

Security Tools Lab 2
Project 1
SOC: Security Ops Centre

Student Name: Gowtham Baskar

Student ID: 1006523

Table of Content

| | |
|---|----|
| 1. Introduction | 3 |
| 2. Network Topology..... | 4 |
| 3. Attack Scenario | 6 |
| 3.1 Dictionary Attack on SSH | 6 |
| 3.2 Brute force FTP | 7 |
| 3.3 Netcat Listening rule (Rule Based) | 9 |
| 3.4 SQL Injection Attack..... | 11 |
| 3.5 NMAP Probing Attack (NMAP Script Engine Detect) | 13 |
| 4. Conclusion..... | 14 |
| 5. References | 15 |

1. Introduction

In today's world, organizations have multiple endpoints/machines running multiple services to perform business operations efficiently. At present, even a small-scale company has a minimum of ten systems in its infrastructure and they are constantly collecting data and generating logs. The biggest challenge that today's organizations have is analysing and utilizing these big data. These endpoints generate multiple records crucial for analysing and creating alerts for critical security violations and attacks. Organizations use SIEM (Security Information Event Management) solutions to handle this problem.

It is important to know two key terms and their definitions to understand the meaning of SIEM:

SEM – Security Event Management deals with collecting logs from endpoints.

SIM – Security Information Management deals with analysing the collected logs.

SEM + SIM = SIEM (Security Information Event Management)

SIEM is a solution that helps organizations in collecting logs and converting the records into useful information that can be analysed. It also provides real-time monitoring & analysing capabilities that helps in creating alerts when any security violation or attack occurs.

The most popular and widely used SIEM out there is WAZUH. Wazuh is used worldwide by many companies, starting from small firms to big corporations.

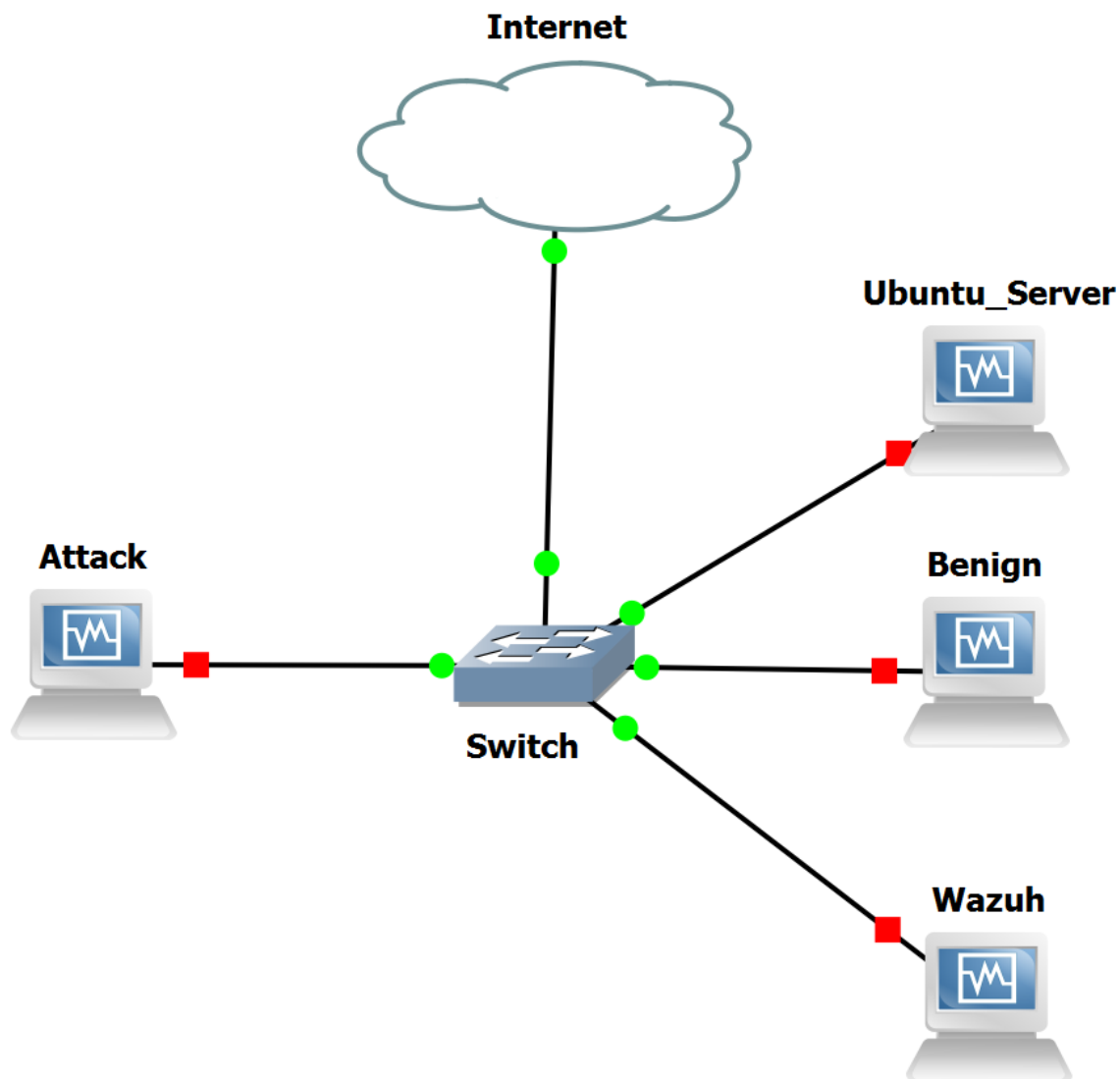
Wazuh is a free and open-source platform that is used for threat detection, prevention, and response.

It is typically used to protect networks, virtualized environments, containers, and cloud environments.

Wazuh is a SIEM system used for collecting, aggregating, indexing, and analysing security related data that allows you to detect attacks, intrusions, threats, vulnerabilities, and malicious activities.

We will be using Wazuh in our project to detect attack in our internal network that was set up in GNS3.

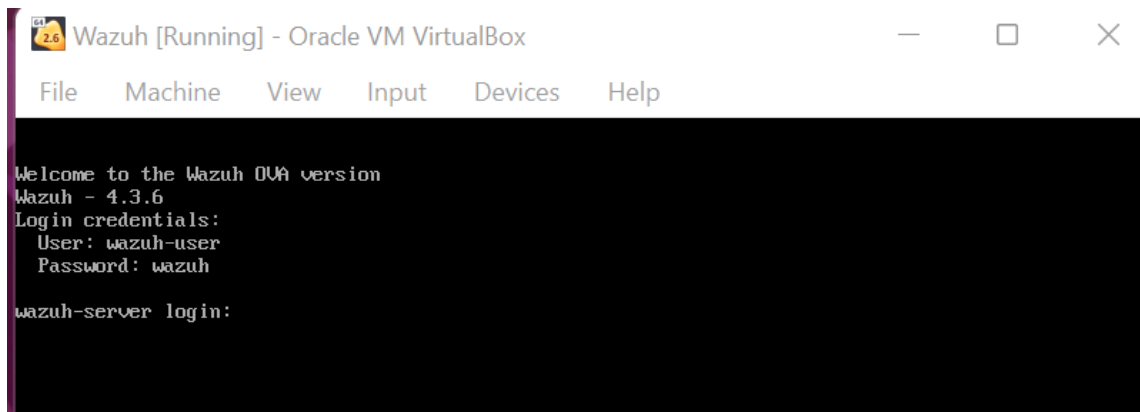
2. Network Topology



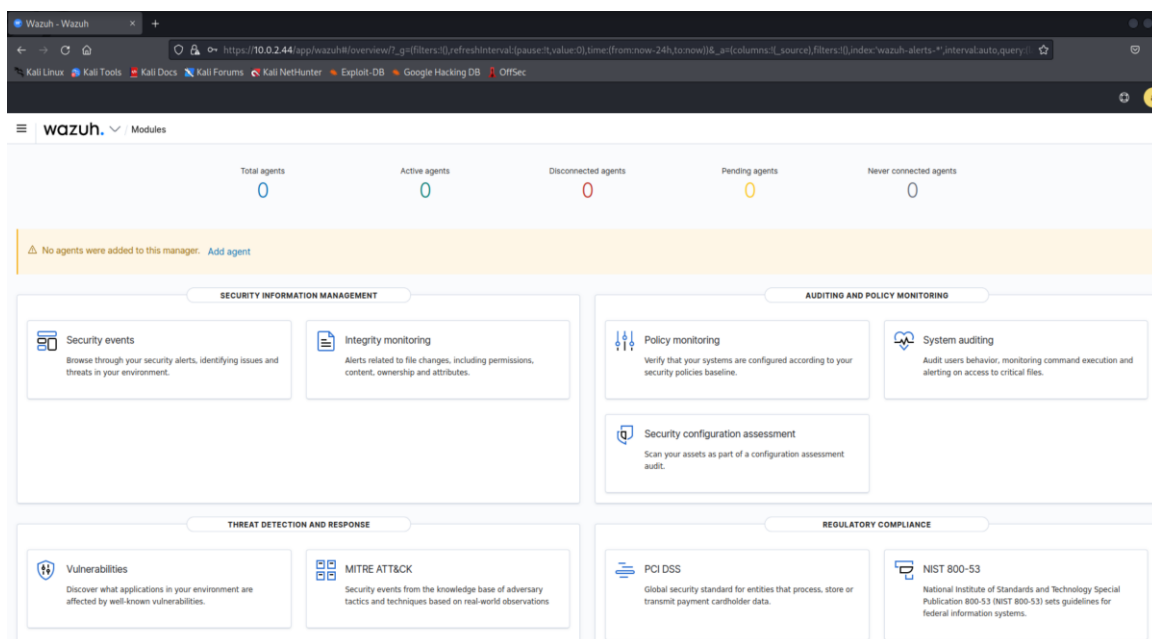
| Host | IP (NAT Network is set as 10.0.2.0/24) |
|--------------------|--|
| Security Agent | |
| Wazuh | 10.0.2.44 (Server) |
| External Network | |
| Malicious | 10.0.2.47 (|
| Internal Network | |
| Benign_STL2 | 10.0.2.42 (Wazuh Agent Installed) |
| Ubuntu Server_STL2 | 10.0.2.46 (Wazuh Agent Installed) |

On the Ubuntu server, these are the following services that are running, the FTP, SSH, http, Domain, MariaDB and the SMTP.

Wazuh Server

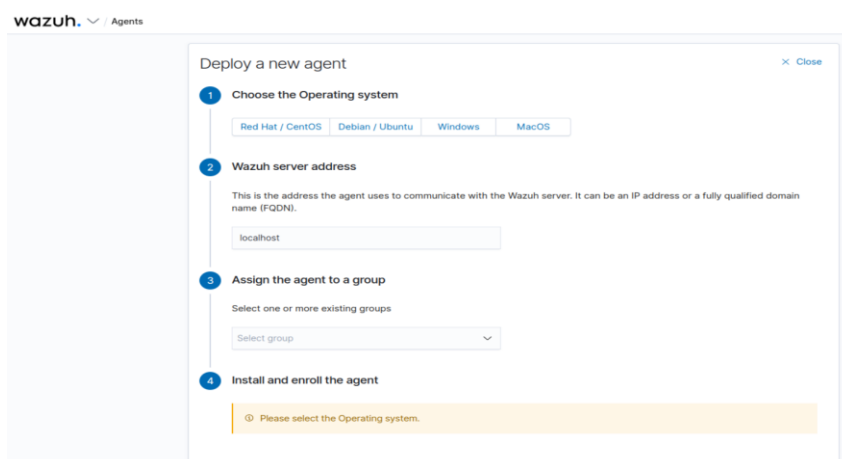


Wazuh Dashboard



Installing Agents

I have installed the agents to my Benign Machine and my ubuntu Server. Installation is guided by Wazuh as below



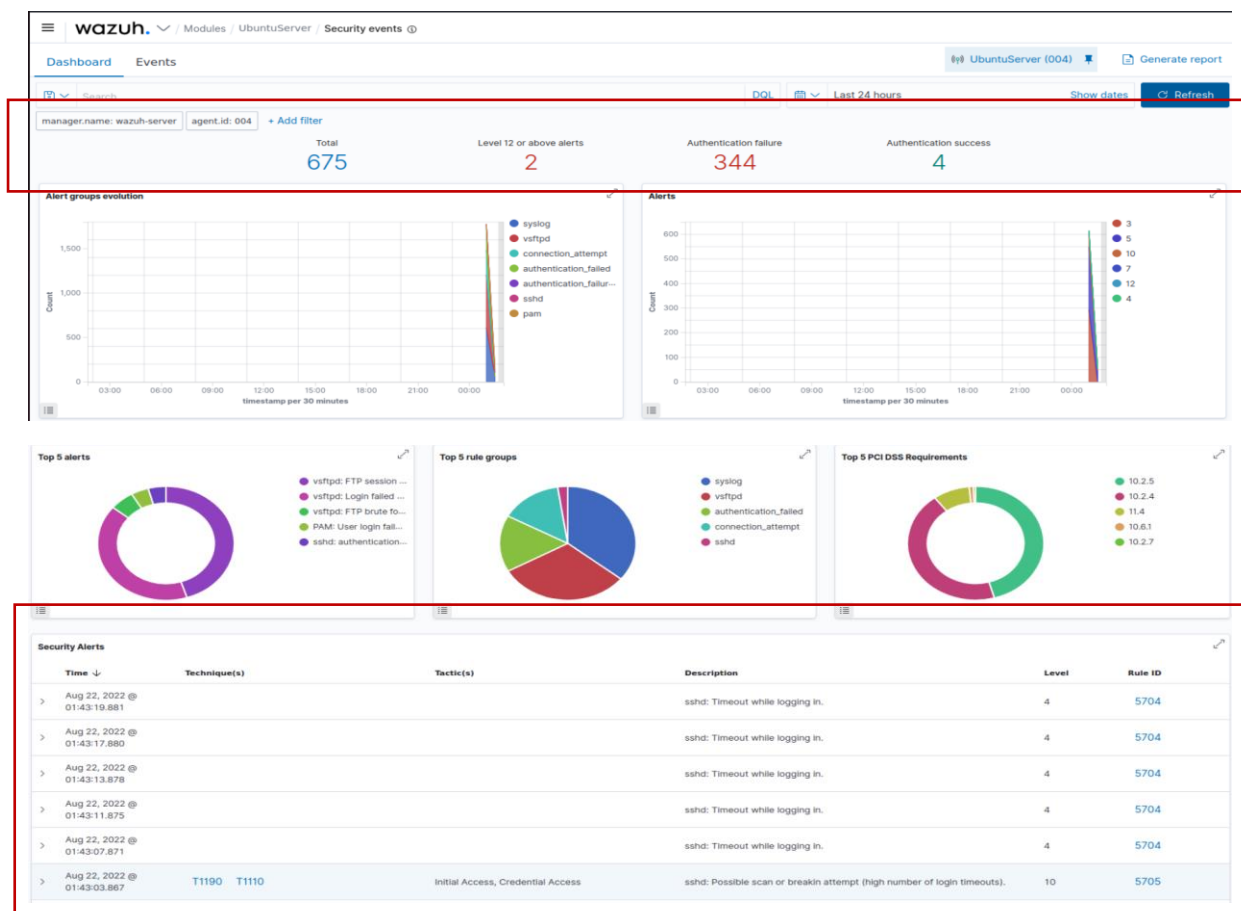
3. Attack Scenario

3.1 Dictionary Attack on SSH

I've attached the code along with the project folder as "Dictionary_SSH.py".

```
(kali@kali) - [~/Desktop/Project/Attacks]
$ python3 Dictionary_SSH.py 10.0.2.46 -u mssd -P wordlist.txt
[!] Invalid credentials for mssd:123456
[!] Invalid credentials for mssd:12345
[!] Invalid credentials for mssd:123456789
[!] Invalid credentials for mssd:password
[!] Invalid credentials for mssd:iloveyou
[!] Invalid credentials for mssd:princess
[!] Invalid credentials for mssd:12345678
[!] Invalid credentials for mssd:1234567
[!] Invalid credentials for mssd:1234567890
```

After running the code, I was able to detect the events in my security alert dashboard as shown below in red.



We have successfully managed to attack the ubuntu server from the attack machine and was successfully able to detect the Dictionary attempt (Credential Guessing).

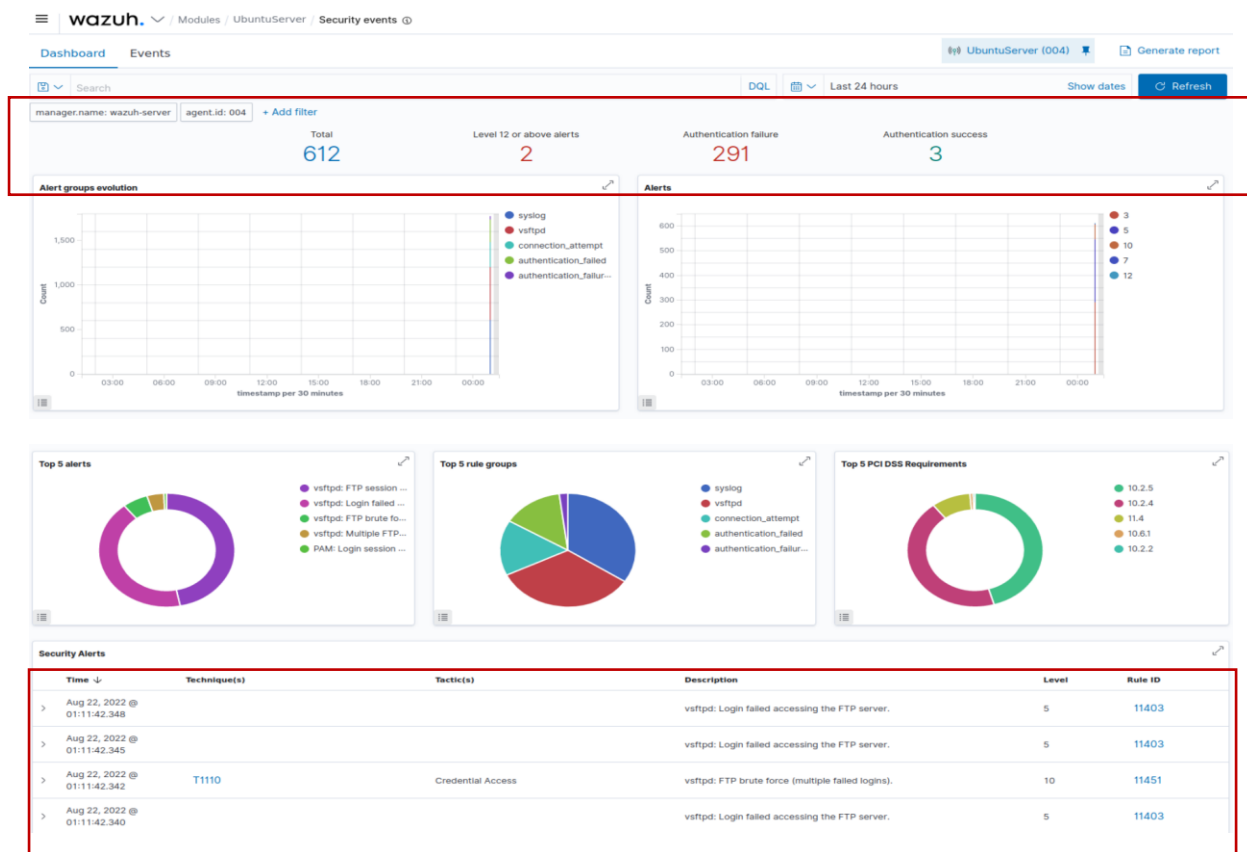
3.2 Brute force FTP

I've attached the code along with the project folder as "Bruteforce_FTP.py".

After executing the code, I was able to find the ftp credential as shown below

```
[!] Trying dmsd
[!] Trying amsd
[!] Trying mssd
[!] Trying sssd
[+] Found credentials:
    Host: 10.0.2.46
    User: mssd
    Password: mssd
Exception in thread Thread-5 (connect_ftp):
Traceback (most recent call last):
  File "/usr/lib/python3.10/threading.py", line 1009, in _bootstrap_inner
    self.run()
  File "/usr/lib/python3.10/threading.py", line 946, in run
```

After running the code, I was able to detect the events in my security alert dashboard as shown below in red.



The scores in the dashboard are different as I've first executed the Brute force FTP attack followed by the Dictionary SSH Attack.

We have successfully managed to attack the ubuntu server from the attack machine and was successfully able to detect the brute force attempt.

Security events ①

09:0012:0015:0018:0021:0000:0003:0006:00

timestamp per 30 minutes

| Time | rule.description | rule.level | rule.id |
|-----------------------------|--|------------|---------|
| Aug 19, 2022 @ 08:15:31.593 | syslog: User authentication failure | 5 | 2501 |

Expanded document

[View surrounding documents](#) [View single document](#)

Table **JSON**

| | |
|-------------------------|--|
| ._index | wazuh-alerts-4.x-2022.08.19 |
| ._type | log |
| agent.id | 001 |
| agent.ip | 10.0.2.43 |
| agent.name | server |
| full_log | Aug 19 12:11:58 server vsftpd: message repeated 642 times: [pam_unix(vsftpd:auth): authentication failure; logname= uid=0 euid=0 tty=ftp ruser=msad rhost=::ffff:10.0.2.40 user=msad] |
| id | 1660911331.1128705 |
| input.type | log |
| location | /var/log/auth.log |
| manager.name | wazuh-server |
| predecoder.hostname | server |
| predecoder.program_name | vsftpd |
| predecoder.timestamp | Aug 19 12:11:58 |

≡ **wazuh.** ▼ / Modules / server / Security events ①

| Time | Technique(s) | Tactic(s) | Description | Level | Rule ID |
|-----------------------------|--------------|-------------------|-------------------------|-------|---------|
| Aug 19, 2022 @ 08:10:55.363 | T1110.001 | Credential Access | PAM: User login failed. | 5 | 5503 |

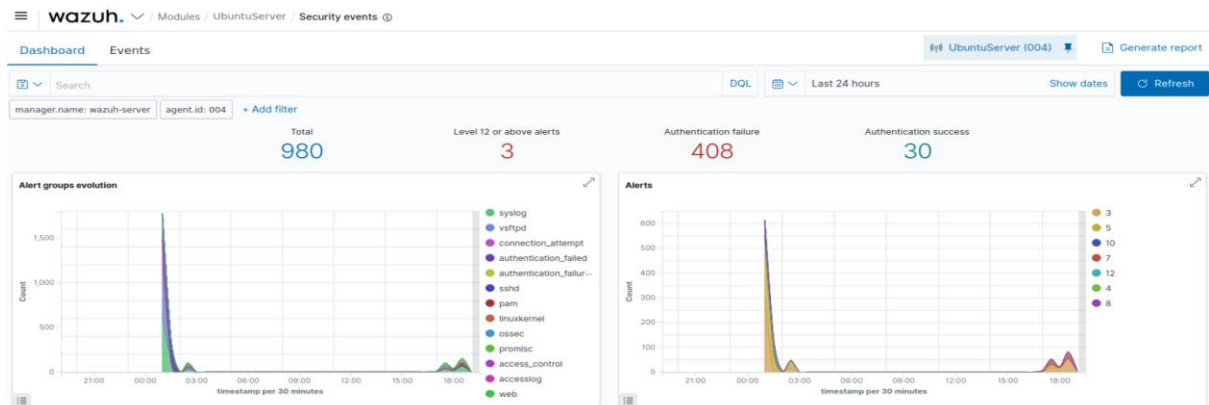
Table **JSON** **Rule**

| | |
|-------------------------|---|
| @timestamp | 2022-08-19T12:10:55.363Z |
| ._id | TUcDtoBxY3dGKqPh_vb |
| agent.id | 001 |
| agent.ip | 10.0.2.43 |
| agent.name | server |
| data.dstuser | msad |
| data.euid | 0 |
| data.srrip | ::ffff:10.0.2.40 |
| data.srcuser | msad |
| data.tty | ftp |
| data.uid | 0 |
| decoder.name | pam |
| full_log | Aug 19 12:10:54 server vsftpd: pam_unix(vsftpd:auth): authentication failure; logname= uid=0 euid=0 tty=ftp ruser=msad rhost=::ffff:10.0.2.40 user=msad |
| id | 1660911055.1121745 |
| input.type | log |
| location | /var/log/auth.log |
| manager.name | wazuh-server |
| predecoder.hostname | server |
| predecoder.program_name | vsftpd |
| predecoder.timestamp | Aug 19 12:10:54 |
| rule.description | PAM: User login failed. |

I was able to get more information by opening each security alerts as shown above.


```
mssd@UbuntuServer:~/Desktop$ nc -l 8000
```

I refreshed the security events in the agents



Here I'm able to detect the rule I've just set up

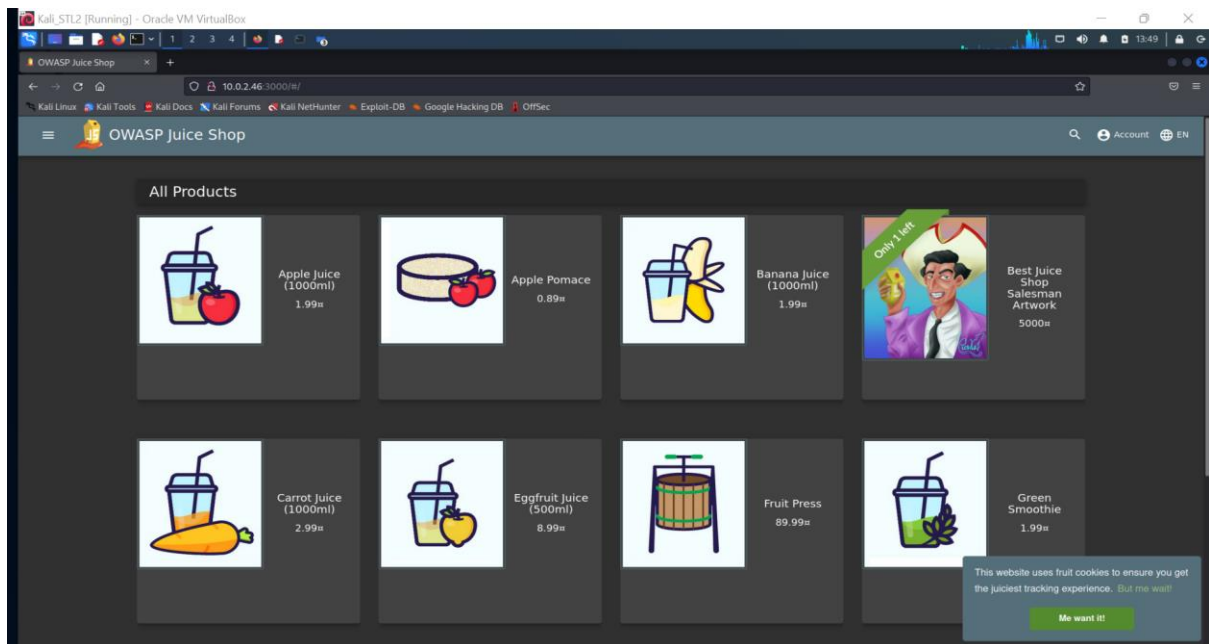
| Security Alerts | | | | | |
|-----------------------------|--------------|--|--|-------|---------|
| Time ↓ | Technique(s) | Tactic(s) | Description | Level | Rule ID |
| Aug 22, 2022 @ 19:10:54.153 | | | Listened ports status (netstat) changed (new port opened or closed). | 7 | 533 |
| Aug 22, 2022 @ 19:08:19.992 | T1078 | Defense Evasion, Persistence, Privilege Escalation, Initial Access | PAM: Login session opened. | 3 | 5501 |
| Aug 22, 2022 @ 19:08:19.992 | T1078 | Defense Evasion, Persistence, Privilege Escalation, Initial Access | PAM: Login session opened. | 3 | 5501 |
| Aug 22, 2022 @ 19:08:19.929 | T1548.003 | Privilege Escalation, Defense Evasion | Successful sudo to ROOT executed. | 3 | 5402 |
| Aug 22, 2022 @ 19:04:53.803 | | | Listened ports status (netstat) changed (new port opened or closed). | 7 | 533 |
| Aug 22, 2022 @ 19:04:53.703 | | | Netcat listening for incoming connections. | 7 | 100051 |
| Aug 22, 2022 @ 19:01:53.544 | | | Netcat listening for incoming connections. | 7 | 100051 |
| Aug 22, 2022 @ 19:01:57.563 | | | PAM: Login session closed. | 3 | 5502 |

We have successfully managed to detect the Netcat Listening command in the Wazuh Server as shown in red above. This was done as an understanding on how a rule-based detection works.

In the next scenario, I will be focusing more on the attacks.

3.4 SQL Injection Attack

I've enabled a docker which runs a OWASP Juice Website in my ubuntu server as shown below



I've added the below code in the local ubuntu server at /var/ossec/etc/ossec.conf

```
<localfile>
  <log_format>apache</log_format>
  <location>/var/log/apache2/access.log</location>
</localfile>
```

This code will monitor the access logs of the Apache server. After adding the code, I restarted the agent.

This is the details of the ruleset

| | | | |
|------------------------|---|-----------------------------------|----------------|
| Information | | | |
| ID | Level | File | Path |
| 31106 | 6 | 0245-web_rules.xml | ruleset/rules |
| Groups | | | |
| attack, web, accesslog | | | |
| Details | | | |
| If_sid | Id | | |
| 31103, 31104, 31105 | pattern: *200 | | |
| Compliance | | | |
| GDPR | TSC | MITRE Techniques | MITRE Tactics |
| IV_35.7.d | CC6.6, CC7.1, CC8.1, CC6.1, CC6.8, CC7.2, CC7.3 | Exploit Public-Facing Application | Initial Access |

Then I performed SQL Injection attack in my attacker's machine as shown below

```
(kali㉿kali)-[~]
└─$ curl -XGET "http://10.0.2.46/?id=SELECT**+FROM+users";
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional
dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <!--
    Modified from the Debian original for Ubuntu
    Last updated: 2022-03-22
    See: https://launchpad.net/bugs/1966004
  -->
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    <title>Apache2 Ubuntu Default Page: It works</title>
    <style type="text/css" media="screen">
* {
  margin: 0px 0px 0px 0px;
  padding: 0px 0px 0px 0px;
}
```

After that, I refreshed my Wazuh server, and I was able to detect SQL Injection as shown below in red

Security Alerts

| Time ↓ | Technique(s) | Tactic(s) | Description | Level | Rule ID |
|-----------------------------|--|----------------|---|-------|---------|
| Aug 22, 2022 @ 19:56:46.277 | T1190 | Initial Access | A web attack returned code 200 (success). | 6 | 31106 |
| Table | JSON | Rule | | | |
| @timestamp | 2022-08-22T11:56:46.277Z | | | | |
| _id | CihpxYIBTSyJk04hs2mM | | | | |
| agent.id | 004 | | | | |
| agent.ip | 10.0.2.46 | | | | |
| agent.name | UbuntuServer | | | | |
| data.id | 200 | | | | |
| data.protocol | GET | | | | |
| data.scrip | 10.0.2.47 | | | | |
| data.url | /?id=SELECT+++FROM+users | | | | |
| decoder.name | web-accesslog | | | | |
| full_log | 10.0.2.47 - - [22/Aug/2022:19:56:45 +0800] "GET /?id=SELECT+++FROM+users HTTP/1.1" 200 10926 "-" "curl/7.82.0" | | | | |
| id | 1661169406.160720 | | | | |
| input.type | log | | | | |
| location | /var/log/apache2/access.log | | | | |
| manager.name | wazuh-server | | | | |
| rule.description | A web attack returned code 200 (success). | | | | |
| rule.firedtimes | 1 | | | | |

We have successfully managed to attack the ubuntu server from the attack machine and was successfully able to detect the SQL Injection. (Web Attack – in this scenario)

3.5 NMAP Probing Attack (NMAP Script Engine Detect)

I have added the below code in my ruleset.

< NMAP_Detection.xml

```
1 <!-- Modify it at your will. -->
2 <!-- # NMAP Detection Rule # -->
3 <!-- ##### -->
4 <group name="NMAP_Detection,">
5   <rule id="100100" level="6">
6     <if_matched_sid>31101</if_matched_sid>
7     <match>Nmap Scripting Engine</match>
8     <description>NMap Scripting Engine Detected</description>
9     <location>/var/log/apache2/access.log</location>
10   </rule>
11 </group>
```

I have also checked if this code as implemented is available in my local agent path. (We did this during SQL injection)

```
<localfile>
  <log_format>apache</log_format>
  <location>/var/log/apache2/access.log</location>
</localfile>
```

I ran an NMAP script in my attacker machine

```
(kali@kali)-[~/Desktop/Project/Attacks]
$ sudo nmap -sV -A -O -Pn -sS 10.0.2.46
Starting Nmap 7.92 ( https://nmap.org ) at 2022-08-22 10:36 EDT
Nmap scan report for 10.0.2.46
Host is up (0.00030s latency).
Not shown: 996 closed tcp ports (reset)
```

I was able to detect the NMAP scripting rule in my Wazuh as shown below.

| Security Alerts | | | | | |
|-------------------------------|--------------|-------------------|--|-------|---------|
| Time ↓ | Technique(s) | Tactic(s) | Description | Level | Rule ID |
| > Aug 22, 2022 @ 22:42:38.087 | T1110.001 | Credential Access | PAM: User login failed. | 5 | 5503 |
| > Aug 22, 2022 @ 22:42:38.082 | T1110.001 | Credential Access | PAM: User login failed. | 5 | 5503 |
| > Aug 22, 2022 @ 22:42:38.077 | T1110.001 | Credential Access | PAM: User login failed. | 5 | 5503 |
| > Aug 22, 2022 @ 22:42:38.041 | | | Web server 501 error code (Not Implemented). | 4 | 31121 |
| > Aug 22, 2022 @ 22:42:38.039 | | | NMap Scripting Engine Detected | 6 | 100100 |
| > Aug 22, 2022 @ 22:42:38.033 | | | NMap Scripting Engine Detected | 6 | 100100 |
| > Aug 22, 2022 @ 22:42:38.029 | | | NMap Scripting Engine Detected | 6 | 100100 |
| > Aug 22, 2022 @ 22:42:38.025 | | | NMap Scripting Engine Detected | 6 | 100100 |

| Aug 22, 2022 @ 22:42:38.039 | | NMap Scripting Engine Detected | 6 | 100100 |
|-----------------------------|------|---|---|--------|
| Table | JSON | Rule | | |
| @timestamp | | 2022-08-22T14:42:38.039Z | | |
| _id | | LFg8x08TSyJkO4ndW9 | | |
| agent.id | | 004 | | |
| agent.ip | | 10.0.2.46 | | |
| agent.name | | UbuntuServer | | |
| data.id | | 404 | | |
| data.protocol | | GET | | |
| data.scrip | | 10.0.2.47 | | |
| data.url | | /evow/about | | |
| decoder.name | | web-accesslog | | |
| full_log | | 10.0.2.47 - - [22/Aug/2022:22:42:36 +0800] "GET /evow/about HTTP/1.1" 404 451 "-" "Mozilla/5.0 (compatible; Nmap Scripting Engine; https://nmap.org/book/nse.html)" | | |
| id | | 1661179358.289308 | | |
| input.type | | log | | |
| location | | /var/log/apache2/access.log | | |
| manager.name | | wazuh-server | | |
| rule.description | | NMap Scripting Engine Detected | | |

We have successfully managed to run an NMAP Engine from the attack machine and was successfully able to detect the NMAP Probing Attack that was set on rule 100100 as shown in the rule code above.

4. Conclusion

As cyber threats are becoming more sophisticated in today's world, real-time monitoring and security analysis are required for quick threat detection and remediation. In this PoC, I performed NMAP Probing attacks and they were all successfully detected by Wazuh. Hence, from this PoC, we can conclude that Wazuh is really a good SIEM platform for threat detection.

I have uploaded the Demo Video in OneDrive. Link to the video

<https://sutdapac-my.sharepoint.com/:v:/g/personal/gowtham_baskar_mymail_sutd_edu_sg/EcsTc0fDnXVLpRpJpwRyGpIBhJfeTrf6G9siuO-ZAaEszQ?e=KStRuB>

5. References

Wazuh

1. <https://wazuh.com/>
2. <https://documentation.wazuh.com/current/index.html>
3. <https://attack.mitre.org/>
4. <https://www.youtube.com/c/HackerSploit> (Wazuh)
5. <https://mangolassi.it/topic/21941/wazuh-when-i-write-the-rule-i-encounter-with-a-problem-nmap-scripting/9> (NMAP Scripting)
6. <https://www.geeksforgeeks.org/introduction-to-wazuh/>

Server Services

1. <https://ubuntu.com/server/docs>

OS's

1. <https://www.kali.org/>
2. <https://ubuntu.com/>

Attacks

1. <https://www.thepythoncode.com/article/brute-force-ssh-servers-using-paramiko-in-python>
2. <https://www.thepythoncode.com/article/brute-force-attack-ftp-servers-using-ftplib-in-python>
3. <https://documentation.wazuh.com/current/proof-of-concept-guide/>