www.euro6ix.net

Title:	Document Version:
Technical Report TR4.1A.2 **Multicast over IPv6 Networks**	2.6

Project Number:	Project Acronym:	Project Title:
IST-2001-32161	Euro6IX	European IPv6 Internet Exchanges Backbone

Contractual Delivery Date:	Actual Delivery Date:	Deliverable Type* - Security**:
31/12/2002	25/02/2003	R – PU

* Type: P - Prototype, R - Report, D - Demonstrator, O - Other
** Security Class: PU- Public, PP – Restricted to other programme participants (including the Commission), RE – Restricted to a group defined by the consortium (including the Commission), CO – Confidential, only for members of the consortium (including the Commission)

Responsible and Editor/Author:	Organization:	Contributing WP:
Cristina Peña	TID	WP4

Authors (organizations):

Miguel A. Morales (Consulintel), Cesar Olvera (Consulintel), Francisco Fontes (PTIN), Carlos Parada (PTIN), Jacinto Vieira (PTIN), Eduardo Azañón (TID), Ignacio Grande (TID), Olaf Bonnes (T-Nova), Lothar Grimm (T-Nova), Tim Chown (UoS).

Abstract:

This document describes investigation carried out on Multicast within the Euro6IX along the first year of the project. After comparing IPv4 and IPv6 Multicast, some implementations were selected to be evaluated and test scenarios were defined and implemented. Results of trials and experiments will be used when incorporating Multicast to the Euro6IX test-bed as a network service. First connections to the M6Bone are described. Besides a video broadcasting service has been developed.

Keywords:

M6Bone, Multicast, Multicast Listener Discovery (MLD), Protocol Independent Multicast (PIM)

© Euro6IX Consortium

Revision History

Revision	Date	Description	Author (Organization)
v0.1	05/14/2002	Document creation: Index defined.	Antonio Skarmeta (UMU)
v0.5	06/10/2002	Contents updated	Olaf Bonness (T-Nova)
v1.0	12/05/2002	Contents updated and new chapters added about M6Bone and the Multicast Service Experience	Miguel A. Morales (Consulintel) César Olvera (Consulintel) Eduardo Azañón (TID) Cristina Peña (TID) Olaf Bonnes (T-Nova) Lothar Grimm (T-Nova)
v1.1	12/13/2002	Contents updated	Cristina Peña (TID)
v2.0	12/19/2002	UoS and PTIN contributions integrated in main document	Francisco Fontes (PTIN) Carlos Parada (PTIN) Jacinto Vieira (PTIN) Cristina Peña (TID) Tim Chown (UoS)
v2.1	01/10/2003	Document updated to match project deliverable template	Antonio Skarmeta (UMU)
v2.2	01/10/2003	Final changes	Cristina Peña (TID)
v2.3	01/10/2003	Minor style changes	David Fernández (UPM)
v2.4	01/10/2003	Minor style changes	Cristina Peña (TID)
v2.5	02/20/2003	Final revision	David Fernández (UPM)
v2.6	25/02/2003	Logos added an PDF generated	Jordi Palet (Consulintel)

Executive Summary

This document describes the activities carried out by the Euro6IX project within the first year of WP4, sub-activity A4.1-2.

First and second chapters describe and compare IPv4 and IPv6 Multicast Protocols, explaining the advantages of Multicast routing over Unicast Routing and the state of the art of the process of standardisation the IPv6 Multicast protocols.

After that, third chapter presents a set of existing IPv6 Multicast implementations. In order to verify the proper functionality of the IPv6 Multicast implementations and features of the networking infrastructure, several test scenarios have been defined and implemented. A complete description of scenarios and obtained results can be found in fourth chapter. It is also interesting to connect Euro6IX and M6Bone to continue with Multicast experiments, and first connections have been established.

Finally, a video service streaming has been implemented, in order to test a multicast multimedia streaming service.

Table of Contents

Table of Figures

1. INTRODUCTION

This introduction explains the main advantages of Multicast Routing in comparison with Unicast Routing.

When it is necessary, to send the same data (for example Video-Streaming, Data-Subscription or distributed Software-Updates) to a large number of receivers, then all the packages have to be sent with Unicast on the same links multiple times depending on the number of receivers.

If we are using Multicast instead of Unicast, the packages will be sent only once over the same link.

Therefore we will save network resources and the speed of the data transfer will increase.

This and the following chapters will focus on some information about IPv4 Multicast and the implementation and testing of IPv6 Multicast within the context of the Euro6IX project.

1.1 IPv4 Multicast addresses

The IPv4 Multicast address is used to address a group of receivers. If the first four bits of a normal IPv4 address are set to 1110, then this address is a Class D or IPv4 Multicast address.

Therefore all IPv4 Multicast addresses are located in the range between 224.0.0.0 and 239.255.255.255. Out of this range of 2^{28} = 268.435.456 theoretically possible addresses some are reserved fore special usage, as described in Figure 1-1.

Address	Usage
224.0.0.1	All Hosts in the same subnet
224.0.0.2	All Gateways in the same subnet
224.0.0.4	All DVMRP[1] Router
224.0.0.5	All MOSPF[2] Router
224.0.0.13	All PIM[3] Router
224.0.0.22	All IGMPv3 Router
224.0.1.1	Network Time Protocol – NTP
224.2.127.254	Session Announcement Protocol – SAP
232.0.0.0/8	Reserved for SSM (*SSM - Source Specific Multicast*) applications
239.0.0.0/8	Administrative scooping

Figure 1-1: Important Reserved IPv4 Multicast Addresses

[1] DVMRP – Distance Vector Multicast Routing Protocol

[2] MOSPF – Multicast Open Shortest Path First

[3] PIM – Protocol Independent Multicast

1.1.1 IGMP

Any Multicast address represents a group of receivers. This group has to be co-ordinated with a group management system, which can subscribe or describe a listener in or from a Multicast group. Also the management system has to provide information about the Multicast Group itself.

For the fulfilment of these tasks the Internet Group Management Protocol (IGMP) was developed. IGMP is a protocol of the Network-Layer and therefore a part of IP.

IGMP is implemented in the Router as well as in the end systems.

The protocol tasks of the Router are to query for listeners of the Multicast groups and to collect this information in order to forward the requests for Multicast groups to the neighbouring routers.

1.1.2 Routing Protocol

For the forwarding of Multicast packets special Routing protocols are necessary to determine the optimum path to the destination. In this direction the Multicast Routing protocol PIM (Protocol Independent Multicast) becomes the standard Multicast Routing Protocol for IPv4 in the Internet.

In contrast to MOSPF (Multicast Open Shortest Path First) and DVMRP (Distance Vector Multicast Routing Protocol) PIM co-operates with any underlying Unicast Routing protocol and not only with OSPF respectively RIP.

PIM uses directly the Unicast Routing table of the Router to find its forwarding information.

The following variations of PIM are available:

PIM-SM	⇒	PIM – Sparse Mode	Is useful for widely distributed listeners.
PIM-DM	⇒	PIM – Dense Mode	Is useful for more than one listener per network.
PIM-SSM	⇒	PIM – Source Specific Multicast	Offers more Security against DOS-attacks. It also ensures, that the selected data-stream is from a dedicated source. The PIM-SSM needs IGMPv3, which is in the implementation phase today.

For the operation of PIM-SM, the existence of a Rendezvous Point (RP) is a prerequisite to handle the registered listeners. To keep the delay-time and Overhead in a lager network small, it is necessary to operate with several dispersed RPs. The exchange of the collected data (about the registered listeners) between the RPs in different PIM Routing Domains is handled by the Multicast Source Discovery Protocol (MSDP) [1].

1.2 Motivation IPv6 Multicast

Like IPv4, IPv6 uses also Multicast to distribute data to a group of receivers. However, the Multicast has in IPv6 a special significance, because IPv6 does not provide any Broadcast mechanisms. Hence dedicated Multicast groups are used to implement Broadcast-like mechanisms.

Therefore the Multicast under IPv6 is not a special feature like in IPv4; it is a necessary component of the system.

2. IPv6 Multicast Standardisation

2.1 Multicast IPv6 addresses

IPv6 Multicast addresses are used to address a well-defined list of Interfaces, as it is known from IPv4. The structure of an IPv6 Multicast address is shown in Figure 2-1.

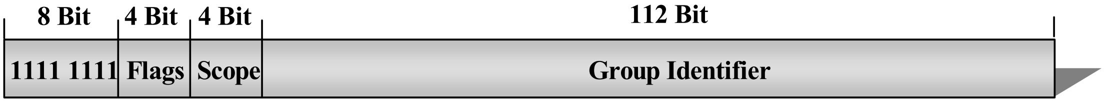


Figure 2-1: Structure of an IPv6 Multicast Address

When the first 8 bits of the address are equal 1, then the corresponding IPv6 address is a Multicast address. The next 4 bits (Bit 9 - 12) represent the Flags-Field within the IPv6 address. At the moment only the most-right flag is defined yet. The remaining 3 Flag bits are reserved for future use.

If the most-right Flag bit is equal 0, then the IPv6 Multicast address is well known and registered from IANA / ICANN. If this Flag bit is equal 1, then it is a transient Multicast address for ad hoc use.

The Scope-Field (Bit 13 - 16) represents the scope of the Multicast address and is described in the [2]:

- 0000 reserved
- 0001 Node-local Scope
- 0010 Link-local Scope
- 0101 Site-local Scope
- 1000 Organisations-local Scope
- 1110 Global Scope
- 1111 reserved

The combination of the 112 bit Multicast Group Identifier together with the Flags and Scope Fields represents different Multicast groups. (Reference [3] and chapter 10.1 contain a list of all registered Multicast addresses which are defined by IANA / ICANN).

Different Multicast groups
FF01::1 Addresses all local Interfaces from the system
FF02::1 Addresses all link-local Interfaces (All Nodes Multicast Address)
FF05::2 Addresses all site-local Router
FF02::1:FF00:0/104 plus the most-left 24 bit of the Unicast address Addresses the Solicited-Node link-local Multicast address

The last Multicast addresses of the example above will be generated using Unicast addresses. This enables the IPv6 end-systems to realise the Address Duplication Detection within a Neighbour Discovery mechanism.

2.2 Multicast Listener Discovery

Multicast Listener Discovery (MLD) [4RFC 2710] is used by an IPv6 Router to discover the presence of Multicast listeners (that is, nodes wishing to receive Multicast packets) on its directly attached links, and to discover specifically which Multicast addresses are of interest to those neighbouring nodes.

MLD is derived from version 2 of IPv4's IGMP. One important difference to note is that MLD uses ICMPv6 (IP Protocol 58) message types, rather than IGMP (IP Protocol 2) message types.

The collected information of MLD is provided to whichever Multicast Routing protocol is being used by the Router, in order to ensure that Multicast packets are delivered to all links where there are interested receivers.

The MLD messages are sent with a link-local IPv6 Source Address. Therefore the Hop Limit of the Package is 1. A Router, in the query mode for a link, periodically [Query Interval] sends a General Query on that link (see Figure 2-2), to solicit reports of all Multicast addresses of interest on that link.

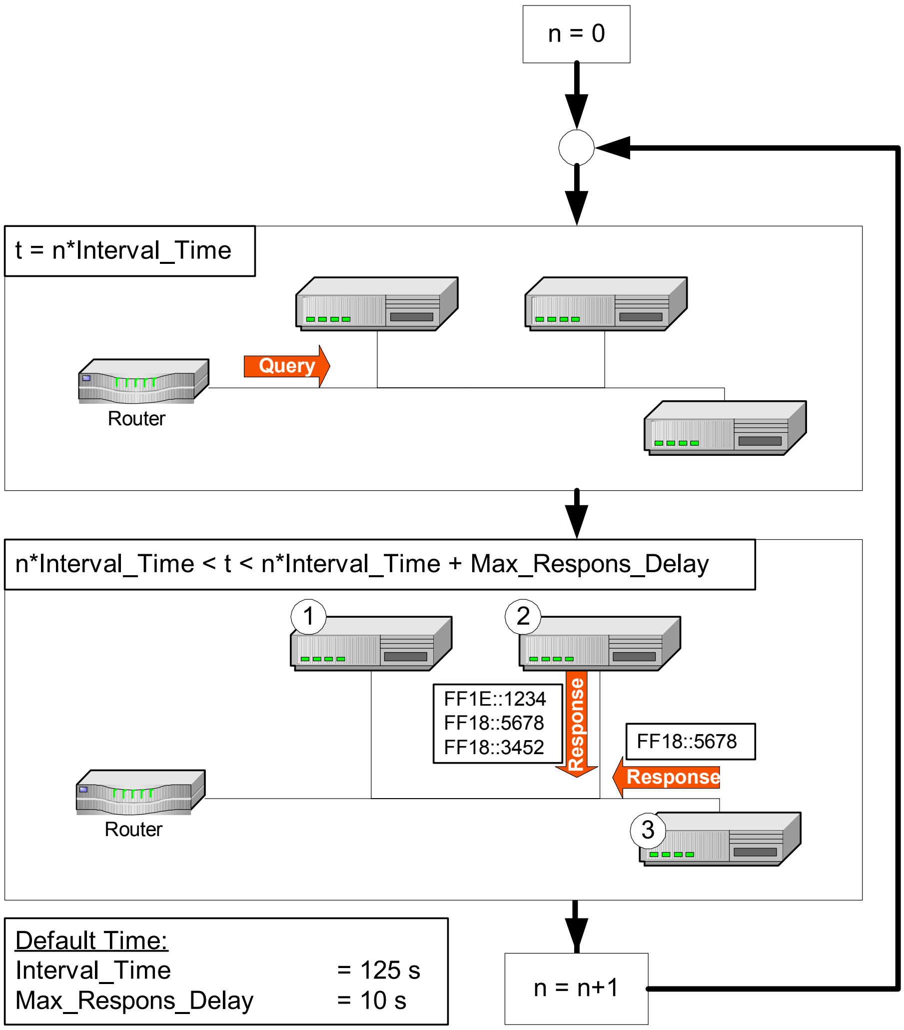


Figure 2-2: MLD Package Transfer

General Queries are sent to the link-local all-nodes Multicast address (FF02::1), with a Multicast Address field of 0::0, and a Maximum Response Delay of [Max_Response_Delay].

The listening Clients answer with a RESPONSE message containing a random delay value lower than Max_Respons_Delay to the Router.

Description of Figure 2-2:

Client (1) is not listening to any Multicast.

Client (2) is listening to three Multicast groups and answers to the Query with a Response list of three Multicast addresses. (FF1E::1234; FF1E:5678; FF18::3452)

Client (3) is listening to one Multicast group and answers with a Response containing the address FF18::3452.

The Clients are also able to unsubscribe from Multicast groups by sending a DONE MLD message to the Router.

Currently, **Multicast Listener Discovery Version 2 (MLDv2)** for IPv6 [1] is in draft status. MLDv2 is derived from version 3 of IPv4's Internet Group Management Protocol, IGMPv3. Compared to the previous version, MLDv2 adds support for "source filtering", that is, the ability for a node to report interest in listening to packets *only* from specific source addresses, or from *all but* specific source addresses, sent to a particular multicast address.

2.3 IPv6 Multicast Routing

PIM is designed to work together with different Unicast Routing protocols and Address formats (e.g. IPv4, IPv6, IPX, and AppleTalk). Nevertheless the PIM protocol has to be modified to support IPv6 because PIM has to use the MLD mechanism instead of IGMP. Simultaneously the MLD protocol has to be implemented in the site-local network infrastructure.

As for IPv4, the following variations of PIM are available:

- Protocol Independent Multicast - Sparse Mode (PIM-SM) [5].
- Protocol Independent Multicast - Dense Mode (PIM-DM) [6].
- Source Specific Mode (SSM) [7].

The PIM protocol uses the Unicast Routing table for the RPF (Reverse Path Forwarding) check. So, every IPv6 Multicast Router has to know the Unicast routes to the other sites. For the Multicast Routing-Protocol PIM-SSM the MLDv2 protocol is required in order to specify the source address of the senders of the Multicast groups. Until today the MLDv2 protocol is still in draft status [1] and PIM-SSM is therefore not available on the most router platforms.

2.3.1 Global IPv6 Multicast Routing

Currently global IPv6 Multicast Routing is impossible [8] except using SSM. The problem is that if there are multiple PIM domains in the Internet, there will be multiple PIM RPs and nowadays RPs have no way of communicating the information about Multicast sources to other Multicast domains since MSDP has not been ported to IPv6.

One solution could be porting MSDP to IPv6 and probably existing implementations would be easily modifiable but there is some resistance since MSDP for IPv4 was born as a "stop-gap" measurement. Another possibility could be to specify or implement another Multicast Routing protocol. For instance, Border Gateway Multicast Protocol (BGMP) has been specified but many problems have raised, no implementations exist and may never be implemented.

From a completely different approach, a new method of embedding the address of RP in IPv6 Multicast address [9] by profiting from reserved bits in IPv6 address has been proposed.

To summarise, it will be reminded that currently global IPv6 interdomain Multicast is completely impossible except using SSM.

3. IPv6 Multicast implementations to analyse - End systems and Routers

There are many different IPv6 Multicast implementations available. The most of them are software based.

3.1 Open-Source Solutions

3.1.1 KAME Project (BSD)

The KAME project is started to realise IPv6 on BSD-Platforms. With respect to IPv6 Multicast until now implementations of PIM-DM, PIM-SM and recently SSM (it was finished by November 2002 and might need more debug) are available.

Therefore investigations and test will be possible using the Multicast Routing functionality of a BSD-Platform based router.

Furthermore KAME has already implemented the Multicast tools VIC, VAT and icecast (ported from the well-known IPv4 Multicast streaming tools).

The KAME kit is released in two ways:

- Kame SNAP kit (for hackers/researchers): it comes with more experimental protocols/APIs support and userland programs.
- Kame kit Integrated *BSD source code tree (for normal users): will always be based on public release version of *BSD, so IPv6 tree becomes available on top of stable *BSD platforms.

The following official releases integrate KAME kit:

- FreeBSD 4.0 and beyond
- NetBSD 1.5 and beyond
- OpenBSD 2.7 and beyond
- BSD/OS 4.2 and beyond

3.1.2 Usagi Project (Linux)

USAGI (UniverSAl playGround for IPv6) Project is an aggressive IPv6 development project, mainly for Linux systems.

The group is on the way to implement the IPv6 Multicast Routing protocol PIM-SM for Linux.

3.1.3 MRT

The MRT (Multi-Threaded Routing Toolkit) is a partnership between the University of Michigan and Merit Network. The target of the group is the research of new Routing software architecture. MRT realised routing software with support for IPv4/IPv6 BGP4+, DVMRP, RIP/RIPng, PIM-DM, and OSPF. Therefore IPv6 Multicast Routing is with DVMRP and PIM-DM available. The software is realised for Windows NT/2000, BSD, Linux and SunOS/Solaris.

The last update of the software was released at August of the year 2000.

3.1.4 Zebra

GNU Zebra is a routing software package that manages TCP/IP based routing protocols. Zebra supports RIPng, OSPFv3 and BGP4+ for IPv6. Therefore the Unicast Routing at IPv6 is possible, but the Multicast Routing is not supported yet.

The latest release is zebra-0.93a dated on July 2002.

3.2 Commercial Solutions

3.2.1 Cisco

Since IOS 12.2(2)T Cisco provides RIPng based on [3]. Therefore the Multicast address FF02::9, the destination address for RIP update messages, is provided. But these messages are sent only in the link-local scope so that there is no need for an IPv6 Multicast routing protocol to support RIPng.

Roadmap

With the Cisco IOS IPv6 Roll-Out Phase III, Cisco will realise the following Multicast relevant protocols:

- MLDv2 ([4] + [1])
- PIMv2-SM
- PIM-SSM

3.2.2 Juniper

The current software release from Juniper (5.3) is IPv6 enabled, but does not provide IPv6 Multicast Routing yet.

Roadmap

Juniper will implement IPv6 Multicast Forwarding and the following Multicast relevant Protocols in the year 2002:

- MLD
- PIMv2-SM

3.2.3 Hitachi

Hitachi is one of the first companies, which implemented the IPv6 Routing in hardware. In the GR2000 Router the Routing protocols RIPng, OSPFv3 and BGP4+ are integrated. Also the Multicast protocols PIM-SM and MLD are implemented, hence it is possible to realise IPv6 Multicast Routing.

3.2.4 Ericsson Telebit

The Ericsson Telebit A/S has developed an IPv6 router with the ability of IPv6 Multicast Routing. The supported IPv6 Multicast Routing protocols are PIM-SM and PIM-DM.

3.2.5 NEC

With the Integrated Switch Routers IX5000 Series NEC offers an IPv6 Multicast Routing System that provides the IPv6 Multicast (routing) protocols MLD, PIM-SM and PIM-DM.

3.2.6 6Wind

The 6WINDGate 6200 Series is a smart IP Access Router which combines, in a single unit, all the features needed to provide a new set of IP services. Located at the boundary between the operator access network and the customer network, this CPE is a response to the networks critical needs in terms of QoS, Security, Mobility and Simplicity of Management.

Regarding IPv6 Multicast Routing protocols, it provides PIMv6-SM, PIMv6-SSM, MLDv1, and MLDv2.

	MLD	PIM-SM	MLDv2	PIM-SSM	MBGP	MSDP
Open-Source Solutions						
BSD	done	done	done	done	not planned	future work
Linux	done	at work	future work	future work	not planned	future work
Commercial Solutions						
6WIND; 6100 Series	done	done	done	done	not planned	future work
Cisco; v12.2 (T)	early release	early release	future work	future work	not planned	not planned
Hitachi	done	done	v6.07	v6.07	done	future work
Juniper; Release 5.4	future work	early release	future work	future work	done	future work
NEC	done	done	future work	future work	not planned	future work

Figure 3-1: Overview of IPv6 Multicast Implementations

4. DESIGNED EVALUATION PLAN

4.1 Background

In order to verify the proper functionality of the IPv6 Multicast implementations (end systems as well as Routers) and the IPv6 Multicast features of the networking infrastructure, several test scenarios will be implemented and investigated during the project lifetime of the Euro6IX project.

Because several partners within the Euro6IX consortium are interested in realising Multicast investigations and tests it is necessary to allow also the transport of IPv6 Multicast Traffic through the Euro6IX Backbone between the IPv6 Internet Exchanges.

The following test scenarios shall illustrate some of the possible approaches which will be under consideration during the following semesters of the project in dependency of the implementation status of the network platform as well the IPv6 Multicast routing implementations.

These first scenarios were defined using a straightforward approach keeping in mind that more sophisticated investigations will be carried out in the later semesters of the project.

At the moment there are for instance no scenarios for the interoperation of IPv6 Multicast with Route Reflectors or Route Servers and also the implementation and usage of the MSDP protocol is not taken under consideration.

4.2 Test Scenarios

4.2.1 Scenario 1: Link-Local IPv6 Multicast Test scenario

The simplest scenario for investigating IPv6 Multicast is illustrated by the following picture. Hereby both devices (Multicast sender and Multicast receiver) are situated within one network segment without involvement of an IPv6 Multicast router.

The IPv6 Multicast traffic (to IPv6 Multicast address ff00:80a0::1) is addressed to the Ethernet Multicast MAC address 33:33:00:00:00:01 and can be received by the Ethernet cards of the receivers without any additional protocol support like MLD or PIM-SMv6.

Hence this scenario can only be used to demonstrate the IPv6 Multicast ability of the end systems and the chosen application.

This scenario has been used to test Linux and Windows clients.

Test Results

For test results, see sections:

- 4.4.1 Test of Scenario 1: Test of Linux Clients
- 4.4.2 Test of Scenario 1: Comparison of Linux and Windows Clients.

4.2.2 Scenario 2: Site-Local IPv6 Multicast Test scenario

In difference to scenario 4.2.1, now also IPv6-capable routers are involved in the test scenario, which could also be integrated within a single participant's network without any necessary connection to the Euro6IX Internet Exchange or Backbone. Hence, this scenario can be tested and verified also during the initial phase of the Euro6IX project since no Backbone network and large IX infrastructure has to be involved in the investigations.

From a general point of view this second scenario is sufficient to make a statement if it is possible to implement also scenarios 3 to 5 which distinguish from scenario 2 only in terms of network complexity and involvement of consortium partners.

An exception is scenario 5 with "One PIM-SMv6 RP per IX" because here a protocol has to be implemented (MSDP - Multicast Source Discovery Protocol [12]) which realises the exchange of MC routing information between different Autonomous Systems. At the moment this protocol is still under standardisation (also for IPv4) within the IETF MSDP Multicast working group.

To test the functionality of different IPv6 Multicast capable routers in site-Local networks, this scenario can be divided in three scenarios:

4.2.2.1 IPv6 Multicast over Networks with IPv6 Multicast routers

The results of these tests will provide an overview of how the MLD, PIM-SMv6, PIM-DMv6 implementations work and if there are some basic problems with them.

PIM-SMv6 and PIM-DMv6 have been tested in FreeBSD routers. Just PIM-SMv6 basic functionality has been tested in Hitachi GR2000 routers.

4.2.2.2 IPv6 Multicast over Networks with IPv6 Not-Multicast routers

The purpose of this test is to deploy the IPv6 Multicast where only some routers support IPv6 Multicast. In these cases, the solution is to use tunnels IPv6 over IPv6 between Multicast routers.

4.2.2.3 IPv6 Multicast over IPv4/IPv6 mixed Networks

The objective of this experiment is to deploy the IPv6 Multicast on IPv4/IPv6 mixed networks. In this case, IPv6 over IPv4 tunnels are configured.

<u>Test Results</u>

For test results, see sections:

- 4.4.3 Test of Scenario 2: Networks with IPv6 FreeBSD Multicast routers and PIM-SMv6
- 4.4.4 Test of Scenario 2: Networks with IPv6 FreeBSD Multicast routers and PIM-DMv6
- 4.4.5 Test of Scenario 2: Networks with IPv6 Hitachi Multicast routers and PIM-SMv6
- 4.4.6 Test of Scenario 2: Networks with IPv6 Not-Multicast routers
- 4.4.7 Test of Scenario 2: IPv6 Multicast over IPv4/IPv6 Mixed Networks

4.2.3 Scenario 3: IPv6 Multicast between L2 customers of an IPv6 IX

A similar scenario to scenario 2 is implemented in scenario 3 with the difference that in this case various L2 customers of an IX are exchanging their IPv6 Multicast traffic.

Both partners have a private L2 peering within the IX and have agreed about a common PIM-SMv6 RP.

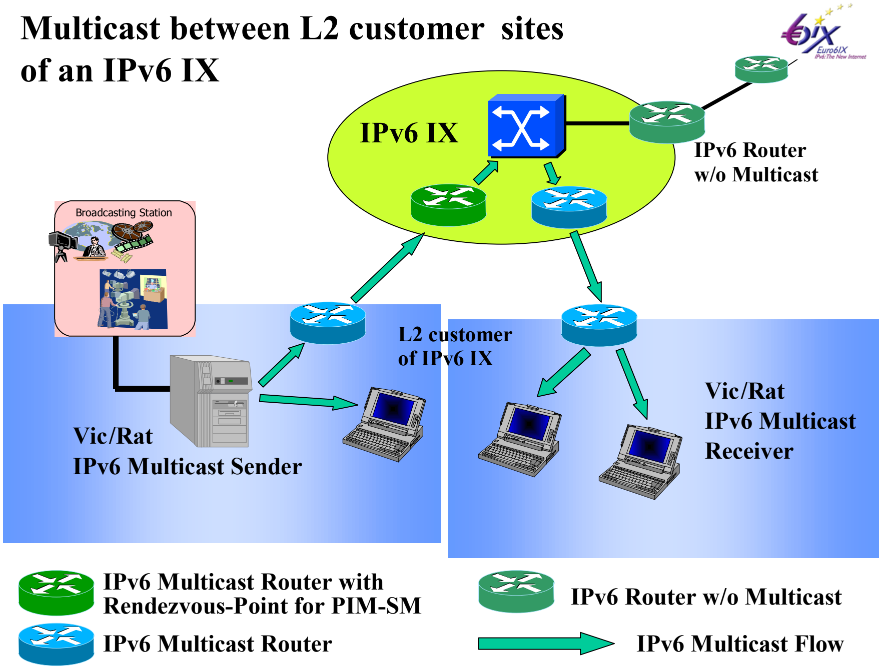


Figure 4-1: Scenario 3 – IPv6 Multicast between L2 customer sites of an IPv6 IX

In this case the IPv6 Multicast traffic will be exchanged only between these both parties and is transparent to the other routers of the IX infrastructure as well as to the other peering parties within this IX.

Problems can arise if within two different private peerings, the same IPv6 Multicast addresses are used because in this case the switch has to be able to separate these traffic streams (with equal IPv6 Multicast destination address) from each other. This can be normally done by L2 VPN approaches.

Note: Until now no interoperation between IPv6 Multicast and an IPv6 Route Reflector or server is assumed or necessary.

Test Results

For test results, see sections:

- 4.4.8 Test of Scenario 3

4.2.4 Scenario 4: IPv6 Multicast between L2 and L3 customer of an IPv6 IX

The 4th scenario uses as first scenario the new IX definition, which was formed within the Euro6IX project. Within the Euro6IX project an IX is assumed to offer also L3 services to its customers and this could be for instance an IPv6 Multicast Rendezvous Point.

Hence the RP for the PIM-SMv6 is implemented within the premises of the IX and all customers (L2 and L3 attached) are allowed to use these additional services.

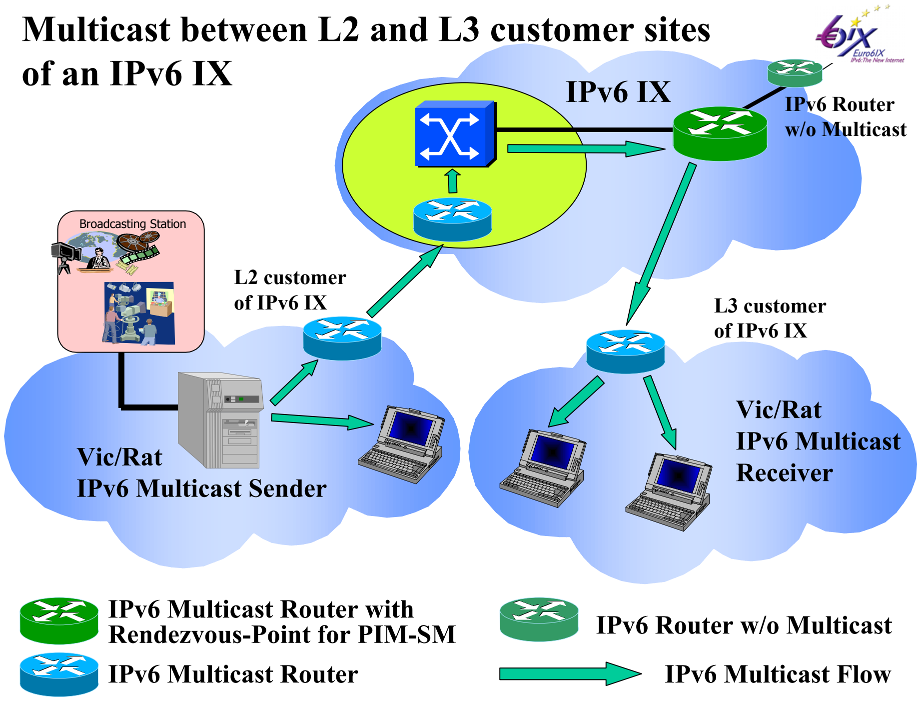


Figure 4-2: Scenario 4 – IPv6 Multicast between L2 and L3 customers of an IPv6 IX

These tests can be performed using only one IX infrastructure and are based on the same requirements with respect to protocol support as scenario 2.

Hence an implementation of this test scenario is possible during the starting phase of the project because no interconnectivity between the consortium partners is needed.

<u>Test Results</u>

For test results, see sections:

- 4.4.9 Test of Scenario 4

4.2.5 Scenario 5: IPv6 Multicast between customers of different IPv6 IXs

The most complex scenario (at the moment) is scenario 5, which can be divided into 3 sub-scenarios.

4.2.5.1 Using one global PIM-SMv6 Rendezvous-Point

Scenario 5a is very similar to scenario 4 but only with the difference that 2 and more IXs will be involved within the IPv6 Multicast network.

All the IXs have to agree in advance in the location of the RP and the interconnection between the involved IXs has to be established.

According to the timetable this scenario can be deployed at the earliest within the third semester of the project in dependency of the availability of PIM-SM implementations of the implemented routers.

Because there is only 1 common PIM-SMv6 RP within the project all the project partners have to send their IPv6 Multicast traffic in direction to this RP and have hence to care about the proper choice of the scope/TTL. This means that there is in worst case a very long and inefficient way for the IPv6 Multicast routing messages and hence a great delay in registering for IPv6 Multicast groups.

This problem can be solved using the scenario 5b, which is based on several IX local PIM-SMv6 RPs, which communicate with each other using the MSDP protocol.

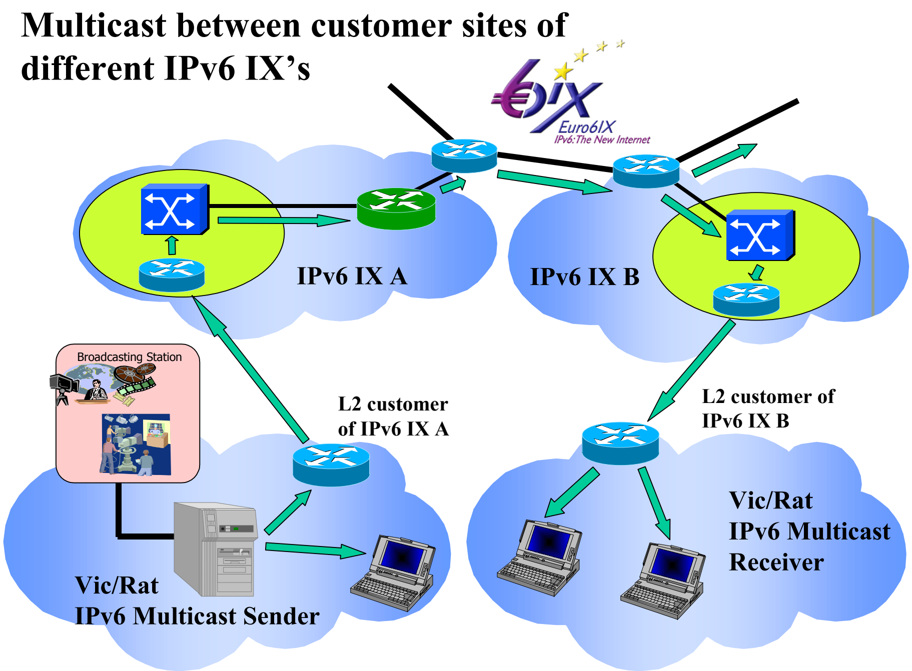


Figure 4-3: Scenario 5 – IPv6 Multicast between customers of different IXs

4.2.5.2 Using one PIM-SMv6 Rendezvous Point per Internet Exchange

To realise scenario 5b where each IX provides to its customers an IPv6 Multicast Rendezvous Point, some multicast inter-domain protocol such as the MSDP or other kind of solution has to be implemented or ported to support IPv6 also.

Because at the moment no information is available respectively a timeline for the availability of MSDPv6 (See section 2.3.1) it is not possible to estimate whether and when these tests can be performed within the project.

Currently global IPv6 interdomain Multicast is completely impossible except using SSM

4.3 Test Tools

In order to verify the IPv6 Multicast functionality of the end systems as well as the IPv6 Multicast routing capability of the network elements it is appropriate to use some well known IPv6-capable Multicast applications and supervise the network traffic using network monitor software like Ethereal (http://www.ethereal.com).

Besides that, the debug information of the router should be used to investigate the states of the network connections and Multicast trees.

4.3.1 Basic test applications

More or less all applications, which transmit data using IPv6 Multicast, are suitable to investigate the functionality of the different scenarios.

As an example only 3 applications should be mentioned:

4.3.1.1 RAT - Robust Audio Tool

The RAT Tool is one of the tools, which were initially developed for usage and testing within the M6Bone. RAT is able to transmit Audio on an IPv6 Multicast basis and can be downloaded from the following UCL Web pages (http://www-mice.cs.ucl.ac.uk/multimedia/software/rat).

4.3.1.2 VIC - Videoconferencing Tool

Another Multicast capable and easy to use tool out of the set of M6Bone tools is the VIC video conferencing application, which was also ported to support IPv6 already a long time ago.

Both basic test applications can be downloaded from the UCL web pages and are available in different releases for nearly all operating systems (Linux, BSD, Solaris, Microsoft etc.)

The packages can be downloaded in source code as well as in binary format. (http://www-mice.cs.ucl.ac.uk/multimedia/software/vic)

4.3.1.3 MGEN

MGEN is a toolkit designed to perform measurements over IPv4 networks, giving results about packet loss rate, delay and jitter. It provides support for rsvpd and subscription to Multicast groups. It is developed by the Naval Research Laboratory (http://manimac.itd.nrl.navy.mil/MGEN) and has been ported to IPv6 by Universidad Carlos III de Madrid in the scope of the LONG project (http://www.ist-long.com).

Lots of other applications can be found when more Internet research will be done, but for the beginning we focus on these 3 simple applications. Other available IPv6 Multicast applications will be investigated in more detail during the next semesters of the Euro6IX project.

4.3.2 Advanced test applications

4.3.2.1 ISABEL

Besides these relatively simple applications some more complex applications can be used as for instance the ISABEL video conferencing tool which was developed and ported to IPv6 by the Universidad Politécnica de Madrid (http://isabel.dit.upm.es)

This ISABEL tool will definitely used within the project in order to demonstrate the network connectivity and hence could also be used to verify the IPv6 Multicast functionality.

4.3.3 Monitoring and Measurement tools

At the time of writing this paper, the existence of a couple of tools for Multicast traffic with IPv6 support was discovered. They will be studied, and tested if it is found of interest for the project, along next semesters.

4.3.3.1 Multicast Beacon Server

The NLANR/DAST Multicast Beacon (http://dast.nlanr.net/projects/beacon/) is active measurement software that monitors the performance of a multicast session. It has two components:

- Beacon Client: an active probing program running at each machine. A set of Beacons sends packets continuously to each other through a multicast session, and measures the performance of the transmission. It then reports to the Beacon Server periodically. The current version (v0.63) is written in Java.
- Beacon Server: a central server collecting the performance information from the Beacon Clients. The v0.8 version of the Beacon Server is written in Perl.

Teamed up with the Access Grid, the Multicast Beacon provides measurement data for the current multicast traffic on the network. The Access Grid is a project led by ANL to implement large-scale distributed collaboration over the network. It relies on multicast for distributing audio, video, and other data across the network. The Access Grid web interface provides a peek into the current experimental session.

The Multicast Beacon can also be used as a general-purpose measurement tool as well.

4.3.3.2 TTCP

The widely used Test TCP (TTCP) is a command-line sockets-based benchmarking tool for measuring TCP and UDP performance between two systems. It was originally developed for the BSD operating system starting in 1984. The original TTCP and sources are in the public domain, and copies are available from many anonymous FTP sites. IPv6 support and Multicast support (IPv6 and IPv4) have been recently added (http://www.netcore.fi/pekkas/linux/ipv6/ttcp.c).

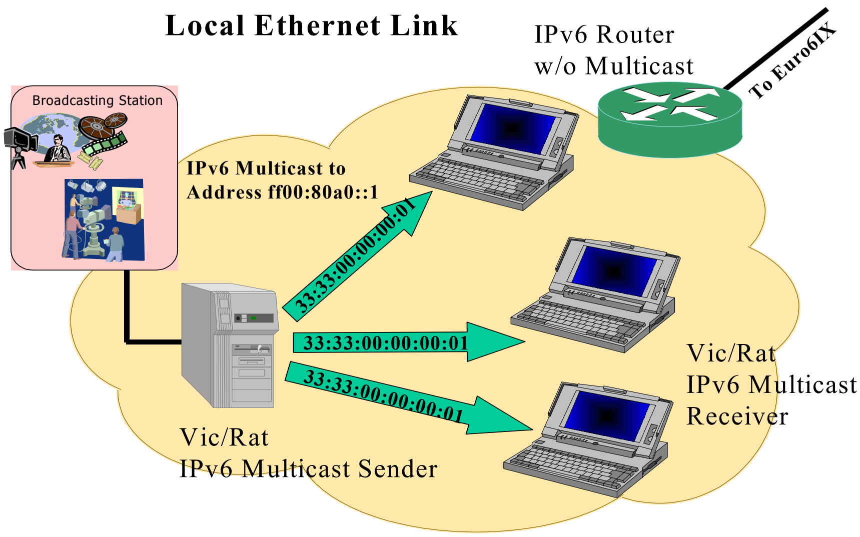
4.4 Test Results

As there are still few commercial routers that implement IPv6 multicast routing protocols, many of the tests have been performed on the FreeBSD platform, configured as IPv6 multicast routers. The version used for our tests was. The installation and configuration procedures of the Release FreeBSD 4.5 are explained in appendixes 10.3 and 10.5.

4.4.1 Test of Scenario 1: Test of Linux Clients

Topology

Referring to Figure 4-4 both end systems were located within an isolated IPv6 network. That is why no IPv6 Multicast routing protocol was needed and also the MLD mechanism was not required.



Mapping IPv6 Multicast Address ff00:80a0::1 to Ethernet MAC Address 33:33:00:00:00:01

Figure 4-4: Scenario 1 – Link-local IPv6 Multicast

The Multicast sender was connected to a standard video recorder and equipped with a Hauppauge Frame Grabber card in order to digitise the incoming video signal. The Audio signal of our Multicast TV program was generated through SoundBlaster cards.

The TV program (Video and Audio) was then multicasted to 2 Multicast receivers, which were able to join the Multicast sessions without any problems.

Configuration

End system:	Multicast sender: PC Pentium3 with 128 MB Ram with Hauppauge Video (WinTV) and SoundBlaster Card (IPv6 Address 3ffe:80a0::9)
	Multicast receiver: 2 PC Pentium 3 with 128 MB Ram and SoundBlaster card (IPv6 Address 3ffe:80a0::8)
Operating System:	Linux Redhat 7.2
Network:	10Mbit/s Ethernet
	Multicast group ff00:08a0::1
Application:	VIC Version 2.8ucl-1.1.3
	RAT Version 4.2.19

Results

Below some screen shots with the front ends of VIC and RAT as well with one screen dump with results of the packet-capturing program Ethereal.

Figure 4-5: Snap shots of VIC and RAT output / config windows in Scenario 1

These results proof that an IPv6 Multicast Audio / Video transmission was easily to set-up and use. No problems were discovered with the networking site of IPv6 Multicast; only some slight problems aroused in choosing the right and working binary for RAT.

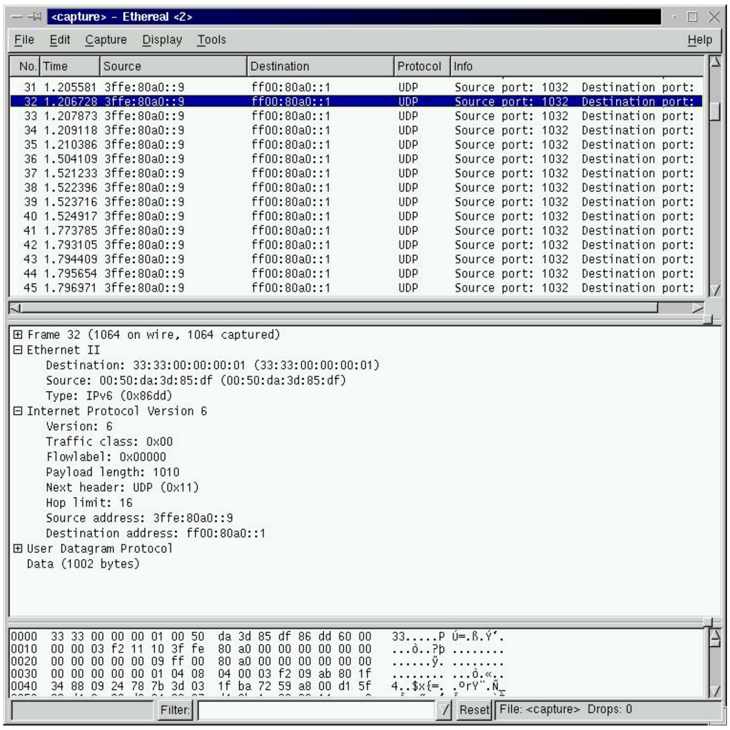


Figure 4-6: Ethereal trace of IPv6 Multicast data in Scenario 1

4.4.2 Test of Scenario 1: Comparison of Linux and Windows Clients.

One of the aims of this activity was to test several IPv6 multicast platforms besides of to check the performance, reliability and hardiness of IPv6 multicast implementations. That is why in this evolution of scenario 1 Windows 2000 and Linux 2.4.18 –3, with a Red Hat Linux 7.3 2.96-110 distribution, has been used at Consulintel laboratory.

This test has been performed using version 4.2.20 of RAT tool on Linux 2.4.18 –3.

Figure 4-7 illustrates the window that shows the appearance of this application. This appearance is the same both Linux and Windows platforms.

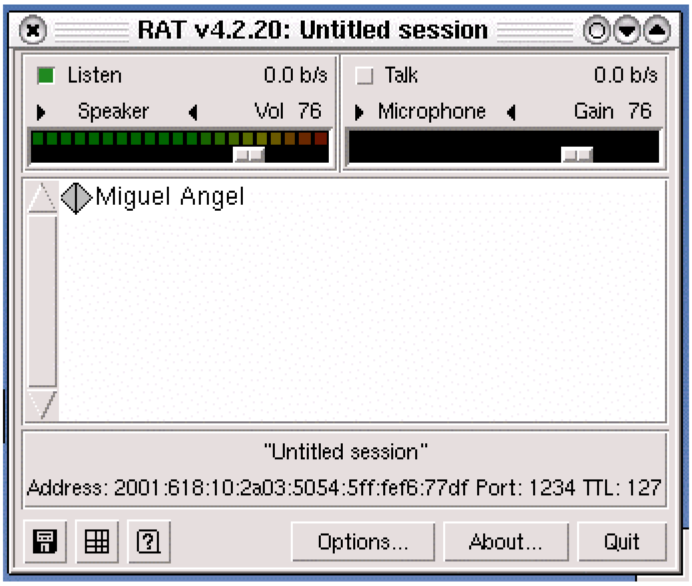


Figure 4-7: RAT window

As can be seen in Figure 4-8, the implemented scenario does not involve an IPv6 capable router, so it is the simplest possible scenario, where a local network segment joints all involved equipment's. Furthermore, no additional multicast protocol, as MLD or PIM-SM, is needed.

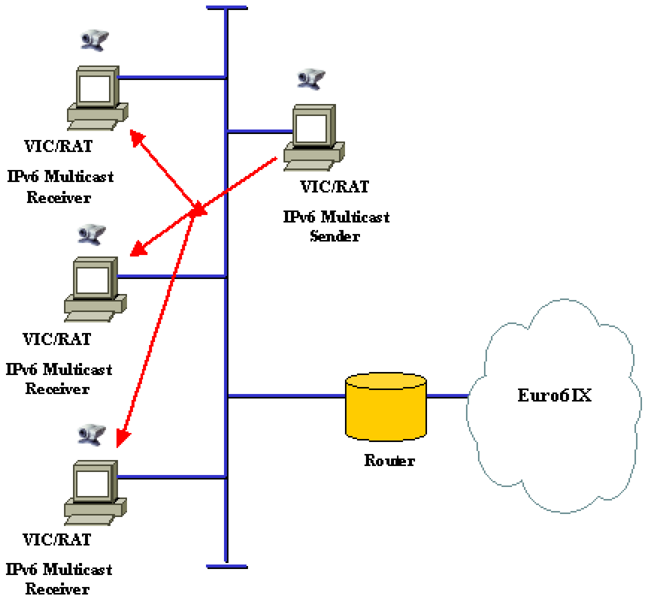


Figure 4-8: Multicast local test scenario

Figure 4-9 resumes the main tests performed in the previous scenario, the purpose of each test, and the results obtained.

Test	Purpose	Results
Applications configuration and performance	Verify the correct performance of the multicast applications both in IPv4 and IPv6.	Correct performance of the tool on Linux Not too stable performance of the tool under Windows. Further tests are underway
Multicast address configuration	Verify the correct configuration of IPv6 multicast addresses in hosts.	Correct configuration of multicast addresses
Multicast delivery	Verify that the packets sent to a multicast IPv6 address are delivered to all hosts with the that multicast address.	Correct delivery of multicast packets

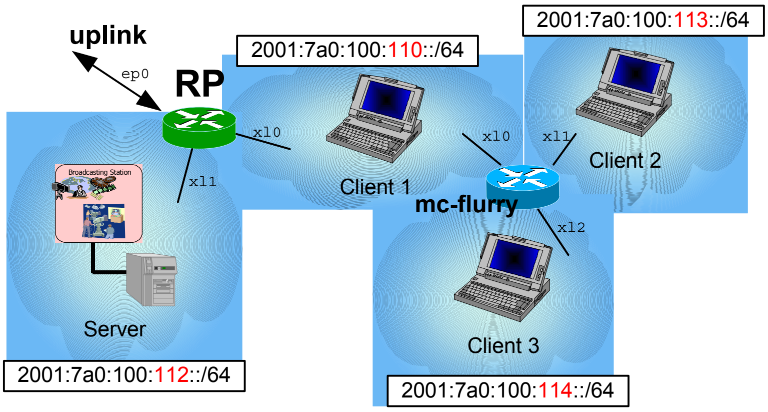
Figure 4-9: Local tests results

The performance seems to be much more stable on Linux than for Windows.

4.4.3 Test of Scenario 2: Networks with IPv6 FreeBSD Multicast routers and PIM-SMv6

Topology

For realising this test case, the server equipped with the Hauppage frame grabber card, which was already described in the previous scenario, was used. No further changes to the clients systems were necessary. To demonstrate Multicast packet routing the network was subdivided into several collision domains (sub-nets), which required two additional routers each running a FreeBSD release 4.6 operating system and version 20020403a of pim6sd (Figure 11).



	xl0:	2001:7a0:100:110:204:76ff:fe9e:4228
	xl1:	2001:7a0:100:113:204:76ff:fe9e:41dd
	xl2:	2001:7a0:100:114:204:76ff:fe9e:3ed5
	xl0:	2001:7a0:100:110:210:4bff:fe3e:1f77
	xl1:	2001:7a0:100:112:260:8ff:fe7e:987
Server:		2001:7a0:100:112:250:daff:fe3d:85df
Client 1:		2001:7a0:100:110:260:8ff:fe6f:5a20
Client 2:		2001:7a0:100:113:290:27ff:fedc:22fe
Client 3:		2001:7a0:100:114:204:76ff:fea0:181e

Figure 4-10: Structure of the IPv6 Multicast Scenario 2

Configuration

End System: Multicast Server: PC Pentium 3 with 128 MB Ram with Hauppauge Video (WinTV) and SoundBlaster Card (Linux Redhat 7.2)

Multicast Client: 3 PC Pentium 3 with 128 MB Ram and SoundBlaster card (Linux SuSE 8.0 Kernel 2.4.18)

Router: 2 PC Pentium 3 with 128 MB Ram (FreeBSD)

Network: 10Mbit/s Ethernet

Multicast group ff15:08a0::12

Application: VIC Version 2.8ucl-1.1.3

RAT Version 4.2.19

During the test realisation we observed some IPv6 Multicast routing problems with one of the clients from scenario 1. This client was possible to send MC packages into the other subnet, but wasn't able to receive MC packages.

After in-depth debugging it turned out that this client was not in the actual list of receivers because of syntax errors within the Multicast Listener Reports sent by this client.

After changing the Linux kernel 2.4.2 to a SuSE 8.0 implementation on the client the problem vanished and the same binaries for Vic and Rat could be used successful.

For illustration reasons below the configuration file of the PIM daemon (pim6sd.conf):

```
#  $FreeBSD:  ports/net/pim6sd/files/pim6sd.conf,v  1.2  2002/04/03
07:48:41 suz Exp$
#
# Please comment out the following line, if you uses stf0
phyint ep0 disable;
cand_bootstrap_router xl0;

cand_rp xl0;
group_prefix ff15::/16;
```

phyint ep0 disable	The uplink Interface ep0 is disabled for IPv6 multicast routing
cand_bootstrap_router xl0	Set the Bootstrap Router (One BSR which determines the RP has to be defined per routing domain.)
cand_rp xl0	The Interface xl0 is set as RP-Interface
roup_prefix ff15::/16	The RP manages only multicast groups with the prefix ff15::/16

The configuration file of the second router (mc-flurry) was not changed and thus contained just the commentary lines.

Results

After starting the Vic and Rat applications on all client hosts an IPv6 Multicast data stream could be initiated and transmitted from the MC server to all the clients. Hence this Multicast scenario 2 was realised and tested successful.

Figure 12 shows a dump of the network traffic for subnet 2001:7a0:100:110::/64. (To grab only relevant packets the packet filter was set to "pim or icmpv6".

In every network an IPv6 MC-Router was responsible for sending Multicast Listener Queries.

Since subnet 2001:7a0:100:110::/64 contains two IPv6 MC-Routers, these both routers have to agree on the task, who of them is responsible for sending the ML queries.

This decision is made independently from the RP status of one of the routers.

In the scenario above the router “mc-flurry” was chosen as queerer, who sent a Multicast Listener Query on interface xl0.

Only Client 1 (*:5a20) responded to this MLD queries since this client was the only one within the pre-arranged Multicast address space.

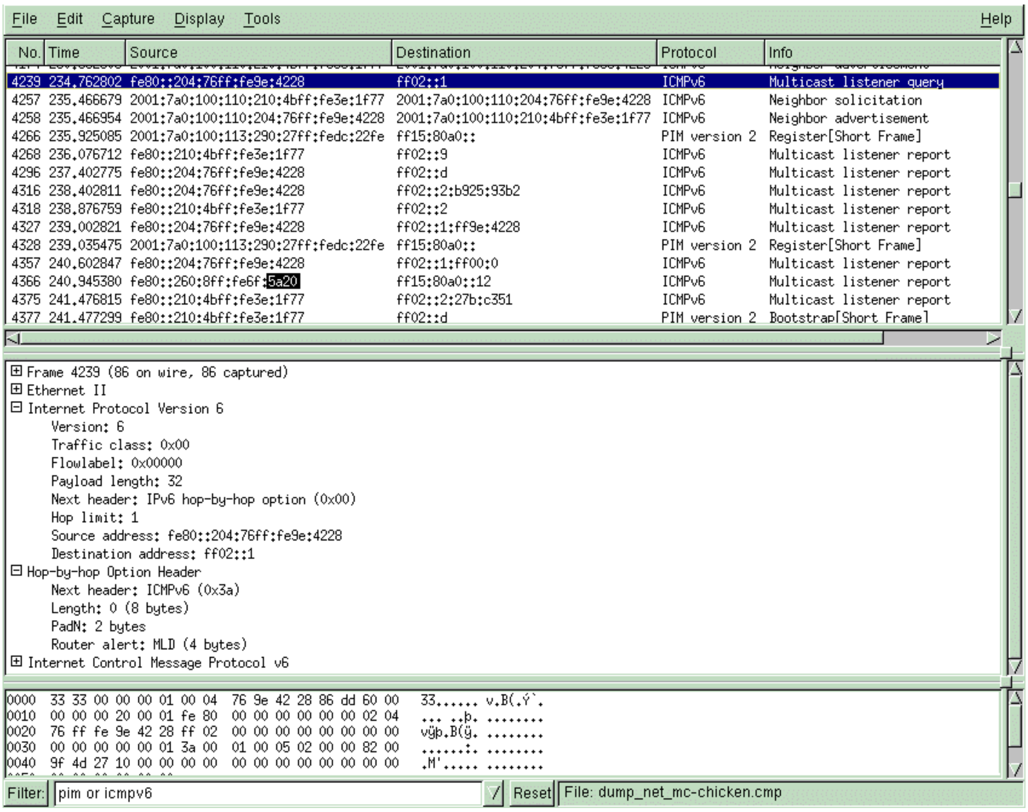


Figure 4-11: Ethereal trace at 2001:7a0:100:110::/64 in scenario 2

The second part of dumped data was grabbed in the subnet 2001:7a0:100:113::/64 (see figure 13). Since there was only one router in the subnet "mc-flurry's" interface x11(*:41dd) was automatically responsible for the MLD queries.

The only listener on that subnet that responded to the queries was Client 2 (*:22fe).

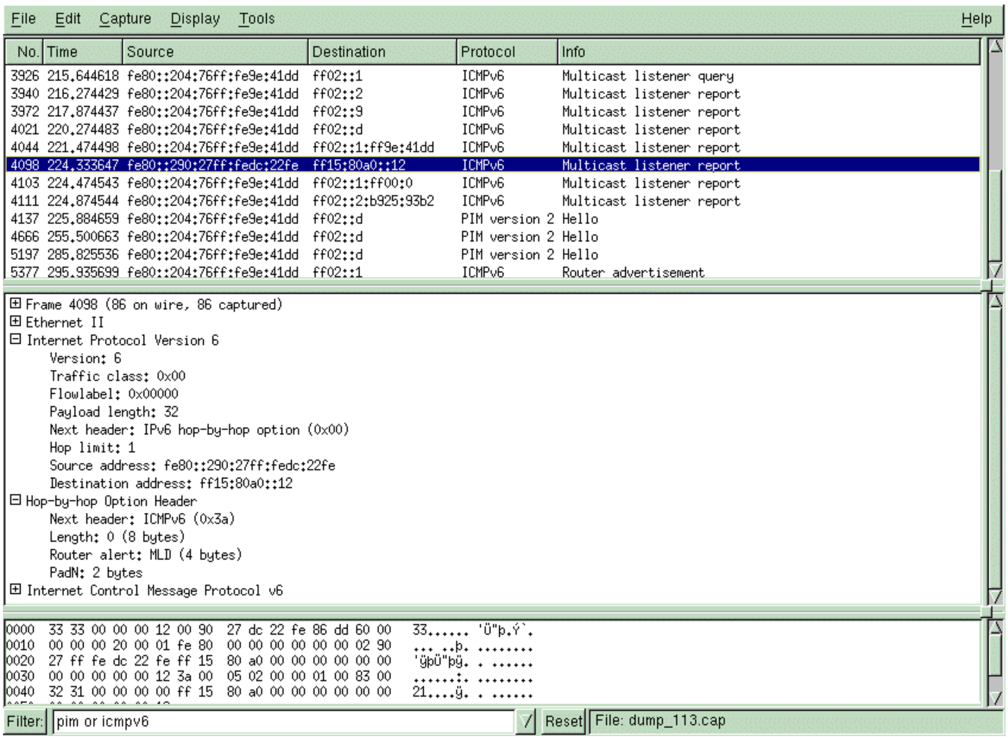


Figure 4-12: Ethereal trace at 2001:7a0:100:113::/64 in scenario 2

The Multicast routing table was displayed using the "pim6stat" command.

The corresponding output is included in 10.2.1. The following table gives a short explanation of each entry from this output:

Multicast Interface Table	Shows all router interfaces with state.
PIM Neighbour List	Lists the neighbouring MC-Router.
MLD Queerer List	Lists every interface corresponding to a neighbouring Multicast enabled network that is cyclically passed by the query packets.
Multicast Routing Table	Shows all participating members (except the redundant ones) and their attributes of the different Multicast groups.
Number of Groups	Number of administrated multicast groups.
RP-address/Upstream	Information of the RP with the upstream interface.)

The whole scenario 2 contains only two MC-Routers which were connected to the subnet 2001:7a0:100:110::/64. Thus each PIM neighbour list contains the interface of the other router. As already described above "mc-flurry" worked as queerer for subnet 2001:7a0:100:110::/64. Hence its interface x10 (*:4228) is displayed in the MLD queerer list. In the other subnets only one router is present in each net. That's why this IPv6 MC router acts automatically as queerer in the corresponding subnet.

The listeners of all different Multicast groups are listed in the Multicast routing table of a MC router. If there should normally appear redundant listeners in the routing table (that means, listeners which are already represented by the registration of an other listener), they will be removed automatically from the table, so that a unique set of MC listeners (without duplications) is represented by this table. Hence this MC routing table could be interpreted as a representation of the MC distribution tree from the local perspective of the corresponding MC router.

For example:

The MC routing table of "mc-flurry" contains no entry for Client 1 because this client is on the way to the MC server and has hence no influence to the Multicast distribution tree behind "mc-flurry".

The MC routing table of the RP router contains in a first step entries from all registered MC clients, which will be pruned out after a certain amount of time. The MC flow is directly sent from the server to these clients (without any further relevance to the RP itself).

4.4.4 Test of Scenario 2: Networks with IPv6 FreeBSD Multicast routers and PIM-DMv6

The objective of this experiment is to deploy multicast on IPv6 networks where all the routers support the IPv6 multicast routing protocol PIM-DM. We intend to verify the messages between the routers during the setting up of the multicast tree and between routers and clients of a multicast session.

Topology

Figure 4-13 shows the topology used. Three PCs with FreeBSD software were configured as IPv6 multicast routers (A, B e C). PCs with Linux RedHat 7.1, Kernel 2.4.2-2 were used as multicast clients. The application used to test the multicast was RAT (Robust Audio Tool).

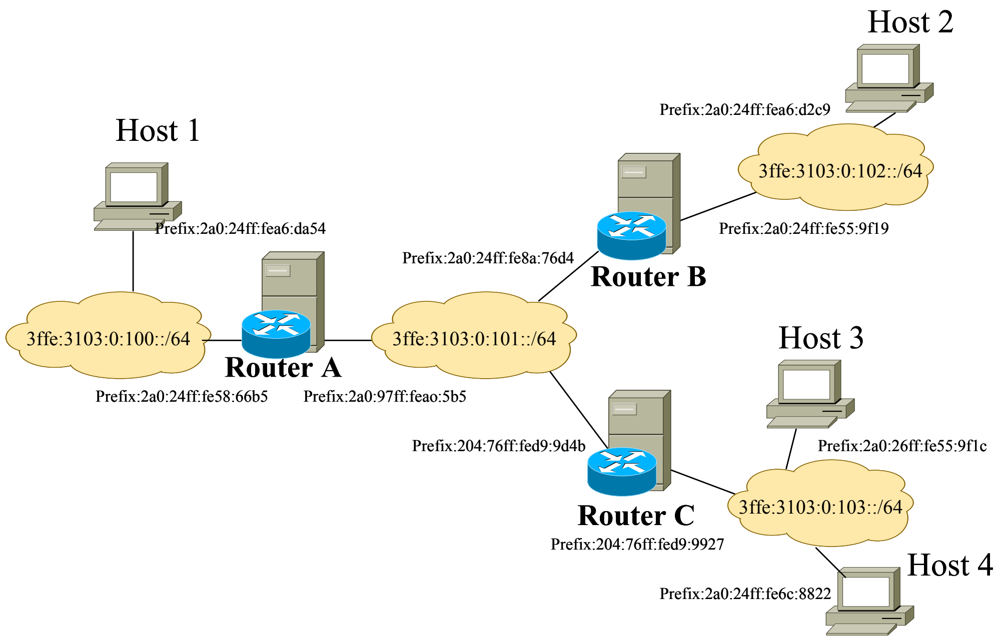


Figure 4-13: IPv6 multicast network with multicast-capable routers

Configuration

The unicast routing protocol used was the RIPng (RIP next generation) based on route6d daemon (`/usr/local/v6/sbin/route6d`). The multicast routing protocol used was the pim6dd (PIM6 dense mode). Router advertisements were configured in all routers.

The configurations are defined in "`/etc/rc.conf`" file. Below, an example of this file is presented for the Router A. The configurations of all the others routers is similar, except the IP6 addresses.

- **Router A**

```
# Enable the IPv6 on the interfaces and define the IPv6 addresses
ipv6_enable="YES"
ipv6_network_interfaces="xl0 vx0"
ipv6_ifconfig_vx0="3ffe:3103:0:100:2a0:24ff:fe58:66b5 prefixlen 64"
ipv6_ifconfig_xl0="3ffe:3103:0:101:260:97ff:fea0:5b5 prefixlen 64"
#
#Enable the IPv6 routing
ipv6_gateway_enable="YES"
#
# Enable the RIPng
ipv6_router_enable="yes"
#
# Enable the router advertisements
rtadvd_enable="YES"
```

On the client machines it is not necessary to configure any address or routes because they receive the "router advertisements" from the routers.

Results

The IPv6 multicast routing protocol (pim6dd) was initialised in all routers to verify the type of messages exchanged between the multicast routers.

The multicast router exchange messages “PIMv2 Hello” used by PIM-DM [6] to detect the presence of the others routers PIM. These messages are sent periodically with interval of 30 seconds from its interfaces, where the PIM is enabled, to the multicast group ALL-PIM-ROUTERS (ff02::d). The “Holdtime” parameter defines the waiting time to receive a new “Hello” message. The following message, Figure 4-14, concerns to the “PIMv2 Hello” sent from the Router A.

```
⊞ Frame 17 (64 on wire, 64 captured)
⊟ Ethernet II
    Destination: 33:33:00:00:00:0d (33:33:00:00:00:0d)
    Source: 00:60:97:a0:05:b5 (00:60:97:a0:05:b5)
    Type: IPv6 (0x86dd)
⊟ Internet Protocol Version 6
    Version: 6
    Traffic class: 0x00
    Flowlabel: 0x00000
    Payload length: 10
    Next header: PIM (0x67)
    Hop limit: 1
    Source address: fe80::260:97ff:fea0:5b5
    Destination address: ff02::d
⊟ Protocol Independent Multicast
    Version: 2
    Type: Hello (0)
    Checksum: 0x42dc (correct)
  ⊟ PIM parameters
      Holdtime: 105
```

Figure 4-14: “PIMv2 Hello” message

The information concerning to multicast connection parameters (PIM6 Dense Mode) is got using the command: “pim6stat -d”. The output of this command on the Router A is presented in Figure 4-15:

```
Multicast Interface Table
 Mif     PhyIF  Local-Address/Prefixlen                          Flags
Neighbors
  0            xl0   3ffe:3103:0:101:260:97ff:fea0:5b5/64     PIM  QRY
fe80::2a0:24ff:fe8a:76d4

fe80::204:76ff:fed9:9d4b
             fe80::260:97ff:fea0:5b5/64
  1            vx0  3ffe:3103:0:100:2a0:24ff:fe58:66b5/64  DR  NO-NBR
QRY
             fe80::2a0:24ff:fe58:66b5/64

MLD Queerer List
 Mif   PhyIF  Address                                            Timer
Last
 0        xl0  fe80::260:97ff:fea0:5b5                             255
4m35s
 1        vx0  fe80::2a0:24ff:fe58:66b5                            255
4m35s

Multicast Routing Table
 Source          Group           Flags
Number of Groups: 0

List of local listener information per interface
```

Figure 4-15: Status by command “pim6stat –d2”

The “Multicast Interface Table” shows that Router A has two active interfaces (vx0 e x10) for the PIM. The interface x10 has two neighbours’ routers (Routers B and C), while the interface vx0 hasn’t neighbours.

The “MLD Queerer List” shows the interfaces that enable MLD (this protocol allows the routers to verify if there are any hosts connected to its interface that want to receive multicast traffic).

The sections “Multicast Routing Table”, “Number of Groups” and “List of Local Listener Information per Interface” are empty because, at this phase, there is not any multicast session yet.

In the next phase, the application RAT is initialised on Host1 for the multicast group ff03::5555 and port 7000, executing the command:

```
rat-4.2.20 ff03::5555/7000
```

The Host1 sends a MLD Report to router denoting the start of a multicast session for the group ff03::5555. As this group isn’t in multicast routing table yet, it is added. The MLD Report respecting to host1 is show in Figure 4-16:

```
⊞ Frame 66 (86 on wire, 86 captured)
⊟ Ethernet II
    Destination: 33:33:00:00:55:55 (33:33:00:00:55:55)
    Source: 00:a0:24:a6:da:54 (3Com_a6:da:54)
    Type: IPv6 (0x86dd)
⊟ Internet Protocol Version 6
    Version: 6
    Traffic class: 0x00
    Flowlabel: 0x00000
    Payload length: 32
    Next header: IPv6 hop-by-hop option (0x00)
    Hop limit: 1
    Source address: fe80::a0:24a6:da54
    Destination address: ff03::5555
⊟ Hop-by-hop Option Header
    Next header: ICMPv6 (0x3a)
    Length: 0 (8 bytes)
    Router alert: MLD (4 bytes)
    PadN: 2 bytes
⊟ Internet Control Message Protocol v6
    Type: 131 (Multicast listener report)
    Code: 0
    Checksum: 0xd5de (correct)
    Maximum response delay: 0
    Multicast Address: ff03::5555
```

Figure 4-16: MLD Report

The MLD Report messages are sent periodically with intervals of 130 seconds.

In this phase, the Router A uses the technique Reverse Path Forwarding (RPF) to verify if there are downstream routers demanding to join for this multicast session. As there are not others routers to send the multicast traffic of this group, the Router A send a “prune” message as it is show in Figure 4-17.

```
⊞ Frame 203 (124 on wire, 124 captured)
⊟ Ethernet II
    Destination: 33:33:00:00:00:0d (33:33:00:00:00:0d)
    Source: 00:60:97:a0:05:b5 (00:60:97:a0:05:b5)
    Type: IPv6 (0x86dd)
⊟ Internet Protocol Version 6
    Version: 6
    Traffic class: 0x00
    Flowlabel: 0x00000
    Payload length: 70
    Next header: PIM (0x67)
    Hop limit: 1
    Source address: fe80::260:97ff:fea0:5b5
    Destination address: ff02::d
⊟ Protocol Independent Multicast
    Version: 2
    Type: Join/Prune (3)
    Checksum: 0xd40b (correct)
  ⊟ PIM parameters
      Upstream-neighbor: fe80::260:97ff:fea0:5b5
      Groups: 1
      Holdtime: 210
    ⊟ Group 0: ff03::5555/128
        Join: 0
      ⊟ Prune: 1
          IP address: 3ffe:3103:0:100:2a0:24ff:fea6:da54/128
```

Figure 4-17: “PIMv2 Prune” message

Also, the routers B and C send a “prune” message to router A because, at this time, they have no any clients for this multicast group

Then, the application RAT was enabled on Host2 connected to Router B. At this time, this routers sends a MLD Report message denoting that intend to join the multicast group ff03::5555. Consequently, the Router B sends a “PIMv2 Graft”, Figure 4-18, message to Router A to inform that wishes to receive traffic for this multicast group.

```
⊞ Frame 285 (124 on wire, 124 captured)
⊟ Ethernet II
    Destination: 00:60:97:a0:05:b5 (00:60:97:a0:05:b5)
    Source: 00:a0:24:8a:76:d4 (00:a0:24:8a:76:d4)
    Type: IPv6 (0x86dd)
⊟ Internet Protocol Version 6
    Version: 6
    Traffic class: 0x00
    Flowlabel: 0x00000
    Payload length: 70
    Next header: PIM (0x67)
    Hop limit: 64
    Source address: fe80::2a0:24ff:fe8a:76d4
    Destination address: fe80::260:97ff:fea0:5b5
⊟ Protocol Independent Multicast
    Version: 2
    Type: Graft (6)
    Checksum: 0x356e (correct)
  ⊟ PIM parameters
      Upstream-neighbor: fe80::260:97ff:fea0:5b5
      Groups: 1
      Holdtime: 0
    ⊟ Group 0: ff03::5555/128
      ⊟ Join: 1
          IP address: 3ffe:3103:0:100:2a0:24ff:fea6:da54/128
        Prune: 0
```

Figure 4-18: “PIMv2 Graft” message

For the building of the multicast tree, the Router A has to send the PIMv2 Graft-Ack message, Figure 4-19, to Router B.

```
⊞ Frame 286 (124 on wire, 124 captured)
⊟ Ethernet II
    Destination: 00:a0:24:8a:76:d4 (00:a0:24:8a:76:d4)
    Source: 00:60:97:a0:05:b5 (00:60:97:a0:05:b5)
    Type: IPv6 (0x86dd)
⊟ Internet Protocol Version 6
    Version: 6
    Traffic class: 0x00
    Flowlabel: 0x00000
    Payload length: 70
    Next header: PIM (0x67)
    Hop limit: 64
    Source address: fe80::260:97ff:fea0:5b5
    Destination address: fe80::2a0:24ff:fe8a:76d4
⊟ Protocol Independent Multicast
    Version: 2
    Type: Graft-Ack (7)
    Checksum: 0x346e (correct)
  ⊟ PIM parameters
      Groups: 0
```

Figure 4-19: "PIMv2 Graft-Ack" message

Later, the RAT was enabled on Host3 that is connected to Router C. The new client informs that it wishes to join the multicast session using a MLD Report message. Then, the router C sends the "PIMv2 Graft" message to Router A to inform that it wishes to receive the multicast traffic for the group ff03::5555. As a reply, the Router A sends the "Graft-Ack" message and, from this moment, the multicast tree for this group is built.

Finally, the fourth client was enabled on Host 4. A MLD Report message is sent from this host to the Router C to join the multicast session. At this stage, there aren't multicast messages between the router because the Router C has already a multicast client connected.

With the 4 clients enabled for the group ff03:5555, the command "pim6stat -d" was executed (to verify the details of the protocol pim6dd) and the netstat –gl (to verify the multicast packets sent) on three routers. The output of this command on Router A is shown in Figure 4-20:

```
Multicast Interface Table
 Mif     PhyIF  Local-Address/Prefixlen                         Flags
Neighbors
  0          xl0  3ffe:3103:0:101:260:97ff:fea0:5b5/64          PIM
fe80::2a0:24ff:fe8a:76d4

fe80::204:76ff:fed9:9d4b
             fe80::260:97ff:fea0:5b5/64
  1         vx0  3ffe:3103:0:100:2a0:24ff:fe58:66b5/64     DR  NO-NBR
QRY
             fe80::2a0:24ff:fe58:66b5/64

MLD Queerer List
 Mif   PhyIF Address                                             Timer
Last
 0        xl0  fe80::204:76ff:fed9:9d4b                            245
12m39s
 1        vx0  fe80::2a0:24ff:fe58:66b5                            255
18m36s

Multicast Routing Table
 Source          Group           Flags
---------------------------(S,G)----------------------------
 3ffe:3103:0:100:2a0:24ff:fea6:da54 ff03::5555      SG
Pruned   oifs: ..
Asserted oifs: ..
Filtered oifs: ..
Leaves   oifs: .l
Outgoing oifs: o.
Incoming     : .I
Upstream nbr: NONE

TIMERS:  Entry   Prune VIFS:   0   1
           205                 0   0
---------------------------(S,G)----------------------------
 3ffe:3103:0:102:2a0:24ff:fea6:d2c9 ff03::5555      SG
Pruned   oifs: ..
Asserted oifs: ..
Filtered oifs: ..
Leaves   oifs: .l
Outgoing oifs: .o
Incoming     : I.
Upstream nbr: fe80::2a0:24ff:fe8a:76d4

TIMERS:  Entry   Prune VIFS:   0   1
           205                 0   0
---------------------------(S,G)----------------------------
 3ffe:3103:0:103:2a0:24ff:fe55:9f1c ff03::5555      SG
Pruned   oifs: ..
Asserted oifs: ..
Filtered oifs: ..
Leaves   oifs: .l
Outgoing oifs: .o
Incoming     : I.
Upstream nbr: fe80::204:76ff:fed9:9d4b

TIMERS:  Entry   Prune VIFS:   0   1
           205                 0   0
---------------------------(S,G)----------------------------
 3ffe:3103:0:103:2a0:24ff:fe6c:8222 ff03::5555      SG
Pruned   oifs: ..
Asserted oifs: ..
Filtered oifs: ..
```

```
Leaves   oifs: .l
Outgoing oifs: .o
Incoming     : I.
Upstream nbr: fe80::204:76ff:fed9:9d4b

TIMERS:  Entry   Prune VIFS:   0   1
           205                 0   0
Number of Groups: 1

List of local listener information per interface
  Mif 1(vx0)
    ff03::5555 timeout: 145
```

Figure 4-20: "pim6stat –d" on Router A

As shows the output of this command, there are four multicast sources for the same group.

Figure 4-21 shows an estimate of multicast packets sent.

```
IPv6 Multicast Interface Table
 Mif   Rate   PhyIF   Pkts-In   Pkts-Out
   0      0     xl0       135         85
   1      0     vx0       127        135

IPv6 Multicast Forwarding Cache
 Origin                                    Group       Packets Waits
In-Mif  Out-Mifs
 3ffe:3103:0:103:2a0:24ff:fe55:9f1c       ff03::5555       10       0
0       1
 3ffe:3103:0:103:2a0:24ff:fe6c:8222       ff03::5555       45       0
0       1
 3ffe:3103:0:102:2a0:24ff:fea6:d2c9       ff03::5555       80       0
0       1
 3ffe:3103:0:100:2a0:24ff:fea6:da54       ff03::5555       84       0
1       0
```

Figure 4-21: "netstat" on Router A

4.4.5 Test of Scenario 2: Networks with IPv6 Hitachi Multicast routers and PIM-SMv6

These tests aim to evaluate several IPv6 multicast functionalities in Hitachi routers using a local environment. The functionalities tested include MLD, PIM-SM and multicast packets forwarding. As can be seen in Figure 4-22, the implemented scenario involves two IPv6 multicast capable routers joining all involved equipment.

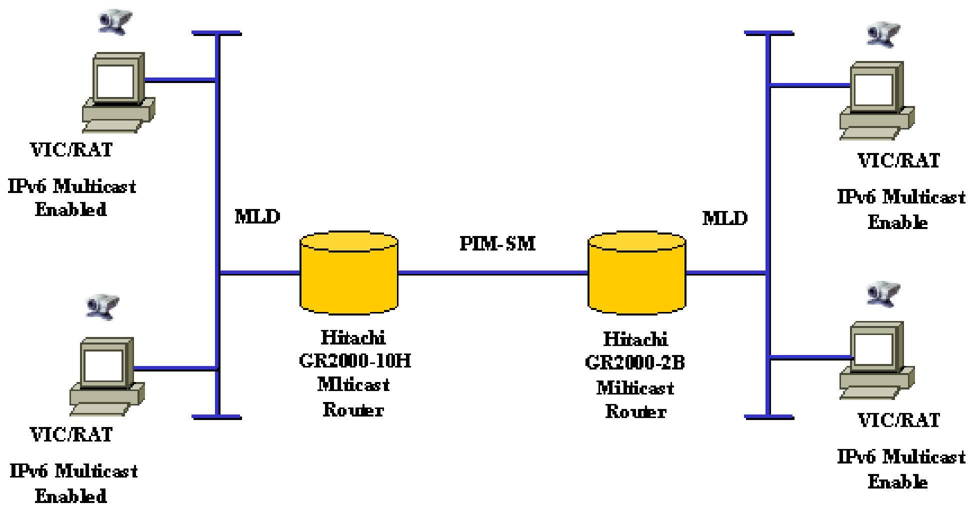


Figure 4-22: Multicast local tests, 2nd scenario

Figure 4-23 resumes the main tests performed in the previous scenario, the purpose of each test, and the results obtained.

Test	Purpose	Results
PIM-SM	Verify the correct performance of IPv6 PIM-SM	Correct performance in a local environment
MLD	Verify the correct performance of IPv6 MLD	Test finished. Result analysis is underway
Multicast delivery	Verify that the packets sent to a multicast IPv6 address are delivered to all hosts with that multicast address.	Test finished. Result analysis is underway

Figure 4-23: Local tests results, 2nd scenario

4.4.6 Test of Scenario 2: Networks with IPv6 Not-Multicast routers

The purpose of this test is to deploy the IPv6 multicast where only some routers support IPv6 multicast. In these cases, the solution is to use tunnels between multicast routers.

Topology

As it is shown in Figure 4-24, two PCs were configured as multicast routers (Router A e C). Another PC was configured as IPv6 unicast router (Router B). An IPv6 over IPv6 tunnel was established between the IPv6 multicast routers. The application used to multicast tests was the rat-4.2.20, which was installed on multicast client hosts.

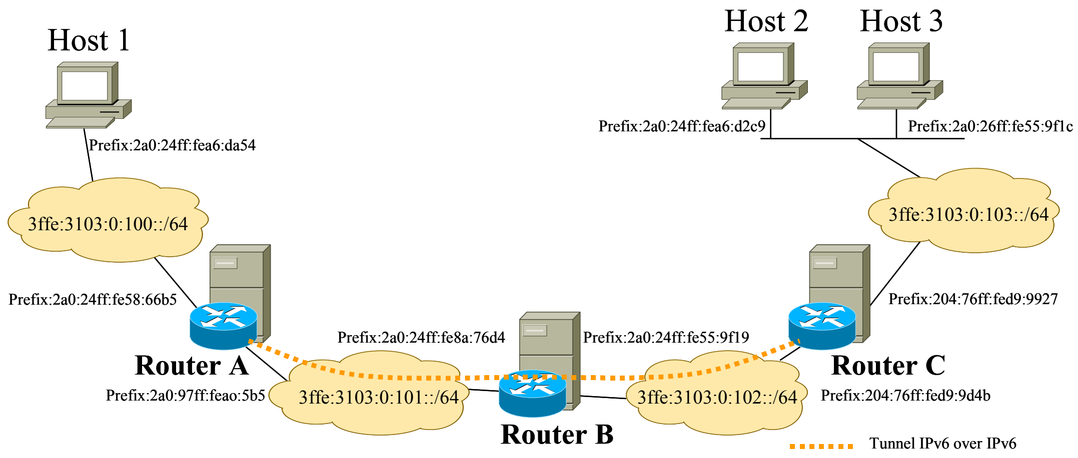


Figure 4-24: Multicast tunnels

Configuration

The unicast routing protocol used was RIPng and IPv6 multicast protocol was "pim6dd" (PIM6 dense mode). All routers were configured with the option to send "router advertisements". The IPv6 addresses on the routers were configured in the file `/etc/rc.conf`, as already described. Additional configurations were necessary to deploy the IPv6 multicast over tunnels.

Router A:

- Configuration of the IPv6 tunnel over IPv6 (gif0):

```
ifconfig gif0 create
ifconfig gif0 inet6 aaaa:1111::1 aaaa:1111::2 prefixlen 128
gifconfig gif0 inet6 3ffe:3103:0:101:260:97ff:fea0:5b5
                     3ffe:3103:0:102:204:7dff:fed9:9d4b
ifconfig gif0 up
```

- Disable the multicast on interface of the not-multicast network

To disable the multicast on interface x10 (connected to 3ffe:3103:0:101::/64), it is necessary to add in file "`/usr/local/v6/etc/pim6dd.conf`" the following line:

```
phyint xl0 disable
```

- Run the pim6dd daemon to enable the IPv6 multicast

```
/usr/local/v6/sbin/pim6dd –c /usr/local/v6/etc/pim6dd.conf
```

Router C:

- Configuration of the IPv6 tunnel over IPv6 (gif0)

```
ifconfig gif0 create
ifconfig gif0 inet6 aaaa:1111::2 aaaa:1111::1 prefixlen 128
gifconfig gif0 inet6 3ffe:3103:0:102:204:7dff:fed9:9d4b
                     3ffe:3103:0:101:260:97ff:fea0:5b5
ifconfig gif0 up
```

- Disable the multicast on interface of the not-multicast network

To disable the multicast on interface x10 (connected to 3ffe:3103:0:102::/64)

```
phyint xl0 disable
```

- Run the pim6dd daemon to enable the IPv6 multicast

```
/usr/local/v6/sbin/pim6dd -c /usr/local/v6/etc/pim6dd.conf
```

Results

The application rat-4.2.20 was run on all hosts for the multicast group ff03::5555 and port 7000. A multicast session was established in all clients.

4.4.7 Test of Scenario 2: IPv6 Multicast over IPv4/IPv6 Mixed Networks

The objective this experiments is to deploy the IPv6 multicast on IPv6/IPv6 mixed networks. In this case, IPv6 over IPv6 tunnels are configured.

Topology

The test scenario is shown in Figure 4-25. As in the previous tests, 3 PCs were configured to be used as IPv6 multicast routers. Also, the application used to tests was the RAT

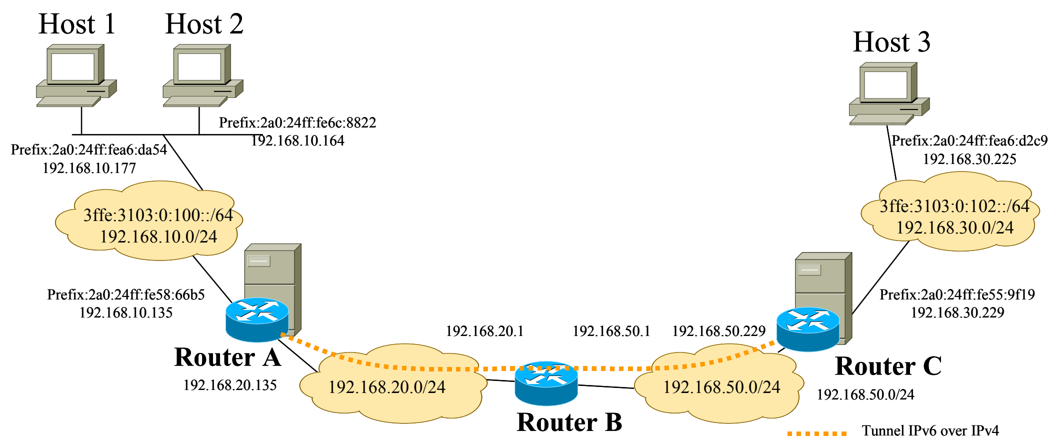


Figure 4-25: IPv6 multicast over IPv6/IPv4 mixed network

Configuration

For this test, static routes were used to route the IPv4 traffic. In the file `/etc/rc.conf` was configured the IPv4 routes, the IPv4 and IPv6 addresses, the option to enable the "router advertisements" and to enable IPv6 unicast routing protocol (RIPng). Below, the relevant configurations are presented.

Router A:

- In file /etc/rc.conf:

```
#Default route
defaultrouter="192.168.20.1"
#
#Enable IPv4 routing
gateway_enable="YES"
#
#IPv4  addresses
ifconfig_xl0="inet 192.168.20.135  netmask 255.255.255.0"
ifconfig_vx0="inet 192.168.10.135  netmask 255.255.255.0"
#
#Enable the IPv6 router
ipv6_gateway_enable="YES"
#
# IPv4  addresses
ipv6_enable="YES"
ipv6_network_interfaces="vx0"
ipv6_ifconfig_vx0="3ffe:3103:0:100:2a0:24ff:fe58:66b5 prefixlen 64"
#
# Enable the router advertisements
rtadvd_enable="YES"
#
# Enable the RIPng
ipv6_router_enable="yes"
```

- Configuration of the IPv6 tunnel over IPv4 (gif0)

```
 ifconfig gif0 create
 ifconfig gif0 inet6 abcd:1111::1 abcd:1111::2 prefixlen 128
 gifconfig gif0 inet 192.168.20.135 192.168.50.229
 ifconfig gif0 up
```

- Disable theIPv6 multicast on x10 (connected to network 192.168.20.0)
  Edit the file "/usr/local/v6/etc/pim6dd.conf" and add

```
phyint xl0 disable
```

- Run the daemon pim6dd

```
/usr/local/v6/sbin/pim6dd -c /usr/local/v6/etc/pim6dd.conf
```

Router C:

- In the file `/etc/rc.conf` add:

```
#Default route
defaultrouter="192.168.50.1"
#
#Enable IPv6 routing
gateway_enable="YES"
#
#IPv4 addresses
ifconfig_vx0="inet 192.168.50.229  netmask 255.255.255.0"
ifconfig_xl0="inet 192.168.30.229  netmask 255.255.255.0"
#
# IPv6 addresses
ipv6_enable="YES"
ipv6_network_interfaces="xl0"
ipv6_ifconfig_xl0="3ffe:3103:0:102:2a0:24ff:fe55:9f19 prefixlen 64"
#
# Enable IPv6 routing
ipv6_gateway_enable="YES"
# Enable Routers Advertisements
rtadvd_enable="YES"
#
# Enable RIPng
ipv6_router_enable="YES"
```

- Configuration of the IPv6 tunnel over IPv4 (gif0)

```
ifconfig gif0 create
ifconfig gif0 inet6 abcd:1111::2 abcd:1111::1 prefixlen 128
gifconfig gif0 inet 192.168.50.229 192.168.20.135
ifconfig gif0 up
```

- Disable the IP multicast on the interface vx0 (connected to network 192.168.50.0)
  Edit the "`/usr/local/v6/etc/pim6dd.conf`" and add:

```
phyint vx0 disable
```

- Run the daemon pim6dd

```
/usr/local/v6/sbin/pim6dd -c /usr/local/v6/etc/pim6dd.conf
```

<u>Results</u>

After being configured the IPv6 over IPv4 tunnel and launched the daemons "pim6dd", the application rat-4.2.20 was executed to group ff03::7777 on 3 clients. A multicast session was established between the 3 clients.

4.4.8 Test of Scenario 3

<u>Topology</u>

The following scenario illustrates how a Multicast connection between two different L2 peering customers could be established passing an IPv6 IX (see Figure 4-26).

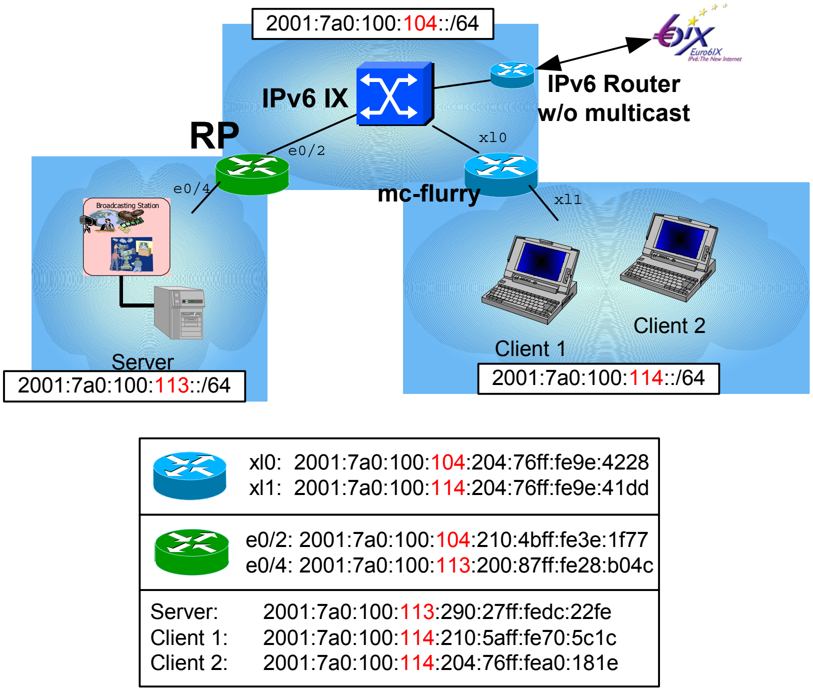


Figure 4-26: Structure of the IPv6 Multicast Scenario 3

Scenario 3 uses the same clients and the router “mc-flurry” as scenario 2. Additionally a Hitachi router GR-2000 was installed, serving as the RP of Scenario 3.

From the perspective of the Euro6IX project this scenario demonstrates, how an IPv6 Service Provider could offer additional, new IPv6 services based on an IPv6 IX infrastructure.

The left subnet of the figure above could be considered as an IPv6 MC service network which offers IPv6 MC streaming services and an IPv6 RP service to the customers of the IX (right subnet with “mc-flurry” as Border router).

Configuration

End system:	Multicast Server: PC Pentium 3 with 128 MB Ram with Hauppauge Video (WinTV) and SoundBlaster Card (Linux Redhat 7.2)
	Multicast Client: 3 PC Pentium 3 with 128 MB Ram and SoundBlaster card (Linux SuSE 8.0 Kernel 2.4.18)
	Router (RP): Hitachi GR2000-4S (06-05-/C ROUTE-OS6-X)
	Router (mc-flurry): PC Pentium 3 with 128 MB Ram (FreeBSD)
Network:	10Mbit/s Ethernet
	Multicast group ff15:08a0::12
Application:	VIC Version 2.8ucl-1.1.3
	RAT Version 4.2.19

After successfully configuring the Unicast routing on the Hitachi router, the following four commands were entered on the router's command line to configure the IPv6 Multicast routing:

```
config: pim6 yes
config: pim6 sparse interface EURO6IX-MC-1 IPv6-IX -enable
config: pim6 sparse candidate-rp group ff15::/16
config: pim6 sparse candidate-bsr
```

"show pim6" displays the configuration of the Hitachi IPv6 MC RP:

```
DURIN/config: show pim6
        pim6 yes {
                sparse {
1                       interface EURO6IX-MC-1 IPv6-IX {
                                enable;
                        };
                        candidate-rp yes {
                                group {
                                        ff15::/16;
                                };
                        };
                        candidate-bsr yes;
                };
        };
```

Results

In this scenario the IPv6-IX subnet (2001:7a0:100:104::/64) contains no Multicast listener.

Therefore there exists no answer to the Multicast listener query (from interface *:b04a of the RP) regarding the Multicast Address ff15:80a0::12 within the network dump below.

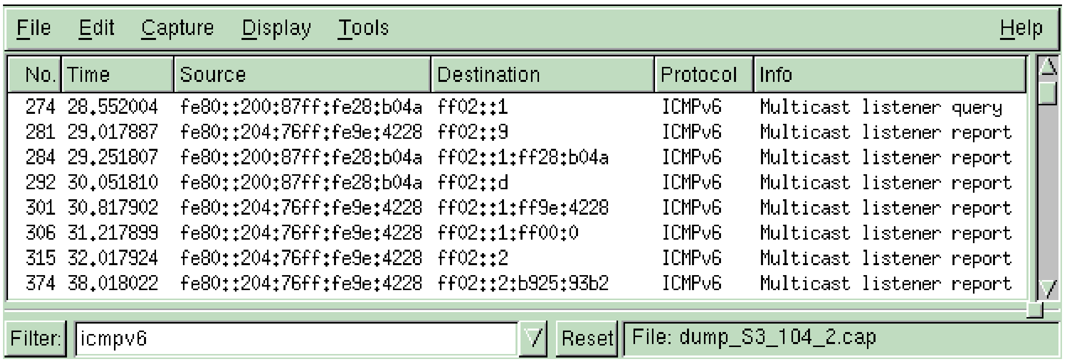
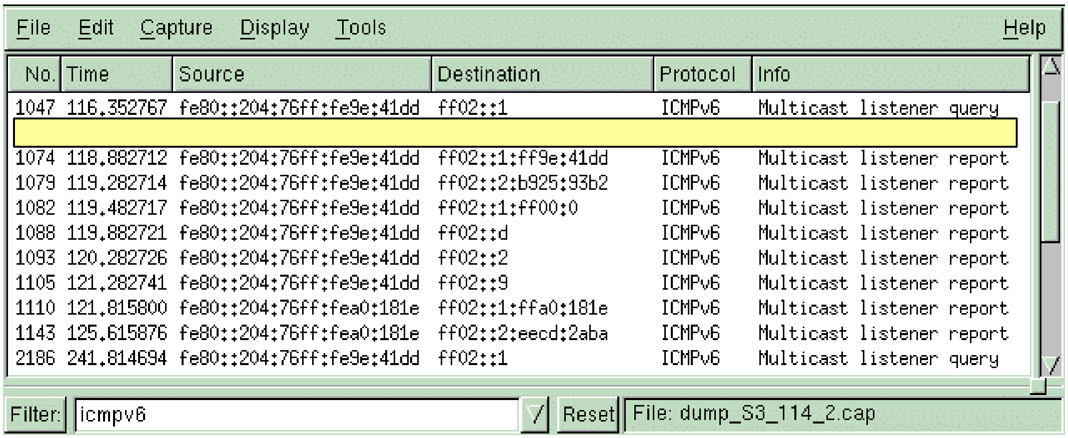


Figure 4-27: Ethereal trace at 2001:7a0:100:104::/64 in scenario 3

There were two listeners in subnet 2001:7a0:100:114::/64. In figure 6 only Client 1 (*:5c1c) answered the query. The report was sent by Client 1 to Multicast address ff15:80a0::12 and therefore also received by Client 2. Thus Client 2 decided not to send any further reports because the subnet 2001:7a0:100:114::/64 is already represented by Client 1.



File Edit Capture Display Tools Help

No.	Time	Source	Destination	Protocol	Info
1047	116.352767	fe80::204:76ff:fe9e:41dd	ff02::1	ICMPv6	Multicast listener query
1074	118.882712	fe80::204:76ff:fe9e:41dd	ff02::1:ff9e:41dd	ICMPv6	Multicast listener report
1079	119.282714	fe80::204:76ff:fe9e:41dd	ff02::2:b925:93b2	ICMPv6	Multicast listener report
1082	119.482717	fe80::204:76ff:fe9e:41dd	ff02::1:ff00:0	ICMPv6	Multicast listener report
1088	119.882721	fe80::204:76ff:fe9e:41dd	ff02::d	ICMPv6	Multicast listener report
1093	120.282726	fe80::204:76ff:fe9e:41dd	ff02::2	ICMPv6	Multicast listener report
1105	121.282741	fe80::204:76ff:fe9e:41dd	ff02::9	ICMPv6	Multicast listener report
1110	121.815800	fe80::204:76ff:fea0:181e	ff02::1:ffa0:181e	ICMPv6	Multicast listener report
1143	125.615876	fe80::204:76ff:fea0:181e	ff02::2:eecd:2aba	ICMPv6	Multicast listener report
2186	241.814694	fe80::204:76ff:fe9e:41dd	ff02::1	ICMPv6	Multicast listener query

Filter: icmpv6 Reset File: dump_S3_114_2.cap

Figure 4-28: Ethereal trace at 2001:7a0:100:114::/64 in scenario 3

The state information of the RP and the Multicast routing table of "mc-flurry" are attached in appendix 10.2.2.

Hence Scenario 3 was successfully realised and no problems were observed.

4.4.9 Test of Scenario 4

At the moment of writing this contribution the tests for the fourth IPv6 Multicast scenario are implemented and performed.

The following picture gives a short overview about the investigated test scenario that will be described in more detail within the activity descriptions of Semester 3.

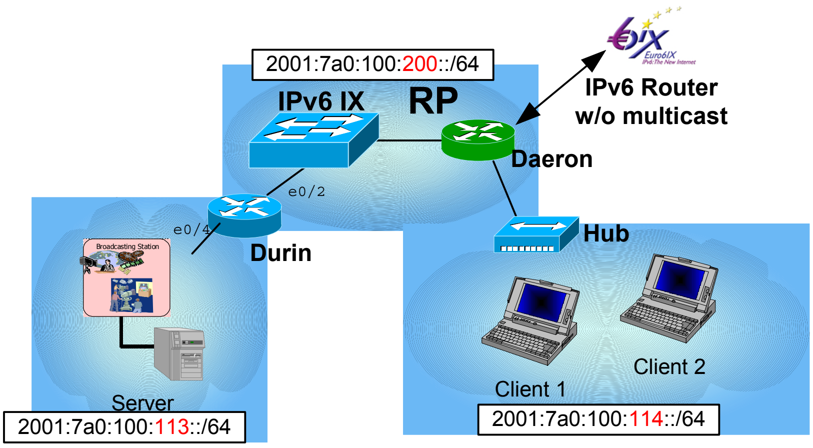


Figure 4-29: Structure of the IPv6 Multicast Scenario 4

The tests of this scenario are not finished yet.

4.4.10 Test of Scenario 5

Not implemented or finished yet.

4.5 Identified open issues / Topics of future work

During the definition of these different test scenarios several open issues were identified, where more work has to be contributed. The following bullets specify only a few of them:

- What's the right place for the RP?
- Are there any interoperations between RP and RR?
- Which applications to use besides these mentioned above?
- (When) Are there some implementations for PIM-SSMv6 and MLDv2 available?
- Is the investigation of IPv6 Multicast tunnel scenarios within the scope of the project?
- Which functionality should the L2 switches within the IXs have in order to allow an native IPv6 Multicast service?
- What could a Business Case be? (IPv6 Multicast [RP] as L3 service)
- Is it realistic that IX customers want to have also L3 services when they are connected to an IX?
- Are other IPv6 Multicast Routing protocols, which could be of interest in the project?
- What about IPv6 Multicast address assignment? (Malloc WG IETF; Scope problem etc.)
- Are there other scenarios, which should be investigated within the Euro6IX project?
- What about IPv6 Multicast implementations for other operating systems?
- Are there any MSDPv6 implementations / standardisation work in the pipe?
- Will be any BGMP implementatios?
- What happens in networks with old Ethernet switches that do not forward IPv6 link layer multicast packets?
- Would be interesting to implement MLD snooping in Ethernet switches for those cases where multicast traffic is too heavy?

This list will change correspondingly to the development of the network infrastructure and router implementations as well as in the context of defining more / new test scenarios for IPv6 within the Euro6IX project.

5. M6BONE EXPERIMENTS, 6NET AND EURO6IX

The Euro6IX project is seeking to build its own IPv6 multicast infrastructure. However, the lack of availability of PIM-SM or PIM-SSM support across the Euro6IX network at this stage of the project means that IPv6 multicast connectivity, as it often the case with IPv4, has to be provisioned through unicast tunnels, either IPv6 multicast in IPv4 unicast or in IPv6 unicast tunnels.

5.1 IPv6 multicast between Euro6IX and 6NET

UoS is jointly a participant in Euro6IX and 6NET. The bulk of its IPv6 multicast work is being carried out in the 6NET project, but UoS is reporting it here in Euro6IX as well because there is a strong relationship between the two projects. Joint collaborative activities have been defined in Deliverable 7.9 of Euro6IX, which is also reported as Deliverable D5.3 of the 6NET project. One such joint activity is IPv6 multicast.

It is thus highly desirable to establish IPv6 multicast connectivity between the Euro6IX and 6NET partners. Currently 6NET also has no native IPv6 multicast connectivity on its backbone network because no PIM-SM images are yet available for the Cisco GSR routers (it is on their roadmap for 2003). As a result, the M6Bone network seems an ideal vehicle for IPv6 multicast experiments until native multicast becomes available in each project.

5.2 The M6Bone

M6bone is an experimental IPv6 multicast network. The main purpose is to offer IPv6 multicast connectivity between IPv6 sites. This service is based on Renater3 [10] (IPv6 enable network), and benefits from the logistic support of the Aristote association which is involved in the broadcasting of the ultra-modern technologies and of the G6, French group of IPv6 testers. The M6Bone has evolved since 2001 with discussion in the TERENA task force TF-NGN working groups. The first objective is to develop an advanced service on IPv6, in order to participate in the promotion of the protocol. It enables to use multicast videoconference tools on the network in order to broadcast events.

At the time of writing, the network topology is the one shown on Figure 1-1 [11].

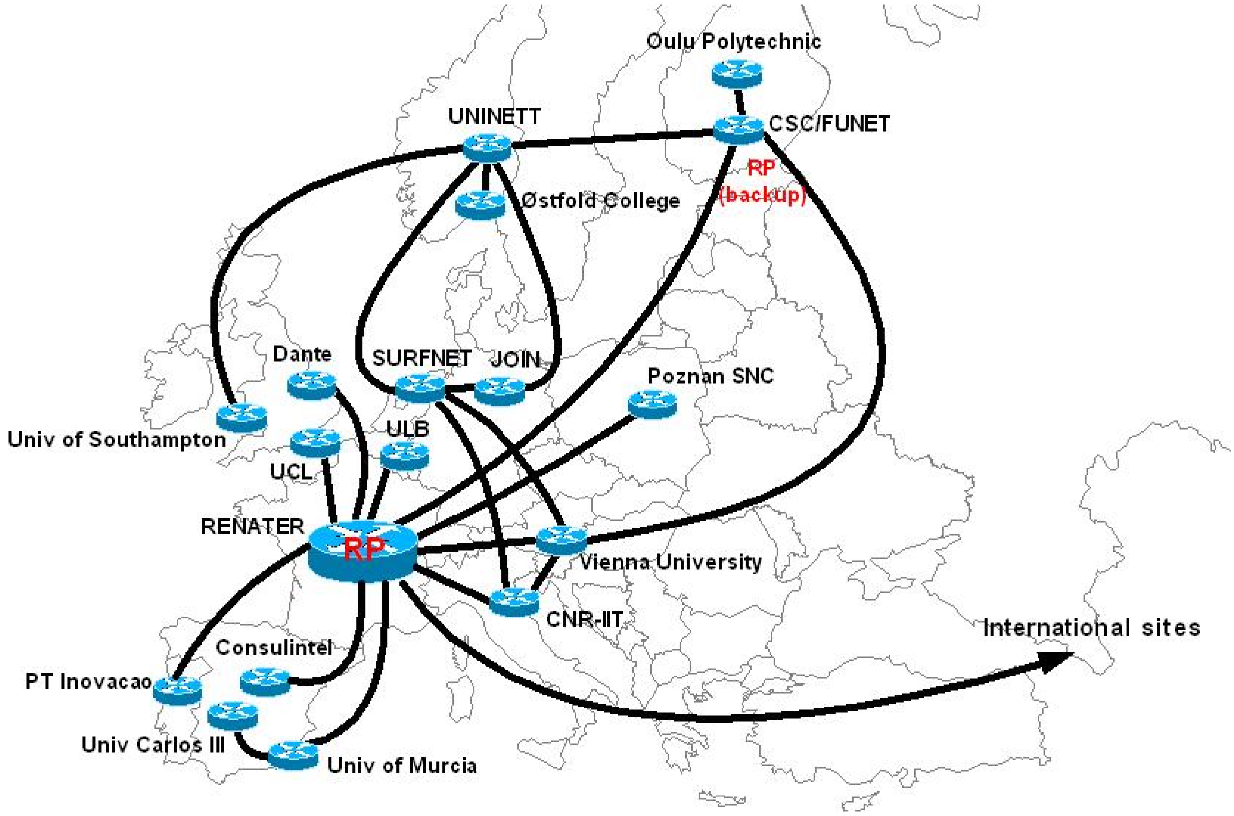


Figure 5-1: The M6Bone sites in Europe (French sites not included), December 2002

The M6Bone has over 35 participants, and during 2002 partners have joined the network from both the 6NET and Euro6IX projects. The 6NET participants include UoS, UCL, DANTE, ULB, JOIN, UNINETT and CSC/FUNET, while Euro6IX partners include Consulintel and the University of Murcia.

M6Bone is implemented with multicast-capable routers interconnected by IPv6 over IPv6 tunnels or IPv6 over IPv4 tunnels. The IPv6 multicast routing protocol deployed on the M6Bone is the PIM Sparse-Mode [5]. The requirement for the operation of a multicast based on this protocol, is the existence of a Rendezvous Point (RP). This RP is hosted in Renater's multicast router. There is not method in IPv6 for inter-domain PIM-SM to operate through multiple RPs yet (MSDP exists for IPv4, but is a "stop-gap" measure that is not likely to be taken forward to IPv6, while BGMP is still also in its implementation infancy). It is possible that PIM-SSM may become the way forward for multicast services in IPv6.

The M6Bone is used for a variety of multicast applications including:

- VIC and RAT for videoconferencing (Linux, BSD and Windows clients)
- NTE whiteboard sharing
- VideoLAN and other video streaming
- Icecast, CRadio and other MP3 streaming (including Trondheim Underground Radio)

Many of the French Aristote institute seminars are transmitted on the M6Bone.

5.2.1 M6Bone at IST2002

At the IST 2002 conference in Copenhagen in November 2002, a Linux Redhat PC was set up at the IPv6 Cluster booth receiving the M6Bone "6NET people" multicast group that reached over 20 participants, using IPv6-enabled versions of vic and rat. Because no M6Bone router was installed at the conference venue, video and audio was streamed from the booth via an IPv6 unicast to multicast reflector hosted at UNINETT (topologically close to the event, which was connected for IPv6 via NORDUnet). Other reflectors, including an IPv4 to IPv6 multicast gateway, also exist.

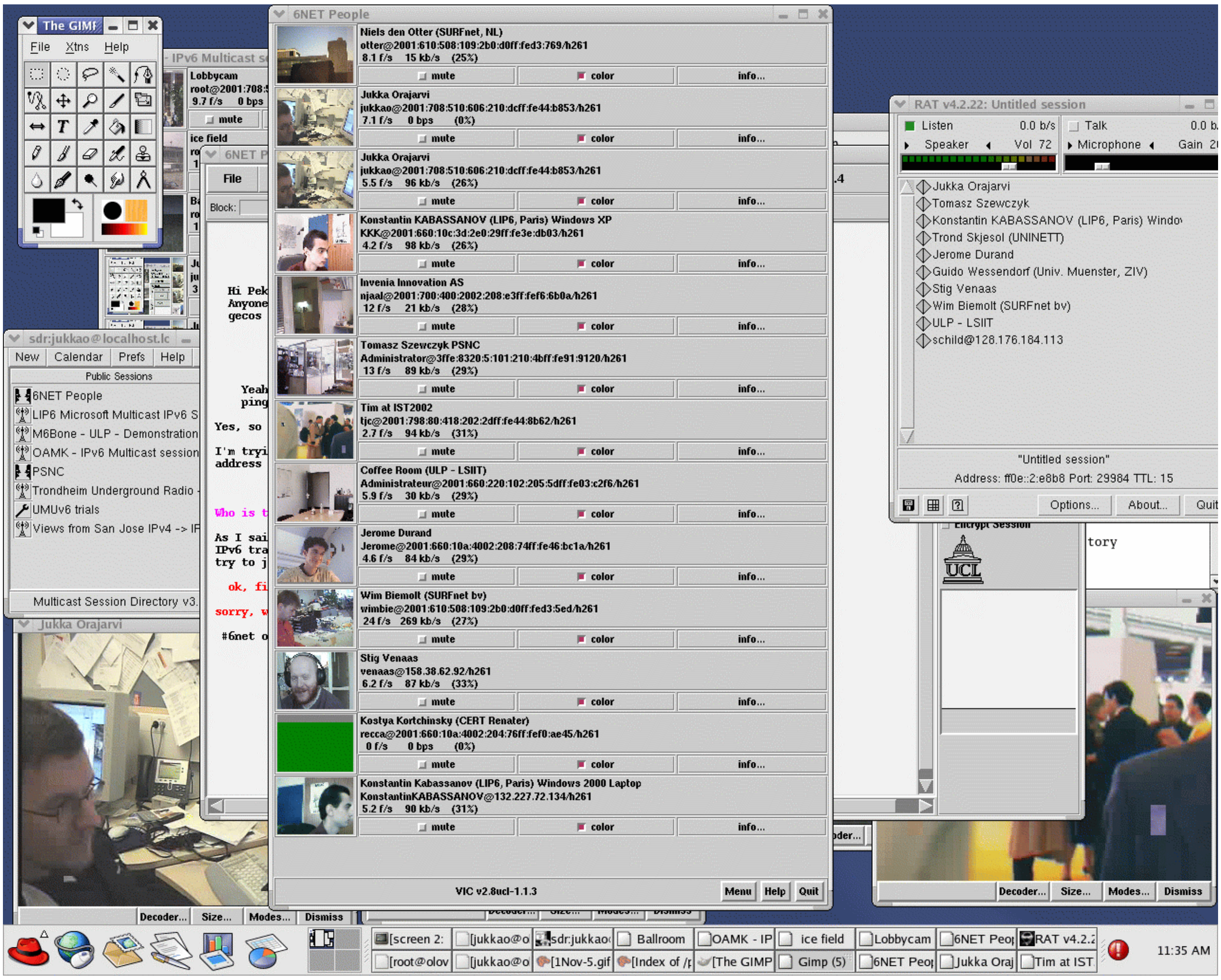


Figure 5-2: The M6Bone in use (vic, rat and nte) at the IPv6 Cluster booth at IST2002

5.3 How to connect Euro6IX to M6Bone

The Euro6IX partners can establish a connection to the M6bone through, at least, three different ways. The different approaches are suitable to live together on the Euro6IX backbone. However, a global strategy could be reached in order to get all the partners connected to the M6bone as well between each other's.

These three approaches are the following:

- IPv6 over IPv4 tunnel to the M6bone multicast router (1).
- IPv6 over IPv6 tunnel to the M6bone multicast router (2).
- IPv6 over IPv6 tunnels between Euro6IX partners, connected to the M6bone (3).

The approach (1) is the simplest one. An IPv6 over IPv4 tunnel is established over the Internet to the central M6bone router. This is a connection similar to the 6to4 access to the 6bone. The disadvantage of this solution is that the accessing Multicast router must have a global IPv4 address. Furthermore, this solution uses IPv4 tunnelling, what must be avoided as much as possible on the project. Hence, this solution must be not taken into account.

The second approach (2) consists on the establishment of an IPv6 over IPv6 tunnel to the main M6bone router. In this case, there is no need for a global IPv4 addressing neither for an IPv4 tunnelling. In this approach the Internet is also not used to drive the traffic to the M6bone router, but the Euro6IX backbone. However, this assumes that native IPv6 connectivity exists between the M6bone router and the Euro6IX backbone.

The third approach (3) consists on the establishment of IPv6 tunnels between the partners, with just one or some of them connected to the M6bone main router (using either (1) or (2) approaches). This seems to be the best approach for the long time, however, some aspects must be well specified like, how many partners and who have the responsibility to be connected to M6bone. In a first approach it seems that one connection per IX could be a good policy. This solution gives also the possibility of a smoother translation to a pure (no tunnels based) Multicast network on the Euro6IX backbone.

5.4 Initial tests in Euro6IX with the M6Bone

As it has been said, although the plan to connect Euro6IX to M6Bone has not been decided yet, some partners are starting to experiment in that network.

5.4.1 UoS

UoS is one of the M6Bone participant sites. The UoS site topology is shown in the figure. Native IPv6 multicast is run across our FreeBSD router hierarchy, which has IPv6 subnets feeding over 300 dual-stack capable end systems.

An example of the FreeBSD router configuration is as follows:

```
ipv6_enable="YES"
ipv6_gateway_enable="YES"
ipv6_router_enable="YES"

ipv6_ifconfig_dc0="2001:630:d0:111::2 prefixlen 64"
ipv6_ifconfig_dc1="2001:630:d0:132::2 prefixlen 64"

ipv6_static_routes="default"
ipv6_route_default="default fe80::280:c8ff:feb9:a8b9%dc0"
```

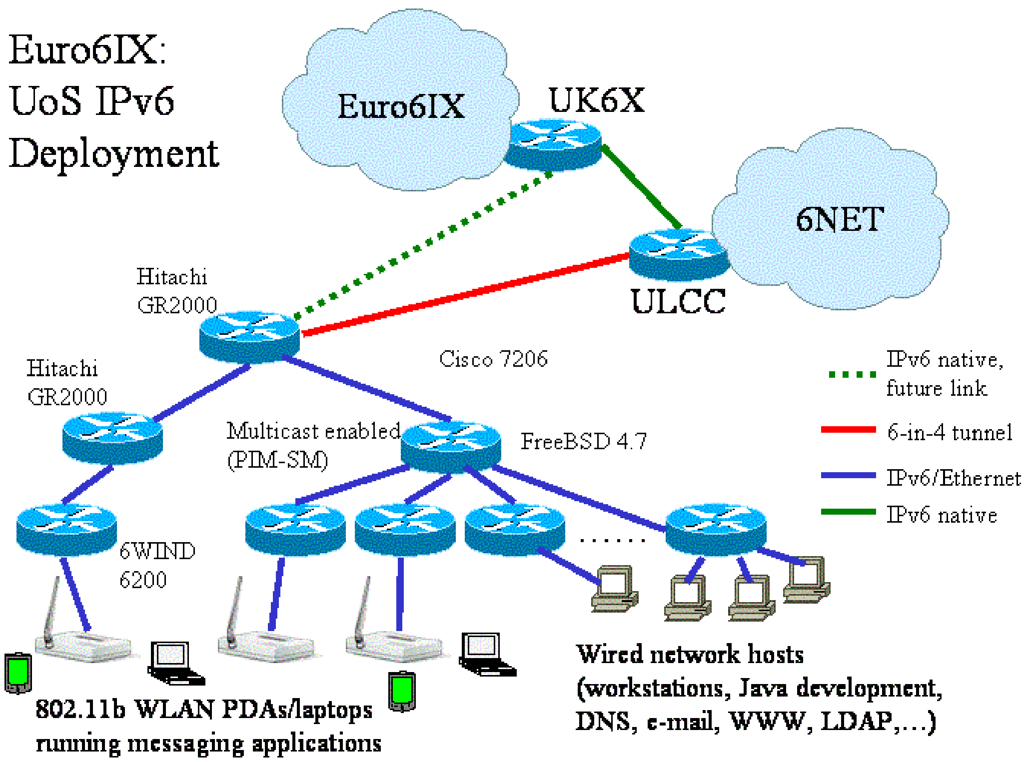


Figure 5-3: UoS Euro6IX infrastructure including PIM-SM hierarchy

An example of the M6Bone-specific configuration is as follows:

```
ipv6_router="/usr/local/v6/sbin/route6d"
ipv6_router_flags="-N dc0,dc1"

gif_interfaces="gif0"
gifconfig_gif0="152.78.65.67 158.38.62.72"
ipv6_ifconfig_gif0="3ffe:2a00:100:7efb::2 prefixlen 64"

mroute6d_enable="YES"
mroute6d_program="/usr/local/v6/sbin/pim6sd"
```

UoS plans to continue to operate its own IPv6 multicast infrastructure, to participate actively in the M6Bone, and to join the Euro6IX and 6NET native multicast infrastructures, as they become available during the lifetime of both projects. Specific new technology will also be tested as it becomes available, e.g. PIM-SSM and MLDv2 implementations.

5.4.2 PTIN Experience

PTIN is connected to M6bone using an IPv6 over IPv6 tunnel through the Euro6IX backbone. The multicast router where the tunnel is terminated consists in a PC running FreeBSD OS. The configuration procedures are presented in Appendix A.4. The connection scenario is shown Figure 5-4.

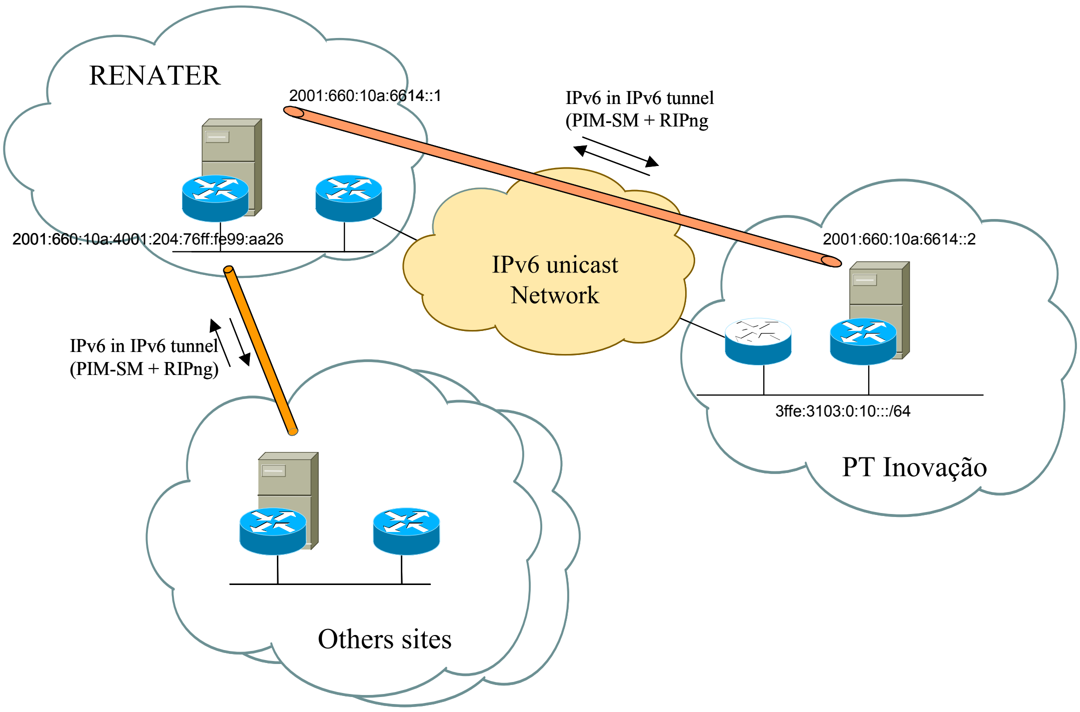


Figure 5-4: PT Inovação connection to M6Bone

At time of writing, intensive tests have not been performed yet. However, the objective is to get experience about multicast, participating in multicast events, like multicast videoconference, among others.

5.4.3 Consulintel Experience

The Figure 5-5 shows the topology used in a M6Bone connection. The goal is to accomplish this connection. At the time of writing, the setting up of the multicast link between Consulintel and Renater is underway, the remaining issue is the tune up of PIM-SM function.

Regarding the IPv6 multicast enabled router used in this testbed, the aim is to use and evaluate a Hitachi router in a first step and then a FreeBSD “router” in a second step. Currently, the platform used is Hitachi GR2000. The functionalities tested include PIM-SM and MLD in a local environment.

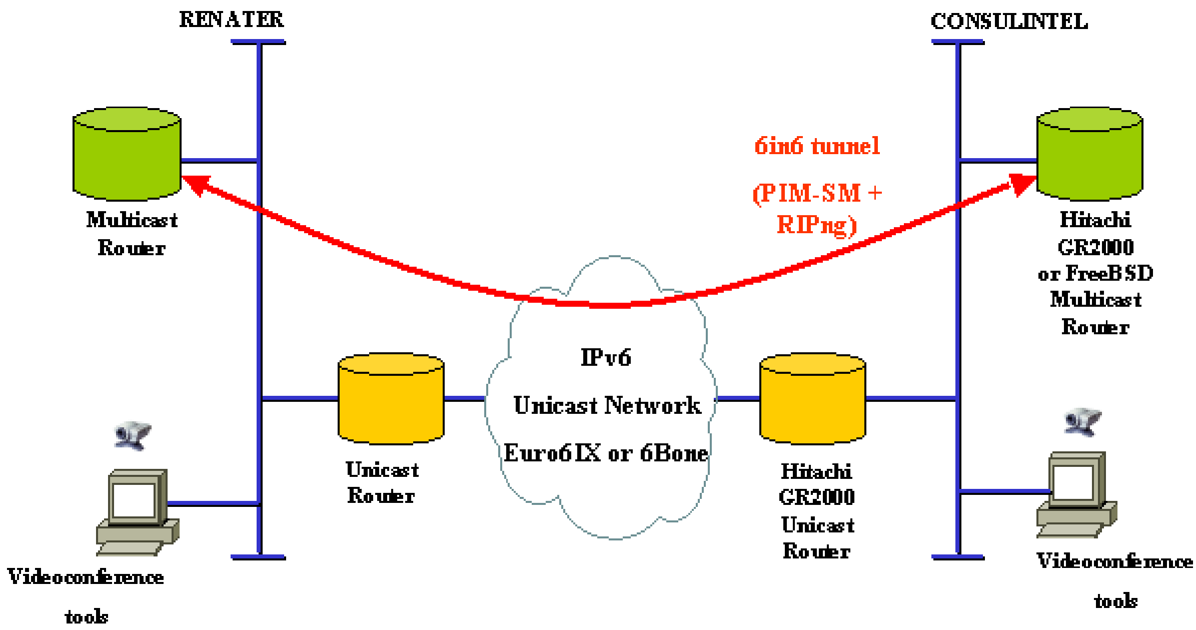


Figure 5-5: Topology used in a M6Bone connection

Figure 5-6 resumes the main tests performed in this third scenario, the purpose of each test, and the results obtained.

Test	Purpose	Results
PIM-SM	Verify the correct performance of IPv6 PIM-SM	Unfinished test on IPv6 on IPv6 tunnels. Further research and tests are underway
MLD	Verify the correct performance of IPv6 MLD	Unfinished test
Multicast delivery	Verify that the packets sent to a multicast IPv6 address are delivered to all hosts with the that multicast address.	Unfinished test

Figure 5-6: WAN/global tests results

6. A MULTICAST SERVICE EXPERIENCE: IMPLEMENTATION OF A VIDEO BROADCASTING SERVICE

6.1 Introduction

The delivering of multimedia contents to multiple users has in multicast networks the natural way of work; with multicast, the load of networks and servers depends on the number of contents streams, instead of the number of users connected to the service.

In order to test a multicast multimedia streaming service, a video streaming service has been implemented.

In this sense the video streaming server application developed and ported to IPv6 by TID, was adapted to multicast traffic. Besides, in order to be able to test the service, the mplayer client for linux boxes was ported to IPv6.

The service was first tested at TID's laboratories using a FreeBSD multicast router, and then over VLANs in the Campus Party event (Valencia, August 2002).

Following, a brief description of the porting and test activities is presented.

6.2 IPv6 video applications (servers and client)

This service has two parts, a client of video and a set of video server tools.

6.2.1 Platform for client and servers

The client and servers run in Linux machines, with kernel version 2.4.8 or superior.

6.2.2 Multicast Video Client

As video client, a modified version of the video player Mplayer has been used.

Mplayer is a video player for Linux, clone of Microsoft's Media Player. The original package can be downloaded from: http://www.mplayerhq.hu.

The modifications to Mplayer have been:

- Addition of IPv6 sockets support.
- Modification of command line syntax in order to accept IPv6 addresses enclosed in '`[]`' characters.
- Addition of a new type of URL called "`UDP://[ipv6 addr]:<port>`". This type of URL makes the client to accept UDP raw streams.
- Addition of Multicast support.
- Modifications in the A/V synchronisation, (out of the scope of this document).

Running mplayer in multicast mode:

```
mplayer udp://<IPv6 multicast group:port>
```

If mplayer exits with error before start playing, follow the instructions in the returned error message.

6.2.3 Server tools

A set of multicast tools has been created:

- Multicast server for MPEG-1.
- Multicast server for MPEG-2.

6.2.3.1 Multicast server for MPEG-1

Is a MPEG-1 system layer (ISO/IEC 11172-1) server, developed in TID, that can deliver contents in UDP packets in unicast or multicast mode.

The command line for use this server is:

```
mpeg_server6 <ipv6_addr> <port> <ttl > <mpeg_file> [-l | -d]
```

Where:

- `ipv6_addr`: is the target group (multicast), or the target host (unicast).
- `port`: UDP target port
- `ttl`: Is the number of hops permitted for this stream, (if 0, unicast socket is opened instead multicast).
- `mpeg_file`: Is the MPEG-1 system layer file to transmit.
- `-l`: When used, the file is transmitted in endless loop mode
- `-d`: Same as “-l”, but running in background.

6.2.3.2 Multicast server for MPEG-2

It is an MPEG-2 program stream (ISO/IEC 13818-2) server, developed in TID that can deliver contents in UDP packets in unicast or multicast mode.

The command line for use this server is:

```
Pgm2server <ipv6_addr> <port> <ttl > <mpeg_file> [-l | -d]
```

Where:

- `ipv6_addr`: is the target group (multicast), or the target host (unicast).
- `port`: UDP target port
- `ttl`: Is the number of hops permitted for this stream, (if 0, unicast socket is opened instead multicast).
- `mpeg_file`: Is the MPEG-2 program layer file to transmit.
- `-l`: When used, the file is transmitted in endless loop mode
- `-d`: Same as "-l", but running in background.

6.2.4 Special Server tools

(See the Campus-Party Experience in section 6.4)

The lack of multicast support for the majority of commercial routers today, forces the need of creating a mechanism that permits multicast connectivity in unicast networks.

Two main problems can make act of presence:

- How to reach the remote node

This issue can be faced in two forms, via tunnel or via UDP proxy. The implementation of a proxy or relay was elected because it was very simple to do. The name of this program is relay_multicast.

- In large Intranets, the L2 network can be fragmented in multiple VLANs, as a mechanism of isolation, another time, routers are the point of hop among VLANs, breaking the IPv6 connectivity.

In this case, direct access to a trunking port of the switch is needed. An apparent solution can be the activation of the Linux 802.1Q support.

The standard 802.1Q support Linux does not permit the replication of the same packet over various VLANs. For this reason, this solution is not valid.

To solve this problem, a special network device daemon has been developed, the 802.1Q VLAN splitter, the name of this program is trunk_802_1q.

6.2.4.1 Unicast/Multicast relay

Served by the application relay multicast, this application listens to a UDP port and forwards the received packet to an IPv6 multicast group.

The use is:

```
relay_multicast <ifz_name | unicast> <incoming group> <input port >
<output group> <output port> <ttl|0> [-d]
```

Where:

- `<ifz_name | unicast>`: Is the name if incoming network adapter, when incoming multicast reading is done. In the majority of cases, the listens will be unicast, so the token unicast must be used.
- `<incoming group>`: Must be the input multicast group in multicast listens, or the IPv6 origin address of the sender.
- `<input port>`: The UDP port to listen.
- `<output group>`: The address of the IPv6 multicast outgoing group.
- `<output port>`: The output port
- `<ttl|0>`: Number of hops permitted to the outgoing packets.
- `[-d]`: When used, the program runs in background.

6.2.4.2 The 802.1Q VLAN splitter

It is served by the application trunk_802_1q. To run this application, tun/tap device support is required.

This application creates various virtual network interfaces, that make the replication of one stream of data in multiple 802.1Q tagged packets. By default, these virtual adapters are bi-directional, but can work in one way mode, to preserve the isolation between VLANS. In a L2 flat network, the server does not need to listen to the ICMPv6 multicast join/leave group solicitations to work properly, because it is a host and not a router. When a static route to a multicast group is set to one interface, the multicast output packets will be delivered anyway.

The use is:

```
trunk_802_1q <file_cfg> <ifz> [-d]
```

Where:

- `<file_cfg>`: Is the application configuration file.
- `<ifz>`: Is the name of the physical network interface.
- `[-d]`: When used, the program runs in background.

Example of configuration file:

```
MTU 1400 #Fix the physical interface MTU
VLAN-GROUP test0 { #Create the virtual interface test0
        #MTU  1450  #MTU  of  virtual  interface  interface  test0
(commented)
        MAC  MAC-NIC  #Fix  the  MAC  with  the  physical  NIC  MAC,  (if
omitted,  a  random  MAC  will  be  used,  you  can  fix  this  field  with  a
concrete MAC address if desired),
        IP_ADDR  88.88.88.88        255.255.255.0  #IPv4  address  and
mask(optional)
        IPv6_ADDR 4444:4444::4/64 #IPv6 address (optional)
        VLAN     0         0 #Tag 0 Priority field 0,(tag 0 mean no
802.1q encapsulation)
}
VLAN-GROUP video { #Create the virtual interface video
        MTU 1450
        MAC MAC-NIC
        XMIT-ONLY #One way interface
        IPv6_ADDR 5555:4444::4/64
        VLAN    2100    0 #triple vlan output defined
        VLAN    2110    3 #Tag 2110, priority field 3
        VLAN    2120    0
}
```

6.3 Experience in the Laboratory

After modifying the mplayer application and developing the IPv6 video streaming multicast capable server, they were tested at TID laboratories between two IPv6 networks communicated through a multicast router consistent in a FreeBSD 4.5 box patched with the kame-20020527-freebsd45-snap. The scenario is shown in the Figure 6-1.

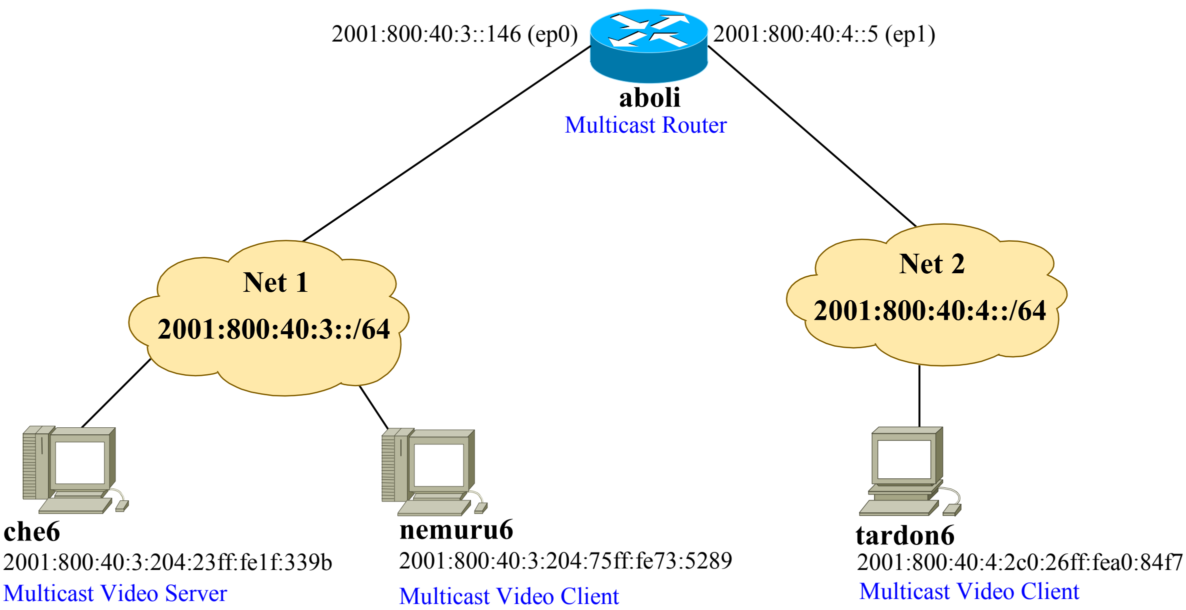


Figure 6-1: Scenario with a multicast router.

6.3.1 Configuration

Multicast Router: FreeBSD 4.5 + kame-20020527-freebsd45-snap

As it was decided to use the Kame patch installed in the `/home/prueba/kame` directory, the multicast and unicast IPv6 daemons must be compiled.

```
# cd /home/prueba/kame/kame/kame/pim6sd
# make
# cd /home/prueba/kame/kame/kame/route6d
# make
# link -s /sbin/route6d /home/prueba/kame/kame/kame/route6d/route6d
```

Then, the generic FreeBSD configuration file `/etc/rc.conf` must be edited and the following lines must be added to configure the host as an IPv6 multicast router:

```
ipv6_gateway_enable="YES"
rtadv_enable="NO"
ipv6_router_enable="YES"
mroute6d_enable="YES"
mroute6d_program="/home/prueba/kame/kame/kame/pim6sd/pim6sd
```

Finally, the behaviour of the PIM6sd must be configured by editing the file /etc/pim6sd.confof the router as it is shown in the Annex.

Multicast Video Server: SuSE 7.2 running mpeg_server6

(che6): `mpeg_server6 FF18::55:55 3 1.mpg -d`

Multicast Video Clients: SuSE 7.2 running mplayer

(nemuru6): `mplayer udp://[FF18::55:55]:9999 -gui -skin plastic -ajustar`

(tardon6): `mplayer udp://[FF18::55:55]:9999 -vo x11`

6.3.2 Test Results

After starting the video server with the destination address FF18::55:55, client applications were launched and both hosts joined the FF18::55:55 multicast group. The result was that the video streaming was presented in both screens without errors or lost frames, perfectly synchronised. **Hence the video server and client application as well as the multicast router behaviour were tested with successful results.**

Monitoring

Besides the obvious results that can be observed just by looking at the screens, the FreeBSD router offers some tools to monitor it.

1. When the multicast routing daemon (PIM) starts, and afterwards periodically, it sends MLD queries to know which addresses and multicast routers are present in the attached networks:

This checking may be manually forced with:

```
# mld6query ep0
# mld6query ep1
```

2. The file pim6stat shows the multicast groups, routes, etc, that is, the multicast statistics.

3. The multicast route table may be showed with:

```
# netstat -g
No IPv4 multicast routing compiled into this system.

IPv6 Multicast Interface Table
 Mif   Rate   PhyIF   Pkts-In   Pkts-Out
   0      0     ep0  31513544          9
   1      0     ep1         0   31513553
   3      0    reg0         0          0

IPv6 Multicast Forwarding Cache
 Origin                          Group   Packts Waits In-Mif Out-Mif
 2001:800:40:3:204:23ff:fe1f:3 ff1f::33:33 11599  0    0    1
 2001:800:40:3:204:23ff:fe1f:3 ff18::55:55 17214  0    0
```

4. The ICMPv6 messages statistics may be showed with:

```
# netstat -sp icmp6
```

6.4 Experience with Multicast in a Public Event (Campus Party 2002)

In August 2002, TID participated in the Campus Party event. One of the Services TID was offering was video broadcasting. The video server, as the other offered services, was located at TID premises (Madrid) and the video clients were in Valencia.

Due to the existence of fourteen VLANs interconnected trough a router (Cisco 7600) which did not support IPv6 Multicast routing at the Campus Party, the decision of developing a new software able to encapsulate IPv6 Multicast video stream in 802.1q (with VLAN labelling) was taken. Hence, the switch (Catalyst 2900) serving all the VLANs at the Campus Party was able to route all the traffic. For that, an 802.1q trunk was made of all the VLANs.

The final scenario, as it is shown in the Figure 6-2, consisted in:

- The IPv6 unicast video server (mabello6), located at TID, transmitted a flow of 1.5 Mbps towards the IPv6 Multicast video server at the Campus Party.
- The IPv6 Multicast video relay/server (che6), located at the Campus, received the unicast video, encapsulated it into 802.1q, and transmitted the IPv6 video stream to the switch (Catalyst 2900).
- The switch (Catalyst 2900) redistributed the stream into all of its ports, included the 802.1q trunk (composed of all the VLANs' ports).

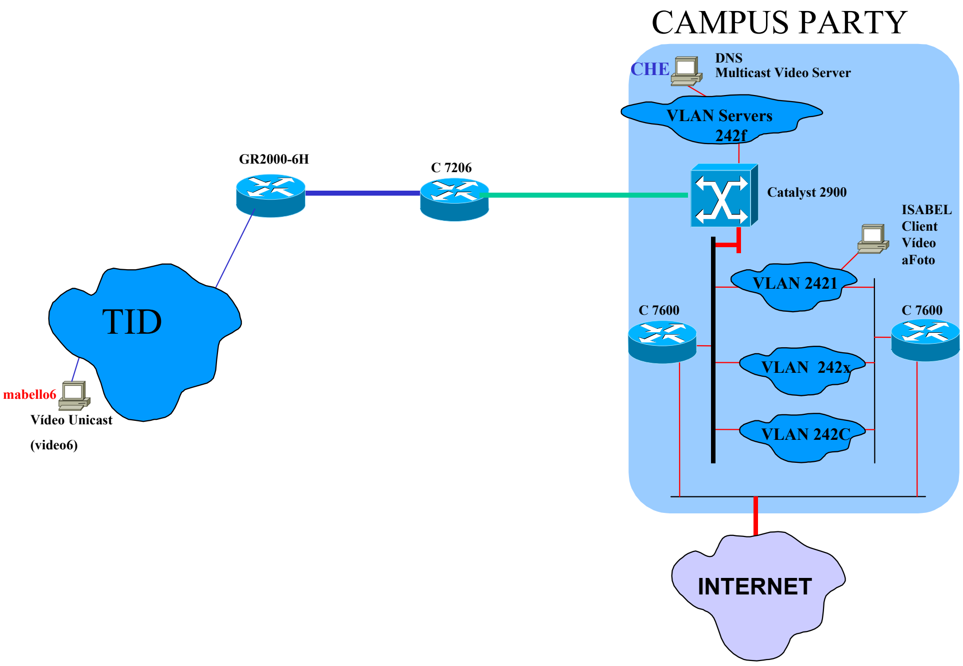


Figure 6-2: Scenario with VLANs.

6.5 Open Issues

Implementation of a MPEG-2 Transport Stream video server.

Addition of MPEG-2 Transport Stream demultiplexing support to mplayer.

Development or porting a multicast video client to Microsoft Windows.

Testing the use of “Traffic Class” and “Flow Label” fields.

7. CONCLUSIONS

Based on the actual point of view and the experiences made in defining and realising only the first to third of the proposed scenarios, the following statements can be done:

- There are a lot of easy-to-use IPv6 Multicast applications, which are well suited to verify the IPv6 Multicast functionality of a given scenario.
- The actual/used IPv6 Multicast applications were easy to compile and to install.
- The basic IPv6 Multicast functionality of Linux end systems works without any problems.
- The performance seems to be much more stable on Linux than for Windows (other end systems have still to be verified).
- Some scenarios need a very complex network infrastructure and can only be investigated if the whole Euro6IX network is up and running.
- There is more input needed from the other project partners in order to light up a broader scope of IPv6 multicast related issues.
- There remains a lot of work for the next project semesters.

8. FUTURE WORK

Depending on the network implementation and development, scenarios 4 and 5 will be performed during the next few semesters of the project.

This means for the near future (3rd semester) to implement and investigate scenario 4 where IPv6 Multicast routers will be implemented and investigated. Furthermore the theoretical part of this deliverable will be rewritten / completed in order to include more some statements to still open issues with respect to Multicast address allocation as well as IPv6 Multicast routing.

More router vendors will be asked for IPv6 Multicast capable implementations and the list of IPv6 Multicast applications will be completed.

An approach for establishing IPv6 Multicast connectivity to the M6Bone will be designed and implemented. This will be one of the main topics of the future IPv6 Multicast investigations, because from the today's point of view IPv6 Multicast is considered as one of the new Network Services which could be implemented and offered to the customers which are connected to an IPv6 Internet Exchange.

The already defined trial scenarios 4 and 5 will be investigated and tested between different Euro6IX partners when the Euro6IX network is up and running and hence are planned for the semesters 3 to 5 of the project.

Inter-operational IPv6 Multicast trials together with the 6NET project have also to be defined and scheduled during the next 2 semesters of the Euro6IX project.

9. REFERENCES

[1] "Multicast Listener Discovery Version 2 (MLDv2) for IPv6", R. Vida, L.Costa, S. Fdida. draft-vida-mld-v2-02.txt, January 2002.

[2] RFC 2373. IP Version 6 Addressing Architecture. R. Hinden, S. Deering. July 1998.

[3] RFC 2080 RIPng for IPv6. G. Malkin, R. Minnear. January 1997.

[4] RFC 2710 Multicast Listener Discovery (MLD) for IPv6. S. Deering, W. Fenner, B. Haberman. October 1999.

[5] "Protocol Independent Multicast - Sparse Mode (PIM-SM): Protocol Specification (Revised)", Bill Fenner, Mark Handley, Hugh Holbrook, Isidor Kouvelas. draft-ietf-pim-sm-v2-new-06.txt, December 2002.

[6] "Protocol Independent Multicast Version 2 Dense Mode Specification", Stephen Deering. draft-ietf-pim-dm-new-v2-01.txt, November 1999.

[7] "Source-Specific Multicast for IP", H. Holbrook, B. Cain. draft-ietf-ssm-arch-01.txt, November 2002.

[8] "IPv6 Multicast Deployment Issues", P.Savola. draft-savola-v6ops-multicast-issues-0-1.txt, November 2002.

[9] "Embedding the Address of RP in IPv6 Multicast Address", P.Savola, B.Haberman. draft-savola-mboned-mcast-rpaddr-00.txt, October 2002.

[10] "Le Réseau National de Télécommunication pour la Technologie, l'Enseignement et Recherche", http://www.renater.fr/.

[11] An experimental IPv6 multicast network: http://www.m6bone.net.

[12] "Multicast Source Discovery Protocol (MSDP)". David Meyer, Bill Fenner. draft-ietf-msdp-spec-14.txt, November, 2002

[13] RFC 2375 IPv6 Multicast Address Assignments. R. Hinden, S. Deering. July 1998.

[14] "Anycast RP mechanism using PIM and MSDP", Dorian Kim, et al. draft-ietf-mboned-anycast-rp-08.txt, May 2001.

[15] RFC 2362 Protocol Independent Multicast-Sparse Mode (PIM-SM): Protocol Specification, D. Estrin. June 1998.

[16] "Protocol Independent Multicast - Dense Mode (PIM-DM): Protocol Specification (Revised)", Andrew Adams, Jonathan Nicholas, William Siadak. draft-ietf-pim-dm-new-v2-02.txt, October 2002.

10. APPENDIXES

10.1 IPv6 Multicast Address Assignment

10.1.1 Fixed Scope Multicast Addresses

These permanently assigned Multicast addresses are valid over a specified scope value.

```
Node-Local Scope:
   FF01:0:0:0:0:0:0:1          All Nodes Address              [ADDARCH]
   FF01:0:0:0:0:0:0:2          All Routers Address            [ADDARCH]
Link-Local Scope:
   FF02:0:0:0:0:0:0:1          All Nodes Address              [ADDARCH]
   FF02:0:0:0:0:0:0:2          All Routers Address            [ADDARCH]
   FF02:0:0:0:0:0:0:3          Unassigned                     [JBP]
   FF02:0:0:0:0:0:0:4          DVMRP Routers                  [RFC1075,JBP]
   FF02:0:0:0:0:0:0:5          OSPFIGP                        [RFC2328,Moy]
   FF02:0:0:0:0:0:0:6          OSPFIGP Designated Routers     [RFC2328,Moy]
   FF02:0:0:0:0:0:0:7          ST Routers                     [RFC1190,KS14]
   FF02:0:0:0:0:0:0:8          ST Hosts                       [RFC1190,KS14]
   FF02:0:0:0:0:0:0:9          RIP Routers                    [RFC2080]
   FF02:0:0:0:0:0:0:A          EIGRP Routers                  [Farinacci]
   FF02:0:0:0:0:0:0:B          Mobile-Agents                  [Bill Simpson]

   FF02:0:0:0:0:0:0:D          All PIM Routers                [Farinacci]
   FF02:0:0:0:0:0:0:E          RSVP-ENCAPSULATION             [Braden]

   FF02:0:0:0:0:0:1:1          Link Name                      [Harrington]
   FF02:0:0:0:0:0:1:2          All-dhcp-agents                [Bound,Perkins]

   FF02:0:0:0:0:1:FFXX:XXXX    Solicited-Node Address         [ADDARCH]

Site-Local Scope:
   FF05:0:0:0:0:0:0:2         All Routers Address             [ADDARCH]

   FF05:0:0:0:0:0:1:3         All-dhcp-servers                [Bound,Perkins]
   FF05:0:0:0:0:0:1:4         All-dhcp-relays                 [Bound,Perkins]
   FF05:0:0:0:0:0:1:1000
   FF05:0:0:0:0:0:1:13FF      Service Location                [RFC2165]
```

Figure 10-1: Fixed Scope IPv6 Multicast Addresses

10.1.2 All Scope Multicast Addresses

These permanently assigned Multicast addresses are valid over all scope ranges. This is shown by an "X" in the scope field of the address, that means any legal scope value.

Note that, as defined in [ADDARCH], IPv6 Multicast addresses, which are only different in scope, represent different groups. Nodes must join each group individually.

The IPv6 Multicast addresses with variable scope are as follows:

```
FF0X:0:0:0:0:0:0:0           Reserved Multicast Address         [ADDARCH]
FF0X:0:0:0:0:0:0:100         VMTP Managers Group             [RFC1045,DRC3]
FF0X:0:0:0:0:0:0:101         Network Time Protocol (NTP)     [RFC1119,DLM1]
FF0X:0:0:0:0:0:0:102         SGI-Dogfight                               [AXC]
FF0X:0:0:0:0:0:0:103         Rwhod                                      [SXD]
FF0X:0:0:0:0:0:0:104         VNP                                       [DRC3]
FF0X:0:0:0:0:0:0:105         Artificial Horizons - Aviator              [BXF]
FF0X:0:0:0:0:0:0:106         NSS - Name Service Server                 [BXS2]
FF0X:0:0:0:0:0:0:107         AUDIONEWS - Audio News Multicast          [MXF2]
FF0X:0:0:0:0:0:0:108         SUN NIS+ Information Service              [CXM3]
FF0X:0:0:0:0:0:0:109         MTP Multicast Transport Protocol           [SXA]
FF0X:0:0:0:0:0:0:10A         IETF-1-LOW-AUDIO                           [SC3]
FF0X:0:0:0:0:0:0:10B         IETF-1-AUDIO                               [SC3]
FF0X:0:0:0:0:0:0:10C         IETF-1-VIDEO                               [SC3]
FF0X:0:0:0:0:0:0:10D         IETF-2-LOW-AUDIO                           [SC3]
FF0X:0:0:0:0:0:0:10E         IETF-2-AUDIO                               [SC3]
FF0X:0:0:0:0:0:0:10F         IETF-2-VIDEO                               [SC3]
FF0X:0:0:0:0:0:0:110         MUSIC-SERVICE                  [Guido van Rossum]
FF0X:0:0:0:0:0:0:111         SEANET-TELEMETRY                  [Andrew Maffei]
FF0X:0:0:0:0:0:0:112         SEANET-IMAGE                      [Andrew Maffei]
FF0X:0:0:0:0:0:0:113         MLOADD                                   [Braden]
FF0X:0:0:0:0:0:0:114         any private experiment                      [JBP]
FF0X:0:0:0:0:0:0:115         DVMRP on MOSPF                              [Moy]
FF0X:0:0:0:0:0:0:116         SVRLOC                                 [Veizades]
FF0X:0:0:0:0:0:0:117         XINGTV                            <hgxing@aol.com>
FF0X:0:0:0:0:0:0:118         microsoft-ds              <arnoldm@microsoft.com>
FF0X:0:0:0:0:0:0:119         nbc-pro                <bloomer@birch.crd.ge.com>
FF0X:0:0:0:0:0:0:11A         nbc-pfn                <bloomer@birch.crd.ge.com>
FF0X:0:0:0:0:0:0:11B         lmsc-calren-1                              [Uang]
FF0X:0:0:0:0:0:0:11C         lmsc-calren-2                              [Uang]
FF0X:0:0:0:0:0:0:11D         lmsc-calren-3                              [Uang]
FF0X:0:0:0:0:0:0:11E         lmsc-calren-4                              [Uang]
FF0X:0:0:0:0:0:0:11F         ampr-info                               [Janssen]
FF0X:0:0:0:0:0:0:120         mtrace                                   [Casner]
FF0X:0:0:0:0:0:0:121         RSVP-encap-1                             [Braden]
FF0X:0:0:0:0:0:0:122         RSVP-encap-2                             [Braden]
FF0X:0:0:0:0:0:0:123         SVRLOC-DA                              [Veizades]
FF0X:0:0:0:0:0:0:124         rln-server                                 [Kean]
FF0X:0:0:0:0:0:0:125         proshare-mc                               [Lewis]
FF0X:0:0:0:0:0:0:126         dantz                                    [Yackle]
FF0X:0:0:0:0:0:0:127         cisco-rp-announce                    [Farinacci]
FF0X:0:0:0:0:0:0:128         cisco-rp-discovery                   [Farinacci]
FF0X:0:0:0:0:0:0:129         gatekeeper                                 [Toga]
FF0X:0:0:0:0:0:0:12A         iberiagames                             [Marocho]
FF0X:0:0:0:0:0:0:201      "rwho"Group (BSD) (unofficial)          [JBP]
FF0X:0:0:0:0:0:0:202        SUN RPC PMAPPROC_CALLIT                   [BXE1]
FF0X:0:0:0:0:0:2:0000
 -FF0X:0:0:0:0:0:2:7FFD   Multimedia Conference Calls             [SC3]
FF0X:0:0:0:0:0:2:7FFE     SAPv1 Announcements                     [SC3]
FF0X:0:0:0:0:0:2:7FFF       SAPv0 Announcements (deprecated)        [SC3]
FF0X:0:0:0:0:0:2:8000
 -FF0X:0:0:0:0:0:2:FFFF     SAP Dynamic Assignments                   [SC3]
```

Figure 10-2: All Scope IPv6 Multicast Addresses

10.2 IPv6 Multicast Routing Tables (extracted from tests)

Following sections present the IPv6 Multicast Routing tables that have been extracted from sections 4.4and 6.3 to facilitate reading.

10.2.1 Test of Scenario 2: Networks with IPv6 Multicast FreeBSD routers and PIM-SMv6

85df	Server	181e	Client 3
5a20	Client 1	1f77	RP-Interface
22fe	Client 2		

RP (FreeBSD) pim6stat result:

```
Multicast Interface Table
 Mif    PhyIF Local-Address/Prefixlen                     Scope Flags
  0       xl0 2001:7a0:100:110:210:4bff:fe3e:1f77/64       0   DR PIM
              fe80::210:4bff:fe3e:1f77/64                  1
              Timers: PIM hello = 0:25, MLD query = 1:30,  MLD version = 1
  1       xl1 2001:7a0:100:112:260:8ff:fe7e:987/64         0   DR QRY NO-NBR
              fe80::260:8ff:fe7e:987/64                    2
              Timers: PIM hello = 0:25, MLD query = 1:40,  MLD version = 1
  2       ep0 2001:7a0:100:10e:260:8ff:fe63:43f8/64        0   DISABLED
              fe80::260:8ff:fe63:43f8/64                   3
              Timers: PIM hello = 0:00, MLD query = 0:00,  MLD version = 1
  3       lo0 fe80::1/64                                   4   DISABLED
              ::1/128                                      0
              Timers: PIM hello = 0:00, MLD query = 0:00,  MLD version = 1
  4    regist 2001:7a0:100:110:210:4bff:fe3e:1f77/64       0   REGISTER
              fe80::210:4bff:fe3e:1f77/64                  1
              Timers: PIM hello = 0:00, MLD query = 0:00,  MLD version = 1

PIM Neighbour List
 Mif   PhyIF Address                               Timer
 0       xl0 fe80::204:76ff:fe9e:4228               95
                 2001:7a0:100:110:204:76ff:fe9e:4228

MLD Queerer List
 Mif   PhyIF Address                               Timer            Last
 0       xl0 fe80::204:76ff:fe9e:4228                140       1d0h38m58s
 1       xl1 fe80::260:8ff:fe7e:987                  255       1d0h39m37s

Multicast Routing Table
 Source           Group            RP-addr          Flags
---------------------------(*,G)----------------------------
 IN6ADDR_ANY      ff15:80a0::12  2001:7a0:100:110:210:4bff:fe3e:1f77 WC RP
Joined   oifs: j....
Pruned   oifs: .....
Leaves   oifs: .l...
Asserted oifs: .....
Outgoing oifs: oo...
Incoming     : ....I
Upstream nbr: NONE

TIMERS: Entry=165 JP=20 RS=0 Assert=0
  MIF    0   1   2   3   4   5   6   7   8   9
    0 165    0   0   0   0
---------------------------(S,G)----------------------------
 2001:7a0:100:110:260:8ff:fe6f:5a20                          ff15:80a0::12
2001:7a0:100:110:210:4bff:fe3e:1f77 SPT CACHE SG
Joined   oifs: j....
Pruned   oifs: .....
Leaves   oifs: .l...
Asserted oifs: .....
Outgoing oifs: .o...
Incoming     : I....
Upstream nbr: NONE

TIMERS: Entry=160 JP=15 RS=0 Assert=0
  MIF    0   1   2   3   4   5   6   7   8   9
    0 105    0   0   0   0
---------------------------(S,G)----------------------------
 2001:7a0:100:112:250:daff:fe3d:85df                         ff15:80a0::12
2001:7a0:100:110:210:4bff:fe3e:1f77 SPT CACHE SG
Joined   oifs: j....
Pruned   oifs: .....
```

```
Leaves   oifs: .l...
Asserted oifs: .....
Outgoing oifs: o....
Incoming     : .I...
Upstream nbr: NONE

TIMERS: Entry=150 JP=20 RS=0 Assert=0
  MIF   0   1   2   3   4   5   6   7   8   9
    0 150   0   0   0   0
---------------------------(S,G)----------------------------
 2001:7a0:100:113:290:27ff:fedc:22fe                          ff15:80a0::12
2001:7a0:100:110:210:4bff:fe3e:1f77 RP CACHE SG
Joined   oifs: .....
Pruned   oifs: p....
Leaves   oifs: .l...
Asserted oifs: .....
Outgoing oifs: .o...
Incoming     : ....I
Upstream nbr: NONE

TIMERS: Entry=205 JP=20 RS=0 Assert=0
  MIF   0   1   2   3   4   5   6   7   8   9
    0   0   0   0   0   0
---------------------------(S,G)----------------------------
 2001:7a0:100:114:204:76ff:fea0:181e                          ff15:80a0::12
2001:7a0:100:110:210:4bff:fe3e:1f77 RP CACHE SG
Joined   oifs: .....
Pruned   oifs: p....
Leaves   oifs: .l...
Asserted oifs: .....
Outgoing oifs: .o...
Incoming     : ....I
Upstream nbr: NONE

TIMERS: Entry=210 JP=40 RS=0 Assert=0
  MIF   0   1   2   3   4   5   6   7   8   9
    0   0   0   0   0   0
--------------------------(*,*,RP)--------------------------
Number of Groups: 1
Number of Cache MIRRORs: 4

---------------------------RP-Set----------------------------
Current BSR address: 2001:7a0:100:110:210:4bff:fe3e:1f77Prio:0 Timeout:40
RP-address/Upstream                   IN  Group prefix     Prio Hold Age
2001:7a0:100:110:210:4bff:fe3e:1f77    4  ff15::/16        0    150  100
  (none)
```

mc-flurry (FreeBSD) pim6stat result:

```
Multicast Interface Table
 Mif   PhyIF Local-Address/Prefixlen                     Scope Flags
  0      xl0 2001:7a0:100:110:204:76ff:fe9e:4228/64      0   PIM QRY
             fe80::204:76ff:fe9e:4228/64                 1
             Timers: PIM hello = 0:05, MLD query = 1:50,  MLD version = 1
  1      xl1 2001:7a0:100:113:204:76ff:fe9e:41dd/64      0   DR QRY NO-NBR
             fe80::204:76ff:fe9e:41dd/64                 2
             Timers: PIM hello = 0:30, MLD query = 1:50,  MLD version = 1
  2      xl2 2001:7a0:100:114:204:76ff:fe9e:3ed5/64      0   DR QRY NO-NBR
             fe80::204:76ff:fe9e:3ed5/64                 3
             Timers: PIM hello = 0:30, MLD query = 1:50,  MLD version = 1
  3      lo0 fe80::1/64                                  5   DISABLED
             ::1/128                                     0
             Timers: PIM hello = 0:00, MLD query = 0:00,  MLD version = 1
  4   regist 2001:7a0:100:110:204:76ff:fe9e:4228/64      0   REGISTER
             fe80::204:76ff:fe9e:4228/64                 1
             Timers: PIM hello = 0:00, MLD query = 0:00,  MLD version = 1

PIM Neighbour List
 Mif  PhyIF Address                                  Timer
 0      xl0 fe80::210:4bff:fe3e:1f77                 80
               2001:7a0:100:110:210:4bff:fe3e:1f77

MLD Queerer List
 Mif  PhyIF Address                                 Timer            Last
 0      xl0 fe80::204:76ff:fe9e:4228                  255       2d0h26m34s
 1      xl1 fe80::204:76ff:fe9e:41dd                  255       2d0h26m34s
 2      xl2 fe80::204:76ff:fe9e:3ed5                  255       2d0h26m34s

Multicast Routing Table
 Source          Group            RP-addr          Flags
---------------------------(*,G)----------------------------
 IN6ADDR_ANY       ff15:80a0::12    2001:7a0:100:110:210:4bff:fe3e:1f77  WC  RP
CACHE
Joined   oifs: .....
Pruned   oifs: .....
Leaves   oifs: .ll..
Asserted oifs: .....
Outgoing oifs: .oo..
Incoming     : I....
Upstream nbr: fe80::210:4bff:fe3e:1f77

TIMERS: Entry=0 JP=60 RS=0 Assert=0
  MIF    0   1   2   3   4   5   6   7   8   9
    0    0   0   0   0   0
---------------------------(S,G)----------------------------
 2001:7a0:100:112:250:daff:fe3d:85df                          ff15:80a0::12
2001:7a0:100:110:210:4bff:fe3e:1f77 CACHE SG
Joined   oifs: .....
Pruned   oifs: .....
Leaves   oifs: .ll..
Asserted oifs: .....
Outgoing oifs: .oo..
Incoming     : I....
Upstream nbr: fe80::210:4bff:fe3e:1f77

TIMERS: Entry=150 JP=5 RS=0 Assert=0
  MIF    0   1   2   3   4   5   6   7   8   9
    0    0   0   0   0   0
---------------------------(S,G)----------------------------
 2001:7a0:100:113:290:27ff:fedc:22fe                          ff15:80a0::12
2001:7a0:100:110:210:4bff:fe3e:1f77 SPT CACHE SG
```

```
Joined   oifs: ....j
Pruned   oifs: .....
Leaves   oifs: .....
Asserted oifs: .....
Outgoing oifs: ..o.o
Incoming     : .I...
Upstream nbr: NONE

TIMERS: Entry=205 JP=60 RS=0 Assert=0
  MIF   0   1   2   3   4   5   6   7   8   9
    0   0   0   0   0   0
---------------------------(S,G)----------------------------
 2001:7a0:100:114:204:76ff:fea0:181e                       ff15:80a0::12
2001:7a0:100:110:210:4bff:fe3e:1f77 SPT CACHE SG
Joined   oifs: ....j
Pruned   oifs: .....
Leaves   oifs: .ll..
Asserted oifs: .....
Outgoing oifs: .o..o
Incoming     : ..I..
Upstream nbr: NONE

TIMERS: Entry=210 JP=60 RS=0 Assert=0
  MIF   0   1   2   3   4   5   6   7   8   9
    0   0   0   0   0   0
--------------------------(*,*,RP)--------------------------
Number of Groups: 1
Number of Cache MIRRORs: 4

---------------------------RP-Set---------------------------
Current BSR address:2001:7a0:100:110:210:4bff:fe3e:1f77Prio:0 Timeout:120
RP-address/Upstream                        IN   Group prefix   Prio Hold Age
2001:7a0:100:110:210:4bff:fe3e:1f77        0    ff15::/16      0    150  110
  fe80::210:4bff:fe3e:1f77
```

10.2.2 Test of Scenario 3

22fe	Server	181e	Client 2
5c1c	Client 1	7/b04a	RP-Interface

RP (Hitachi) show pim6:

```
DURIN/command> ip6 show pim6 interface
Interface      Component Vif Nbr   Hello DR                        This
                             Count Intvl Address                   Router
EURO6IX-MC-1   PIM-SM      1     0    30 fe80::200:87ff:fe28:b04c Y
IPv6-IX        PIM-SM      2     1    30 fe80::204:76ff:fe9e:4228 N

DURIN/command> ip6 show pim6 neighbour
NeighborAddress          Interface      Uptime  Expires
fe80::204:76ff:fe9e:4228 IPv6-IX         3days  01:23

DURIN/command> ip6 show pim6 mfc
Group                                   Source
ff15:80a0::12                           2001:7a0:100:113:290:27ff:fedc:22fe
    Uptime 12:31    Expires 03:30   Component: PIM-SM
    upstream:
        EURO6IX-MC-1
    downstream:
        IPv6-IX
ff15:80a0::12                           2001:7a0:100:114:204:76ff:fea0:181e
    Uptime 13:36    Expires 03:30   Component: PIM-SM
    upstream:
        IPv6-IX
    downstream:
        EURO6IX-MC-1
ff15:80a0::12                           2001:7a0:100:114:210:5aff:fe70:5c1c
    Uptime 06:13    Expires 03:30   Component: PIM-SM
    upstream:
        IPv6-IX
    downstream:
        EURO6IX-MC-1

DURIN/command> ip6 show pim6 bsr
Status : Elected Bootstrap Router
BSR Address  : 2001:7a0:100::7 (This Router)
    Priority : 0    Hash mask length : 126
    Uptime   :  4days
    Bootstrap Timeout  : 45 seconds
    Bootstrap Interval : 60 seconds

DURIN/command> ip6 show pim6 rp-set
Status : Candidate Rendezvous Point
         Local RP Address: 2001:7a0:100::7    Priority: 0
Total: 1 route , 1 group , 1 RP
Group/masklen                    C-RP Address        Priority Uptime  Expires
ff15::/16                        2001:7a0:100::7            0  4days  02:05

DURIN/command> ip6 show pim6 rp-hash ff15:80a0::12
RP Address                               Uptime  Expires
2001:7a0:100::7                           4days  02:30

DURIN/command> ip6 show pim6 route
Total: 4 routes, 1 group , 3 sources

(S,G) 3 routes  ---------------------------------------------------
Group Address                           Source Address
ff15:80a0::12                           2001:7a0:100:113:290:27ff:fedc:22fe
    Uptime 15:49    Expires 03:15    Assert 00:00    Flags SFLT
    in-coming: EURO6IX-MC-1    upstream: Direct  Reg-Sup: 0s
    downstream: IPv6-IX         uptime 15:47    expires 03:08

ff15:80a0::12                           2001:7a0:100:114:204:76ff:fea0:181e
    Uptime  3days   Expires 03:23    Assert 00:00    Flags SLT
```

```
    in-coming: IPv6-IX         upstream: fe80::204:76ff:fe9e:4228
    downstream: EURO6IX-MC-1    uptime  3days   expires --:--

ff15:80a0::12                           2001:7a0:100:114:210:5aff:fe70:5c1c
    Uptime  3days   Expires 03:32    Assert 00:00    Flags SLT
    in-coming: IPv6-IX         upstream: fe80::204:76ff:fe9e:4228
    downstream: EURO6IX-MC-1    uptime  3days   expires --:--

(*,G) 1 route   ---------------------------------------------------
Group Address                           RP Address
ff15:80a0::12                           2001:7a0:100::7
    Uptime  4days   Expires 03:08    Assert 00:00    Flags SLR
    in-coming: localhost       upstream: This Router
    downstream: EURO6IX-MC-1    uptime  3days   expires --:--
                IPv6-IX         uptime 15:47    expires 03:08
DURIN/command>
```

mc-flurry (freeBSD) pim6stat result:

```
Multicast Interface Table
 Mif    PhyIF Local-Address/Prefixlen                    Scope Flags
  0       xl0 2001:7a0:100:104:204:76ff:fe9e:4228/64      0   DR PIM
              fe80::204:76ff:fe9e:4228/64                 1
              Timers: PIM hello = 0:20, MLD query = 1:15,  MLD version = 1
  1       xl1 2001:7a0:100:114:204:76ff:fe9e:41dd/64      0   DR QRY NO-NBR
              fe80::204:76ff:fe9e:41dd/64                 2
              Timers: PIM hello = 0:20, MLD query = 0:40,  MLD version = 1
  2       xl2 fe80::204:76ff:fe9e:3ed5/64                 3   DR QRY NO-NBR
              Timers: PIM hello = 0:20, MLD query = 0:40,  MLD version = 1
  3       lo0 fe80::1/64                                  5   DISABLED
              ::1/128                                     0
              Timers: PIM hello = 0:00, MLD query = 0:00,  MLD version = 1
  4    regist 2001:7a0:100:104:204:76ff:fe9e:4228/64      0   REGISTER
              fe80::204:76ff:fe9e:4228/64                 1
              Timers: PIM hello = 0:00, MLD query = 0:00,  MLD version = 1

PIM Neighbour List
 Mif   PhyIF Address                                 Timer
 0       xl0 fe80::200:87ff:fe28:b04a                90
                2001:7a0:100:104:200:87ff:fe28:b04a

MLD Queerer List
 Mif   PhyIF Address                                Timer          Last
 0       xl0 fe80::200:87ff:fe28:b04a                 210      3d18h31m52s
 1       xl1 fe80::204:76ff:fe9e:41dd                 255      3d18h32m42s
 2       xl2 fe80::204:76ff:fe9e:3ed5                 255      3d18h32m42s

Multicast Routing Table
 Source          Group           RP-addr         Flags
---------------------------(*,G)---------------------------
 IN6ADDR_ANY     ff15:80a0::12   2001:7a0:100::7 WC RP CACHE
Joined   oifs: .....
Pruned   oifs: .....
Leaves   oifs: .l...
Asserted oifs: .....
Outgoing oifs: .o...
Incoming     : I....
Upstream nbr: fe80::200:87ff:fe28:b04a

TIMERS: Entry=0 JP=25 RS=0 Assert=0
  MIF   0   1   2   3   4   5   6   7   8   9
    0   0   0   0   0   0
---------------------------(S,G)---------------------------
 2001:7a0:100:114:204:76ff:fea0:181e  ff15:80a0::12      2001:7a0:100::7   SPT
CACHE SG
Joined   oifs: j...j
Pruned   oifs: ....p
Leaves   oifs: .....
Asserted oifs: .....
Outgoing oifs: o....
Incoming     : .I...
Upstream nbr: NONE

TIMERS: Entry=205 JP=25 RS=59 Assert=0
  MIF   0   1   2   3   4   5   6   7   8   9
    0 205   0   0   0   0
---------------------------(S,G)---------------------------
 2001:7a0:100:114:210:5aff:fe70:5c1c  ff15:80a0::12      2001:7a0:100::7   SPT
CACHE SG
Joined   oifs: j...j
Pruned   oifs: ....p
```

```
Leaves   oifs: .....
Asserted oifs: .....
Outgoing oifs: o....
Incoming     : .I...
Upstream nbr: NONE

TIMERS: Entry=170 JP=25 RS=38 Assert=0
  MIF   0   1   2   3   4   5   6   7   8   9
    0 170   0   0   0   0
--------------------------(*,*,RP)--------------------------
Number of Groups: 1
Number of Cache MIRRORs: 3

---------------------------RP-Set----------------------------
Current BSR address: 2001:7a0:100::7 Prio: 0 Timeout: 150
RP-address/Upstream                   IN  Group prefix     Prio Hold Age
2001:7a0:100::7                       0   ff15::/16        0    150  140
  fe80::200:87ff:fe28:b04a
```

10.2.3 A multicast Service Experience: Experience with Multicast in the laboratory

The experience of testing the video service at TID laboratories was described in section 6.3. Following, the configuration and the statistics files of the multicast router used are shown.

aboli (freeBSD) /etc/pim6sd.conf:

```
#The timer granularity.
#More this value is small,more pim6sd will be accurate
#default if not specified : 5
#BE SURE to have to same granularity on ALL routers,
#otherwise....

granularity 5;

#syntax  :  phyint  <interface>  <disable>  <metric>  [metric]  <preference>
[preference]
#metric and pref are for the asserts messages
#samples :

#phyint ed1 disable;
#phyint de0 disable;
#phyint ed0 disable;
#phyint gif0 disable;

phyint ep0;
phyint ep1;

#---------------Protocol timer specifications---------------------------#
#Notes : theses value are the default spec value!
#do not touch it if you don't know what you do!!
#you MUST change theses values according to the granularity value!
#syntax :  'hello_period <number> <coef>'.
# number is the period in second between 2 hello messages
# and coef is the coef to deterimine the hello holdtime=hello_period*coef
# default if not specified: 30 3.5

hello_period 30 3.5;

#syntax :  'join_prune_period <number> <coef>'.
# number is the period in second between 2 join/prune messages
#    and    coef    is    the    coef    to    deterimine    the    join/prune
holdtime=join_prune_period*coef
# default if not specified : 60 3.5

join_prune_period 60 3.5;

#syntax :  'data_timeout <number>'.
# number is the time after which (S,G) state for a silent source will be
deleted
# default if not specified  : 210

data_timeout 210;

#syntax :  'register_suppression_timeout <number>'.
# This is the mean interval between receiving a Register-Stop and allowing
#Register to be send again.
# default if not specified  : 60

register_suppression_timeout 60;

#syntax :  'probe_time <number>'.
#This  is  the  time  between  sending  a  null  Register  and  the  Register-
Suppression-Timer
#expiring unless it is restarted by receiving a Register-Stop.
#default if not specified : 5

probe_time 5;
```

```
#syntax :  'assert_timeout <number>'.
#this is the interval between the last time an Assert is received and the
time at wich the
#assert is timeout
#default if not specified : 180

assert_timeout 180;

#syntax : <cand_rp> <interface> <time> [time] <priority> [priority]
#and time can't be <  10
#you can just type cand_rp,
#samples :
#cand_rp;
#cand_rp de0;
#cand_rp ed0 priority 0 time 6;

cand_rp ep0;

#syntax : <group_prefix> <multicast address>/<prefix length>
#group_prefix ff06::15
#default if not specified : ff00::/8
#samples:
#group_prefix ff1e::15/128;
#group_prefix ff2e::/16;

group_prefix ff00::/8;
group_prefix ff18::55:55/16;

#syntax : <cand_bootstrap_router> <interface> <priority> [priority] <time>
[time]
#Typically, you can simply set cand_bootstrap_router for a candidate
bootstrap
#router. All other parameters are optional.
#the bootstrap period is configurable, BUT the holdtime of a bootstrap
#router is not in the fields of a bootstrap message : it is hardcoded
#in the pim6sd include file!
#So be sure to have a time < PIM_BOOSTRAP_TIMEOUT (file pimd.h )
cand_bootstrap_router ep0 priority 15 time 5;

#syntax : <switch_register_threshold> <rate> [number] <interval> [number]
#default rate = 50000 interval = 20s
#samples :
#TODO : not tested
#switch_register_threshold rate 54389 interval 45;
switch_register_threshold;

#syntax : <switch_data_threshold> <rate> [number] <interval> [number]
#default rate = 50000 interval = 20s
#TODO : not tested
#samples:
switch_data_threshold interval 100 rate 1000;

#syntax : <default_source_metric> [number]

default_source_metric 1243;
#syntax : <default_source_preference> [number]

default_source_preference 123 ;
aboli (freeBSD) pim6stat result:
Multicast Interface Table
 Mif   PhyIF Local-Address/Prefixlen           Scope Flags
  0      ep0 2001:800:40:3::146/64             0     DR QRY NO-NBR
             fe80::2a0:24ff:fec5:79df/64       1
```

```
             Timers: PIM hello = 0:10, MLD query = 1:05,  MLD version = 1
  1      ep1 2001:800:40:4::5/64                 0     DR QRY NO-NBR
             fe80::2a0:24ff:fec5:784c/64         2
             Timers: PIM hello = 0:10, MLD query = 1:05,  MLD version = 1
  2      lo0 fe80::1/64                          3     DISABLED
             ::1/128                             0
             Timers: PIM hello = 0:00, MLD query = 0:00,  MLD version = 1
  3   regist 2001:800:40::146/64               0     REGISTER
             fe80::2a0:24ff:fec5:79df/64         1
             Timers: PIM hello = 0:00, MLD query = 0:00,  MLD version = 1

PIM Neighbour List
 Mif  PhyIF Address                                 Timer

MLD Queerer List
 Mif  PhyIF Address                                 Timer            Last
 0      ep0 fe80::2a0:24ff:fec5:79df%ep0              255        2d1h7m1s
 1      ep1 fe80::2a0:24ff:fec5:784c%ep1              255        2d1h7m1s

Multicast Routing Table
 Source          Group           RP-addr         Flags
---------------------------(*,G)----------------------------
 IN6ADDR_ANY     ff18::55:55     2001:800:40:3::146 WC RP
Joined   oifs: ....
Pruned   oifs: ....
Leaves   oifs: ll..
Asserted oifs: ....
Outgoing oifs: oo..
Incoming     : ...I
Upstream nbr: NONE

TIMERS: Entry=0 JP=20 RS=0 Assert=0
  MIF   0   1   2   3   4   5   6   7   8   9
    0   0   0   0   0



---------------------------(S,G)----------------------------
 2001:800:40:3:204:23ff:fe1f:339b  ff18::55:55         2001:800:40:3::146  SPT
CACHE SG
Joined   oifs: ....
Pruned   oifs: ....
Leaves   oifs: ll..
Asserted oifs: ....
Outgoing oifs: .o..
Incoming     : I...
Upstream nbr: NONE

TIMERS: Entry=125 JP=40 RS=0 Assert=0
  MIF   0   1   2   3   4   5   6   7   8   9
    0   0   0   0   0
---------------------------(*,G)----------------------------
 IN6ADDR_ANY     ff1f::33:33     2001:800:40:3::146 WC RP
Joined   oifs: ....
Pruned   oifs: ....
Leaves   oifs: ll..
Asserted oifs: ....
Outgoing oifs: oo..
Incoming     : ...I
Upstream nbr: NONE

TIMERS: Entry=0 JP=50 RS=0 Assert=0
  MIF   0   1   2   3   4   5   6   7   8   9
    0   0   0   0   0
```

```
---------------------------(S,G)----------------------------
 2001:800:40:3:204:23ff:fe1f:339b  ff1f::33:33         2001:800:40:3::146  SPT
CACHE SG
Joined   oifs: ....
Pruned   oifs: ....
Leaves   oifs: ll..
Asserted oifs: ....
Outgoing oifs: .o..
Incoming     : I...
Upstream nbr: NONE

TIMERS: Entry=165 JP=20 RS=0 Assert=0
  MIF   0   1   2   3   4   5   6   7   8   9
    0   0   0   0   0

--------------------------(*,*,RP)--------------------------
Number of Groups: 2
Number of Cache MIRRORs: 2

---------------------------RP-Set----------------------------
Current BSR address: 2001:800:40:3::146 Prio: 15 Timeout: 10
RP-address/Upstream                      IN  Group prefix     Prio Hold Age
2001:800:40:3::146                       3   ff00::/8         0    150  105
  (none)                                     ff18::/16        0
```

10.3 FreeBSD Installation guide

- Down load the image and record on CDROM:
  http://www.freebsdmirrors.org/FBSDsites.php3?showi386ISO=do
- Setup the PC form the CDROM

If it is not possible to setup from CDROM, the images of the files kern.flp and mfsroot.flp have to be created on floppy; these images are located on /floppies and it is built with the program fdimage.exe, located on:

```
c:\ fdimage kern.flp a:
c:\ fdimage mfsroot.flp a:
```

- Installation option:

- Kernel in visual mode

- Options → IPv6 yes

→ debugging yes

- Standard installation

10.4 Installation of the new Kame kernel

- Download the image kame-20020402-freebsd4.5-snap.tgz from http://www.kamet.net and extract to local directory.

For example:

```
cd /IPv6
tar zxvf kame-20020402-freebsd4.5-snap.tgz
```

The structure of this local directory has the following layout:

```
IPv6 ── kame ──┬── INSTALL
               ├── kame ────┬── sys
               │            └── kame ──┬── libinet6
               │                       └── .........
               └── freebsd4 ──┬── INSTALL
                              ├── sys
                              ├── lib ──┬── libinet6
                              │         └── ...........
                              └── ...........
```

Note: if there is another version of the KAME kit, the files on "/usr/local/v6" should be removed to avoid possible conflicts.

- In the directory /IPv6/kame:

```
cd /IPv6/kame
make TARGET=freebsd4 prepare
```

- Backup the previous kernel:

```
cp /kernel /kernel.previous
cd /usr
mkdir include.clean
cd include.clean
(cd ../include; tar Bpcf - . ) | tar Bpxf –
```

- Build the new kernel

```
cd /IPv6/kame/freebsd4
cd sys/i386/conf
cp GENERIC.KAME CONFIGFILE
vi CONFIGFILE                //verificar os vários parâmetros
/usr/sbin/config CONFIGFILE
cd ../../compile/CONFIGFILE
make depend
make
make install
```

- Installation of new includes

```
cd /IPv6/kame/freebsd4
make includes
make install-includes
make install
```

- Restart the PC

```
fastboot
```

Note:

- The configuration files are located in the directory "/usr/local/v6/etc"
- The modified commands are in directories "/usr/local/v6/sbin" and "/usr/local/v6/bin". So, to work correctly is need to include in file "/etc/rc.conf" the following line:

```
PATH=/usr/local/v6/sbin:/usr/local/v6/bin:$(PATH)
```

- The documentation is installed in directories "/usr/local/v6/man/" and "/usr/local/v6/shared/doc".

10.5 IPv6 multicast Configuration on FreeBSD

- Enable the PIM Dense Mode

The daemon is the program pim6dd and the default configuration file is the /etc/pim6dd.conf. In the file /etc/rc.conf, add the following lines:

```
mroute6d_enable="YES"
mroute6d_program="/usr/sbin/pim6dd"
```

- Enable the PIM Sparse Mode

The daemon is the pim6dd programe and the default configuration file is the /etc/pim6sd.conf. In the file /etc/rc.conf, add the following lines:

```
mroute6d_enable="YES"
mroute6d_program="/usr/sbin/pim6sd"
```

10.6 IPv6 tunnels on FreeBSD

- Configuration of the IPv6 tunnel over IPv6 (gif0)

Execute the following commands:

```
 ifconfig gif0 create
 ifconfig gif0 inet6 <ocal end-tunnel> <remote end-Tunnel> prefixlen 128
 gifconfig gif0 inet6 <local ipv6addr on interface> <remote ipv6addr on interface>
 ifconfig gif0 up
```

- Configuration of the IPv6 tunnel over IPv4 (gif0)

Execute the following commands:

```
 ifconfig gif0 create
 ifconfig gif0 inet6 <local end-tunnel> <remote end-Tunnel> prefixlen 128
 gifconfig gif0 inet <local ipv4addr on interface> <remote ipv4addr on interface>
 ifconfig gif0 up
```

10.7 RAT (Robust Audio Tool) installation and configuration

- Download the software from the site: http://www-mice.cs.ucl.ac.uk/multimedia/software/rat/download.html
- Execute the following commands

```
# tar zxvf rat-4.2.20.tar.gz

# cd /rat-4.2.20/tcl-8.0/unix/
# ./configure
# make

# cd /rat-4.2.20/tk-8.0/unix/
# ./configure
# make
# make install

# cd /rat-4.2.20/common/
# ./configure –enable-ipv6
# make
# make install

# cd rat-4.2.20/rat/
# ./configure –enable-ipv6
# make
# make install
```

The following command should be executed to launch the RAT application:

```
        rat-4.2.20 @ipv6_address/port
```

where:

- @ipv6_address: unicast or multicast address.
- port: the port used (> 1024).