UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Escola Tècnica Superior d'Enginyeria
de Telecomunicació de Barcelona

telecos
BCN

Analysis and Implementation of a Security Standard

Master Thesis
submitted to the Faculty of the
Escola Tècnica d'Enginyeria de Telecomunicació de Barcelona
Universitat Politècnica de Catalunya
by

David Hernández Vilalta

In partial fulfillment
of the requirements for the
Master's Degree in Cybersecurity

Advisor: Juan Hernandez Serrano
Barcelona, July 1, 2021

telecos
BCN

Contents

List of Figures

Revision history and approval record

Revision	Date	Purpose
0	01/04/2021	Document creation
1	Weekly	Document revision with David Reguera
2	Monthly	Technical revision with Juan Hernandez
3	30/06/2021	Final revision with David Reguera
4	30/06/2021	Final revision with Juan Hernandez
5	2/07/2021	Document delivery

DOCUMENT DISTRIBUTION LIST

Name	e-mail
David Hernandez Vilalta	david.hernandez.vilalta@estudiantat.upc.edu
Juan Hernandez Serrano	j.hernandez@upc.edu
David Reguera Lopez	dreguera@ub.edu

Written by:		Reviewed and approved by:	
Date	July 1, 2021	Date	July 2, 2021
Name	David Hernandez Vilalta	Name	Juan Hernandez Serrano
Position	Project Author	Position	Project Supervisor

Abstract

This master's thesis describes the design and implementation of a security standard in a university research department. It has been developed in the framework of the ETSETB Master's Degree in Cybersecurity, in cooperation with the University of Barcelona.

The work has consisted on several stages. First, an analysis of the vulnerabilities of the system has been performed. This diagnosis has been specially important, since the lack of cybersecurity protections in the department has lead to several hijacks and data losses throughout the years. Then, the report describes the application of all the security features that are considered essential in a company, covering as much elements as possible. Those include from devices' physical security, through software protection to employees training. The project will be mainly focused in the deployment of the main services found in an IT department with a brief cybersecurity training session for the employees at the end.

The work developed in this master thesis will reinforce the security of all crucial services and will reduce the possibility of data loss.

1 Introduction

Nowadays cybersecurity is getting more relevance than ever due to the pandemic situation where a significant percentage of the population is working remotely. Furthermore, cyber-attacks have increased during the last year for the very same reason and this is something that we must take care of and be as most forewarned as possible. The aim of this project is to protect a company from all kind of cyberattacks securing both people and machines.

The main objective is to take a currently insecure department of scientific research and achieve a standardised securitization at all levels, including infrastructure, software and of course employees. Some already defined standards will be the pathway to follow such as the ones defined in the ISO 27000-series as well as other non standardised protocols and practices such as the ones described by OWASP.

This thesis will show some methods of achieving an acceptable level of security on the most relevant elements in a company. It will be done adhering to the current security standards and guidelines or at least guaranteeing the minimum level required by them. The infrastructure security will not be detailed since it is not possible to implement significant changes at the moment. Nevertheless, some suggestions will be pointed out for the cases that apply. As for software security, the document will detail the addition of security features for every service required by the department. Finally, regarding the employees, a training session will be developed in which the employees will learn the most relevant security concepts to keep themselves and the company as much secure as possible in front of all the potential threats out there.

The requirements for this project are quite cheap, giving the fact that most of the implementations are software-based. In fact, this thesis could be carried out using the existing computational resources of the company (i.e. a single ten year-old server). Nonetheless, the company has decided to take a step forward and invest a little bit more in cybersecurity, giving this project the chance of acquiring a new server machine for this specific purpose.

This work will be carried out from scratch, built on top of a completely clean machine which won't even come with a pre-installed operative system. Every single element mentioned in this master thesis is going to be designed and implemented here or thereafter, if applicable. Several applications and software tools developed by other authors will be used throughout the project for most solutions excepting some particular ones which will be developed specifically for this master thesis.

The rest of the document is organised as follows:

1. **State of the art**: This section will point out some of the main security improvements of the recent years and some of the existing standards that describe their implementations.
2. **Use case**: Here is the explanation of the methodology that will be followed along the work and a description of the scenario of this thesis in which the current system and its security flaws are detailed.

3. **Implementations**: This is the main section of the project. It shows all the procedures that have been executed and their correspondent justifications.
4. **Conclusion and Future development**: This is the final section in which the work that has been done is summarised together with the things that remain to be done and a final personal evaluation of the project is also expressed.

1.1 Planned dedication

This project has been developed in the Department of Condensed Matter Physics at the University of Barcelona. The dedication has been adjusted to the limitations of the contract as part-time system administrator. The actual dedication to the project will be following a daily basis, in which around 2.5 hours will be entirely devoted to its development every day in order to still have time enough -2.5 more hours- to carry out the tasks as the department's system administrator. Given the fact that the project has began April 1st -for internal affairs- and that the final delivery is due to July 2nd, means that by the end of term the total hours dedicated to the project will amount to roughly 14 weeks * 5 days per week * (2.5 hours per day + 2 daily hours at home) = 315 hours, which is greater than the expected (12 ECTS * 25 hours per ECTS = 300 hours for the thesis).

1.2 Gantt Diagram

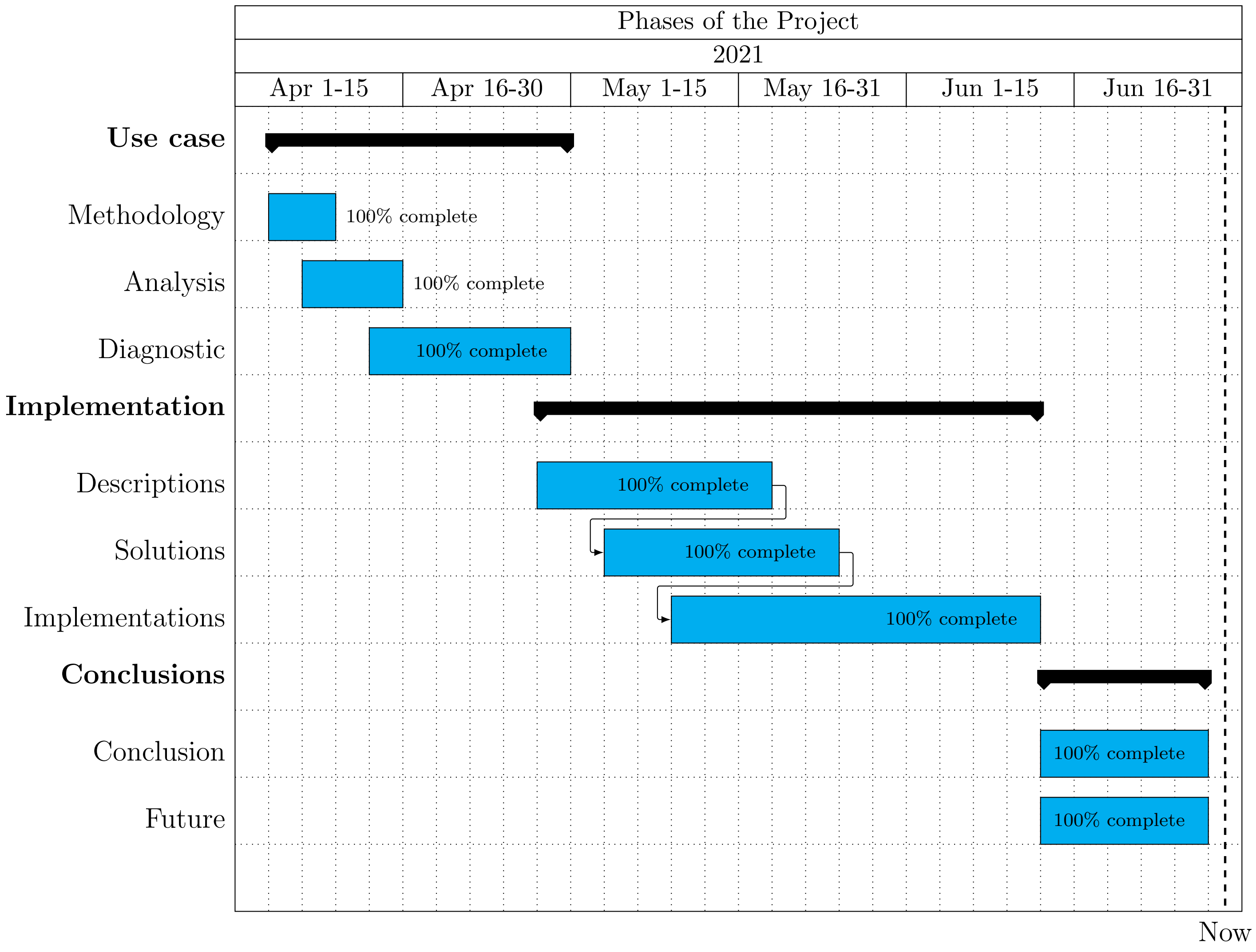


Figure 1: Gantt diagram of the project

2 State of the art of the technology used or applied in this thesis

Along the years, the digital world has gone through all kind of threats and attacks that have lead to the constant improving security systems that have turned into the algorithms and protocols that protect us all today. Some recent relevant improvements that have been done to palliate those menaces are the ones listed below:

- **Multi-factor authentication (MFA)**
- **Network Behavioural Analysis (NBA)**: identifying malicious files based on behavioural deviations or anomalies.
- **Threat intelligence and update automation**
- **Real-time protection**: also referred to as on-access scanning, background guard, resident shield and auto-protect.
- **Sandboxing**: creating an isolated test environment where you can execute a suspicious file or URL.
- **Forensics**: replaying attacks to help security teams better mitigate future breaches.
- **Back-up and mirroring**
- **Web application firewalls (WAF)**: protecting against cross-site forgery, cross-site-scripting (XSS), file inclusion, and SQL injection.

All of the mentioned above -and so many more- are included in the information technology security standards ISO/IEC 27000-series.

2.1 Topic

This project will be mainly based on the suggestions found in the ISO 27000-series documents, which comprise information security standards that provide best practice recommendations on information security management and the management of information risks through information security controls. More precisely, the standards in which this thesis will be focused are the 27001, which includes security techniques, information security management systems and requirements [1]; the 27003, including information security management system implementation guidance; the 27004, focused on information security management, monitoring, measurement, analysis and evaluation; the 27005, which is dedicated to information security risk management; and the 27007, which provides guidelines for information security management systems auditing.

Of course, in the field of information security there's never a perfect system in which to trust, and as a consequence there is always a need of further improvement. Therefore, this thesis will include some ideas that could be carried out in order to improve the currently existing mechanisms.

3 Use case

In this section is going to be analysed all the elements involved in the security of the department's information and their more than probable security flaws. From all the services that are currently running and offering employees a functional system, to the physical security of all the relevant devices and machines.

Despite that, some not-so-important services/devices will be ignored in this project since they would require an extra effort and time that is beyond this project's scope. Nevertheless, it is worth to emphasise that those elements left behind here should not be ignored in a fully system security deployment.

Additionally, a brief diagnostic will be performed on the previously mentioned elements just to superficially see how a cyber attack could (or not) be executed on the department.

3.1 Methodology

All implementations will be first developed and tested under the practicality and simplicity of a docker container. This will allow for making mistakes and even breaking the system without worrying about it while trying to install and configure each service.

The operating system chosen for all the required services to run on is Ubuntu Server. This OS will be deployed on a new server mentioned in the following sections.

All the decisions and steps taken in this project will be focused on security, without forgetting functionality. This means that in some cases the chosen solution may not be the one which performs best neither in speed nor in practicality. The idea is to find a balance in which each service can operate fairly similar as it would without the security added features.

This project will be developed upon some concepts of security found on several standards such as ISO/IEC 27000-series [2] or other non standardised rules such as the ones described by OWASP [3]. This doesn't mean that whatever those directives say has to be blindly followed since the great diversity of possible scenarios require specific implementations that don't necessarily need to match the standardised ones. Besides, the suggestions contributed by those entities will always be taken into account.

The aim of this project does not end with systems' security itself, it also cares for users' privacy as well. Some of the implemented measures do not provide any layer of security but the employees' privacy within the Internet.

3.2 Analysis

The Department of Condensed Matter Physics (FMC from now on) of the University of Barcelona (UB from now on) is a very particular department in terms of IT compared with the other ones in the UB. It has their own IT systems and services which are also managed by an engineer of the department (i.e. myself). Nevertheless, the firewall restrictions are managed by the UB IT department, so this part of the implementation will be assumed to be already secured. This scenario requires for those services to be maintained, a task which has been quite forgotten throughout the years. As detailed below, the FMC has several computer services that allow for the department's employees to work in full conditions. Recently, the department FMC was created in 2016 in which two former departments was unified in a new one, which has lead to the obsolescence of some of those services and other non-migrated ones from the old departments. Currently, the department has around 100 employees and it is focused on research tasks and teaching, and they use the department's resources mainly for the former one. All of them will be analysed in this section.

The company's infrastructure at the moment of starting this project is the following:

1. A tower server installed in 2008 with a 2 GHz CPU, 2 GB of RAM, an HDD of 300 GB and another one of 160 GB, running Debian 4.3.5-4. This machine runs several services that are under the load of more than 100 users. Those services are presented in the following list:

   - **DNS**: Bind 9.7.3 — Using old version publicly and HMAC-MD5 for TSIG, which is considered insecure nowadays. This service is used for name resolving.
   - **LDAP**: OpenLDAP 2.4.23 (Jan 12 2013) — Old version, which means it is probably plenty of vulnerabilities. Currently there are 2 LDAPs, one for the old department and one for FMC that should be unified. This service is used to enable authentication to other IT resources on the server for the users of FMC.
   - **Web**: Apache2 Server version 2.2.16 — Old version, old build (2014). Apache is one of the most widely used web servers, and as a consequence it has lots of crackers trying to find new vulnerabilities every day. Here's a list provided by the developers in which some known vulnerabilities that affect this version are shown: Apache 2.2.16 vulnerabilities. This web service acts as a host for the department's web page and for some of the research groups.
   - **Database**: MySQL 5.0.51a — Old version, which leads to the same conclusion as the previous service. A list of vulnerabilities can be found here. The database is used mainly for the web pages and some of the other server internal resources.
   - **Backup**: Configured to work alongside with a NAS. It has been stopped since 2019 for unknown reasons and it wasn't even configured to backup the database.
   - **FTP**: ProFTPD Version 1.3.3a — Old version, employees no longer use this service and therefore it will not be used in the new implementation.

- **Mail**: Exim version 4.72, built 13-Jul-2014 — Old build, the FMC used to have an internal mail service, which has been replaced by the institutional one so it won't be needed anymore.
- **Printing**: CUPS 1.4.4 — Ancient version, as well as the mail service, the department once had its own printing system, but it was replaced by a centralised one managed by a third party a couple of years ago and this service hasn't been used since then. Therefore, it will not be necessary in the new deployment.
- **WiFi**: FreeRadius 2.1.12, 10 years old version for a service that has been pretty much unused since its deployment due to the birth of *eduroam* which granted with Internet access the whole campus, and as a consequence nobody used the FMC's own WiFi. Therefore, it won't be included in the new implementation.

2. Another tower server (running Ubuntu 9.10, and its vulnerabilities) that is in charge of the DHCP service of the whole department and runs a virtual machine that acts as a VPN for the proper access to the internal network of the company from the outside Internet.
    - **DHCP**: ISC-DHCP-Server 4.2.2 — Old version, some of its vulnerabilities can be found here. This service is used for automatic IP and DNS Server assignment for users in the FMC network.
3. A high performance computing racked cluster which consists of 40 nodes in which one of them is used as master and another is used as a backup. The specification of these nodes is described below:
    - Server: 2 x CPU AMD Opteron 2350 (4c) @ 2.00GHz — RAM 16 GB
    - 20 Nodes: 2 x CPU Intel Xeon E5-2650 (8 core) @ 2.00GHz — RAM 32 GB
    - 4 nodes: 2 x CPU Intel Xeon E5-2670 v2 (8 c) @ 2.50GHz — RAM 40 GB
    - 7 nodes: 2 x CPU Intel Xeon E5-2660 v3 (10 c) @ 2.60GHz — RAM 48 GB
    - 1 node: 2 x CPU Intel Xeon E5-2697 v4 (18 c) @ 2.30GHz — RAM 72 GB
    - 7 nodes: 2 x CPU Intel Xeon Gold 6130 (16 c) @ 2.10GHz — RAM 80 GB
4. A cluster of 9 tower machines equipped with powerful GPU graphics cards to run heavy simulations, described below:
    - 1: Intel Core i7 870@2.93GHz (4cores) — Nvidia TESLA / Nvidia GeForce GTX 460
    - 2: Intel Core i7 870@2.93GHz (4cores) — 2x Nvidia GeForce GTX 460
    - 3: Intel Core i7 4790@3.60GHz (4 cores) — Nvidia GeForce GTX 780
    - 4: Intel Core i7 4790@3.60GHz (4 cores) — Nvidia GeForce GTX 980
    - 5: Intel Core i7 870@2.93GHz (4cores) — 2x Nvidia GeForce GTX 460
    - 6: Intel Core i7 7700@3.6 GHZ (4 cores) — 2x Nvidia GeForce GTX 1060

- 7: Intel Xeon W-2155@3.3GHz (10 cores) — 4x Nvidia RTX 2080 Ti
- 8: Intel Xeon W-2155@3.3GHz (10 cores) — 4x Nvidia RTX 2080 Ti
- 9: Intel Xeon W-2155@3.3GHz (10 cores) — 3x Nvidia RTX 2080 Ti

5. A network attached storage (NAS) with 50 TB of disk space, an Intel Xeon Silver CPU and 48 GB of RAM.

3.3 Diagnostic

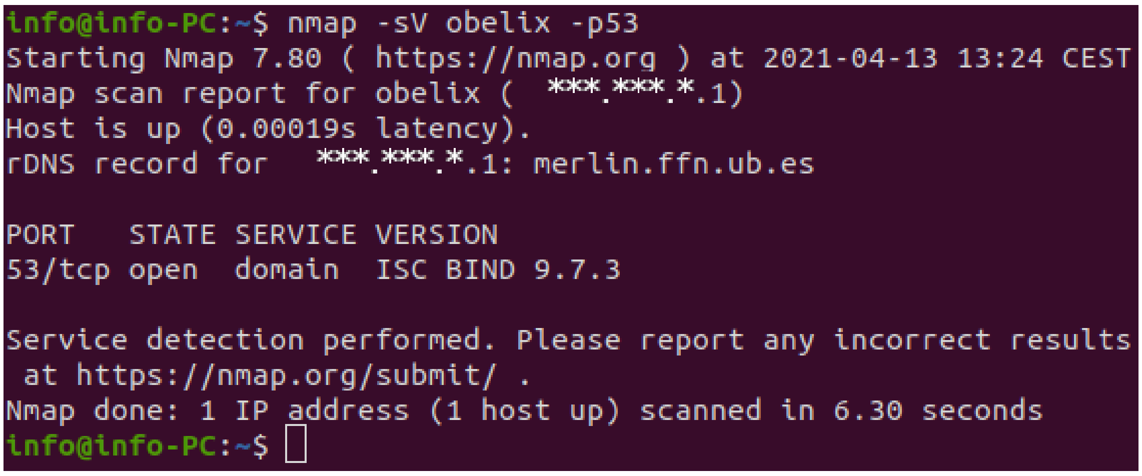
This section will expose all the current vulnerabilities of each service described in the previous section, as well as a small analysis and a proper possible solution for each one.

- DNS Bind 9.7.3: This is an old version of Bind which has 31 vulnerabilities reported by cvedetails.com/BIND-9.7.3_Vulnerabilities. Fortunately, none of the vulnerabilities described there have an impact on the system such that its exploiting would result in a total information disclosure, revealing all system files. The great majority are mainly preventing the service from running normally, also known as a denial of service (or how it is known in the cybersecurity slang, DoS). Despite this, there is one vulnerability that compromises all system files by gaining system-level permissions through privileges escalation exploiting a flaw present on the Windows installer, thus not affecting the system analysed here.

  The previous paragraph was just regarding the software itself which the system administrator has no control of (or at least not enough to be considered their fault). The following part of the analysis will be focused on the service configuration on the system, which in this case is completely under the administrator's control.

  First of all, BIND offers the configuration of the Transaction SIGnatures (TSIG) based transaction security to authenticate updates to the DNS server. This configuration file can be found at */etc/bind/rndc.key* and it uses an HMAC-MD5 key which is 128 bits long, nowadays considered insecure due to the fact that the average probability of finding a collision is half the key length, i.e. 1 over 2^{64}. Although the security of the MD5 hash function itself is severely compromised, the currently known attacks on HMAC-MD5 do not seem to indicate a practical vulnerability when used as a message authentication code, but for a new protocol design, a cipher suite with HMAC-MD5 should not be used.

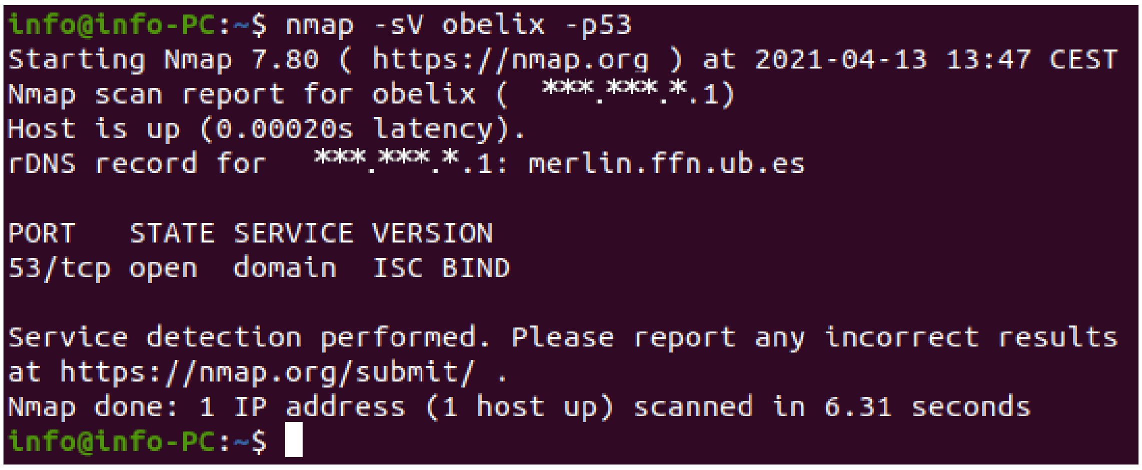
  On the second hand, there's an option that makes it more difficult for a potential attacker to exploit a vulnerability of the service that is to hide the version in use. Bind 9.7.3 has an specific configuration file (*/etc/bind/named.conf.options*) in which the administrator should explicitly write the willingness to resist as much as possible to a version scanning tool, such as *nmap*. The figure below shows the current difficulty of extracting the version in use:



```
info@info-PC:~$ nmap -sV obelix -p53
Starting Nmap 7.80 ( https://nmap.org ) at 2021-04-13 13:24 CEST
Nmap scan report for obelix (  ***.***.*.1)
Host is up (0.00019s latency).
rDNS record for   ***.***.*.1: merlin.ffn.ub.es

PORT   STATE SERVICE VERSION
53/tcp open  domain  ISC BIND 9.7.3

Service detection performed. Please report any incorrect results
 at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 6.30 seconds
info@info-PC:~$ 
```

Figure 2: Nmap Scan of the DNS version on the analysed machine.

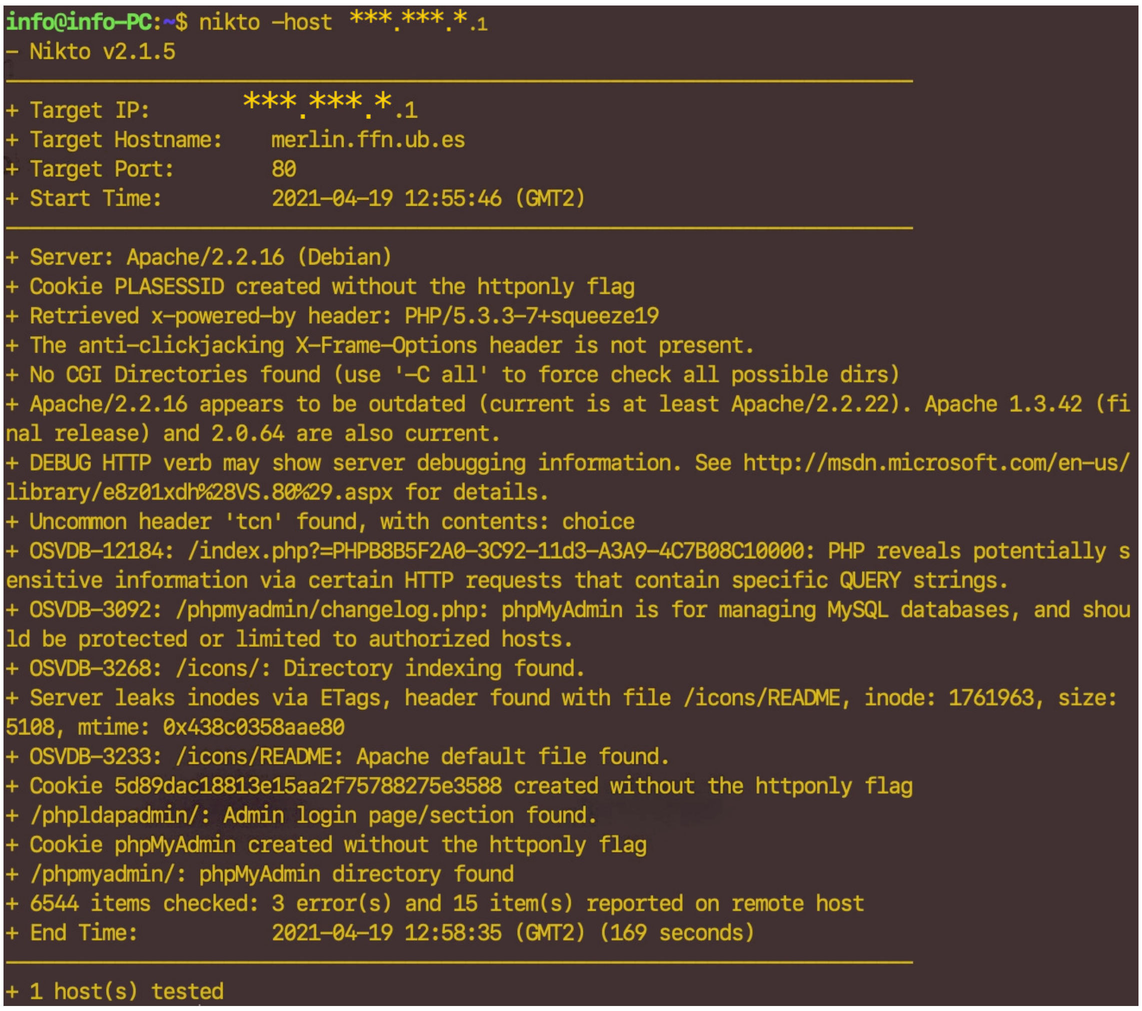
A simple tweak of the /etc/bind/named.conf.options file by adding just a line containing *version none;* would have been enough to mitigate this security issue. Now, performing the same scan as before results in no version leak.



```
info@info-PC:~$ nmap -sV obelix -p53
Starting Nmap 7.80 ( https://nmap.org ) at 2021-04-13 13:47 CEST
Nmap scan report for obelix (  ***.***.*.1)
Host is up (0.00020s latency).
rDNS record for   ***.***.*.1: merlin.ffn.ub.es

PORT   STATE SERVICE VERSION
53/tcp open  domain  ISC BIND

Service detection performed. Please report any incorrect results
at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 6.31 seconds
info@info-PC:~$ 
```

Figure 3: Nmap Scan of the DNS version on the analysed machine (fixed).

- **LDAP OpenLDAP 2.4.23**: It is currently using the MD5 algorithm to calculate the hashes of the users' passwords, which is considered to be insecure nowadays.
- **Web Apache2 Server 2.2.16**: There is a tool that allows to run a web scan in search of vulnerabilities on a server in a pretty easy and simple way called Nikto. The following figure shows the result of running Nikto in the web server of the department and it is not precisely soothing.

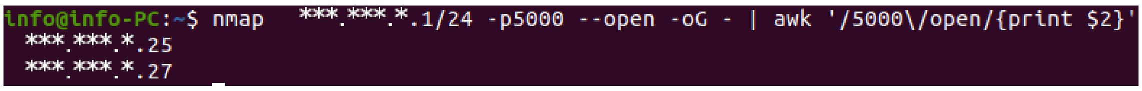


```
info@info-PC:~$ nikto -host ***.***.*.1
- Nikto v2.1.5
---------------------------------------------------------------------------
+ Target IP:          ***.***.*.1
+ Target Hostname:    merlin.ffn.ub.es
+ Target Port:        80
+ Start Time:         2021-04-19 12:55:46 (GMT2)
---------------------------------------------------------------------------
+ Server: Apache/2.2.16 (Debian)
+ Cookie PLASESSID created without the httponly flag
+ Retrieved x-powered-by header: PHP/5.3.3-7+squeeze19
+ The anti-clickjacking X-Frame-Options header is not present.
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ Apache/2.2.16 appears to be outdated (current is at least Apache/2.2.22). Apache 1.3.42 (fi
nal release) and 2.0.64 are also current.
+ DEBUG HTTP verb may show server debugging information. See http://msdn.microsoft.com/en-us/
library/e8z01xdh%28VS.80%29.aspx for details.
+ Uncommon header 'tcn' found, with contents: choice
+ OSVDB-12184: /index.php?=PHPB8B5F2A0-3C92-11d3-A3A9-4C7B08C10000: PHP reveals potentially s
ensitive information via certain HTTP requests that contain specific QUERY strings.
+ OSVDB-3092: /phpmyadmin/changelog.php: phpMyAdmin is for managing MySQL databases, and shou
ld be protected or limited to authorized hosts.
+ OSVDB-3268: /icons/: Directory indexing found.
+ Server leaks inodes via ETags, header found with file /icons/README, inode: 1761963, size:
5108, mtime: 0x438c0358aae80
+ OSVDB-3233: /icons/README: Apache default file found.
+ Cookie 5d89dac18813e15aa2f75788275e3588 created without the httponly flag
+ /phpldapadmin/: Admin login page/section found.
+ Cookie phpMyAdmin created without the httponly flag
+ /phpmyadmin/: phpMyAdmin directory found
+ 6544 items checked: 3 error(s) and 15 item(s) reported on remote host
+ End Time:           2021-04-19 12:58:35 (GMT2) (169 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested
```

Figure 4: Web service scan using Nikto on the analysed machine.

- **Database MySQL 5.0.51a**: Data is stored not encrypted and there are no backups of it. Additionally, there is no encryption on the communication between the database and the client.
- **Backup**: It has been stopped since 2019 and it was never configured to backup the MySQL. It also does not have a good configuration of full and incremental/differential backups to fit the needs of the server and the available disk space.

  Performing a quick scan of the default port used by synology machines (5000) we can see the following hosts up on the network:



```
info@info-PC:~$ nmap  ***.***.*.1/24 -p5000 --open -oG - | awk '/5000\/open/{print $2}'
 ***.***.*.25
 ***.***.*.27
```

Figure 5: Synology machines scan using Nmap on the local network.

In the case of the .25 machine the ssh service was activated and after some minutes of performing a password cracking of the default user *admin* using a popular tool called John the Ripper and a publicly available dictionary, the password was guessed.

4 Implementations

First of all, a new machine will be acquired to replace the current main server of the department. It will run all the necessary services implementing the most advanced security features and assuring that the latest security patches of every service are always up to date. Due to the department's infrastructure, it is required for the new server to be racked instead of a tower type. This will leave more space for new GPU machines incoming the next few months and offers a better fit on a new data centre that is being built. The chosen operative system has been Ubuntu Server 20.04.2 LTS since it brings almost all the features that are needed for this work, as well as a promise from the developers to keep releasing security patches until 2030. In addition, Ubuntu is a distribution in which I personally have plenty of experience with, and this is a fact that must be taken into account. All the implementations that are going to be detailed in the following sections will be installed on top of this OS, after their proper configuration and testing under the protection of breaking the system offered by a docker container.

The characteristics for this machine have been specifically designed for this master thesis and it consists of the following ones:

- **CPU**: Intel Xeon Silver 4214R 2.4 GHz, 12 Cores / 24 Threads, powerful enough to run all the department's required services.
- **RAM**: 32 GB RDIMM, 3200 MT/s: This is a server segment type of RAM memory that provides a high value of security since it has a record of what is stored in them and of the inputs and outputs between the controller and the RAM memory module. This type of memory works with less electrical load on the controller and therefore allows to have a system with more RAM capacity but with fewer physical modules. These modules, when accessing with a register, the controller is able to access the storage directly without the need for a buffer and as it is a direct access the dual channel communication speed improves performance.
- **Storage**: 2x 1TB 7.2K RPM SATA 6 Gbps Hot-plug HDD
- **RAID: PERC H730P Controller**: A RAID controller is in this case a hardware device, although it can also be a software program, used to manage hard disk drives (HDDs) or solid-state drives (SSDs) in a computer or storage array so they work as a logical unit. A RAID controller provides a degree of protection for stored data and may also help to improve computing performance by accelerating access to stored data.
- **Trusted Platform Module 2.0**: The main function of this chip is to provide a physical place in which to store credentials, certificates and encryption keys that serve both to encrypt other data and to store our passwords themselves. One of the faculties that make this chip so secure is that it can only communicate with the processor, uniquely and exclusively, so that no other hardware component can access it without the permission of the processor.

4.1 Certification Authority

In the following sections there is going to be a need of a certification authority (CA) responsible of signing certificates for some protocols such as TLS in order to provide secure connections for the corresponding service. In this section, the process of creation of such CA in the server machine will be described [4].

There are three types of certification authorities that can help to achieve proper certification for the services described in later sections:

- **Commercial and Private Certificate Authorities**:
  - Strength: Most commercial certificate authorities are trusted by default in most browsers.
  - Weakness: Expensive.
- **Let's Encrypt**:
  - Strength: Provides an automated mechanism to request and renew free domain validated certificates (ACME).
  - Weakness: Does not provide certificates for bare IP addresses.
- **Self-signed**:
  - Strength: Full customisation and very flexible.
  - Weakness: The CA must be manually marked as trusted.

4.1.1 Chosen Technology: Self-Signed Certification Authority

Given the fact that the services will be only used for internal purposes -thus discarding Private CAs- the decision has been to implement a self-signed certification authority due to the flexibility that it provides in front of its counterpart, *Let's Encrypt*. This will make life easier for the system administrator when a need to create new certificates or modify valid ones arise.

4.1.2 Implementation

Some of the department's required services detailed later need a certification authority to sign their TLS certificates. Although there will be a CA created specifically in the services that need it, in the final implementation there must be a common Root CA for all the services, due to the fact that the certificates are going to be self-signed. Therefore, the need for a Root CA implementation arise.

To begin with, we will copy the */etc/ssl/openssl.cnf* file to a new directory called *rootca*. The CA's base directory of this file must be changed to ., so that the *openssl ca* command looks for the CA files and directories in the same directory it is run. As a consequence, our CA will only work if the *openssl ca* command is run from the *rootca* directory.

```
[ CA_default ]
...
dir     = .
```

Next, we change the default bit length for the keys generated by that command to 4096, which is the security recommendation for a root CA.

```
[ req ]
default_bits    = 4096
...
```

It is a security recommendation to delimit the purpose of the self-signed certificate for our root CA.

```
[ v3_ca ]
basicConstraints          = critical,CA:true
subjectKeyIdentifier      = hash
authorityKeyIdentifier    = keyid:always,issuer
keyUsage                  = critical, cRLSign, keyCertSign
```

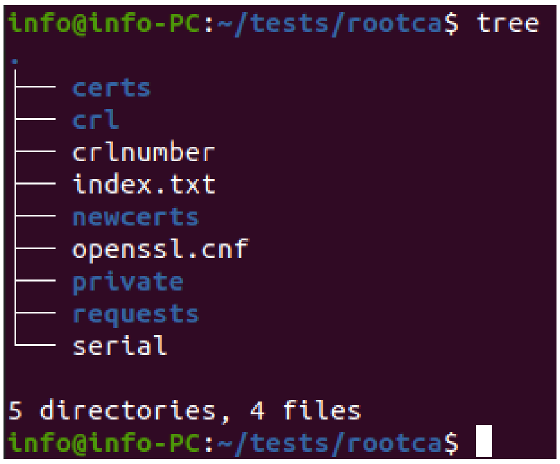
The next extension will be only added to TLS client certificates, which are used when the clients authenticate to a TLS server.

```
[ usr_cert ]
basicConstraints     = CA:FALSE
subjectKeyIdentifier = hash
authorityKeyIdentifier = keyid, issuer
keyUsage             = critical, nonRepudiation, digitalSignature,
    keyEncipherment, keyAgreement
extendedKeyUsage     = critical, clientAuth
```

The same procedure needs to be done for the server certificates.

```
[ server_cert ]
basicConstraints     = CA:FALSE
subjectKeyIdentifier = hash
authorityKeyIdentifier = keyid, issuer
keyUsage             = critical, nonRepudiation, digitalSignature,
    keyEncipherment, keyAgreement
extendedKeyUsage     = critical, serverAuth
```

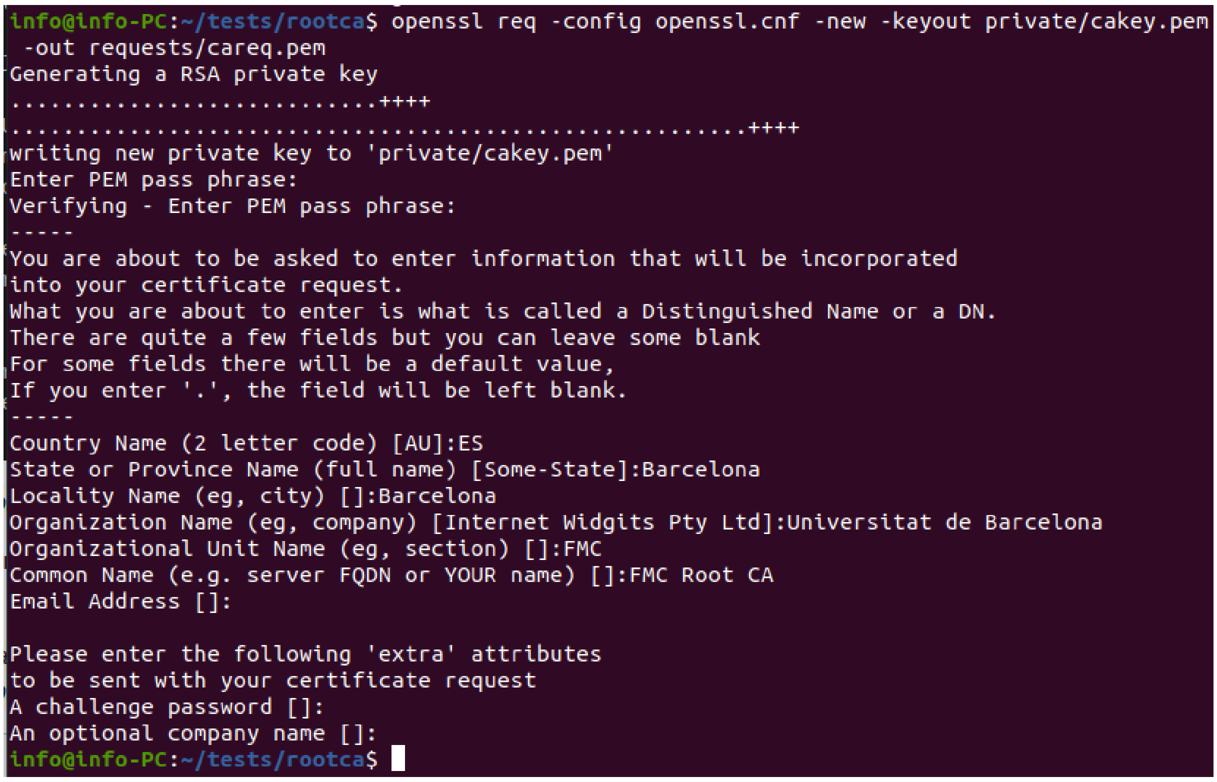
With this, the configuration of the *openssl.cnf* file is finished. Now we need to create a set of files and directories which will be necessary for the correct functioning of the root CA. Those are the ones depicted in the next figure.



```
info@info-PC:~/tests/rootca$ tree
.
├── certs
├── crl
├── crlnumber
├── index.txt
├── newcerts
├── openssl.cnf
├── private
├── requests
└── serial

5 directories, 4 files
info@info-PC:~/tests/rootca$
```

Figure 6: RootCA directory files structure.

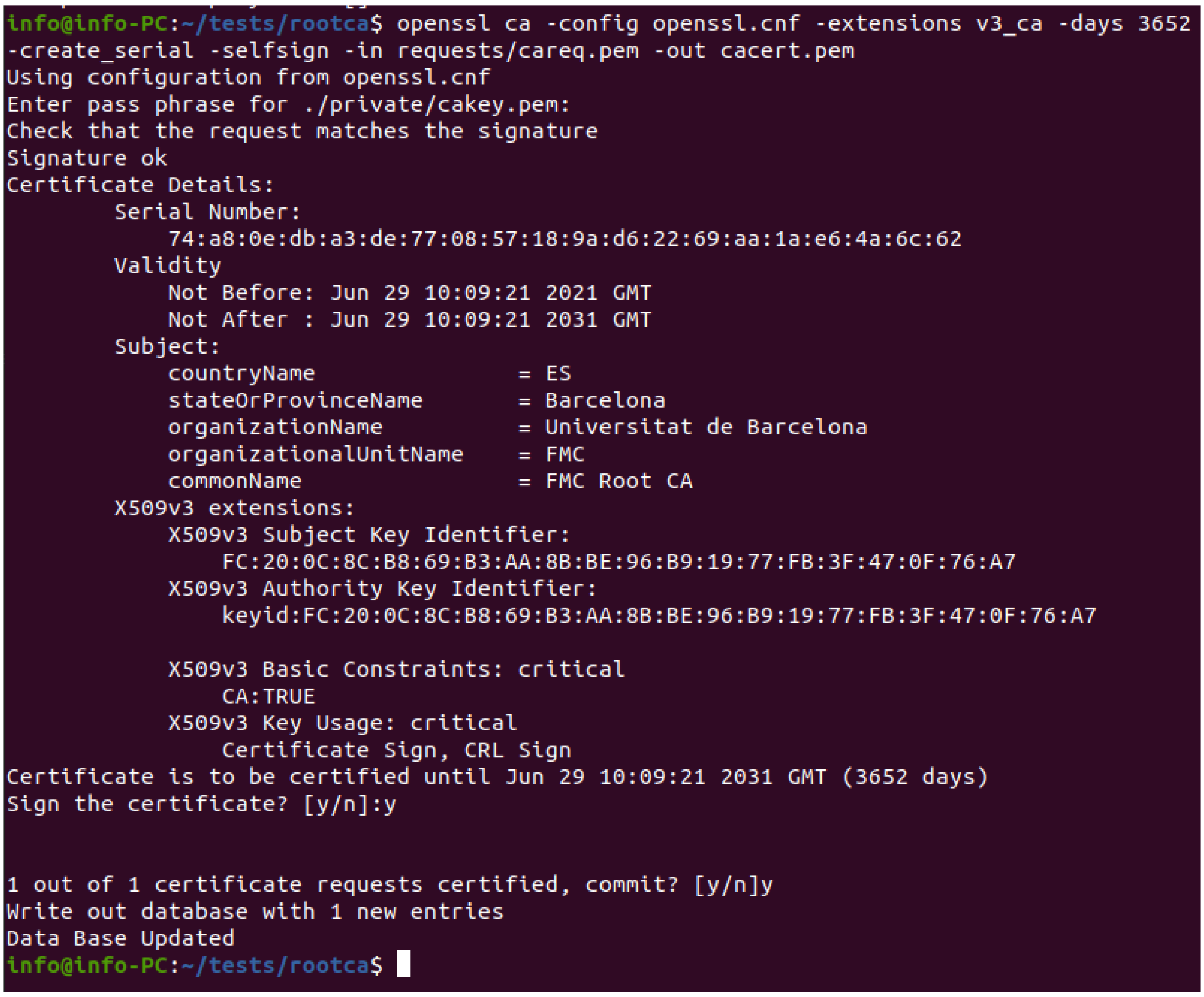
Now we can create the CA's certificate, the private key and the serial. The first two can be created with a single command like this:



```
info@info-PC:~/tests/rootca$ openssl req -config openssl.cnf -new -keyout private/cakey.pem
 -out requests/careq.pem
Generating a RSA private key
...........................++++
........................................................++++
writing new private key to 'private/cakey.pem'
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:ES
State or Province Name (full name) [Some-State]:Barcelona
Locality Name (eg, city) []:Barcelona
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Universitat de Barcelona
Organizational Unit Name (eg, section) []:FMC
Common Name (e.g. server FQDN or YOUR name) []:FMC Root CA
Email Address []:

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
info@info-PC:~/tests/rootca$
```

Figure 7: RootCA certificate and private key creation.

After that, we will sign the request for creating the CA's certificate. With the same command we will create the serial file.



```
info@info-PC:~/tests/rootca$ openssl ca -config openssl.cnf -extensions v3_ca -days 3652
-create_serial -selfsign -in requests/careq.pem -out cacert.pem
Using configuration from openssl.cnf
Enter pass phrase for ./private/cakey.pem:
Check that the request matches the signature
Signature ok
Certificate Details:
        Serial Number:
            74:a8:0e:db:a3:de:77:08:57:18:9a:d6:22:69:aa:1a:e6:4a:6c:62
        Validity
            Not Before: Jun 29 10:09:21 2021 GMT
            Not After : Jun 29 10:09:21 2031 GMT
        Subject:
            countryName               = ES
            stateOrProvinceName       = Barcelona
            organizationName          = Universitat de Barcelona
            organizationalUnitName    = FMC
            commonName                = FMC Root CA
        X509v3 extensions:
            X509v3 Subject Key Identifier:
                FC:20:0C:8C:B8:69:B3:AA:8B:BE:96:B9:19:77:FB:3F:47:0F:76:A7
            X509v3 Authority Key Identifier:
                keyid:FC:20:0C:8C:B8:69:B3:AA:8B:BE:96:B9:19:77:FB:3F:47:0F:76:A7

            X509v3 Basic Constraints: critical
                CA:TRUE
            X509v3 Key Usage: critical
                Certificate Sign, CRL Sign
Certificate is to be certified until Jun 29 10:09:21 2031 GMT (3652 days)
Sign the certificate? [y/n]:y


1 out of 1 certificate requests certified, commit? [y/n]y
Write out database with 1 new entries
Data Base Updated
info@info-PC:~/tests/rootca$
```

Figure 8: RootCA certificate self signing.

With this, the root certification authority creation process is finalised, and the services that need certificates to be signed will be able to make use of it, in production.

4.2 Database

Database security is a complex and challenging task that involves all aspects of information security technologies and practices. It's also naturally at odds with database usability. The more accessible and usable the database, the more vulnerable it is to security threats; the more invulnerable the database is to threats, the more difficult it is to access and use (this paradox is sometimes referred to as Anderson's Rule [5]).

By definition, a data breach is a failure to maintain the confidentiality of data in a database. Many software misconfigurations, vulnerabilities, or patterns of carelessness or misuse can result in breaches. The following are among the most common types or causes of database security attacks and their causes [6].

- **Insider threats**: An insider threat is a security threat from any one of three sources with privileged access to the database:
  - A malicious insider who intends to do harm.
  - A negligent insider who makes errors that make the database vulnerable to attack.
  - An infiltrator who somehow obtains credentials via a scheme such as phishing or by gaining access to the credential database itself.

  Insider threats are among the most common causes of database security breaches and are often the result of allowing too many employees to hold privileged user access credentials.
- **Human error**: Accidents, weak passwords, password sharing, and other unwise or uninformed user behaviours continue to be the cause of nearly half (49%) of all reported data breaches.
- **Exploitation of database software vulnerabilities**: Hackers make their living by finding and targeting vulnerabilities in all kinds of software, including database management software. All major commercial database software vendors and open source database management platforms issue regular security patches to address these vulnerabilities, but failure to apply these patches in a timely fashion can increase your exposure.
- **SQL/NoSQL injection attacks**: A database-specific threat, these involve the insertion of arbitrary SQL or non-SQL attack strings into database queries served by web applications or HTTP headers. Organisations that don't follow secure web application coding practices and perform regular vulnerability testing are open to these attacks.
- **Buffer overflow exploitation**: Buffer overflow occurs when a process attempts to write more data to a fixed-length block of memory than it is allowed to hold. Attackers may use the excess data, stored in adjacent memory addresses, as a foundation from which to launch attacks.

- **Denial of service (DoS/DDoS)** attacks: In a denial of service (DoS) attack, the attacker deluges the target server—in this case the database server—with so many requests that the server can no longer fulfil legitimate requests from actual users, and, in many cases, the server becomes unstable or crashes. In a distributed denial of service attack (DDoS), the deluge comes from multiple servers, making it more difficult to stop the attack.

4.2.1 Solution: All levels security

When evaluating database security there must be considered every of the following areas: physical security, administrative and network access controls, end user account/device security, encryption, database software security, application/web server security, backup security, auditing.

Nowadays almost any database server allows for each one of the previously mentioned areas. Since in our case we will have small to medium sized data sets and the information gathered can be structured in such a way that it can be segmented into tables and related between each other; the best choice is a relational database.

4.2.2 Chosen Technology: MariaDB

MariaDB is a fork of MySQL 5.1, although over time it has evolved into a self-contained database management system. The definition of files and tables is supported and identical protocols, structures, and programming interfaces are used. All MySQL connectors can be used with MariaDB without modifying them. Although both use the same standard storage subsystem engine, InnoDB/XtraDB, MariaDB offers quite a lot more engines than MySQL, this is the main reason for the choice.

Besides, the MariaDB project is focused on guaranteeing that there will always be a free and open source version of MySQL; not as MySQL, that was bought by the controversial software company Oracle, that has other plans for the main branch of the project.

4.2.3 Implementation

First of all, the required packages for MariaDB need to be installed. In this case they are all included in the *mariadb-server* package itself so the installation command is pretty simple. The latest version for the server will be installed in the machine just by executing the following command:

```
$ sudo apt install -y mariadb-server
```

After the installation is complete, we will proceed to perform a secure installation of it. The installed package provides a script for that, called *mysql_secure_installation* that must be run as root, so the command to be executed is as follows:

```
$ sudo mysql_secure_installation
```

This script provides a guidance of some steps to run the server more securely than the default configuration offers. It allows the admin to modify the points below:

- Change root password: Setting the root password ensures that nobody can log into the MariaDB root user without the proper authorisation.
- Remove anonymous users: By default, a MariaDB installation has an anonymous user, allowing anyone to log into MariaDB without having to have a user account created for them. This is intended only for testing, and to make the installation go a bit smoother, but sometimes they are forgotten to be removed and can become a security hole on the system.
- Disallow root login remotely: Root should only be allowed to connect from *localhost*. This ensures that someone cannot guess the root password from the network.
- Remove test database: By default, MariaDB comes with a database named 'test' that anyone can access. This is also intended only for testing and the same reason as before applies here.

Since the server uses the root account for tasks such as log rotation and server startup and discovery, it is better practice not to change the account's root authentication details. Modifying the configuration file credentials in */etc/mysql/debian.cnf* may work at first, but package updates may overwrite those changes. Instead of modifying the root account, package maintainers recommend creating a separate administrative account for password-based access. In order to do so we'll execute a command inside the database like this:

```
MariaDB [mysql]> GRANT ALL ON *.* TO 'admin'@'localhost' IDENTIFIED BY
    'password' WITH GRANT OPTION;
```

And then to set a password for this new created user, again using the database built-in shell:

```
MariaDB [mysql]> SET PASSWORD FOR 'admin'@'localhost' = PASSWORD('<
    password>');
```

It is well known that MySQL runs on port 3306, and its superuser is called 'root'. Even though we previously disabled remote login for 'root', it is never enough regarding security; and maybe tomorrow a security hole appears which allows for 'root' remote login even being disabled. So, in order to make things harder, it is quite simple to change this. To some extent, this is an example of security through obscurity but it may at least stop automated attempts to get access to the 'root' user. To change the default port, we need to edit */etc/mysql/mariadb.conf.d/50-server.cnf* and set 'port' variable to some other value. To verify that changes have been successfully applied we could run this command inside the database that will return us the current port used to listen to external connections:

```
MariaDB [mysql]> SHOW VARIABLES LIKE 'port';
```

If both server and client has the ability to run 'LOAD DATA LOCAL INFILE', a client will be able to load data from a local file to a remote MySQL server. This, potentially, can help to read files the client has access to. For example, on an application server, one could access any file that the HTTP server has access to. To avoid it, we need to set *local-infile = 0* again in the */etc/mysql/mariadb.conf.d/50-server.cnf* under the *[mysqld]* section.

Another issue is to make sure data is securely transferred over the network. With an exception of setups where all the clients are local and use Unix socket to access MySQL, in majority of cases, data which forms a result set for a query, leaves the server and is transferred to the client over the network. Network traffic can be sniffed, and through those means, sensitive data would be exposed.

To prevent this from happening, it is possible to use TLS to encrypt traffic, both server and client-side. The idea is to create a TLS connection between a client and a MySQL server. We can also create a TLS connection between the master and the slaves. This will ensure that all data that is transferred is safe, and although it can still be sniffed by an attacker who gained access to your network, they won't be able to decrypt it and see the raw data.

In order to do so we need to create some certificates for the TLS connection. First of all we need to create the certification authority key and certificate. This can be achieved pretty easily using *OpenSSL*. The below commands should do the work:

```
$ sudo openssl genrsa 4096 > ca-key.pem
$ sudo openssl req -new -x509 -nodes -days 365000 -key ca-key.pem -out
    ca-cert.pem
```

And the same for the server, in this case we create the key and the certificate request in a single command:

```
$ sudo openssl req -newkey rsa:2048 -days 365000 -nodes -keyout server-key.pem
    -out server-req.pem
```

Finally sign the server certificate, running:

```
$ sudo openssl x509 -req -in server-req.pem -days 365000 -CA ca-cert.pem
    -CAkey ca-key.pem -set_serial 01 -out server-cert.pem
```

Now we also need the have a certificate for the mysql client, the Python/Perl/Ruby/Whatever app is going to use the client certificate to secure client-side connectivity. The following files must be installed on all the clients including the web server. To create the client key and process the rsa key, we run:

```
$ sudo openssl req -newkey rsa:2048 -days 365000 -nodes -keyout client-key.pem
    -out client-req.pem
$ sudo openssl rsa -in client-key.pem -out client-key.pem
```

Finally, sign the client certificate:

```
$ sudo openssl x509 -req -in client-req.pem -days 365000 -CA ca-cert.pem
    -CAkey ca-key.pem -set_serial 01 -out client-cert.pem
```

To end with the TLS configuration there are still two more things to do. The proper server and client configuration files need to be modified. Once more, some lines have to be added under the *[mysqld]* section of the */etc/mysql/mariadb.conf.d/50-server.cnf* file.

Those are the following:

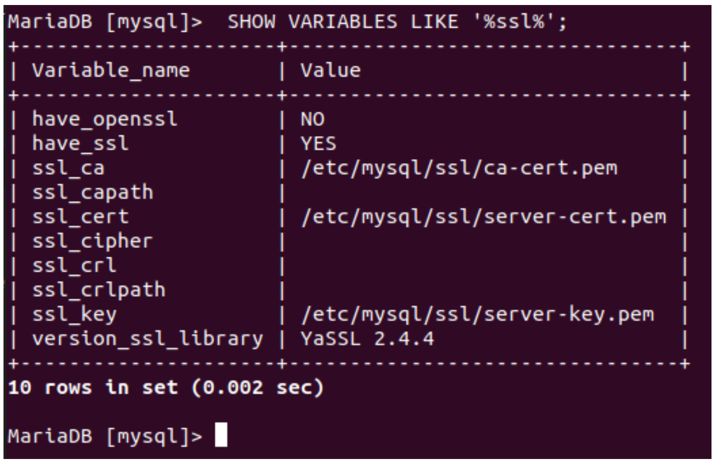
```
[mysqld]

ssl-ca   = /etc/mysql/ssl/ca-cert.pem
ssl-cert = /etc/mysql/ssl/server-cert.pem
ssl-key  = /etc/mysql/ssl/server-key.pem
```

And the same will have to be performed for the *[mysql]* section on the */etc/mysql/mariadb.conf.d/50-mysql-clients.cnf* with the corresponding files:

```
[mysql]

ssl-ca   = /etc/mysql/ssl/ca-cert.pem
ssl-cert = /etc/mysql/ssl/client-cert.pem
ssl-key  = /etc/mysql/ssl/client-key.pem
```

Now, just to verify that everything has been successfully configured, we can run a MariaDB instance and execute the following command to check if SSL is enabled:

```
MariaDB [mysql]> SHOW VARIABLES LIKE '%ssl%';
```



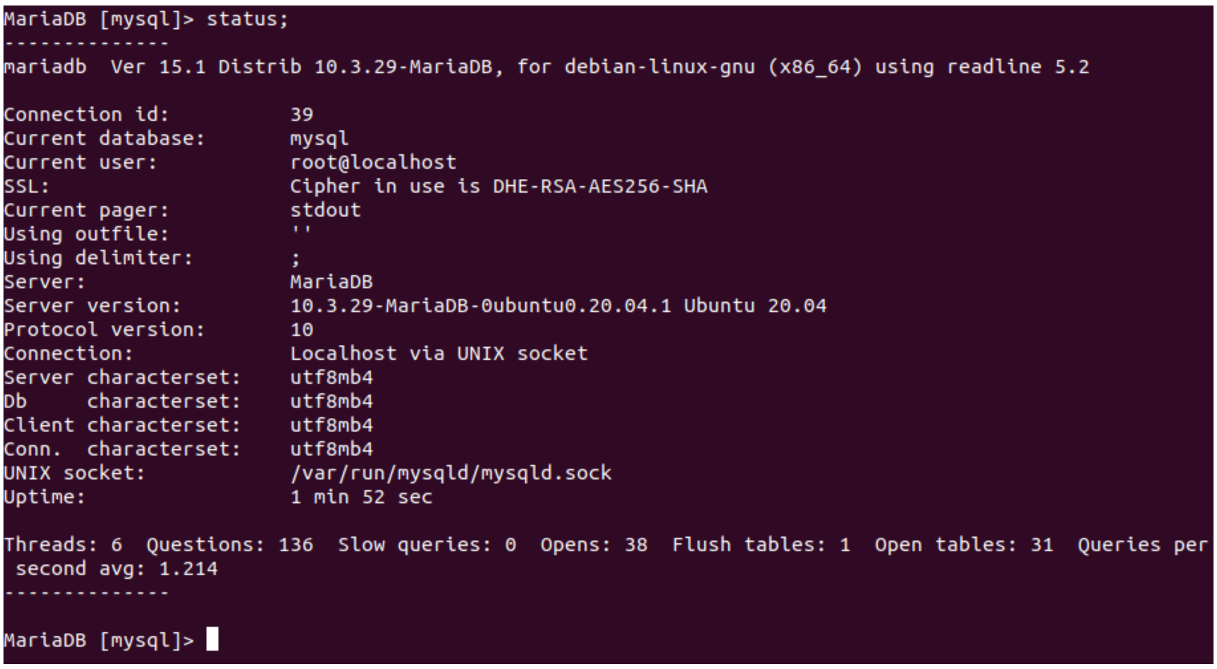
```
MariaDB [mysql]>  SHOW VARIABLES LIKE '%ssl%';
+---------------------+--------------------------------+
| Variable_name       | Value                          |
+---------------------+--------------------------------+
| have_openssl        | NO                             |
| have_ssl            | YES                            |
| ssl_ca              | /etc/mysql/ssl/ca-cert.pem     |
| ssl_capath          |                                |
| ssl_cert            | /etc/mysql/ssl/server-cert.pem |
| ssl_cipher          |                                |
| ssl_crl             |                                |
| ssl_crlpath         |                                |
| ssl_key             | /etc/mysql/ssl/server-key.pem  |
| version_ssl_library | YaSSL 2.4.4                    |
+---------------------+--------------------------------+
10 rows in set (0.002 sec)

MariaDB [mysql]> 
```

Figure 9: Dump of the *SHOW VARIABLES LIKE '%ssl%';* command.

Also, it can be checked by issuing the following command:

```
MariaDB [mysql]> status;
```

Which returns the next output:



```
MariaDB [mysql]> status;
--------------
mariadb  Ver 15.1 Distrib 10.3.29-MariaDB, for debian-linux-gnu (x86_64) using readline 5.2

Connection id:          39
Current database:       mysql
Current user:           root@localhost
SSL:                    Cipher in use is DHE-RSA-AES256-SHA
Current pager:          stdout
Using outfile:          ''
Using delimiter:        ;
Server:                 MariaDB
Server version:         10.3.29-MariaDB-0ubuntu0.20.04.1 Ubuntu 20.04
Protocol version:       10
Connection:             Localhost via UNIX socket
Server characterset:    utf8mb4
Db     characterset:    utf8mb4
Client characterset:    utf8mb4
Conn.  characterset:    utf8mb4
UNIX socket:            /var/run/mysqld/mysqld.sock
Uptime:                 1 min 52 sec

Threads: 6  Questions: 136  Slow queries: 0  Opens: 38  Flush tables: 1  Open tables: 31  Queries per
 second avg: 1.214
--------------

MariaDB [mysql]> 
```

Figure 10: Dump of the *status;* command.

It shows that, in this case, the cipher being used is the *DHE-RSA-AES256-SHA*, which is considered to be a weak cipher suite according to Ciphersuite.info. Despite that, it is secure enough for at least the first implementation of a secure database communication.

Finally, the last step for securing the database is to encrypt the data at rest. Having tables encrypted makes it almost impossible for someone to access or steal a hard disk and get access to the original data. This functionality is also known as "Transparent Data Encryption (TDE)". As a counterpart, using encryption has an overhead of roughly 3-5%, according to MariaDB knowledge base.

It is also noteworthy that encryption at-rest is an additional protection, but it is not a replacement for a good firewall, strong passwords, correct user permissions, and in-transit encryption between the client and server.

To start with the process of encryption the data at rest, we'll need once more the *OpenSSL* tool to generate a few encryption keys. This one-liner bash script command issues five random 128 bit keys and stores them in a file called */etc/mysql/encryption/keyfile*:

```
$ for i in {1..5}; do echo -n "$i;" >> /etc/mysql/encryption/keyfile; openssl
    rand -hex 32 >> /etc/mysql/encryption/keyfile; done;
```

The only algorithm that MariaDB currently supports to encrypt the key file is Cipher Block Chaining (CBC) mode of Advanced Encryption Standard (AES). The encryption key will be created using the SHA256 algorithm, but before that we need to generate a random password to be used for the encryption, so we will have to run two commands here:

```
$ openssl rand -hex 128 > /etc/mysql/encryption/keyfile.key
$ openssl enc -aes-256-cbc -pass file:/etc/mysql/encryption/keyfile.key -in
    /etc/mysql/encryption/keyfile -out /etc/mysql/encryption/keyfile.enc
```

Finally we need to modify again the */etc/mysql/mariadb.conf.d/50-server.cnf* file to indicate the server to use the encryption shceme that is been created. We will do so with the following lines:

```
[mysqld]
...
plugin_load_add = file_key_management
file_key_management_filename = /etc/mysql/encryption/keyfile.enc
file_key_management_filekey = FILE:/etc/mysql/encryption/keyfile.key
file_key_management_encryption_algorithm = aes_cbc
encrypt_binlog = 1

innodb_encrypt_tables = ON
innodb_encrypt_log = ON
innodb_encryption_threads = 4
innodb_encryption_rotate_key_age = 0 # Do not rotate key
```

Now, to ensure that everything is working properly, two tables have been created: one is in clear text and the other has been stored encrypted. Both tables have been provided with a single entry consisting of a string indicating whether the table is encrypted or not. One of the first things a hacker would do to a new authorized database would be to run the *strings* command on it. This is the obtained output on the clear-text table:

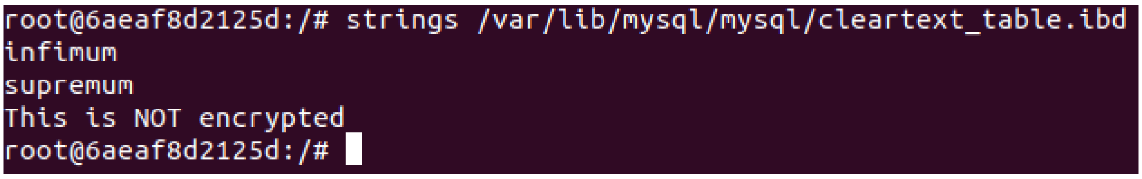
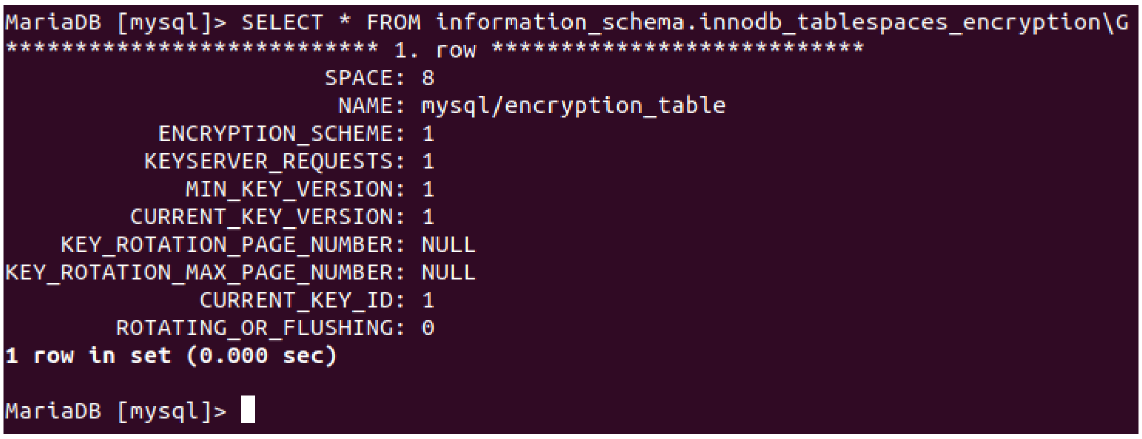

```
root@6aeaf8d2125d:/# strings /var/lib/mysql/mysql/cleartext_table.ibd
infimum
supremum
This is NOT encrypted
root@6aeaf8d2125d:/#
```

Figure 11: Dump of the *strings* command on the file that contains the *cleartext_table* table.

It is clearly visible that is very easy to obtain the unencrypted entries of the table.

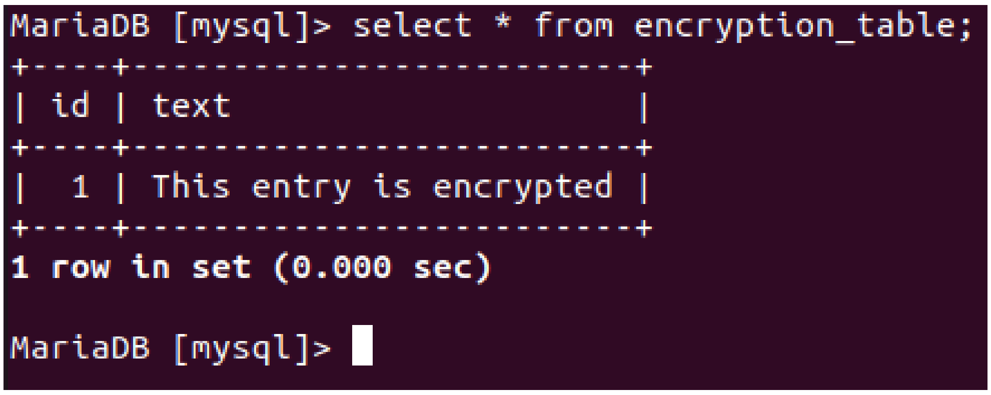
On the other hand, the process of extracting sensitive information of a database becomes at least non-trivial when the data is encrypted. After applying the settings mentioned before, a newly created table is by default encrypted, as shown in the following figure:



```
MariaDB [mysql]> SELECT * FROM information_schema.innodb_tablespaces_encryption\G
*************************** 1. row ***************************
                       SPACE: 8
                        NAME: mysql/encryption_table
           ENCRYPTION_SCHEME: 1
          KEYSERVER_REQUESTS: 1
             MIN_KEY_VERSION: 1
         CURRENT_KEY_VERSION: 1
    KEY_ROTATION_PAGE_NUMBER: NULL
KEY_ROTATION_MAX_PAGE_NUMBER: NULL
              CURRENT_KEY_ID: 1
        ROTATING_OR_FLUSHING: 0
1 row in set (0.000 sec)

MariaDB [mysql]>
```

Figure 12: Dump of the currently encrypted tables on the database.

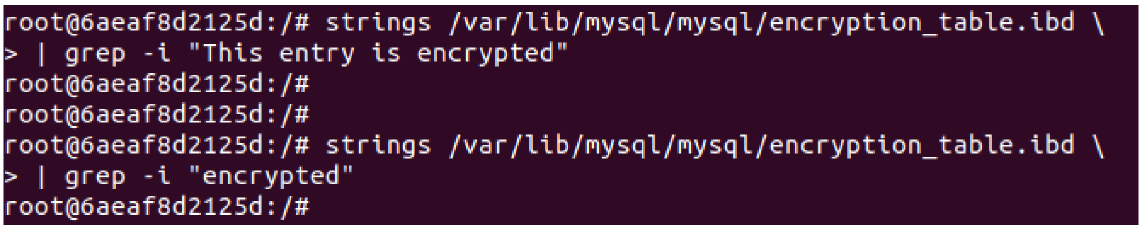
And the contents of the *encryption_table* table are properly seen within MariaDB:



```
MariaDB [mysql]> select * from encryption_table;
+----+-------------------------+
| id | text                    |
+----+-------------------------+
|  1 | This entry is encrypted |
+----+-------------------------+
1 row in set (0.000 sec)

MariaDB [mysql]>
```

Figure 13: Contents of the *encryption_table* table.

Then, if we run again the *strings* command on this table in order to extract its sensitive information, we get the output shown below:



```
root@6aeaf8d2125d:/# strings /var/lib/mysql/mysql/encryption_table.ibd \
> | grep -i "This entry is encrypted"
root@6aeaf8d2125d:/#
root@6aeaf8d2125d:/#
root@6aeaf8d2125d:/# strings /var/lib/mysql/mysql/encryption_table.ibd \
> | grep -i "encrypted"
root@6aeaf8d2125d:/#
```

Figure 14: Extraction of sensitive information from the *encryption_table* table.

As seen in figure 11, the encrypted entries remain unrevealed by the *strings* command, thus demonstrating that MariaDB data-at rest encryption has been successfully configured.

4.3 Domain Name System (DNS)

DNS is an Internet protocol whose function is the resolution of domain names, converting them into IPs. Each domain has assigned DNS (name servers), allowing the domain name to be converted to the corresponding IP [7].

DNS is insecure because by default DNS queries are not encrypted, which can be exploited by middle entities. DNS cache poison is one of the DNS abuses that is widely used by the Great Firewall of China (GFW) to censor Chinese Internet [8]. GFW checks every DNS query that is sent to a DNS server outside of China. Since plain text DNS protocol is based on UDP, which is a connection-less protocol, GFW can spoof both the client IP and server IP. When GFW finds a domain name on its block list, it changes the DNS response. For example, if a Chinese Internet user wants to visit google.com, the Great firewall of China returns to the DNS resolver an IP address located in China instead of Google's real IP address. Then the DNS resolver returns the fake IP address to the user's computer.

4.3.1 Solution: DNS over TLS

DNS over TLS (DoT) means that DNS queries are sent over a secure connection encrypted with TLS, the same technology that encrypts HTTPS traffic, so third parties can not see the DNS queries.

By establishing a connection over a well-known TCP port, 853, clients and servers expect and agree to negotiate a TLS session to secure the channel. RFC 7858 defines the following method for using DNS over TLS to establish secure sessions:

- **Session Initiation**: A DNS server that supports DNS over TLS listens for and accepts TCP connections on Port 853, unless it has a mutual agreement with its server to use a different port for DoT. When using DNS over TLS, all TCP connections on Port 853 should be encrypted, as significant security issues arise in mixing encrypted and unencrypted data.
- **TLS Handshake and Authentication**: Once the DNS client successfully connects, it proceeds with the TLS handshake and will authenticate with the DNS server, if required. After the TLS negotiation is completed, the connection is encrypted.

Together with HTTPS and encrypted SNI (Server Name Indication), the user's browsing history is fully protected from ISP spying.

Another solution that could be a perfect fit for a clean system design -in which no pre-requisites would be needed to be fulfilled- would be DNSSEC. Although it offers great features it can not be applied to the use case of this project because the company has several devices that need for the existing regular DNS configuration and would not be able to work with DNSSEC.

4.3.2 Chosen Technology: BIND9 + Stunnel

BIND9 is an open source implementation of DNS, available for almost all Linux distributions. BIND allows to publish DNS information on internet as well as allows to resolve DNS queries for the users. BIND is by far the most used DNS software on Internet, so it has plenty of documentation and community which makes it easy to implement and maintain, but as a counterpart it is the preferred target for DNS attackers due to the great potential victims it provides. Nonetheless, it remains the greatest option regarding this project scenario.

On the other hand, in order to implement the TLS security over BIND9 we have Stunnel, which has been officially accepted by the creators of BIND9 (Internet Systems Consortium, or ISC) following the RFC 7858 specification for DoT. Stunnel is a proxy designed to add TLS encryption functionality to existing clients and servers without any changes in the programs' code. Its architecture is optimised for security, portability, and scalability (including load-balancing), making it suitable for large deployments. Stunnel uses the OpenSSL library for cryptography, so it supports whatever cryptographic algorithms are compiled into the library. Stunnel uses public-key cryptography with X.509 digital certificates to secure the TLS connection.

4.3.3 Implementation

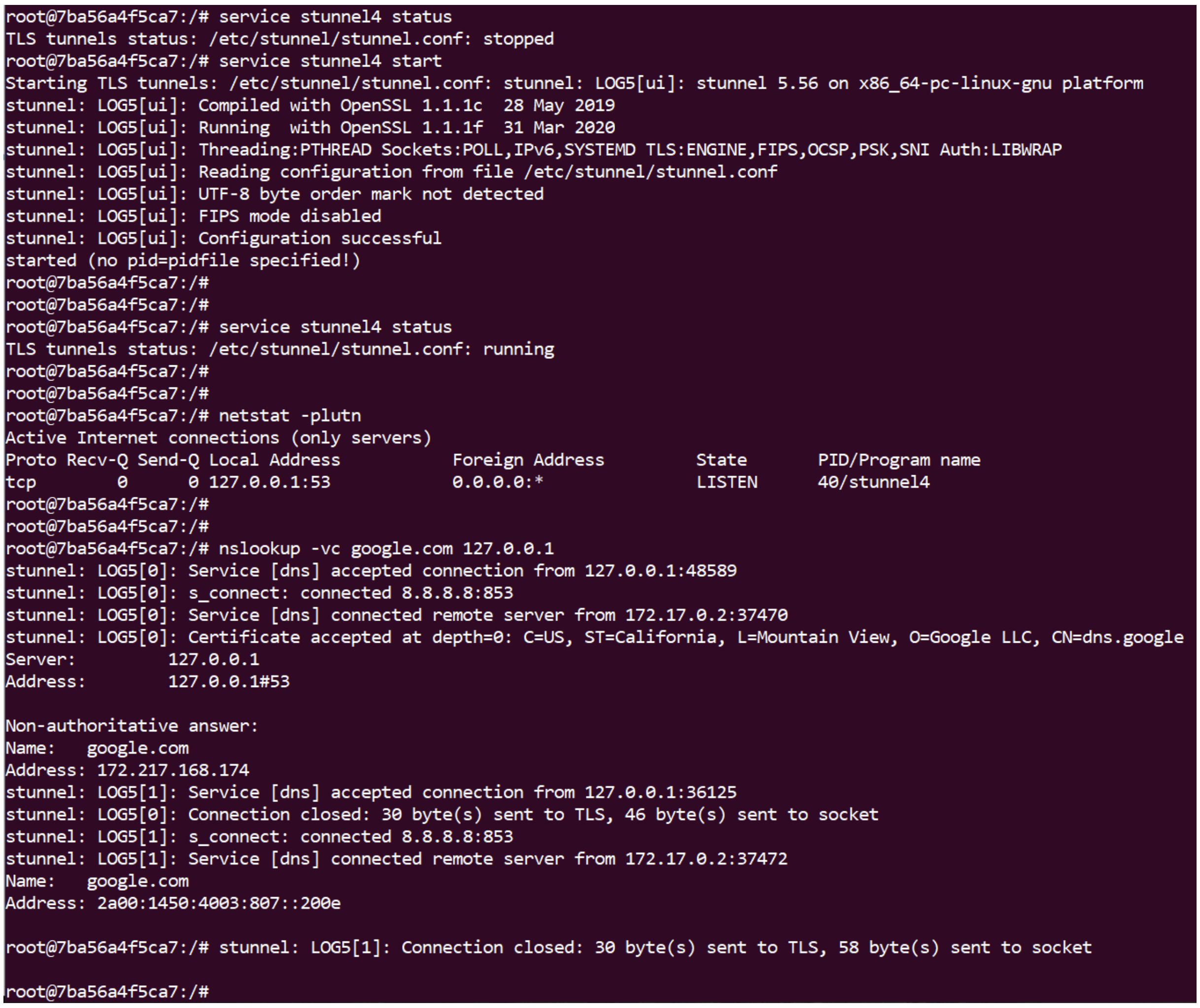
This implementation will show only the DNS over TLS (Stunnel) configuration assuming an already properly configured DNS server with BIND9 which will not be detailed in this section since it is out of the scope of this project due to the lack of security steps it involves.

First things first, install the necessary packages. In this case is especially simple, just by typing the following command and nothing else:

```
$ sudo apt install stunnel
```

Once the installation is finished, the configuration file will need to be created and populated with the proper rules to work as desired. In the first place, we create the configuration file as */etc/stunnel/stunnel.conf* and we populate it with the following lines:

```
[dns]

client = yes
accept = 127.0.0.1:53
connect = 8.8.8.8:853
CApath = /etc/ssl/certs
verifyChain = yes
checkIP = 8.8.8.8
```



```
root@7ba56a4f5ca7:/# service stunnel4 status
TLS tunnels status: /etc/stunnel/stunnel.conf: stopped
root@7ba56a4f5ca7:/# service stunnel4 start
Starting TLS tunnels: /etc/stunnel/stunnel.conf: stunnel: LOG5[ui]: stunnel 5.56 on x86_64-pc-linux-gnu platform
stunnel: LOG5[ui]: Compiled with OpenSSL 1.1.1c  28 May 2019
stunnel: LOG5[ui]: Running  with OpenSSL 1.1.1f  31 Mar 2020
stunnel: LOG5[ui]: Threading:PTHREAD Sockets:POLL,IPv6,SYSTEMD TLS:ENGINE,FIPS,OCSP,PSK,SNI Auth:LIBWRAP
stunnel: LOG5[ui]: Reading configuration from file /etc/stunnel/stunnel.conf
stunnel: LOG5[ui]: UTF-8 byte order mark not detected
stunnel: LOG5[ui]: FIPS mode disabled
stunnel: LOG5[ui]: Configuration successful
started (no pid=pidfile specified!)
root@7ba56a4f5ca7:/#
root@7ba56a4f5ca7:/#
root@7ba56a4f5ca7:/# service stunnel4 status
TLS tunnels status: /etc/stunnel/stunnel.conf: running
root@7ba56a4f5ca7:/#
root@7ba56a4f5ca7:/#
root@7ba56a4f5ca7:/# netstat -plutn
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 127.0.0.1:53            0.0.0.0:*               LISTEN      40/stunnel4
root@7ba56a4f5ca7:/#
root@7ba56a4f5ca7:/#
root@7ba56a4f5ca7:/# nslookup -vc google.com 127.0.0.1
stunnel: LOG5[0]: Service [dns] accepted connection from 127.0.0.1:48589
stunnel: LOG5[0]: s_connect: connected 8.8.8.8:853
stunnel: LOG5[0]: Service [dns] connected remote server from 172.17.0.2:37470
stunnel: LOG5[0]: Certificate accepted at depth=0: C=US, ST=California, L=Mountain View, O=Google LLC, CN=dns.google
Server:         127.0.0.1
Address:        127.0.0.1#53

Non-authoritative answer:
Name:   google.com
Address: 172.217.168.174
stunnel: LOG5[1]: Service [dns] accepted connection from 127.0.0.1:36125
stunnel: LOG5[0]: Connection closed: 30 byte(s) sent to TLS, 46 byte(s) sent to socket
stunnel: LOG5[1]: s_connect: connected 8.8.8.8:853
stunnel: LOG5[1]: Service [dns] connected remote server from 172.17.0.2:37472
Name:   google.com
Address: 2a00:1450:4003:807::200e

root@7ba56a4f5ca7:/# stunnel: LOG5[1]: Connection closed: 30 byte(s) sent to TLS, 58 byte(s) sent to socket

root@7ba56a4f5ca7:/#
```

Figure 15: Initialisation and test of DoT configuration using BIND9 and Stunnel.

As shown in the above figure, we can clearly see how the service is successfully initialised and running on port 53 on the local host's IP address. It also shows a test connection in which the certificate is accepted by the recipient and it returns the IP address of the requested name. Finally, on connection closing we can see a brief report that indicates that 30 bytes have been sent through the TLS protection, which was the aim of this implementation.

4.4 Dynamic Host Configuration Protocol (DHCP)

The DHCP is a client/server network protocol by which a server dynamically assigns an IP address and other network configuration parameters to each device on a network so that they can communicate with other IP networks [9].

A DHCP server is vulnerable to different types of attacks. Let's take a look at some of the attack types and tips on how to prevent or mitigate these risks.

- **Denial of Service**: Since the DHCP protocol doesn't require authentication from the client to provide network configurations, any user who has access to the network may obtain a lease of IP address. The data sent by the DHCP server may reveal information on the DNS servers IPs, which may comprise the network's security. Malicious users who have access to a DHCP-enabled network may create a denial-of-service attack on DHCP servers by flooding the server with a high number of lease requests, thereby depleting the number of leases that are available to other DHCP clients.
- **DHCP Starvation Attack**: The DHCP starvation attack is an attack where the hacker exhausts the address space available to the DHCP servers for a particular time period. This kind of attack is conducted by broadcasting DHCP requests with spoofed MAC addresses. Also, to gain access to the network, the attackers exploit the DHCP snooping, a mechanism used to provide the security of the network by filtering untrusted DHCP messages and by creating and maintaining a DHCP snooping binding database.
- **Rogue DHCP Server**: A hacker may set up a fake DHCP server on the attacked network, in order to cause man-of-the-middle, sniffing and reconnaissance attacks. This fake server is called a rogue server, and the attacker uses it to supply the clients with fake addresses and other network information to snoop into the data packets. The rogue server then provides its own DNS servers and network gateways, which redirect clients to malicious websites where they perform phishing attacks to obtain their confidential information such as credit card numbers and passwords.

4.4.1 Solution: Physical security, log auditing and limited permissioning

Maintaining proper physical security protocols for the hardware components such as servers, switches, and routers limits unauthorised access to the server system. Restricting wireless access for illicit individuals inside or outside the system by maintaining the user access policies also anneal the security perimeter.

Audit logging for every DHCP server on the network should be enabled along with keeping a tab on log files. These log files ensure safety at times when the DHCP server receives an unusually high number of lease requests from the clients. An audit log file contains the information required to track the source of attacks made against the DHCP server. The system event log should be analysed for explanatory information about the DHCP Server service as well.

In addition, administrative access to DHCP should be restricted to a limited number of individuals to prevent for undesired behaviours of the server and possible cyber attacks, because -remember- every single additional person with authorised access becomes a major potential security flaw.

4.4.2 Chosen Technology: ISC DHCP Server & Leases monitoring script

The Internet Systems Consortium (ISC) DHCP Server, implements the Dynamic Host Configuration Protocol (DHCP). ISC DHCP offers a complete open source solution for implementing DHCP servers, relay agents, and clients. ISC DHCP supports both IPv4 and IPv6, and is suitable for use in high-volume and high-reliability applications.

On the other hand, a bash script will be developed in order to audit the logs generated by the previously mentioned service. More specifically, the script will be focused on the issued leases located in the */var/lib/dhcp/dhcpd.leases* file. The script will mainly check that there are just as many leases as usual, with a certain threshold of course, to identify a possible denial of service and notify the administrator before it's too late. Additionally, the script will also notify the administrator in the case the number of granted leases decreases significantly in a short period of time, considering that it could be due to a rogue server attack/incident.

4.4.3 Implementation

In the first place, the required packages have to be installed, as always. In this case though they won't be installed following the common *apt install* procedure. Since aptitude does not install the latest stable version of the *isc-dhcp-server* package the process of installation will be performed by building the service from source code. The package can be easily found on the ISC web page under the DHCP section. Currently, the latest stable version, and thus the one that will be installed, is the 4.4.2-P1. The process is simple: download the package, decompress it, run the *configure* script and finally compile by executing the *make* command. Additionally, in order to have the *dhcpd* available on command line it is necessary to run one last command that is:

```
$ make install
```

After that, if the previous commands didn't raise any error the ISC DHCP Server should be installed on the machine. The current version of the service can be checked by issuing the following command:

```
$ dhcpd --version
```

Which in this case should output *isc-dhcpd-4.4.2-P1*.

Hereafter, in order for the DNS -configured in the previous section- to take effect on the devices connected to the network, it is required to properly indicate which is its IP in the configuration file. By doing this we will achieve that every time the DHCP assigns a valid IP to a device it will also assign the early configured DNS. This will be carried out by editing the */etc/dhcp/dhcpd.conf* file and replacing the following line with the desired DNS IP address, in this case the server's IP address (blurred out for privacy):

```
...
option domain-name-servers ***.***.*.1
...
```

Once the configuration file has been successfully modified, all that is left is to restart the service by issuing the command below:

```
$ service isc-dhcp-server restart
```

Now that the service is properly installed the development of the log auditing script can begin. The script will be written in bash language since it is the most simple and easy to integrate with the environment we are dealing with.

As mentioned in the previous section, the script will fulfil several tasks. The first one is to maintain a control over the amount of leases that are being issued on a "normal state" of the network. This means that it will be able to identify whether the amount of leases is too high or too low to be considered a normal behaviour of the network. In case the script detects an anomaly it will immediately notify the administrator via email.

On the second place, the script will also take care of the backup of the issued leases over time using the *rsync* tool. To avoid accumulating garbage, the script will backup the records just up to a week back. This decision is made in front of two facts: the first is that the script needs a significant amount of records (in this case one week) to learn which is the normal load of the network and the the amount of users on the department's network is considerably dynamic, some new employees arrive as well as some other leave so the script must be able to adapt itself to the increasing/decreasing number of issued leases without considering it an attack; on the other hand, in case of an attack it should be detected the same day so the logs of more than one week back are pretty much irrelevant.

Therefore, the script must be running 24 hours a day, which does not necessarily mean it needs to be active 100% of that time, in fact it will only be active for less than one second every minute, so the impact on the system performance is almost negligible.

Finally, the final script will look reasonably similar to the code below:

```
#!/bin/bash

while true; do
    rm /tmp/showdhcp.log &
    rm /tmp/tmp* &
    tempFile=$(mktemp)
    cat /var/lib/dhcp/dhcpd.leases \
      | grep \
         -e '^lease ' \
         -e '^ binding state' \
         -e hardware \
         -e client-hostname \
         -e } >$tempFile.1

    sed -e "s/^lease \(.*\) {$/\1~/" \
        -e "s/^ binding state \(.*\);$/\1~/" \
        -e "s/^ hardware ethernet \(.*\);$/\1~/" \
        -e "s/ client-hostname \(.*\);$/\1~/" \
        -e "s/^}//" \
        $tempFile.1 > $tempFile.2

    sed -e ":'loop';/~$/N;s/~\n/,/;t'loop'" \
        -e "s/,$//" \
        -e "s/,/\t/g" \
        $tempFile.2 | sort > $tempFile.3

    sed -e'$!N;/^\(.*\)\n\1$/!P;D' $tempFile.3 > $tempFile.4

    print=$(printf "IP Addr\t\tStatus\t\tMAC\t\tHost Name\n")
    cat $tempFile.4 > /tmp/showdhcp.log &
    sed -i "1i $print" /tmp/showdhcp.log
    #rm -rvf /tmp/Non_Active_User*
    #rm -rvf /tmp/Active_User*
    #cp -rvf $tempFile.4 /tmp/showdhcp.log &
    free=$(grep -o -w 'free' /tmp/showdhcp.log | wc -w)
    active=$(grep -o -w 'active' /tmp/showdhcp.log | wc -w)
    echo -e ""
    echo -e "--------- Total Count- Active & Non Active -----------"
    echo -e ""
    echo -e "Total Active User :"    $active
    echo -e "Total Non Active User :" $free
    echo -e "------------------------------------------------------"
    #sed -n '/free/p' /tmp/showdhcp.log > /tmp/Non_Active_User_$(date
        +"%Y-%m-%d__%H%M%S").log
    #sed -n '/active/p' /tmp/showdhcp.log > /tmp/Active_User_$(date
        +"%Y-%m-%d__%H%M%S").log
    rm $tempFile.*
    #rm -rvf /tmp/showdhcp.log
    #rm -rvf /tmp/tmp* &
    sleep 60
done
```

All that remains to be done regarding this section is to execute the script in background and detached from the current session in order for it to keep running even when the administrator is not logged in. Additionally, the standard output can be redirected to a file to see the current logs at any moment. A command like this would do the job:

```
$ sudo nohup ./dhcp_monitor.sh > dhcp_monitor.log &
```

With this final command this section is concluded and the DHCP is a little bit more secure than before.

4.5 Lightweight Directory Access Protocol (LDAP)

LDAP is a set of open source protocols that are used to access information that is centrally stored on a network. This protocol is used at the application level to access remote directory services. A remote directory is a set of objects that are organised hierarchically, such as name, key, addresses, etc. These objects will be available by a series of clients connected through a network, usually internal or LAN, and will provide the identities and permissions for those users who use them [11].

The currently installed software in charge of the LDAP service is actually not that bad in terms of security, but needs to be configured properly.

First of all, the installed version is from 2013 which means that there are plenty of well known vulnerabilities and their corresponding exploits as well. The list of vulnerabilities can be found here. Thankfully, there is no high risk vulnerability known for this specific version but these kind of services must be always up to date to prevent possible security flaws.

The LDAP service must implement a minimum level of security in communications between the LDAP server and the clients. Even if we are working on a test or development environment, it is essential to have a valid trusted certificate. This is because the *OpenLDAP* tool does not allow a self-signed certificate to be used on the server. In this way we have several alternatives: buy a certificate signed by a trusted entity; generate a free certificate with *certbot*; or create our own self-signed CA.

Creating our own CA has its advantages, since it allows us to generate any number of certificates and gives us greater flexibility, although it implies a greater amount of work when configuring the LDAP server. In this case though, since the certification authority has already been created for other services that need the same utility, this step will be assumed and skipped.

4.5.1 Solution: LDAP with TLS

The proposed solution to secure the LDAP Server is by using SSL/TLS certificates and keys given by a self signed SSL Certificate. The service will be configured to make use of a certificate generated through a Certification Authority of our own (similar to how OpenVPN does).

4.5.2 Chosen Technology: OpenLDAP with STARTTLS

As said before, the currently installed LDAP software is pretty good and will be a perfect fit for the requirements of this case. Some things need to be tweaked in order to fulfil the security requirements though. The first will be to get rid of the current deprecated version of OpenLDAP and install the latest stable version of it. Afterwards, the connection encryption will be performed using an extension for plain text communication protocols named *Opportunistic TLS* and referred as *STARTTLS* by OpenLDAP.

STARTTLS "upgrades" a non-encrypted connection by wrapping it with SSL/TLS after/during the connection process. This allows unencrypted and encrypted connections to be handled by the same port. This guide will utilise STARTTLS to encrypt connections.

4.5.3 Implementation

Similarly to other sections, this implementation won't show how the regular installation and configuration of OpenLDAP is done. What will be detailed is the process of configuring the LDAP connections with TLS. Another step that will be skipped is the certification authority setup, which is already defined in the Certification Authority section.

Before starting with the TLS configuration itself, the server must be set up so it correctly resolves its host name and fully qualified domain name (FQDN). This will be necessary in order for the certificates to be validated by clients. The FQDN chosen for this implementation is *ldapserver.tfm.davidhv*.

To begin with, the required packages need to be installed, as always. In this case those are the ones shown in the installation command shown below:

```
$ sudo apt install gnutls-bin ssl-cert
```

The first thing to do will be to create a template for the LDAP Server certificate that will be called *ldap_server.conf* and will be populated with the following lines:

```
organization = "Universitat de Barcelona"
cn = ldapserver.tfm.davidhv
tls_www_server
encryption_key
signing_key
expiration_days = 3652
```

Here, different pieces of information are issued. A part from the name of the organisation, which has obvious meaning, there are the *tls_www_server*, *encryption_key*, and *signing_key* options that will confer the certification with the basic functionalities it needs. The expiration date for the certificate is also set, concretely this will be a 10 year certificate to avoid having to manage frequent renewals.

Next, the LDAP Server will need a private key, so it will be issued using the *certtool* utility and stored in the */etc/ssl/private* for security purposes. The command is as follows:

```
$ sudo certtool -p --sec-param high --outfile /etc/ssl/private/ldap_server.key
```

Now, in order to generate the LDAP Server certificate some extra files will be needed. More precisely, the needed files will be the certification authority certificate and its key, found under the */etc/ssl/certs* and */etc/ssl/private* directories, respectively. Putting all together, the resulting command looks like this:

```
$ sudo certtool -c --load-privkey /etc/ssl/private/ldap_server.key
    --load-ca-certificate /etc/ssl/certs/ca_server.pem --load-ca-privkey
    /etc/ssl/private/ca_server.key --template
    /etc/ssl/templates/ldap_server.conf --outfile
    /etc/ssl/certs/ldap_server.pem
```

Afterwards, all these files need to have the proper permissions in order for the OpenLDAP process to access them. A group called *ssl-cert* already exists as the group owner of the */etc/ssl/private* directory. All that is needed is to add the *openldap* user to the *ssl-cert* group.

Now, the OpenLDAP configuration must be tweaked so the service uses the files created earlier. This will be achieved by creating an LDIF file with some configuration changes and loading it into the LDAP instance. This changes will be specified in a file called *tlsconf.ldif* which contents are the following:

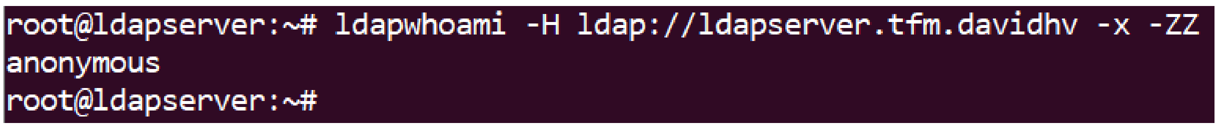
```
dn: cn=config
changetype: modify
add: olcTLSCACertificateFile
olcTLSCACertificateFile: /etc/ssl/certs/ca_server.pem
-
add: olcTLSCertificateFile
olcTLSCertificateFile: /etc/ssl/certs/ldap_server.pem
-
add: olcTLSCertificateKeyFile
olcTLSCertificateKeyFile: /etc/ssl/private/ldap_server.key
```

Then, the changes must be committed issuing the command shown below:

```
$ sudo ldapmodify -H ldapi:// -Y EXTERNAL -f tlsconf.ldif
```

To end with the configuration, the general LDAP configuration file needs to be modified in order to indicate it where is the CA certificate to be used for the TLS connections. Therefore, the *TLS_CACERT* entry of the */etc/ldap/ldap.conf* file will be tweaked to match the real path of the CA certificate file.

Finally, for testing purposes, the *ldapwhoami* command offers enough functionalities to perform a request to the LDAP server using TLS. Concretely, this is achieved by issuing the command shown below, where the *-ZZ* indicates the LDAP server to issue *STARTTLS* extended operation, requiring it to be successful.



```
root@ldapserver:~# ldapwhoami -H ldap://ldapserver.tfm.davidhv -x -ZZ
anonymous
root@ldapserver:~#
```

Figure 16: Successful TLS protected connection to the LDAP server.

4.6 HTTP Server

An HTTP server is, in this case, both a machine and a service that accepts HTTP requests from the network and provides HTTP responses to the client that made the request. These answers typically consist of an HTML document, but can also be simple text files, or other types of documents. Normally web servers can also record information about their use in a log file, which allows the web administrator to collect statistics using monitoring tools for this specific type of files [10].

In our scenario we have a web server used mainly by a team of employees within a local network, but must be secured against external threats as well. Web servers are one of the most targeted parts of an organisation's network, because of the sensitive data that they typically host. As a result, it's important that as well as securing web applications and your wider network, you take thorough measures to secure the web servers themselves.

There are several key threats to web servers that are important to be aware of, to prevent and mitigate those risks. These include, but are not limited to:

- **DoS and DDoS Attacks**: Denial of Service attacks and Distributed Denial of Service attacks are techniques cybercriminals will use to overwhelm your servers with traffic until they become unresponsive, rendering your website or network unusable.
- **SQL Injections**: SQL injections can be used to attack websites and web apps, by sending Structured Query Language requests through web forms to create, read, update, alter or delete data stored in your servers, such as financial information.
- **Unpatched software**: Software updates and security patches are designed to fix vulnerabilities in older versions of that software. However, once a new patch is released, would-be hackers can reverse-engineer attacks based on the changes, leaving unpatched versions in a vulnerable position. It's why we recommend using a trusted patch management service to make sure you're always up-to-date.
- **Cross-site scripting**: Cross-site scripting, also known as XSS, is a technique similar to an SQL injection. Code is injected into server-side scripts to gather sensitive data or to execute malicious client-side scripts.

However, one of the most prevalent threats to server security is human error or carelessness. Whether it's poorly-written code, easy-to-guess passwords, or a failure to install and update firewalls and other security software, the human element in cybersecurity is typically the weakest link.

4.6.1 Solution: HTTPS + NIDS

A standard HTTP connection on the Internet can easily be hijacked by unauthorised parties, performing an interception of the transmitted data and thus allowing for targeted attacks on individuals. The data entered by a user in their browser window is often personal (account information, email, credit card information, etc.) and must be protected from such access. The purpose of an HTTPS connection is to avoid this by means of encrypting data to ensure secure data transmission. The resulting transmission is thus encrypted and the server authenticated.

Additionally, a Network Intrusion Detection System (NIDS) will be implanted. NID Systems perform an analysis of passing traffic on the entire sub-net, and matches the traffic that is passed on the sub-nets to the library of known attacks. Once an attack is identified, or abnormal behaviour is sensed, the alert can be sent to the administrator.

4.6.2 Chosen Technology: Apache and Snort

The question here was, Apache or Nginx? Since they are both pretty much equal in terms of security, the decision has been made in front of other aspects that are relevant for the project's particular use case. To summarise, Nginx yields better than Apache under heavy load while consuming less resources than its counterpart, but on the other hand, Apache provides great flexibility, allowing for connection customisation and admitting more than 60 modules that increase its functionalities even more. Being this said, the chosen option has been to use Apache due to its high flexibility and its customisation ability.

Snort is a network intrusion detection system that has the capacity to perform real-time traffic analysis and packet logging on Internet Protocol (IP) networks. Snort performs protocol analysis, content searching and matching. The program can also be used to detect probes or attacks, including, but not limited to, operating system fingerprinting attempts, semantic URL attacks, buffer overflows, server message block probes, and stealth port scans.. Snort uses a series of rules that help define malicious network activity and uses those rules to find packets that match against them and generates alerts for users.

4.6.3 Implementation

As usual, the first step is to install the software package, this time by issuing the command:

```
$ sudo apt install apache2
```

Once installed, the first thing that needs to be tweaked is the system's firewall so as to permit Apache's access through it. This is easily achieved with the next command, which indicates the keyword *Apache Secure* thus only allowing TLS encrypted traffic through port 443:

```
$ sudo ufw allow "Apache Secure"
```

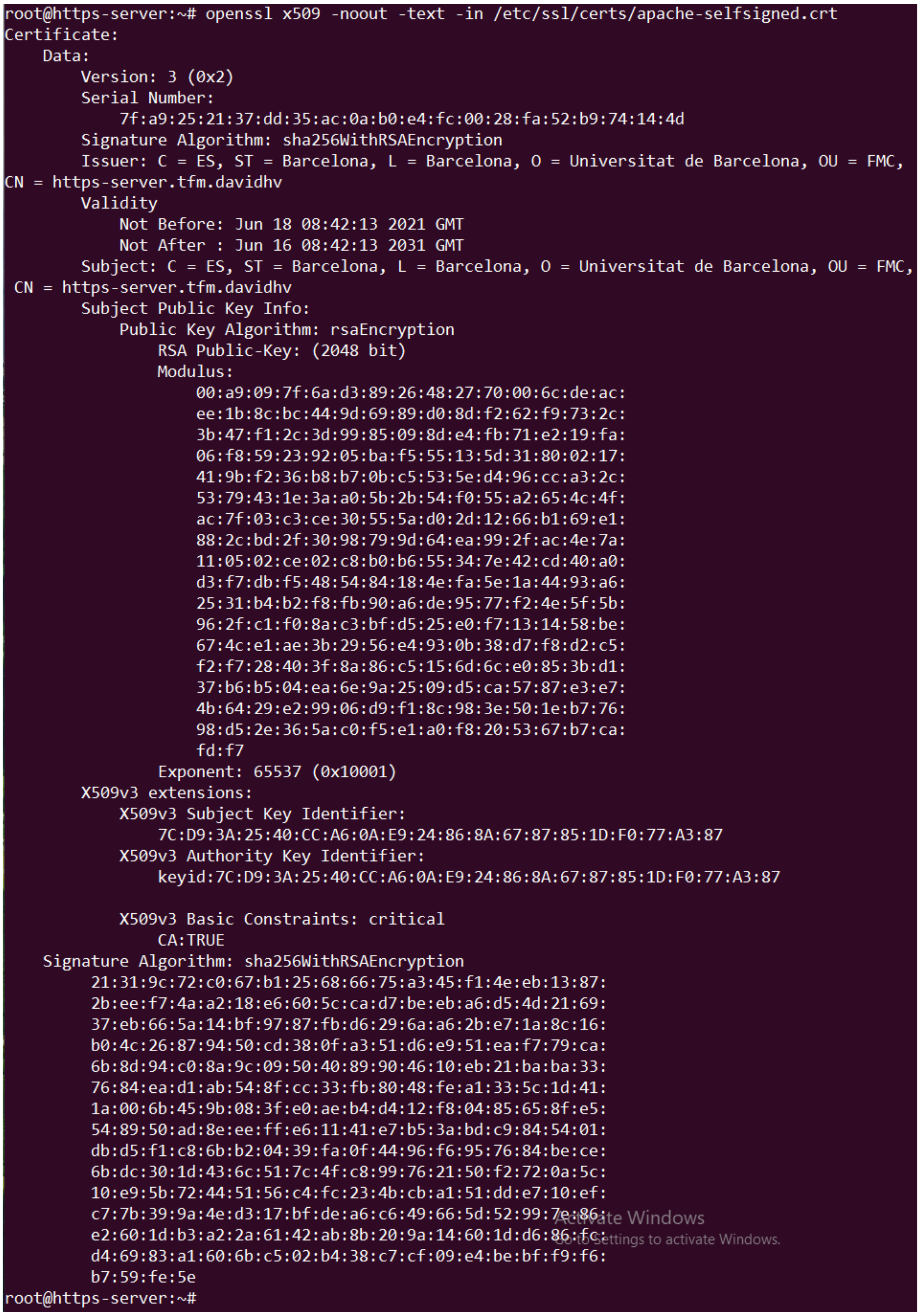
Before starting with the TLS configuration itself, a particular module need to be enabled. The *mod_ssl* provides support for TLS encryption thus allowing the usage of TLS certificates. After that, the service must be reloaded for the changes to take effect.

```
$ sudo a2enmod ssl
$ sudo apache2 restart
```

Now we can create a self signed certificate for the test web server, named as *https-server.tfm.davidhv*.

```
$ sudo openssl req -x509 -nodes -days 3650 -newkey rsa:2048 -keyout
    /etc/ssl/private/apache-selfsigned.key -out
    /etc/ssl/certs/apache-selfsigned.crt
```

The resulting certificate contents can be checked by issuing the next command:



```
root@https-server:~# openssl x509 -noout -text -in /etc/ssl/certs/apache-selfsigned.crt
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            7f:a9:25:21:37:dd:35:ac:0a:b0:e4:fc:00:28:fa:52:b9:74:14:4d
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: C = ES, ST = Barcelona, L = Barcelona, O = Universitat de Barcelona, OU = FMC,
CN = https-server.tfm.davidhv
        Validity
            Not Before: Jun 18 08:42:13 2021 GMT
            Not After : Jun 16 08:42:13 2031 GMT
        Subject: C = ES, ST = Barcelona, L = Barcelona, O = Universitat de Barcelona, OU = FMC,
 CN = https-server.tfm.davidhv
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                RSA Public-Key: (2048 bit)
                Modulus:
                    00:a9:09:7f:6a:d3:89:26:48:27:70:00:6c:de:ac:
                    ee:1b:8c:bc:44:9d:69:89:d0:8d:f2:62:f9:73:2c:
                    3b:47:f1:2c:3d:99:85:09:8d:e4:fb:71:e2:19:fa:
                    06:f8:59:23:92:05:ba:f5:55:13:5d:31:80:02:17:
                    41:9b:f2:36:b8:b7:0b:c5:53:5e:d4:96:cc:a3:2c:
                    53:79:43:1e:3a:a0:5b:2b:54:f0:55:a2:65:4c:4f:
                    ac:7f:03:c3:ce:30:55:5a:d0:2d:12:66:b1:69:e1:
                    88:2c:bd:2f:30:98:79:9d:64:ea:99:2f:ac:4e:7a:
                    11:05:02:ce:02:c8:b0:b6:55:34:7e:42:cd:40:a0:
                    d3:f7:db:f5:48:54:84:18:4e:fa:5e:1a:44:93:a6:
                    25:31:b4:b2:f8:fb:90:a6:de:95:77:f2:4e:5f:5b:
                    96:2f:c1:f0:8a:c3:bf:d5:25:e0:f7:13:14:58:be:
                    67:4c:e1:ae:3b:29:56:e4:93:0b:38:d7:f8:d2:c5:
                    f2:f7:28:40:3f:8a:86:c5:15:6d:6c:e0:85:3b:d1:
                    37:b6:b5:04:ea:6e:9a:25:09:d5:ca:57:87:e3:e7:
                    4b:64:29:e2:99:06:d9:f1:8c:98:3e:50:1e:b7:76:
                    98:d5:2e:36:5a:c0:f5:e1:a0:f8:20:53:67:b7:ca:
                    fd:f7
                Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Subject Key Identifier:
                7C:D9:3A:25:40:CC:A6:0A:E9:24:86:8A:67:87:85:1D:F0:77:A3:87
            X509v3 Authority Key Identifier:
                keyid:7C:D9:3A:25:40:CC:A6:0A:E9:24:86:8A:67:87:85:1D:F0:77:A3:87

            X509v3 Basic Constraints: critical
                CA:TRUE
    Signature Algorithm: sha256WithRSAEncryption
         21:31:9c:72:c0:67:b1:25:68:66:75:a3:45:f1:4e:eb:13:87:
         2b:ee:f7:4a:a2:18:e6:60:5c:ca:d7:be:eb:a6:d5:4d:21:69:
         37:eb:66:5a:14:bf:97:87:fb:d6:29:6a:a6:2b:e7:1a:8c:16:
         b0:4c:26:87:94:50:cd:38:0f:a3:51:d6:e9:51:ea:f7:79:ca:
         6b:8d:94:c0:8a:9c:09:50:40:89:90:46:10:eb:21:ba:ba:33:
         76:84:ea:d1:ab:54:8f:cc:33:fb:80:48:fe:a1:33:5c:1d:41:
         1a:00:6b:45:9b:08:3f:e0:ae:b4:d4:12:f8:04:85:65:8f:e5:
         54:89:50:ad:8e:ee:ff:e6:11:41:e7:b5:3a:bd:c9:84:54:01:
         db:d5:f1:c8:6b:b2:04:39:fa:0f:44:96:f6:95:76:84:be:ce:
         6b:dc:30:1d:43:6c:51:7c:4f:c8:99:76:21:50:f2:72:0a:5c:
         10:e9:5b:72:44:51:56:c4:fc:23:4b:cb:a1:51:dd:e7:10:ef:
         c7:7b:39:9a:4e:d3:17:bf:de:a6:c6:49:66:5d:52:99:7e:86:
         e2:60:1d:b3:a2:2a:61:42:ab:8b:20:9a:14:60:1d:d6:86:fc:
         d4:69:83:a1:60:6b:c5:02:b4:38:c7:cf:09:e4:be:bf:f9:f6:
         b7:59:fe:5e
root@https-server:~#
```

Figure 17: Dump of the HTTPS web server certificate contents.

After that, the Apache configuration must be updated so as to use the self signed certificate and key generated earlier. In the distribution used in this project, the Apache configuration files are stored under the */etc/apache2/sites-available* directory, where the specific file for the test web must be created. The file should be called as the domain, in this case it will be *https-server.tfm.davidhv.conf* and its contents should be the ones below:

```
<VirtualHost *:443>
   ServerName https-server.tfm.davidhv
   DocumentRoot /var/www/https-server.tfm.davidhv

   SSLEngine on
   SSLCertificateFile /etc/ssl/certs/apache-selfsigned.crt
   SSLCertificateKeyFile /etc/ssl/private/apache-selfsigned.key
</VirtualHost>

<VirtualHost *:80>
    ServerName https-server.tfm.davidhv
    Redirect / https://https-server.tfm.davidhv/
</VirtualHost>
```

Notice that it does not only respond to HTTPS requests but to HTTP ones as well. The trick is that when an unencrypted connection to the site is requested it will be redirected to the secure version of it, thus making the web site accessible from both port 443 and 80. To fully allow this feature though, the firewall must be set accordingly with Apache to permit those connections.

```
$ sudo ufw allow "Apache Full"
```

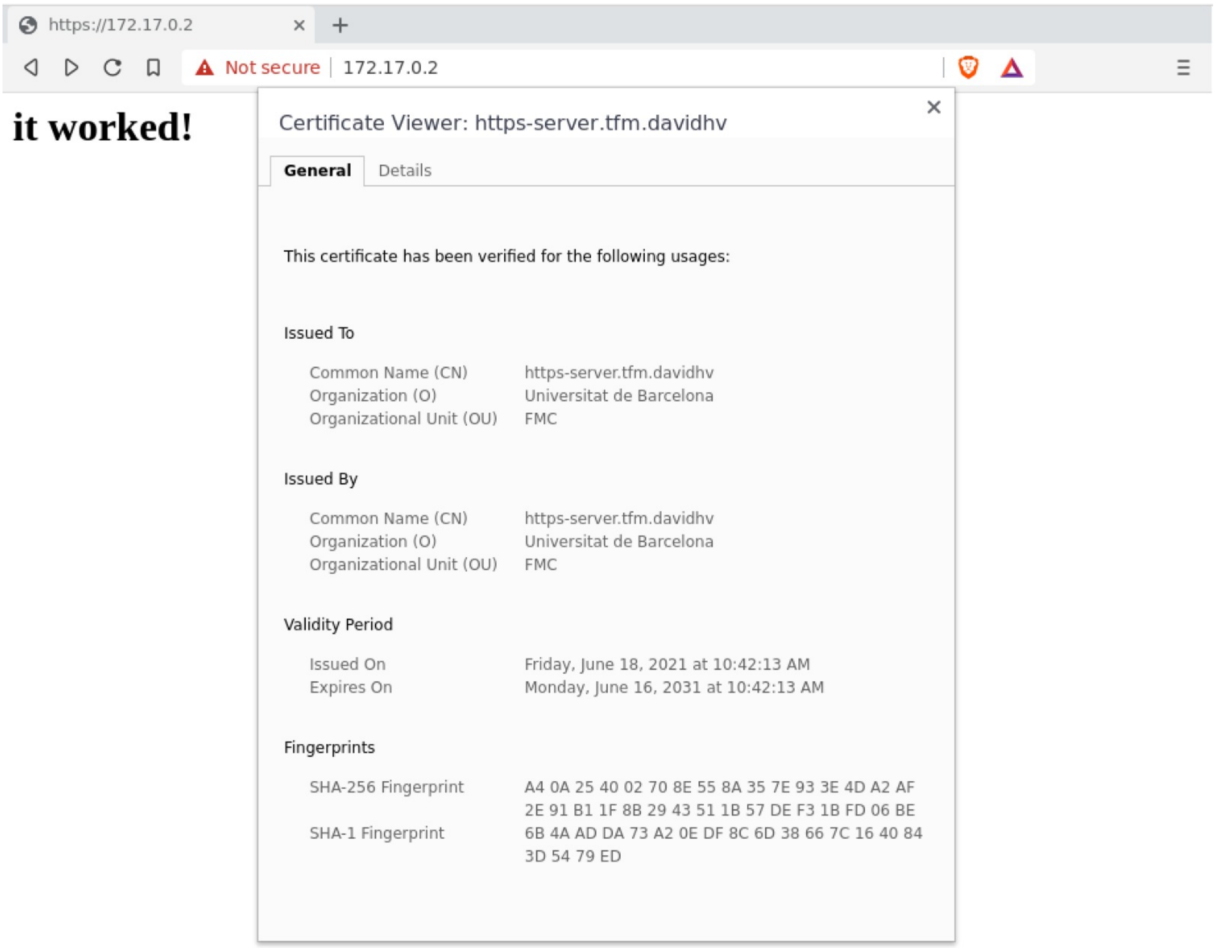
Afterwards, just for testing purposes, we can create a sample *index.html* file into */var/www /https-server.tfm.davidhv/* containing a message that will be displayed when the web site is requested. In this case its contents are as simple as:

```
<h1>it worked!</h1>
```

To end with the configuration, all that remains to be done is to enable the created web site and reload the service as follows:

```
$ sudo a2ensite https-server.tfm.davidhv.conf
$ sudo service apache2 restart
```

Once the service is up again the web site can be accessed through HTTPS and the message contained in the *index.html* file should be displayed. Nonetheless, the browser will likely raise a warning, telling the user that the place is not secure due to the repudiation of the certificate authority used for the HTTPS certificate. In this case, since it has been signed by ourselves we can safely assume the ”risk” and proceed to the site. The following figure shows the working website alongside with its certificate details.


https://172.17.0.2

Not secure | 172.17.0.2

it worked!

Certificate Viewer: https-server.tfm.davidhv

General Details

This certificate has been verified for the following usages:

Issued To

Common Name (CN)	https-server.tfm.davidhv
Organization (O)	Universitat de Barcelona
Organizational Unit (OU)	FMC

Issued By

Common Name (CN)	https-server.tfm.davidhv
Organization (O)	Universitat de Barcelona
Organizational Unit (OU)	FMC

Validity Period

Issued On	Friday, June 18, 2021 at 10:42:13 AM
Expires On	Monday, June 16, 2031 at 10:42:13 AM

Fingerprints

SHA-256 Fingerprint	A4 0A 25 40 02 70 8E 55 8A 35 7E 93 3E 4D A2 AF 2E 91 B1 1F 8B 29 43 51 1B 57 DE F3 1B FD 06 BE
SHA-1 Fingerprint	6B 4A AD DA 73 A2 0E DF 8C 6D 38 66 7C 16 40 84 3D 54 79 ED

Figure 18: Successful HTTPS connection to the Apache web server.

With this, the HTTPS configuration on the Apache web server is finished.

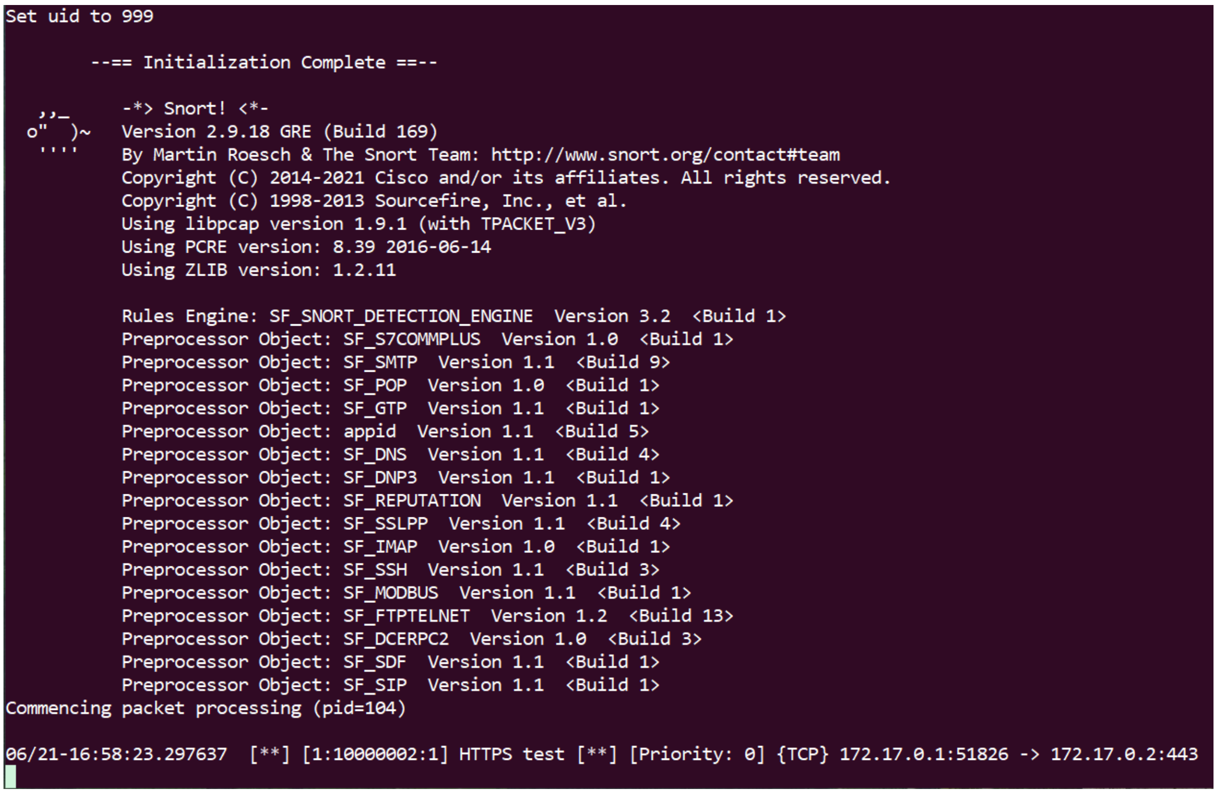
On the other hand, a NIDS will be deployed in order to monitor the traffic in search of weird behaviour that could denote the presence of an intruder. In this case it needs to be implemented on the same machine which provides the web service because otherwise the NIDS wouldn't be able to see the contents of any packet given the fact that the web server uses TLS encrypted connections through HTTPS.

After the installation and proper configuration of Snort, which won't be detailed in this implementation due to the lack of security-related steps of it (and also not precisely short), the testing procedure can begin.

From another machine, we can now test Snort by requesting the web page configured earlier through this command:

```
$ wget -q -O- --no-check-certificate https://https-server.tfm.davidhv
```

Which will output the contents of the *index.html* file, but what we are interested in right now is the output of the Snort process running on the server machine, which is the following:



```
Set uid to 999

        --== Initialization Complete ==--

   ,,_     -*> Snort! <*-
  o"  )~   Version 2.9.18 GRE (Build 169)
   ''''    By Martin Roesch & The Snort Team: http://www.snort.org/contact#team
           Copyright (C) 2014-2021 Cisco and/or its affiliates. All rights reserved.
           Copyright (C) 1998-2013 Sourcefire, Inc., et al.
           Using libpcap version 1.9.1 (with TPACKET_V3)
           Using PCRE version: 8.39 2016-06-14
           Using ZLIB version: 1.2.11

           Rules Engine: SF_SNORT_DETECTION_ENGINE  Version 3.2  <Build 1>
           Preprocessor Object: SF_S7COMMPLUS  Version 1.0  <Build 1>
           Preprocessor Object: SF_SMTP  Version 1.1  <Build 9>
           Preprocessor Object: SF_POP  Version 1.0  <Build 1>
           Preprocessor Object: SF_GTP  Version 1.1  <Build 1>
           Preprocessor Object: appid  Version 1.1  <Build 5>
           Preprocessor Object: SF_DNS  Version 1.1  <Build 4>
           Preprocessor Object: SF_DNP3  Version 1.1  <Build 1>
           Preprocessor Object: SF_REPUTATION  Version 1.1  <Build 1>
           Preprocessor Object: SF_SSLPP  Version 1.1  <Build 4>
           Preprocessor Object: SF_IMAP  Version 1.0  <Build 1>
           Preprocessor Object: SF_SSH  Version 1.1  <Build 3>
           Preprocessor Object: SF_MODBUS  Version 1.1  <Build 1>
           Preprocessor Object: SF_FTPTELNET  Version 1.2  <Build 13>
           Preprocessor Object: SF_DCERPC2  Version 1.0  <Build 3>
           Preprocessor Object: SF_SDF  Version 1.1  <Build 1>
           Preprocessor Object: SF_SIP  Version 1.1  <Build 1>
Commencing packet processing (pid=104)

06/21-16:58:23.297637  [**] [1:10000002:1] HTTPS test [**] [Priority: 0] {TCP} 172.17.0.1:51826 -> 172.17.0.2:443
```

Figure 19: Successful HTTPS packet interception (bottom) requesting the test web page.

As it can be seen in the last line of the figure above, the request to the test web page is seen (and logged) by Snort, which after some time will be capable of deciding whether this kind of connections are legit or not.

4.7 Backup

There is no doubt cloud services have saved us a lot of trouble. We can back up important files in cloud storage and download them later whenever needed. But backing up the entire operating system is not efficient enough with cloud services. An operating system loaded with a ton of applications can be more than 100GB in size. Or, like in this case, a Linux server which stores, updates, moves, and deletes files almost constantly, it is just not a good choice [13].

When running a server, the administrator must be ready for any disaster such as server crash that may corrupt data, accidentally deleting important configuration files, or in the worst scenario, the server may get hacked.

Uploading system files to the cloud will not only consume time but will require the admin to sign up for a paid plan that provides enough storage for the backup since there is no such platform that allows these amounts for free. Google drive, Gdrive, Dropbox, etc. are not designed for systematically storing backups. So even if we could successfully package our operating system and upload it to the server, restoring that backup manually would not be easy.

4.7.1 Solution: Local Backup using a NAS

The inconveniences the cloud services present are the reason why the best way to perform the backup is locally. Most Linux distributions come with a built-in tool that allows for backup configurations. This approach is the one that will be used for this project. Additionally, since the department disposes of a network attached storage machine with massive disk space the service will be configured to store the backup there [12].

4.7.2 Chosen Technology: Rsync

Rsync is a fast and versatile command-line utility for synchronising files and directories between two locations over a remote shell, or from/to a remote/local Rsync daemon. It provides fast incremental file transfer by transferring only the differences between the source and the destination. Rsync can be used for mirroring data, incremental backups and copying files between systems.

Rsync allows for much more customisation than any other on the list of backup tools. With a previous knowledge of bash scripting it is very simple to deploy and tweak a backup configuration with rsync. It allows to do anything on the file system and either transfer the output to any remote server or save it locally.

With Rsync, the backup service will be configured to make a differential backup of the system and users' files every day and a full backup once a week. This decision has been made in front of two facts: first, the changes to data will generally be few compared to the entire amount of data in the system; second, regarding data restoration time, at most two backup media are ever needed to restore all the data, which compared with the incremental backup offers quicker recovery time, requiring only a full backup and the last differential backup to restore the entire data.

Moreover, despite specifying that the backup will be done locally this will not be entirely true in the implementation. It is stated as locally because it won't be using a third party online service such as any cloud storage mentioned in the previous subsection. But actually the backup won't be stored in the same server machine, instead it will be cached in a network attached system (NAS) owned by the department.

4.7.3 Implementation

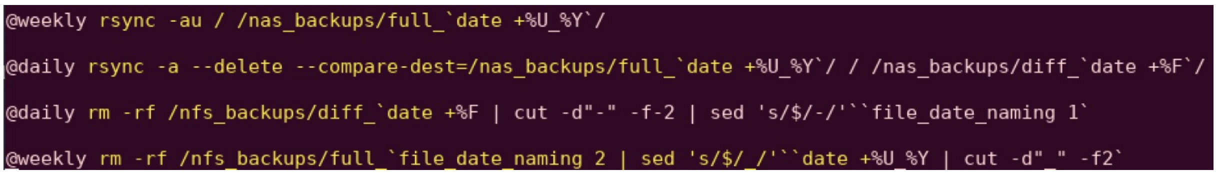
The implementation of this service begins just as the others, installing the required packages if needed. In most Linux distributions this won't be the case, and the Ubuntu Server used for this project is not the exception.

So, assuming rsync is installed in the system, alongside with NFS and the linux *cron* tool, the backups will be performed as follows. First of all, since rsync does not allow to compare files on remote machines in order to make the differential backup, it is needed to have the remote NAS's backup directory mounted locally. This is achieved with the help of the network file system (NFS) protocol. This protocol allows for a file system to be mounted over the network within another machine. This way the path specified to rsync to compare the files with will be "local".

```
$ sudo mkdir /nas_backups
$ sudo mount -t nfs <NAS IP address>:/backups /nas_backups
```

Secondly, in order to automatise the backups, Linux provides a tool called crontab, which allows the administrator to define recurrent commands to be executed when desired. In this case, it is needed to make the differential backup on a daily basis and a full backup once a week. To fulfil that requisite, crontab defines easy macros that satisfy the most used ones, which include both of the two needed. More specifically this is realised by means of typing *@daily* and *@weekly* before the command specification.

Therefore, the rsync commands must be included in the crontab configuration, which will look like this:



```
@weekly rsync -au / /nas_backups/full_`date +%U_%Y`/
@daily rsync -a --delete --compare-dest=/nas_backups/full_`date +%U_%Y`/ / /nas_backups/diff_`date +%F`/
@daily rm -rf /nfs_backups/diff_`date +%F | cut -d"-" -f-2 | sed 's/$/-/'``file_date_naming 1`
@weekly rm -rf /nfs_backups/full_`file_date_naming 2 | sed 's/$/_/'``date +%U_%Y | cut -d"_" -f2`
```

Figure 20: Scheduled jobs in crontab file for system backup.

As shown in the above code, the backup is broken down into two rsync commands. The first command creates a full backup every Sunday with the suffix of the current week of the year and the current year; and the second command takes care of the differential backup by means of comparing the files found in the current week full backup and stores it into the directory with the same suffix basis explained before.

The flags used for the commands have nothing special except for -*a*, which is responsible of several tasks such as recurse into directories, copy symbolic links as symbolic links (not as regular files) and preserve permissions, modification times, group, owner, device files and special files. The other one is just for skipping files that are newer on the destination.

Additionally, two more jobs have been defined in order to delete those backups that become useless as time passes by. More concretely, the third command shown in Scheduled jobs in crontab file for system backup 20 is responsible of deleting the backup of the previous day once the differential backup of the current day is finished and the last command takes care of deleting full system backups from two months back. Both of them use a command called *file_date_naming* which calculates the name of the backup directory that must be erased. It has two different modes of operation, the first is specified with a 1 and refers to the differential backup whereas the other mode, referenced with a 2 takes care of the full system backup naming. This script has been deliberately developed for this specific need of the project. The source code of that binary is the following:

```
#!/bin/bash

if [ $# -ne 1 ]
then
   echo "Usage: file_date_naming <mode number>"
   echo "(Mode 1: Differential Backup; Mode 2: Full Backup)"
   exit 1
fi

if [ $1 -eq 1 ]
then
   day=‘date +%d‘
   if [ $day -lt 11 ] && [ $day -gt 1 ]
   then
      echo "$day-1" | bc | sed 's/^/0/'
   elif [ $day -eq 1 ]
   then
      month=‘date +%m‘
      if [ $month -eq 3 ]
      then
         echo 28
      elif [ $month -eq 5 ] || [ $month -eq 7 ] || [ $month -eq 10 ] || [
          $month -eq 12 ]
      then
         echo 30
      else
         echo 31
      fi
   else
      echo "$day-1" | bc
   fi
elif [ $1 -eq 2 ]
```

```
then
   week=`date +%U`
   if [ $week -lt 18 ] && [ $week -gt 7 ]
   then
      echo "$week-8" | bc | sed 's/^/0/'
   elif [ $week -lt 8 ]
   then
      echo "53-(7-$week)" | bc
   else
      echo "$week-8" | bc
   fi
else
   echo The specified mode does not exist
   exit 1
fi

exit 1
```

With all this, the backup configuration is finished and we can breath a little bit easier in case of a system failure/hijack.

4.8 Network Monitoring

Network monitoring is a critical component of a network management strategy that provides valuable insights into network-related problems which can affect the organisation. When networks are monitored regularly, risks like overloaded networks, router problems, downtime, cybercrime, and data loss can be mitigated more easily.

By monitoring the network we can be aware of relevant information like identification of over-used and under-used network elements, cybersecurity threats or minor network faults that could turn into significant problems [14].

4.8.1 Solution: Passive Server Monitoring

Since we don't want to affect the performance of the network at all we will be using a passive network monitoring tool. Passive network monitoring refers to capturing network traffic that flow through a network and analysing it afterwards. Through a collection method like log management or network taps, passive monitoring compiles historic network traffic to paint a bigger picture of a company's network performance.

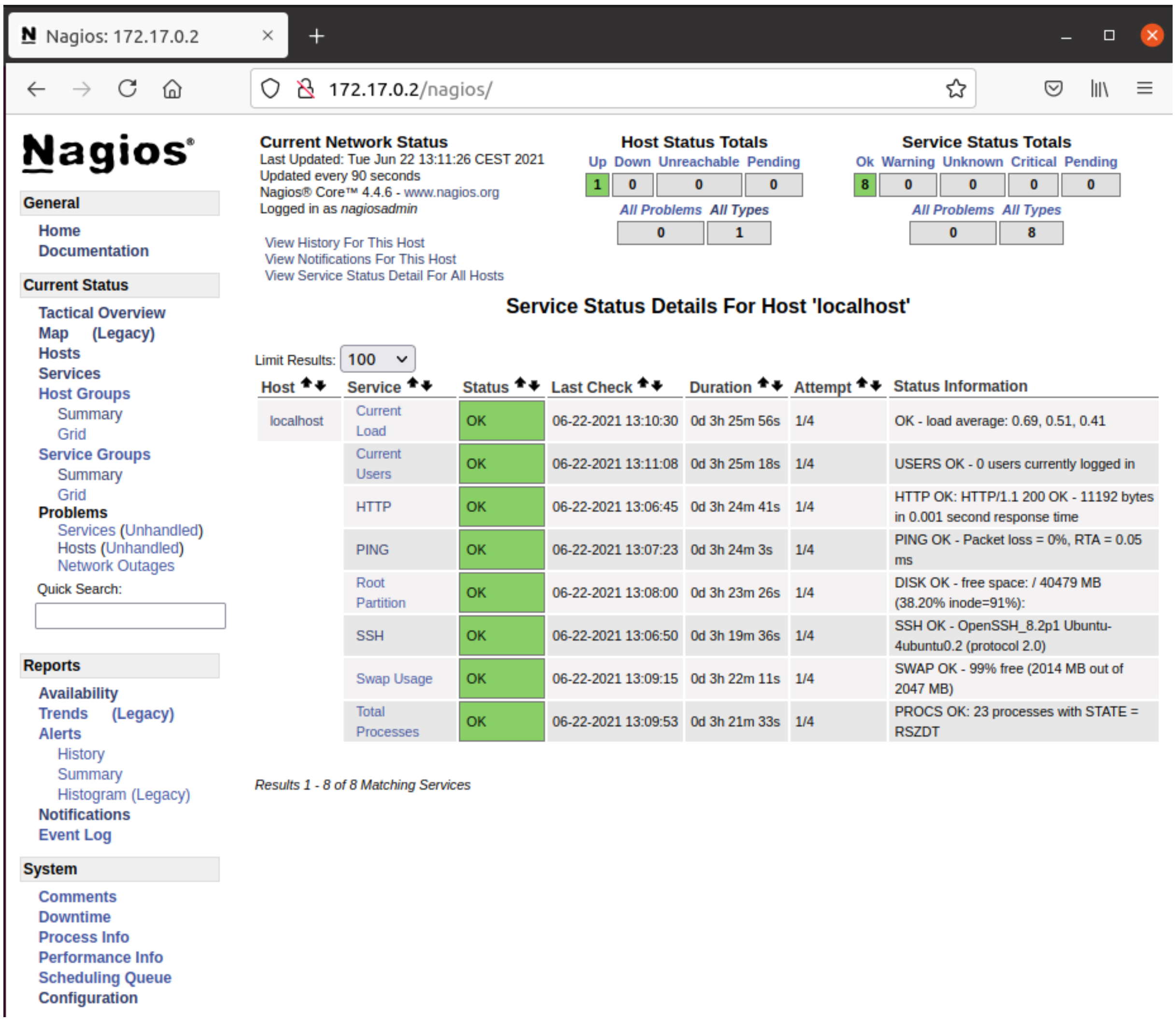
4.8.2 Chosen Technology: Nagios

Nagios is a network monitoring tool that also allows to monitor specific machines and services and alerts when their behaviour is beyond the rule. Nagios is a very powerful tool since it not only provides monitoring and alerting but also offers reporting of historical events, a maintenance mode to prevent from receiving alerts when the equipment is being upgraded, and trending and capacity planning graphs and reports that help the administrator to identify necessary infrastructure upgrades before failures occur.

4.8.3 Implementation

The implementation for this service has nothing special, just the regular installation of the program following the guideline provided by the developers. One particular thing to take into account though is that the notification built-in service comes with a default email that is not presented to modify during the installation process. This can be modified in the */usr/local/nagios/etc/objects/contacts.cfg* file.

The following figure shows the fully working Nagios software on the server machine, with the installed services under Nagios' monitoring.



Current Network Status
Last Updated: Tue Jun 22 13:11:26 CEST 2021
Updated every 90 seconds
Nagios® Core™ 4.4.6 - www.nagios.org
Logged in as *nagiosadmin*

View History For This Host
View Notifications For This Host
View Service Status Detail For All Hosts

Host Status Totals

Up	Down	Unreachable	Pending
1	0	0	0

All Problems	*All Types*
0	1

Service Status Totals

Ok	Warning	Unknown	Critical	Pending
8	0	0	0	0

All Problems	*All Types*
0	8

Service Status Details For Host 'localhost'

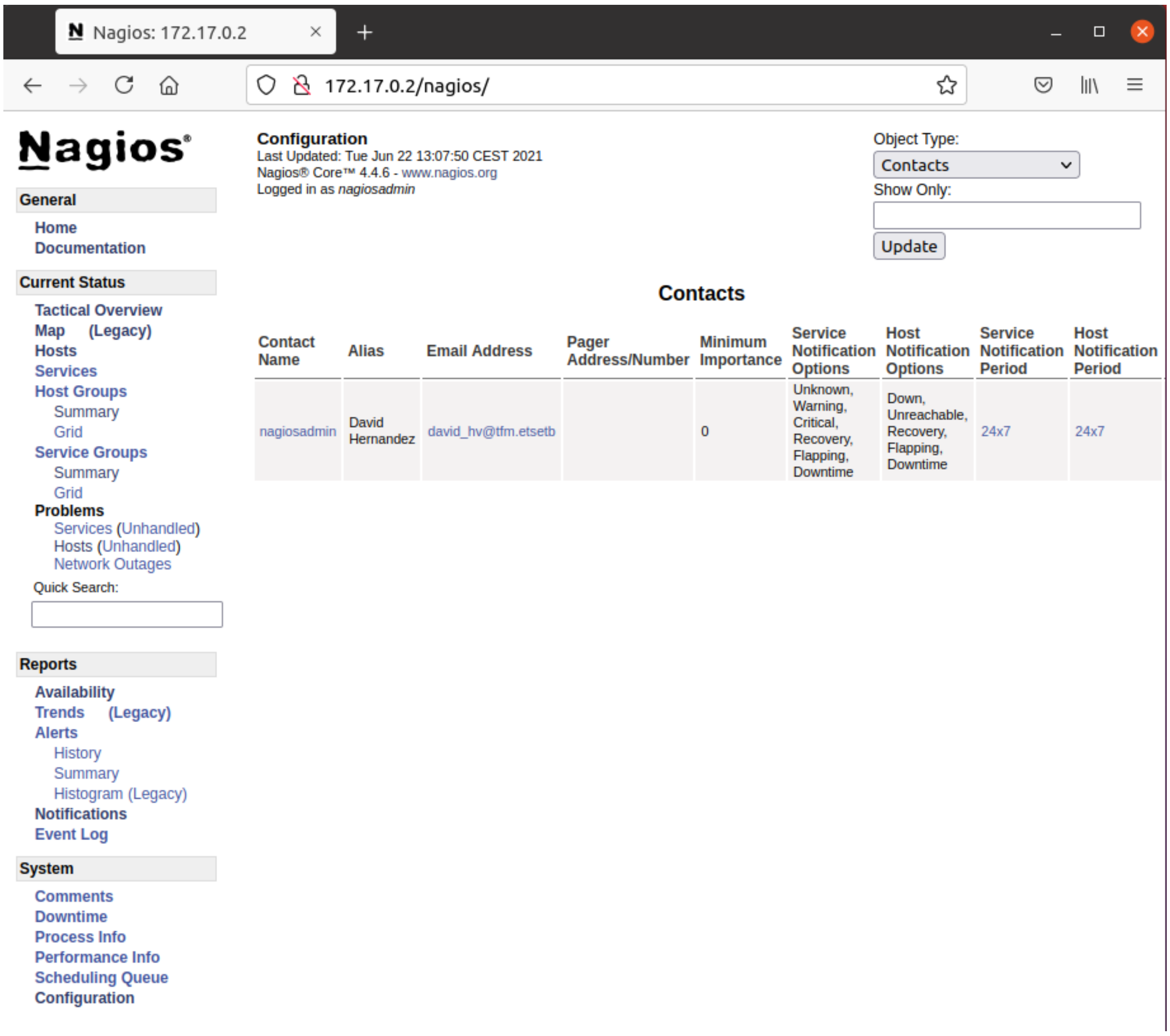
Limit Results: 100

Host	Service	Status	Last Check	Duration	Attempt	Status Information
localhost	Current Load	OK	06-22-2021 13:10:30	0d 3h 25m 56s	1/4	OK - load average: 0.69, 0.51, 0.41
	Current Users	OK	06-22-2021 13:11:08	0d 3h 25m 18s	1/4	USERS OK - 0 users currently logged in
	HTTP	OK	06-22-2021 13:06:45	0d 3h 24m 41s	1/4	HTTP OK: HTTP/1.1 200 OK - 11192 bytes in 0.001 second response time
	PING	OK	06-22-2021 13:07:23	0d 3h 24m 3s	1/4	PING OK - Packet loss = 0%, RTA = 0.05 ms
	Root Partition	OK	06-22-2021 13:08:00	0d 3h 23m 26s	1/4	DISK OK - free space: / 40479 MB (38.20% inode=91%):
	SSH	OK	06-22-2021 13:06:50	0d 3h 19m 36s	1/4	SSH OK - OpenSSH_8.2p1 Ubuntu-4ubuntu0.2 (protocol 2.0)
	Swap Usage	OK	06-22-2021 13:09:15	0d 3h 22m 11s	1/4	SWAP OK - 99% free (2014 MB out of 2047 MB)
	Total Processes	OK	06-22-2021 13:09:53	0d 3h 21m 33s	1/4	PROCS OK: 23 processes with STATE = RSZDT

Results 1 - 8 of 8 Matching Services

Figure 21: Nagios web page showing local host's services.

And the configuration page which shows the modified administrator's details, presented below:



Configuration
Last Updated: Tue Jun 22 13:07:50 CEST 2021
Nagios® Core™ 4.4.6 - www.nagios.org
Logged in as *nagiosadmin*

Contacts

Contact Name	Alias	Email Address	Pager Address/Number	Minimum Importance	Service Notification Options	Host Notification Options	Service Notification Period	Host Notification Period
nagiosadmin	David Hernandez	david_hv@tfm.etsetb		0	Unknown, Warning, Critical, Recovery, Flapping, Downtime	Down, Unreachable, Recovery, Flapping, Downtime	24x7	24x7

Figure 22: Nagios web page showing administrator's details.

4.9 Cybersecurity Training Session for Employees

After all the software-related security, finally comes the section which is going to make a real difference with other companies in terms of security. An organisation's employees are one of the biggest risks to its cybersecurity. In fact, human error is considered the leading cause of data breaches. However, organisation's employees can also be a huge asset for an the company's cybersecurity. If employees are provided with the knowledge they require to identify cyberthreats -through an effective and engaging security training program- they can act as another line of defence for an organisation [15].

4.9.1 Implementations

For this purpose, a training session has been developed in which the employees will have to pass a cybersecurity evaluation. The program is designed as follows:

1. First, a set of slides has been constituted in which some of the most relevant security threats are shown and their correspondent avoidance method is taught. The format for this presentation will be with a video of myself teaching it (about 20 minutes, out of the scope of this thesis).
2. In a second place, after watching the video of the session, employees will have to prove themselves that they have learnt the knowledge instructed there. This will be achieved with a test created for this specific purpose that employees will have to pass with a minimum mark of 7.5.
3. Finally, if the employee successfully completes the test, they will be given access and credentials to the department's resources. On the contrary, if the employee doesn't overcome the examination they will have to get back to step 1 and start over the training.

Among the goals of this master thesis was not only to secure systems but also to educate the staff in the security field. A secure system will not be complete if its users are not aware of the risks, the security measures in place, and how to use them.

The creation of a training course is one of the most important final steps of this project. The initial set of slides has already been created and available here, but more is about to come. We are also preparing video lessons for the different topics. The plan is to initiate the training is the department's personnel in September/October, once everyone is back from holidays, and the technical solutions are moved to production servers.

5 Budget

5.1 Cost of the Gantt's activities

All items are computed following the next rule: total hours spent in the activity * 12 euros / hour * 1.35 National Insurance. This breaks down as follows:

- Methodology: 10 hours = 162 euros.
- Analysis: 20 hours = 324 euros.
- Diagnostic: 50 hours = 810 euros.
- Descriptions: 40 hours = 648 euros.
- Solutions: 80 hours = 1296 euros.
- Implementations: 100 hours = 1620 euros.
- Conclusions: 10 hours = 162 euros.

Total = 5022 euros.

It needs to be clarified that the project is carried out by a single person and take on a unique role.

5.2 Generic costs

As mentioned in previous sections, this project could have been carried out without any adversity with the current computational resources of the department, but nevertheless a new machine has been acquired for the fulfilment of this project, and the total cost of it has been of 2662 euros.

5.3 Grand Total

Adding both the generic costs and the activities costs the cost of the master thesis amounts to a total of 7684 euros.

6 Conclusions and future development

In this master's thesis project we have implemented several features that add some extra layers in terms of the department's security. This has been carried out mainly by analysing and diagnosing the department's resources, designing and acquiring hardware for this purpose, securing all the software services that the department needs, but also by increasing the employees' awareness of information security. All this work will provide the department with a robust defence against cyber attacks and hopefully will make it not to be a victim of hijacks never again.

Once the project has been completed, it can be stated that the goals that had been set have been achieved. First, the main objective, which was to implement security features on the main services of the department, has been successfully carried out. The systems have been designed to be cryptographically agnostic, i.e. that although it now uses a specific set of cipher suites, it will be able to switch to new ones in the future if required. Second, the integration of those services with the current working ones is still yet to be performed. This means that we currently have a clear method of installing and configuring those services but we still have to encounter so much more adversities. As explained, this project has been more focused on securing the services, but the other elements, like infrastructure security or employees training, are just as or more important than the former one; and although this project will be completed in terms of the master thesis, much work remains to be done and will continue to be implemented over the next few months.

Obviously, the solutions reflected in this thesis have not been easy to choose nor to implement in some cases. In most of the implementations the final solution has not been the same as the initial proposal, which almost always has lead to uncountable hours of research and failed accomplishments. Besides, those frustrations have been very helpful since their fulfilment have contributed to something even more important than the knowledge of what is correct: what is not.

On a personal level, the development of this project has allowed me to grow a lot as a cybersecurity specialist, especially as a systems designer. I have been surrounded by very professional people, in the fields of engineering and science, and I have learned a lot from them. In addition, this job has given me the opportunity to enter the world of systems administration and cybersecurity in a realistic way, and at the same time the hours I have spent in the company have served me as a professional experience, which I value enormously. Thanks to the project I have been able to see the real applications of many of the knowledge learned during the master; especially of subjects like Network Security, Malware and Traffic Monitoring and Analysis.

As a general conclusion, this master thesis project has brought me valuable skills and knowledge, and so much priceless failures which I learnt from; for I am very proud of it and I expect to be able to apply all the knowledge acquired here in a company's cybersecurity specialised department.

References

[1] Carla Carvalho and Eduardo Marques. Adapting iso 27001 to a public institution. In *2019 14th Iberian Conference on Information Systems and Technologies (CISTI)*, pages 1–6, 2019.

[2] Bayona Sussy, Chauca Wilber, Lopez Milagros, and Maldonado Carlos. Iso/iec 27001 implementation in public organizations: A case study. In *2015 10th Iberian Conference on Information Systems and Technologies (CISTI)*, pages 1–6, 2015.

[3] Shubham Kumar Lala, Akshat Kumar, and Subbulakshmi T. Secure web development using owasp guidelines. In *2021 5th International Conference on Intelligent Computing and Control Systems (ICICCS)*, pages 323–332, 2021.

[4] Bingyu Li, Jingqiang Lin, Qiongxiao Wang, Ze Wang, and Jiwu Jing. Locally-centralized certificate validation and its application in desktop virtualization systems. *IEEE Transactions on Information Forensics and Security*, 16:1380–1395, 2021.

[5] Ross John Anderson. Anderson's rule (computer science).

[6] Nedhal A. Al-Sayid and Dana Aldlaeen. Database security threats: A survey study. In *2013 5th International Conference on Computer Science and Information Technology*, pages 60–64, 2013.

[7] Michael Dooley and Timothy Rooney. *Introduction to the Domain Name System (DNS)*, pages 17–29. 2017.

[8] Chris Hoffman. How the great firewall of china works to censor china's internet.

[9] Timothy Rooney. *Dynamic Host Configuration Protocol (DHCP)*, pages 53–68. 2010.

[10] R.T. Fielding and G. Kaiser. The apache http server project. *IEEE Internet Computing*, 1(4):88–90, 1997.

[11] J.A. Cortes. A brief introduction to ldap and its future. *IEEE Distributed Systems Online*, 5(4):1–3, 2004.

[12] Yongmin Zhao and Ning Lu. Research and implementation of data storage backup. In *2018 IEEE International Conference on Energy Internet (ICEI)*, pages 181–184, 2018.

[13] S. Suguna and A. Suhasini. Overview of data backup and disaster recovery in cloud. In *International Conference on Information Communication and Embedded Systems (ICICES2014)*, pages 1–7, 2014.

[14] Baphumelele Masikisiki, Siyabulela Dyakalashe, and Mfundo Shakes Scott. Network monitoring system for network equipment availability and performance reporting. In *2017 IST-Africa Week Conference (IST-Africa)*, pages 1–12, 2017.

[15] Benedikt Lebek, Jörg Uffen, Michael H. Breitner, Markus Neumann, and Bernd Hohler. Employees' information security awareness and behavior: A literature review. In *2013 46th Hawaii International Conference on System Sciences*, pages 2978–2987, 2013.

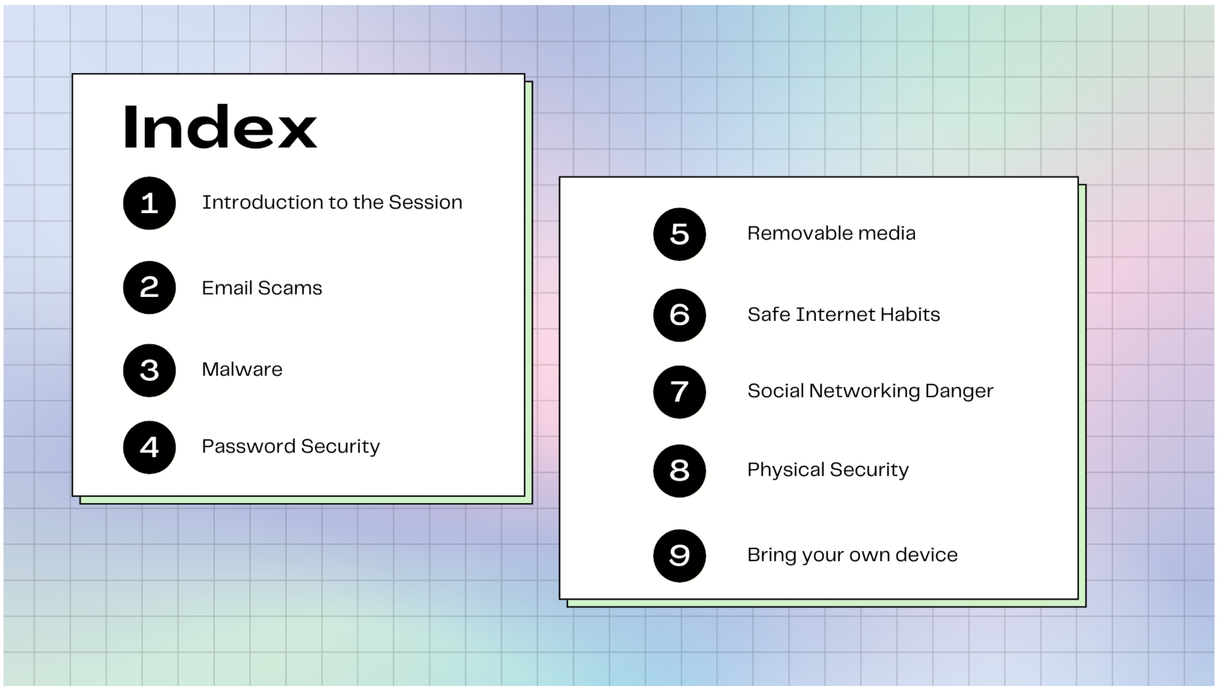
Appendices

A Cybersecurity training session for employees

Cybersecurity Training Session

by David Hernández Vilalta

Figure 23: Cybersecurity Training Session, Slide 1.



Index

1 Introduction to the Session

2 Email Scams

3 Malware

4 Password Security

5 Removable media

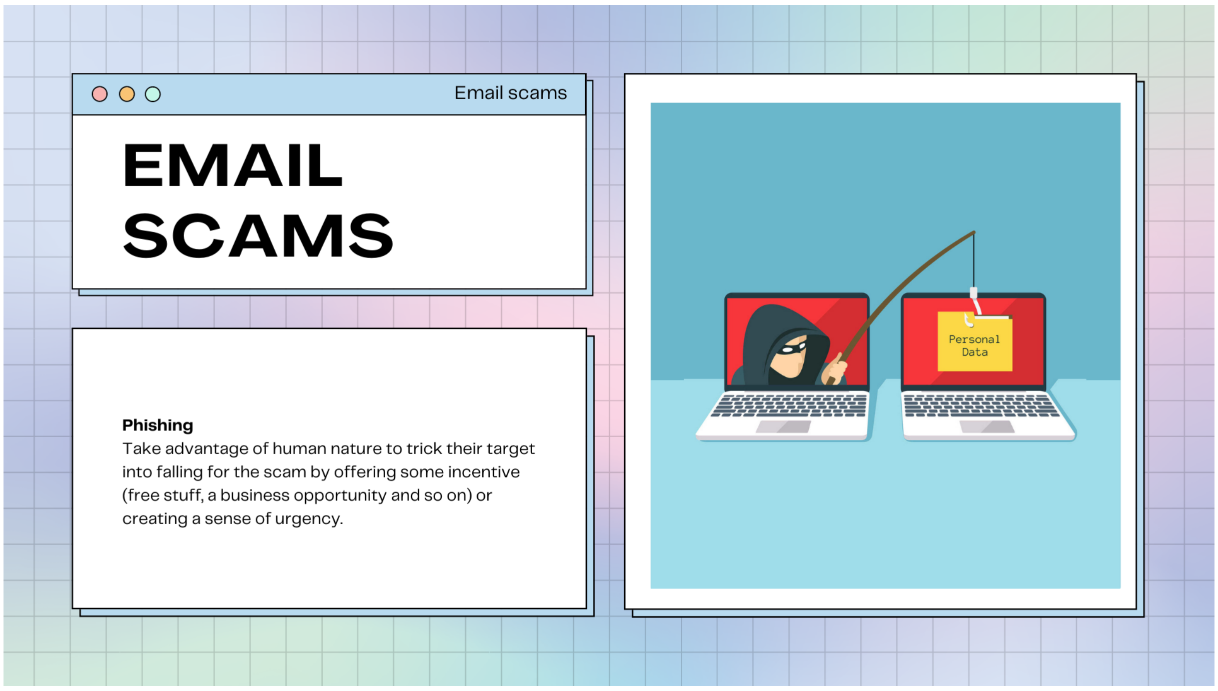
6 Safe Internet Habits

7 Social Networking Danger

8 Physical Security

9 Bring your own device

Figure 24: Cybersecurity Training Session, Slide 2.



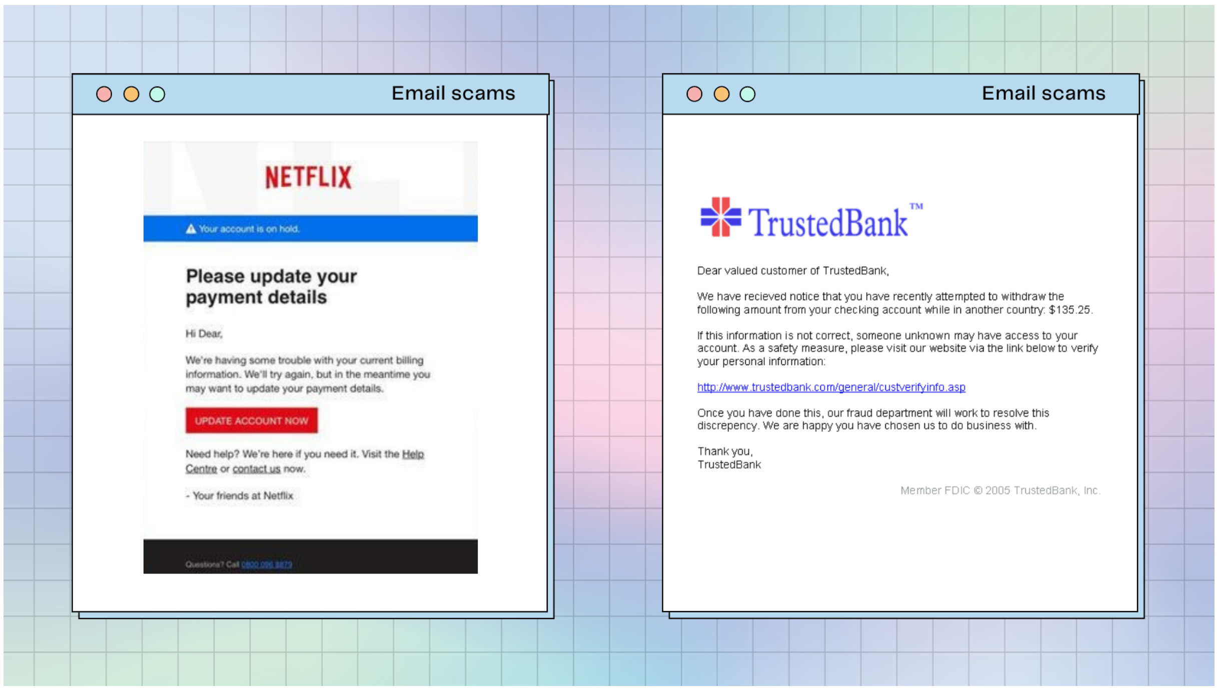
Email scams

EMAIL SCAMS

Phishing
Take advantage of human nature to trick their target into falling for the scam by offering some incentive (free stuff, a business opportunity and so on) or creating a sense of urgency.

Personal
Data

Figure 25: Cybersecurity Training Session, Slide 3.



Email scams

NETFLIX

Your account is on hold.

Please update your payment details

Hi Dear,

We're having some trouble with your current billing information. We'll try again, but in the meantime you may want to update your payment details.

UPDATE ACCOUNT NOW

Need help? We're here if you need it. Visit the Help Centre or contact us now.

- Your friends at Netflix

Questions? Call 0800 096 8879

Email scams

TrustedBank™

Dear valued customer of TrustedBank,

We have recieved notice that you have recently attempted to withdraw the following amount from your checking account while in another country: $135.25.

If this information is not correct, someone unknown may have access to your account. As a safety measure, please visit our website via the link below to verify your personal information:

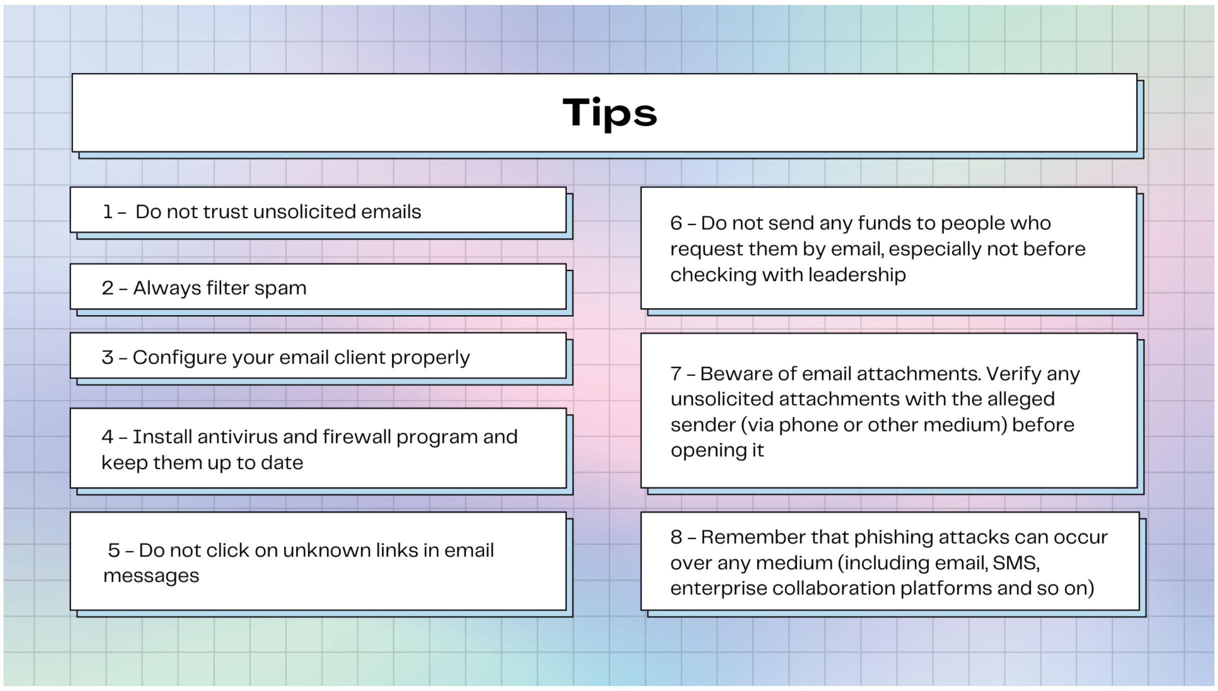
http://www.trustedbank.com/general/custverifyinfo.asp

Once you have done this, our fraud department will work to resolve this discrepency. We are happy you have chosen us to do business with.

Thank you,
TrustedBank

Member FDIC © 2005 TrustedBank, Inc.

Figure 26: Cybersecurity Training Session, Slide 4.



Tips

1 – Do not trust unsolicited emails

2 – Always filter spam

3 – Configure your email client properly

4 – Install antivirus and firewall program and keep them up to date

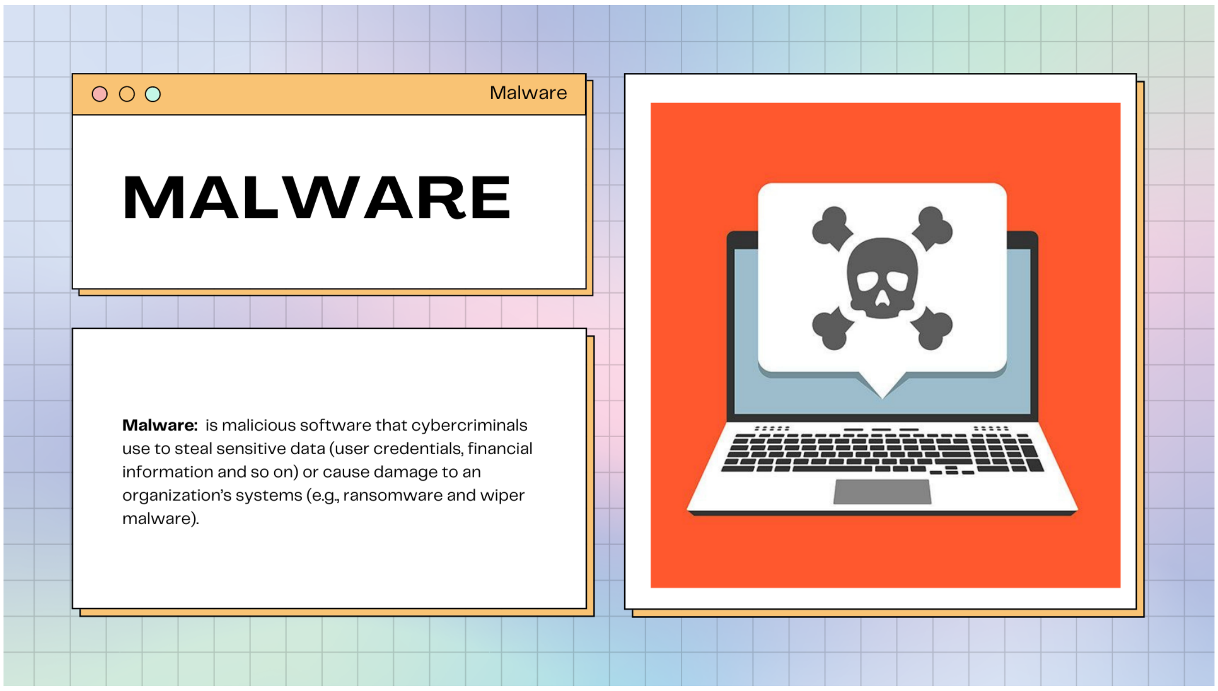
5 – Do not click on unknown links in email messages

6 – Do not send any funds to people who request them by email, especially not before checking with leadership

7 – Beware of email attachments. Verify any unsolicited attachments with the alleged sender (via phone or other medium) before opening it

8 – Remember that phishing attacks can occur over any medium (including email, SMS, enterprise collaboration platforms and so on)

Figure 27: Cybersecurity Training Session, Slide 5.

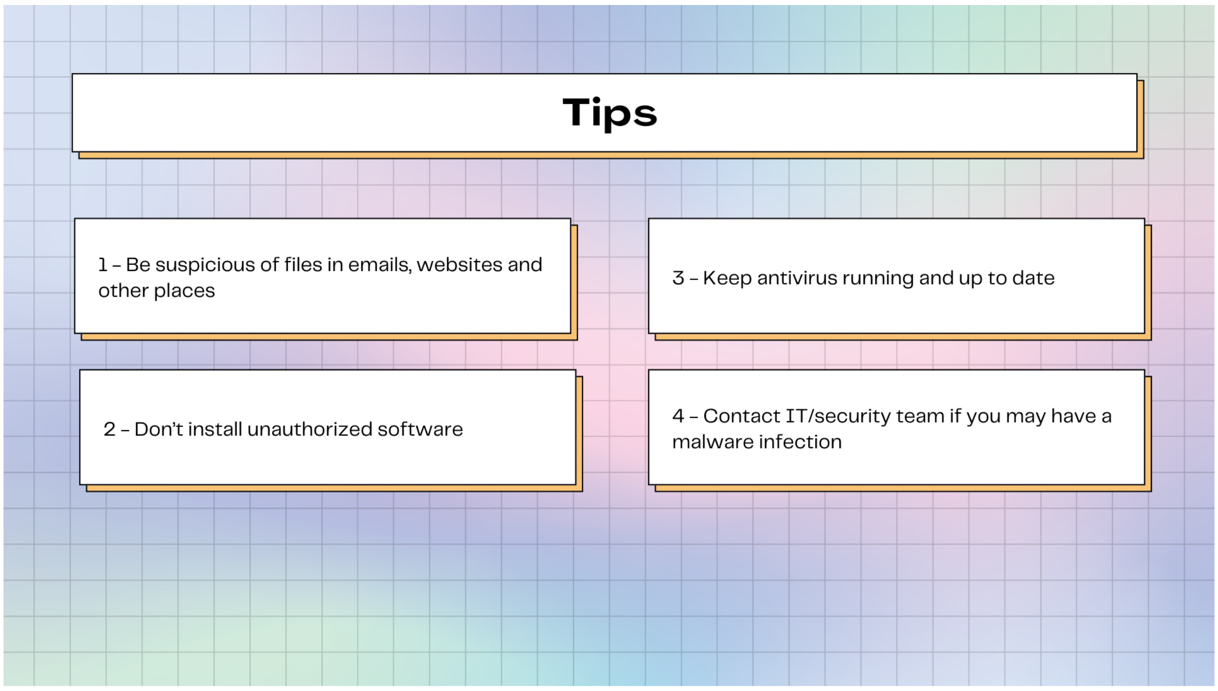


Malware

MALWARE

Malware: is malicious software that cybercriminals use to steal sensitive data (user credentials, financial information and so on) or cause damage to an organization's systems (e.g., ransomware and wiper malware).

Figure 28: Cybersecurity Training Session, Slide 6.



Tips

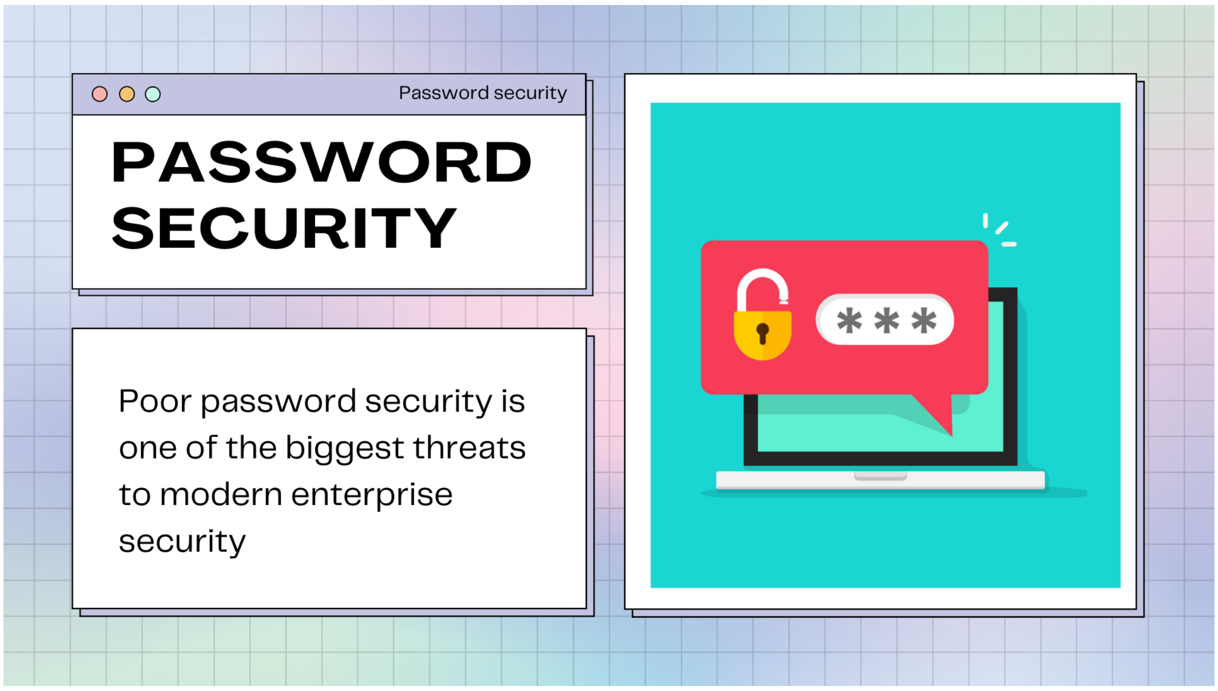
1 – Be suspicious of files in emails, websites and other places

2 – Don't install unauthorized software

3 – Keep antivirus running and up to date

4 – Contact IT/security team if you may have a malware infection

Figure 29: Cybersecurity Training Session, Slide 7.

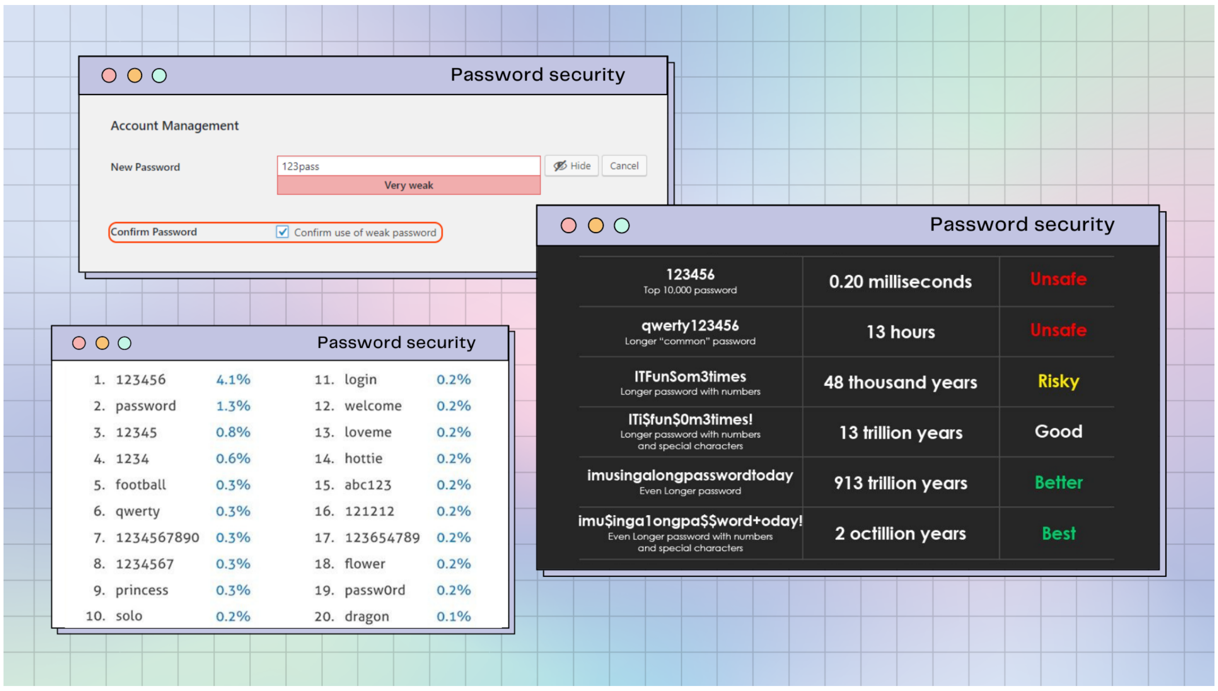


Password security

PASSWORD SECURITY

Poor password security is one of the biggest threats to modern enterprise security

Figure 30: Cybersecurity Training Session, Slide 8.



Password security

Account Management

New Password	123pass Very weak	Hide	Cancel
Confirm Password	✓ Confirm use of weak password		

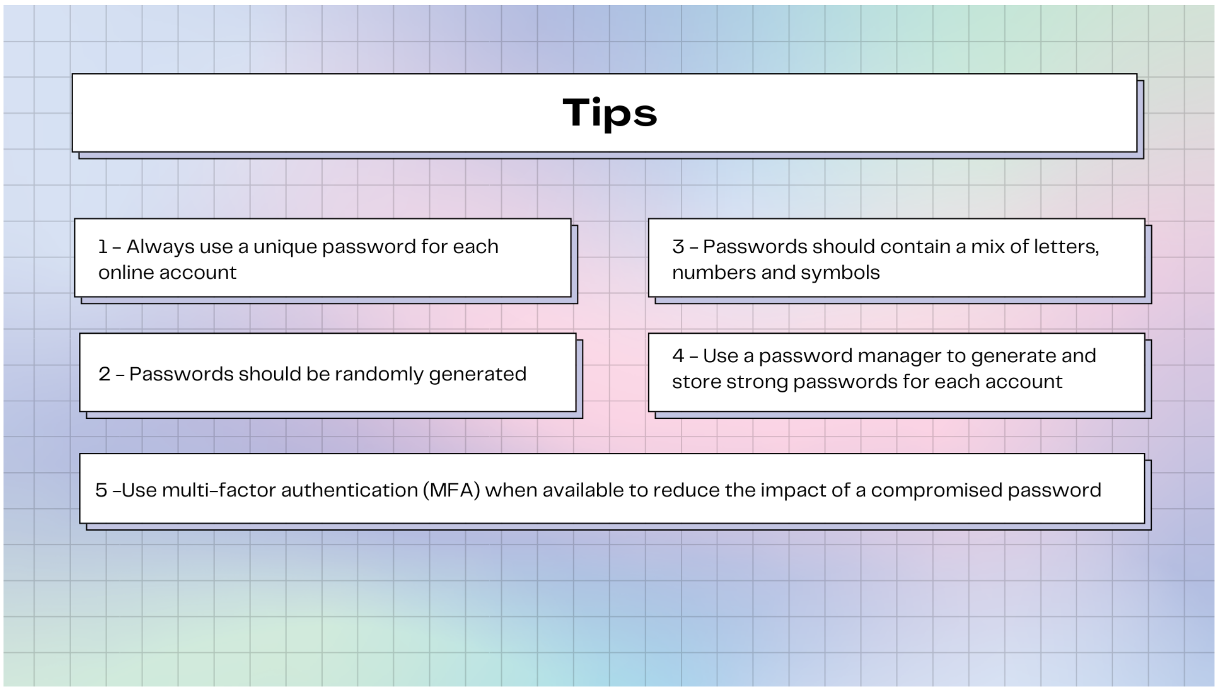
Password security

1.	123456	4.1%	11. login	0.2%
2.	password	1.3%	12. welcome	0.2%
3.	12345	0.8%	13. loveme	0.2%
4.	1234	0.6%	14. hottie	0.2%
5.	football	0.3%	15. abc123	0.2%
6.	qwerty	0.3%	16. 121212	0.2%
7.	1234567890	0.3%	17. 123654789	0.2%
8.	1234567	0.3%	18. flower	0.2%
9.	princess	0.3%	19. passw0rd	0.2%
10.	solo	0.2%	20. dragon	0.1%

Password security

123456 Top 10,000 password	0.20 milliseconds	Unsafe
qwerty123456 Longer "common" password	13 hours	Unsafe
ITFunSom3times Longer password with numbers	48 thousand years	Risky
ITi$fun$0m3times! Longer password with numbers and special characters	13 trillion years	Good
imusingalongpasswordtoday Even Longer password	913 trillion years	Better
imu$inga1ongpa$$word+oday! Even Longer password with numbers and special characters	2 octillion years	Best

Figure 31: Cybersecurity Training Session, Slide 9.



Tips

1 – Always use a unique password for each online account

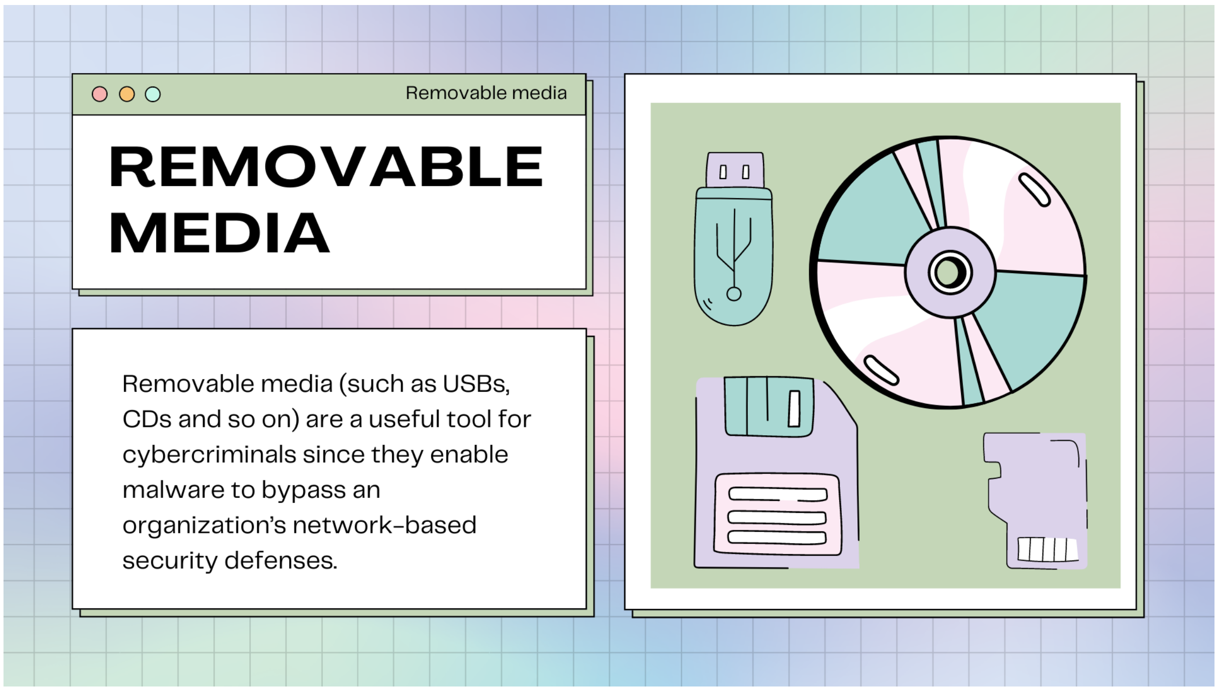
2 – Passwords should be randomly generated

3 – Passwords should contain a mix of letters, numbers and symbols

4 – Use a password manager to generate and store strong passwords for each account

5 -Use multi-factor authentication (MFA) when available to reduce the impact of a compromised password

Figure 32: Cybersecurity Training Session, Slide 10.

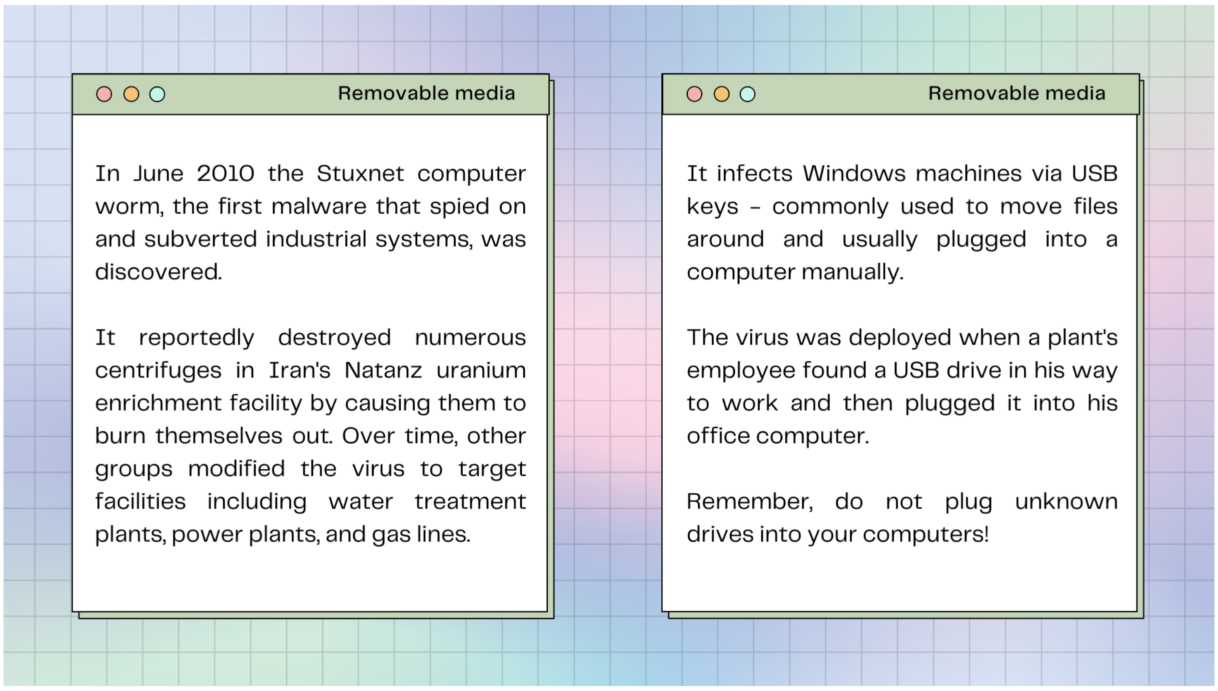


Removable media

REMOVABLE MEDIA

Removable media (such as USBs, CDs and so on) are a useful tool for cybercriminals since they enable malware to bypass an organization’s network-based security defenses.

Figure 33: Cybersecurity Training Session, Slide 11.



Removable media

In June 2010 the Stuxnet computer worm, the first malware that spied on and subverted industrial systems, was discovered.

It reportedly destroyed numerous centrifuges in Iran's Natanz uranium enrichment facility by causing them to burn themselves out. Over time, other groups modified the virus to target facilities including water treatment plants, power plants, and gas lines.

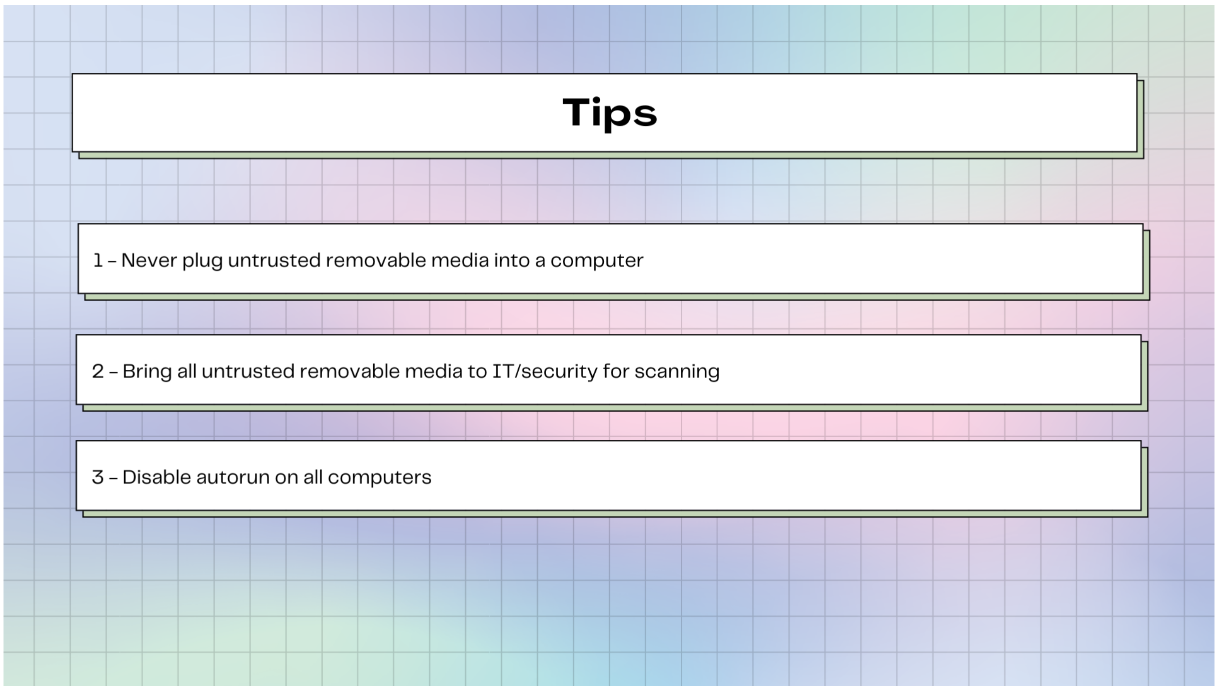
Removable media

It infects Windows machines via USB keys – commonly used to move files around and usually plugged into a computer manually.

The virus was deployed when a plant's employee found a USB drive in his way to work and then plugged it into his office computer.

Remember, do not plug unknown drives into your computers!

Figure 34: Cybersecurity Training Session, Slide 12.



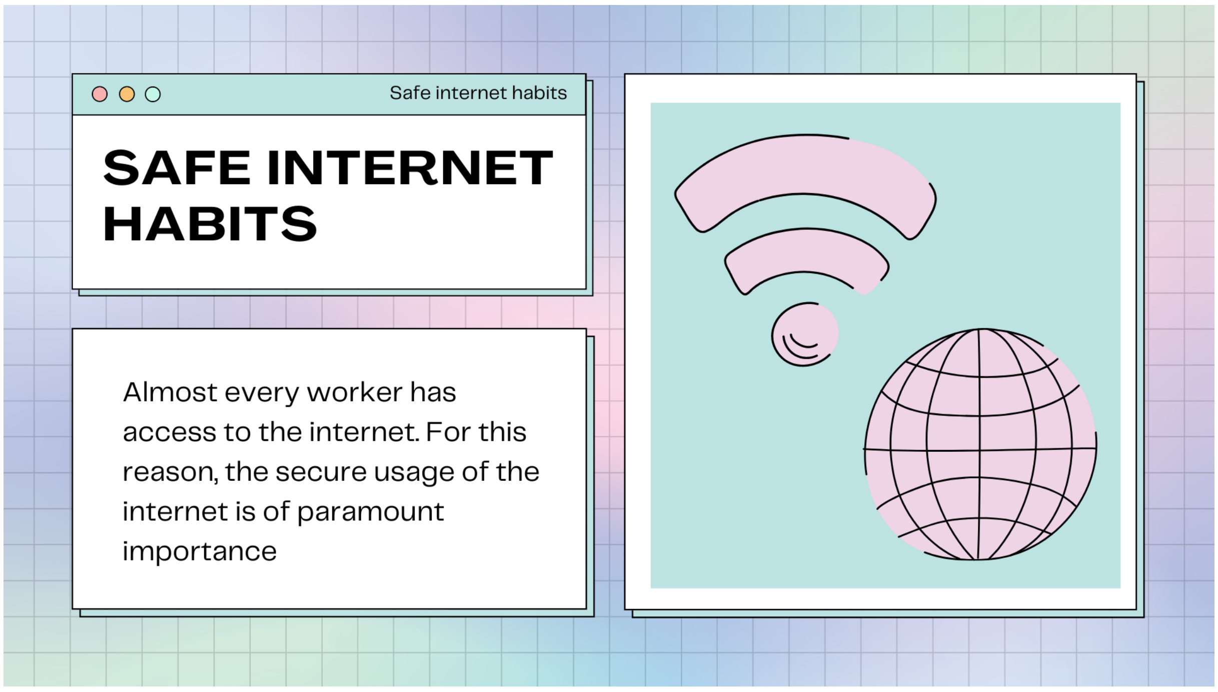
Tips

1 – Never plug untrusted removable media into a computer

2 – Bring all untrusted removable media to IT/security for scanning

3 – Disable autorun on all computers

Figure 35: Cybersecurity Training Session, Slide 13.

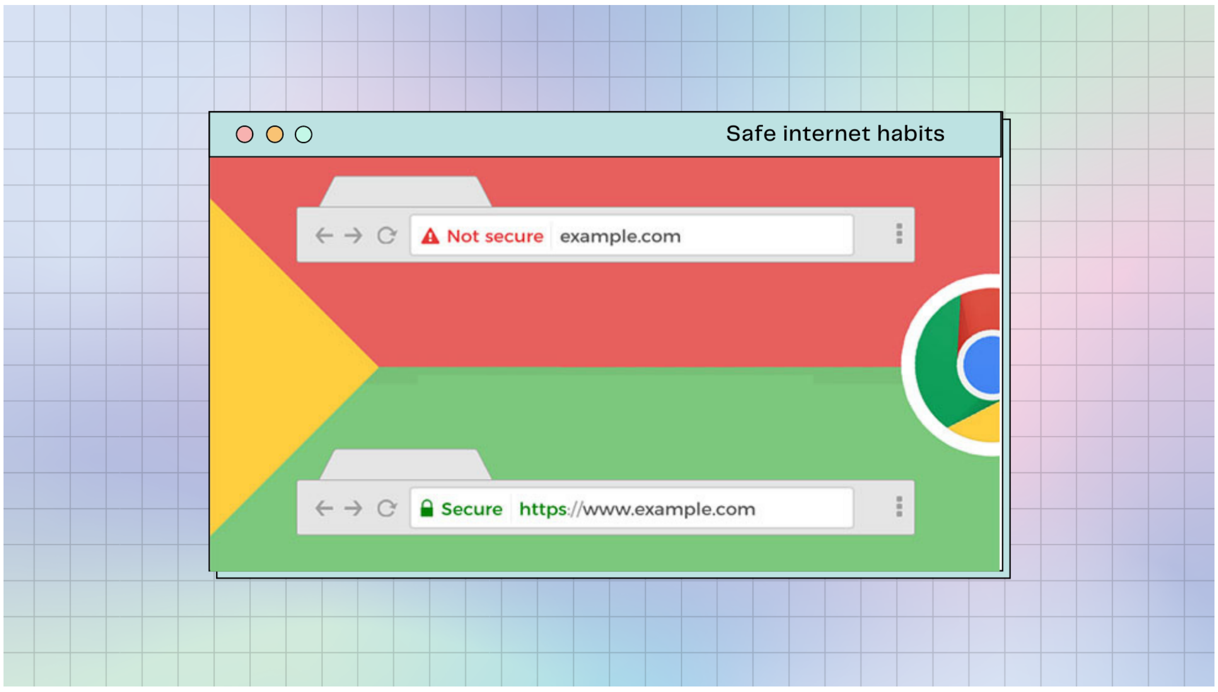


Safe internet habits

SAFE INTERNET HABITS

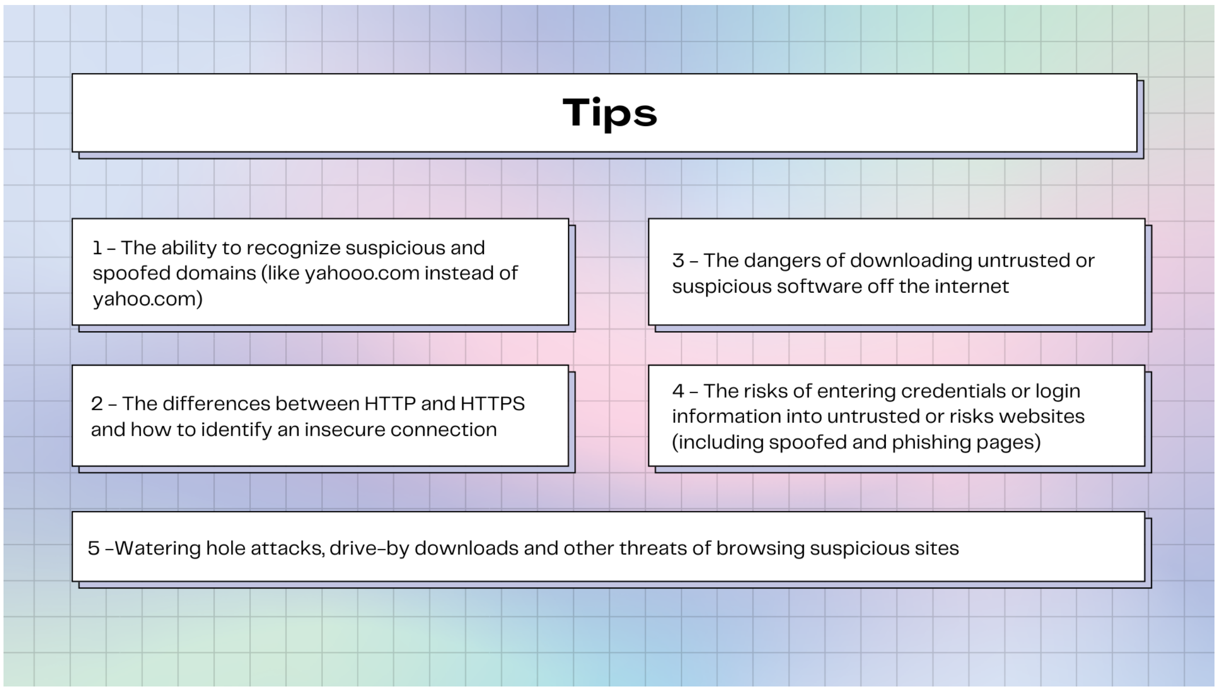
Almost every worker has access to the internet. For this reason, the secure usage of the internet is of paramount importance

Figure 36: Cybersecurity Training Session, Slide 14.



Safe internet habits

Figure 37: Cybersecurity Training Session, Slide 15.



Tips

1 – The ability to recognize suspicious and spoofed domains (like yahooo.com instead of yahoo.com)

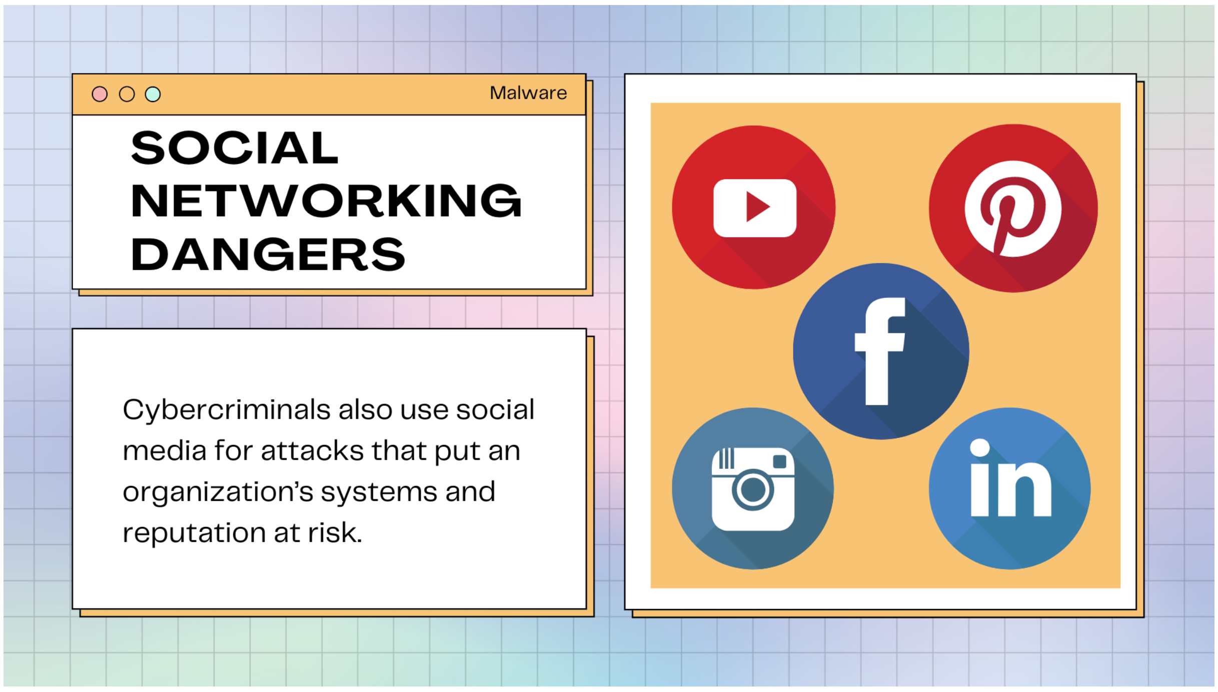
2 – The differences between HTTP and HTTPS and how to identify an insecure connection

3 – The dangers of downloading untrusted or suspicious software off the internet

4 – The risks of entering credentials or login information into untrusted or risks websites (including spoofed and phishing pages)

5 –Watering hole attacks, drive-by downloads and other threats of browsing suspicious sites

Figure 38: Cybersecurity Training Session, Slide 16.

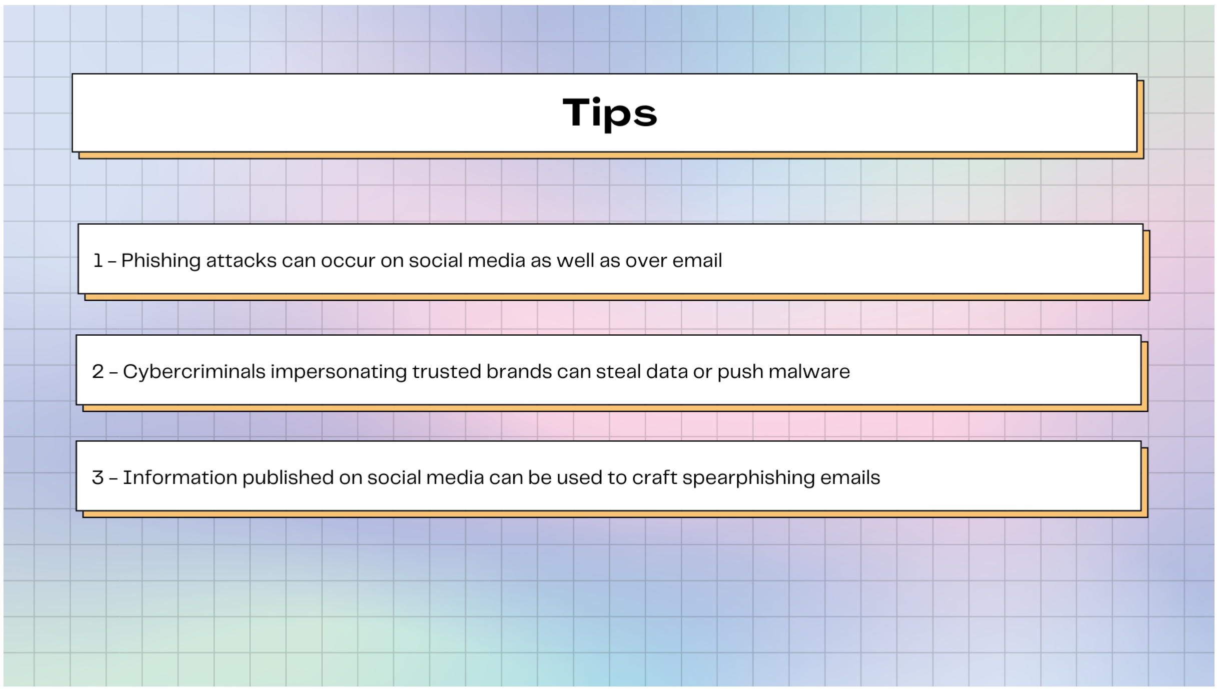


Malware

SOCIAL NETWORKING DANGERS

Cybercriminals also use social media for attacks that put an organization's systems and reputation at risk.

Figure 39: Cybersecurity Training Session, Slide 17.



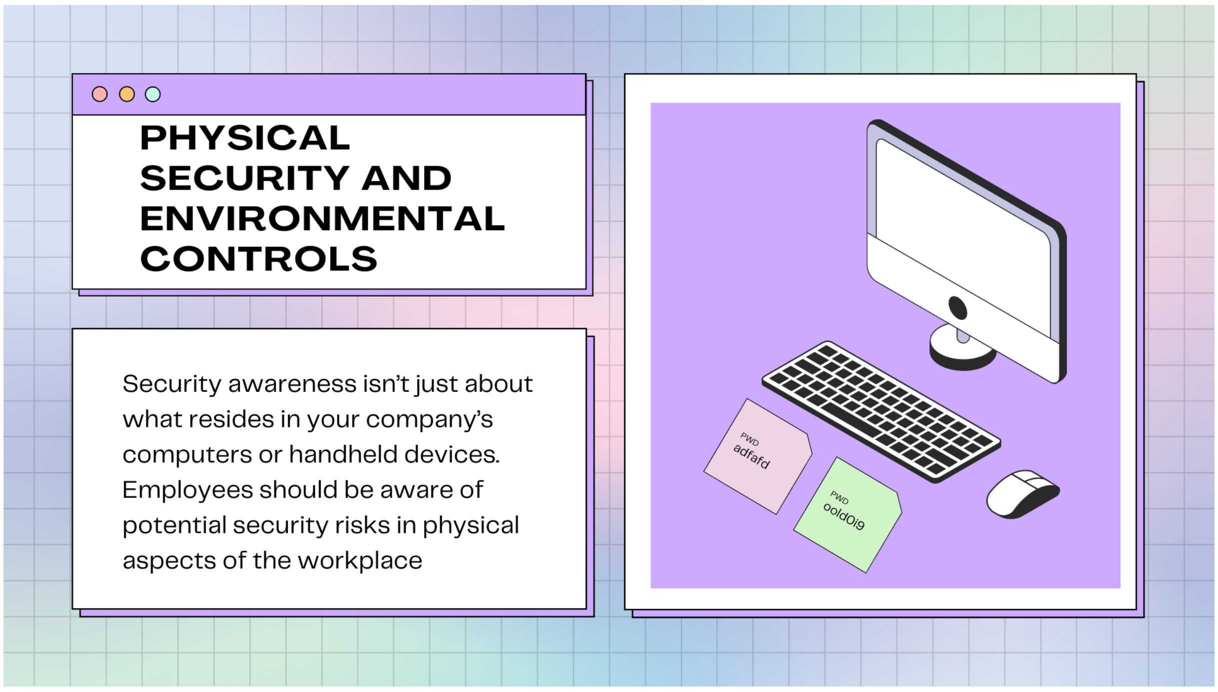
Tips

1 – Phishing attacks can occur on social media as well as over email

2 – Cybercriminals impersonating trusted brands can steal data or push malware

3 – Information published on social media can be used to craft spearphishing emails

Figure 40: Cybersecurity Training Session, Slide 18.

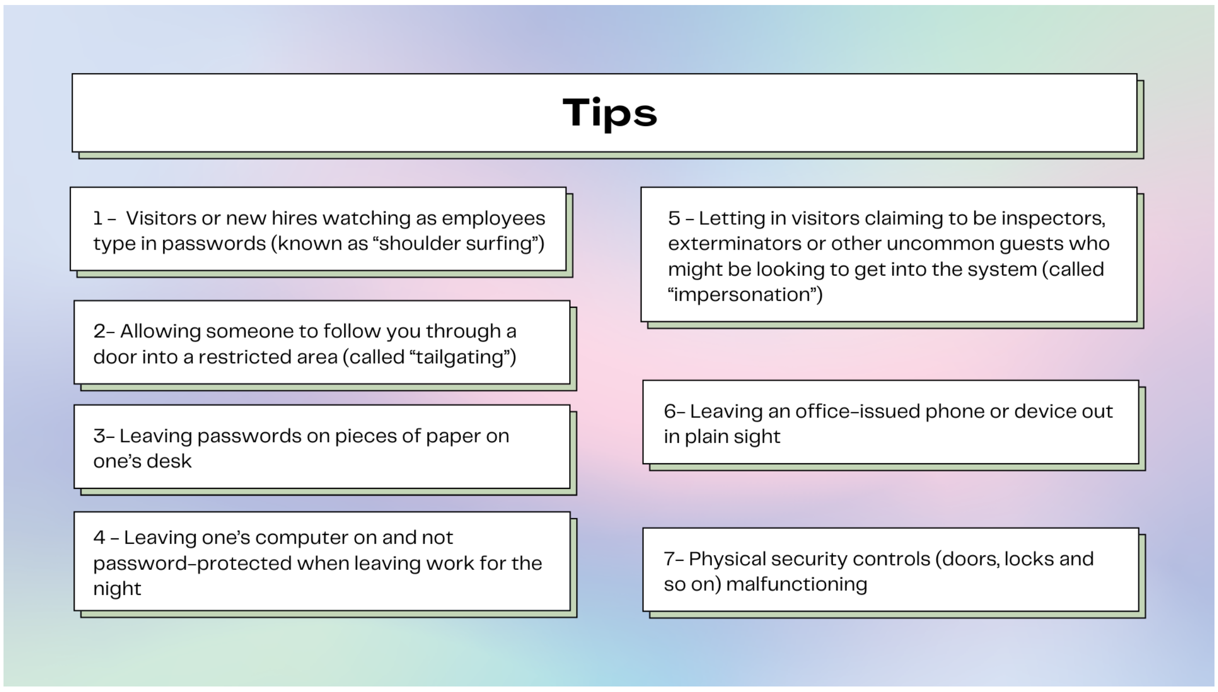


PHYSICAL SECURITY AND ENVIRONMENTAL CONTROLS

Security awareness isn't just about what resides in your company's computers or handheld devices. Employees should be aware of potential security risks in physical aspects of the workplace

PWD
adfafd
PWD
ooldoi9

Figure 41: Cybersecurity Training Session, Slide 19.



Tips

1 – Visitors or new hires watching as employees type in passwords (known as “shoulder surfing”)

2– Allowing someone to follow you through a door into a restricted area (called “tailgating”)

3– Leaving passwords on pieces of paper on one’s desk

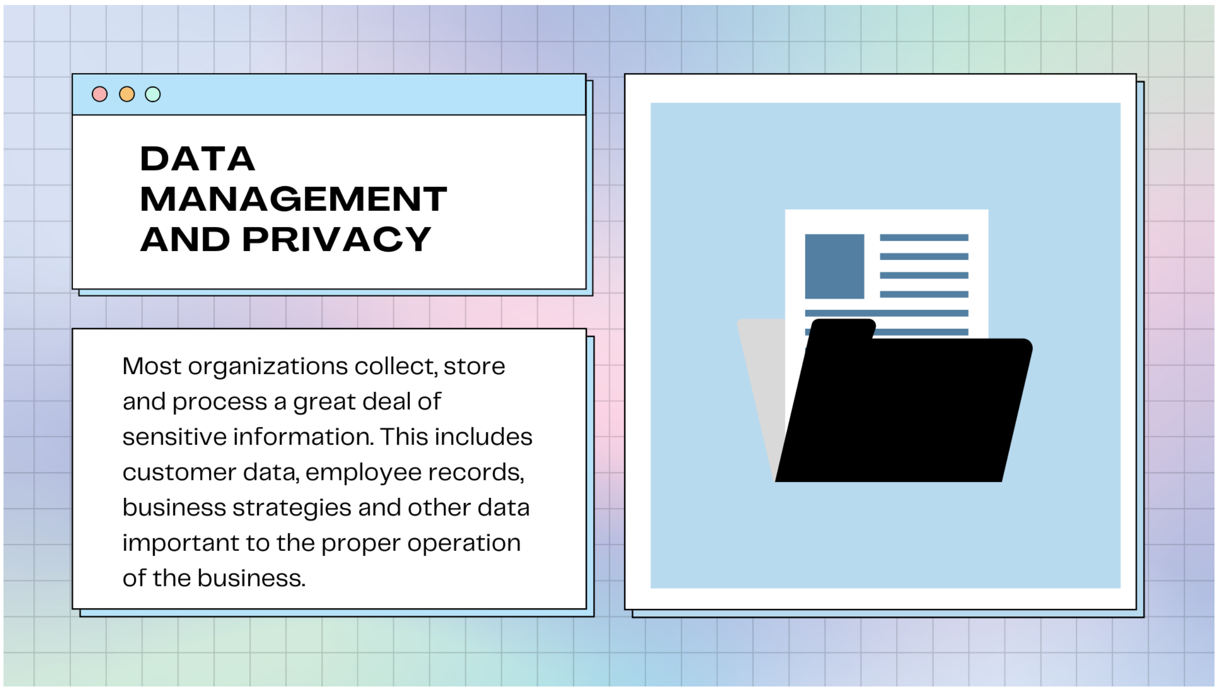
4 – Leaving one’s computer on and not password-protected when leaving work for the night

5 – Letting in visitors claiming to be inspectors, exterminators or other uncommon guests who might be looking to get into the system (called “impersonation”)

6– Leaving an office-issued phone or device out in plain sight

7– Physical security controls (doors, locks and so on) malfunctioning

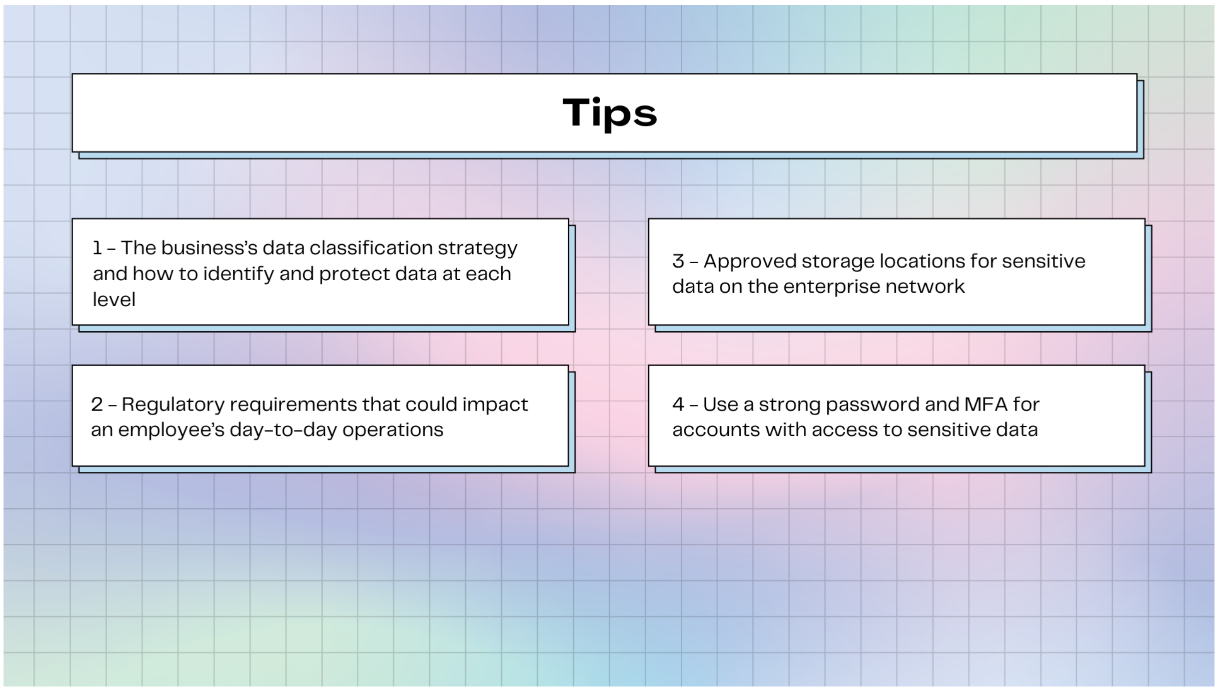
Figure 42: Cybersecurity Training Session, Slide 20.



DATA MANAGEMENT AND PRIVACY

Most organizations collect, store and process a great deal of sensitive information. This includes customer data, employee records, business strategies and other data important to the proper operation of the business.

Figure 43: Cybersecurity Training Session, Slide 21.



Tips

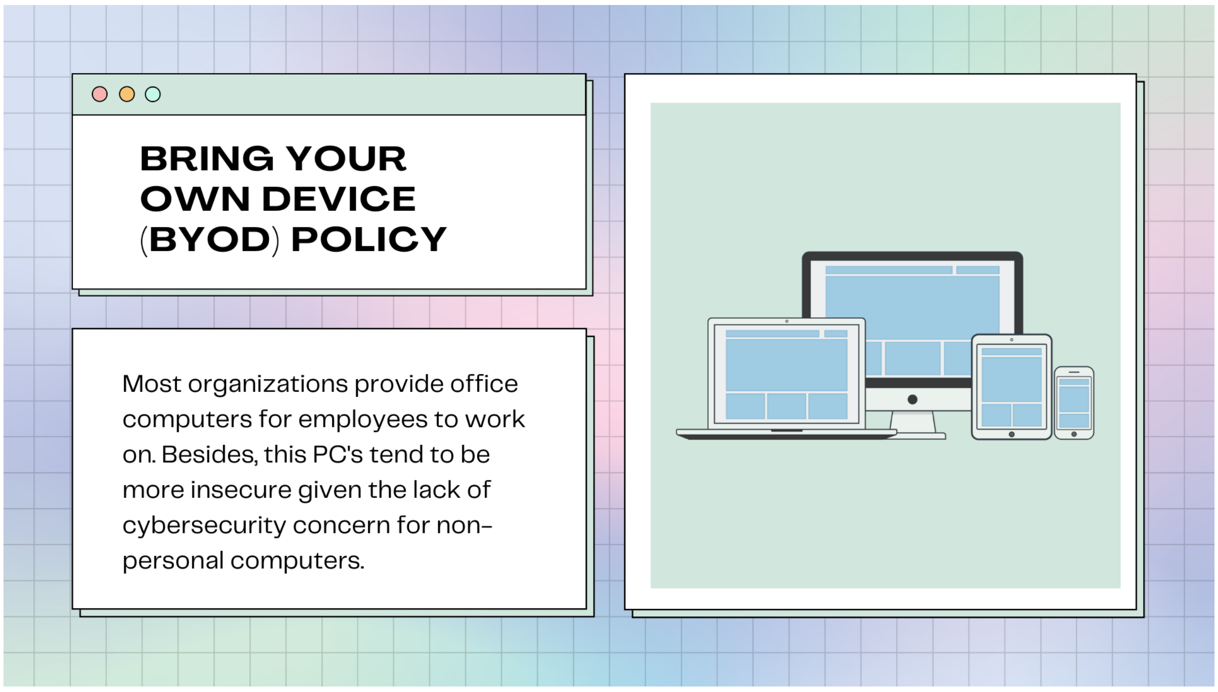
1 – The business's data classification strategy and how to identify and protect data at each level

2 – Regulatory requirements that could impact an employee's day-to-day operations

3 – Approved storage locations for sensitive data on the enterprise network

4 – Use a strong password and MFA for accounts with access to sensitive data

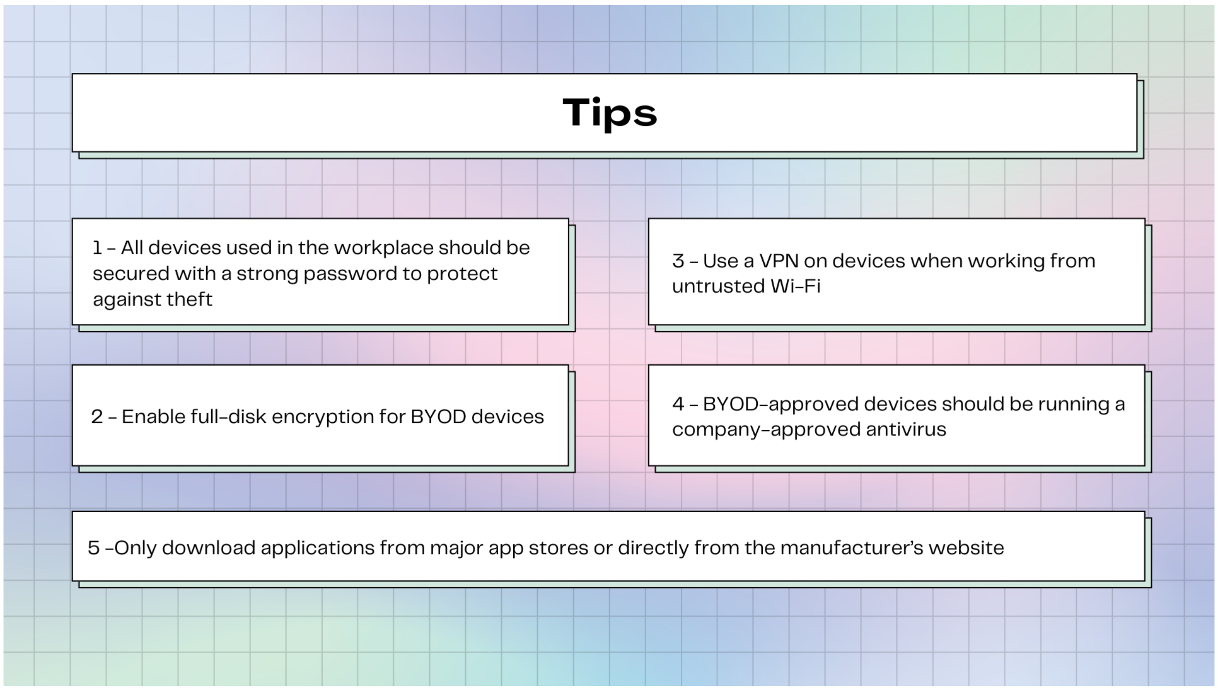
Figure 44: Cybersecurity Training Session, Slide 22.



BRING YOUR OWN DEVICE (BYOD) POLICY

Most organizations provide office computers for employees to work on. Besides, this PC's tend to be more insecure given the lack of cybersecurity concern for non-personal computers.

Figure 45: Cybersecurity Training Session, Slide 23.



Tips

1 – All devices used in the workplace should be secured with a strong password to protect against theft

3 – Use a VPN on devices when working from untrusted Wi-Fi

2 – Enable full-disk encryption for BYOD devices

4 – BYOD-approved devices should be running a company-approved antivirus

5 –Only download applications from major app stores or directly from the manufacturer's website

Figure 46: Cybersecurity Training Session, Slide 24.

Stay safe!

Figure 47: Cybersecurity Training Session, Slide 25.