Hands-on Lab Description

Project 2
Packet Filter Firewall (iptables)

CONTENTS

Category:

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

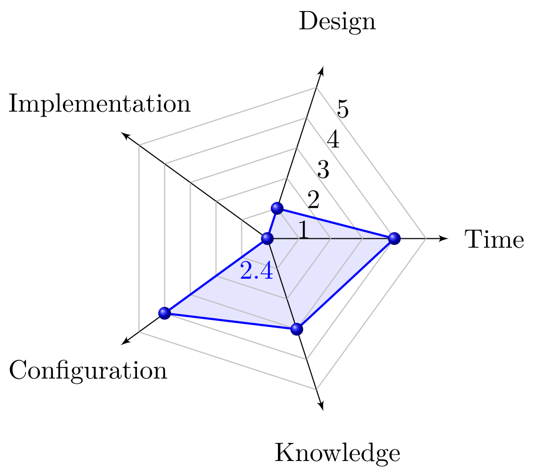
Objectives:

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

Estimated Lab Duration:

```
1  Expert: 120 minutes
2  Novice: 360 minutes
```

Difficulty Diagram:



Difficulty Table.

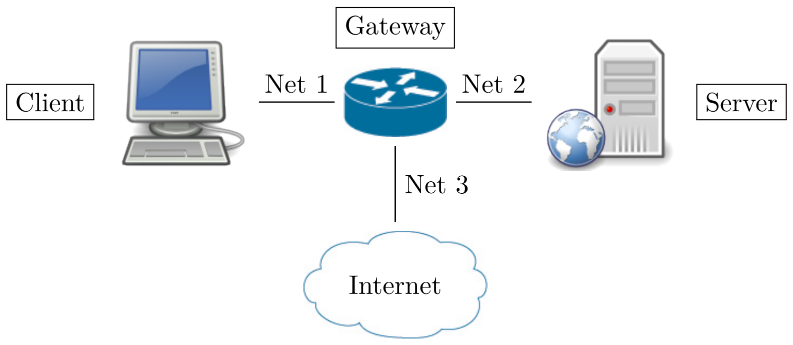
Measurements	Values (0-5)
Time	4
Design	1
Implementation	0
Configuration	4
Knowledge	3
Score (Average)	2.4

Required OS:

```
Linux: Ubuntu
```

Lab Running Environment:

Vmware or Other VM solution on student PC



```
Client: Linux (Ubuntu 18.04 LTS)
Server: Linux (Ubuntu 18.04 LTS)
Gateway: Linux (Ubuntu 18.04 LTS)
Network Setup:
Internet is connected through Net 3: 172.16.0.0/12
Client-side Net 1: 192.168.0.0/24
Server-side Net 2: 10.0.0.0/8
```

Lab Preparations:

```
Know how to use Linux OS
Basic knowledge about computer networking

Know how to setup services such as Web, FTP and SSH.
```

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 such as apache2 web service, vsftp service, ssh service
- Use and test iptables-based packet filter firewall to enable and disable access to the established services

1 Task 1 Preparation of setting up lab environment

Suggestions:

1. Review and exercise the the following labs: Web (CS-SYS-00003), FTP (CS-SYS-00006) 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 the lab resource repository and *unzip* it. You can download lab resource files by using following *wget* and *unzip* command. Note that check if *wget* is installed using command:

```
$    wget --version
$    unzip --version
```

If *wget* and *unzip* are not installed, you can install it using the following command:

```
$    sudo apt install wget
$    sudo apt install unzip
```

And then, download the zipped lab file:

```
$    wget
       https://gitlab.thothlab.org/thoth-group/ThoThLabResource/raw/master/lab-cs-cns-00001a.zip
$    unzip lab-cs-cns-00001a.zip
$    cd lab-cs-cns-00001a
```

The firewall script template file "*rc.firewall*" is located in the folder 'lab-cs-cns-00001a'. It 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:

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

1.1 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 and server. The default gateway should be set to the gateway's IP address directly connected to the client's or server's network. 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:

   (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
```

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.

1.2 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 *bind9*, *apache2*, *ssh*, and *vsftp* and make them accommodate to requirements presented below.

1. On the server, test the web server make sure they are 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).

2. On the server, test the ftp server and make sure the vsFTP server runs in the passive mode, i.e., the data channel will be initiated by the client. Thus, port forwarding is required to be established on the gateway to forward ftp data channel request to the specified data receiving port on the server side. Please refer to system lab CS-SYS-00006 (FTP (vsFTP) Setup on Linux) for more details on how to setup vsFTP as a passive mode. The following configuration on the ftp server need to be enforced:

   - Enable the anonymous access (i.e., no need to provide an user account and password).
   - Enable the passive mode, client can initiate the data channel through ports range [30000,30099].

   You can test ftp service locally. For this project, you may not need to setup the ftp authentication and security. However, you need to make sure that the passive mode is enabled.

3. Finally, on the server, test if the SSH server is running.

```
$       sshd -v    % show ssh server version
$       ssh ubuntu@localhost % setup an ssh connection to the server itself. SSH
          server does not allow root to remote acess to the server, thus you should
          use the user account 'ubuntu' to access the ssh service.
```

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

1.3 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. 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 any given VMs and you should not be able to access to any of the services established on the server from the client.

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 services hosted by the server from the client.

2 Task 2 Requirements for setting up a Stateless Packet filter firewall

On the gateway, please set up the following packet filtering rules. For each required rule, demonstrate how each rule can be satisfied.

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. Consider the server-side network is a private and protected network, and configure NAT service properly on the gateway to change NATed IP addresses access the server-side private network. (hint: you can use tcpdump to capture the traffic on the client-side network to verify that packets sent by the server has been changed to the gateway’s IP address).

3. Allow the client to access the web page (http) on the server using gateway's IP. The demo web page should contain a keyword “Welcome”, such as“Welcome to the demo and test web page!”

   - http://192.168.0.100

4. Allow the client to access the ftp server hosted by the server using the passive mode and allow anonymous access, i.e., use the following command to access the FTP server:

```
$      ftp -p 192.168.0.100 % use ’anonymous’ as the user ID and password
         to access the server
```

   Set up the passive data ports opening from the server in the range of [30000, 30099].

5. Allow the client to ping the gateway's client-side IP address.
6. Allow the server to ping the gateway's server-side IP address and the client's IP address.

3 Deliverable

Students need to submit a sequence of screenshots with explanations on they can achieve requirements described in the *Lab Assessment* section.

4 Lab Assessments (100 points)

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

1. (40 points) The client
   - can ping the gateway IP address that is on client side network,
   - can ssh ubuntu@gateway IP, and gain access to server
   - can ftp -p gateway IP and access to FTP server running on Server VM(using anonymous and passive mode, note that the client cannot issue command such as "ls" in active mode and the system will hang)
   - can curl gateway IP and access to Apache server running on Server VM (The returning page must contain "Welcome ....", you can also use a web browser)
   - can ping the gateway's IP address on the client side.

2. (25 points) The gateway should
   - set up port forwarding for www, ssh, and ftp to the server's IP address (NATed IP address).
   - enable POSTROUTING to allow server to access outside network and change their source IP addresses.

3. (25 points) The server can
   - ping the gateway and client IP addresses
   - can respond to the client-side node on requests to its www, ssh, and ftp services.

4. (10 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- 192.168.0.x % x is the value of your gateway's
             IP address on the client-side network
$          sudo nmap -sU -p- 192.168.0.x % x is the value of your gateway's
             IP address on the client-side network
$          ping 8.8.8.8
$          ping 192.168.0.x % x is the value of your gateway's IP address on
             the client-side network
$          ping 10.0.0.w % w is the value of your server's IP address
```

On server VM:

```
$         sudo nmap -sT -p- 10.0.0.y % y is the value of your gateway’s IP
            address on the server-side network
$         sudo nmap -sU -p- 10.0.0.y % y is the value of your gateway’s IP
            address on the server-side network
$         ping 8.8.8.8
$         ping 10.0.0.y % y is the value of your gateway’s IP address on
            the server-side network
$         ping 192.168.0.z % z is the value of your client’s IP address
```

On gateway VM:

```
$         ping 8.8.8.8
$         ping 192.168.0.z % z is the value of your client’s IP address,
$         ping 10.0.0.w % w is the value of your server’s IP address
```

5 Related Informsation and Resource

```
IptablesHowTo https://help.ubuntu.com/community/IptablesHowTo
Iptables Tutorial 1.2.2
    https://www.frozentux.net/iptables-tutorial/iptables-tutorial.html
```