Lab 06: Bridging and ethernet spanning tree protocol in Linux

OSL, Mon Sept 24, 2012

Objective:

1. Get familiarized with VNUML, a virtualization tool based on User Mode Linux used to simulate linux based network scenarios
2. Learn how *ethernet bridging* and *spanning tree protocol* are setup in Linux.

General instructions:

1. **USE LOCAL LOGIN. NFS CANNOT TAKE THE LOAD OF 40 MACHINES RUNNING VIRTUALIZATION AT THE SAME TIME.**
2. This lab is to be done in **groups of two students**
3. Download the configuration file "lab06_bridging.xml"
4. Create a directory called <rollnumber1>_<rollnumber2>_lab06. As you proceed with the lab instructions below, note down observations or relevant output from whatever you do in a file named "lab06.txt" using a text editor.
5. Also add to this directory any written code along with output files. You will find more details of this in the specific exercises.

Lab Instructions:

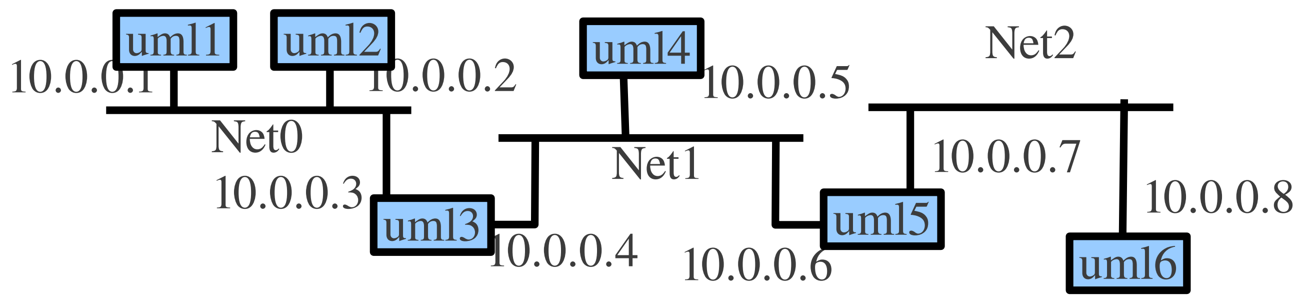
Setting up bridging (or for that matter doing any network configuration) on normal linux hosts requires *root* permission. To circumvent this problem, we will be using *User Mode Linux (UML)* which essentially runs Linux as an application within Linux. How cool is that! So you will be working on real Linux, it does not matter to us that the real Linux is running as a virtual machine.

Another advantage of using UML is that you can create any number of virtual machines on a single physical machine (subject to memory availability). So you can create an entire *network of Linux machines* in a single physical machine!

We will be using VNUML, a software which makes the creation of a Virtual Network using UML very easy. VNUML takes a configuration file as input, which specifies the topology and connections in the virtual network. It then creates as many UML instances as are specified in the configuration, and connects them up according to the specified topology. That's it!

Exercise 1: Bridging

We will create a simple topology as shown in the figure. As you can see, there are 3 LAN segments, Net0, Net1 and Net2. There are 3 hosts uml1, uml2, uml3 connected to Net0. Uml3, uml4, uml5 connected to Net1 and uml5, uml6 connected to Net2. Note that some of the machines have two ethernet interfaces and connected to two networks. Each interface of a machine is assigned an IP address as shown. Look at the "xml" file to see how this topology was created.



You can bring up the above virtual network using the command: " vnumlparser.pl -t lab06_bridging.xml -v". ***In case you want to release the simulator (for a good reason, setting up the machines takes some time), you can do so using the command "vnumlparser.pl -d lab06_bridging.xml -v"***

When you run the above command (it may take some time to execute), you should see six xterm windows popping up; these are the six virtual machines running Linux corresponding to the umls. You can **root** login to each by using the password "xxxx". What you do on these machines will have no effect whatsoever on the physical machine's Linux or the filesystem! So the system administrators are happy!

1. Checkout the xterms corresponding to the umls. They are much like your regular terminals. You can use ifconfig, route, arp, vim, cat, bash scripts etc much like on our own laptops/desktops where you have root permissions.
2. Check the reachability between the machines by using "ping <dest-ip>" For example, you can try "ping 10.0.0.2" from uml1. Note down in the report the machines you could reach from each of the machines: uml1, uml2, uml3, uml4, uml5, uml6. Could you reach all the machines in the topology from any given machine? Why or why not? For debugging, run tcp dump on the umls to see whats happening.

Your next job is to enable bridging. Machines uml3 and uml5 with two interfaces are ideal candidates for this. They can bridge the two networks they are connected to. These machines are running Linux, which has a built-in software to do bridging (functionality of learning bridges). The relevant command to use "brctl". Learn about this command by using "man brctl" or by using google.

3. Enable bridging on machine uml3 first using brctl (After setting up the bridge, you need to use ifconfig to bring up the bridge). What are the **interfaces** uml1 can reach now? Depend on how you enabled bridging, you may or maynot reach the interfaces of the bridge i.e. 10.0.0.3 or 10.0.0.4. For now, ignore this. Just assume that once a machine is configured as a bridge, it doesn't have to act as a host. After this setup, you have one bridge: uml3 and five hosts.
4. Enable bridging on machine uml5 next. After this, you have two bridges, uml3, uml5 and four hosts (uml1, uml2, uml4, uml6). What are the **hosts** uml1 can reach now?

You can type "brctl showmacs" at any bridge at any time to see what entries the learning bridge has learnt so far. To make it more interesting, set the ageing timer to 0. This clears out all entries except for the local interfaces of the bridge. Then set the ageing timer to 30. After this, run ping between some hosts. And check what the bridge has learnt using "brctl showmacs". Verify that entries match the MAC addresses of the corresponding hosts (you can use ifconfig to figure out the mac address of a host). You will see that after some time, these entries disappear because of the ageing limit you set.

Now we will enable spanning tree protocol. You can type “brctl show” to see that the spanning tree protocol (STP) is not enabled on either uml3 or uml5 currently. Learn how to enable STP, and enable it one after another on each bridge. Use the “brctl showstp” command to learn the status of the STP at each point of time.

5. From the results of the above command executed at both uml3 and uml5, specify which is the root bridge (uml3 or uml5?) and what is its ID? Who is the designated bridge for the lan segments Net0, Net1 and Net2 respectively.

After you are done playing with the simulator, do not forget to release it using the command “vnumlparser.pl -d lab06_bridging.xml -v”

Exercise 2: Bridging with loops

Now that you have warmed up, create a topology (title the file “lab06-stp-loop.xml”), with one or two cycles in it. Draw the topology in any drawing tool and export as a jpg (lab06-stp-loop.jpg), this will be part of your submission. If time is short, draw it on paper (specify roll numbers) and hand the sheet to the TA.

Make sure that in the topology, as you are enabling the bridges, STP is also enabled (it is disabled by default). Observe how the spanning tree topology changes as you enable the bridges one by one. Check the connectivity between some distant pairs of hosts. By looking at individual bridge stp entries (using the command brctl showstp) answer the following. Apart from this, for each bridge in your network, cut & paste the output of the “brctl showstp” to a file named “noloop-uml<id>” . For example in the previous exercise, you would have named the files noloop-uml3 and nolopp-uml5.

1. What is the id of the root bridge? Who is the designated bridge for each of the LAN segments in your network?

Based on the above information, copy and modify the figure in lab06-stp-loop to show the spanning tree. Essentially you need to remove all ports/interfaces of a bridge that are not active. Specify this file as “lab06-stp-noloop.jpg” (Draw on paper neatly if time is short)

Now disable the root bridge in your topology (by using brctl again). Now observe how the spanning tree changes. Make sure that it matches what you expect according to what you learnt in class. Check the connectivity between some distant pairs of machines again. Again for each bridge in your network, cut & paste the output of the “brctl showstp” to a file named “rbf-uml<id>”

1. What is the id of the new root bridge? Who is the designated bridge for each of the LAN segments in your network?

Again create a figure showing the new spanning tree and name this file as “lab06-stp-rbf.jpg”. Draw on paper if short of time. If done, do not forget to release the simulator instance.

Submission instructions

The directory named <rollnumber1>_<rollnumber2>_lab06 that you will submit should contain the following files:

1. lab06.txt
2. lab06-stp-loop.xml
3. lab06-stp-loop.jpg (or paper)
4. lab06-stp-noloop.jpg (or paper)
5. All the files noloop-uml*
6. lab06-stp-rbf.jpg (or paper)
7. All the files rbf-uml*

Now tar it as follows:

```
tar -zcvf <rollnumber1>_<rollnumber2>_lab6.tgz <rollnumber1>_<rollnumber2>_lab06/
```

Submit the file <rollnumber1>_<rollnumber2>_lab06.gz via moodle for grading.