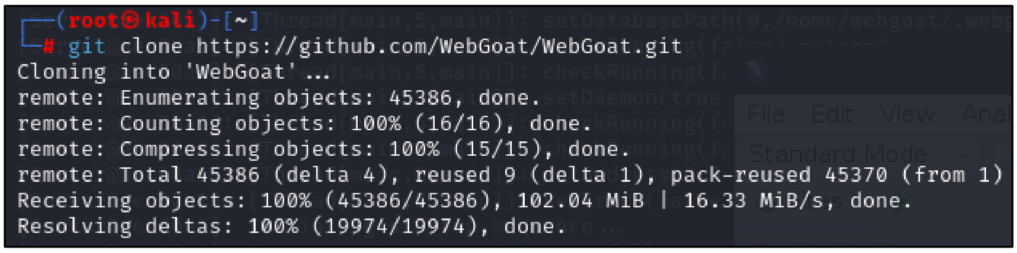
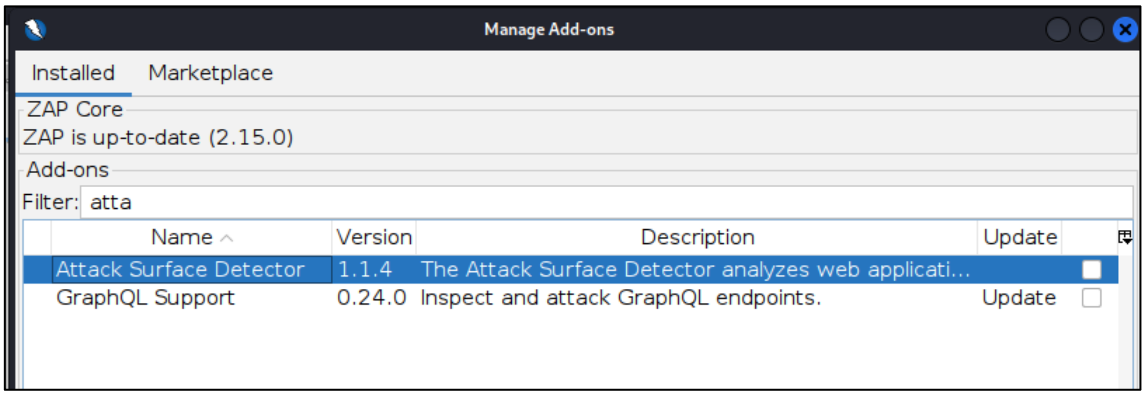
Cyber Security Assessment and Management

Worksheet #3 – Security Assessment

1. DOWNLOAD WEBGOAT SOURCE CODE



```
┌──(root㉿kali)-[~]
└─# git clone https://github.com/WebGoat/WebGoat.git
Cloning into 'WebGoat'...
remote: Enumerating objects: 45386, done.
remote: Counting objects: 100% (16/16), done.
remote: Compressing objects: 100% (15/15), done.
remote: Total 45386 (delta 4), reused 9 (delta 1), pack-reused 45370 (from 1)
Receiving objects: 100% (45386/45386), 102.04 MiB | 16.33 MiB/s, done.
Resolving deltas: 100% (19974/19974), done.
```



Manage Add-ons

Installed | Marketplace

ZAP Core

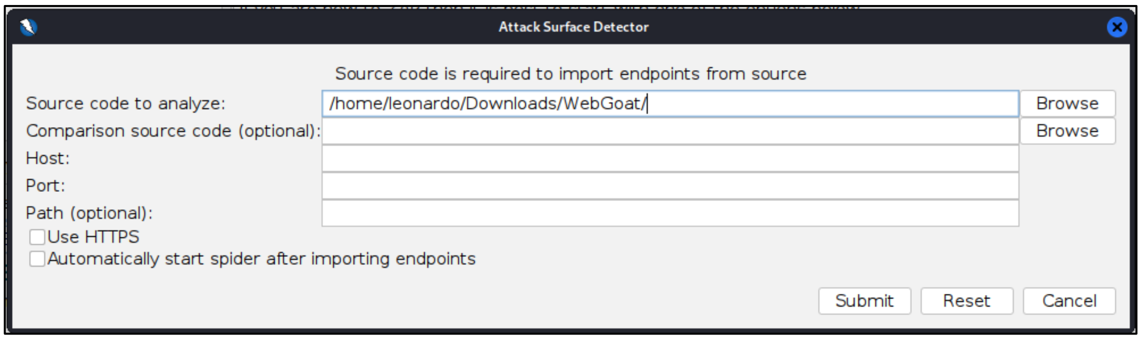
ZAP is up-to-date (2.15.0)

Add-ons

Filter: atta

Name	Version	Description	Update	
Attack Surface Detector	1.1.4	The Attack Surface Detector analyzes web applicati...		☐
GraphQL Support	0.24.0	Inspect and attack GraphQL endpoints.	Update	☐

2. IDENTIFY THE ATTACK SURFACE IN WEBGOAT USING ASD



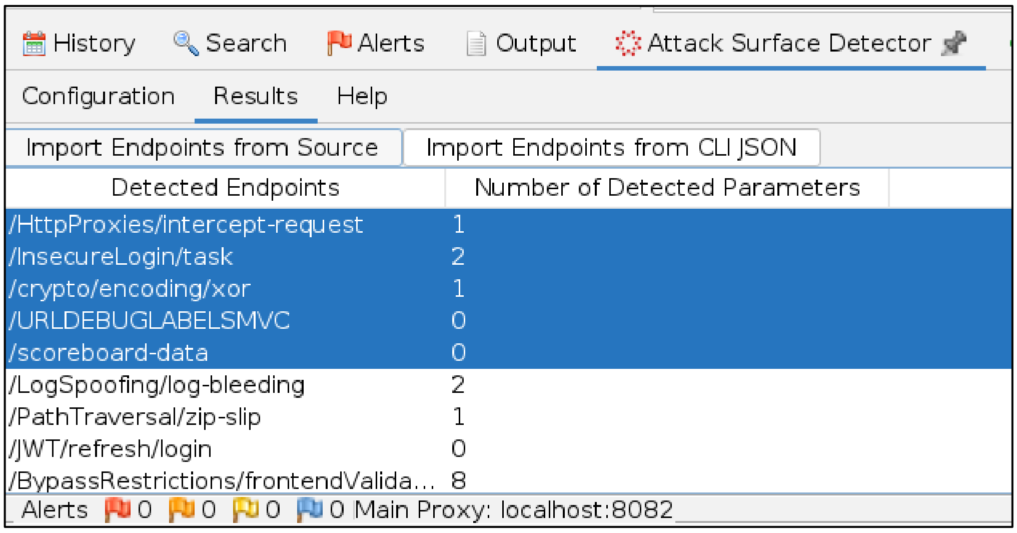
Attack Surface Detector

Source code is required to import endpoints from source

Source code to analyze:	/home/leonardo/Downloads/WebGoat/
Comparison source code (optional):	
Host:	
Port:	
Path (optional):	

☐ Use HTTPS

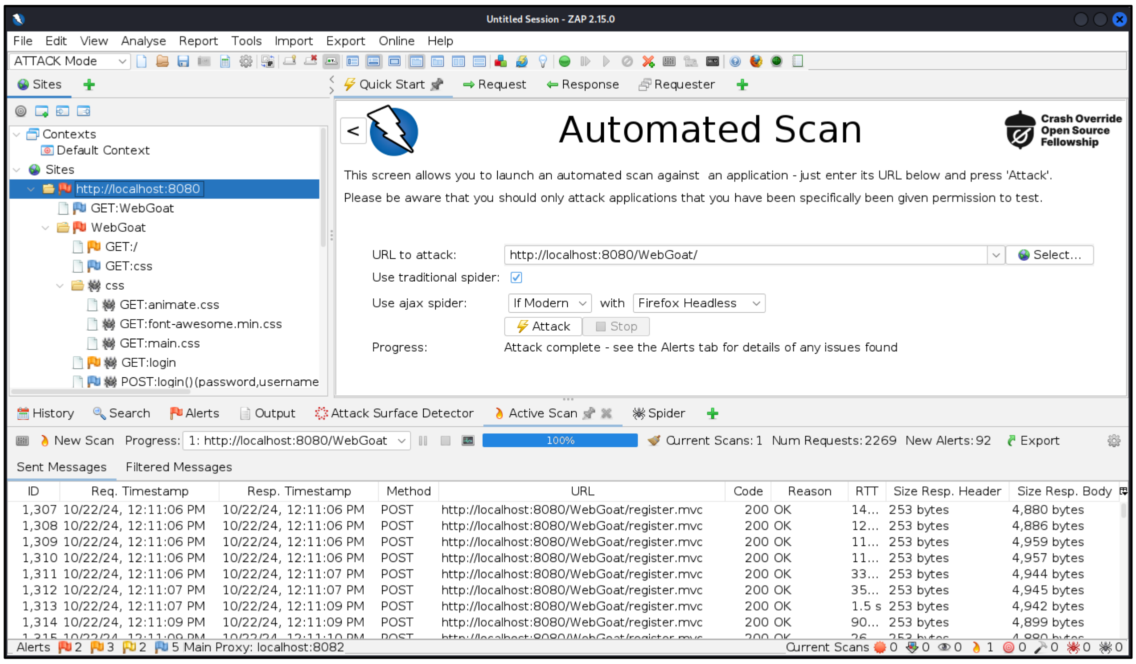
☐ Automatically start spider after importing endpoints



Detected Endpoints	Number of Detected Parameters
/HttpProxies/intercept-request	1
/InsecureLogin/task	2
/crypto/encoding/xor	1
/URLDEBUGLABELSMVC	0
/scoreboard-data	0
/LogSpoofing/log-bleeding	2
/PathTraversal/zip-slip	1
/JWT/refresh/login	0
/BypassRestrictions/frontendValida...	8

Alerts 0 0 0 0 Main Proxy: localhost:8082

R: The number of detected endpoints by choosing the "Import Endpoints from Source" option in the Attack Surface Detector are **195** without removing duplicates or **165** after removing duplicates.



This screen allows you to launch an automated scan against an application - just enter its URL below and press 'Attack'.
Please be aware that you should only attack applications that you have been specifically been given permission to test.

URL to attack: http://localhost:8080/WebGoat/
Use traditional spider: ✓
Use ajax spider: If Modern with Firefox Headless
Progress: Attack complete - see the Alerts tab for details of any issues found

Progress: 1: http://localhost:8080/WebGoat 100% Current Scans: 1 Num Requests: 2269 New Alerts: 92

ID	Req. Timestamp	Resp. Timestamp	Method	URL	Code	Reason	RTT	Size Resp. Header	Size Resp. Body
1,307	10/22/24, 12:11:06 PM	10/22/24, 12:11:06 PM	POST	http://localhost:8080/WebGoat/register.mvc	200	OK	14...	253 bytes	4,880 bytes
1,308	10/22/24, 12:11:06 PM	10/22/24, 12:11:06 PM	POST	http://localhost:8080/WebGoat/register.mvc	200	OK	12...	253 bytes	4,886 bytes
1,309	10/22/24, 12:11:06 PM	10/22/24, 12:11:06 PM	POST	http://localhost:8080/WebGoat/register.mvc	200	OK	11...	253 bytes	4,959 bytes
1,310	10/22/24, 12:11:06 PM	10/22/24, 12:11:06 PM	POST	http://localhost:8080/WebGoat/register.mvc	200	OK	11...	253 bytes	4,957 bytes
1,311	10/22/24, 12:11:06 PM	10/22/24, 12:11:07 PM	POST	http://localhost:8080/WebGoat/register.mvc	200	OK	33...	253 bytes	4,944 bytes
1,312	10/22/24, 12:11:07 PM	10/22/24, 12:11:07 PM	POST	http://localhost:8080/WebGoat/register.mvc	200	OK	35...	253 bytes	4,945 bytes
1,313	10/22/24, 12:11:07 PM	10/22/24, 12:11:09 PM	POST	http://localhost:8080/WebGoat/register.mvc	200	OK	1.5 s	253 bytes	4,942 bytes
1,314	10/22/24, 12:11:09 PM	10/22/24, 12:11:09 PM	POST	http://localhost:8080/WebGoat/register.mvc	200	OK	90...	253 bytes	4,899 bytes

Alerts 2 3 2 5 Main Proxy: localhost:8082 Current Scans 0 0 0 1 0 0 0 0

R: The number of endpoints and urls found by the ZAP scan is **20**.

Answer the following questions:

1. What is the most reliable source for determining the attack surface ? Justify your answer.

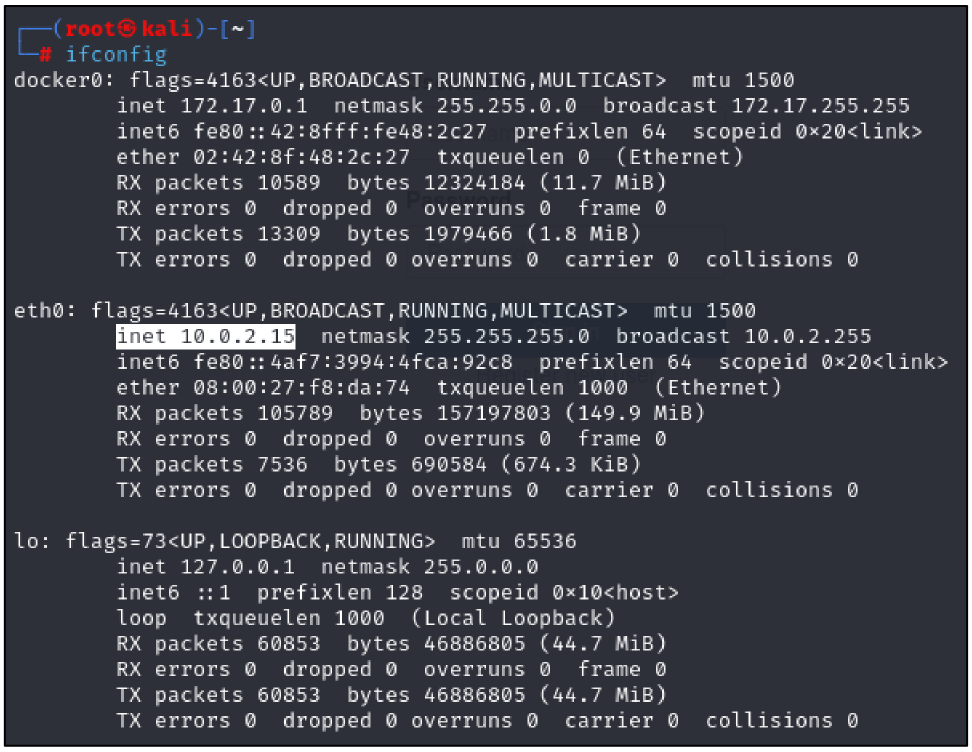
R: The most reliable source for determining the attack surface is the **source code analysis** using the Attack Surface Detector because it gives a complete list of endpoints since it has access to the entire application codebase, including hidden or unused endpoints that might not be actively exposed during a dynamic scan. It reveals all possible URLs, even those not reachable from user input or that require special conditions.

2. **Which approach would you use to determine the most accurate attack surface ? Justify your answer.**

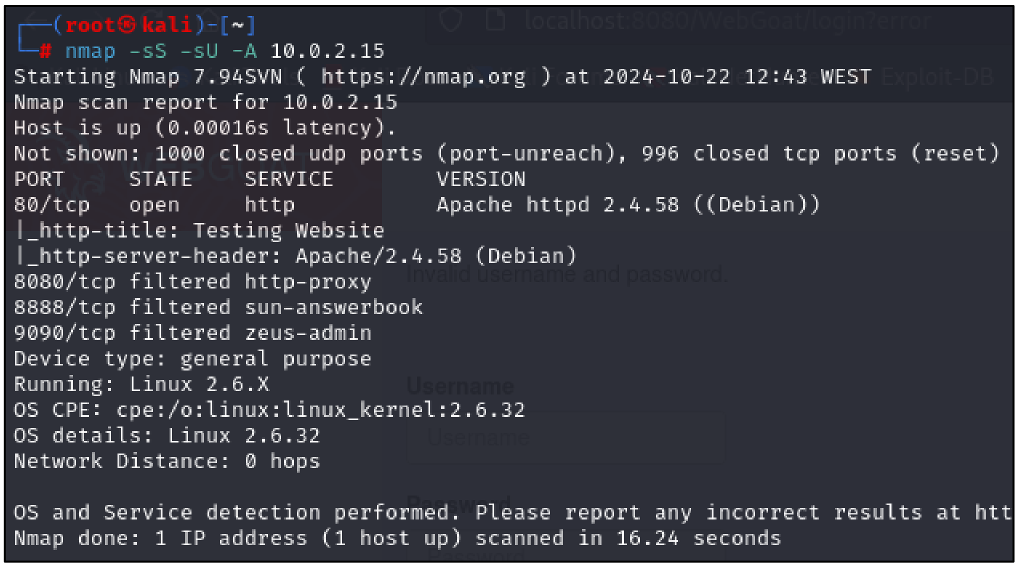
R: The most accurate approach is to combine both **source code analysis and dynamic analysis (like ZAP automated scan using ATTACKER MODE).** Source Code Analysis ensures no endpoint is missed, including internal APIs, debug endpoints, or developer backdoors. Dynamic Analysis helps understand how the application behaves under real conditions and identifies potential security vulnerabilities that an attacker might exploit.

By **combining both**, we can ensure completeness (from source code analysis) and real-world relevance (from dynamic analysis), offering the most **accurate representation** of the attack surface.

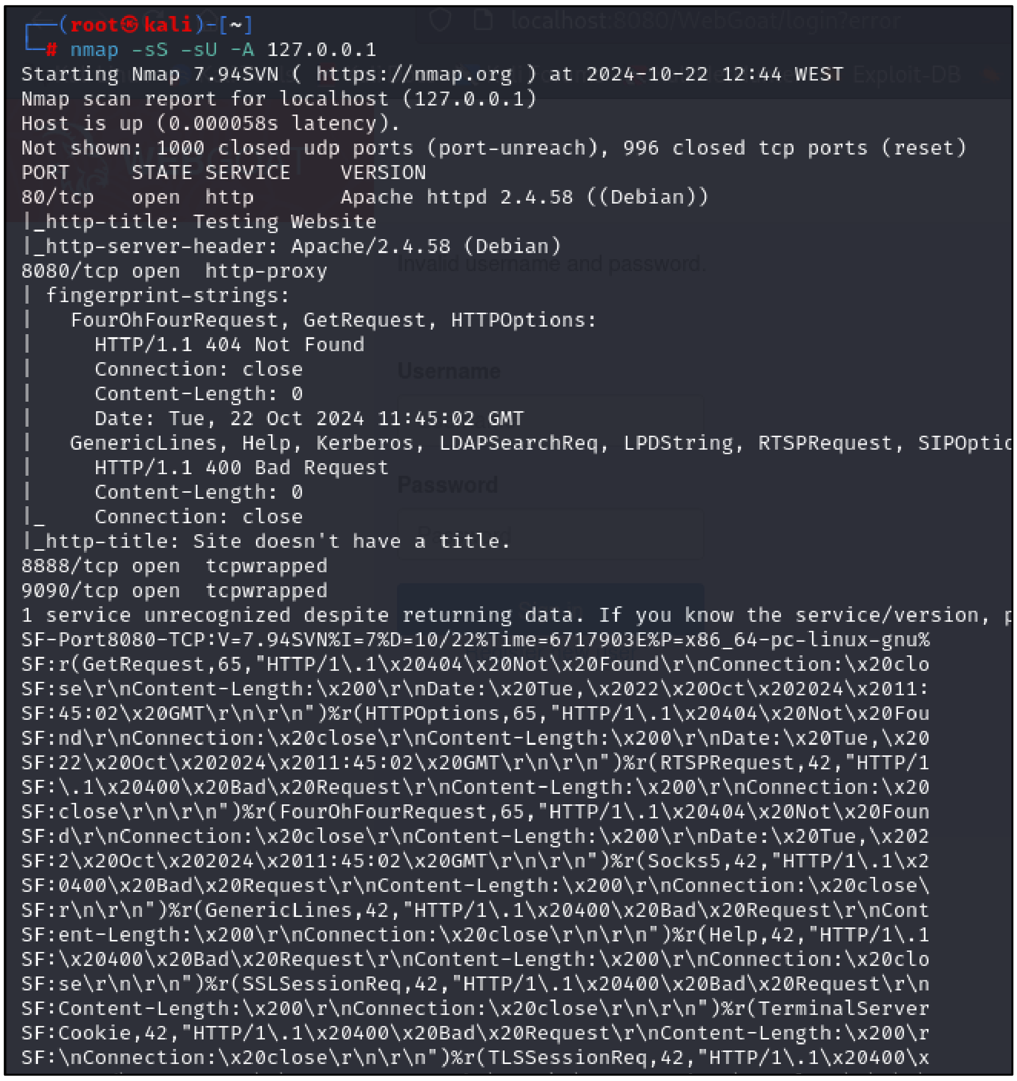
3. IDENTIFY THE ATTACK SURFACE IN ENVIRONMENT WHERE THE APPLICATIONS RUN



```
┌──(root㉿kali)-[~]
└─# ifconfig
docker0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 172.17.0.1  netmask 255.255.0.0  broadcast 172.17.255.255
        inet6 fe80::42:8fff:fe48:2c27  prefixlen 64  scopeid 0x20<link>
        ether 02:42:8f:48:2c:27  txqueuelen 0  (Ethernet)
        RX packets 10589  bytes 12324184 (11.7 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 13309  bytes 1979466 (1.8 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.2.15  netmask 255.255.255.0  broadcast 10.0.2.255
        inet6 fe80::4af7:3994:4fca:92c8  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:f8:da:74  txqueuelen 1000  (Ethernet)
        RX packets 105789  bytes 157197803 (149.9 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 7536  bytes 690584 (674.3 KiB)
        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 60853  bytes 46886805 (44.7 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 60853  bytes 46886805 (44.7 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```



```
┌──(root㉿kali)-[~]
└─# nmap -sS -sU -A 10.0.2.15
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-10-22 12:43 WEST
Nmap scan report for 10.0.2.15
Host is up (0.00016s latency).
Not shown: 1000 closed udp ports (port-unreach), 996 closed tcp ports (reset)
PORT     STATE    SERVICE        VERSION
80/tcp   open     http           Apache httpd 2.4.58 ((Debian))
|_http-title: Testing Website
|_http-server-header: Apache/2.4.58 (Debian)
8080/tcp filtered http-proxy
8888/tcp filtered sun-answerbook
9090/tcp filtered zeus-admin
Device type: general purpose
Running: Linux 2.6.X
OS CPE: cpe:/o:linux:linux_kernel:2.6.32
OS details: Linux 2.6.32
Network Distance: 0 hops

OS and Service detection performed. Please report any incorrect results at htt
Nmap done: 1 IP address (1 host up) scanned in 16.24 seconds
```



```
┌──(root㉿kali)-[~]
└─# nmap -sS -sU -A 127.0.0.1
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-10-22 12:44 WEST
Nmap scan report for localhost (127.0.0.1)
Host is up (0.000058s latency).
Not shown: 1000 closed udp ports (port-unreach), 996 closed tcp ports (reset)
PORT     STATE SERVICE    VERSION
80/tcp   open  http       Apache httpd 2.4.58 ((Debian))
|_http-title: Testing Website
|_http-server-header: Apache/2.4.58 (Debian)
8080/tcp open  http-proxy
| fingerprint-strings:
|   FourOhFourRequest, GetRequest, HTTPOptions:
|     HTTP/1.1 404 Not Found
|     Connection: close
|     Content-Length: 0
|     Date: Tue, 22 Oct 2024 11:45:02 GMT
|   GenericLines, Help, Kerberos, LDAPSearchReq, LPDString, RTSPRequest, SIPOptic
|     HTTP/1.1 400 Bad Request
|     Content-Length: 0
|_    Connection: close
|_http-title: Site doesn't have a title.
8888/tcp open  tcpwrapped
9090/tcp open  tcpwrapped
1 service unrecognized despite returning data. If you know the service/version, p
SF-Port8080-TCP:V=7.94SVN%I=7%D=10/22%Time=6717903E%P=x86_64-pc-linux-gnu%
SF:r(GetRequest,65,"HTTP/1\.1\x20404\x20Not\x20Found\r\nConnection:\x20clo
SF:se\r\nContent-Length:\x200\r\nDate:\x20Tue,\x2022\x20Oct\x202024\x2011:
SF:45:02\x20GMT\r\n\r\n")%r(HTTPOptions,65,"HTTP/1\.1\x20404\x20Not\x20Fou
SF:nd\r\nConnection:\x20close\r\nContent-Length:\x200\r\nDate:\x20Tue,\x20
SF:22\x20Oct\x202024\x2011:45:02\x20GMT\r\n\r\n")%r(RTSPRequest,42,"HTTP/1
SF:\.1\x20400\x20Bad\x20Request\r\nContent-Length:\x200\r\nConnection:\x20
SF:close\r\n\r\n")%r(FourOhFourRequest,65,"HTTP/1\.1\x20404\x20Not\x20Foun
SF:d\r\nConnection:\x20close\r\nContent-Length:\x200\r\nDate:\x20Tue,\x202
SF:2\x20Oct\x202024\x2011:45:02\x20GMT\r\n\r\n")%r(Socks5,42,"HTTP/1\.1\x2
SF:0400\x20Bad\x20Request\r\nContent-Length:\x200\r\nConnection:\x20close\
SF:r\n\r\n")%r(GenericLines,42,"HTTP/1\.1\x20400\x20Bad\x20Request\r\nCont
SF:ent-Length:\x200\r\nConnection:\x20close\r\n\r\n")%r(Help,42,"HTTP/1\.1
SF:\x20400\x20Bad\x20Request\r\nContent-Length:\x200\r\nConnection:\x20clo
SF:se\r\n\r\n")%r(SSLSessionReq,42,"HTTP/1\.1\x20400\x20Bad\x20Request\r\n
SF:Content-Length:\x200\r\nConnection:\x20close\r\n\r\n")%r(TerminalServer
SF:Cookie,42,"HTTP/1\.1\x20400\x20Bad\x20Request\r\nContent-Length:\x200\r
SF:\nConnection:\x20close\r\n\r\n")%r(TLSSessionReq,42,"HTTP/1\.1\x20400\x
```

R: To scan both TCP and UDP ports and also detect the services running on those ports, we can use the following command: **nmap -sS -sU -A <target-ip>.**

It was used the target-ip **10.0.2.15** and the **127.0.0.1** (localhost).

The results from the target-ip **10.0.2.15** indicates that the only confirmed open port is **80/tcp**, which is used for HTTP traffic. This suggests that a web server (Apache) is running on this port, and it may be the main attack surface for web-based attacks. Ports 8080, 8888, and 9090 are not directly accessible due to filtering, which could indicate firewall rules in place to secure these ports.

The results from the target-ip **127.0.1** indicates that there are **4 open TCP ports**: **80, 8080, 8888, 9090.**

Port 80 is commonly used for HTTP traffic, indicating that your system might be serving a website or web application (Apache).

Port 8080 is often used as an alternative HTTP port and is used for proxy services. It's also commonly used for web applications (in this case, the WebGoat application).

Port 8888 could be used for various services, and since it shows as tcpwrapped, it suggests a service is running but not fully identified.

Port 9090 could be associated with web services as well, often used for APIs or alternate web services.

4. ANALYSE THE RESULTS

Reducing the overall attack surface of an application involves a combination of best practices in security, configuration management, and monitoring. Based on the results obtained from your Nmap scan and the endpoints identified in the previous steps, here's some recommendations to minimize the attack surface.

- **Limit Open Ports**:
  - ❖ **Restrict Unnecessary Services**: Close any ports not actively in use. For example, if we don't need the proxy on port 8080 or any wrapped services on 8888 and 9090, we should consider shutting them down.
  - ❖ **Firewall Rules**: Implement firewall rules to restrict access to these ports. Only allow access from trusted IP addresses or networks.

- **Application Hardening**:
  - ❖ **Remove Default Pages and Unused Features**: Ensure that default pages or features of web servers like Apache are removed or disabled. Review the configuration to disable unnecessary modules.
  - ❖ **Input Validation and Output Encoding**: Implement proper input validation and output encoding for all application inputs to prevent attacks such as SQL injection, XSS, and command injection.

- **Authentication and Authorization**:

  - **Implement Strong Authentication Mechanisms**: Apply multi-factor authentication (MFA) and strong password policies to secure access to your applications.
  - **Restrict User Privileges**: Use the principle of least privilege to limit access to sensitive functions or data. This reduces the risk of exploitation through compromised accounts.

- **Regular Updates and Patching**:
  - **Keep Software Up-to-Date**: Regularly update the application, libraries, and server software to protect against known vulnerabilities. Configure the system to automatically apply security patches when they become available.

- **Security Testing**:
  - **Regular Vulnerability Scans and Penetration Testing**: Conduct regular security assessments to identify and mitigate potential vulnerabilities.
  - **Monitoring and Logging**: Implement logging to capture attempts to access endpoints and ports. Use intrusion detection systems (IDS) to monitor for unusual activity.

- **Network Segmentation**:
  - **Isolate Sensitive Applications**: Place critical applications on separate network segments to limit exposure. For example, if WebGoat is being used for training, we should consider isolating it from production environments.

- **Use of Web Application Firewalls (WAF)**:
  - **Deploy a WAF**: Implement a WAF to monitor and filter incoming traffic to your web application, protecting against common web exploits.