A QoS-Aware Ad Hoc Wireless Network for Isolated Rural Environments

Javier Simó[1], Joaquín Seoane[1], and Rodrigo de Salazar[1]

Fundación EHAS,
Despacho C-203-1, ETSI de Telecomunicación
Ciudad Universitaria s/n - 28040 Madrid (Spain)
Tel: +34 91 549 57 00 ext 493 - Fax: +34 91 336 73 33

Abstract. Ad-hoc wireless networks are constituted by routers enabled to establish wireless links among them in a mesh topology. This kind of networks are becoming very popular because of their potential, low cost and ease of deployment, specially when IEEE 802.11b wireless technology is used. There has been a lot of research in the design of protocols and algorithms for improving the performance of this kind of networks, and there have also been many efforts in studying the particular characteristics of their behavior for packet switching, specially when real-time traffic is a concern. However, real implementations and testbeds are scarce, and even those few well known interesting experiences are not QoS-aware and their implementation details are not consistent. This paper presents the design of a 802.11b multi-hop QoS-aware mesh network specifically designed for isolated rural environments by means of adapting the most appropriated software implementations available for IP auto-configuration, QoS-aware traffic control and dynamic multi-hop routing. This proposed network is used as a testbed for driving some initial experiments that aim to measure its performance when supporting both elastic and real-time traffic. The evaluation includes guidelines for a QoS-aware deployment of the network and recommendations for further work that can improve these results.

1 Introduction

Wireless ad hoc networks have been the subject of many researches during the last years. A wireless ad hoc network is constituted by several computing devices communicated with each other through wireless links, that cooperate for configuring a multi-hop ad hoc network automatically. Each node of the network can be the client, the server or an intermediate router within a communication, and one or several of them can eventually be for the others a gateway to other networks like the Internet or the PSTN (public switched telephone network). At the IP level, the address that each node gives to itself must be unique, but not necessarily similar to the others. Multi-hop routes are created by special dynamic routing protocols adapted to the special characteristics of this kind of networks.

Many researches in this field are concerned by the mobility of nodes; in these approaches, dynamic routing protocols are designed or chosen considering that topology changes very quickly. Unfortunately, few researches study in depth the specific problems of ad hoc routing when nodes are not mobile. AODV, DSR, OLSR and other accepted ad hoc routing protocols are not appropriated for this kind of scenario because they are mainly concerned by the variability of the network topology in MANETs, which is not critical in static ad hoc networks. On the other hand, these networks require routing protocols that take into account the real quality of wireless links in the metrics due to the fact that typical hop count is not enough in them [4]. Quality-of-Service (QoS) is another interesting aspect often referred from a theoretical point of view in researches about ad hoc networks [5] [7] [9].

Contrasting with the quality and quantity of theoretical research, there are few software packages implementing IP auto-configuration, ad hoc routing protocols and QoS architectures, and even those few products are not well tested. Moreover, real testbeds of static ad hoc networks as in [19] are very scarce, procedures used in them for routing and address configuration are not good enough for real applications, and none of them is concerned by QoS, as far as we know.

In the EHAS group [1], we work on telecommunication technologies adapted to isolated rural environments in developing countries, and specifically with health facilities and their communication to hospitals. Previous advances in rural telemedicine for developing countries [2] [3] proved that providing voice and data communications in small health rural spots gives great benefits such as a drastic reduction in the average evacuation time of critical patients, improvement of diagnostics' reliability, and decrease of travels needed by the staff.

Deploying reliable and sustainable communication networks in such environments needs some challenging implementation considerations: firstly, wireless technologies allow easy, cheap and infrastructure-free communication links and thus they are highly suitable for these scenarios; such installations will be generally difficult to access and therefore unsupervised, meaning that maintenance and system administration might be minimized; secondly, lack of reliable electric sources means that such terminals require low-power consumption and thirdly, low population density in such areas makes advisable the use of low-cost links.

This paper presents the development of a wireless solar-powered router designed to perform QoS-aware ad hoc routing in outdoor installations. This system is an embedded computer with a QoS-aware routing software powered by solar panels, having one or several WiFi cards provided with external antennas. A network can be easily deployed by installing some units in previously fixed positions so that antennas are well oriented and line-of-sight is assured between two systems supposed to be connecting to each other. IETF's zeroconf proposals have been used for IP self-configuration [11], a DiffServ-like simplified QoS architecture based on Linux advanced traffic control permits the QoS-aware packet switching [15] [16] [17] [18], and a free software implementations of the AODV ad hoc routing protocol has been used. The first prototype has been produced using a x86 embedded computer, but the software has been integrated in such a way

that cross-compiling the same software for other platforms is straight-forward. A second version of it is previewed using a StrongARM platform, which will reduce power consumption in 75will become smaller and much cheaper. The network designed has also served as a testbed permitting the evaluation of different dynamic routing protocols and technical procedures for partial QoS support.

A first experiment permitted to evaluate and compare several ad hoc routing protocols already implemented for Linux. Once that IP addressing and routing where solved, a second experiment permitted to test our simplified DiffServ-like implementation, which gave us some usable results about the network performance and the optimal configuration of queuing disciplines that permit a good coexistence between elastic and real-time traffic.

The final results of this work are intended to be used in real installations by the EHAS program.

2 The link level in wireless ad hoc networks

In the last ten years many new technologies have appeared that make possible the deployment of broadband wireless networks: Bluetooth, Hyperlan, Hyperlan/2, 802.11a, 802.11b, 802.11g, etc. Moreover, new technologies like 802.11e and 802.16 should be available soon, but the latter will take some time to be affordable for our scenarios. The choice of the most appropriated wireless technology for this project is determined by the next three constraints: the use for long distances, the use of a free band and a very low cost of devices. All this make 802.11b the most appropriated technology; however, it has several important limitations proved by other researches: the MAC cannot assure a good performance when different connexions share the channel, there are some negative interactions between the MAC and TCP, the technology doesn't support QoS, and there is not a definitive solution for the "hidden node problem" [6]. Those limitations must be taken into account in the way upper levels will use wireless links in order to minimize interferences what improves performance and stability.

The IEEE 802.11b standard describes the architecture and protocols for wireless local area networks using the spread spectrum technology at the ISM 2.4GHz band. There are basically two different operation modes which are infrastructure and ad hoc. In the infrastructure mode two types of entities are defined: access points and clients, while in ad hoc mode all nodes are identical. The service set constituted by an access point and the clients that access the network through it is called "basic service set" (BSS). Several access points can be associated in a so called "wireless distribution system" (WDS) which produces an "extended service set" (ESS) that is the union of all BSS involved. When using the ad hoc mode, a set of nodes connected among them is called "independent service set" (IBSS).

Obviously the ad hoc mode can serve to build ad hoc networks. It must be understood that "ad hoc" here means peer-to-peer, that is, two systems that see to each other and communicate directly. Many systems having the same SSID

(service set identifier) and keys can connect among them as far as each one see all the other systems. Thus, using the ad hoc mode each node see only its neighbours.

3 The IP level in wireless ad hoc networks

As seen at the IP level, an ad hoc network, also called "mesh network", is a set of nodes that can all act as clients, servers or routers in different communications and that constitutes a multi-hop network dynamically build and configured. Thus, the objective of a mesh network is that several independent IP nodes constitute a single IP network where any node can communicate with any other one and even with other networks if any of them act as a gateway. The mechanisms that make this possible are mainly two: address auto-configuration and multi-hop ad hoc routing. There are several theoretical proposals for address auto-configuration, but the most important is the IETF's zeroconf's model [11]. Unfortunately, most of the real implementations of ad hoc networks don't use this solution and prefer to solve the addressing in a different way. After studying the different proposals, zeroconf's seems to be the best, which made us decide to adopt it. There is a software implementation of it for Linux called zcip that has been chosen to perform address auto-configuration. However, we must say that we have ceased using it after the tests presented in this paper since the authors have announced a conflict of patents. Nowadays we are looking for a definitive solution, but in the meantime we produce unique IP addresses using the method used in MIT's RoofNet testbed [19]. When using zeroconf's system, each node was able to find a unique 169.254.X.Y IP address. A DHCP server is also installed in the node for which configuration changes each time the node is rebooted, so that it proposes addresses of the form192.168.X.Y to clients. The node will perform network address translation (NAT) to its clients.

There are many multi-hop ad hoc dynamic routing protocols designed and simulated, some of which have been implemented and tested in the real world. AODV is the best known protocol of this type, and at least two well tested software implementations of it can be used in Linux. Other implemented protocols are DSR and OLSR. It has been established that proactive protocols are not well suited for ad hoc networks where the throughput is considered a critical resource because routing table broadcasts may starve all available resources. But even reactive protocols whose metrics is the number of hops have some important limitations in wireless networks [4]. A wireless link in a 802.11b network may have a speed as of 1,2,5.5 and 11 Mbps, but the real throughput is much lower because it depends on the quality of the signal, which may vary with the distance and the presence of interferences. The routing protocol must take this into consideration because the shortest path may not be the best route. The available protocols implemented and having a metrics well suited for wireless networks are scarce. In fact, at the moment of our study we were not able to identify any available implementation of a multi-hop ad hoc routing protocol meeting these

requirements. Some QoS-aware ad hoc routing protocols as QOLSR and ETX are being implemented, but those implementations were still not usable.

We have tested some implementations of standard AODV and OLSR looking for good node discovery, stability, low impact in performance and good gateway discovery. After testing several implementations, we found that both UU-AODV and Qolyester (OLSR implementation aiming to become QOLSR, but still without QoS support) are good enough for now. However, we don't give any more details as this is a temporary solution for our testbed until implementations of QoS-aware protocols well suited to wireless ad hoc networks become available. As the Linux kernel is going to be used in nodes, we assure that we have all the elements needed for IP packet switching [12] [13].

4 The QoS in IP routers

The QoS (Quality of Service) can be defined as a guarantee assured by the network of respecting certain maximal or minimal values to certain parameters when switching a packet throughout the network. The main problem associated to the QoS in a protocol stack is that all protocols must be QoS-aware., which is not the case of 802.11b as it has been said above; any wireless network using this technology will never support QoS completely. However, the use of certain technical procedures at the IP level will permit at least that different kinds of traffic could be differentiated and treated as needed.

Typical IP QoS architectures are IntServ and DiffServ. Both are standardized by the IETF [15] [16] [17] [18] but the second one is preferred generally because it scales better. None of them can be directly applied to ad hoc networks because they make some important functional differences between edge nodes and core nodes while in ad hoc networks all nodes have the same functionality; so any of these solutions will have to be adapted.

The QoS at the IP level implies that different communications (in IntServ) or different traffic classes (in DiffServ) can be identified in each router and be treated separately, with different priorities. An important handicap will be that the throughput of wireless links must be estimated in order to perform bandwidth sharing in a fair way, though the throughput may be variable due to the distance between nodes or to the presence of interferences [8]. We have also mentioned that the WiFi technology is not QoS-aware [6]. However, a partial support for QoS may be obtained at the IP level applying QoS-aware IP switching, what permits to give different priorities to different traffic classes. The parameters that can be adjusted for each traffic class are mainly the following ones: throughput, delay and packet-dropping probability. Additionally, traffic shaping functions give us a way to avoid network overload, what permits us to guarantee that the network performance will approach what is expected.

We will be interested in differentiating six types of traffic which are voice (EF), control and interactive terminal sessions (AF1), video (AF2), navigation (AF3), file downloading (AF4), and the rest (BE). Linux kernel 2.4.20 or newer incorporate several queue disciplines, filters and other facilities that permit to

implement a simplified version of DiffServ just by configuring the kernel with the tc utility [12] [13] [14]. Using dsmark, tc_index, gred, htb, red and pfifo in the linux kernel we can implement the following procedure:

1. Packets coming into the network (identified by their source IP address) are marked with a DSCP value corresponding their type. Packet classification will be performed in order to distinguish the six different traffic classes mentioned above. In order to achieve this, several packet fields will be evaluated: transport protocol, IP address, ports, size and even the payload. The main difficulty is to separate voice and small video packets but even if we make some errors considering an small percentage of them as voice packets, the bandwidth will be shared in a way that avoids those errors to be critical. UDP packets smaller than 160 bytes are identified as VoIP. We could get better results analyzing the payload in order to identify packets containing RTP/RTCP with something having a well known audio codec, but the increment in the CPU load of nodes is more significant than the one in the amount of successful identifications. The rest of UDP packages will be considered video, which is obviously a very simplistic approach (traffics as common as DNS or NFS use to be UDP) but useful for us.
2. Once all packets circulating in the network have been marked with a DSCP, all routers will be able to easily separate them as belonging to different traffic classes. Egress queues in all routers will put each type of traffic in a different subqueue. Each of them will have a share of bandwidth, an appropriated queue length, an appropriated queue dropping probability and other parameters like maximum bust traffic, etc. Voice is given priority over all other traffic classes and so the other classes have relative priorities as listed above.
3. When a packet is going to quit the ad hoc network through an Internet gateway, its DSCP will be given an acceptable value for the Internet.

The implementation of this QoS architecture will use a DSMARK queuing discipline with six subclasses. The EF will be managed by a HTB queue with a guaranteed bandwidth equal to 30% of the total (to be measured in the laboratory). AF classes will be implemented with GRED queues and share 45% of available bandwidth. The rest of the bandwidth (25%) would be given to best effort traffics, managed with a FIFO or eventually an SFQ queue. The DSMARK queuing discipline is used as a container of all other classes because it lets subclasses handle the DSCP. The specific configuration of the traffic control block will depend on the results of experiments presented below.

The ingress queue of each node is programmed to switch transparently every packet with the DS field marked with values 0xa0 or 0x60, or coming from another ad hoc node. Other packets coming from client terminals will be analysed in order to mark audio packets with 0xa0 and video packets as 0xa0. Any other packet will be given a DS value of 0x00 and distinguished with their port field in the egress queue.

5 Building the system

This work consists of building a node of the wireless ad hoc network. A number of identical systems should permit to deploy a network for voice and data switching.

These systems will be installed in isolated rural areas where there is no power source, so nodes need to have a solar power subsystem which means that the cost and the size of a system will be proportional to its power consumption. On the other hand, ultra-low consuming embedded computers are extremely expensive and software development for those platforms can be difficult. Each system will be constituted by three main parts: the power subsystem, the computer subsystem and the radio subsystem.

The first one will consist of a 10Wp solar panel, a 3A regulator and a 17Ah battery, calculated for powering a system consuming about 3W 24 hours a day in the worst weather conditions that we can find in EHAS networks, which are deployed in tropical regions. Obviously, solar radiation in Spain is much lower, so during our experiments in Spain we had to recharge the batteries from time to time.

For the computer subsystem, Soekris net4521 embedded computers have been chosen because they meet all the requirements: low power consumption, three expansion slots where we can insert WiFi cards and hardware watchdog. However, less consuming boards like StrongARM platforms will be evaluated in future developments. The hardware is completed with a 32MB SanDisk CompactFlash that will contain the operating system.

Finally, the radio subsystem will include one or two PCMCIA Engenius Senao SL-2511-CD Ext2 200mW WiFi cards which may be connected to pigtails and external antennas if needed. See Figure 1.

The operating system is built using a cross-compiling toolchain, following Karim Yaghmour's guidelines. We could simplify this just by compiling the software in a x86 platform, but this way of software development permits to migrate easily to other hardware platforms. A basic Linux filesystem has been created, with basic devices and directories. Additionally, an appropriated kernel has been cross-compiled, as well as glibc, busybox, hostap, wireless-tools, pcmcia-cs, iproute2, iptables, tinylogin, pump, kernel-aodv, openssl, zlib, openssh, sysklogd, zcip and MobileMesh. Besides configuration files, starting scripts and so forth are manually created. Although the filesystem obtained has a size of 13MB, it can be compressed in less than 5MB. With GRUB as the bootloader and the filesystem configured to be decompressed in RAM, we have obtained an operating system that can live in a small Compact Flash while having all the power of a QoS-aware wireless IP router.

Two or more images of the operating system may be grabbed in the CompactFlash. When the system must be updated, the old image may be preserved until we are completely sure that the new one is stable and works well. By now the update procedure is manual, but an automatic procedure will be developed in the future in order to perform atomic installations where the system can not be blocked by a wrong update. This is very important for avoiding the need of

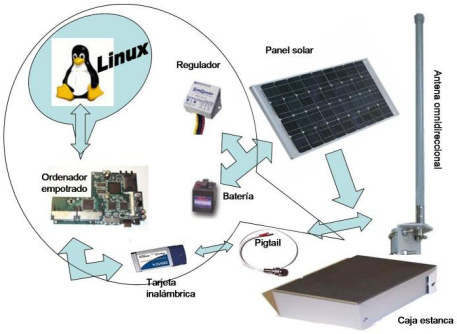

Fig. 1. Parts of a node

local intervention. A software and a hardware watchdog are implemented in the system in order to assure that the system will restart if it gets blocked.

6 Performance evaluation

A solar-powered wireless ad hoc node has been fully designed. A prototype has been developed with its hardware, operating system, auto-configuration procedures and traffic control politics. Now it is time to find out if the design serves for deploying auto-configurable ad hoc networks that enable transport VoIP and other data with acceptable quality.

The first consisted on testing that the embedded computer with two WiFi cards could be powered in a sustainable basis by the solar subsystem, and that it was able to do the routing and packet switching while keeping low the CPU load. Thereafter, it was time to prove that the designed node is autonomous, auto-configurable and able to switch voice and data correctly.

There are three main tests that must be run in order to validate this work:

1. Nodes must detect each other automatically and adapt their routing table to any changes in the topology. In particular, the presence of a gateway node must be recognized as such by the others, which must add a default route through it.
2. Any WiFi or Ethernet client terminal correctly configured and having a DHCP client must be provided with a private IP address from any of the nodes to be able to access the other nodes and eventually the Internet.
3. When several connections are overloading the network, voice traffic must be protected from the others.

We deployed our testbed in the laboratory with four nodes in cascade. Our testbed can operate in a reduced space because the WiFi cards chosen lack of any kind of internal dipole antennas. If we don't connect external antennas to them, their range of communication is kept below one meter. We configured the first card of each node in WiFi channel 1 and the second in channel 6 or superior

so that both cards will not interfere. The first WiFi card is used to access the ad hoc network while the second and the Ethernet ports are for client terminals.

The first test was straight forward once we chose the UU-AODV, developed by the University of Upsala. It is well understood that this routing protocol is not QoS aware, so it is specially important that two nodes associate only if they can maintain a high quality wireless link between them. That can be assured by fixing a high association threshold (i.e. -70 dBm or superior) to the first wireless card of each node.

The second test was also simple and the success was almost assured, as the configuration of DHCP and NAT is just a matter of system administration.

Once we had four nodes forming an ad hoc network, one of which was connected to the Internet through its first Ethernet port, and all of them detecting the network topology properly and being able to give access to client terminals, we associated a computer to node D and connected two IP telephones to nodes A and D through their second Ethernet port, as seen in Figure 2.

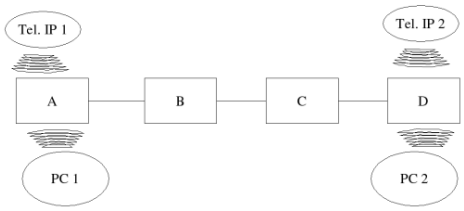

Fig. 2. Network topology for test 3

The IP phones are configured to use the value 0xa0 for the DS field.
We will conduct the following tests:

1. Measure of the total available bandwidth.
2. Objective tests: by using a traffic generator as iperf, we simulate a continuous TCP-BE traffic flow between A and D. This tool permits us to measure bandwidth, delay and jitter at arrival. 20 seconds later a new traffic flow simulating a voice conversation is injected. 20 seconds later traffic simulating video is introduced, and finally other UDP traffic flows follow. These tests are performed with and without traffic control.
3. Subjective tests: the same tests are run but voice traffic is not simulated. A real voice communication happens between the two IP phones that is perturbed with different concurrent traffic flows.

7 Results

The first test shows a total throughput of about 590 Kbps in absence of traffic control. That is completely normal because the links have been fixed to 1Mbps

in order to simulate the worst real conditions. Based on this first result and after evaluating different alternatives, the traffic will be controlled according to the following simplified configuration (we distinguish only among voice, video and the rest in this case):

```
#!/bin/bash
IFACE=$1
tc qdisc add dev $IFACE handle 1:0 root dsmark indices 64 set_tc_index default_index 1
tc filter add dev $IFACE parent 1:0 protocol ip prio 4 tcindex mask 0xfc shift 2
tc qdisc add dev $IFACE handle 2: parent 1:1 htb
tc class add dev $IFACE parent 2: classid 2:1 htb rate 500Kbit ceil 550Kbit
tc class add dev $IFACE parent 2:1 classid 2:10 htb rate 300Kbit ceil 500Kbit prio 1
tc class add dev $IFACE parent 2:1 classid 2:20 htb rate 150Kbit prio 2
tc class add dev $IFACE parent 2:1 classid 2:40 htb rate 64Kbit prio 3
tc qdisc add dev $IFACE parent 2:10 sfq perturb 10
tc qdisc add dev $IFACE parent 2:20 sfq perturb 10
tc qdisc add dev $IFACE parent 2:40 red limit 64KB min 5KB max 15KB burst 8 avpkt 1000 \
bandwidth 64Kbit probability 0.1
tc filter add dev $IFACE parent 2: protocol ip prio 1 handle 0x28 tcindex classid 2:10 pass_on
tc filter add dev $IFACE parent 2: protocol ip prio 1 handle 0x18 tcindex classid 2:20 pass_on
tc filter add dev $IFACE parent 2: protocol ip prio 2 u32 match ip tos 0x00 0x00 flowid 2:40
```

It is important to observe that although other possible configurations have been evaluated (with PRIO instead of HTB, etc.) but this configuration has resulted to be the best for our needs. The DSMARK queuing discipline is used only for giving subclasses access to the DSCP but actually all traffic goes to a HTB queuing discipline, whose main mission is to manage bandwidth sharing. Here in there are two possible approaches: HTB subclasses might share their dedicated bandwidth in excess of what they need, or they can keep their dedicated bandwidth empty when they don't use it. HTB main class is supposed to manage bandwidth sharing so that each traffic class always finds its dedicated bandwidth when necessary but lends it when idle. We have tested both possibilities. The configuration for bandwidth among classes is similar to the previous one but giving a 'ceiling' of 500 kbps to classes 2:20 and 2:40, and a limit of 500 kbps to the RED qdisc.

The objective tests without traffic control show that traffic flows perturb enormously to each other, as we can see in Figures **??** 4.

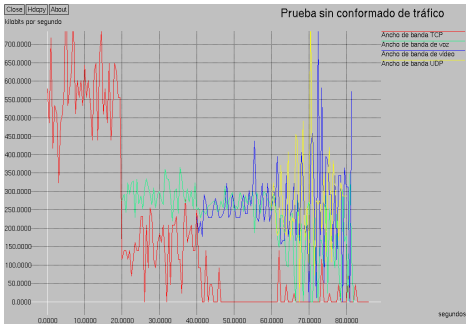

Fig. 3. Bandwidth without traffic control

We can see how the bandwidth utilization of the BE flow drops immediately when the voice appears. That is expected, as TCP uses its congestion avoidance

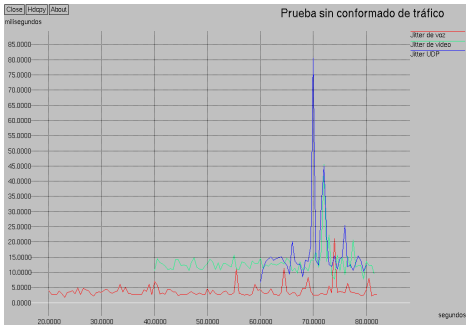

Fig. 4. Jitter without traffic control

mechanism for adapting itself to available resources. When video appears, the voice keeps using the same bandwidth just because the network is not yet saturated and because UDP has no congestion avoidance. At the same time, we can see that both UDP flows have starved the bandwidth and the TCP traffic has almost disappeared. As the forth traffic flow appears, all the flows are perturbed, with a very important visible impact on the quality of service.

When applying traffic control, the result of the same tests is as shown in Figures 5 and 6. We can see that each traffic flow is preserved. The TCP flows are more sensitive than the UDP flows as expected. We can also see that voice is not perturbed by TCP traffic, but video perturbs voice a little bit. However, we have achieved a situation where different traffic flows are somehow preserved from each other's influence.

When the traffic control block permits sharing of available bandwidth among classes, we can see that the result is much worse for the bandwidth (Figure 7) and similar for the jitter (Figure 8). This result shows that, at least for the voice traffic, we must avoid other classes to borrow its available bandwidth. Even more: when dedicated bandwidth is restricted but adjusted to the exact needs of voice traffic, the result is similar to the case of bandwidth sharing. We get the result showed in Figure 5 only if we limit the total bandwidth to at most 85% of the the maximum throughput obtained in the first test. This way we are wasting bandwidth significantly, but the sacrifice is necessary in order to assure the best QoS for telephony.

In both Figures 4 and **??**, and also in Figure 8, we can see that the jitter is strongly modified in voice communications independently we have or not applied traffic control. That means that voice quality can be maintained by applying traffic control, but packet delay will augment significantly in both cases when the network is overloaded.

Subjective tests run without traffic control showed that VoIP:

- has a very good quality when the network is dedicated,
- is good enough but with some audible 'clicks' whenever sharing the network with TCP flows, and

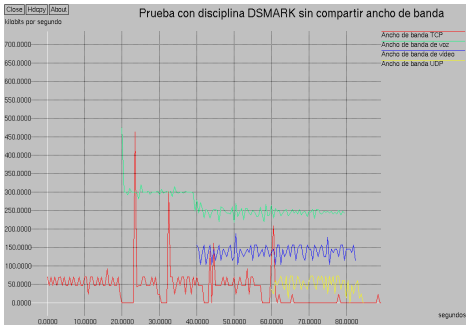

Fig. 5. Bandwidth with traffic control

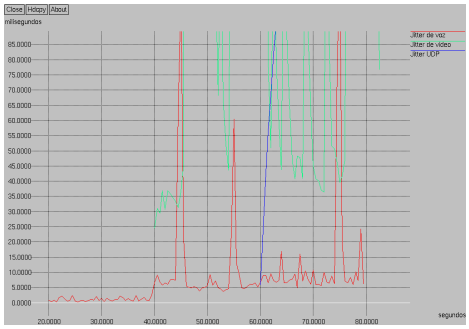

Fig. 6. Jitter with traffic control

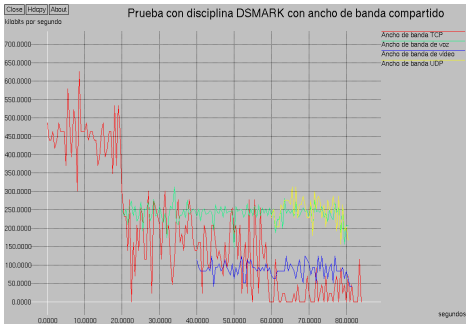

Fig. 7. Bandwidth with traffic control (BW shared)

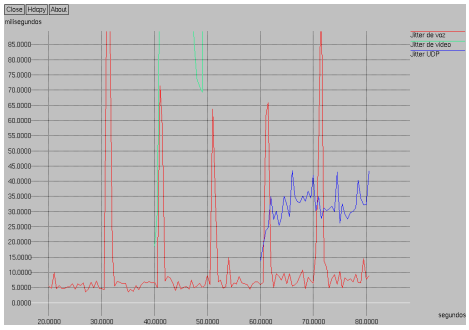

Fig. 8. Jitter with traffic control (BW shared)

– is bad, with long delays (seconds) but it offers almost acceptable audio quality when the network is overloaded.

When applying traffic control, the quality of voice communications remains good and stable in presence of TCP or intense UDP traffic. Only real network overload causes longer delays of voice packets, but with good audio quality anyways. Actually this result is end-system dependent: VoIP terminals that do much less buffering would have much lower delays in overloaded networks at the price of perceiving a worse audio quality due to packets lost.

8 Discussion and conclusions

The aim of this research was to develop and evaluate a real wireless ad hoc network with address and route auto configuration, and partial QoS support. The network developed should be easy to deploy and be able to switch voice and data traffic taking into account the quality of service required for each differentiated service. The network must be deployable in isolated rural areas for giving access to the Internet and the telephonic network.

The process of evaluation itself has been presented above. We have been able to deploy an ad hoc network easily and make it work with no manual configuration of IP addresses and routes.

All software and hardware products have been verified and routing protocols have been compared. The performance evaluation test showed that our network deals well with differentiated services, giving priority to real-time traffic classes and specially protecting voice flows from data traffic load. However, these results don't mean that voice communications can be of the same quality in any installation, and therefore, it is important to consider the limitations imposed by the WiFi technology, which is not QoS-aware.

The network developed will permit to do some real installations in isolated rural areas of developing countries, and will serve at the same time for a deeper evaluation of all software and hardware technologies that are being used and the interactions among them. Nevertheless, there are many aspects in the prototype

that must be optimized: power consumption may be much lower using a more efficient computer architecture, the IP auto-configuration procedure must be consolidated, a QoS-aware ad hoc routing protocol must be used, and traffic control architecture should be optimized in order to avoid bandwidth waste and extremely high delays and jitters when the network is overloaded.

In the next future we intend to develop a second version of the hardware, based on the StrongARM architecture, with IPv6 for an easier auto-configuration and more appropriated routing protocols as ETX or future QoS-aware versions of Qolyester. Additionally, traffic control will be studied more in depth in order to obtain a more efficient model.

References

1. EHAS Foundation: EHAS: Enlace Hispano-Americano de la Salud. Website created in November 2003. http://www.ehas.org.
2. A. Martinez, V. Villarroel, J. Seoane, F. del Pozo: Rural Telemedicine for Primary Healthcare in Developing Countries. IEEE Technology and Society Magazine, Volume 23, Number 2. Summer 2004.
3. A. Martinez, V. Villarroel, J. Seoane, F. del Pozo: A study of a rural telemedicine system in the Amazon region of Peru. Journal of Telemedicine and Telecare, Volume 10, Number 4, 2004.
4. S. J. De Couto: Performance of Multihop Wireless Networks: Shortest Path is Not Enough. In Proc. of First Workshop on Hot Topics in Networking (HotNets-I), Princeton, New Jersey, October 2002.
5. D.S.J. De Couto, D. Aguayo, J. Bicket, R. Morris: A high-throughput path metric for multi-hop wireless routing. In Proc. ACM/IEEE Mobicom '03. September 2003.
6. S. Xu, T. Saadawi: Revealing the problems with 802.11 medium access control protocol in multi-hop wireless ad hoc networks. Computer Networks 38. Ed. Elsevier Science, 2002.
7. P. Mohapatra, J. Li, C. Gui: QoS in Mobile Ad Hoc Networks. IEEE Wireless Communications, June 2003.
8. R.G. Mukhtar, S.V. Hanly, L.L.H. Andrew: Efficient Internet Traffic Delivery over Wireless Networks. IEEE Communications Magazine, December 2003.
9. Y. Ge, T. Kuntz, L. Lamont: Quality of Service Routing in Ad-Hoc Networks Using OLSR. In Proc. of the 36th Hawaii International Conference on System Sciences (HICSS'03). Hawaii, 2003.
10. E. Perkins, E. M. Royer, S.R.Das, M. K. Marina: Performance Comparison of Two On-Demand Routing Protocols for Ad Hoc Networks. IEEE Personal Communciations, February 2001.
11. S. Cheshire, B. Aboba, E. Guttman: Dynamic Configuration of Link-Local IPv4 Addresses. IETF Draft v10, September 2003.
12. S. Radhakrishnan: Linux - Advanced Networking Overview. ITTC, Univ. Kansas. August 1999. URL: http://qos.ittc.ukans.edu/howto.pdf .
13. B. Hubert: Linux Advanced Routing and Traffic Control. Julio de 2003. URL: http://www.lartc.org .
14. W. Almesberger, J. H. Salim, and A. Kuznetsov: Differentiated Services on Linux. June 1999.
15. S. Blake, D. Black, M. Carlson, E. Davies, Zh. Wang, W. Weiss: RFC2475: An architecture for Differentiated Services. IETF RFC, 1998.

16. K. Nichols, S. Blake, F. Baker, D. Black: RFC2474: Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers. IETF RFC, 1998.
17. J. Heinanen, F. Baker, W. Weiss, J. Wroclawski: RFC2597: Assured Forwarding PHB group. IETF RFC, 1999.
18. Jacobson, K. Nichols, K. Poduri: RFC 2598: An Expedited Forwarding PHB. IETF RFC, 1999.
19. Aguayo, J. Bicket, S. Biswas, D.S.J. de Couto, R. Morris: MIT's RoofNET Implementation. PDOS-LCS. Massachussets Institute of Technology, August 2003.