Assignment 4

Instructions:

1. Make sure that you read, understand, and follow these instructions carefully. Your cooperation will help to speed up the grading process.
2. Following are generic instructions. Make sure that you also check carefully and follow any specific instructions associated with particular questions.
3. Complete the assignment with the best of your own capabilities and follow the instructions as mentioned below.
    a. Create a folder named as your roll no.
    b. Create separate folders for each questions inside it and name them as Q-1, Q-2, etc.
    c. Each folder must contain:
        i. All your program files
        ii. Makefile to compile your programs if required.
        iii. Readme/report file to explain your assignment and answers to the questions.
    d. Compress the entire submission folder as a single file. After this step, the file name should be as your roll no. When done, submit the file in Moodle on or before the submission deadline.

Ethical Guidelines (lab policy):

1. Deadlines: Deadlines should be met. Assignments submitted after their respective deadlines will not be considered for evaluation.
2. Cheating: You are expected to complete the assignment by yourself. Cases of unfair means and copying from another student will not be tolerated, even if you make cosmetic changes to them. If we suspect any form of cheating, we are compelled to award ZERO marks.
3. If you have problem meeting a deadline, it is suggested that you consult the instructor/TAs and not cheat.

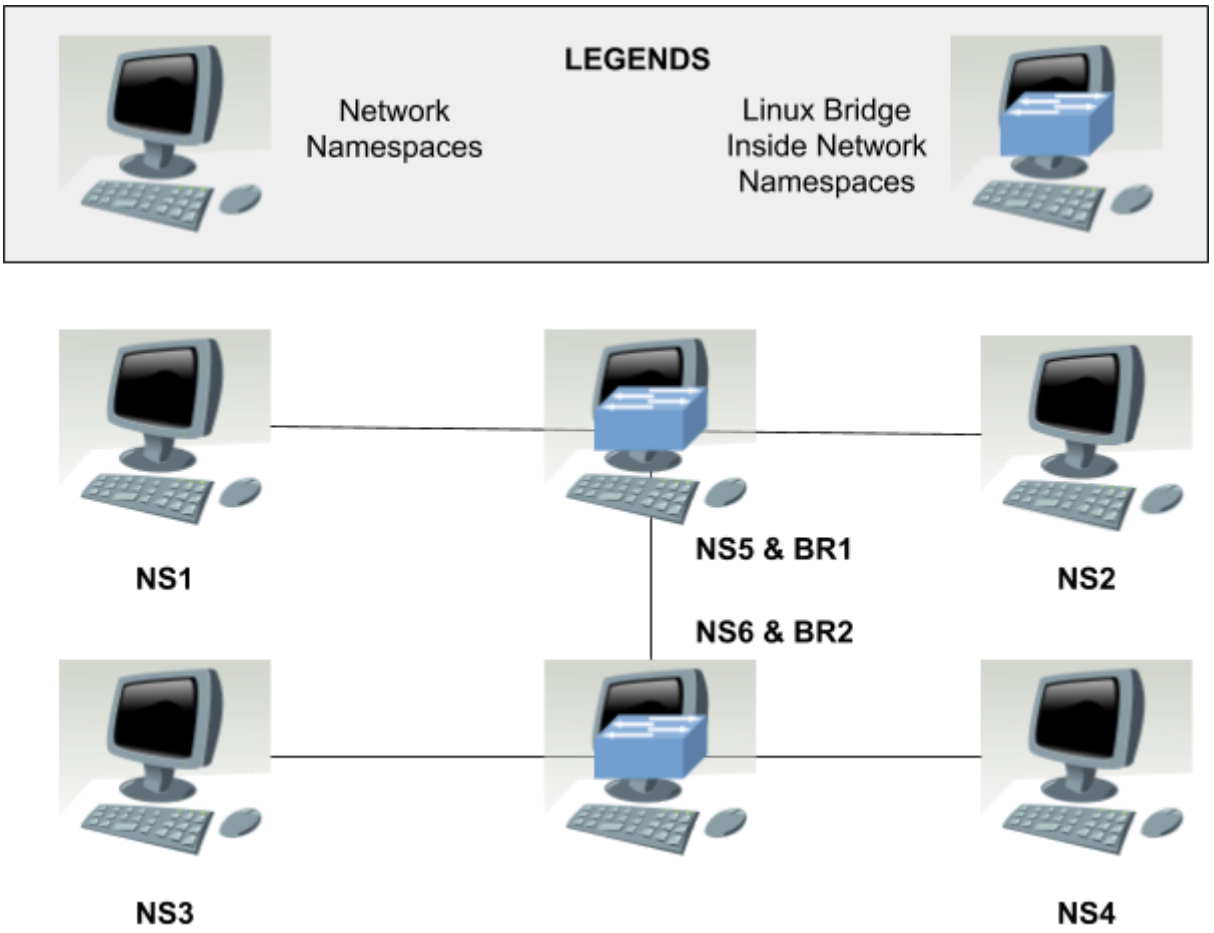

Fig.1

Question 1:

This assignment targets emulation of network protocol stack. For this purpose one can use network namespaces.

"A network namespace is logically another copy of the network stack, with its own routes, firewall rules, and network devices"[1].

The instances of the network namespaces can be connected via virtual ethernet interfaces (veth)[2]. Attach the veths to the bridge namespaces. Attach the bridge interfaces via Linux bridge control utility[3].

The step by objectives are as follows.

1. Create a "Oracle virtualbox" host machine with latest image of Ubuntu. Inside virtual host
    a. Create network namespaces (NS1 to NS6) as given in Figure-1.
    b. Create virtual ethernet (veth) connections between them.
    c. Attach bridges to the bridge namespaces and attach corresponding veths to the bridges. After this step, the namespaces should be able to ping each other.
    d. Assign 10ms of delay to each of the veths using linux "tc" and test the

[1] http://man7.org/linux/man-pages/man8/ip-netns.8.html
[2] http://man7.org/linux/man-pages/man4/veth.4.html
[3] https://www.tldp.org/HOWTO/BRIDGE-STP-HOWTO/set-up-the-bridge.html

connections using ping.

e. List all the commands in form of a shell script.
f. Write a separate cleanup script to remove all the veths, namespaces and bridges from your system.

[13]

Question 2:

Network debugging is one of the important task of a network administrator. For this purpose often an administrator uses “traceroute” utility[4]. A python based implementation of traceroute can be found here.

Traceroute is helpful for identification of network layer faults. However, in this assignment you are supposed to create a similar utility for data link layer debugging. Let us call this utility as “tracemac”. Tracemac requires ethernet headers. Find a sample C code to send and receive ethernet frames from here[5][6]. The tasks of the tracemac utility are as follows.

1. Tracemac finds the layer 2 path between the source node and a target IP address.
2. Command sample: `tracemac <target IP_address>`
3. For this purpose you need to write a client which uses raw sockets for communication. You also have to deploy servers in daemon mode corresponding to the tracemac utility which can identify the tracemac events and perform accordingly.
4. Your implemented utility must time-out after a predefined number of retries if no target address is found.
5. Test your application for different test cases. For testing purpose you must use the scenario given in Figure-1.
6. Compare the results with the results obtained from running the traceroute utility. (and answer the questions)
    a. Can you see all the devices using traceroute?
    b. Can you see all the devices using tracemac?
    c. Can your implemented tracemac be used for a target mac address?

[17]

---

[4] http://web.mit.edu/freebsd/head/contrib/traceroute/traceroute.c

[5] https://gist.github.com/austinmarton/1922600

[6] https://gist.github.com/austinmarton/2862515