Evaluating the Performance of Real Time Videoconferencing in Ad Hoc Networks Through Emulation

Jorge Hortelano, Juan-Carlos Cano, Carlos T. Calafate, Pietro Manzoni
Department of Computing Engineering
Technical University of Valencia, Spain
jorgehortelano@gmail.com, {jucano, calafate, pmanzoni}@disca.upv.es

Abstract

The validation of new video protocols and applications for mobile ad hoc networks in a real environment is an important task. In this work we present Castadiva, a test-bed architecture that allows validating software solutions for ad hoc networks using low-cost, off-the-shelf devices and open source software. We use this tool to test a videocall using the OLSR protocol in different scenarios, varying the number of hops between the caller and the receiver.

The results obtained in this paper show that, for an ad hoc network with a large number of hops, the quality of videocalls suffers a significant degradation even in the absence of mobility.

1. Introduction

Advancements in video compression standards and CPU speeds have favoured the widespread adoption of videoconferencing applications. Videoconferencing is used in a plethora of scenarios and situations, such as internal company communications, educating and training remote employees, telecommuting, etc.

Mobile ad hoc wireless networks (MANETs) consist of mobile nodes interconnected by multihop communication paths. Unlike conventional wireless networks, ad hoc networks have no fixed network infrastructure or administrative support. The topology of the network changes dynamically as mobile nodes join or depart the network or radio links between nodes become unusable. However, a price for all those features is paid in terms of complex technological solutions [1]. The importance of MANETs becomes evident by noticing the wide application area that these embrace. Special situations need communication networks without any previous infrastructure, like emergency missions, military operations or ad hoc meetings. However, there are many challenges for supporting video communications over MANETs. MANETs require efficient routing protocols to operate correctly. A routing protocol specifies how nodes communicate with each other to disseminate routing information, allowing them to create routes between any two nodes on a network. Many of the academic papers dealing with ad hoc and mesh networks [2, 3] evaluate protocols assuming varying degrees of mobility within a bounded area.

Researchers in this field have two options for testing new MANET protocols: (a) simulation tools and (b) test-beds.

A simulation tool is an application which behaves or operates like a given real system when provided with a set of controlled inputs. Currently several network simulators exist, including ns-2 [4] and OPNET [5]. Computer simulation is the most popular way to evaluate MANET routing protocols. Simulation offers four important advantages: First, it enables experimentation with large networks. Second, it enables experimentation with configurations that may not be possible with existing technology. Third, it allows rapid prototyping. Finally, it makes reproducible experiments in a controlled environment possible. Simulations also have some disadvantages: First, researchers can not test there their own real world implementation of a protocol in a realistic scenario. Second, simulators typically do not incorporate realistic models of node mobility and radio signal propagation.

A test-bed is a platform for experimentation which allows for rigorous, transparent and replicable testing of scientific theories, computational tools, and other new technologies. Several prototypes for generating a real ad hoc network test-bed can also be founded in the literature, like mLab [6], the wireless emulator from Carnegie Mellon University [7], and test emulators like ORBIT [8] or MobiEmu [9]. Test-beds also have some disadvantages: the test-bed called mLab can only generate network topologies and capture packets, Carnegie Mellon University's emulator does not use commercial off-the-shelf devices, while ORBIT or MobiEmu use expensive equipment to do their tests, being limited in terms of radio signal modelling as occurs with simulators.

In this paper we study the performance evaluation of videoconference applications over MANETs. We use our test-bed called Castadiva. Castadiva [10] is an architecture that facilitates making test-bed experiments; it relies on low-cost, of the-shelf devices combined with the power of a Linux platform. Castadiva allows generating network topologies, exporting them to real devices and obtaining the test results. It relies on a cheap architecture that includes two different networks: a wired network, called connection network, that connects the core with a group of wireless nodes, and a wireless network where the actual testbed experiments are made.

In the paper we specifically focus on enhancements to Castadiva to enable testing with traffic from real applications. The application being tested is Ekiga [11], a videoconferencing platform offering real-time communication between users supporting both SIP [12] and H.323 [13] standards. We obtain the results when testing a videocall in different MANET environments using the OLSR routing protocol [14].

The rest of this paper is organized as follows: Section 2 describes Castadiva's architecture. Section 3 presents some implementation details. Section 4 describes the enhancements that Castadiva offers to allow attaching external traffic sources, and Section 5 describes our experimentations using video traffic. Finally, Section 6 presents some concluding remarks.

2. Castadiva's Architecture Overview

Castadiva is a test-bed developed to deal with performance evaluation of protocols and applications in MANETs. Castadiva is composed by a server that runs the main application, several wireless nodes, two different networks and an application to coordinate all devices.

Castadiva's server executes the application and configures the network devices. Concerning wireless nodes used, they can be any sort of computing device, like a laptop, a PDA or a wireless router. In our prototype, each node is a Linksys router. The main requirement for a node is that it must have a Linux/Unix operating system installed, and two network cards: an Ethernet card and an IEEE 802.11 card. If the node is a wireless router, the OpenWRT [15] kernel is an optimum solution.

Figure 1 shows a schema of Castadiva's components. The main application, controls all devices and manages the links among them according to a pre-defined network topology. It also manages traffic generation between pairs of nodes. Since the controlling application also requires communicating with nodes to send control packets, Castadiva

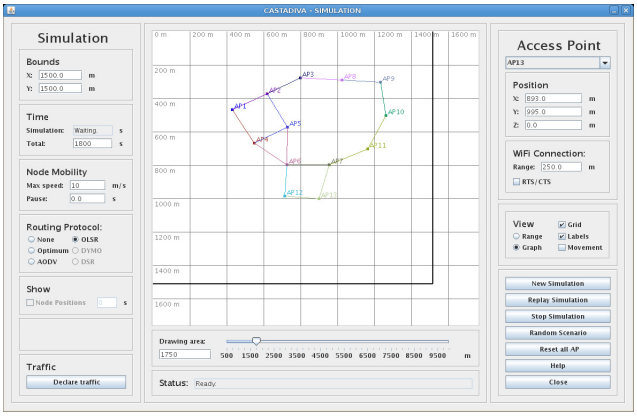

Figure 2. Scenario definition with Castadiva.

combines two different networks: the coordination network (wired), that connects the Castadiva core with the wireless nodes, allowing the main application to send configuration messages to all the nodes without creating any interference within the wireless network itself; and the wireless network, where actual tests are run and is composed by Castadiva's wireless nodes.

Castadiva's core has two main functions: (a) to allow a user to interact with the system so as to define all the test parameters required and (b) to coordinate the wireless nodes during an experiment. By using Castadiva's GUI a user can control all of Castadiva's functionality, defining the network topology and the traffic flow among nodes. Figure 2 shows Castadiva's GUI. We describe the whole functionality offered by Castadiva's GUI in Section 3.1.

Figure 3 shows our test-bed. One switch connects Castadiva's server with all the wireless nodes for coordination purposes. On the center of the picture the group of wireless nodes being used are shown. It consists of eleven routers. The wireless ad hoc network conformed by these nodes is the one used in Castadiva's testbed experiments. The experiments presented in this paper required extending Castadiva to support traffic injected from outside applications (see Section 4). In particular, laptops to go through the emulated MANET, allowing us to assess the performance of video conference sessions as experienced by users.

3. Castadiva's Implementation details

In this section we detail the requirements of Castadiva on the server and on the wireless nodes. We describe the software tools we have developed to connect all the wireless nodes with the server, and how Castadiva allows emulating connections among them. We also explain the process of designing network topologies by using the Scenario Generation tool, an interactive and user-friendly interface that

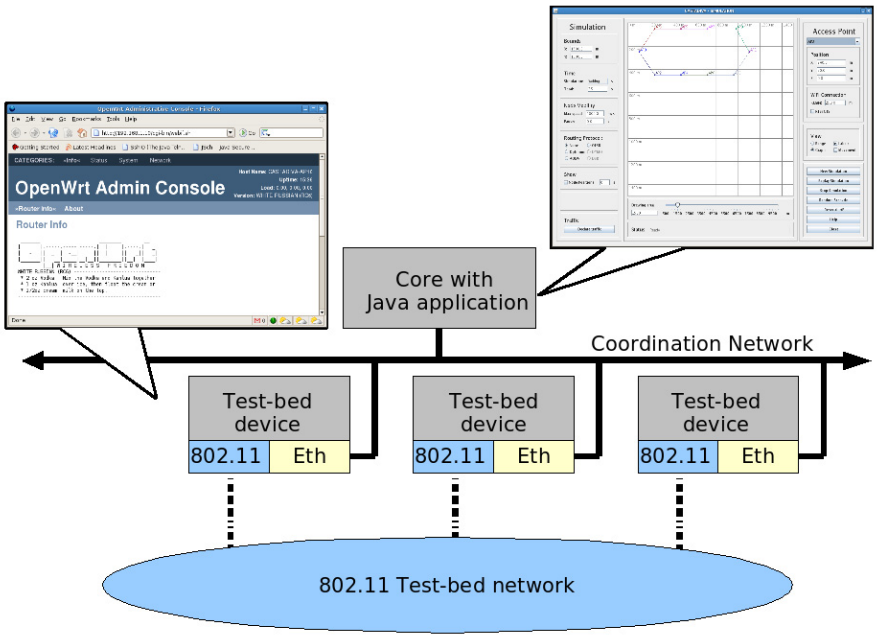

Figure 1. Schema of Castadiva's components.

Figure 3. Castadiva's physical network.

allows defining the network's scenario and the desired traffic connections among nodes.

Castadiva requires some libraries and services to operate. The requirements of Castadiva are different for the server and the wireless nodes. The server must be a standard Linux-based system and must have a Java Virtual Machine, an SSH client and an NFS server installed. Concerning nodes, each one must be a Linux based system with an SSH server and an NFS client.

The connection between Castadiva's core element (server) and each node is made using both SSH and NFS connections. On Castadiva's server, the user interacts with the GUI application by defining the network topology and other simulation parameters. Then, through SSH, the application sends a *start* instruction to each node through the coordination network (wired). Wireless nodes achieve coordination among themselves by executing the required binaries, which are stored into a server folder shared through NFS.

Each node has a set of requirements that must be met for successful operation: a Linux-based operating system, a set of special-purpose scripts, some specific applications and connectivity to Castadiva's server. The operating system installed on each router is OpenWRT. OpenWRT allows executing scripts natively and includes a simple SSH server used to receive instructions from Castadiva's server.

Network topology configuration is made through the Iptables tool. According to the selected topology, Iptables allows us to dynamically break the network links between pairs of nodes.

Castadiva also includes routing agents for well-known routing protocols, such as AODV and OLSR.

3.1. The Main application

Castadiva's core element, a Java application running at the server, includes all the control functions required for testbed experimentation. It is responsible for network topology maintenance, traffic control, as well as result calculation and presentation. Once the topology is defined, Castadiva must configure the wireless nodes according to that topology. The application communicates with each node through SSH connections to send the required instructions.

When all the experiments are finished, Castadiva's core must calculate the result statistics for the experiment by gathering all the data obtained, and finally show these results to the user. We now offer more details about the services offered by Castadiva:

Ad Hoc Network Scenario Generation: this window is a representation of a virtual environment where nodes are located. Here the user can edit node properties, such as position and signal range. This window also allows the user to define the scenario bounds, the test time, node mobility and the routing protocol used.

Adding Nodes to the Test-bed: the user can specify the parameters used by Castadiva's main application to connect nodes among themselves and with the main server: the MAC address (required for Castadiva to enforce topology changes), the folder shared by NFS and the SSH user and password used by the main application, to connect to each individual router and submit commands.

Mobility in Castadiva: Castadiva generates all node movements required for the simulation before it starts. It then calculates the visibility range for each node every second. This behavior is similar to the one provided by the "setdest" tool embedded in the ns-2 simulator. The obtained visibility is translated to Iptables rules and written to one file per node.

Network Traffic Declaration: this tool allows defining different types of traffic flows between pairs of nodes. Traffic parameters for each connection can be set depending on the type of protocol used.

Random test generator: sometimes it is useful to automate the testbed evaluation process varying different parameters. The user must specify the bounds of the scenario, the routing protocol used, the number of nodes for testing and how many times each test will be repeated using these parameters.

Compatibility with the ns-2 simulator: Castadiva can also import/export scenarios to/from ns-2, making it compatible with the most widely used MANET simulator. This characteristic offers the possibility of comparing results and reaching more meaningful conclusions.

4. Castadiva extensions for external traffic injection

The synthetic traffic tools developed for Castadiva are quite useful for launching automated tests to measure performance in different conditions. We considered, nevertheless, that extending our platform to support the injection of external traffic would be more interesting by allowing to assess performance at the user level also. For this reason, Castadiva was enhanced to incorporate an extra functionality that allows attaching an external data source.

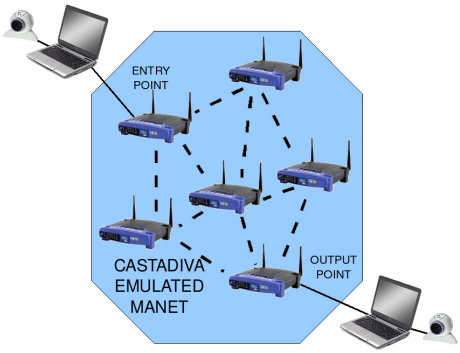

Figure 4. Schema of how to inject real video traffic int Castadiva's framework.

Such functionality allows external nodes to generate real traffic of any kind and redirect it to specific nodes of Castadiva. For example, you can use real applications like Ekiga to launch a video-conference and study the behaviour of H.323, SIP and video streaming protocols in MANETs.

Figure 4 shows an example of two laptops connected to Castadiva. Both laptops have a webcam and run the Ekiga application to generate a videocall. Castadiva redirects all traffic related to this video-conference through the emulated MANET, from the entry to the output point.

5. Video Traffic Evaluation

We now describe the performance results obtained when transmitting real-time video traffic through Castadiva. We test with several different scenarios where the number of hops varies from one to ten. The purpose is to stress the application and evaluate the impact of the number of hops in the quality of the received video stream. We are going to study two different kinds of scenarios: a standard videocall, where the video sequence is almost static, and a dynamic videocall, where we will point the webcams towards a screen that displays a movie

The scenario is defined in a 1500m x 1800m area, and the test time is of 5000 seconds. We set the wireless nodes' range to 250 meters. To monitor the video traffic we use the tcpdump application [16].

To generate the video traffic we have two laptops with an attached webcam running Ekiga, an open source VoIP and videocall application for Linux.

We generate an emulated MANET topology where all nodes are aligned according to a chain topology. Figure 5

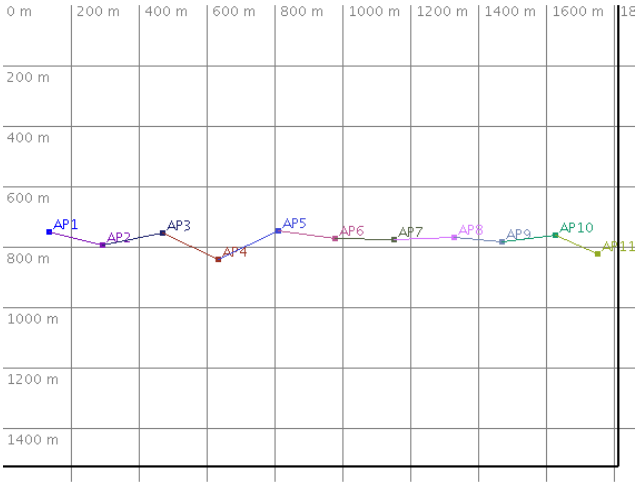

Figure 5. Topology for evaluating the traffic video.

Parameter	RFC Default Values	Used Values
HELLO_INTERVAL	2 s	5 s
REFRESH_INTERVAL	2 s	90 s
TC_INTERVAL	5 s	2 s
MID_INTERVAL	TC_INTERVAL	15 s
HNA_INTERVAL	TC_INTERVAL	15 s

Table 1. Values of the OLSR protocol parameters.

shows the topology used when eleven nodes are deployed. On each laptop we redirect all traffic sent to the other laptop through the wireless routers located at both edges of the simulation (AP1 and AP11 in this scenario). We also configure Castadiva to route all video traffic thought our testbed.

Each node runs the OLSR daemon to obtain consistent routing tables. We tested with the values shown in the RFC of OLSR, but the results obtained were somehow not so good, since the OLSR was unable to find the routes or the routes were lost too quickly. Therefore, instead of the defaults RFC parameters, we used the default configuration of OLSR implemented in the OpenWRT system. Table 1 shows both values.

The effects we want to study are the variations on throughput, the delay times and jitter when varying the number of hops in the network.

5.1. Evaluating the performance of with a videocall

We now present the results obtained when using Castadiva to evaluate a standard videocall between two users.

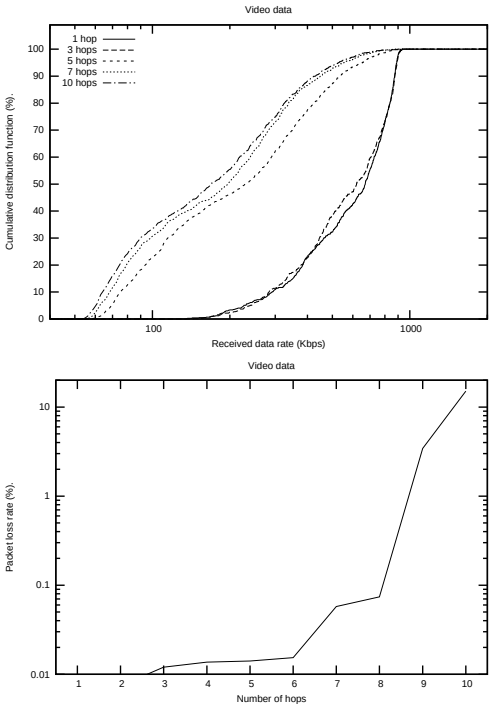

Figure 6. Cumulative distribution function for the throughput (top) and packet loss rate (bottom) in different scenarios.

Based on traffic traces at source and destination, we measured the throughput and the inter-packet delay (jitter).

Figure 6 shows the throughput cumulative distribution function obtained for each second of the test. In the one hop scenario, we can see how the traffic sent and the traffic received overlap (no packet losses).

As we increase the number of hops, we appreciate that there is a significant difference between the data sent and the data received. In fact, we find that the mean throughput is decreased to less than half compared to the one hop scenario (from around of 700 Kb/s to close to 150 Kb/s). Figure 6 (bottom) shows how the packet loss increases when we vary the number of intermediate hops between sender and receiver. In a scenario with more than 8 hops the packet loss rate is significant for the videocall. For a scenario of 10 hops the videocall experiences a 10% packet loss.

Figure 7 shows the cumulative distribution function for the inter-packet generation interval and inter-packet arrival interval in two scenarios where source and destination are one and ten hops away, respectively. In the scenario with one hop, the smallest network possible, we can appreciate the differences between the minimum inter-packet generation time at the source (around 0.01 milliseconds) and the minimum inter-packet arrival time at the receiver (around

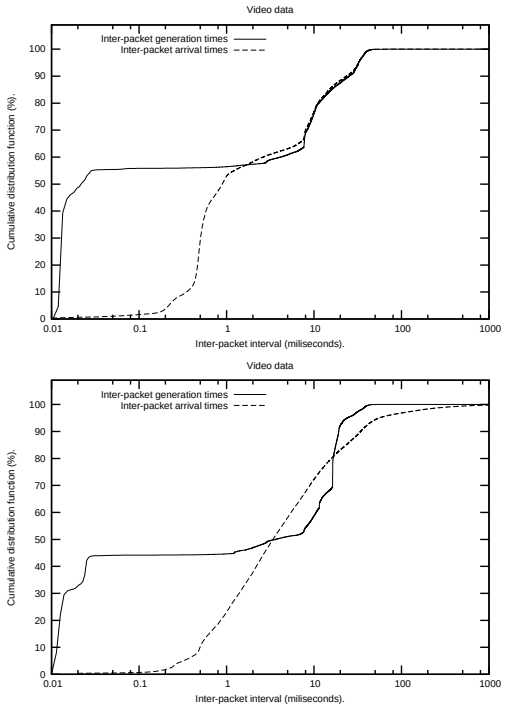

Figure 7. Cumulative distribution function for the inter-packet generation interval and inter-packet arrival interval in a scenario with one hop (top) and ten hops (bottom).

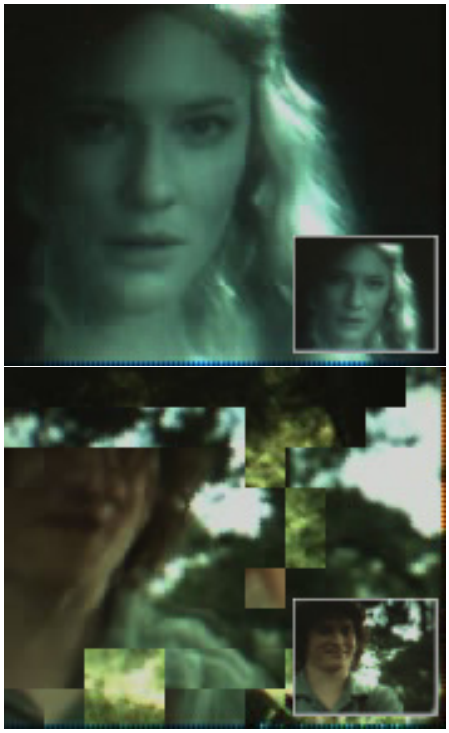

Figure 8. Image of the conversation with a scenario of one hop (top) and ten hops (bottom).

0.1 milliseconds). Also, 50% of the packets have an inter-packet generation time of less than 0.03 milliseconds, but the inter-packet arrival time is typically of more than 1 millisecond.

In the scenario with ten hops, if we compare the inter-packet arrival time with the one obtained in the one hop scenario, we can appreciate an increase in the inter-packet arrival of more than 2 milliseconds. This is caused by the forwarding time of the additional nodes on the path. Differences, in terms of inter-packet generation times are due to the rate control mechanism embedded in the Ekiga tool, which offers adaptive video compression according to network conditions. This is expected since these packets, usually generated back-to-back by the videoconferencing application, experience significant jitter when traversing an ad hoc network with a high number of hops.

5.2. Evaluating the performance of a movie transmission

In the previous section our experiments relied on a standard videocall. The characteristics of such videoconference - low degrees of video motion - do not impose significant demands in terms of network bandwidth. Therefore, we repeat our experiments with a higher motion video. With that purpose we pointed both webcams to a screen showing a movie. Such strategy also allows to run long experiments without intervention of users. By repeating our experiments and taking long sampling periods we were also able to obtain meaningful values for the end-to-end delay, and so we also include them in this section, as well as the measured throughput and jitter to compare with the previous test.

Figure 8 shows two frames obtained in the test, where we can see the quality of the video received in both scenarios. In the scenario with one hop, there is no significant delay between the real video sequence and the decoded one because the path is too short with respect to the decoded video sequence. In a scenario with ten hops, we can see the poor performance obtained by comparing the received and transmitted sequences. We also appreciate a delay of the received image compared to the real image (notice that both webcams aim at a same target for comparison purposes).

Figure 9 shows the cumulative distribution function for the inter-packet generation and arrival times in a scenario with one hop and in a scenario with ten hops. Notice that high delay values are quite prone to occur in the latter, often introducing significant jitter (above 100 ms).

Figure 10 shows the cumulative distribution function of

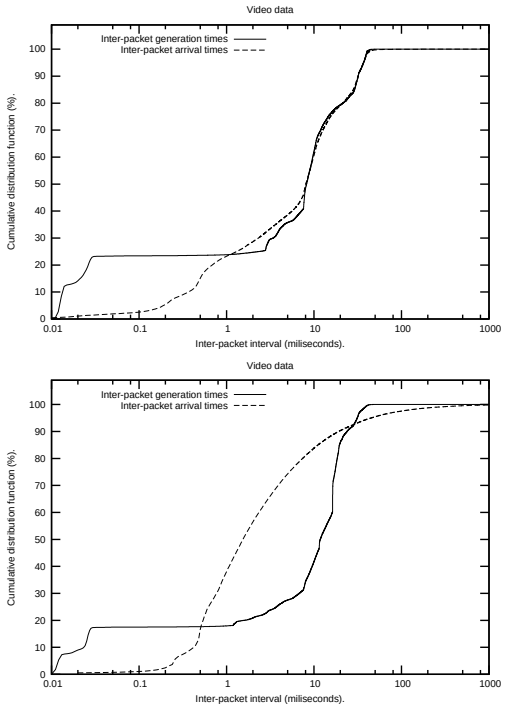

Figure 9. Cumulative distribution function for the inter-packet generation interval and inter-packet arrival interval in a scenario with one hop (top) and ten hops (bottom).

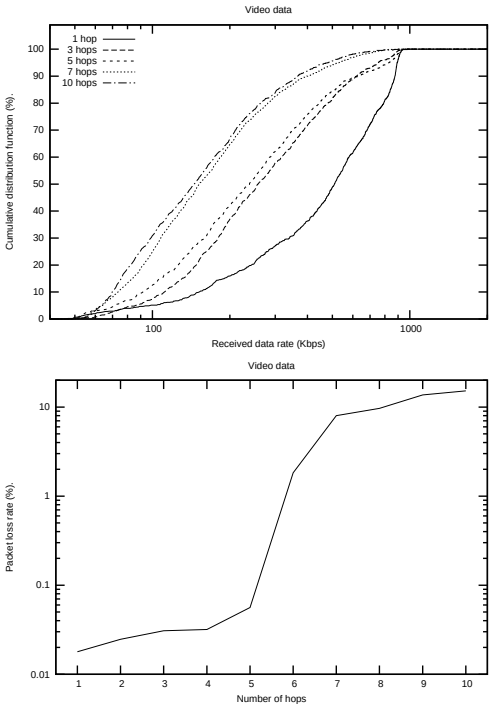

Figure 10. Cumulative distribution function for the throughput in a scenario with differents hops (top) and packet loss rate in different scenarios (bottom).

the generated data rate for different scenarios where we increase the number of intermediate hops. In a scenario with one hop, we have a data rate above 500 Kb/s 50% of the time. In a scenario with ten hops, the mean value of the generated data rate decreases to less than 200 Kb/s. We also evaluate the packet loss in each scenario. Figure 10 (bottom) shows how the packet loss increases when we vary the intermediate hops between sender and receiver. If we compare this graphic with the one obtained in the evaluation of a real videoconference, we can observe that now the packet loss rate increases more significantly. In fact, with more than 6 hops the packet loss rate surpasses 10%.

We also evaluated the behavior of increasing the number of hops by using ping sessions and measuring the impact on round-trip time. We generate a ping session between the two laptops in the chain scenario described earlier, varying the number of hops from one to ten. We obtain the average time of the ping session used to send a packet and receive the answer. To obtain the one-way delay, we divided the results obtained by two. Figure 11 shows the average delay and the standard deviation when varying the number of hops.

As expected, delay increases almost linearly with the number of hops between sender and receiver. Of special interest is the increase in terms of standard deviation, which can be quite problematic for real-time video transmission.

6. Conclusions

In this paper we present Castadiva, a novel architecture to improve research in the MANETs field by allowing to make real testbed experiments in a simple and straightforward manner.

This work shows that the advantages of using Castadiva with respect to other MANET test-beds are: (1) is a very low-cost test-bed since each node costs about 70$, (2) it is fully compatible with ns-2 allowing it to compare results with the most extended simulator, and (3) it does not occupy a lot of room to deploy the entire infrastructure.

Castadiva was extended to allow injecting traffic from externals applications. Such functionality enabled us to assess the performance of videoconferencing in an emulated ad hoc network using a real application such as Ekiga.

We studied the behavior of a videocall with different scenarios, changing the number of hops between caller and receiver from one to ten. We also varied the characteristics of

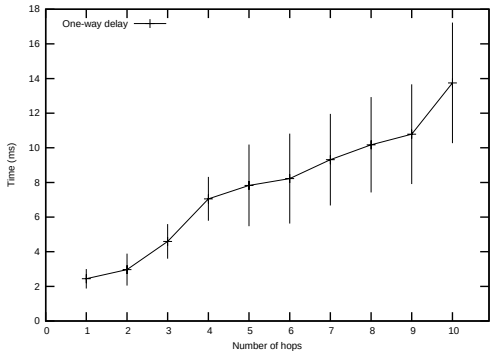

Figure 11. Evaluation of the ping sessions in different scenarios.

the video being transmitted.

The results obtained in the video evaluation show that, for a scenario with a large number of hops, delay times are sometimes high, which could provoke annoyance to the videocall users. Packet losses also become problematic when source and destination are more than six hops away.

As future work we will introduce QoS enhancements to our testbed to improve the performance of real-time video streaming.

Acknowledgments

This work was partially supported by the *Ministerio de Educación y Ciencia*, Spain, under Grant TIN2005-07705-C02-01.

References

[1] S. Basagni, M. Conti, S. Giordano, and I. Stojmenovic. *Mobile ad hoc networking*. IEEE Press, 2004.

[2] I. D. Chakeres and C. E. Perkins. Dynamic MANET on-demand routing protocol. *IETF Internet Draft*, June 2006.

[3] C. Perkins, E. Royer, and S. Das. Ad hoc on-demand distance vector (AODV) routing. *Internet Draft*, February 2003.

[4] UC Berkeley, LBL, USC/ISI, and Xerox PARC researchers. Network Simulator - ns (Version 2). Available at: http://www.isi.edu/nsnam/ns/, 1998.

[5] OPNET Technologies Inc. OPNET making networks and applications performs. Available at: http://www.opnet.com/.

[6] A. Karygiannis and E. Antonakakis. mLab: a mobile ad hoc network test bed. National Institute of Standards and Technology.

[7] Glenn Judd and Peter Steenkiste. Design and implementation of an rf front end for physical layer wireless network emulation. In *IEEE 2007 IEEE 65th Vehicular Technology Conference (VTC2007)*, Dublin, Ireland, April 2007.

[8] D. Raychaudhuri, I. Seskar, M. Ott, S. Ganu, and K. Ramachandran. Overview of the orbit radio grid testbed for evaluation of next-generation wireless network protocols. *Wireless Communications and Networking Conference, 2005 IEEE*, 3:1664–1669, 2005.

[9] Y. Zhang and W. Li. An integrated environment for testing mobile ad hoc networks. Available at: http://www.wins.hrl.com/projects/adhoc.

[10] J. Hortelano, J. C Cano, C. T. Calafate, and P. Manzoni. Castadiva: a test-bed architecture for mobile ad hoc networks. In *Symposium on Personal Indoor and Mobile Radio Communications (PIMRC 2007)*, Athens, Greece, September 2007. IEEE Communications Society.

[11] Ekiga, free your speech. Available at: http://ekiga.org/.

[12] J. Rosenberg, H. Schulzrinne, and G. Camarillo. SIP: Session initiation protocol. Request for Comments 3261, Network Working Group, http://tools.ietf.org/html/rfc3261, June 2002.

[13] ITU-T. Packet-based multimedia communications systems. available at http://www.itu.int/rec/T-REC-H.323-200606-I/en, June 2006.

[14] T. Clausen and P. Jacquet. Optimized link state routing protocol (olsr). Request for Comments 3626, MANET Working Group, http://www.ietf.org/rfc/rfc3626.txt, October 2003. Work in progress.

[15] OpenWRT, wireless freedom. Available at: http://openwrt.org.

[16] S. McCanne V. Leres and V. Jacobson. The tcpdump manual page. Lawrence Berkeley Laboratory, 6 89.