Extension of a network monitoring tool with IPv6 features (Ntop)

Abdelkader Lahmadi, Olivier Festor

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

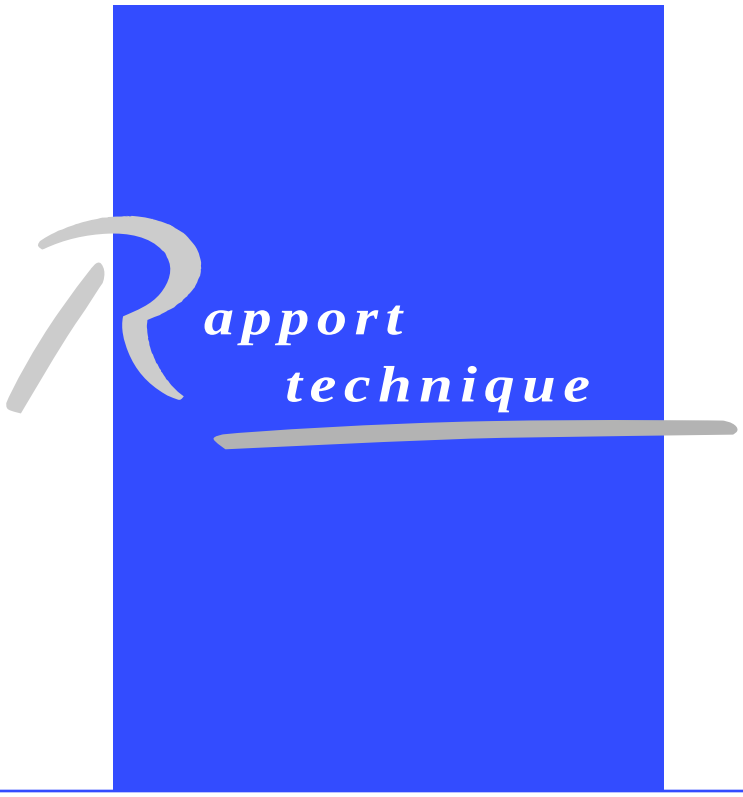
Extension of a network monitoring tool with IPv6 features (Ntop)

Abdelkader Lahmadi and Olivier Festor

N° 0292

Février 2004

THÈME 1



ISSN 0249-0803

Extension of a network monitoring tool with IPv6 features (Ntop)

Abdelkader Lahmadi and Olivier Festor

Thème 1 — Réseaux et systèmes
Project Madynes

n° 0292 — Février 2004 — 25 pages

Abstract: To support IPv6, most of the managed frameworks need advanced extensions. In the context of the 6net project we contribute to this evolution by extending Open Source frameworks. In this report we present our porting of a network monitoring tool called ntop to IPv6. Ntop is an open source web-based network usage monitor that enables users to track relevant network activities including network utilisation, established connections, network protocol usage and traffic classification.

Key-words: ntop, IPv6, Network monitoring, Traffic measurement, Traffic analysis

Unité de recherche INRIA Lorraine
LORIA, Technopôle de Nancy-Brabois, Campus scientifique,
615, rue du Jardin Botanique, BP 101, 54602 Villers-Lès-Nancy (France)
Téléphone : +33 3 83 59 30 00 — Télécopie : +33 3 83 27 83 19

Extension de l'environnemnt NTOP pour le monitorage d'un réseau IPv6

Résumé : Afin de supporter IPv6, la plupart des environnement et logiciels de supervision existants requierent des extensions importantes. Dans le cadre de projet 6net, nous contribuons à cet effort vice le portage et l'adaptation de logiciels libres. Dans ce rapport nous présentons le portage d'un outil en IPv6 de monitorage de réseaux ntop. Ntop est un outil libre source basé sur une interface web pour monitorer l'utilisation d'un réseau. Il permet aux utilisateurs de savoir toute activités sur le réseau, les connections établies entre les machines, les protocoles utilisés et une classification de trafic.

Mots-clés : ntop, IPv6, monitorage de réseaux, mesure de trafic, analyse de trafic

Contents

Chapter 1

Introduction

Network monitoring has been considered a crucial activity since the early days of networking. Administrators had to keep track of network traffic and hosts activity for several reasons, including detection of network bottlenecks and plannig network extensions. Many tools have been created for traffic measurement.These are mostly based on packet capture like tcpdump or snoop for analysing network traffic and tracking network and protocol connectivity issues. This knid of tools needs off-line applications for correlating captured data and identifying the network flows. Other tools for network monitors such as NetTraMet and CoralReef offer adavanced network flow analysis and statistic report generation. Nevertheless, those tools are a little bit heavy and need a lot of CPU and memory and have been designed as instrumentable network daemons.A more simple and effcient tools have been designed to monitor networks and reports the actual network status in human-readable format like ntop, netprobe. Ntop [3] is a simple, free and portable traffic measurement and monitoring tool. Similar to the Unix top tool that reports processes CPU usage, ntop identifies the hosts that were currently using most of the available network ressources.

Most of those tools do not yet support IPv6 [2] while in some cases offers a minimal support for this protocol, next Generation Internet based on IPv6 has been designed to solve the actual addressing problem and obtain potentiel adavantages mainly in fields like mobility, security or quality of services. Porting network monitoring tools is necessary to

assure the smooth transition to IPv6. But porting this kind of applications needs more effort than simple client/server applications where only sockets need to be modified to be protocol independentto listen to IPv6 and IPv4 [7] connections,IPv6 is not only carrying an extended addressing scheme, but it offers a lot of new services and protocols (DNSv6, DHCPv6, ICMPv6,..) and also new kinds of addresses (Link local, site local, anycast). Monitoring an IPv6 environment is different than an IPv4 environement in several ways. Moreover, monitoring applications should be aware of the changes between IPv4 and IPv6 to monitor efficently a hybrid or native IPv6 network. To monitor and analyse IPv6 networks the choice is to develop new applications or porting existing ones to support IPv6. The current case to enhance the ntop framework we decided in order to support IPv6 devices, adding the IPv6 extensions and requirements. This document is further structured as follows:Section 2 gives an overview on ntop and its functions needed to be ported to IPv6. Section 3 presents in detail the most significant modifications that hade to be made to the source code. In section 4 we describes an example of the utilization of ntop to monitor an IPv6 platform.In section 5 we present some conclusions and futur works.

Chapter 2

NTOP Overview

2.1 Introduction

Ntop (Network TOP) is an open-source software application [6] developed and maintained by Luca Deri written using the C language available free of charge under the GNU public license. Similar to the Unix top tool that reports processes CPU usage, the authors needed a simple tool able to report the network top users (hence the term ntop) for quickly identifying those hosts that were currently using most of the available network ressources. The current version of ntop features both command line and web-based user interfaces, and is available on both UNIX and Win32 platforms.Ntop focuses on:

- traffic measurement
- traffic monitoring
- network optimization and planning
- detection of network security violations

The next section describes the ntop architecture.

2.2 Architecture

The ntop architecture is depicted in Figure 2.1. Ntop is based on three layers: the sniffing

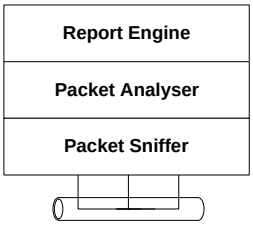


Figure 2.1: ntop architecture

layer, packet and data analysing layer and the reporting layer.

2.2.1 Sniffing layer

This component is based on the libpcap [8] library under a Unix and platform-specifc packet capture library for other operating systems like winpcap for windows. The packet sniffer supports different network interface types including PPP, Ethernet and Token Ring and allows captured packet to be filtered before being processed by the analyser. For IPv6 networks we need to capture packets with type `ETHERTYPE_IPv6 = 0x86DD`. This can be done by using the BPF [5] filter facility of libcap to capture only IPv6 packets or on within the analysing layer by handling only the coresponding IPv6 packets.

2.2.2 Analysing layer

The packet analyser processes one packet at a time. According to information analysed from the packet header, host informations are updated. This information stored in a large hash table whose key is 48 bit hardware MAC address. Each entry contains several counters that keep track of the data sent/received by the host, stored according to the supported network protocols (TCP , UDP, ICMPv6, OSPFv3, PIM). For each IP packet, the appropriate protocol counter is updated. Moreover, ntop handles IP fragments. When the first fragment is encountered, fragment information is stored in the hash table. Fragment information is

removed as soon as the last fragment has been received. For IPv6, fragments are detected through the protocol type which should be `IPPROTO_FRAGMENT = 44`. The host entry also contains a list of the host's active TCP connections. Hence if the received packet is a TCP packet, then the host TCP connection list is updated.

2.2.3 Reporting layer

Ntop acts as a daemon with an embedded http server that allows remote users to analyse traffic statistics through a web browser. For each request, the server dynamically creates web pages and sends them to the user. The statistics shown by the web mode are the following:

- IP multicast.
- Host information.
- Traffic Statistics.
- Currently active TCP sessions.
- IP/non-IP Protocol distribution.
- Local subnet traffic matrix.
- Network flows.
- Local network usage.

To support the IPv6 protocol the report engine functions needs to be modified to take care of the IPv6 addresses.

2.3 Functions

There are two classes of information that can be retrieved from ntop:

- Network traffic measurement.
- Network traffic monitoring.

The traffic measurement consists in measuring the network bandwidth utilisation on a local network. Both the total network bandwidth utilization and individual host bandwidth utilization are tracked by analyzing the packets on the network. Here are some elements that are tracked by ntop:

- Data received: the ntop application tracks how much data is received by each host identified on the network (the destination host address).
- Data sent : the ntop application also tracks the sending host and the type of data sent by each host.This feature allows administrator to see which hosts are sending the most data on the network.
- Network throughput: the network throughput is displayed using graphs, showing the average network load at different points of time.

For the IPv6 support, the network traffic measurement function needs to measure the IPv6 traffic by analyzing also IPv6 packets and update dedicated traffic counters.

The traffic monitoring function provides information on the type of traffic that is present. The following informations are provided by the network monitoring function :

- Statistics: They show how much traffic of a specific type has been seen by ntop, as well as indicating which hosts have produced the different types of network traffic.This include multicast traffic, information about packets captured by ntop, host statistics and domains.
- IP traffic: the ntop application monitors all traffic seen on the network interface and divides it into three categories : remote to local, local to remote and local to local. Where local represent the machines on the local LAN and remote the outside ones.
- IP protocols: ntop keeps statistics for each protocol within the IP packet, such as TCP and UDP. Each IP application is tracked to determine which hosts are using it and how much traffic it has generated.

The monitor function need to track IPv6 applications as well update statistics for this protocol.

thus, all monitoring processing done for the IPv4 protocol needs to be ported for the IPv6 protocol as well support of the larger address fileds is required.

Chapter 3

IPv6 extensions for NTOP

In this chapter we provide the details of the extensions provided to ntop for IPv6 support.

3.1 Data structures

In the first stage of porting ntop to IPv6 [2],we have defined an IP independent data structure to represent either IP4 or IPv6 addresses. The definition of the structure called HostAddr is the following.

```
typedef struct hostAddr {
  u_int     hostFamily;
  union {
   struct in_addr  _hostIp4Address;
   struct in6_addr _hostIp6Address;
  }addr;
}HostAddr;
```

This structure uses the field hostFamily to specify the address family of the represented address.This field should be `AF_INET` for IPv4 or `AF_INET6` for IPv6. We have modified all ntop data structures that used `struct in_addr` for IPv4, to the new IP independent data

structure HostAddr. Some useful macros have been defined to easily access various address fields of this structure.

```
#define Ip4Address addr._hostIp4Address
#define Ip6Address addr._hostIp6Address
```

A set of functions have also been written for handling the new address structure. These functions enable creating, copying and comparing IP independent addresses.

```
unsigned short addrcmp(HostAddr *addr1, HostAddr *addr2)
HostAddr *addrcpy(HostAddr *dst, HostAddr *src)
unsigned short addrput(int family, HostAddr *dst, void *src)
unsigned short addrnull(HostAddr *addr)
unsigned short addrfull(HostAddr *addr)
```

In the next stage of our work we have modified the operating existing functions to support IPv6.First, we have located functions that use IPv4 addresses as parameters or which handle this kind of addresses. We have modified all these functions to be IP version independent.

3.2 IPv6 considerations in the analyzing layer

The packet processing function `processIpPkt` is the core of the analysing traffic layer. It decodes IP packets and updates the related network informations. Hosts information is stored in a large hash table. This table is updated by analysing each packet header. This function was extended to process IPv6 packets headers and payload. The same hash table contains now, both IPv6 and IPv4 hosts information. This hash table is used by the reporting layer functions to retrieve information about hosts. The processIpPkt function uses an IP addresses resolver function. This function resolve addresses and maintains a cache where IP address resolution records (mapping numeric/symbolic IP address) are stored. The resolver function was modified to be IP independ, and resolves both IPv4 and IPv6 addresses. Also, the resolver cache is modified to be IP independend. The ntop application decodes the IP payload to gather usefull information from the upper layer protocols (TCP and UDP). These protcocol are IP independend. The only consideration was that the IPv6 protocol

can encapsulate more than one payload. We need to look at the option field in the IPv6 packet to get the respective payload. We have add the support of the ICMPv6 [1] protocol which contains more messages than the ICMP protocol for IPV4. The ICMPv6 protocol was designed to replace more than one protocol and not only the ICMP one. The ICMPv6 messages could contains DHCPv6 messages, replace ARP protocol,etc. These messages contain useful information for network administrators. Currently, we have a minimal support for this protocol, but in futur work ntop should investigate more the ICMPv6 messages.

3.3 IPv6 considerations in the reporting layer

The reporting layer uses network information collected by the analysing layer. So all the main IPv6 support is done in this last layer. For the reporting layer we have modified some utility functions to be IP independend. For example functions for representing IP addresses in character strings format were modified in this way. We used the IP independend `inet_ntop` function to interpret both IPv6 and IPv4 addresses. Also, we added the display of statistics about the ICMPv6 protocol. This layer was written to be IP neutral, thus facilitating the porting.

3.4 Getting the network interface IPv6 addresses

In the first stage, ntop should retrieve information about the network interfaces on which running. The main information to be retrieved is the set of IP addresses of this interface. For IPv4, we can use the `ioclt` system function with the `SIOCGIFADDR` parameter to get the addresses. For IPv6, in some systems like Linux this function does not return addresses of the `AF_INET6` family. We have written a system dependend module that provides the IPv6 addresses of a network interface. This module tries to retrieve the addresses according to the operating system on which ntop is running.

For Linux we have used the kernel supplied file `/proc/net/if_inet6` to obtain a list of all existing IPv6 addresses in the system. An example of this file is shown in the table 3.4.

The first column is the IPv6 address and the last column is the interface name it is assigned to. The loop back device (`lo`) is also listed but filetred by the detection loop as

00000000000000000000000000000001 01 80 10 80	lo
fe8000000000000020475fffebee808 04 0a 20 80	eth1
200106881fb80032020475fffebee808 04 40 00 00	eth1
2001066010130032020475fffebee808 04 40 00 00	eth1
2001066003010032020475fffebee808 04 40 00 00	eth1

Table 3.1: *Sample entries of the /proc/net/if_inet6 file*

it is for IPv4, too. This method more easily is specific to the linux kernel version 2.4, but there seems to be no way to do this detection.

In the next release we will investigate the use of the pcap `pcap_findalldevs` function to obtain the IP addresses of a network interface. This function allows ntop to be more portable depending on the portability of the pcap library.

3.5 Porting the embeding ntop HTTP server to IPv6

The ntop application embodies a HTTP web server that allows users to connect to ntop by using the HTTP protocol and have a look to the netwok being monitored. We have extended the connection function to be protocol independent which allows the server to accept both IPv4 and IPv6 connections from web clients. The server uses the network API independend protocol to resolve addresses and open the listening socket. We have used the main two standard functions getaddrinfo and getnameinfo to initialize the web server socket attributes. These functions are protocol-independent and accepts both IPv4 and IPv6 addresses. When launching ntop through the command line the user could specify the address family (`AF_INET` or `AF_INET6`) on which the server will listen by using the `-6` or `-4` options.Default the server listens to both IPv4 and IPv6 connections.

Chapter 4

The Experimental Environment

4.1 Installing of ntop

The main Web site for ntop is located at *http://www.ntop.org*. The current development release (currently 2.2.97) source code available from CVS has the IPv6 support.For downloading ntop from CVS you should try the commands below:

```
cvs -d :pserver:anonymous@cvs.ntop.org:/export/home/ntop login
```

the required password is `ntop` then you can check out the current CVS tree by using :

```
cvs -d :pserver:anonymous@cvs.ntop.org:/export/home/ntop co ntop
```

Now you can begin the compilation process by executing the following commands as root:

```
./configure
make
make install
```

By default, ntop has the IPv6 support. To disable this feature launch the configure script with the -disable-ipv6=yes option. Before compiling the ntop application install the GD library to create all the fancing graphics used on the web page. Hence this is done, run ntop with the HTTP server listens to IPv6 connections by using the `-6` option or `-4` option for

IPv4 connections: `ntop -6 -K -u ntopuser` By using the option `-i` you can specify the network interface on which ntop will running.

4.2 Setup of Testbed

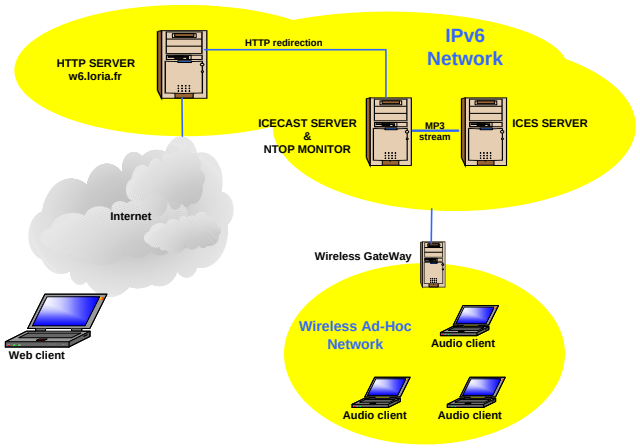


Figure 4.1: Ntop with the IPv6 support testbed

In order to test ntop with the IPv6 support, we have used our already deployed IPv6 network. Our testbed consists in a set of machines running a mixture of Linux and FreeBSD operating systems. Ntop runs on a linux machine with a redhat 8.0 distribution. For compilation we uses the version 3.2 of the gcc compiler. The version of the pcap library is 0.6 which supports the IPv6 protocol. Also, our IPv6 testbed is a mixture of wired and wireless networks. An IPv6 service is deployed on our network which is MP3 streaming service.We have installed the two open source servers icecast and ices [4]. The version 0.2.3 of ices with the libsout builtin has the IPv6 support. The libshout library provides the network connection functions to the ices server. The streaming server will be monitored by the ntop application to see some usesful statistics. The wireless network is in the Ad-Hoc mode. It consists of a set of machines running Linux. On each machine runs the audio client xmms with the IPv6 support to enable users to connect to the icecast server. To allow

HTTP connections to the ntop application from a public network we have used the HTTP proxying function. All connections to the HTTP server embedded into ntop are redirected by our public HTTP server `http://w6.loria.fr`. Remote users should be identified by a username and a password which is a feature enabled in the ntop web server. The testbed is depicted in Figure 4.1.

4.3 Testing ntop with IPv6 support

The aim of the testbed is to test our porting of ntop to IPv6 with a real IPv6 service which is an audio streaming service. The ntop application monitors the bandwidth consumed by the server, the audio clients and lists all IPv6 sessions between them. The Figure 4.2 depicts the running hosts on the testbed. By clicking on the streaming server IPv6 address it displays detailed statistics about the host, and the data that was transfered as depicted in Figure 4.3. Current sessions can be seen as depicted in Figure 4.4 by clicking `sessions` under the `IP protos` category tab.

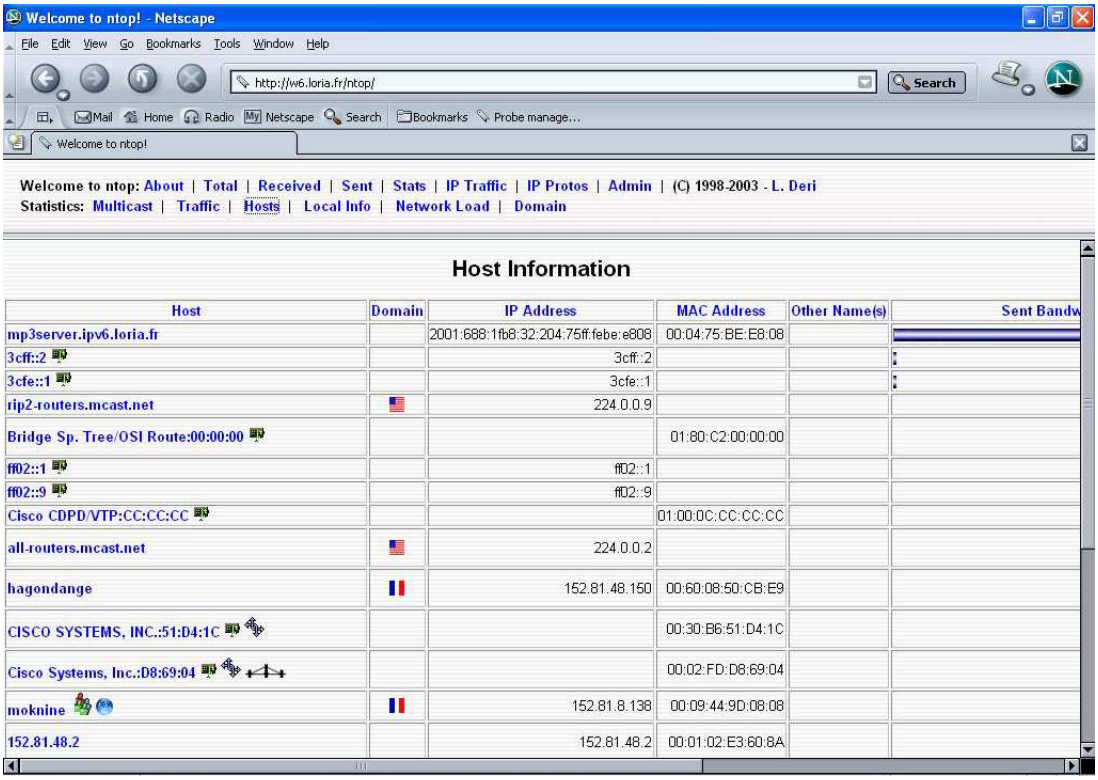


Figure 4.2: The hosts running on the testbed as shown by ntop

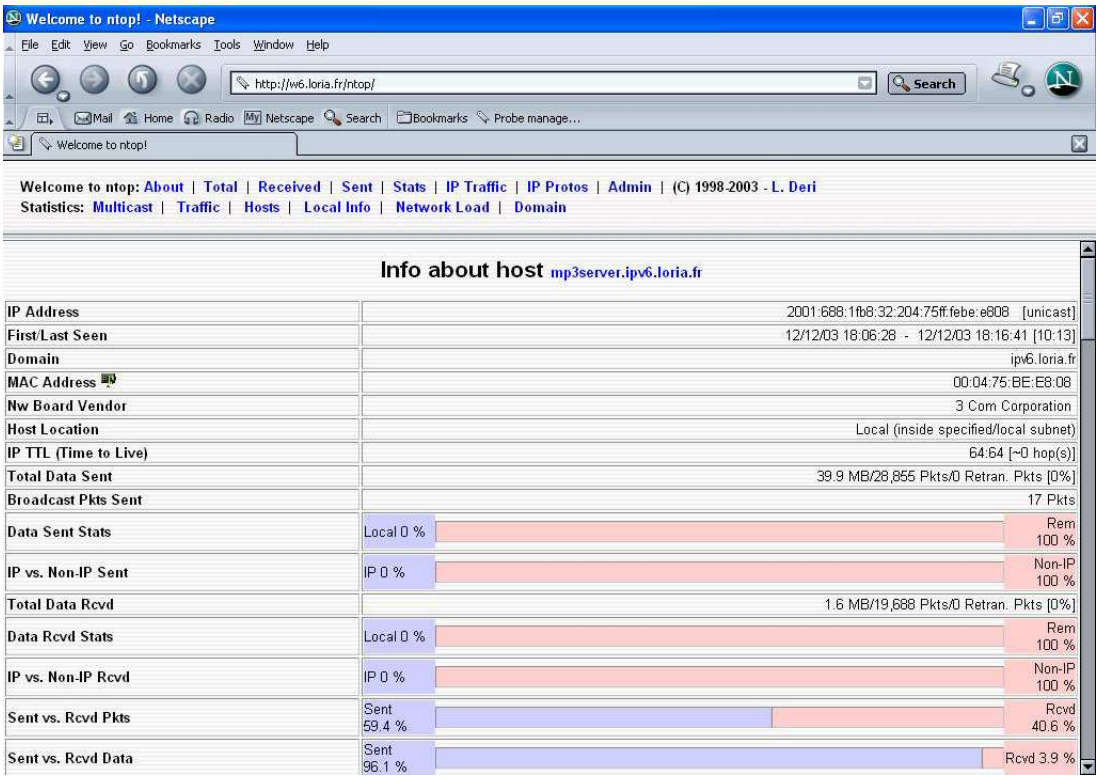


Figure 4.3: The streaming server detailed informations

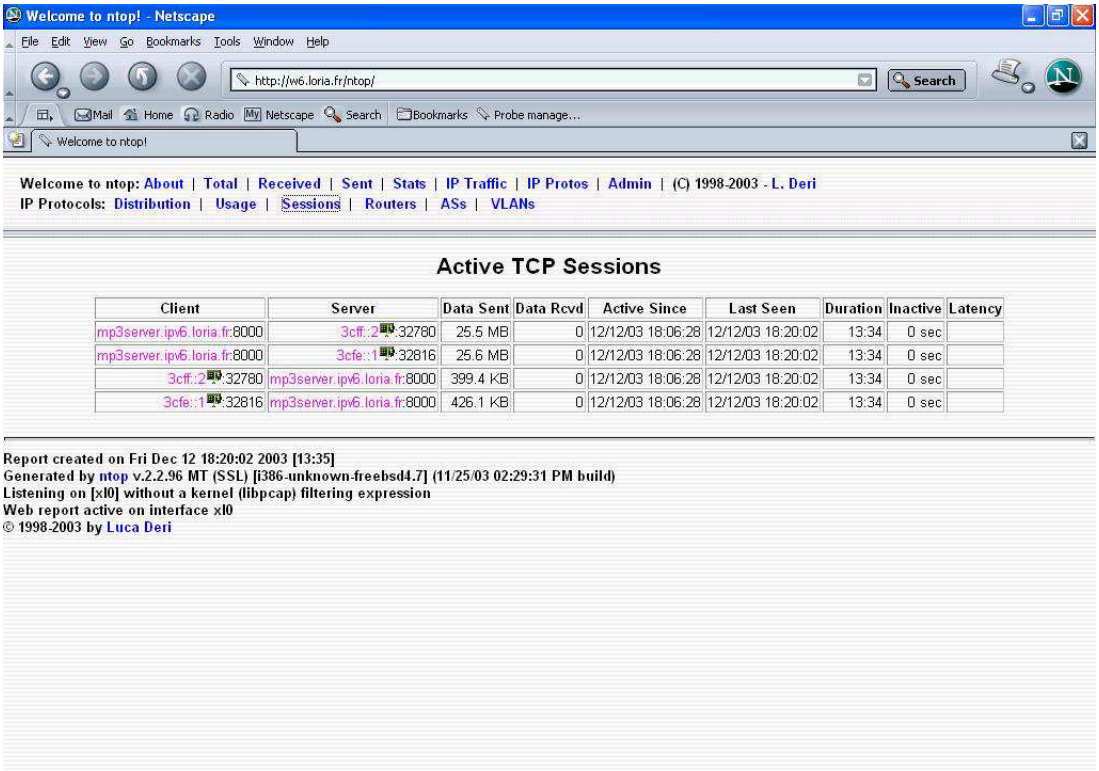


Figure 4.4: The active sessions on the testbed

Chapter 5

Conclusion and Futur works

To faster the deployement and acceptance of IPv6 networking the availability of supporting management solution is required. To this end we ported the ntop monitoring tool to Ipv6. This tool is simple and easy to deploy for monitoring a network and gatthers useful informations about the running services. By porting ntop to IPv6, we have investigated the monitoring of a LAN IPv6 network with a real IPv6 service like audio streaming. Such service with an IPv6 support, that can be deployed in an university campus or an entreprise, can be monitored by an application like ntop. Our contribution is integrated in the current cvs version of ntop and available on thenntop site. This version supports both IPv4 and IPv6 protocols. In a future contribution, we will investigate more IPV6 protocols like DNSv6, DHCPv6 which bring a lot of useful informations about the network monitored by a tool like ntop. Nevertheless, due to the original ntop design, it cannot be easily in environments such as a wireless system like a PDA with limited resources. In addition, it would be nice to distribute light network probes on the network that send traffic information towards a central traffic analysis console such as ntop. In order to satisfy the above requirements nProbe has been designed [6]. Nprobe supports yet the IPv6 protocol. But its adaptation is required for wireless networks to support the 802.11 protocol. We will investigate this adapatation in future work.

Bibliography

[1] A. Conta and S. Deering. Internet control message protocol (icmpv6) for the internet protocol version 6 (ipv6) specification. RFC 1885, Internet Engineering Task Force, December 1995.

[2] S. E. Deering and R. Hinden. Internet protocol, version 6 (ipv6) specification. RFC 2460, Internet Engineering Task Force, December 1998.

[3] Luca Deri and Stefano Suin. Ntop: Beyond ping and traceroute. In *DSOM*, pages 271–284, 1999.

[4] Icecast. The official icecast web site. http://www.icecast.org.

[5] Steven McCanne and Van Jacobson. The BSD packet filter: A new architecture for user-level packet capture. In *USENIX Winter*, pages 259–270, 1993.

[6] ntop. The official ntop web site. http://www.ntop.org.

[7] J. B. Postel. Internet protocol. RFC 791, Internet Engineering Task Force, September 1981.

[8] V.Jacobson, C.Leres, and S.McCanne. libpcap, lawrence berkeley national labs, 1994. ftp://ftp.ee.lbl.gov/.

Unité de recherche INRIA Lorraine
LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)

Unité de recherche INRIA Rennes : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)
Unité de recherche INRIA Rhône-Alpes : 655, avenue de l'Europe - 38330 Montbonnot-St-Martin (France)
Unité de recherche INRIA Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)
Unité de recherche INRIA Sophia Antipolis : 2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-0803