EXPERIMENTAL QOS-BASED COMPARISON OF OLSR AND B.A.T.M.A.N. MESH ROUTING ALGORITHMS FOR MULTIMEDIA HOME NETWORKS

A Thesis

by

Sırrı Erdem Ulusoy

Submitted to the
Graduate School of Sciences and Engineering
In Partial Fulfillment of the Requirements for
the Degree of

Master of Science

in the
Department of Electrical and Electronics Engineering

Özyeğin University
August 2013

EXPERIMENTAL QOS-BASED COMPARISON OF OLSR AND B.A.T.M.A.N. MESH ROUTING ALGORITHMS FOR MULTIMEDIA HOME NETWORKS

Approved by:

Assist. Prof. Dr. Ali Özer Ercan, Advisor
Department of Electrical and Electronics Engineering
Özyeğin University

Prof. Dr. Tanju Erdem
Department of Computer Science
Özyeğin University

Assoc. Prof. Dr. Oğuz Sunay
Department of Electrical and Electronics Engineering
Özyeğin University

Date Approved: 2 August 2013

To my parents and sister

ABSTRACT

Wireless Mesh Networks (WMNs) allow communication of two nodes via a multi-hop route, without the need for an infrastructure such as base stations. Routing has been a fundamental issue for this type of communication networks. In this thesis, we compare two well-known routing algorithms for WMNs, namely OLSR and B.A.T.M.A.N., in their achieved QoS for video streaming, on a real life WMN test-bed. Our test-bed consists of seven mesh access points (MAPs) placed on several rooms spread on two floors of Ozyegin University Engineering Building. We installed B.A.T.M.A.N.-Adv, a layer two implementation of B.A.T.M.A.N. routing algorithm, and OLSRd, a layer three implementation of OLSR routing algorithm, on the MAPs. We used a laptop as a video server and another laptop placed on the other corner of the building as a client. A test video is streamed from the server to the client using the said routing algorithms several times and the sample means and standard deviations of the PSNR values between compressed and received videos as well as between uncompressed and received videos are calculated. In the calculation of the PSNR values, we developed a novel method to overcome the problem of mismatched frames in the case of dropped frames. The experiments are repeated for varying levels of video compression rates, cross-traffic levels on the network, and by changing settings of the RTS/CTS handshaking mechanism. The PSNR values are reported and compared between the two routing algorithms for these different cases, and their effects are discussed.

ÖZETÇE

Kablosuz Çokgen Bağlantılı Ağlar (WMN), baz istasyonu gibi bir altyapı ihtiyacı olmaksızın iki düğümün çok hoplamalı rota üzerinden iletişimini sağlar. Rotalama bu tür iletişim ağlarında temel bir konu olmuştur. Bu tezde, WMN'ler için iyi bilinen iki rotalama algoritmasını, yani OLSR ve B.A.T.M.A.N., gerçek bir WMN deney düzeneğinde akan bir videoda sağladıkları PSNR başarılarına göre karşılaştırdık. Deney düzeneğimiz, Özyeğin Üniversitesi Mühendislik Fakültesi'nde birbirinden mesafeli olarak yerleştirilmiş yedi adet Çokgen Bağlantı Erişim Noktası'ndan (MAP) oluşmaktadır. MAP'lerin üzerine, B.A.T.M.A.N. rotalama algoritmasının ikinci katman uygulaması olan B.A.T.M.A.N.-Adv ve OLSR rotalama algoritmasının üçüncü katman uygulaması olan OLSRd yükledik. Bir dizüstü bilgisayarı video sunucusu olarak ve binanın diğer köşesine yerleştirilmiş diğer bir dizüstü bilgisayarı istemci olarak kullandık. Deney videosu sunucudan istemciye bahsedilen rotalama algoritmaları kullanılarak birçok kez kesintisiz olarak gönderildi ve sıkıştırılmış-alınmış videolar arasındaki ve sıkıştırılmamış-alınmış videolar arasındaki örnek ortalamaları ve standart sapmaları hesaplandı. PSNR değeri hesabında, kaybolan framelerden dolayı eşleşmeyen çerçevelerin sorununu çözmek için özgün bir yöntem geliştirdik. Deneyler değişik seviyelerde video sıkıştırma oranları için, ağdaki farklı çapraz trafik hızları için ve RTS/CTS tokalaşma protokolünün ayarları değiştirilerek tekrarlandı. PSNR değerleri sunuldu ve bahsedilen farklı durumlar için rotalama algoritmaları arasında karşılaştırıldı ve etkileri ele alındı.

ACKNOWLEDGEMENTS

First, I would like to thank my advisors, Asst. Prof. Dr. Ali Özer Ercan and Assoc. Prof. Dr. M. Oğuz Sunay for their guidance, support, motivation and patience to complete my M.Sc. studies and thesis. They helped me to find this intereting topic and during the process, and countless times they showed me the way out from dead ends.

I would also like to thank Prof. Dr. Tanju Sunay for taking the time as commitee member in my thesis defence.

I would like to thank to Asst. Prof. Dr. Ali Özer Ercan, Assoc. Prof. Dr. Oğuz Sunay, Assist. Prof. Dr. İsmail Arı and Assist. Prof. Dr. Yasemin Şengül for letting me placing test devices in their rooms.

Also I would like to Volkan Yazıcı for sharing his knowledge on Computer Science, especially on Linux and various programming tools.

I also want to thank you my friends and cousins, Alican Gürsoy, Furkan Şahin, Kyoomars Alizadeh Noghani, Süleyman Taşkent, Ayşe Karagöz and Haydar Pekdemir for proofreading and helping me to fix my thesis.

Also I want to thank my aunts, Leyla Kuzucanlı, Hatice Gürsoy, my uncles İbrahim Ulusoy, Erol Ulusoy and their families for being my family in İstanbul.

I would also like to thank my collegues and friends, Fatih Temizkan, Asım Yıldız, Gökhan Güner, Bilgiday Yüce, Okan Palaz, Mustafa Kaygısız, Koray Kökten, Serkan Kırkgül, Kıvanç Çakmak and Ali Arsal for their friendship and amusing moments that we had together during this stressful period

Last but not least I would like to thank to my parents, Rafet Ulusoy and Hatice Ulusoy and my sister Gökçe Ulusoy for their endless support during my whole life.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

With the advance of mobile devices, Wireless Local Area Networks (WLANs) altered people's habit to connect to the Internet distinctly in the last two decades. Wireless connection is used as the last hop to connect to the Internet. To cover large areas, single Acces Point (AP) may not be enough to cover all area so to cover large areas many APs may be used. When these APs are connected to the Wireless Distribution System (WDS) via cable, it increases installation and maintenance costs. So Wireless Mesh Networks (WMNs) where Mesh Points (MPs) relay each other's and client's data in the same network are started to be used to cover large areas.

WMNs are used to extend the coverage area of a single AP by relaying downlink and uplink data in multihop. They also set a WLAN in themselves. Since there is multihop communication, nodes should be aware of other nodes in the network and the paths to reach them. This requires routing and due to their nature in wireless networks, link states and conditions change rapidly. Due to this fact, routing algorithms proposed for wired networks don't work well on wireless networks. A routing algorithm proposed for wireless networks should be aware of the link states in the network.

There are plenty of routing algorithms proposed or adapted from Ad hoc Network routing algorithms for WMNs. Two of most common routing algorithms are B.A.T.M.A.N. and OSLR routing algorithms. Performances of these two and other routing algorithms are compared even in simulation and test environments. But best to our knowledge, there is no detailed Quality of Service (QoS) comparison of these routing algorithms for streaming video applications. In this study, we built a 802.11g

Multimedia WMN test bed composed of 7 Mesh Access Points (MAPs) and compared these two routing algorithms on their Peak signal-to-noise ratio (PSNR) achievements for streaming videos. In our test, we tested three different video encoding bit rate and to see how the routing algorithms respond to the cross traffic in the network, we created UDP traffic at 4 different rates and TCP traffic and observed their effect on the PSNR of the received video. Finally to see effect of RTS/CTS handshake mechanism, we turned it in for every transmission and observed how the recieved video is effected. Since wireless medium is a lossy medium, during video streaming some of the frames are dropped and as a part of our study, we proposed a novel method to match the recieved frames with the frames in the encoded video.

1.1 Overview of IEEE 802.11

IEEE 802 is the standard family for LANs and MANs, up to date version of IEEE 802.11 standard is [1]. Basically, it describes data link layer and physical layer for these networks. It has several work groups to define standards for different parts of LANs and MANs.

WG 802.11 is the work group which studies about WLANs and IEEE 802.11 is the physical layer standard for WLANs. IEEE 802.11 is proposed in 1997 and since 1997, WG 802.11 has been adding new amendments to standard 802.11 and these amendments aren't restricted with physical layer. For example, the IEEE 802.11e amendment defines QoS for WLANs and IEEE 802.11i defines the security protocol, WPA2 for WLANs.

1.1.1 Overview of IEEE 802.11 Physical Layer Standards

There are 6 amendments to define the physical layer properties. These amendments are IEEE 802.11a/b/g/n/ac/ad. IEEE 802.11ac is currently a draft standard, IEEE 802.11ad hasn't been published yet and is expected to be published in February 2014. These physical layer amendments define physical properties, such as operating

frequency, bandwidth, allowable MIMO streams and modulation of the transmitted signal by wireless devices. Data rate of the transmitted signal is highly dependent on these physical properties. For example, doubling bandwidth may double the data rate or using a modulation technique which uses the spectrum more efficiently may result in higher data rate. General properties of 802.11 physical layer amendments can be found in Table 1. During our tests, we used 802.11g as physical layer standard. It was the most suitable standard during our tests because it has higher data rate with respect to other amendments working in 2.4GHz frequency. Also, in our experiment setup, we have used OpenWrt, a linux distribution for embeded devices, and implementation of 802.11g was more stable than implementation of 802.11n in OpenWrt.

Table 1: General Properties of Physical Layer Amendments of 802.11 Standard

Amendment	Frequency (GHz)	Bandwidth (MHz)	Maximum Possible Data Rate Per Stream (Mbits/s)	Allowable MIMO Streams	Modulation
a	5	20	54	1	OFDM
b	2.4	20	11	1	DSSS
g	**2.4**	**20**	**54**	**1**	**OFDM, DSSS**
n	2.4, 5	20, 40	150	1	OFDM
ac (DRAFT)	5	20, 40, 80, 160	866.7	8	OFDM
ad (expected)	2.5, 5, 60		7000		

1.1.1.1 IEEE 802.11g Amendment

IEEE 802.11g is an amendment proposed by WG 802.11 in 2003. The standard proposes a modulation scheme for 2.4 GHz band. 2.4 GHz band is one of the ISM bands which are radio bands reserved for license free use of radio frequency for industrial, scientific and medical purposes. Microwave ovens, radio frequency remote controls

are the most common examples where ISM bands are used. Devices using ISM band are highly densely distributed and powerful emissions of these devices may result in electromagnetic interference. To prevent interference of these devices to each other, there are power limitations in their standards, since wireless devices use the same band, they are subjected to the similar limitations.

IEEE 802.11b amendment is the first amendment of IEEE 802.11 standard defined in 2.4GHz and the maximum data rate is 11 Mbits/s. Both 802.11b and 802.11g amendments have same bandwidth and channel spectrum usage but in IEEE 802.11g amendment, OFDM modulation scheme is defined in addition to DSSS modulation scheme which was defined in IEEE 802.11b standard. OFDM modulation scheme was defined by IEEE 802.11a amendment in 5 GHz and it is directly copied in IEEE 802.11g amendment.

Direct Sequence Spread Spectrum: DSSS is a spread spectrum modulation technique. In spread spectrum techniques, the information is spread over a larger bandwidth than its own. In spread spectrum modulation techniques, energy is spreaded over a larger spectrum and this results in that signal has energy as low as noise and this property not only helps to decrease interference, noise and jamming but also makes it harder to be detected. In Figure 1, power distribution of narrow-band waveform and spread waveform over frequency domain can be seen.

In DSSS, to spread the signal over the spectrum, the signal is multiplied with a pseudo-random sequence of 1 and -1 values. Each of these pseudo-random value is called chip and their duration is much shorter than the duration of the signal. The pseudo-random sequence is preshared between receiver and transmitter so when the receiver receives the signal, it multiplies the signal with the same sequence and constructs the original signal.

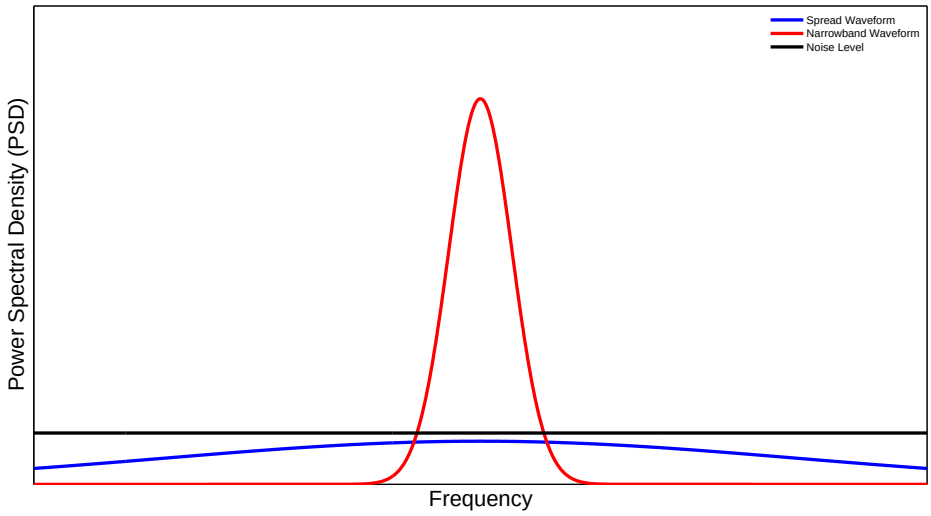

Figure 1: Power Spectral Density of Narrow-band and Spread Waveforms

Orthogonal Frequency Division Multiplexing: Orthogonal is a method to transfer data using multiple narrowband subcarriers instead of using one wideband carrier. It can be thought as instead of using the whole frequency spectrum just for one signal, in OFDM signal is divided into pieces and each piece is transmitted through one subcarrier in a longer period in other words instead of one rapidly modulated signal, multiple slowly modulated signals are sent. This technique makes the signal propagation stronger in severe channel conditions such as narrowband interference and frequency selective fading due to multipath. The most important disadvantages of OFDM are to be sensitive to Doppler effect and to have high PAPR. Figure 2 shows power spectral density of OFDM signal.

Spectrum Usage of IEEE 802.11 Standard in 2.4 GHz Band IEEE 802.11 standard defines 13 channels separated by 5 MHz from each other. These channels are numbered from 1 to 13. All of these channels 22 MHz bandwidth so only three non-overlapping channels are available in this band. To reduce the interference between devices only non-overlapping channels 1, 6 and 11 are used. The spectrum of these

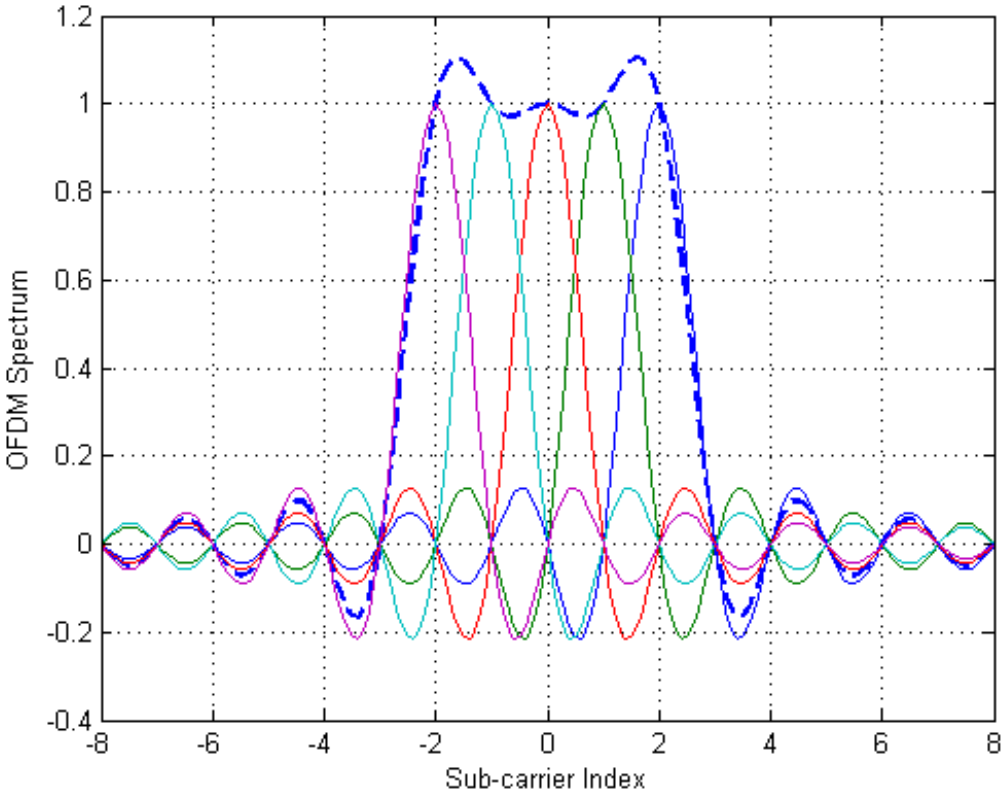

Figure 2: Power Spectral Density of OFDM Signal (Taken from [2])

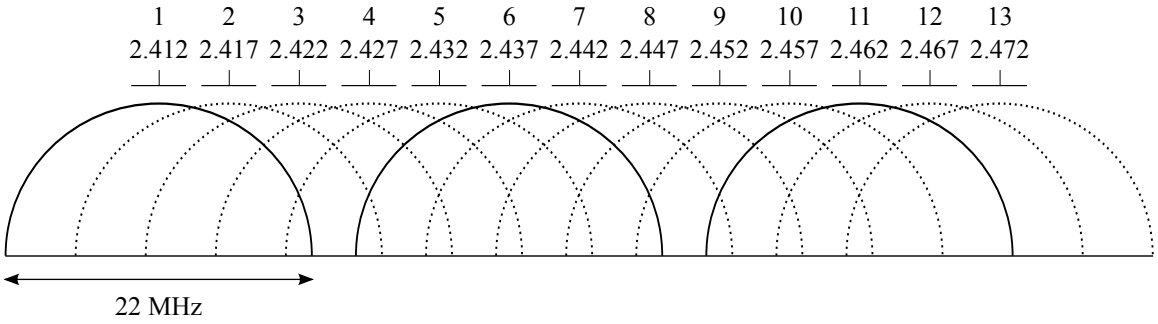

Figure 3: Spectrum of channels in 2.4 GHz

channels can be seen in Figure 3. As it can be seen from Figure 3, Channel 1 overlaps with Channels 2-5, Channel 6 overlaps with Channels 2-5 and 7-10 and Channel 11 overlaps with Channels 7-10 and Channels 12-13.

1.2 Wireless Mesh Networks

A WMN is a communication network which contains radio devices that not only transmit their own data but also relay other nodes' data. Aim of these networks is by help of the relaying nodes without cable connection to extend the service area of a gateway which is connected to other networks and making communication possible between the nodes in the network without cable connection. Wireless Mesh Networks are used in LANs, MANs and WANs. All of these networks has different requirements so properties of mesh networks for the each network type is defined under different standards. WMNs for LANs are defined by the amendment IEEE 802.11s. In Figure 4, an example of mesh network consisting of mesh network elements and a server can be seen.

Devices in mesh networks can be grouped in four;

1. **Mesh Gateway: Mesh gateway is a node which has wired or wireless connection to other networks.**

2. **Mesh Point: Mesh point is a node which is capable to relay other**

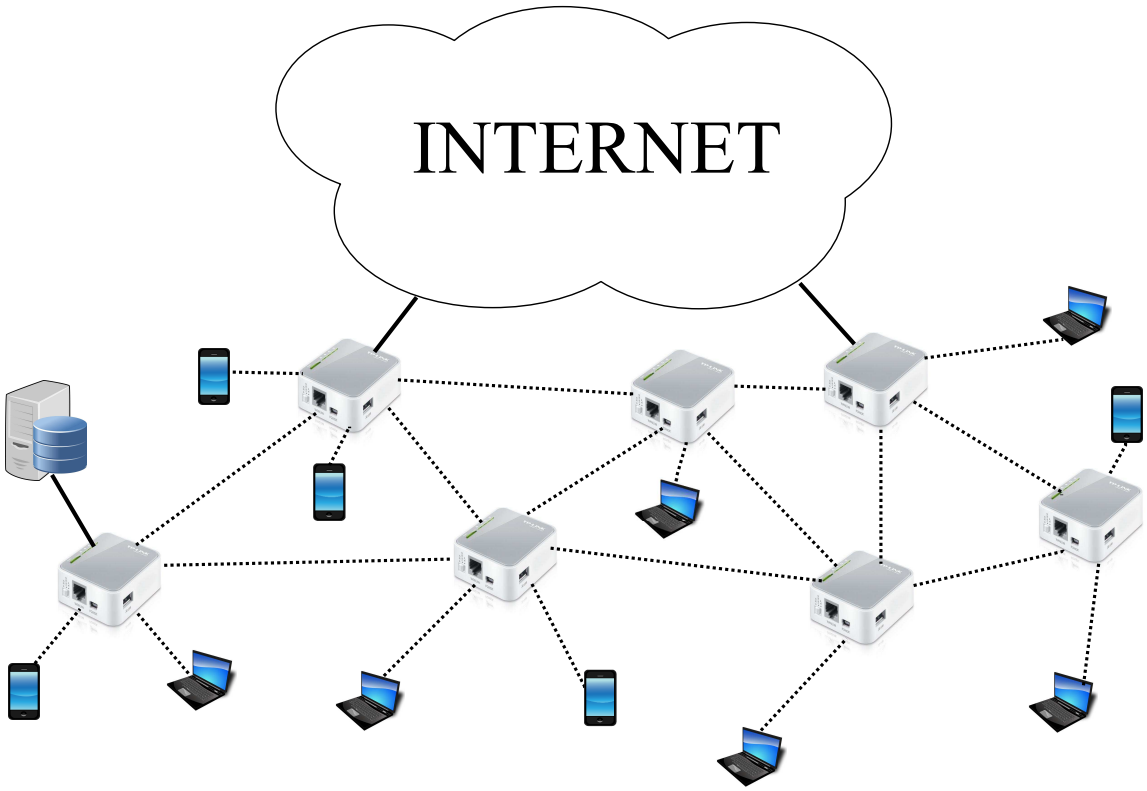

Figure 4: Mesh Network Example

nodes' data but not capable to provide service to clients.

3. **Mesh Access Point: Mesh access point is a node which is capable both to relay other nodes' data and to provide service to clients.**

4. **Client: Client is the node used by end user, it may or may not relay other nodes' data.**

Mesh gateways, mesh points and mesh access points build infrastructure of a WMN. The nodes building the infrastructure have no mobility or very low mobility. In a mesh network, data packets are likely to be transferred through same path and during the transfers quality of the links oscillates between good and bad states. Routing algorithms in mesh networks should be aware of this phenomenon.

1.3 Routing in Mesh Networks

Different network structures require different routing techniques for efficient delivery of the packets. A routing method which is very good for a specific network structure might be very poor for another network structure. Due to the nature of wireless communication and multihop wireless transmission purpose of WMNs, the metric used for routing should consider somehow the link quality. In this section, we will discuss routing protocols that we tested and explain how they consider the link quality in their routing metric.

1.3.1 OLSR

OLSR is a proactive, link state routing algorithm, OLSR is defined in [3]. Since it is a link state routing protocol, the links between the nodes in the network are announced through the network. The routing is performed to optimize pure link state protocol by using only certain nodes to flood the control messages and only including link state information of these flooding nodes in the control packets. These certain nodes are

called MPRs and they are the main concept of OLSR algorithm. In OLSR, routing is calculated to achieve minimum number of hops.

In OLSR protocol, the control messages are flooded only periodically, in the cases of link failures or additions, no extra control message is generated and it reduces the routing overhead. Control messages are sent by unreliable transmission because since they are periodical, the system is robust in case of some control messages are dropped. Also, in the control messages that is flooded over the network sequence numbers are used so if control messages are received out of order, the old message can be ignored. Each nodes calculates its own routing table and this helps to compensate mobility of nodes if the mobile node can be traced by even its neighbors.

1.3.1.1 MPR Concept

In default routing algorithms to flood topology control messages, each node broadcasts their own control messages and relays the messages coming from other nodes. This results in bulky generation of control messages and makes very high load in the network. To reduce this bulky information, OLSR routing protocol suggests to use MPR concept. In MPR concept, nodes select some of their neighbors to retransmit their topology control messages. Using a node in this selected MPR set, node can reach any of its two-hop neighbors. By using MPRs, the required number of retransmissions is reduced when flooding the control traffic through the network.

1.3.1.2 Protocol Operation

After the protocol starts, nodes send each other "HELLO" packets. Except the header the "HELLO" packets include the information of neighbor nodes, neighbor nodes with bi-directional link and nodes selected as MPRs. "HELLO" packet format can be seen in Figure 5. "HELLO" packets have three main duties first one is to announce their neighbor nodes so neighbor nodes can determine their neighbors with whom they are connected via bi-directional links, the second main duty is to announce

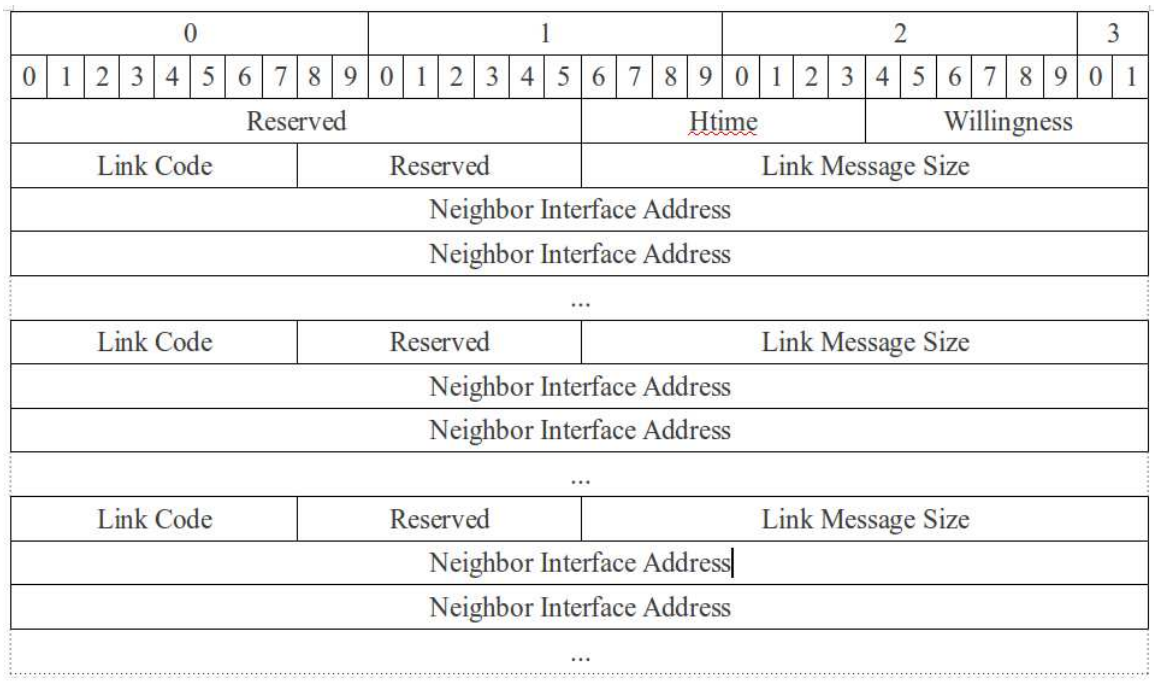

Figure 5: HELLO packet structure for OLSR protocol (Taken from [3])

neighbor nodes with bi-directional links so other neighbors who has bi-directional link connection with the node can determine their two-hop neighbors with bi-directional link connection and the final main duty is to announce MPR nodes so MPR nodes learn that they are selected as MPRs.

To calculate its MPR set, a node needs to know all of its neighbors with whom it has bi-directional link connection and all of their bi-directional neighbors (2-hop neighbors). All of these information is included in "HELLO" messages, i.e. the information in "HELLO" messages is necessary and enough to select MPR set. The MPR set doesn't have to be optimum but the smaller it is, the more beneficial the algorithm is. A way to calculate the MPR set is to choose all of the neighbors as MPR and eliminating the ones instead of whom an alternative neighbor can be used, another way can be to put the neighbors who are the only way to some 2-hop neighbors and trying to find MPRs for uncovered two-hop neighbors. The MPR

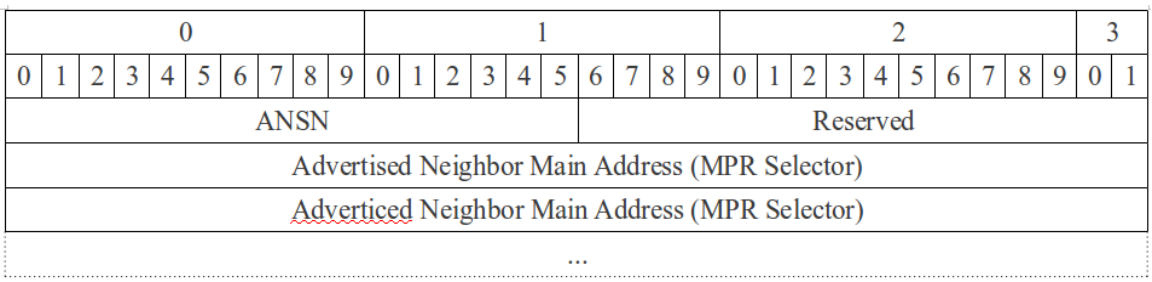

Figure 6: General Structure of TC packets (Taken from [3])

calculation algorithm is triggered in two cases; first one if there is change in bi-directional neighbor set, the other is if there is change in two-hop neighbor set with bi-directional links.

After calculating MPR set, the next function that the protocol requires to do is declaring the MPR information through the network. To achieve this purpose, OLSR algorithm uses "Topology Control" (TC) messages whose structure can be seen in Figure 6. "HELLO" messages are broadcasted to only to neighbor nodes but "TC" messages are flooded through the entire network. As mentioned above this flooding mechanism is different than default flooding and only MPR nodes are used the flood this information. Also one more optimization is made when flooding "TC" messages. "TC" messages aren't initialized by the nodes themselves but MPRs initialize the "TC" messages by announcing the nodes which have chosen them as MPRs. The nodes announced in the "TC" messages are called MPR selector set which is the set of the nodes selected the node as MPR. Another difference between "HELLO" messages and "TC" messages is that since "HELLO" messages are sent directly to neighbors and not retransmitted they don't require sequence number but using different routes may a cause a delay for a "TC" message which has been sent and it may be received after an up-to-date message to avoid this kind of misunderstandings, "TC" messages include sequence numbers.

When a node receives a "TC" message, the node first checks the sequence number if the messages is older than that of last processed "TC" message, it is silently discarded. For the other messages, the entries who have older MPR selector sequence number are omitted and for the rest either new topology entry is recorded in the topology table or the time for entry is refreshed if there already exists an entry whose destination address is same with address of an MPR Selector.

After these processes the routing table can be calculated by using gathered information. As being a link-state algorithm that uses hop count as routing metric, an algorithm to calculate (or re-calculate) the routing table is as follows; first remove all entries in the routing table, then insert one-hop neighbors with the hop count or distance parameter 1, then increase the distance parameters and insert two-hop neighbors to the table and by induction insert all of the nodes in the topology table to the routing table. If there are unused entries in topology table, they might be either used to define alternatives routes or just thrown away.

1.3.2 B.A.T.M.A.N.

The other routing algorithm we used in our experiments is called Better Approach To Mobile Ad hoc Networks (B.A.T.M.A.N.) which is a proactive (table-driven) link-state routing protocol, it is defined in [4]. In the conventional link-state algorithms, all of the links in the network are relayed to all of the nodes in the network. Then nodes calculates best paths to the destination nodes using this link-state information themselves. Due to nature of wireless networks, even when the nodes are static and there is no other network traffic, when a packet goes through the network, it creates interference on the path of the packet it relayed when sending the next packet. B.A.T.M.A.N. proposes a new routing algorithm which can cope with these problem.

In B.A.T.M.A.N. routing algorithm, it is assumed that the message first comes through the best path. Every node starts to send their routing protocol messages

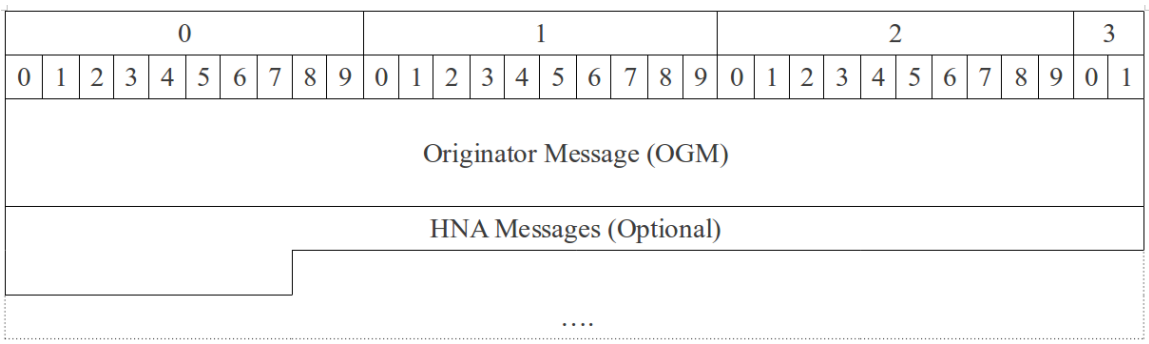

Figure 7: General B.A.T.M.A.N. Packet Format (Taken from [4])

which are called OGMs. When a node receives an OGM, it relays it to its neighbors according to some forwarding rules to prevent problems that may be caused by uni-directional and asymmetric links and the process goes so on until Time-to-Live (TTL) of the message is reached which means maximum allowed retransmission is made for the packet or the message is flooded all over the network. The node initializing the OGM is called originator. When a node receives an OGM, by using the best path assumption it records address of Originator as destination and address of the neighbor who relayed the message as the best gateway (or next-hop) to the Originator.

1.3.2.1 General B.A.T.M.A.N. Packet Format

The General B.A.T.M.A.N. Packet consists two main parts the first contains the OGM message and the next part contains the HNA messages if there are some networks that can be reached by the node. The structure of General B.A.T.M.A.N. Packet can be seen in Figure 7.

The more important subfield of General B.A.T.M.A.N. Packet is the Originator Message and the information included in the Originator Message can be seen in Figure 8.

In the network, there may occur momentary congestion and the OGM may not be received through the best path. To deal with this problem, nodes not only consider the last received OGM but also considers the OGMs received previously to decide

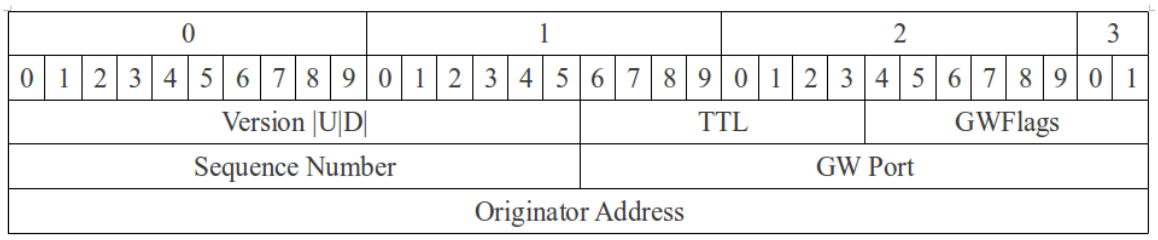

Figure 8: Structure of B.A.T.M.A.N. Protocol OGM (Taken from [4])

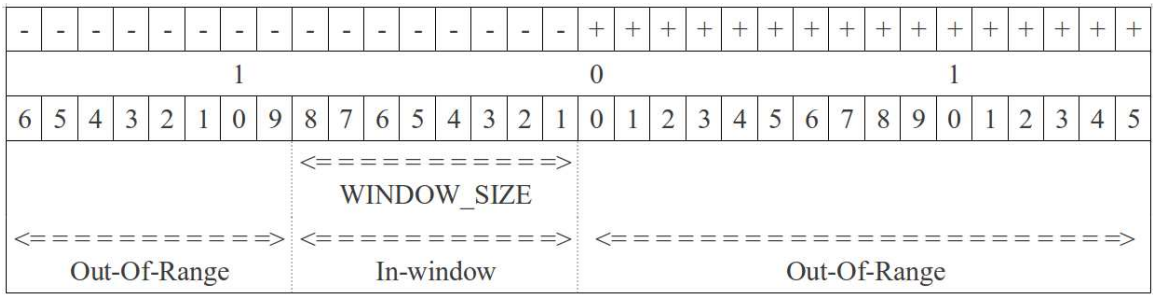

Figure 9: Sliding Window Example for B.A.T.M.A.N. Protocol (Taken from [4])

the routing path. Also there might happen a permanent change in the network and a previous next-hop may not be the best choice to an originator anymore. To deal with this problem, in B.A.T.M.A.N. protocol, sliding window technique is suggested that keeps records of the up-to-date OGMs.

1.3.2.2 Sliding Window Format

The Sliding Window is used to keep up-to-date control messages by sequence numbers and since the memory is limited, the sequence numbers are used from a limited window. The control messages received with a sequence number outside of this window are omitted since they are out of range. Visualization of sliding window can be seen in Figure 9

1.4 Thesis Contribution

In this work, we built a three dimensional WMN Test Bed in Ozyegin University Engineering Faculty building. Our devices are separated enough from each other to

make sure multihop communication. In the study, two well-known routing algorithms, OLSR and B.A.T.M.A.N. are compared according to their QoS performances for real time streaming video on a real life WMN test bed. We made our experiments for different video compression rates and by creating cross-traffic at various rates and types on the network, the effects of these on the Video QoS for different routing algorithms are observed. Also we tested how the routing algorithms are effected, when RTS/CTS handshake mechanism is switched to be always on. For QoS comparison, we streamed our test video several times and calculated the mean and standard deviations of PSNR values with respected to the compressed and uncompressed videos. Some frames were lost during streaming since wireless medium is a lossy medium because of that before making PSNR calculation, we needed to match frames in recieved videos to frames in the compressed video. We proposed an algorithm to match frames in received video and compressed video by considering lost frames.

1.5 Summary

The remainder of the thesis is organized as follows. In Chapter 2, we summarized similar works done in the literature. In Chapter 3, how the setup of our testbed is explained. In Chapter 4, we explained implementations of tested routing algorithms. In Chapter 5, concepts on streaming that are used in test-bed are explained. In Chapter 6, we explained how the test results are evaulated. In Chapter 7, we shared our results. In Chapter 8, the results are summarized and concluded.

CHAPTER II

PREVIOUS WORK

In this chapter, a review of works in the literature that were about the subjects OLSR routing algorithm, BATMAN routing algorithm, test beds for wireless mesh networks and video streaming in the wireless networks will be reviewed.

In [5], Benzaid, Minet and AlAgha offered a modified version of OLSR routing algorithm. They called the new algorithm Fast-OLSR, because it aimed to help specially to fast moving nodes. The proposed algorithm is backward compatible. The main idea of the modification is if a node is moving fast, its neighbors will change at a faster than that of a stationary node, so when a node discovers that its neighbors are changing quite fast, it changes its algorithm to the Fast-OLSR and starts to send Fast-Hello messages instead of Hello messages defined in OLSR routing protocol. Fast-Hello messages are sent oftener so the node can renew its neighbor list oftener. After sometime if the node discovers that its neighbors stop changing much then it switches back to the default mode. Fast-Hello messages are shorter than the Hello messages since they include less number of neighbor addresses. The aim of Fast-Hello message is to inform MPRs about the link will change fast and ensure the moving node that it can be reached by the chosen MPRs. They tested their algorithm in simulation environment and they choose a node and make it to move at different speeds from 20km/h to 150km/h. They concluded that the moving node can reach 150km/h with reasonable overhead and below 15% packet loss rate.

In [6], for more scalable and effective self-managing networks, Klein proposed

cross-layer methods and tools. The solutions were specially validated over a real-world IEEE 802.11 WMN test-bed. They offered a technique at MAC layer to combine several frames into a single burst for traffic aggregation and they prepared a Knowledge Plane which is capable to enable service ontology among the nodes in WMN and they developed a monitoring framework to verify the suitability between the Knowledge Plane and realistic networking scenario. As a result of their study, they observed that the proposed packet aggregation method increases scalability and the architecture they proposed can prioritize the traffic and isolate the perfomance in IEEE 802.11 WMN. Also they observed the prepared Knowledge Plane makes avaliable monitoring needs and adapting the behaviors of the nodes.

In [7], Bicket, Aguayo, Biswas and Morris deployed an 802.11b wireless mesh network, over four square kilometers of an urban area with 37 nodes. They classified the previous wireless networks into two groups at the date they deployed their network. The first has to be carefully constructed with multihop communication. This kind of network has to be well-planned and requires high level technical expertise. In the second one, APs serve to clients individually and they have loose connection between them, this type network has small coverage area and coverage of all area cannot be ensured. In the network they built, they tried to combined benefits of these networks. In their network, nodes have omnidirectional antennas with lower range, multihop routing is possible and routing might be altered slowly if the network topology changes slowly. For the routing, they tried to maximize the throughput and used ETT (expected transmission time), as the routing metric. After the experiments, they found the average throughput between nodes as 627 kbits/sec and they also saw that internet can be served to network by a few conveniencely placed Internet gateways.

In [8], Hempel, Sharif, Zhou and Mahasukhon built a testbed on a railwat with collaboraiton of Federal Railroad Administration. In their test bed, they want to

test various wireless protocols for mobile usage. In their testbed, they used 8 APs all of which have both omni-directional and patch antennas. To test their testbed, they used IEEE 802.11a standard for inter AP communication and IEEE802.11b/g to serve the clients. After their first test, they observed that their testbed is useful for testing mobile wireless communication.

In [9], Lan et.al deployed a WMN testbed at the intersections with traffic lights and shared their experiences. Their motivation was increasing number of adaptive traffic control systems in cities worldwide and higher cost of wired systems than cost of wireless systems. If the devices placed in the traffic lights are connected by wire, installation, maintenance costs increase due to cost of wires connecting the modems each other and operational cost is high because of the monthly fee paid to the telephone company but by a wireless system, the installation cost can be reduced to cost of only placing devices, maintenance cost can be reduced to repairing or renewing cost of devices and for operation fee, only small license fee may apply. After their experiments, the first result, they obtained is not only the latency but also variance of latency increase when hop number increases, but if only the distance increases the latency doesn't increase but variation. They observed that packet loss is distributed uniformly over time and increasing number of hops or distance makes packet lost severe. As results of their experiences, they suggest people to check carefully hardware and software they use since different kind of hardware and software may perform different performance than each other and even than their own specifications. Finally, they make suggestions on the topics maintenance, remote management and security.

Abolhasan, Hagelstein and Wang compared performances of BATMAN, BABEL and OLSR routing protocols according to multi-hopping performances and route recovery abilities of these routing protocols in [10]. They used 8 devices in their testbed and deployed the testbed in one floor of their research institute. They used default versions of BATMAN and BABEL protocols but they made slight changes in the

configuration of OLSR protocol to prevent constant route changes. They used three scenarios to compare the routing protocols, in the first scenario they saturated the network with file transfer and measured the bandwidth with respect to the hop count. In the second scenario, with a lighter traffic than it was in first scenario, they measured packet loss rate and round trip time of the network and in the last scenario, they compared the link recovery reflex of the network by breaking a link intentionally. They say OLSR is beaten in all scenarios by both of BATMAN and BABEL, BATMAN is the best at the subjects stability and packet delivery and BABEL outperforms both of the other protocols in multi-hop bandwidth usage and route repair time.

In [11], Rong, Qian, Lu, Hu and Kadoch studied multipath routing over IEEE 802.11 based WMNs for multiple description video transmission. To investigate technical challenges they designed a framework. They saw that for good performance paths are qualified if they have minimal joint paths but since video applications are vulnerable to delay, it is hard to find multiple qualified paths. They proposed a new version of Guaranteed-Rate packet scheduling algorithm to solve this problem. After their simulation, it is shown that the proposed method reduces the delay and achieves load balancing over the network.

In [12], Wang, Hagelstein and Abolhasan share their results that they obtained by testing open80211s, an implementation of draft standard IEEE 802.11s, OLSR and BATMAN routing protocols and compare the new results with what they have found in [10]. They made small changes in the location of their devices. They used their first two scenarios to test the wireless protocols. The first conclusion they drew is that HWMP performs worse than both of OLSR and BATMAN routing protocols in a multi-hop network and even BATMAN doesn't have a predetermined transmission path, it shows better performance than OLSR. During the test, they recognized that HWMP has the highest outage rate due to reactive routing and this is the partial

reason of why HWMP has the lowest transmission rate. Also in the next part of their paper, they say that since HWMP is a reactive routing protocol, it suffers in the subject round trip delay and OLSR has the worst packet delivery ratio. In the conclusion of the paper, they say that HWMP works in multi-hop environment, but since it is a reactive protocol, its path selection becomes unstable in a congested network and HWMP should be improved because of its inadequate performance on bandwidth and round trip delay.

In [13], Garroppo, Giordano and Tavanti summarized pros and cons of layer 2 routing with respect to layer 3 routing and then shared the results of the experimental comparison of 802.11s and BATMAN-Adv protocols. Since they think the essential aspects of a wireless mesh network routing algorithm are route stability and recovery times, in their study, Garroppo et.al focused on only these two parameters. To control the environmental parameters easily, they built a small network consisting of four devices. To conclude their work, they say that the IEEE 802.11s implementation cannot show enough stability in proactive path selection but BATMAN-Adv acts much stabler and this makes BATMAN-Adv a reliable routing algorithm on proactive path selection. On the other hand, they say that BATMAN-Adv suffers to much in case of recovering from an interrupted communication but IEEE 802.11s implementation recovers so quick that in some cases the service feels to be seamless. After these conclusions, to finish their paper by announcing both of the protocols as "losers" since they can't handle some typical realistic situations that they are supposed to handle.

In [14], Murray, Dixon and Koziniec compared the routing protocols OLSR, BABEL and BATMAN and they discussed about the largest determinant of routing performance, impact of OSI layer on routing performance and challenges in multi-hop ad-hoc network protocols. They prepared four different topology to test the routing protocols. In their first topology, all nodes were in one-hop range of the gateway. They setup this topology to have a baseline for their experiments. They distributed

the nodes randomly for the other topologies. They measured packet delivery ratios, bandwidths and overheads of the said routing protocols. They found similar and reliable packet delivery ratio results for all routing protocols. In the comparison of bandwidth, BABEL outperformed all other three protocols and the other three ranked differently in themselves in different topologies. They also emphasize the correlation between the overhead and the bandwidth of the routing protocol. In the final part of their paper, they discuss about the impact of OSI layer on routing performance and they conclude the discussion with their opinion OSI layer has negligible effect on routing performance, when compared to other parameters.

Seither, König and Hollick made experiments to compare BATMAN-Adv, AODV and a version of AODV modified by themselves. They shared their experiences and results in [15]. Their aim was to test BATMAN-Adv and they used AODV as a control group. AODV uses the hop count metric which is very unreliable in wireless communication so to make it a little bit reliable in the modified version AODV, they applied neighbor selection process explained in [16]. They used the different traffic schemes in their experiments, in the first one they send each node 50 ping messages in an interval of 1 second from a chosen node consecutively, in the second one, from the chosen node they send TCP traffic for 30 seconds to all nodes in the network one by one. They built three different experiment setups for their experiments, in the first setup they placed only different devices which can hear each other, the aim of this scenario was establishing a baseline for the experiment since it is obvious that there is no need for routing protocols in this scenario. For the second scenario, they sparsely distributed 14 nodes in two adjacent floors and for the last scenario, they distributed 17 nodes in one floor of the building densely. In the last scenario, they aimed to increase RSS and reduce interference. They make analysis of their results as, AODV and its modified version are not as successful as BATMAN-Adv in terms of reachability and packet loss. When RTT performances of the routing

algorithms are compared, in the second scenario AODV shows similar performance with BATMAN-Adv but BATMAN-Adv has lower variance and in the third scenario, they observed that AODV performs slightly better than BATMAN-Adv algorithm. As a comparison of the throughput, for the second scenario they say that since AODV cannot establish stable routes, it was not possible to measure its throughput for the comparison and for the third scenario, they say that in the cases where AODV can establish stable routes, AODV and BATMAN-Adv shows similar performance but in most cases AODV couldn't have established stable routes. They sum up their result with the conclusion; BATMAN-Adv can operate well in an environment with interference but AODV cannot even establish stable routes when multi-hop communication is required.

In [17], Kaula built a mesh network consisting of 6 MAPs and 2 clients. They compared HWMP and B.A.T.M.A.N.-Adv routing protocols. They measured throughput of the routing protocols for both of UDP and TCP traffics by changing the number of wireless hops from 1 to 5. They also made multi-hop video transfer and measured the quality of video according to MOS. As the result of their experiments, they found out that when the number of hops are 1 or 2 B.A.T.M.A.N.-adv outperforms HWMP but when the number of hops is increased more both of the protocols pass more or less same amount of data through the network. They also experienced similar quality for streaming video for both of the protocols.

In [18], Winkler and Dufaux prepared a simulation of video transmission over WCDMA channel for 3G and WLANs. For their, simulation environment they selected the source, codecs, bit rates and bit error patterns according to the applications they used, namely MPEG-4 and Motion JPEG2000. They made their tests four 12 different conditions and they evaluated quality of received videos accorging to SSCQE and also they obtained MOS from 21 non-expert viewers, they saw that the predicted MOS and subjective MOS are highly correlated so predicted MOS is reliable. They

also concluded that motion prediction is beneficial depending on scene and bitrate however for transmission error, it may cause some problems.

In [19], Cranley and Davis made an experiment to see how background traffic effects streaming video quality in 802.11b WLANs. In their test bed, there was a streaming video server that is connected to AP with wired connection. There are n traffic sources and n traffic sinks. All of these sources and sinks are connected to each other via the AP. They started their experiments first by streaming the video and increasing the background traffic over time. They streamed the video without complicated adaptation techniques. In their experiment, they also tested how the packet size effects the video quality. In the results of their tests, they observed that the AP reaches its maximum throughput at the rates 4.4Mbps, 4Mbps and 3Mbps for the background traffic packet sizes 1400B, 1024B and 512B respectively when the background traffic load reaches 1Mbps. After this point, the streaming video starts not to be able to find the desired bandwidth. They observed that when the traffic packet sizes are smaller, the buffers in the network get empty quicker so network can restore quicker. They didn't observe any significant effect of number of traffic sources on the streaming video and finally they found that the streaming video terminates when the background traffic load reaches 2.1Mbps, 2Mbps and 1.5Mbps for the packet sizes 1400B, 1024B and 512B respectively.

In [20], Cheng, Mohapatra, Lee and Banerjee evaluated the video transmission performance of WMNs for multihop situation using a testbed. In their research, instead of some network parameters such as throughput or jitter, they directly focused on perception of the users. They built their network with three clients, three APs, a switch and a streaming server. The server and one of the APs were connected to each other with cable via the switch. Each client was connected to a different AP and APs are connected to each other to achieve multihop transmission. They first tested the effect of inter-flow interference which is interference of two streaming video

to each other. For this test, first they streamed video directly to the furthest client and then they streamed video both to the furthest client and to the client connected to first AP. They observed that the inter-flow interference effect the transmission quality significantly and packet loss is highest between the AP connected to switch and relaying AP since this link is the closest link on the path to the link between first AP and first client. The next experiment they conducted was about intra-flow interference. For this test, they streamed video from server to the each client respectively. They observed that the packet loss increases as the hop count increases but they also observed that the perceived video qualities at first client and second client don't differ so much in other words they have similar PSNR values however they receive different amount of data. Then they tested the effect of best-effort traffic which doesn't have QoS requirements but effects the traffic in the network. They tested effects of both UDP and TCP traffic and they observed that UDP traffic effects the network more than the TCP traffic since TCP traffic has congestion control mechanisms. After that they tested the effect of the coding rate of the streaming video and observed that when the coding rate increases however more throughput pass through the network, the observed PSNR decreases. In the next part of their experiments, they observed that RTS/CTS mechanism and buffer size of decoder doesn't effect or slightly effects the observed PSNR but UDP packet size and MAC layer packet size effect the quality significantly. For both parameters as the packet size increases, the observed PSNR increases for the values they tested. In the last part of their experiment, they observed that by using multichannel and multiradio or QoS support with IEEE 802.11e, the perceived video quality can be increased.

In [21] and [22], new metrics for quality measurement of streaming videos are defined. In [21], Wang, Banerjee and Jamin proposed application level quality for streaming videos, first explained the capabilities of the application they used to make

tests and summarized the performace metrics for streaming video; then, they explained the parameters they changed in their experiments. They summarized their outcomes and proposed their metrics as a complementary component to MOS and PSNR since these new metrics are defined specially for streaming videos. In [22], Chan, Zeng, Mohapatra, Lee and Banerjee first proposed a method to match frames of transmitted and streamed videos to calculate PSNR and then they defined two metric for streaming video, in the first metric, they used PSNR as the only input and in the second metric, they used distorted frame rate, their distortion and frame loss rate as input. After that they compared their metric with MOS and they obtained Pearson correlation between their metric and MOS.

CHAPTER III

EXPERIMENTAL SETUP

In wireless communication environment as the distance between devices increases, the data transmission rate between devices decreases faster than linear propotion. Due to this phenomenon, in wireless communication at long distances instead of transmitting in farther hops and less hop counts, transmitting in closer hops and more hop counts may lead to transmission at higher data rate. In our experiments, we wanted to find out if routing algorithms transmit data packets in farther hops with lower rate or transmit data packets in closer and more hops and achieve higher transmission rate. For this purpose, first we had to discover the distance from devices where transmission rate of the devices starts to decline. To find an approximate range distance, we made measurements in an open area. The results of these measurements showed that in the radius of 15 meters, throughput of the devices doesn't change much but after that, every time the distance is doubled the signal strength measured by the devices decreases by 10dB and this results in decrease at the transmission rate. By considering these results, we decided to put neighbor devices about 10 meters apart from each other.

3.1 Devices

For our testbed, we have used 7 TP-Link TL-MR3020 ver.1.7 devices as MAPs. The device can be seen in Figure 10, top view of the inner part of the device can be seen in Figure 11 and bottom view of the inner part of the device can be seen in Figure 12.

Figure 10: TP-Link TL-MR3020

Table 2: Input and Output Sockets of TP-Link TL-MR3020

Socket	Function
Ethernet Port	WAN/LAN Connection
USB Port	3G Modem/Card connection
Mini USB Port	Power Connection

3.1.1 Hardware Specifications

In Table 2, input and output sockets of the device and their functions are listed. In Table 3, hardware features of TP-Link TL-MR3020 are listed. In Table 4, CPU properties of device are listed.

3.1.2 Software Specifications

OpenWrt is an open source, extensible Linux distribution for embedded devices. It allows users to built flexible firmwares for routers. By using OpenWrt, users can build a firmware including all of the desired features but nothing unnecessary. OpenWrt

Figure 11: Top View of Device Circuit

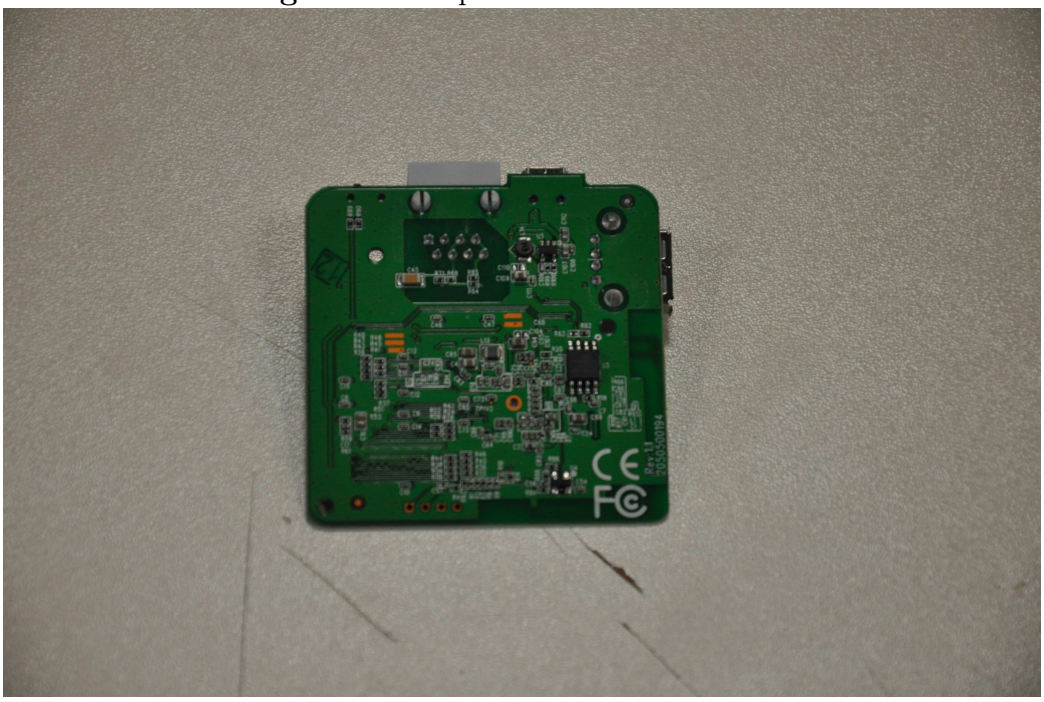

Figure 12: Bottom View of Device Circuit

Table 3: Hardware Features of TP-Link TL-MR3020

CPU	Atheros AR7240
Ram Size	32MiB
Flash Size	4MiB
SoC	Atheros AR9330 rev 1
Antenna Number	2

Table 4: Properties of CPU

Vendor	Atheros
Architecture	MIPS
Bootloader	uboot
Operation Frequency	400 MHz

saves users from single static firmwares and gives them opportunity of fully writable file system with package management. This allows users to use their devices in specific ways they wanted and the vendor didn't plan.

We used the last revision of OpenWrt, r34470, at the time we started to make experiments. This revision was including the B.A.T.M.A.N.-Adv module in itself but OLSRd module was not included. From OLSR's github repository we have added OLSRd to the OpenWrt's Buildroot system and we compiled the kernel that includes routing algorithms and required network tools such as iperf, iwconfig.

3.1.2.1 OpenWrt Buildroot System

OpenWrt offers some firmwares in its website, but they are generally in the simplest form to boot the device with network properties. If someone want to add extra features to the firmware, they need to use OpenWrt Buildroot System which helps to compile special firmwares.

To use OpenWrt Buildroot system, subversion (SVN), a version control system, and build-essential, a cross compiler, are needed. We need SVN for upgrading

or downgrading OpenWrt Buildroot system to desired revision and we need build-essential for compiling the kernel of the router in our computer and run it in the router which have different hardwares. In a Debian-based Linux distribution SVN and build-essential can be installed with the following commands;

```
sudo apt-get update
sudo apt-get install subversion build-essential
```

After installing SVN and build-essential, we need to go to the folder where we want to setup OpenWrt Buildroot system. By running the following command, we can checkout the trunk revision of OpenWrt Buildroot system;

```
svn co svn://svn.openwrt.org/openwrt/trunk
```

If one wants to check out another revision, they should run the following command in the terminal;

```
svn co svn://svn.openwrt.org/openwrt@#####
```

In this command, ##### symbolizes the revision number of the wanted revision.After successful installation of OpenWrt, if desired, by running the following commands in the folder where OpenWrt is installed, all feeds can be installed using feeds script;

```
./scripts/feeds update -a
./scripts/feeds install -a
```

The missing packages on the system can be checked by using one of the following commands;

```
make defconfig
make prereq
make menuconfig
```

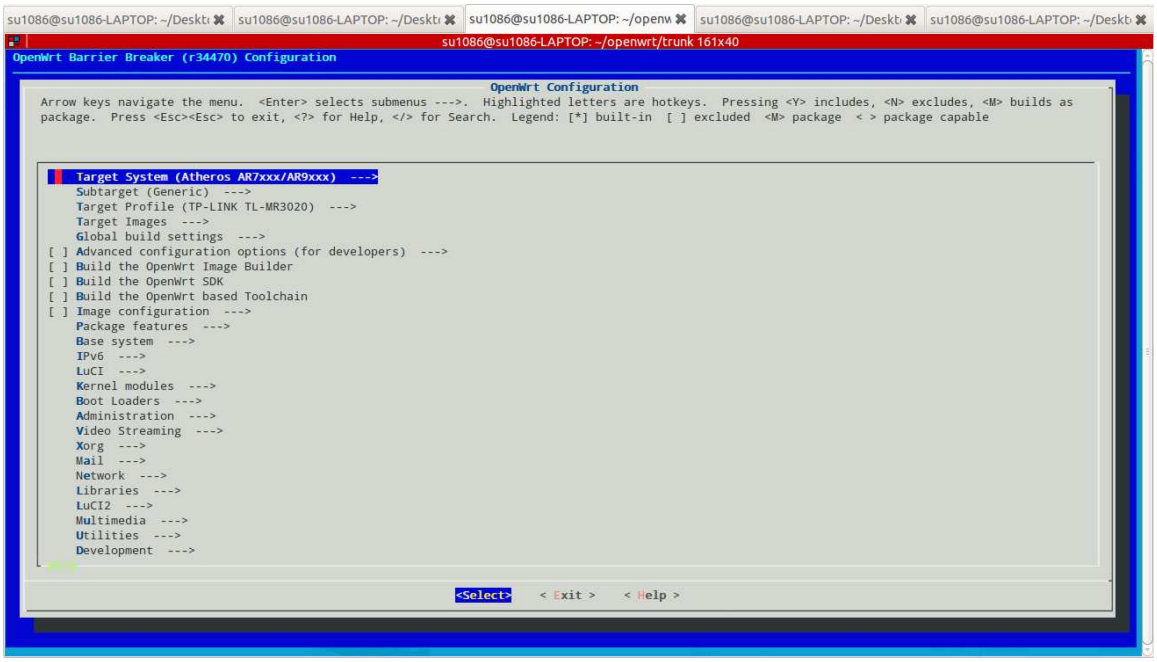

Figure 13: Screen view of configuration interface

When the command "make menuconfig" is called, OpenWrt Buildroot configuration interface, as can be seen in Figure 13, appears on the screen, "Target System" and "Target Profile" should be chosen according to the device in which the firmware will be installed (for our device they are respectively "Atheros AR7xxx/AR9xxx" and "TP-LINK TL-MR3020").

From the other menu items desired modules can be chosen to be added in the kernel. Main modules that we have used in the kernel and their path in configuration interface can be listed as follows;

B.A.T.M.A.N. Adv	Kernel modules → Network Support → kmod-batman-adv
IP-in-IP Encapsulation	Kernel modules → Network Support → kmod-ipip
Atheros 802.11n PCI wireless card support	Kernel Modules → Wireless Drivers → kmod-ath9k
Atheros 802.11n USB device support	Kernel Modules → Wireless Drivers → kmod-ath9k-htc
Library for manipulation Linux Wireless Extensions	Libraries → libiw
Generalized Wireless Information Library (iwinfo)	Libraries → libiwinfo
IEEE 802.1x Authenticator (full)	Network → hostapd
TCP and UDP bandwidth measurement tool	Network → iperf
OLSRd	Network → olsrd-stable-git
Network Monitoring and data acquisition tool	Network → tcpdump

After desired modules are chosen from configuration interface and desired configuration is saved, running the following command will start the kernel build process;

```
make
```

If it is the first time a kernel is compiled, this process will take longer time because OpenWrt Buildroot System connects to repositories, downloads the necessary packets and compiles them. After the compilation is over the required kernel images may be found in the relevant subfolder of bin folder of the OpenWrt folder (in our case the subfolder is AR71xx since the CPU of TL-MR3020 is AR71xx class CPU).

3.1.2.2 Adding OLSRd to Openwrt Buildroot System

First, the following line should be added to feeds.conf file in main folder of OpenWrt.

```
src-git olsrd http://olsr.org/git/olsrd.git;stable
```

If the line above is added before updating and installing feeds, the following lines aren't required; in other cases, to update and install the olsrd feed, the following commands should be run in the directory of OpenWrt. Then, the config menu option for OLSRd will be seen in the OpenWrt Buildroot Configuration System.

./scripts/feed update olsrd

./scripts/feed install olsrd-stable-git

3.1.3 Installing OpenWrt Firmware to the Device

When updating firmware, wireless connection shouldn't be preferred because if the connection is lost during update process, data may be lost and firmware cannot be installed completely which may result in crash of the device.

3.1.3.1 Installing for the First Time

By using web-interface, the firmware in the device could be changed easily if its original firmware has no inhibition. This might vary from device to devicei. For the ones we used the following steps should be followed.

From a web-browser go to the page 192.168.0.254

Enter admin for both of the fields "User Name" and "Password" (both of the fields are case-sensitive)

From the menu on the left, click "System Tools", the menu will expand down after "System Tools"

From the expanded menu, click "Firmware Upgrade" (when "Firmware Upgrade is clicked Figure 14 will appear on the screen

Click "Browse..."

Go to <Directory of OpenWrt>/bin/ar71xx/

Choose openwrt-ar71xx-generic-tl-mr3020-v1-squashfs-factory.bin

Click "Upgrade" on the web-interface

Click OK for upgrade confirmation

Wait for kernel image to be uploaded and the system is restarted

In the first upgrade the kernel image ending with "factory.bin" should be used. The restart page works offline and at the end of the process the page refreshes itself but since the default IP Address of OpenWrt is 192.168.1.1, the page cannot be

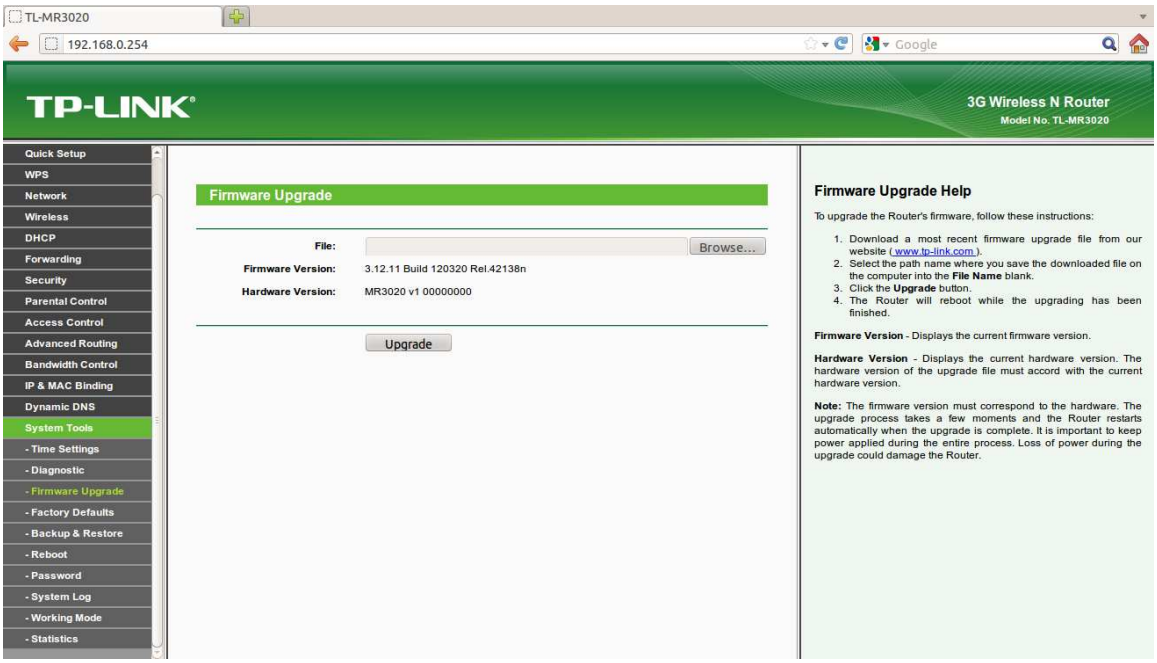

Figure 14: TP-Link TL-MR3020 Web-Interface Firmware Update Screen

loaded. To continue, you need to change IP Address of your computer's Ethernet interface which is connected to the device (here it will be assumed as "eth0"). You can do it by running the following command from terminal:

sudo ifconfig eth0 192.168.1.100

The first connection to device from terminal must be made via telnet. After first connection, by using "passwd" command the desired password for root user of the router should be set and next connections should be made by ssh. The required commands are listed below:

telnet 192.168.1.1 (for telnet connection)

passwd root (to change password of root user, it should be run in the command line of the router)

ssh root@192.168.1.1 (for ssh connection)

3.1.3.2 Upgrading Firmware If OpenWrt is Installed on the Router

The fastest and easiest way to upgrade the firmware is using OpenWrt's sysupgrade system. To upgrade the firmware the following steps should be followed (in this process it is assumed that root password of the router is set);

1. secure copy the new firmware to the /tmp folder of the router
2. connect to the router
3. upgrade the firmware

The following commands should be run in the directory where the new firmware is for this process;

1. scp <desired firmware>.bin root@192.168.1.1:/tmp
2. ssh root@192.168.1.1
3. sysupgrade /tmp/<desired firmware>.bin

3.1.4 Configuration Files

OpenWrt has its configuration interface called UCI. By using UCI, configurations of the systems can be made from terminal and this allows us to write scripts to change configuration from terminal easily. For our system, we changed the firewall configuration, added 2 new lines to the "/etc/profile" file and 6 files, one of which is a script to run at startup, one for setting first configuration and initializing first routing protocol and the other four are for changing the routing protocol running on the device. How these scripts operate will be explained in next chapter after properties of routing implementations are explained. All of these files can be found in Appendices A, B, C and D.

In the devices that we used, OpenWrt could be installed by updating the firmware from the web interface of the device. Another way to install OperWrt is to use serial console. How OpenWrt can be installed from serial console is also explained shortly at the end of this chapter. The pinout of the serial console is as following:

1. TX
2. RX
3. GND
4. VCC

On the board, pin 1 is marked.

3.1.5 Installing New Firmware Using Serial Console

Serial console allows us to connect and run bootloader and when bootloader has capability to write and erase flash drive directly. So when the firmware is crashed or doesn't allow us to install new firmware by using bootloader we can install new firmware via bootloader. OpenWrt uses the same bootloader as TP-Link so installing new firmware doesn't depend on the firmware running on the device.

Upgrading the firmware from bootlader requires higher attention because it is written directly to flash drive and if the addresses are entered wrong, it may harm some important software on the device.

When installing from serial console, we have chosen to use tftp. There are various ways to install from serial console. To install in our way, tftp requires some configuration settings and they can be found in Appendix E.

When the device is powered, first it reads the RAM memory then boots the firmware in flash drive. To connect to the bootloader, first one has to connect the device from serial port before powering the device, then "tpl" has to be typed before the firmware boots. (It will boot 1 second later then it prints "Autobooting in 1 seconds" on the screen) Figure 15 shows the terminal screen right after the bootloader is booted. The computer and router should be connected via Ethernet cable and IP-Address of the computer has to be 192.168.1.100. Then running the following commands from serial port will upgrade the firmware.

setenv ipaddr 192.168.1.111

Figure 15: Screen view of booting from serial console

setenv serverip 192.168.1.100

tftpboot 0x80000000 <name of new kernel>.bin

erase 0x9f020000 **+0x3c0000**

cp.b 0x80000000 0x9f020000 0x3c0000

bootm 9f020000

The value 0x3c0000 (3.75 MB) is the highest size that can be written on the flash by serial console so after transferring the kernel image, size of the image should be checked. Figure 16 shows the screen view just before booting the new firmware.

3.2 Test Bed

After we made seven devices ready with their new software, we placed them various locations at the Engineering Faculty Building of Ozyegin University as MAPs. The plan of where we installed the devices can be seen in Figure 17. Addition to our MAPs, we placed four computers as traffic sources and sinks. Two of those computers, A

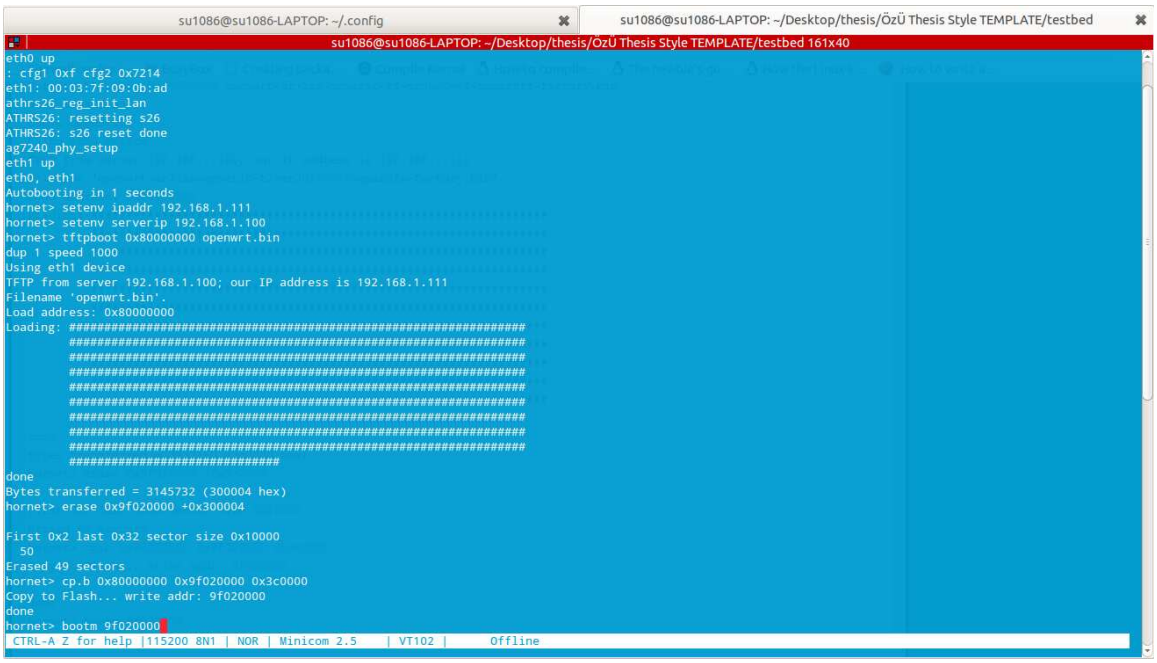

Figure 16: Last screen view before booting new kernel

and B in Figure 17, are connected to the APs with wireless connection and the other two are connected via cable. In our tests, we streamed the video from Computer A to Computer B and all the data was transferred from Computer D to Computer C.

Figure 17: Plan of our network

CHAPTER IV

ROUTING ON THE EXPERIMENTAL SETUP

4.1 B.A.T.M.A.N.-Adv

To test B.A.T.M.A.N. routing algorithm, we used B.A.T.M.A.N.-Adv which is a layer two implementation of B.A.T.M.A.N. routing algorithm. B.A.T.M.A.N.-Adv is built up by open-mesh group and an official part of Linux kernel since 2.6.38.

Most of the known routing algorithms operate on OSI layer 3 so devices announce their routing information by using UDP packets and by using the information they gather from other nodes, they manipulate the kernel routing table. On the other hand, B.A.T.M.A.N.-Adv operates completely on OSI layer 2. It not only floods the routing information on OSI layer 2 but also encapsulates and transfers the data traffic on OSI layer 2.

Operating on OSI layer 2 brings B.A.T.M.A.N.-Adv implementation lots of advantages such as:

Any network or higher layer protocol can be run on B.A.T.M.A.N.-Adv such as IPv4, IPv6 or a new implementation.

Nodes don't require to have an IP.

Non-mesh clients can be easily integrated without using HNA messages.

Non-mesh clients can roam without changing their IP-Addresses.

Data flow can be optimized.

Broadcast/multicast required protocols can be run over mesh and non-mesh clients.

B.A.T.M.A.N.-Adv is implemented as a kernel driver because Ethernet packets cannot be relayed and if the data packets are processed in user space, it will consume CPU cycles unnecessarily.

4.1.1 Configuration and Starting B.A.T.M.A.N.-Adv Routing Algorithm

There are some differences between configuration of B.A.T.M.A.N.-Adv versions before and after 2013.0.0. In our test bed we used B.A.T.M.A.N.-Adv version 2012.4.0 so here we will explain how to configure B.A.T.M.A.N.-Adv version 2012.4.0.

One way to configure B.A.T.M.A.N.-Adv is to use its "batctl" tool which is a configuration and debugging tool of B.A.T.M.A.N.-Adv. However, the configuration we will use is "UCI" which is abbreviation of **Unified Configuration Interface** and an utility for all kinds of configurations in OpenWrt.

To configure a device to operate as MAP running B.A.T.M.A.N.-Adv, we first changed its LAN address to an IPv4 address in the network 192.168.1.0/24 and defined the interfaces running on LAN as "eth0.1" and "bat0". Then we made the configurations of "bat0" interface. These configurations are interface name: "bat0", interface protocol: unspecified and MTU size: 1500 bytes. In the next step, we set the configuration options of "mesh0" interface. These configurations are the interface protocol: unspecified and MTU size: 1528 bytes. The difference between MTU sizes of the mesh interface and other interfaces is important and it has to be at least 28 bytes. As we mentioned above in B.A.T.M.A.N.-Adv, data packets are encapsulated and send via layer 2 routing, the extra 28 bytes are required for this encapsulation process. When OpenWrt kernel is installed on the device, there is a ready AP configuration so we do nothing to change the AP configuration but we set the "ssid" as "BATMAN-ADV" so we can see if there is a device not running B.A.T.M.A.N.-Adv but an other routing protocol on it. In the last part, we configured the mesh interface of the wireless connection. These configurations are to set its network as "mesh0", setting its network operation mode as "adhoc", defining its "ssid" name as any user defined name and finally setting the multicast rate as 11000. For these change to be effective first we write them in configuration files via the command "uci commit" and then we restart the network daemon of the device. The necessary commands for this

configuration can be found in Appendix B.

When the configuration is done and network deamon on the device is restarted B.A.T.M.A.N.-Adv starts automatically.

4.1.2 B.A.T.M.A.N.-Adv Operation

For B.A.T.M.A.N.-Adv, we define an extra interface called "bat0", however; "bat0" doesn't relay or transmit data packets. The mission of "bat0" interface is sending and relaying protocol messages of B.A.T.M.A.N. protocol and gathering the required data for kernel module to calculate the routing. After the routing is calculated in the kernel when there is a packet to be transferred, the packet is transmitted via the mesh interface (in our case it is mesh0) of the device on layer 2.

By using devices running B.A.T.M.A.N.-Adv, two end devices who don't run B.A.T.M.A.N.-Adv can be connected on layer 2. Because if the mesh interface of the MAP device is bridged with the same device's interface to which the non-mesh device is connected, the B.A.T.M.A.N.-Adv module floods its MAC address through the network and mesh devices can built a network on layer 2 between these two non-mesh devices.

4.2 OLSRd

To test OLSR routing algorithm, we used OLSRd which is a layer three implementation of OLSR routing algorithm. OLSRd gathers routing information from the nodes in the network, it calculates the routing table and manipulates kernel routing table which is the easiest way for Layer 3 routing.

Also OLSRd can serve devices which don't run OLSRd on themselves however there are some issues regarding these services. Since OLSRd is a Layer 3 implementation, if all the devices which aren't running OLSRd are connected only to one mesh point, communication to these devices can be made by NAT protocol. But when connection is needed between two or more devices without OLSRd that are not connected

to the same MAP, NAT protocol is not enough. To establish this connection, devices are supposed to be in different subnets and devices running OLSRd should announce the subnet that they are serving to via their HNA messages. Also this brings that all of the devices should run their own DHCP servers for the devices connected to them.

4.2.1 Configuration and Starting OLSRd Routing Algorithm

4.2.1.1 Initial Configuration

In this part, we will explain configuration of OLSRd which is needed only one time and in the next part we will explain the necessary part when switching between routing protocols.

For the initial configuration, we first need to remove all of the interfaces related with OLSRd then we can add only two interfaces which are "lan" and "wlan". lan interface will be used to bridge the interfaces of router such as Ethernet connection and wireless connection. For mesh connection we will use the wlan interface. As we explained above because of being a layer three routing algorithm OLSRd requires HNA service to serve the devices who don't run OLSRd. So in the next step of configuration we add HNA4 configuration for OLSRd and then configure the subnet that will be served by the device. To save these changes in the configuration files, "uci commit" command should be run.

4.2.1.2 General Configuration

When switching from a routing protocol to OLSR routing protocol, the following configurations are necessary and required. First, we may change the device's IP address that is used to communicated with th clients in its subnet (LAN where the device operates as an AP). Then we should define the interfaces that will be used in the subnet. Then we should add a new interface which will be used in the mesh network. The protocol, IP address and netmask of these interface should be set accordingly to the required properties of the mesh network. Then we should set the

SSID of the AP so we can differ devices running OLSRd from the ones not running OLSRd. Finally, we should define the properties of the network such as the interface that will be used, the wireless mode and its SSID. After making these changes, they should be recorded in the configuration files via the command "uci commit" and network daemon should be restarted for configurations to be enabled. Restarting the network with the new configuration is not enough for OLSRd to operate, its daemon should be started and enabled. The necessary commands for these configuration can be found in Appendix C

4.3 Stopping the Routing Algorithms

There are more than one routing algorithms running on the devices so when switching between routing algorithms, we should remove some configurations of the previous routing algorithm so the protocol doesn't interfere with the new routing protocol.

To remove B.A.T.M.A.N.-Adv, it is enough to delete the bat0 and mesh0 interfaces from configuration files and delete ssid and mcast_rate of the mesh interface.

To remove OLSRd first it has to be disabled and stopped, then WLAN interface and SSID of the mesh interface should be deleted.

The necessary commands to stop the routing algorithms can be found in Appendix D

CHAPTER V

VIDEO STREAMING OVER THE EXPERIMENTAL SETUP AND EVALUATION OF RESULTS

Multimedia streaming is a special kind of network application. Most of the other network applications doesn't require steady flow of network traffic but streaming applications serve the data received from the network directly to the user. In other applications, network quality can be measured by parameters like throughput, round-trip delay. On the other hand, in multimedia streaming if a packet comes late, its play-back time may expire and there may be distortion in the service. To measure the distortion, other performance parameters are defined. To measure video quality, we used PSNR the most common video quality measurement parameter. Also, there are Internet protocols to make multimedia streaming easier. In our testbed, we used RTP protocol for streaming video.

5.1 QoS Metric

There are various measurement techniques for video quality. They can be classified as subjective and objective techniques. Subjective techniques depend on people's opinions and they require time and volunteers to watch the video. Objective techniques are based on the measurable differences between the raw video and tested video. PSNR is the most common technique to measure quality of a video.

PSNR is an objective technique and it is the ratio between the maximum possible power of a signal and the power of corrupting noise. It is calculated in logarithmic scale and the unit is dB. Formula for calculating PSNR of a video is given in equation 1 and formula for calculating MSE is given in equation 2.

$$PSNR = 10 \cdot log_{10}(\frac{MAX^2}{MSE}) \tag{1}$$

$$MSE = \frac{1}{m \cdot n \cdot o} \cdot \sum_{i=1}^{m} \sum_{j=1}^{n} \sum_{k=1}^{o} Raw(i,j,k) - Test(i,j,k) \tag{2}$$

In equation 2, m is frame number of the video, n is frame width and o is frame height. $Raw(i,j,k)$ means the pixel in the i^th frame, j^th column and k^th raw of the original video and similarly $Test(i,j,k)$ means the corresponding pixel in the tested video.

Colored videos has three different components. They are Y (luminance), Cb and Cr (chroma) components. Luminance defines the brightness of picture, Cb and Cr components define the color components of a picture. For noises in visual media, human eye doesn't react linearly and it is more sensitive to Y-component. Because of this, when calculating PSNR, we considered only Y-components of the videos.

5.1.1 PSNR Calculation in Lossy Environment

When multimedia data is compressed, the number of frames doesn't change and frames can be matched one by one. But in streaming applications order of frames might differ if they use different paths on the network or due to congestion in the network some packets may come after the required time of arrival or never come; as a result, the player in the receiver side may have to skip some frames and play the subsequent frame. In such cases, the numbers of sent and received frames differ and for a fair comparison before calculating PSNR, the frames are required to be matched. We couldn't extract the header of the frames so we proposed our method to match the frames.

To match the frames in sent and received videos, first we find the uncorrupted frames by comparing the received frames with original frames. After finding uncorrupted frames, we compared each frame in the received sequence with frames that

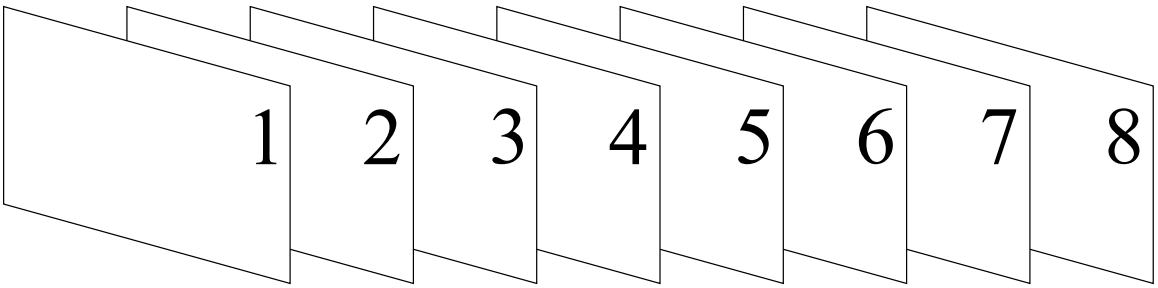

Figure 18: Transmitted sample sequence

would be a possible match in the transmitted sequence. After this comparison, we match the closest frames with each other, remove the corresponding received frame from the tranmitted sequence and narrowed the comparison set. For example, let's assume that frames 1-8 in Figure 18 are transmitted as a part of streaming video and frames a-e in Figure 19 are a part of received video and frame 1 is same with frame a, frame 8 is same with frame e. To match other frames, the following steps are done:

1. Calculate PSNR between frame b and frames 2-5 one by one.
2. Calculate PSNR between frame c and frames 3-6 one by one.
3. Calculate PSNR between frame d and frames 4-7 one by one.
4. Compare PSNRs
5. Match the frames with highest PSNR.

Let's assume that highest PSNR was between frame c and frame 4. Then;

7. Compare PSNR between frame b and frames 2 and 3.
8. Match the frame b with the frame of higher result.
9. Compare PSNR between frame d and frames 5, 6 and 7.
10. Match the frame d with the frame of highest result.

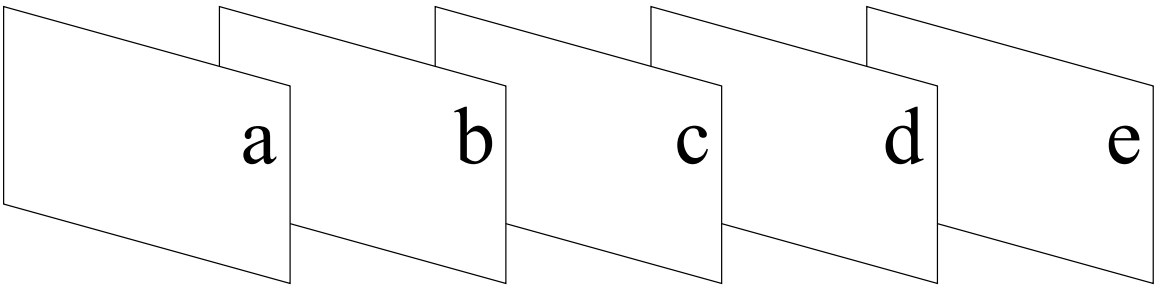

Figure 19: Received sample sequence

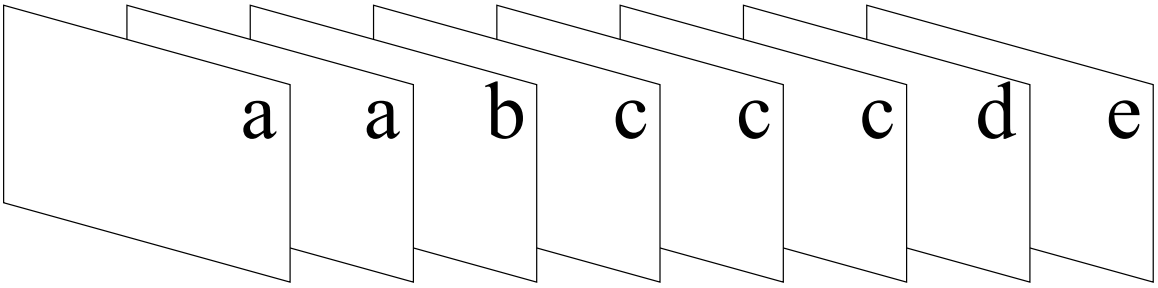

Figure 20: Interpolated sample sequence

After matching the corrupted frames with their correspondings resulted in the procedure above, next step is repeating the received frames in place of lost frames. In our example let's assume that frame b matched with frame 3 and frame d matched with frame 7. Then after interpolating frame a is repeated in the place of frame 2 and frame c is repeated in the places of frames 5 and 6, the new sequence will be like in Figure 20. We repeat the frames in the explained way because if a frame isn't received in the required time, player shows the previous frame until new one comes.

After we matched one received frame for each transmitted frame, we calculate the PSNR between two sequences.

5.2 PSNR calculation

At the end of the experiments, we collected 2160 different videos. We decomposed each video into its frames via MPlayer, a media playing program. For making PSNR comparison, we used "pnmpsnr" command in the "netpbm" packet which is a toolkit for manipulation of graphic images. "pnmpsnr" command gives the PSNR between

two given images for each of Y, Cb and Cr components. The human eye is much more sensitive to Y component than Cb and Cr components, so when evaluating the quality of the video frames, we took into account only PSNR of the Y component.

"pnmpsnr" command uses the following steps to calculate PSNR of an image:

1. Calculates the error between two frames pixel by pixel.
2. Calculates the mean square of the errors.
3. Chooses the maximum value as the peak signal value and divides it by the mean square error (calculated in the previous steps)
4. Converts the calculates PSNR value to dB by using the formula

$$PSNR_{dB} = 10 * log_{10}\frac{PEAK}{MSE} \tag{3}$$

After getting the calculated PSNR value of each image, we calculated PSNR of a video by the following steps:

1. By using inverse formula of 3, we get the MSE value for each frame back.
2. We calculated the MSE of all of the frames.
3. By using formula 3, we calculated PSNR value of the video.

5.3 Transport Protocol

RTP is an application layer Internet protocol built on UDP transport protocol, the concepts of RTP are defined in [23]. RTP is defined for transportation of real-time data such as audio, video or simulation data over multicast and unicast network services. Instead of resource reservation or guaranteeing QoS for real-time services, RTP provides end-to-end services with its control protocol, RTCP. RTP mainly identifies payload type, numbers the sequence, timestamps and monitors the delivery.

RTP doesn't assume that lower layers and network deliver the packets in sequence reliable but it assumes that they deliver the packets on time and ensure QoS requirements. By using sequence numbers in RTP, receiver can put the coming packets in sequence without decoding them or can locate the appropriate place of the packet.

CHAPTER VI

RESULTS

After collecting 30 samples for each combination of three video encoding bit rates and six cross-traffic types, we calculated PSNR of each video with respect to both of corresponding transmitted and raw videos, and then calculated the mean and standard deviation for each combination after eliminating 5 best and 5 worst cases in the combination. We plotted the results that we got from 20 videos for each combination.

We divided our graphs mainly into three categories, the first category is about how the PSNR is effected as the video coding rate is increased; in this category videos streamed when there is no cross traffic in the network were taken into account. Results of this category can be found in Figure 21, for the case where RTS/CTS handshake mechanism is set to device's default setting, in Figure 22, for the case where RTS/CTS handshake mechanism is set to be always on. The second category is about how the PSNR is effected as the video coding rate is increased when there is TCP traffic in the network. Results of this category can be found in Figure 23, for the case where RTS/CTS handshake mechanism is set to device's default setting and in Figure 24, for the case where RTS/CTS handshake mechanism is set to be always on. The last main category is about how the PSNR is effected as video encoding bit rate is increased when there is UDP traffic at constant rate in the network. We tested the routing algorithms under the effect of four different UDP traffic rates. The first UDP traffic was 300kb/s, the second UDP traffic was 600kb/s, the third UDP traffic was 1200kb/s and last UDP traffic rate was 1800kb/s. Results for the case where the UDP traffic rate is 300kb/s can be found in Figure 25 and 26 for the cases

where RTS/CTS handshake mechanism is set to device's default and to be always on respectively. Results for the case where the UDP traffic is 600kb/s can be found in Figure 27 and 28, where RTS/CTS handshake mechanism is set to device's default setting and to be always on respectively. Results of the case where cross-traffic rate 1200kb/s can be found in 29 and 30, where RTS/CTS handshake mechanism is set to device's default setting and to be always on respectively. For the last cross-traffic rate case result can be seen in Figure 31 and 30, for the cases where RTS/CTS handshake mechanism is set to device's default setting and to be always on respectively.

When we look at the Figure 21, we see that increasing video encoding bit rate doesn't effect the PSNR between encoded and received video and B.A.T.M.A.N. shows better performance than OLSR but both of the algorithms have high deviation which means that when the RTS/CTS handshake mechanism is set to the default setting of the device, quality of the received videos differ very much. In Figure 22, we see that switching RTS/CTS handshake mechanism to be always on helps OLSR to perform better but reduces performance of B.A.T.M.A.N. protocol a little bit. From that we can say that B.A.T.M.A.N. routing protocol is robust to hidden node problem since RTS/CTS handshake protocol is mainly to eliminate hidden node problem. On the other hand, we see that when RTS/CTS handshake mechanism is always on deviation at performances of both of the routing protocols decreases sensibly, which shows that however B.A.T.M.A.N. routing algorithm is not effected from hidden node problem on the average, due to hidden node problem there might be quality oscillations.

When Figure 23 is analyzed, it is seen that increasing the video encoding bit rate doesn't effect B.A.T.M.A.N. protocol so much but worsens performance of OLSR routing protocol if there is TCP transfer in the network. Also when Figure 24 is analyzed, we see that switching RTS/CTS handshake mechanism to be always on helps OLSR protocol but doesn't effect B.A.T.M.A.N. protocol so much, contrarily it decreases performance of the protocol. When there was no cross traffic switching

RTS/CTS handshake mechanism to be always on was helping to reduce the deviation at the performances of the routing algorithms but in this case, we see that the handshake mechanism doesn't help so much about reducing the deviation at the performances of the routing protocols. This might be due to rate control mechanism of TCP transfer protocol.

When we look at the Figures 25 and 26, we see that routing protocols aren't effected from the cross traffic when the cross traffic bit rate is 300kb/s and they show similar results as in the case where there was no cross traffic. Also when we look at the Figures 27 and 28, we see that cross traffic at 600kb/s doesn't effect performances of routing algorithms effectively. But when we look at the Figures 29 and 30 where the cross traffic is increased to 1200kb/s, we see that network starts to saturate and as the video coding rate increases the loss in the network also increases and PSNR values for the both of the protocols starts to decrease, also we can observe that at this rate B.A.T.M.A.N. routing protocol also starts to be effected from hidden node problem. When Figures 31 and 32 are looked over, we see that even for the case where RTS/CTS handshake mechanism is set to be always on, in both of the routing algorithms, PSNRs decrease as the video encoding bit rate increases, also we see that however B.A.T.M.A.N. was suffering from RTS/CTS mechanism when there is cross traffic at lower rates in the network in this cases we see that B.A.T.M.A.N. routing algorithm also benefits from RTS/CTS handshake mechanism.

When the Figures 33 and 34 are looked over, it can be seen that, however the increment at the video encoding bit rate didn't effect the achieved PSNR between transmitted and received videos when there is no cross traffic in the network, PSNR between raw and receieved videos increases as the video compressed at a better quality. This can be easily expected since the videos lose same amount of data during transmission and compressing at a lower bit rate will result in greater loss in the video.

When there is TCP traffic in the network, Figures 35 and 36, we see that B.A.T.M.A.N. routing algorithm can achieve similar PSNR if received videos are compared with raw video but in OLSR routing algorithm as the video encoding bit rate increases the loss in the network increases much more and however being compressed at a better quality, as the video encoding bit rate increases, the PSNR between raw and received videos tends to decrease.

As told previously, constant rate cross traffic at the rates 300kb/s and 600kb/s doesn't effect the performance of the routing algorithms. So in Figures 37, 38, 39 and 40, we observe similar things as we observed in Figures 33 and 34. Figures 41 and 42 tells us an interesting story, in Figure 41 we see that both of the routing algorithms are effected from cross traffic and as the video encoding rate increases in B.A.T.M.A.N. routing algorithm, PSNR between raw video and recieved video doesn't change but in OLSR routing algorithm, the loss in the network increases so much that even the video encoded at a higher quality is received by the receiver at lower quality. But in Figure 42, we see that in OLSR routing algorithm better PSNR values can be achieved if the RTS/CTS handshake mechanism is set to be always on. Finally, when we look at the Figure 43, we see that when there is high cross traffic load in the network, both of routing algorithms are under effect of the load and PSNR between raw video and received video decreases as the video is encoded with better quality. Also when Figure 44 is looked over, one more interesting result is seen that is however B.A.T.M.A.N. Routing Algorithm was tending to suffer from RTS/CTS mechanism in this loaded network case, we see that it also benefits from RTS/CTS handshake mechanism and same PSNR can be achieved when the video is compressed at higher rate.

After calculating means and standard deviations of the resulting PSNRs, by using unpaired t-test, we calculated the probability of B.A.T.M.A.N. being better than OLSR for all of the cases. Basic information for unpaired t-test can be found in F.

These probabilities can be found in Table 5, for the case where the RTS/CTS handshake mechanism is set to device's default setting and in Table 6, or the case where the RTS/CTS handshake mechanism is set to be always on. Also we calculated probabilities of PSNR when the RTS/CTS handshake mechanism is always on being higher than the PSNR when the RTS/CTS handshake mechanism is set to device's default setting. When we check these results, if the RTS/CTS handshake mechanism is set to be device's default setting B.A.T.M.A.N. routing protocol has a higher probability to be better than OLSR routing protocol but for the specific case of video encoding bit rate 500kb/s and UDP cross traffic at the rate of 300kb/s we see that they almost perform equally. But when we set the RTS/CTS mechanism to be always on, it is seen that B.A.T.M.A.N. routing protocol loses its advantage against the OLSR routing protocol and they show almost equal performances and for the specific situation where video encoding bit rate is 750kb/s and TCP cross traffic, OLSR routing protocol is expected to perform slightly better than the B.A.T.M.A.N. routing protocol.

By using same method, we also calculated the expectation of the routing protocols performing better when RTS/CTS handshake mechanism is set to be always on compared with RTS/CTS handshake mechanism is set to device's default setting. The results of this comparison can be found in Table 7. When we analyzed these expectations, we see that for B.A.T.M.A.N. protocol, setting it to be always on doesn't help so much even it makes performance of B.A.T.M.A.N. protocol worse for some cases; on the other hand, setting RTS/CTS handshake mechanism to be always on helps to increase the performance of OLSR routing protocol.

Table 5: Probability of B.A.T.M.A.N. Routing Protocol Performing Better than OLSR Routing Protocol for RTS/CTS Handshake Mechanism is set to Device's Default

		Cross-Traffic					
		None	300kb/s	600kb/s	1200kb/s	1800kb/s	TCP
Video Coding Rate	500kb/s	0.92	0.57	0.87	0.90	0.91	0.93
	750kb/s	0.91	0.88	0.89	0.93	0.94	0.94
	1000kb/s	0.93	0.90	0.93	0.90	0.92	0.96

Table 6: Probability of B.A.T.M.A.N. Routing Protocol Performing Better than OLSR Routing Protocol for RTS/CTS Handshake Mechanism is set to be Always On

		Cross-Traffic					
		None	300kb/s	600kb/s	1200kb/s	1800kb/s	TCP
Video Coding Rate	500kb/s	0.55	0.97	0.99	0.92	0.84	0.54
	750kb/s	0.69	0.71	0.85	0.92	0.71	0.22
	1000kb/s	0.94	0.93	0.69	1.00	0.75	0.98

Table 7: Probability of Routing Protocols Performing Better When RTS is always on than RTS is Default

			Cross-Traffic					
			None	300kb/s	600kb/s	1200kb/s	1800kb/s	TCP
Video Coding Rate	B.A.T.M.A.N.	500kb/s	0.16	0.78	0.86	0.68	0.93	0.31
		750kb/s	0.06	0.50	0.43	0.56	0.73	0.36
		1000kb/s	0.06	0.79	0.27	0.92	0.92	0.06
	OLSR	500kb/s	1.00	0.37	0.91	0.95	1.00	0.99
		750kb/s	0.98	0.96	0.93	1.00	1.00	1.00
		1000kb/s	0.99	0.97	1.00	0.96	1.00	0.99

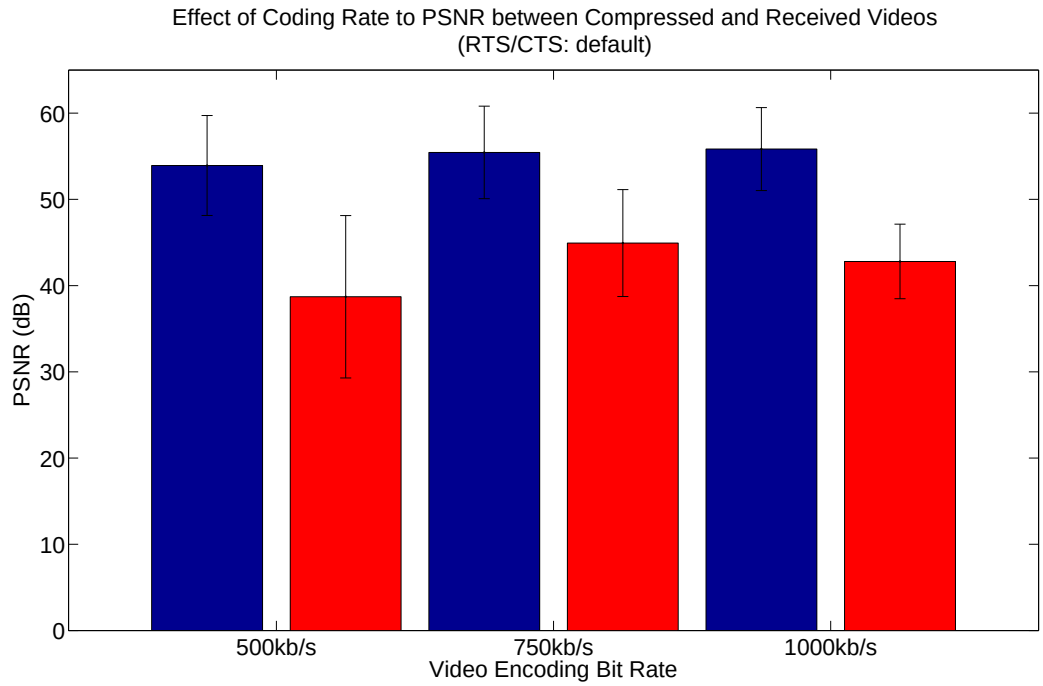

Figure 21: Effect of Video Encoding Rate to PSNR between Transmitted and Received Videos (RTS/CTS:Default)

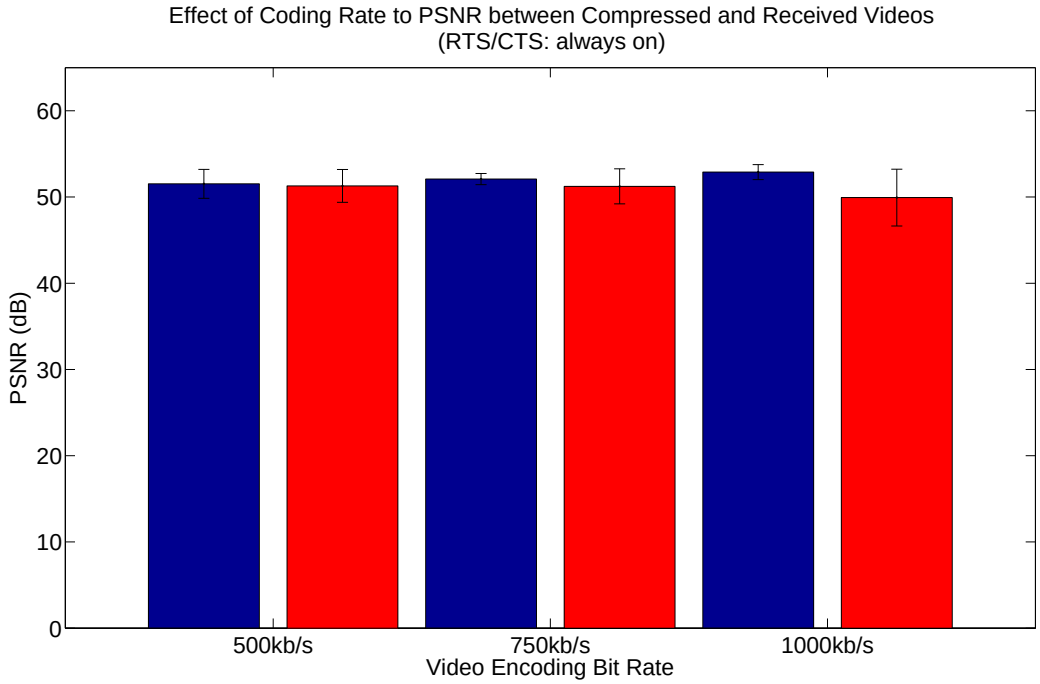

Figure 22: Effect of Video Encoding Rate to PSNR between Transmitted and Received Videos (RTS/CTS:Always On)

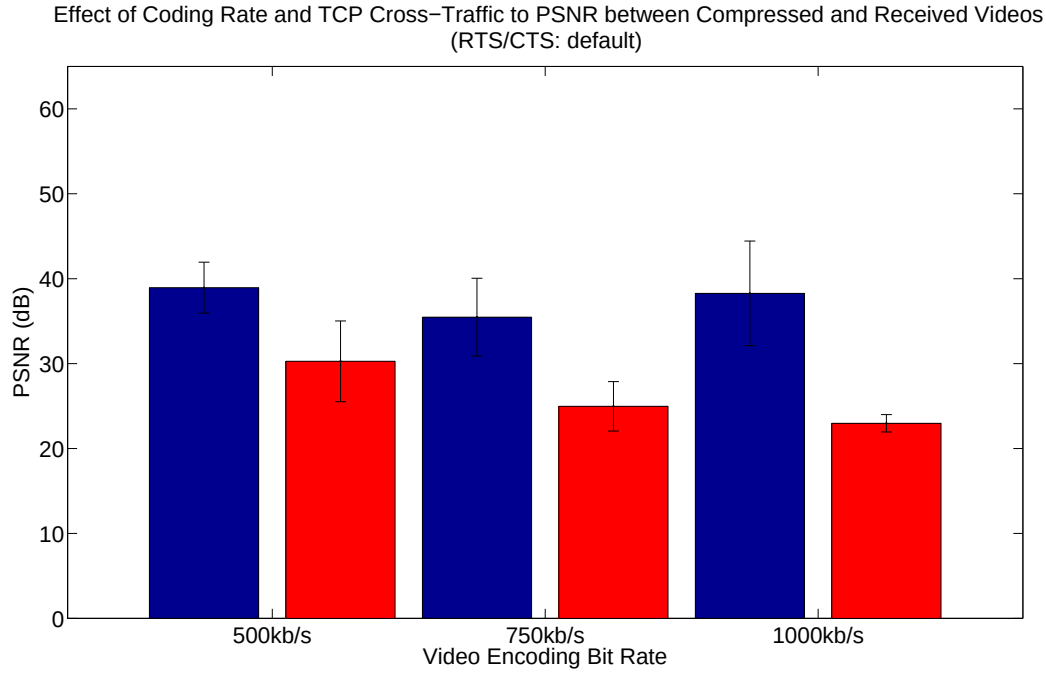

Figure 23: Effect of Video Encoding Rate to PSNR between Transmitted and Received Videos under TCP cross traffic (RTS/CTS:Default)

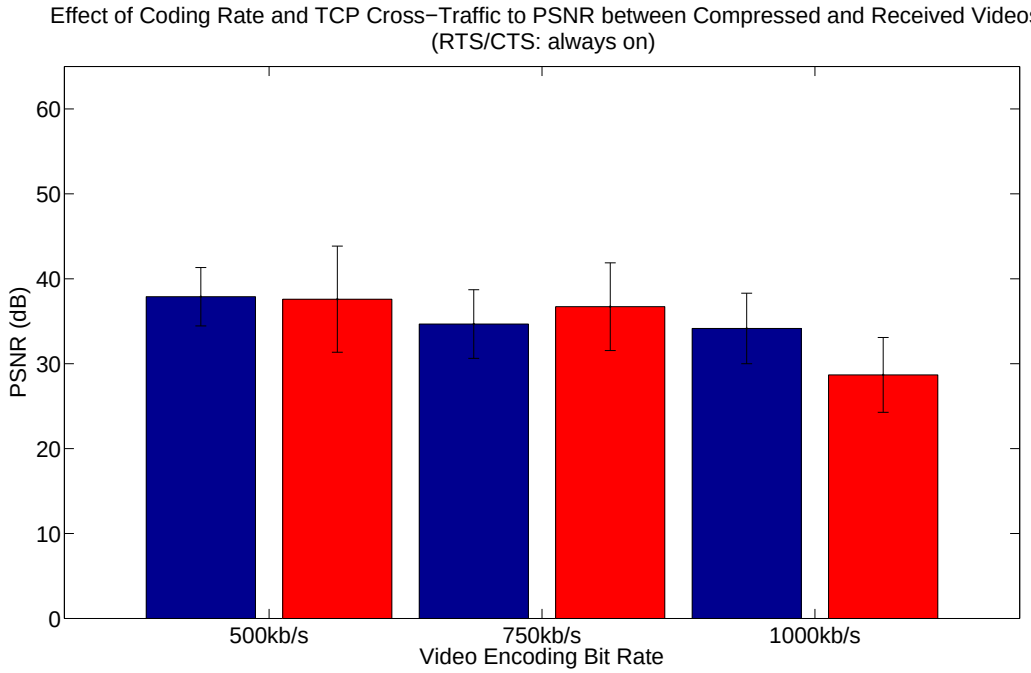

Figure 24: Effect of Video Encoding Rate to PSNR between Transmitted and Received Videos under TCP cross traffic (RTS/CTS:Always On)

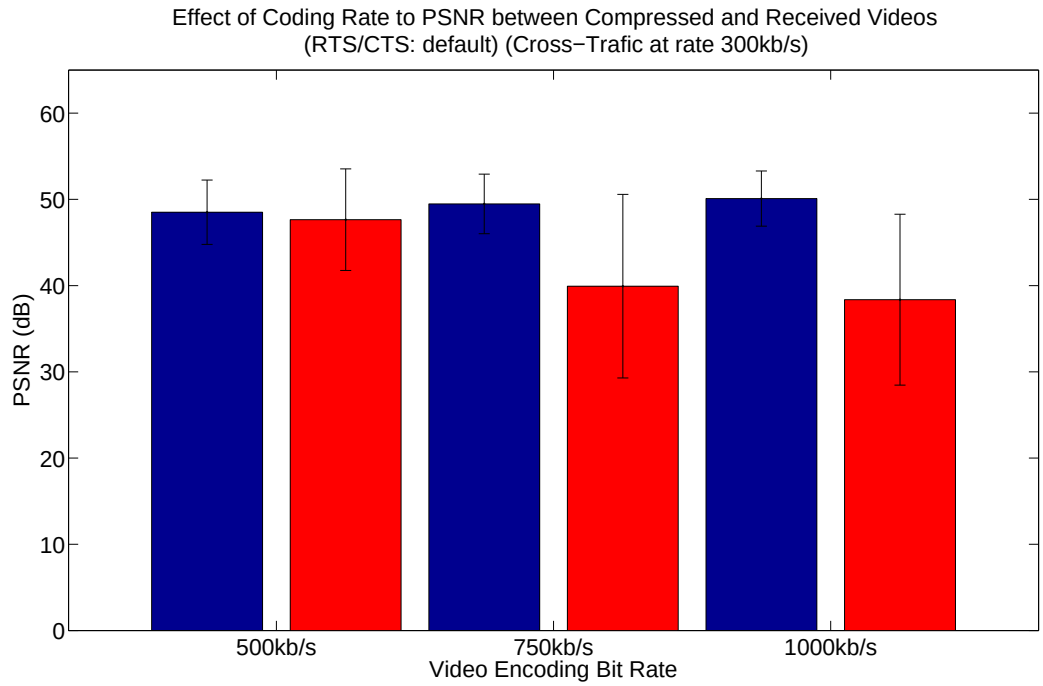

Figure 25: Effect of Video Encoding Rate to PSNR between Transmitted and Received Videos under 300kb/s UDP cross traffic (RTS/CTS:Default)

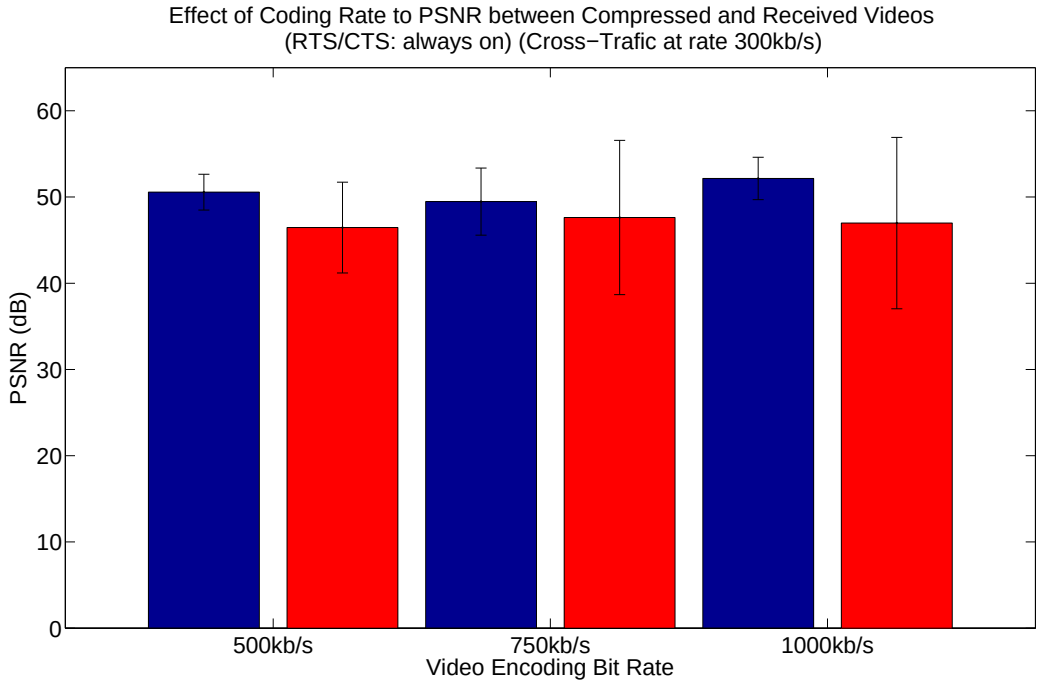

Figure 26: Effect of Video Encoding Rate to PSNR between Transmitted and Received Videos under 300kb/s UDP cross traffic (RTS/CTS:Always On)

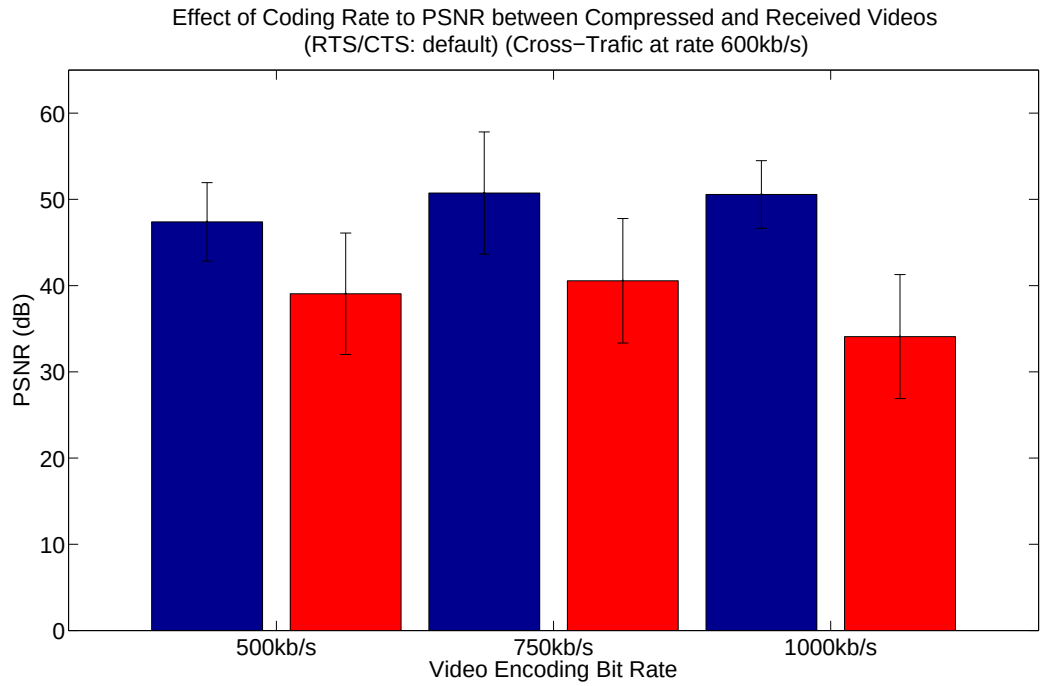

Figure 27: Effect of Video Encoding Rate to PSNR between Transmitted and Received Videos under 600kb/s UDP cross traffic (RTS/CTS:Default)

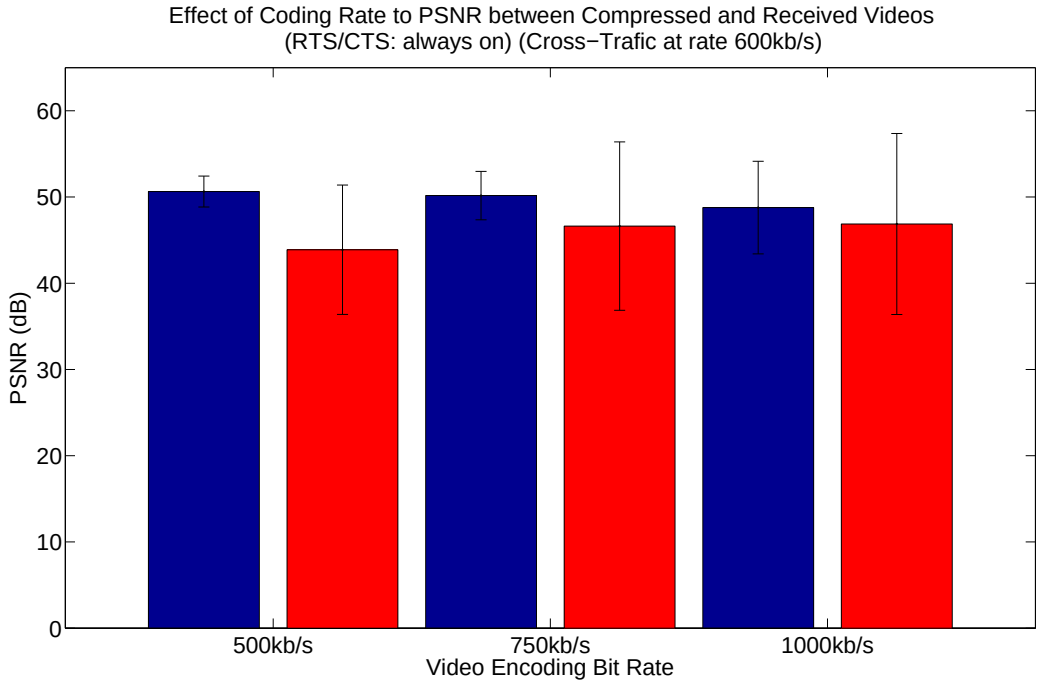

Figure 28: Effect of Video Encoding Rate to PSNR between Transmitted and Received Videos under 600kb/s UDP cross traffic (RTS/CTS:Always On)

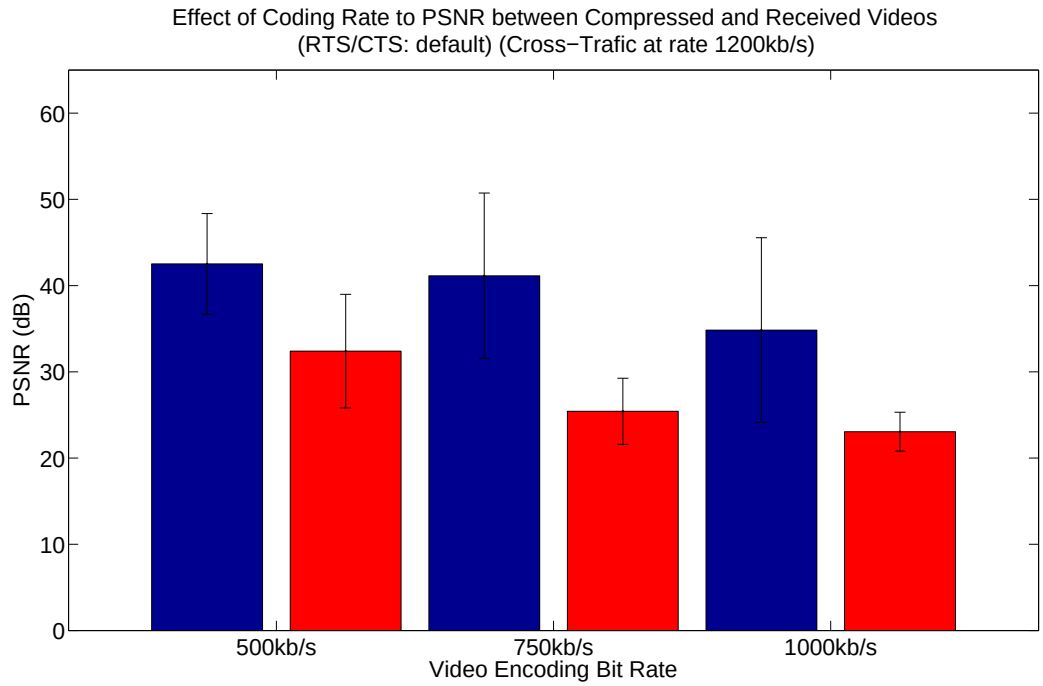

Figure 29: Effect of Video Encoding Rate to PSNR between Transmitted and Received Videos under 1200kb/s UDP cross traffic (RTS/CTS:Default)

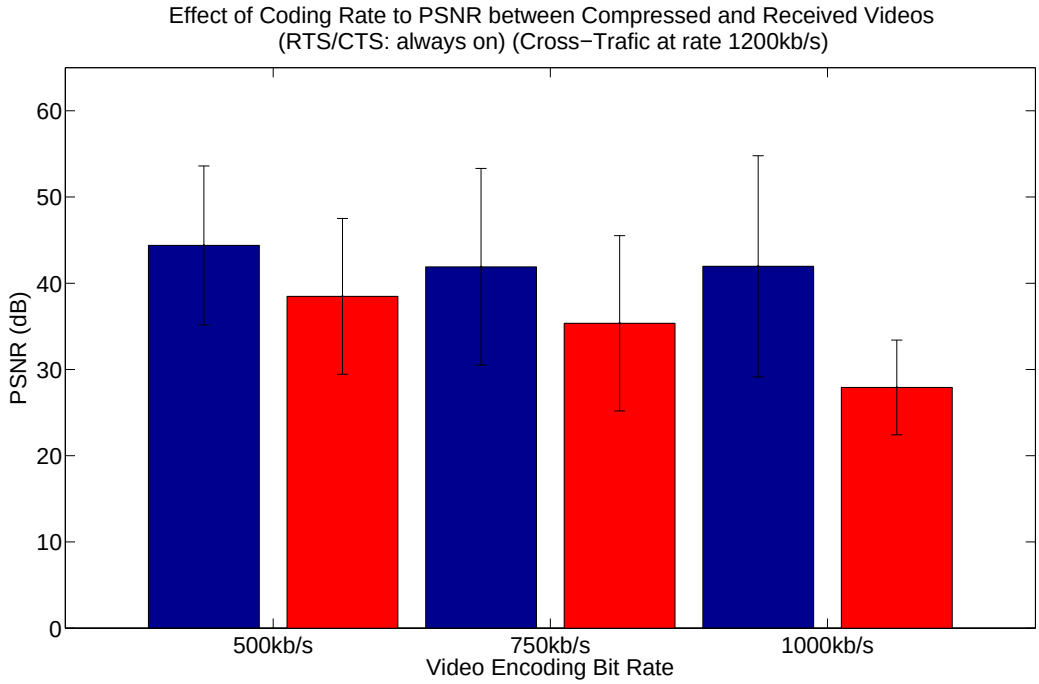

Figure 30: Effect of Video Encoding Rate to PSNR between Transmitted and Received Videos under 1200kb/s UDP cross traffic (RTS/CTS:Always On)

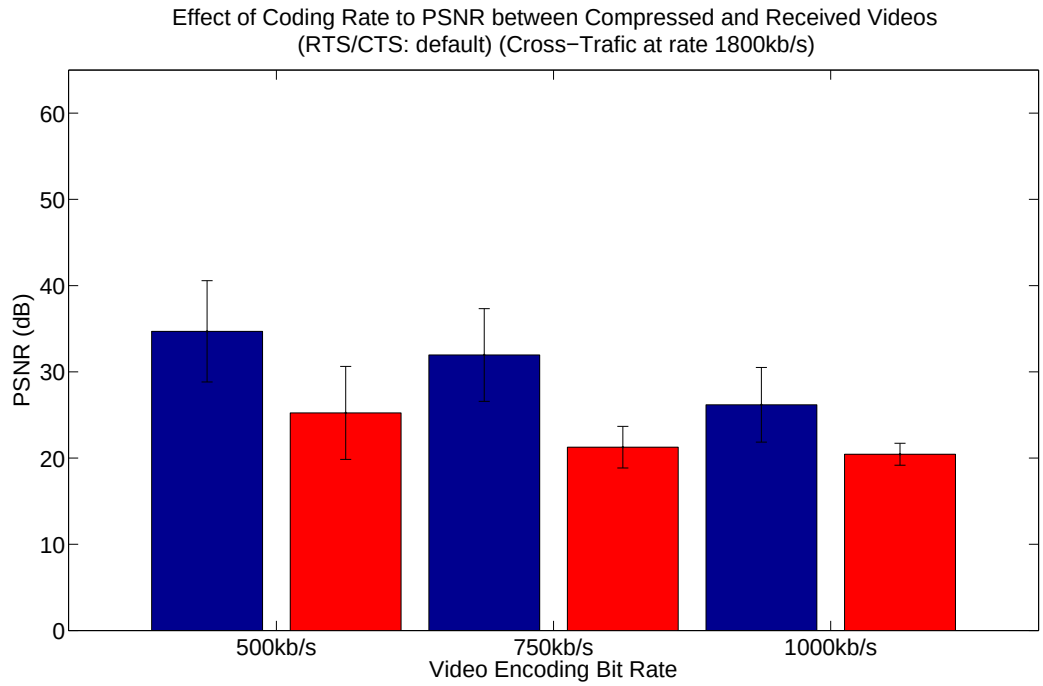

Figure 31: Effect of Video Encoding Rate to PSNR between Transmitted and Received Videos under 1800kb/s UDP cross traffic (RTS/CTS:Default)

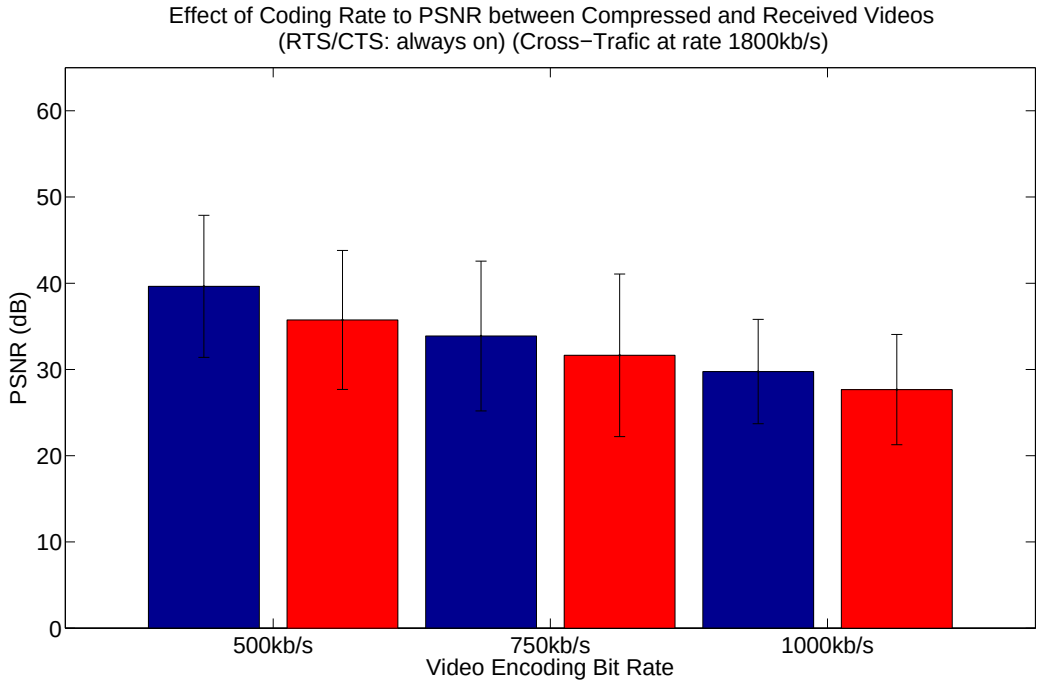

Figure 32: Effect of Video Encoding Rate to PSNR between Transmitted and Received Videos under 1800kb/s UDP cross traffic (RTS/CTS:Always On)

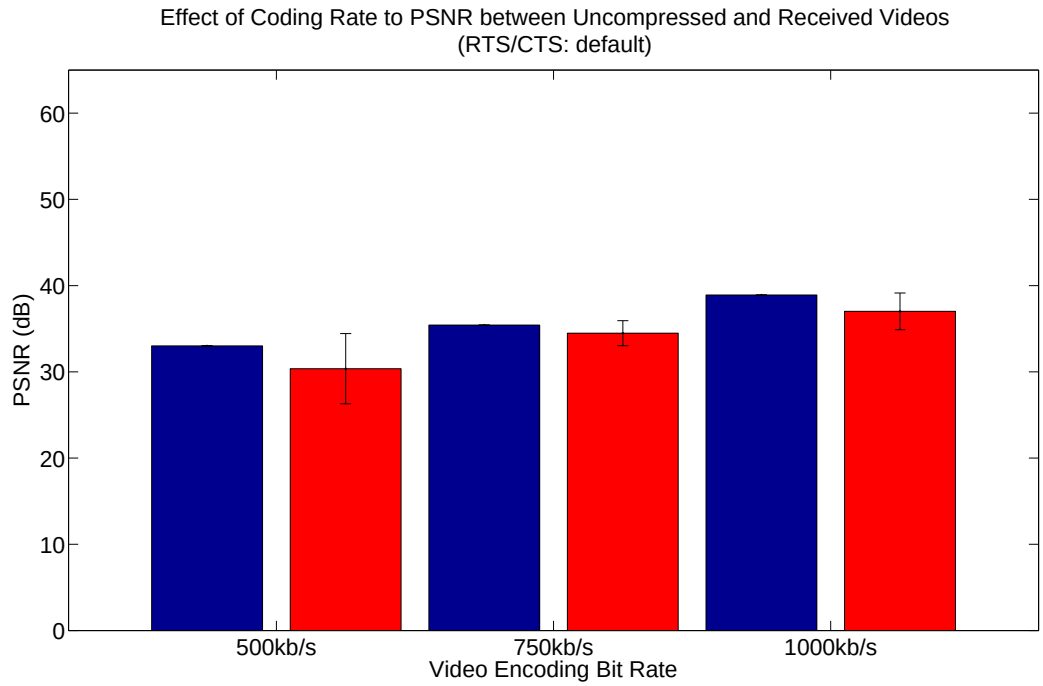

Figure 33: Effect of Video Encoding Rate to PSNR between Raw and Received Video (RTS/CTS:Default)

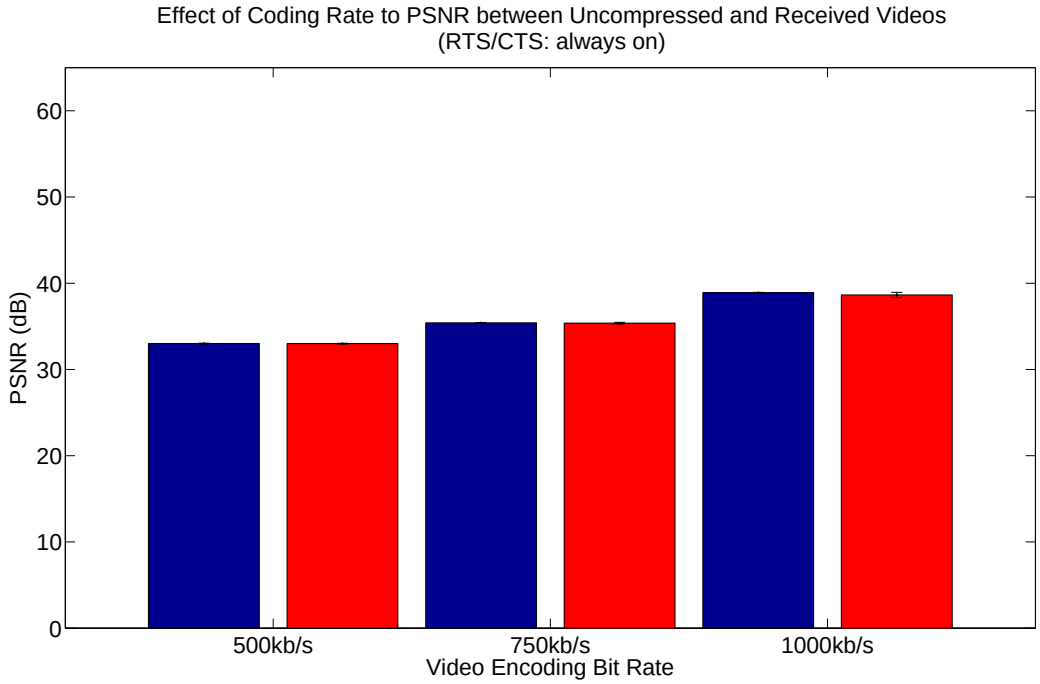

Figure 34: Effect of Video Encoding Rate to PSNR between Raw and Received Video (RTS/CTS:Always On)

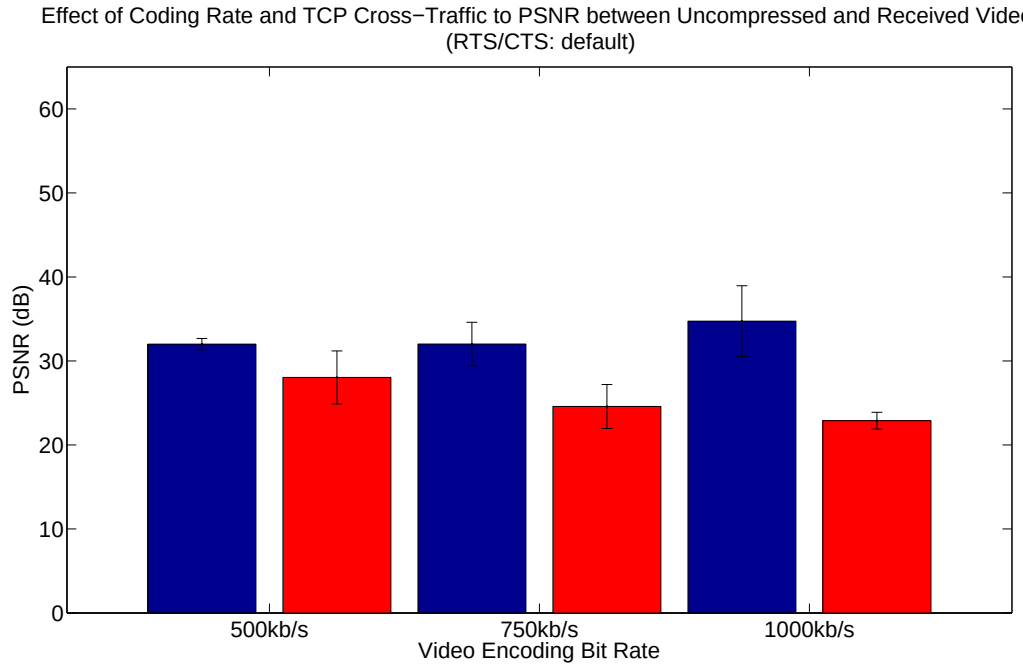

Figure 35: Effect of Video Encoding Rate to PSNR between Raw and Received Video under TCP cross traffic (RTS/CTS:Default)

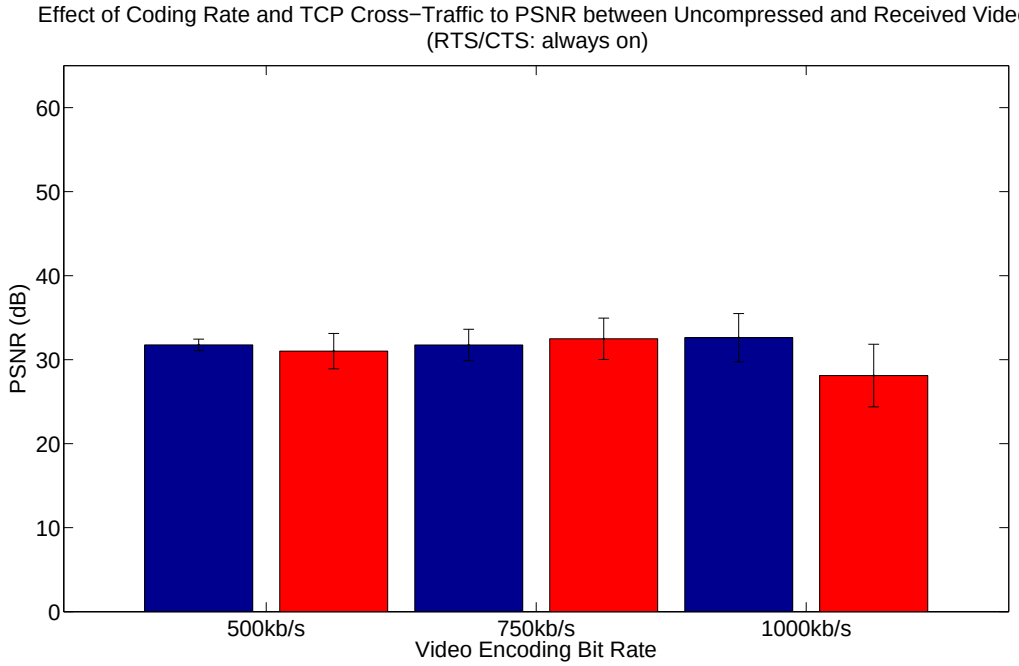

Figure 36: Effect of Video Encoding Rate to PSNR between Raw and Received Video under TCP cross traffic (RTS/CTS:Always On)

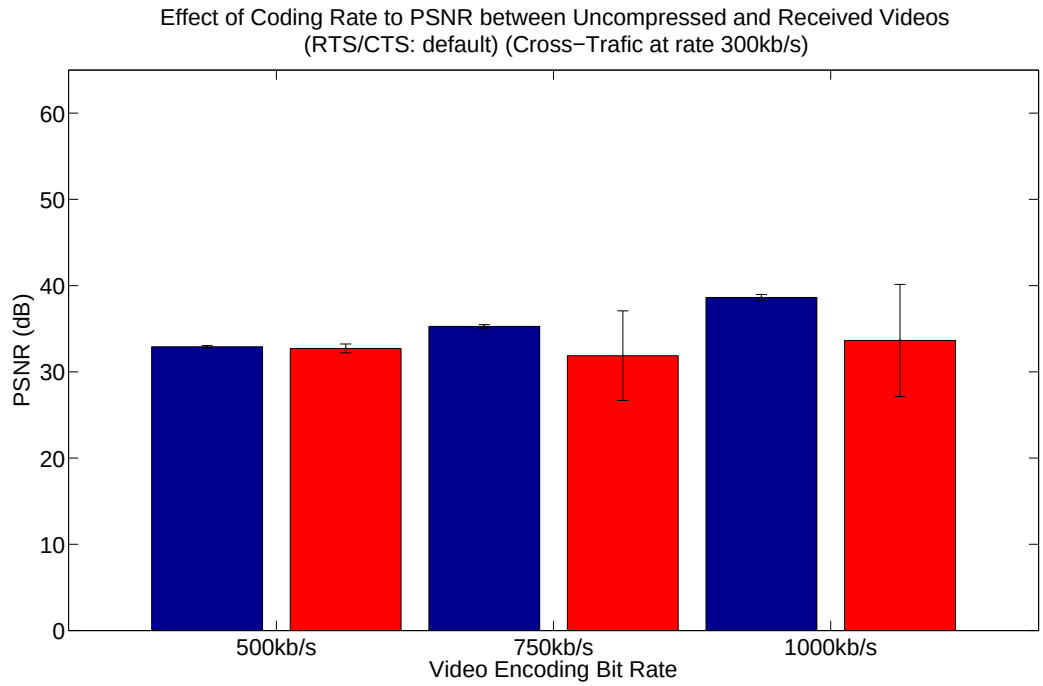

Figure 37: Effect of Video Encoding Rate to PSNR between Raw and Received Videos under 300kb/s UDP cross traffic (RTS/CTS:Default)

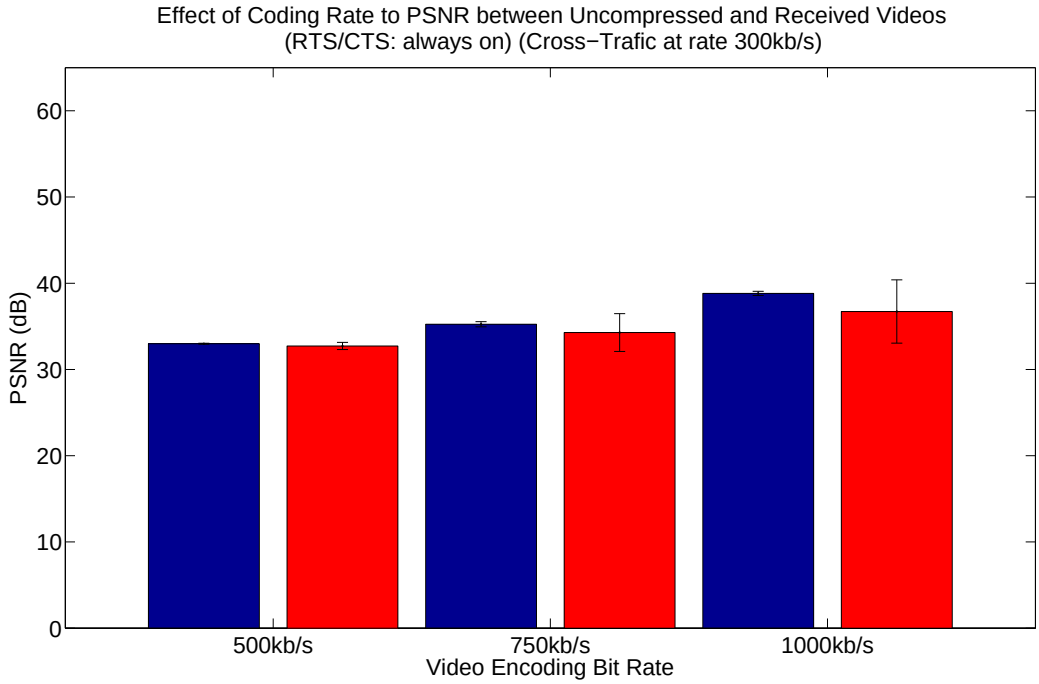

Figure 38: Effect of Video Encoding Rate to PSNR between Raw and Received Videos under 300kb/s UDP cross traffic (RTS/CTS:Always On)

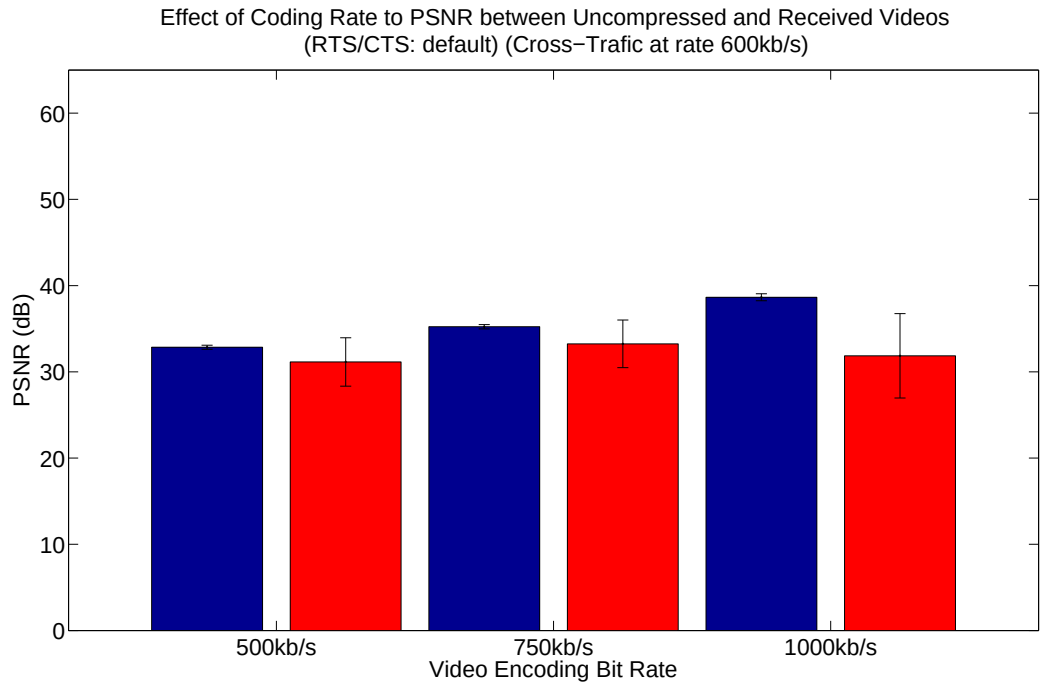

Figure 39: Effect of Video Encoding Rate to PSNR between Raw and Received Videos under 600kb/s UDP cross traffic (RTS/CTS:Default)

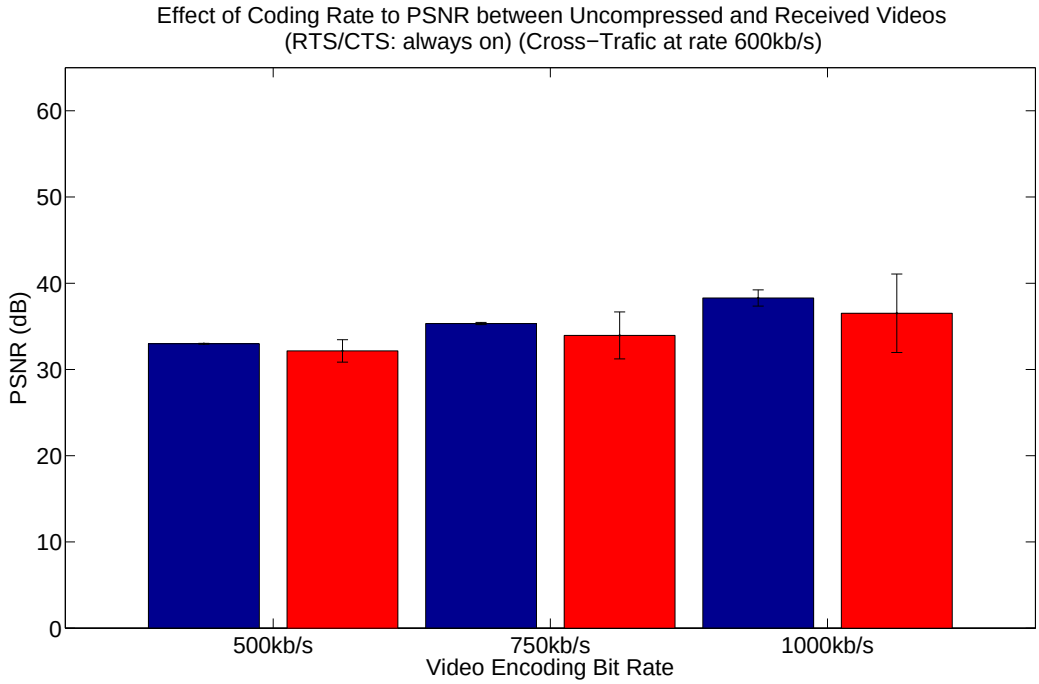

Figure 40: Effect of Video Encoding Rate to PSNR between Raw and Received Videos under 600kb/s UDP cross traffic (RTS/CTS:Always On)

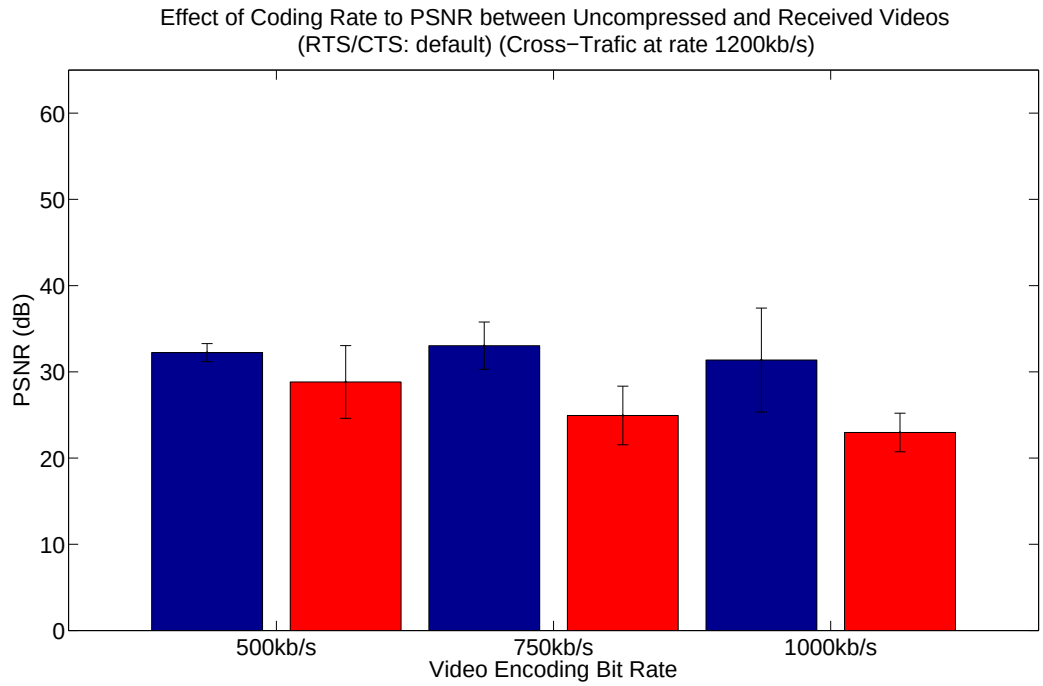

Figure 41: Effect of Video Encoding Rate to PSNR between Raw and Received Videos under 1200kb/s UDP cross traffic (RTS/CTS:Default)

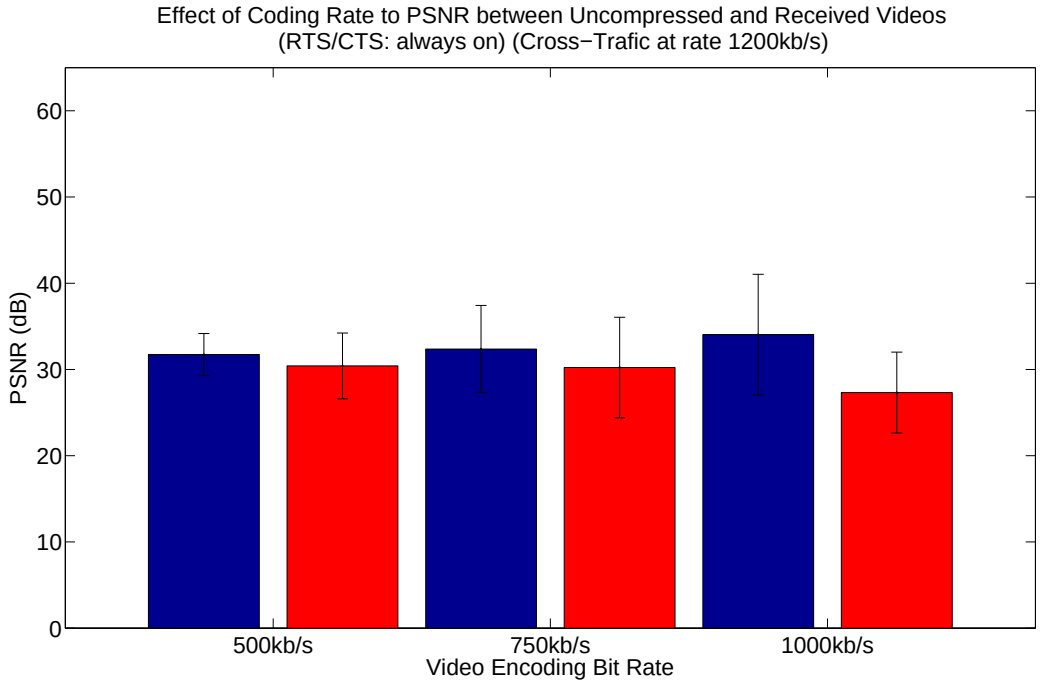

Figure 42: Effect of Video Encoding Rate to PSNR between Raw and Received Videos under 1200kb/s UDP cross traffic (RTS/CTS:Always On)

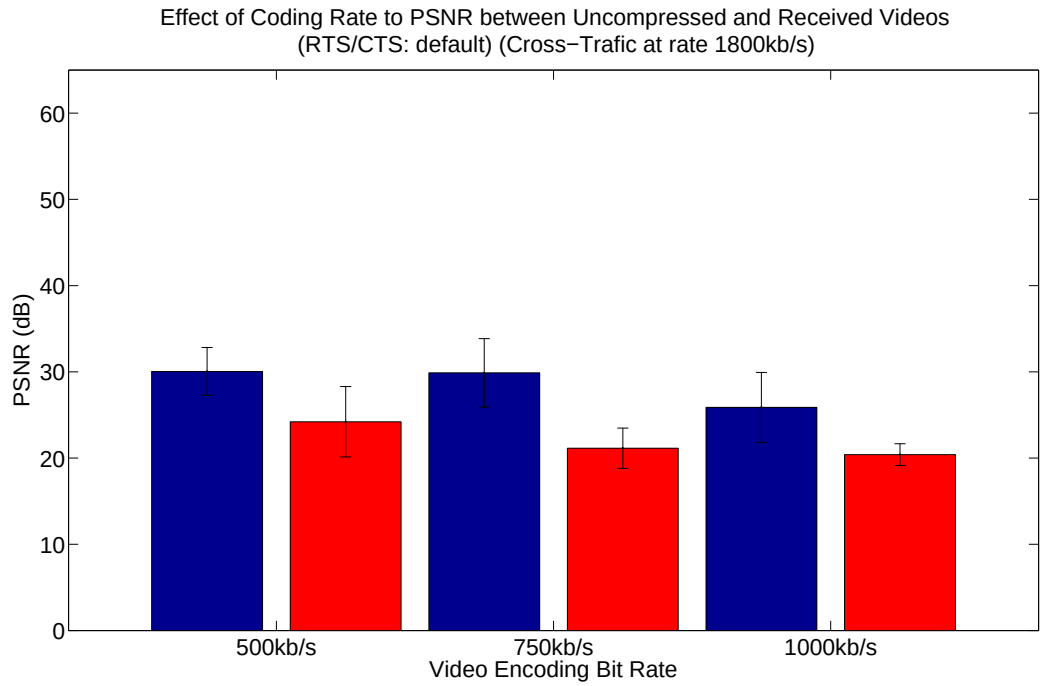

Figure 43: Effect of Video Encoding Rate to PSNR between Raw and Received Videos under 1800kb/s UDP cross traffic (RTS/CTS:Default)

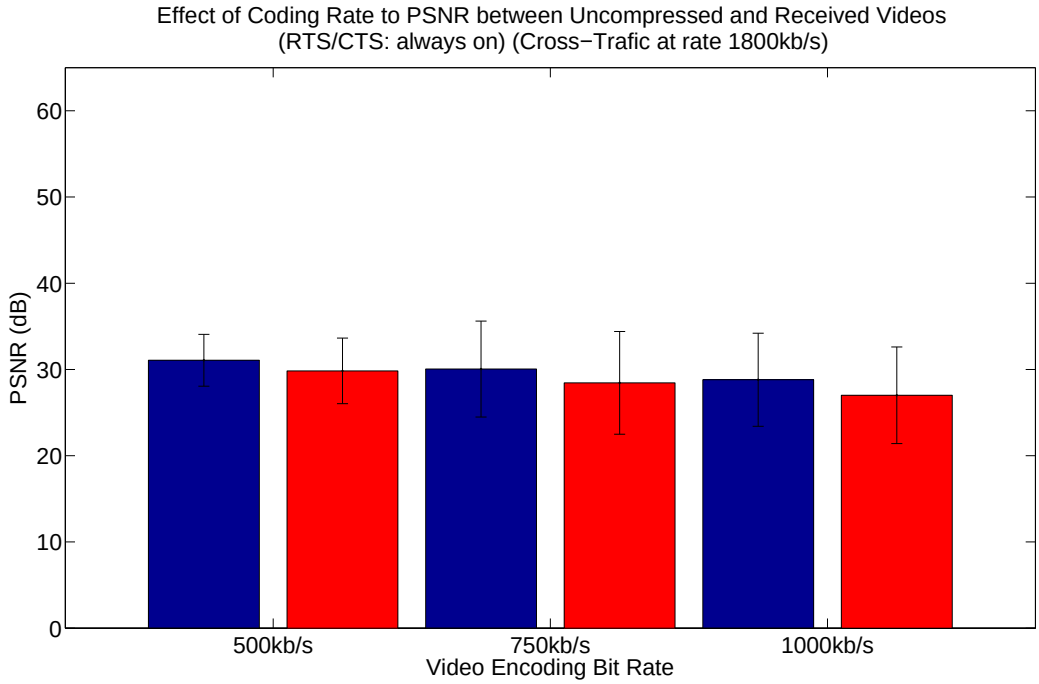

Figure 44: Effect of Video Encoding Rate to PSNR between Raw and Received Videos under 1800kb/s UDP cross traffic (RTS/CTS:Always On)

CHAPTER VII

CONCLUSION

In this thesis, we compared B.A.T.M.A.N. and OLSR routing algorithms over a multimedia WMN. We made our comparison based on their PSNR achievement for streaming videos when there are different kinds of cross-traffic at various rates in the network. To summarize the conducted research, first we can say that both of the routing protocols can achieve multihop video streaming and the received video may have PSNR up to 60 dB and both routing protocols can carry cross traffic up to some level.

B.A.T.M.A.N. routing protocol generally performs better than OLSR routing protocol. But when the routing protocols are forced to use RTS/CTS handshake mechanism for each packet they transmitted, performance of B.A.T.M.A.N. routing protocol doesn't increase but performance of OLSR routing protocol increases and OLSR routing protocol performs as good as B.A.T.M.A.N. routing protocol. This shows that B.A.T.M.A.N. routing protocol is robust to hidden node problem but OLSR routing protocol is not. Both of the routing protocols suffer from the high deviation in the received video quality, when there is no cross traffic, the setting RTS/CTS handshake mechanism to be always on helps to decrease this deviation but if there is either UDP or TCP cross traffic, even changing the setting of RTS/CTS handshake mechanism may not help the routing protocols in terms of variation in their transfer. However OLSR routing protocol is an optimized link state algorithm, it uses hop count as routing metric so nodes try to transmit further nodes with whom they have unreliable connection.

Also, we saw that as either cross traffic rate or streamed video encoding rate is increased the PSNR between encoded video and received video decreases.

Finally, we saw that however the PSNR between the encoded video and received video is getting worse as the encoding bit rate increases, when we checked the PSNR between the raw video and received video, we saw that PSNR is higher in the case where the videos are encoded at a higher bit rate if there is no cross traffic or UDP traffic at a low rate. But if the rate of cross traffic increases, we saw that the PSNR between raw and received video starts to decrease as the video encoding bit rate increases. This leads to the conclusion that however encoding videos at higher bit rates suffer more during their transmission through the network, since the encoded videos at a lower rate lose more data during compression, the received video with a higher encoding is received in a higher quality if there is no cross traffic or cross traffic at a lower rate. However, if there is cross traffic at a high rate in the network, the videos encoded at a higher bit rate start to lose more data during transmission and after being transmitted, they end up with more data detriment.

APPENDIX A

INITIALIZATION SCRIPT

#!/bin/sh

echo $LOCALIP $NETMASK $OLSRIP $OLSRNET

uci set wireless.radio0.disable='0' uci rename wireless.@wifi-iface[0]=ap uci set wireless.ap.ssid='deneme'

uci add wireless wifi-iface uci rename wireless.@wifi-iface[-1]=mesh uci set wireless.mesh.device='radio0' uci set wireless.mesh.network='wlan' uci set wireless.mesh.mode='adhoc' uci set wireless.mesh.ssid='ErdEm' uci set wireless.mesh.encryption='none'

uci set network.lan.ifname='eth0 wlan0' uci set network.lan.ipaddr=192.168.11.1 uci set network.lan.netmask=255.255.255.0

uci del olsrd.@Interface[0].interface uci add_list olsrd.@Interface[0].interface='lan wlan' uci add olsrd Hna4 uci rename olsrd.@Hna4[-1]=Hna4 uci set olsrd.Hna4.netaddr=192.168.11.0 uci set olsrd.Hna4.netmask=255.255.255.0

uci set network.wlan='interface' uci set network.wlan.ifname='wlan0-1' uci set network.wlan.proto='static' uci set network.wlan.ipaddr=10.1.1.11 uci set network.wlan.netmask=2

uci commit

/etc/init.d/network restart /etc/init.d/olsrd start /etc/init.d/olsrd enable

APPENDIX B

SCRIPT TO START B.A.T.M.A.N.-ADV

```
#!/bin/sh
    uci set network.lan.ipaddr=$LOCALIP uci set network.lan.ifname='eth0 bat0'
    uci set network.bat0=interface uci set network.bat0.ifname=bat0 uci set network.bat0.proto=nor
uci set network.bat0.mtu=1500
    uci set network.mesh0=interface uci set network.mesh0.proto=none uci set net-
work.mesh0.mtu=1528
    ci set wireless.ap.ssid='BATMAN-ADV'
    uci set wireless.mesh.network=mesh0 uci set wireless.mesh.mode=adhoc uci set
wireless.mesh.ssid='ErdEm' uci set wireless.mesh.mcast_rate=11000
    uci commit
    /etc/init.d/network restart
```

APPENDIX C

SCRIPT TO START OLSR

#!/bin/sh

uci set network.lan.ifname='eth0 wlan0' uci set network.lan.ipaddr=$OLSRIP

uci set network.wlan='interface' uci set network.wlan.ifname='wlan0-1' uci set network.wlan.proto='static' uci set network.wlan.ipaddr=$OLSRNETIP uci set network.wlan.netmask=$NETMASK

uci set wireless.ap.ssid='OLSR'

uci set wireless.mesh.network=wlan uci set wireless.mesh.mode=adhoc uci set wireless.mesh.ssid=ErdEm

uci commit

/etc/init.d/network restart /etc/init.d/olsrd start /etc/init.d/olsrd enable

APPENDIX D

SCRIPTS TO REMOVE ROUTING ALGORITHMS

D.1 Script to Remove OLSR

```
#!/bin/sh
    /etc/init.d/olsrd disable /etc/init.d/olsrd stop
    uci del wireless.mesh.ssid
    uci del network.wlan
    uci commit
```

D.2 Script to Remove B.A.T.M.A.N.-Adv

```
#!/bin/sh
    uci del network.bat0 uci del network.mesh0
    uci del wireless.mesh.ssid uci del wireless.mesh.mcast_rate
    uci commit
```

APPENDIX E

TFTP SETTINGS

The /etc/xinetd.d/tftp file should be opened with root privileges and following lines should be add.

service tftp protocol = udp port = 69 socket_type = dgram wait = yes user = nobody server = /usr/sbin/in.tftpd server_args = /tftpboot disable = no

tftp folder should be created with owner nobody. The following commands can be used for this.

sudo mkdir /tftpboot sudo chmod -R 777 /tftpboot sudo chown -R nobody /tftpboot

APPENDIX F

UNPAIRED T-TEST

Unpaired t-test is used to compare statistical information of individual sample groups that are not paired or matched to each others. For t-test, t and υ are calculated according to appropriate t-test and υ is placed in the Student's t Distribution formula and the distribution formula is integrated from $-\infty$ to t.

The Student's t-Distribution

$$y = f(x|\upsilon) = \frac{\Gamma\left(\frac{\upsilon+1}{2}\right)}{\Gamma\left(\frac{\upsilon}{2}\right)} \frac{1}{\sqrt{\upsilon\pi}} \frac{1}{\left(1+\frac{x^2}{\upsilon}\right)^{\frac{\upsilon+1}{2}}} \tag{4}$$

For unpaired t-test t is calculated with the following formula:

$$t = \frac{\overline{X} - \overline{Y}}{\sqrt{\frac{{S_X}^2}{n_1} + \frac{{S_Y}^2}{n_2}}} \tag{5}$$

υ is the degree of freedom for t-test and for unpaired t-test, υ is calculated with the following formula:

$$df = \frac{\left(\frac{{S_X}^2}{n_1} + \frac{{S_Y}^2}{n_2}\right)^2}{\frac{\left(\frac{{S_X}^2}{n_1}\right)^2}{n_1 - 1} + \frac{\left(\frac{{S_Y}^2}{n_2}\right)^2}{n_2 - 1}} \tag{6}$$

In these formulas $\overline{X}$ and $\overline{Y}$ are the means of respectively the first and second sample groups, S_x and S_y are the standard deviations of respectively the first and second sample groups. n_1 and n_2 are number of samples of respectively first and second sample groups.

Bibliography

[1] I. . W. Group, "Ieee standard for information technology-telecommunications and information exchange between systems-local and metropolitan area networks-specific requirements - part 11: Wireless lan medium access control (mac) and physical layer (phy) specifications," *IEEE Std 802.11-2007 (Revision of IEEE Std 802.11-1999)*, pp. C1–1184, 12 2007.

[2] L. Litwin and M. Pugel, "The principles of ofdm," *RF signal processing*, vol. 2, pp. 30–48, 2001.

[3] "Optimized link state routing protocol," October 2003.

[4] L. M. Neumann A., Aichele C. and W. S., "Better approach to mobile ad-hoc networking (b.a.t.m.a.n.)," October 2008.

[5] M. Benzaid, P. Minet, and K. Al Agha, "Integrating fast mobility in the olsr routing protocol," in *Mobile and Wireless Communications Network, 2002. 4th International Workshop on*, pp. 217–221, 2002.

[6] J. Klein, "Implementation of an ad-hoc routing module for an experimental network," Master's thesis, University of Stuttgart, Polytechnic University of Catalunya, 2005.

[7] J. Bicket, S. Biswas, D. Aguayo, and R. Morris, "Architecture and evaluation of the mit roofnet mesh network," 2005.

[8] M. Hempel, H. Sharif, T. Zhou, and P. Mahasukhon, "A wireless test bed for mobile 802.11 and beyond," in *Proceedings of the International Conference on Wireless Communications and Mobile Computing, IWCMC 2006, Vancouver, British Columbia, Canada, July 3-6, 2006* (S. Onoe, M. Guizani, H.-H. Chen, and M. Sawahashi, eds.), pp. 1003–1008, ACM, 2006.

[9] K.-c. Lan, Z. Wang, M. Hassan, T. Moors, R. Berriman, L. Libman, M. Ott, B. Landfeldt, and Z. Zaidi, "Experiences in deploying a wireless mesh network testbed for traffic control," *SIGCOMM Comput. Commun. Rev.*, vol. 37, no. 5, pp. 17–28, 2007.

[10] M. Abolhasan, B. Hagelstein, and J. C. P. Wang, "Real-world performance of current proactive multi-hop mesh protocols," in *Communications, 2009. APCC 2009. 15th Asia-Pacific Conference on*, pp. 44–47, 2009.

[11] B. Rong, Y. Qian, K. Lu, R. Hu, and M. Kadoch, "Multipath routing over wireless mesh networks for multiple description video transmission," *Selected Areas in Communications, IEEE Journal on*, vol. 28, no. 3, pp. 321–331, 2010.

[12] J. C. P. Wang, B. Hagelstein, and M. Abolhasan, "Experimental evaluation of ieee 802.11s path selection protocols in a mesh testbed," in *Signal Processing and Communication Systems (ICSPCS), 2010 4th International Conference on*, pp. 1–3, 2010.

[13] R. Garroppo, S. Giordano, and L. Tavanti, "Experimental evaluation of two open source solutions for wireless mesh routing at layer two," in *Wireless Pervasive Computing (ISWPC), 2010 5th IEEE International Symposium on*, pp. 232–237, 2010.

[14] D. Murray, M. Dixon, and T. Koziniec, "An experimental comparison of routing protocols in multi hop ad hoc networks," in *Telecommunication Networks and Applications Conference (ATNAC), 2010 Australasian*, pp. 159–164, 2010.

[15] D. Seither, A. Konig, and M. Hollick, "Routing performance of wireless mesh networks: A practical evaluation of batman advanced," in *Local Computer Networks (LCN), 2011 IEEE 36th Conference on*, pp. 897–904, 2011.

[16] C. Gottron, P. Larbig, A. Konig, M. Hollick, and R. Steinmetz, "The rise and fall of the aodv protocol: A testbed study on practical routing attacks," in *Local Computer Networks (LCN), 2010 IEEE 35th Conference on*, pp. 316–319, 2010.

[17] W. Kaula, "Building easily deployable mesh networks for first respondents based on 802.11n access points," Master's thesis, Athens Information Technology, 2011.

[18] S. Winkler and F. Dufaux, "Video quality evaluation for mobile applications," 2003.

[19] N. Cranley and M. Davis, "Performance evaluation of video streaming with background traffic over ieee 802.11 wlan networks," in *Proceedings of the 1st ACM workshop on Wireless multimedia networking and performance modeling*, WMuNeP '05, (New York, NY, USA), pp. 131–139, ACM, 2005.

[20] X. Cheng, P. Mohapatra, S.-J. Lee, and S. Banerjee, "Performance evaluation of video streaming in multihop wireless mesh networks," in *Proceedings of the 18th International Workshop on Network and Operating Systems Support for Digital Audio and Video*, NOSSDAV '08, (New York, NY, USA), pp. 57–62, ACM, 2008.

[21] Z. Wang, S. Banerjee, and S. Jamin, "Studying streaming video quality: from an application point of view," in *Proceedings of the eleventh ACM international conference on Multimedia*, (New York, NY, USA), pp. 327–330, ACM, 2003.

[22] A. J. Chan, K. Zeng, P. Mohapatra, S.-J. Lee, and S. Banerjee, "Metrics for evaluating video streaming quality in lossy ieee 802.11 wireless networks.," in *INFOCOM*, pp. 1613–1621, IEEE, 2010.

[23] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "Rtp: A transport protocol for real-time applications," 2003.

VITA

Sırrı Erdem ULUSOY has born in Kırıkkale, in 1986. In 2001, he won bronze medal in the Elementary Mathematics Olympics conducted by TUBITAK. In 2004, he placed in the first thousand and is enrolled in Electrical and Electronics Department, METU as an undergradute student. During his undergraduate study, for an academic year, he has been as an exchange student in Paderborn, Germany. After he got his undergraduate degree, he continued Graduate School of Engineering in Özyeğin University. For his master thesis, he studied on Video Streaming on Wireless Mesh Networks. He is currently an M.Sc. student in Özyeğin University.