# Computer Security 2020

## Lab 3: Network security

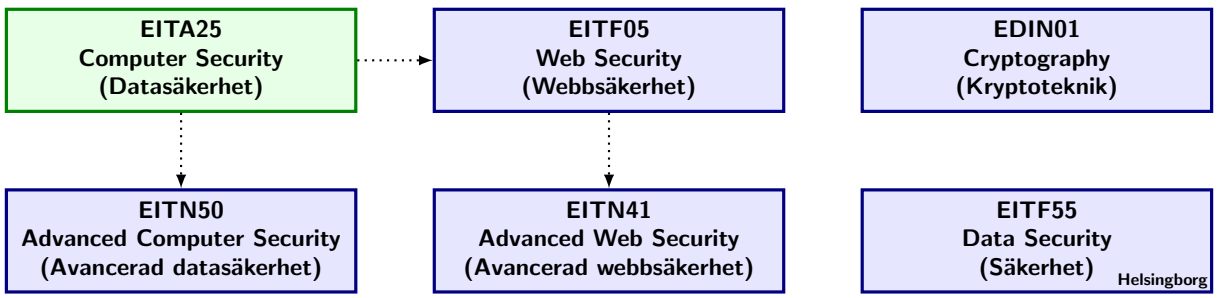
- This lab will be done in groups of 2 people.
- There are preparatory assignments for this lab, read through the complete lab guide carefully, and bring your written answers to the lab.
- During the lab, write down answers to all problems on a sheet of paper so your work can be approved.

---

### Learning goals:

- Learn how to use GnuPG for encrypting files and mail.
- Scan a network using nmap.
- Sniffing data and analyzing traffic on a local network using Wireshark.
- Working with Iptables.

---

## Read this earlier than one day before the lab!

There are preparatory assignments for this lab. For most students, these assignments take more than a couple of minutes. Read through this lab guide, and then prepare your assignments. During the lab, answer all Problems on a separate sheet of paper.

*Note that you will not have any internet access during the lab, so come prepared.* You may bring as many books and printed materials as you can carry. Study the questions in this lab manual, consider what you will need to be able to solve them, and make sure you bring that information with you. Alternatively, if you feel confident in the availability of eduroam, you may bring your own laptop, smartphone, or tablet to get Internet access.

# 1 GNU Privacy Guard

[Estimated time for this part: 1.5–2 hours]

In this part of the laboratory we will learn how to use cryptography to send encrypted emails, we will learn how to create signatures, and we will learn how to encrypt files using the free replacement of PGP, called GNU Privacy Guard (GPG). To understand the laboratory it is essential that you understand some basic cryptography.

> **Preparatory assignment 1**
>
> - Study the Gnu Privacy Handbook, available at `http://www.gnupg.org/gph/en/manual.html`. Do not go into any details, but make sure that you have a feeling for what it does and how to work with GPG. The handbook will be available at the lab computers.
> - Make sure you understand the concept of public key cryptography, symmetric cryptography, hash functions and signatures.
> - In the handbook, look up the commands to: create keypairs; generate key revocation certificates; import, export, and get fingerprints of public keys; and encrypt, decrypt, sign, and verify files. You don't need to remember them, but you should know where to look for them.

It is possible to operate GPG via the command-line, or by using some graphical front end for GnuPG. For this lab though, you should only use the command-line. The graphical programs does not always provide the more advanced features needed for some of the tasks you will perform.

If you need to study the Gnu Privacy Handbook *during the lab*, it can be viewed in the browser on the lab computers using the address `http://www.local.lab/gpgmanual.htm`.

## 1.1 Create keypair

Begin by generating a new keypair, use **lina#** as real name and **lina#@mail.local.lab** as email adress. Make sure you remember the password chosen since this is needed for all access to the private key. Write the public key to a file using the operation `export`, both with and without the option `armor`, compare the results.

**Problem 1** *Which public key algorithm do you use for signatures and encryption?*

**Problem 2** *What does the option* `armor` *do?*

**Preparatory assignment 2**

- What is a key revocation certificate?
- Why is it a good idea to have a revocation certificate?

**Problem 3** *Create a key revocation certificate and store this in a secure place, i.e., only you should be able to read it.*

## 1.2 Encrypt and sign files

Copy `/mnt/server.local.lab/eita25/lab3/encrypt_me.txt` to your local home directory. Look at what this file contains. Encrypt this file with the option `encrypt` using your public key.

**Problem 4**

*Why is the encrypted file much smaller than the original file?*

*Is the text readable?*

*Which types of* ***algorithms*** *are used when encrypting the file?* Hint: *Read about* Hybrid ciphers *in the Gnu Privacy Handbook.*

Decrypt the file again and compare the result with the original file. Are the files identical? The next step will be to also create a signature of the file.

**Problem 5** *Sign the file* `encrypt_me.txt`*. Then verify the signature using the option* `verify`*. What command did you use to create the signature?*

**Problem 6** *What different ways are there to create signatures in GPG? How do they differ?*

If you have trouble with the problem below, go back to the previous question and consider the different ways to create signatures. You may need to use another method.

**Problem 7** *Change one letter in the signed document and then verify the signature again. Is the signature valid?*

## 1.3 Configure Evolution (mail agent) to use your GPG key

Start the Evolution mail agent (Applications ⇒ Office ⇒ Evolution Mail and Calendar). Use the following settings for the mail agent:

**Identity**
Full name: lina#
E-mail address: lina#@mail.local.lab

**Receiving mail**
Server type: POP
Server: mail.local.lab
Username: lina#

**Sending mail**
Server type: SMTP
Server: mail.local.lab
Server requires authentication
Username: lina#

Options not mentioned above should be left as the default values.

Now we can send and receive emails, but we would also like to configure Evolution to use GPG. Open the account editor (Edit ⇒ Preferences ⇒ "Your account" ⇒ Edit ⇒ Security). In the field for the GPG key, assign one of the identifiers for your key, e.g., `lina#@mail.local.lab`.

## 1.4 Signing keys and sending encrypted and signed emails

A very important part of public key cryptography is the distribution and integrity of public keys. The core of key management in GPG is the notion of signing keys. Signing keys permits you to detect tampering on your keyring and it allows you to certify that the key belongs to the person named by a user ID on the key.

The signatures of the user IDs can be checked with the command **check** from the key edit menu in the command-line tool.

Before you sign someone else's key it is important to check the authenticity of the key, to do this use the key's fingerprint. Compare the fingerprint of the received key with the sent one, this may be done in person or over the phone or through any other means as long as you can guarantee that you are communicating with the key's true owner. If the fingerprint you get is the same as the fingerprint the key's owner gets, then you with high certainty know that you have a correct copy of the key. After checking the fingerprint, you may sign the key to validate it. Since key verification is a weak point in public-key cryptography, you should be extremely careful and always check a key's fingerprint with the owner before signing the key.

You should now try sending a couple of encrypted and signed emails to some other group. Start by exchanging public keys with your neighbour using for example mail, scp, ftp etc. If you use the `edit-key` option to sign the key, don't forget to `save` after signing!

> **Problem 8** *Import your neighbours public key into your keyring and sign it using your private key. Note that signing a key is not the same thing as signing a file.*

> **Problem 9** *Try sending some mails to each other using both signatures and encryption and verify that it works. You do this by clicking the appropriate option in the* Options*-menu when you are composing your e-mail.*

> **Problem 10** *Can you read the encrypted mails you sent? Why/why not?*

## 1.5 Web of trust

Since it is impossible to have a certification authority who checks the integrity and authenticity of all keys used in GPG, some other method must be used. GPG/PGP uses something call the web of trust.

> **Preparatory assignment 3**
>
> - What is the **web of trust** and how does it work? *Note! This term is also used in some other contexts, ensure you read about the web of trust related to PGP/GnuPG!*
> - What are the different trust levels in the web of trust?
> - Look up the commands used to set trust on keys in GnuPG.
> - What is the difference between trust and validity?

In the folder `/mnt/server.local.lab/eita25/lab3/gpg` you can find a number of public keys. Import all of these keys into your keyring. The web should be organized as in the figure below. You have three friends you trust, **Adam**, **Berit** and **Caroline**, so you should sign their public keys with your key. They on the other hand have another friend, **David**, who they trust and whose key they have signed, you do not know **David** at all, and thus you should *not* sign his key.


Depending on how much you trust your friends **Adam**, **Berit** and **Caroline**, GPG validates **David's** key differently.

**Problem 11** *Consider the four cases in the table below (one for each row). For each row, set the trust in your friends according to the row, and fill in the corresponding validity of* ***David's*** *public key.*

| *trust* | | | *validity of* |
|---|---|---|---|
| *Adam* | *Berit* | *Caroline* | *David's key* |
| *marginal* | *do not trust* | *do not trust* | |
| *full* | *do not trust* | *do not trust* | |
| *marginal* | *marginal* | *do not trust* | |
| *marginal* | *marginal* | *marginal* | |

Your are now finished with part one of the laboratory, check your answers.

## 2 Scanning the network

[Estimated time for this part: 0.5 hour]

A network attack is likely to require the IP address of the target. Moreover, in order to take advantage of known vulnerabilities in servers we need to know if a particular computer is running the server and also, if a nonstandard port is used, the port number the server is running on. Known vulnerabilities are often applicable only to some specific versions of the server and it is favourable for the attacker if he knows the exact version that is running. There are several tools that can be used to gather all this information and **Nmap** is the most used and well-known of them. Naturally, a network scanning tool is not only used by attackers. A network administrator can also use it to get information about the network, e.g., if the firewall functions as it is supposed to or if there are users, perhaps unknowlingly, running vulnerable services.

> **Preparatory assignment 4**
>
> - Read about Nmap so that you feel comfortable working with it. A good online resource is `http://nmap.org`. Read the lab problems below, and figure out what commands you need to use during the lab.

Now we are going to use nmap to scan the local network. *Note that while you are shielded from the Internet in this lab, using Nmap on a network without the permission from the administrator may get you in trouble.* For more information about legal issues, read `http://nmap.org/book/legal-issues.html` if you are interested.

**Problem 12** *Determine, using a ping scan, which hosts are online on the local network.*

**Problem 13** *Do a port scan for the hosts that are online. (You can use the option* `-T4` *if the port scan seems slow). What services are running on the different hosts? Give some examples from each host, you don't have to write down all output.*

**Problem 14** *Find out, using OS detection, what operating system is used on the computer* `crackme.local.lab`*.*

**Problem 15** *Which ftp-server and which version of it is used on the computer* `crackme.local.lab`*.*

## 3 Network sniffing

[Estimated time for this part: 0.5 hour]

For some reasons, that can be best explained by computational complexity and speed, it seems that not all data traffic on a network is secured by some form of crypto-system. It is also a fact that protocols like rlogin, telnet, FTP, POP3, SMTP, IMAP etc. are sometimes used in their infancy form in order to be backward compatible. This means, for example, that the mentioned protocols may send passwords as plain text if you are not careful.

> **Preparatory assignment 5**
>
> - Get acquainted with POP3, SMTP, telnet, FTP and SSH. You do not have to learn any details about the protocols, but you should know what they are and in which context they are used.

We will, in this laboratory assignment, use the tool Wireshark to study the data traffic on the local network provided during the lab. Wireshark is mainly a tool for the system or network

administrator to check and analyze the network traffic. But for our purposes it also serves well as a tool to obtain password or any other confidential information sent in plain text on the network.

The network in the laboratory is a hubbed network. When a hub receives a packet of data at one of its ports from a PC on the network, it transmits the packet to all of its ports and, thus, to all of the other PCs on the network. This is similar to how traffic is sent on a wireless network. This makes it easy for a computer on the network to listen to the other computers. It is also possible to sniff data on a network connected via a switch, although it is a little more difficult (this will not be tested in the laboratory since one has to poison the ARP table of the victim and the router to perform the attack. 12 computers all trying to poison each other at the same time is a bad idea).

**Preparatory assignment 6**

- Get acquainted with Wireshark. Understand how to enter filter strings. Information can e.g., be found at `http://wiki.wireshark.org`.

**Problem 16** *Start Wireshark with by entering the command* `sudo wireshark`*, in the terminal, and start getting acquainted with the program. Visit a few web pages e.g.,* `http://www.local.lab` *or the lab queue, and analyze the traffic. Try a few different filters, e.g.* `http` *to only see HTTP traffic.*

**Problem 17** *Update your mailbox created in the first section and analyze the traffic. What can you say about the POP3 protocol?*

Now we will log into the computer `crackme.local.lab`. Recall that the username is `linaXX` and the password is `Kanejbytas123`.

**Problem 18** *Log into the computer* `crackme.local.lab` *by using FTP, SSH and telnet. If possible, let your neighbour log in and study the traffic between their computer and* `crackme`*. What can you say about the traffic?*

In recent years, the use of TLS has increased significantly, and a large fraction of today's internet traffic is now encrypted. It is, however, important to note that there may still be metadata that is not encrypted.

**Problem 19** *Visit the site* `https://imaginar.se`*, which uses TLS, and look at the traffic in Wireshark. Look for the TLS Client Hello message. Can an eavesdropper know which site you are visiting? Can you think of a situation when this is undesirable?*

# 4 Iptables

Iptables is a powerful user-space utility program that allows a system administrator to configure the tables provided by the Linux kernel firewall and the chains and rules it stores. Users can use iptables to block a specific ip-address, allow incoming SSH only from a specific network or load balancing incoming web traffic, etc. The iptables firewall works by interacting with the packet filtering hooks in the Linux kernels networking stack. These kernel hooks are known as the netfilter framework. When a packet enters networking system (incoming or outgoing), it will trigger these hooks, which enables iptables to interact with the traffic and apply the corresponding rules.

Iptables contains five tables: *raw*, *filter*, *nat*, *mangle* and *security*. In most common cases, only *filter* and *nat* are generally used. *filter* is the default table and is one of the most widely used tables. For example, if the rule is used to decide whether to allow the packet to continue to its intended destination or to deny its request, it would probably be added to the *filter* table. *nat* is used for network address translation. For example, if a rule is used for port forwarding, it will be added into the *nat* table.

The chains are lists of rules which are followed in order. Each table has multiple chains. There are five predefined chains: PREROUTING, INPUT, FORWARD, OUTPUT and POSTROUTING, each of which is triggered by one of the netfilter hooks. Users can also define their own chains as desired.

The rules are placed within a specific chain of a specific table. When each chain is called, the packet in question will be checked against each rule within the chain in order. Rules are specified by multiple matches and one target. Matches are the conditions the packet must satisfy so that the rule can be applied. And target is the action taken when the packet matches all conditions.

> **Preparatory assignment 7**
>
> - Read about firewall and iptables and different rule settings such as drop, accept and different options that can be applied to iptables rules and their usages.

In this assignment you will play around with some basic iptables commands. First, list the available iptables rules:

```
sudo iptables -L
```

Check the IP addresses and the interface you have:

```
ifconfig
```

Your IP should start with 192.168. Remember your IP address and also ask the IP address of your neighbor. Try to drop the incoming traffic from your neighbor by adding this rule to your iptables:

```
sudo iptables -A INPUT -s [your neighbor ip] -j DROP
```

Type the linux password which is eita25. Then list the iptables rules again and make sure the rule was added successfully.

> **Problem 20** *Now ask your neighbor to ping your IP address. Are they able to ping your computer why or why not?*

> **Problem 21** *Try to ping your neighbor, are you able to ping them? why or why not?*

Now try to drop also outgoing traffic to your neighbor by the command bellow:

```
sudo iptables -A OUTPUT -d [your neighbor ip] -j DROP
```

Again list the iptables rules and make sure the rule was added successfully, now you should have two rules in the iptables.

> **Problem 22** *Does the new rule added in the beginning of the list or end of the list? what does* `-A` *option do? what if you use* `-I` *instead of* `-A`*? Would the order be different?*

Now allow the outgoing traffic to your neighbor by adding the command below:

```
sudo iptables -A OUTPUT -d [your neighbor ip] -j ACCEPT
```

> **Problem 23** *Now you have both drop and accept rules for your neighbor for outgoing traffic. Are you able to ping your neighbor or it will be dropped? How about if the accept rule was before the drop rule in the list?*

You have now finished this lab! Present your solutions for the lab assistant. In the meantime, you can perform the cleanup operations of the next section. However, if you have not yet presented your answers on the GPG part, please do not delete those files just yet.

# 5 Cleanup

Remove all added rules by the command below:

```
sudo iptables -F
```

List iptables rules and make sure every added rule was deleted successfully.