UNIVERSITY OF AMSTERDAM

University of Amsterdam
Faculty of Science
System and Network Engineering

Multicast in a CineGrid testbed

Igor Idziejczak
$firstname.$lastname@os3.nl

My master thesis is the culmination of my work in the
System and Network Engineering Program
a one year study to obtain the degree of Master of Science

Academic year 2007-2008

University of Amsterdam
Faculty of Science
System and Network Engineering

Supervisors SNE group: Dr. Paola Grosso
MSc Ralph Koning

University of Amsterdam
Faculty of Science

System and Network Engineering
Building Leeuwenburg
Weesperzijde 190, 1097 DZ Amsterdam, the Netherlands

Tel. +31-20-595 1697

This work is the conclusion of one month of research for the Systems and Networking Engineering (SNE) research group - Optical Networking at the University of Amsterdam in the Netherlands.

Version 1.0.0 and compiled on June 30, 2008 with LaTeX.

My master thesis is the culmination of my work in the
System and Network Engineering Program
a one year study to obtain the degree of Master of Science

Academic year 2007-2008

Summary

This document describes the possibilities of sending high quality video to multiple destinations using multicast for a reliable, scalable high-bandwidth CineGrid testbed without loss of performance for the System and Network Engineering (SNE) research group of the University of Amsterdam.

The CineGrid organization focuses on tools for the distribution of very high quality digital media. This distribution requires large amounts of bandwidth and because of physical limitations, like link speeds to a maximum of 10 Gigabit, multicast would be the only viable option to send data to multiple destinations.

To guarantee bandwidth reservation, the SNE research group is researching layer 2 techniques like Provider Backbone Transport (PBT). PBT can be used to deliver a connection-oriented Ethernet protocol to photonic networks. Multicasting in a PBT network can be challenging because it can't be used with point-to-point links.

I investigate different multicast techniques divided in network layer and application layer multicasting and give an overview of traditional multicasting. I investigate SAGE bridge, an application layer multicast solution and do some test to stream high quality video to multiple displays. Sage bridge is a part of SAGE, a middleware application for streaming high quality video. I do a literature study on Provider Link State Bridging (PLSB), a network layer multicast solution. My findings concerning network layer and application layer multicasting are weighed against each other to provide a solution for the CineGrid testbed of the SNE research group.

Contents

List of Figures

1 Introduction

1.1 Preface

As part of our Master of Science study in System and Network Engineering, at the University of Amsterdam, I have done research on the topic of multicast in the Cinegrid network. The research was performed on behalf of the SNE group - Optical Networking [1], Kruislaan 403, 1098 SJ Amsterdam, under supervision of Dr. Paola Grosso and MSc Ralph Koning.

1.2 Background

The purpose of this research is to investigate the possibilities to multicast traffic in a CineGrid testbed based on Provider Backbone Transport (PBT) [2] and Provider Link State Bridging (PLSB) [3] for the CineGrid Open Content Exchange (COCE).

Multicast is used to unburden the network when streaming high quantity data to multiple destinations. This can be done by only creating copies when the links to the destinations split.

PBT can be used to deliver a connection-oriented Ethernet protocol to photonic networks. PBT uses the IEEE 802.1Qay® standard which implies that PBT is a traffic-engineered point-to-point model. Multicasting in a PBT network can be challenging because multicasting can't be used with point-to-point links. A solution is

PLSB which enables multipoint Ethernet by adding point-to-multipoint infrastructure capabilities. Another possibility is to investigate existing tools to multicast high quality video to multiple sites using application multicasting.

My research project main focus lies on the use of multicast in the CineGrid testbed to distribute content for the COCE. "CineGrid is an interdisciplinary community that focuses on the research, development, and demonstration of networked collaborative tools to enable the production, use and exchange of very-high-quality digital media over photonic networks" [4]. A COCE provides secure high quality digital media to members of the CineGrid community.

This document describes the details about my research project such as *"traditional"* multicast and PBT in the CineGrid network. For a detailed outline, see 1.5 on the following page.

1.3 CineGrid

Cinegrid, administratively based in California, started in 2006 as a non-profit international organization. Nowadays Cinegrid has members from different countries with different interests. Active members can be found in countries like Japan, the Netherlands and the USA, among others.

Cinegrid's mission statement [4] gives a good idea what CineGrid is all about. In essence, it enables research on the production, use and exchange of very high-quality digital media over photonic networks.

Most of the transport and streaming of very high-quality digital media is made possible by a virtual international organization, the Global Lambda Integrated Facility (GLIF). This organization promotes the paradigm of lambda networking to support demanding scientific applications. Lambda networking is the use of different "colours" or wavelengths of (laser) light in fibres for separate connections. Each wavelength is called a "lambda" [5].

CineGrid uses the optical networks (lambda's) of GLIF for intercontinental transport and streaming of the Cinegrid content. A most recent overview of GLIF maps can be found at [6].

The very high-quality digital media that is available in Cinegrid is called 4K, this is roughly four times 1080p HD quality. In [7] a comparison is made between different high quality media formats. It also shown that for a single 4K stream 7,6 Gbit/s is needed. In light of my research, this is a good indication that multicast would be the only viable option to actually send a stream to multiple destinations. When sending unicast, every destination will need a factor x more bandwidth. x is defined as the number of receivers to which the stream will be sent (7,6 Gb/s times the number of destinations). Current hardware cant send multiple 7,6 Gb/s streams.

1.4 Research question & scope

After conducting preliminary research, I defined my research question as follows:

> *"Investigate the possibility of sending high quality video to multiple destinations using multicast for a reliable, scalable high-bandwidth CineGrid network without loss of performance."*

Because of the limited timespan of 4 weeks I narrowed my research down to the following:

- Introduce *"traditional"* multicast protocols in layer 2 and layer 3.
- Investigate and explain existing tools (such as SAGEbridge for the SAGE environment [8]) to multicast content to multiple sites. Analyze the performance on a regular and PBT enabled network.
- Investigate the multicast possibility of PLSB applied to the CineGrid data exchange. Are there any problems that arise from using such protocol? Can PLSB be implemented at the moment? Are there other methods for multicasting in a PBT enabled network?

1.5 Outline

In chapter 2, I will talk about the different aspects of multicast to deliver multicast traffic to the receiver. First I explain the mapping between IP multicast addresses to MAC addresses and how the bridge knows which hosts want to receive which multicast addresses. Secondly, the router must know if there are multicast receivers attached. This discovery is done with Internet Group Management Protocol (IGMP) and its workings will be explained. Finally, I will define different multicast routing protocols and I'll touch on scalability issues rising from it which is important to the CineGrid network.

In chapter 3, I present a solution to the multicasting problem in CineGrid, SAGE bridge. SAGE bridge is a component of the Scalable Adaptive Graphics Environment (SAGE) and enables what I call application layer multicasting. With this component, streaming high quality video to multiple destinations becomes possible. It falls strictly taken not under my definition of multicast but I will explain why in this chapter. The installation, configuration and test result are described.

In chapter 4, I will introduce PBT and how it became to be by means of different Ethernet standards / drafts. I'll explain its use in the CineGrid testbed and provide an introduction of the switching solution, the Metro Ethernet Routing Switch 8600 (MERS 8600), focusing on configuring PBT.

In chapter 5, I will define Provider Link State Bridging (PLSB). It enables multicasting and is a technique on top of PBT. PLSB is a cutting edge technology and isn't available at the moment so I give some insight on the possibilities and approach when using it with PBT.

In chapter 6, I approach the differences between network layer and application layer multicasting. I'll state some design considerations which should be taken into account when implementing a multicast technique which are based on my literature study.

2

Traditional multicast

2.1 Introduction

In this chapter I'll discuss a more theoretical part of my research on multicasting. I don't explain or give my opinion on the necessity of multicast in WAN environments and the problems rising from it. More appropriate reading on this matter can be found in [9] [10] and [11].

Multicast can be defined as the delivery of data to a group of destinations only creating copies when the links to the destinations split [12]. While this definition is sufficient for a general view on multicast, in practice, there are exceptions which makes the definition previously stated oversimplified. An interesting point of view on an exception is from Radia Perlman about the similarities between multicast and broadcast. In [9] she concludes broadcast to be essentially the same as multicast because conceptually they are both used to send packets to everyone that is listening. With her statement, I want to show it is important to have an unified vision of the concept multicast. Misconception can lead to confusion as explained with the statement above. A correct notion of multicast will become significant for my research. In 3 on page 11, I talk about possible multicast techniques. With multicast explained, my choices of techniques are irrefutable multicast techniques.

Multicast transmission mechanisms are available for both layer 2 and layer 3 of the OSI model. To deliver multicast traffic from an IP network to a LAN, two problems

occur. First of all, what data link layer destination address should be used? Secondly how does the bridge know which hosts want to receive which multicast addresses? The same problems exists on layer 3. How does a router know which host wants to receive which multicast addresses and how is it propagated throughout the network?

2.2 Multicast addresses

2.2.1 IP addresses

I only talk about multicast addresses in IP on layer 3. In my case the only importance is to understand the concept of multicasting and multicast mapping and I do this by means of IP. Other protocols were a viable decision but I choose IP because it is globally adopted and the Internet depends on IP.

In IPv4 class D is reserved for multicast groups and have the top 4 bits (1110) and are defined in "IANA Guidelines for IPv4 Multicast Address Assignments" RFC 3171 [13]. In other words, multicast addresses are in the range 224.0.0.0 to 239.255.255.255.

In IPv6, ff00::/8 is reserved as multicast space and multicast addresses are defined in "IP Version 6 Addressing Architecture" RFC 4291 [14]. This defines fixed scope and variable scope multicast addresses. They have the following format as depicted in figure 2.1:

8	4	4	128
1111 1111	flgs	scope	group ID

Figure 2.1: Multicast address format

flgs are flags which can be set to indicate a permanently or non-permanently assigned multicast address. Scope is a value used to limit the scope of the multicast group.

2.2.2 MAC addresses

When a IP multicast address is transmitted onto the LAN, one possibility for choosing the data link layer address would be the layer 2 broadcast address [9]. This would impose great problems on the local network. Imagine a theoretically multicast pool of 2^{28} addresses (the top 4 bits are constant as I talked about in 2.2.1) and all these addresses would translate to the layer 2 broadcast address. This would cause a lot of useless bandwidth usage. Privacy would be a second problem. One could promiscuously listen

to network traffic and receive all multicast traffic without being eligible to inspect this data.

A better method would be to map every IP multicast address to a corresponding layer 2 multicast address. To do this, a mechanism must be defined so that multiple hosts can receive the same packet and be able to differentiate between multicast groups [15]. In the IEEE Std 802® 2001 (R2007) standard [16], the first bit of the first octet is used as the group/individual bit. This bit makes a distinction between a particular station and logical groups of stations. Multicast uses this bit, set to 1, to transmit IP packets to a group of hosts.

Because there are 2^{28} possible IP multicast addresses, this would require 20 bits of high-order constant (a MAC address is 48 bit) in order to do a one to one mapping [9]. An ordinary block assigned from the IEEE [1] gives 24 bits of high order constant and is called an OUI . An Organizationally Unique Identifier (OUI) or company id is a 24 bit unique assigned number that uniquely identifies a organization. In an Ethernet MAC address, a 48 bit address, together with the 24 bit OUI, is used to uniquely identify a piece of hardware. In order to map an IP multicast address to a data link multicast address 16 consecutive address blocks are necessary. Because this was found impractical to both the IETF and the IEEE [9], the IEEE assigned a 24 bit address block (01-00-5E) to IANA [17]. This block (01-00-5E-00-00-00 to 01-00-5E-7F-FF-FF) is used to map IP multicast to layer 2 multicast addresses by mapping only the bottom 23 bits into the layer 2 block. Concretely, this means that 32 (2^5), because 28 bits - 23 bits = 5 bits, different IP multicast addresses will map to the same data link multicast address. The top bit is used for MPLS multicast and *"Internet reserved by IANA"* which is not yet assigned for any purpose [17].

2.3 IGMP

IGMP stands for Internet Group Management Protocol and this protocol is used by IPv4 hosts [2] to communicate multicast membership states to multicast routers [15]. This is important so the router knows which hosts are members of which multicast addresses. *IGMP v1* contains two types of messages [18]:

- Host Membership Query (0x11)
- Host Membership Report (0x16)

Multicast routers send Host Membership Queries to discover which host is member of a multicast group. Queries are sent with address 224.0.0.1 or the all-hosts group. In 2.1 on page 5 we talked about the similarities between broadcast and multicast and

[1] A registration form can be found at `http://standards.ieee.org/regauth/oui/forms/`

[2] I say IPv4 hosts because the IGMP protocol is an integral part of IP

Radia Perlman's conclusion that broadcast is essentially the same as multicast. In IGMP, we see that the multicast address 224.0.0.1 is actually a broadcast to all IP enabled hosts on the local network. These hosts must listen to 224.0.0.1 and thus only multicast enabled hosts will be addressed. Host Membership Queries are also sent with a Time To Live (TTL) of 1. This ensures that the queries can only be propagated on the local network.

Hosts respond to these Queries by generating Host Membership Reports. They report every multicast group for which they are a member (host groups) on the interface the Query was received. To avoid traffic bursts of Reports sent to the multicast routers, two techniques are used [18]:

- Timers are used to spread the sending of Reports to the multicast routers.
- Host Membership Reports are sent with the IP destination equal to the host group address. If another host of the host group hears the Report, it will stop the timer. In a normal case only one Report for a host group will be send.

There are exceptions to these rules and can be found in RFC 1112 [18].

IGMPv2 adds the ability to *instantly* resign from a host group as specified in RFC 2236 [19]. In IGMPv2, two new types are defined:

- Leave Group (0x17)
- Version 1 Membership Report (0x12)

In a PBT enabled network IGMPv2 doesn't add any advantages for the CineGrid network. IGMPv2 adds a lot of overhead and protocol complexity with type 0x17. IGMPv2 routers must be manually configured for compatibility with IGMPv1 routers.

IGMPv3 adds the ability to join a host group with only a specified set of senders. In older versions the mapping between layer 3 and layer 2 is done with a layer 2 multicast address derived from the IP multicast address. In *version 3* all IGMP replies are sent to a single layer 2 multicast address listened to by the switch [20] and [21]. Closely related to this is IGMP snooping which we will talk about in 2.4.

2.4 IGMP snooping

A switch uses IGMP snooping to know on what port to send the multicast and is used to conserve bandwidth [3] as defined in "Considerations for Internet Group Management Protocol (IGMP) and Multicast Listener Discovery (MLD) Snooping Switches" RFC 4541 [22]. The switch analyses traffic between hosts and multicast routers. When the switch hears IGMP traffic it can add or delete the host's port number to the multicast

[3] In *"normal"* switch behavior, multicast traffic is typically forwarded on all interfaces

list for that group accordingly the type of IGMP message received. A switch listening to IGMP messages do not adhere to the conceptual model that provides the separation between different layers of the OSI model.

2.5 Multicast routing protocols

The only thing missing now is how multicast is routed in the network. A multicast network requires a mechanism to build distribution trees that have a unique forwarding path between the subnet of the source and each multicast member subnet [15]. There are different multicast routing protocols to do this, like Distance Vector Multicast Routing Protocol (DVMRP), Multicast OSPF, Core-Based Trees (CBT) and Protocol-Independent Multicast (PIM) among others. It is currently the most widely used protocol [15] and uses routing information supplied by other routing protocols. This makes PIM *independent* so that various routing protocols can be used like OSPF and BGP among others. All these protocols belong to one of two categories which I will discuss next:

- Dense-Mode (DM)
- Sparse-Mode (SM)

2.5.1 Dense Mode

PIM-DM, a DM implementation, is defined in RFC 3973 [23]. More generally, DM protocols are designed on the assumption that large numbers of multicast routers will need to distribute multicast traffic. The source initially broadcasts to every router, and thus every node. Then each node that does not wish to receive packets destined for that group will send a prune message to its router. Upon receiving a prune message, the router will modify its state so that it will not forward those packets out that interface. If every interface on a router is pruned, the router will also be pruned [24].

There are different DM implementations, like PIM, but generally the concept stays the same. DM's flooding technique makes it poorly scalable. In the CineGrid network, were scaling is very important, this could be a less viable options.

2.5.2 Sparse Mode

PIM-SM, a SM implementation, is defined in RFC 4601 [25] and RFC 5059 [26]. More generally, SM protocols are designed on the assumption that only few routers in the network will need to distribute multicast traffic for each multicast group [27]. A host sends a join message to an adjacent router. In its turn, the router will tunnel it to a *"designated router"* (DR) using information of the underlying routing protocol. A

router is elected *"designated router"* by means of the highest priority / address. This router will send it on the multicast tree [9].

Like DM, SM has different implementations among which PIM. The concept of SM is generally the same in spite of different implementations. SM is different from DM by using a *pull* model [4] in contrast with a *push* [5] model in DM protocols.

SM also has some scaling issues. First of all, there is a *"designated router"* election which is a bit noisy. Possible DR routers flood advertisements throughout the domain. Secondly, in a topology were multicast members are geographically dispersed one DR router will introduce latency. This is the consequence of an election in which the location of group members is not taken into account.

2.6 Review

In this chapter I talked about all the techniques and protocols necessary to successfully send multicast from the sender to the multicast member host. It is important to understand traditional multicast because enabling multicast in a PBT network doesn't mean multicast in CineGrid will work. I'll introduce some multicast possibilities and what could be a viable solution for the CineGrid testbed throughout the next chapters.

[4] The data is retrieved when asked for
[5] Data is flooded to all the devices

3
SAGE bridge

3.1 Application multicast

In 2 on page 5, we gave a definition about multicasting where I stated that multicast is the delivery of data to a group of destinations only creating copies when the links to the destinations split. I also stated later that I would discuss multicast techniques that weren't strictly multicast. In this section, I will talk about such a case. Application layer multicast is the delivery of data to a group of destinations but in this case it uses unicast. By using an *"overlay"* structure, I can still speak of multicast. In this sense, I can call multicast, as in 2 on page 5, traditional *"network layer multicast"*.

Multicasting is brought to life for a special purpose. In the case of network layer multicast it is to unburden the network. Application layer multicast is a workaround because of the lack of support for network layer multicast on the Internet [28] and to address practical problems with network layer multicast, like vulnerable to flooding attacks without network management and hard to provide reliability and congestion control at a higher level [29]. Network layer and application layer multicast can be both a means to an end for the CineGrid testbed.

I have looked at application layer multicasting to provide the CineGrid testbed a multicast solution. Application layer multicasting has the advantage that it can be used in every network. As seen in 2 on page 5, a lot of components are necessary to enable *"traditional"* network layer multicast. In the case of a PBT enabled CineGrid

network this is a problem for sure. PLSB, as I will talk about in 5.2 on page 30, is a *"cutting-edge"* technology and not yet available for production. A disadvantage application layer multicasting is that unburdening the network, as *"traditional"* network layer multicasting can, will never be achieved. On the other hand, it can help in alleviating the network by placing intermediate nodes on the network from where the application layer multicast is started. I'll talk about how this is done in SAGE bridge in 3.3.4 on page 17.

Application layer multicasting doesn't have to be a compromise when correctly implemented. SAGE bridge is a possibility to do application layer multicasting and is a component of SAGE [8].

3.2 SAGE

Scalable Adaptive Graphics Environment (SAGE) is a specialized middleware for streaming high-definition video and high-resolution graphics in real-time from remotely rendering and storage clusters to scalable multiple displays over very fast networks [30].

SAGE consists of the Free Space Manager, SAGE Application Interface Library (SAIL), Sage Receiver and the User Interface (UI) client [31] as depicted in figure 3.1.

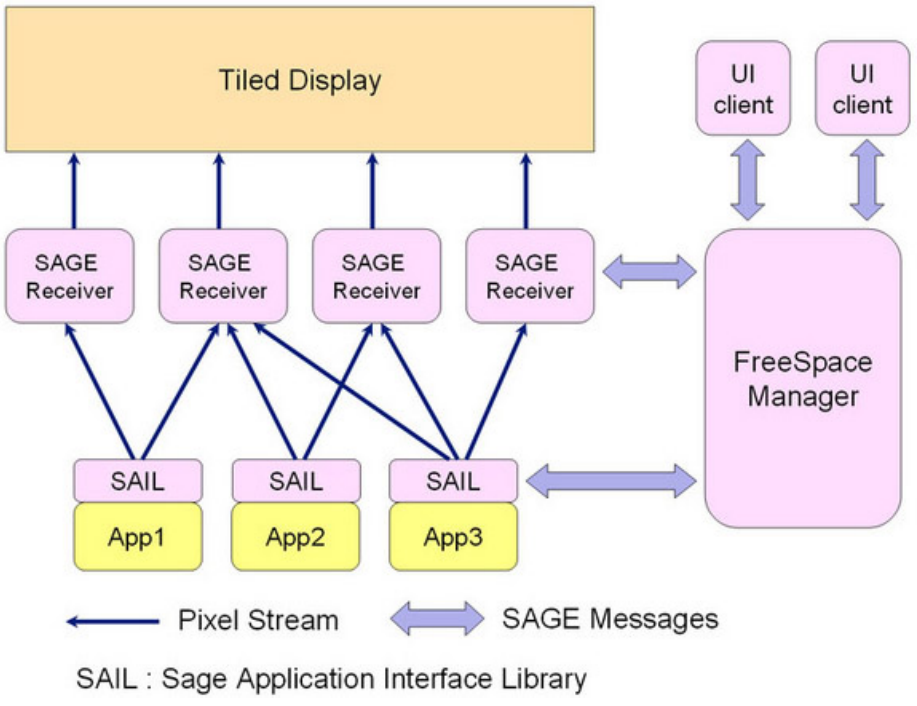

Figure 3.1: SAGE architecture. Source [30]

The Free Space Manager receives user commands from the UI and controls pixel streams between SAIL and the SAGE Receivers. SAIL captures output pixels from applications and streams them to the appropriate SAGE Receivers. A SAGE Receiver can get multiple pixel streams from different applications, and displays streamed pixels on multiple tiles. A UI sends user commands to control the Free Space Manager, and receives messages that inform the current status of SAGE [31].

3.3 SAGE bridge

3.3.1 Introduction

SAGE Bridge receives pixel streams from applications and distributes to multiple SAGE sessions [32]. By doing this, it is possible to alleviate network usage by placing these SAGE bridge after very costly links. In this case, you only have to send one unicast over the costly link to the SAGE bridge. From there you can have multiple *"unicast"* streams to multiple destinations.

To realize this, I made a setup in which I have two SAGE receivers with both a screen to display the streams. I have one storage node which is also configured with SAGE and in my case with SAGE bridge from where I'll start the streaming. This topology is depicted in figure 3.2 on the following page.

My goal with this setup is to describe how SAGE bridge is installed, configured and if content can be successfully streamed. Successful, in this context, means will the SAGE bridge be sending the stream to multiple SAGE receivers? It is important to note that the SAGE bridge will need a factor x more bandwidth. x is defined as the number of receivers to which the stream will be sent. In my test it isn't of great importance but it must be kept in mind when placing these bridges.

3.3.2 Installation

To test application multicasting by means of SAGE bridge, 3 nodes are installed with SAGE, as depicted in figure 3.2 on the next page. The installation of SAGE can be found in A.3 on page 42.

I have a central node with SAGE bridge installed from where I stream high quality video. Two nodes are running Free Space Manager and are SAGE receivers. These nodes are display nodes and have a screen attached to display the video streams. All the nodes run tcpdump [33] to check the data flows and to see of the streams originate from the SAGE bridge node.

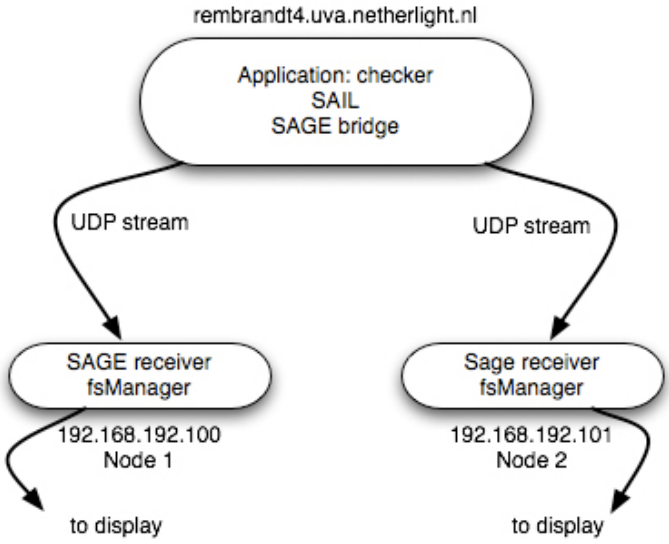

Figure 3.2: Logical representation of SAGE config

3.3.3 Configuration

Central node

The central node will be configured for SAGE bridge. It will also execute the application. I will use a very simple program checker, used for screen testing.

Let's first edit sageBridge.conf in $SAGE_DIRECTORY/bin:

```
masterIP 192.168.192.14
slaveList 0
streamPort 41000
msgPort 42000
syncPort 43000
screenRes 1000 1000

rcvNwBufSize 65536
sendNwBufSize 65536
MTU 9000
```

The masterIP is the IP address of the master node of SAGE bridge cluster. In this example, I only have one bridge and this is the localhost. The slavelist can be configured to note all the nodes in de bridge cluster. The following three fields are already configured and these standard ports should be left alone unless an application is already using these ports or firewall rules are defined to block these ports.The screenRes can be set to for debugging. rcvNwBufSize, sendNwBufSize and MTU are network parameters used for the pixel streams between the checker application and the SAGE bridge.

The application used for the pixel streams must also be configured with following configuration parameters:

```
bridgeOn true
bridgeIP 192.168.192.14
bridgePort 42000
fsIP 192.168.192.101
fsPort 20002
```

With `bridgeOn true`, I am saying that this application uses SAGE bridge. With bridgeIP, I define the master IP of the SAGE bridge. The bridgePort is standard configured and left alone. fsIP and fsPort are used to define the IP and system port of the first fsManager. In our case it is Node 2.

SAGE receivers

On the SAGE receivers, two configuration files must be adapted. The first is the fsManager.conf and is being used to run the Free Space Manager. The second configuration file is the stdtile-1.conf and is used to display the pixel stream on the screen.

Because we have two display nodes, we also have two Free Space Managers running on each node. The only thing that changes, is the IP address of the Free Space Manager.

```
fsManager       local 192.168.192.100
systemPort      20002
uiPort          20001
trackPort       20003
conManager      206.220.241.46 15557

tileConfiguration stdtile-1.conf
receiverSyncPort  12000
receiverStreamPort   22000
receiverBufSize    100
fullScreen     0
winTime 0
winStep     1

rcvNwBufSize 65536
sendNwBufSize 65536
MTU 9000
```

In the fsManager.conf configuration file, we filled the IP address of the fsManager which is the local IP of the node. We also changed the Maximum Transmission Time (MTU) to 9000 which means we can send files with a package size of 9000 Bytes.

The tile configuration file stdtile-1.conf is the same for both the display nodes but can be different. My configuration looks like this:

```
TileDisplay
Dimensions 1 1
Mullions 0.0 0.0 0.0 0.0
Resolution 1920 1080
PPI 90
Machines 1

DisplayNode
Name local
IP 192.168.192.100
Monitors 1 (0,0)
```

In the TileDisplay, I define the screen. I connect a single screen so the number of columns and rows is 1. The value for the parameter mullions is 0 because we only have one screen. PPI is the pixels per inch. Machines represent the number of display nodes which drive each display.

In the second block, DisplayNode, I define the configuration for the display node. In my case, I only have one display node with one monitor.

To use SAGE bridge to stream to multiple destinations, the following must be executed in the right order:

- Execute the binary ”sageBridge” in $SAGE_DIRECTORY/bin
- Execute ”fsManager” on both the display nodes
- Launch the checker application. The pixels are streamed to the SAGE bridge and to the first SAGE session. In my case node 192.168.192.101.
- To share this application to another SAGE session, execute uiConsole to be connected to the first Free space Manager (usint the fsManager.conf file). Execute following command:

  ```
  uiConsole fsManager.conf
  share appID fsIP2 fsPort2
  ```

  The appID is the number of the application and these numbers are assigned in execution order. In my case, it would always be 0 because I only start one application. fsIP2 and fsPort2 are the IP and port number of the second fsManager.

3.3.4 Results

When executing the binary sageBridge, it initializes successfully:

```
SAGE Bridge : tcp network object was initialized successfully
SAGE Bridge : udp network object was initialized successfully
```

At this moment the SAGE bridge accepts any incoming requests. When starting the fsManager on both the display managers, I see a successful initialization:

```
cgdemo@demo2:~/sage3.0/bin$ ./fsManager
fsManager using configuration file <fsManager.conf>
SAGE Display Manager : start sync server
SAGE Display Manager : register to a Free Space Manager
sysClient 0 connected
SAGE Display Manager : initialization message was successfully parsed
sdlSingleContext:init(): Window created
Display Manager is creating display object
Display Manager is initializing network objects....

SAGE Display Manager : tcp network object was initialized successfully
SAGE Display Manager : udp network object was initialized successfully
Connected to sync master 192.168.192.101:12000
```

A similar message is shown for the second display node 192.168.192.100. Now, I can start streaming and I do this by executing the application `./checker`:

```
cgdemo@rembrandt4:~/sage3.0/bin$ ./checker
ResX = 400  ResY = 400

try to connect to 192.168.192.14:42000
```

The checker application will try to connect to the SAGE bridge (192.168.192.14) and SAIL with send a register message to the SAGE bridge. After a negotiation the SAIL will send the stream to the SAGE bridge:

```
send SAIL register message to sage bridge
init config ID 0
sail initialized

SAIL is initializing network connections for streaming
sageStreamer::initNetworks : initialize UDP object
sageStreamer : network object was initialized successfully

1 connections are established
SAIL: Network connection Done
```

This negotiating can also be seen on the SAGE bridge:

```
msgClient 0 connected
init config ID 0
sageStreamer::initNetworks : initialize UDP object
sageStreamer : network object was initialized successfully
1 connections are established
```

At this point, I can see the application checker on the screen of the first display node. To display the stream to the second display, I opened a uiConsole to the primary Free Space Manager (192.168.192.101):

```
cgdemo@rembrandt4:~/sage3.0/bin$ ./uiConsole fsManager.conf
Message : 40004
1 1 1
1920 1200
1920 1200 0

Message : 40001
checker 0 50 1650 50 1250 1001 0 0 0 0

share 0 192.168.192.100 20002
```

The 40004 message shows SAGE display info. The 40001 message is information about the application like checker is the name of the application with application ID 0. Also the SAIL ID (1001) and the dimensions (50, 1650, 50, 1250 [1]) are shown. The last line is the command to share the stream to the second display and is in following format `share appID fsIP2 fsPort2`. The Free Space Manager will send this message to the SAGE bridge and will interpret this message:

```
app share message : 192.168.192.100 20002

sageStreamer::initNetworks : initialize UDP object
sageStreamer : network object was initialized successfully
1 connections are established
```

Now, an extra connection is made to the second display. When looking at the Free Space Manager of the second display node, we can also see that the stream is successfully send:

```
sysClient 1 connected
Establishing network connections for streams......
connect message sent
init stream message sent
```

[1] from left to right and from bottom to top

The screen of the second display doesn't show the checker application. When looking at the tcpdump of the SAGE bridge, I can see that the SAGE bridge (192.168.192.14) is sending two streams to 192.168.192.100 and 192.168.192.101 with MTU 9000:

```
"87259","109.652273","192.168.192.14","192.168.192.101","UDP",
"Source port: 44880  Destination port: 56235"
"87260","109.652341","192.168.192.14","192.168.192.100","UDP",
"Source port: 52846  Destination port: 39904"
```

As a result, I can say that application layer multicasting with SAGE bridge works but the stream to the second display doesn't show on the screen. What causes the black screen on the second display node is unknown. I didn't had time to investigate the cause of the black screen. The SAGE documentation [34] notes that the SAGE bridge software is still a bit buggy.

4

Provider Backbone Transport

4.1 Introduction

I'll introduce Provider Backbone Transport (PBT) and provide the technical details important for me to examine the possibilities of multicasting in a PBT enabled CineGrid network. I will talk about about the advantages / disadvantages of PBT in CineGrid in 4.3 on page 26.

Ethernet was originally developed in 1973 and during the decades it has proven its worth in Local Area Networks (LAN's). Ethernet's simplicity made it easy to operate and through commoditization Ethernet became very cost-effective [35]. However in larger, circuit-switched networks (MANs or WANs) it is a completely different story. Adding the Ethernet protocol, because of its cost-effectiveness, to circuit-switched networks is very much wanted in the industry. This evolution is called *"Carrier Ethernet"* and defined by the Metro Ethernet Forum (MEF) [35]. The differences between Carrier Ethernet and *"traditional"* LAN based Ethernet are standardized services, scalability, reliability, Quality of Service and service management. More information can be found in [36].

PBT is built upon the IEEE 802.1® standard and is actually an extension of Provider Backbone Bridges (PBB) among others. To understand PBT, these extensions will be explained. PBT enables the creation of reliable layer 2 circuit-switched point-to-point Ethernet tunnels. This has an impact on the CineGrid network, explained later in this

chapter and an impact on how multicast can be realized in CineGrid explained in 3 on page 11.

4.2 Ethernet standards

Different IEEE 802® committees like 802.1®, 802.2® and 802.3® deal with different layer 1 and layer 2 issues. The 802.1® committee deals with issues common accross all 802® LANs including addressing, management and bridges. PBT is gradually built upon the protocols 802.1®, 802.1q®, 802.1ad® and 802.1ah® as depicted in figure 4.1.

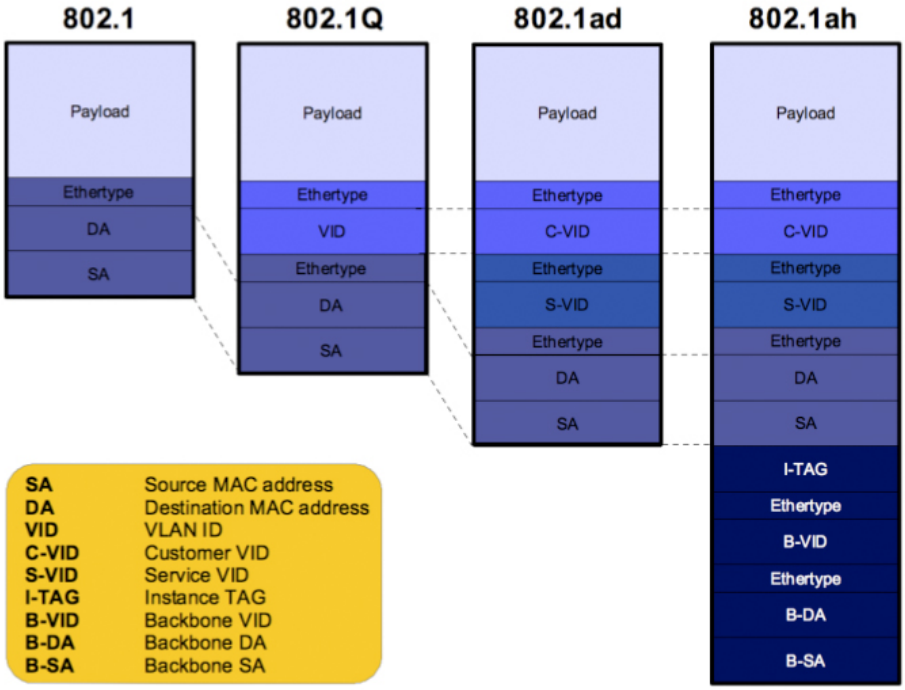

Figure 4.1: Addition to the Ethernet frame for PBT. Source [37]

In the next sections, I'll explain Virtual LANs (VLANs) in relation with PBB / PBT as defined in 802.1q®. Next, I'll talk about Provider Bridges (PB) as proposed in draft 802.ad® and PBB as specified in 802.1ah®. Finally I'll elaborate on PBT which is defined in 802.1Qay® and is also called Provider Backbone Bridges Traffic Engineering or PBBTE or PBT in short. 802.1® just defines a standard layer 2 frame as depicted in figure 4.1 & figure 4.2 on the next page and I will not elaborate on that. It is enough to know that such a frame consists of a destination address, a source address and a protocol type field. The last is being used for multiplexing.

4.2.1 802.1q®

VLANs are defined in IEEE Std 802.1Q™-2005 [38]. VLANs are actually the same as LANs in terms of the territory over which a broadcast / multicast is sent (broadcast domain) [9]. VLANs are used to communicate with each other as if they were attached to the same wire in spite of physical boundaries. "A VLAN has the same attributes as a physical LAN, but it allows for end stations to be grouped together even if they are not located on the same LAN segment" [39]. The difference in frames between LANs and VLANs is depicted in figure 4.2.

Ethernet packet without VLAN tag:

6 octets	6 octets	2	···
destination	source	ptype	data

Ethernet packet with VLAN tag:

6 octets	6 octets	2	2	2	···
destination	source	81-00	V-TAG	ptype	data

Figure 4.2: Packet without and with VLAN tag

The first field after the source is the protocol type field of the original LAN packet and is used to indicate that the packet is VLAN tagged. The following 2 bytes field is the VLAN tag (V-TAG) also named Q-tag. In this field 3 bits are reserved for *priority* called the Priority Code Point (PCP), 12 bits for a VLAN ID (VID) and 1 bit to indicate a canonical format or not, called the Canonical Format Indicator (CFI) field. A VLAN tag is depicted in figure 4.3.

3 bits	1 bit	12 bits
PCP	CFI	VID

Figure 4.3: VLAN tag field

The CFI field is always set to zero for Ethernet switches. CFI is used for compatibility between Ethernet and Token Ring networks. If a frame, received at an Ethernet port, has a CFI set to 1, then that frame should not be bridged to an untagged port [40].

4.2.2 802.1ad®

Provider Bridges are defined in [41] and it is an amendment of IEEE Std 802.1Q™-2005 [38]. Provider Bridges were created because of two reasons. First of all, a the-

oretical maximum of 4094 VLANs can be created to divide a LAN. I say only 4094 VLANs because of the 12 bits (2^{12}) but without the VID values 0 and 1. The null VID indicates that the tag header contains only priority information. The Port VLAN ID (PVID) with value 1 is the default and members are untagged which means Ethernet packets are not VLAN tagged.

Secondly, a Service Provider (SP) couldn't make a separation between internal VLANs and customer VLANs. By creating Provider Bridges a SP could create a VLAN for a customer without cooperation of all the customers. Also, a customer isn't bound to the VLANs assigned from the SP.

Provider Bridges are also called Q-in-Q and this actually holds the solution. PB extend the original concept of VLANs by simply adding a new Q-tag in the header. Practically it means that the new Q-tag is used to allow Service Providers to administer their own VLANs (for example to identify a customer network). The *"original"* Q-tag is used to identify VLANs within the customers network. The customer and the Service Provider are still bound to a VLAN limit of 4094 VLANs. An example packet is depicted in figure 4.4.

6 octets	6 octets	2	2	2	2	2	···
destination	source	81-a8	S-TAG	81-00	C-TAG	ptype	data

Figure 4.4: PB packet

The first protocol type field is 81-a8 to identify the field S-TAG. S-TAG is Service VLAN tag and this field allows SPs to administer their own VLANs. The next field is a protocol type field again. It is the *"original"* VLAN tag but called C-TAG here. C-TAG stands for Customer VLAN tag and the customer can use this field to create VLANs.

4.2.3 802.1ah®

Provider Backbone Bridges are defined in [42] and is amendment 6 of IEEE Std 802.1Q™-2005 [38]. It is sometimes called MAC-in-MAC because it encapsulates the 802.1ad® header in another MAC header [37]. 802.1ah® and 802.1ad® are developing standards according to Nortel [43] and both developed to address the scalability issues in large networks.

I will try to explain PBB with the help of a simple example. Simple in the meaning of a simple network topology with only a view customers as depicted in figure 4.5 on the next page. PBB makes use of a 3-tiered architecture. In figure 4.5 on the following page these 3 tiers are each shown with a unique color. We distinguish the Customer, the Provider and the Provider Backbone. The red crosses through some of the links

mean that Spanning Tree Protocol (STP) is active and these links are in a blocked state to prevent loops. Let's say Customer A on Site 1 sends a frame to Customer A but on Site 3. This frame is sent through the customer bridge and the provider bridge to the Provider Backbone Edge Bridge (PBEB). These are switches connecting the Provider network with the provider backbone and run PBB.

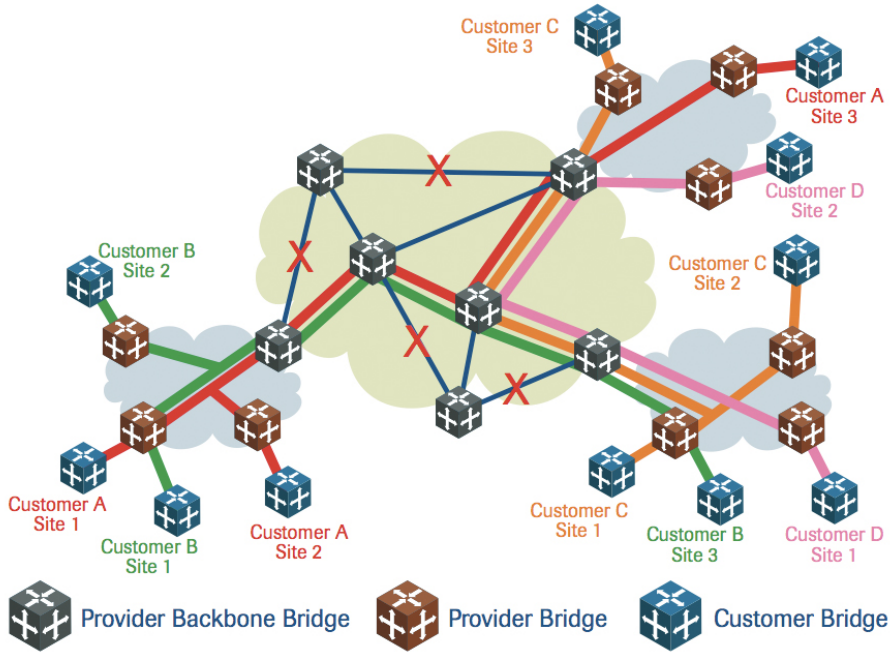

Figure 4.5: PBB network example. Source [44]

The PBEB adds a backbone MAC header to the customer's ethernet frame. The PBEB forwards the frame to the PBEB connected to the destination using the Backbone MAC header Destination Address field. Forwarding decisions are based on a combination of VID and MAC address and the combination is globally unique. The frame travels through the backbone network (like an *"ordinary"* frame) [1] until it reaches the PBEB at the other side. There the backbone MAC is removed and the customer's frame is sent through the provider bridge to the customer. Important to note for enabling multicast is that PBB networks provide multi-point tunnels between PB networks.

To provide this kind of hierarchical scalability new fields are introduced in 802.1ad® and these are depicted in figure 4.6 on the next page.

The B-DA en B-SA are respectively the Backbone Destination Address and the Backbone Source Address. These addresses are used to send frames within the Provider Backbone. The B-TAG is the Backbone Tag and consists of two fields the Tag Protocol

[1] Ordinary means that not PBB enabled switches see a regular PB frame. This has been done for backwards compatibility.

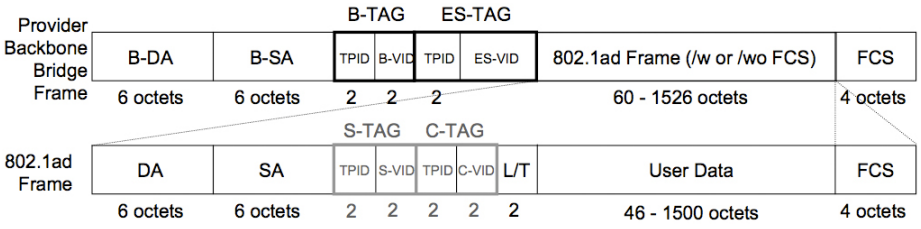

Figure 4.6: 802.1ad® header and PBB header. Source [45]

Identifier (TPID) and the Backbone VLAN Identifier (B-VID). The TPID is used to identify the B-VID. The B-VID contains the VLAN that carries the traffic through the backbone network. The ES-TAG is the Extended Service-VLAN Identifier and consists of two fields the Tag Protocol Identifier (TPID) and the ES-VID. The TPID is again used to identify, in this case, the Extended Service-VLAN Identifier (ES-VID) also called the Instance-Service Identifier (I-SID). The I-SID in the I-TAG is associated with a C-VID, S-VID or both [46].

4.2.4 802.1Qay®

Provider Backbone Transport also called Provider Backbone Bridges-Traffic Engineering (PBB-TE) is defined in [47] and is amendment 6 of IEEE Std 802.1Q™-2005 [38]. PBT, concerning its frame header, is build upon a couple of extensions as discussed in the previous sections. An overview of these frame headers can also be found in figure 4.1 on page 21.

The inherent problems of Ethernet make it unusable in large network, like MANs and WANs, because of its connectionless behavior. Ethernet has a *"flooding proces"* which is used when the destination is unknown. All ports will receive a copy of the data to eventually learn the destination and save this information in a forwarding table. This is called the *"learning"* process [35]. A second problem were the VLANs with only a maximum of 4094 VLANs as explained in 4.2.1 on page 22. The first step towards a solution was with the introduction of PBB, explained in 4.2.3 on page 23. For Ethernet to become truly a *"Carrier Ethernet"* more requirements are needed. These requirements are, to note a few, scalability, reliability, resiliency and Quality of Service. A solution to this could be PBT.

The PBT frame format is the same as the PBB frame header. PBT packets are switched based on VID and B-DA, together they also form a 60 bits globally unique identifier.

PBT encapsulates data and uses a tunneling technology to transport the data from the customer across a provider backbone. It also has a mechanism to ensure end-to-end reliability [46]. In practice, PBT sets up a primary and an optional secondary

(backup) path with QoS to the destination using point-to-point tunnels. PBT disables MAC address learning and Spanning Tree Protocol (STP) and flooding (broadcasting when not knowing the destination) [46]. In the remaining section, I will discuss the important PBT elements and is mainly referenced from [46]:

- PBT creates trunk groups containing unidirectional trunks .
- Service instances (SI) define the remote user-to-network interfaces (UNI) destinations.

A PBT trunk is not one trunk. It is actually a pair of unidirectional trunks going in opposite directions and share the same endpoints. These trunks use a defined route across a provider's network and use a separate VLAN for each direction. PBT uses the VID and the destination MAC address to identify a trunk. Traffic engineering and QoS is configured for each of these PBT trunks.

A Service Instance (SI) is like a regular Virtual Private Network but encapsulates the customer frame. Each SI is associated with a trunk that carries the service from point to point. The customer is assigned a number of Service Instance Identification (I-SID) numbers. The I-SIDs are used to transport customer traffic between endpoints with the same customer domain. SI links can be seen as links from the customer to the provider backbone.

4.3 PBT in CineGrid

Currently all high quality CineGrid content is mainly streamed over optical links because of its specific properties (low latency, fixed bandwidth, low jitter). To setup an optical path between two CineGrid locations costs a lot of manual labor. People of various organizations have to work together and communicate in order to properly configure their equipment. This is of course very error prone [48].

PBT for CineGrid has many advantages and can be a solution to the problems above that occur now in the SNE-group network located in the LightHouse. The LightHouse is a collaboration of the UvA [49] and SARA [50]. This network is a shared network meaning that different applications use it. The CineGrid testbed is such an application.

Nortel, founder of PBT, defines a lot of advantages for the use of Carrier Ethernet and PBT in particular. I'll talk about the possible important improvements for the CineGrid network.

The SNE research group has a testbed for PBT, as depicted in figure 4.7 on the next page, and sees a lot of potential in PBT because of its reliability and traffic engineering like QoS.

Reliability is important especially for demanding applications like sending high quality video in CineGrid. Nortel states that this can be done by provisioning an ad-

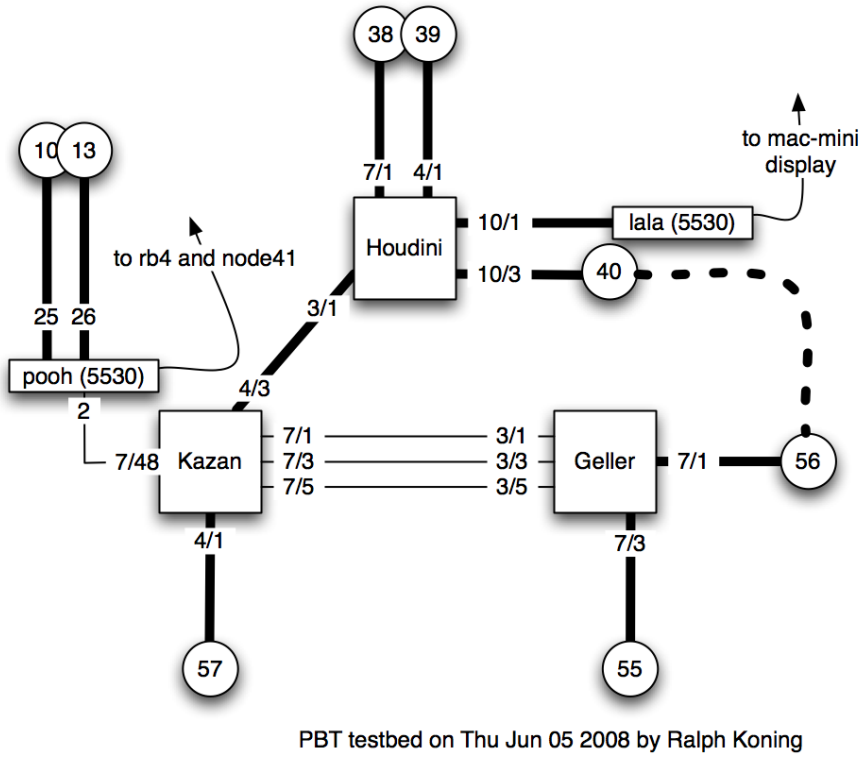

Figure 4.7: PBT testbed. Source MSc Ralph Koning

ditional backup route. In combination with the IEEE 802.1ag® [2] (Connectivity Fault Management) the working and protection paths enable PBT to provide a <50ms recovery [35]. *Traffic engineering* is an important Ethernet service feature. With, for example, QoS, bandwidth reservation can be configured. This insures that the necessary throughput can be achieved when streaming high quality video. This is important because every network hick-up can be seen in the video output.

4.4 Nortel MERS 8600

Two Nortel MERS 8600 switches are used to test PBT in CineGrid. The test bed of the two MERS's (Kazan and Geller) is depicted in figure 4.7. A third Nortel MERS 8600, named Houdini, is a production switch but can also be used in a test environment.

The MERS 8600 is an aggregation switch used to provide Carrier Ethernet in MANs or WANs. Nortel calls it a *"metro solution"*. The role of the aggregation switch,

[2] This standard will provide capabilities for detecting, verifying and isolating connectivity failures in such networks

located at the Service Provider, is to provide service encapsulation for Ethernet virtual private networks (EVPNs). Concretely, a MERS 8600 is a connection point between the customer, where it provides EVPNs with the customer, and the provider backbone, where it provides Provider Backbone Transport tunnels over the provider network.

In the SNE test setup as depicted in figure 4.7 on the previous page, PBT tunnels are going to be established between all switches. In the preliminary tests only PBT links between Kazan and Geller are configured because in this setup one Gigabit links can be easily filled up.

The customer would be in this case the researchers conducting their research, the provider backbone is the PBT testbed. The Service Provider is the SNE research group.

5 Provider Link State Bridging

5.1 Network multicast

In chapter 2, I talked about traditional multicast and in chapter 3, I talked about application layer multicasting which redefined traditional multicasting to traditional network layer multicasting. Traditional multicast is a prerequisite to investigate multicasting techniques in PBT. I already proposed a solution without involving the network. This could be done with SAGE bridge, an application layer multicast solution for SAGE. In this chapter I propose a network layer multicast solution for the CineGrid.

PBT uses deterministic point-to-point links which disables STP or the learning process in bridges. A *"network layer multicast"* solution could be Provider Link State Bridges (PLSB). In PBT, the control plane [1] is used to manage the Forwarding Information Base (FIB) [2]. Provider Link State Bridging (PLSB) uses link a state protocol and computation to populate forwarding tables to construct shortest path loop free connectivity for an 802.1ah® Provider Backbone Bridge Network (PBBN) [51]. For PLSB another control plane is added using the configured option to other traditional control planes [52] and is depicted in figure 5.1 on the following page.

Although PLSB creates a new control plane, the data plane [3] is still Ethernet, and

[1] The control plane is concerned with drawing the network map, or the information in a Forwarding Information Base that defines what to do with incoming packets

[2] Forwarding decisions are made based on the FIB

[3] Data Plane or User Plane is the part of the network that carries its users' traffic [53]

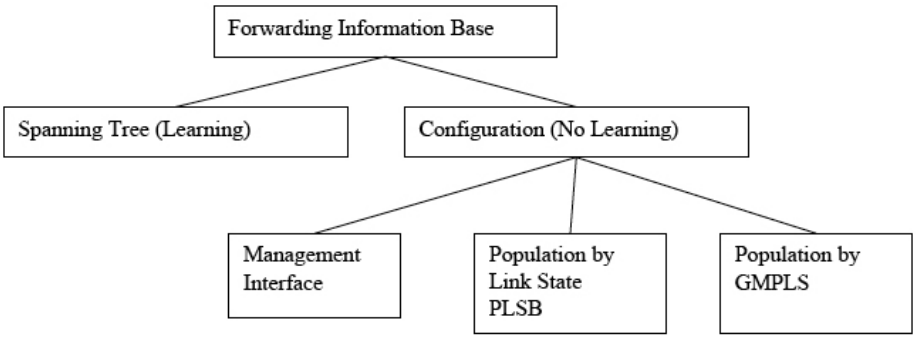

Figure 5.1: Different control planes. Source [52]

Ethernet forwarding is still based on the standard Ethernet header.

The only thing that has changed is that for a reserved number of VLANs the forwarding tables are configured by IS-IS and not STP (as with traditional Ethernet) or the management system (as with PBT) [54]. Because PLSB only uses a small part of VLAN numbers other VLANs can be used for other purposes.

5.2 PLSB

Provider Link State Bridging is defined in [55] and is amendment 9 of IEEE Std 802.1Q^{TM}-2005 [38].

For each B-VLAN assigned to the application PLSB, flooding and learning is disabled and instead a link state protocol, Intermediate System to Intermediate System (IS-IS), is used to learn and distribute network information of PBB and PBT paths.

IS-IS is ideally suited to this task because its flexibility allows it to be easily adapted to support different network protocols [54]. More information on IS-IS can be found at [56]. PLSB uses this flexibility to adapt IS-IS to remove the IP functionality and to share layer 2 information (B-MAC addresses and I-SID values) and is also called Shortest Path Bridging.

5.2.1 Link state information

Each PLSB node informs its immediate neighbors using link state advertisements what nodes it is connected to and how it is connected to them. The advertisements are then send to every PLSB enabled node throughout the network. Every node shares a common view of the network topology (consisting of the B-MACs and I-SIDs), hence the name Provider *Link State* Bridging.

Once all the nodes have learned the topology, PLSB applies the Shortest Path First (SPF) algorithm and installs the calculated paths in the Forwarding Information Base

at each node [54] (the root). Each node has notion of every other node in the network and establishes its own calculated view to other nodes creating a point-to-multipoint SPT as depicted in figure 5.2.

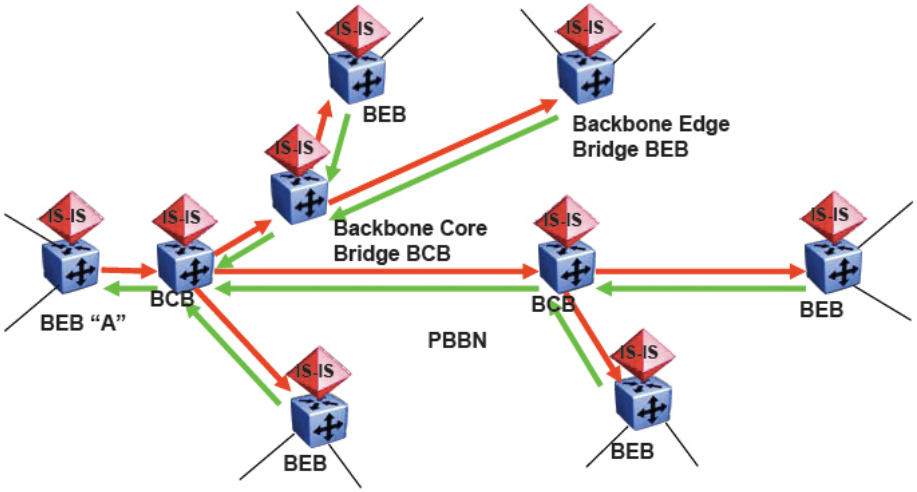

Figure 5.2: Shortest Path Tree for "A". Source [52]

In the picture, the tree for "A" is shown. There is a tree for every Backbone Edge Bridge (BEB). PLSB will create a Shortest Path Tree from every switch to "A" and will simultaneously create a PBT point-to-point path to every switch.

5.2.2 Multicasting in PLSB

As I talked about in the previous section, PLSB uses IS-IS link state updates to distribute information about nodes and services in the network. I-SID information is included in the link stated updates, so all nodes can learn of other nodes that share the same services [54]. Nodes on the path between two end points can detect this and install state for the multicast tree. Only one multicast address per I-SID number is supported.

In figure 5.3 on the following page an example of a service specific multicast tree is depicted. In this case, I-SID 100 is defined only on the nodes ES7 and ES11 in the network.

As new service points were added in this topology, all other service points automatically learned this via the link state updates. When intermediate nodes ES3 and ES4 perform their Shortest Path Tree computation, they will assess whether they are on the best path between any nodes supporting the same I-SID. If yes, they will install the appropriate FIB state in their forwarding tables for forwarding that service's traffic to ensure connectivity [54]. Already a large number of services can be supported.

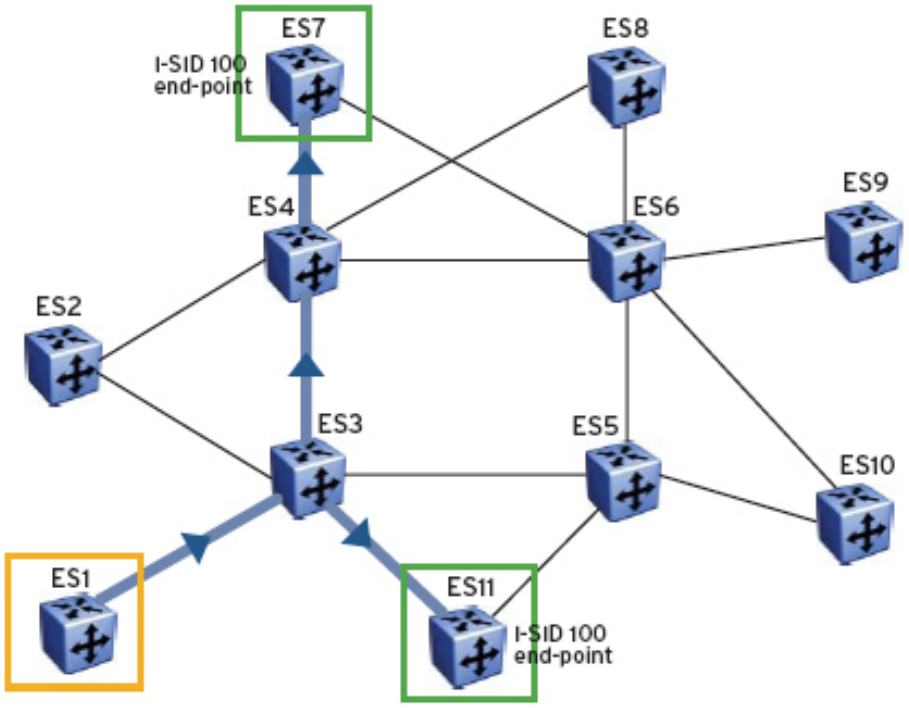

Figure 5.3: Multicast tree for “ES1” for the nodes supporting I-SID 100. Source [54]

6
Findings

In network multicasting, packets are only replicated at routers with known multicast members inside the network. In application layer multicast, on the other hand, packets are replicated at end hosts. This difference is depicted in figure 6.1. In the figure the square nodes are routers and the round nodes are end hosts. The dotted lines represent peers on the overlay.

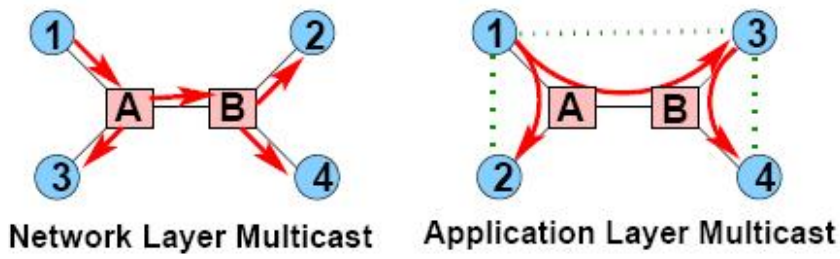

Figure 6.1: Difference between network and application multicast. Source [57]

The end hosts are not necessarily members of the multicast membership but form an overlay network. The goal of application layer multicasting is to construct and maintain an efficient overlay for data transmission [57]. In case of SAGE bridge, this can be done manually. Application layer multicasting can never be as efficient as network

multicasting, especially on the LAN, where the central node will unicast packets to all multicast members.

Besides SAGE bridge, there are several other, application independent, solutions for applications layer multicasting both from the research as the practice arenas. Because of the large application layer multicast (ALM) protocols available, [58] divided these protocols based on their "class" in audio/video streaming, audio/video conferencing, generic multicast service and reliable data broadcast and file transfer. [59] classified available ALM protocols on their class among other things. In the available list, I could find several ALM protocols suitable for the CineGrid testbed in the class of video streaming like CoopNet [60], OMNI [61] and Yoid [62] among others. A complete list can be found in [59].

Application layer multicasting is a viable options for the CineGrid testbed. The streaming of high quality video is mostly done to only a couple of remote locations. By placing end hosts (in this case SAGE bridge) in the vicinity of these remote locations, the stress [1] can be minimized as depicted in figure 6.2.

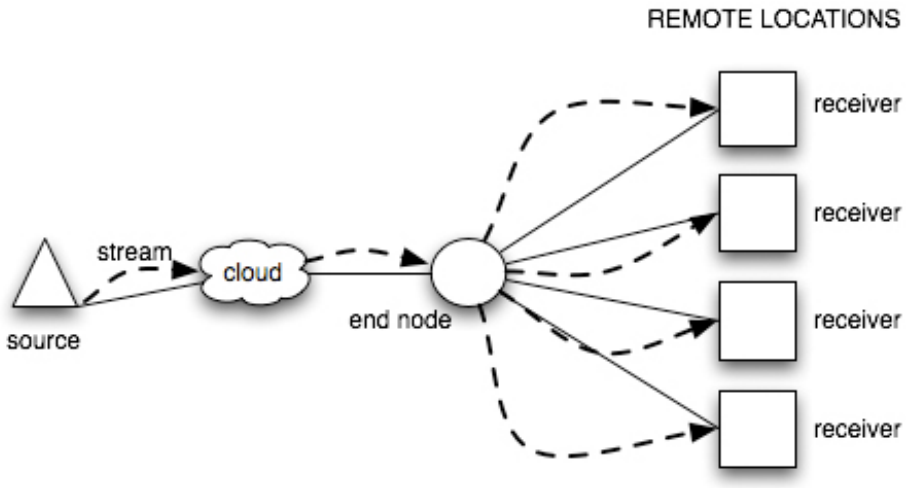

Figure 6.2: Placing end nodes on strategic places in the network

Even when the stress is high, there are in general [2] a couple of ways to solve this problem. A possibility is to send the multicast to one host and let him send it to the next host and so on as depicted in figure 6.3 on the next page. In this way a chain reaction is created and this also introduces high latency problems.

Another possibility would be to subdivide the network. Instead of one machine sending the multicast you send it to multiple *"sub"*network's. From every sub network

[1] Stress is defined per-link and counts the number of identical packets sent by a protocol over each underlying link in the network

[2] Not SAGE or SAGE bridge specific

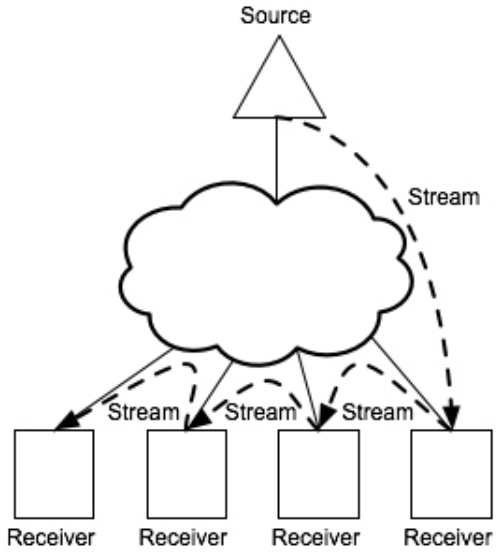

Figure 6.3: Chain reaction to reduce stress

a node sends the stream to the multicast members but only on that sub network as depicted in figure 6.4.

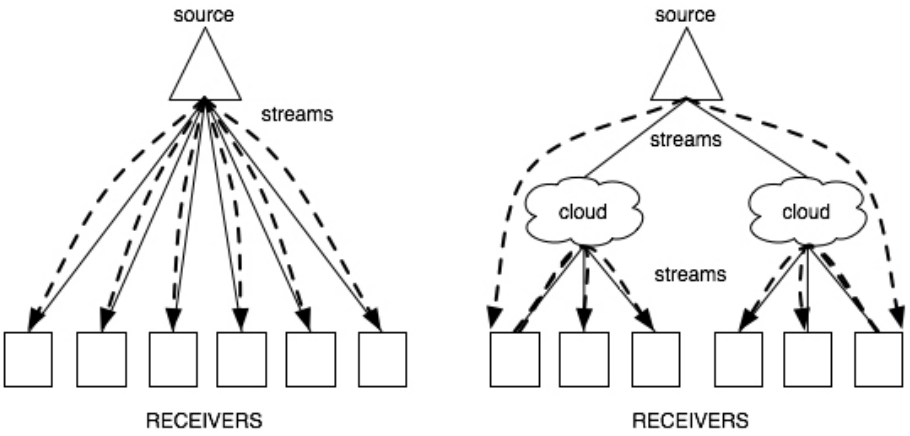

Figure 6.4: Creating hierarchy with sub network's to reduce stress

A last possibility would be broadcasting the stream. Practically, create a VLAN and add all multicast members to the VLAN. Then you let the central multicast node of the VLAN broadcast the stream as depicted in figure 6.5 on the next page. In the figure the source sends a broadcast in the VLAN indicated with the blue full line. Nodes with a dotted line belong to other VLANs.

The limitations to network multicasting on the Internet are largely the cause beause

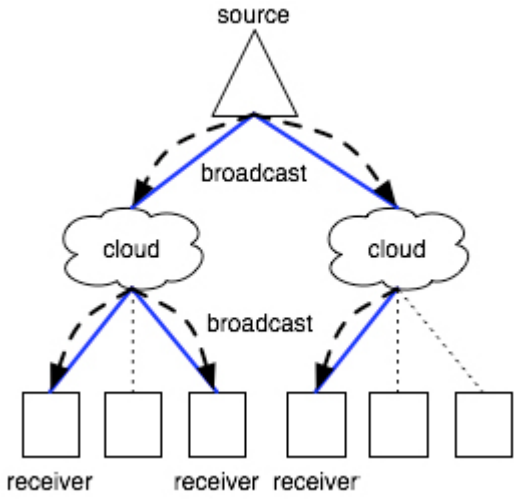

Figure 6.5: Creating VLANs with to reduce stress

it is not widely adopted by Internet Service Providers (ISP's) [28]. The consequence was the creation of ALM. These problems don't apply to the CineGrid network because it isn't connected to the Internet. It uses 10 Gb dedicated optic connections. Other limitations bound the use of network multicast techniques which I will talk about in the following paragraph.

At the moment two possibilities exist to enable multicast in the CineGrid testbed:

- Use PBB in which multicast is enabled;
- Use PBT in combination with PLSB.

The problems to network multicast in a CineGrid testbed are ambiguous. On the one hand, a disadvantage for this research is that PLSB is a cutting-edge technology and availability is limited to only a number of partners. On the other hand, the SNE research group uses PBT for traffic engineering which is not possible in PBB. Also, traditional multicast problems seen on the Internet aren't a problem in the CineGrid. Still, it is important to note that to use multicast over a global Cinegrid network, a global understanding must be agreed on how multicast will be done.

In general, I can say that the differences between network layer and application layer multicast can be summarized as follows. In network layer multicasting each host informs to its designated router when it joins or leaves the group. Then the multicast routers exchange group membership information over the multicast tree. All this overhead is handled by the Internet Group Membership Protocol (IGMP). In this way a perfect topology is created without redundant paths. This improves network efficiency and is very scalable. Despite of bandwidth efficiency, it suffers from deployment problems as I mentioned earlier.

Application layer multicasting on the other hand is less efficient as network layer multicasting but is very easy to deploy. In the ALM protocols, all multicast techniques necessary to send multicast to members, are controlled by the end hosts. In this case, it doesn't require multicast enabled routers which makes it a lot easier to implement. End hosts in application layer multicasting have little or no knowledge about the lower layers in terms of network layout. This causes longer end-to-end latency and lower efficiency [59].

7

Conclusion and future work

Throughout this document, I investigated multicast techniques for the CineGrid testbed. SAGE bridge is a possibility to do multicast on the application layer. I could stream to multiple display nodes. More investigation must be done to solve the problems encountered with the black screen on the second display node. I also gave a short overview of other application layer multicast techniques. I did not investigate these possibilities but these can also prove to be a good alternative. An advantage is that they aren't bound to one application (SAGE).

I investigated PLSB as an application of IS-IS and on top of PBT. PLSB enables dynamic path discovery and shortest path bridging. Through PLSB, network multicasting becomes possible. PLSB is in a beta stage so I don't have any practical experience with PLSB. As soon as it becomes available some performance tests should be done. PBT as an extension of PBB enables traffic engineering and QoS for bandwidth reservation. Using PBB with multicast, in order to reserve bandwidth, minimizes the overhead and extra complexity when using PBT with PLSB. Again, these two possibilities should be investigated. Due to lack of time, I only summarized the use of PLSB and PBT. More technical research can give a deeper insight and can give some extended answers on the possibilities of multicast in PBT and with PLSB.

I noted the advantages and disadvantages of application layer multicast and with a proper implementation the disadvantages can be minimized. Performance tests, comparing network layer and application layer multicasting techniques, should give

a definitive answer.

After carefully investigating the possibilities on multicasting, I conclude that application layer multicasting is a viable option and shouldn't only be looked at as a temporary solution.

A

Installation SAGE

A.1 Pre-installation tasks

We use a Mac mini as extra display node to test multicasting with SAGE bridge. To do this, I need to install SAGE. The other node and the central node are already installed with SAGE. The Mac mini was booted with Mac OS X Leopard. To install SAGE, I first need to install Ubuntu. On the Mac mini no space was left for an Ubuntu partition so I first resized the hard disk with

```
diskutil resizeVolume disk0s2 40G
```

This 40G partition is used to install Ubuntu. To let the the Mac mini dualboot, we use rEFIt [63].

```
cd /efi/refit/
./enable.sh
```

When enabling rEFIt, it automatically detects the two partitions (Mac OS X Leopard and Ubuntu). I can now boot into Ubuntu using rEFIt.

A.2 Dependencies

I installed subversion to get sage from their svn repository

```
sudo apt-get install subversion
```

I downloaded SAGE from the EVL svn repository:

```
svn co svn://cube.evl.uic.edu/dev/sage3.0
```

To build SAGE, the necessary dependencies must be met:

- Quanta [1] can be build using cmake:

```
wget http://www.evl.uic.edu/cavern/quanta/
downloads/QUANTA-1.0.tar.gz
sudo apt-get install cmake
cmake .
make install
#make a softlink so SAGE can locate it
ln -s QUANTAConfig.h QUANTAConfig.hxx
```

- readline [2] and Simple DirectMedia Layer (SDL) [3] can be downloaded with aptitude

```
apt-get install readline libsdl1.2-dev
```

A.3 Installation

SAGE uses a config.mk file when compiling. I will not be using sound so comment the following lines and set the directory for Quanta:

```
#AUDIO=1
QUANTA_DIR=/usr/local/include
```

It isn't necessary to compile everything. Only compile with `make;make install` in the directories sage3.0/src and sage3.0/app.

I use Direct Rendering Infrastructure (DRI) to allow SAGE to access the video hardware directly without passing data through the X server. To install, do:

```
apt-get install driconf
```

Enable S3TC texture compression in System - Preferences - 3D Acceleration

[1] Quanta is a cross-platform adaptive networking toolkit for supporting the diverse networking requirements of latency-sensitive and bandwidth-intensive applications [64]

[2] The GNU Readline library provides a set of functions for use by applications that allow users to edit command lines as they are typed in [65]

[3] Simple DirectMedia Layer is a cross-platform multimedia library designed to provide low level access to audio and 3D hardware among others via OpenGL, and 2D video framebuffer [66]

Acknowledgements

I would like to express a word of gratitude for the opportunity, the University of Amsterdam and the System and Network Engineering (SNE) research group gave me, to conduct my research project.

I want to thank my supervisors dr. Paola Grosso and MSc Ralph Koning for their time and effort guiding me through the project. They were always willing to help me and gave me the possibility to use SNE's infrastructure. Without their aid, I couldn't have completed my research. Also a word of thanks to the numerous people for their advice and guidance.

I gave the SNE research group and the University of Amsterdam a deeper insight in multicasting techniques for the CineGrid testbed.

The SNE group can use this knowledge to envision in future multicast possibilities and to weigh these possibilities to implement a multicast technique for the CineGrid testbed.

All in all, it was a very educational experience and I have learned a lot in a very short time span!

Amsterdam, July 2008
Igor Idziejczak

Bibliography

[1] SNE research group. Research group systems- and networking engineering (sne). http://www.science.uva.nl/research/sne/.

[2] Nortel. Nortel: Solutions : Provider backbone bridges (pbb) and provider backbone transport (pbt): Overview. Website. http://www2.nortel.com/go/solution_content.jsp?segId=0&catId=0&parId=0&prod_id=55120.

[3] Peter Ashwood-Smith & Nigel Bragg Dave Allan. New innovations in ethernet: Provider link state bridging. *Nortel Technical Journal*, 2008.

[4] CineGrid. Cinegrid. Website. http://www.cinegrid.org.

[5] GLIF. Glif: the global lambda integrated facility. http://www.glif.is/.

[6] GLIF. Glif: Publications. http://www.glif.is/publications/maps/.

[7] Ralph Koning, Paola Grosso, Eric Bernier, Cees de Laat, and Inder Monga. Cinegrid on layer2 paths. *Terena Networking Conference 2008*, 2008.

[8] cavern group. Scalable adaptive graphics environment. Website. http://www.evl.uic.edu/cavern/sage/index.php.

[9] Radia Perlman. *Interconnections: Bridges, Routers, Switches, and Internetworking Protocols*, pages 449–450. Addison-Wesley Professional, second edition, 1999.

[10] Lawrence Harte. *Introduction to Data Multicasting, IP Multicast Streaming for Audio and Video Media Distribution*. Althos, 2008.

[11] Abderrahim Benslimane. *Multimedia Multicast on the Internet*. ISTE Publishing Company, 2007.

[12] Wikipedia contributors. Multicast. http://en.wikipedia.org/wiki/Multicast.

[13] Z. Albanna, K. Almeroth, D. Meyer, and M. Schipper. IANA Guidelines for IPv4 Multicast Address Assignments. RFC 3171 (Best Current Practice), August 2001.

[14] R. Hinden and S. Deering. IP Version 6 Addressing Architecture. RFC 4291 (Draft Standard), February 2006.

[15] Daniel Minoli. *IP multicast with applications to IPTV and mobile DVB-H*. John Wiley & Sons, Inc., 2008.

[16] The Institute of Electrical and Inc. Electronics Engineers. Ieee standard for local and metropolitan area networks: Overview and architecture. *IEEE Standards*, 2007. `http://standards.ieee.org/getieee802/download/802-2001.pdf`.

[17] IANA. Ether types. `http://www.iana.org/assignments/ethernet-numbers`, 2008.

[18] S.E. Deering. Host extensions for IP multicasting. RFC 1112 (Standard), August 1989. Updated by RFC 2236.

[19] W. Fenner. Internet Group Management Protocol, Version 2. RFC 2236 (Proposed Standard), November 1997. Obsoleted by RFC 3376.

[20] B. Cain, S. Deering, I. Kouvelas, B. Fenner, and A. Thyagarajan. Internet Group Management Protocol, Version 3. RFC 3376 (Proposed Standard), October 2002. Updated by RFC 4604.

[21] H. Holbrook, B. Cain, and B. Haberman. Using Internet Group Management Protocol Version 3 (IGMPv3) and Multicast Listener Discovery Protocol Version 2 (MLDv2) for Source-Specific Multicast. RFC 4604 (Proposed Standard), August 2006.

[22] M. Christensen, K. Kimball, and F. Solensky. Considerations for Internet Group Management Protocol (IGMP) and Multicast Listener Discovery (MLD) Snooping Switches. RFC 4541 (Informational), May 2006.

[23] A. Adams, J. Nicholas, and W. Siadak. Protocol Independent Multicast - Dense Mode (PIM-DM): Protocol Specification (Revised). RFC 3973 (Experimental), January 2005.

[24] Wikipedia contributors. Dense multicast. `http://en.wikipedia.org/wiki/Dense_multicast`.

[25] B. Fenner, M. Handley, H. Holbrook, and I. Kouvelas. Protocol Independent Multicast - Sparse Mode (PIM-SM): Protocol Specification (Revised). RFC 4601 (Proposed Standard), August 2006. Updated by RFC 5059.

[26] N. Bhaskar, A. Gall, J. Lingard, and S. Venaas. Bootstrap Router (BSR) Mechanism for Protocol Independent Multicast (PIM). RFC 5059 (Proposed Standard), January 2008.

[27] Wikipedia contributors. Sparse multicast. `http://en.wikipedia.org/wiki/Dense_multicast`.

[28] S. Deering and D. Cheriton. Multicast routing in datagram internetworks and extended lans. *ACM Transactions on Computer Systems*, 1990.

[29] Kai Shen. *Application-layer Multicast*. Dept. of Computer Science, University of Rochester, 2003. `http://www.cs.rochester.edu/~kshen/csc573-spring2003/notes/meeting19-multicast.pdf`.

[30] Byungil Jeong, Luc Renambot, Ratko Jagodic, Rajvikram Singh, Julieta Aguilera, Andrew Johnson, and Jason Leigh. High-performance dynamic graphics streaming for scalable adaptive graphics environment. *White Paper*, 2006.

[31] EVL @ UIC. Sage :: Description. `http://www.evl.uic.edu/cavern/sage/description.php#framework`.

[32] Byungil Jeong. *README file for SAGE v2.0*, 2004.

[33] The Tcpdump team. Tcpdump/libpcap public repository. `http://www.tcpdump.org/`.

[34] Byungil(Brent) Jeong. Readme file for sage v2.0. `http://www.evl.uic.edu/cavern/sage/release/readme-v2.txt`.

[35] Nortel Networks. Provider backbone transport: adding scale, qos and operational simplicity to ethernet. *Nortel White Paper*, 2007. `http://www.nortel.com/solutions/collateral/nn115500.pdf`.

[36] Metro Ethernet Forum (MEF). What is carrier ethernet? `http://metroethernetforum.org/page_loader.php?p_id=140`.

[37] G.A. van Malenstein and C. Steenbeek. Pbt networking: Host-to-host connections through surfnet6. *Master thesis*, 2007.

[38] IEEE 802.1 Working Group. Virtual Bridged Local Area Networks. IEEE Std 802.1Q™-2005, 2005.

[39] Wikipedia contributors. Virtual lan. `http://en.wikipedia.org/wiki/VLAN`.

[40] Wikipedia contributors. Ieee 802.1q. `http://en.wikipedia.org/wiki/IEEE_802.1Q`.

[41] IEEE 802.1 Working Group. Virtual Bridged Local Area Networks - Amendment 4: Provider Bridges. IEEE Std 802.1ad™-2005, 2005.

[42] IEEE 802.1 Working Group. Virtual Bridged Local Area Networks Provider Backbone Bridges. *IEEE Standard for Local and metropolitan area networks*, Draft 4.0(IEEE P802.1ah), 2007.

[43] Nortel. Provider backbone bridges bring massive service scalability to ethernet. `http://www.nortel.com/solutions/collateral/nn120620.pdf`.

[44] World Wide Packets. Provider backbone transport of carrier ethernet services. *White Paper*, 2006.

[45] Michael Chen Muneyoshi Suzuki, Paul Bottorff. Addressing issues of provider backbone bridges. *Presentation*, 2004.

[46] Nortel Networks. Nortel metro ethernet routing switch 8600 fundamentals. *Guide*, 2007.

[47] IEEE 802.1 Working Group. Virtual Bridged Local Area Networks Amendment 6: Provider Backbone Bridge Traffic Engineering. *IEEE Standard for Local and metropolitan area networks*, Draft 3.0(IEEE P802.1Qay), 2007.

[48] Ralph Koning and Paola Grosso. Cinegrid on layer2 traffic engineered paths. *Project plan*, 2007.

[49] Universiteit van Amsterdam. Universiteit van amsterdam (uva). http://www.uva.nl/start.cfm.

[50] SARA. Sara reken- en netwerkdiensten. http://www.sara.nl/.

[51] Don Fedyk and Paul Bottorff. Provider link state bridging (plsb). *Nortel Networks Articles*, 2007. http://ieee802.org/1/files/public/docs2007/aq-fedyk-provider-link-state-bridging-0107-01.pdf.

[52] Don Fedyk. Provider link state bridging. http://ieee802.org/1/files/public/docs2007/aq-fedyk-plsb-present-0107.pdf.

[53] Wikipedia contributors. Telecommunications network. http://en.wikipedia.org/wiki/Telecommunications_network.

[54] Nortel Networks. Introduction to provider link state bridging. the next step in the evolution of ethernet. *Nortel White Paper*, 2007. http://www.nortel.com/solutions/collateral/nn123440.pdf.

[55] IEEE 802.1 Working Group. Virtual Bridged Local Area Networks Amendment 9: Shortest Path Bridging. *IEEE Standard for Local and metropolitan area networks*, Draft 0.4(IEEE P802.1aq), 2007.

[56] Wikipedia contributors. Is-is. http://en.wikipedia.org/wiki/IS-IS.

[57] Christopher Kommareddy Suman Banerjee, Bobby Bhattacharjee. Scalable application layer multicast. *SIGCOMM'02*, 2002. http://pages.cs.wisc.edu/~suman/courses/640/papers/banerjee02sigcomm.pdf.

[58] C. Diot, B.N. Levine, B. Lyles, H. Kassem, and D. Balensiefen. Deployment issues for the ip multicast service and architecture. *IEEE Network Magazine*, 14(1):78–88, 2000.

[59] Mojtaba Hosseini, Dewan Tanvir Ahmed, Shervin Shirmohammadi, and Nicolas D. Georganas. A survey of application-layer multicast protocols. *IEEE Communications Surveys and Tutorials*, 2007. http://www.discover.uottawa.ca/publications/CST.pdf.

[60] Helen Wang Phil Chou, Venkat Padmanabhan. Coopnet project at microsoft research. http://research.microsoft.com/~padmanab/projects/CoopNet/.

[61] S. Banerjee, C. Kommareddy, K. Kar, B. Bhattacharjee, and S. Khuller. Construction of an efficient overlay multicast infrastructure for real-time applications. *IEEE INFOCOM*, 2:1521–1531, 2003.

[62] Paul Francis. Your own internet distribution. http://www.icir.org/yoid/.

[63] Christoph Pfisterer. refit - an efi boot menu and toolkit. http://refit.sourceforge.net/.

[64] Eric He Jason Leigh, Oliver Yu. The quality of service adaptive networking toolkit. http://www.evl.uic.edu/cavern/quanta/.

[65] Chet Ramey. The gnu readline library. `http://tiswww.case.edu/php/chet/readline/rltop.html`.

[66] slouken. Simple directmedia layer. `http://www.libsdl.org/`.