**Lab 5 (Quiz 2)**

**Outcomes**:

* Explore Firewall using iptables and Nmap
* Introduction to symmetric and asymmetric encryption

**Objectives:**

1. Setting Stateless Firewall Rules
2. Setting Stateful Firewall Rules
3. Install GPG
4. Generate Keypairs
5. Manage keys
6. Encrypt and decrypt documents
7. Sign documents

**Deliverables:**

You will not be required to submit an answer sheet for this lab report. However, you need to complete all the activities in this lab and answer all the questions. Quiz2 will evaluate your knowledge that is acquired in this lab.

Contents

[Task 1: Setting Stateless Firewall Rules 3](#_Toc98936521)

[Task 2: Setting Stateful Firewall Rules 6](#_Toc98936522)

[Task 3: Install GPG 6](#_Toc98936523)

[Task 4: Generate Keypairs 7](#_Toc98936524)

[Task 5: Manage keys 8](#_Toc98936525)

[Task 6: Encrypt and Decrypt documents 10](#_Toc98936526)

[Task 7: Sign documents 12](#_Toc98936527)

# Task 1: Setting Stateless Firewall Rules

Linux has a built-in firewall based on netfilter. This firewall is called iptables. Technically, the kernel part implementation of the firewall is called Xtables, while iptables is a user-space program to configure the firewall. However, iptables is often used to refer to both the kernel-part implementation and the user-space program.

In this task you will be using the basic firewall setting present in Alpine Docker container to Set Stateless Firewall Rules. For this purpose, you will require, two Docker containers.

1. Install Alpine Docker container:
   1. Open a first terminal (terminal 1) and run the following command:



* 1. Open a second terminal (terminal 2) and run the following command:



1. In Terminal 1, Type the below command:



Write down the IP address of your Docker container on terminal 1.

1. In Terminal 2, Type the below command:



Write down the IP address of your docker container on terminal2.

1. Ping from Terminal 1 to Terminal 2 and vice versa. You should note that the ping is successful
2. In Terminal1, Add the following rules:



1. **Explain in detail the rules that you’ve added in step 5**
2. Ping from Terminal 1 to Terminal 2 and vice versa. **What do you notice? Explain why?**
3. In terminal 1, Use the following command to view the rules added to the firewall



1. In terminal 1, type the following commands to delete the previously added firewall rules. INPUT 1
2. **What alternative command can you use to delete all the rules? (use the command iptables -h to view the manual to find the answer)**
3. In Terminal1, Add the following rule.



1. In Terminal1, add 3 additional rules that allow SSH traffic and SMTP and prevent DNS traffic (UDP Protocol). **Provide a screenshot showing your work.**
2. In Terminal 1, add the following rule and **explain it**



1. Use the following command to view the rules added to the firewall, **Provide a screenshot.**:



1. Now switch to terminal 2. and Run the command:



1. Use the below command to scan terminal 1 using Nmap, **Provide a screenshot of the output.**



* **What is nmap? What is it used for?**
* **Describe the output displayed?**
* **Explain the difference between filtered ports, closed ports and open ports?**
* **Why are all ports filtered including HTTP port while SSH and SMTP ports are closed?**

1. Switch to terminal 1 and run the following command to start ssh service on Terminal1.



1. Switch back to Ttrminal2 and run the following command to scan Terminal 1 again



* **how is the output different from the Nmap scan in step 16 (the first Nmap scan)?**

1. Go back to terminal 1 and delete all the rules.

iptables -D INPUT

1. In Terminal 1, add the following rule, **what is the purpose of this rule?** 
2. Use the below command to confirm the changes.



1. Now switch to terminal2 and type:



* **- Provide a screenshot of the output**
* **- What is your observation? And how does the output vary from the previous two Nmap scans that you have performed? And why?**

# Task 2: Setting Stateful Firewall Rules

1. In terminal 1, delete all the rules.
2. Ping from terminal 1 to terminal 2 and vice versa, the ping should be successful.
3. In Terminal 1, add the following rules:



1. Ping from terminal 1 to terminal 2 and vice versa, **what is your observation?**
2. **Explain the above rules and how they led the result that you’ve got in step 4**
3. **What is the difference between stateless firewall and stateful firewall?**

# Task 3: Install GPG

PGP (Pretty Good Privacy) is one of the most well-known public key cryptography programs. The OpenPGP format is the open cryptography standard from PGP. It is adopted by two software: GPG (free) and PGP (paid). Under Linux, the best-known distribution is GnuPG. On Windows, we can use Gpg4win.

In a public key system, each user has a key pair. GnuPG allows each user to have a pair of primary keys and zero or more pairs of secondary keys.

In this lab, we will cover some of the basic functionalities of the GnuPG software. This includes creating, exchanging key pairs, encrypting, decrypting and signing documents.

1. Start Alpine Docker image

*docker run -it --rm alpine*

1. Install the gpg package using the following command:

*apk add gnupg*

1. GPG is installed and ready to go! Check the version of the installed tool and the supported encryption algorithms by typing:

*gpg -h*

* what are the supported encryption algorithms?

# Task 4: Generate Keypairs

1. The following command is used to create new primary keypair:

*gpg --full-generate-key*

As you see from the output of the above command, GnuPG can create different types of keypairs. The primary pair is used make signature operations, whereas the second pair is used for encrypting. The different choices proposed by the gpg are the following:

Choice 1: It allows to create two RSA keypairs, the first keypair is used for signature and the second one for encryption.

Choice 2: It allows to create two keypairs, the first one is DSA used for signature and the second one is Elgamal is used for encryption.

* what are the other three choices?

1. Type *1* to select choice 1 and press Enter
2. For the Key size: The longer the key, the more resistant it will be against an attack, but it requires a longer processing time. We currently consider that 2048 bits is a minimum to have security for a few years. Type *2048* and press Enter
3. For the expiration Date: It must be chosen with care, because it is difficult to send the key with the updated date to users who already have your expired public key. For this lab, type *0* and press Enter, then type *Y* to confirm

* What does option 0 mean?
* What is a revocation certificate in gpg?

1. For the identity: Before being distributed, the key must be linked to an identity. To do this, we will attach the key pairs to an identifier (i.e. your mail address). For the purpose of this lab, you can use your personal email address if you are creating the keys on your own device and you want to adopt the keys that you create for your personal use. Insert your *name, email address, and comment if needed* then press Enter. Type *O* to confirm if all the details are correct
2. A pop-up window will appear for you to enter a Password. As a final step, you should provide a password that will be used to prove your identity when you wish to modify or use your private keys (e.g. for signing documents). For this lab, Choose an easy-to-remember phrase that you won’t forget. This password is used to generate a **symmetric** key which encrypts and decrypts the database where your private keys are stored.
3. Once you provided all the above information, gpg generates the keypairs and give you information about the identifier of your keypairs and their types (e.g. 2048 bits, RSA). - -

* Provide a screenshot your result.
* What was created in addition to the keypair?

# Task 5: Manage keys

Keys management includes all the operations that allows you to add or delete a key or revoke an existing one. We will see in this section how to achieve all the above-mentioned operations in addition to other important commands that you need to know.

1. **Listing existing keys:** You can list the existing keys using the following commands:

*gpg --list-keys*

gpg --list-secret-keys

* Provide a screenshot of the output
* What is the difference between the output of the two commands?

1. **Exporting keys:** You might have to copy your key to another machine to use the same key there, you can export your private key using the following command:

*gpg --armor --export-secret-keys InsertIdentifierEmailHere > private.key*

1. View the private key using the following command:

*cat private.key*

* Provide a screenshot of the output.

you can later import the key on another machine using the command: *gpg --import private.key*

1. **Creating keys:** You can add different sub keypairs that can be used for different purposes. For example, one sub keypair can be used for signing business documents while the other keypair for signing personal documents. To add more sub keypairs, follow these steps:
   1. Enter the editing mode of the primary key using this command:

*gpg --edit-key InsertIdentifierEmailHere*

* 1. Enter the command:

*addkey*

* 1. Specify the type of the key, select: RSA (encrypt only)
  2. Specify the key size: 2048
  3. Specify the expiration of the key: 0, and type y to confirm
  4. Enter your password
  5. Use CTRL + C to exit editing mode
* Provide a screenshot of the output

1. **Deleting keys:** To delete an existing key,you should execute the following commands:
   1. Enter in the editing mode of the primary key using this command:

*gpg --edit-key InsertIdentifierEmailHere*

* 1. Select the wanted key using the command:

*key N*

where N represents the order of the key and can take values starting from 0.

* 1. Enter the command, and type y to confirm:

*delkey*

* 1. Enter CTRL+C to exit editing mode

1. **Revoking keys*:*** Sometimes, our private keys can be lost (e.g. your hard disk is crashed or stolen, or you may simply forget your passphrase). In this case, we should inform our correspondents as quickly as possible by revoking our key and sending them the revocation certificate. In order to revoke a key, you need to know the passphrase of the private key that you want to revoke, and you need to have an access to the private key. Follow the below steps to revoke a key:
   1. Enter the edit mode:

*gpg --edit-key InsertIdentifierEmailHere*

* 1. Select the key to revoke using the below command, where N is the order number of the key, starting from 0:

*key N*

* 1. Revoke the key using the below command:

*Revkey*

* 1. Select a reason for revocation and confirm.
  2. Press CTRL + C to exit editing mode

1. **Distributing keys*:*** In an ideal scenario, we should distribute our keys by giving them personally to our correspondents. However, the PGP keys can be distributed by email, or by other electronic means of communication. In addition to email, one of the most known distribution methods is to post your public key to central public key servers. Use the following command to distribute your key:
   1. use the command: *gpg --list-keys*
   2. copy your public key id that is shown in the screenshot below

***A picture containing text

Description automatically generated***

* 1. send your public key:

*gpg --keyserver keys.gnupg.net --send-key InsertPublicKeyID*

* 1. using your browser, open hkp://pgp.surf.nl and search for your key using your email address
* provide a screenshot of the output

# Task 6: Encrypt and Decrypt documents

You can encrypt and decrypt documents using the following commands:

1. First, we will encrypt a document using symmetric encryption:
   1. create the document that is going to be encrypted

*touch doc1*

* 1. verify that the document has been created

*ls*

* 1. Insert text into the document

*echo This is the first secret message! > doc1*

* 1. View the content of the document

*cat doc1*

* 1. Encrypt using symmetric encryption, enter your password if needed:

*gpg --symmetric doc1*

* 1. Use the command *ls* to view the output file, which should be titled *doc1.gpg*
  2. View the contents of the encrypted file using the below command

*cat doc1.gpg*

* If you want to send the above encrypted message to a friend. What does your friend need to decrypt the file? what is the problem that rises when using symmetric encryption?

1. Next, encrypt using asymmetric encryption. You can encrypt a message for a single specific recipient. You do this by encrypting asymmetrically with the recipient’s public key. By doing this, only the recipient's private key will decrypt the message. You will need the recipient's public key in order to do this. They can share their public key with you directly, or you can search public key servers. For this exercise you will encrypt using your own public key.
2. create the document that is going to be encrypted

*touch doc2*

1. verify that the document has been created

*ls*

1. Insert text into the document

*echo This is the second secret message! > doc2*

1. View the content of the document

*cat doc2*

1. Encrypt using asymmetric encryption:

*gpg --output encrypted --encrypt --recipient InsertIdentifierEmailHere doc2*

*For example:*

*gpg --output encrypted --encrypt --recipient liza.ahmd@gmail.com doc2*

1. Use the command *ls* to view the output file.
2. View the contents of the encrypted file using the below command.

*cat encrypted*

1. Now you will decrypt the file using the following command:

*gpg --output decrypted --decrypt encrypted*

1. Use the command *ls* to view the output file.
2. View the contents of the encrypted file using the below command.

*cat decrypted*

* If Alice wants to send a message to bob using asymmetric cryptography. Which key should she use to encrypt the message? And which key will be used to decrypt the message?
* What are the advantages and disadvantages of symmetric cryptography?
* What are the advantages and disadvantages of asymmetric cryptography?

# Task 7: Sign documents

You can use the following commands in order to sign and verify the signature of a documents.

1. Generating the signature:

*gpg --output signed --sign doc2*

1. Verifying the signature

gpg --verify signed

* What is the purpose of digital signatures?