Assignment 3:
Application Layer DHCP & DNS + Exploring the netlab room

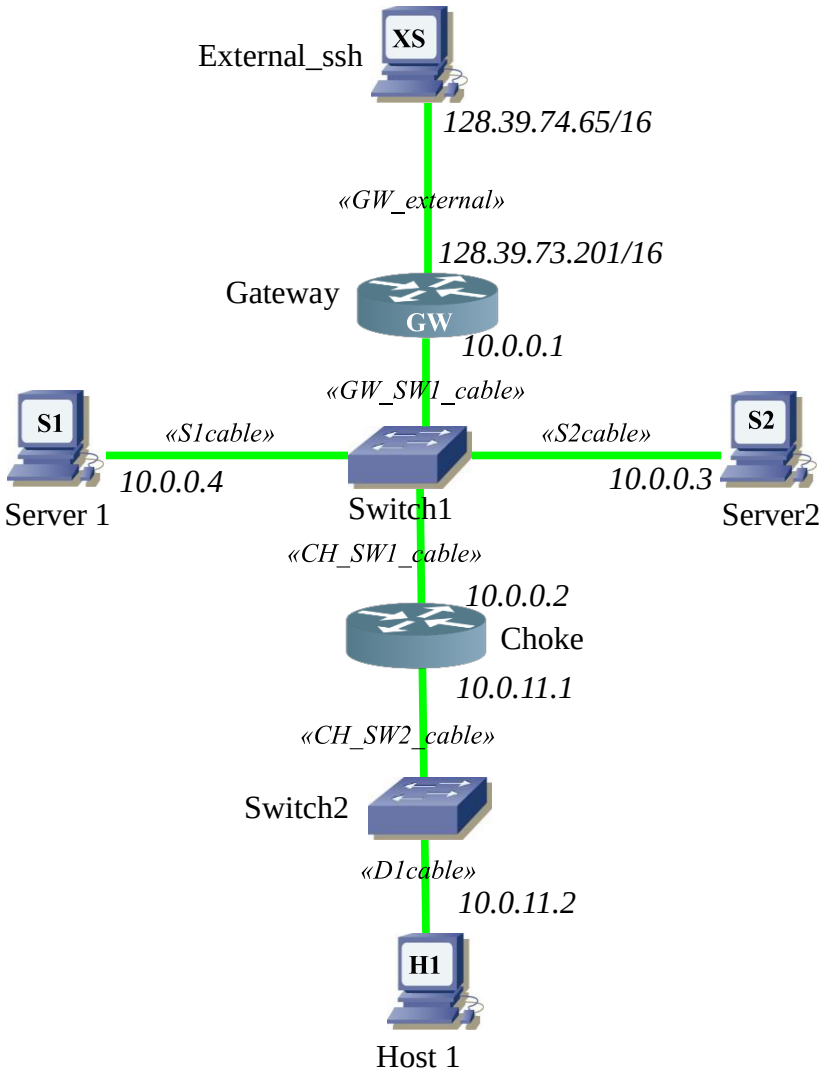
Starting: Week 36/ Deadline: Week 37

The goal of this assignment is to become familiar with minimal versions of the DHCP & DNS Application layer protocols. You will further develop your knowledge for configuring different types of network devices and making swift network adapter/cabling changes.
There are several problem solving tasks for which you are expected to do some troubleshooting. In the end, you will identify and get acquainted with the different equipment in the physical lab.

All assignments shall be handed in according to the "Documentation-Guidelines" document found in Fronter (see attachment)

PART I: DHCP and DNS in Virtual Lab

1. **Start up your network (set of VMs) that you created in the previous assignment.** It is assumed that you still have got the network and VMs that you created in last week's assignment. Otherwise, you must create them first by following the guidelines in the previous assignments. Your network, referred to as your "site", should look like this:



2. Exploring the config files of your Quagga router configuration.

a. As an exercise *(and in order succeed in the tasks outlined later),* we will re-configure the gateway, so that eth1 is the interface facing your internal network *(facing SW1)* while eth0 is the external interface facing e.g. External_ssh. First, you need to change the addressing of the two interfaces.
Q- In gateway bootlocal.sh, interchange the IP addresses between eth0 and eth1, so that it says "ifconfig eth110.0.0.1 …etc" and "ifconfig eth0 128.39.73.201 … etc". (Remember filetool.sh command for persistence.)

b. However, you also assigned addresses to interfaces when you set up the Quagga routing in the previous assignment, so we need to change this as well. Change directory into */usr/local/etc/*quagga and take a look at the files there. Run ls, and observe that you find configuration files for RIP, OSPF, BGP, which we will return to in later assignments.
Before you proceed, take a backup of *zebra.conf (e.g. sudo cp zebra.conf zebra.conf-bkup* and filetool.sh to store it persistently.) The configuration of your static routes is found in zebra.conf file. Generally, you need to be very careful if you want to make changes directly into these files, thus the backup.
Q - Now, open the zebra.conf file and replace the 10.0.0.1/24 address for eth0 with 128.39.73.201/16 without making any other changes, and then replace the 128.39.73.201/16 address for eth1 with 10.0.0.1/24. Store the changes persistently (sudo filetool.sh -b), and power off the gateway (sudo poweroff).

c. Before powering up the gateway, you must not forget to also change the adapters, because Adapter 1 is automatically assigned eth0 by VirtualBox and should now be connected to *"GW_external"* (and not to *"SW1_cable"* as it was in the previous assignment). Likewise, Adapter 2 is automatically assigned eth1 and should now be connected to "SW1_cable" (and not to *"GW_external"* as it was in the previous assignment).
Q – Change the adapters and power on the gateway and do some network-wide ping testing e.g. between Host1 and External_ssh to ensure that everything works OK. Otherwise, you need to do some troubleshooting.

3. A minimal DHCP server/deamon (*udhcpd*)

a. Up until this point, we have configured a manual IP address in bootlocal.sh, but at the same time we have had a dhcp-client (called udhcpc) running in the background. Since you have not had any dhcp servers on your network, this has not caused any problems. However, if there were a dchp server running in your network, the udhcpc process would quickly obtain a new address from it and would override your manual IP address configuration. Below, we are going to install a dhcp-server in your virtual network, so we need to turn off the dhcp client process at the nodes (gateway, choke and Server2) where we want to maintain a manually configured IP:
Q - On Gateway, Choke and Server2 you could for example disable (i.e. kill) the dhcp client process in by adding the line "pkill udhcp" in bootlocal.sh, before the lines with the manual assignments of IP addresses (ifconfig lines). Either you do a reboot, or you run the same command at these nodes (with sudo) before you proceed.

b. Even though it might be more natural to implement DHCP-server as an integrated feature of the router, this is not a necessity. Instead, in this assignment we are going to install it at Server2. At Server2 you should create a persistent file called */etc/udhcpd.conf* containing the configuration of your DCHP-server.
Q - Make a search on the Internet (or see Resources below) to find out possible formats of the config file of udhcpd. The requirements of your configuration are the following: Your server

shall hand out addresses in the range 10.0.0.100-10.0.0.200 with a 24-bit subnet mask, and addresses should be leased out over a time of 72 hrs. Show what your configuration file looks like.

c. Now, you can start the udhcpd-deamon at Server2, by issuing the command: sudo udhcpd /etc/udchpd.conf &. Demonstrate that all the following nodes: Server 1, Gateway and Choke work after having rebooted them.
Q – Start udhcpd-daemon and answer following questions. What address is obtained by Server1? Explain. If you meet any problems, we expect you to do some troubleshooting and debugging yourself to resolve any problems – or document it.

d. We did not disable the background udhcpc process at Host1. You might run "sudo udhcpc" *(i.e. in the foreground)* at Host1, to check what is happening.
Q - How is Host1 affected by the udhcpd process at Server2? Explain.

e. What is the purpose of a DHCP relay?

f. It would be convenient to have the opportunity to allow dhcp to allocate a fixed IP address to some nodes in your network. Server1 uses dhcp to obtain an address, udhcpd at Server2 does not know from which node this request comes, so it will get an arbitrary address, and not a fixed address as we would like. Now, we are going to change that that: To demonstrate this feature, your task is to ensure that the address 10.0.0.50 is always allocated to Server1. To do so you can use the following line in the configuration file: *"static_lease 11:22:33:44:55:66 10.0.0.50"* where you replace *"11:22:33:44:55:66"* with the actual hardware address of the interface at *server1 (e.g. use ifconfig eth0)*. Run udhcpc again at server1, and confirm that it works. Later in this assignment, we will use dnsmasq instead to offer this functionality and much more. dnsmasq provides a menu of services, including a built-in dhcp server. Therefore, we are now going to skip using udhcpd at Server2 (thus, if you have stored the command "udhcpd /etc/udchpd.conf &" in bootlocal.sh, you should comment it out.) You simply kill the udhcpd process at Server2 or do a reboot. Before we look at dnsmasq, we need to explore the /etc/hosts file in the step below.
Q - Ensure that the address 10.0.0.50 is always allocated to Server1.

4. The /etc/hosts file.

a. Q - At Server2, issue the following command *"ping gateway".* What error message do you get?

b. Q - Now, update the /etc/hosts - file at Server2, by adding the two lines "10.0.0.1 gateway" and "10.99.99.2 choke". *(We intentionally have set an incorrect choke-address for now for testing purposes)*: Now, at Server2 issue the command "ping choke". What happens? Then, issue "ping gateway" again. What happens?
It would be very convenient to have this functionality at every node, without having to configure the /etc/hosts file at every node specifically. Later in this assignment we will obtain this functionality by using dnsmasq.

c. There are many ways to make the changes to /etc/hosts persistent. Make it persistent! Reboot, and check that it is OK.
Q – How can you make changes persistent in /etc/hosts file?

d. If you did not succeed in the previous step, the reason might be that /etc/hosts is overwritten by a process after reboot, but before bootlocal.sh is run. So, to fix the problem, you might add the following lines in bootlocal.sh, *"echo 10.0.0.1 gateway >> /etc/hosts"*, *"echo 10.99.99.2 choke >> /etc/hosts"*. (These lines will be added after the new hosts file has been written by sethostname). Reboot and check that the hosts file now is OK, and that you can ping gateway from

Server2 with our local hostnames instead of IP addresses.

5. Using dnsmasq for local DNS

a. First, install dnsmasq (hint: "ab dnsmasq.tcz") at Server2. Based on previous assignments, you should now be able to solve the networking issues (switch temporarily to NAT interface at adapter) to do so. What is the procedure (in brief)? Now, copy the configuration file like you did for ssh: "sudo cp /usr/local/etc/dnsmasq.conf.example /etc/dnsmasq.conf" and ensure that the configuration is stored persistently after reboot.
Q – How can you install dnsmasq at Server2? (brief description)

b. For a very minimal configuration just to make dnsmasq work, you can do the following: Configure /etc/dnsmasq.conf, by first enabling "domain-needed" and "bogus-priv" (probably at lines 14 and 16, or close to these lines). Set the dhcp-range=10.0.0.3, 10.0.0.100,12h (around line 136 or alternatively 142). Also set the dhcp-leasefile to a valid place in your file system, e.g. "dhcp-leasefile=/tmp/dnsmasq.leases" (line 413 in the configuration file). Since in this assignment we are not running dnsmasq at the gateway (which would be a common configuration and which is thus assumed by default), set "dhcp-option=option:router , 10.0.0.1" (around line 250). Since we do want static configuration of the routers (gateway and choke) add two lines in the conf script (around line 195). For example, to ignore dhcp requests from the gateway add *"dhcp-host=11:22:33:44:55:66, ignore*" where you replace *"11:22:33:44:55:66"* with the actual hardware address of the 10.0.0.1 interface at gateway (e.g. use ifconfig eth1), and then add a similar line for choke. Now you can start dnsmasq. You might run it first in the foreground in debug mode "sudo dnsmasq –d". Later, for example at the end of the assignment, you might run it in the background instead (e.g. "sudo dnsmasq &" in bootlocal.sh).
Q - Demonstrate that dnsmasq with above mentioned configurations work.

c. Q - Run udhcpc at Server1. Which address does it get? How about local dns-server? (e.g. cat /etc/resolv.conf)

d. Q - If you run *nslookup choke* from Server1, what do you observe, and why?

e. Q - If you now turn off dnsmasq at Server2 (or use the VirtualBox GUI to put the VM at pause) what happens when running the command *nslookup choke* at Server1?

f. Even though gateway and choke will not get a dhcp response, configure them manually to use Server2 as a local dns server, by adding the line "echo nameserver 10.0.0.3 >> /etc/resolv.conf" in the bootlocal.sh file of these two nodes. Also configure Host1 to use Server2 (10.0.0.3) as a local dns server.
Q – Demonstrate that gateway, choke and Host1 use Server2 as their dns server.

g. Now, configure Server1 so that it gets a fixed address: Around line 164 in your script add the line: "dhcp-host=11:22:33:44:55:66, serverX, 10.0.0.4,12h" where you replace "11:22:33:44:55:66" with the actual hardware address of the interface at server1. Run udhcpc again at server1, and confirm that it works.
Q – Configure Server1 to get fixed address and run nslookup serverX or ping serverX to confirm that the name lookup works.

h. You have now scratched the surface of what can be done with dnsmasq, by testing out the simplest configuration. Now, you should be able to configure dnsmasq more properly to accommodate dhcp and dns for all your nodes on subnet 10.0.0.0/24, and also tidy up. For example; replace serverX with server1 in the conf-file, use correct IP address for choke in /etc/hosts at server2 and also add server1, server2, external_ssh and host1 to this file, so that their IP addresses can be resolved. You might also configure dnsmasq with an arbitrarily chosen local domain name (which will never leave your site, if you have configured everything correctly), and so forth.
   Q – In order to above mentioned instructions, configure a solid, correct and consistent dnsmasq configuration, and document what you have done.

6. Obtaining Internet access from your virtual network

a. You are not required to maintain connectivity to External_ssh, so you can use eth0 of Gateway to connect to Internet. Some hints to get started: Poweroff gateway, and assign an adapter (e.g. Adapter3) to the gateway configured to a NAT interface of VirtualBox and disable Adapter 1 (or you might reconfigure Adapter1 to a NAT interface). VirtualBox will automatically assign the NAT interface to eth0, so that the steps you took in task 2) above are useful to succeed here. You might also have to set up the gateway with proxy arp at the external interface that faces the NAT of VirtualBox by adding an additional sysctl-line to bootlocal.sh at the gateway: "sysctl –w net.ipv4.conf.eth0.proxy_arp=1". Document what you did to succeed or document what stopped you.
   Q - Try to obtain Internet connectivity from your site. Document how you do it.

b. Q - Check/document that you can successfully ping a global IP address on the Internet (e.g. of vg.no) from Host1. (The network capacity might we limited in this virtual environment, so only one ping session from the site at a time might be a good idea.)

c. Q - Note down the nameservers that the gateway obtains from running dhcp on the external interface (to the NAT of VirtualBox) by running "cat /etc/resolv.conf" at the gateway. Add these nameservers persistently to the /etc/resolv.conf file of Server2 (e.g. by adding lines like "echo nameserver 158.36.161.21 >> /etc/resolv.conf" to bootlocal.sh). dnsmasq at Server2 will automatically read the external nameserver addresses from this file. Confirm that you now at Host1 can resolve from dnsmasq at Server2 both local names (e.g. by running "nslookup gateway" at Host1) and FQDNs (e.g. by "*nslookup www.vg.no*").

d. According to the content of /etc/resolv.conf of gateway, the gateway is not able to resolve local names (e.g. like "nslookup host1").
   Q - Reconfigure the gateway, so that it is able to resolve both local names and FQDNs. (Hint: If it uses Server2 as its local dns server instead, dnsmasq will ensure that it is able to resolve both types of names.)

7. Documenting your final dnsmasq configuration

a. Q - For proper documentation, summarize the dnsmasq features you have configured and summarize all the dnsmasq settings you have enabled in a small, and nicely edited file. (You might use the diff command, e.g. something along the lines of *sudo diff /etc/dnsmasq.conf /usr/local/etc/dnsmasq.conf.example | grep "^-" > /home/tc/my_dnsmasq_settings.txt* before you edit the resulting textfile and take a screendump of it).

You have finished the first part of this assignment! Make sure configurations are persistent and store away the virtual machines you have created, so that you can continue working with them on the next assignment.

PART II: Using TCPDump to capture TCP connections:

The TCPDump is a powerful tool to capture packets passing through the network. You use the TCPDump here in order to capture TCP connection packets.

a. Check out that you have connectivity between Host1 and Server2 *(if not do troubleshooting).* Then check out that the openssh server is running on Server2 and Host1. *(hint: issue "cp sshd_config.example sshd_config"* command, then *"sudo /usr/local/etc/init.d/openssh start"* command to start openssh server at Server2. Also set password for user tc by issuing *"passwd tc"* command to use for SSH session). Finally, install tcpdump at choke.(Hint: use *"ab tcpdump"* command to install tcpdump at choke).

b. Now at choke, you require TCPdump to capture traffic in interface which connected to the Host1 that is eth1. Then, you can issue " sudo tcpdump –i eth1 –w ssh.dump not port 67" command in the choke.
Q – capture SSH traffic passing through the choke's interface eth1.

c. Now, you can run an SSH connection from Host1 to the Server2 by issuing " ssh 10.0.0.3" command in the Host1. Then TCPDump at choke captures the SSH TCP connection.

d. You can simulate the packet loss in this TCP connection by shut down the Server2 network interface. You can do this by issuing the " sudo ifconfig eth0 down" command at Server2. After shutting down the Server2 network adapter then make sure the SSH connection disconnected.

e. Now you can end the TCPdump capture in choke. Also, you can get information about TCPdump in *http://www.tcpdump.org/tcpdump_man.html* and learn how to interpret the output.
Q – Read the TCPdump output and describe TCP three-way handshake order. Describe the Host1 reaction to the connection lost between Host1 and Server2.

You can up the Server2 network interface with *"sudo ifconfig eth0 up"* command.

The deadline for completing and submitting this assignment into Fronter is at 23:30 (11:30 pm) two days after the lab in the week of the deadline (see top).

Resources:

- http://manpages.ubuntu.com/manpages/hardy/man5/udhcpd.conf.5.html
- http://forum.tinycorelinux.net/index.php?topic=2338.0
- http://en.wikipedia.org/wiki/Dnsmasq
- https://www.linux.com/learn/tutorials/516220-dnsmasq-for-easy-lan-name-services
- http://www.tcpdump.org/tcpdump_man.html