CHAPTER 5

LAN SWITCHING

In this chapter we cover the basic principles of LAN switching and bridging that form the basis of the experiments in Lab 5.

The chapter has four sections. Each section covers material that you need to run the lab exercises. The first section gives an overview of the different types of interconnection devices and discusses the differences between them. Section 2 goes into an in depth discussion of LAN switching outlining the fundamentals of backwards learning algorithm and how it is used for routing. In Section 3 we describe the spanning tree algorithm that is used by transparent bridges to maintain connectivity and avoid forming loops. Section 4 presents the commands used to configure the hosts and the routers as bridges.

TABLE OF CONTENT

1 Interconnection devices

Interconnection devices are used to *inter* connect networks at the different layers of the network architecture. The devices can operate at:

- the physical layer such as optical repeaters, hubs, digital cross connects, etc.
- the data link layer such as LAN switches/bridges, frame relay switches, etc.,
- the network layer such as a router or a gateway
- the transport layer for TCP segment switching
- the application layer for overlay networks such as content delivery networks

The various interconnection devices, as shown in Figure 1, play different roles in a network infrastructure and as such have very different functionalities.

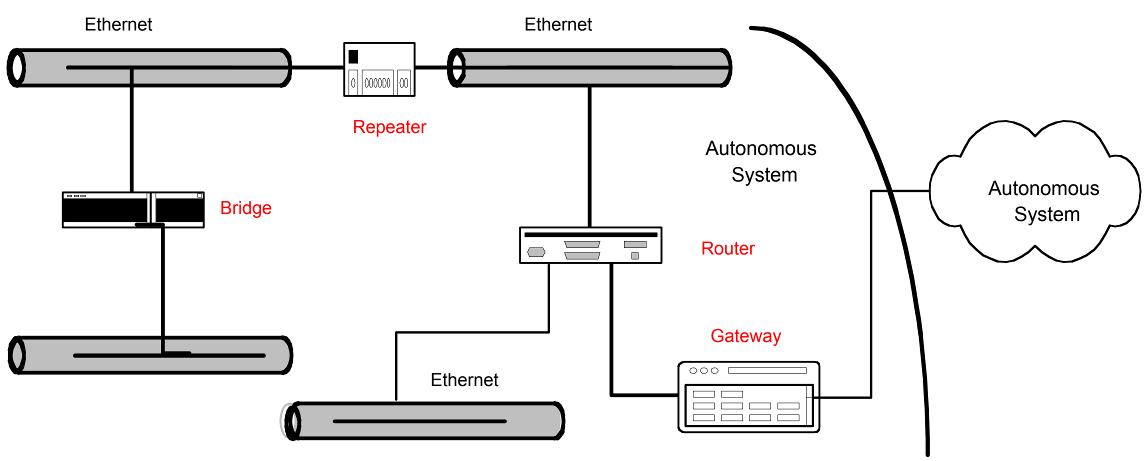


Figure 1. Example of different interconnection devices

1.1 Physical Layer Devices

Physical layer devices are used to increase the reach or geographic span of a physical network. As a signal propagates through a medium such as a coaxial cable, a twisted pair, or a fiber, it suffers from attenuation, i.e., signal strength loss. As shown in Figure 2a, beyond a certain distance, a signal has dropped below a strength threshold that makes it impossible to recover the information. A physical layer device such as a *repeater* is used to amplify the signal before it drops below the threshold (see Figure 1b). The repeater does not *process* the content of the bits in anyway.

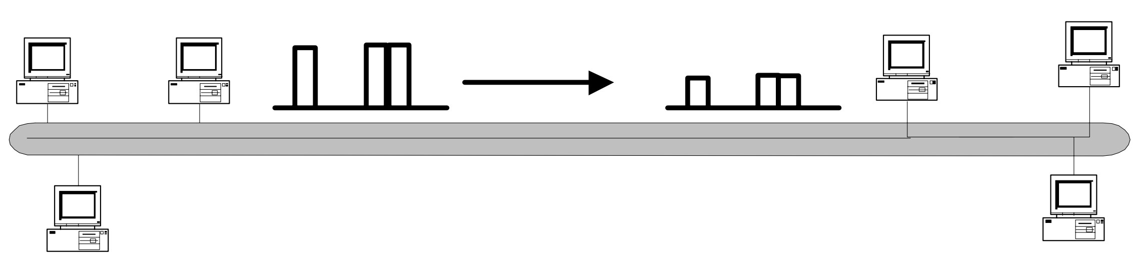


Figure 2a Attenuation and Signal Strength

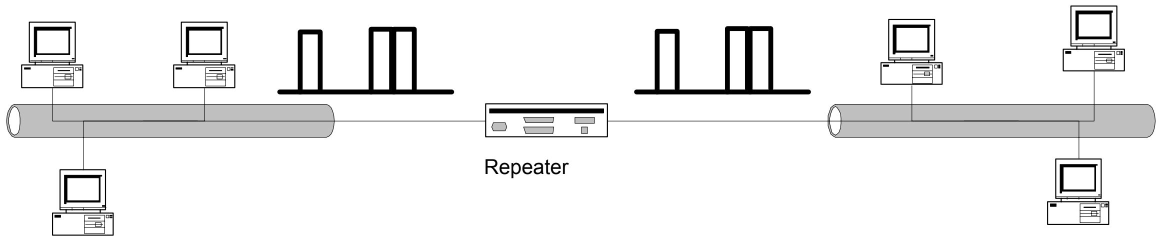


Figure 2b. Amplification of a Signal using a Repeater

An Ethernet *hub* is another example of physical layer device, it serves solely as a conduit for passing packets from its input interfaces to its output interfaces. It broadcasts all information that it receives on its input ports to all of its output ports. It does not store any frames, using cut through switching technology for forwarding the frames, the bits in a frame's header are directly routed to all output ports without waiting for the remainder of the frame to be completely received at the input port. All links that connect devices, such are hosts and routers, to hubs, come in pairs, e.g., 10BaseT, 100BaseT, one pair is used for upstream traffic and the second pair is used for downstream traffic. Any data that is transmitted by a host on the uplink pair is looped back on the downstream pair, re-creating the collision environment of Ethernet. A host therefore hears its own transmission. At the same time if another host transmits a frame on its upstream link, its bits will be broadcast to every port, these will be combined with the loopback transmissions on other downstream links, creating a collision. A host must therefore sense the downlink stream before commencing a transmission. But, as in any CSMA environment, collisions cannot be avoided since two devices could start their transmissions at the same time.

Most hubs nowadays include some buffering on the interfaces to minimize collisions. Dual speed hubs isolate the traffic between the two different speed environments. In other words, a 10/100 hub will not broadcast the 10Mbps traffic on the 100Mbps links unless addressed to a 100Mbps MAC address and vice versa, no 100Mbps traffic is broadcast on the 10Mbp links unless explicitly addressed to a 10Mbps MAC.

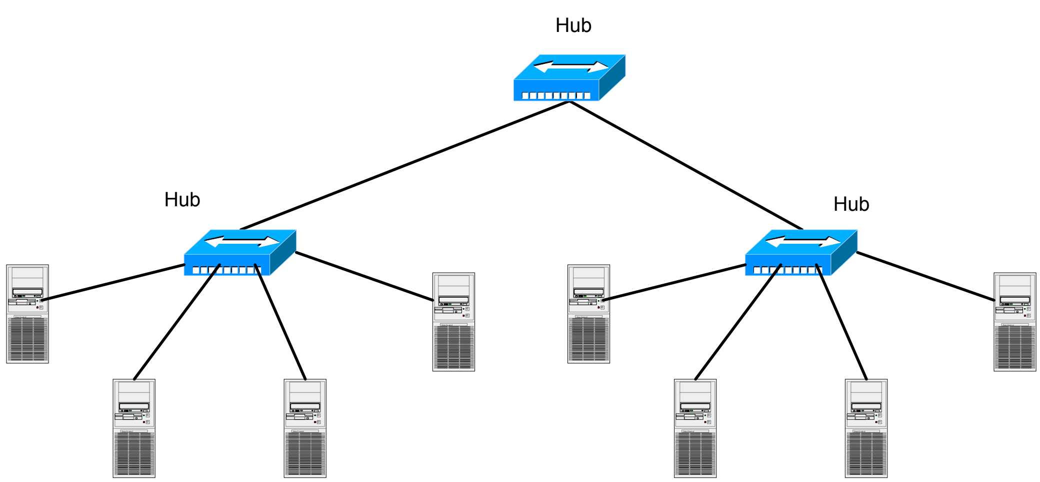


Figure 2. A Hub Architecture

1.2 Data Link Layer Devices

Bridges and *LAN switches* are devices that operate at the data link layer. They interconnect two or more LANs. A bridge was originally designed to provide a *bridge* between two different LAN technologies, e.g., an Ethernet LAN and a token ring LAN. However, over the past decade Ethernet has become the dominant LAN technology and we have seen the gradual demise of token ring LANs. Bridges evolved to not only bridge between two different protocols but to provide another option to hubs and repeaters for extending the size of an Ethernet network domain. Bridges are intelligent devices, that, contrary to hubs, isolate LAN segments thereby limiting the collision environments and improving the overall throughput. By isolating LAN segments, one inherently obtains a more secure network in which data from one segment is not broadcast to another.

A bridge is a store and forward device. Every frame is fully received before forwarding. Transmission on any outgoing link will only take place one frame at a time. Bridges cannot prevent collisions from occurring on an Ethernet segment, but they will not relay collided frames. Similar to hubs, current bridge designs provide dual speed ports, allowing a mix of 10Mbps and 100Mbps devices to be interconnected.

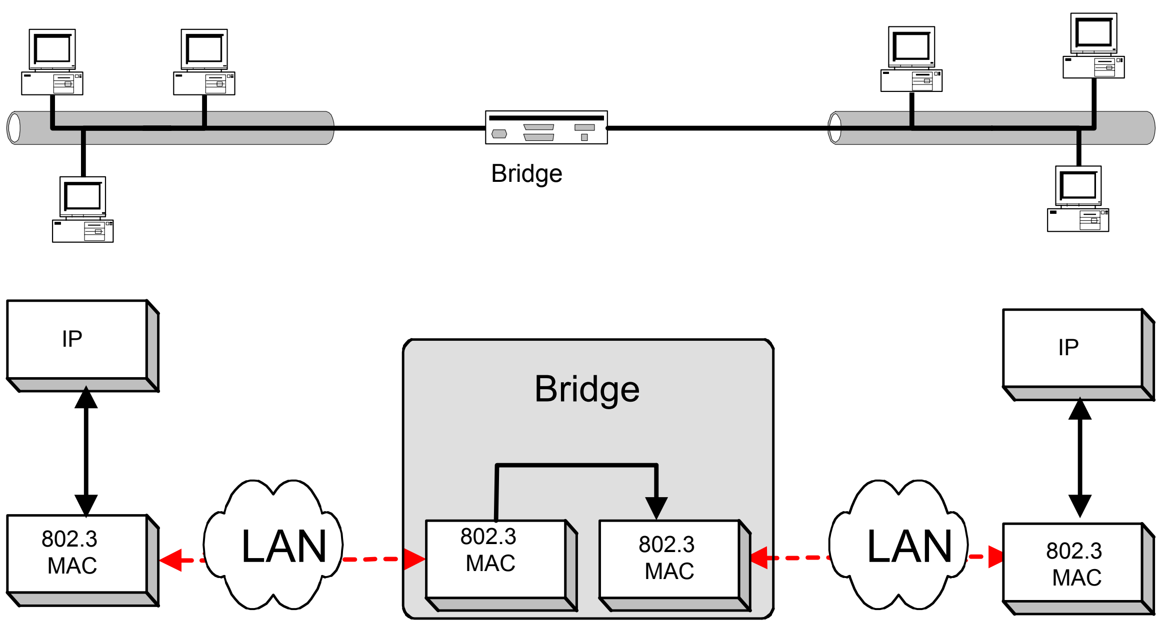


Figure 4a. Bridge Interconnection Device

LAN switches are multi port (more than 4 port) bridges. LAN switches are touted by manufacturers as high throughput multi interface devices that can interconnect ports at a variety of speeds, e.g., 10M, 100M, 1G and 10Gpbs. They are also able to operate the links in full duplex mode if directly connected to a network device[1]. To increase their speed of operation, LAN switches, like hubs, use cut through switching. Once the destination address has been processed the packet is forwarded to the appropriate output port where transmission can be commenced if the link is idle.

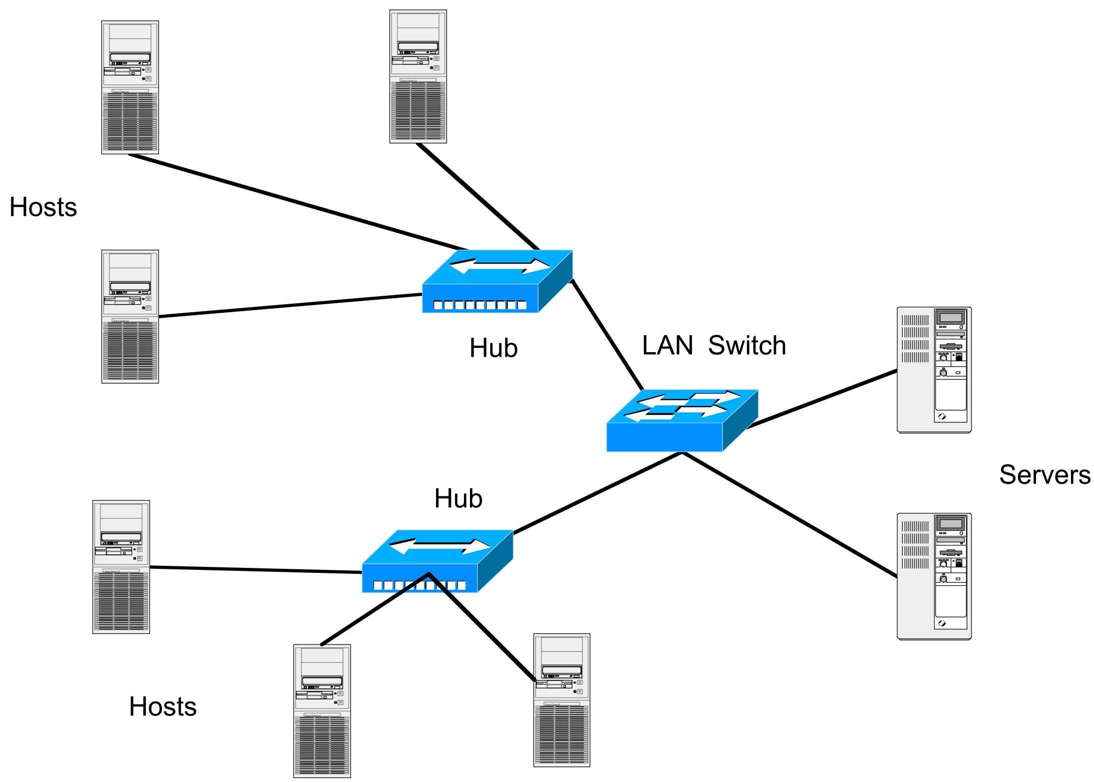


Figure 4b. A LAN switch architecture

1.3 IP Layer Devices

Devices that process IP datagrams are considered to be IP layer devices. *Routers* and *gateways* are both examples of an IP layer device. They each forward and route IP datagrams between different subnets as explained in detail in Chapter 3. The main distinction between a router and a gateway lies in the functionality of the device vis a vis an Autonomous System (AS). A router generally operates within an AS whereas a gateway operates as a bridge between two ASs. A gateway therefore must run two routing protocols, IGP internal to its AS and EGP external to its AS with corresponding gateways in other ASs.

---

[1] The loopback feature is disabled, the link is treated as a point to point link with no collisions possible as frames are buffered if other transmissions are in progress.

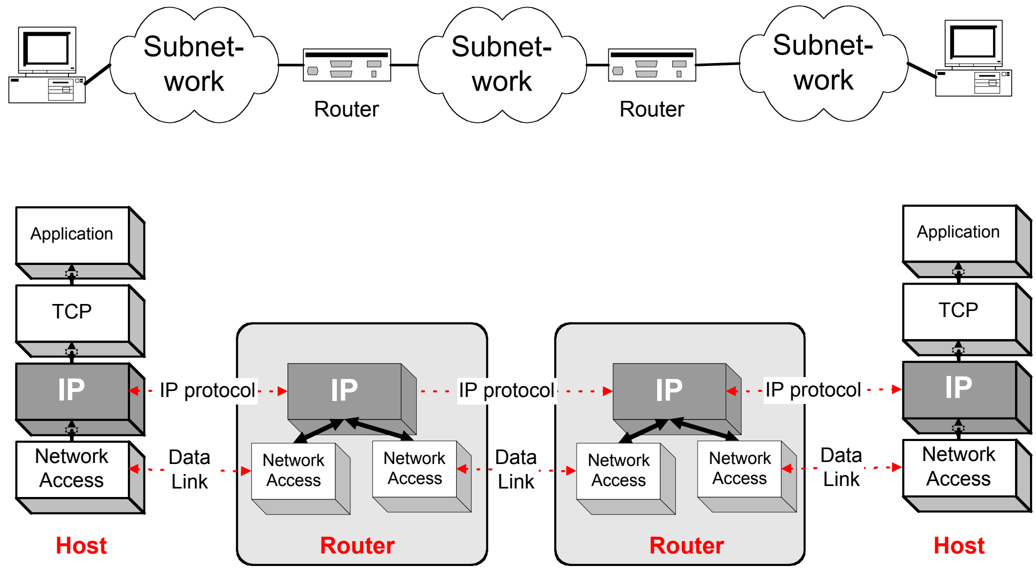


Figure 5a. A Router interconnecting 3 subnets

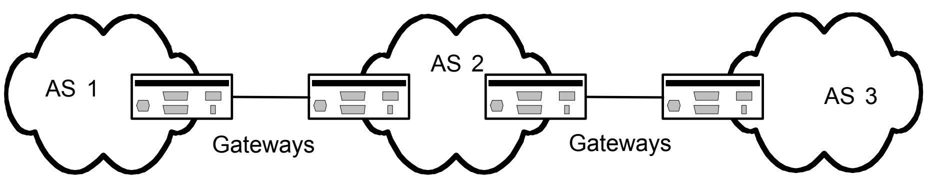


Figure 5b. Operation of a Gateway

1.4 A Comparison between the Different Interconnection Devices

In the Table 1 we summarize the different features of the devices. Each interconnection device has a role to play in an enterprise network. The choice of which device to use depends very much on the needs of the network infrastructure. Hubs are cheap plug and play devices that provide the most basic connectivity. Bridges serve to isolate segments and are relatively cheap too. They have more intelligence built into them and similar to hubs and LAN switches are plug and play devices. LAN switches provide high throughputs at convenient prices when compared to a router. A router serves to create subnetworks which provide autonomy to different departments in a campus network. They are not plug and play, requiring a system administrator to setup subnets and re-configure host network interfaces, but they do give much more flexibility when it comes to network design. Although layer two devices provide dynamic routing, only one route exists between any two devices. In a router configuration, several routes can exist between a source and a sink.

Features/Device	Hubs	Bridges	LAN Switches	Routers
Cost	Low	Low	Low/Medium	Medium/High
Ease of deployment	Plug and Play	Plug and Play	Plug and Play	Requires system admin.
Dynamic	No	Yes	Yes	Yes
Traffic Isolation	None	Yes	Yes	Yes
Autonomy	No	No	No	Yes

Table 1. Comparison of the Features of Interconnection Devices

2 Bridges/LAN Switches

LAN switches[2] are classified by the forwarding procedure that they use to find and reach a destination in a meshed network topology. Three different approaches were identified:

- Fixed paths
- Source routing
- Combination of Backwards Learning and Spanning Tree Algorithms

Fixed routing is never the favored choice because it requires system administrator intervention to create the paths and maintain the network under failures. The source routing approach was very popular in token ring environments. A source would send out a search packet looking for a destination MAC address. Every bridge the frame reached would insert its ID/address and then forward the frame to every outgoing port (except the incoming port). The first frame to reach the destination would then be used to send back a response following the path that the frame took to reach the destination. Every frame henceforth between that source destination pair carried the full path to reach the end point. Alas, token rings have lost favor in the LAN market, giving way to the ever popular Ethernet LAN and thus the demise of source routing in interconnected LAN settings.

The third choice is the dominant approach. Bridges that use backwards learning and spanning tree are referred to as *transparent* bridges due to their truly plug and play nature. We discuss both algorithms in detail explaining their operation and how they forward frames *transparently*, i.e., hosts and routers are oblivious to their presence in the network.

2.1 Transparent Bridges/LAN Switches

The forwarding operation of bridges or LAN switches consists of asking the following question:

- Do I know which port to forward a received frame to?

The answer to that question, as illustrated in Figure 6 below for a frame arriving at **Port x**, is either YES or NO.

- YES -> forward the frame to the appropriate port
- NO -> flood the frame to all outgoing ports except the incoming port

[2] We will use the term bridge and LAN switch interchangeably for the remainder of the discussion in this Chapter.

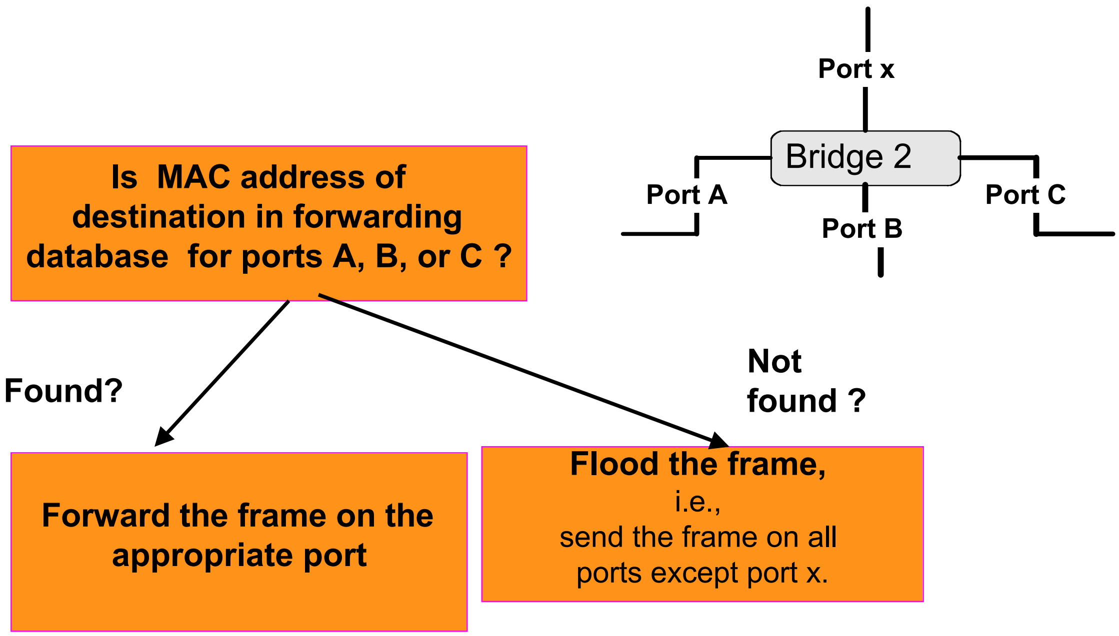


Figure 6. Forwarding process in a Bridge

AYES answer is hinged upon finding the destination address in a table. The forwarding table is created using the backwards learning algorithm, a mechanism by which the bridge learns how to associate destination MAC addresses with outgoing port numbers.

The bridges must reside in a loop free topology because they use flooding to find a destination. In Figure 7 below we show an example of a typical meshed bridged network. Meshed networks are popular for they can tolerate a certain number of link and device failures without creating a disconnected segment. As illustrated in the figure, unless a loop free topology is used, the controlled flooding algorithm will not terminate. Bridges only look at a frame's destination and source address and the incoming port. They do not keep a history of flooded frames[3]. A loop free topology is created by the bridges using the spanning tree algorithm discussed in Section 2.1.2 below. The topology is maintained by constantly monitoring the health of the bridges and the connecting links. Upon detection of a failure, the bridges will self configure to create a new fully connected loop free topology. Some failures may result in disconnected parts that will require system administrator intervention.

---

[3] Note that there are no sequence numbers in Ethernet frames and as such it is impossible for a bridge to maintain a history of previously seen frames unless it maintains a record of the destination addresses of flooded frames. Bridges do not do that, it would complicate their otherwise very elegant and simple design.

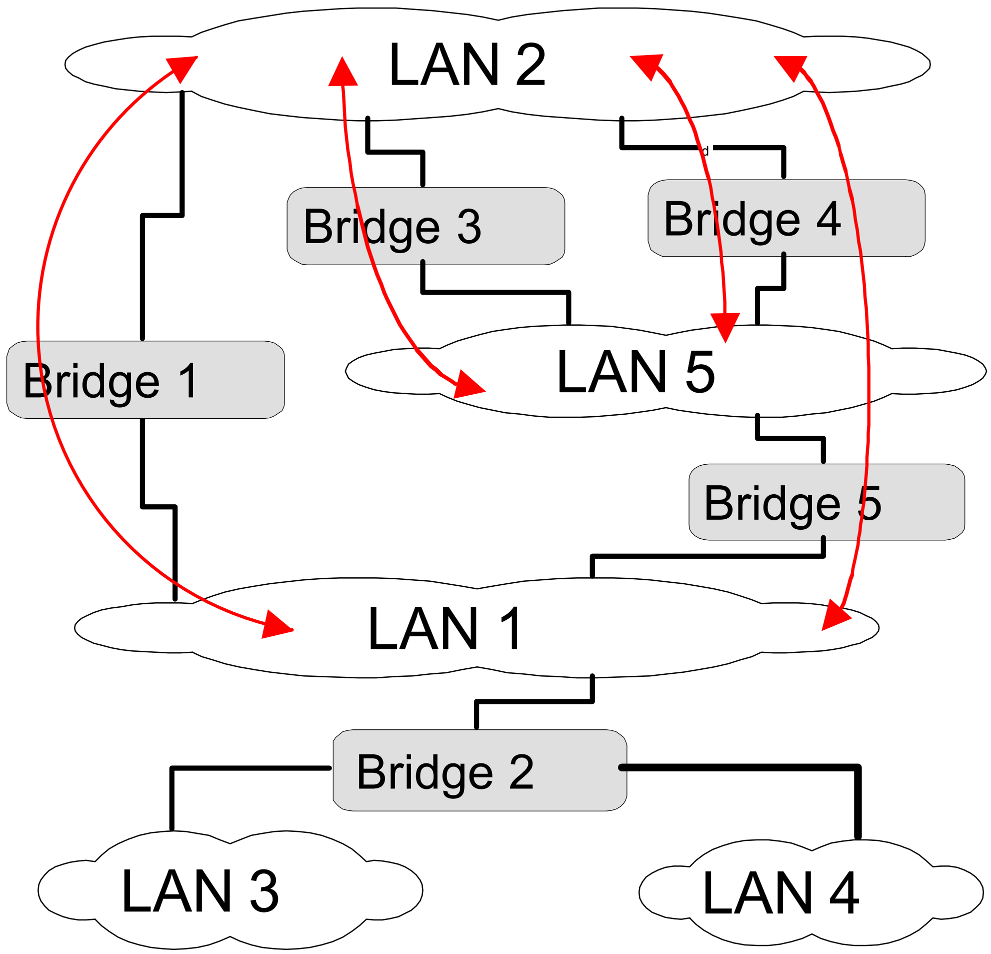


Figure 6. A meshed bridge topology

2.1.1 Backwards Learning Algorithm

The concept of backwards learning is very simple: Learn about an (source) address from the direction from which it came, then place that address in a table and use it for (destination) forwarding. Bridges operate in promiscuous mode, they listen to all traffic that is broadcast on every link connected to its active ports. By examining the source MAC address of every packet traversing the link associated with a particular port on the bridge, the bridge learns what addresses are reachable via that particular port. These addresses are stored in a forwarding database or table.

Every LAN switch maintains a forwarding database or table. This table contains the following fields:

- MAC address
- Outgoing Port number
- Timer – indicating age of entry

The MAC address refers to the destination address in the MAC frame. The outgoing port number refers to the port that needs to be used to transmit the frame for that particular

MAC address. The timer is used to control the age of the entries. When a timer expires, the entry is deleted from the table. Ever MAC address hit refreshes the timer of that MAC address entry. The table can be interpreted as follows:

> A machine with **MAC address** lies in direction of **outgoing port number**. The entry is **timer** time units old.

If a bridge sees a frame with a destination address that matches one of the entries in its forwarding table, it will copy the packet into its buffer and forward the packet to the necessary port. If the outgoing port is the same as the incoming port, it discards the frame. If the bridge sees a frame for which it has no entry in its forwarding table, it will make multiple copies of the frame and broadcast it on every outgoing port (excluding the port on which the frame arrived). As the bridges are connected in a loop free tree topology, the flooding will terminate at the leaves of the tree. Below, in Figure 8, we illustrate the operation of the backwards learning algorithm by stepping through an example of a frame transmission through a single LAN switch with an initially empty forwarding table.

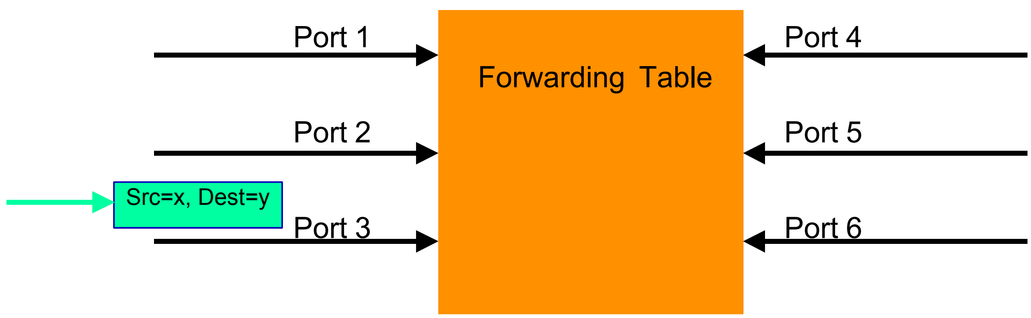


Figure 8a. Frame arrives on Port 3 Source = x, Destination = y

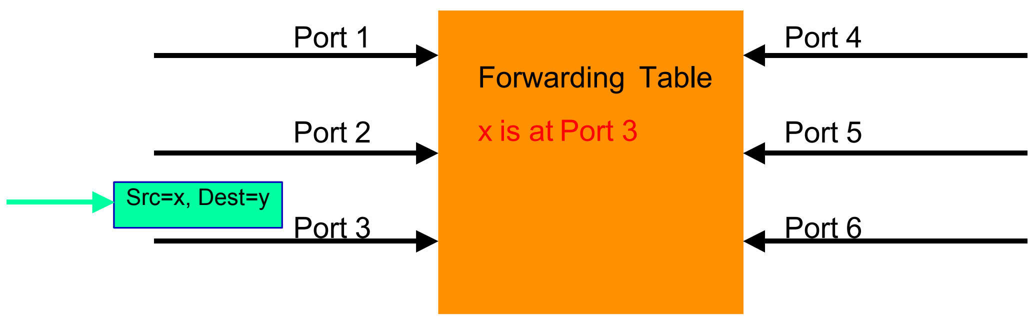


Figure 8b. First entry in table: Bridge learns that MAC address x is associated with port 3

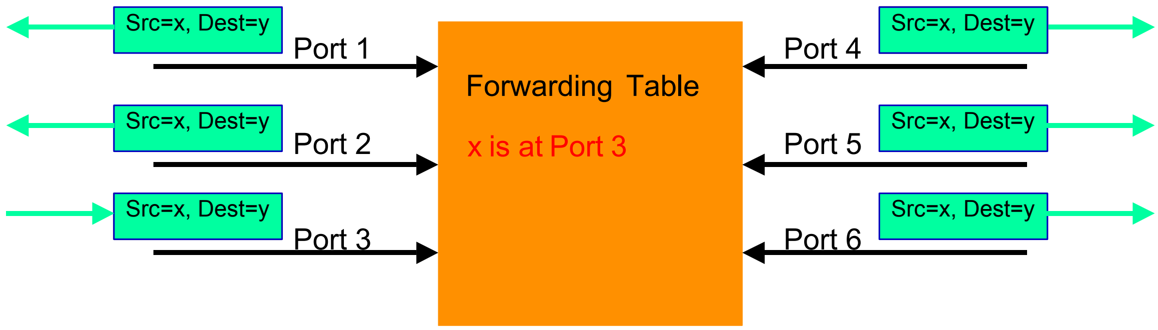


Figure 8c. Bridge does not find entry for y, floods frame on all ports except 3

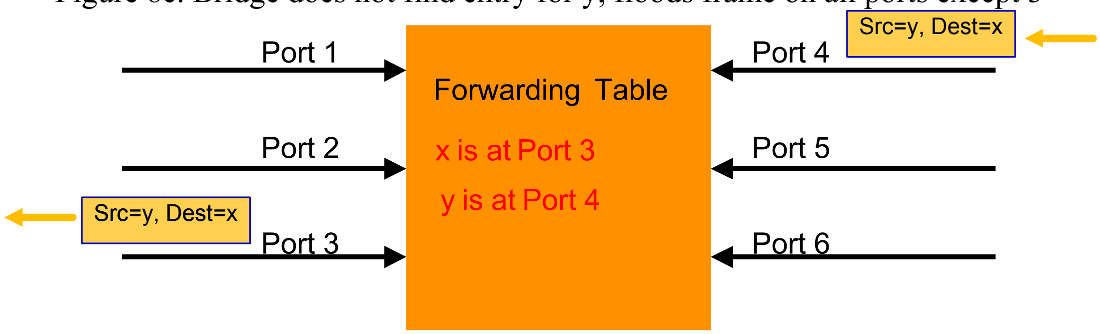


Figure 8d. Frame from y arrives on port 4. Bridge adds entry for MAC address y and forwards frame to port 3 using entry for MAC address x in its table.

If we now take an example of two bridges and observe the process by which a the forwarding table is filled, we will understand the backwards learning algorithm and how it is used by the bridges in promiscuous mode. In Figure 9 we show a sample network with two bridges. Host A initially sends a frame to host F. This is followed by a frame from host C to host A and then a third frame from host E to host C.

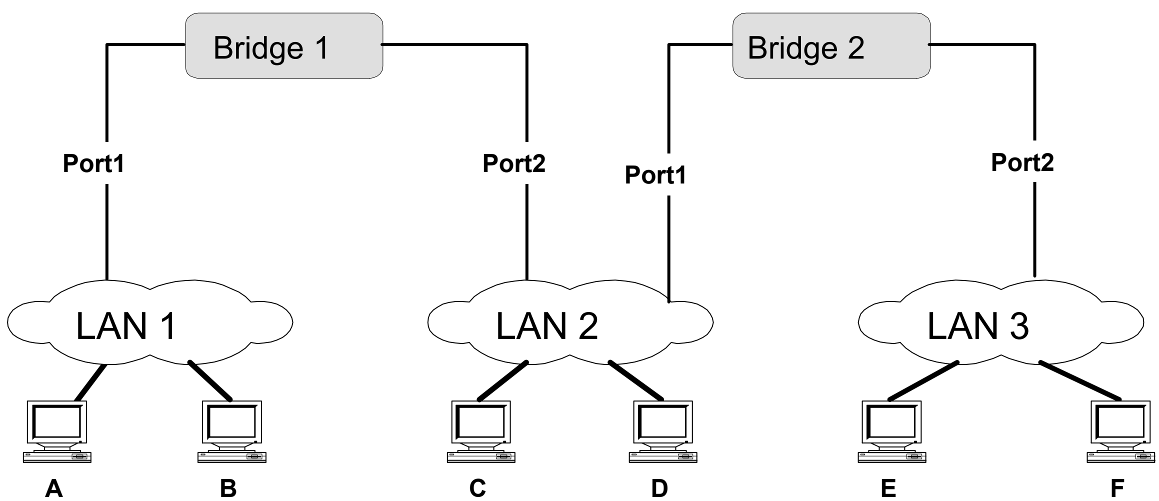


Figure 9. Forwarding Example

Bridge 1 will receive the frame on port 1. With its forwarding table empty, Bridge 1 will flood the frame on outging port 2. Bridge 2 receives the frame on port 1, it too does not find an entry in its table and proceeds to flood the frame on outgoing port 2. Destination

F finally receives the frame. During this flooding process, both Bridges 1 and 2 learnt that MAC address A is associated with port 1 on their respective bridges. The second frame from host C to host A will cause no flooding as both bridges have an entry for MAC address A. Bridge 2 will ignore the frame as its association for MAC address A is with port 1 on which it received the frame. But before discarding the frame it will make an entry in its forwarding table for MAC address C. Bridge 1 will receive the frame on port 2 and forward the frame to port 1 based upon the entry for MAC address A in its forwarding table. It too will make a new entry in the forwarding table for MAC address C. The third frame from host E to host C will not cause flooding either as both bridges have now an entry for MAC address C. Bridge 2 will forward the frame from port 2 to port 1 and at the same time enter MAC address E in the forwarding table. Bridge 1 will ignore the frame as the outgoing port is the same as the received port. It too will make a new entry in the forwarding table for MAC address E. Below we show the resulting forwarding tables (ignoring the timer field).

Bridge 1		Bridge 2	
MAC Address	Port	MAC Address	Port
A	1	A	1
C	2	C	1
E	2	E	2

Table 2. Bridge Forwarding Tables for Example shown in Figure 9

2.1.2 Spanning Tree Algorithm

The spanning tree algorithm [PERL] is the mechanism by which the LAN switches create a loop free tree topology. As explained above, meshed topologies are the preferred design choice in an institutional network to tolerate link and device failures. Flooding mechanisms do not perform well in mesh topologies unless the nodes track the flooded frames and stop flooding when it is recognized that a frame has already been flooded. Bridges do not track frames and so require to operate in a loop free topology. So long as the destination does send a response back to the source, the bridges will never find the destination, even though the frame will have reached the destination. If a loop exists, the frame will be flooded over and over, with each reception of the frame at a bridge generating a new flood. In other words, we will observe an exponential growth in the number of flooded frames.

The idea behind the spanning tree is very simple. Create a tree in which some bridges and/or ports on bridges are active and others are blocked. The blocked bridges and/or ports constitute the disconnected portions of the original meshed topology to create the tree topology.

The spanning tree algorithm uses a specific frame called the Bridge Protocol Data Unit (BPDU) for exchanging information between the bridges. The BPDUs come in various types. Configuration BPDUs are used to create the tree by exchanging path cost information, bridge IDs, etc. Hello BPDUs are used to monitor the health of the tree. If at any point a bridge in the tree does not send a hello BPDU within a specified interval of time, the neighboring bridges (including blocked ones) will sound the alarm by initiating a new round for creating a spanning tree. The health monitoring BPDUs are triggered by the *root bridge*, the bridge at the root of the spanning tree. The root bridge periodically broadcasts a hello BPDU on its branches. The reception of this broadcast hello BPDU by bridges at the next level on the tree, in turn, triggers a transmission of a hello BPDU along their branches. This continues down the tree till it reaches the leaf bridge which transmits a hello BPDU on its local LAN segments primarily for the benefit of any blocked bridges that might be attached at that level signaling its continued health.

The tree creation process consists of the exchange of configuration BPDUs that inform other bridges of the ID of the root bridge, the ID of the bridge transmitting the BPDU, the cost for that bridge to reach the root and the port used for forwarding in the direction of the root. This port is referred to as the *root* port. All other ports unless *blocked* are called *designated or forwarding* ports. The root ports and the designated ports constitute the active links of the topology. A bridge with no root or designated ports is considered to be a blocked bridge. Note that a blocked bridge does not participate in frame forwarding, but it does listen to all transmissions, monitoring all activity and ensuring the health of other non blocked bridges. Not all ports on a bridge need to be either root or designated, some can be blocked. Each LAN must have one and only *designated bridge*, which is the only bridge with the a designated port on that LAN. Note that several bridges can have their root ports on a LAN, but only one can have a designated port.

2.1.2.1 Configuration BPDUs

These are the BPDUs used by the bridges to exchange information that will assist in the determination of the spanning tree. Figure 10 shows the fields of a configuration BPDU. The fields in red are the main fields used for creating the spanning tree.

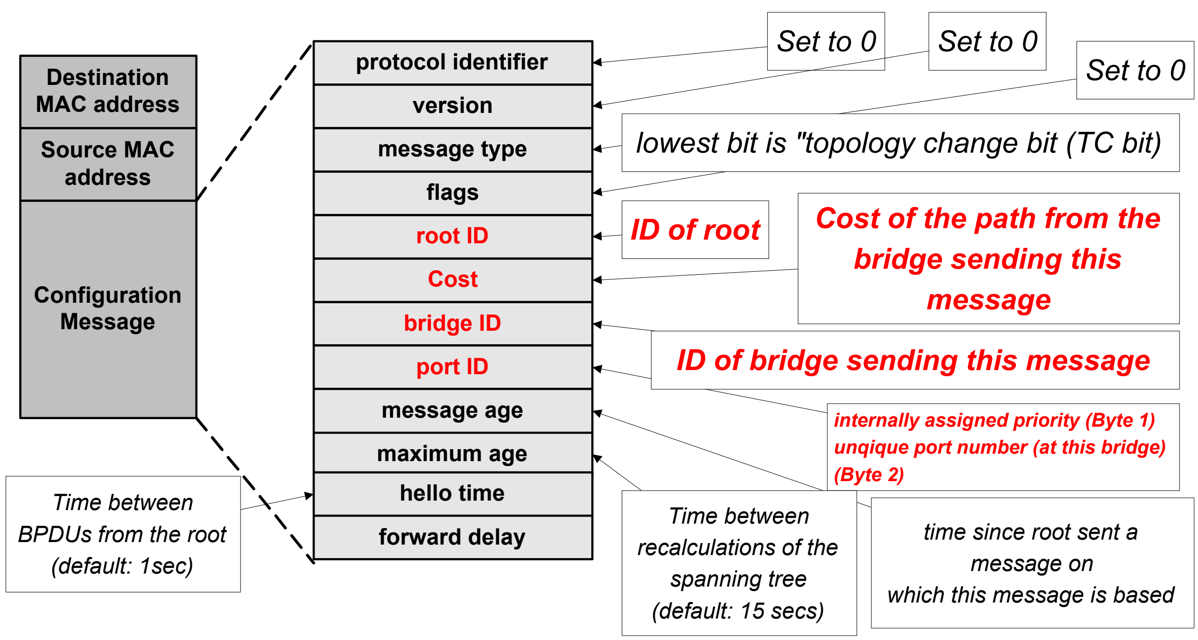


Figure 10. Configuration BPDUs

Each bridge as a unique identifier: Bridge ID = <MAC address + priority level>. A bridge has several MAC addresses, one for each port, but only one ID, used to elect the root. The lowest MAC address is usually used for the ID. Each port within a bridge has a unique identifier (port ID).

The bridge with the lowest identifier is elected the root of the spanning tree and is henceforth referred to as the root bridge (root ID in configuration BPDU). The root port on each bridge identifies the next hop from that bridge to the root and is identified by the port ID in the BPDU. For each bridge, the cost of the min-cost path to the root is called the root path cost. This cost is generally measured in hops (each LAN is a hop) to reach the root, but it can be set to represent any other metric. For example a 100Mpbs LAN is more desirable as a path than a 10Mbps LAN, and so the cost can be determined accordingly, the 100Mbps LAN will have a lower value. The designated bridge on a LAN provides the minimal cost path to the root for that LAN. If two bridges have the same root path cost, the algorithm selects the one with the highest priority. If the designated bridge has two or more ports on the LAN, then the algorithm selects the port with the lowest identifier.

With the help of the BPDUs, bridges can:

- Elect a single bridge as the **root bridge**.
- Calculate the distance of the shortest path to the root bridge
- Each LAN can determine a **designated bridge**, which is the bridge closest to the root
- The designated bridge will forward packets towards the root bridge.
- Each bridge can determine a **root port**, the port that gives the best path to the root.
- Select ports to be included in the spanning tree.

Below we describe the steps used to elect the root, determine the designated bridges and calculate the minimum cost to the root.

2.1.2.2 Steps of the Spanning Tree Algorithm

Steps of the spanning tree algorithm:

- Determine the root bridge
- Determine the root port on all other bridges
- Determine the designated port on each LAN

To achieve the above, each bridge is sending out BPDUs that contain the root ID (what the bridge considers to be the root, initially always set to the bridge's ID), root path cost (current path cost to what is considered by the bridge to be the root, initially set to "0" as it assumes it is the root), bridge ID (own ID), port ID (port to be used to reach root).

Figure 11 Main fields of configuration BPDU to calculate the Spanning Tree

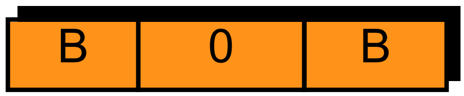


Figure 12 Initial settings of the fields in a configuration BPDU

Initially, all bridges assume they are the root bridge. Each bridge B floods all its ports with a configuration BPDU as shown in Figure 12 on its connected LANs. It identifies itself, sets the root ID to itself and the cost is “0”. Each bridge receives configuration BPDUs from its neighbors and compares the values in the these three fields with those in its own transmitted BPDU. It the updates its BPDU accordingly, the root bridge is the smallest received root ID that has been received so far (whenever a smaller ID arrives, the root is updated) and it increments the root path cost by the cost of the link connecting the bridge to the neighbor from which it received the BPDU with the lowest root ID. Bridge B’s root port is the port from which B received the lowest cost path to the root R. With the new values of R, cost and root port, it can update its BPDU and flood that information to all neighbors but the one it received the lowest root ID from. All ports from which it received a BPDU with a higher root ID it will designate as its designated ports and assume itself to be the designated bridge for those LANs (unless two ports are connected to the same LAN[4], in that case, it will pick the one with the lowest ID) and block the other.

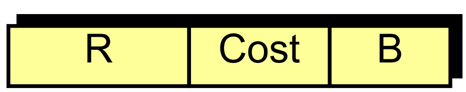
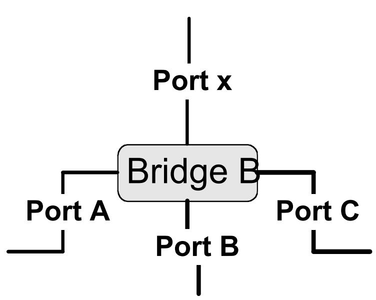


Figure 13 Updated fields of BPDU

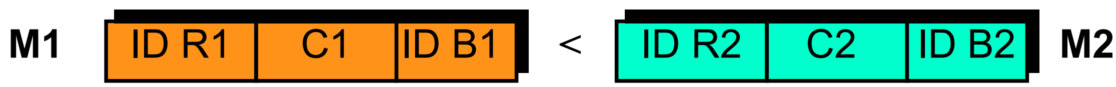


[4] A bridge will only know that if it receives to identical BPDUs.

Figure 14. Bridge B’s connected ports

For example, in Figure 14, if the BPDU with the lowest root ID was received from port x, then bridge B will assume it is the designated bridge on the LANs attached to ports A,B, and C.

To summarize our discussion on root and port selection, we can order BPDU messages with the following ordering relation “<<“:




```
If (R1 < R2)
          M1<< M2
elseif ((R1 == R2) and (C1 < C2))
          M1 << M2
 elseif ((R1 == R2) and (C1 == C2) and (B1 < B2))
          M1 << M2
```

And always pick the smallest message in the “<<” sense.

In Figure 15 we show the earlier example of Figure 6 with a spanning tree over-layed on top of the mesh topology.

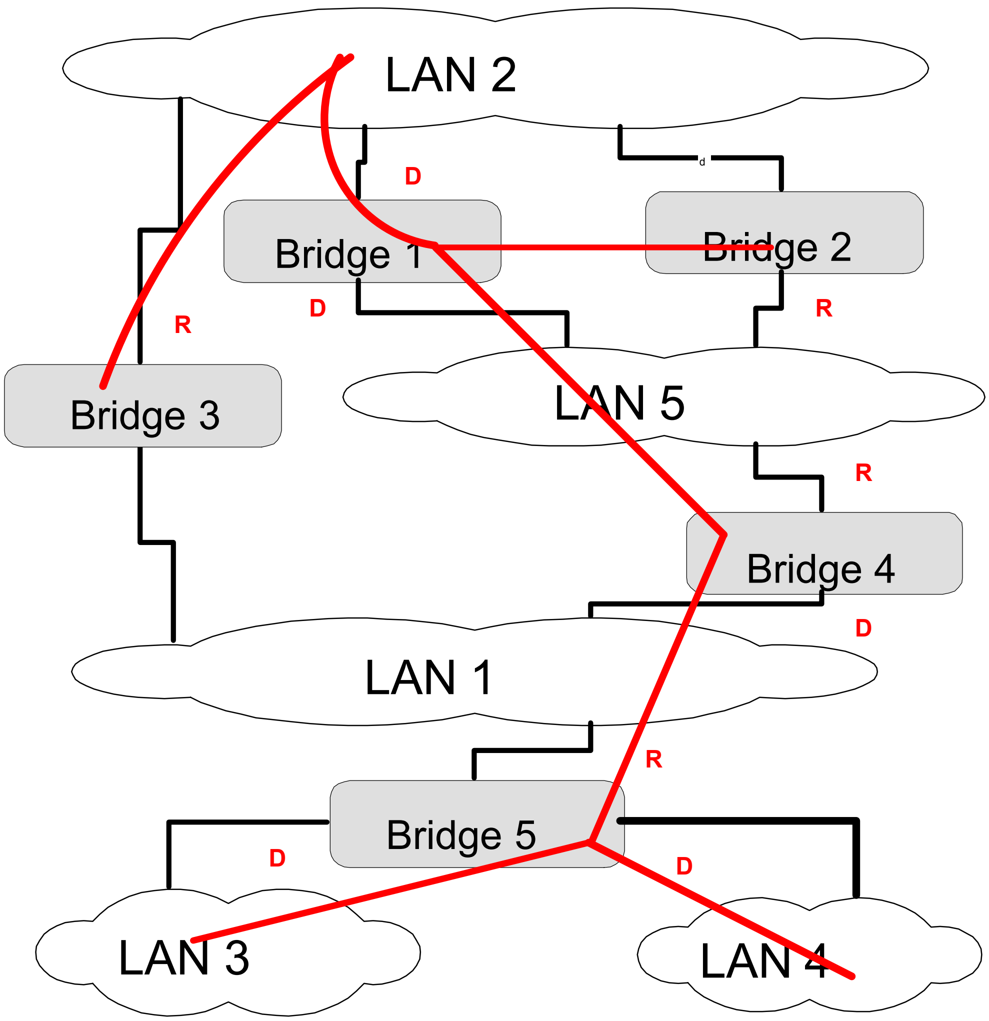


Figure 15 Mesh topology with a tree overlay. “D” identifies designated port, and “R” root port

3 Tools and Utilities

3.1 Configuring a PC as a Bridge using the gbrctl utility

`gbrctl` is a GNOME utility to configure Ethernet bridging on Linux PCs. The following screenshots illustrated the features and configuration procedure of `gbrctl`.

- To start the utility, type “*`gbrctl`*” at a shell terminal and the main window appears:

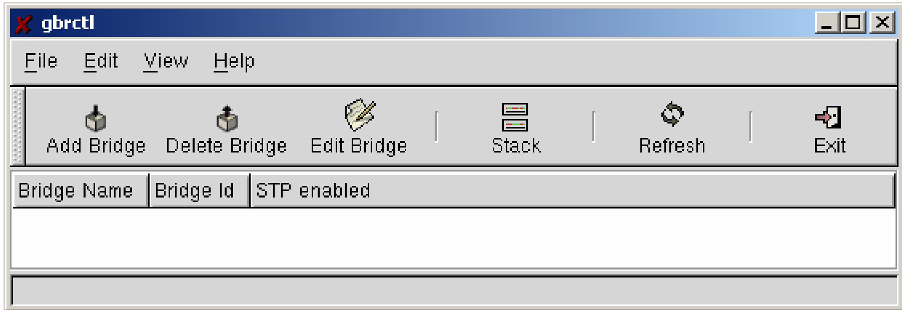


Figure 1: Main Window

- To begin the configuration of Ethernet bridging, click on *Add Bridge, a*nd enter the bridge name, such as "*Bridge1*"in the prompt that appears:

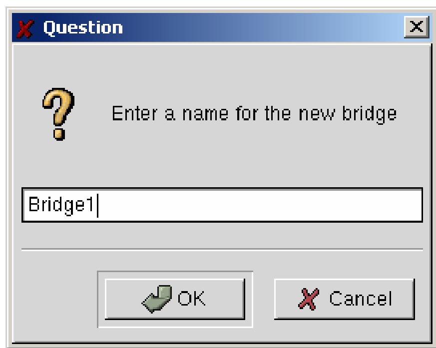


Figure 2: Prompt to Add New Bridge

- To configure `gbrtcl` so that Ethernet interfaces of the Linux PC participate as interfaces of the LAN switch, select "*Bridge1*", and click on *Edit Bridge*. This will bring up the Bridge Configuration Window:

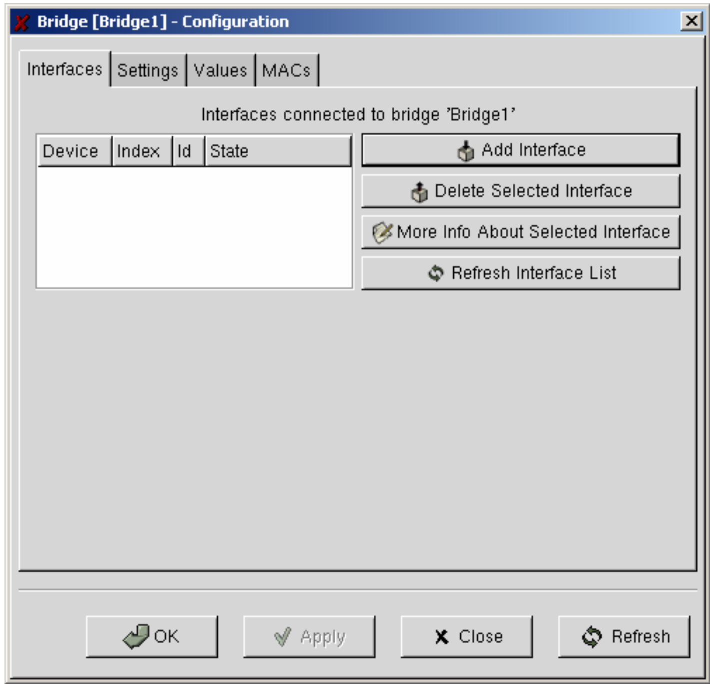


Figure 3: Bridge Configuration Window – Interfaces

- Click on the *Interfaces* tab, and then the *"Add interfaces"* button. Type the name of the interface to be added, e.g., "`eth0`" or "`eth1`".

- Under the *Settings* tab, one may enable or disable the Spanning Tree Protocol (STP) by toggling the button next to the STP parameter:

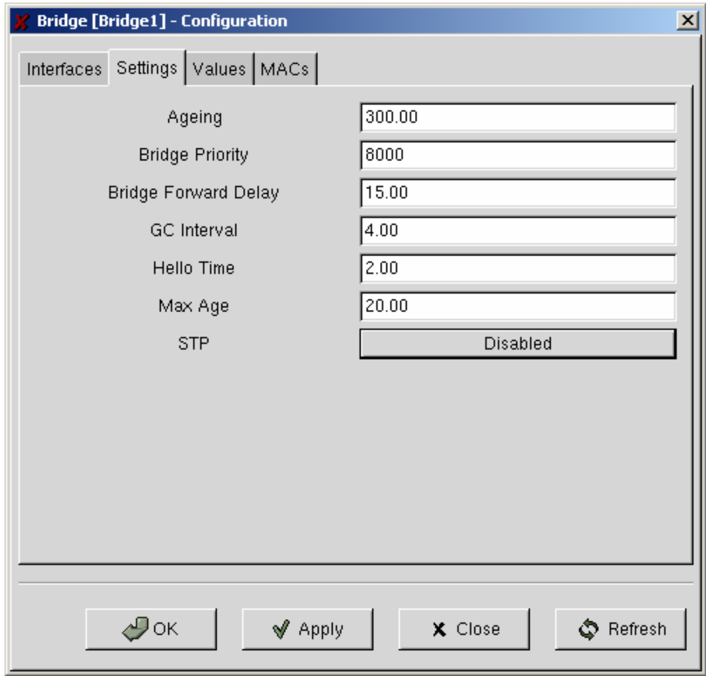


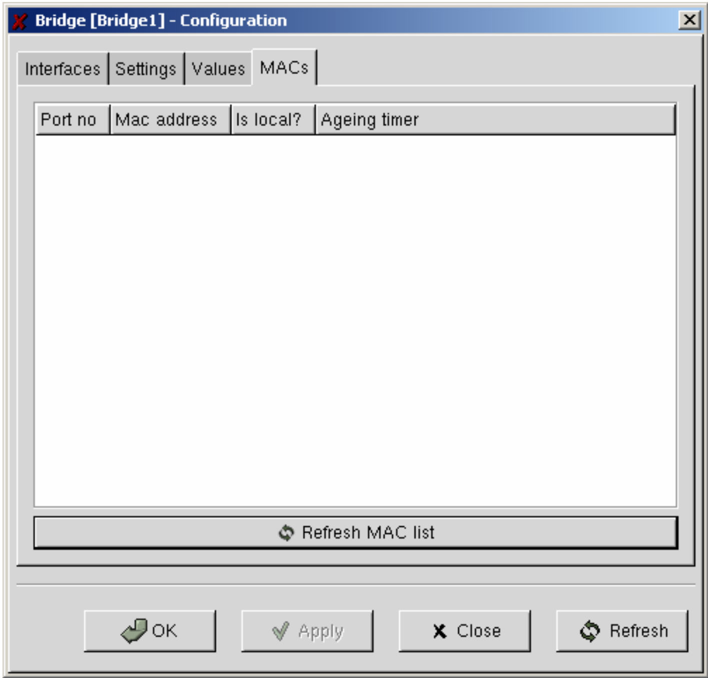
Figure 4: Bridge Configuration Window – Settings

- An important field under this tab is the *Bridge Priority*. Probably due to a implementation error, `gbrctl` has an unconventional way to entering and displaying the value of this field. By default, `gbrctl` sets the *Bridge Priority* to 8000 (hex), and sets it to 0001 (hex) when SPT is disabled and then enabled. To assign the bridge priority, enter the appropriate input value given in Table 1 below.

Input Value	Value Displayed in GUI	Actual Priority Value
8	0000	0
16	0008	8
32	0010	16
64	0020	32
128	0000	0
256	0040	64
512	0080	128
1024	0100	256
2048	0000	0

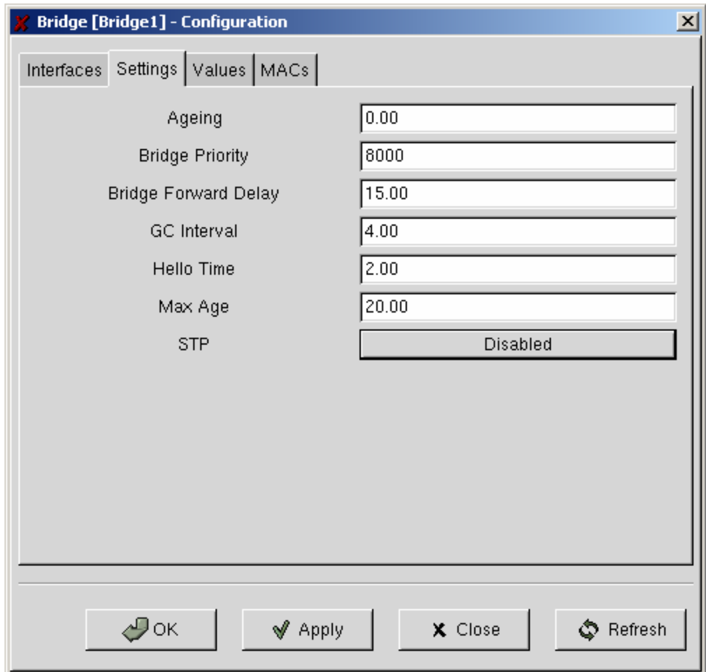
Table 1: Valid Bridge Priority Values for `gbrctl`

- To display the content of the forwarding table, click on the *MACs* tab and click the *Refresh MAC list* button at the bottom:



- The `gbrctl` tool does not have an explicit way to delete the forwarding table. In order to delete the entries from the forwarding table, one may set the age of the forwarding table entries to a small value as follows:

- Select "Bridge1" and click on *Settings*
- Set "Ageing" to 0
- Once the entries are deleted, set the Ageing entry to the original value (default is 300 seconds.)



3.2 Configuring a Cisco Router as a LAN Switch

A Cisco router can be operated as a LAN switch by turning off the routing functions with the no ip routing command and enabling the bridging function. Table 1 below describes the commands that need to be used. When enabling bridging, you also have to choose the routing protocol, in our case we will be using the IEEE standard which refers to the Spanning Tree algorithm. By choosing a priority, you can determine which bridge gets to be the root. If all the bridges have the same priority, then the MAC address will be used for root selection.

Router> **enable** Password: **rootroot**	Enter the privileged EXEC mode. In this state you can read configuration files, reboot, etc.
Router1# **configure terminal** Router1(config)#	Enter a configuration mode. In the configuration mode, you can do various system-related tasks, for example, assigning IP address, setting the protocol to support, etc.

Router1(config)# **no ip routing**	Disable IP routing. This tells the Cisco Router to stop acting as a router.
Router1(config)# **bridge 1 protocol ieee**	Assigns the IEEE Spanning Tree Protocol (STP) to "bridge group 1". A bridge-group is a number between 1 and 9, which is chosen to refer to a set of bridge interfaces.
Router1(config)# **bridge 1 priority 128**	Define a priority for bridge group #1. Priority is used when electing the root bridge in the Spanning Tree Algorithm.

Table 1 Enabling Bridging on a Cisco Router

Note: the bridge group number (e.g. *bridge 1*) is only relevant internal to each router. If a router has at least 18 interfaces, you can create up to 9 bridge groups (9 pairs of interfaces). Since the Cisco routers in this lab only have 2 interfaces, we will assign *bridge group 1* to all the routers.

The above steps set up Router1 as a LAN switch. Now, each interface of the router has to be individually configured to participate in LAN switching. In Table 2 we show the steps how to configure the interface Ethernet0 on Router1:

Router1(config)# **interface eth0/0**	Enter the interface configuration mode for interface Ethernet0/0. This is used when configuring parameters specific to that interface.
Router1(config-if)# **no mop enabled** Router1(config-if)# **no mop sysid**	These commands disable the Maintenance Operation Protocol (MOP) in DEC networks. By default, it is disabled on each interface. These commands are applicable only on Cisco 25xx routers, and are not available on Cisco 16xx or 36xx routers.
Router1(config)# **no cdp run** Router1(config-if)# **no cdp enable**	Disable the Cisco Discovery Protocol (CDP). By default, it is enabled on each interface. When CDP is disabled, another type of device/network management protocol appears and transmits "loopback" packets, which have identical source and destination MAC addresses, and do not interfere with the experiments.
Router1(config-if)# **bridge-group 1**	Assigns this network interface to bridge-group 1. Frames are forwarded only between interfaces in the same group within a bridge. In this lab, all interfaces should belong to the same group.
Router1(config-if)# **bridge-group 1**	Assigns this network interface to bridge-group 1, but

spanning-disabled	disables the Spanning Tree Algorithm (SPT).
Router1(config-if)# **no shutdown**	Activates the interface.
Router1(config-if)# **end**	Returns to privileged EXEC mode.

Table 2. Setting the Router Interface

Once a Cisco router is configured as a LAN switch, the following commands are used to display the current status of the LAN switch:

Router1# **show bridge**	Displays the entries of the forwarding table.
Router1# **show spanning-tree**	Displays the spanning-tree topology information known to this bridge.
Router1# **show interface**	Displays statistics of all interfaces, including the MAC addresses of all interfaces.

Table 3. Displaying the Bridge Status

The following steps are used to reset the state of a bridge at the beginning of a new exercise:

Router1# **clear bridge**	Removes all entries from the forwarding table.
Router1# **clear arp-cache**	Clears the ARP table.

Table 4. Resetting a Bridge

References:

http://home.planet.nl/~kristian/gbrctl.html