Universidade do Minho

Mestrado integrado em Engenharia Informática

Trabalho Prático 2

Diogo Pinto Ribeiro, A84442

Luís Pedro Barbosa Ferreira, A86265

Segurança de Sistemas Informáticos
4th Year, 1st Semester
Departmento de Informática

February 7, 2021

Contents

1 Abstract

In this report we were given two groups of questions: A and B. In the first group we chose 2 companies and tried to discover as much sensible information as we could using different approaches, while trying to highlight the differences between the companies. In the second group we performed a vulnerability scan on a vulnerable host while having an IDS monitoring traffic. We analyzed the vulnerabilities found, as well as IDS alerts, and tried to implement possible fixes.

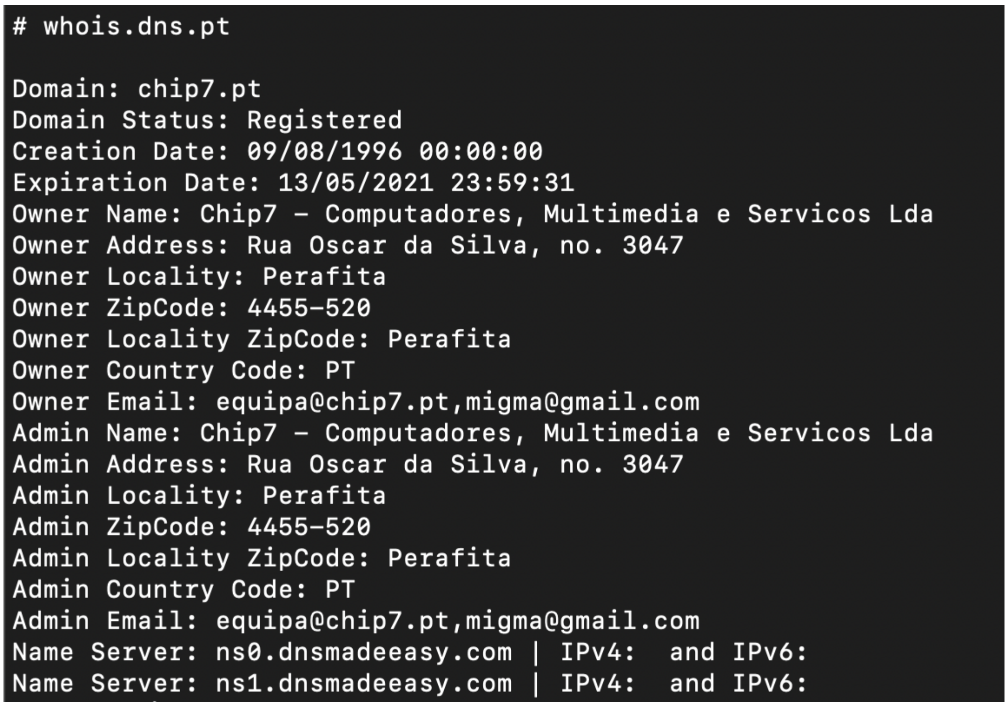
2 Part A

In this part we have chosen two companies that offer online services in order to perform **Passive Scanning** on their websites. The two companies are of different scales on purpose, so that we can spot differences in their security approach. The smaller company that we chose is **Chip 7**, a national electronics chain of stores, with an online shop. The bigger company is **Farfetch**, a multinational online clothing store.

2.1 Chip 7

2.1.1 Whois

The first thing we did for each company was to run **whois** with their domain in order to get more information about them:



```
# whois.dns.pt

Domain: chip7.pt
Domain Status: Registered
Creation Date: 09/08/1996 00:00:00
Expiration Date: 13/05/2021 23:59:31
Owner Name: Chip7 - Computadores, Multimedia e Servicos Lda
Owner Address: Rua Oscar da Silva, no. 3047
Owner Locality: Perafita
Owner ZipCode: 4455-520
Owner Locality ZipCode: Perafita
Owner Country Code: PT
Owner Email: equipa@chip7.pt,migma@gmail.com
Admin Name: Chip7 - Computadores, Multimedia e Servicos Lda
Admin Address: Rua Oscar da Silva, no. 3047
Admin Locality: Perafita
Admin ZipCode: 4455-520
Admin Locality ZipCode: Perafita
Admin Country Code: PT
Admin Email: equipa@chip7.pt,migma@gmail.com
Name Server: ns0.dnsmadeeasy.com | IPv4:  and IPv6:
Name Server: ns1.dnsmadeeasy.com | IPv4:  and IPv6:
```

Figure 1: whois result

As we can see, we where able to obtain some information about the company, such as a physical address and a staff email.

Some useful information that hackers can be in the lookout for is, for example, the Expiration Date, Owner and Admin Emails, Owner and Admin Addresses and the Name Server.

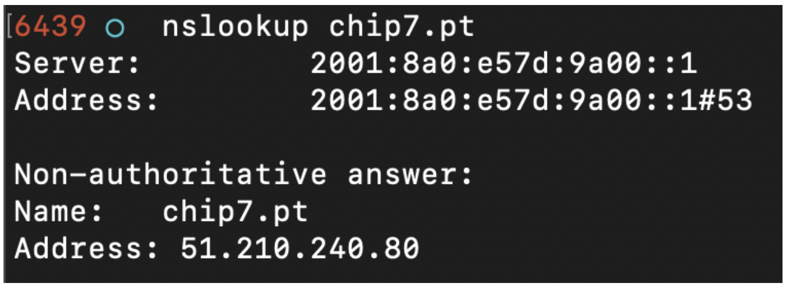
Being this a small company, it can happen that they forget to renovate the Domain, and the attacker is able to hijack it easily. The attacker can also be waiting for that date to expire. It can happen that the company takes some days to renew it.

The Owner and Admin Emails can be used for fishing, an attacker can send emails with some link to a dubious site that asks for critical information or even download a malicious program that steals some other secrets.

In this case, the Owner and Admin Addresses are just a warehouse. That place can be the headquarters and can be used to perform Denial of Service attacks or vandalism on the part of the hackers, but being a franchise, it would not be a very efficient attack. Making a search on Google Maps it looks a place with some security too. If they weren't so careful, the addresses could have been the home of some element of the staff. That could be used for other kind of attack, like spying, take advantage of their private network vulnerabilities or even use the the email and social network of a naive element of the family to get some critical information useful for an attack or blackmail.

2.1.2 Nslookup & Whois

The next step that we will take, is going to consist in retrieving the **IP Address** of the domain, and query whois again, using the obtained IP address. To retrieve the IP address of the domain, we will be using **nslookup**:

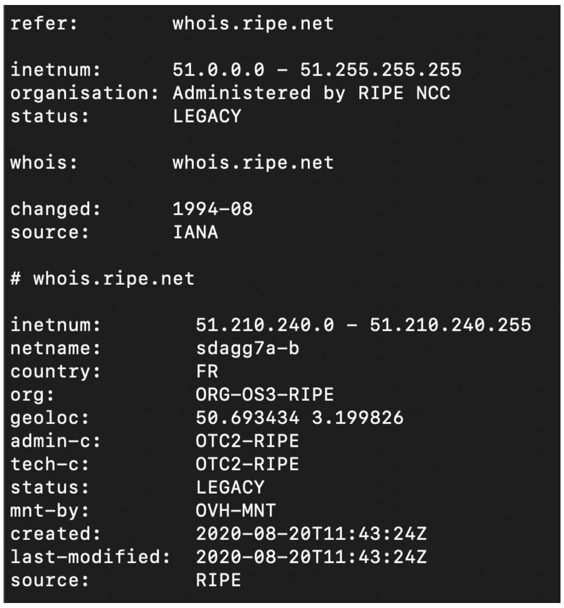


```
[6439 o  nslookup chip7.pt
Server:         2001:8a0:e57d:9a00::1
Address:        2001:8a0:e57d:9a00::1#53

Non-authoritative answer:
Name:   chip7.pt
Address: 51.210.240.80
```

Figure 2: nslookup result

As we can see, we managed to get an **IP address**, and by querying **whois** with this new address we get the following:



```
refer:        whois.ripe.net

inetnum:      51.0.0.0 - 51.255.255.255
organisation: Administered by RIPE NCC
status:       LEGACY

whois:        whois.ripe.net

changed:      1994-08
source:       IANA

# whois.ripe.net

inetnum:        51.210.240.0 - 51.210.240.255
netname:        sdagg7a-b
country:        FR
org:            ORG-OS3-RIPE
geoloc:         50.693434 3.199826
admin-c:        OTC2-RIPE
tech-c:         OTC2-RIPE
status:         LEGACY
mnt-by:         OVH-MNT
created:        2020-08-20T11:43:24Z
last-modified:  2020-08-20T11:43:24Z
source:         RIPE
```

Figure 3: whois result

Looking at the result we obtained, we discovered that the information is related to **RIPE**, a Network Coordination Centre. We also can use the *mnt-by* attribute to have a slight idea that the **OVH**, which is a French cloud computing company that offers VPS, dedicated servers and other web services, may be the hosting provider for the company. Since we didn't get much information through the **IP address**, we changed our approach.

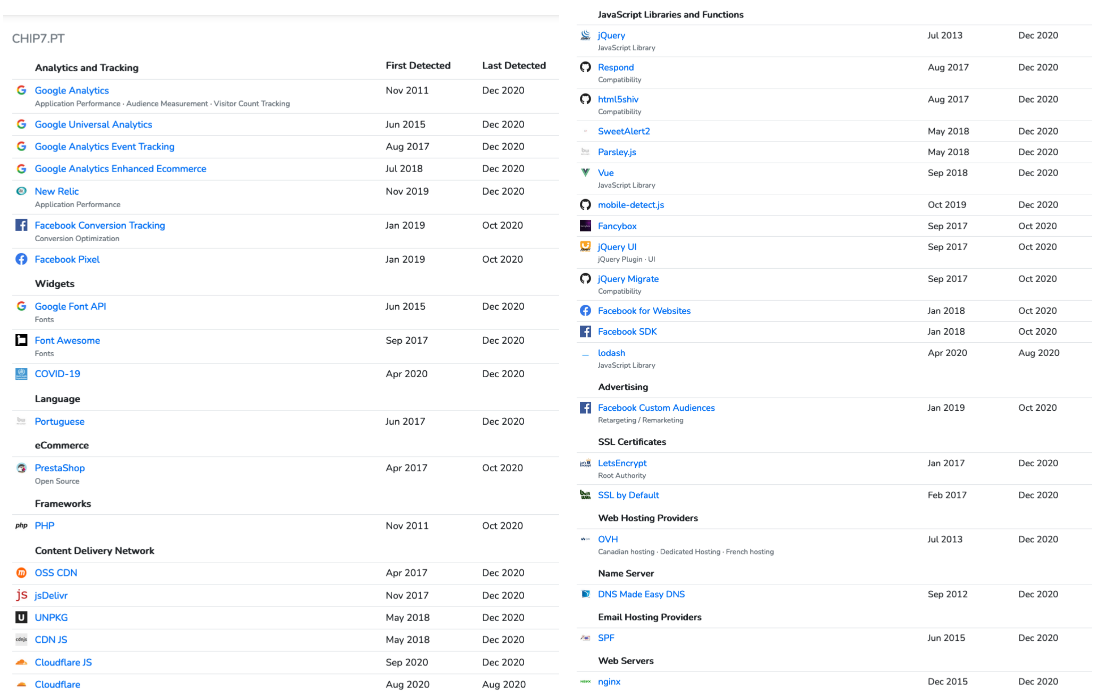
2.1.3 Job offers

We tried to get the technologies and tools used and other useful information, so we could try to get some vulnerabilities for them. For that purpose, we went on the lookout for active and past job offers. In the case of this company, we didn't find any information about this aspect, which makes us think that chip7 was careful with this topic. So, we went looking for a tool that would try to get that information just by inspecting the site.

2.1.4 BuiltWith

As we are only using a passive search, a good idea would be use the **BuiltWith** tool in order to know what technologies are used.

As we can see from the figures below, we are able to see what kind of technologies are being used, such as **jQuery**, **PHP**, **PrestaShop**, **Vue**. We also know that they are using a **nginx** web server, and as we suspected. they are using **OVH** for hosting. Just with this tools we were able to uncover a lot of important information.



CHIP7.PT

Analytics and Tracking	First Detected	Last Detected
Google Analytics Application Performance · Audience Measurement · Visitor Count Tracking	Nov 2011	Dec 2020
Google Universal Analytics	Jun 2015	Dec 2020
Google Analytics Event Tracking	Aug 2017	Dec 2020
Google Analytics Enhanced Ecommerce	Jul 2018	Dec 2020
New Relic Application Performance	Nov 2019	Dec 2020
Facebook Conversion Tracking Conversion Optimization	Jan 2019	Oct 2020
Facebook Pixel	Jan 2019	Oct 2020
Widgets		
Google Font API Fonts	Jun 2015	Dec 2020
Font Awesome Fonts	Sep 2017	Dec 2020
COVID-19	Apr 2020	Dec 2020
Language		
Portuguese	Jun 2017	Dec 2020
eCommerce		
PrestaShop Open Source	Apr 2017	Oct 2020
Frameworks		
PHP	Nov 2011	Oct 2020
Content Delivery Network		
OSS CDN	Apr 2017	Dec 2020
jsDelivr	Nov 2017	Dec 2020
UNPKG	May 2018	Dec 2020
CDN JS	May 2018	Dec 2020
Cloudflare JS	Sep 2020	Dec 2020
Cloudflare	Aug 2020	Aug 2020

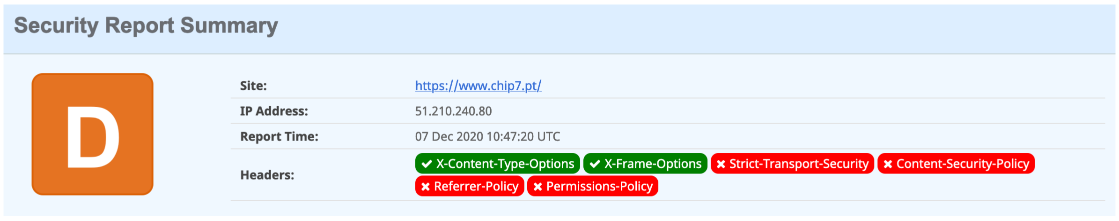
JavaScript Libraries and Functions		
jQuery JavaScript Library	Jul 2013	Dec 2020
Respond Compatibility	Aug 2017	Dec 2020
html5shiv Compatibility	Aug 2017	Dec 2020
SweetAlert2	May 2018	Dec 2020
Parsley.js	May 2018	Dec 2020
Vue JavaScript Library	Sep 2018	Dec 2020
mobile-detect.js	Oct 2019	Dec 2020
Fancybox	Sep 2017	Oct 2020
jQuery UI jQuery Plugin · UI	Sep 2017	Oct 2020
jQuery Migrate Compatibility	Sep 2017	Oct 2020
Facebook for Websites	Jan 2018	Oct 2020
Facebook SDK	Jan 2018	Oct 2020
lodash JavaScript Library	Apr 2020	Aug 2020
Advertising		
Facebook Custom Audiences Retargeting / Remarketing	Jan 2019	Oct 2020
SSL Certificates		
LetsEncrypt Root Authority	Jan 2017	Dec 2020
SSL by Default	Feb 2017	Dec 2020
Web Hosting Providers		
OVH Canadian hosting · Dedicated Hosting · French hosting	Jul 2013	Dec 2020
Name Server		
DNS Made Easy DNS	Sep 2012	Dec 2020
Email Hosting Providers		
SPF	Jun 2015	Dec 2020
Web Servers		
nginx	Dec 2015	Dec 2020

Figure 4: BuiltWith result

2.1.5 SecurityHeaders.io

Another interesting aspect worth analysing are the website's headers. To do that we are going to use **SecurityHeaders.io** because it awards the website a grade, and we want to use it as a benchmark value.

As we can see, this website managed a classification of **D**, which isn't great. This is caused by the lack of 4 security headers which should be implemented:


Security Report Summary

D

Site:	https://www.chip7.pt/
IP Address:	51.210.240.80
Report Time:	07 Dec 2020 10:47:20 UTC
Headers:	✓ X-Content-Type-Options ✓ X-Frame-Options ✗ Strict-Transport-Security ✗ Content-Security-Policy ✗ Referrer-Policy ✗ Permissions-Policy

Figure 5: Security Headers result

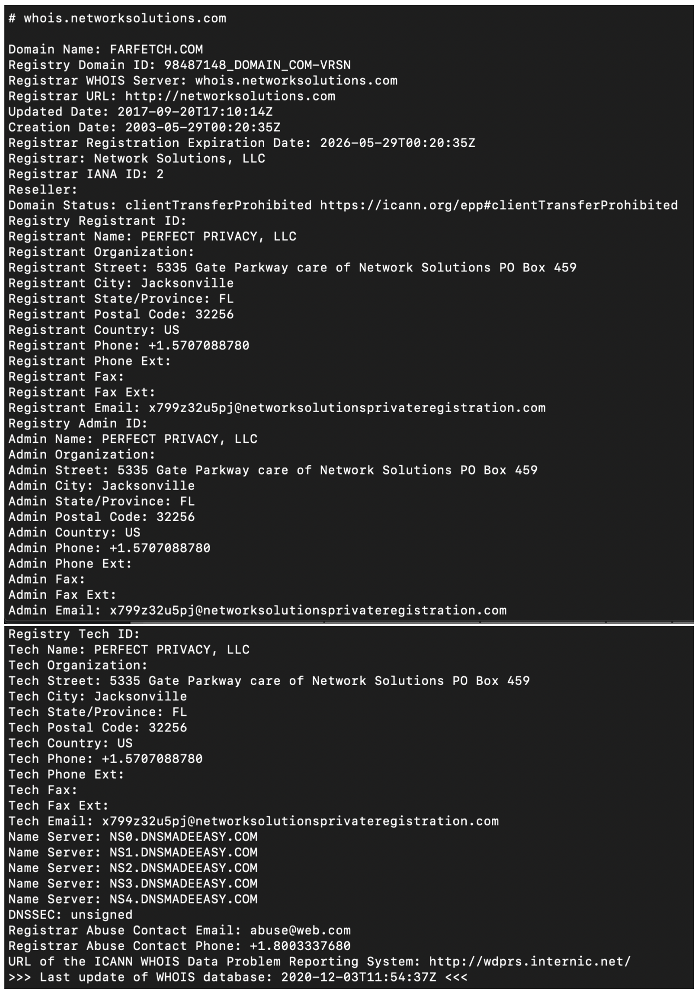
- **Strict-Transport-Security** - HTTP Strict Transport Security is an excellent feature to support a site and strengthens an implementation of TLS by getting the User Agent to enforce the use of HTTPS. Recommended value "Strict-Transport-Security: max-age=31536000; includeSubDomains".
- **Content-Security-Policy** - Content Security Policy is an effective measure to protect a site from XSS attacks. By whitelisting sources of approved content, it can prevent the browser from loading malicious assets.
- **Referrer-Policy** - Referrer Policy is a new header that allows a site to control how much information the browser includes with navigations away from a document and should be set by all sites.
- **Permissions-Policy** - Permissions Policy is a new header that allows a site to control which features and APIs can be used in the browser.

2.2 Farfetch

2.2.1 Whois

For this company we also started by running **whois** with their domain in order to get more information.

As we can see, despite having more data returned, we obtained less information about the company that we targeted. In this case, they used a company (Perfect Privacy LLC) to act as a Proxy and protect private information. So we haven't got no private email nor an valid address, only a Post-office box, and we also noticed that the expiration date of the domain won't end in the near future (2026).



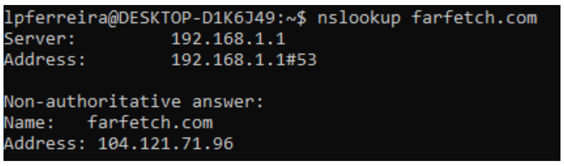
```
# whois.networksolutions.com

Domain Name: FARFETCH.COM
Registry Domain ID: 98487148_DOMAIN_COM-VRSN
Registrar WHOIS Server: whois.networksolutions.com
Registrar URL: http://networksolutions.com
Updated Date: 2017-09-20T17:10:14Z
Creation Date: 2003-05-29T00:20:35Z
Registrar Registration Expiration Date: 2026-05-29T00:20:35Z
Registrar: Network Solutions, LLC
Registrar IANA ID: 2
Reseller:
Domain Status: clientTransferProhibited https://icann.org/epp#clientTransferProhibited
Registry Registrant ID:
Registrant Name: PERFECT PRIVACY, LLC
Registrant Organization:
Registrant Street: 5335 Gate Parkway care of Network Solutions PO Box 459
Registrant City: Jacksonville
Registrant State/Province: FL
Registrant Postal Code: 32256
Registrant Country: US
Registrant Phone: +1.5707088780
Registrant Phone Ext:
Registrant Fax:
Registrant Fax Ext:
Registrant Email: x799z32u5pj@networksolutionsprivateregistration.com
Registry Admin ID:
Admin Name: PERFECT PRIVACY, LLC
Admin Organization:
Admin Street: 5335 Gate Parkway care of Network Solutions PO Box 459
Admin City: Jacksonville
Admin State/Province: FL
Admin Postal Code: 32256
Admin Country: US
Admin Phone: +1.5707088780
Admin Phone Ext:
Admin Fax:
Admin Fax Ext:
Admin Email: x799z32u5pj@networksolutionsprivateregistration.com
Registry Tech ID:
Tech Name: PERFECT PRIVACY, LLC
Tech Organization:
Tech Street: 5335 Gate Parkway care of Network Solutions PO Box 459
Tech City: Jacksonville
Tech State/Province: FL
Tech Postal Code: 32256
Tech Country: US
Tech Phone: +1.5707088780
Tech Phone Ext:
Tech Fax:
Tech Fax Ext:
Tech Email: x799z32u5pj@networksolutionsprivateregistration.com
Name Server: NS0.DNSMADEEASY.COM
Name Server: NS1.DNSMADEEASY.COM
Name Server: NS2.DNSMADEEASY.COM
Name Server: NS3.DNSMADEEASY.COM
Name Server: NS4.DNSMADEEASY.COM
DNSSEC: unsigned
Registrar Abuse Contact Email: abuse@web.com
Registrar Abuse Contact Phone: +1.8003337680
URL of the ICANN WHOIS Data Problem Reporting System: http://wdprs.internic.net/
>>> Last update of WHOIS database: 2020-12-03T11:54:37Z <<<
```

Figure 6: whois result

2.2.2 Nslookup & Whois

The next step will be, just like with the other company, retrieving the **IP Address** of the domain, and with that Address trying to know "whois" responsible for it. For the part of fetching the **IP Address** of the domain, we used **nslookup**:

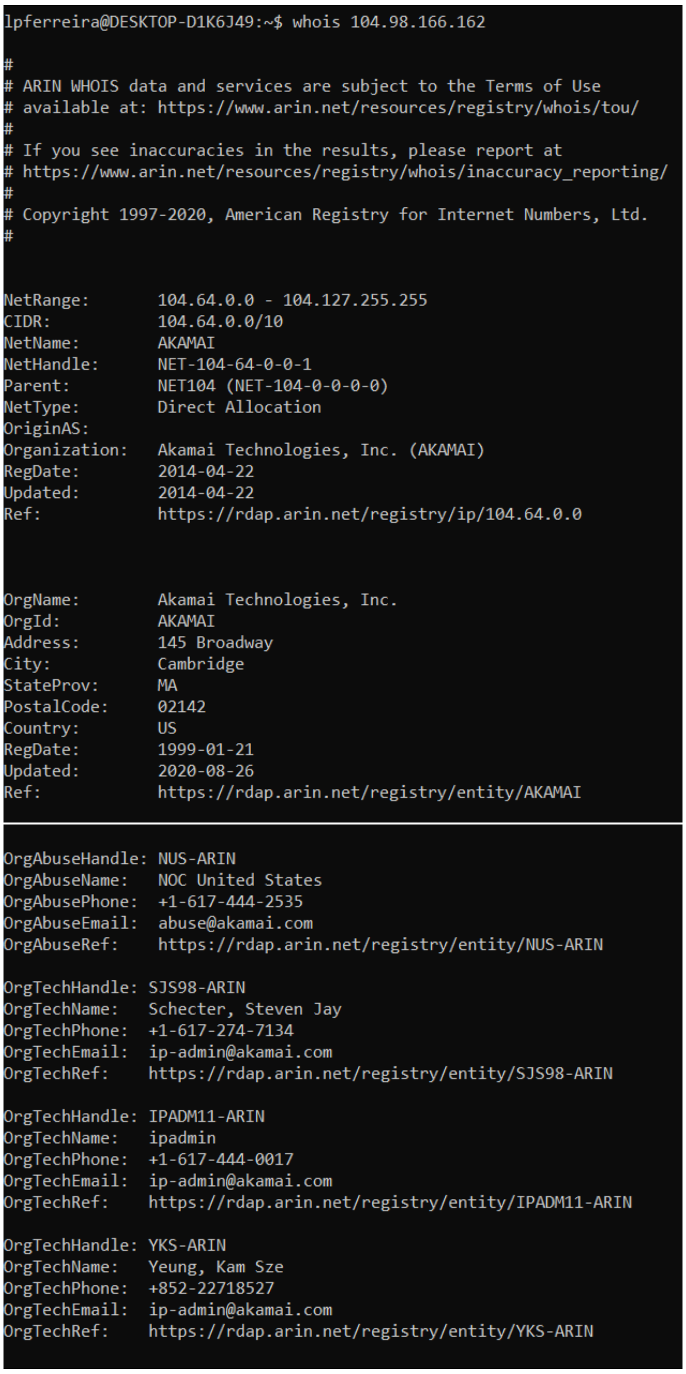


```
lpferreira@DESKTOP-D1K6J49:~$ nslookup farfetch.com
Server:         192.168.1.1
Address:        192.168.1.1#53

Non-authoritative answer:
Name:   farfetch.com
Address: 104.121.71.96
```

Figure 7: nslookup result

Having obtained the wanted **IP Address**, we now query it with **whois** to get more information:



```
lpferreira@DESKTOP-D1K6J49:~$ whois 104.98.166.162

#
# ARIN WHOIS data and services are subject to the Terms of Use
# available at: https://www.arin.net/resources/registry/whois/tou/
#
# If you see inaccuracies in the results, please report at
# https://www.arin.net/resources/registry/whois/inaccuracy_reporting/
#
# Copyright 1997-2020, American Registry for Internet Numbers, Ltd.
#


NetRange:       104.64.0.0 - 104.127.255.255
CIDR:           104.64.0.0/10
NetName:        AKAMAI
NetHandle:      NET-104-64-0-0-1
Parent:         NET104 (NET-104-0-0-0-0)
NetType:        Direct Allocation
OriginAS:
Organization:   Akamai Technologies, Inc. (AKAMAI)
RegDate:        2014-04-22
Updated:        2014-04-22
Ref:            https://rdap.arin.net/registry/ip/104.64.0.0



OrgName:        Akamai Technologies, Inc.
OrgId:          AKAMAI
Address:        145 Broadway
City:           Cambridge
StateProv:      MA
PostalCode:     02142
Country:        US
RegDate:        1999-01-21
Updated:        2020-08-26
Ref:            https://rdap.arin.net/registry/entity/AKAMAI

OrgAbuseHandle: NUS-ARIN
OrgAbuseName:   NOC United States
OrgAbusePhone:  +1-617-444-2535
OrgAbuseEmail:  abuse@akamai.com
OrgAbuseRef:    https://rdap.arin.net/registry/entity/NUS-ARIN

OrgTechHandle: SJS98-ARIN
OrgTechName:   Schecter, Steven Jay
OrgTechPhone:  +1-617-274-7134
OrgTechEmail:  ip-admin@akamai.com
OrgTechRef:    https://rdap.arin.net/registry/entity/SJS98-ARIN

OrgTechHandle: IPADM11-ARIN
OrgTechName:   ipadmin
OrgTechPhone:  +1-617-444-0017
OrgTechEmail:  ip-admin@akamai.com
OrgTechRef:    https://rdap.arin.net/registry/entity/IPADM11-ARIN

OrgTechHandle: YKS-ARIN
OrgTechName:   Yeung, Kam Sze
OrgTechPhone:  +852-22718527
OrgTechEmail:  ip-admin@akamai.com
OrgTechRef:    https://rdap.arin.net/registry/entity/YKS-ARIN
```

Figure 8: whois result

Looking at the result we obtained, we discovered that the information is related to **Akamai**, a global content delivery network (CDN), cybersecurity, and cloud service company, which should be one who's hosting the web pages.

2.2.3 Job Offers

Once again, we want to get some information on the technologies and tools used with development of this site. This time, we got some useful information when searching for job offers mainly using a business oriented social network called **LinkedIn**[11].

In that job offer they mention they are searching for someone who has experience in some technologies, those being `.NET Core`, `Java`, `Apache Cassandra`, `Apache Kafka`, `RabbitMQ`, `Neo4j`, `PostgreSQL`, `Docker` and `Kubernetes`. This is some good information to know what is used in their website. Despite this information, **Farfetch** was careful enough so we couldn't get any versions for that technologies so we could get some vulnerabilities out of them.

2.2.4 BuiltWith

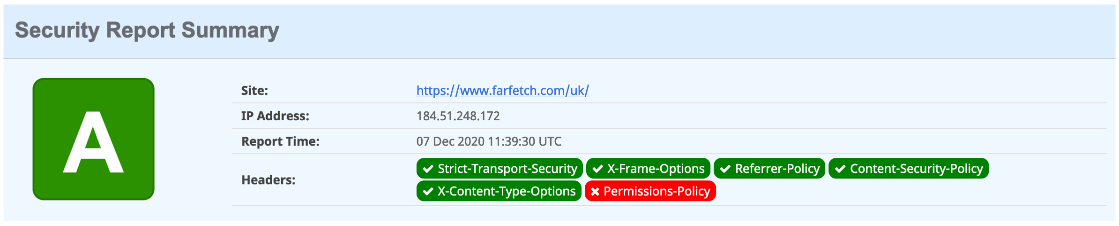
Having got some technologies, we still used **BuiltWith** tool so we can get more information about the technologies implemented on their website.

This time we got a lot of information in every section of the technology profile, so we choose not to fill pages with prints. But all the information can be accessed **here**.

Just by hovering over all that technology we could notice that a lot of technologies are implemented with the purpose of having a more secure environment. There are plenty of frameworks that are in use, like **ASP.NET**, **PHP**, **Perl** and **Java**. Just like a lot JavaScript libraries, like **JQuery**, **core-js** and **React**. We can also confirm that **Akamai** is indeed the web hosting provider and main Content Delivery Network.

2.2.5 SecurityHeaders.io

In order to make a comparison with the previous company, we also analysed the headers using **SecurityHeaders.io**.



Security Report Summary

A

Site:	https://www.farfetch.com/uk/
IP Address:	184.51.248.172
Report Time:	07 Dec 2020 11:39:30 UTC
Headers:	✓ Strict-Transport-Security ✓ X-Frame-Options ✓ Referrer-Policy ✓ Content-Security-Policy ✓ X-Content-Type-Options ✗ Permissions-Policy

Figure 9: Security Headers result

As we can see, Farfetch managed to get a score of **A**, beating the previous score of D. Regarding to these scores we can conclude that Farfetch is more careful with the security of their site. Only having one similar missing header being:

- **Permissions-Policy** - Permissions Policy is a new header that allows a site to control which features and APIs can be used in the browser.

3 Part B

In this part we will be auditing a **Metasploitable 3** system using an **Auditor System**. In this case, the auditor system is a Kali Linux VM.

3.1 Q1

The fist step that we need to take in order to identify vulnerabilities without using vulnerability scanners is to run `nmap`:

```
nmap -v -sS -O 172.20.5.2
```

By using the `nmap` with these flags, we were able to discover lots of open ports with the respective service that is using that specific port. The following list contains all the ports `nmap` managed to discover:

```
PORT       STATE  SERVICE
22/tcp     open   ssh
135/tcp    open   msrpc
139/tcp    open   netbios-ssn
445/tcp    open   microsoft-ds
3306/tcp   open   mysql
3389/tcp   open   ms-wbt-server
4848/tcp   open   appserv-http
7676/tcp   open   imqbrokerd
8009/tcp   open   ajp13
8022/tcp   open   oa-system
8031/tcp   open   unknown
8080/tcp   open   http-proxy
8181/tcp   open   intermapper
8383/tcp   open   m2mservices
8443/tcp   open   https-alt
9200/tcp   open   wap-wsp
49152/tcp  open   unknown
49153/tcp  open   unknown
49154/tcp  open   unknown
49155/tcp  open   unknown
49160/tcp  open   unknown
```

We also managed to get some information about the Operating System on the target.

```
MAC Address: 08:00:27:9A:D8:98 (VirtualBox virtual NIC)
Device type: general purpose
Running: Microsoft Windows 7
OS CPE: cpe:/o:microsoft:windows_7::sp1
OS details: Microsoft Windows 7 SP1
Uptime guess: 0.002 days (since Thu Dec 10 14:37:59 2020)
Network Distance: 1 hop
```

By analysing the information about the open ports and the target's OS, we knew that we could dig deeper using `nmap`. To do so, we used 2 additional flags: -sV to probe open ports to determine service/version info, and -A which enables OS detection, version detection, script scanning, and traceroute.

```
nmap -v -sS -O -sV -A 172.20.5.2
```

With these flags we managed to get version information about the services, as well as specific details about some of them:

3.1.1 ssh

```
22/tcp    open  ssh              OpenSSH 7.1 (protocol 2.0)
| ssh-hostkey:
|   2048 df:ad:f5:41:2b:0e:20:7d:fa:e6:8d:ea:36:e1:8b:4d (RSA)
|_  521 82:d7:58:44:6e:9b:57:a4:a4:c0:28:7f:f4:52:1f:2d (ECDSA)
```

As we can see, the target's version of OpenSSH is 7.1. Since there are a lot of CVE's, we picked the most recent one: **CVE-2020-1577** [9].

3.1.2 msrpc

```
135/tcp   open  msrpc            Microsoft Windows RPC
49152/tcp open  msrpc            Microsoft Windows RPC
49153/tcp open  msrpc            Microsoft Windows RPC
49154/tcp open  msrpc            Microsoft Windows RPC
49155/tcp open  msrpc            Microsoft Windows RPC
```

`Nmap` returned these 5 ports running Microsoft Windows RPC. The last 4 of them were previously listed as unknown, but due to the added flags the service is now identified. Given that there is no version related to this service, we also have chosen the most recent CVE in our search: **CVE-2020-1113** [7].

3.1.3 netbios-ssn

```
139/tcp   open   netbios-ssn           Microsoft Windows netbios-ssn
```

Another service detected was Microsoft Windows netbios-ssn. For this service we also went for the one of the most recent and critical CVE's listed: **CVE-2020-13159** [8].

3.1.4 microsoft-ds

```
445/tcp   open   microsoft-ds        Windows Server 2008 R2 Standard 7601 Service Pack 1
                                              microsoft-ds
```

Another open port open is the port number 445 which has the microsoft-ds (SMB) service running on it, used mainly on Windows networks for sharing resources and execute remote commands. The most recent CVE that we managed to find and that affected our specific system was **CVE-2018-0749** [6].

3.1.5 mysql

```
3306/tcp   open   mysql                   MySQL 5.5.20-log
| mysql-info:
|   Protocol: 10
|   Version: 5.5.20-log
|   Thread ID: 4
|   Capabilities flags: 63487
|   Some Capabilities: FoundRows, LongPassword, Support41Auth,
| Speaks41ProtocolNew, ConnectWithDatabase, ODBCClient,
| SupportsLoadDataLocal, LongColumnFlag, SupportsTransactions,
| IgnoreSigpipes, SupportsCompression, DontAllowDatabaseTableColumn,
| IgnoreSpaceBeforeParenthesis, InteractiveClient, Speaks41ProtocolOld,
| SupportsMultipleResults, SupportsMultipleStatments, SupportsAuthPlugins
|   Status: Autocommit
|   Salt: |gmpjEU^X|*Y‘Z~/jImk
|_  Auth Plugin Name: mysql_native_password
```

Another service we discovered was MySQL 5.5.20. For this service we opted to list the most severe CVE we could find: **CVE-2016-6662** [5].

3.1.6 tcpwrapped

```
3389/tcp  open  tcpwrapped
| ssl-cert: Subject: commonName=vagrant-2008R2
| Issuer: commonName=vagrant-2008R2
| Public Key type: rsa
| Public Key bits: 2048
| Signature Algorithm: sha1WithRSAEncryption
| Not valid before: 2020-12-07T20:45:24
| Not valid after:  2021-06-08T20:45:24
| MD5:   4c9b 0ff2 2654 da3b 2cb4 38af bf19 5684
|_SHA-1: 549d a566 6360 d123 b84c fa53 a57f dbf8 c4a8 ff9d
|_ssl-date: 2020-12-10T20:26:53+00:00; +1s from scanner time.
```

On this specific port, we got a tcpwrapped response. As far as we could investigate, this happened because the program running on this port is protected by a tcpwrapper, which doesn't allow us to talk further with the protected program.

3.1.7 ssl/appserv-http

```
4848/tcp  open  ssl/appserv-http?
| ssl-cert: Subject: commonName=localhost/organizationName=Oracle
            Corporation/stateOrProvinceName=California/countryName=US
| Issuer: commonName=localhost/organizationName=Oracle
            Corporation/stateOrProvinceName=California/countryName=US
| Public Key type: rsa
| Public Key bits: 2048
| Signature Algorithm: sha256WithRSAEncryption
| Not valid before: 2013-05-15T05:33:38
| Not valid after:  2023-05-13T05:33:38
| MD5:   790d fccf 9932 2bbe 7736 404a 14e1 2d91
|_SHA-1: 4a57 58f5 9279 e82f 2a91 3c83 ca65 8d69 6457 5a72
|_ssl-date: 2020-12-10T20:26:53+00:00; +1s from scanner time.
```

In this case `nmap` managed to identify the service based on it's port rather than it's signature (represented by the question mark). For this service we opted to list the most recent CVE we could find: **CVE-2008-2398** [3].

3.1.8 java-message-service

```
7676/tcp  open  java-message-service Java Message Service 301
```

Another service detected was Java Message Service. For this service we also went for the most critical CVE's listed: **CVE-2015-5254** [4].

3.1.9 ajp13

```
8009/tcp   open   ajp13                 Apache Jserv (Protocol v1.3)
|_ajp-methods: Failed to get a valid response for the OPTION request
```

The next service we discovered was Apache Jserv, and we chose the most recent and most critical CVE listed one Mitre: **CVE-2020-1938** [10].

3.1.10 http

```
8022/tcp   open   http                  Apache Tomcat/Coyote JSP engine 1.1
| http-methods:
|   Supported Methods: GET HEAD POST PUT DELETE OPTIONS
|_  Potentially risky methods: PUT DELETE
|_http-server-header: Apache-Coyote/1.1
|_http-title: Site doesn't have a title (text/html;charset=UTF-8).
```

For this service we only managed to find one vulnerability afecting this specific version, which was **CVE-2005-2090** [1].

3.1.11 http

```
8080/tcp   open   http                  Sun GlassFish Open Source Edition  4.0
| http-methods:
|_  Supported Methods: GET
|_http-title: GlassFish Server - Server Running
```

For this service specific version we did not manage to find any CVE. The only CVE we found was related to version 4.1, but there was no mention of version 4.0.

3.1.12 ssl/intermapper

```
8181/tcp   open   ssl/intermapper?
| ssl-cert: Subject: commonName=localhost/organizationName=Oracle
Corporation/stateOrProvinceName=California/countryName=US
| Issuer: commonName=localhost/organizationName=Oracle
Corporation/stateOrProvinceName=California/countryName=US
| Public Key type: rsa
| Public Key bits: 2048
| Signature Algorithm: sha256WithRSAEncryption
| Not valid before: 2013-05-15T05:33:38
| Not valid after:  2023-05-13T05:33:38
| MD5:    790d fccf 9932 2bbe 7736 404a 14e1 2d91
|_SHA-1: 4a57 58f5 9279 e82f 2a91 3c83 ca65 8d69 6457 5a72
|_ssl-date: 2020-12-10T20:26:53+00:00; +1s from scanner time.
```

As we saw earlier, the service on this port was identified by the port number and not by it's signature. Intermapper is a network monitoring software, but we did not find any vulnerabilities linked to it.

3.1.13 ssl/http

```
8383/tcp  open  ssl/http          Apache httpd
| http-methods:
|   Supported Methods: GET HEAD POST PUT DELETE OPTIONS
|_  Potentially risky methods: PUT DELETE
|_http-server-header: Apache
|_http-title: Site doesn't have a title (text/html;charset=UTF-8).
| ssl-cert: Subject: commonName=Desktop Central/organizationName=Zoho
Corporation/stateOrProvinceName=CA/countryName=US
| Issuer: commonName=Desktop Central/organizationName=Zoho
Corporation/stateOrProvinceName=CA/countryName=US
| Public Key type: rsa
| Public Key bits: 1024
| Signature Algorithm: sha1WithRSAEncryption
| Not valid before: 2010-09-08T12:24:44
| Not valid after:  2020-09-05T12:24:44
| MD5:   3d69 ffa2 b100 7135 728e c704 3075 da29
|_SHA-1: 701e 2e6d f885 4c4f 0b29 8dff 03a2 c6f0 bac7 d315
|_ssl-date: TLS randomness does not represent time
```

The service running on this port is Apache httpd. With our search we were not able to get information about it's version, and because of that we couldn't find specific vulnerabilities associated.

3.1.14 https-alt

```
8443/tcp  open  ssl/https-alt?
```

As seen before, the service on this port was identified by the port number and not by it's signature. About the service running on this port we weren't able to extract more information.

3.1.15 wap-wsp

```
9200/tcp   open   wap-wsp?
| fingerprint-strings:
|   FourOhFourRequest:
|     HTTP/1.0 400 Bad Request
|     Content-Type: text/plain; charset=UTF-8
|     Content-Length: 80
|     handler found for uri [/nice%20ports%2C/Tri%6Eity.txt%2ebak]
and method [GET]
|   GetRequest:
|     HTTP/1.0 200 OK
|     Content-Type: application/json; charset=UTF-8
|     Content-Length: 311
|     "status" : 200,
|     "name" : "Jim Hammond",
|     "version" : {
|     "number" : "1.1.1",
|     "build_hash" : "f1585f096d3f3985e73456debdc1a0745f512bbc",
|     "build_timestamp" : "2014-04-16T14:27:12Z",
|     "build_snapshot" : false,
|     "lucene_version" : "4.7"
|     "tagline" : "You Know, for Search"
|   HTTPOptions:
|     HTTP/1.0 200 OK
|     Content-Type: text/plain; charset=UTF-8
|     Content-Length: 0
|   RTSPRequest, SIPOptions:
|     HTTP/1.1 200 OK
|     Content-Type: text/plain; charset=UTF-8
|_    Content-Length: 0
```

As we saw earlier, the service on this port was identified by the port number and not by it's signature. The only vulnerability found for this service was **CVE-2007-0795** [2].

3.1.16 Host script results

Despite not being process related information, it's important to know what kind of information we were able to get about the target system itself. As we can see, we were able to acquire detailed information about the target's OS, the system's time, system name and workgroup. This kind of information is sensitive and shows us the lack of protection that exists.

```
Host script results:
|_clock-skew: mean: 1h20m01s, deviation: 3h15m59s, median: 0s
| nbstat: NetBIOS name: VAGRANT-2008R2, NetBIOS user: <unknown>,
NetBIOS MAC: 08:00:27:9a:d8:98 (Oracle VirtualBox virtual NIC)
| Names:
|   VAGRANT-2008R2<00>    Flags: <unique><active>
|   WORKGROUP<00>         Flags: <group><active>
|_  VAGRANT-2008R2<20>    Flags: <unique><active>
| smb-os-discovery:
|   OS: Windows Server 2008 R2 Standard 7601 Service Pack 1
(Windows Server 2008 R2 Standard 6.1)
|   OS CPE: cpe:/o:microsoft:windows_server_2008::sp1
|   Computer name: vagrant-2008R2
|   NetBIOS computer name: VAGRANT-2008R2\x00
|   Workgroup: WORKGROUP\x00
|_  System time: 2020-12-10T12:26:43-08:00
| smb-security-mode:
|   account_used: <blank>
|   authentication_level: user
|   challenge_response: supported
|_  message_signing: disabled (dangerous, but default)
| smb2-security-mode:
|   2.02:
|_    Message signing enabled but not required
| smb2-time:
|   date: 2020-12-10T20:26:39
|_  start_date: 2020-12-10T19:38:43
```

Despite not being able to uncover all services running and their respective versions, `nmap` was the best tool for the job, since other tools we tried were not able to perform **port scanning** and **banner grabbing** with the same quality.

3.2 Q2

The results we obtained by scanning the host were quite interesting. In total, Nessus discovered **47 vulnerabilities**, sorting them by severity, so that we can have a better idea of the entire state of the system.

Nessus also has a **Remediations** list that has some advised actions that mitigate some vulnerabilities. For the vulnerabilities found, Nessus has some information such as it's **severity**, **how it can exploit it**, related **CVE's**, and more. Nessus also listed all the open ports, as well as all services running on them. There is also relevant information such as the target's **device typ**e and **OS**. When comparing

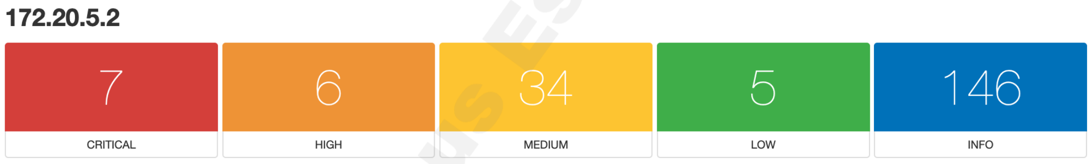


Figure 10: Scan Summary

the results from our scan with the results of the automated one we found many similarities. In first place we investigated the ports where we struggled and found out that they either had a TLS service or a webserver tunning on them. Nessus managed to identify the services and even get the versions of TLS used. The only exception was port 8484, where Nessus wasn't able to find any information, hinting that it may be protected by a TCP Wrapper. Nessus managed to extract information about TLS and SSL to an extent that we did not manage. The same happened with SSH, where Nessus we only got versions and hostkeys. Nessus even listed SSH Algorithms and supported languages.

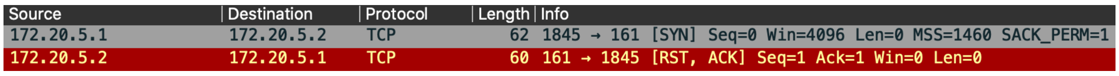
Nessus managed to find **22 services** against the **15** we found. The OS detection made by Nessus returned the same result as we have with our search. With this comparison, the main conclusion we take is that with Nessus we manage to obtain more information compared to what we did before. Also the quality of the information obtained is superior, and quicker to obtain.

3.3 Q3

For this question we picked the following events from our IDS:

```
[**] [1:1418:11] SNMP request tcp [**]
[Classification: Attempted Information Leak] [Priority: 2]
12/13-19:04:13.826861 172.20.5.1:1845 -> 172.20.5.2:161
TCP TTL:64 TOS:0x0 ID:0 IpLen:20 DgmLen:48 DF
******S* Seq: 0xF614612C  Ack: 0x0  Win: 0x1000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK
[Xref => http://cve.mitre.org/cgi-bin/cvename.cgi?name=2002-0013]
[Xref => http://cve.mitre.org/cgi-bin/cvename.cgi?name=2002-0012]
[Xref => http://www.securityfocus.com/bid/4132]
[Xref => http://www.securityfocus.com/bid/4089]
[Xref => http://www.securityfocus.com/bid/4088]
```

By analysing the traffic captured by wireshark, we found 2 entries related to this event.



Source	Destination	Protocol	Length	Info
172.20.5.1	172.20.5.2	TCP	62	1845 → 161 [SYN] Seq=0 Win=4096 Len=0 MSS=1460 SACK_PERM=1
172.20.5.2	172.20.5.1	TCP	60	161 → 1845 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0

Figure 11: Packets in Wireshark

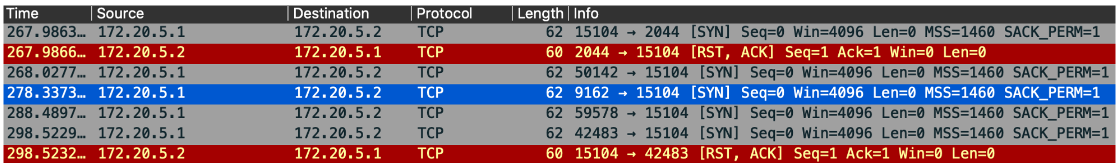
By analysing the packet sent to the target we noticed a **SYN flag**, which tells us that our host is trying to initiate connection with the target. But when we analyse the tcp segment that the target sends to our host, we noticed that in addition to the **ACK flag**, which isn't odd in this case, we have a **RST flag**. This may be caused by a SYN packet trying to establish a connection to a port on which no process is listening.

By searching the Snort Rules for this specific SID we found the **rule** for this entry. The rule states that there was detected the presence of **SNMP** protocol, which has a lot of vulnerabilities used for **Denial of Service** and to **gain privileges**.

The CVE's related to this vulnerability are **CVE-2002-0013** and **CVE-2002-0012**.

```
[**] [1:249:8] DDOS mstream client to handler [**]
[Classification: Attempted Denial of Service] [Priority: 2]
12/13-19:04:24.593643 172.20.5.1:9162 -> 172.20.5.2:15104
TCP TTL:64 TOS:0x0 ID:0 IpLen:20 DgmLen:48 DF
******S* Seq: 0x1C77AAE7  Ack: 0x0  Win: 0x1000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK
[Xref => http://cve.mitre.org/cgi-bin/cvename.cgi?name=2000-0138]
[Xref => http://www.whitehats.com/info/IDS111]
```

By analysing the traffic captured by wireshark, we found 7 entries related to this event, being the highlighted one that is registered in this event.



Time	Source	Destination	Protocol	Length	Info
267.9863…	172.20.5.1	172.20.5.2	TCP	62	15104 → 2044 [SYN] Seq=0 Win=4096 Len=0 MSS=1460 SACK_PERM=1
267.9866…	172.20.5.2	172.20.5.1	TCP	60	2044 → 15104 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
268.0277…	172.20.5.1	172.20.5.2	TCP	62	50142 → 15104 [SYN] Seq=0 Win=4096 Len=0 MSS=1460 SACK_PERM=1
278.3373…	172.20.5.1	172.20.5.2	TCP	62	9162 → 15104 [SYN] Seq=0 Win=4096 Len=0 MSS=1460 SACK_PERM=1
288.4897…	172.20.5.1	172.20.5.2	TCP	62	59578 → 15104 [SYN] Seq=0 Win=4096 Len=0 MSS=1460 SACK_PERM=1
298.5229…	172.20.5.1	172.20.5.2	TCP	62	42483 → 15104 [SYN] Seq=0 Win=4096 Len=0 MSS=1460 SACK_PERM=1
298.5232…	172.20.5.2	172.20.5.1	TCP	60	15104 → 42483 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0

Figure 12: Packets in Wireshark

The first thing that we noticed was that the packet was similar to the one before, with the exception that this time, it was being sent periodically to the target, not getting a response sometimes. The packet that our host sent from port 50142 didn't get a response, and when the next packet was sent (from port 9162), the IDS registered the activity.

By searching the Snort Rules for this specific SID we found the **rule** for this entry. The rule states that there was detected the presence of a distributed denial of service(DDOS) attack master, agent, or zombie.

The CVE related to this vulnerability is **CVE-2000-0138**.

3.4 Q4

When analysing the output of the IDS and the output of Snort we noticed that some of the vulnerabilities listed on the IDS didn't show up on the vulnerability scanner. An example of this are the events listed in the previous question, which aren't listed on the report generated by Nessus.

To discover what might be causing this we looked at what both IDS and vulnerability scanners do. The IDS basically monitors the network or system registering all kind of suspicious or malicious activity. Any intrusion activity or violation registered. The Scanner on the other hand, works in a different way, following a set of automated rules and redirecting it's focus on the go. For example, if the Scanner detects that the target system is a Windows host, it won't try to use vulnerabilities or exploits for Linux hosts. These kind of situations create some differences on what the IDS considers to be a threat and what a Scanner considers to be a threat.

3.5 Q5

For this question we chose the following vulnerabilities:

- 57608 - SMB Signing not required (Medium)
- 90192 - ManageEngine Desktop Central 8 / 9 < Build 91100 Multiple RCE (Critical)
- 134862 - Apache Tomcat AJP Connector Request Injection (Ghostcat) (High)

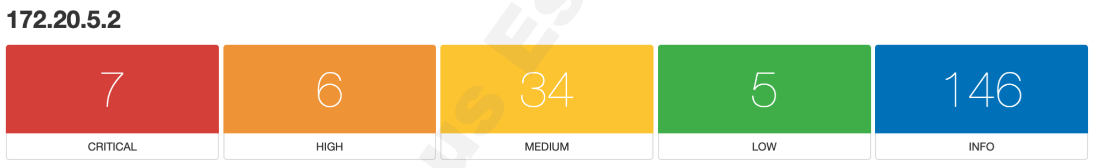


Figure 13: Original Vulnerabilities

In order to resolve the first vulnerability we had to **enforce message signing in the host's configuration**. To do that we edited the windows registry key **HKLM\System\CurrentControlSet\Services\LanManServer\Parameters\ RequireSecuritySignature** and set the flag to 1. What this does is requiring signatures for SMB communication. After this we rebooted the machine and performed a new scan, with the following results.

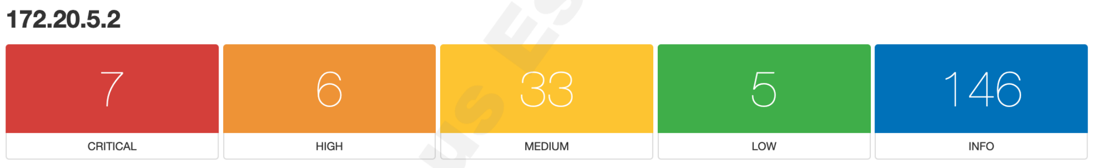
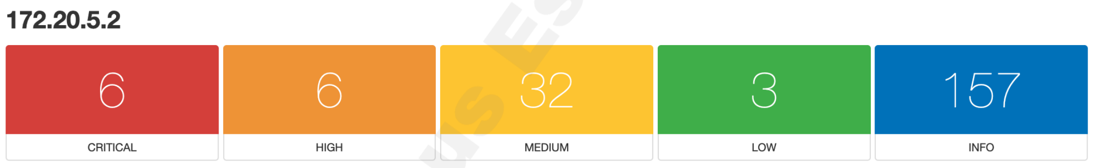


Figure 14: Vulnerabilities after first fix

To fix the second vulnerability we updated **ManageEngine** to the latest version (10.0.650). The software update fixed the wanted vulnerability as we can see by running a new scan:

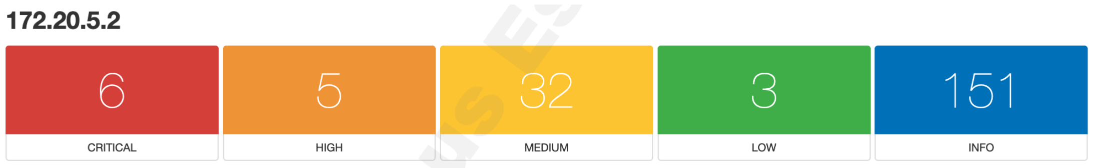

172.20.5.2

6	6	32	3	157
CRITICAL	HIGH	MEDIUM	LOW	INFO

Figure 15: Vulnerabilities after second fix

Despite fixing the target vulnerability and some additional ones that were related with this same fix, there were also some new additions.

The last vulnerability we fixed was related to **Apache Tomcat**. The solution we applied was to update Tomcat to version 9.5.61. This fixed the vulnerability in question as we can see:


172.20.5.2

6	5	32	3	151
CRITICAL	HIGH	MEDIUM	LOW	INFO

Figure 16: Vulnerabilities after third fix

What we learned from this question is that with some minor system tweaks and some regular update cycles, we are able to mitigate a lot of security flaws on a system.

4 Conclusions

With part A we discovered how different security postures can affect a real product of a company and we discovered some of the different approaches taken by companies of different sizes. With part B we gained some technical insights about Vulnerability Scanners and IDS. With these tools we gained an insight about what these tools are capable of in comparison to performing manual searches for vulnerabilities. Another aspect we learned was the differences between an IDS and a Vulnerability Scanner, as well as how to fix the Vulnerabilities on a system.

References

[1] *CVE-2005-2090*. URL: `https://nvd.nist.gov/vuln/detail/CVE-2005-2090` (visited on 12/12/2020).

[2] *CVE-2007-0795*. URL: `https://nvd.nist.gov/vuln/detail/CVE-2007-0795` (visited on 12/12/2020).

[3] *CVE-2008-2398*. URL: `https://nvd.nist.gov/vuln/detail/CVE-2008-2398` (visited on 12/12/2020).

[4] *CVE-2015-5254*. URL: `https://nvd.nist.gov/vuln/detail/CVE-2015-5254` (visited on 12/12/2020).

[5] *CVE-2016-6662*. URL: `https://nvd.nist.gov/vuln/detail/CVE-2016-6662` (visited on 12/11/2020).

[6] *CVE-2018-0749*. URL: `https://nvd.nist.gov/vuln/detail/CVE-2018-0749` (visited on 12/11/2020).

[7] *CVE-2020-1113*. URL: `https://nvd.nist.gov/vuln/detail/CVE-2020-1113` (visited on 12/11/2020).

[8] *CVE-2020-13159*. URL: `https://nvd.nist.gov/vuln/detail/CVE-2020-13159` (visited on 12/11/2020).

[9] *CVE-2020-15778*. URL: `https://nvd.nist.gov/vuln/detail/CVE-2020-15778` (visited on 12/10/2020).

[10] *CVE-2020-1938*. URL: `https://nvd.nist.gov/vuln/detail/CVE-2020-1938` (visited on 12/12/2020).

[11] *Farfetch LinkenIn*. URL: `https://www.linkedin.com/jobs/view/2332825996/?refId=5466889361608491278523&trackingId=dyMSaZRcD4J%5C%2FG4uaijQKGQ%5C%3D%5C%3D&trk=d_flagship3_company` (visited on 12/20/2020).