Hands-on Lab Description

CS-CNS-00001 –
Packet Filter Firewall (iptables)

Category:

```
CS-CNS: Computer Network Security
```

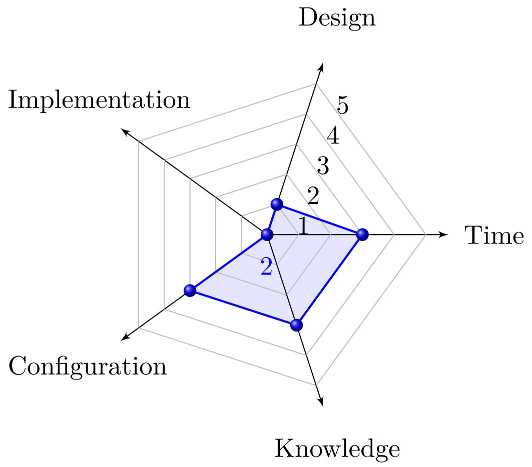
Objectives:

```
Setup packet filter firewall (iptables) to allow and block network traffic
Setup Network Address Translation (NAT) service
Use basic networking and diagnostic tools such as ifconfig, ip, route, netstat, ping,
     traceroute, and tcpdump
Use iptables to regulate network traffic and enable services such as Web service
    based on provided traffic policies
```

Estimated Lab Duration:

```
Expert: 50 minutes
Novice: 150 minutes
```

Difficulty Diagram:



Difficulty Table.

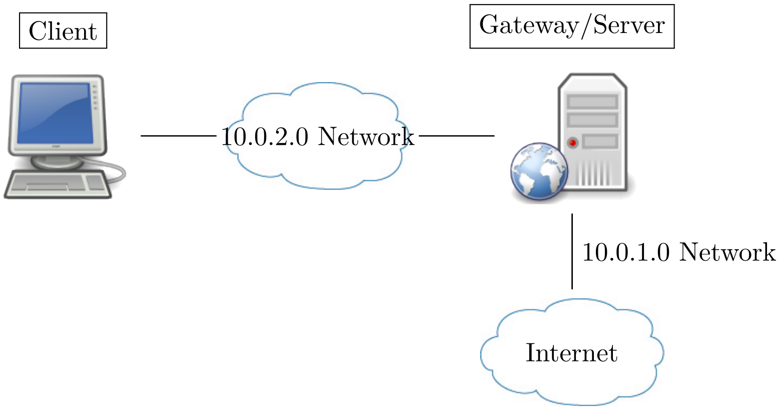
Measurements	Values (0-5)
Time	3
Design	1
Implementation	0
Configuration	3
Knowledge	3
Score (Average)	2

Required OS:

```
Linux: Ubuntu 18.04 LTS (Bionic Beaver)
```

Lab Running Environment:

```
VirtualBox https://www.virtualbox.org/ (Reference Labs: CS-SYS-00101)
```



```
Client: Linux (Ubuntu 18.04 LTS)
Gateway/Server: Linux (Ubuntu 18.04 LTS)
Network Setup:
Both VMs connected to Local NAT Network: 10.0.2.0/24
Internet is connected to Gateway/Server VM through network: 10.0.1.0/24
```

Lab Preparations:

```
Know how to use Linux OS (Reference Labs: CS-SYS-00001)
Basic knowledge about computer networking (Reference Labs: CS-NET-00001 and CS-NET-00002)
Know how to setup services such as Web (CS-SYS-00003).
```

Lab Overview

In this lab you will explore the packet filter firewall by using Linux firewall *iptables*. The first part of the lab will set up necessary *iptables* running environments; the second part of the lab specify the requirements to implement firewall filtering rules to enable and disable network traffic.

The assessment of the lab is based on the completeness of implementing firewall filtering rules that satisfy the required firewall security policies. Students need to submit a sequence of screenshots and corresponding illustrations to demonstrate how they fulfilled the firewall's packet filtering requirements.

In summary, students will do:

- Set up network packet forwarding and inspect traffic using tools such as ifconfig, route, ip, ping, traceroute, and tcpdump
- Check services setup of apache2 web service
- Use and test iptables-based packet filter firewall to enable and disable access to the established services

Task 1 Preparation of setting up lab environment

Preparation:

1. Review and exercise the CS-SYS-00101 (Linux tutorial) to create Client and Gateway/Server VM in VirtualBox on your computer.
2. Review and exercise the following labs CS-SYS-00001 (Linux tutorial), CS-NET-00001 (Network setup) and CS-NET-00002(Gateway setup) on both Client and Gateway/Server VM before you do Task 1.1. (Set Gateway/Server VM as the Gateway for the Client VM)
3. Review and exercise the Web Service Lab (CS-SYS-00003) on the Gateway/Server VM before you do Task 1.2.

In this lab, an *iptables* firewall running script template is provided, which allow you to manage and run your *iptables* rules easier. You can download from Coursera's Project 1 page and put it on the Gateway/Server VM. The firewall script template file is a shell script. To make it executable, you can change its permission by (refer to more details in the lab CS-SYS-00001 on Linux file permission):

```
$   sudo chmod 755 rc.firewall % this will change to the file to green when you show
      'ls -l' command
```

The *rc.firewall* is a shell script to help you manage your firewall rules easier. It only include some basic setup. To fulfill the overall goal of this lab, you need to add and update firewall rules in it. To edit and run the script file, you can:

```
$   gedit rc.firewall % use vim to edit the script. The script has comments that are
      sufficient to self-explain.
$   sudo ./rc.firewall   % run the script
```

Task 1.1 Test network connectivity

The first step is to check the connectivity among VMs.

1. use *ping* to check connectivity (if this step is successful, please skip to Task 1.2):

```
$       ping ip_address % you need to performs a mutual ping between any pair of
          VMs that are connected on the same local networks
```

Usually, unsuccessful *ping* responses can be resulted in the following cases:

Ping response	Possible Reason
Request timed outs	timeout exceeds, e.g., windows default time out is 4s
No reply from <destination>	no response from the destination, the routers along the path working properly.
<destination> is unreachable	source nodes does not know how to get to the destination, i.e., something wrong about the routing
ICMP host unreachable from gateway	the gateway/router forward you packets is improperly setup

2. Check if you default gateway is properly set up on your client VM(Don't change gateway setting in the Gateway/Server VM). The default gateway should be set to the gateway/server's IP address. For example:

```
$       route -n   % check default gw setup
$       sudo route add default gw <default_gw_ip> <interface_to_gw_net> % set
          default gw to the default gw IP though the directly connected interface
```

   For details on how to check the default gateway setup, you should refer to the lab CS-NET-00002. After checking/setting default gateway configuration, performs ping to each other again.

3. If you can still not ping the gateway from the client or server, you may want to check if the firewall setup on the gateway blocked the ping. You may want to disable the firewall on the gateway and try to ping again:

```
$       sudo ufw disable % disable the firewall
```

4. After checking the connectivity, next you should check the packet forwarding setup on the Gateway/Server VM:

   (a) Enable packet forwarding on the gateway

```
$          sudo echo "1" > /proc/sys/net/ipv4/ip_forward
```

   (b) To check your current iptables rules setup, you can issue the following command:

```
$          sudo iptables -L  % display the filter table policies. For
             whitelist, the default policy for INPUT, OUTPUT and FORWARD
             chaines should be DROP.
```

   (c) On the gateway, clean up all existing iptables rules (**used in care if you have already establish iptables rules**):

```
$          sudo iptables -F % flush all existing chains
$          sudo iptables -X  % delete all user defined chains
```

   (d) Set iptables default policies to *blacklist*[1] Afte the following iptables setup, you should be able to ping from any VM to other VMs.

```
$          sudo iptables -P INPUT ACCEPT % option -P means default policy
$          sudo iptables -P OUTPUT ACCEPT
$          sudo iptables -P FORWARD ACCEPT
```

[1] The firewall blacklist policy means only block known illegitimate traffic and allow all unspecified network traffic pass through.

After the presented steps, you firewall rules are flushed and no restriction to sending packets among VMS. Thus, you should be able to ping between any pair of VMs.

Task 1.2 Test installed software and services

The second step is to make sure the project required software packages are installed properly on given VMs. Note that you may need to adjust the configurations of *apache2* and make it accommodate to requirements presented below.

1. On the Gateway/Server, test the web server make sure it is working properly:

   (a) Test Apaches server running the following command:

   ```
   $         service apache2 status
   ```

   (b) Establish a demo website by editing the file */var/www/html/index.html* and add a statement such as "Welcome to Demo and Test!". For more informsation about how to set up a web service, please refer to the system lab CS-SYS-00003 (Basic Web Service (Apache) Setup on Linux).

Task 1.3 Reset firewall to whitelist

After ensuring the network connectivity is good and the client access all the services described in Task 1.2, you need to enforce the *whitelist* firewall policy as the start point of your lab setup for the next task and flush out all existing firewall rules and chains on the Gateway/Server VM. The firewall whitelist policy means that the firewall only allows known legitimate traffic to pass through and it will drop all unspecified/unknown network traffic. After setting up the *whitelist* policy, you **SHOULD NOT** ping between the VMs and you should not be able to access to web service established on the Gateway/Server VM from the Client VM.

On the Gateway/Server VM, first, flush iptables chains and delete all user-defined chains:

```
$    sudo iptables -F        % flush ipables rules
$    sudo iptables -X      % delete user defined chaines
```

Second, set the default iptables policy to*whitelist*:

```
$    sudo iptables -P INPUT DROP % option -P means default policy
$    sudo iptables -P OUTPUT DROP
$    sudo iptables -P FORWARD DROP
```

Now, you should not be able to ping between VMs and you cannot access web service from the client VM.

Task 2 Requirements for setting up a Stateless Packet filter firewall

On the Gateway/Server VM, please set up the following packet filtering rules.

1. Check and set the default *iptables* policies to *DROP* for *INPUT*, *OUPUT*, and *FORWARD* chains. This setup is basically implement a ***whitelist*** policy, i.e., only allowing specific network traffic as "good" traffic to pass through, and thus disable all other non-specified traffic. Note that **only** allow the required traffic flow and connectivity described in below, and drop all other network traffic and access.

2. Allow the client to access the web page (http) on the server by ip address of the Gateway/Server VM . The demo web page should contain a keyword "Welcome", such as"Welcome to the demo and test web page!"

3. Stop the client from pinging the Gateway/Server VM’s IP address.
4. Allow the client to ping 8.8.8.8 (an public IP address on internet).

Deliverable

Students need to finish a project report that contain a sequence of screenshots and explanations to show that they achieved requirements described in the *Lab Assessment* section. A project report template will be provided in the project page in Coursera.

Lab Assessments (100 points)

Lab assessment for accomplishing Task 1 and Task 2 depends on the following facts:

1. (30 points) The client
   - can not ping the Gateway/Server VM IP address
   - can access the demo webpage on Gateway/Server VM by access the IP address of Gateway/Server VM in browser (the returning page must contain ”Welcome ....”, you can also use a web browser)
   - can ping 8.8.8.8.
2. (30 points) The Gateway/Server VM should
   - set up http(webpage) service to it’s own IP address (with the demo page avalable).
   - enable POSTROUTING to allow client to access outside network(8.8.8.8) and change their source IP addresses.
3. (40 points) Additional requirements
   - You should set the default firewall policy to DROP for INPUT, OUTPUT, and FORWARD chains.
   - Besides the allowed network access described the above, you should not allow any other network access. Provide screenshots for the following results:
     On client VM:

```
$          sudo nmap -sT -p- 10.0.2.x % x is the value of your
             Gateway/Server VM’s IP address
$          sudo nmap -sU -p- 10.0.2.x % x is the value of your
             Gateway/Server VM’s IP address
$          ping 8.8.8.8 % This should be working
$          ping 8.8.4.4 % This should be not working, as you should drop all
             traffic that is not required in the requirement.
$          ping 10.0.2.x  % x is the value of your Gateway/Server VM’s IP
             address, This should be not working
```

On Gateway/Server VM:

```
$          ping localhost % This should be not working
$          ping 10.0.2.y % y is the value of your client’s IP address,this
             should be not working
$          ping 8.8.8.8 % This should be not working
```

Related Informsation and Resource

```
Whitebox, blackbox, and greyboax testing:
    https://www.nbs-system.com/en/blog/black-box-grey-box-white-box-testing-what-differences/
DNS Installation and setup on Ubuntu:
    https://help.ubuntu.com/lts/serverguide/dns.html.en
IptablesHowTo https://help.ubuntu.com/community/IptablesHowTo
Iptables Tutorial 1.2.2
    https://www.frozentux.net/iptables-tutorial/iptables-tutorial.html
```