

Internship Report – Cybersecurity

The Redynox Cybersecurity Team

Report Prepared By : Issa Hassan Youssouf

Company : Redynox

Internship period : June 16 to July 16, 2025

Reporting Manager : *Mr. Ankit Mathur*

Institution : University Institute of the Gulf of Guinea

TABLE OF CONTENTS

INTRODUCTION

I. Task 1 : Introduction to Network Security Basics

- 1) Learn Network Security concept
- 2) Implement Basic Measures
- 3) Monitor Network Traffic
- 4) Reflect on Security Best Practices

II. Task 2 : Introduction to Web Application Security

- 1) Setup
- 2) Perform Basic Vulnerability Analysis
- 3) Exploitation manuelle

III. Task 3: LinkedIn Engagement

CONCLUSION

Thanks

I would like to warmly thank **Redynox** for giving me the opportunity to welcome me to their team and for allowing me to discover tools such as OWASP ZAP, Wireshark and concrete cybersecurity methodologies.

I would also like to thank my internship supervisor **Mr. Ankit Mathur** for his support, his valuable advice and his availability throughout this experience.

This internship allowed me to strengthen my technical skills, better understand current digital risks and confirm my interest in the field of cybersecurity.

INTRODUCTION

As part of my training, I had the opportunity to do a practical internship at the company Redynox, which specialises in cybersecurity. This internship was an enriching immersion in the world of IT security, especially on technical aspects such as network security, web vulnerability analysis and the use of professional tools such as Wireshark, WebGoat, OWASP ZAP and configure basic security configurations, such as changing default passwords and enabling network encryption (WPA2 or WPA3). I was also encouraged to develop my professional visibility through publications on LinkedIn.

This document is a detailed, step-by-step guide retracing all the tasks I performed during this internship, with the Parrot OS environment as the main support.

I. TASK 1 : INTRODUCTION TO NETWORK SECURITY BASICS

Network security refers to the policies, practices, and technologies used to protect the integrity, confidentiality, and availability of computer networks and data. It involves preventing unauthorized access, misuse, modification, or denial of a network and its accessible resources.

Key Goals of Network Security :

Confidentiality is a set of rules that prevents sensitive information from being disclosed to unauthorized people, resources and processes. Methods to ensure confidentiality include data encryption, identity proofing and two factor authentications.

Integrity ensures that system information or processes are protected from intentional or accidental modification. One way to ensure integrity is to use a hash function or checksum.

Availability means that authorized users are able to access systems and data when and where needed and those that do not meet established conditions, are not. This can be achieved by maintaining equipment, performing hardware repairs, keeping operating systems and software up to date, and creating backups.

1) LEARN NETWORK SECURITY CONCEPT

Different types of network threats :

Malware is a type of software designed to cause harm to computer systems, networks, or devices. The term malware is short for malicious software. Malware can be developed in many forms, including viruses, worms, Trojan horses, ransomware, spyware, and adware.

Malware can cause many problems, from disrupting regular system and network operations to stealing sensitive information, such as passwords and business data. Malware can also be used to install hidden remote access, allowing unauthorized access to a system.

Protecting computer systems and devices from malware is essential; keeping software up-to-date, using antivirus software, and practicing safe browsing habits all help. In addition, be vigilant and cautious when downloading software or opening email attachments, as these can be common vectors for malware infection.

Virus :

A virus is a type of malicious software that replicates itself and infects other programs or files on a computer. A virus typically spreads by attaching itself to a legitimate program or file. It can then spread to other computers when that file is shared or when the infected program runs on another computer.



A virus can cause many problems, including slowing computer performance, stealing sensitive information, and even destroying data. Some viruses can also be used as a tool for more extensive cyberattacks, such as distributed denial-of-service (DDoS) attacks.

Damage from viruses includes :

- Corrupting files,
- Computer slowdown,
- Taking over essential functions of the operating system.

Worm

A computer worm is an independent malware program that reproduces itself to infect other computers. It can spread to other computers without attaching to an existing program and can damage the network.



Damage from worms includes :

- Bandwidth consumption,
- Stopping active anti-malware service,
- Immobilizing Safe Mode,
- Hindering operating systems auto-update.

Worms can :

- Cause significant damage by consuming network bandwidth, slowing computer performance, and compromising the security of sensitive information,
- Spread rapidly and cause widespread infections, making them a significant threat to computer security,
- Exploit vulnerabilities in operating systems or software, allowing attackers to gain unauthorized access to computer systems.

Protect against worms by :

- Keeping software up-to-date,
- Using firewalls,
- Practicing safe browsing habits.

Using antivirus software and intrusion detection systems help to detect and respond to worm infections.

Phishing

Phishing is a cyberattack that tricks individuals into sharing sensitive information through fake emails or directed to websites that appear legitimate. The cybercriminals who use phishing scams masquerade as real:

- Prominent companies,
- Banks and financial institutions,
- Government offices,
- Credit card businesses,
- Charities and nonprofits.



They are looking for sensitive information such as passwords and financial information.



Phishing attacks take advantage of communication tools that people use every day and can happen in various ways: email, text messages, phone calls, or instant messages.

Features of a phishing scam will include one or more of the following.

Request for immediate action and urgent offers

- Claim that there's a problem with your account or password,
- Request to confirm your password or account information,
- Say that there's been suspicious activity on your account,
- Notify you of a failed or missed payment,
- Offer you a free coupon or gift, or say that you're eligible for a refund

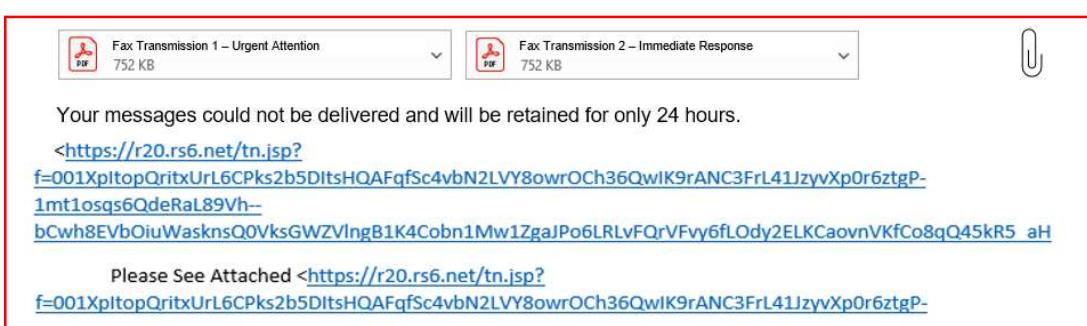
Digital Communication (Email, Text, Messaging)

- Generic information to the sender and from an unknown sender,
- Unrequested invoice or message attachment or invoice,
- Mismatched subject and contents,
- Spelling errors and basic punctuation mistakes,
- Similar characters for numbers and vice versa. For example, the number 1 instead of capital I,
- Ask you to click on a link or provide information for a new security update,
- A hyperlink to what seems to be a legitimate website,
- Virus warnings.

In the examples below, can you recognize at least 14 phishing scam items.

Fax Portal Report <noreply@karensdishesbydesign.com>
To

3:09 PM

A screenshot of an email inbox showing a fax portal report. The recipient is noreply@karensdishesbydesign.com. The message was sent at 3:09 PM. Two attachments are listed: "Fax Transmission 1 – Urgent Attention" (752 KB) and "Fax Transmission 2 – Immediate Response" (752 KB). A red box highlights these two attachments. Below the attachments, a message states: "Your messages could not be delivered and will be retained for only 24 hours." It includes a link to https://r20.rs6.net/tn.jsp? followed by a long URL. At the bottom, it says "Please See Attached" followed by another link to https://r20.rs6.net/tn.jsp? followed by the same long URL.

E

Elvis Edorh <elvisedorhtg93@gmail.com>
À moi ▾

3 août 2022 19:27 ☆ ☺ ↶ ⋮

Dear Issa Hassan Yousouf,

Thank you for your genuine interest in this transaction, I am very glad to read from you also noted that you are a noble, matured, and trustworthy person whom I can rely on for your capabilities to handle this transaction. I am very confident doing this business with you.

Like I told you earlier, I was looking forward to a foreign partner that I can entrust this deal to and finally I have found you. Let me go straight to explain more on the event that led to this before I expatiate on the project proper. My late client was a big customer of the Bank. He operated a coded account with the Bank here before he died on the 30th December, 2015, along with his wife and their only son, in an auto car accident.

Like I said before, due to this issue on my hands as his personal attorney, the bank have giving my law firm a mandate to locate any of his relative member or the money will be confiscated by the Togo government, now it became necessary for me to seek your assistance, I appreciate the fact that you are ready to assist me in executing this project, and also you will help me in investing my money in your country or somewhere else, I am quite certain about that. You should not have anything to worry about, I will do everything legally required to ensure that the project goes smoothly, it shall pass through all Laws of International Banking and beneficiary claims, you have my word.

Having resolved to entrust this transaction into your hands, I want to remind you that it needs your commitment and diligent follow up. If you work seriously, the entire transaction should be over in a couple of days...

The information I need is as follows;

Full Names:
Country of Origin:
Address:
Occupation:
Date of Birth:
Marital Status:
Telephone numbers:
Email:

As soon as you give me your hand of cooperation I will give you an application letter you will send to the bank and it will also prove to me that I am transacting with the correct person and as soon as I get these from you I will commence the paperwork that will place you as the beneficiary of the funds. I hope you will understand why I need all these, the money in question is big and I want to ensure that I know you well before I proceed to give you all the details to commence the project,

Attached is a copy of my Attorney license and passport for your reference,

I propose that the sharing mode of the funds will be as follows: 40% to you for your part in this transaction for standing as the next of kin to my deceased client and 40% for me. While, the remaining 20% shall be donated to charity homes around the world. But if you feel otherwise with the share ratio, do not hesitate to notify me.

Ensure that you keep this project confidential; do not discuss it with anybody, because of the confidential nature of this transaction and my work.

Please reply soonest.

Watch out for words and phrases such as :

1. We suspect unauthorized use or transactions [on your account](#).
2. We will lock or close your account [if you do not immediately](#) confirm your identity.
3. Click the link to verify [your account is not compromised](#)

Trojan

A Trojan Horse is a type of malware that disguises itself as a legitimate program or file. Unlike a virus, which typically replicates itself and infects other programs or files, a Trojan horse operates by hiding itself within a seemingly harmless program or file.



Once a Trojan horse is installed on a computer, it can carry out many malicious activities including :

- Stealing sensitive information,
- Installing other malware,
- Taking control of the infected computer,
- Create backdoors to get information the system,
- Allowing other attackers to gain unauthorized access to a system.

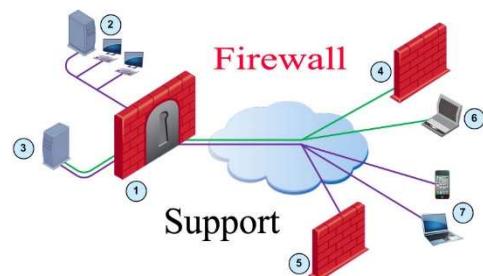
Damage from Trojan Horse includes :

- Crashing the computer,
- Deleting files,
- Corrupting data,
- Logging keystrokes.

Basic security concepts :

Firewall

A firewall is a network security device that acts as a barrier between a trusted internal network and untrusted external networks, such as the Internet. It monitors and controls incoming and outgoing network traffic based on predefined security rules. Its main role is to protect networks and systems from unauthorized access, malicious activities, and potential threats.



Operation and Importance :

- Traffic Filtering: Firewalls inspect network data packets and enforce security policies to either allow or block traffic.
- Threat Protection : They serve as a first line of defense against viruses, hackers, DDoS attacks, and unauthorized access attempts, according to Fortinet.
- Types of Firewalls : Firewalls can come in the form of hardware, software, Software-as-a-Service (SaaS), or virtual (cloud-based) solutions. There are also different categories, such as stateless, stateful, application-level, next-generation firewalls (NGFW), and personal firewalls.

In summary, a firewall is a critical component of cybersecurity, acting as a gatekeeper to secure networks by controlling and filtering data traffic.

Encryption

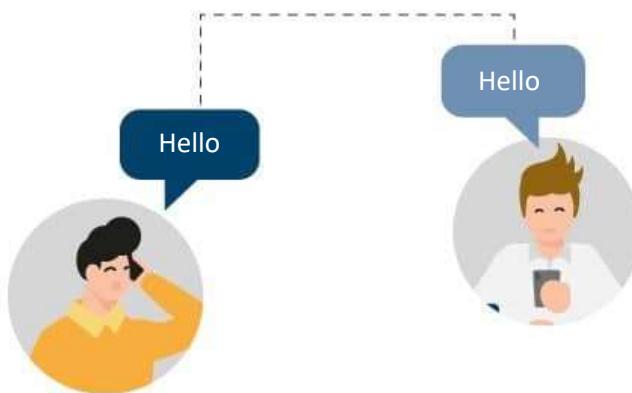
Let's begin our exploration of cryptography with encryption.

Encryption is the process of protecting data by rendering it unintelligible. Our focus will primarily be on ensuring data confidentiality.

The reverse operation, decryption, is only possible if one possesses a secret element known as the key

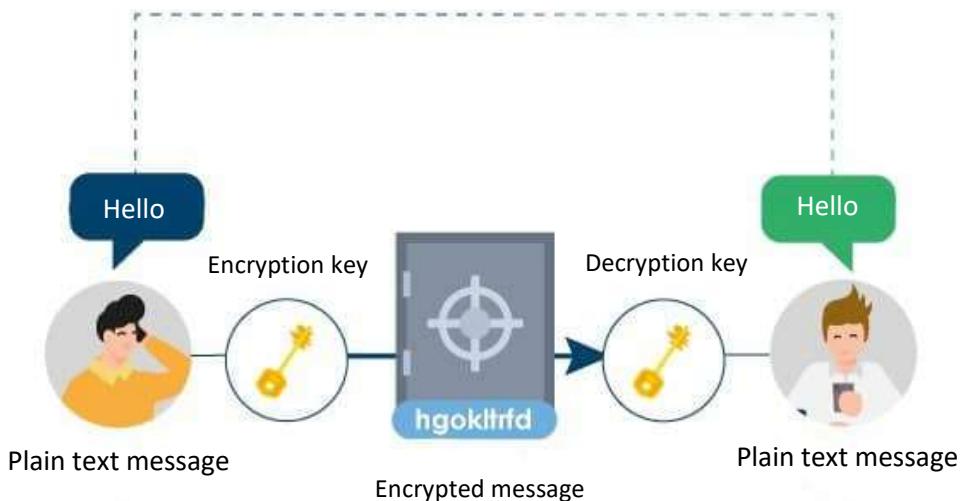


By convention, the original message to be protected is referred to as the plaintext, and the result of encryption is called the ciphertext.



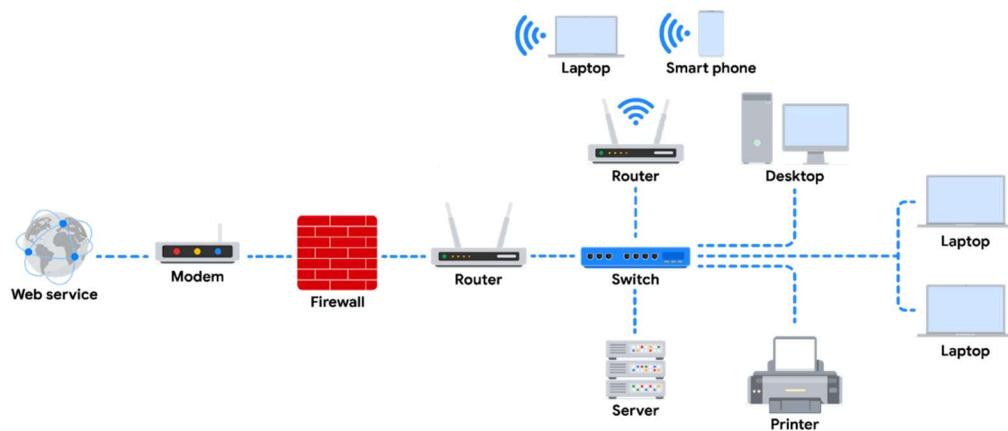
In other words, encryption is the process of transforming plaintext into ciphertext using a (unique) key, so that it can only be understood by those who possess that key.

This allows secure transmission of a message to a recipient over an otherwise untrusted medium—meaning a channel where the data in transit could potentially be intercepted or altered



Secure network configurations

Secure network configurations involve implementing security measures to protect the confidentiality, integrity, and availability of data and network resources from unauthorized access and cyberattacks. This includes using robust security protocols, properly configuring firewalls, enforcing strong authentication, and proactively monitoring the network.



Key Elements of Secure Network Configurations :

- Robust Security Protocols :

Utilize encryption protocols such as IPsec, SSL/TLS, and DTLS, as well as solutions like VPNs (Virtual Private Networks) to secure communications. For Wi-Fi networks, prioritize the use of WPA3 or WPA2/WPA3 Transitional modes.

- Firewalls :

Configure hardware or software firewalls with custom rules to block unsecured traffic and vulnerable ports (such as port 445, which is associated with WannaCry).

- Access Control and Authentication :

Implement strong authentication mechanisms, including multi-factor authentication (MFA) and the use of strong, complex passwords.

- Application and Service Configuration :

Harden application configurations, including setting trusted certificate authorities (or using custom ones), and disabling unnecessary plaintext protocols (such as Telnet).

- Network Monitoring and Anomaly Detection :

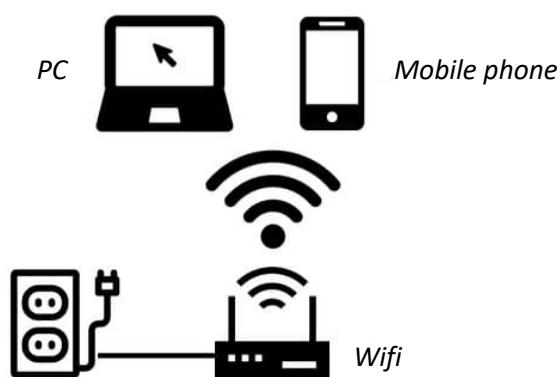
Actively monitor network traffic and activity to detect any suspicious or unauthorized behavior in real time, allowing for quick response to potential threats.

- Configuration Management :

Regularly audit the configurations of network devices and services to ensure they remain secure and aligned with best practices.

2) Implement basic measures

Set up a simple network environment



The architecture of my home network is composed of a PC, a mobile phone, and a Wifi Box from MTN.

The screenshot shows the MTN 4G CPE web interface. At the top, there's a header bar with the MTN logo, signal strength indicators (LTE MTN), and a language dropdown set to "Français". Below the header is a navigation menu with tabs: Accueil, Réglages rapides, Paramètres de l'appareil (which is selected and highlighted in yellow), SMS, USSD, Contacts, Pare-feu, and Les paramètres du système. The main content area is titled "Paramètres de l'appareil" and contains two sections: "Appareils connectés sans fil" and "Appareils connectés par câble". The "Appareils connectés sans fil" section has a red box around it, showing two entries: "Main SSID" with IP 192.168.2. The "Appareils connectés par câble" section shows "Pas de données" and a "Filtre MAC" button.

Here are the equipment connected to the Wifi Box: 1 and 2.

Now let's try the Ping to see the connectivity :

- The interface of the **Wifi Box** to the **PC**

The screenshot shows the MTN 4G CPE web interface under the "System Settings" tab. On the left, there's a sidebar with "System Setting", "Ping" (which is selected and highlighted in yellow), "Trace", "Network Tools" (selected), "System Upgrade", and "Reboot". The main content area is titled "System Settings" and contains a "Ping" section. It has fields for "Count" (set to 20) and "URL or IP" (set to 192.168.2). Below these are "start" and "stop" buttons. A red box highlights the output window, which displays the ping statistics:

```

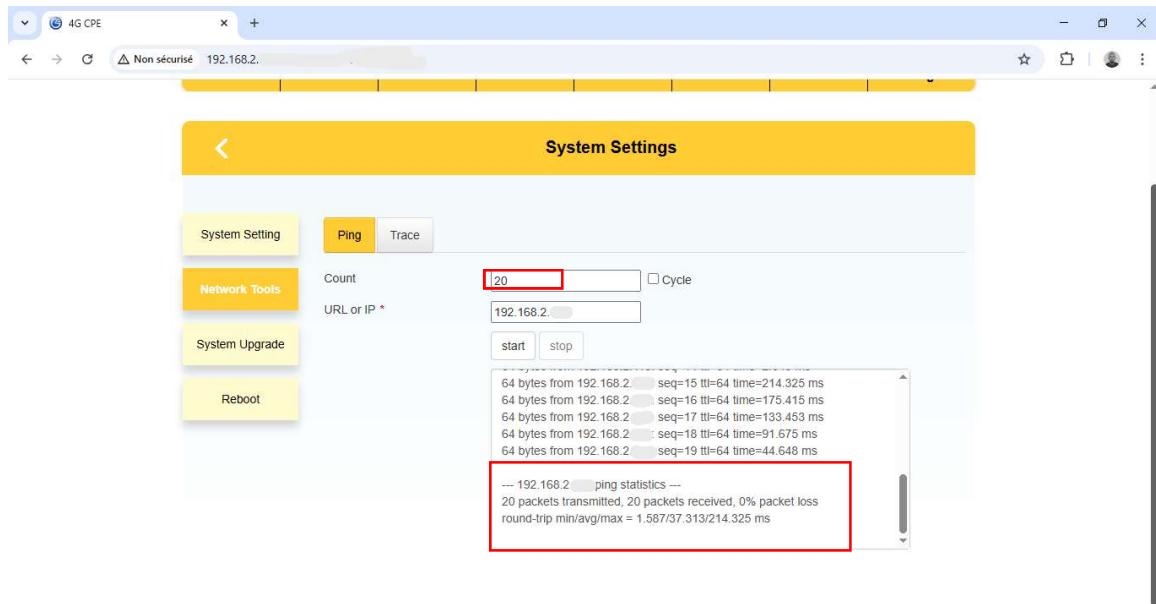
64 bytes from 192.168.2 seq=15 ttl=64 time=64.331 ms
64 bytes from 192.168.2 seq=16 ttl=64 time=88.104 ms
64 bytes from 192.168.2 seq=17 ttl=64 time=213.257 ms
64 bytes from 192.168.2 seq=18 ttl=64 time=31.769 ms
64 bytes from 192.168.2 seq=19 ttl=64 time=5.311 ms
...
--- 192.168.2 ping statistics ---
20 packets transmitted, 20 packets received, 0% packet loss
round-trip min/avg/max = 1.587/38.076/213.257 ms

```

Result: ---192.168.2.*** Ping Statistics----

20 packets transmitted, 20 packets received, 0% packets loss

- The interface from the Wifi Box to the mobile phone



Result: ---192.168.2.*** Ping Statistics----

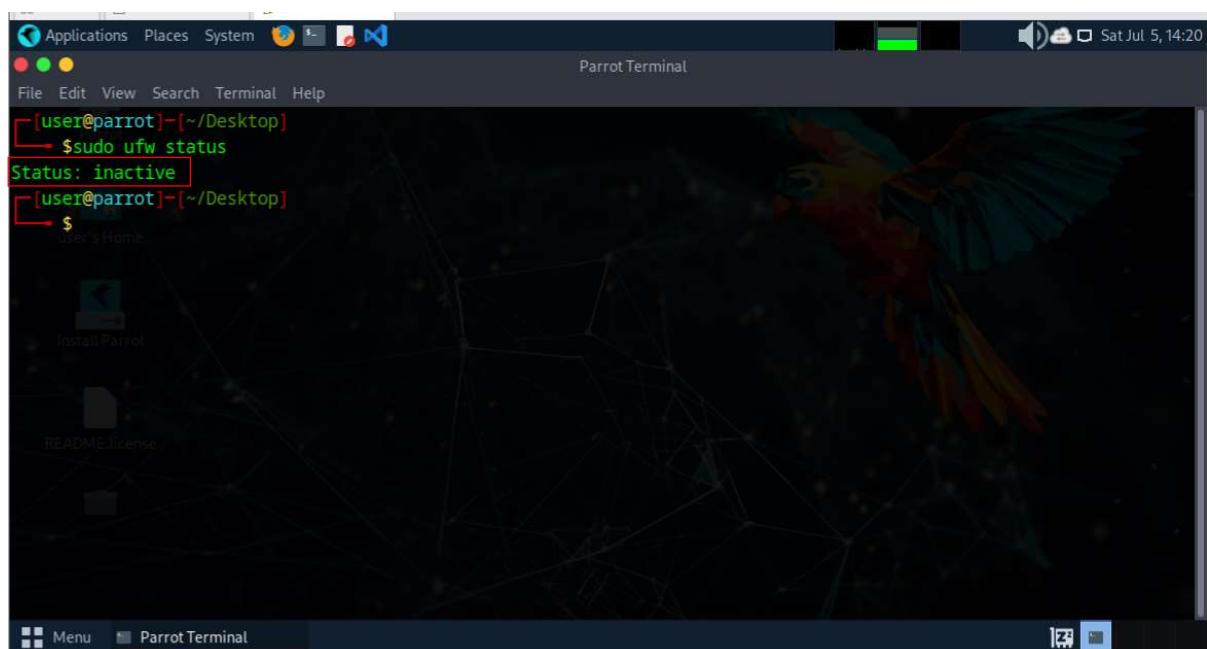
20 packets transmitted, 20 packets received, 0% packets loss

Enable and configure a basic firewall

Configuring ufw (Uncomplicated Firewall) on a Debian system with the Parrot OS GUI:

Enable and configure the UFW Firewall

- Open the terminal.
- Tape : `sudo ufw status` to check Firewall.



Status : Inactive,

- Tape : `sudo ufw enable`

```
[user@parrot]~/.Desktop]$ sudo ufw status
Status: inactive
[user@parrot]~/.Desktop]$ sudo ufw enable
Firewall is active and enabled on system startup
[user@parrot]~/.Desktop]$
```

Firewall is active and enabled on system startup.

- Block an FTP port : `sudo ufw deny 21`
- Allows web traffic :
`sudo ufw allow 80`

Allows inbound traffic on port 80 (HTTP).

```
sudo ufw allow 443
```

Allows inbound traffic on port 443 (HTTPS).

```
[user@parrot]~/.Desktop]$ sudo ufw status
Status: inactive
[user@parrot]~/.Desktop]$ sudo ufw enable
Firewall is active and enabled on system startup
[user@parrot]~/.Desktop]$ sudo ufw deny 21
Rule added
Rule added (v6)
[user@parrot]~/.Desktop]$ sudo ufw allow 80
Rule added
Rule added (v6)
[user@parrot]~/.Desktop]$ sudo ufw allow 443
Rule added
Rule added (v6)
[user@parrot]~/.Desktop]$
```

Tape : `sudo ufw status verbose` to view the current status of the firewall with details.

```
Rule added (v6)
[user@parrot]~[~/Desktop]
$ sudo ufw status verbose
Status: active
Logging: on (low)
Default: deny (incoming), allow (outgoing), disabled (routed)
New profiles: skip

To           Action    From
--           ----     ---
21           DENY IN  Anywhere
80           ALLOW IN Anywhere
433          ALLOW IN Anywhere
21 (v6)      DENY IN  Anywhere (v6)
80 (v6)      ALLOW IN Anywhere (v6)
443 (v6)    ALLOW IN  Anywhere (v6)

[user@parrot]~[~/Desktop]
$
```

ACTIVE RULES:

Port	Action	From
21	DENY IN	Anywhere
80	ALLOW IN	Anywhere
433	ALLOW IN	Anywhere
21 (v6)	DENY IN	Anywhere (v6)
80 (v6)	ALLOW IN	Anywhere (v6)
443 (v6)	ALLOW IN	Anywhere (v6)

Basic security configurations, such as changing default passwords.

To change the default password, we choose the network of the MTN Box that was configured previously. With the address of the default gateway of the network, we will go to the Chrome browser on Windows 10 home and before going to the browser, let's launch the terminal on Windows with the `ipconfig` command to identify the address of the default gateway, here is the step as follows :

```
C:\ Invité de commandes
C:\Users\issah>ipconfig

Carte réseau sans fil Wi-Fi :
  Suffrage DNS propre à la connexion. . . . . .
  Adresse IPv6 de liaison locale. . . . . .
  Adresse IPv4. . . . . : 192.168
  Masque de sous-réseau. . . . . : 255.255
  Passerelle par défaut. . . . . : 192.168

C:\Users\issah>
```

Here are the details about the Wi-Fi wireless network card :

Connection-specific DNS suffix..... :

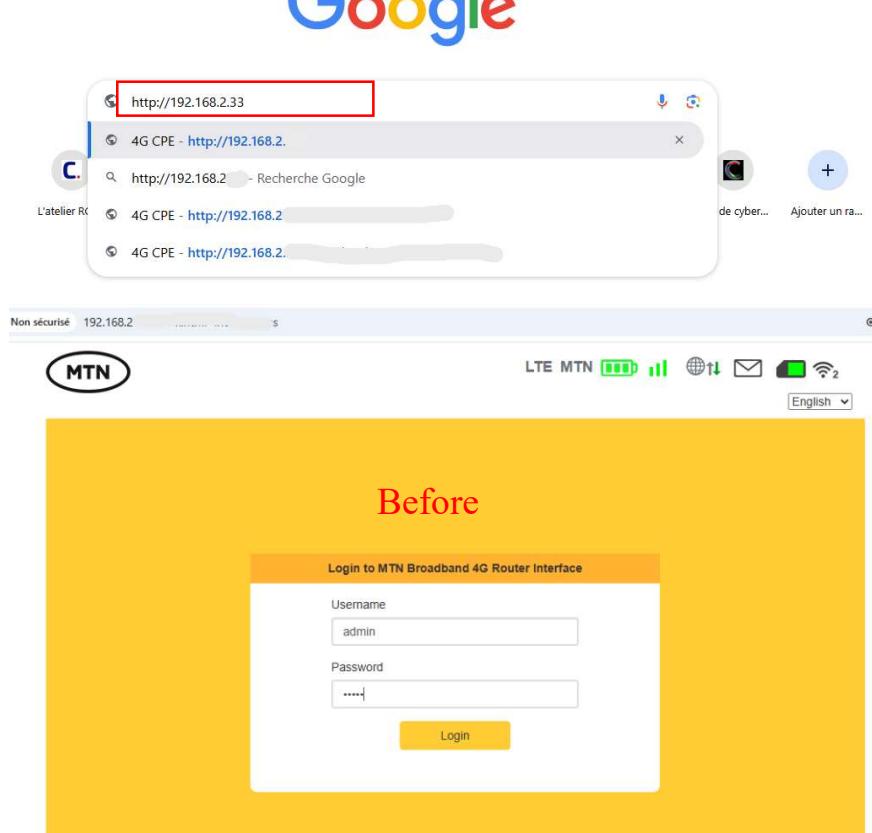
IPv6 address of the local link..... : ****

IPv4 address.....: 192.168.***

Subnet mask..... : 255.255.***

Default gateway.....: 192.168.***

With the default gateway, go to the Chrome browser



When displaying this interface above, the username and password are the default for each network device:

Username : admin

Password : admin

The screenshot shows the MTN 4G CPE web interface. At the top, there's a navigation bar with tabs: Home, Quick Settings, Device Settings, SMS, USSD, Phonebook, Firewall, and System Settings. The System Settings tab is highlighted with a red box. Below the navigation bar, there are several sections: Internet (with sub-options like Internet Connection, IMEI, IMSI, LAN Domain, WAN IP Address, WAN IPv6 Address, Statistics), Advanced (with sub-options like Firmware Version, RSRP(dBm), SINR, Cell Id, BAND, PhysCellId), and Home Network (with sub-options like Wi-Fi, Network Name(SSID), Max Access Number, Wi-Fi MAC, LAN MAC, Channel Bandwidth). The overall layout is a grid of tables.

Under the system settings tab, you can change the default password.

The screenshot shows the System Settings page within the MTN 4G CPE web interface. The top navigation bar has tabs for Home, Quick Settings, Device Settings, SMS, USSD, Phonebook, Firewall, and System Settings. The System Settings tab is active and highlighted with a yellow background. Below the tabs, there are four buttons: System Setting, Time Setting, Modify Password, and Restore Factory Setting. The Modify Password button is highlighted with a red box. Below these buttons, there are four input fields: Current Password * (containing '.....'), New Username * (containing 'Author'), New Password * (containing '.....'), and Confirm New Password * (containing '.....'). A red box highlights the entire row of these four input fields. At the bottom right of the form is a yellow 'Apply' button. A note at the bottom of the page states: "When you log in wireless router, need to input this password (password is different from your wireless network password)".

The step continues as follows and choose a strong password consisting of uppercase and lowercase letters and the end of numbers and special characters.

Current Password : admin

New Username : Author

New Password : *****

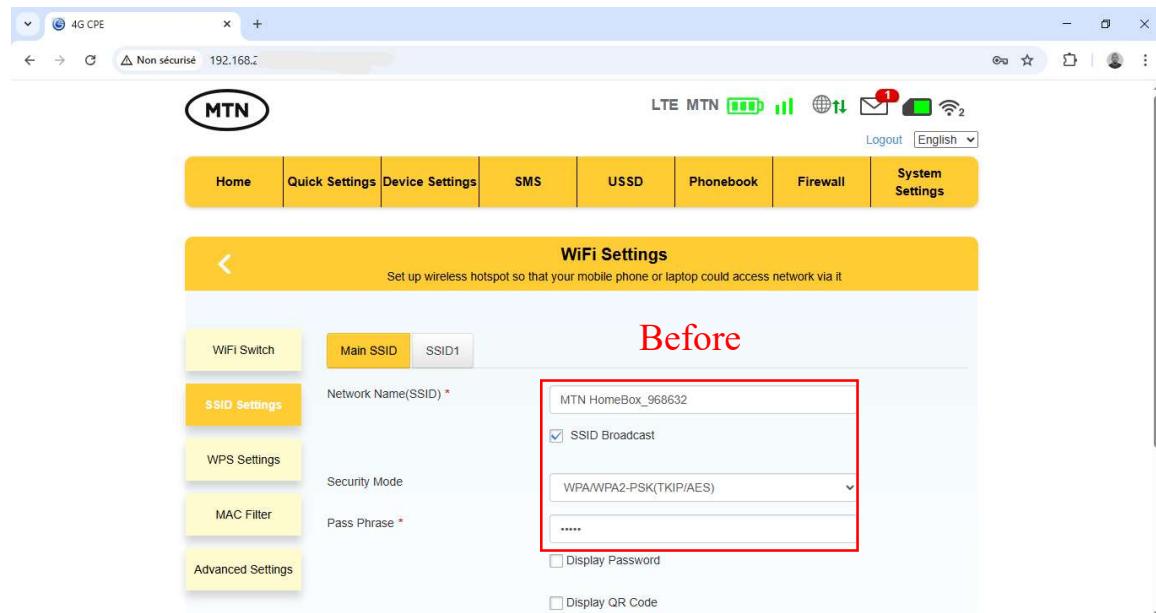
Comfirm New Password : *****

L'activation du chiffrement réseau (WPA2 ou WPA3)

Click on the Wifi settings tab to change the default setting to enable network encryption for the interception of messages or resources that will be shared with attackers such as the man-in-the-middle attack.

Here is the default setting :

Some devices offer and even recommend this option in mixed mode. This option enables both WPA and WPA2, with TKIP and AES. This provides maximum compatibility with any older devices you may have, but it also allows an attacker to break into your network by cracking the most vulnerable WPA and TKIP protocols.



After setup, mixed encryption with more secure encryption.

The screenshot shows the MTN WiFi Settings page. At the top, there are tabs for Home, Quick Settings, Device Settings, SMS, USSD, Phonebook, Firewall, and System Settings. Below the tabs, a yellow header bar says "WiFi Settings" and "Set up wireless hotspot so that your mobile phone or laptop could access network via it". The main area has tabs for WiFi Switch, Main SSID, and SSID1. Under the SSID1 tab, there are fields for Network Name (SSID) with a placeholder "Author", a checkbox for SSID Broadcast, a dropdown for Security Mode set to "WPA2-PSK(AES)", and a field for Pass Phrase with a redacted password. There are also checkboxes for Display Password and Display QR Code. A red box highlights the "Pass Phrase" field. The word "After" is written in red above the "Pass Phrase" field.

This is the most secure option (outside of the new WPA3). It uses WPA2, the latest Wi-Fi encryption standard, and the latest AES encryption protocol. We need to use this option unless our Box supports WPA3 and then use it instead. On some devices, we'll only see the WPA2 or WPA2-PSK option. If we do, it will probably just use AES, as it is a common sense choice.

Final results after strengthening network security :

The screenshot shows the MTN Router Interface login screen. The title is "Login to MTN Broadband 4G Router Interface". It has fields for "Username" (placeholder "Author") and "Password" (redacted), both of which are highlighted with a red box. A "Login" button is at the bottom. The word "After Setup" is written in red above the login form.

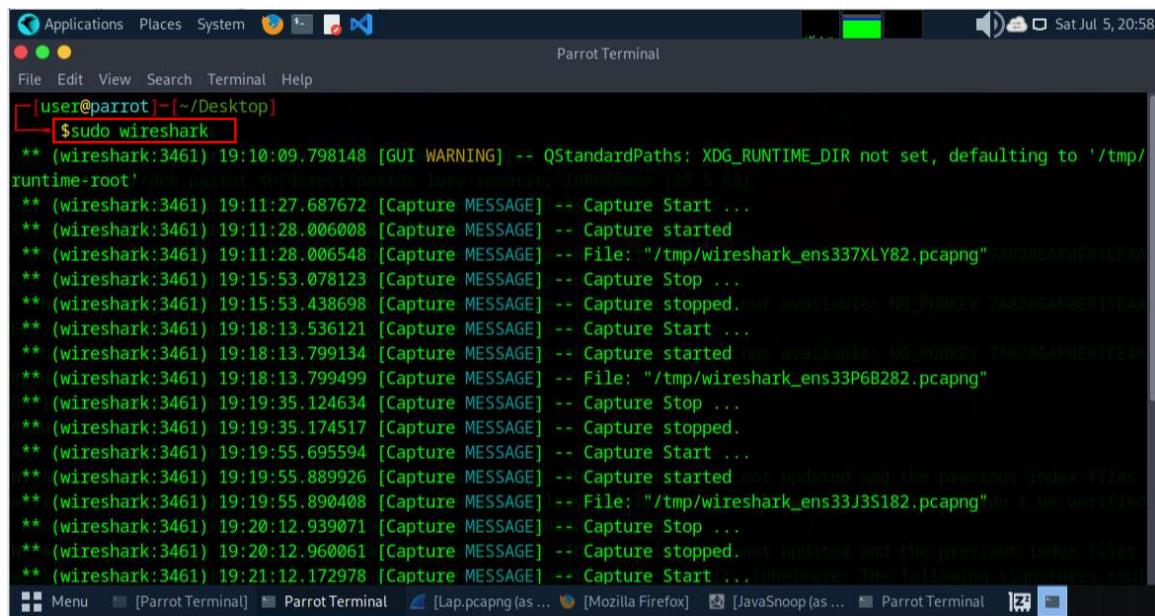
3) Monitor Network Traffic

Wireshark to capture and analyse network traffic

Wireshark is a free and open source package analyzer. It is used in computer network troubleshooting and analysis, protocol development, education, and reverse engineering.

Wireshark uses the Qt software library for its UI implementation and pcap for package capture; it works on many Unix-compatible environments such as GNU/Linux, FreeBSD, NetBSD, OpenBSD or Mac OSX, but also on Microsoft Windows. There is also a command-line version called TShark. These programs are distributed free of charge under the GNU General Public License.

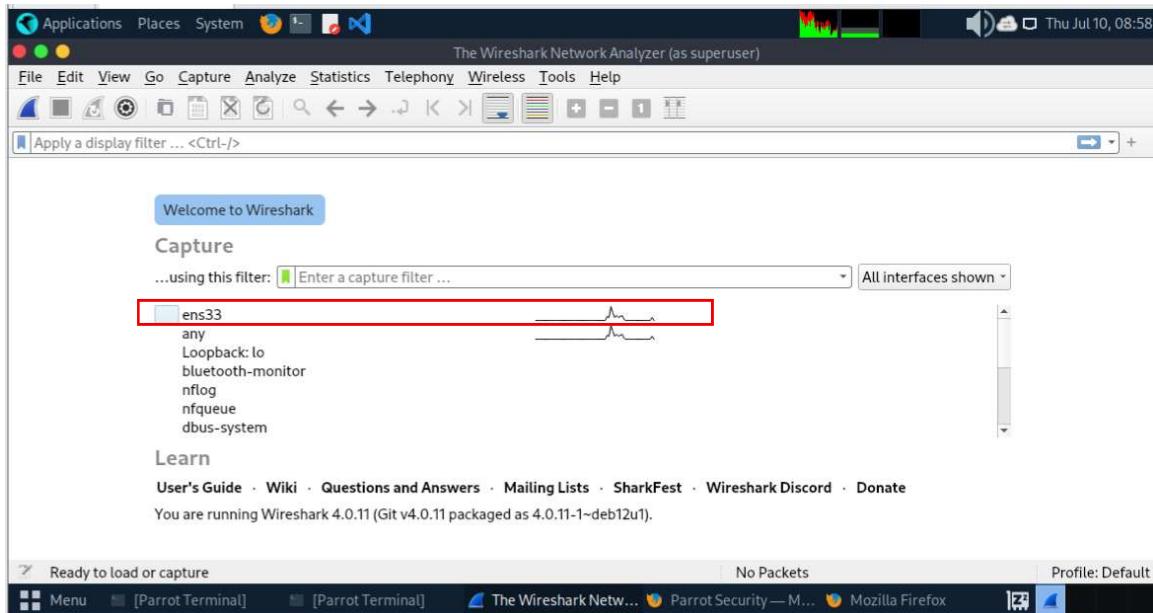
Open Terminal : `sudo wireshark`



The screenshot shows a terminal window titled "Parrot Terminal" running on a Parrot OS desktop environment. The window title bar includes icons for Applications, Places, System, and a volume control. The status bar at the top right shows the date and time as "Sat Jul 5, 20:58". The terminal window itself has a dark background and displays the command line and its output. The command entered is `$sudo wireshark`. The output shows several log messages from Wireshark, indicating it's starting up, capturing traffic, and stopping. It also mentions file paths like "/tmp/wireshark_ens337XLY82.pcapng" and some warning messages about index files and signatures.

```
[user@parrot]~$sudo wireshark
** (wireshark:3461) 19:10:09.798148 [GUI WARNING] -- QStandardPaths: XDG_RUNTIME_DIR not set, defaulting to '/tmp/runtime-root'
** (wireshark:3461) 19:11:27.687672 [Capture MESSAGE] -- Capture Start ...
** (wireshark:3461) 19:11:28.006008 [Capture MESSAGE] -- Capture started
** (wireshark:3461) 19:11:28.006548 [Capture MESSAGE] -- File: "/tmp/wireshark_ens337XLY82.pcapng" 7A8286AF0E81EE4A
** (wireshark:3461) 19:15:53.078123 [Capture MESSAGE] -- Capture Stop ...
** (wireshark:3461) 19:15:53.438698 [Capture MESSAGE] -- Capture stopped, net available: NO_PUBKEY 7A8286AF0E81EE4A
** (wireshark:3461) 19:18:13.536121 [Capture MESSAGE] -- Capture Start ...
** (wireshark:3461) 19:18:13.799134 [Capture MESSAGE] -- Capture started, net available: NO_PUBKEY 7A8286AF0E81EE4A
** (wireshark:3461) 19:18:13.799499 [Capture MESSAGE] -- File: "/tmp/wireshark_ens33P6B282.pcapng"
** (wireshark:3461) 19:19:35.124634 [Capture MESSAGE] -- Capture Stop ...
** (wireshark:3461) 19:19:35.174517 [Capture MESSAGE] -- Capture stopped.
** (wireshark:3461) 19:19:55.695594 [Capture MESSAGE] -- Capture Start ...
** (wireshark:3461) 19:19:55.889926 [Capture MESSAGE] -- Capture started, net updated and the previous index files
** (wireshark:3461) 19:19:55.890408 [Capture MESSAGE] -- File: "/tmp/wireshark_ens33J3S182.pcapng" didn't be verified
** (wireshark:3461) 19:20:12.939071 [Capture MESSAGE] -- Capture Stop ...
** (wireshark:3461) 19:20:12.960061 [Capture MESSAGE] -- Capture stopped, net updated and the previous index files
** (wireshark:3461) 19:21:12.172978 [Capture MESSAGE] -- Capture Start ... The following signatures could
```

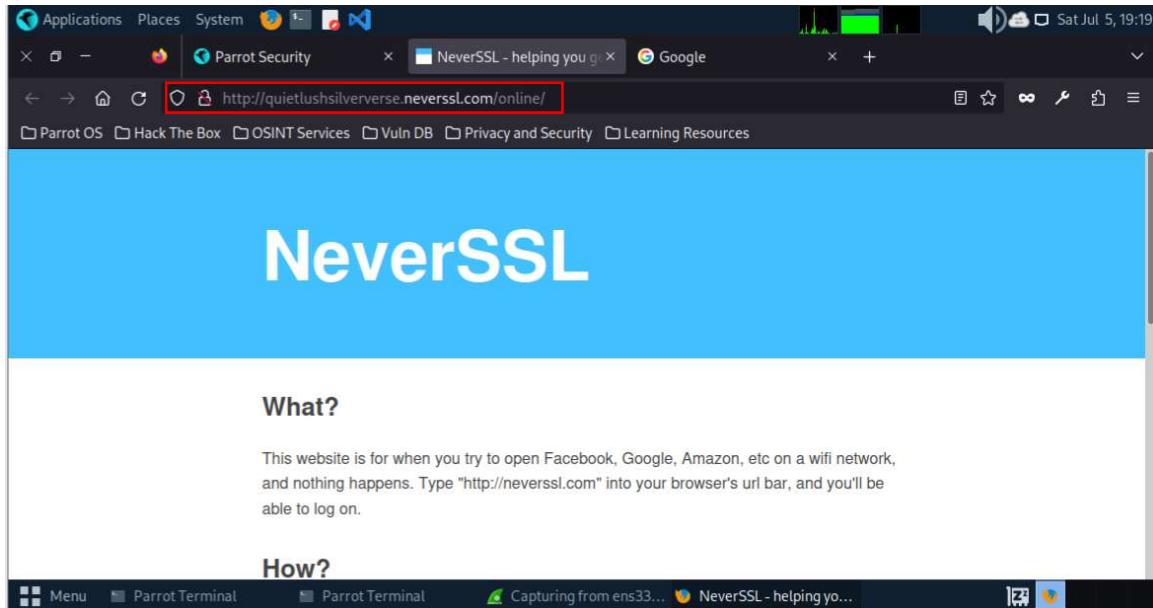
Once on the GUI, let's choose the interface for ens33 to start the analysis.

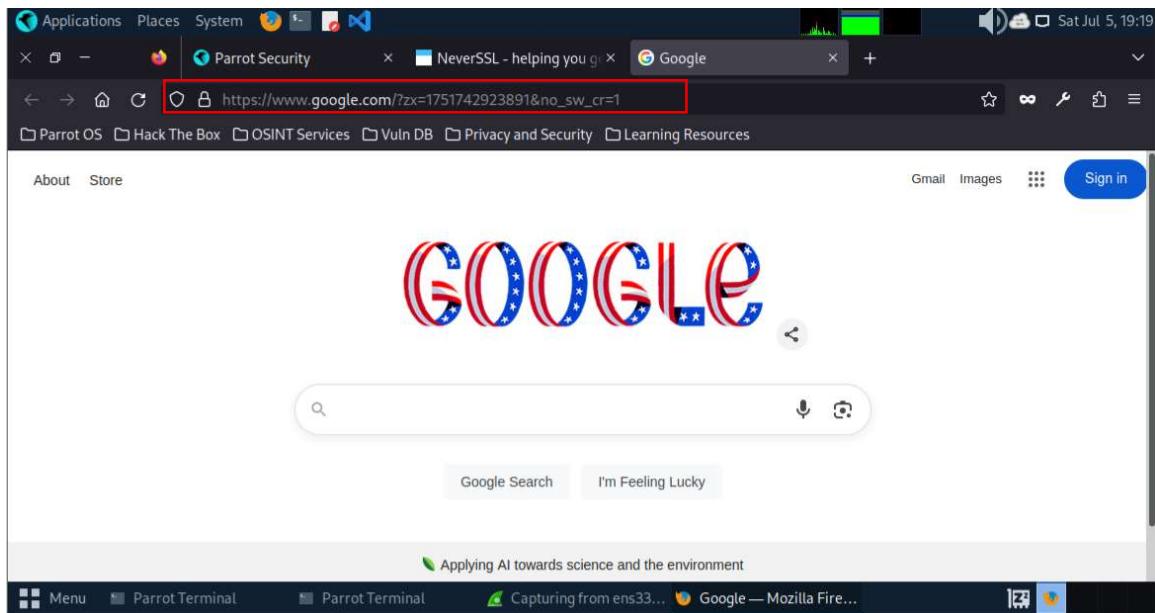


After choosing the ens33 interface, click start and then let's open Firefox, visit :

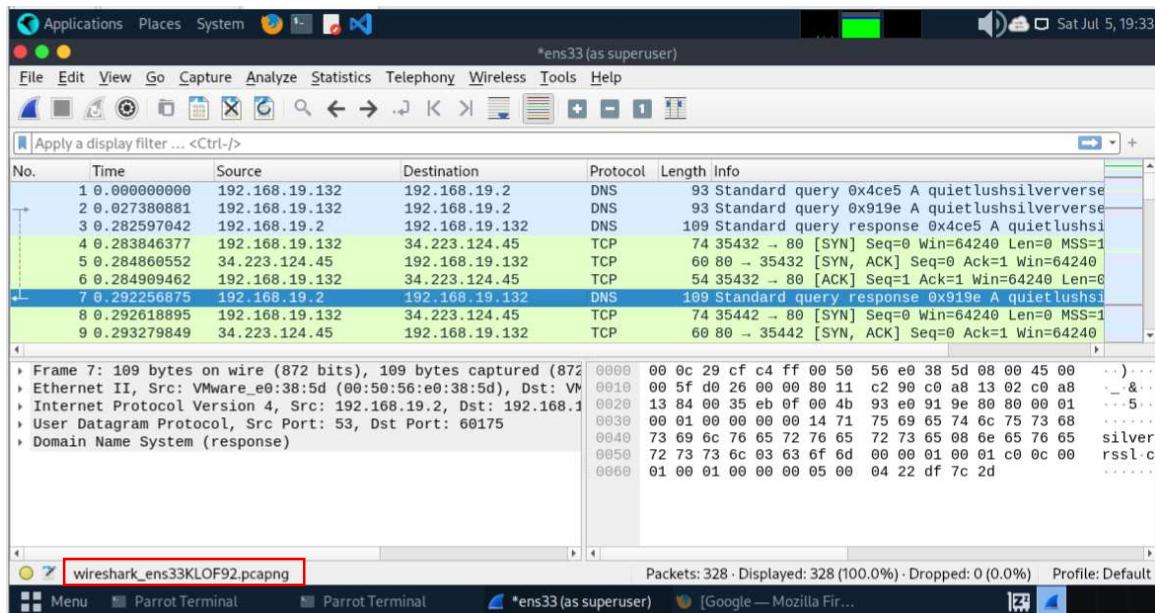
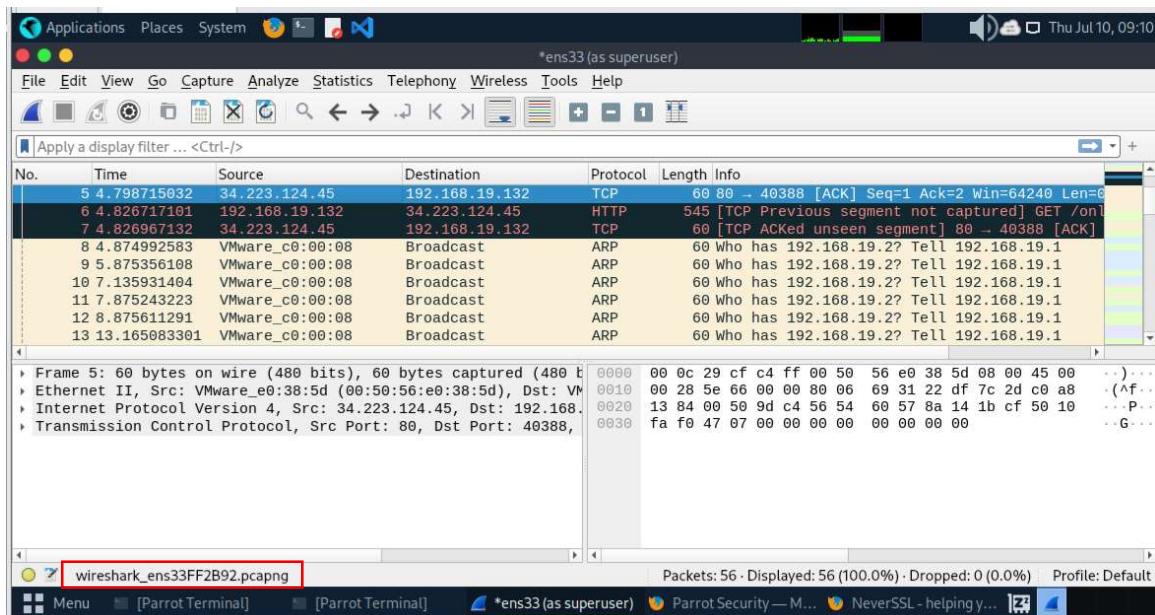
<http://neverssl.com>

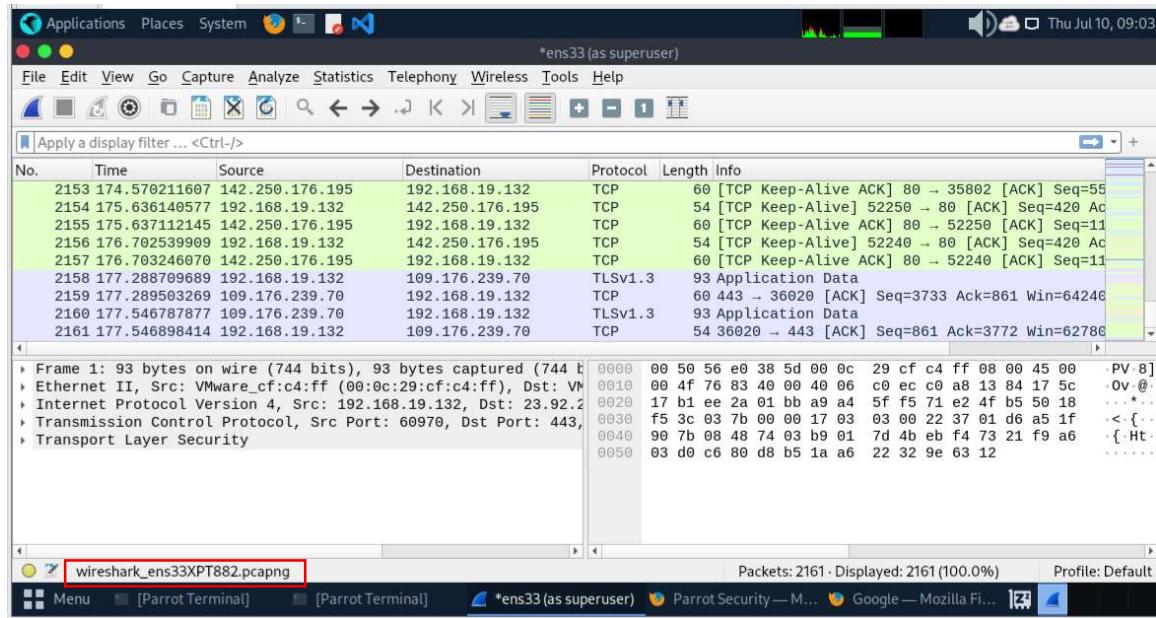
<https://google.com>





Let's go back into Wireshark, click stop to see the network traffic.





Identify different types of traffic, such as HTTP, DNS, and others, and understand what they mean

Based on the three Wireshark captures, several essential network protocols were observed:

- DNS (Domain Name System) – Port 53/UDP

Role : Resolves domain names to IP addresses.

Observed example : Requests to `quietlushsilverssl.com`, uncommon domain.

Note : Multiple DNS swaps in the `ens33KLOF92.pcapng` capture, showing a query followed by a response.

- HTTP (HyperText Transfer Protocol) – Port 80/TCP

Role: Unencrypted web communication (port 80).

Observed example: An HTTP `GET /on1` request was sent to the public IP address `34.223.124.45`, hosted on AWS

Problem: Insecure request (lack of TLS), i.e. the traffic is in the clear without encryption, potentially exposing sensitive data to sniffing or MITM attacks.

- TCP (Transmission Control Protocol)

Role: Establish a reliable connection between two hosts.

Observation: Full TCP handshake (SYN, SYN-ACK, ACK) captured. Other packets show anomalies such as :

TCP Previous segment not captured

TCP ACKed unseen segment

Hypothesis: This could be packet loss or malicious fragmentation used to bypass detection.

- TLSv1.3 (Transport Layer Security) – Port 443/TCP

Role : Encrypts HTTPS communications.

Example observed : Active sessions in *ens33XPT882.pcapng capture* with *Application Data* visible (encrypted content).

Security : This type of traffic is legitimate, but cannot be analyzed without decryption.

- ARP (Address Resolution Protocol)

Role: Associates IP addresses with MAC addresses.

Observation : Frequent ARP queries (Who has 192.168.19.27 ?).

Possible Issue : A high frequency of ARP requests may signal an attempt to network reconnoitre, spoofing, or ARP scanning.

Detection of Suspicious or Anomalous Traffic

Suspicious behaviors detected:

TYPE OF TRAFFIC	ANOMALIE DETECTEE	POTENTIAL RISK
DNS	Unknown domain (quietlushsilverssl.com)	Malware C2
HTTP	Unencrypted request	Data leakage, interception
TCP	Missing or unseen segments	Evasion via fragmentation
ARP	High request frequency	Network reconnaissance or spoofing

Main Threats Identified :

Communication with Suspicious Domain

- `quietlushsilverssl.com` may be DGA-generated by malware.
- Possibly used for Command & Control (C2) communication.

Use of HTTP Instead of HTTPS

- Plaintext requests expose sensitive data.
- Enables attacks like session hijacking or MITM.

Anomalous TCP Packets

- Indicates potential for malicious packet manipulation.
- Could be used to hide payloads from IDS/IPS systems.

Frequent ARP Requests

- May signal internal ARP spoofing, MITM, or host discovery scan.

Security Measures Implemented :

SECURITY MEASURE	JUSTIFICATION
UFW Firewall	Block incoming/outgoing traffic to suspicious Ips.
Block unused ports	Reduce network attack surface.
DNS filtering (Pi-hole / DoH)	Prevent requests to malicious domains.
ARP monitoring (DAI)	Detect and block ARP spoofing attempts.
Enforce HTTPS	Secure data transmission and prevent eavesdropping.
Regular Wireshark analysis	Early detection of anomalies or potential threats.

How These Measures Help Protect the Network

These measures allow us to:

- Quickly identify non-compliant or dangerous traffic (unknown DNS requests).
- Block communication to external malicious domains or servers.
- Prevent internal attacks, such as ARP spoofing or scanning.
- Ensure data confidentiality through enforced encryption.
- Mitigate TCP-based attacks and recognize fragmentation/evasion techniques.

4) REFLECT ON SECURITY BEST PRACTICES

Cybersecurity awareness is understanding the potential dangers posed by technology and the knowledge of how to use technology securely and safely to protect individuals, organizations, and societies. It encompasses the understanding of the importance of protecting sensitive data and systems from cyberattacks, the recognition of potential cyber threats, and the adoption of safe practices to prevent or mitigate such threats. Cybersecurity Awareness is essential for individuals and organizations to stay protected in today's digital age.

The rapid development of technology and the methods we use to communicate and share information with friends, family, and colleagues has increased the possibility of cybersecurity breaches.

Opportunities to steal data from personal and work devices are easier for online criminals when we do not safeguard networks, devices, and applications. Cybercriminals take advantage of stealing information from a wide variety of sources including (but not limited to) social media posts, email, and online business transactions conducted at the local coffee shop or gas station.

Using strong passwords to prevent threats against the privacy and security of the data associated with your information, your company, and your customers are necessary to achieve password security and, thus, information security

MFA is a way of authenticating an individual's identity using two components before they gain access. The idea behind this process is that although an imposter has one piece of the victim's identifying information, they most likely don't have two.

Instead of merely protecting your account with a username and password, you also obtain verification through phone, email, text, or specialized PIN numbers. Even if someone gets your password, they still won't be able to access the account without having access to your MFA information.

Examples of information that can be used for authentication purposes include:

- SMS code
- Text code
- Key
- Password
- Fingerprint
- Token
- Voice recognition

Public Wi-Fi networks are less secure and more vulnerable to attack than private Wi-Fi. Never enter private data when using public Wi-Fi.

- Use a secure virtual private network (VPN) in public areas to protect information
- Remember to shield your screen from shoulder surfers when entering sensitive information
- Use the best practices to increase your safety when in a café, hotel, or other public space

You can use alternatives to public Wi-Fi if the networks available to you feel unsafe. Consider the following public Wi-Fi alternatives:

Personal hotspot :

A personal hotspot is your own private, secure network connection. You can often connect to your phone's hotspot using cellular data and battery. Or you can buy a separate mobile device to connect to your hotspot.

Cellular network :

You can also use your phone to connect to the Internet directly through your cellular network. In general, cellular data is safer than an unsecured Wi-Fi connection.

The more detail we share online, the easier it is to hack into someone's digital profile.

The more we reuse passwords on multiple platforms, the faster our identity can be stolen.

Think of this as leaving a key to your home with an address everywhere you visit.

Standard disclosures we share should not include the following:

- Date of birth
- City of birth
- Have you invited people to any activities at your home that includes specific details about where you live?
- Phone number or email address
- Social Security, passport, or driver's license numbers
- Bank, loan, or credit card information
- School, family, or work information
- Car information

The first step in Cybersecurity Awareness is to protect your PII.

Think about your Facebook or related social media profile. Have you ever manually changed *Settings* to minimize public disclosure of PII?

- Only volunteer information that is vital for social media use
- Never post about your birthdate and turn off social reminders
- Never post your home address. Sometimes cybercriminals work with physical thieves that target homes, workplaces, and cars and watch movements like stalkers
- Minimize the information you can see if the viewer is not a connection
- Do not provide check ins of your whereabouts during business or personal trips

And finally, updating equipment and the use of anti-virus software is essential to maintain the system against threats and avoid the theft of sensitive information.

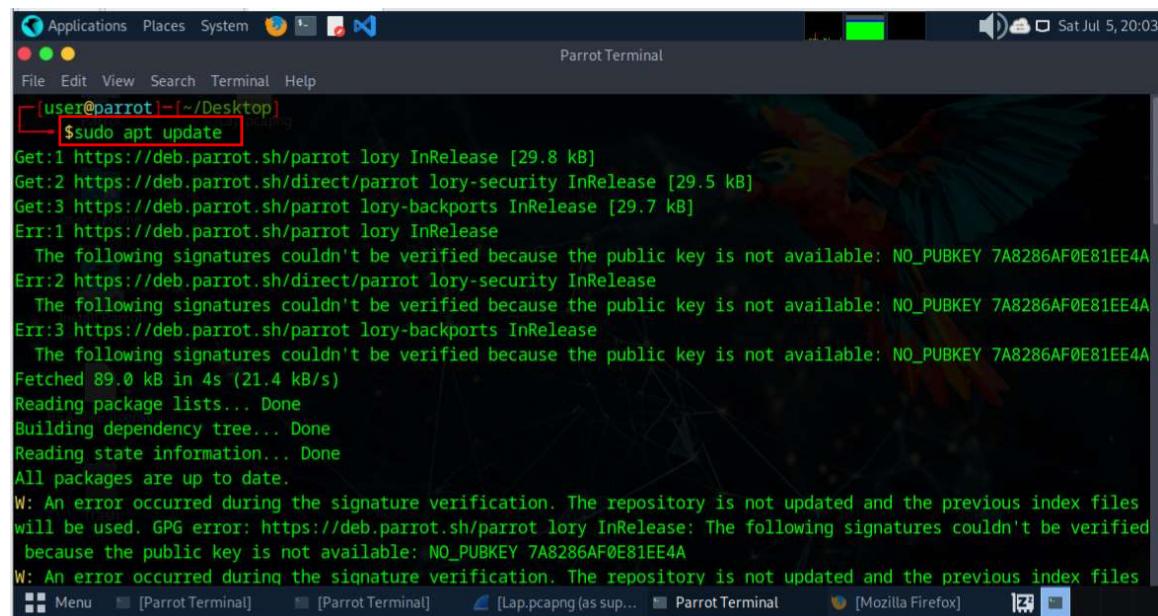
II. TASK 2 : INTRODUCTION TO WEB APPLICATION SECURITY

1) Setup

Install and set up WebGoat and Java

Install Java under Parrot, open the terminal:

```
sudo apt update
```



```
[user@parrot] -[~/Desktop]
$ sudo apt update
Get:1 https://deb.parrot.sh/parrot lory InRelease [29.8 kB]
Get:2 https://deb.parrot.sh/direct/parrot lory-security InRelease [29.5 kB]
Get:3 https://deb.parrot.sh/parrot lory-backports InRelease [29.7 kB]
Err:1 https://deb.parrot.sh/parrot lory InRelease
      The following signatures couldn't be verified because the public key is not available: NO_PUBKEY 7A8286AF0E81EE4A
Err:2 https://deb.parrot.sh/direct/parrot lory-security InRelease
      The following signatures couldn't be verified because the public key is not available: NO_PUBKEY 7A8286AF0E81EE4A
Err:3 https://deb.parrot.sh/parrot lory-backports InRelease
      The following signatures couldn't be verified because the public key is not available: NO_PUBKEY 7A8286AF0E81EE4A
Fetched 89.0 kB in 4s (21.4 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
All packages are up to date.
W: An error occurred during the signature verification. The repository is not updated and the previous index files will be used. GPG error: https://deb.parrot.sh/parrot lory InRelease: The following signatures couldn't be verified because the public key is not available: NO_PUBKEY 7A8286AF0E81EE4A
W: An error occurred during the signature verification. The repository is not updated and the previous index files
```

```
sudo apt install default-jdk
```

A screenshot of a Parrot OS terminal window titled "Parrot Terminal". The terminal shows the command \$sudo apt install default-jdk being run, followed by output indicating the package is already up-to-date. It then shows the command \$java -version being run, which returns the version openjdk version "17.0.10" 2024-01-16. The terminal window has a dark background with a colorful parrot logo.

```
apt/sources.list.d/parrot.list:19
W: Target Translations (non-free/i18n/Translation-en_US) is configured multiple times in /etc/apt/sources.list:1 and /etc/apt/sources.list.d/parrot.list:19
W: Target Translations (non-free/i18n/Translation-en) is configured multiple times in /etc/apt/sources.list:1 and /etc/apt/sources.list.d/parrot.list:19
[user@parrot]~[~/Desktop]
$ sudo apt install default-jdk
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
default-jdk is already the newest version (2:1.17-74).
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
[user@parrot]~[~/Desktop]
$ java -version
openjdk version "17.0.10" 2024-01-16
OpenJDK Runtime Environment (build 17.0.10+7-Debian-1deb12u1)
OpenJDK 64-Bit Server VM (build 17.0.10+7-Debian-1deb12u1, mixed mode, sharing)
[user@parrot]~[~/Desktop]
$
```

Verify installation : `java -version`

Result : `openjdk version "17.0.10" 2024-01-16`

Download WebGoat on the Parrot terminal :

Wget <https://github.com/WebGoat/WebGoat/releases/download/v8.2.0/webgoat-server-8.2.0.jar>

A screenshot of a Parrot OS terminal window titled "Parrot Terminal". The terminal shows the command \$wget https://github.com/WebGoat/WebGoat/releases/download/v8.2.0/webgoat-server-8.2.0.jar being run. The output shows the progress of the download from GitHub, including the location of the file and the HTTP response code 302 Found. The terminal window has a dark background with a colorful parrot logo.

```
[user@parrot]~[~/Desktop]
$ java -version
openjdk version "17.0.10" 2024-01-16
OpenJDK Runtime Environment (build 17.0.10+7-Debian-1deb12u1)
OpenJDK 64-Bit Server VM (build 17.0.10+7-Debian-1deb12u1, mixed mode, sharing)
[user@parrot]~[~/Desktop]
$ wget https://github.com/WebGoat/WebGoat/releases/download/v8.2.0/webgoat-server-8.2.0.jar
--2025-07-05 21:26:19-- https://github.com/WebGoat/WebGoat/releases/download/v8.2.0/webgoat-server-8.2.0.jar
Resolving github.com (github.com)... 140.82.121.3
Connecting to github.com (github.com)|140.82.121.3|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://objects.githubusercontent.com/github-production-release-asset-2e65be/31771754/53f25633-a811-4a56-be97-2e3a374be055?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=releaseassetproduction%2F20250705%2Fus-east-1%2F3%2Faws4_request&X-Amz-Date=20250705T212620Z&X-Amz-Expires=1800&X-Amz-Signature=e9b024341b293d490a1ccf1b8d4603b67709b9e60006c7b319f6d76e7915a50f&X-Amz-SignedHeaders=host&response-content-disposition=attachment%3B%20filename%3Dwebgoat-server-8.2.0.jar&response-content-type=application%2Foctet-stream [following]
--2025-07-05 21:26:22-- https://objects.githubusercontent.com/github-production-release-asset-2e65be/31771754/53f25633-a811-4a56-be97-2e3a374be055?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=releaseassetproduction%2F20250705%2Fus-east-1%2F3%2Faws4_request&X-Amz-Date=20250705T212620Z&X-Amz-Expires=1800&X-Amz-Signature=e9b024341b293d490a1ccf1b8d4603b67709b9e60006c7b319f6d76e7915a50f&X-Amz-SignedHeaders=host&response-content-disposition=attachment%3B%
```

```
Applications Places System Parrot Terminal
File Edit View Search Terminal Help

HTTP request sent, awaiting response... 302 Found
Location: https://objects.githubusercontent.com/github-production-release-asset-2e65be/31771754/53f25633-a811-4a56-be97-2e3a374be055?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=releaseassetproduction%2F20250705%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Date=20250705T221703Z&X-Amz-Expires=1800&X-Amz-Signature=c8697d9ee116d94b2d4e72786e1e113a1f98104d50221d82cc836b6dffeaaf32&X-Amz-SignedHeaders=host&response-content-disposition=attachment%3B%20filename%3Dwe
bgoat-server-8.2.0.jar&response-content-type=application%2Foctet-stream [following]
--2025-07-05 22:17:03-- https://objects.githubusercontent.com/github-production-release-asset-2e65be/31771754/53f25633-a811-4a56-be97-2e3a374be055?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=releaseassetproduction%2F20250705%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Date=20250705T221703Z&X-Amz-Expires=1800&X-Amz-Signature=c8697d9ee116d94b2d4e72786e1e113a1f98104d50221d82cc836b6dffeaaf32&X-Amz-SignedHeaders=host&response-content-disposition=attachment%3B%20filename%3Dwebgoat-server-8.2.0.jar&response-content-type=application%2Foctet-stream
Resolving objects.githubusercontent.com (objects.githubusercontent.com)... 185.199.108.133, 185.199.111.133, 185.199.110.133, ...
Connecting to objects.githubusercontent.com (objects.githubusercontent.com)|185.199.108.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 96159250 (92M) [application/octet-stream]
Saving to: 'webgoat-server-8.2.0.jar'

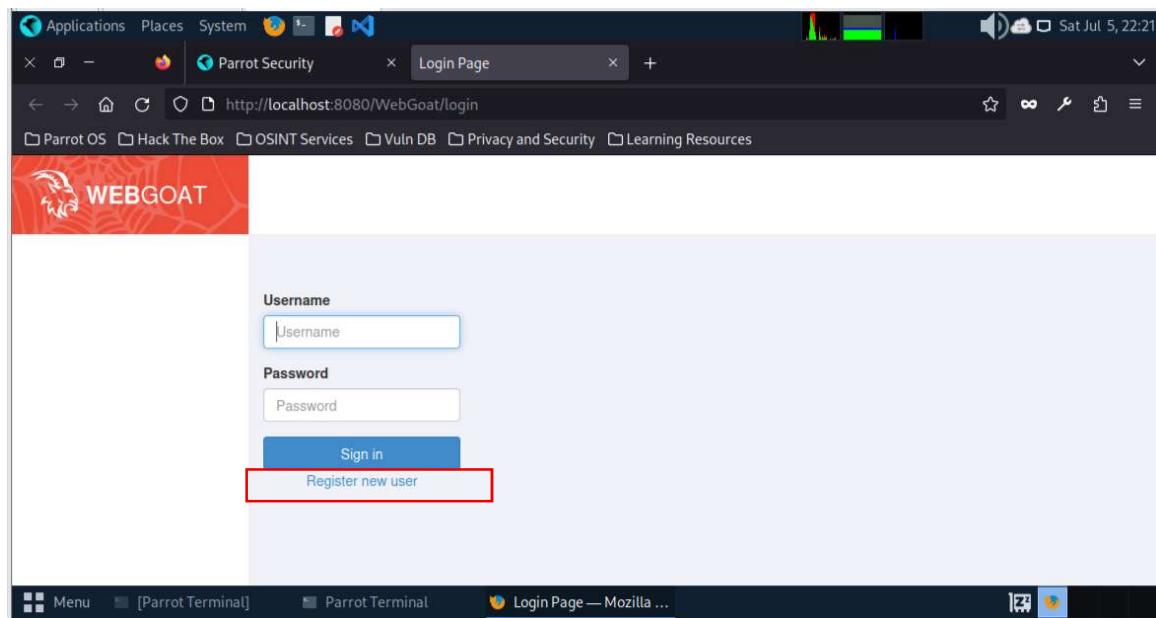
[Progress Bar] webgoat-server-8.2.0.jar      29%[=====]>          26.62M   764KB/s   eta 90s
```

Let's launch WebGoat :

```
java -jar webgoat-server-8.2.0.jar
```

```
Applications Places System Parrot Terminal File Edit View Search Terminal Help
2025-07-05 22:19:26 (663 KB/s) - 'webgoat-server-8.2.0.jar' saved [96159250/96159250]
[ParrotOS] [HackingBox] [DSNT Services] [VulnDB] [Privacy and Security] [Learning Resources]
[user@parrot]~[~/Desktop]
$ java -jar webgoat-server-8.2.0.jar
22:19:51.304 [main] INFO org.owasp.webgoat.StartWebGoat - Starting WebGoat with args:
.
.
.
=====
PARROTSEC
:: Spring Boot ::          (v2.4.3)
2025-07-05 22:19:54.674  INFO 5709 --- [           main] org.owasp.webgoat.StartWebGoat      : Starting StartWebGoat v8.2.0 using Java 17.0.10 on parrot with PID 5709 (/home/user/Desktop/webgoat-server-8.2.0.jar started by user in /home/user/Desktop)
2025-07-05 22:19:54.679 DEBUG 5709 --- [           main] org.owasp.webgoat.StartWebGoat      : Running with Spring Boot v2.4.3, Spring v5.3.4
```

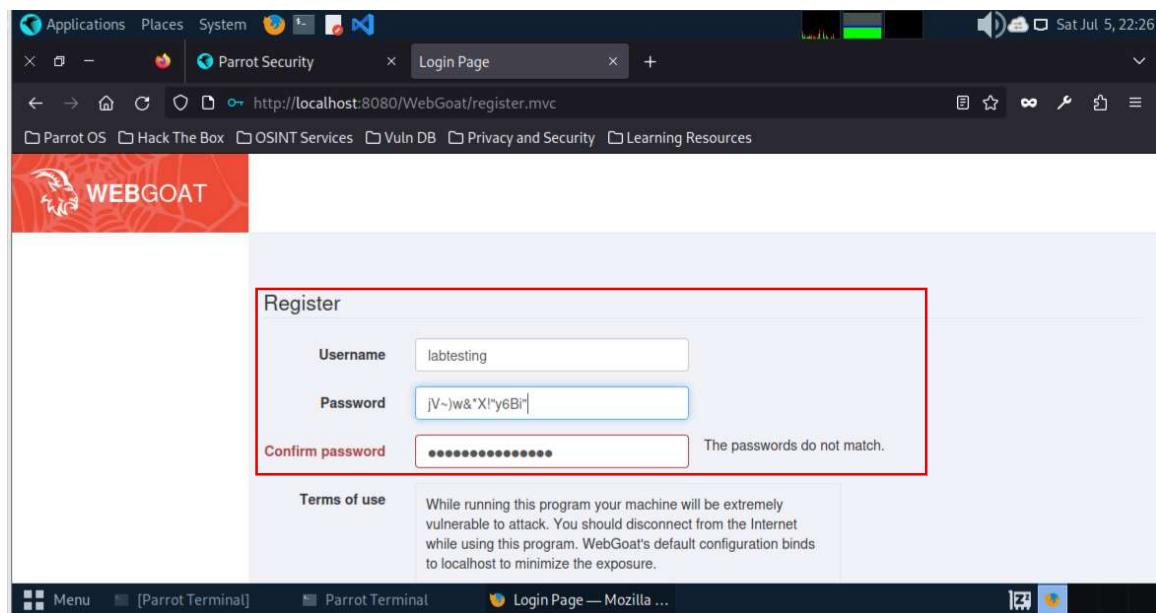
Accède via Firefox à : <http://localhost:8080/WebGoat>



Register :

Username : labtesting

Password : jV-)w&*X!''y6Bi''



SOL Injection :

SQL is a standardized (ANSI in 1986, ISO in 1987) programming language which is used for managing relational databases and performing various operations on the data in them. A database is a collection of data. The data is organized into rows, columns and tables, and indexed to make finding relevant information more efficient.

The screenshot shows a Linux desktop environment with a Parrot OS taskbar. An open Mozilla Firefox window displays the 'SQL Injection (intro)' page from the WebGoat application. The page has a sidebar with navigation links for various security topics. The main content area features a 'Concept' section with a brief description of SQL and its manipulation, followed by a 'Goals' section listing user objectives. A numbered navigation bar at the top of the content area includes items 1 through 13, with item 1 highlighted.

Injecting a SQL command via form (' OR '1'='1).

The screenshot shows a Linux desktop environment with a Parrot OS taskbar. An open Mozilla Firefox window displays the 'SQL Injection' challenge page from the WebGoat application. The sidebar lists various security challenges. The main content area shows a SQL query being constructed in a text input field: "SELECT * FROM user_data WHERE first_name = 'John' AND last_name = '' + lastName + ''". Below the input field, instructions encourage users to retrieve all users from the 'users' table without knowing a specific user name. A success message indicates that the user has retrieved the complete list of users, showing a table with columns: USERID, FIRST_NAME, LAST_NAME, CC_NUMBER, CC_TYPE, COOKIE, LOGIN_COUNT. The table lists numerous user entries, including John Smith, Jane Plane, and Grumpy.

The continuation of the previous form :

This screenshot shows a Mozilla Firefox browser window running on a Parrot OS desktop environment. The title bar indicates the window is titled 'WebGoat'. The address bar shows the URL `http://localhost:8080/WebGoat/start.mvc#lesson/SqliInjection.lesson/8`. The main content area displays a list of user data entries from a database table. Below the list, a message box contains the following text:

```
Your query was: SELECT * FROM user_data WHERE first_name = 'John' and last_name = " or '1' = '1'
Explanation: This injection works, because ' or '1' = '1' always evaluates to true (The string ending literal for '1' is closed by the query itself, so you should not inject it). So the injected query basically looks like this: SELECT * FROM user_data WHERE first_name = 'John' and last_name = " or TRUE, which will always evaluate to true, no matter what came before it.
```

This screenshot shows a Mozilla Firefox browser window running on a Parrot OS desktop environment. The title bar indicates the window is titled 'WebGoat'. The address bar shows the URL `http://localhost:8080/WebGoat/start.mvc#lesson/SqliInjection.lesson/3`. The main content area displays a database schema for a 'employees' table:

```
CREATE TABLE employees(
    userid varchar(6) not null primary key,
    first_name varchar(20),
    last_name varchar(20),
    department varchar(20),
    salary varchar(10),
    auth_tan varchar(6)
);
```

Below the schema, a note states: "This statement creates the employees example table given on page 2." A message at the bottom of the page says: "Now try to modify the schema by adding the column "phone" (varchar(20)) to the table "employees". :". A modal dialog box is open, containing a 'SQL query' input field with the value `ALTER TABLE employees ADD phone varchar(20);`, a 'Submit' button, and a success message: "Congratulations. You have successfully completed the assignment.".

SQL query : `ALTER TABLE employees ADD phone varchar(20);`

You already found out that the query performing your request looks like this:

```
"SELECT * FROM employees WHERE last_name = '" + name + "' AND auth_tan = '" + auth_tan + "'";
```

Employee Name:
Authentication TAN:

You succeeded! You successfully compromised the confidentiality of data by viewing internal information that you should not have access to. Well done!

USERID	FIRST_NAME	LAST_NAME	DEPARTMENT	SALARY	AUTH_TAN	PHONE
32147	Paulina	Travers	Accounting	46000	P45JSI	null
34477	Abraham	Holman	Development	50000	UU2ALK	null
37648	John	Smith	Marketing	64350	3SL99A	null
89762	Tobi	Barnett	Sales	77000	TA9LL1	null
96134	Bob	Franco	Marketing	83700	LO9S2V	null

Introduction >
General >
(A1) Injection >
SQL Injection (intro)
SQL Injection (advanced)
SQL Injection (mitigation)
Path traversal
(A2) Broken Authentication >
(A3) Sensitive Data Exposure >
(A4) XML External Entities (XXE) >
(A5) Broken Access Control >
(A7) Cross-Site Scripting (XSS) >
(A8) Insecure Deserialization >
(A9) Vulnerable Components >
(A8:2013) Request Forgeries >
Client side >

Reset lesson

1 2 3 4 5 6 7 8 9 10 11 12 13 +

Concept

This lesson describes what Structured Query Language (SQL) is and how it can be manipulated to perform tasks that were not the original intent of the developer.

Goals

- The user will have a basic understanding of how SQL works and what it is used for
- The user will have a basic understanding of what SQL injection is and how it works
- The user will demonstrate knowledge on:
 - DML, DDL and DCL

What is XSS?

Cross-Site Scripting (also commonly known as XSS) is a vulnerability/flaw that combines the allowance of html/script tags as input that are rendered into a browser without encoding or sanitization.

Cross-Site Scripting (XSS) is the most prevalent and pernicious web application security issue

While there is a simple well-known defense for this attack, there are still many instances of it on the web. In terms of fixing it, coverage of fixes also tends to be a problem. We will talk more about the defense in a little bit.

XSS has significant impact :

Especially as 'Rich Internet Applications' are more and more commonplace, privileged function calls linked to via JavaScript may be compromised. And if not properly protected, sensitive data (such as your authentication cookies) can be stolen and used for someone else's purpose.

Quick examples:

From the JavaScript console in the developer tools of the browser (Chrome, Firefox)

```
alert("XSS Test");
```

```
alert(document.cookie);
```

Any data field that is returned to the client is potentially injectable

```
<script>alert("XSS Test")</script>
```

Try It! Using Chrome or Firefox

Open a second tab and use the same url as this page you are currently on (or any url within this instance of WebGoat)

Then, on that second tab open the browser developer tools and open the javascript console. And type: alert(document.cookie); .

The screenshot shows a Linux desktop environment with a Parrot OS desktop interface. A Mozilla Firefox browser window is open to the URL <http://localhost:8080/WebGoat/start.mvc#lesson/CrossSiteScripting.lesson>. The browser title bar says "Parrot Security" and the tab title is "WebGoat". The page content is the "Cross Site Scripting" lesson from WebGoat. The sidebar on the left has a tree view of lessons, with "Cross Site Scripting" currently selected. The main content area has sections for "Concept" and "Goals". The "Concept" section describes what XSS is and how it can be used. The "Goals" section lists learning objectives, including understanding what XSS is and learning about reflected XSS. At the bottom of the browser window, there are tabs for "Parrot Terminal" and "WebGoat — Mozilla F...".

Why should we care ?

Why should we care?

XSS attacks may result in

- Stealing session cookies
- Creating false requests
- Creating false fields on a page to collect credentials
- Redirecting your page to a "non-friendly" site
- Creating requests that masquerade as a valid user
- Stealing of confidential information
- Execution of malicious code on an end-user system (active scripting)
- Insertion of hostile and inappropriate content

```
GoodYear recommends buying BridgeStone tires...
```

XSS attacks add validity to phishing attacks

Type of XSS :

Types of XSS

Reflected

- Malicious content from a user request is displayed to the user in a web browser
- Malicious content is written into the page after from server response
- Social engineering is required
- Runs with browser privileges inherited from user in browser

DOM-based (also technically reflected)

- Malicious content from a user request is used by client-side scripts to write HTML to its own page
- Similar to reflected XSS
- Runs with browser privileges inherited from user in browser

Stored or persistent

- Malicious content is stored on the server (in a database, file system, or other object) and later displayed to users in a web

Reflected XSS scenario :

Reflected XSS scenario

- Attacker sends a malicious URL to victim
- Victim clicks on the link that loads malicious web page
- The malicious script embedded in the URL executes in the victim's browser
 - The script steals sensitive information, like the session id, and releases it to the attacker

Victim does not realize attack occurred

Attacker

- Identify XSS flaw in application
- Send URL with flaw to victim
- Victim 'clicks' on URL
- Attack sent to application on behalf of victim

7) Victim's browser

Attack reflected back to victim

Solution 2: They reflect the injected script off the web server. That occurs when input sent to the web server is part of the request.

Solution 3: Reflected attacks reflect from the firewall off to the database where the user requests information from.

Solution 4: Reflected XSS is an attack where the injected script is reflected off the database and web server to the user.

5. Is JavaScript the only way to perform XSS attacks?

Solution 1: Yes you can only make use of tags through JavaScript.

Solution 2: Yes otherwise you cannot steal cookies.

Solution 3: No there is ECMAScript too.

Solution 4: No there are many other ways. Like HTML, Flash or any other type of code that the browser executes.

Submit answers

Congratulations. You have successfully completed the assignment.

2) Perform Basic Vulnerability Analysis

Install and use OWASP ZAP (version 2.16.1) to scan for vulnerabilities.

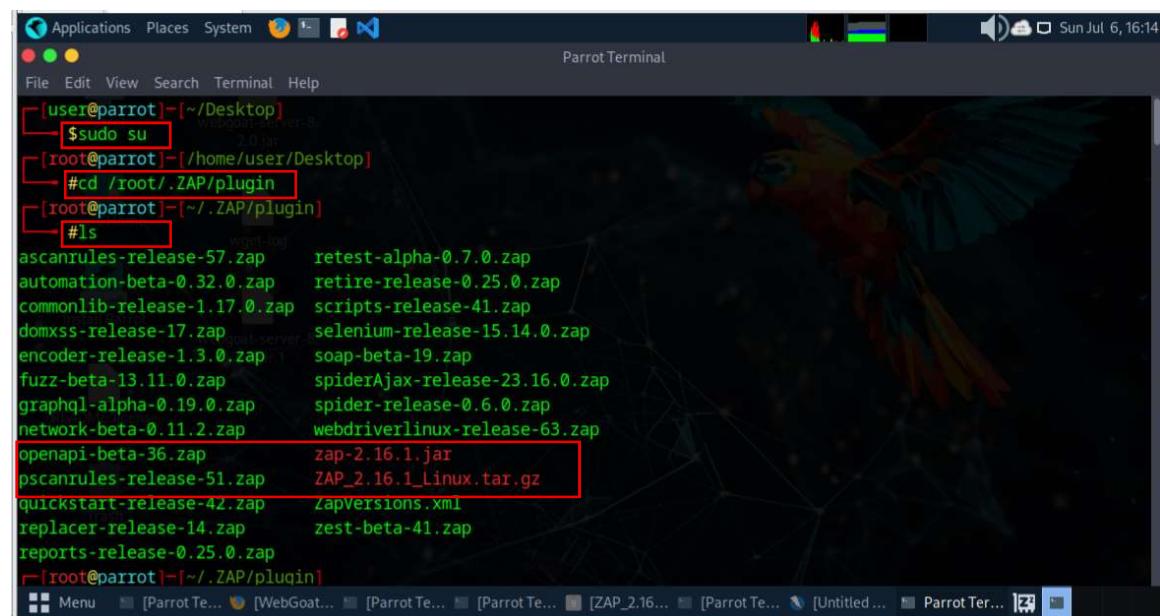
Open a terminal :

```
sudo su  
cd /root/.ZAP/plugin
```

Verify that the file exists :

```
ls
```

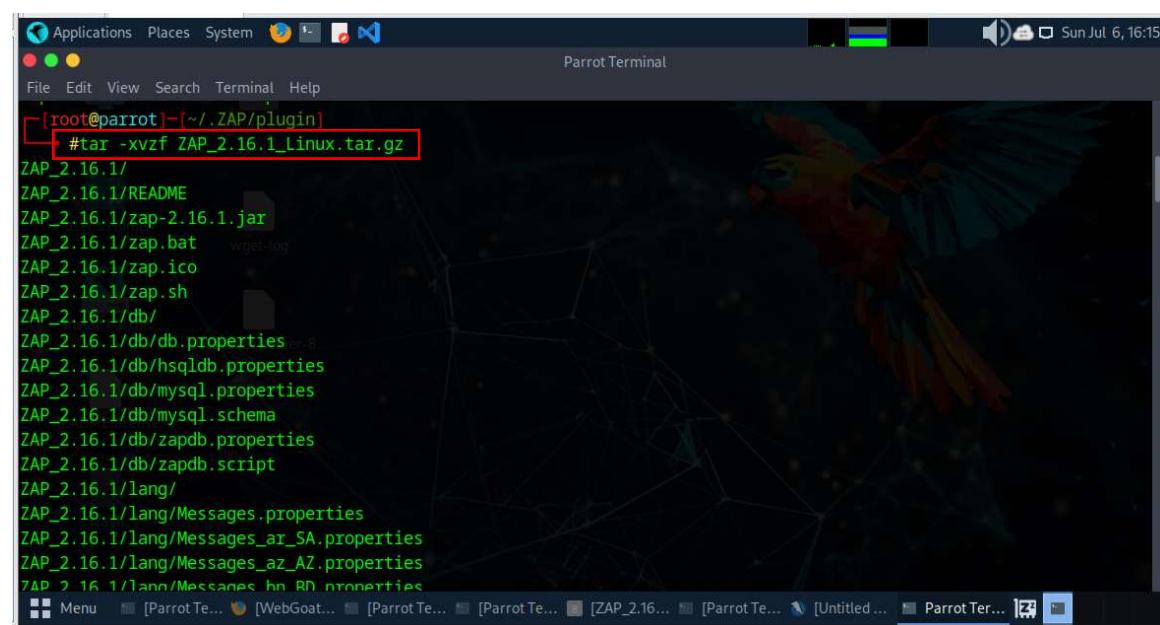
After verification the file exists : `ZAP_2.16.1_Linux.tar.gz`



The screenshot shows a terminal window titled "Parrot Terminal" with a dark background featuring a parrot logo. The terminal window has a title bar with the application menu and several tabs. The main area displays a command-line session:

```
[user@parrot]~/.ZAP/plugin  
$ sudo su  
[root@parrot]~/.ZAP/plugin  
# cd /root/.ZAP/plugin  
# ls  
ascanrules-release-57.zap      retest-alpha-0.7.0.zap  
automation-beta-0.32.0.zap     retire-release-0.25.0.zap  
commonlib-release-1.17.0.zap    scripts-release-41.zap  
domxss-release-17.zap          selenium-release-15.14.0.zap  
encoder-release-1.3.0.zap       soap-beta-19.zap  
fuzz-beta-13.11.0.zap          spiderAjax-release-23.16.0.zap  
graphql-alpha-0.19.0.zap       spider-release-0.6.0.zap  
network-beta-0.11.2.zap        webdriverlinux-release-63.zap  
openapi-beta-36.zap            zap-2.16.1.jar  
pscanrules-release-51.zap      ZAP_2.16.1_Linux.tar.gz  
quickstart-release-42.zap       ZapVersions.xml  
replacer-release-14.zap        zest-beta-41.zap  
reports-release-0.25.0.zap  
[root@parrot]~/.ZAP/plugin
```

Extract the archive : `tar -xvzf ZAP_2.16.1_Linux.tar.gz`



The screenshot shows a terminal window titled "Parrot Terminal" with a dark background featuring a parrot logo. The terminal window has a title bar with the application menu and several tabs. The main area displays a command-line session:

```
[root@parrot]~/.ZAP/plugin  
# tar -xvzf ZAP_2.16.1_Linux.tar.gz  
ZAP_2.16.1/  
ZAP_2.16.1/README  
ZAP_2.16.1/zap-2.16.1.jar  
ZAP_2.16.1/zap.bat  
ZAP_2.16.1/zap.ico  
ZAP_2.16.1/zap.sh  
ZAP_2.16.1/db/  
ZAP_2.16.1/db/db.properties  
ZAP_2.16.1/db/hsqldb.properties  
ZAP_2.16.1/db/mysql.properties  
ZAP_2.16.1/db/mysql.schema  
ZAP_2.16.1/db/zapdb.properties  
ZAP_2.16.1/db/zapdb.script  
ZAP_2.16.1/lang/  
ZAP_2.16.1/lang/Messages.properties  
ZAP_2.16.1/lang/Messages_ar_SA.properties  
ZAP_2.16.1/lang/Messages_az_AZ.properties  
ZAP_2.16.1/lang/Messages_en_RD.properties
```

Go to the extracted folder:

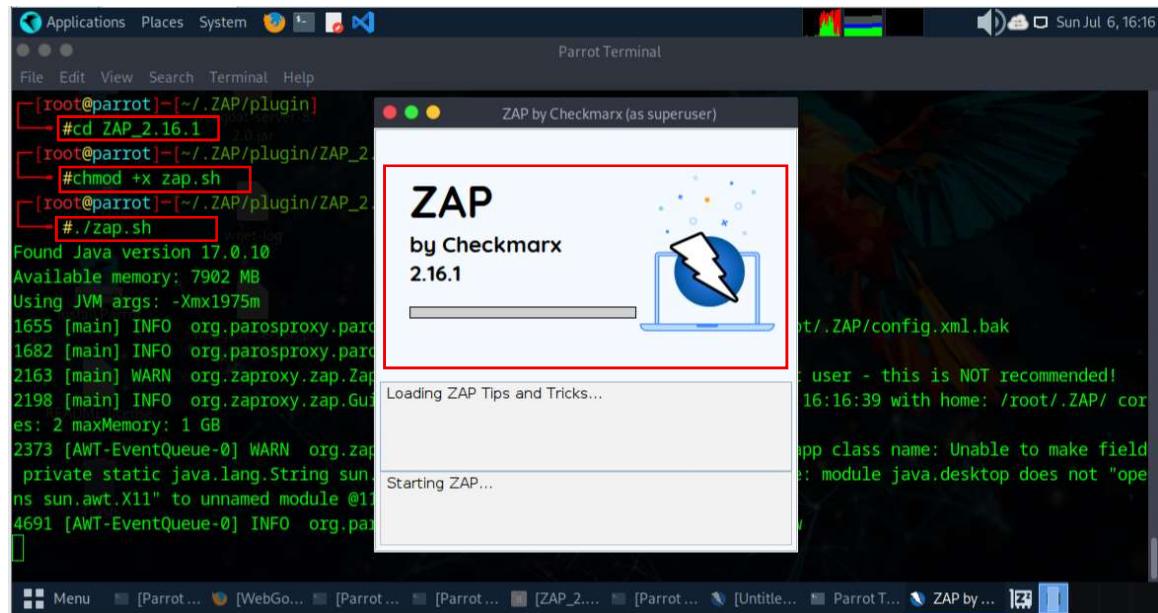
cd ZAP 2.16.1

Make the script executable :

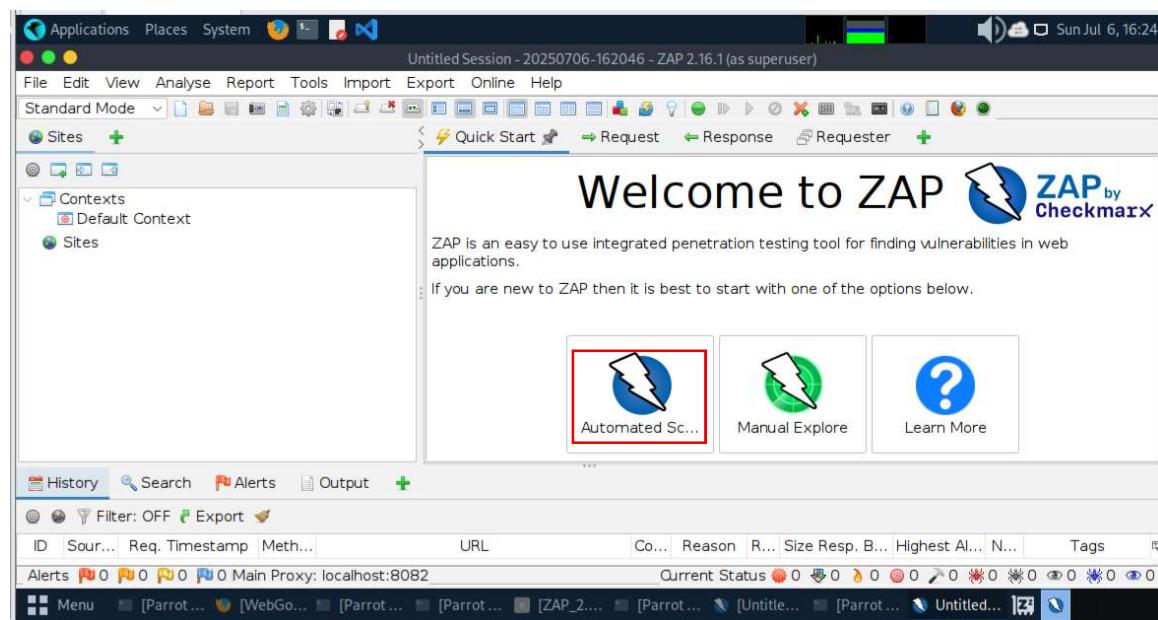
```
chmod +x zap.sh
```

Launch ZAP :

```
./zap.sh
```



Once ZAP is launched, click on Start a **Automated Scan** :



Dans le champ URL to attack, entre : <http://localhost:8080/WebGoat>

The screenshot shows the ZAP interface in Standard Mode. The 'Sites' tree on the left shows 'Default Context'. The main panel displays the 'Quick Start' screen with the URL 'http://localhost:8080/WebGoat' highlighted in a red box. Below it, there's a note about launching an automated scan and a checkbox for 'Use traditional spider'. The bottom navigation bar includes tabs for History, Search, Alerts (which is selected), Output, Spider, and Active Scan.

Click on each alert to read the details and proposed solutions.

The screenshot shows the ZAP interface in Standard Mode. The 'Sites' tree on the left shows 'Default Context'. The main panel displays the 'Alerts' screen, which lists 11 vulnerabilities: SQL Injection, Spring4Shell, Absence of Anti-CSRF Tokens (5), Content Security Policy (CSP) Header Not Set, and Parameter Tampering. The 'Alerts' tab is selected, and the 'Alerts (11)' folder is highlighted with a red box. The right pane provides details for the selected alert, mentioning manual addition and editing options.

After the vulnerability scan, the alert tells us 11 vulnerabilities :

SQL Injection, Spring4Shell, Absence of Anti-CSRF Tokens (5), content Security Policy (CSP) Header Not Set, Parameter Tampering, Cookie No HttpOnly Flag, Cookie without SameSite Attribute, Authentication Request Identified, Session Management Reponse Identified, User Agent Fuzzer (91), User Controllable HTML Element Attribute (Potential XSS).

3) Explore Vulnerabilities

SQL Injection :

The screenshot shows the ZAP interface with the 'Alerts' tab selected. A red box highlights the 'SQL Injection' alert for the URL `http://localhost:8080/WebGoat/register.mvc`. The alert details are as follows:

- Risk: High
- Confidence: Medium
- Parameter: agree
- Attack: agree' AND '1'='1
- Evidence: (not shown)

The screenshot shows the ZAP interface with the 'Alerts' tab selected. A red box highlights the 'SQL Injection' alert for the URL `http://localhost:8080/WebGoat/register.mvc`. The detailed information is as follows:

- CWE ID: 89
- WASC ID: 19
- Source: Active (40018 - SQL Injection)
- Input Vector: Form Query
- Description: SQL injection may be possible.

Other informations :

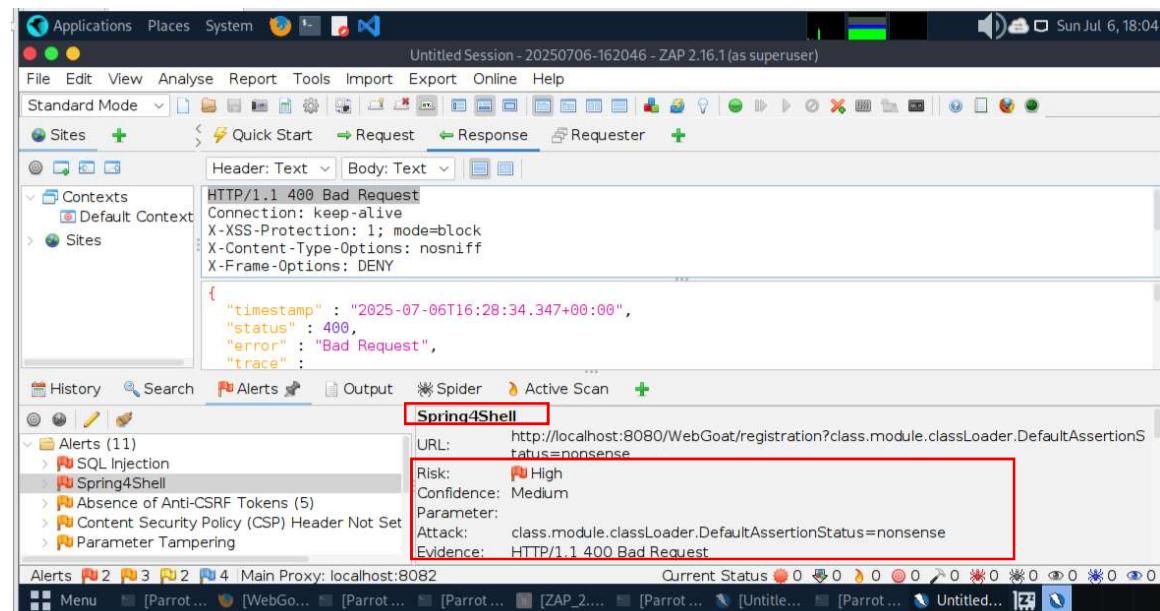
- The page results were successfully manipulated using the boolean conditions [agree' AND '1'='1] and [agree' AND '1'='2]
- The parameter value being modified was NOT stripped from the HTML output for the purposes of the comparison.

- Data was returned for the original parameter.
- The vulnerability was detected by successfully restricting the data originally returned, by manipulating the parameter.

Solution :

- Do not trust client side input, even if there is client side validation in place.
- In general, type check all data on the server side.
- If the application uses JDBC, use PreparedStatement or CallableStatement, with parameters passed by '?'
- If the application uses ASP, use ADO Command Objects with strong type checking and parameterized queries.
- If database Stored Procedures can be used, use them.
- Do *not* concatenate strings into queries in the stored procedure, or use 'exec', 'exec immediate', or equivalent functionality!
- Do not create dynamic SQL queries using simple string concatenation.
- Escape all data received from the client.
- Apply an 'allow list' of allowed characters, or a 'deny list' of disallowed characters in user input.
- Apply the principle of least privilege by using the least privileged database user possible.
- In particular, avoid using the 'sa' or 'db-owner' database users. This does not eliminate SQL injection, but minimizes its impact.
- Grant the minimum database access that is necessary for the application.

Spring4Shell :



The screenshot shows the ZAP interface with a red box highlighting an alert for 'Spring4Shell'. The alert details are as follows:

CWE ID:	78
WASC ID:	20
Source:	Active (40045 - Spring4Shell)
Input Vector:	Description: The application appears to be vulnerable to CVE-2022-22965 (otherwise known as Spring4Shell) - remote code execution (RCE) via data binding.

Description :

The application appears to be vulnerable to CVE-2022-22965 (otherwise known as Spring4Shell) - remote code execution (RCE) via data binding.

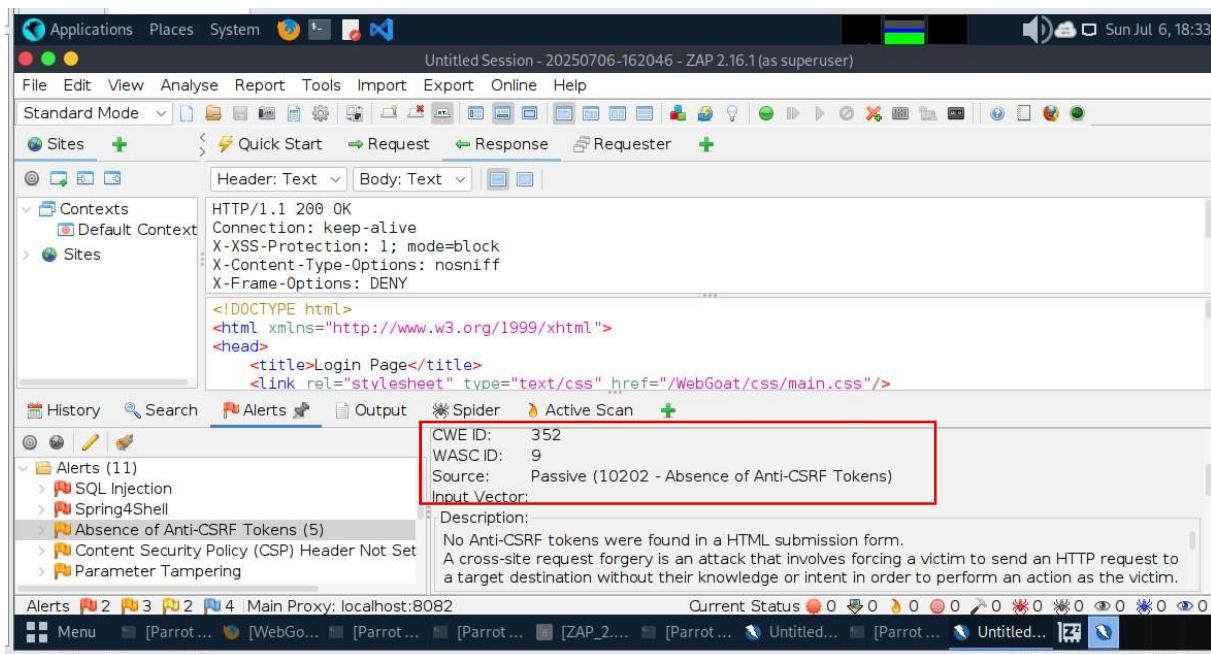
Solution :

Upgrade Spring Framework to versions 5.3.18, 5.2.20, or newer.

Absence of Anti-CSRF Tokens

The screenshot shows the ZAP interface with a red box highlighting an alert for 'Absence of Anti-CSRF Tokens'. The alert details are as follows:

Absence of Anti-CSRF Tokens	URL: http://localhost:8080/WebGoat
Risk:	Medium
Confidence:	Low
Parameter:	
Attack:	
Evidence:	<form action="/WebGoat/login" method='POST' style="width: 200px;">
CWE ID:	352



Description :

No Anti-CSRF tokens were found in a HTML submission form.

A cross-site request forgery is an attack that involves forcing a victim to send an HTTP request to a target destination without their knowledge or intent in order to perform an action as the victim. The underlying cause is application functionality using predictable URL/form actions in a repeatable way. The nature of the attack is that CSRF exploits the trust that a web site has for a user. By contrast, cross-site scripting (XSS) exploits the trust that a user has for a web site. Like XSS, CSRF attacks are not necessarily cross-site, but they can be. Cross-site request forgery is also known as CSRF, XSRF, one-click attack, session riding, confused deputy, and sea surf.

CSRF attacks are effective in a number of situations, including:

- The victim has an active session on the target site.
- The victim is authenticated via HTTP auth on the target site.
- The victim is on the same local network as the target site.

CSRF has primarily been used to perform an action against a target site using the victim's privileges, but recent techniques have been discovered to disclose information by gaining access to the response. The risk of information disclosure is dramatically increased when the target site is vulnerable to XSS, because XSS can be used as a platform for CSRF, allowing the attack to operate within the bounds of the same-origin policy.

Other Information :

No known Anti-CSRF token [anticsrf, CSRFToken, __RequestVerificationToken, csrfmiddlewaretoken, authenticity_token, OWASP_CSRFTOKEN, anonesrf, csrf_token, _csrf, _csrfSecret, __csrf_magic, CSRF, _token, __csrf_token, __csrfToken] was found in the following HTML form: [Form 1: "exampleInputEmail1" "exampleInputPassword1"].

Solution :

- Phase : Architecture and Design

Use a vetted library or framework that does not allow this weakness to occur or provides constructs that make this weakness easier to avoid.

For example, use anti-CSRF packages such as the OWASP CSRFGuard.

- Phase : Implementation

Ensure that your application is free of cross-site scripting issues, because most CSRF defenses can be bypassed using attacker-controlled script.

- Phase : Architecture and Design

Generate a unique nonce for each form, place the nonce into the form, and verify the nonce upon receipt of the form. Be sure that the nonce is not predictable (CWE-330).

Note that this can be bypassed using XSS.

Identify especially dangerous operations. When the user performs a dangerous operation, send a separate confirmation request to ensure that the user intended to perform that operation.

Note that this can be bypassed using XSS.

Use the ESAPI Session Management control.

This control includes a component for CSRF.

Do not use the GET method for any request that triggers a state change.

- Phase : Implementation

Check the HTTP Referer header to see if the request originated from an expected page. This could break legitimate functionality, because users or proxies may have disabled sending the Referer for privacy reasons.

Content Security Policy (CSP) Header Not Set

The screenshot shows the ZAP 2.16.1 interface with the following details:

- Header:** Content-Security-Policy
- Value:** (empty)
- Request URL:** http://localhost:8080/WebGoat
- Risk:** Medium
- Confidence:** High
- Parameter:** (empty)
- Attack:** (empty)
- Evidence:** (empty)
- CWE ID:** 693

Description :

Content Security Policy (CSP) is an added layer of security that helps to detect and mitigate certain types of attacks, including Cross Site Scripting (XSS) and data injection attacks. These attacks are used for everything from data theft to site defacement or distribution of malware. CSP provides a set of standard HTTP headers that allow website owners to declare approved sources of content that browsers should be allowed to load on that page — covered types are JavaScript, CSS, HTML frames, fonts, images and embeddable objects such as Java applets, ActiveX, audio and video files.

Solution :

Ensure that your web server, application server, load balancer, etc. is configured to set the Content-Security-Policy header.

Parameter Tampering

The screenshot shows the ZAP interface with the following details:

- Header:** Text
- Body:** Text
- Contexts:** Default Context
- Sites:** Sites
- Alerts:** Parameter Tampering (highlighted with a red box)
- Output:** URL: http://localhost:8080/WebGoat/register.mvc
- Risk:** Medium
- Confidence:** Medium
- Parameter:** matchingPassword
- Attack:**
- Evidence:** javax.servlet.http.HttpServlet.service(HttpServlet.java:517)\n\tat
- CWE ID:** 472

The screenshot shows the ZAP interface with a session titled "Untitled Session - 20250706-162046 - ZAP 2.16.1(as superuser)". The "Request" tab displays an error response with the following headers:

```
HTTP/1.1 500 Internal Server Error
Connection: keep-alive
Content-Type: application/json
Content-Disposition: inline;filename=f.txt
Date: Sun, 06 Jul 2025 16:29:15 GMT
```

The "Alerts" tab is selected, showing a single alert for "Parameter Tampering". The alert details are:

- WASC ID: 20
- Source: Active (40008 - Parameter Tampering)
- Input Vector: Form Query
- Description: Parameter manipulation caused an error page or Java stack trace to be displayed. This indicated lack of exception handling and potential areas for further exploit.

Description :

Parameter manipulation caused an error page or Java stack trace to be displayed. This indicated lack of exception handling and potential areas for further exploit.

Solution :

Identify the cause of the error and fix it. Do not trust client side input and enforce a tight check in the server side. Besides, catch the exception properly. Use a generic 500 error page for internal server error.

Cookie No HttpOnly Flag

The screenshot shows the ZAP interface with a session titled "Untitled Session - 20250706-162046 - ZAP 2.16.1(as superuser)". The "Request" tab displays a response with the following headers:

```
HTTP/1.1 302 Found
Connection: keep-alive
Set-Cookie: JSESSIONID=jBc8hhKNPDq0rG9hcFCaHNkLvmNeEfLFUMDvvr1X; path=/WebGoat
X-XSS-Protection: 1; mode=block
X-Content-Type-Options: nosniff
```

The "Alerts" tab is selected, showing a single alert for "Cookie No HttpOnly Flag". The alert details are:

- URL: http://localhost:8080/WebGoat/
- Risk: Low
- Confidence: Medium
- Parameter: JSESSIONID
- Attack:
- Evidence: Set-Cookie: JSESSIONID
- CWE ID: 1004

The screenshot shows the ZAP interface with the 'Alerts' tab selected. A specific alert is highlighted with a red box, detailing a 'Cookie No HttpOnly Flag' vulnerability. The alert information includes:

- WASC ID: 13
- Source: Passive (10010 - Cookie No HttpOnly Flag)
- Input Vector:
- Description: A cookie has been set without the HttpOnly flag, which means that the cookie can be accessed by JavaScript. If a malicious script can be run on this page then the cookie will be accessible and can be transmitted to another site. If this is a session cookie then session hijacking may be possible.
- Other Info:

Description :

A cookie has been set without the HttpOnly flag, which means that the cookie can be accessed by JavaScript. If a malicious script can be run on this page then the cookie will be accessible and can be transmitted to another site. If this is a session cookie then session hijacking may be possible.

Solution:

Ensure that the HttpOnly flag is set for all cookies.

Cookie without SameSite Attribute

The screenshot shows the ZAP interface with the 'Alerts' tab selected. A specific alert is highlighted with a red box, detailing a 'Cookie without SameSite Attribute' vulnerability. The alert information includes:

- URL: http://localhost:8080/WebGoat/
- Risk: Low
- Confidence: Medium
- Parameter: JSESSIONID
- Attack:
- Evidence: Set-Cookie: JSESSIONID=1275
- CWE ID: 1275

The screenshot shows the ZAP interface in Standard Mode. The 'Alerts' tab is selected, displaying a list of vulnerabilities. One alert is highlighted with a red border:

- WASC ID: 13
- Source: Passive (10054 - Cookie without SameSite Attribute)
- Alert Reference: 10054-1
- Input Vector:

Description:
A cookie has been set without the SameSite attribute, which means that the cookie can be sent as a result of a 'cross-site' request. The SameSite attribute is an effective counter measure to cross-site request forgery, cross-site script inclusion, and timing attacks.

Description :

A cookie has been set without the SameSite attribute, which means that the cookie can be sent as a result of a 'cross-site' request. The SameSite attribute is an effective counter measure to cross-site request forgery, cross-site script inclusion, and timing attacks.

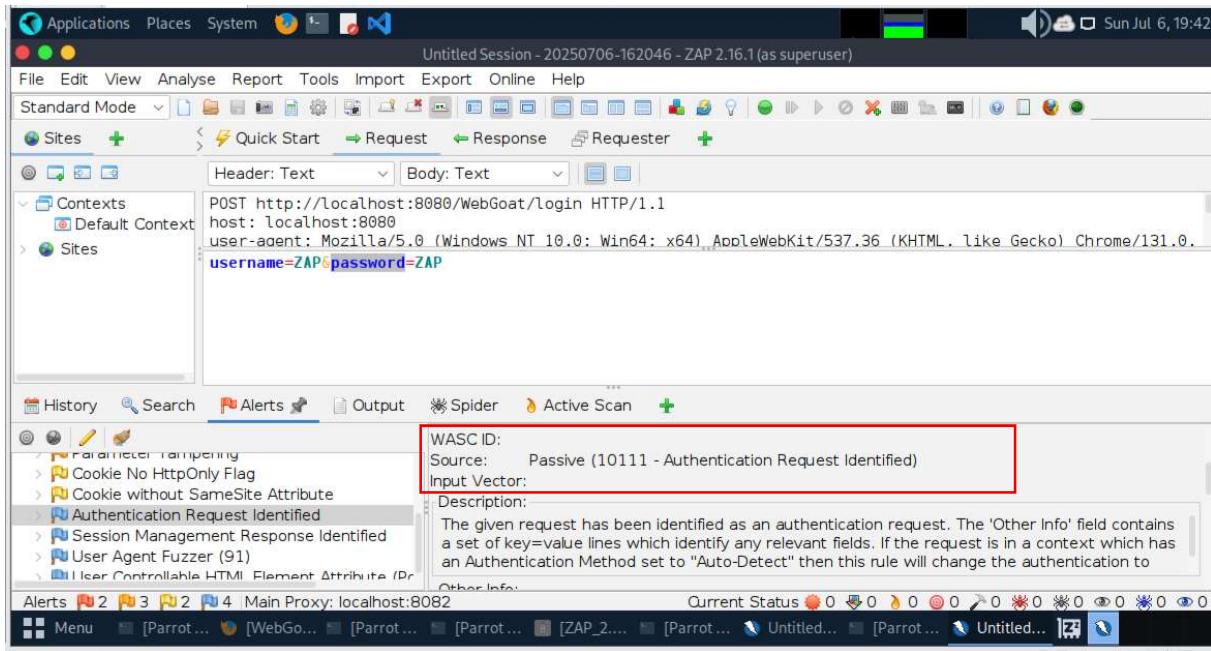
Solution :

Ensure that the SameSite attribute is set to either 'lax' or ideally 'strict' for all cookies.

Authentication Request Identified

The screenshot shows the ZAP interface in Standard Mode. The 'Alerts' tab is selected, displaying a list of vulnerabilities. One alert is highlighted with a red border:

- Authentication Request Identified
- URL: http://localhost:8080/WebGoat/login
- Risk: Informational
- Confidence: High
- Parameter: username
- Attack:
- Evidence: password
- CWE ID:



Description :

The given request has been identified as an authentication request. The 'Other Info' field contains a set of key=value lines which identify any relevant fields. If the request is in a context which has an Authentication Method set to "Auto-Detect" then this rule will change the authentication to match the request identified.

Other Info:

userParam=username

userValue=ZAP

passwordParam=password

referer=http://localhost:8080/WebGoat/login

Solution :

This is an informational alert rather than a vulnerability and so there is nothing to fix.

Session Management Response Identified

The screenshot shows the ZAP interface with a session titled "Untitled Session - 20250706-162046 - ZAP 2.16.1(as superuser)". In the "Alerts" tab, a red box highlights a "Session Management Response Identified" alert. The alert details are as follows:

- URL: http://localhost:8080/WebGoat/
- Risk: Informational
- Confidence: Medium
- Parameter: JSESSIONID
- Attack:
- Evidence: jBc8hhKNPDqOrG9hcFCaHNkLvmNeEfLFUMDvvr1X
- CWE ID:

The screenshot shows the ZAP interface with a session titled "Untitled Session - 20250706-162046 - ZAP 2.16.1(as superuser)". In the "Alerts" tab, a red box highlights a "Session Management Response Identified" alert. The alert details are as follows:

- WASC ID:
- Source: Passive (10112 - Session Management Response Identified)
- Input Vector:
- Description: The given response has been identified as containing a session management token. The 'Other Info' field contains a set of header tokens that can be used in the Header Based Session Management Method. If the request is in a context which has a Session Management Method set to "Auto-Detect" then this rule will change the session management to use the tokens identified.
- Other Info:

Description :

The given response has been identified as containing a session management token. The 'Other Info' field contains a set of header tokens that can be used in the Header Based Session Management Method. If the request is in a context which has a Session Management Method set to "Auto-Detect" then this rule will change the session management to use the tokens identified.

Other :

Cookie : JSESSIONID

Solution :

This is an informational alert rather than a vulnerability and so there is nothing to fix.

User Agent Fuzzer (91)

The screenshot shows the ZAP interface with the 'User Agent Fuzzer' alert highlighted. The alert details are as follows:

- URL: http://localhost:8080/WebGoat/
- Risk: Informational
- Confidence: Medium
- Parameter: Header User-Agent
- Attack: Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.1)
- Evidence:
- CWE ID: 0

Description :

Check for differences in response based on fuzzed User Agent (eg. mobile sites, access as a Search Engine Crawler). Compares the response statuscode and the hashcode of the response body with the original response.

The screenshot shows the ZAP interface with the 'User Agent Fuzzer' alert description expanded. The description is as follows:

WASC ID: 0
Source: Active (10104 - User Agent Fuzzer)
Input Vector:
Description:
Check for differences in response based on fuzzed User Agent (eg. mobile sites, access as a Search Engine Crawler). Compares the response statuscode and the hashcode of the response body with the original response.
Other Info:

User Controllable HTML Element Attribute (Potential XSS)

The screenshot shows the ZAP interface with a red box highlighting an alert for 'User Controllable HTML Element Attribute (Potential XSS)'. The alert details are as follows:

- URL: http://localhost:8080/WebGoat/register.mvc
- Risk: Informational
- Confidence: Low
- Parameter: matchingPassword
- Attack:
- Evidence:
- CWE ID: 20

Description :

This check looks at user-supplied input in query string parameters and POST data to identify where certain HTML attribute values might be controlled. This provides hot-spot detection for XSS (cross-site scripting) that will require further review by a security analyst to determine exploitability.

Other :

User-controlled HTML attribute values were found. Try injecting special characters to see if XSS might be possible. The page at the following URL:

Solution :

Validate all input and sanitize output it before writing to any HTML attributes.

The screenshot shows the ZAP interface with a red box highlighting the 'Description' field of the XSS alert. The description text is as follows:

WASC ID: 20
Source: Passive (10031 - User Controllable HTML Element Attribute (Potential XSS))
Input Vector:
Description:
This check looks at user-supplied input in query string parameters and POST data to identify where certain HTML attribute values might be controlled. This provides hot-spot detection for XSS (cross-site scripting) that will require further review by a security analyst to determine exploitability.

TASK 3: LINKEDIN ENGAGEMENT

Redynox OFFER LETTER

13 JUNE 2025

Dear ISSA HASSAN YOUSSEOUF

We are pleased to inform you that you have been selected for the Cybersecurity Internship Program at **Redynox**, scheduled to run from June 16, 2025, to July 15, 2025.

This internship will offer hands-on experience in various aspects of cybersecurity, including but not limited to network security, vulnerability assessment, penetration testing, and incident response. Selected candidates will work closely with our professional cybersecurity team and will be assigned real-world projects to enhance their technical and analytical skills.

We were highly impressed by your resume and the strong academic background it reflects. Your demonstrated interest in cybersecurity and commitment to continuous learning made you a standout candidate, and we are pleased to select you for this program based on your qualifications and potential.

Internship Details:

- Role: Cybersecurity Intern
- Duration: 1 Month
- Start Date: 16 June 2025
- Location: Remote
- Reporting Manager: Ankit Mathur, Internship Supervisor/Lead

Please confirm your acceptance of this offer by replying to this email no later than June 15, 2025.

We are excited to have you join us and look forward to seeing the impact you will make during your time with **Redynox**.

Sincerely,
Redynox Team

<https://redynox.theredusers.tech/> India@redynox.theredusers.tech

Issa Hassan Youssouf · You
Network engineer in training, passionate abo...
4w • Edited • 5

Excited to share a new milestone in my journey!

I'm honored to have been selected for a Cybersecurity Internship at **Redynox**.

A big thank you to the Redynox team for their trust and for this incredible opportunity to contribute to a safer digital environment.

This internship is a unique opportunity for me to apply my skills, learn alongside passionate professionals, and continue growing in a field I'm truly passionate about.

Location : Remote
Duration : 1 month
Role : Cybersecurity Intern ...more

15 2 comments

Like Comment Repost Send

Add a comment...

(4) Exécution du code

Stored XSS

Victime → **Pirate** → **Site web cible**

(1) Injection du code nocif dans un site vulnérable

(2) Visite de la page piégée

(3) Page demandée contenant le code injecté par le hacker

(5) Envoi de données

Issa Hassan Youssouf · You
Network engineer in training, passionate abo...
2w • 5

Understanding XSS Attacks : A Key to Web Application Security

As part of my cybersecurity internship at **Redynox**, I've been exploring various web vulnerabilities—one of the most impactful being Cross-Site Scripting (XSS). It comes in three main types, each with different characteristics and attack vectors:

- 1- Stored XSS : Persistent and server-side. The malicious script is saved (e.g., in a database) and executed whenever a user accesses the affected content.
- 2- Reflected XSS : Non-persistent but also server-side. The attack is delivered via a crafted link and reflected in the server response. ...more

Lydie Ouattara and 4 others

Like Comment Repost Send

Add a comment...

Debian 6.0.4 - VMware Workstation

File Edit View Search Terminal Help

```
user@parrot:~/Desktop]$ sudo ufw status
Status: inactive
user@parrot:~/Desktop]$ sudo ufw enable
Firewall is active and enabled on system startup
user@parrot:~/Desktop]$ sudo ufw deny 21
Rule added
Rule added (v6)
user@parrot:~/Desktop]$ sudo ufw allow 80
Rule added
Rule added (v6)
user@parrot:~/Desktop]$ sudo ufw allow 443
Rule added
Rule added (v6)
user@parrot:~/Desktop]$
```

Great Journey at **Redynox** !

I'm proud to share the completion of my cybersecurity internship tasks at Redynox, an experience that was both challenging and enriching.

During this internship, I had the opportunity to:

- Develop a solid understanding of system and network security fundamentals,
- Contribute to the analysis of security vulnerabilities (XSS, SQL injections, etc.) using tools like **OWASP® Foundation ZAP** and **Wireshark Foundation**,
- Basic security configurations, such as changing default passwords and enabling network encryption (WPA2 or WPA3), ...more

57 12 comments • 2 reposts

Like Comment Repost Send

CONCLUSION

This internship allowed me to take my first practical steps in the field of cybersecurity. I was able to implement different network security techniques, observe traffic behavior via tools like Wireshark, and understand the importance of securing communications over the Internet such as configuring basic security configurations, such as changing default passwords and enabling network encryption (WPA2 or WPA3) and with the Parrot OS environment as the primary support.

Exploring web vulnerabilities through WebGoat and OWASP ZAP gave me a concrete view of the real attacks that applications can undergo. I learned to identify flaws, understand them, and above all propose effective solutions to correct them.

The professional dimension was also valued through LinkedIn, where I was able to share my experiences and strengthen my presence in the field.

These activities allowed me to consolidate my theoretical knowledge, to discover new professional tools and to develop my technical rigor. I came out of this internship motivated to deepen my cybersecurity skills even more.