed AODV-MAP (AODV Multiple Alternative Paths) [147], a robust multipath routing mechanism that computes node-disjoint failsafe paths [126]. In [149], the authors proposed a secured variant of AODV-MAP, called SAODV-MAP (secured AODV-MAP), which employs threshold cryptography [135] and self-certified public keys. The proposed VoIP framework uses SAODV-MAP as the base. The core bitstream of G.727 coded data is transmitted over the primary path (fail-safe and higher data rate) of AODV-MAP and enhancement bitstream over the node-disjoint path. A source initiated mechanism performs secured route discovery as well as session key distribution. Content dependent scalable encryption is employed for providing data confidentiality. The base layer of G.727 is encrypted using a joint session key. A HMAC (Hash Based Message Authentication Code) [90] is generated and sent along the encrypted base layer. The enhancement layer undergoes lighter bitwise XOR operation with the joint session key and is sent to the destination along with a generated hash code.

Gibson et al. [62] investigated secure voice communications over MANETs using selective encryption for scalable speech coding. The aim is to provide bandwidth efficient speech coding, yet secured against passive eavesdropping. The authors have chosen the MPEG-4 scalable speech coding, SNR [52]. SNR is consists of a core layer, with minimum bit rate for acceptable speech quality, and one to multiple enhancement layers. Enhancement layers improve speech quality at the expense of increased bit rate. The main advantage of scalable coding is the ability to prune the data rate by excluding enhancement layers. In their proposal, the authors have chosen only to encrypt the core bit stream. The authors indicated the importance of choosing an encryption technique that does not exhaust resources with signal processing. For the evaluation of their work, the authors used the MPEG-4 standard CELP scalable speech coding tool to generate encoded audio consisting of

one core layer and two enhancement layers, covering 49% and 51% of the bitstream respectively. Thus, only 49% of the bitstream is encrypted. The authors have demonstrated that no feature of the original speech is identifiable from the encrypted stream.

Meyer and Gadegast proposed SECMPEG [103], an extension of MPEG supporting selective encryption. SECMPEG has four levels. The first level offer minimal security by encrypting only the sequence and slice headers of MPEG video. The second level also encrypts DC components of I-blocks and motion vectors. The third level encrypts the I-frames and all other I-blocks. The last level encrypts the entire video.

Tang incorporated cryptography at the video coding level to achieve compression and encryption in one step [145]. Tang's work was among the first that opposes what used to be a common belief that the spatial and temporal correlation, that exists in video coding such as MPEG, makes cryptography difficult to be applied at the coding level [99]. The author has proposed a technique to achieve encoding and encryption in a single step. In the MPEG compression procedure, after quantization, zig-zag scan is used to map the coefficients from macroblocks (matrix) to a vector. The proposed scheme replaces zig-zag scanning by a random permutation list. This vector contains coefficients ordered according to random permutation which is also the secret key. Experiments show that the resulted encrypted data is incomprehensible. The author pointed out a number of shortcomings in the proposal. The technique suffers from reduced compression rate. spatial distribution property of MPEG compression makes the technique vulnerable to several known cryptographic attacks. Thus, it is not suitable for highly sensitive video data.

Iqbal et al. [75] proposed a slice-based encryption technique for MPEG-4 H.264 [65] video. MPEG-21 [78] gBSD (generic Bitstream Syntax Description) is used as a metadata descriptor for the compressed bitstream. The metadata information can be used for adapting compressed video data according to the network condition or application requirements. The gBSD is generated during the encoding process. Adaptation information is first applied to the gBSD, which in turn produces an adapted bit stream from the original bitstream. MPEG-4 H.264 video frames can be split into one to multiple self-contained, independently decodable slices. Slices are composed of a group of macro blocks. Slices can be marked based on their importance.

Encryption is applied to a single slice or a group of slices composing the region of interest (ROI). This way, the amount of data to be encrypted can be reduced. In their previous work [76], the authors have chosen to encrypt the macroblocks containing the motion vectors.

Mahmud et al. [17] presented an adaptive security architecture for IP-based air-ground communications. The core component of their technique is called Security Manager (SecMan). The issue of resource overhead caused due to encryption is addressed. SecMan aims to maintain a trade-off between performance and security policies. A multilevel QoS policy is defined to manage the priorities between the services and to allocate the network resources. SecMan maintains a database of predefined security policies, called SSPD (Supported Security Protocol Database). SecMan uses the Multi-Criteria Decision Making Algorithm (MCDMA) [57] for selecting the best security policy. MCDMA uses the Analytical Hierarchical Process (AHP) [130] for adaptive security management. AHP is used to calculate the security service cost based on security strength, network cost and system cost of a security protocol. Security strength depends on the key length, block size and number of rounds defined in the security algorithm. Delay and bandwidth metrics are used to calculate system and network costs. Security negotiation is carried out among involved parties when a new data flow has to be secured.

Chapter 3

Background Information

In this chapter, we present the technical background relevant to our research work. We primarily focus on the contents necessary for understanding MANET operations, multimedia streaming and cryptography.

3.1 Mobile Ad hoc Networks

A Mobile Wireless Ad hoc Network or MANET is a communications network where the member nodes are self-organized. This means, absence of fixed infrastructure (e.g., central server, base station and access points), nodes communicate with each other directly or through intermediate nodes. Nodes act both as hosts and routers. Nodes could be both stationary and mobile, i.e., no static topology is guaranteed. The functional behaviour of a MANET can be compared with other wireless communications infrastructures such as WLAN [33], HSDPA, WiMAX, LTE, Flash-OFDM [66] etc. All these technologies utilize the wireless channel for communications and to support mobility. Common hurdles of wireless communications such as channel fading, collision, interference, distortion and effect of mobility are experienced by all wireless communications infrastructures.

At the 52nd IETF meeting, Macker and Corson provided the following definition of MANET [98]:

> *A "mobile ad hoc network" (MANET) is an autonomous system of mobile routers (and associated hosts) connected by wireless links - the union of which form an arbitrary graph. The routers*

are free to move randomly and organize themselves arbitrarily; thus, the network's wireless topology may change rapidly and unpredictably. Such a network may operate in a stand-alone fashion, or may be connected to the larger Internet.

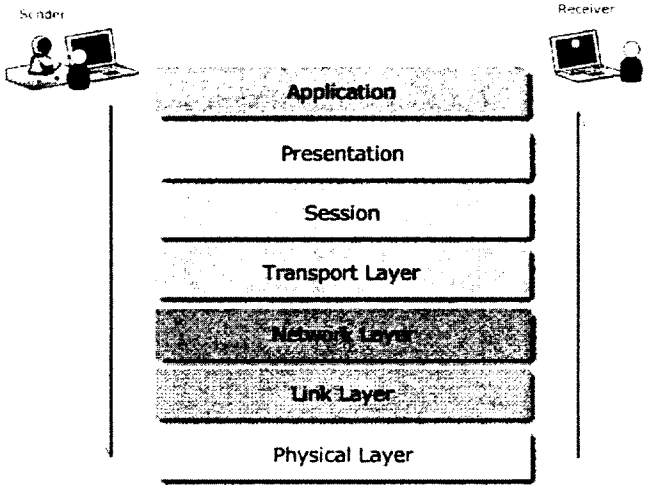


Figure 3.1: The OSI model.

The concept of ad hoc network has evolved from traditional infrastructure-based networks. Thus, an ad hoc network can be described using the ITU OSI (Open Systems Interconnection) model [80]. The OSI model describes a network in seven different layers where each layer is responsible for a specific group of tasks (Figure 3.1).

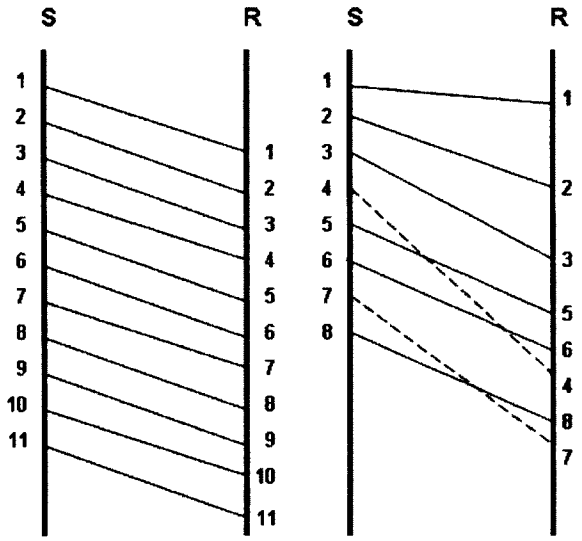
Layer one is the Physical layer and defines the physical attributes of the communications interface (e.g., Wi-Fi [33]) and radio propagation model for wireless transmission. It is responsible for transmitting and receiving bit-streams. Layer two is referred as the Data Link layer and defines the logical representation of data. For wireless networks, MAC (Medium Access Control) [1] protocols (e.g., 802.11) are also defined in layer two. Layer three is called the Network layer and is responsible for routing and packet forwarding. IP is the routing protocol that powers the World Wide Web [123]. For ad hoc networks, route discovery, route maintenance, routing table management and routing decisions are carried out at the network layer. Layer

four is the Transport layer that describes communications behaviour of end-systems and provides end-to-end flow control and error recovery. TCP [123] is a popular transport layer protocol for guaranteed data delivery, whereas UDP [123] is often chosen for delay sensitive services. Layer five, six and seven are termed as Session layer, Presentation layer and Application layer respectively. Similar to layer four, all these three layers present end-to-end perspective, providing great advantage for end-user application development. In practice, the presence of layers five, six and seven may not always be necessary and they are often generalized into a single application or service layer. Network-based multimedia applications typically utilize all seven layers.

3.2 Multimedia in Ad hoc Networks

The word meaning of *multimedia* is multiple media or ways of presenting information. For example, a document can have both a text-based and a speech-based representations. Multiple media may be used in combination for presenting information, such as audio visual media. In computing, multimedia commonly refers to presentation of information through media other than text-only representation. Examples of multimedia are static pictures, 2D and 3D animations, audio and video.

Multimedia contents are usually resource intensive in terms of storage, computation and distribution over a network. Consider the following example: We have a one minute long video. The pixel density of the video data is 360 $\times$ 240 prixels. Each pixel is 24-bit. 30 frames make up the one second long video. Storage requirement for this video data is 360 $\times$ 240 $\times$ 24 $\times$ 30 $\times$ 60 bits or approximately 3.5 Gb. In order to stream this video over a network and achieve the highest data rate (30 frames per second), the network would require to achieve 360 $\times$ 240 $\times$ 24 $\times$ 30 = 60 Mbps (approximately) effective data rate. High data rates require equally high bandwidth and bandwidth is expensive. 60 Mbps effective data rate is literally impossible for most of today's wireless networks. In addition to network resource limitations, maintaining spatial and temporal consistency of transmitted video contents are equally important. Figure 3.2 shows two streaming scenarios. Figure 3.2 (a) demonstrates expected streaming behaviour. Figure 3.2 (b) shows inconsistent delay experienced at the receiver's end and frames arriving out-of-order. Unexpected delays makes it difficult to perceive multimedia contents. Frames



(a) Expected streaming behaviour.

(b) Frames arriving out-of-order.

Figure 3.2: Two multimedia streaming scenarios.

arriving out-of-order cause further ambiguity and affect playback experience. For coded video [151] [77], frames arriving late and out-of-order are treated as obsolete and only contribute to causing network traffic overhead.

Limited bandwidth, error prone radio channels and absence of centralized control makes multimedia streaming even more difficult in ad hoc networks. Ad hoc networks have to cope with dynamic topology. Constant route management results in high control traffic. High traffic and high data rate requirements could cause network congestion, leading to high latency, distortion and packet loss.

In order to address the above-mentioned issues related to network-based multimedia services, various techniques have been developed. Most of these techniques were originally developed for wired and infrastructure-based wireless networks.

A digitization technique is employed to convert and compress the *raw*

video to a portable and network-friendly format. The lossy compression technique discards *components* of the video data that have little or no influence on human visual system. At the application and network levels, several techniques are employed to meet Quality of Service (QoS) requirements of the multimedia service.

3.2.1 Quality of Service

According to Crawley et al. [42] *"QoS is a set of service requirements to be met by the network while transporting a flow"*. The term *flow* refers to a packet data stream. QoS mechanisms utilize measurable performance metrics such as available bandwidth, packet loss rate, packet jitter, estimated delay, hop count, and availability and reliability of the routing path.

QoS is often directly related to perceptual or qualitative measures of user experience of a multimedia service. In a networked environment, multimedia applications are the primary examples of applications that demands QoS. Quality of experience of multimedia contents, such as streaming video and audio, is directly related to human visual and auditory sensory systems, respectively. It is considered that for real-time video, such as IP based video-conferencing, latency between two consecutive pictures should not be more than 150ms [118]. In addition to that, for audio-visual media, if audio and visual contents are out-of-sync, then the media become impossible to perceive. Therefore, for multimedia content delivery, QoS is very important.

There are two fundamental approaches to achieve QoS [121]. The first one is over-provisioning or simply increasing resource availability (e.g., bandwidth). The second approach is traffic engineering and is more practical and economical than over-provisioning. The latter approach can be further classified in three groups [121] [155]: QoS aware routing protocols, resource reservation mechanisms and QoS aware MAC protocols. A QoS service model usually combines the above methods for QoS provisioning.

For network-based video streaming, there have been a number of proposals for providing QoS. QoS can be provided at different network layers, both separately or utilizing multiple layers. In ad hoc networks, as discussed earlier, QoS provisioning is challenging because of limitations due to lack of infrastructure.

Integrated services (IntServ) [23] and Differentiated services (DiffServ) [21] are two QoS techniques for the Internet, standardized by the IETF. The IntServ technique works on each traffic flow to provide QoS. Traffic classification and scheduling are performed on a per flow basis. IntServ uses RSVP (Resource Reservation Protocol) [24] for resource management. Network resource utilization depends on the priority of the traffic flow. In DiffServ, traffic is divided into best effort (BE) and QoS classes. QoS class traffic are given higher priority over BE class traffic. DiffServ employs specialized routers which manage and control packet scheduling, queuing and dropping behaviours. DiffServ maintains a single routing table, meaning all the traffic to the same destination uses the same path, regardless of the traffic class.

Both IntServ and DiffServ were originally designed for IP networks and not with hop-by-hop networks in mind. Dependency on centralized administration makes these models not ideal for ad hoc networks. In an adverse network condition, a common route as in DiffServ, may lead to deprive the less priority traffic, resulting in high latency or service disruption. This consequence is called inter-class effect [150]. DiffServ's traffic profile is not practical for pure wireless links [155]. The QoS parameters DiffServ focuses on, are not often possible to guarantee for MANETs because of physical constrains of wireless ad hoc networks. Introducing Bandwidth Broker like technique would be far more complex in ad hoc networks, would require additional communications between routers and the agent, and also could raise security concerns. IntServ's per-flow approach to all QoS traffic is impractical for MANET due to bandwidth limitation [155]. RSVP approach is not practical for ad hoc networks because of its huge control traffic requirement. IntServ maintains state information for each flow and hence storage and processing requirement increases proportionally with number of flows. This problem questions scalability of IntServ.

3.2.2 QoS in Ad hoc Networks

QoS provisioning is challenging in ad hoc networks. The self organization characteristic of ad hoc networks makes it particularly difficult to adopt the QoS mechanisms proposed for infrastructure-based networks, where service provisioning is controlled centrally. Ad hoc networks have to constantly adapt with changing network topology and wireless link quality. Available

routing paths may not comply with service requirements and backup paths are unlikely to exist. Another important issue is competition between control traffic and QoS data traffic. Ad hoc networks generate a large number of control traffic for route discovery and maintenance. Both kinds of traffic are equally important and may compete to have access to network resources. Control traffic associated with QoS mechanism could cause additional network traffic. Traffic overhead is problematic in a low bandwidth network and makes QoS provisioning difficult. Furthermore, maintaining traffic profile and traffic flow state in arbitrary intermediate nodes is not practical for ad hoc networks, since routing paths cannot be guaranteed. With the mentioned challenges in mind, ad hoc network QoS techniques are typically modeled utilizing interaction and cooperation of key QoS elements like QoS routing, resource reservation scheme and QoS aware MAC layer protocol [121].

SWAN [5] is a distributed, decentralized and stateless QoS technique for ad hoc networks. SWAN uses feedback control mechanism and sender-based admission control for providing QoS for real-time traffic. Explicit congestion notification (ECN) [59] is used for congestion control. Intermediate nodes do not store per-flow information which makes SWAN stateless. SWAN employs source-based admission control and a rate control technique utilizing feedback from real-time traffic and MAC delay measurements that uses Additive Increase Multiplicative Decrease (AIMD) [34] rate control mechanism. The limitations of SWAN are strict source routing approach and for multipath routing, the packet probing technique would cause significant traffic overhead.

INSIGNIA [96] is an in-band signaling system that provides reservation-based services in MANET. Reservation technique used by INSIGNIA is independent of the routing protocol. INSIGNIA offers service differentiation, fast reservation and restoration with rerouting, end-to-end adaptation, and distributed resource control and resource management. A source initiated reservation method establishes end-to-end reservation with the destination, based on the service mode and the payload type. Resource information such as bandwidth requirements is used for reservation. Recourse reservation method requires the intermediate nodes, in the routing path, to maintain reservation state information for the requesting traffic flow. The service differentiation technique is built on IEEE 802.11 MAC DCF [60]. INSIGNIA

classifies traffic as BE (best effort) or EQ (QoS traffic). INSIGNIA is adaptive to change in available resource and packet flow is adapted accordingly. Key limitations of INSIGNIA are maintaining reservation states for multiple flows and multipath routing would be expensive. Furthermore, reservation-based QoS approaches, in a highly dynamic network, often suffer from false resource lookup and restoration. We believe only two, BE and EQ traffic classes are not sufficient to cover all types of traffic categories. INSIGNIA's adaptation technique changes between service classes only based on resource availability, but for QoS traffic, there are other possible adaptation alternatives (e.g., maintaining data rate by sacrificing quality enhancement elements).

FQMM (Flexible QoS model for MANET) [155] combines functionalities of IntServ and service differentiation of DiffServ. FQMM features dynamic roles of nodes, hybrid QoS provisioning and adaptive conditioning. In FQMM, a hybrid provisioning scheme aggregates per-flow mechanism of IntServ and per-class approach of DiffServer. Traffic with the highest priority follows per-flow method and other priority classes follow per-class provisioning. Although the hybrid approach is applied to reduce traffic load, considerable traffic overhead is caused by RSVP used by IntServ. We also believe control traffic associated with QoS provisioning of FQMM is still an overhead for multipath routing.

QOLSR [12] is a QoS extension of the original OLSR (Optimized Link State Routing) [37] routing protocol. QOLSR employs QoS at the routing level. QOLSR uses additional metrics other than hop count for optimal route selection. Bandwidth and delay information are used as additional metrics for optimal route calculation. These metrics are added to the OLSR routing table entry on each node. Delay is measured from latency of HELLO messages received from neighbouring nodes. Bandwidth is calculated based on MAC layer information. QOLSR uses a distributed algorithm for multiple metrics based optimal route selection. Both metrics can be used individually or together. For example, for two paths with equal bandwidth, delay is used as a second metric for selecting a routing path. QOLSR have a number of shortcomings. QOLSR does not specify support for multipath routing and traffic classification.

In [166] Zhang et al. proposed the adaptive source network rate control scheme (ASNC) for adapting voice coding bit rate to available network

resources, in order to minimize packet loss while maximizing voice quality. Adaptive coding, source-based rate control, packet combination and error checking are the key features of ASNC. ASNC uses AMP-WB (adaptive multi-rate wideband) [83] coding and determines a coding bit rate base on information such as packet loss, error, and MOS (Mean Opinion Score) [81]. AMR-WB coding offers nine bit rates from 6.6 to 23.85 Kbps with wideband characteristics and source controlled rate operation by voice activity detection (Discontinuous Transmission, DTX) [83]; very low rate during voice inactivity. PESQ (perceptual evaluation of speech quality) [81] is used to obtain MOS value of reception. The authors showed that MOS value is proportional to effective coding rate and used packet loss rate to calculate MOS value. Based on the available bandwidth, multiple voice data packets are combined in a single transport layer frame to decrease MAC frame rate which reduces packet loss. At the receiver's end, the decoder determines packet loss and reception error and provides feedback accordingly.

Canales et al. [31] proposed a cross-layer architecture for QoS provisioning in ad hoc networks. The proposal aims to provide admission control based on end-to-end available bandwidth estimation. The proposal takes advantage of the reliable broadcast service offered by ADHOC MAC [22]. The authors proposed dynamic TDMA (Time Division Multiple Access) slot assignment technique based on ADHOC MAC that allows conflict-free resource reservation for point-to-point communications. The proposed cross-layer solution uses MAC layer inter-operation with the routing protocol to estimate the TDMA slots available along the routing path. The MAC scheme is based on a frame structure that consists of two subframes: a control subframe and a data subframe. The control subframe broadcasts control information and allows to distribute necessary information to support resource reservation, maintenance of connectivity and routing operation. The TDMA based proposal also offers a distributed admission control in cooperation with a modified version of AODV. The authors showed applicability of their proposal using a number of example scenarios.

Multiple description coding (MDC) [63] is a way of providing error resilient network data streams. MDC fragments a single data stream into multiple independent streams. The independent streams may contain redundant information (distributed according to application requirements) or the single stream can be distributed over multiple sub-streams. A receiver of the

MDC streams can reconstruct the the original data by receiving sufficient information from the redundant streams or the sub-streams. Typically, the first approach offer fault-tolerance while the latter approach distribute traffic load across multiple paths. MDC along with multipath routing is a attractive choice for providing error resilient and performance enhanced multimedia service [10]. Generation of multiple projections and reconstruction of single data stream both are computationally expensive and could be problematic for live video streaming.

3.3 Video Streaming

Video streaming can be classified as [151]:

- Broadcast only services over terrestrial, cable, satellite, cable modem or DSL.
- Conversational services over IP or Cellular networks.
- Video-on-demand or multimedia streaming services.
- Multimedia messaging services (MMS).

Streaming video can be live, i.e., captured in real time or prerecorded and made available on-demand. The most common applications of real-time video are conversational, broadcast services and surveillance. Real-time or live video demands to meet playout deadlines, i.e., captured video must be transmitted and being played back at the receiver's end within a certain time interval. For live video streaming, temporal importance of video data is uncompromising. Our interest is transmission of real-time digital video over packet data networks such as IP networks.

IP based video streaming is composed of a number basic steps. Captured raw video data is first encoded (digitalized and compressed) and encode video data is transmitted as datagram packets. Up-on reaching the destination, video data must be decoded (decompressed) before playback.

It is important to understand video encoding, because encoding makes it possible to compress large amounts of video data to a portable format that

can be transmitted as network packets. MPEG [151] and Motion JPEG (M-JPEG) [77] are examples of the two most commonly used video coding standards. At present, MPEG-4 H.264/AVC (Advanced Video Coding) is the most popular video coding for digital video streaming. MPEG-4 H.264/SVC (Scalable Video Coding) is an extension of the original H.264/AVC, enabling scalable video [134]. An example of scalable coding is support for multiple resolutions by a single coded video data. To date, the decoding process of MPEG-4 H.264/SVC is computationally expensive and may not be suitable for delay sensitive real-time video services, such as videoconferencing, in a resource constrained ad hoc network.

3.3.1 MPEG-4 H.264/AVC

MPEG-4 H.264/AVC [65] is a video coding standard of the ITU-T Video Coding Experts Groups and ISO/IEC Moving Pictures Experts Group (MPEG) [79] [106]. H.264/AVC or MPEG-4 part 10 is the successor of MPEG-2 [64]. An important goal of the H.264/AVC standard is to provide network-friendly representation of video data for both conversational (e.g., videoconference) as well as non-conversational (e.g., video-on-demand or video broadcasting) video applications.

The primary targets of the H.264/AVC standardization are maximized coding efficiency, applicability to diverse range of network types, and robustness against loss and errors. H.264/AVC offers improved rate-distortion efficiency compared to the other existing video coding standards. In the video coding domain, only the decoder is standardized for all the receivers to be able to interpret the encoded bitstream and to produce similar output [151].

H.264/AVC covers a Video Coding Layer (VCL) which represents the video content and Network Abstraction Layer (NAL) that formats the VCL representation of the video and provide header information for transport layer protocols or storage media.

3.3.1.1 Network Abstraction Layer (NAL)

NAL provides customization of the use of VCL by mapping VCL data to the multimedia streaming protocols such as RTP [118] over IP and MPEG-2 for

High Definition (HD) video broadcasting. Key elements of NAL are *NAL units, bytestream, packet format, parameter set* and *access units.*

Coded video data is organized into packets called NAL units. The first byte of an NAL unit is a header byte that contains information about the type of data and the remaining bytes of the NAL unit are the payload. Payload is accompanied by emulation prevention bytes.

NAL unit is structured to support both bitstream-oriented and packet-oriented transport protocols. For bitstream-oriented protocols such as H.320 and MPEG-2/H.222.0 systems, each NAL unit contains three bytes long unique identifier called *start code prefix.* Start code prefix indicates the boundary of the NAL unit. In packet data based systems, e.g., multimedia streaming using RTP, coded data is transported as packets by the underlying transport layer protocol.

NAL units can be classified as VCL and non-VCL NAL units. The VCL NAL unit contains data that composes the video pictures. Non-VCL NAL units contain additional data such as important header information applicable to a group of NAL units and supplemental enhancement information that are optional for decoding. Non-VCL NAL unit contains sequence and picture parameter sets which decouple infrequently varying information from coded video data. Sequence and picture parameter set can be sent before corresponding NAL units. Non-VCL NAL unit can be used to verify packet loss and send request for retransmission if necessary. Non-VCL NAL unit parameter sets can be transported over a reliable service for guaranteed reception.

An access unit is a collection of NAL units in a specified form and composes a *primary coded picture.* Supplement enhancement information could precede the primary coded picture. Redundant coded pictures might follow the primary coded picture for the recovery purpose, in case of data loss or corruption in the primary coded picture. An access unit can be independently decoded to a picture.

A series of access units using the same parameter set is called a coded video sequence and is independently decodable. The first access unit of a coded sequence is called *instantaneous decoding refresh* (IDR). An IDR access unit contains an *intra* picture which is independently decodable and

is the reference picture for the subsequent pictures. A NAL unit may contain multiple coded video sequences thus multiple intra pictures.

3.3.1.2 Video Coding Layer (VCL)

H.264/AVC uses a block-based motion compensated hybrid coding approach. Each coded picture is represented in block shaped units of associated *luma* and *chroma* samples. These units are called *macroblocks*. The H.264/AVC coding algorithm is hybrid since it considers both temporal and spatial dependencies.

Human visual system processes brightness and color information separately with greater sensitivity to the details of brightness than color. In H.264/AVC color representation is separated into three components, called Y, Cb and Cr. Y is called the luma (luminance) that stands for brightness. Cb and Cr are the chroma (chrominance) information that represents the magnitude to which color changes from grey towards blue and red, respectively. Human visual system is more sensitive to luma compared to chroma components. H.264/AVC compression takes advantage of this property. The sampling structure of H.264/AVC coding maintains luma and chroma sample ratio at 4:2, i.e., each chroma component sample is one fourth of that of a luma component.

Partitioning pictures into *macroblocks* is the first step of the coding process. A picture is partitioned into fixed-sized blocks. Each block covers a rectangular picture area of 16×16 samples of luma component and 8×8 samples of each chroma components. A picture can be split into one to multiple *slices* (Figure 3.3). A sequence of macroblocks composes a slice. A slice can be decoded independently, given that reference pictures are available (if there is any). A collection of slices forms a *slice group* which can be treated as an independent logical unit as well. One to several slice groups may present on a H.264/AVC frame.

Depending on the used coding type, there can be three main types of slices, namely I, P and B-slice. An I-slice has the macroblocks coded using intra prediction. The P-slice extends properties of the I-slice and also contain macroblocks coded using inter prediction with at most one motion-compensated prediction signal per prediction block. The B-slice extends

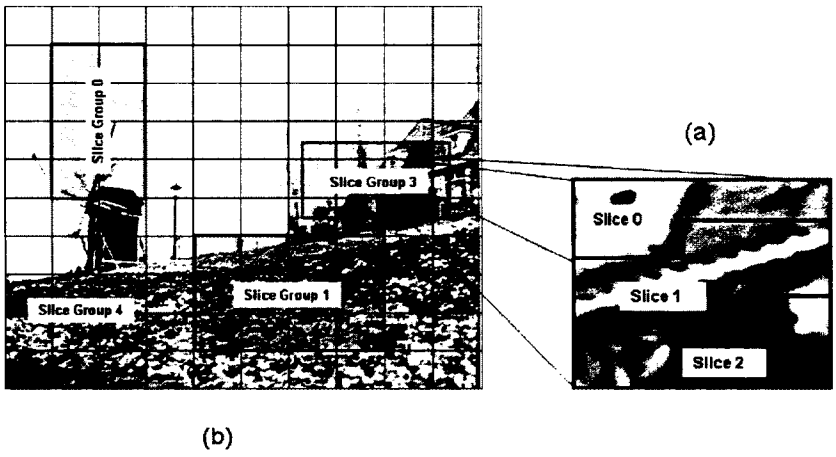


Figure 3.3: (a) MPEG-4 H.264/AVC slices. (b) MPEG-4 H.264/AVC slice groups.

properties of the P-slice and also contain macroblocks coded using inter prediction with two motion-compensated prediction signal per prediction block. Additionally, there are two new slice types, called *Switching I* (SI) and *Switching P* (SP) slice.

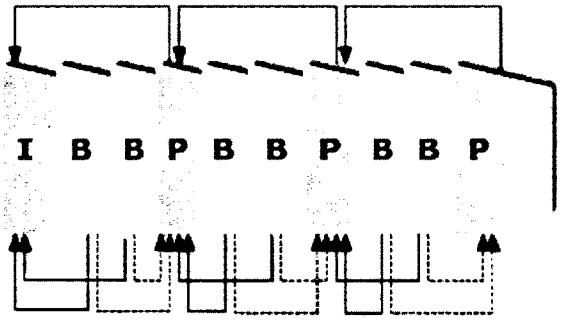


Figure 3.4: An example of MPEG-4 H.264/AVC GOP structure.

The concept of independent slices is comparable to *frames* in previous standards. A video frame containing a single I-slices is the same as an I-frame. There could be a series of P and B-frames following each I-frame. A sequence of P and B-frames and their reference I-frame compose a Group Of

Pictures (GOP) (Figure 3.4).

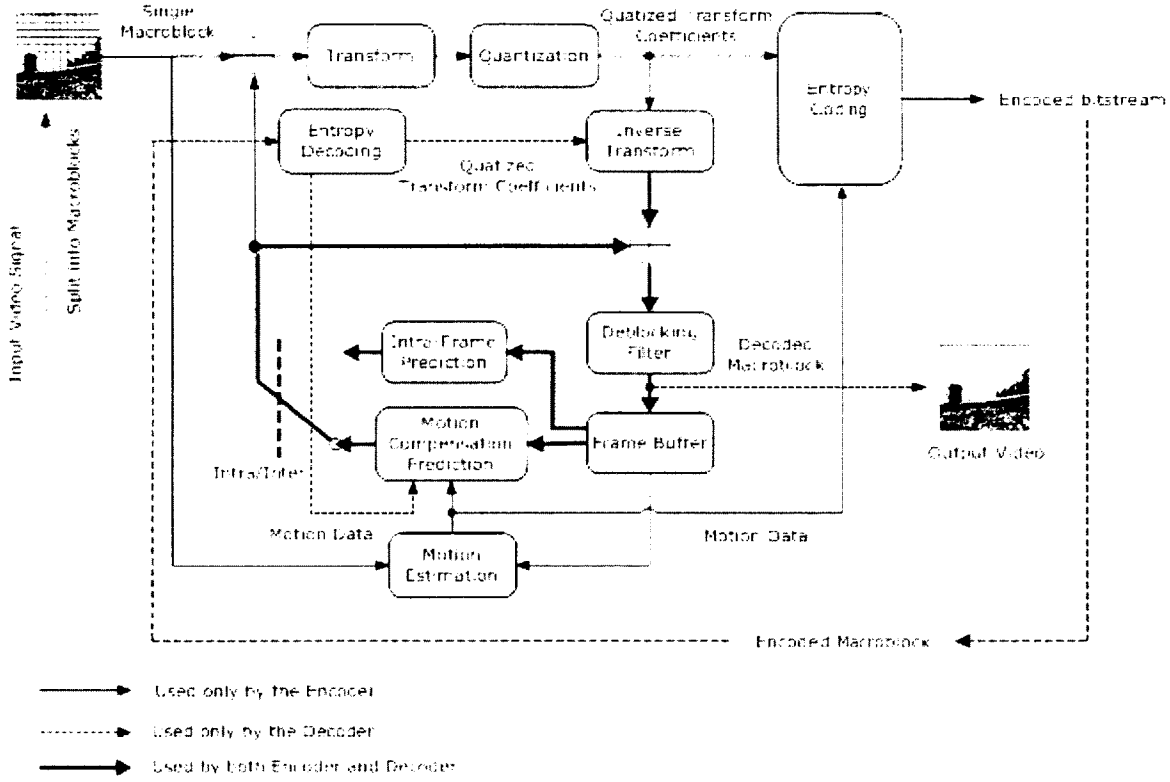


Figure 3.5: High level architecture of MPEG-4 H.264/AVC Encoder and Decoder.

Luma and chroma samples are either spatially or temporally selected. The resulting prediction residual is encoded using *transform coding*. Each color component of the prediction residual is subdivided into 4×4 blocks and *integer transform* is applied to each block. The *transform coefficients* are *quantized* and coded using *entropy coding* technique. Figure 3.5 illustrates the operations of VCL.

Macroblocks are coded in *Intra* or *Inter* mode. Intra predicted macroblocks are predicted using information from macroblocks that belongs to the same picture when temporal prediction is difficult or inefficient. Macroblocks are predicted using motion compensation in inter mode coding.

H.264/AVC allows motion compensated prediction with multiple references, i.e., an inter macroblock may have several reference pictures (Figure 3.6).

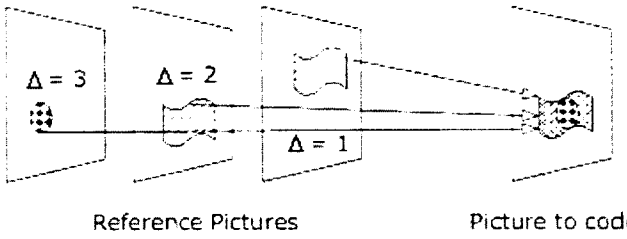


Figure 3.6: Inter mode motion estimation with multiple references. Δ is the referenced picture's parameter set.

The next step in the coding process is *Transform Coding.* Transform coding is applied to reduce spatial redundancy of the prediction error signal. Transformation is applied mainly to the 4×4 blocks. Integer transform is used instead of *discrete cosine transform* (DCT) as used by the earlier standards [151]. A key advantage of using integer transform is that the inverse transform is defined by exact integer operations, thus inverse-transform mismatches are completely avoided by the decoder. The transformation outputs a set of coefficients, each of which is a weighting value for a standard basis pattern. A quantization parameter is used to determine quantization of transform coefficients. H.264/AVC uses scalar quantizer for this purpose. The quantized transform coefficients generally are scanned in a zig-zag fashion. Zig-zag scan of transform coefficients shows that statistical distribution of larger values for the low frequency part and smaller values for the high frequency part. H.264/AVC standard specifies two entropy coding methods: a lesser complex method based on context-adaptively switched sets of variable length codes, called CAVLC, and the computationally more demanding context-based adaptive binary arithmetic coding (CABAC). Both coding methods shows improvements in terms of coding efficiency compared to techniques used in prior standards. H.264/AVC employs *In-Loop Deblocking Filter* technique to reduce production of visible block structures.

In summary, at the encoder, the transform coding process includes a forward transform, zig-zag scanning, scaling and rounding as the quantization process is followed by the entropy coding. At the decoder, the inverse of the

encoding process is performed except for the rounding.

3.4 Transport Layer Protocol

Two remote hosts communicate with each other by means of a Transport Layer protocol. The transport layer sits in between the application and network layers. The transport layer presents end-to-end perspective to communicating parties by hiding lower network layers. The transport layer protocol determines packet formats and other transmission control properties, such as retransmission and acknowledgements. Examples of transport layer protocols are UDP (Unified Datagram Protocol) and TCP (Transmission Control Protocol). The choice of a transport layer protocol is directly related to the application traffic class.

UDP is an unreliable yet fast transport layer protocol. During transmission, UDP packets may be lost or arrive out-of-order. TCP, on the other hand, was originally developed for guaranteed delivery and is suitable for best effort traffic. Multi-step handshaking dialogs are used by TCP for connection establishment. TCP ensures ordering of data using sequence numbers and offers retransmission when necessary [123].

Although it provides fast transmission, UDP is unreliable. Delay sensitive applications such as multimedia applications commonly use UDP but introduces additional features, such as sequence number, congestion control, error recovery and retransmission mechanisms. The RTP (Real-time Transport Protocol) [133] is an example of popular IP based multimedia streaming protocol. RTP was developed by the IETF. Although, the original design of RTP is independent of transport layer protocol, RTP is commonly used as an application layer protocol on UDP. RTP is consists of a pair of protocols, RTP and RTCP (RTP Control protocol). RTP is responsible for transmission of multimedia data, while RTCP provides control information, such as feedback on the quality of service, for RTP streams [118].

3.5 Routing in Ad hoc Networks

According to the seven layer OSI model [80], a routing protocol belongs to the network layer. A routing protocol establishes the data routing characteristics of the network. A routing protocol defines the logical communication infrastructure, presents nodes with the network topology and defines communication rules among the member nodes. In a MANET, the routing infrastructure is established in a distributed manner. In ad hoc networks, a routing protocol is responsible for neighbour discovery, route discovery, route management, computing optimal routes and traffic forwarding [14]. Murthy and Manoj discussed in detail the design goals of MANET routing protocols [107]. An ad hoc routing protocol is designed with the following key expectations:

- Provides stable loop-free connectivity.
- Has reduced control traffic overhead.
- Responds to changes in topology and link connectivity.
- Offers mobility management.
- Offers scalability.
- Flexible enough to incorporate QoS extensions and security mechanisms.

The main components of an ad hoc routing protocol are a routing table, that contains full or partial topology information, control messages for discovering and maintaining routes, and a mechanism for identifying optimal routes.

Ad hoc routing protocols can be grouped in three primary classes:

a. Reactive

b. Proactive

c. Hybrid

a. Reactive rououting protocol

A reactive routing protocol performs route discovery on demand; i.e., communications begins with route discovery and usually nodes do not maintain network topology (e.g., routing table). Examples of reactive routing protocol are DSR (Dynamic Source Routing) [14] and AODV (Ad hoc On-Demand Distance Vector) [14].

b. Proactive rououting protocol

A proactive routing protocol maintains topology information through sending periodic control messages and nodes maintains routing tables that convey topology information. OLSR (Optimized Link State Routing) [14] is an example of proactive routing protocol.

c. Hybrid routing protocol

A hybrid routing protocol aims to combines key features of both proactive and reactive protocols. ZRP (Zone Routing Protocol) [14] is a hybrid routing protocol.

Each type of routing protocol has its own strengths and weaknesses. The reactive approach generates less control traffic but may experience higher latency at the beginning of transmission. Performance degrades in high mobility environments due to increased rate of route discovery. The proactive routing protocol, on the other hand, generates considerably higher (periodic) control traffic but is a better choice for dense networks with high node mobility.

An ad hoc routing protocol performs forwarding through source routing or in a hop-by-hop manner [14]. In many routing protocols, hop count is the deciding factor for selecting the optimal path. Often the computed shortest path, in terms of hop count, is not the optimal path. Additional metrics such as bandwidth, latency, and packet loss information can also be used for computing the optimal routing path.

In addition to a unique optimal path, routing protocols can utilize multiple paths for routing. The key advantage of multipath routing is that it distributes traffic load among several routes, enabling parallel transmission

to achieve high data rate while reduces network congestion and traffic distortion.

A number of routing protocols have been standardized by the IETF [74]. RFC 3561 [119] describes AODV and RFC 3626 [37] specifies the OLSR standard. In the following two sections, we describe AODV as an example of reactive routing protocol and OLSR as an example of proactive routing protocol.

3.5.1 AODV

AODV (Ad hoc On-Demand Distance Vector) [119] is a reactive routing protocol that uses distance vector technique (e.g., Bellman-Ford shortest Path). AODV essentially combines functionalities of DSR (Dynamic Source Routing) and DSDV (Destination-Sequenced Distance Vector) [120] routing protocols. In AODV, route discovery is performed on-demand. AODV offers a route maintenance mechanism and loop free routing, and the use of message sequence number prevents count to infinity problem.

In AODV, data transmission begins with broadcasting RREQ (route request) message for route discovery. A RREQ contains source address, current source sequence number, destination address, destination sequence number, broadcast ID and hop count. If a neighbouring node does not contain route information for the target destination, it rebroadcasts the RREQ with its address as the current source and increases hop count value and sequence number. Source address and broadcast ID are used to identify a RREQ and duplicates are discarded. The sequence number offers freshness of the RREQ as the latest sequence number is maintained by the intermediate nodes and is broadcasted only if current RREQ's sequence number is greater than that of the existing one. As a RREQ message propagates through the network, intermediate nodes also update their routing table. A node maintains a reverse route entry of already discovered route in the routing table. The routing table contains addresses of all destination nodes (the originator of the RREQs), destination sequence number, number of hops to the destination and next hop address for the destination from the current node. A route entry in the routing table maintains a lifetime flag which is updated every time the route is used. Links are maintained by sending HELLO messages to the neighbours. If a node receives a HELLO message from its neighbour,

it updates the lifetime flag and the route containing that link is assumed to be active. A route is discarded if the lifetime flag expires. Once the RREQ finds the destination node, RREP (route reply) message, originated at the destination node, is returned to the source. The RREP message typically follows the route maintained in the routing tables on the intermediate nodes.

3.5.2 OLSR

OLSR (Optimized Link State Routing) is a proactive routing protocol standardized by the IETF [37]. RFC 3626 specifies the OLSR standard. OLSR is a table driven routing protocol. Core components of OLSR are Neighbour discovery, MPR selection and Topology discovery. OLSR uses the Dijkstra's algorithm for route computation. Each node in OLSR, maintains a routing table. OLSR offers support for multiple interfaces as well.

Destination node address	Next hop address	Distance (number of hops)	Interface address (link to be used to reach the next hop)

Table 3.1: OLSR routing table.

OLSR primarily uses two kinds of control messages for establishing the routing infrastructure: HELLO messages are used for neighbour sensing and TC (topology control) messages are used for discovering the topology, thus computing routing paths. Control messages are sent out periodically for both link sensing and route maintenance, and routing tables are updated accordingly. Each node maintains a routing table that consists of the entries in Table 3.1.

Periodic HELLO messages are broadcasted by each node. HELLO message contains a node's neighbour list and MPRSelector set (discussed later). HELLO messages are not forwarded or rebroadcasted. If a node's one-hop neighbours can reach all of their two-hop neighbours through that node, the node is selected as an MPR (multipoint relay) node and is added to each selecting node's MPRset. An MPR node maintains MPRselector set containing

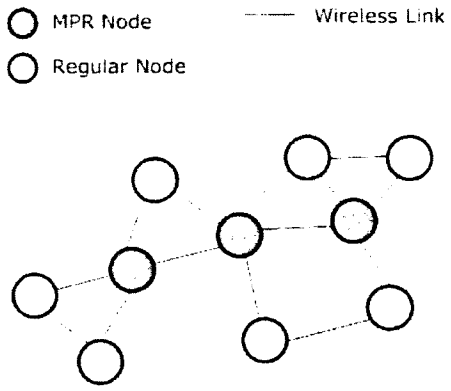


Figure 3.7: OLSR routing protocol showing selection of MPR nodes.

the one-hop neighbours it was select the MPR for (Figure 3.7). Information available in the HELLO messages are utilized in the MPR selection process. The MPR selection process focuses on minimizing the number of MPRs. There is a *willingness* parameter which represents whether a node is interested to act as an MPR.

TC messages are used to establish the network topology. TC messages are only forwarded by the MPR nodes, both periodically and on detecting change in the network topology. TC messages contain MPRselector set of the MPR node. Up-on receving a TC message, a node updates its routing table. Routing table 3.1 contains information about one-hop neighbours as well as next hop address and hop counts for all the other nodes in the newteork. There is a TTL (time-to-live) value for each entry in the table.

A shortest path algorithm, e.g., Dijkstra's Shortest Path algorithm is employed for route calculation. Route calculation in OLSR can be as simple as considering the hop counts to other more complex methods. For example, bandwidth information, mobility metrics and congestion information can be considered for path calculation.

OLSRv2 (OLSR version 2) [35] was proposed in 2009. OLSRv2 retains the same basic algorithms from the original OLSR, but offers a modular architecture which provides grater flexibility for extensions, such as QoS and security. OLSRv2 is consists of three key operational procedures: Neighbour

discovery, MPR flooding and Link State advertisement. OLSRv2 provides resilience against a number security threats [36]. OLSRv2 employs packet sequence numbers that helps discarding older control messages. OLSRv2 ignores unidirectional links, providing some resilience against jamming attacks. Message interval bounds may limit the impact of an indirect jamming attack. Additionally, OLSRv2 control packets allow extensions for integrating security features such as digital signatures.

3.5.3 Multipath Routing

In contrast to classical single path routing, packets can be routed from a source to a destination through several different paths. Multipath routing offers better reliability, fault-tolerance, reduces traffic load on a single path and improves latency, characteristics which are important for providing improved QoS. Multiple paths can further be used as backup routes for redundant data packets as well as for parallel transmission. Routes in multipath routing can be classified as *Link Disjoint*, *Node Disjoint*, *Inter Twisted* and *Hybrid* [161]. Link Disjoint paths do not share a common link (an edge) connecting two intermediate nodes. Node Disjoint paths are absolutely unique as no two paths share any node and as a result do not share any common link. Inter Twisted paths may contain common links. Hybrid routes may contain both link and node disjoint and inter twisted paths.

AOMDV (Ad hoc On-demand Multipath Distance Vector) [101] described by Marina and Das is a multipath extension of the original AODV routing protocol. AOMDV algorithm computes multiple loop-free disjoint paths. AOMDV shows significant improvement in routing performance. A key problem experienced by reactive multipath routing protocols is flooding of a large number of redundant route requests and route reply packets [159]. Calculating multiple paths is more efficient in proactive routing protocols. Proactive routing protocols maintain topology information beyond single-hop neighbours at every node which makes multiple route discovery less cumbersome [161].

There have been a number of proposals for mulipath proactive routing protocols, particularly for OLSR. Kun et al. [91] proposed multipath OLSR based on IP source routing where paths are node disjoint and information about intermediate node's interface queue is used for selecting paths. Badis

and Agha [11] described a multipath extension for QOLSR [13]. QOLSR utilizes bandwidth and delay metrics for route calculation. The proposed multipath algorithm uses bandwidth and delay information to calculate loop free and node disjoint routes. The algorithm uses a correlation factor, the number of shared links among the paths, to reduce interference among the paths.

This is important to mention here that pure disjoint multiple paths could lead to inefficient routing [160]. Routing protocols, employing Dijkstra's Shortest Path algorithm, might not work for sparse networks with pure disjoint multiple paths [161]. Completely disjoint paths may produce limited number of paths as well as very long paths. Also for techniques where route computation incorporates other metrics, such as in QOLSR [11], absolute disjoint paths are very much likely to introduce inefficiency.

Multimedia streaming can harness various advantages of multipath routing to enhance over all quality of the multimedia service. Multimedia streaming is both delay sensitive and requires high bandwidth. Availability of multiple paths can lessen congestion, thus reduce latency and packet loss.

3.5.3.1 Multipath OLSR (MP-OLSR)

MP-OLSR (Multipath OLSR) proposed by Yi et al. [161] is a variant of the original OLSR routing protocol. Although, MP-OLSR is based on proactive routing protocol OLSR, MP-OLSR is not a pure proactive, link-state routing protocol. An MP-OLSR node does not always maintain routing paths to all possible destinations in the network, rather it computes the routes when necessary. This is to avoid computational complexity associated with computing multiple routes. The source computes a route and appends the route information to the transmitted packet, similar to source routing. MP-OLSR provides support for loop detection and route recovery. MP-OLSR verifies presence of loop in the routing path by looking at the source defined route in the packet header. Route recovery is performed only using topology information saved on the local node. An intermediate node verifies the existence of the next hop in the appended route information before it forwards a data packet. In case the next hop in the source defined route is not available, the intermediate node recomputes the route and appends the new route information to the forwarded packet.

MP-OLSR uses the same topology sensing mechanism of OLSR. For route computation, Multipath Dijkstra's algorithm is used to calculate multiple paths from the information gathered through topology sensing. The Multipath Dijkstra's algorithm is used to obtain N paths from a source, u to a destination, v. If G represents the graph underlying the network, given G and u, then the Dijkstra's algorithm provides the source tree, ST containing the shortest paths in G for the source node u. A procedure, called $GETPATH(ST, v)$, extracts the shortest path from u to v in ST. The above two steps are performed to obtain N different paths from u to v. How N different paths are obtained? In each step, i, a path, P_i is returned by the procedure $GETPATH(ST, v)$. Two incremental functions, namely f_p and f_e are employed for identifying disjoint paths connecting u and v. f_p is used to increase *link costs* (similar to the original Dijkstra's algorithm) in the previous path P_{i-1} in order to obtain link disjoint paths. f_e on the other hand, is used to obtain node disjoint paths by increasing cost of the links incident to vertices in P_{i-1}. It is also possible to obtain hybrid paths in order to maximize the number of available paths. The result is an N-tuple $(P_1, P_2, P_3...., P_N)$ of paths from u to v in G.

3.5.4 Evaluation of Routing Protocols

In this section, we present results of our evaluation of the above discussed three ad hoc routing protocols. We compare a number of performance metrics of the three routing protocols. The results are obtained in the identical environment. The simulation environment and implementation details are available in Chapter 5. Table 5.1 lists the network simulation parameters. The results are confirmed with 90% confidence level.

Figure 3.8 shows packet delivery ratio of the three routing protocols for different network sizes. On the x-axis is the network size. Packet delivery ratio is on the y-axis. All three routing protocols perform similarly for the 16 node network. Beyond 16 nodes, MP-OLSR completely outperforms AODV and OLSR. With increase in the network size, the packet delivery ratio of MP-OLSR increases as well. This is due to increase in number of available paths for parallel transmission. For larger networks, OLSR shows increase in packet delivery ratio while AODV's packet delivery ratio declines. Performance of AODV declines with increasing network size while OLSR performs

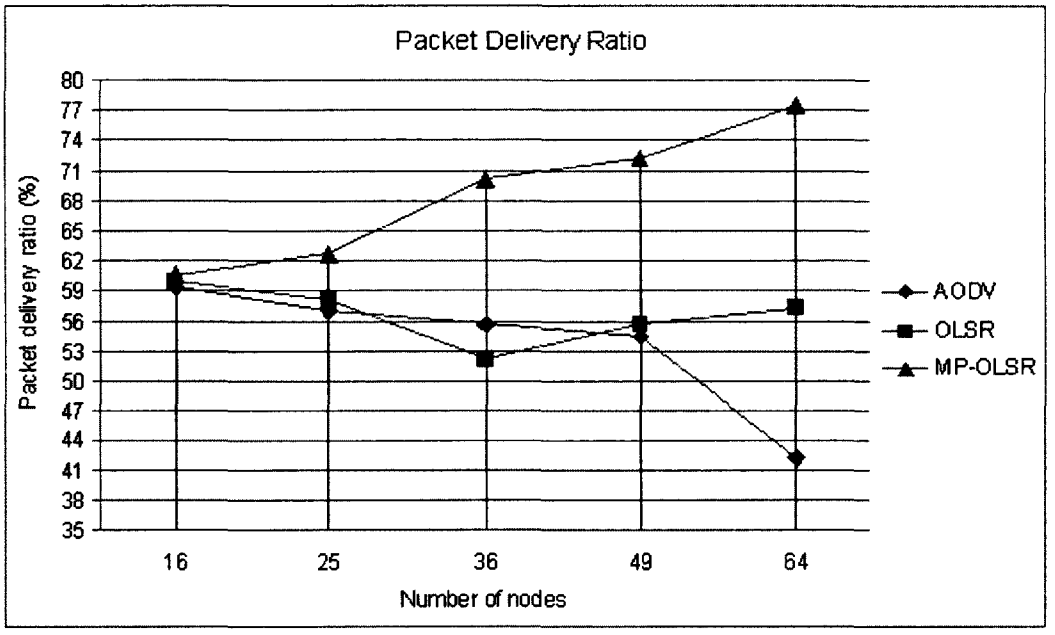


Figure 3.8: Comparison of packet delivery ratio for different network sizes.

better among the two.

Figure 3.9 compares network latency experienced by the three routing protocols for different network sizes. On the x-axis is the network size and end-to-end delay or network latency is on the y-axis. We measure one-way packet delay (average end-to-end delay) in milliseconds. For the 16 node network, AODV performs slightly better than both OLSR and MP-OLSR. For the 25 node network, AODV performs much better than the two other protocols, while OLSR performs the worst. Beyond 36 nodes, network latency of MP-OLSR is lower compared to the other two routing protocols.

Figure 3.10 shows median absolute deviation of end-to-end delay. Median absolute deviation of end-to-end delay demonstrates consistency or predictability of transmission behavior. As we can see MP-OLSR outperforms AODV by more than 30%. OLSR on the other hand, because of a fixed single routing path, shows slightly better consistency compared to MP-OLSR.

Figures 3.11, 3.12 and 3.13 show the interface queue (IFq) statistics

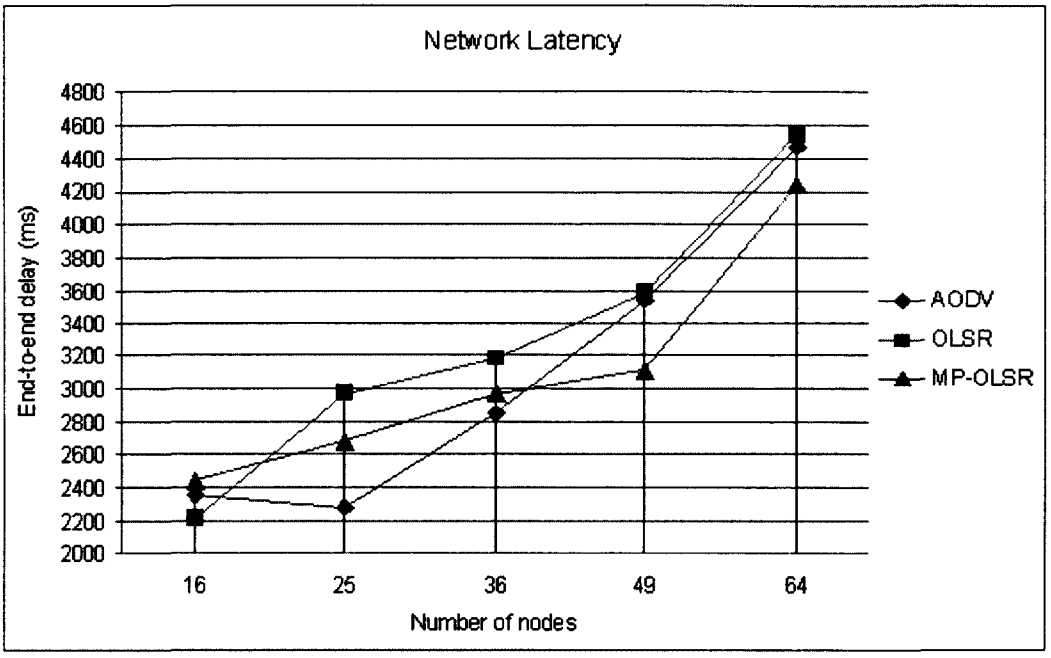


Figure 3.9: Comparison of network latency for different network sizes.

for all three routing protocols for different network sizes. IFq statistics provides information regarding network congestion. The higher the number of packets present in the IFq at any time during transmission, the greater is the magnitude of congestion. We have collected data for 16, 25, 36, 49 and 64 node networks. For better visibility, we present results for 16, 36 and 64 node networks. The results show that AODV has most number of packets in IFq for the longest duration while MP-OLSR has the least number of packets for any network size.

3.6 Medium Access Control (MAC)

Medium Access Control (MAC) protocol defines how a wireless node accesses the shared wireless channel. MAC layer communications is limited between adjacent nodes, nodes that are within each other's radio transmission range. MAC protocol determines when to release packets and when to back off. The IEEE 802.11 was originally developed for WLAN (Wireless Local Area Network) communications. The IEEE 802.11 is a MAC layer and physical

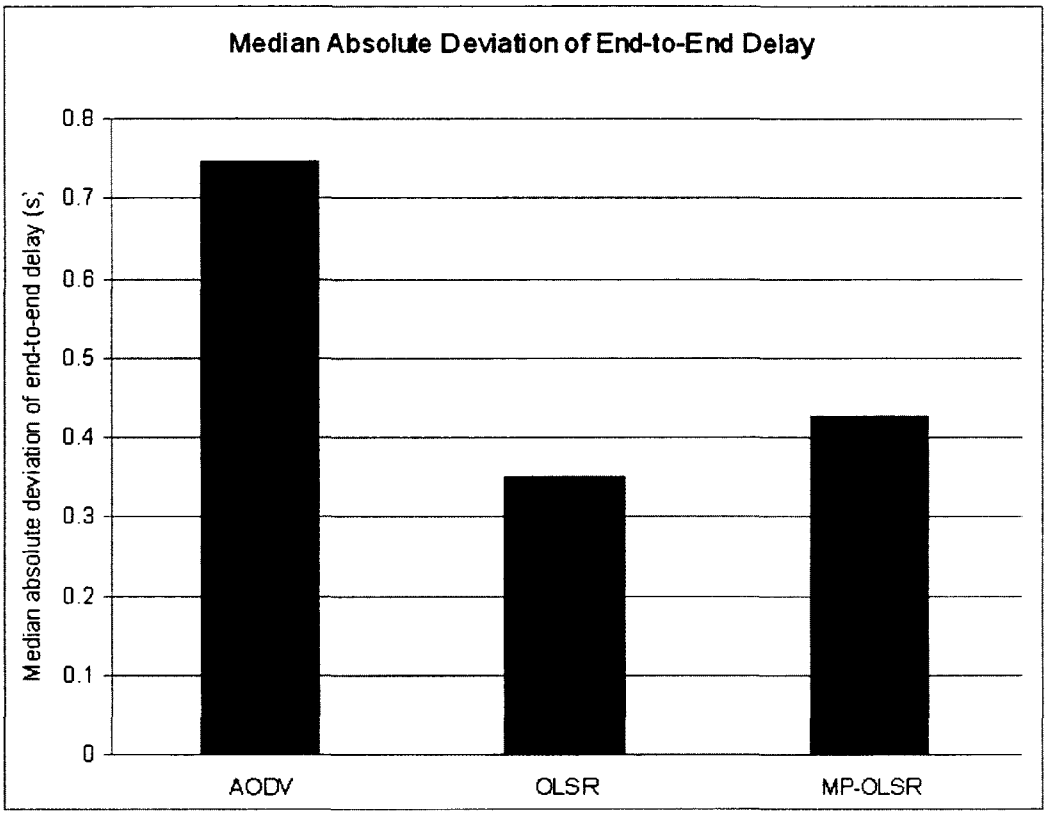


Figure 3.10: Median absolute deviation of end-to-end delay.

layer protocol standardized by the IEEE 802.11 working group [1]. The IEEE 802.11 operates at either the 2.4 GHz industrial, scientific, and medical (ISM) band or the 5 GHz unlicensed national information infrastructure (UNII) band [60]. The IEEE 802.11 WLAN operates both in the infrastructure and ad hoc mode [1].

3.6.1 IEEE 802.11 MAC

The IEEE 802.11 standard defines MAC specifications [1] for WLAN. The IEEE 802.11 MAC protocol determines how nodes share a common wireless channel. IEEE 802.11 MAC mechanism consists of a mandatory distributed coordination function (DCF) and an optional point coordination function (PCF).

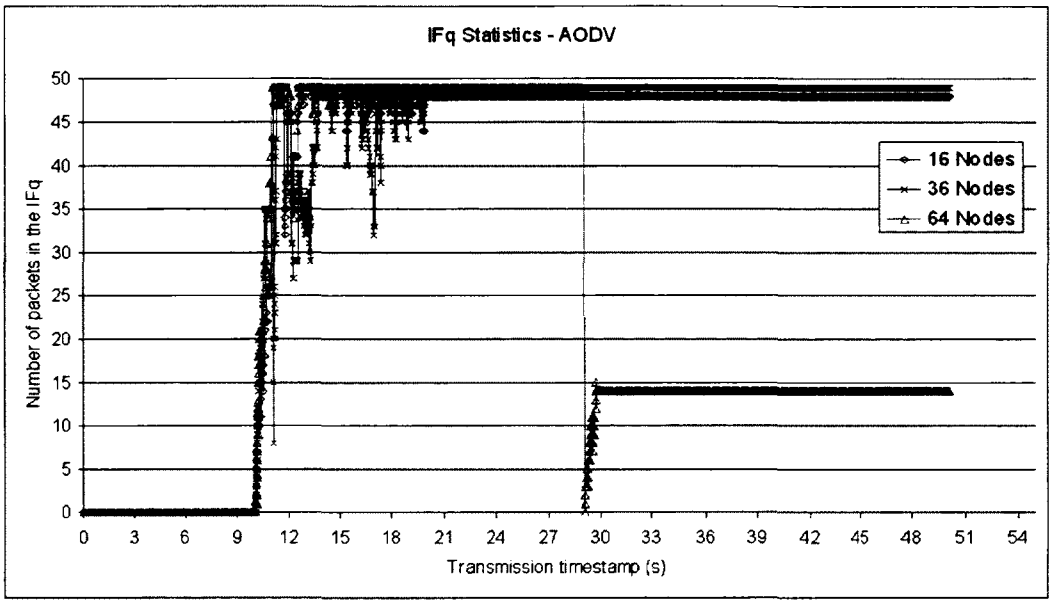


Figure 3.11: Interface queue statistics for AODV.

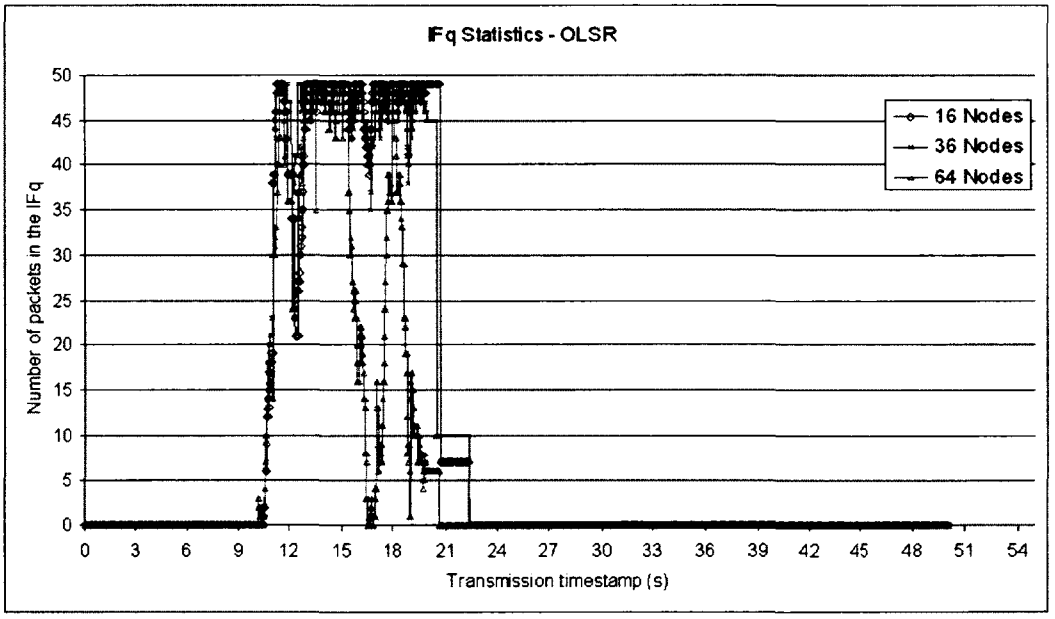


Figure 3.12: Interface queue statistics for OLSR.

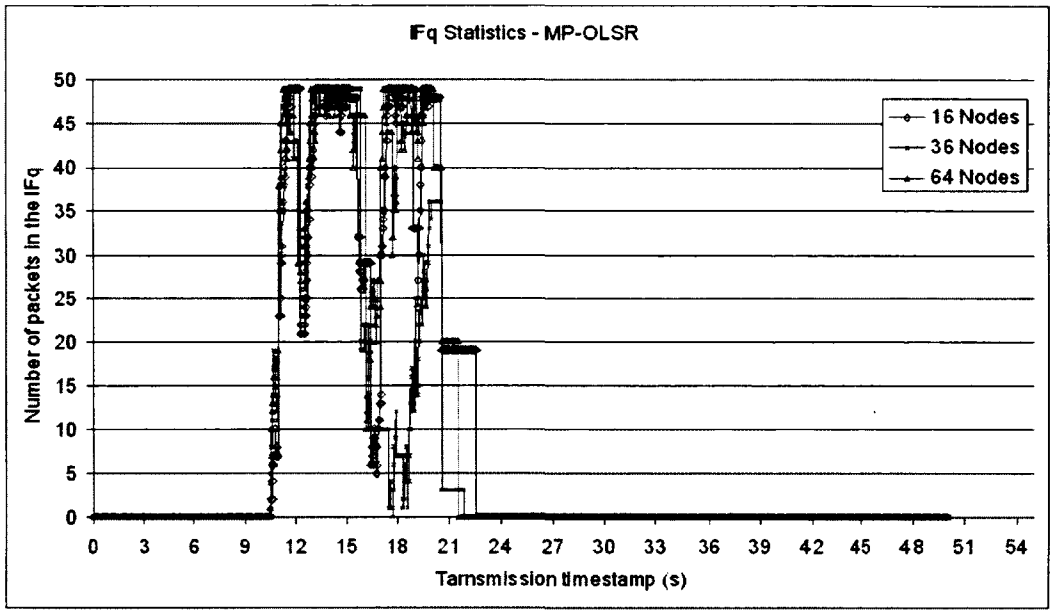


Figure 3.13: Interface queue statistics for MP-OLSR.

DCF utilizes a technique called carrier sense multiple access with collision avoidance (CSMA/CA). A node listens to the wireless medium to verify the presence of traffic. The logical unit maintains a *backoff* counter, a uniformly distributed random number between zero and the contention window (CW). Contention window is the maximum wait time for the backoff counter. A node starts transmitting when the backoff counter is reduced to zero and the medium is free, and transmits for a duration of DCF interframe space (DIFS). A transmission is initiated by the sender by sending RTS (request to send) frame. If the receiver is ready for reception, it sends back a CTS (clear to send) frame. Up-on receiving CTS, the sender initiates the ACK (acknowledgement) timer and begin data transmission. The receiver acknowledges the reception by sending ACK. If the sender does not receive ACK within the ACK timer window, retransmission is scheduled. Collision might occur when two nodes try to transmit at the same time. If no ACK is received, then it is assumed that a collision has occurred. On detecting collision, nodes double their backoff time up to CW_{max}. Figure 3.14 is an example of the operation of DCF. PCF requires central control, thus, is not suitable for ad

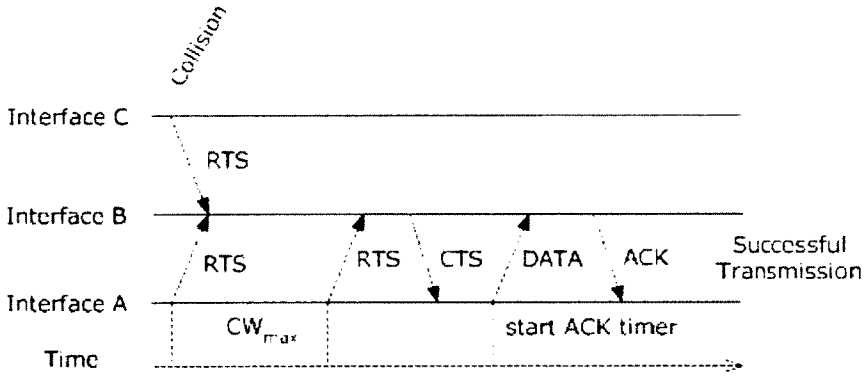


Figure 3.14: Example of IEEE 802.11 MAC operations (DCF).

hoc networks [60].

IEEE 802.11 was originally designed for best effort traffic. In DCF, all nodes share the same channel access parameters, thus, introducing support for QoS is challenging.

3.6.2 IEEE 802.11e MAC

IEEE 802.11e [2] is a QoS extension of the original IEEE 802.11 [1]. IEEE 802.11e introduces hybrid coordination function containing two medium access mechanisms, namely contention-based channel access and controlled channel access. Contention-based channel access uses the technique called enhanced DCF channel access (EDCA). The HCF controlled channel access (HCCA) technique is used in the controlled channel access. Among the MAC layer QoS mechanisms described in the IEEE 802.11e standard, we are particularly interested in the admission control mechanism. It controls data traffic for different service classes. The advantages are QoS of the traffic flow of interest is maintained and efficient utilization of network resources.

EDCA introduces the concept of access category (AC). Different ACs possess different channel access priorities and serve different traffic types. Different ACs have different CW_{min} and CW_{max} values. An AC carrying higer priority traffic is assigned a lower CW_{min} value to achieve higher transmission opportunities (TXOPs). Backoff period of an AC is also different

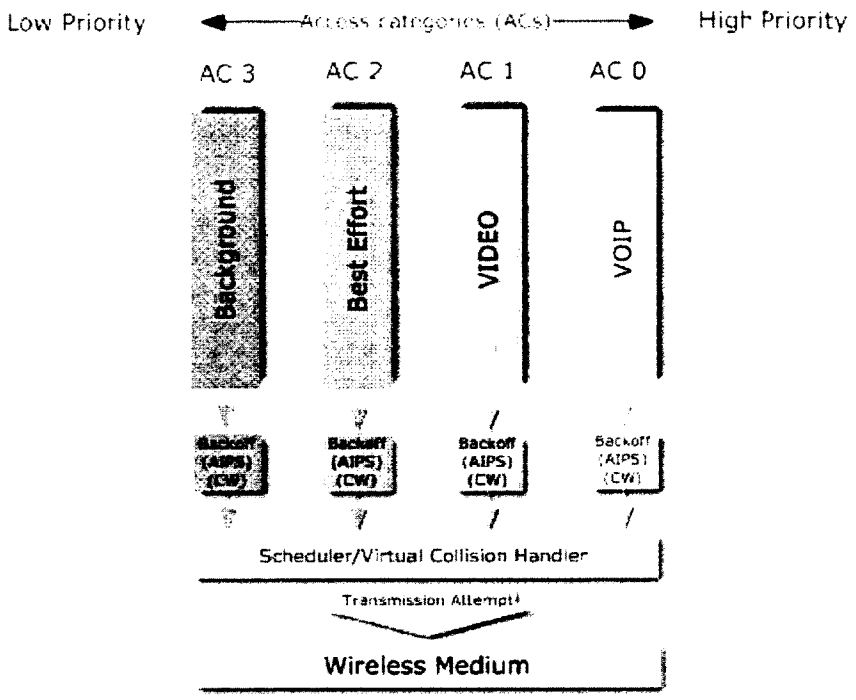


Figure 3.15: IEEE 802.11e EDCA access categories.

based on AC's inter frame space (IFS), called arbitration IFS or (AIFS). Virtual collision occurs when two ACs' backoff periods elapse at the same time. In this case, Virtual Collision Handler, an inside scheduler, allows the AC carrying higher priority traffic to access the physical medium and the AC carrying lower priority prepares to try again. Figure 3.15 shows four ACs carrying VOIP, video, best effort and background traffic. Here, each type of traffic has a different priority. VOIP is assigned the highest priority class, while background traffic belongs to the lowest priority class. Centralized control of HCCA leads the admission control to have a deterministic nature, which is not suitable for ad hoc networks [60].

Key advantages of the IEEE 802.11e standard are employing QoS at the MAC layer, through traffic prioritization and admission control, and virtual collision avoidance, enabling efficient resource utilization and reducing packet overhead and on-channel collision.

3.6.3 Evaluation of 802.11 MAC Protocols

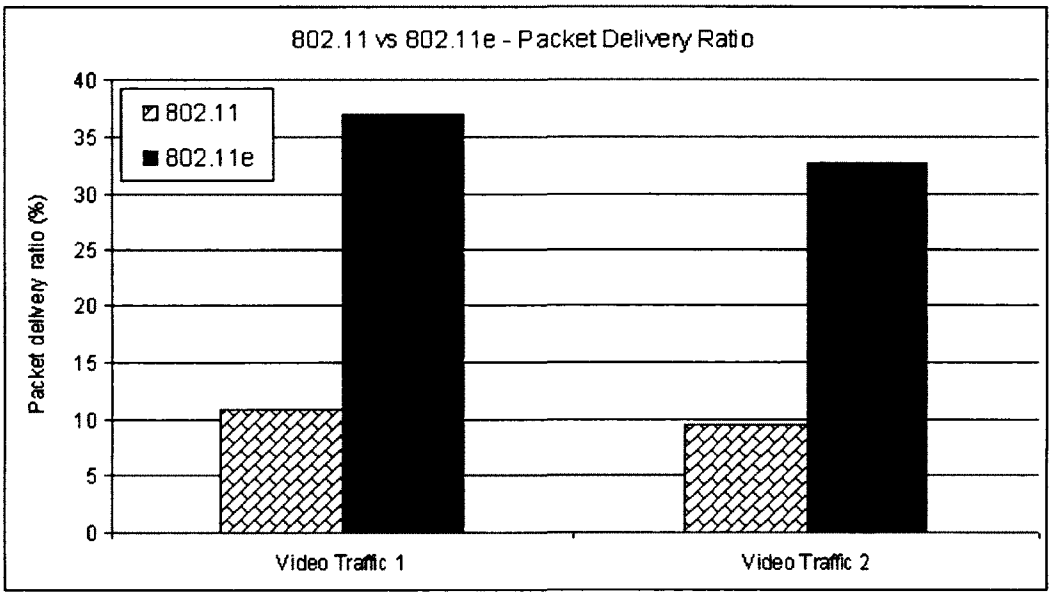


Figure 3.16: Comparison of packet delivery ratio of 802.11 and 802.11e.

Figure 3.16 compares packet delivery ratio for video transmission in 802.11 and 802.11e MAC environments. The results are obtained in the identical environment. The simulation environment and implementation details are available in Chapter 5. Table 5.1 lists the network simulation parameters. The results are confirmed with 95% confidence level.

We carried out simulations in a 16 node ad hoc network organized in grid topology. AODV is used as the routing protocol as we have seen before that AODV performs better in the smaller networks. There are five simultaneous traffic flows, one VoIP, two Video, one CBR and one FTP over TCP. In the 802.11e MAC, VoIP, Video, CBR and FTP over TCP are set from the highest to the lowest priority order. From Figure 3.16, we can see a significant difference in performance from 802.11 to 802.11e. 802.11e delivers about 50% more packets for the first video traffic flow and 45% more packets for the second video traffic flow. The admission control mechanism of 802.11e clearly demonstrates its superiority in presence of multiple traffic flows.

3.7 Cryptography

In this section, we review cryptography concepts and techniques, and present overview of the cryptography algorithms that we have used in this work.

3.7.1 Cryptography Overview

In computing, *Cryptography* is a technique for providing confidentiality of digital contents. Cryptographic techniques transform digital contents to an unrecognizable altered form. In cryptographic terms, the original content is referred to as the *plaintext* and the transformed content is called the *ciphertext*. The procedure that transforms a plain text to a ciphertext is called *encryption*. The method of reviving the original plaintext from the ciphertext is called *decryption*. Usually the encryption and decryption procedures are lossless and non-additive, but there are exceptions.

The basis of a cryptographic technique is sharing some secret information among trusted parties. In a networked environment, confidential information could travel through unknown intermediate routers before reaching the target destination. A cryptography technique ensures that only the legitimate target destination or trusted parties have access to the confidential data by sharing a piece of secret information in advance. This secret information is commonly termed as the *key* of the cryptographic technique. A cryptographic key is essential for the encryption and decryption operations [139].

Depending on the characteristic of shared secret information, cryptographic techniques can be classified into *symmetric key* and *public key* based cryptographic infrastructures. In symmetric key infrastructure, a single key is shared and used for both encryption and decryption. Public key infrastructure, on the other hand, utilizes two separate keys. Encryption is carried out using the public key, whereas a private key is used for decryption. A public key based approach involves a key exchanging technique such as Diffie-Hellman key exchange [139]. Both approachs have their own strengths and weaknesses and are appropriate for different application domains.

According to operational procedure, cryptographic techniques can be further classified as *stream cipher* and *block cipher*. Stream ciphers operate on a bit stream of arbitrary length, e.g., a single byte. Block ciphers, on the other

hand, only accept a block of data of a predetermined size and transform the input into a ciphertext. Stream ciphers are usually light weight and faster than block ciphers but vulnerable to various attacks [100] [139].

For our evaluation, we have chosen several cryptography algorithms that are FIPS (Federal Information Procession standard) [58] and NIST (National Institute of Standards and technology) [108] approved. The following is the outline of the selected cryptography techniques:

- Symmetric key infrastructure:
  - Stream ciphers: RC4 and Salsa20
  - Block ciphers: DES and AES
- Public key infrastructure:
  - Elliptic Curve Cryptography

In the next few sections, we briefly describe these cryptography techniques.

3.7.2 RC4

RC4 [100] is a stream cipher based on 256-byte internal state called *S-Box* (Sand Box). For a given secret key, RC4 produces a *keystream* and data are encrypted by XORing with the keystream. The RC4 algorithm has two parts: The *key scheduling algorithm* uses the secret key and loads a key register (to the S-box) with a permutation on integers 0 to 255. A *pseudorandom number generator* produces one-byte of keystream on each call to the generator and also updates the S-box. The one-byte keystream is XORed with one-byte of the plaintext data to produce a ciphertext of the same length. The decryption procedure is exactly the same as the encryption, only in this case the input is the ciphertext.

3.7.3 Salsa20

The Salsa20 stream cipher is one of the submissions included in the final eSTREAM (ECRYPT Stream Cipher Project) portfolio in 2008 [18] [19]. Salsa20 is a stream cipher that works in counter mode. Salsa20 does not

need to prepare an internal state like RC4 and therefore, has no setup phase. The core of Salsa20 is a hash function that accepts a 512-bit block of plaintext and outputs an equal size cipher text [156] [69]. The keystream generation function takes a 256-bit secret key and a 64-bit nonce (a unique message) and together with the 64-bit counter (sequence number) outputs a 512-bit keystream. The core of the keystream generation function is called *quarter-round* function (QR). A 512-bit or 64-byte input block is configured as a 4×4 matrix of 32-bit words,

$$\begin{pmatrix} x_0 & x_1 & x_2 & x_3 \\ x_4 & x_5 & x_6 & x_7 \\ x_8 & x_9 & x_{10} & x_{11} \\ x_{12} & x_{13} & x_{14} & x_{15} \end{pmatrix} \quad \text{where } x_i \text{ is a 32-bit word and } i \in \{0, 1, 2, 3, ..., 15\}$$

Four words in each row and in each column are modified by QR 10 times

$$\begin{cases} QR\,(x_0\ x_1\ x_2\ x_3) \\ QR\,(x_5\ x_6\ x_7\ x_4) \\ QR\,(x_{10}\ x_{11}\ x_8\ x_9) \\ QR\,(x_{15}\ x_{12}\ x_{13}\ x_{14}) \end{cases} \overbrace{QR\begin{pmatrix} x_0 \\ x_4 \\ x_8 \\ x_{12} \end{pmatrix} QR\begin{pmatrix} x_5 \\ x_9 \\ x_{13} \\ x_1 \end{pmatrix} QR\begin{pmatrix} x_{10} \\ x_{14} \\ x_2 \\ x_6 \end{pmatrix} QR\begin{pmatrix} x_{15} \\ x_3 \\ x_7 \\ x_{11} \end{pmatrix}}$$

The transformation process (on each 4-word tuple) is the following:

If $y = (y_0, y_1, y_2, y_3)$, then $QR(y) = (z_0, z_1, z_2, z_3)$, where y_j and z_j are 32-bit words, $j \in \{0, 1, 2, 3\}$ and

$$\begin{aligned} z_1 &= y_1 \oplus ((y_0 + y_3) <<< 7) \\ z_2 &= y_2 \oplus ((z_1 + y_0) <<< 9) \\ z_3 &= y_3 \oplus ((z_2 + z_1) <<< 13) \\ z_0 &= y_0 \oplus ((z_3 + z_2) <<< 18) \end{aligned}$$

The transformed matrix is added to the original input matrix to produce a 4-word or 512-bit keystream block. Plaintext is XORed with the keystream to produce the ciphertext.

3.7.4 DES

DES (Data Encryption Standard) [110] is a symmetric key based block cipher technique. Triple-DES is a more secured variant of the original DES. Triple-DES (also known as TEDA as specified in ANSI X9.52 [154]) is approved by FIPS and NIST. The original DES uses 56-bit long keys. DES encryption produces ciphertext having the same length of the original plaintext. The block size used in DES is 64-bit. Alike other block ciphers, DES is employed using a mode of operation (described in Section 3.7.6). A 16 round identical iterative process transforms a plaintext loaded in the 64-bit block in a ciphertext of the same length. With increasing rounds, the security of the algorithm increases exponentially. The main components of DES operation are Key scheduling, Initial Permutation (IP), Feistel function (F) and Inverse Initial Permutation (IP^{-1}) or Final Permutation (FP). Fundamental operational procedure of DES is described below:

Key scheduling step uses the 56-bit key to produce 48-bit subkeys for each of the 16 rounds. Plaintext undergoes Initial Permutation (IP). Inverse of IP, IP^{-1} is called the Final Permutation (FP). The block containing the plaintext is separated into 32-bit subsegments (left and right halves) and is operated on alternately. This operation involves Feistel function (F) processing one 32-bit half (of the 64-bit block) using the 48-bit subkey, output of which (F) is XORed with the other 32-bit half of the block and the resulting halves are swapped. (No swapping on the 16^{th} round). This operation is carried out 16 times. After 16 rounds, IP^{-1} or FP transforms the block into the final ciphertext. The decryption procedure is almost identical as the encryption except for the 48-bit subkeys are applied in the reverse order.

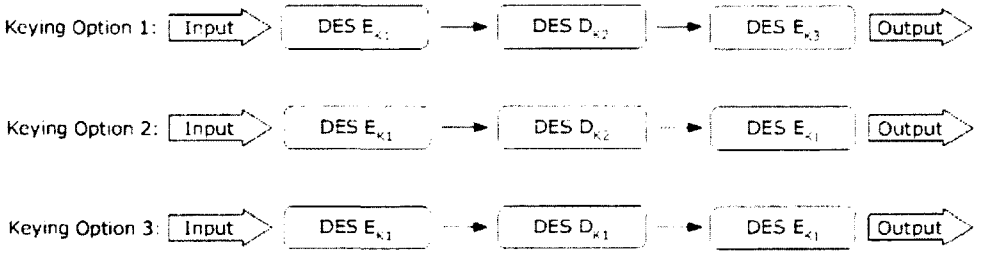


Figure 3.17: Triple-DES Keying options.

Triple-DES utilizes three keys, each 56-bit, for its functional procedure. Plain text loaded in a 64-bit block, is encrypted, decrypted and again encrypted using three keys. All three keys could be the same or different or only keys used for encryption could be the same, as illustrated in Figure 3.17. Decryption is carried out in reverse of the encryption procedure. Triple-DES is backward compatible with the original DES.

3.7.5 AES

AES (Advanced Encryption Standard) [47] is a symmetric key based block cipher algorithm. AES supports 128, 192 and 256 bit long keys. AES has a fixed block size of 128 bits. Length of the ciphertext is the same as the original plain text. An iterative transformation process called a *round* transforms plaintext into a ciphertext.

The AES algorithm with 128-bit key length and 128-bit block size operates on a 4×4 matrix (a two-dimensional array) of bytes called a *state*. A 128-bit input (loaded in the block) is transformed into 4 rows of bytes each containing 4 bytes. Each column of the two-dimensional array, on the other hand, is 32-bit long and is called a *word*. Number of rounds depends on the size of the key. For 128, 192 and 256 bit long keys, number of rounds are 10, 12 and 14 respectively. A round operating on a state, has four main functions namely SubBytes, ShiftRows, MixColumns and AddRoundKey. All rounds are identical except for the final one which does not carry out MixColumns.

A *Key Expansion* routine generates a *key schedule*. Each round utilizes a key schedule consists of 4-byte words. SubBytes function operates on each byte of the state. An *S-box* is used to achieve a non-linear byte substitution. S-box is a non-linear substitution table used in several byte substitution transformations and in the Key Expansion routine to perform a one-for-one substitution of a byte value. S-box operates in the finite field $GF(2^8)$ (Galois field). ShiftRows function cyclically shifts over several bytes present in the last three rows of a state. MixColumns operates on each of the 32-bit columns of a state to transform all 4-bytes of each column. In the AddRoundKey step each column of the state is XORed with a round key (a 32-bit word from the key schedule).

3.7.6 Block Cipher Mode of Operation

This is important to mention that block ciphers such as DES and AES can operate in stream cipher mode and take advantage of stream cipher techniques. Examples of modes of operation are ECB (Electronic Code Book), CBC (Cipher Block Chaining), OFB (Output Feedback), CFB (Cipher Feedback) and CTR (Counter). For our evaluation, we used CFB which is a self synchronizing stream and can be used to encrypt any number of bits [55].

3.7.7 Elliptic Curve Cryptography

Elliptic curve cryptography (ECC) [38] is a public key cryptography system. ECC demonstrates performance advantages at higher security level. ECC considers elliptic curves over finite field (known as Galois field, GF) integers modulo a prime number $GF(p)$ (where p is the prime number and $p > 3$) or binary polynomial $GF(2^m)$. The key size of an elliptic curve cryptosystem is the size of the prime number or the binary polynomial represented in bits. Common ECC key sizes are (in bits) 163, 256, 384, 512.

Elliptic curve cryptosystem is based upon the mathematical complex problem of elliptic curve divergence logarithm (ECDLP). The elliptic curve divergence logarithm problem is the following: nG, a multiple of a point G, is easy to compute if $G \in E(GF(2^m))$ where E is an elliptic curve on the finite field $GF(2^m)$. But without knowing E, solving n, for given G and nG is a difficult problem. In a typical elliptic curve cryptosystem, G is the public key and n is the private key. Stealthness of an elliptic curve cryptosystem much depends on the property of the composition of the underlying elliptic curve. A safe elliptic curve should be composed of a big rank and the rank has a big prime factor and the curve is not hyper singular [165].

An elliptic curve E has an Abelian group [38] structure with identity element O called the point of infinity. The elliptic curve E over the binary finite field $GF(2^m)$ can be represented by the following equation (called the Weierstrass equation.):

$$E : y^2 + xy = x^3 + ax^2 + b$$
where (x, y) is a point in $E(GF(2^m))$ and $x, y \in GF(2^m)$
with $a, b \in GF(2^m)$ and $b \neq 0$

Establishing an ECC cryptosystem involves selecting an underlying finite field, identifying the elliptic curve domain parameters of the field being used and a representation for the elements in the finite field. Then, an elliptic curve has to be carefully chosen together with a point on the curve called the *generator* and finally generating elliptic curve key pairs (n, G). Finite field arithmetic operation is employed to carry out *point multiplication* which is the fundamental operation of elliptic curve cryptosystem.

The main operation of ECC is point multiplication. The basis of security in ECC relies on the point multiplication operation. Multiplication of a scalar k with any point P on the elliptic curve, to calculate another point Q on the curve, is an example of point multiplication. Point multiplication can be carried out by two elliptic curve operations: *Point Addition* and *Point Doubling*. Point addition is adding two points X and Y to obtain another point Z $(Z = X + Y)$. Point doubling is adding a point to itself (doubling of Z produces $2Z$). So, if $Q = kP$ and k is given, then Q can be obtained from repeated addition and doubling of P. Each point addition and doubling involves a *multiplicative inverse* operation. Finding multiplicative inverse is a costly operation in both finite fields. It is notworthy that there are other efficient methods to carry out point multiplication (e.g. Window-NAF [137]).

ECC can be used for signature schemes, key agreement and encryption schemes. Elliptic Curve Augmented Encryption Scheme (ECAES), also known as Elliptic Curve Integrated Encryption Scheme (ECIES), provides semantic security against adversary capable of launching chosen-plaintext and chosen-ciphertext attacks. Elliptic Curve Diffie-Hellman (ECDH) is used by ECIES for key agreement [27].

The key advantages of ECC are use of binary polynomial that does not require integer multiplication which reduces hardware implementation complexity, security of a 171 -180 bit ECC key is equivalent to RSA cryptosystem with 1024-bit key [152] which makes ECC a better choice for network communications, especially resource stringent wireless communication as well as on smartcards and embedded devices, and non-existence of effective sub-index arithmetic to attack for a carefully chosen elliptic curve [165]. Although offer shorter keys, ECC encryption is relatively more computationally expensive than the previously described cryptography algorithms.

3.7.8 Evaluation of Cryptography Algorithms

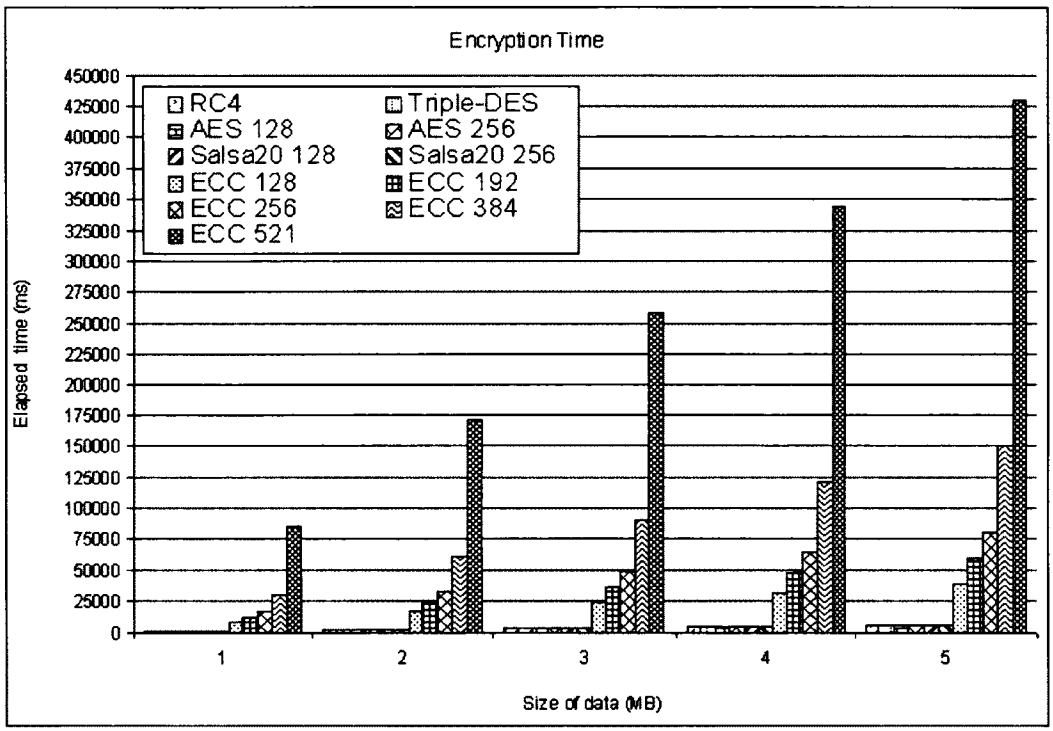


Figure 3.18: File size vs encryption time of different cryptography schemes.

In this section, we present results of our own evaluation of the above discussed cryptography algorithms. The cryptography algorithms are implemented using the Crypto++ library version 5.6.1 [43]. Crypto++ is a open source cryptography library implemented in C++. The results are obtained in the identical environment. The environment and implementation details are available in Chapter 5.

We evaluate and compare runtime performance of the five cryptography algorithms in the identical environment. We carry out the evaluation for five different file sizes; 1 MB, 2 MB, 3 MB, 4 MB and 5 MB. The files are populated with random binary data. The size of the unit data buffer that the encryption operation is performed on is 1024 bytes. Decryption procedure follows the same method as the encryption operation. Time required to complete encrypting and decrypting all different sizes of files is recorded in

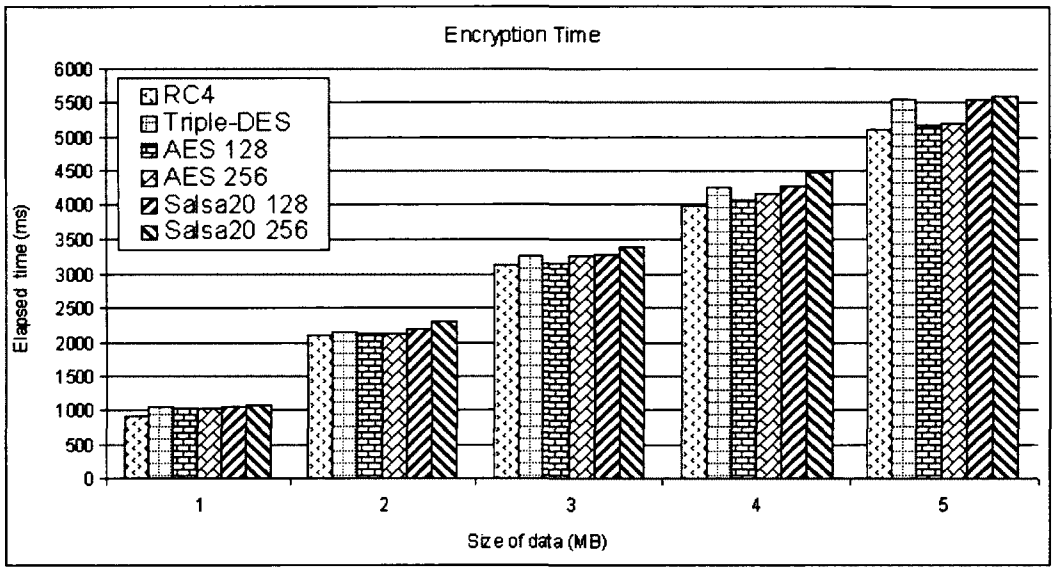


Figure 3.19: File size vs encryption time of different cryptography schemes.

milliseconds. In the presented results, the numeric value following the name of a cryptography algorithm indicates the used key size, except for the ECC, where it indicates the size of the curve in bits. The results are confirmed with 95% confidence level.

Figures 3.18 and 3.19 show time required to encrypt five different sizes of files using different *cryptography schemes* (or crypto scheme as defined in Section 4.3). RC4, Triple-DES, AES 128, AES 256, Salsa20 128 and Salsa20 256 are presented separately in Figure 3.19 for better visibility. On the x-axis is file size and on the y-axis is time required to encrypt the file. From Figure 3.19, we can see for RC4, Triple-DES, AES 128, AES 256, Salsa20 128 and Salsa20 256, the encryption times are always within 500 ms of each other. From Figure 3.18, it is clear that ECC schemes, for any curve size, are more expensive than the other four algorithms. For instance, RC4, Triple-DES, AES 128, AES 256, Salsa20 128 and Salsa20 256 took less than 1500 ms to encrypt the 1 MB binary file, whereas the ECC with a 128-bit curve took more than 8000 ms, 80 s more than any other non-ECC scheme.

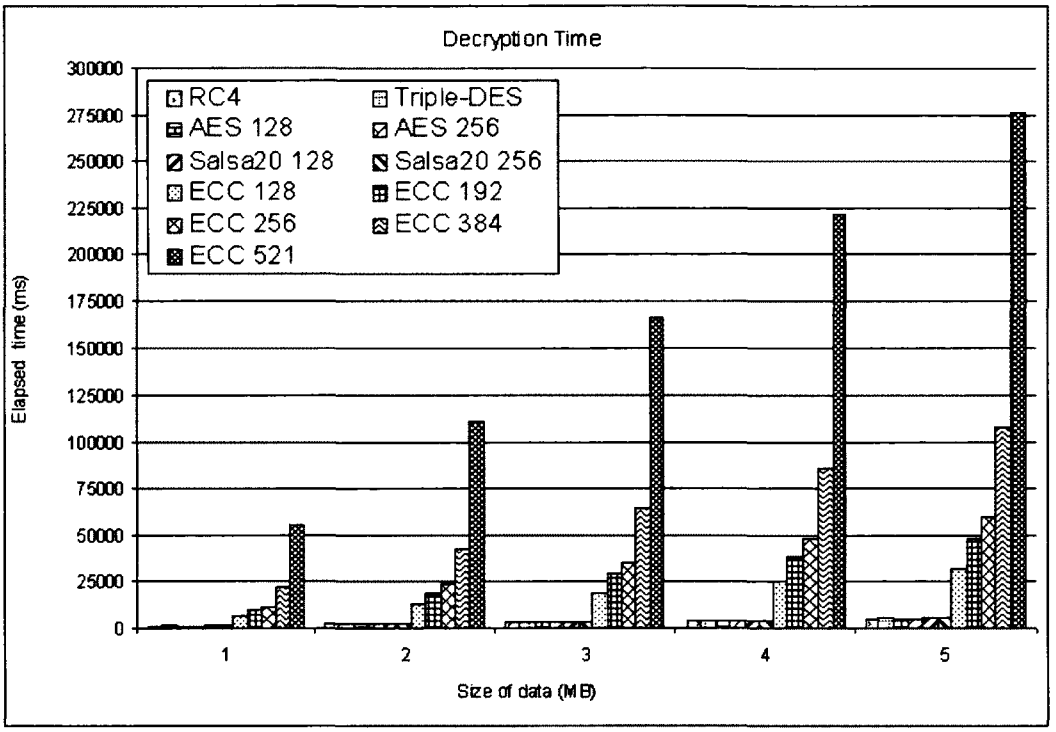


Figure 3.20: File size vs decryption time of different cryptography schemes.

Figures 3.20 and 3.21 show time required to decrypt five different sizes of files using different cryptography schemes. RC4, Triple-DES, AES 128, AES 256, Salsa20 128 and Salsa20 256 are presented separately in Figure 3.21 for better visibility. The decryption operations are performed in the same manner as their encryption counterparts.

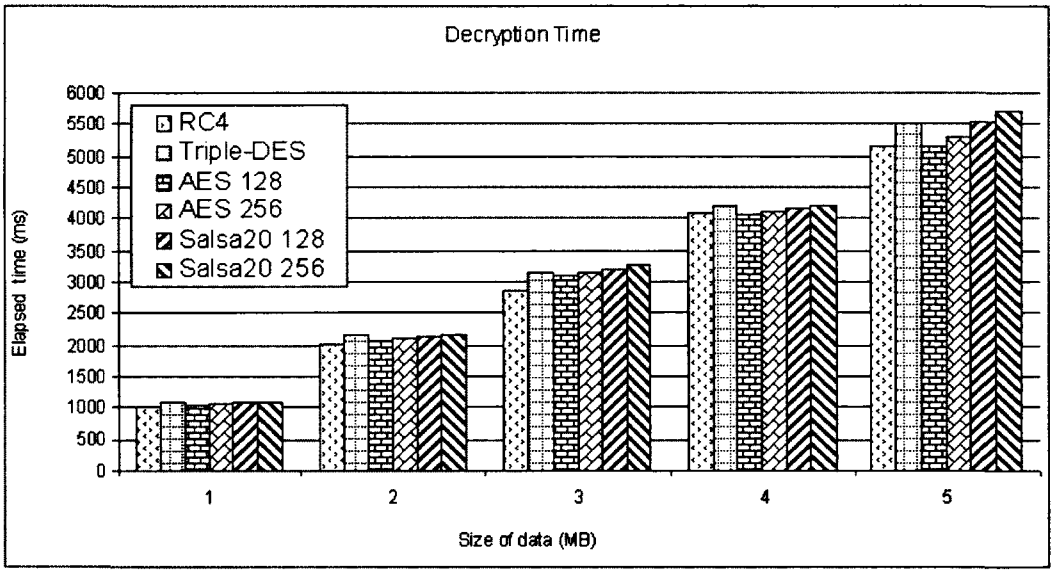


Figure 3.21: File size vs decryption time of different cryptography schemes.

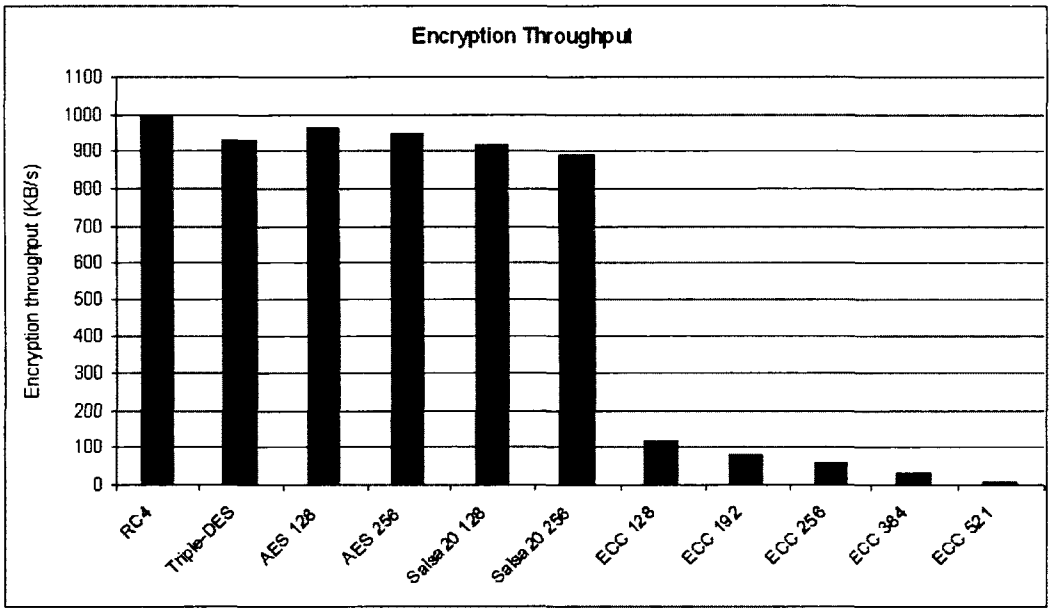


Figure 3.22: Comparison of encryption throughput of different cryptography schemes.

Figures 3.22 and 3.23 show encryption and decryption throughput respectively. Throughput is measured in KBps. As we can see, ECC throughput are much lower compared to RC4, Triple-DES, AES 128, AES 256, Salsa20 128 and Salsa20 256. RC4, DES, AES 128, AES 256, Salsa20 128 and Salsa20 256 throughput fall between 800 - 1000 KBps, whereas the throughput of the highest performing ECC scheme is less than 200 KBps.

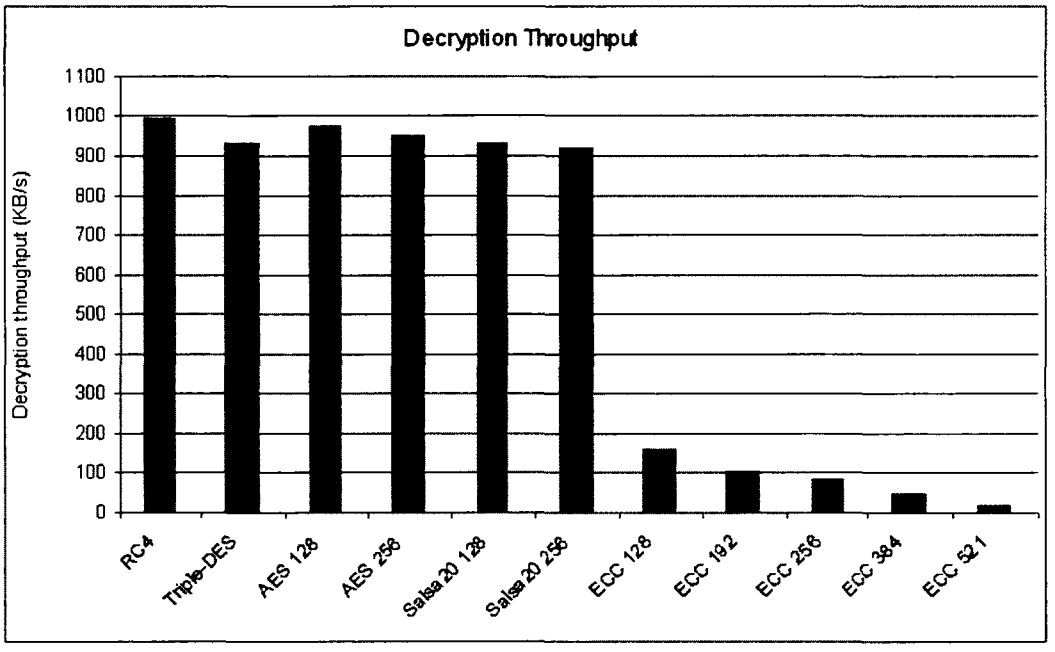


Figure 3.23: Comparison of decryption throughput of different cryptography schemes.

Crypto Scheme	Encrypted Payload Size (byte)
RC4	1024
Triple-DES	1024
AES 128	1024
AES 256	1024
Salsa20 128	1024
Salsa20 256	1024
ECC 128	1077
ECC 192	1093
ECC 256	1109
ECC 384	1141
ECC 521	1177

Table 3.2: Comparison of encrypted payload size of different cryptography schemes.

Table 3.2 contains the encrypted payload size for different cryptography schemes. Here, the original payload size is 1024 bytes. For RC4, Triple-DES, AES 128, AES 256, Salsa20 128 and Salsa20 256, the encrypted payload size is the same as the original payload size. The ECC encryption operation on the other hand, results in increased encrypted payload size. The ECC encrypted payload increases with increased curve size.

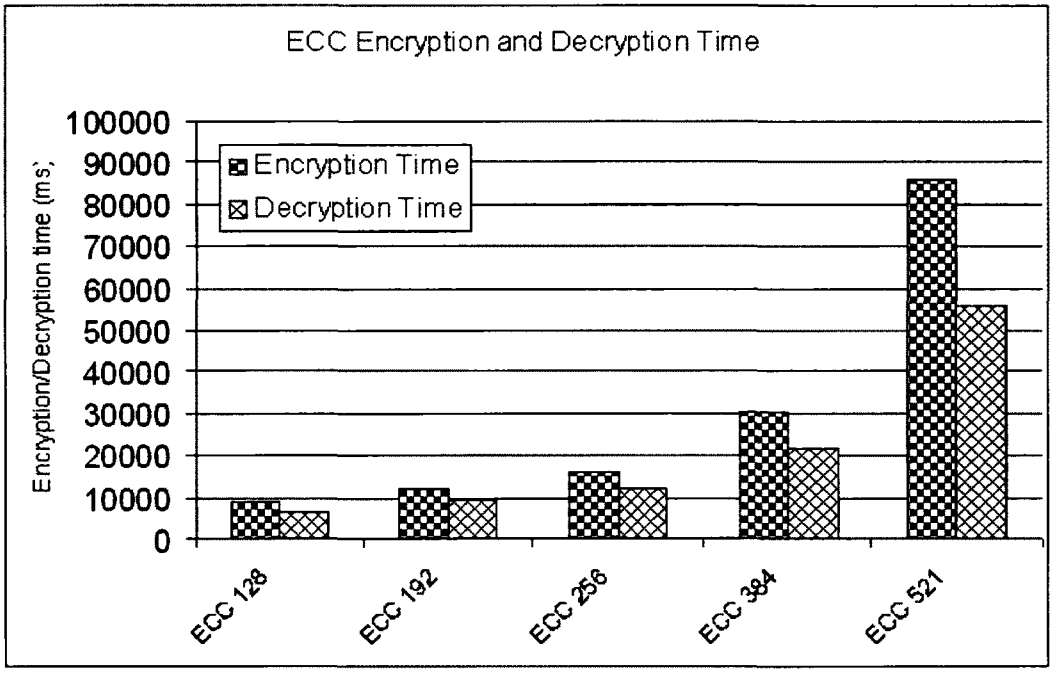


Figure 3.24: Comparison of encryption and decryption time of ECC for different curve sizes.

We observe that, the ECC encryption operation is considerably more delay intensive than the decryption operation. Figure 3.24 compares ECC encryption and decryption times on a 1 MB file for different curve sizes.

Chapter 4

QoS Aware Adaptive Security Scheme (QaASs)

In this chapter, we present an application setup for secured video streaming in ad hoc networks. QoS awareness is a key feature of the security scheme that provides confidentiality at the content level. The application is adaptive to change in computing and network resources. Trade-off between security and QoS parameters is of main interest of the adaptation procedure. We name our proposed framework the *QoS Aware Adaptive Security Scheme* or QaASs in short.

4.1 Overview of QaASs

QaASs aims to provide QoS aware security for real-time multimedia communications over MANETs. In order to meet QoS requirements for a diverse range of multimedia applications, we have chosen to deploy QoS mechanisms at several network layers and MANET components have been chosen accordingly. The building blocks of QaASs can be grouped in three categories:

a. Application and network level QoS mechanisms

b. Network and content level security

c. The adaptive scheme

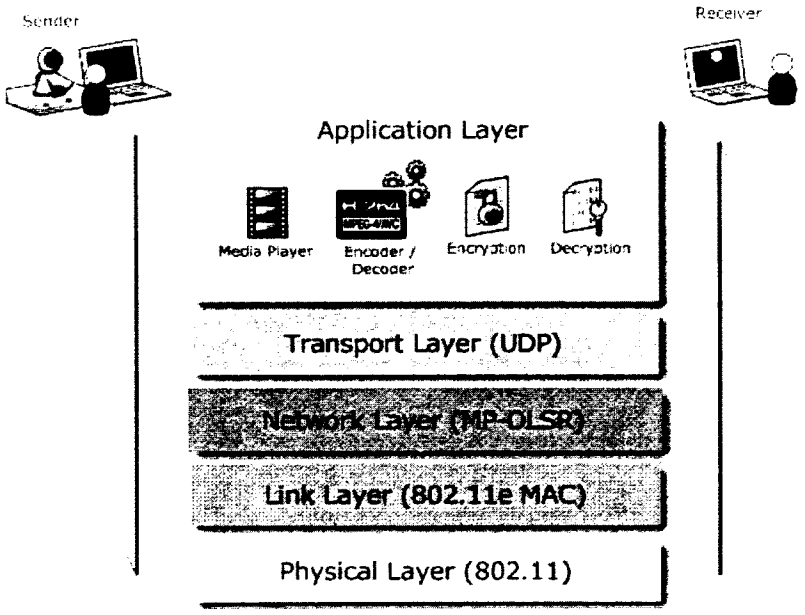


Figure 4.1: Application and network level components.

a. Application and network level QoS mechanisms

Each member station is equipped with devices and applications for real-time multimedia streaming. Video is captured using a video camera and encoded using an MPEG-4 H2.64/AVC encoder. A video streaming application is installed for real-time streaming to the unicast or multicast destinations or for video broadcasting. Each station can stream and receive multiple flows simultaneously. The member stations support other types of traffics as well, e.g., FTP over TCP).

The network utilizes IPv6 [45] for node addressing. Encoded video data are transmitted as IPv6 packets. The streaming application operates on UDP, thus video data are transported as UDP datagram. The network support multipath routing and MP-OLSR [161] is employed as the ad hoc routing protocol.

Each station is equipped with at least one IEEE 802.11 [1] family Wi-Fi interface. The Wi-Fi interface supports IEEE 802.11e [2] admission control enabled MAC. Unlike on-demand reservation based techniques, 802.11e

employs a stateless admission control scheme. Figure 4.1 shows application and network level components using the OSI model.

b. Network and content level security

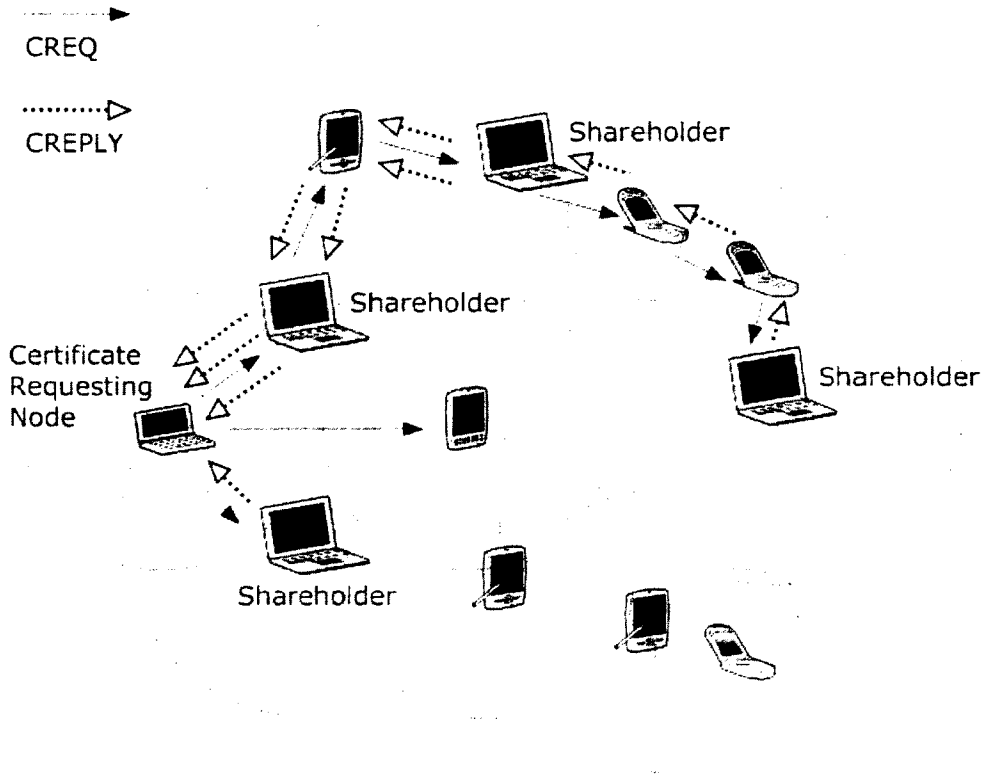


Figure 4.2: CREQ dispatch and CREPLY from shareholders

QaASs offers security both at the network and at the content level. We have chosen public key infrastructure approach to provide authentication at the network layer. QaASs establishes network level security (authentication) in a distributed manner. We follow Dhillon et al.'s [49] proposed distributed CA technique in our work. Dhillon et al. [49] proposed a fully distributed CA for OLSR based ad hoc networks. The certification technique utilizes core concept of Shamir's threshold cryptography [135] and is similar to MOCA proposed by Yi and Kravets [162]. The initial assumption is that the network

contains predefined special nodes called *shareholders*. Shareholders can generate partial signatures. A node joining the network, can obtain a certificate only if it receives at least k partial signatures form k different shareholders. Both MPR and non-MPR nodes can be a shareholder. A shareholder offering service, can be identified from the broadcasted HELLO messages. HELLO messages can only reach one-hop neighbours. It is very much likely that all k shareholders are not within one-hop distance of the certificate requesting node. Information about shareholders is collected by each MPR node from its MPR selector's set and propagated across the network using TC messages. In this way, all the nodes in the network become aware of all the shareholders. Each node maintains shareholders' identity and distance (hop count). A node requests for certificate by sending out CREQ messages to at least k shareholders. Each shareholder reply to the requesting source with CREPLY message that carries a partial signature (Figure 4.2). From k partial signatures, the certificate requesting node can construct a complete signature that is verifiable by the other (authenticated) nodes in the network.

Since, our network aims to provide delay sensitive multimedia services, we have critically taken in consideration the overhead caused by security mechanisms. For example, onion routing [9] like techniques provide anonymity in addition to confidentiality, but, not suitable for delay sensitive applications.

We extend Dhillon et al.'s [49] distributed CA technique to provide routing security in our work. OLSR control messages are modified to carry the digital certificate of the respective control message originator. The complete digital signature contained in the digital certificate is verifiable by a node that has already computed complete digital signature from k partial signatures. A node receiving HELLO message from a neighbour, adds the message originator to the routing table only if the accompanied certificate in the HELLO message is carrying a valid digital signature. MPR nodes only include nodes from the MPR selector's set in TC messages whose digital certificate have been verified. This way, TC messages being propagated across the network, only carry information about verified nodes. A TC message also carries digital certificate of the respective originator MPR node. The constructed routing tables only contain nodes with valid digital certificate. Eventually, computed routes contain authenticated nodes.

Each node maintains the digital certificate of a corresponding node in

the routing table. The digital certificate contains the public key of the associated node, verified by the k shareholders. Hence, when establishing a service session, this public key can be utilized and communicating parties are not required to reverify each others public keys with the shareholders. This key can also be used for encrypting a service key. For example, if the service uses symmetric key infrastructure, the exchanged symmetric key is encrypted using the public key.

The public key infrastructure based routing security offers both authentication and non-repudiation. We assume computational complexity associated with verifying digital certificate is negligible. The proposed technique does not introduce any additional control message, only append additional information to the original OLSR control messages. Elliptic Curve Digital Signature Scheme (ECDSA) [27] can be used for this purpose. ECDSA offers shorter key yet strong security [152]. We assume that rekeying and certificate revocation mechanisms are in place but out of the scope of our current discussion.

Since, computed routes are supposed to be already authenticated, data packets carrying multimedia payload are not authenticated during transmission. If required, payload data are encrypted to provide security at the content level. Multimedia services usually begin with establishing a session. In addition to encrypting the payload, end-to-end secured session is also established.

QaASs aims to support both symmetric key and asymmetric key cryptography for content level security. Both symmetric and asymmetric key cryptography infrastructures require key agreement among involving parties. Since, nodes are verified for a valid digital signature before being added to the routing table, for cryptographic key exchange, we only verify if the service request is coming from a valid node by comparing the node's digital signature with the one stored in the routing table. For already verified nodes, service key establishment can follow any conventional key exchange protocol appropriate for the cryptosystem of interest, e.g., Diffiie-Hellman Key Agreement Protocol [50].

c. The adaptive scheme

In Section 2.2.2.2, we have reviewed literature that address the issue of resource overhead caused by cryptography. We have discussed a number of selective and adaptive mechanisms for encrypting multimedia contents. We also have commented on the reviewed literature and identified their contributions and limitations. To the best of our knowledge, we have not encountered any work that takes a comprehensive approach to provide QoS aware adaptive security for real-time multimedia streaming over ad hoc networks.

Our motivation behind developing an adaptive scheme is that no existing work really addresses, under what circumstances, selective encryption should be applied. Additionally, an number of contributions [103] [145] use nonstandard cryptography techniques and format compliance of encoded video data in some techniques is questionable [4] [145]. Unfortunately, these techniques are unlikely to make into a real world application where genuine security is of interest.

The adaptive scheme defines why, when and how to deploy adaptation. We make an effort to identify video quality metrics influenced by the complexity associated with cryptography operations and also identify possible metrics that can be utilized for making adaptation decisions. The adaptive scheme adapts both cryptography and multimedia service parameters. We describe the adaptive scheme in details in Section 4.3.

4.2 The Role of the Adaptation Mechanism

In Section 3.7.8, we compared run-time performance of different cryptography schemes. Since, video streaming is delay sensitive, in order to meet desired QoS requirements, performance of the cryptography operation is of significant importance.

We begin with describing scenarios where adaptation is applicable. The video streaming application is consists of four key processes, namely $ECD_{H.264}$, ENC_{CRP} at the streaming source and $DCD_{H.264}$ and DEC_{CRP} at the receiving end. $ECD_{H.264}$ and $DCD_{H.264}$ are the H.264/AVC encoder and decoder processes respectively, and ENC_{CRP} and DEC_{CRP} are the encryption and

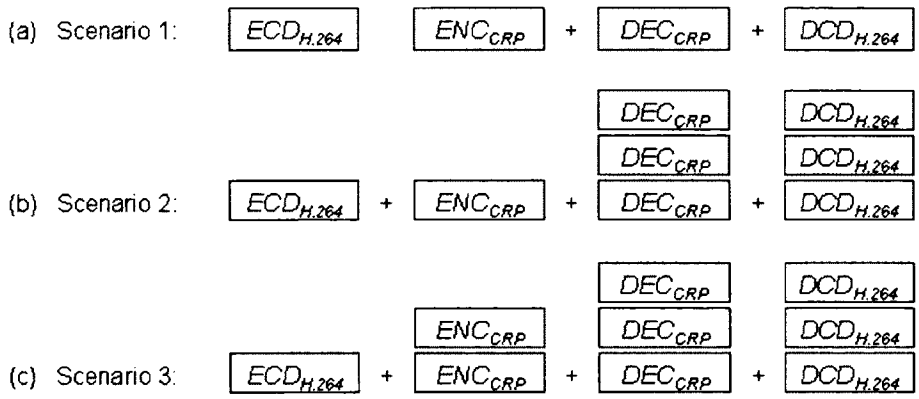


Figure 4.3: Three communication scenarios: Processes.

decryption processes for the cryptography scheme CRP. Let us assume that all processes have equal priority. For one-on-one video communications, a station would require to execute all four processes simultaneously and system resource would be equally shared by four processes (Figure 4.3 (a)). If there is four way real-time communications, i.e., each station is communicating with three other stations, a total of eight processes would share the system resource (Figure 4.3 (b)). Similarly, Figure 4.3 (c) shows that a station is engaged in real-time communications with three parties and providing one video-on-demand (VOD) service. In the latter case, video data is already encoded and only needs to be encrypted.

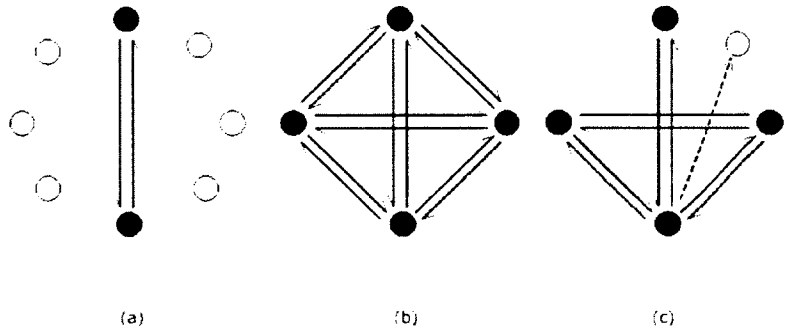


Figure 4.4: Three communication scenarios: Network Flows.

Figure 4.4 shows network resource utilization by the three previously

described scenarios. In Figure 4.4 (a), there are only two flows, so network resource (e.g., available bandwidth) is shared by two flows. In Figure 4.4 (b) and (c), the same resource is shared by 12 flows and nine flows respectively. In the next section, we provide simulation results which reflect the above discussed concepts.

4.2.1 Encrypted Video Traffic

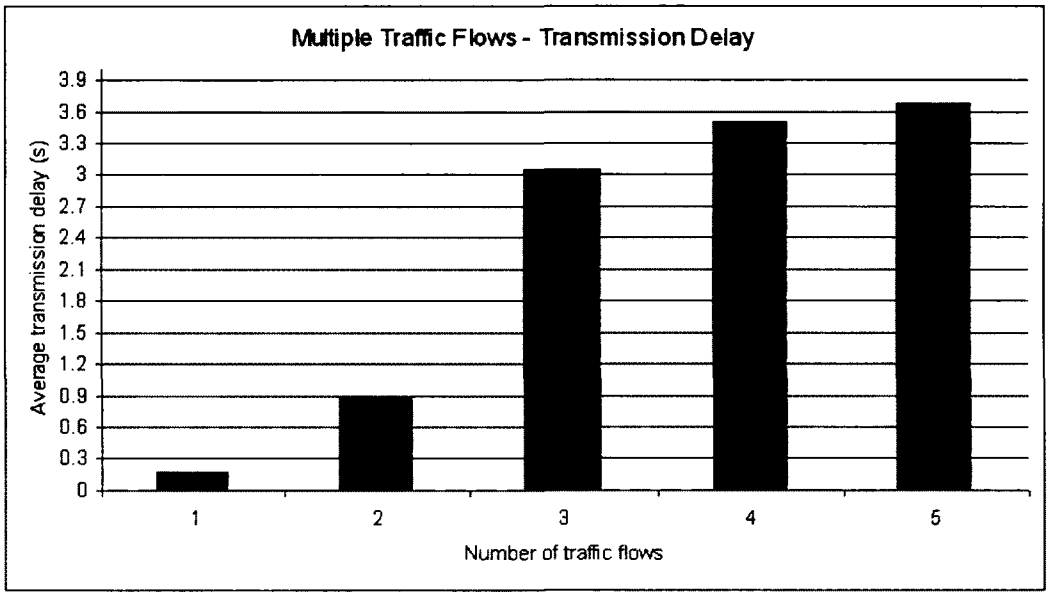


Figure 4.5: Comparison of transmission delay of 1-5 video flows.

Figure 4.5 compares end-to-end transmission delay experienced at a station for different numbers of video traffic flows. The results are obtained in the identical environment. The simulation environment and implementation details are available in Chapter 5. Table 5.1 lists the network simulation parameters. We carry out simulation in a grid network with 36 nodes. MP-OLSR is used as the ad hoc routing protocol. Video data is encrypted using ECC with a 384-bit curve. The results are confirmed with 90% confidence level.

Figure 4.6 compares number of packets successfully transmitted for different numbers of video flows. Figure 4.7 compares throughput of the cryp-

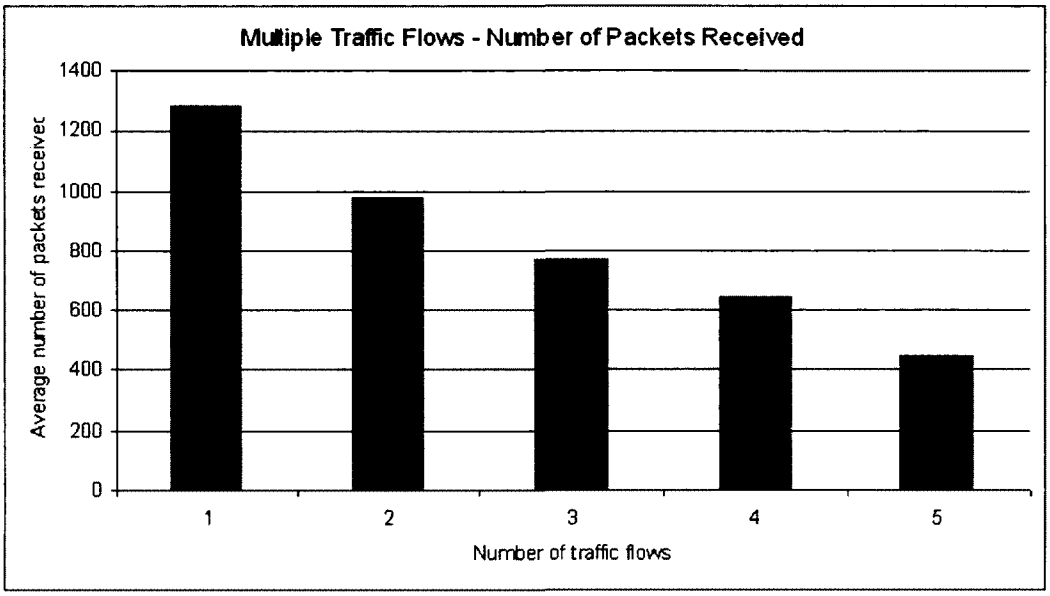


Figure 4.6: Comparison of number of packets successfully transmitted.

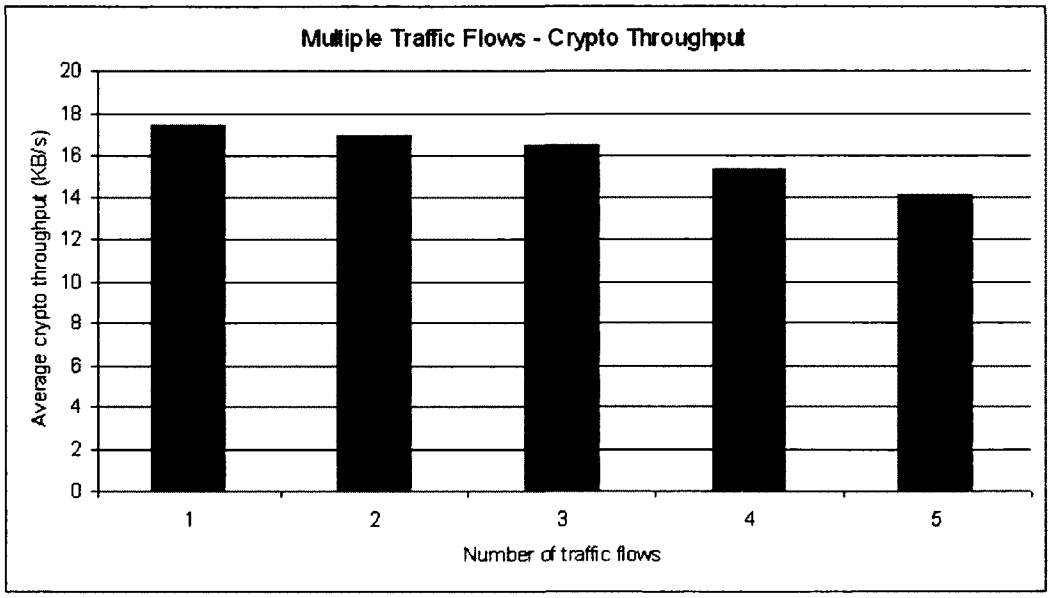


Figure 4.7: Comparison of cryptography process throughput.

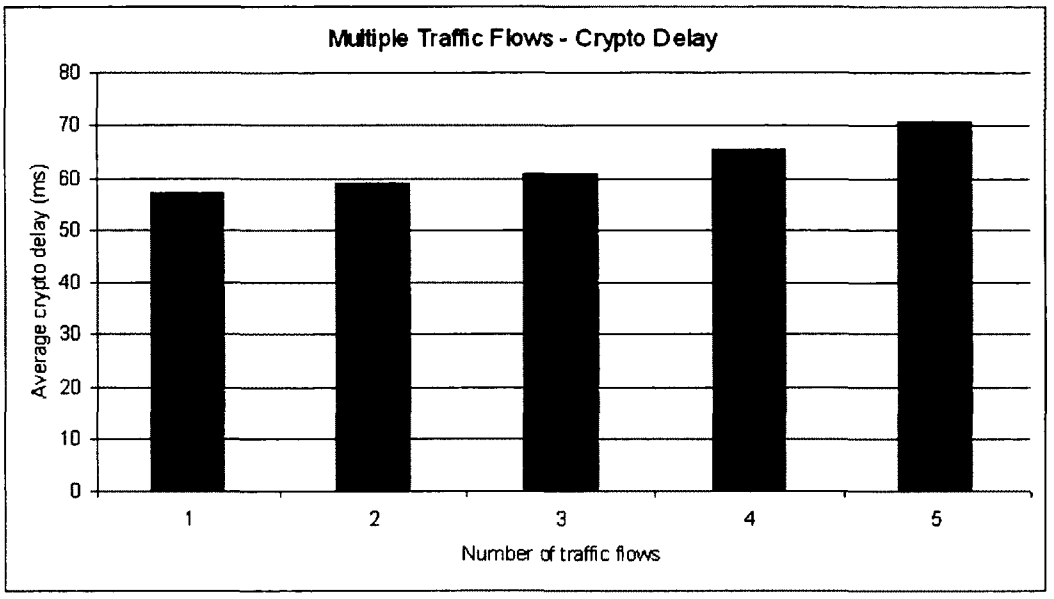


Figure 4.8: Comparison of cryptography delay of 1-5 video flows.

tography process for different numbers of video flows. Figure 4.8 compares cryptography delay experienced at a station for different numbers of video flows. Cryptography delay for each video data unit differs by 13 ms across a single video flow to five video flows. Over the course of a second, for a 30 fps video, this time would be magnified by 30 times causing significant playback delay. The adaptation mechanism of QaASs is designed to deal with the aforementioned issues.

4.2.2 Selective Encryption

In this section we present evaluation of selective encryption. The results are obtained in the identical environment. The simulation environment and implementation details are available in Chapter 5. Table 5.1 lists the network simulation parameters. We carry out simulation in a grid network with 36 nodes. MP-OLSR is used as the ad hoc routing protocol. We use ECC with a 384-bit curve for encrypting video data. The results are confirmed with 95% confidence level.

Figure 4.9 compares cryptography delay for three encryption scenarios.

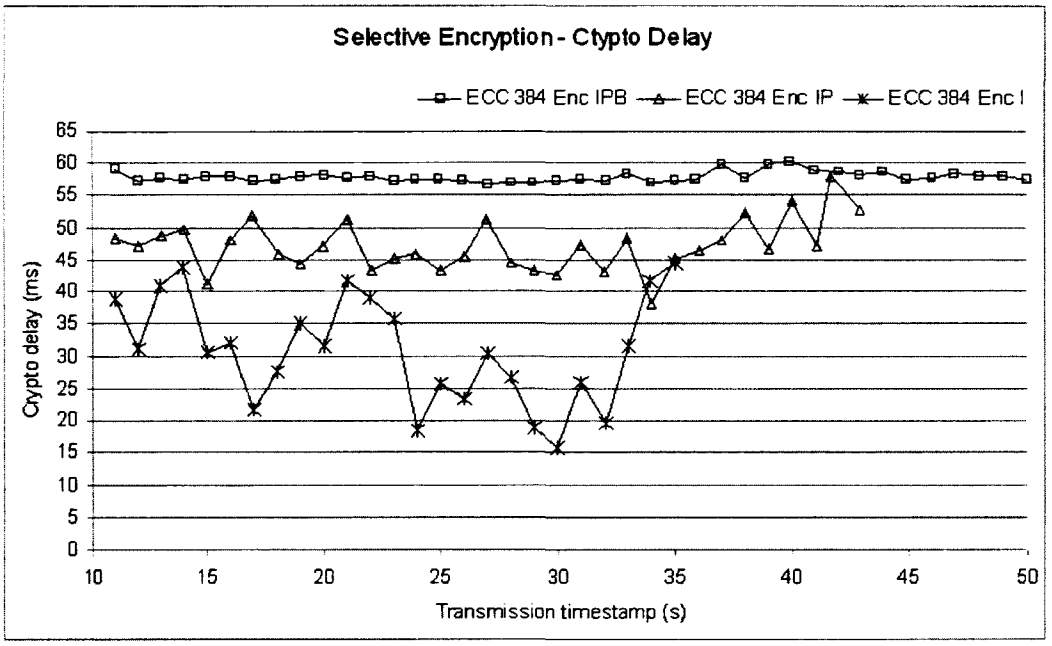


Figure 4.9: Comparison of cryptography delay for selective encryption.

In the first scenario, we encrypt all types of coded frames in a H.264/AVC video sequence. In the second scenario, we encrypt I and P-frames and in the third scenario only the I-frames are encrypted. The results show that there is an average 30 ms difference in cryptography delay per frame from scenario one to scenario three. The difference is about 15 ms for scenario two. Furthermore, in the first scenario, it took about 50 s to transfer the entire video, whereas in the second scenario the time is 42 s and 36 s for scenario three. Figure 4.10 compares effective packet transmission rate (per second) experienced by the receiver. Scenario three achieves the highest packet rate at an average 42 packets per second while the first scenario manages just 30 packets per second. We, therefore, think that selective encryption can be a possible candidate for the adaptation mechanism in QaASs.

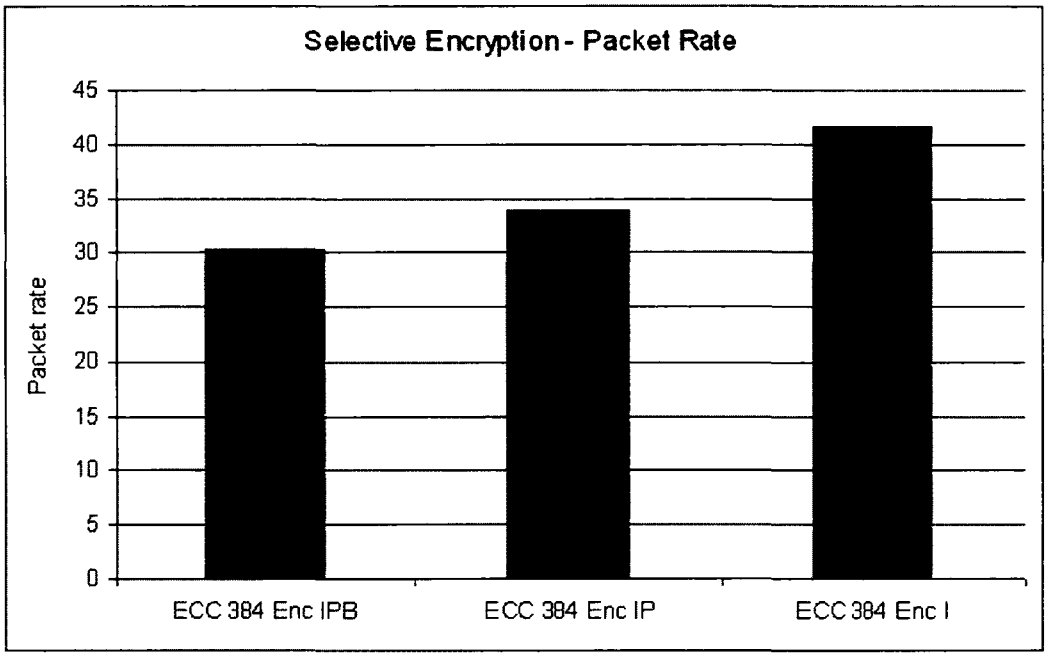


Figure 4.10: Comparison of packet rate for selective encryption.

4.2.3 Video Frame Rate

Video compression techniques such as H.264/AVC, allow to encode video data at different frame rates. A higher frame rate comprises a larger number of frames for a fixed duration. The same video for the same duration can be produced with less number of frames but would compromise playback quality. As we have seen in the previous section, the more number of frames have to be encrypted the higher the over all delay and lower the effective packet rate, thus actual video frame rate. We, therefore, think that video frame rate can be considered as a metric for the adaptation mechanism in QaASs.

4.3 QaASs Adaptive Scheme

QaASs adapts both security and streaming behaviour to cope with change in system and network resources. We explain the adaptation mechanism in

the context of live video streaming.

Crypto Scheme:

A *crypto scheme* is a unique combination of a cryptography algorithm and its properties such as internal state size, block size, number of rounds, mode of operation and key length. For example, AES having a 128-bit key and AES having a 256-bit key are two unique crypto schemes. RC5 [127] with a 64-bit block size is a different crypto scheme from RC5 with a 128-bit block size. ECC with a 128-bit curve and ECC with a 384-bit curve are two separate crypto schemes. In this thesis, a crypto scheme always refers to a unique combination of a cryptography algorithm and its set of properties.

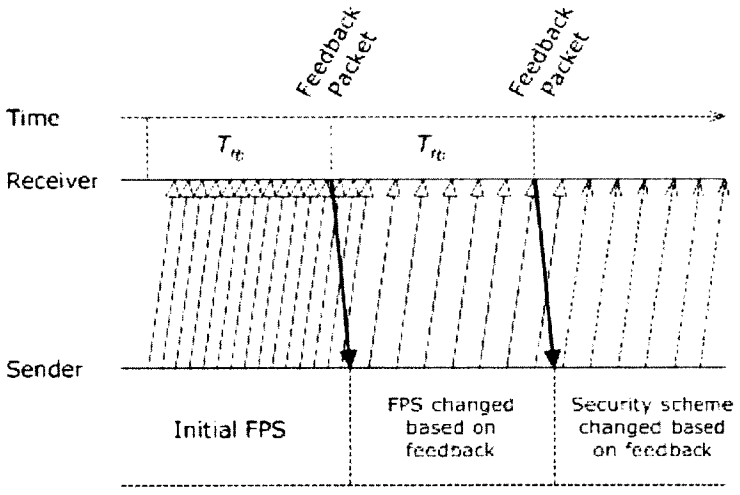


Figure 4.11: Feedback packet.

Feedback Packet:

The receiver of real-time video provides periodic feedback about reception to the streaming source. A feedback packet PKT_{fb} is sent by the receiver every T_{fb} interval (Figure 4.11). T_{fb} is measured in seconds and customizable before the beginning of a service session. PKT_{fb} only contains a parameter set. Therefore, it can be sent over a guaranteed service like TCP. Table 4.1

shows an example of PKT_{fb} parameter set.

Parameter	Symbol	Unit
Average transmission delay	T_{tr_d}	milliseconds
Average playback frame rate	FPS_{pb}	frames/second
Frame delay jitter	J_{f_d}	milliseconds
Decryption throughput	TP_{dec}	KBps
Frame loss percentage	F_{loss}	%(percentage)

Table 4.1: Feedback packet parameters set.

- T_{tr_d} is the average transmission delay.
- FPS_{pb} is the average frame rate achieved at the receiver's end, i.e., playback FPS.
- J_{f_d} is the frame delay jitter experienced at the receiver's end. Delay jitter indicates the deviation from the average or median delay. One way of measuring delay jitter is calculating median absolute deviation.
- TP_{dec} is the decryption throughput, a measure of performance of a decryption process (explained in the next section).
- F_{loss} is the measure of frames lost during transmission or treated obsolete by the decoder.

We explain the adaptation scheme presenting four different scenarios. Both security requirements and streaming properties are application dependent and are user defined. We present four adaptation options to explain the operational procedure of the QaASs's adaptation scheme.

Adaptation Option One:

Conditions: FPS is fixed and all video data must be encrypted.

Since, our network aims to support a diverse range of devices, different stations have different processing capability; therefore, the throughput of crypto schemes varies across stations. Each station maintains a performance

Crypto Scheme ID	Algorithm	Size of Internal State or Block Size (bits)	Key Size or Curve Size (bits)	Encryption Through-put (KBps)	Decryption Through-put (KBps)
1	RC4	2064	256	993.80	996.79
4	Triple-DES	64	168	930.50	931.73
5	AES	128	128	963.57	972.95
6	AES	128	256	950.87	952.50
7	Salsa20	128	512	921.23	933.66
8	Salsa20	256	512	895.35	919.56
9	ECC	N/A	256	61.93	84.36
10	ECC	N/A	384	33.11	46.72

Table 4.2: Cryptography algorithm performance profile.

profile of crypto schemes, CRP_{prf}. CRP_{prf} contains encryption and decryption throughput in KBps for different crypto schemes. Throughput values are pre-populated in an ideal execution environment. Throughput values in Table 4.2 are obtained following the same procedure as discussed in Section 3.7.8. Table 4.2 is an example of information stored in CRP_{prf}.

Ideally, CRP_{prf} is an XML file. During the service setup phase, communicating parties exchange their CRP_{prf}'s. Adaptation option one utilizes CRP_{prf} for its operational procedure.

In Table 4.2,

- Encryption throughput is the amount of data encrypted by the crypto scheme per second, measured in KBps.
- Decryption throughput is the amount of data decrypted by the crypto scheme per second, measured in KBps.

Lets assume, u is the streaming source and v is the receiver.

The maximum size of a video frame is n bytes.

T_{enc} is the time required to encrypt n bytes at station u. n_{enc} is the size of the encrypted payload and $n_{enc} \geq n$.

T_{dec} is the time required to decrypt n_{enc} bytes at station v.

Here, time is measured in milliseconds.

$T_z = T_x + T_y$, is the playback time at the receiver's end for a video frame that was originally available at the source at time T_x. T_y is the maximum allowed playback delay for the video frame.

So, we can write,

$$T_{sch} = T_x + T_{enc} \tag{4.1}$$

where T_{sch} is the scheduled transmission time for the video frame that was originally available at time T_x.

and

$$T_y \geq T_{enc} + T_{tr\_d} + T_{dec} \tag{4.2}$$

where T_{tr_d} is the transmission delay measured in milliseconds.

In order to meet QoS requirements (e.g., maximum allowed playback delay), we can choose an encryption algorithm from $CRP_{profile}$ that satisfies Equation 4.2. From Equation 4.2, we can see that the playback delay is composed of two independent entities, the transmission delay, T_{tr_d} and the cryptography delay, $(T_{enc} + T_{dec})$.

We define a parameter called *Crypto Threshold*, $CRP_{threshold}$. The value of $CRP_{threshold}$ indicates the maximum time allowed for a *video data unit* to spend in cryptography operations (encryption and decryption to be exact). A video data unit could be a single video frame, a GOP or could be one second worth of video data. In order to cope with unexpected transmission delay and jitter, streaming applications incorporate a technique, called *jitter buffer* [111]. Moreover, introducing additional delay at the beginning of playback, to allow the decoder to receive enough frames, could help to mitigate the effect of unexpected transmission delay and jitter. The goal is to avoid choppiness or prevent the video from stalling during playback. The

value of crypto threshold is determined with the above two techniques in consideration. How to determine the value of $CRP_{threshold}$, in conjunction with other related parameters, is a problem in its own right and we leave it as an open problem for the time being and consider as a candidate for future research.

So we can say, the maximum allowed playback delay, T_y is a function of $CRP_{threshold}$, jitter buffer delay properties and introduced initial playback delay. Conversely, the value of $CRP_{threshold}$ is chosen to satisfy Equation 4.2.

For adaptation option one, we consider that the value of $CRP_{threshold}$ indicates the maximum time allowed for a video frame to spend in cryptography operations, i.e., the sum of encryption time and decryption time, $(T_{enc} + T_{dec})$. Hence,

$$CRP_{threshold} \geq T_{enc} + T_{dec} \tag{4.3}$$

For each streaming session, the sender and receiver maintain the average encryption and decryption throughput respectively, over a predefined duration (e.g., one second for a 30 fps video). TP_{enc} is the average encryption throughput per frame measured in KBps (Equation 4.4).

$$TP_{enc} = \frac{\sum_{t=0}^{t=1s} TP_{enc\_frame}}{\sum_{t=0}^{t=1s} frame} \tag{4.4}$$

In Equation 4.4, TP_{enc_frame} is the encryption throughput for each frame, $\sum_{t=0}^{t=1s} frame$ is the total number of frames encrypted over the predefined duration, which is assumed to be 1s. It is possible that, $\sum_{t=0}^{t=1s} frame = GOP_{length}$, where GOP_{length} is the GOP length in the sourced video.

The feedback packet, PKT_{fb}, originated by the receiver, informs the sender about average decryption throughput per frame, TP_{dec}. We calculate TP_{dec} from Equation 4.5. In Equation 4.5, TP_{dec_frame} is the decryption throughput for each individual frame. Hence, $\sum_{t=0}^{t=1s} TP_{dec\_frame}$ is the cumulative decryption throughput for $\sum_{t=0}^{t=1s} frame$ frames. TP_{dec} calculation considers decodable frames in a received GOP.

$$TP_{dec} = \frac{\sum_{t=0}^{t=1s} TP_{dec\_frame}}{\sum_{t=0}^{t=1s} frame} \tag{4.5}$$

We calculate, average encryption time per frame, T_{enc} from Equation 4.6 and average decryption time per frame, T_{dec} from Equation 4.7.

$$T_{enc} = \frac{n_{avg}}{TP_{enc}} \tag{4.6}$$

$$T_{dec} = \frac{n_{avg}}{TP_{dec}} \tag{4.7}$$

where n_{avg} is the average frame size in bytes.

Up-on receiving a PKT_{fb}, the streaming source verifies if Equation 4.3 is satisfied. In case Equation 4.3 is not satisfied, FPS cannot be altered, nor can we employ selective encryption. The only variables here are T_{enc} and T_{dec}.

The source have the receiver's $CRP_{profile}$. Using Equations 4.6 and 4.7, from the recipient's $CRP_{profile}$, the source chooses a crypto scheme for which Equation 4.3 is satisfied.

Our literature review and evaluation show that, in most cases, a crypto scheme with higher throughput happens to provide lower security. By switching to a higher throughput crypto scheme, to some extent, we are compromising security. In order to address this issue, we employ rekeying. Multimedia applications are delay sensitive. As we have seen before, cryptography key management is a complex procedure in ad hoc networks. Rekeying as well is an expensive procedure and require additional communications among communicating parties.

We only present a rekeying mechanism for public key cryptosystem, ECC to be specific. Key agreement in ECICS is achieved using ECDH. We assume, communicating pairs exchange and agree upon ECC domain parameters during the session setup phase. The elliptic curve E and a point P on E are agreed upon during the session setup phase and are not changed by the rekeying procedure. The source and destination generate large enough random integers a and b respectively. The source computes a point aP on E and sends it to the destination. The destination computes a point bP on E and sends it to the source. The source and destination computes $a(bP)$

and $b(aP)$ respectively. This is to note that the sizes of aP and bP are the same as the size of the point P. The shared secret x is the x-coordinate of the point abP on the curve E. A symmetric key can be derived from x for message encryption.

We propose two ECDH based rekeying techniques:

1. The source generates a new random integer a' and computes $a'P$ and sends it to the destination. Since, the source have already received bP during the session setup phase, the destination is not required to resend bP. Both the source and destination can compute $a'(bP)$ and $b(a'P)$ respectively, thus a new shared secret x'. A new symmetric key can be derived from x'. For this approach, we would only require one way communications.

2. The source generates a new random integer a' and computes $a'P$ and sends it to the destination. The destination generates a new random integer b' and computes $b'P$ and sends it to the source. Up-on receiving $b'P$, the source can compute $a'(b'P)$. Up-on receiving $a'P$, the destination can compute $b'(a'P)$. Therefore, a new shared secret x' can be computed. A new symmetric key can be derived from x'. For this approach, we would require two way communications.

It is possible that information exchanged for rekeying be sent separately or if possible, piggyback on data traffic. The latter reduces traffic overhead. Another important issue is the rekeying frequency. A higher rekeying frequency would require exchange of a lot more information compared to a low rekeying frequency. Since the amount of information exchanged influences network resource utilization, this is of important consideration. Rekeying frequency should be chosen in such a way that it does not cause serious traffic overhead that interferes with actual multimedia traffic.

Adaptation Option Two:

Conditions: Crypto scheme and FPS are fixed. Encrypting frames containing inter coded macroblocks (e.g., P and B-frames) are optional.

The more frames are encrypted, the higher the overall delay overhead is. Frames received out-of-order, due to delay, are often discarded by the

decoder and therefore, would only contribute to resources exhaustion and bandwidth wastage. In Section 4.2.2, we have seen how selective encryption influences overall transmission delay. Here, selective encryption is an option for adaptation.

In theory, if no I-block is present, in H.264/AVC, P and B-frames cannot be decoded unless the reference I-frame information is available. Information present in P and B-frames is useless without the reference I-frame. Hence, I-frames are of the most importance and therefore, must be encrypted.

Options	I-frame	P-frame	B-frame
i	e	e	e
ii	e	e	x
iii	e	x	x

Table 4.3: H.264/AVC selective encryption options.

For adaptation option two, we take advantage of the above property of H.264/AVC and make encrypting inter coded frames, i.e., P and B-frames optional. Along with I-frames, only P-frames may be chosen for encryption. Another option would be not to encrypt either P or B-frames at all. By choosing less frames for encryption, the overall delay overhead due to encryption and decryption can be significantly reduced. For example, if a GOP contains one I-frame, three P-frames and six B-frames, we can choose to encrypt only one I-frame instead of 10 frames. Table 4.3 summarizes the options for the discussed selective encryption technique. In Table 4.3, 'e' refers to encrypted and 'x' refers to not encrypted.

The decision regarding selecting frames for encryption is made based on Equation 4.3. For each streaming session, the sender and receiver maintain time spent on the encryption and decryption operations, over a predefined duration (e.g., one second for 30 fps video). The feedback packet, PKT_{fb}, originated by the receiver, informs the sender about effective decryption throughput, TP_{dec}. Up-on receiving PKT_{fb}, the streaming source verifies if Equation 4.3 is satisfied. In case Equation 4.3 is not satisfied, we can employ selective encryption. The adaptation decision is applied on a trial basis. For example, initially, all I, P and B-frames are encrypted. Up-on

receiving PKT_{fb}, the source verifies if Equation 4.3 is satisfied. If Equation 4.3 is not satisfied, the source decides not to encrypt the B-frames. On the next reception of PKT_{fb}, the source again verifies if Equation 4.3 is satisfied. If Equation 4.3 is still not satisfied, the source decides not to encrypt P and B-frames.

Adaptation Option Three:

Conditions: Crypto scheme is fixed and all the transmitted video data must be encrypted. It is optional to transmit video frames containing inter coded macroblocks (e.g., P and B-frames).

Adaptation option two reduces delay overhead by not encrypting frames containing inter coded macroblocks. Adaptation option three considers a more restricted scenario, where crypto scheme is fixed and all the transmitted data must be encrypted. However, transmission of frames containing inter coded macroblocks (e.g., P and B-frames) is optional. If the source wants to send P and B-frames, they must be encrypted.

Discarding non-inter coded frames results in reduced amount of encrypted data, i.e., reduction in overall cryptography operations; thus, reduced cryptography delay overhead. Therefore, we can gain streaming performance in expense of quality of experience.

Up-on receiving a feedback packet, PKT_{fb}, the streaming source verifies if Equation 4.3 is satisfied. In case Equation 4.3 is not satisfied, the adaptation decision is applied on a trial basis. For example, initially, all I, P and B-frames are encrypted and transmitted. Up-on receiving a PKT_{fb}, the source verifies if Equation 4.3 is satisfied. If Equation 4.3 is not satisfied, the source decides not to encrypt and send the B-frames. On the next reception of PKT_{fb}, the source again verifies if Equation 4.3 is satisfied. If Equation 4.3 is still not satisfied, the source decides not to encrypt and send P and B-frames.

Adaptation Option Four:

Conditions: Crypto scheme is fixed and all the transmitted video data must be encrypted. Variable frame rate (FPS) is allowed.

Another option for adaptation is adapting the frame rate. A high FPS value means more frames have to be encrypted; thus, higher cryptography overhead.

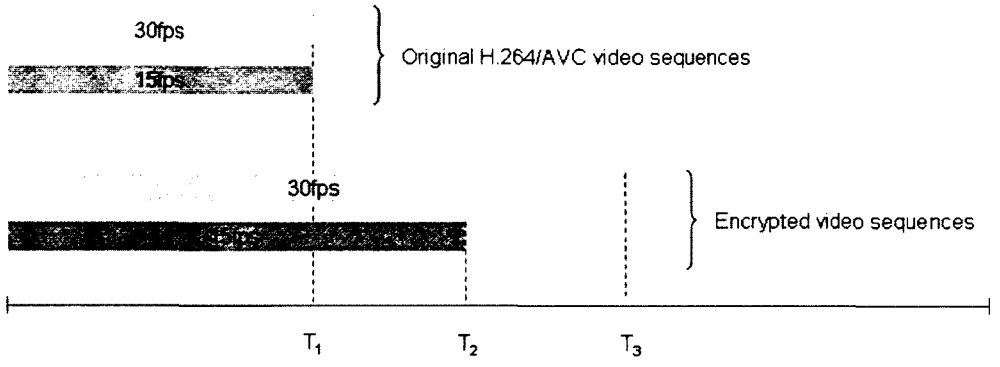


Figure 4.12: Transfer time for encrypted video sequences at different FPS.

In Section 4.2.2, we have seen how a computationally expensive cryptography algorithm can reduce effective frame rate. A high FPS video in this case would increase the number of frames arriving out-of-order. Figure 4.12 illustrates this issue with an example. We have the same T_1 second long video sequence coded at 30 fps and 15 fps. Assuming network latency is negligible, the transfer time for both video is T_1. Lets assume that the used crypto scheme causes the 30 fps video to be transferred by time T_3, twice the transfer time required by the unencrypted videos. Number of frames in the 30 fps sequence is twice that of in the 15 fps sequence. Therefore, in theory, the 30 fps sequence spends twice the amount of time in cryptography operations compared to the 15 fps sequence and the Equation 4.8 is always true.

$$T_3 > \partial T_2 \tag{4.8}$$

where the factor ∂ is a function of cryptography delay.

For adaptation option four, we adapt FPS in order to improve overall transfer time. Here, we replace TP_{dec} in PKT_{fb} by cumulative decryption time, $\sum T_{dec}$, as in Equation 4.9.

$$\sum T_{dec} = \sum_{t=0}^{t=1s} T_{dec\_frame} \tag{4.9}$$

The source also maintains cumulative encryption time, $\sum T_{enc}$, as in Equation 4.10.

$$\sum T_{enc} = \sum_{t=0}^{t=1s} T_{enc\_frame} \tag{4.10}$$

We rewrite Equation 4.3 as,

$$CRP_{threshold} \geq \sum T_{enc} + \sum T_{dec} \tag{4.11}$$

In adaptation option four, we verify crypto threshold using Equation 4.11. Up-on receiving a feedback packet, PKT_{fb}, the streaming source verifies if Equation 4.11 is satisfied. In case Equation 4.11 is not satisfied, the adaptation decision is applied on a trial basis. For example, the initial FPS is 30. Up-on receiving a PKT_{fb}, the source verifies if Equation 4.11 is satisfied. If Equation 4.11 is not satisfied, the source reduces the frame rate to 25 fps. On the next reception of a PKT_{fb}, the source again verifies if Equation 4.11 is satisfied. If Equation 4.3 is still not satisfied, the source selects 20 fps and so forth, only till acceptable minimum FPS. Although, reduced FPS would effect quality of experience, however, we would be able to maintain the required level of security.

Chapter 5

Simulation and Results

In this chapter, we describe the simulation environment that we have used to verify our proposal, present simulation results and comment on the results.

5.1 Simulation Environment

5.1.1 Network Simulator

We have chosen to use Network Simulator (NS) [109] to verify our proposal. NS is a discrete-event simulator developed for communications network research. NS is open source and is maintained by a number of collaborating research groups. NS facilitates simulation for both wired (e.g., Ethernet, ATM) and wireless (e.g., WLAN, WiMAX, Satellite) communications. NS provides support for application layer protocols (e.g., RTP, SIP), transport layer protocols (e.g., TCP, UDP), unicast and multicast protocols, link layer and MAC protocols (e.g, IEEE 802.11). In addition to network protocols, NS offers support for network topology, traffic (e.g., CBR, VBR), mobility and packet loss models.

Core libraries of NS are written in C++. NS offers end-user programming in both C++ and in OTcl. OTcl scripts are typically used for executing end-user commands. User defined applications (e.g., routing protocols) and traffic generation models are commonly written in C++. Network Animator (NAM), an accompanying graphical tool, offers visual support for simulations. A common simulation practice in NS is to follow a trace driven

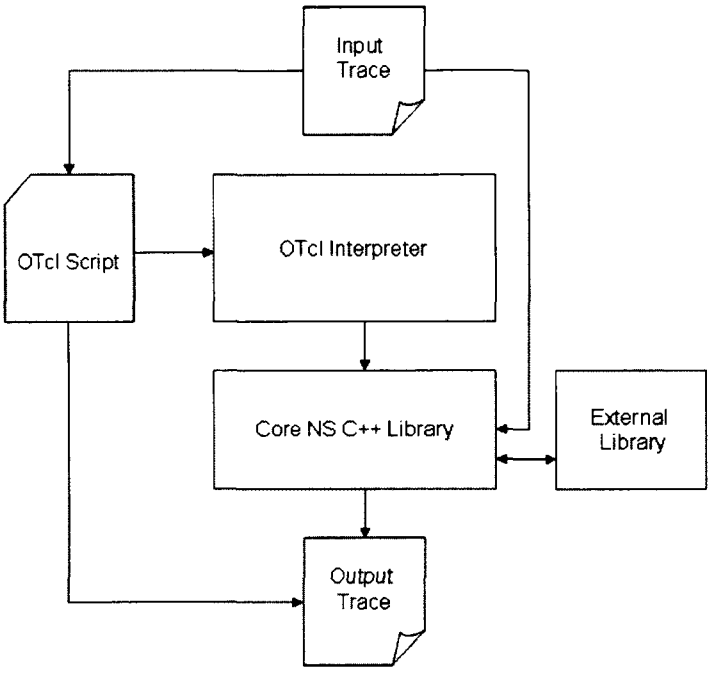


Figure 5.1: Overview of NS.

approach. Figure 5.1 shows a high level view of NS components and their communications relations (details are available in the next section). Typically, network topology and traffic information are defined in trace files which are used as the blueprint for the simulation. The simulation results can be accumulated in trace files, which can be used for visualization using NAM.

5.1.2 Implantation Details

We implemented and carried out simulations on a IBM ThinkPad® T43 mobile workstation equipped with a single 1.86 GHz Intel® Pentium® M processor and 1.50 GB physical memory.

For our implementation, we have used NS version 2.28 using Tcl 8.4.5 and OTcl 1.9, compiled using GCC 3.31 supporting POSIX thread.

Figure 5.2 shows the architecture of a wireless node in NS. In NS, each component is identified as an *Agent*, e.g., routing agent, transport layer agent

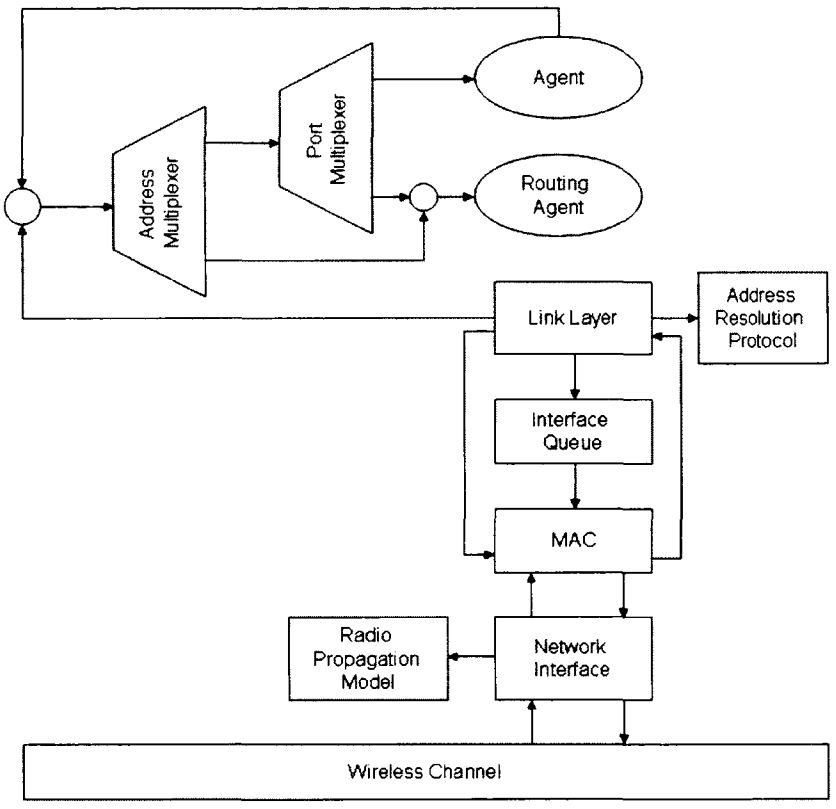


Figure 5.2: Architecture of a wireless node in NS.

etc. The appropriate agent must be attached to a node (i.e., included in the node properties), in order to use the service provided by the agent. An NS node communicates with each component through unique ports. Looking at the packet header, address multiplexer decides if a packet should be handed over to the routing agent or an upper later protocol (an agent), which in turn forwards the packet to the appropriate application port.

In Figure 5.2, the routing agent carries out the route discovery, route maintenance, forwarding and packet routing. We have used an NS-2 implementation of the MP-OLSR [161] which is developed based on UM-OLSR [128]. UM-OLSR is implemented according to the RFC 3536 specifications. In Figure 5.2, at the link layer, there is a separate module that implements the MAC protocol. The IEEE 802.11e (EDCA) admission control mechanism

is implemented in this module. We use an available NS-2.28 implementation of 802.11e [3].

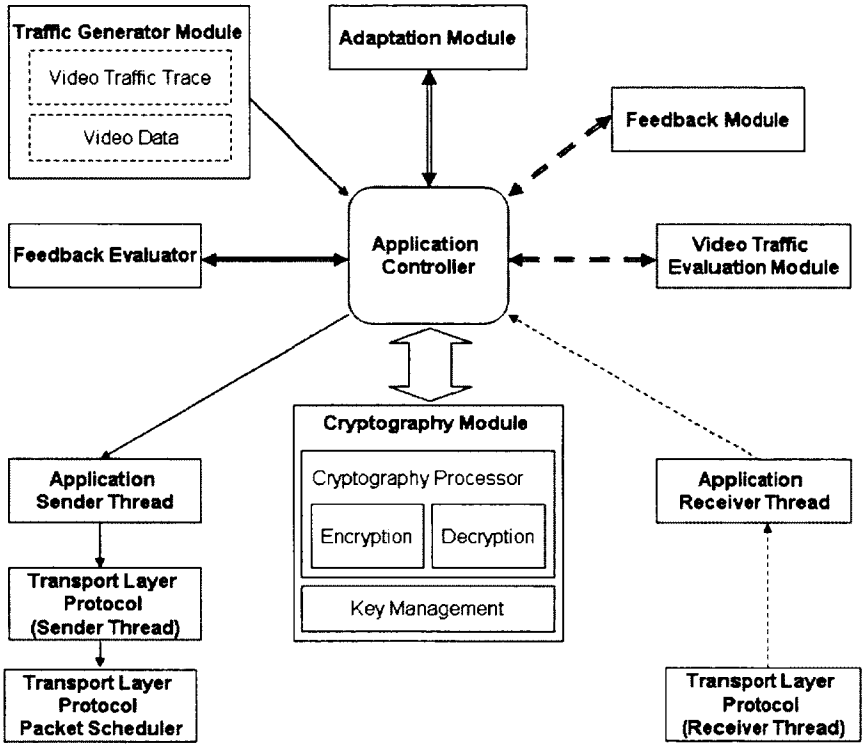


Figure 5.3: Implementation of QaASs in NS. The arrow-headed solid lines indicate operations at the source. The arrow-headed broken lines indicate operations at the receiver.

In NS, the implementation of a transport layer protocol or routing protocol extends the *Agent* class. QaASs is implemented in the application layer and extends the *Application* class. The Agent class maintains a pointer to the Application class for callback. Figure 5.3 is an illustration of implementation of QaASs in the application layer.

The OTcl script defines the network topology, channel parameters, network parameters, applications, traffic and mobility models, and the duration of simulation. The OTcl script points to the entry point of the application.

In Figure 5.3, Application Controller is the entry point.

Video traffic is generated using the video traffic trace derived from the original encoded video data. We use 4:2:0 YUV *foreman* video sequence [8] in CIF format containing 300 frames. The video data is encoded in H.264/AVC format and includes I, P and B-slices. We use the JM 1.7 H.264 codec [136] for encoding the RAW video sequence. The video traffic trace is generated during the encoding process. The trace file contains the following information: frame sequence number, slice type, slice payload size and frame timestamp. The video trace acts as a descriptor for the compressed video data.

The traffic generator module is responsible generating network traffic. The traffic generator utilizes both the video traffic trace and the compressed video data. The module can generate one to multiple simultaneous video traffic flows. The module can generate video traffic at different frame rate.

The cryptography module encompasses the cryptography processor and is also responsible for cryptographic key management. The cryptography module is implemented using the Crypto++ library, version 5.6.1 [43]. Crypto++ is a C++ class library of cryptography algorithms. All the cryptography algorithms described in Section 3.7, are supported by Crypto++. According to the application requirements, at the source, the application controller invokes the cryptography module for encrypting video data. The crypto scheme is decided either at the beginning of a simulation or based on the feedback from the adaptive module. At the receiver's end, the the application controller invokes the cryptography module for decrypting received encrypted video data.

The feedback evaluator module analyzes periodically received feedback packets. A feedback packet contains parameters indicating streaming performance experienced at the receiver's end. Details are available in Section 4.3. Up-on receiving a feedback packet, the application controller forward the parameter set to the feedback evaluator. The evaluation results are returned back to the application controller which forwards the evaluation results to the adaptation module.

The adaptive module can be perceived as callback function, invoked upon receiving a feedback packet. The adaptive module utilizes information

from the feedback evaluator, forwarded by the application controller. The adaptive module follows the operational procedure described in Section 4.3. The adaptation module decides if an adaptation procedure is required to be carried out.

The video traffic evaluation module is responsible for evaluating received video traffic. Received video traffic is first forwarded to the video traffic evaluation module. This module prepares the parameter set that should be transmitted back to the source by the feedback module. The video traffic evaluation module also decides if it should forward the received packet to the decoder or ignore it, in case the packet has been identified as an out-of-order packet.

The feedback module is responsible for sending periodic feedback indicating streaming performance experienced at the receiver's end. Details are available in Section 4.3. This module utilizes information from the video traffic evaluation module as forwarded by the application controller.

5.2 Simulation Setup

Network Topology

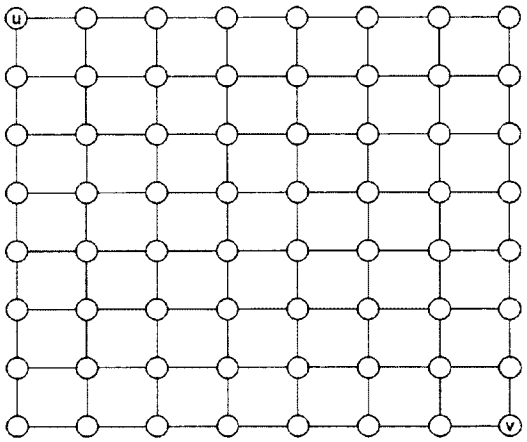


Figure 5.4: Example of a 64 node grid topology.

Initially, there are n^2 nodes in a $k \times k$ area, positioned in a $n \times n$ grid

topology. The grid cells are of equal sizes (Figure 5.4). A node has at least one neighbour within its communication range and together the nodes form a connected graph G [131]. This means, if guaranteed delivery is ensured, a packet sent from a node u will always reach another node v, where $u, v \in G : l(u, v) \notin E$, i.e., u and v are not neighbours. Here, $l(u, v)$ represents a direct bidirectional link connecting any two nodes u and v and E is the set of edges in the graph underlying the network. V is the set of vertices in the underlying graph and $|V| = n^2$. For our evaluation purpose, chosen values of n are be 4, 5, 6, 7 and 8. Hence, we have networks consist of 16, 25, 36, 49 and 64 nodes. The value of k depends on the chosen value of n and size of the grid cell. We use the Random Waypoint mobility model [84] for node mobility. Node velocity is set to 0.5 m/s. In all our simulations, the initial position of the source and destination are the same as locations of u and v in Figure 5.4, i.e., at the opposite ends of the diagonal of the $n \times n$ square grid.

Communications Model

Each node is equipped with at least one 802.11 wireless network interface. The interface operates on the 2.4 GHz frequency band. The maximum data rate of the interface is set to 2 Mbps. The wireless interface uses Omni directional antennae using the Two Ray Ground Reflection model [39] for radio communications. Maximum communications range is set to 250 m. In theory, node u and v are neighbours and can directly communicate with each other, if their scalar distance is no more than 250 m. All nodes communicate over a common channel.

Logical Network Model

The 802.11e wireless MAC is employed to aid admission control. We use the 802.11e with EDCA MAC protocol. The admission control mechanism of EDCA offers traffic prioritization for four different types of traffic. Traffics are prioritized in the following order: Voice, Video, Best effort and Background traffic. We have added support for simultaneous flow of different types of traffic. In a real world setting, it is very much likely that a station would engage in multiple multimedia conferences with several parties while exchanging documents which would require reliable service. Multimedia traffic is transported over UDP, while best effort traffic uses a guaranteed service,

e.g., TCP.

5.3 Simulation Results

In this section we present the simulation results. We carry out a series of simulations to measure a number of performance metrics. We discuss the simulation parameters, metrics and simulation results in the remaining of this section. Table 5.1 lists the common network and simulation parameters.

Network and Simulation Parameters	
Number of nodes	16, 25, 36, 49, 64
Network topology	Grid
Mobility model	Random Waypoint
Node velocity	0.5 m/s
Network interface	WiFi (802.11)
Antenna	Omni directional
Radio propagation model	Two Ray Ground Reflection
Frequency band	2.4 GHz
Radio range	250 m
Modulation	OFDM
MAC	802.11e EDCA
Routing protocol	MP-OLSR
Maximum data rate	2 Mbps
Basic data rate	1 Mbps
Node addressing	IPv6
Transport layer protocol	UDP
Video data	4:2:0 foreman CIF H.264/AVC coded
Original video duration	17 s
Simulation duration	100 s

Table 5.1: Network and simulation parameters.

5.3.1 Evaluation of Encrypted Video Data Transmission

Network and simulation parameters used for this evaluation are listed in Table 5.1. We performed the simulation on a network with 36 nodes. Video traffic begins at the 10^{th} second. This to allow the proactive routing protocol some time to build the routing tables. We used the same 11 cryptography schemes as in Section 3.7.8. Each slice in the H.264/AVC sequence are encrypted independently. Presented results are confirmed with 95% confidence level.

5.3.1.1 Packet Scheduling Time

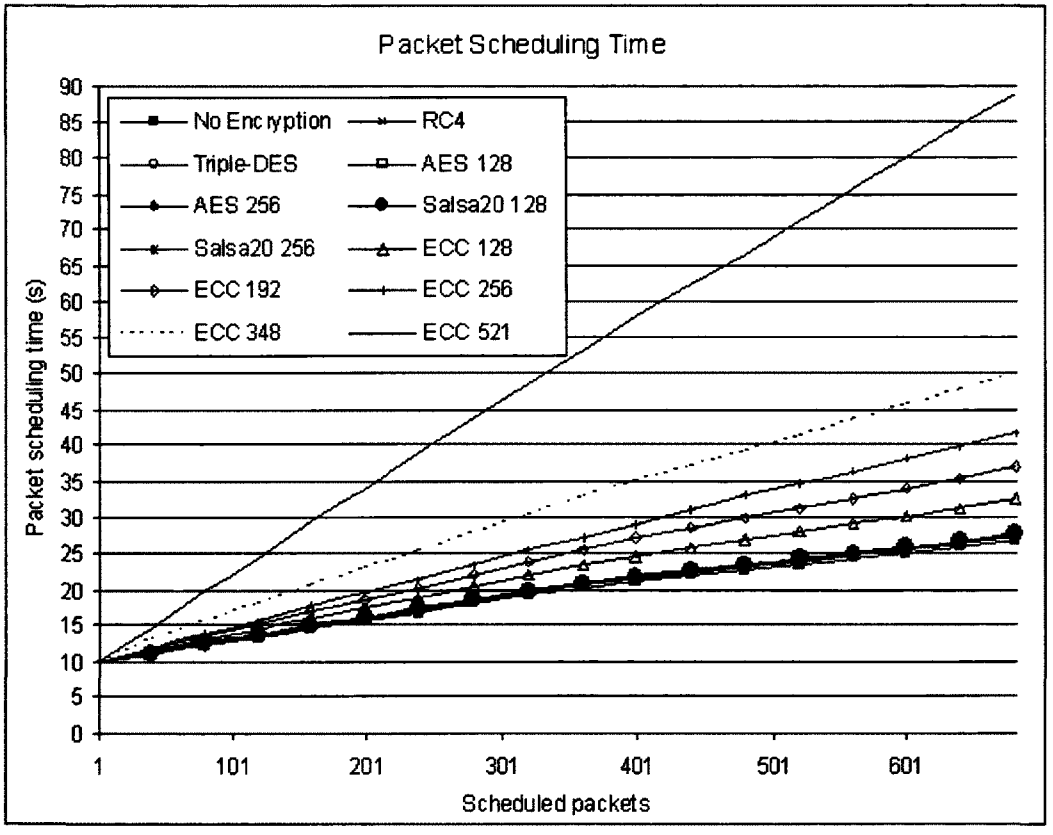


Figure 5.5: Packet scheduling time.

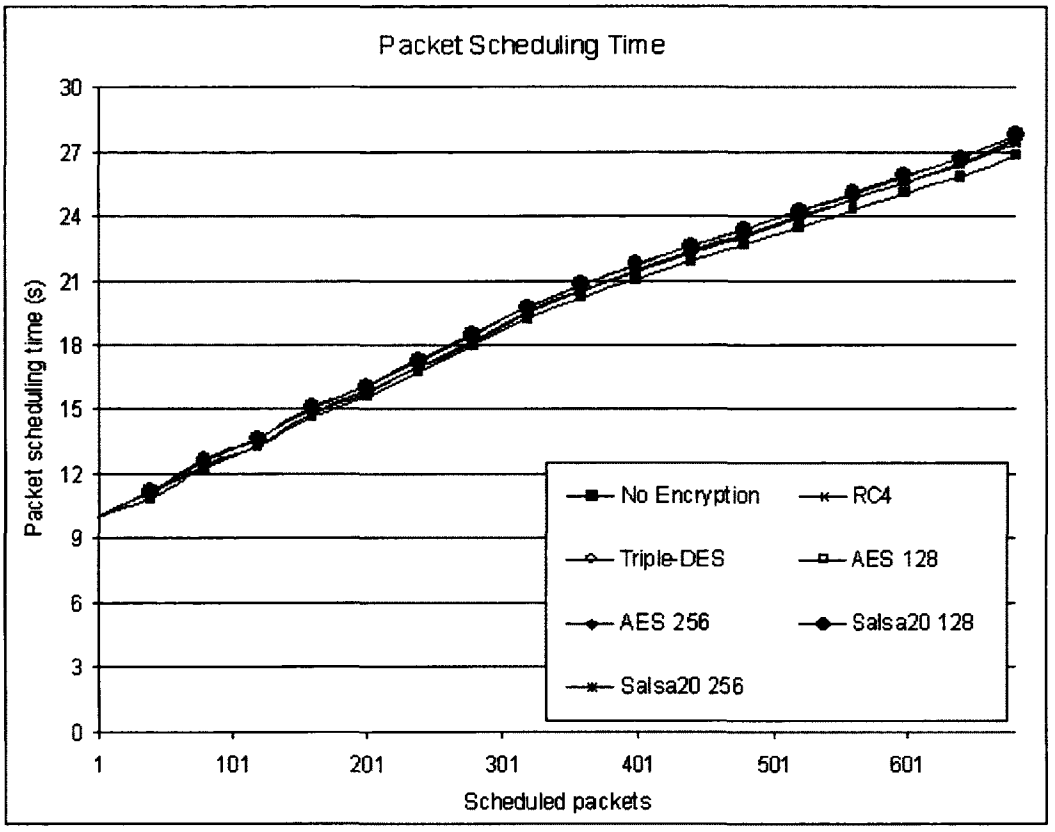


Figure 5.6: Packet scheduling time.

Figures 5.5 and 5.6 show how different cryptography algorithms influence packet scheduling time. Figure 5.6 presents RC4, Triple-DES, AES 128, AES 256, Salsa20 128 and Salsa20 256 data separately for better visibility. The x-axis contains the scheduled packets in order and corresponding scheduling time are on the y-axis. Packet scheduling time follows Equation 4.1. We compare scheduling time of packets containing encrypted video data with packets containing unencrypted video data. Packet scheduling time of video data encrypted using RC4, Triple-DES, AES 128, AES 256, Salsa20 128 and Salsa20 256 falls within 1 s of the unencrypted video data (Figure 5.6). Packet scheduling time complements the encryption throughput in Figure 3.22. The cryptography algorithm with the least encryption throughput has the highest scheduling delay. Furthermore, with progression of transmission, the packet scheduling time is magnified by quite a margin. For instance,

ECC with a 521-bit curve has the lowest encryption throughput and highest packet scheduling delay. A packet containing an unencrypted video frame was scheduled for transmission at about the 27^{th} second. Employing ECC with a 521-bit curve, the same video frame was scheduled for transmission at about the 90^{th} second, resulting in an approximate minimum of 63 s playback delay.

5.3.1.2 Cryptography Delay

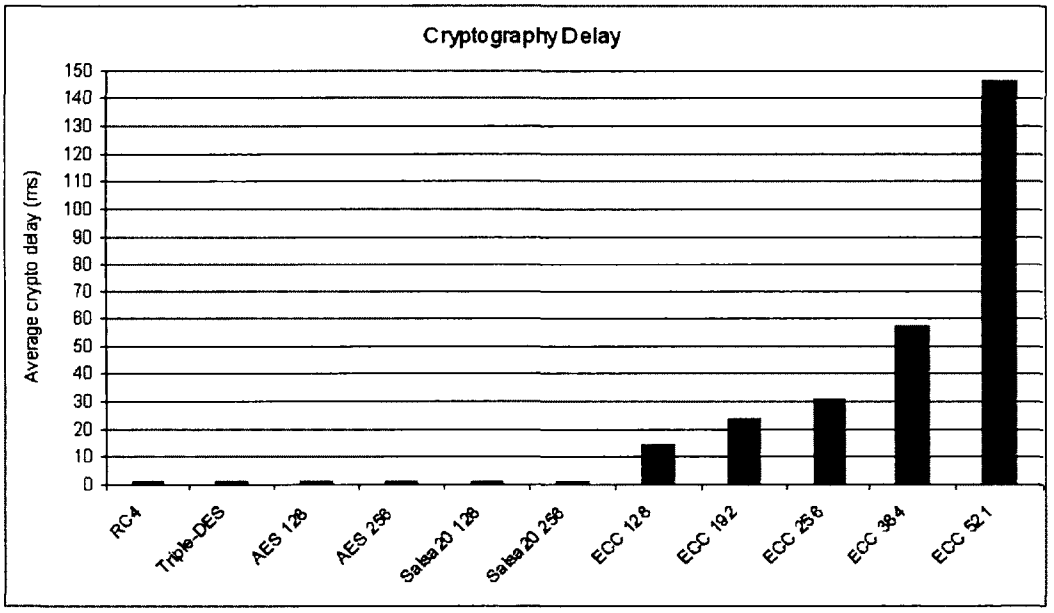


Figure 5.7: Cryptography delay.

Figure 5.7 compares cryptography delay of different crypto schemes. For each crypto scheme, we present average cryptography delay per frame over the transmission period. RC4, Triple-DES, AES 128, AES 256, Salsa20 128 and Salsa20 256 have average cryptography delay less than 2 ms. ECC on the other hand, with increase in the curve size, shows significant increase in cryptography delay. ECC with a 128-bit curve has the lowest average cryptography delay among the ECC schemes, approximately 15 ms. Cryptography delay for ECC with a 521-bit curve is about 147 ms.

5.3.1.3 Transmission Time

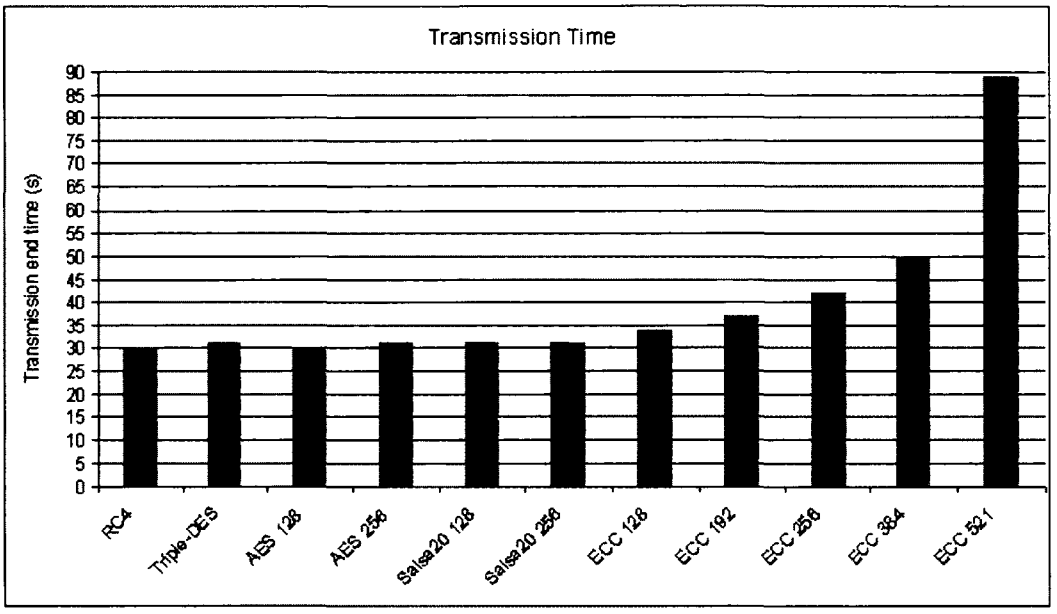


Figure 5.8: Transmission time.

Figure 5.8 compares transmission time as a result of using different crypto schemes. For each crypto scheme, we present time required to transfer the entire video sequence. Transmission time for RC4, Triple-DES, AES 128, AES 256, Salsa20 128 and Salsa20 256 are less than 31 s, at most 4 s more than the last frame in the video sequence was originally available at the source. ECC on the other hand, with increase in the curve size, shows significant increase in transfer time. ECC with a 128-bit curve require the least amount of time among the ECC schemes, approximately 34 s. Transmission time for ECC with a 521-bit curve is about 89 s.

5.3.1.4 Packet Rate

Figure 5.9 shows effect of different cryptography schemes on packet rate. From Figure 5.9 we can see that RC4, Triple-DES, AES 128, AES 256, Salsa20 128 and Salsa20 256 achieve average packet rate over 40 packets per second. With increase in curve size, the ECC cryptography schemes show dramatic decrease in packet transmission rate. For instance, ECC with a

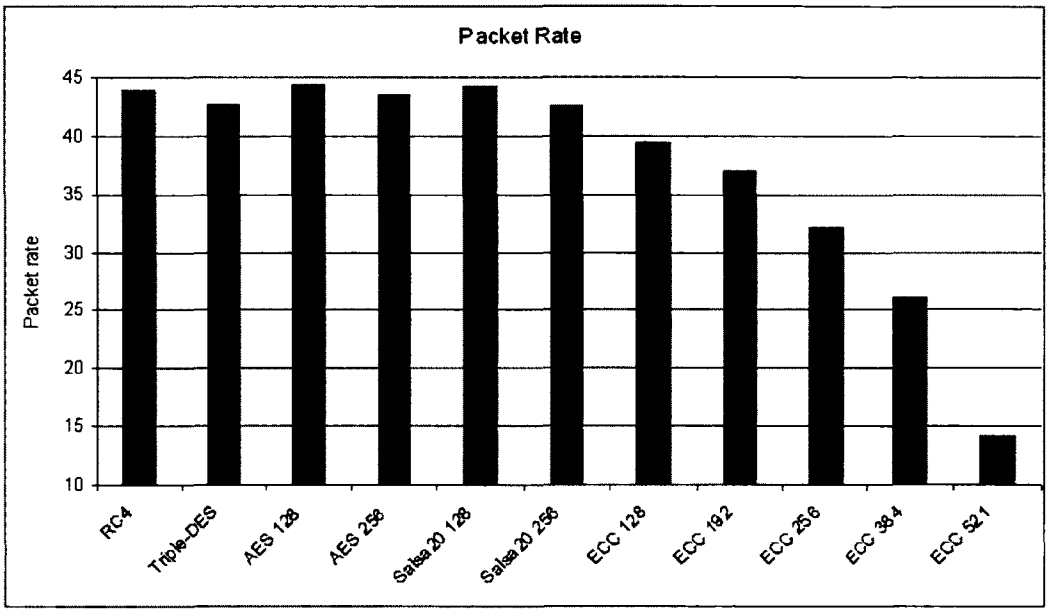


Figure 5.9: Packet rate.

192-bit curve achieves average packet rate of 37 packets per second, whereas ECC with a 521-bit curve achieves less than 15 packets per second, more than 50% drop in packet rate. Furthermore, as a result of low packet rate, with increase in curve size, ECC cryptography schemes take longer to deliver the entire video. Figure 5.10 shows packet rates using boxplot. The plot shows minimum and maximum values, lower (25^{th} percentile) and upper (75^{th} percentile) quartiles, and the median.

5.3.1.5 Transmission Delay

Figures 5.11 and 5.12 show transmission delay of encrypted video data. Figure 5.12 presents RC4, Triple-DES, AES 128, AES 256, Salsa20 128 and Salsa20 256 data separately for better visibility. On the x-axis is transmission timestamp and average transmission delay per second collected over the transmission period is on the y-axis. From Figure 5.12, we can see RC4, Triple-DES, AES 128, AES 256, Salsa20 128 and Salsa20 256 show very similar trends. In absence of any additional QoS mechanism, transmission delay gradually increases with the progression of transmission. This is due to congestion caused by increasing traffic at the default data rate. ECC schemes

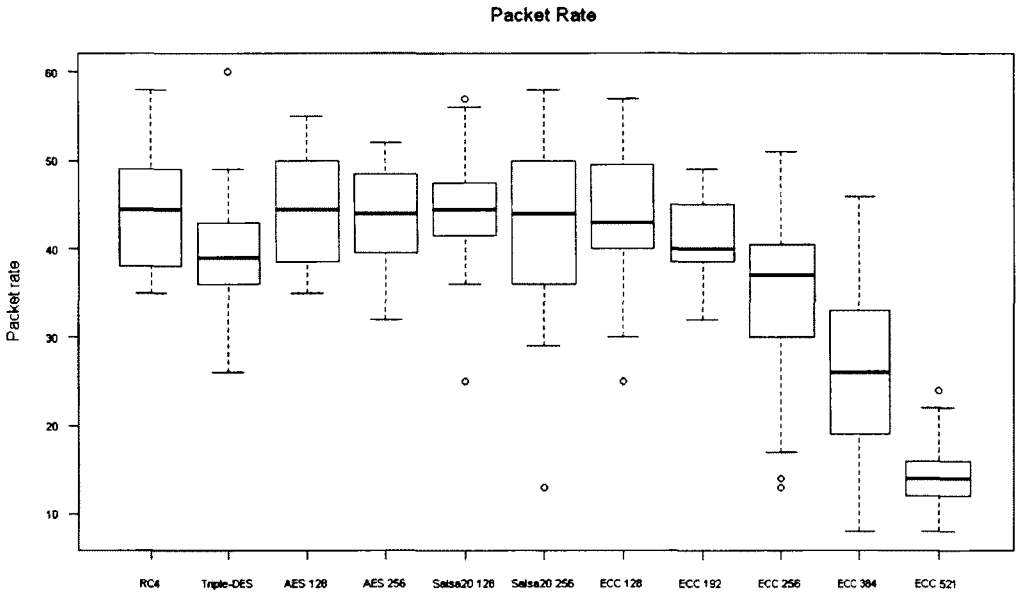


Figure 5.10: Packet rate (boxplot).

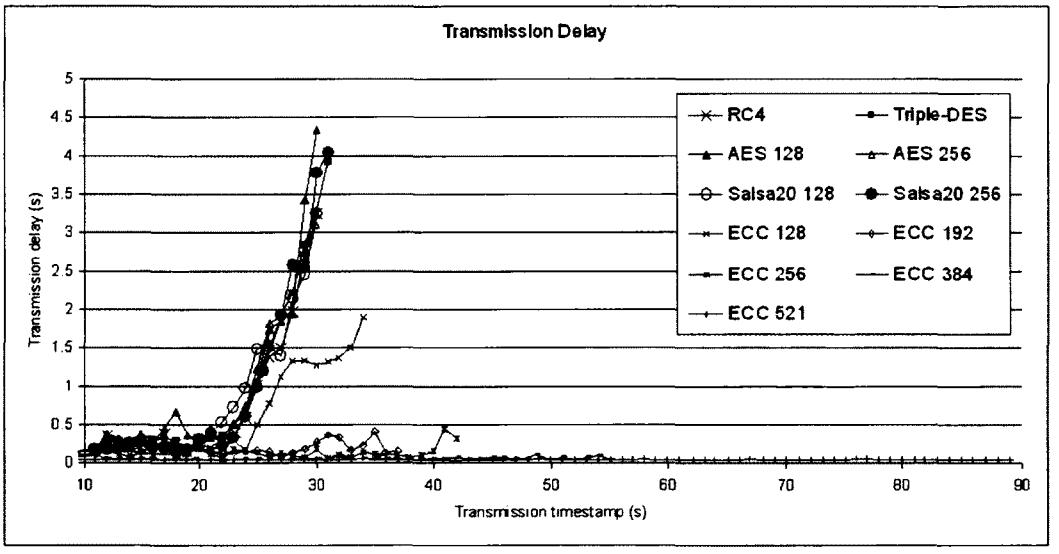


Figure 5.11: Packet transmission delay.

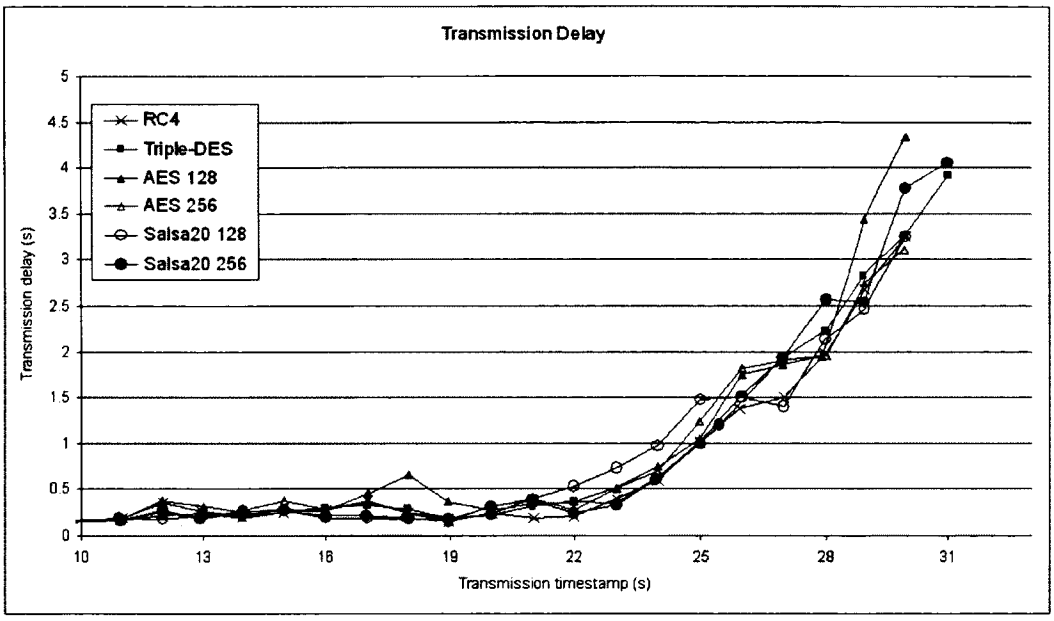


Figure 5.12: Packet transmission delay.

on the other hand, experience low transmission delay because of low data rate caused by expensive ECC encryption operations.

5.3.2 Evaluation of the QaASs Adaptation Schemes

Network and simulation parameters used for this evaluation are listed in Table 5.1. We performed the simulation on a network with 36 nodes. Video traffic begins at the 10^{th} second. This to allow the proactive routing protocol some time to build the routing tables. Feedback packet interval is set to 1s. Presented results are confirmed with 95% confidence level.

5.3.2.1 Adaptation Option One

We compare results of the following five different simulation scenarios, indexed S1-S5:

S1. Without adaptation.

S2. Single video traffic flow with adaptation option one and no rekeying.

S3. In presence of six video traffic flows with adaptation option one and no rekeying.

S4. Single video traffic flow with adaptation option one and with rekeying option one (frequency: once every 1s).

S5. Single video traffic flow with adaptation option one and with rekeying option two (frequency: once every 1s).

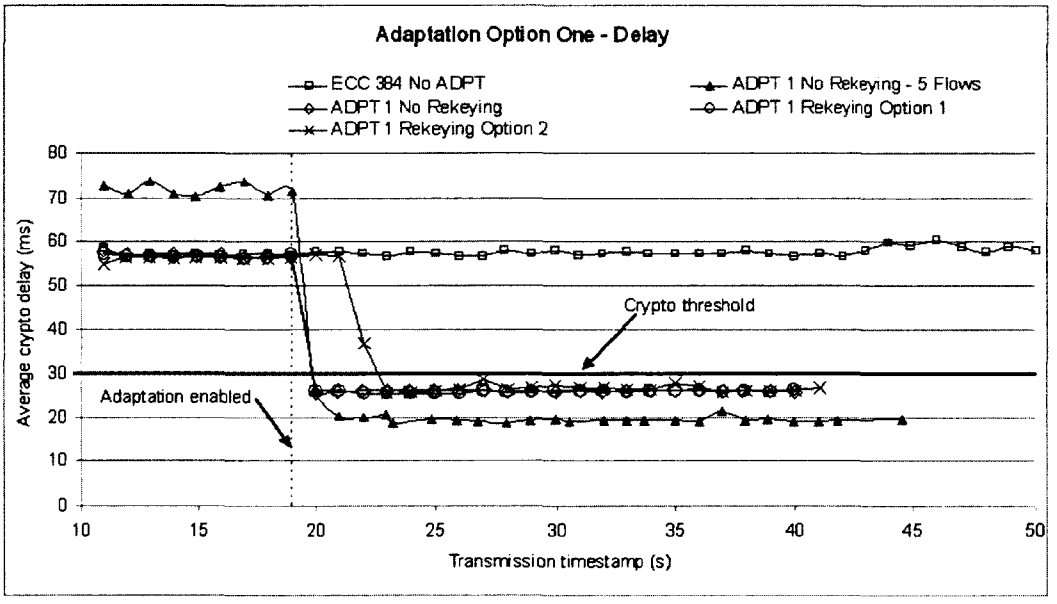


Figure 5.13: Adaptation option one for different scenarios.

In order to observe the effect of adaptation, the adaptation procedure is set to be activated at the 19th second. The crypto threshold, $CRP_{threshold}$ is set to 30 ms. For all five scenarios, we begin simulation with encrypting video data using ECC with a 348-bit curve. Figure 5.13 shows simulation results of average cryptography delay per second over the transmission period. On the x-axis is transmission timestamp and on the y-axis is average cryptography delay per second over the transmission period. When adaptation is enabled at the 19th second, in S2, based on received feedback, cryptography scheme is changed from ECC with a 384-bit curve (average 58 ms cryptography delay) to ECC with a 192-bit curve (average 27 ms cryptography delay), in

order to meet the crypto threshold requirement. Transmission is completed by the 42nd second compared to 51st second of S1, a 9 s gain in playback time.

In S3 there are six traffic flows and all begin with encrypting video data using ECC with a 384-bit curve. Six simultaneous encryption processes causes noticeable delay overhead for each flow, approximately an average 74 ms compared to 58 ms of the single flow. Based on received feedback, cryptography scheme is changed from ECC with a 384-bit curve (average 58 ms cryptography delay) to ECC with a 128-bit (average 20 ms cryptography delay) in order to meet crypto threshold requirement. Transmission is completed by the 45th second. This is to note that, in S3, cryptography scheme is changed from ECC with a 384-bit curve to ECC with a 128-bit curve, whereas in S2, cryptography scheme was changed to ECC with a 192-bit curve. This is due to six simultaneous cryptography processes for ECC with a 192-bit curve, none of the traffic flows meet the crypto threshold requirement.

S4 is similar to S2, with the addition of rekeying option one (as described in Section 4.3). From Figure 5.13, we can see, S4 shows similar performance as S2. S5 is similar to S2, with the addition of the rekeying option two. We can see, S5 shows similar performance as S2 as well. Since, rekeying option two requires two way communications and in-session key agreement, the delay overhead in S5 is slightly more than S4. In summery, with adaptation option one, without and with rekeying, we obtain significant gain in overall delay compared to absence of adaptation.

Figure 5.14 shows comparison of network traffic caused by S1, S2, S4 and S5. For S4 and S5, we obtain results for rekeying frequencies once in every 0.50 , 1.0, 1.50 and 2.0 second. S1 generates more than 35000 additional bytes than any other case. The traffic overhead caused by rekeying is at most 7500 bytes more than in S2. Rekeying option two with rekeying frequency once every 0.5s, generates highest amount of bytes among all simulations with adaptation, yet causes half the amount of traffic compared to the case without adaptation.

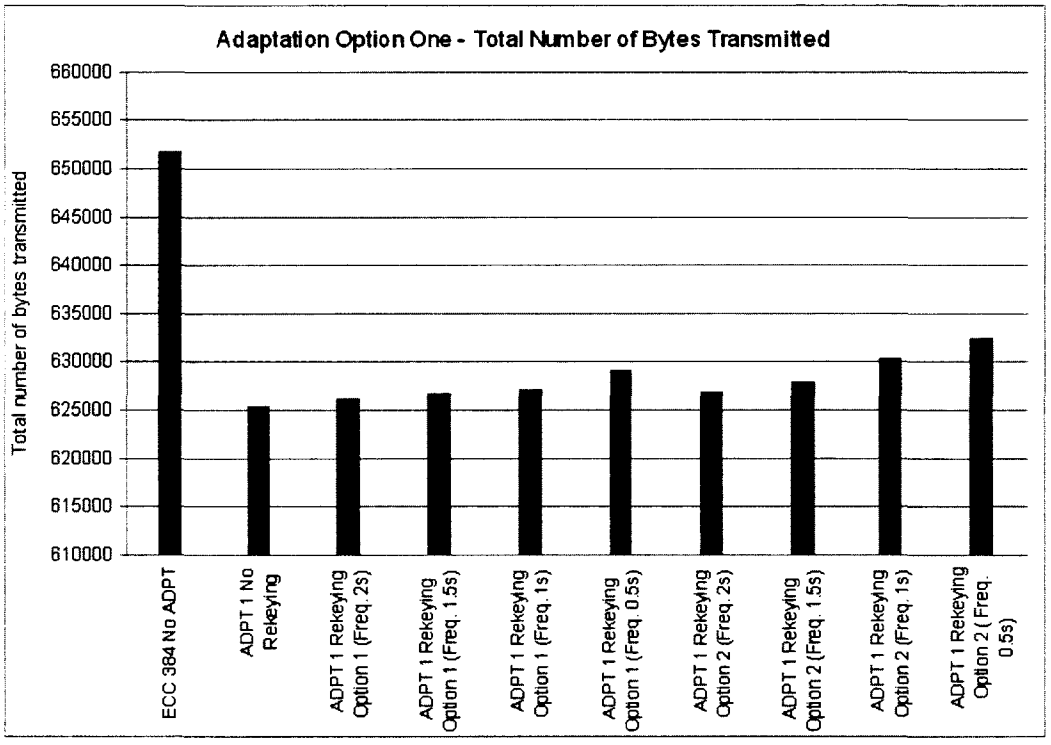


Figure 5.14: Adaptation option one with rekeying.

5.3.2.2 Adaptation Option Two

We compare results of the following two different simulation scenarios:

S1. Without adaptation.

S2. Single video traffic flow with adaptation option two.

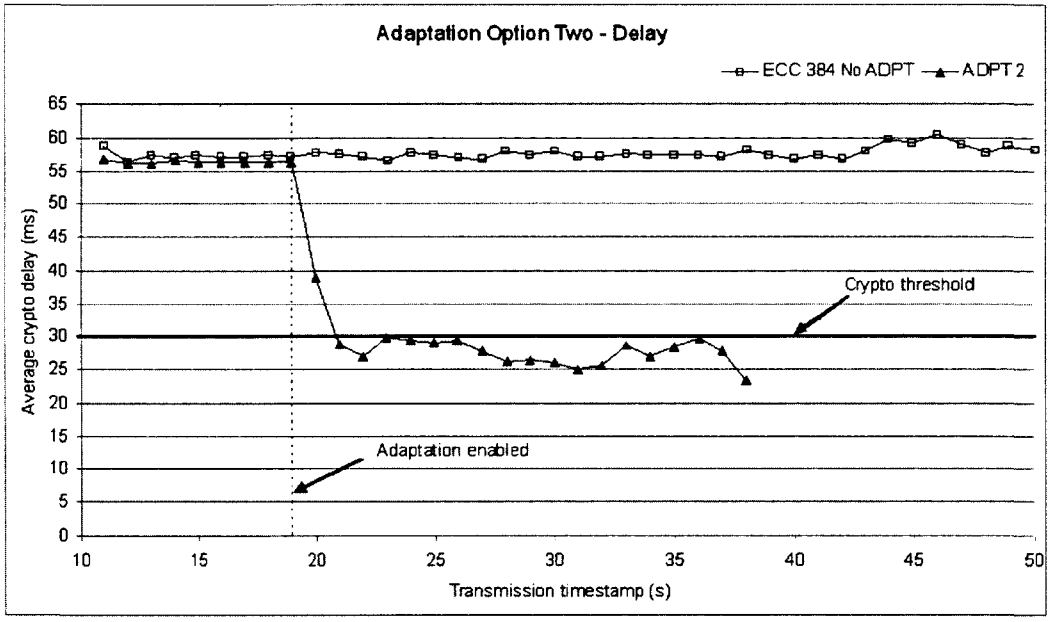


Figure 5.15: Comparison of delay: no adaptation vs adaptation option two.

Similar to simulation of adaptation option one, the adaptation procedure is set to be activated at the 19th second. The crypto threshold, $CRP_{threshold}$ is set to 30 ms. For all scenarios, we begin simulation with encrypting video data using ECC with 348-bit curve. Figure 5.15 shows the simulation results of average cryptography delay per second over the transmission period. On the x-axis is transmission timestamp and on the y-axis is average cryptography delay per second over the transmission period. When adaptation is enabled at the 19th second, in S2, based on received feedback, the source realizes that average crypto delay does not satisfy the crypto threshold and decides not to encrypt the B-frames. In a later iteration, the source realizes that the average crypto delay still does not satisfy the crypto threshold and

decides not to encrypt the P-frames. Transmission is completed by the 38th second compared to 51st second of S1, a 12 s gain in playback time.

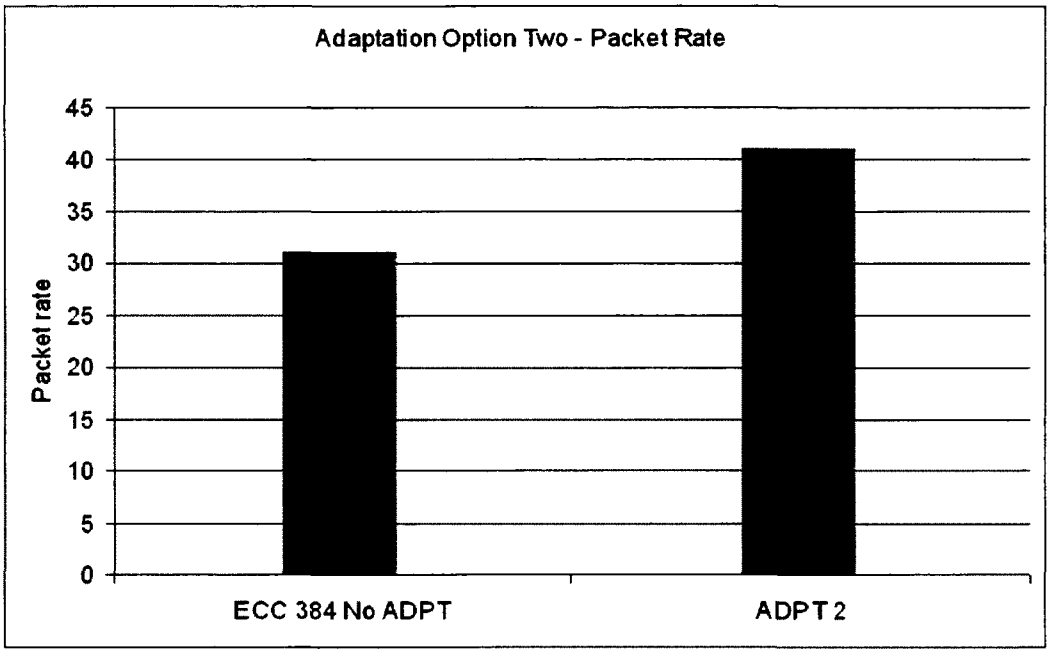


Figure 5.16: Comparison of packet rate (average): no adaptation vs adaptation option two.

Figure 5.16 shows comparison of packet rate, average number packets transmitted per second. With adaptation option two, we were able to the increase packet rate by an average 10 more packets per second. Figure 5.17 shows the packet rates using boxplot. The plot shows minimum and maximum values, lower (25th percentile) and upper (75th) quartiles, and the median. In S2, the median is 40 compared to 33.5 in S1.

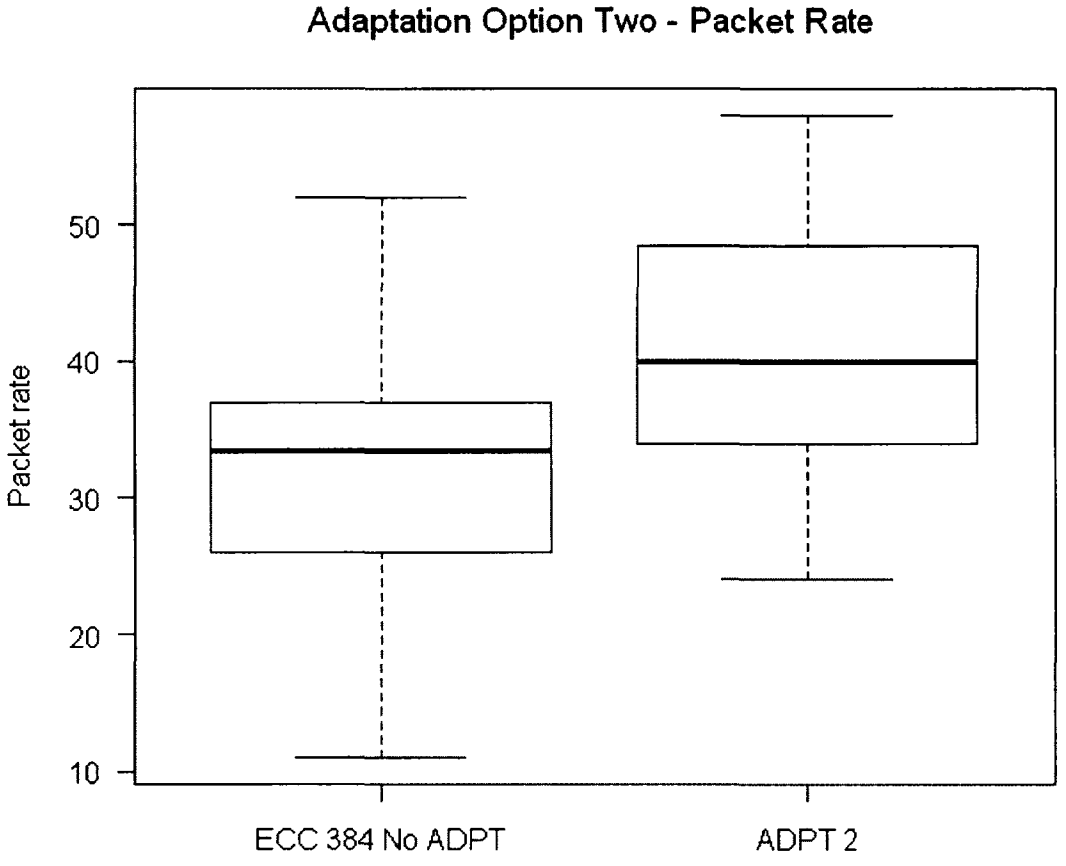


Figure 5.17: Comparison of packet rate (boxplot): no adaptation vs adaptation option two.

5.3.2.3 Adaptation Option Three

We compare results of the following two different simulation scenarios:

S1. Without adaptation.

S2. Single video traffic flow with adaptation option three.

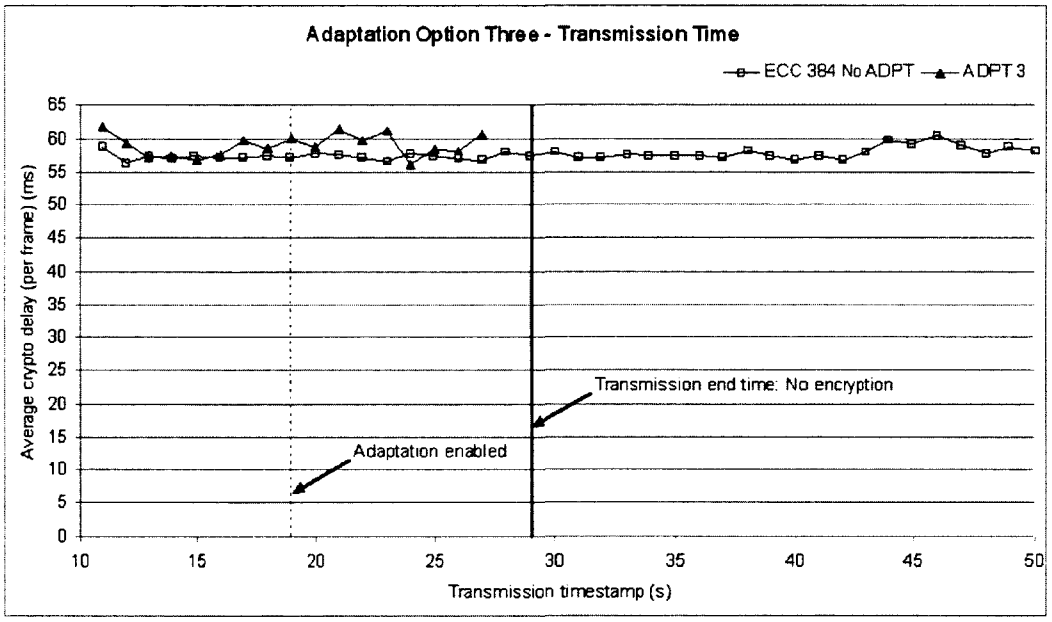


Figure 5.18: Comparison of transmission time: no adaptation vs adaptation option three.

The adaptation procedure is set to be activated at the 19th second. The crypto threshold, $CRP_{threshold}$ is set to 30 ms. For all scenarios, we begin simulation with encrypting video data using ECC with a 348-bit curve. Figure 5.18 shows time taken by each scenario to transfer the entire video sequence. On the x-axis is transmission timestamp and on the y-axis is average cryptography delay per second over the transmission period. When adaptation is enabled at the 19th second, in S2, based on received feedback, the source realizes that average crypto delay does not satisfy the crypto threshold and decides not to encrypt the B-frames and not send B-frames. In a later iteration, the source realizes that that the average crypto delay still does not

satisfy the crypto threshold and decides not to encrypt the P-frames and not include P-frames in the video stream. Transmission is completed by the 27^{th} second compared to 51^{st} second of S1, a 24 s gain in playback time (in expense of quality of cource). What is interesting here is that, we were able to transfer the entire video sequence within the time period of transfer time of transmission without encryption.

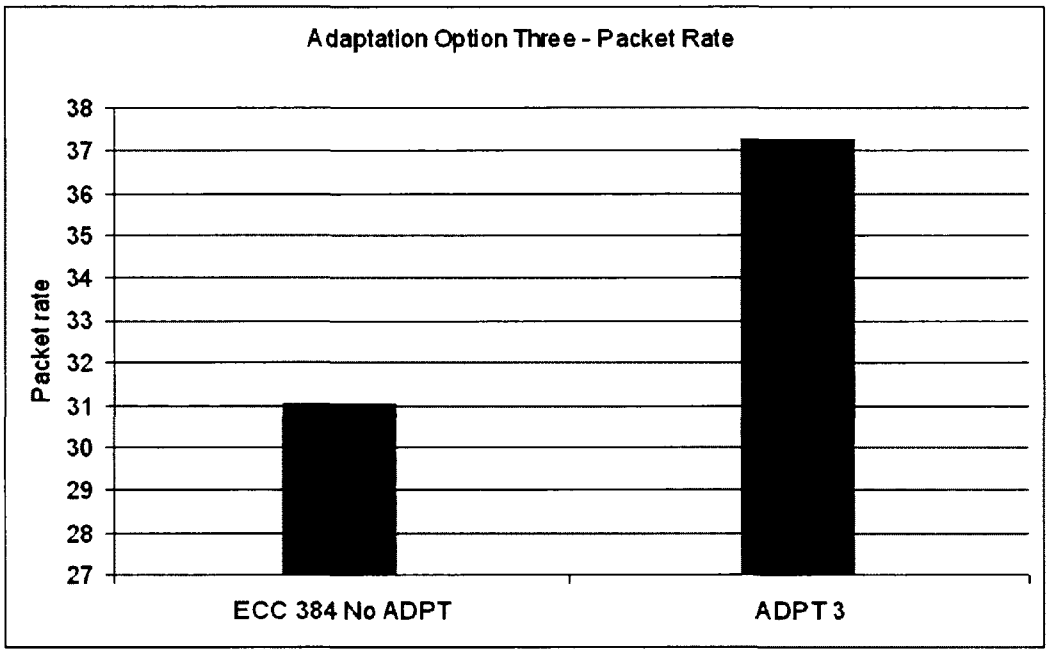


Figure 5.19: Comparison of packet rate: no adaptation vs adaptation option three.

Figure 5.19 shows comparison of packet rate, average number packets transmitted per second. This is to note that, because of adaptation three, in S2, data traffic is considerably lowered.

5.3.2.4 Adaptation Option Four

For the evaluation of adaptation option four, we used *foreman* video sequence coded at 30 fps. The encoded video contains only I and P-frames and the GOP length is 30. The video is approximately 10 s long. We compare results of the following two different simulation scenarios:

S1. Without adaptation.

S2. Single video traffic flow with adaptation option four.

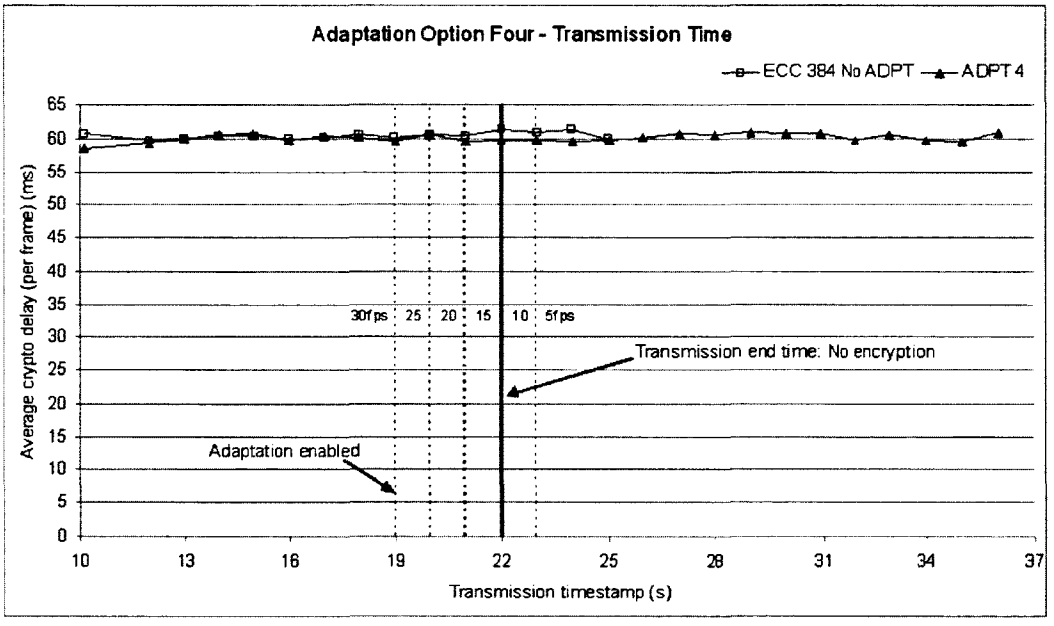


Figure 5.20: Comparison of transmission time: no adaptation vs adaptation option four.

The adaptation procedure is set to be activated at the 19^{th} second. The crypto threshold, $CRP_{threshold}$ is set to 2100 ms. For all scenarios, we begin simulation with encrypting video data using ECC with a 348-bit curve. Figure 5.20 shows the time taken by each scenario to transfer the entire video sequence. On the x-axis is transmission timestamp and on the y-axis is average cryptography delay per second over the transmission period. When adaptation is enabled at the 19^{th} second, in S2, based on received feedback,

the source realizes that cumulative crypto delay (described in Section 5.3.2.4) does not satisfy the crypto threshold and decides to reduce the frame rate to 25 fps. In a later iteration, the source realizes that the cumulative crypto delay still does not satisfy the crypto threshold and decides to reduce the frame rate to 20 fps. For the selected simulation parameters, ultimately the frame rate is reduced to just 5 fps. Transmission is completed by the 25^{th} second compared to 36^{th} second of S1, an 11 s gain in playback time. Additionally, the transmission time of the encrypted video stream with adaptation option four is only about 3 s more than the unencrypted stream.

5.3.2.5 Comparison of Adaptation Options

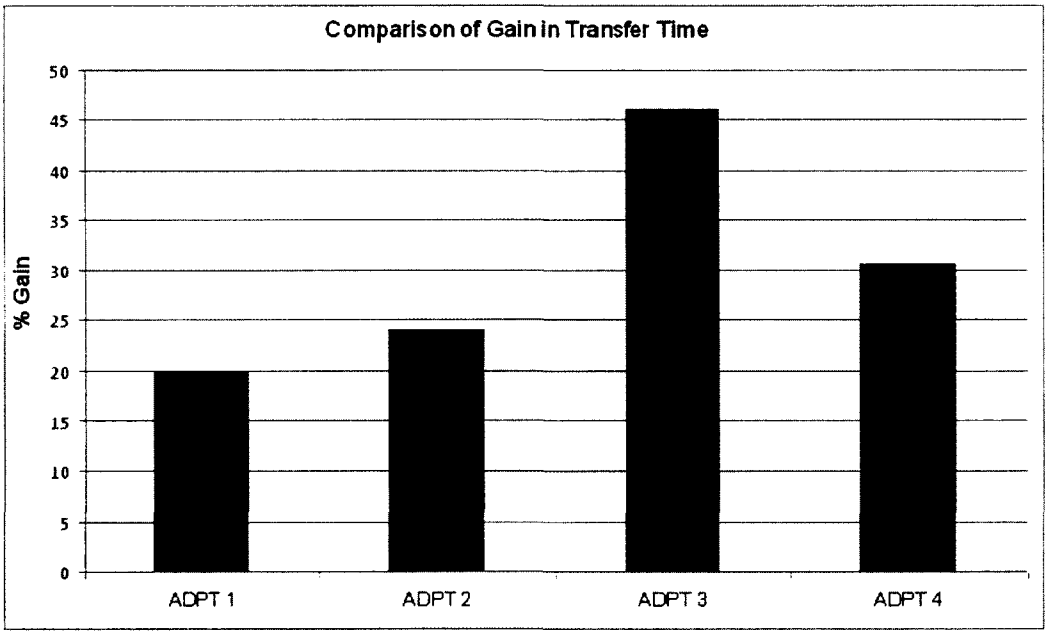


Figure 5.21: Comparison of % gain in transfer time of four adaptation options.

Figure 5.21 compares the transfer time of the entire video sequence for all four adaptation options. We present the % (percentage) gain in transfer time compared to the absence of adaptation.

In adaptation option one, in order to meet the crypto threshold, we switch to a higher throughput ECC crypto scheme with a smaller curve. An ECC crypto scheme with a bigger curve offers stronger security compared to the scheme with a smaller curve. Eventually, transferring to an ECC crypto scheme with a smaller curve reduces the level of security. In adaptation option two, based on receiver's feedback, we decide if frames containing inter coded macroblocks (e.g., P and B-frames) should be encrypted. When P or B-frames are not encrypted, we are putting out unencrypted data for transmission. According to Agi and Gong [4], the presence of I-blocks in unencrypted P and B-frames is a security hole. A series of P and B-frames could carry enough information if their base frames are correlated. A frame containing an unencrypted I-block, being referenced by blocks in subsequent frames, can be decoded. Therefore, in the case of both adaptation option one and two, with sufficient computing resources, it might be possible for a crypto analyst to glean information about the transmitted video data.

5.3.2.6 Discussions

According to the simulation results, in adaptation option one, in order to meet the crypto threshold, an ECC crypto scheme with higher throughput is chosen. An ECC crypto scheme with a smaller curve offers higher throughput. When switching to a higher throughput crypto scheme, we are selecting an ECC crypto scheme with a smaller curve and therefore, security is being compromised while original video quality is maintained. In adaptation option two, in order to meet the crypto threshold, based on receiver feedback, we decide if frames containing inter coded macroblocks (e.g., P and B-frames) should be encrypted. When P or B-frames are not encrypted, we are putting out unencrypted data for transmission. Therefore, security is compromised but original video quality is maintained. In adaptation option three, in order to meet the crypto threshold, we do not transmit the unencrypted P or B-frames. As a result of which video quality is compromised but security is maintained. In adaptation option four, frame rate is adapted in order to meet the crypto threshold. Reducing the frame rate compromises video playback quality. Similar to the adaptation option three, since all video data are encrypted with the original crypto scheme, security of the transmitted video data is maintained. Table 5.2 summarizes the key elements of comparison of the four adaptation options.

	Adaptation Option One	Adaptation Option Two	Adaptation Option Three	Adaptation Option Four
Adapted Property	Crypto Scheme	Inter Coded Frame Encryption	Inter Coded Frame Transmission	Video FPS
Possibility of Compromising Security	Yes	Yes	No	No
Possibility of Compromising Video Quality	No	No	Yes	Yes

Table 5.2: Summary of comparison of the four adaptation options.

Chapter 6

Conclusion

The goal of this work was to develop a solution that addresses the issue of transmission delay overhead caused by cryptography operations. We perceived the addressed problem in the context of multimedia streaming in ad hoc networks. We reviewed the challenges associated with security mechanisms and multimedia services in MANETs. In order to develop a possible solution, we have evaluated a number of MANET properties and cryptography algorithms. Based on our evaluation, we have proposed an adaptive mechanism that aims to provide stability between QoS of multimedia service and security measures. Chapter 4 documents our proposal. In Chapter 5 we have presented the simulation setup, implementation details and evaluation of our proposal. Section 6.1 summarizes the contributions and Section 6.2 outlines possible future work in this ares of research.

6.1 Contributions

We have proposed an adaptation mechanism that adapts cryptography and/or multimedia service properties in order to meet desired QoS while maintaining the required level of security. The adaptation mechanism utilizes service feedback from the receiver in real-time. We have presented four different adaptation options exemplifying different application requirements. In order to verify our proposal, we assembled an NS based simulation environment for secured multimedia streaming in ad hoc networks. One attractive feature of the simulation setup is that, it carries out cryptography operations in real-time on actual video data, making it very close to real-world experience.

Our evaluation shows that, in presence of adaptation option one, video transfer time could be reduced by a significant margin. We have shown, for ECC with a 384-bit curve, using adaptation option one, it is possible to gain 9 s on transfer time. We have also shown that, while choosing an cryptography scheme with higher throughput and lower security may reduce transmission delay, periodic rekeying is a viable option to elevate security with less than apprehensible effect on performance.

Using adaptation option two, it is possible to increase packet rate and further reduce video transfer time. We have shown, for ECC with a 384-bit curve, using adaptation option two, it is possible to gain 12 s on transfer time and increase average packet rate by 10 packets per second.

Adaptation option three, though reduces quality, improves video transfer time without compromising security. For ECC with a 384-bit curve, using adaptation option three, we have been able to transfer the entire video sequence within the time period of transfer time of transmission without encryption.

Adaptation option four improves overall transmission time by adjusting frame rate. We have shown that, for ECC with a 384-bit curve, using adaptation option four, it is possible to gain about 10 s on transfer time and the time is comparable to unecrypted video stream.

6.2 Future Work

In our work we have only considered H.264/AVC. The concept can be extended for H.264/SVC [134] and 3D/Stereoscopic video coding [112].

For our evaluation, we have assumed the value of *Crypto Threshold* (e.g., 30ms). Ideally, the value of Crypto Threshold should differ across networks and application requirements. We think, developing an algorithm to determine value of Crypto Threshold based on network and application parameters can be a candidate for future work.

In our simulation, we have used video traces for generating video traffic

for already encoded video data. In order to simulate live video communications closer to real-life experience, it would be ideal to incorporate real-time encoder/decoder to the simulation framework.

Although not part of our core contribution, we thought the following two topics, that we came across during our research, could be considered for future research:

The proposed adaptive mechanism concerns only point-to-point confidentiality, thus the routes in the ad hoc network are required to be authenticated. In order to provide routing security in MP-OLSR, we have outlined a distributed public key based route authentication mechanism based on Dhillon et al.'s [49] distribute CA technique. The route authentication mechanism can be considered for implementation and security analysis.

MP-OLSR assumes, no pair of nodes are connected by more than one link. The cost functions are designed around this assumption. From the physical constructional point of view, this means, MP-OLSR assumes that each node is equipped with a single wireless interface, which we think is a significant limitation and is a good candidate for future research.

Bibliography

[1] Unapproved Draft Standard for Information Technology- Telecommunications and information exchange between systems- Local and metropolitan area network- Specific requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications. (This document reflects the combining of the 2003 Edition of 802.11 plus the 802.11g, 802.11h, 802.11i and 802.11j Amendments) (Revision of IEEE Std 802.11-1999) (Superseded by P802.11REV-ma/D4.0). *IEEE Std P802.11REV-ma/D4.0*, 1999.

[2] Unapproved Draft Amendment Standard for Information Technology–Telecommunications and Information Exchange Between Systems–LAN/MAN Specific Requirements– Part 11 Wireless Medium Access Control (MAC) and Physical Layer (PHY) specifications: Medium Access Control (MAC) Quality of Service (QoS) Enhancements (Replaced by approved draft 802.11e/D13.0). *IEEE Std P802.11e/D13.0*, 2005.

[3] IEEE 802.11e. ns-2 imlementation of IEEE 802.11e EDCA. `ftp://ftp-sop.inria.fr/rodeo/qni/ns-edcf.tar.gz`.

[4] I. Agi and L. Gong. An Empirical Study of Secure MPEG Video Transmissions. In *Proceedings of the Symposium on Network and Distributed System Security (SNDSS)*, pages 137–144, Washington, DC, USA, 1996. IEEE Computer Society.

[5] G. Ahn, A. Campbell, A. Veres, and L. Sun. SWAN: service differentiation in stateless wireless ad hoc networks. In *Proceedings of the Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, volume 2, pages 457–466, 2002.

[6] W. Arbaugh. Wireless security is different. *Computer*, 36(8):99–101, August 2003.

[7] N. Asokan and P. Ginzboorg. Key Agreement in Ad-hoc Networks. *Computer Communications*, 23:1627–1637, 1999.

[8] ASU. Video Trace Library. `http://trace.eas.asu.edu/index.html`.

[9] B. Awerbuch, D. Holmer, C. Nita-Rotaru, and H. Rubens. An on-demand secure routing protocol resilient to byzantine failures. In *Proceedings of the 1st ACM workshop on Wireless security (WiSE)*, pages 21–30, New York, NY, USA, 2002. ACM.

[10] O. Badarneh, M. Kadoch, and A. Elhakeem. Video multicast based multiple description coding and multi-paths in wireless ad hoc networks. In *Canadian Conference on Electrical and Computer Engineering (CCECE)*, pages 604–609, May 2009.

[11] H. Badis and K. Agha. QOLSR multi-path routing for mobile ad hoc networks based on multiple metrics: bandwidth and delay. In *IEEE 59th Vehicular Technology Conference (VTC)*, volume 4, pages 2181–2184, May 2004.

[12] H. Badis, I. Gawedzki, and K. Agha. QoS routing in ad hoc networks using QOLSR with no need of explicit reservation. In *IEEE 60th Vehicular Technology Conference (VTC)*, volume 4, pages 2654–2658, September 2004.

[13] H. Badis, A. Munaretto, K. Agha, and G. Pujolle. QoS for Ad hoc Networking Based on Multiple Metrics: Bandwidth and Delay. In *Mobile and Wireless Communication Networks (MWCN)*, pages 15–18, 2003.

[14] M. Barbeau and E. Kranakis. *Principles of ad hoc networking.* John Wiley & Sons Ltd., 2007.

[15] K. Becker and U. Wille. Communication complexity of group key distribution. In *Proceedings of the 5th ACM conference on Computer and communications security (CCS)*, pages 1–6, New York, NY, USA, 1998. ACM.

[16] S. Bellovin and M. Merritt. Encrypted key exchange: password-based protocols secure against dictionary attacks. In *Proceedings of IEEE Computer Society Symposium on Research in Security and Privacy (RISP)*, pages 72–84, May 1992.

[17] M. Ben Mahmoud, N. Larrieu, A. Pirovano, and A. Varet. An adaptive security architecture for future aircraft communications. In *IEEE/AIAA 29th Digital Avionics Systems Conference (DASC)*, pages 3.E.2–1–3.E.2–16, October 2010.

[18] D. Bernstein. Salsa20 - The eSTREAM Project - eSTREAM Phase 3 Profile 1. `http://www.ecrypt.eu.org/stream/salsa20p3.html`, 2008.

[19] D. Bernstein. The Salsa20 Family of Stream Ciphers. In Matthew Robshaw and Olivier Billet, editors, *New Stream Cipher Designs*, volume 4986 of *Lecture Notes in Computer Science*, pages 84–97. Springer Verlag, Berlin, Heidelberg, Germany, 2008.

[20] S. Bhattacharya, T. Chattopadhyay, and A. Pal. A Survey on Different Video Watermarking Techniques and Comparative Analysis with Reference to H.264/AVC. In *IEEE 10th International Symposium on Consumer Electronics (ISCE)*, pages 1–6, 2006.

[21] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss. An Architecture for Differentiated Service. RFC 2475 (Informational), December 1998. Updated by RFC 3260.

[22] F. Borgonovo, A. Capone, M. Cesana, and L. Fratta. ADHOC MAC: new MAC architecture for ad hoc networks providing efficient and reliable point-to-point and broadcast services. *Wireless Networks*, 10(4):359–366, July 2004.

[23] R. Braden, D. Clark, and S. Shenker. Integrated Services in the Internet Architecture: an Overview. RFC 1633 (Informational), June 1994.

[24] R. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin. Resource ReSerVation Protocol (RSVP) – Version 1 Functional Specification. RFC 2205 (Proposed Standard), September 1997. Updated by RFCs 2750, 3936, 4495, 5946.

[25] J. Brassil, S. Low, N. Maxemchuk, and L. O'Gorman. Electronic marking and identification techniques to discourage document copying. *IEEE Journal on Selected Areas in Communications*, 13(8):1495–1504, October 1995.

[26] I. Brodsky. *The History of Wireless: How Creative Minds Produced Technology for the Masses.* Telescope Books, 2008.

[27] D. Brown. SEC 1: Elliptic Curve Cryptography. `http://www.secg.org/`, May 2009.

[28] S. Buchegger and J. Le Boudec. Performance analysis of the CONFIDANT protocol. In *Proceedings of the 3rd ACM international symposium on Mobile ad hoc networking & computing (MobiHoc)*, pages 226–236, New York, NY, USA, 2002. ACM.

[29] S Buchegger and J. Le Boudec. A Robust Reputation System for P2P and Mobile Ad-hoc Networks. In *Proceedings of the 2nd Workshop on the Economics of Peer-to-Peer Systems*, June 2004.

[30] L. Buttyan and J. Hubaux. Nuglets: A Virtual Currency to Stimulate Cooperation in Self-Organizing Mobile Ad Hoc Networks. Technical Report DCS/2001/001, Swiss Federal Institute of Technology, 2001.

[31] M. Canales, J. Gállego, Á. Hernández-Solana, and A. Valdovinos. QoS provision in mobile ad hoc networks with an adaptive cross-layer architecture. *Wireless Networks*, 15(8):1165–1187, 2009.

[32] S. Capkun, L. Buttyan, and J. Hubaux. Self-organized public-key management for mobile ad hoc networks. *IEEE Transactions on Mobile Computing*, 2(1):52–64, January - March 2003.

[33] Z. Chen and K. Luk. *Antennas for base stations in wireless communications.* Communication engineering. McGraw-Hill, 2009.

[34] D. Chiu and R. Jain. Analysis of the increase and decrease algorithms for congestion avoidance in computer networks. *Computer Networks and ISDN Systems*, 17(1):1–14, June 1989.

[35] T. Clausen, C. Dearlove, and J. Dean. The Optimized Link State Routing Protocol version 2 (draft-ietf-manet-olsrv2-12). Internet-Draft, 2011.

[36] T. Clausen and U. Herberg. Vulnerability analysis of the optimized link state routing protocol version 2 (OLSRv2). In *IEEE International Conference on Wireless Communications, Networking and Information Security (WCNIS)*, pages 628–633, June 2010.

[37] T. Clausen and P. Jacquet. Optimized Link State Routing Protocol (OLSR). RFC 3626 (Experimental), October 2003.

[38] H. Cohen, G. Frey, and R. Avanzi. *Handbook of elliptic and hyperelliptic curve cryptography.* Discrete mathematics and its applications. Chapman & Hall/CRC, 2006.

[39] R. Collin. *Antennas and radiowave propagation.* McGraw-Hill series in electrical engineering. McGraw-Hill, 1985.

[40] I. Cox, J. Kilian, F. Leighton, and T. Shamoon. Secure spread spectrum watermarking for multimedia. *IEEE Transactions on Image Processing*, 6(12):1673–1687, December 1997.

[41] I. Cox and M. Miller. Electronic watermarking: the first 50 years. In *IEEE 4th Workshop on Multimedia Signal Processing (MMSP)*, pages 225–230, 2001.

[42] E. Crawley, R. Nair, B. Rajagopalan, and H. Sandick. A Framework for QoS-based Routing in the Internet. RFC 2386 (Informational), August 1998.

[43] Crypto++. Crypto++ Library is a free C++ class library of cryptographic schemes. `http://www.cryptopp.com`.

[44] V. Darmstaedter, J. Delaigle, D. Nicholson, and B. Macq. A Block Based Watermarking Technique for MPEG2 Signals: Optimization and Validation on Real Digital TV Distribution Links. In *Proceedings of the Third European Conference on Multimedia Applications, Services and Techniques (ECMAST)*, pages 190–206, London, UK, 1998. Springer Verlag.

[45] S. Deering and R. Hinden. Internet Protocol, Version 6 (IPv6) Specification. RFC 2460 (Draft Standard), December 1998. Updated by RFCs 5095, 5722, 5871.

[46] F. Deguillaume, G. Csurka, J. O'Ruanaidh, and T. Pun. Robust 3D DFT video watermarking. In *Proceedings of the SPIE Security and Watermarking of Multimedia Contents, vol. 3657*, pages 113–124, San Jose, CA, USA, January 1999.

[47] National Institute of Standards Department of Commerce and Information Technology Laboratory (ITL) Technology. Advanced Encryption Standard (AES). In *FIPS PUB 187*. Federal Information Processing Standards Publications, November 2001.

[48] J. Devore. *Probability and Statistics for Engineering and the Sciences.* Cengage Learning, 2011.

[49] D. Dhillon, T. Randhawa, M. Wang, and L. Lamont. Implementing a fully distributed certificate authority in an OLSR MANET. In *IEEE Wireless Communications and Networking Conference (WCNC)*, volume 2, pages 682–688, March 2004.

[50] W. Diffie and M. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, November 1976.

[51] D. Djenouri, L. Khelladi, and A. Badache. A survey of security issues in mobile ad hoc and sensor networks. *IEEE Communications Surveys Tutorials,*, 7(4):2–28, Quarter 2005.

[52] H. Dong, J. Gibson, and M. Kokes. SNR and bandwidth scalable speech coding. In *IEEE International Symposium on Circuits and Systems (ISCAS)*, volume 2, pages II–859–II–862, 2002.

[53] Sheetalkumar Doshi and Timothy X Brown. Minimum Energy Routing Schemes for a Wireless Ad Hoc Network. In *Proceedings of IEEE International Conference on Computer Communications (INFOCOM)*, New York, NY, USA, 23 - 27 June 2002.

[54] V. Dubendorf. *Wireless data technologies.* J. Wiley, 2003.

[55] M. Dworkin. Recommendation for Block Cipher Modes of Operation - Methods and Techniques. In *NIST Special Publication 800-38A*. National Institute of Standards and Technology, November 2001.

[56] L. Ertaul and N. Chavan. Security of ad hoc networks and threshold cryptography. In *International Conference on Wireless Networks, Communications and Mobile Computing (WIRLES)*, volume 1, pages 69–74, June 2005.

[57] J. Figueira, S. Greco, and M. Ehrgott. *Multiple criteria decision analysis: state of the art surveys.* International series in operations research & management science. Springer, 2005.

[58] FIPS. Federal Information Processing Standards Publications. `http://itl.nist.gov/fipspubs`.

[59] S. Floyd. TCP and explicit congestion notification. *SIGCOMM Computer Communication Review*, 24(5):8–23, October 1994.

[60] D. Gao, J. Cai, and K. Ngan. Admission control in IEEE 802.11e wireless LANs. *IEEE Network*, 19(4):6–13, July - August 2005.

[61] M. Gast. *802.11 wireless networks: the definitive guide.* Definitive Guide Series. O'Reilly, 2005.

[62] J. Gibson, A. Servetti, H. Dong, A. Gersho, T. Lookabaugh, and J. De Martin. Selective encryption and scalable speech coding for voice communications over multi-hop wireless links. In *IEEE Military Communications Conference (MILCOM)*, volume 2, pages 792–798, October - November 2004.

[63] V. Goyal. Multiple description coding: compression meets the network. *Signal Processing Magazine, IEEE*, 18(5):74–93, September 2001.

[64] ITU-T Recommendation H.262 and ISO/IEC 13 818-2 (MPEG-2). Generic Coding of Moving Pictures and Associated Audio Information - Part 2: Video. Technical report, 1994.

[65] ITU-T Rec. H.264 and ISO/IEC 14496-10:2005 (E) (MPEG-4 AVC). H.264 : Advanced video coding for generic audiovisual services. Technical report, 2005.

[66] L. Hanzo, Y. Akhtman, L. Wang, and M. Jiang. *MIMO-OFDM for LTE, WIFI and WIMAX: Coherent Versus Non-Coherent and Cooperative Turbo-Transceivers.* Wiley - IEE. John Wiley & Sons, 2010.

[67] F. Hartung and B. Girod. Watermarking of uncompressed and compressed video. *Signal Processing*, 66:283–301, May 1998.

[68] F. Hartung and M. Kutter. Multimedia watermarking techniques. *Proceedings of the IEEE*, 87(7):1079–1107, July 1999.

[69] L. Henzen, F. Carbognani, N. Felber, and W. Fichtner. VLSI hardware evaluation of the stream ciphers Salsa20 and ChaCha, and the compression function Rumba. In *2nd International Conference on Signals, Circuits and Systems (SCS)*, pages 1–5, November 2008.

[70] C. Hohendorf, E. Rathgeb, E. Unurkhaan, and M. Txen. Secure End-to-End Transport Over SCTP. *Journal of Computers*, 2(4), 2007.

[71] Y. Hu, A. Perrig, and D. Johnson. Packet leashes: a defense against wormhole attacks in wireless networks. In *IEEE Societies Twenty-Second Annual Joint Conference of the IEEE Computer and Communications (INFOCOM)*, volume 3, pages 1976–1986, March - April 2003.

[72] Y. Hu, A. Perrig, and D. Johnson. Rushing attacks and defense in wireless ad hoc network routing protocols. In *Proceedings of the 2nd ACM workshop on Wireless security (WiSe)*, pages 30–40, New York, NY, USA, 2003. ACM.

[73] Y. Hu, A. Perrig, and D. Johnson. Ariadne: a secure on-demand routing protocol for ad hoc networks. *Wireless Networks*, 11:21–38, January 2005.

[74] IETF. The Internet Engineering Task Force. `http://www.ietf.org`.

[75] R. Iqbal, S. Shahabuddin, and S. Shirmohammadi. Compressed-domain spatial adaptation resilient perceptual encryption of live H.264 video. In *10th International Conference on Information Sciences Signal Processing and their Applications (ISSPA)*, pages 472–475, May 2010.

[76] R. Iqbal, S. Shirmohammadi, A. El Saddik, and J. Zhao. Compressed-Domain Video Processing for Adaptation, Encryption, and Authentication. *IEEE Multimedia*, 15(2):38–50, April - June 2008.

[77] ISO/IEC. ISO/IEC 15444-1: Information Technology - JPEG 2000 image coding system - Part 1: Core coding system. Technical report,

International Organization for Standardization, Geneva, Switzerland., 2001.

[78] ISO/IEC. ISO/IEC 21000-7:2007 - Information technology - Multimedia framework (MPEG-21) - Part 7: Digital Item Adaptation. Technical report, International Organization for Standardization, 2007.

[79] ITU. International Telecommunication Union. `http://www.itu.int`.

[80] ITU. Open Systems Interconnection. Basic Reference Model:. Technical report, International Telecommunication Union, 1994. Recommendation X.200 Information technology. Open Systems Interconnection. Basic Reference Model: The basic model. - ITU-T, 1994.

[81] ITU. Perceptual evaluation of speech quality (PESQ), an objective method for end-to-end speech quality assessment of narrow-band telephone networks and speech codecs, February 2001.

[82] ITU-T. 5-, 4-, 3- AND 2-BITS SAMPLE EMBEDDED ADAPTIVE DIFFERENTIAL PULSE CODE MODULATION (ADPCM), 1990.

[83] ITU-T. Wideband coding for speech at around 16kbit/s using Adaptive Multi-rate Wideband (AMR-WB), 2002.

[84] D. Johnson and D. Maltz. Dynamic Source Routing in Ad Hoc Wireless Networks. In Imielinski and Korth, editors, *Mobile Computing*, pages 153–181. Kluwer Academic Publishers, 1996.

[85] R. Jurdak, C. Lopes, and P. Baldi. A survey, classification and comparative analysis of medium access control protocols for ad hoc networks. *IEEE Communications Surveys Tutorials*, 6(1):2–16, Quarter 2004.

[86] T. Kalker, G. Depovere, J. Depovere, and M. Maes. A video watermarking system for broadcast monitoring. In *Proceedings of the SPIE Security and Watermarking of Multimedia Contents*, volume 3657, pages 103–112, 1999.

[87] K. Kamphenkel, M. Blank, J. Bauer, and G. Carle. Adaptive encryption for the realization of real-time transmission of sensitive medical video streams. In *International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, pages 1–6, June 2008.

[88] Frank Kargl, Andreas Klenk, Stefan Schlott, and Michael Weber. Advanced Detection of Selfish or Malicious Nodes in Ad hoc Networks. In *Proceedings of the 1st European on Security in Ad-Hoc and Sensor Networks (ESAS)*, pages 152–165. Springer Verlag, 2004.

[89] T. Kaya, G. Lin, G. Noubir, and A. Yilmaz. Secure multicast groups on ad hoc networks. In *Proceedings of the 1st ACM workshop on Security of ad hoc and sensor networks (SASN)*, pages 94–102, New York, NY, USA, 2003. ACM.

[90] H. Krawczyk, M. Bellare, and R. Canetti. HMAC: Keyed-Hashing for Message Authentication. RFC 2104 (Informational), February 1997. Updated by RFC 6151.

[91] M. Kun, Y. Jingdong, and R. Zhi. The research and simulation of multipath-OLSR for mobile ad hoc network. In *IEEE International Symposium on Communications and Information Technology (ISCIT)*, volume 1, pages 540–543, October 2005.

[92] P. Kyasanur and N. Vaidya. Detection and handling of MAC layer misbehavior in wireless networks. In *Proceedings of International Conference on Dependable Systems and Networks*, pages 173–182, June 2003.

[93] G. Langelaar, R. Lagendijk, and J. Biemond. Real-Time Labeling of MPEG-2 Compressed Video. *Journal of Visual Communication and Image Representation*, 9(4):256–270, 1998.

[94] K. Lauter. The advantages of elliptic curve cryptography for wireless security. *IEEE Wireless Communications*, 11(1):62–67, February 2004.

[95] L. Lazos and R. Poovendran. Energy-aware secure multicast communication in ad-hoc networks using geographic location information. In *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volume 4, pages IV–201–4, April 2003.

[96] S. Lee, G. Ahn, X. Zhang, and A. Campbell. INSIGNIA: An IP-Based Quality of Service Framework for Mobile ad Hoc Networks. *Journal of Parallel and Distributed Computing*, 60:374–406, 2000.

[97] X. Li, Y. Wang, and O. Frieder. Efficient hybrid key agreement protocol for wireless ad hoc networks. In *Proceedings of Eleventh International Conference on Computer Communications and Networks*, pages 404–409, October 2002.

[98] J. Macker and S. Corson, editors. *Mobile ad-hoc networks (MANET).*, Salt Lake City, Utah, USA, December 2001. IETF.

[99] B. Macq and J. Quisquater. Cryptology for digital TV broadcasting. *Proceedings of the IEEE*, 83(6):944–957, June 1995.

[100] I. Mantin. Analysis of the stream cipher RC4. Master's thesis, The Weizmann Institute of Science, Rehovot, Israel, 2001.

[101] M. Marina and S. Das. On-demand multipath distance vector routing in ad hoc networks. In *Ninth International Conference on Network Protocols (ICNP)*, pages 14–23, November 2001.

[102] S. Marti, T. Giuli, K. Lai, and M. Baker. Mitigating routing misbehavior in mobile ad hoc networks. In *Proceedings of the 6th annual international conference on Mobile computing and networking (MobiCom)*, pages 255–265, New York, NY, USA, 2000. ACM.

[103] J. Meyer and F. Gadegast. Security mechanisms for Multimedia data with the Example MPEG-1 video, Project description of SECMPEG. 2000.

[104] P. Michiardi and R. Molva. Core: a collaborative reputation mechanism to enforce node cooperation in mobile ad hoc networks. In *Proceedings of the IFIP TC6/TC11 Sixth Joint Working Conference on Communications and Multimedia Security: Advanced Communications and Multimedia Security*, pages 107–121, Deventer, The Netherlands, The Netherlands, 2002. Kluwer, B.

[105] A. Mishra and K. Nadkarni. *Security in wireless ad hoc networks - A Survey*, chapter 30, pages 1–51. CRC Press, 2002.

[106] MPEG. Moving Picture Experts Group. `http://www.mpeg.org`.

[107] C. Murthy and B. Manoj. *Ad Hoc Wireless Networks: Architectures and Protocols*, pages 207–208,304. Prentice Hall PTR, 2004.

[108] NIST. National Institute of Standards and Technology. `http://www.nist.gov/index.html`.

[109] NS. The Network Simulator. `http://www.isi.edu/nsnam/ns/`.

[110] U.S. DEPARTMENT OF COMMERCE/National Institute of Standards and Technology. Data Encryption Standard (DES). In *FIPS PUB 46-3*. Federal Information Processing Standards Publications, October 1999.

[111] B. Oklander and M. Sidi. Jitter Buffer Analysis. In *Proceedings of the 17th International Conference on Computer Communications and Networks (ICCCN)*, pages 1–6, August 2008.

[112] L. Onural. An Overview of Research in 3DTV. In *14th International Workshop on Systems, Signals and Image Processing (IWSSIP)*, page 3, June 2007.

[113] R. Ostrovsky and M. Yung. How to withstand mobile virus attacks (extended abstract). In *Proceedings of the tenth annual ACM symposium on Principles of distributed computing (PODC)*, pages 51–59, New York, NY, USA, 1991. ACM.

[114] P. Papadimitratos and Z. Haas. Secure Routing for Mobile Ad hoc Networks. In *SCS Communication Networks and Distributed Systems Modeling and Simulation Conference (CNDS)*, pages 193–204, 2002.

[115] P. Papadimitratos and Z. Haas. Secure data transmission in mobile ad hoc networks. In *Proceedings of the 2nd ACM workshop on Wireless security (WiSe)*, pages 41–50, New York, NY, USA, 2003. ACM.

[116] P. Papadimitratos and Z. Hass. *Securing Mobile Ad Hoc Networks*, chapter 31, pages 1–17. CRC Press, 2002.

[117] A. Penrig, D. Song, and D. Tygar. ELK, a new protocol for efficient large-group key distribution. In *Proceedings of IEEE Symposium on Security and Privacy (SECPRI)*, pages 247–262, 2001.

[118] C. Perkins. *RTP: audio and video for the internet.* Kaleidoscope Series. Addison-Wesley, 2003.

[119] C. Perkins, E. Belding-Royer, and S. Das. Ad hoc On-Demand Distance Vector (AODV) Routing. RFC 3561 (Experimental), July 2003.

[120] C. Perkins and P. Bhagwat. Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers. In *Proceedings of the conference on Communications architectures, protocols and applications (SIGCOMM)*, pages 234–244, New York, NY, USA, 1994. ACM.

[121] D. Perkins and H. Hughes. A survey on quality-of-service support for mobile ad hoc networks. *Wireless Communications and Mobile Computing*, 2(5):503–513, 2002.

[122] A. Perrig, R. Canetti, J. Tygar, and D. Song. The TESLA Broadcast Authentication Protocol. *RSA CryptoBytes*, 5, Summer 2002.

[123] L. Peterson and B. Davie. *Computer networks: a systems approach.* The Morgan Kaufmann series in networking. Morgan Kaufmann Publishers, 2003.

[124] B. Pfitzmann. Information Hiding Terminology - Results of an Informal Plenary Meeting and Additional Proposals. In *Proceedings of the First International Workshop on Information Hiding*, pages 347–350, London, UK, 1996. Springer Verlag.

[125] M. Rabin. Efficient dispersal of information for security, load balancing, and fault tolerance. *Journal of the ACM*, 36:335–348, April 1989.

[126] L. Reddy and S. Raghavan. SMORT: Scalable multipath on-demand routing for mobile ad hoc networks. *Ad Hoc Netw.*, 5:162–188, March 2007.

[127] R. Rivest. The RC5 Encryption Algorithm. RFC 2460 (Draft Standard). Mass.02139 (Revised March 20, 1997).

[128] F. Ros. UM-OLSR: an implementation of the OLSR (IETF RFC 3626) protocol for the ns-2 Network Simulator. `http://masimum.inf.um.es/fjrm/`.

[129] E. Royer and C. Toh. A review of current routing protocols for ad hoc mobile wireless networks. *IEEE Personal Communications*, 6(2):46–55, April 1999.

[130] T. Saaty. *Fundamentals of decision making and priority theory with the analytic hierarchy process.* The analytic hierarchy process series. RWS Publications, 2001.

[131] N. Santoro. *Design and analysis of distributed algorithms.* Wiley series on parallel and distributed computing. Wiley-Interscience, 2007.

[132] K. Sanzgiri, B. Dahill, B. Levine, C. Shields, and E. Belding-Royer. A secure routing protocol for ad hoc networks. In *Proceedings of 10th IEEE International Conference on Network Protocols (ICNP)*, pages 78–87, November 2002.

[133] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson. RTP: A Transport Protocol for Real-Time Applications. RFC 3550 (Standard), July 2003. Updated by RFCs 5506, 5761, 6051, 6222.

[134] H. Schwarz, D. Marpe, and T. Wiegand. Overview of the Scalable Video Coding Extension of the H.264/AVC Standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 17(9):1103–1120, September 2007.

[135] A. Shamir. How to share a secret. *Communications of the ACM*, 22:612–613, November 1979.

[136] K. Shring. H.264/AVC JM Reference Software. `http://iphome.hhi.de/suehring/tml/download/`.

[137] J. Solinas. Efficient Arithmetic on Koblitz Curves. *Designs, Codes and Cryptography*, 19:195–249, March 2000.

[138] G. Spanos and T. Maples. Performance Study of a Selective Encryption Scheme for the Security of Networked, Real-Time Video. *Fourth International Conference on Computer Communications and Networks*, pages xviii+683, September 1995.

[139] W. Stallings. *Cryptography and network security: principles and practice.* Prentice Hall, 2011.

[140] M. Steiner, G. Tsudik, and M. Waidner. Diffie-Hellman key distribution extended to group communication. In *Proceedings of the 3rd ACM conference on Computer and communications security (CCS)*, pages 31–37, New York, NY, USA, 1996. ACM.

[141] R. Stewart. Stream Control Transmission Protocol. RFC 4960 (Proposed Standard), September 2007. Updated by RFC 6096.

[142] R. Stewart, M. Ramalho, Q. Xie, M. Tuexen, and P. Conrad. Stream Control Transmission Protocol (SCTP) Partial Reliability Extension. RFC 3758 (Proposed Standard), May 2004.

[143] R. Stewart, Q. Xie, M. Tuexen, S. Maruyama, and M. Kozuka. Stream Control Transmission Protocol (SCTP) Dynamic Address Reconfiguration. RFC 5061 (Proposed Standard), September 2007.

[144] K. Su, D. Kundur, and D. Hatzinakos. A novel approach to collusion-resistant video watermarking. In *Proceedings of the SPIE Security and Watermarking of Multimedia Contents*, pages 491–502, 2002.

[145] L. Tang. Methods for encrypting and decrypting MPEG video data efficiently. In *Proceedings of the fourth ACM international conference on Multimedia (MULTIMEDIA)*, pages 219–229, New York, NY, USA, 1996. ACM.

[146] E. Unurkhaan, E. Rathgeb, and A. Jungmaier. Secure SCTP A Versatile Secure Transport Protocol. *Telecommunication Systems*, 27:273–296, 2004. 10.1023/B:TELS.0000041012.85567.54.

[147] B. Vaidya, D. Choi, J. Park, and S. Han. Multipath Routing Scheme for Wireless Multihop Network. In *Proceedings of the international conference on Computational Science and Its Applications, Part II (ICCSA)*, pages 433–445, Berlin, Heidelberg, Germany, 2008. Springer Verlag.

[148] B. Vaidya, M. Denko, and J. Rodrigues. Secure Framework for Voice Transmission over Multipath Wireless Ad-Hoc Network. In *IEEE Global Telecommunications Conference (GLOBECOM)*, pages 1–6, December 2009.

[149] B. Vaidya, S. Yeo, D. Choi, and S. Han. Robust and secure routing scheme for wireless multihop network. *Personal Ubiquitous Comput.*, 13:457–469, October 2009.

[150] J. Wang, Y. Wang, and K. Nahrstedt. Quantitative Study of Differentiated Service Model Using Ultrasan. Technical report, Champaign, IL, USA, 2001.

[151] T. Wiegand, G. Sullivan, G. Bjontegaard, and A. Luthra. Overview of the H.264/AVC video coding standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 13(7):560–576, July 2003.

[152] M. Wiener. Performance Comparison of Public-key Cryptosystems. `http://www.rsa.com/rsalabs/pubs/cryptobytes.html`, 1998.

[153] C. Wong, M. Gouda, and S. Lam. Secure group communications using key graphs. *SIGCOMM Computer Communication Review*, 28(4):68–79, October 1998.

[154] ANSI X9.52. Triple Data Encryption Algorithms Modes of Operation. `https://www.x9.org/home/`, 1998.

[155] H. Xiao, W. Seah, A. Lo, and K. Chua. A flexible quality of service model for mobile ad-hoc networks. In *Proceedings of the IEEE 51st Vehicular Technology Conference (VTC)*, volume 1, pages 445–449, Tokyo, Japan, Spring 2000.

[156] J. Yan and H. Heys. Hardware Implementation of the Salsa20 and Phelix Stream Ciphers. In *Canadian Conference on Electrical and Computer Engineering (CCECE)*, pages 1125–1128, April 2007.

[157] H. Yang, H. Luo, F. Ye, S. Lu, and L. Zhang. Security in mobile ad hoc networks: challenges and solutions. *IEEE Wireless Communications,*, 11(1):38–47, February 2004.

[158] H. Yang, X. Meng, and S. Lu. Self-organized network-layer security in mobile ad hoc networks. In *Proceedings of the 1st ACM workshop on Wireless security (WiSE)*, pages 11–20, New York, NY, USA, 2002. ACM.

[159] Z. Yao, J. Jiang, P. Fan, Z. Cao, and V. Li. A neighbor-table-based multipath routing in ad hoc networks. In *The 57th IEEE Semiannual Vehicular Technology Conference (VTC)*, volume 3, pages 1739–1743, April 2003.

[160] Z. Ye, S.V. Krishnamurthy, and S.K. Tripathi. A framework for reliable routing in mobile ad hoc networks. In *IEEE Societies Twenty-Second Annual Joint Conference of the IEEE Computer and Communications*, volume 1 of *INFOCOM '03*, pages 270 – 280, March - April 2003.

[161] J. Yi, A. Adnane, S. David, and B. Parrein. Multipath optimized link state routing for mobile ad hoc networks. *Ad Hoc Networks*, 9:28–47, January 2011.

[162] S. Yi and R. Kravets. MOCA : Mobile Certificate Authority for Wireless Ad Hoc Networks. In *2nd Annual PKI Research Workshop Program (PKI)*, pages 65–79, Gaithersburg, MD, 2003.

[163] S. Yi, P. Naldurg, and R. Kravets. Security-aware ad hoc routing for wireless networks. In *Proceedings of the 2nd ACM international symposium on Mobile ad hoc networking & computing (MobiHoc)*, pages 299–302, New York, NY, USA, 2001. ACM.

[164] H. Yin, C. Lin, F. Qiu, J. Liu, G. Min, and B. Li. CASM: a content-aware protocol for secure video multicast. *IEEE Transactions on Multimedia*, 8(2):270–277, April 2006.

[165] B. Yu. Establishement of elliptic curve cryptosystem. In *IEEE International Conference on Information Theory and Information Security (ICITIS)*, pages 1165–1167, December 2010.

[166] H. Zhang, J. Zhao, and O. Yang. Adaptive Rate Control for VoIP in Wireless Ad Hoc Networks. In *IEEE International Conference on Communications (ICC)*, pages 3166–3170, May 2008.

[167] L. Zhou and Z. Haas. Securing ad hoc networks. *IEEE Network*, 13(6):24–30, November - December 1999.

[168] S. Zhu, S. Setia, S. Xu, and S. Jajodia. GKMPAN: an efficient group rekeying scheme for secure multicast in ad-hoc networks. In *The First Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services (MOBIQUITOUS)*, pages 42–51, August 2004.

[169] P. Zimmermann. *The official PGP user's guide.* MIT Press, 1995.

Appendix A

Confidence Level

A.1 Confidence Level

Confidence level indicates the reliability of an interval estimate. An interval estimate, also called confidence interval (CI), is a bounded range of plausible values, as opposed to single point estimate. CI is often considered more sensible then point estimate. Most commonly used confidence levels are 90%, 95% and 99%. A higher confidence level indicates higher reliability of the estimation [48].

Random samples can be obtained for observation from a normally distributed population, with population mean μ and population standard deviation σ. CI methodology shows that, regardless of the random sample size n, the random sample mean $\bar{X}$ is also normally distributed with expected mean value μ and standard deviation $\sigma/\sqrt{n}$.

Standardizing $\bar{X}$ gives,

$$Z = \frac{\bar{X} - \mu}{\sigma/\sqrt{n}} \tag{A.1}$$

where Z is the standard normal variable. Z lies between $-z_{\alpha/2}$ and $z_{\alpha/2}$ in the standard normal distribution.

$$P\left(-z_{\alpha/2} < Z < z_{\alpha/2}\right) = 1 - \alpha \tag{A.2}$$

Here, α is a small nonnegative number. CI is usually presented in the form $100 \cdot (1-\alpha)\%$. z is the standard normal critical value. $z_{\alpha/2}$ follows from the cumulative standard normal curve area (in the standard normal distribution).

A 95% confidence level implies that the probability of random interval $\left(\bar{X} - 1.96 \cdot \sigma/\sqrt{n},\ \bar{X} + 1.96 \cdot \sigma/\sqrt{n}\right)$ including μ is 0.95 (A.3).

$$P\left(\bar{X} - 1.96 \cdot \sigma/\sqrt{n} < \mu < \bar{X} + 1.96 \cdot \sigma/\sqrt{n}\right) = 0.95 \qquad \text{(A.3)}$$

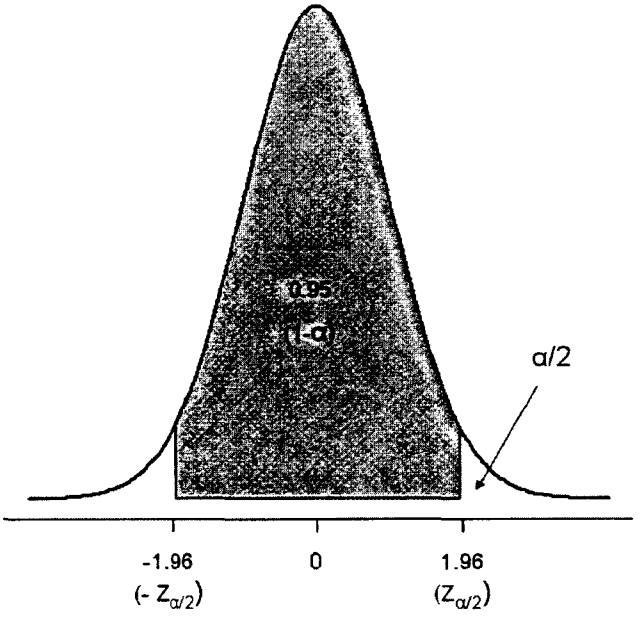


Figure A.1: Standard normal curve between -1.96 and 1.96.

Equation A.3 is derived from Equations A.1 and A.2. The area under the standard normal curve between -1.96 and 1.96 is 0.95 (Figure A.1). This can also be expressed as,

$$P\left(-1.96 < Z < 1.96\right) = P\left(-1.96 < \frac{\bar{X} - \mu}{\sigma/\sqrt{n}} < 1.96\right) = 0.95 \qquad \text{(A.4)}$$

Sample Size

A particular application of CI is determining the sample size for the desired confidence level (Equation A.5).

$$n = \left(2z_{\alpha/2} \cdot \frac{\sigma}{w}\right)^2 \tag{A.5}$$

Equation A.5 is derived from Equations A.1 and A.2. Here, w is the width of the interval. For 95% CI, the interval extends $1.96 \cdot \sigma/\sqrt{n}$ to each side of the sample mean $\bar{x}$; so, width of the interval, $w = 2(1.96) \cdot \sigma/\sqrt{n}$.

Example:

Based on multiple simulation results for transmitted video data that were encrypted using Triple-DES, we collect data for average packet reception rate (i.e., average number of packets received each second). The accumulated data is normally distributed with standard deviation 11.09. We would like to determine the sample size necessary to ensure that the resulting 95% CI has a width of at most 15.

Here, standard deviation, $\sigma = 11.09$, width of CI, $w = 15$ and for 95% confidence level, $2z_{\alpha/2} = 1.96$. Using Equation A.5,

$$\text{sample size, } n = \left(2 \cdot (1.96) \cdot \frac{11.09}{15}\right)^2 = 8.41$$

Thus, required sample size is 9 (n must be an integer), i.e., we would require to obtain 9 independent simulation results to satisfy 95% confidence level. Similarly, for the above example, the sample size for 99% confidence level is 15.