Active measurement tool for the EuQoS project

*René Serral-Gracià Albert Cabellos-Aparicio Hector Julian-Bertomeu
Jordi Domingo-Pascual
E-mail: {rserral,acabello,bertomeu,jordid}@ac.upc.edu
Universitat Politècnica de Catalunya (UPC),
Departament d'Arquitectura de Computadors, Spain.

Abstract

Network deployment spreads everywhere, such expansion unveils the need to discover the network's real performance. This paper presents NetMeter, a tool designed for active network testing with some of the advantages of passive traffic analysis. NetMeter permits to easily manage such tests, also gives a structured framework for representing the most common measurement parameters such as packet losses or one way delays. NetMeter applicability ranges from precise interdomain Quality of Service studies to deep wireless handover analysis.

1 Introduction

On each production network there is a need to monitor any important event. Typically such events include security, network services or system maintenance. But, as network usage spread on a day to day basis, the need of modelling the actual network's performance is important, that's because users always ask for more and better services, which in turn require special capabilities on the network to work.

Usually this concept of performance is mistaken as network bandwidth, but there are many situations where speed is not the main issue. Environments where Quality of Service (QoS) is needed require tight network constrains such as low packet delay, constant delay variation or few packet losses, all of them taking precedence over bandwidth.

Scenarios where is possible to find this network constrains include applications like VoIP, videoconference or, in general, any real-time application. On average, those technologies don't have big demands on actual bandwidth, but force other constrains as stated before.

On those scenarios there is a need of some tool capable of detecting such problems over the network. The tool we present on this paper is NetMeter [1].

NetMeter is a tool used to actively testing any network between two end hosts. The idea behind that is to provide a tool capable of generating controlled data flows between two stations. With the extracted information of those flows, there are computed the most typical QoS parameters [2, 3]. NetMeter, unlike other similar tools, is designed for accuracy and, more important, interacts with the scenario as transparently as possible.

Jointly with the above cases, there are also increasing need to evaluate and tune different Quality of Service parameters over interdomain scenarios. This has been the major focus of several projects on the past years [4–6] and continues with others like [7]. This paper presents a typical Interdomain scenario where NetMeter can be used to analyse the end-to-end Quality of Service.

This paper is divided into the following sections, first an overview of NetMeter's capabilities are introduced along with all the graphical representation possibilities of the tool, later this paper discuses the usability of this tool on a real research project such as EuQoS [7], finally there is the explanation of a real scenario where NetMeter is used for analysing all the implications of a Moblie IPv6 handover over an end to end connection, more important with the aim to discuss its impact on the user's perception (QoS) due to such discontinuity on the data flows.

2 NetMeter

This section describes the key functionalities of NetMeter, this chapter doesn't pretend to give deep im-

*This work was partially funded by IST under contract 6FP-004503 (*IST-EuQoS*), MCyT (Spanish Ministry of Science and Technology) under contract FEDER-TIC2002-04531-C04-02 and the CIRIT (Catalan Research Council) under contract 2001-SGR00226.

plementation details, but to highlight the main capabilities, which make of this tool a good candidate for analysing any network topologies with an active measurement tool.

Is important to remark that lately, besides the active testing approach, NetMeter has been upgraded with some passive capture capabilities to permit more detailed analysis on a per hop basis, combined with the end to end approach of the active testing.

2.1 Tool's overview

NetMeter is a front-end for various applications. The basic scheme of NetMeter's operation can be seen on Figure 1. The figure shows three main components needed for NetMeter to operate, first there is the *Test Manager* which is usually outside of the network under test, its main goal is to manage and distribute all the tests that have to be done. The second component are the end points of the tests. Those are computers on the edges of the network/s under test, they need connection through a management interface with the *Test Manager*. The last component is the actual network under test.

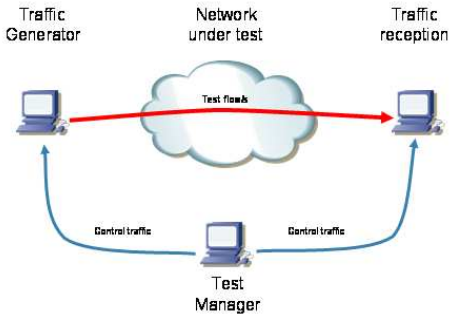

Figure 1. NetMeter's basic scenario

The tests are managed with the *Test Manager* which will send the instructions to the end points through a secure connection, once the end points have all the needed information the traffic flows going from source to destination will be monitored. When the tests finish the *Test Manager* will gather all the needed information of the flows

Usually all the network remote test applications have the drawback that are intrusive with the actual tests. NetMeter changes this behaviour because the control machine can specify different hosts for the tests, and issue the tests with different physical interfaces used for management, thus without interfering with the data being monitored.

NetMeter is capable of computing the different network parameters needed to outline its performance. NetMeter uses Active Traffic generation tools for computing all sorts of delays, delay variations and packet losses. More important, is also capable of extracting statistical information of the network tests and plot graphical representations of the results.

Then, one of the strongest points of this application are the graphical representation capabilities, for example displaying overall results with per packet delay or plotting accumulated distributions for IP Delay Variation (as demonstrated on chapter 4) is an easy task. Another added plus is that NetMeter has a good scripting system for helping on the reproduction/batching of networking tests. This permits to repeat tests with different network parameters to evaluate their differences.

The current version is programmed with TCL/TK, published under the GPL license, for the graphical interface and the basic data manipulation. Also the lower level parts such as graphical test representations are developed with C with the help GD library.

The tool's first goal is to automate the long procedure of prepare and manage networking tests, closely followed by the posterior graphical representation of the results.

This automation leads to a structured methodology for the tests execution, when at the same time giving a solid base to plan all the set of tests which form any traffic analysis project.

For accomplishing all the above functionalities, NetMeter embeds various traffic generation and graphical representation tools that form the whole application. The currently supported tools can be divided on two different parts:

1. *Traffic generation tools:* where the tools being used are:
   - *MGen:* [8] is a tool that permits to generate any amount of controlled UDP packets. This traffic generation can be either periodic or poissonian.
   - *Netperf:* [9] this tool generates elastic TCP flows which goal is to fill up the link capacity and calculate the bandwidth over time. The good point of this measurement tool is that permits to specify the size of the return packets, simulating this way a bidirectional TCP flow.
2. *Graphical representation:* the tools used on this part are specifically designed for NetMeter and are included on the package. More detail on

graphical representation can be found later on this chapter.

2.2 NetMeter measurement parameters

NetMeter handles the following parameters of QoS defined by *IPPM*:

- *One-Way Delay:* delay of each packet from its generation on the source machine until its reception on the destination.
- *Inter Packet Delay Variation:* is the difference between the one-way-delay of pairs of packets, usually the taken pairs are consecutive packets of the stream. Sometimes, this parameter is also named Jitter.
- *Packet Losses:* this parameter specifies the total amount of packet losses over the test.
- *Link Bandwidth Capacity:* measures the maximum usable bandwidth between both end points.

All these parameters are studied per flow independently, but, for being able to compare different results, NetMeter provides a highly versatile framework to represent all the needed results on the same graphical file, as described in the next section.

2.3 Analysis and Graphical Representation

Once the data is stored on the Test Manager there are several possible graphical representations, the most important are regarding the *Graphical Distributions*, that represent detailed flow/flows evolution:

1. *One-Way Delay:* delay of each packet to reach its destination. X-axis holds the sequence number and Y-axis the delay expressed in milliseconds.
2. *One-Way Delay Distribution:* percentage of the distribution of packets with a given delay. X-Axis has the delay and the Y axis the percentage of packets with that delay threshold.
3. *IP Delay Variation:* delay variation (jitter) among the current packet and the mean.
4. *IP Delay Variation Distribution:* distribution of the above representation.
5. *One-Way Average Distribution:* the same as one-way delay but pondered with the test's mean.

The graphical distributions are indeed very useful to analyse the flow variations over time, or to gather its information in percentages. Are even useful when comparing different flows with similar properties. But when dealing with other data such as packet losses or average delays per test, there is a need of other kind of representation, this situations, where each test gives only a numerical result has to be treated differently, that is to represent the different values of the various tests on the same plot to compare their differences. NetMeter is capable of controlling variables such as:

1. *Average Delay:* Value of test's average delay.
2. *Delay Variation:* Delay variation's value per test.
3. *Flow Relative Start:* Flow starting time, this value can be useful when different flows start at different times and this means different network conditions.
4. *Flow Relative Stop:* Time when the flow stops.
5. *IP Packet Size:* Packet size per tests, usually is useful to compare with end to end delays.
6. *Maximum Delay:* Maximum delay per test.
7. *Minimum Delay:* Minimun delay.
8. *Packets dropped:* This is the MGen notation for packet losses.
9. *Received packets:* Integer definning the packets received on the tests.
10. *Received data rate (kbps):* Rate at UDP level expressed in Kilobit per second.
11. *Received rate (packets/second):* Same as above but indicating packet per second.
12. *Send Rate (packets/second):* There are times when the received rate differs from the sent.
13. *UDP Packet Size:* UDP level packet size that is different from the link level size.

With such versatility that any of the above variables can form the X or the Y-axis seamlessly, and thus interpret any arbritrary number of tests on the same plot.

2.4 Passive analysis

As stated above, NetMeter studies the network performance by actively generating the desired traffic patterns for later analysis. For enhancing this paradigm, NetMeter provides an hybrid approach to networking

measurements by providing support to passive captures.

The idea behind this scheme is to provide intermediate results to help on the analysis of events that may occur on any hop of the network. An application of this idea can be found on chapter 3 and 4.

NetMeter reaches this goal by capturing through Ethereal [10] the desired traffic, processing it offline and finally using it to compare with the end to end data. Each partial result gives an idea of the flow's status on each domain.

3 Interdomain Analysis

As stated before, NetMeter includes some interesting capabilities only found on passive monitoring tools. This section comments the use of those capabilities describing the analysis to be done in the framework of the EuQoS project. This project is being conducted by several european universities along with some corporations whose goal is to design a framework capable of monitoring, configuring and interact over an interdomain Quality of Service network.

Nowadays a typical end to end connection between two arbritrary hosts on the Internet is formed by the sender's access network, the core network, composed at the same time by one or several smaller networks, and finally the destination's access network, possibly of different tecnology than sender's.

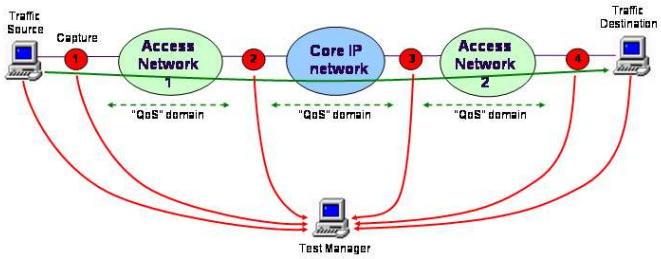

Figure 2. Interdomain analysis

On such scenario the core network is pretty homogeneous, but on the other hand the access network is highly variable, either on technology as in capabilities. To guarantee the QoS on such environment is not an easy task, but at the same time, verification of whether or not the service is properly delivered is also very hard to accomplish.

Over all this framework, NetMeter fits when network's QoS verification is needed. The tool is able to generate controlled traffic through the path under test, capture it at destination's machine and, more important in this situation, capture at the intermediate ingress and egress points of the access technologies (refer to figure 2 for an example) by capturing the generated traffic. NetMeter processes the results by gathering offline all the captures and converting them to mgen format, thus permitting its graphical representation. Those intermediate representations permit to track the flow evolution in terms of one way delay or IPDV and detect which segment of the network is responsible of the lack of QoS, or on the other side, to verify that the service provision is correct.

With the above scenario, more capture points imply higher precision when analysing the results. The simplest scenario in this situation is the standard MGen test where only the sender and the destination capture the traffic. But even on this case is possible to take advantage of passive monitoring. MGen sets and gets the timestamps at application's level, what involves operating system overhead. But when capturing with any passive tool, the timestamps are taken from the kernel, more precisely, at the lowest level available, which means, just before the packet is sent to the network or just after it arrives at the destination's network card. With this approach, to exclude almost all the operating system variabilities is a possible task.

The opposite case to the one described above is when there are capture points at each hop, giving NetMeter more information which can be useful to spot conflictive points along the path.

4 WLAN Access Network Study

Wireless (IEEE 802.11 [11]) is part of the technologies under test in the EuQoS project. This technology, on the last years, has improved and made cheaper. In current Internet status, an user can connect trough a wireless link, but he can not move, change his point of attachment and maintain his network connections. For that reason IETF designed Mobile IP, which jointly with WLAN provides this capability to the Internet. In this paper we use NetMeter to do a complete study of the most critical part of this technology: the handover. During this phase, the mobile node (MN) is not able to send or receive data, and some packets may be lost or delayed (due to intermediate buffers).

IEEE 802.11 is based on a cellular architecture,

where system is divided into cells and each cell is managed by an Access Point (AP). When a MN decides to associate to an AP, it searches for a new one (scan); the MN sends "Probe Request" frames in different channels, expecting to receive "Probe Response" sent by APs. Once a MN has found an AP, it will go through the "Authentication Process", which is the interchange of security information. When the MN is authenticated, it will start the "Association Process". Only after this phase is completed, the MN is able to transmit and receive data frames.

Mobile IP was designed in two versions, Mobile IPv4 [12] and Mobile IPv6 (MIPv6) [13]. The main goal of the protocol is to allow MNs to change its point of attachment to the Internet while maintaining its network connections. This is accomplished by keeping a fixed IP address on the MN (Home Address or HAd). This address is unique, and, when the mobile node is connected to a foreign network (not its usual network) it uses a temporal address (Care-of Address or CoA) to communicate, however it is still reachable through it's HAd (using tunnels or with special options in the IPv6 header). A special entity called Home Agent (HA) is needed to manage mobility. The MN must register (using special MIPv6 messages: "Binding Update" and "Binding Acknowledgement") its temporal CoA to its HA. The MN can also register its CoA to Correspondent Nodes (any node on the Internet whith active connections with the MN) in order to allow direct communication with them, otherwise, communication must be routed through the HA.

NetMeter is used to study the WLAN/ 801.11/ MIPv6 handover. Our goal is to study the handover in a real testbed using active measurements. We aim to study the effects of the handover on traffic sent or received by applications, studying packet losses, one-way delays, IPDV and QoS variations.

Several papers focus on the same topic, [14] uses a mathematical model to study the handover latency but it does not take into account the wireless handover, [15] studies the Mobile IPv6 (and others) handover with a simulator, [16] makes an empirical analysis of the 802.11 handover, and, finally, [17] studies the WLAN/Mobile IPv6 handover in a real testbed proposing a new algorithm to improve the handover latency. Our paper goes further, analyzing bottlenecks, comparing the layer 2 and layer 3 handover and studying the effects suffered by the applications.

4.1 Measurement Scenario

Testbed's main goal is to compute the Mobile Node handover latencies. Figure 3 shows the detailed testbed. We have two "Access Points", each one with two wireless cards, one for communicating with the MN and the other one to monitor the wireless beacons. The MN has one wireless card, and will switch from one AP to the other by regular movement between them.

The CN is the node connected to the MN; packets will travel from the CN to the MN and vice versa. For permitting Mobile IPv6 configuration there is also a HA as shown in the figure.

Regarding the hardware and software configuration, our testbed uses the Atheros (for APs) and Cisco Aironet (for MN) chipsets on the wireless cards. The Mobile IPv6 implementation needed for this testbed is MIPL 1.1 [18] and the software for all the machines is Linux Debian Sid Distribution with Kernel 2.4.26.

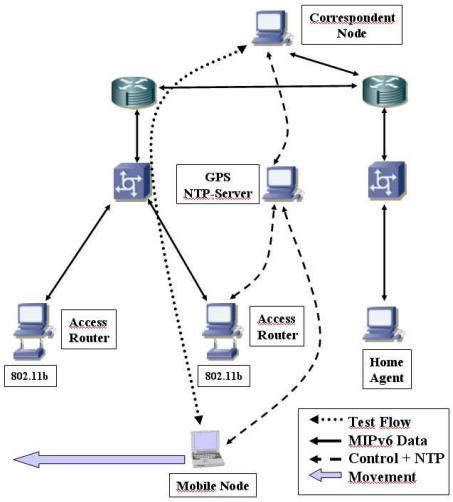

Figure 3. Measurement Scenario

Our testbed has a parallel management network used to manage the testbed without interfering with our measurements (control network). This network is also used for synchronization purposes, for this we use four NTP (Network Time Portocol) sources [19], two of them belonging to a private network (Stratum 1 servers connected directly to a GPS source each). The other two sources are on the outside network and are as far as 3 hops away from the testbed. The NTP statistics show that, with this setup, we obtain 1ms of measurement accuracy.

In order to confirm our measurement accuracy, which is a critical aspect on such small scenarios, we sent several broadcast ARP packets on our measurement network. Those packets were captured at all the machines involved in our tests, then, the timestamps were compared. The maximum difference among these

timestamps agreed with the NTP statistics.

Summarizing, in the testbed the test flows and the MIPv6 data travel on the network under tests, that is, wireless link from the MN to the AP, and through ethernet link from the AP to the CN. Meanwhile, the test's control data and the NTP synchronization go through the control network, which is a parallel ethernet network, not detailed on the figure for the sake of simplicity.

4.2 Methodology

We use NetMeter to perform the handover study. As stated before, we propose a mix of active and passive measurements, we generate an active synthetic flow that travels end-to-end (depending on the tests the source can be the MN or the CN). This flow is captured at the Access Point (using Ethereal [10]) and also at the destination (using NetMeter). This approach permits to calculate, end-to-end parameters but also partial delays. That's because the stored timestamps passed to the MGen file are taken from the monitoring machine (which is the actual Access Point on the testbed).

With this method, we can calculate end-to-end OWD, IPDV, packet loss, but also, using the packets captured at the Access Point, differentiate the wireless part from the wired one. We aim to isolate all the handover effects, without taking into account the other parts of the tests.

This methodology, is also applicable to study other handovers, such as Mobile IPv4, Fast Handovers [20], Hierarchical Mobile IPv6 [21] or 802.11 handover improvements.

4.3 Tests

Our goal is to analyze the handover; for this purpose, we must build up a good set of tests. We ran a set of 16 tests, each 5 minutes long, from where extracted a set of 63 valid handovers. The tests where splitted as follows: half of them the MN was sending traffic to the Mobile Node, while the other half was on the opposite direction. Moreover, each direction of the tests where split as follows:

- *64Kbps Traffic*: With this flow, we try to recreate the VoIP traffic over UDP/IPv6. The rate is 34 packet per seconds, with 252 bytes of payload as stated on [22].
- *1Mbps Traffic*: This UDP flow has a rate of 94 packets per second and a payload of 1300 bytes. We use a flow with a higher size and rate because VoIP have a very low one, and we also aim to study the impact of a different bandwith in the handover.

During the 5 minutes of each test, the handovers were "forced" attenuating the signal sent by the AP. The MN realizes this (it detects that the signal quality is poor) and tries to search for a new AP. In our testbed we do not have external interferences, and thus, the MN changes to the other AP.

4.4 Results

Here the results about our handover analysis obtained with NetMeter application are discussed. We aim to study the traffic's impact on user level, the most important parameters are: packet losses, delays and IPDV.

4.4.1 Packet Loss

Packet loss is the most important metric to measure the effect of a handover at application level; basically, a handover losses packets. Computing this parameter is straight forward problem having the first and the last packet (and its sequence number), which are provided by the capture on the access point.

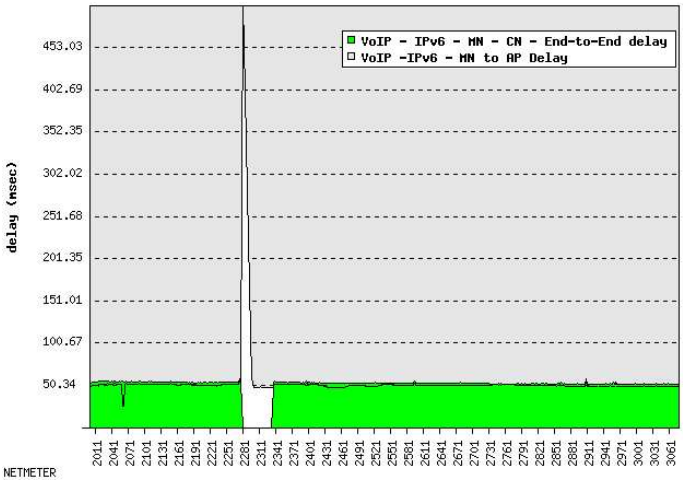

Figure 4. Handover MN-CN VoIP traffic

Figure 4 and 5 have the packet's sequence number on the x-axis and the delay (in milliseconds) on the y-axis. The first figure, represents a handover where the traffic was sent from the MN to the CN, and the other one on the opposite flow direction. This figures show clearly the handover gap, we can see how the traffic stops flowing when the handover starts, and continue flowing when it ends. During this handover latency,

any packet is sent or received by applications, thus, these packets are lost.

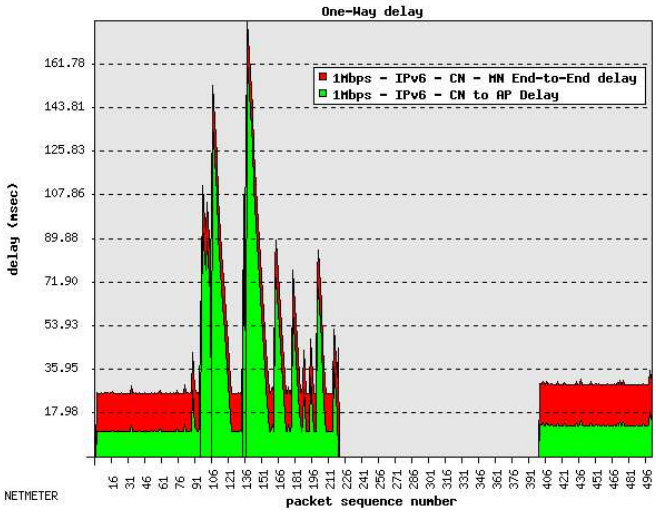

Figure 5. Handover CN-MN 1Mbps traffic

Our results show that 65 packets (in average) are lost when VoIP traffic is flowing, and 207 packets when 1 Mbps traffic is being sent. There are no differences regarding packet loss when the source of the traffic is the MN or when it is the CN. The number of packet loss is proportional to the rate and to the handover latency. [23] states that a 802.11/IPv6/MIPv6 handover takes 2.1 seconds in average. We can conclude that the packet loss of a handover is the rate multiplied by the handover latency.

If we take into consideration in our handover study, not only our end-to-end measurement, but also the packets captured on the Access Point we can see that the handover has a different behavior when then MN is sending the traffic flow or when the CN is sending it. Figures 4 and 5 show also the delay of the packets captured on the Access Point, those packets are labeled as "MN to AP delay" and "CN to AP delay" respectively. Those packet, arrive to the Access Point, but some of them are lost at that point.

When the MN is sending the traffic flow, and the handover begins, the wireless drivers begin to buffer those packets; it is searching for a new AP and it is not able to send data. Those buffered packets, are sent as fast as possible when the MN regains Layer 2 connectivity. Unfortunately, they have an incorrect Access Router MAC address destination (the MN has changed its Access Router) and they are lost. This effect can be clearly seen on figure 5; packets buffered are sent with a higher delay and at a faster ratio, the peak shows this behavior. The highest delay is exactly the duration of the 802.11 handover, 260ms in average, as stated in [23].

This buffer may seem useless, but in fact, is very useful in case of a 802.11 handover. In this case, the MN is not changing its Access Router, it is just changing its Access Point and it doesn't need to change its default router, or announce a new location to its Home Agent. The buffered packet, will be sent after the handover is finished, with a higher delay, but it won't be lost. When the traffic flows from the CN to the MN, the behavior is straight forward. The CN will send the packets to the incorrect Access Router until the MN announces its new location. This information is sent using the Binding Update message, which also indicates the end of the 802.11/IPv6/MIPv6 handover.

4.4.2 One Way Delay and Inter Packet Delay Variation

The other important parameters to study the handovers are One Way Delay (OWD) and Inter Packet Delay Variation (IPDV). Those metrics, jointly with Packet Losses are necessary to highlight the level of provided QoS. The main goal is to see if the QoS parameters are kept under those handover. This is accomplished by taking several seconds worth of packets before the handover and calculate the OWD, the IPDV and the same after it. With these values, we can study which are de QoS fluctuations caused by a handover, specially, when the wireless signal is becoming low (before the handover) and some packets may be delayed. The study is concentrated in the figures showed in the previous section.

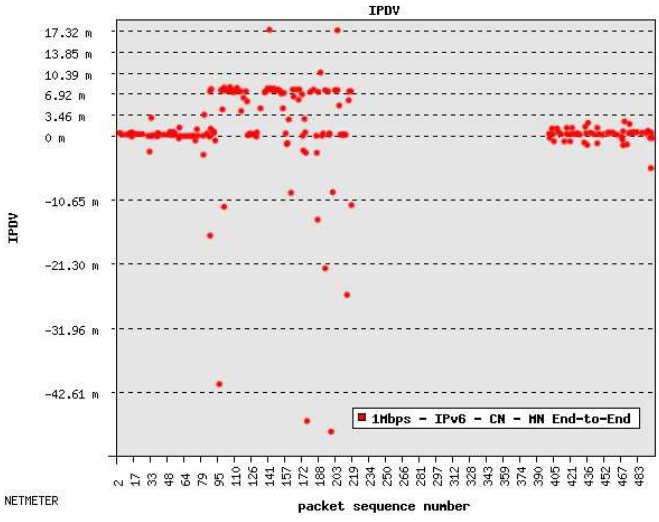

Figure 6. IPDV CN-MN 1Mbps handover

Figure 4 and figure 5 show two handovers, one with

64Kbps traffic (VoIP) and the other with 1Mbps traffic, respectively. As we can see, figure 4 doesn't show any QoS variation at the prior or later moments of the handovers, however, figure 5 displays that, when the Access Point signal's quality is becoming low (before the handover), some packets are being sent with a higher delay. The overall results for the displayed handovers can be seen on table 1.

Table 1. OWD and IPDV

	OWD (ms)		**IPDV** (ms)	
	Before	**After**	**Before**	**After**
VoIP	49.68	48.94	6.92	7.08
1Mbps	46.88	29.88	153.77	6.39

The results are very clear, when VoIP traffic is being sent, the loss of connectivity before the handover due to a low wireless signal quality doesn't affect the delay of the packets, the same holds true for the system recovery once the handover is finished. Another result though, is the case when the link is more overloaded (1Mbps). We can see clearly the increment of the delay of the flow; the main reason is that the signal quality is low, and there are a lot of retransmissions. However, the system's recovery is pretty fast and reliable.

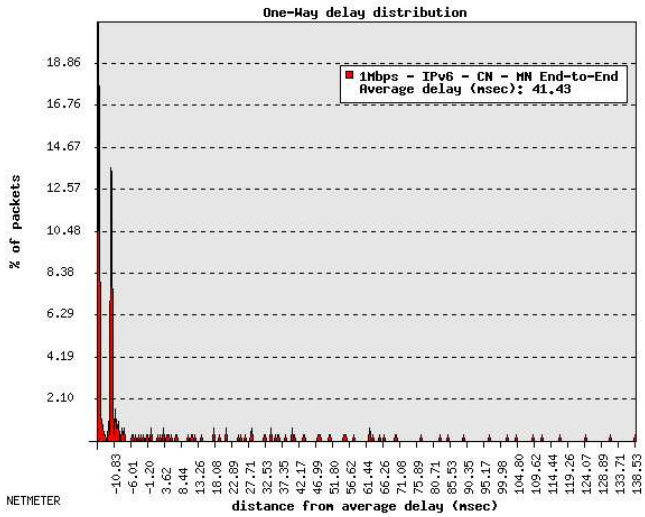

Figure 7. OWDD CN-MN 1Mbps handover

Figure 6 show the IPDV of the 1Mbps handover. We can clearly see how the packet delay vary remarkably moments before the handover starts, but, when the handover is finished, the variation is very low. In figure 7 we can see the OWD distribution of the full handover, the figure shows that 25% of the packets have a close delay to the average, but the rest have very dispersed values.

5 Summary and Future Work

This paper revises NetMeter, an active measurement tool used on the EuQoS project, this tool is being used to verify and evaluate its Quality of Service framework.

NetMeter, besides providing an easy to use interface, and a solid framework for active network testing, also incorporates basic passive traffic analysis capabilities which highly enhance the usability and possible uses of the tool, particularly being very useful in the EuQoS project.

On the other hand, summarizing all the results for the study of the 802.11/IPv6/MIPv6 handover, we can conclude that, the handover loses packets, specifically, the rate multiplied by the handover latency. In our experiments, we found that (in average) a handover loses 65 packets when the traffic rate is 34 packets per second (and 207 when the rate is 94 packets per second), this means that the handover latency is, in average, 2.05 seconds.

Regarding the QoS level before and after the handover, we found that, when the traffic rate is low (VoIP), the OWD and the IPDV doesn't suffer any variation, however, when the bandwidth is higher (1Mbps), there are severe IPDV fluctuations for packets transmitted before the handover.

This results, (specially packet losses and IPDV) show that the 802.11/IPv6/MIPv6 is not able to support a proper quality voice transmission. The only solution for this matter is to improve the handover latency, that is, to improve Mobile IPv6 or change it to better protocols such as Fast Handovers.

Altoghether there are several possible improvements which are left as future work, first there are the traffic generation limitations imposed by the fact that we are only using MGen and NetPerf on our scenario. Anyway there are plans to solve this issue by permitting NetMeter to use other traffic generation tools as OWAMP [24].

Related to [7] NetMeter is being used for Interdomain QoS analysis, so, is left as future work the analysis and constrains of such studies in deeper detail.

References

[1] René Serral, Roberto Borgione, "NetMeter a NETwork performance METER," 2004. [Online]. Available: http://www.ccaba.upc.es/netmeter

[2] G. Almes, S. Kalidindi and M. Zekauskas, "A One-way Delay Metric for IPPM," RFC 2679, Sept. 1999.

[3] C. Demichelis and P. Chimento, "IP Packet Delay Variation Metric for IP Performance Metrics (IPPM)," RFC 3393, Nov. 2002.

[4] "[IST] 6QM - IPv6 QoS Measurement," Jan. 2003. [Online]. Available: http://www.6qm.org/

[5] "[IST] MESCAL - Management of End-to-end Quality of Service Across the Internet at Large," Jan. 2004. [Online]. Available: http://www.mescal.org/

[6] "[IST] INTERMON - Inter-domain QoS Research," Sept. 2003. [Online]. Available: http://www.ist-intermon.org/

[7] "[IST] EuQoS - End-to-end Quality of Service support over heterogeneous networks," Sept. 2004. [Online]. Available: http://www.euqos.org/

[8] J. F. R. Hervella, "MGEN: The Multi-Generator Toolset," 2002. [Online]. Available: http://matrix.it.uc3m.es/~long/software/mgen6/mgen6/

[9] R. Jones, "Public netperf homepage," 2003. [Online]. Available: http://www.netperf.org/netperf/NetperfPage.html

[10] G. Combs, "Ethereal: The world's most popular network protocol analyzer," 2004. [Online]. Available: http://www.ethereal.com

[11] IEEE, "802.11: Wireless LAN medium access control (MAC) and physical layer (PHY) specification," no. 802.11, 1997.

[12] C. Perkins, "IP mobility support for IPv4," RFC 3344, Aug. 2002.

[13] D. Johnson, C. Perkins and J. Arkko, "IP mobility support for IPv6," RFC 3775, June 2004.

[14] Marco Liebesch, Xavier Pérez Costa and Ralph Schmitz, "A MIPv6, FMIPv6 and HMIPv6 handover latency study: Analytical approach," *IST Mobile and Wireless Telecommunications Summit*, June 2002.

[15] Xavier Pérez Costa and Hannes Hartenstein, "A simulation study on the performance of mobile IPv6 in a WLAN-based cellular network," *Computer Networks: The International Journal of Computer and Telecommunications Networking*, vol. 40 Issue 1, 2002.

[16] A. Mishra, M. Shin and W. Arbaugh, "An empirical analysis of the IEEE 802.11 MAC layer handoff process," *ACM SIGCOMM Computer Communications Review*, vol. 33, Apr. 2003.

[17] N. Montavont and T. Noel, "Handover management for mobile nodes in IPv6 networks," *IEEE Communications Magazine*, Aug. 2002.

[18] Helsinki University of Technology, "MIPL Mobile IPv6 for Linux," 2004. [Online]. Available: http://www.mobile-ipv6.org/

[19] Internet2 Consortium, "OWAMP - NTP configuration," 2004. [Online]. Available: http://e2epi.internet2.edu/owamp/details.html#NTP

[20] Rajeev Koodl, "Fast handovers for mobile IPv6," draft-ietf-mipshop-fast-mipv6-03.txt, 2004.

[21] H. Soliman, C. Catelluccia, K. El Malki, L. Bellier, "Hierarchical mobile ipv6 mobility management (HMIPv6)," 2004.

[22] John Q. Walker, NetIQ Corporation, "A handbook for successful voip deployment: Network testing, qos, and more," July 2002.

[23] A. Cabellos-Aparicio, R. Serral-Gracià, L. Jakab, J. Domingo-Pascual, "Measurement based analysis of the handover in a WLAN MIPv6 scenario passive and active measurements (to be published)," 2005.

[24] S. Shalunov, B. Teitelbaum, A. Karp, J. W. Boote, M. J. Zekauskas, "A One-way Active Measurement Protocol (OWAMP)," draft-ietf-ippm-owdp-14.txt, 2004.