

Configuration Manual

MSc Research Project
MSc in Cybersecurity

Nirav Ratilal Patel
Student ID: x23268859@student.ncirl.ie

School of Computing
National College of Ireland

Supervisor: Joel Aleburu

National College of Ireland
MSc Project Submission Sheet



School of Computing

Nirav Ratilal Patel
Student Name:

x23268859@student.ncirl.ie
Student ID:

MSc in Cybersecurity
Programme: **Year:** 2024-2025

MSc Research Project
Module:

Joel Aleburu
Lecturer:

11/08/2025
Submission Due Date:

Implementing Zero Trust Architecture in Detecting and Mitigating Threats:
Project Title: An APT Focused ZTA Framework.....

2031
Word Count: **Page Count:** 24

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Nirav Ratilal Patel
Signature:

11/08/2025
Date:

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	<input type="checkbox"/>
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	<input type="checkbox"/>
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	<input type="checkbox"/>

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Implementing Zero Trust Architecture in Detecting and Mitigating Threats: An APT Focused ZTA Framework

Nirav Ratilal Patel

X23268859@student.ncirl.ie

1. Introduction

This configuration manual goes through and introduces about the lab designed which comprises of the five VM setup which demonstrates the zero trust principles for the detection and mitigation of the APT attacks through the open source Wazuh SIEM. The attack simulation is performed through Kali Linux virtual machines and demonstrated real world exploits using different tools like Hydra, Metasploit, Wireshark and other tools and the victims are Ubuntu server and the windows server 2022 and the illustration of detection and response process against these APT threats. This configuration manual provides step by step process starting from the network setup configuration and also all the tools installation process along with the attack scenarios.

2. System Specifications

2.1 The following is the hardware configurations of the machine.

- CPU: Intel(R) Core(TM) i7-10750H CPU @ 2.60GHz 2.59 GHz
- Operating System: 64-bit operating system, x64-based processor
- Memory: 16GB RAM
- Drive: 1 TB SSD

2.2 Network design and Lab architecture and tools Installation

Below are the details about the virtual machines and their configurations.

Virtual Machine	Operating System	RAM	Storage	IP address
Ubuntu for SIEM	Ubuntu 22.04	8 GB	100 GB	192.168.1.86
Kali Linux for Attack machine	Kali Linux	4 GB	40 GB	192.168.1.76

Ubuntu for ZTA controller	Ubuntu	4 GB	60 GB	192.168.1.81
Windows server for victim machine	Windows server 2022	4 GB	60 GB	192.168.1.72
Ubuntu Server for Victim machine	Ubuntu server 22.04	2 GB	40 GB	192.168.1.77

2.3 Tools Installation on different virtual machines

1. **Ubuntu Desktop: Installation of Wazuh SIEM 4.12.0.** It is an opensource tool for monitoring of all the events has threat detection capabilities which is used mainly for the detection of abnormal behaviours in the infrastructure in different environments. It has different features such as Active response, File integrity monitoring, Vulnerability detection, Log analysis and corelation and provides real time alerts and Incident response. IT consists of three major components which are Wazuh Manager, Wazuh Indexer, Wazuh, Dashboard.

The command for Installing Wazuh is “**curl -sO <https://packages.wazuh.com/4.12/wazuh-install.sh> && sudo bash ./wazuh-install.sh -a**”

- **Wazuh Manager:** It is basically the central component of the of the whole system which receives and processes the data and also analyses the data from all the agents which are deployed. It performs real time log processing and many other different functions.

```

wazuh-manager.service - Wazuh manager
Loaded: loaded (/usr/lib/systemd/system/wazuh-manager.service; enabled; preset: enabled)
Active: active (running) since Sat 2025-07-19 09:23:47 IST; 39s ago
Process: 1178 ExecStart=/usr/bin/env /var/ossec/bin/wazuh-control start (code=exited, status=0/SUCCESS)
Tasks: 177 (limit: 9436)
Memory: 823.7M (peak: 823.9M)
CPU: 39.465s
CGroup: /system.slice/wazuh-manager.service
├─2027 /var/ossec/framework/python/bin/python3 /var/ossec/api/scripts/wazuh_apid.py
├─2028 /var/ossec/framework/python/bin/python3 /var/ossec/api/scripts/wazuh_apid.py
├─2029 /var/ossec/framework/python/bin/python3 /var/ossec/api/scripts/wazuh_apid.py
├─2032 /var/ossec/framework/python/bin/python3 /var/ossec/api/scripts/wazuh_apid.py
├─2035 /var/ossec/framework/python/bin/python3 /var/ossec/api/scripts/wazuh_apid.py
├─2084 /var/ossec/bin/wazuh-authd
├─2103 /var/ossec/bin/wazuh-db
├─2130 /var/ossec/bin/wazuh-execd
├─2145 /var/ossec/bin/wazuh-analysisd
├─2180 /var/ossec/bin/wazuh-syscheckd
├─2364 /var/ossec/bin/wazuh-remoted
├─2395 /var/ossec/bin/wazuh-logcollector
├─2549 /var/ossec/bin/wazuh-monitord
└─2565 /var/ossec/bin/wazuh-modulesd

Jul 19 09:23:43 ubuntu-siem-VirtualBox env[1178]: wazuh-logcollector: Process 2192 not used by Wazuh, removing...
Jul 19 09:23:44 ubuntu-siem-VirtualBox env[1178]: Started wazuh-logcollector...
Jul 19 09:23:44 ubuntu-siem-VirtualBox env[1178]: wazuh-monitord: Process 2215 not used by Wazuh, removing...
Jul 19 09:23:44 ubuntu-siem-VirtualBox env[1178]: Started wazuh-monitord...
Jul 19 09:23:44 ubuntu-siem-VirtualBox env[1178]: wazuh-modulesd: Process 2255 not used by Wazuh, removing...
Jul 19 09:23:44 ubuntu-siem-VirtualBox env[2562]: 2025/07/19 09:23:44 wazuh-modulesd:router: INFO: Loaded router mo
Jul 19 09:23:44 ubuntu-siem-VirtualBox env[2562]: 2025/07/19 09:23:44 wazuh-modulesd:content_manager: INFO: Loaded

```

Figure 1: Wazuh manager

- **Wazuh Indexer:** It is based on the functions like handling the data storage and indexing and other search capabilities which can help for faster data retrieval and other data retention capabilities.

```

ubuntu-siem@ubuntu-siem-VirtualBox:~$ sudo systemctl status wazuh-indexer
● wazuh-indexer.service - wazuh-indexer
   Loaded: loaded (/usr/lib/systemd/system/wazuh-indexer.service; enabled; preset: enabled)
   Active: active (running) since Sat 2025-07-19 09:24:08 IST; 2min 28s ago
     Docs: https://documentation.wazuh.com
    Main PID: 1177 (java)
      Tasks: 99 (limit: 9436)
    Memory: 1.7G (peak: 1.7G)
       CPU: 1min 50.166s
    CGroup: /system.slice/wazuh-indexer.service
           └─1177 /usr/share/wazuh-indexer/jdk/bin/java -Xshare:auto -Dopensearch.networkaddress.cache.ttl=60 -Dopensearch

Jul 19 09:23:32 ubuntu-siem-VirtualBox systemd-entrypoint[1177]: WARNING: System::setSecurityManager has been called by
Jul 19 09:23:32 ubuntu-siem-VirtualBox systemd-entrypoint[1177]: WARNING: Please consider reporting this to the maintai
Jul 19 09:23:32 ubuntu-siem-VirtualBox systemd-entrypoint[1177]: WARNING: System::setSecurityManager will be removed in
Jul 19 09:23:34 ubuntu-siem-VirtualBox systemd-entrypoint[1177]: Jul 19, 2025 9:23:34 AM sun.util.locale.provider.Local
Jul 19 09:23:34 ubuntu-siem-VirtualBox systemd-entrypoint[1177]: WARNING: COMPAT locale provider will be removed in a f
Jul 19 09:23:35 ubuntu-siem-VirtualBox systemd-entrypoint[1177]: WARNING: A terminally deprecated method in java.lang.S
Jul 19 09:23:35 ubuntu-siem-VirtualBox systemd-entrypoint[1177]: WARNING: System::setSecurityManager has been called by
Jul 19 09:23:35 ubuntu-siem-VirtualBox systemd-entrypoint[1177]: WARNING: Please consider reporting this to the maintai
Jul 19 09:23:35 ubuntu-siem-VirtualBox systemd-entrypoint[1177]: WARNING: System::setSecurityManager will be removed in
Jul 19 09:24:08 ubuntu-siem-VirtualBox systemd[1]: Started wazuh-indexer.service - wazuh-indexer.
lines 1-21/21 (END)

```

Figure 2: Wazuh Indexer

- **Wazuh Dashboard:** It is wazuh's web interface which is used for the real time monitoring and management. It is mainly used for the managing the alerts and proactive threat hunting and management of different users and its access and permission.

```

ubuntu-siem@ubuntu-siem-VirtualBox:~$ sudo systemctl status wazuh-dashboard
● wazuh-dashboard.service - wazuh-dashboard
   Loaded: loaded (/etc/systemd/system/wazuh-dashboard.service; enabled; preset: enabled)
   Active: active (running) since Sat 2025-07-19 09:23:22 IST; 4min 0s ago
    Main PID: 930 (node)
      Tasks: 11 (limit: 9436)
    Memory: 321.5M (peak: 448.5M)
       CPU: 17.653s
    CGroup: /system.slice/wazuh-dashboard.service
           └─930 /usr/share/wazuh-dashboard/node/bin/node --no-warnings --max-http-header-size=65536 --unhandled-reje

Jul 19 09:24:45 ubuntu-siem-VirtualBox opensearch-dashboards[930]: {"type": "response", "@timestamp": "2025-07-19T08:24:45"}
Jul 19 09:24:45 ubuntu-siem-VirtualBox opensearch-dashboards[930]: {"type": "response", "@timestamp": "2025-07-19T08:24:45"}
Jul 19 09:24:46 ubuntu-siem-VirtualBox opensearch-dashboards[930]: {"type": "response", "@timestamp": "2025-07-19T08:24:45"}
Jul 19 09:24:46 ubuntu-siem-VirtualBox opensearch-dashboards[930]: {"type": "response", "@timestamp": "2025-07-19T08:24:45"}
Jul 19 09:24:47 ubuntu-siem-VirtualBox opensearch-dashboards[930]: {"type": "response", "@timestamp": "2025-07-19T08:24:45"}
Jul 19 09:24:47 ubuntu-siem-VirtualBox opensearch-dashboards[930]: {"type": "response", "@timestamp": "2025-07-19T08:24:45"}
Jul 19 09:24:48 ubuntu-siem-VirtualBox opensearch-dashboards[930]: {"type": "response", "@timestamp": "2025-07-19T08:24:48"}
Jul 19 09:24:50 ubuntu-siem-VirtualBox opensearch-dashboards[930]: {"type": "response", "@timestamp": "2025-07-19T08:24:48"}
Jul 19 09:24:51 ubuntu-siem-VirtualBox opensearch-dashboards[930]: {"type": "response", "@timestamp": "2025-07-19T08:24:51"}
Jul 19 09:24:51 ubuntu-siem-VirtualBox opensearch-dashboards[930]: {"type": "response", "@timestamp": "2025-07-19T08:24:51"}
lines 1-20/20 (END)

```

Figure 3: Wazuh Dashboard

Wazuh Manager Configuration: For leveraging the full event logging and also checking the location of the logs, we need to make some changes in the configuration file which is “/var/ossec/etc/ossec.conf”. Perform the below steps and add the <global> section in the conf file.

- Go to `sudo nano /var/ossec/etc/ossec.conf`
- Add the below steps in the file and save it.
- **<global>**
 <logall>yes</logall>
 <logall_json>yes</logall_json>
</global>



```
GNU nano 7.2 /var/ossec/etc/ossec.conf
<!--
Wazuh - Manager - Default configuration for ubuntu 24.04
More info at: https://documentation.wazuh.com
Mailing list: https://groups.google.com/forum/#!forum/wazuh
-->

<ossec_config>
  <global>
    <jsonout_output>yes</jsonout_output>
    <alerts_log>yes</alerts_log>
    <logall>yes</logall>
    <logall_json>yes</logall_json>
    <email_notification>no</email_notification>
    <smtp_server>smtp.example.wazuh.com</smtp_server>
    <email_from>wazuh@example.wazuh.com</email_from>
    <email_to>recipient@example.wazuh.com</email_to>
    <email_maxperhour>12</email_maxperhour>
    <email_log_source>alerts.log</email_log_source>
    <agents_disconnection_time>10m</agents_disconnection_time>
    <agents_disconnection_alert_time>0</agents_disconnection_alert_time>
    <update_check>yes</update_check>
  </global>

  <alerts>
    <log_alert_level>3</log_alert_level>
    <email_alert_level>12</email_alert_level>
  </alerts>
```

Figure 4: ossec.conf file

For log parsing and processing, we need to make some changes in the filebeat modules due to which we can enable all the alerts and archives in wazuh logs. It is the vital setting for integrating and visualizing the alerts in Wazuh’s console.

Navigate to the filebeat.yml file by “**sudo nano /etc/filebeat/filebeat.yml**” and make the following changes. We need to make sure that the alerts and archives both are “true”.

- filebeat.modules:
 - module: wazuh
 - alerts:
 - enabled: true**
 - archives:
 - enabled: true**

```
root@ubuntu-siem-VirtualBox: /home/ubuntu-siem
GNU nano 7.2 /etc/filebeat/filebeat.yml
# Wazuh - Filebeat configuration file
output.elasticsearch.hosts:
  - 127.0.0.1:9200
#   - <elasticsearch_ip_node_2>:9200
#   - <elasticsearch_ip_node_3>:9200

output.elasticsearch:
  protocol: https
  username: ${username}
  password: ${password}
  ssl.certificate_authorities:
    - /etc/filebeat/certs/root-ca.pem
  ssl.certificate: "/etc/filebeat/certs/wazuh-server.pem"
  ssl.key: "/etc/filebeat/certs/wazuh-server-key.pem"
setup.template.json.enabled: true
setup.template.json.path: '/etc/filebeat/wazuh-template.json'
setup.template.json.name: 'wazuh'
setup.ilm.overwrite: true
setup.ilm.enabled: false

filebeat.modules:
  - module: wazuh
    alerts:
      enabled: true
    archives:
      enabled: true

logging.level: info
```

Figure 5: Filebeat.yml file

Log Ingestion in Wazuh: Creation of the archive index pattern in wazuh dashboard for recognizing all the searches in the logs and selecting the timestamps which will enable all the time fields.

- Go to the Wazuh dashboard, and click on the menu Icon and select the Dashboard management and go to index patterns.
- Click on the “create index pattern” and add wazuh-archives-* as the index pattern.
- Next, choose timestamp as the time field.
- Click on the create index pattern to save it.

The screenshot shows the OpenSearch Dashboards interface for the 'wazuh-archives-*' index pattern. The left sidebar contains navigation options like 'Index patterns', 'Data sources', 'Saved objects', and 'Advanced settings'. The main content area shows the index name and a table of fields. A search bar and a dropdown for 'All field types' are also visible.

Name	Type	Format	Searchable	Aggregatable	Excluded
@timestamp	date		●	●	✎
@version	string		●		✎
GeoLocation.area_code	number		●	●	✎
GeoLocation.city_name	string		●	●	✎
GeoLocation.continent_code	string		●		✎
GeoLocation.coordinates	number		●	●	✎
GeoLocation.country_code2	string		●		✎

Figure 6: Log ingestion

For viewing the logs in the Dashboard,

- Go to the Discover section and from the index pattern dropdown, choose the **wazuh-archives-*** option and see the all the raw logs which are being sent by the agent.

2. Ubuntu for ZTA Controller:

Installation of spire server which will work as the foundation of the zero trust architecture in out environments. Basically it comprises of the two main components, one is Spire server and other is spire agent. The installation steps are mentioned below. [4]

Initially, we will update all the dependencies and required packages.

- **sudo apt update && sudo apt upgrade -y**
- **sudo apt install curl wget git unzip build-essential -y**

Now, we will create a separate directory for spire.

- **mkdir -p ~/zta-lab/spire/{bin,conf,data/{server,agent},logs}**
- **cd ~/zta-lab/spire**

```
zta-controller@zta-controller-VirtualBox: ~/zta-lab/spire
zta-controller@zta-controller-VirtualBox:~/zta-lab/spire$ cd
zta-controller@zta-controller-VirtualBox:~$ ls
cilium          Downloads  snap          wazuh-install.sh          zta-lab
cilium-linux-amd64.tar.gz  Music      Templates    ziti-console.zip
Desktop         Pictures   Videos      ziti-console.zip.1
Documents       Public    wazuh-install-files.tar  ziti-linux-amd64-1.5.4.tar.gz
zta-controller@zta-controller-VirtualBox:~$ cd zta-lab/
zta-controller@zta-controller-VirtualBox:~/zta-lab$ ls
cilium-policies  openziti  spire
zta-controller@zta-controller-VirtualBox:~/zta-lab$ cd spire
zta-controller@zta-controller-VirtualBox:~/zta-lab/spire$ ls
bin  conf  data  LICENSE  logs  README.md
zta-controller@zta-controller-VirtualBox:~/zta-lab/spire$
```

Figure 7: Create a directory for spire

Now, we will download the latest release version for spire and install it. The version which is the latest is version 1.12.3. The commands for the installation are given below.

- `wget https://github.com/spiffe/spire/releases/download/v1.12.3/spire-1.12.3-linux-amd64-musl.tar.gz`

Now, we will verify and extract the binaries.

- `tar -xzf spire-1.12.3-linux-amd64-musl.tar.gz`
- `mv spire-1.12.3/* .`
- `rm -rf spire-1.12.3 spire-1.12.3-linux-amd64-musl.tar.gz`

Configuration of Spire Server:

We will do the configuration in the nano text editor and below is the configuration of that.

- `nano ~/zta-lab/spire/conf/server/server.conf`

```
zta-controller@zta-controller-VirtualBox: ~/zta-lab/spire/conf/server
zta-controller@zta-controller-VirtualBox:~/zta-lab/spire/conf/server$ ls
server.conf
zta-controller@zta-controller-VirtualBox:~/zta-lab/spire/conf/server$ nano server.conf
zta-controller@zta-controller-VirtualBox:~/zta-lab/spire/conf/server$
```

Figure 8: Make server.conf file

Now add all the configuration in the “server.conf” file.

```
GNU nano 7.2 server.conf
server {
  bind_address = "0.0.0.0"
  bind_port    = "8081"
  trust_domain = "zta-lab.local"
  data_dir     = "./data/server"
  log_level    = "INFO"
  log_file     = "./logs/spire-server.log"

  ca_subject = {
    country      = ["US"],
    organization = ["ZTA Lab"],
    common_name  = "ZTA Lab CA",
  }
  ca_ttl        = "168h"
  default_x509_svid_ttl = "1h"
  default_jwt_svid_ttl  = "5m"
}

plugins {
  DataStore "sql" {
    plugin_data {
      database_type     = "sqlite3"
      connection_string = "./data/server/datastore.sqlite3"
    }
  }
  KeyManager "disk" {
    plugin_data {
      keys_path = "./data/server/keys.json"
    }
  }
}
```

Figure 9: Spire server configuration

Configuration of Spire Agent: For this configuration navigate to the same file location and make a new file called “agent.conf”. The location should be, “**nano ~/.zta-lab/spire/conf/agent/agent.conf**”.

```
GNU nano 7.2 agent.conf
agent {
  data_dir       = "./data/agent"
  log_level      = "INFO"
  log_file       = "./logs/spire-agent.log"
  server_address = "127.0.0.1"
  server_port    = "8081"
  socket_path    = "./data/agent/spire-agent.sock"
  trust_bundle_path = "./data/agent/bundle.crt"
  trust_domain   = "zta-lab.local"
  spiffe_id      = "spiffe://zta-lab.local/agent/zta-controller"
}

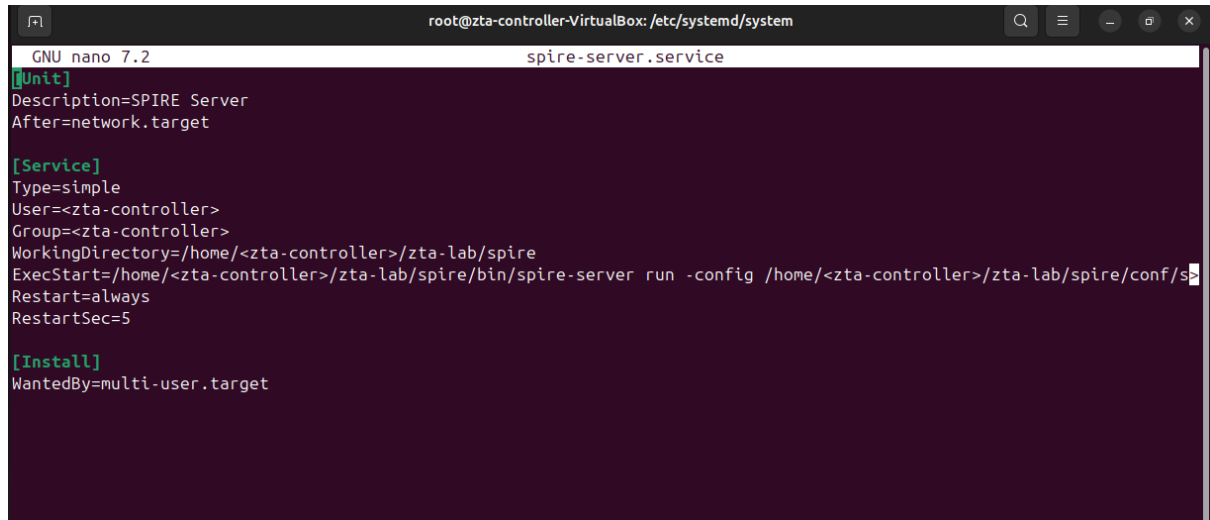
plugins {
  KeyManager "disk" {
    plugin_data { directory = "./data/agent" }
  }
  NodeAttestor "join_token" { plugin_data {} }
  WorkloadAttestor "unix" {
    plugin_data { discover_workload_path = true }
  }
  WorkloadAttestor "docker" {
    plugin_data { docker_socket_path = "/var/run/docker.sock" }
  }
}

health_checks {
  listener_enabled = true
  bind_address     = "127.0.0.1"
}
```

Figure 10: Spire agent

Now, we are going to create a systemd service for both spire service as well as spire agent. Below are the steps for spire service and spire agent.

Spire Server Service: “sudo nano /etc/systemd/system/spire-server.service”



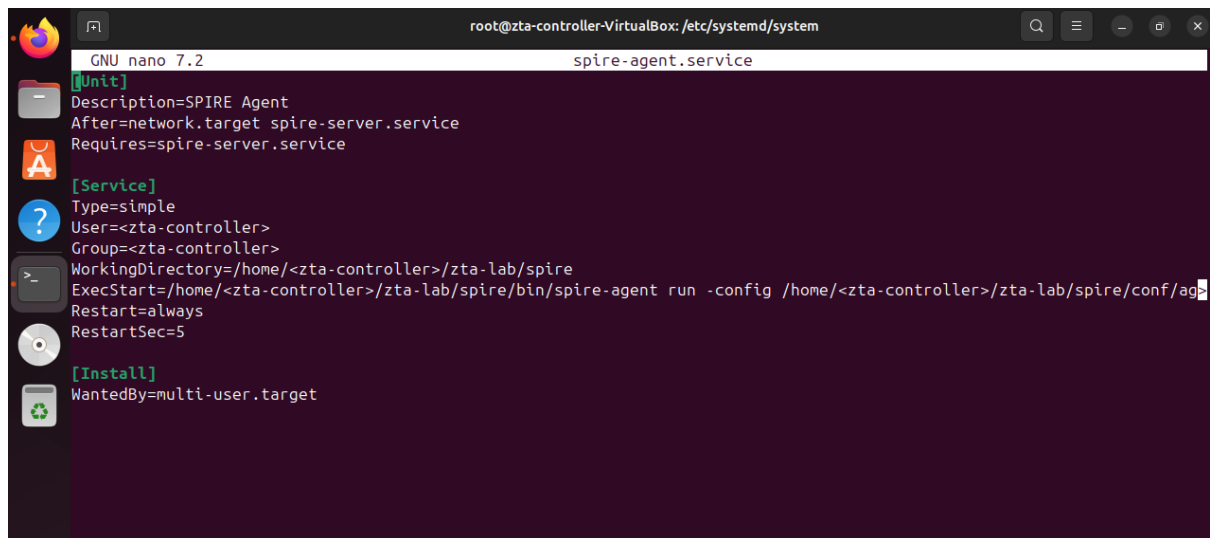
```
root@zta-controller-VirtualBox: /etc/systemd/system
GNU nano 7.2 spire-server.service
[Unit]
Description=SPIRE Server
After=network.target

[Service]
Type=simple
User=<zta-controller>
Group=<zta-controller>
WorkingDirectory=/home/<zta-controller>/zta-lab/spire
ExecStart=/home/<zta-controller>/zta-lab/spire/bin/spire-server run -config /home/<zta-controller>/zta-lab/spire/conf/s
Restart=always
RestartSec=5

[Install]
WantedBy=multi-user.target
```

Figure 11: Spire server service

Spire Agent Service: “sudo nano /etc/systemd/system/spire-agent.service”



```
root@zta-controller-VirtualBox: /etc/systemd/system
GNU nano 7.2 spire-agent.service
[Unit]
Description=SPIRE Agent
After=network.target spire-server.service
Requires=spire-server.service

[Service]
Type=simple
User=<zta-controller>
Group=<zta-controller>
WorkingDirectory=/home/<zta-controller>/zta-lab/spire
ExecStart=/home/<zta-controller>/zta-lab/spire/bin/spire-agent run -config /home/<zta-controller>/zta-lab/spire/conf/ag
Restart=always
RestartSec=5

[Install]
WantedBy=multi-user.target
```

Figure 12: Spire Agent Service

The spire server and spire agent is set up now. We will enable the services and check the status of the services.

- **sudo systemctl daemon-reload**
- **sudo systemctl enable spire-server spire-agent**
- **sudo systemctl start spire-server**
- **sleep 5**
- **sudo systemctl start spire-agent**

Environment Configuration and adding variables to the user profile

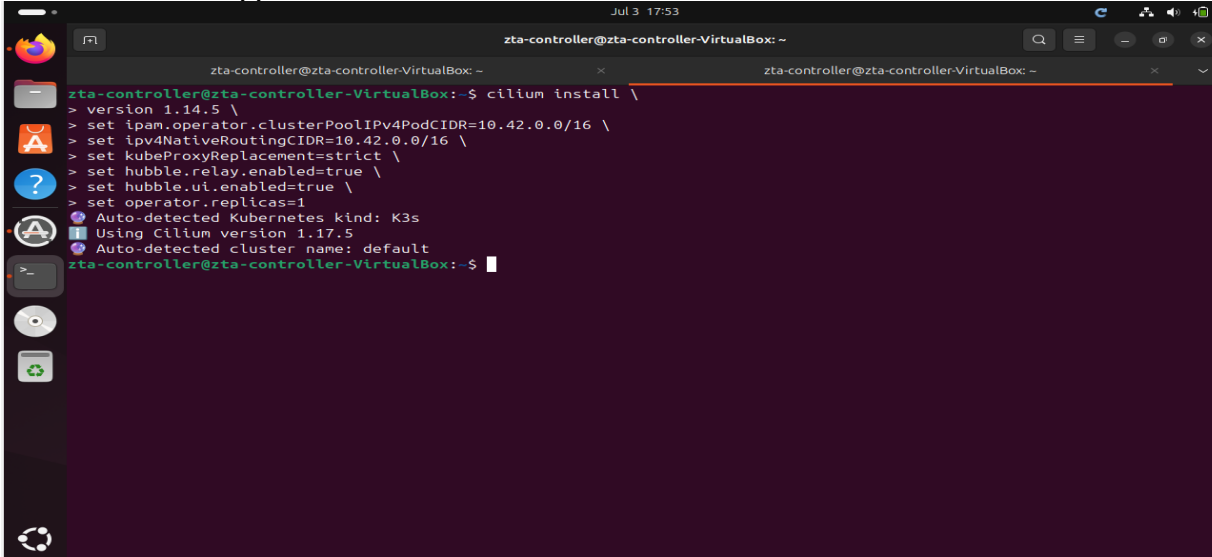
- nano ~/.bashrc
- Append the following command in the end of the file:
- **export SPIRE_SERVER_ADDRESS=127.0.0.1:8081**
- **export SPIFFE_ENDPOINT_SOCKET=unix:///home/<your_user>/zta-lab/spire/data/agent/spire-agent.sock**
- Now, Save and reload it using “**source ~/.bashrc**”

Installation of Cilium

Now, we will move forward with the cilium installation, we will download Cilium from github and below are the commands for it. Also, the extraction and installation commands are provided below.

- **curl -L --remote-name-all <https://github.com/cilium/cilium-cli/releases/latest/download/cilium-linux-amd64.tar.gz>**
- **tar -xzf cilium-linux-amd64.tar.gz sudo mv cilium /usr/local/bin/**
- **cilium version --client**

Also, the deployment of the cilium is done with zero trust configuration below are the commands and snippets for it.



```
zta-controller@zta-controller-VirtualBox: ~  
$ cilium install \  
> version 1.14.5 \  
> set ipam.operator.clusterPoolIPv4PodCIDR=10.42.0.0/16 \  
> set ipv4NativeRoutingCIDR=10.42.0.0/16 \  
> set kubeProxyReplacement=strict \  
> set hubble.relay.enabled=true \  
> set hubble.ut.enabled=true \  
> set operator.replicas=1  
Auto-detected Kubernetes kind: K3s  
Using Cilium version 1.17.5  
Auto-detected cluster name: default  
zta-controller@zta-controller-VirtualBox:~$
```

Figure 13: Cilium installation

For checking the cilium status, use the command “**cilium version --wait**”

```

Jul 3 17:54
zta-controller@zta-controller-VirtualBox: ~
zta-controller@zta-controller-VirtualBox: ~
Using Cilium version 1.17.5
Auto-detected cluster name: default
zta-controller@zta-controller-VirtualBox:~$ cilium status --wait
Cilium:          1 errors, 1 warnings
Operator:        OK
Envoy DaemonSet: 1 errors
Hubble Relay:    disabled
ClusterMesh:     disabled

DaemonSet        cilium          Desired: 1, Unavailable: 1/1
DaemonSet        cilium-envoy   Desired: 1, Unavailable: 1/1
Deployment        cilium-operator Desired: 1, Ready: 1/1, Available: 1/1
Containers:      cilium          Pending: 1
                  cilium-envoy   Running: 1
                  cilium-operator Running: 1
                  clustermesh-apiserver
                  hubble-relay

Cluster Pods:    0/5 managed by Cilium
Helm chart version: 1.17.5
Image versions   cilium          quay.io/cilium/cilium:v1.17.5@sha256:baf8541723ee0b72d6c489c741c81a6fdc5228940d66cb76ef5ea2ce3c639ea6: 1
                  cilium-envoy   quay.io/cilium/cilium-envoy:v1.32.6-1749271279-0864395884b263913eac200ee2048fd985f8e626@sha256:9f69e290a7ea3d4edf9192acd81694089af048ae0d8a67fb63bd62dc1d72203e: 1
                  cilium-operator quay.io/cilium/operator-generic:v1.17.5@sha256:f954c97eeb1b47ed67d08cc8fb4108fb829f869373cbb3e698a7f8ef1085b09e: 1
Errors:          cilium          cilium          1 pods of DaemonSet cilium are not ready
                  cilium-envoy cilium-envoy    1 pods of DaemonSet cilium-envoy are not ready
Warnings:       cilium          cilium-2qjqz    pod is pending

```

Figure 14: Cilium status

Configuration of Zero Trust Network Policies:

- `mkdir -p ~/zta-lab/cilium-policies`
- `nano default-deny-policy.yaml`

```

zta-controller@zta-controller-VirtualBox: ~/zta-lab/cilium-policies
zta-controller@zta-controller-VirtualBox:~/zta-lab/cilium-policies$ ls
apt-mitigation-policy.yaml default-deny-policy.yaml spiffe-network-policy.yaml zta-management-policy.yaml
zta-controller@zta-controller-VirtualBox:~/zta-lab/cilium-policies$
zta-controller@zta-controller-VirtualBox:~/zta-lab/cilium-policies$

```

Figure 15: Policies

```

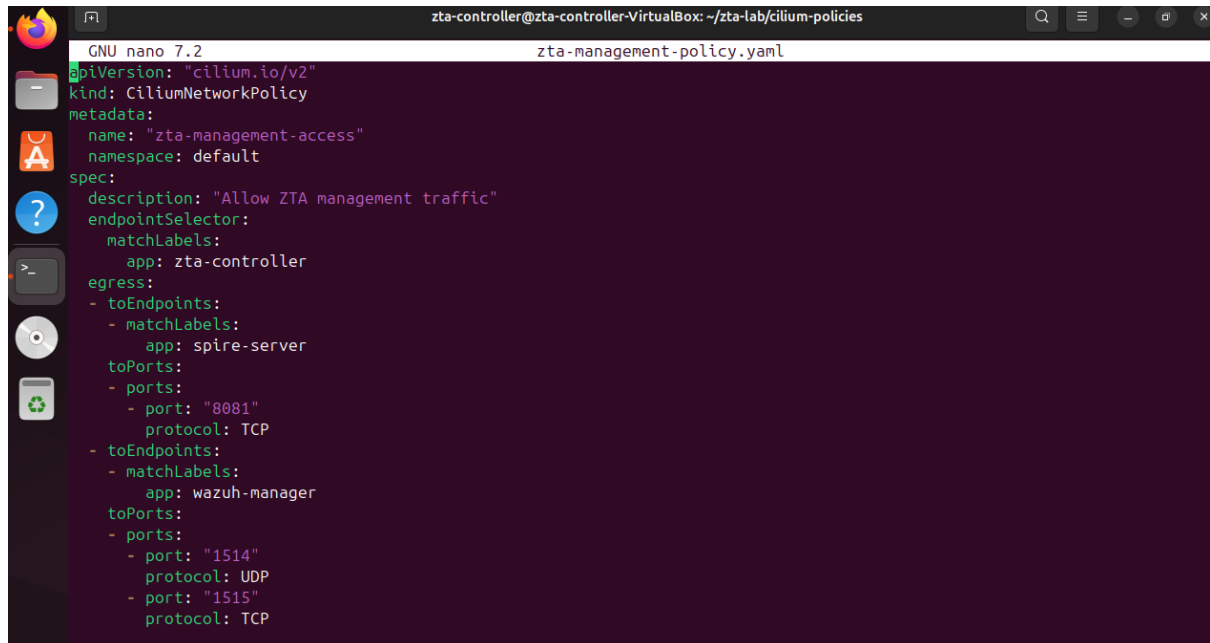
GNU nano 7.2 default-deny-policy.yaml
apiVersion: "cilium.io/v2"
kind: CiliumNetworkPolicy
metadata:
  name: "default-deny-all"
  namespace: default
spec:
  description: "Default deny all traffic - Zero Trust baseline"
  endpointSelector: {}
  egress: []
  ingress: []

```

Figure 16: Default -deny policy

Now, we can create **ZTA management policy**:

- Go to terminal and use the command “nano zta-management-policy.yaml” and edit the file.
- Apply the changes and save it. “kubectl apply -f zta-management-policy.yaml”



```
GNU nano 7.2 zta-management-policy.yaml
apiVersion: "cilium.io/v2"
kind: CiliumNetworkPolicy
metadata:
  name: "zta-management-access"
  namespace: default
spec:
  description: "Allow ZTA management traffic"
  endpointSelector:
    matchLabels:
      app: zta-controller
  egress:
  - toEndpoints:
    - matchLabels:
        app: spire-server
      toPorts:
      - ports:
        - port: "8081"
          protocol: TCP
  - toEndpoints:
    - matchLabels:
        app: wazuh-manager
      toPorts:
      - ports:
        - port: "1514"
          protocol: UDP
        - port: "1515"
          protocol: TCP
```

Figure 17: ZTA-management policy

Cilium policy for APT Mitigation:

- Go to the terminal and create policy using the nano editor. “nano apt-mitigation-policy.yaml”
- Apply the policy and save it. “kubectl apply -f apt-mitigation-policy.yaml”

```
zta-controller@zta-controller-VirtualBox: ~/zta-lab/cilium-policies
GNU nano 7.2 apt-mitigation-policy.yaml
apiVersion: "cilium.io/v2"
kind: CiliumNetworkPolicy
metadata:
  name: "apt-lateral-movement-prevention"
  namespace: default
spec:
  description: "Prevent APT lateral movement between network segments"
  endpointSelector:
    matchLabels:
      zone: target-network
  egress:
  - toEndpoints:
    - matchLabels:
        zone: management-network
    toPorts:
    - ports:
      - port: "443"
        protocol: TCP
  ingress:
  - fromEndpoints:
    - matchLabels:
        zone: management-network
    toPorts:
    - ports:
      - port: "22"
        protocol: TCP
```

Figure 18: Cilium policy

Now, we move forward with creating system service for monitoring which basically creates a service for monitoring cilium,

- Go to the terminal and type “sudo nano /etc/systemd/system/cilium-monitor.service” and save the file.



```
zta-controller@zta-controller-VirtualBox: /etc/systemd/system
GNU nano 7.2 cilium-monitor.service *
[Unit]
Description=Cilium ZTA Network Monitor
After=k3s.service
Requires=k3s.service

[Service]
Type=simple
User=root
ExecStart=/usr/local/bin/cilium monitor --type drop
Restart=always
RestartSec=10

[Install]
WantedBy=multi-user.target
```

Figure 19: cilium service

3. Installation of Tools in Kali Linux

- We will install Metasploit and Hydra in kali linux for the attack scenario purpose:

Below is the installation of the tools.

- Sudo apt install Metasploit-framework
- systemctl enable --now postgresql
- systemctl status postgresql
- msfdb init
- msfconsole

```

(root@nirav)-[/home/nirav]
# msfconsole
Metasploit tip: To save all commands executed since start up to a file, use the
makerc command

##### ;:
._____.;
" 000000'.'000000 000000'.'000000 ".
'-0000000000000000 0000000000000000 0;
.0000000000000000 0000000000000000 .'
"--'.0000 -'.0 0 '-.'--"
".0' ; 0 0 `.' ;'
|0000 000 0
' 000 00 000 ,
'.0000 000 .
',000 0
( 3 C ) /|__ /Metasploit! \
;0'._*_-' " \|- \
'(. , . . . . "/

=[ metasploit v6.4.69-dev ]
+ -- --[ 2529 exploits - 1302 auxiliary - 431 post ]
+ -- --[ 1669 payloads - 49 encoders - 13 nops ]
+ -- --[ 9 evasion ]

Metasploit Documentation: https://docs.metasploit.com/

```

Figure 20: Metasploit

```

HYDRA(1)                                     General Commands Manual                                     HYDRA(1)
NAME
  hydra - a very fast network logon cracker which supports many different services
SYNOPSIS
  hydra
  [[-l LOGIN|-L FILE] [-p PASS|-P FILE|-x OPT -y]] | [-C FILE]]
  [-e nsr] [-u] [-f=F] [-M FILE] [-o FILE] [-b FORMAT]
  [-t TASKS] [-T TASKS] [-w TIME] [-W TIME] [-m OPTIONS] [-s PORT]
  [-c TIME] [-S] [-O] [-4|6] [-I] [-vV] [-d]
  server service [OPTIONS]
DESCRIPTION
  Hydra is a parallelized login cracker which supports numerous protocols to attack. New modules are easy to add, beside that, it is flexible and very fast.

  This tool gives researchers and security consultants the possibility to show how easy it would be to gain unauthorized access from remote to a system.

  Currently this tool supports:
  adam6500 afp asterisk cisco cisco-enable cvs firebird ftp ftps http[s]-{head|get|post} http[s]-{get|post}-form http-proxy http-proxy-urlenum icq imap[s] irc ldap2[s] ldap3[-{cram|digest}|md5][s] mssql mysql(v4) mysql5 ncp nntp oracle oracle-listener oracle-sid pcanywhere pcnfs pop3[s] postgres rdp radmin2 redis rexec rlogin rpcac rsh rtsp s7-300 sapr3 sip smb smtp[s] smtp-enum snmp socks5 ssh sshkey svn teamspeak telnet[s] vmauthd vnc xmpp

  For most protocols SSL is supported (e.g. https-get, ftp-ssl, etc.). If not all necessary libraries are found during compile time, your available services will be less. Type "hydra" to see what is available.

```

Figure 21: Hydra

4. Installation of Wazuh Agent in Ubuntu server (Victim Machine)

Below is the commands for installation of agent.

- Initially, Install the GPG Key by using the following command
Command : “curl -s https://packages.wazuh.com/key/GPG-KEY-WAZUH | gpg --no-default-keyring --keyring gnupg-ring:/usr/share/keyrings/wazuh.gpg --import && chmod 644 /usr/share/keyrings/wazuh.gpg”.

- Add the repository by using the following command
Command: `echo "deb [signed-by=/usr/share/keyrings/wazuh.gpg] https://packages.wazuh.com/4.x/apt/ stable main" | tee -a /etc/apt/sources.list.d/wazuh.list.`
- Update the current package:
Command: `apt-get update`

Below is the attack simulations of all the attack scenarios.

1. **SSH Brute Force Attack:** Detection of the SSH bruteforce attacks when there is 8 or more than 8 failed login from the same IP address in the time span of 120 seconds.

timestamp	agent.name	rule.description	rule.level	rule.id
Jul 30, 2025 @ 13:00:56.946	nirav	sshd: brute force trying to get access to the system. Authenticatio...	10	5763
Jul 30, 2025 @ 12:58:58.755	nirav	sshd: brute force trying to get access to the system. Authenticatio...	10	5763
Jul 30, 2025 @ 11:29:57.718	nirav	sshd: brute force trying to get access to the system. Authenticatio...	10	5763
Jul 30, 2025 @ 11:28:53.680	nirav	sshd: brute force trying to get access to the system. Authenticatio...	10	5763
Jul 30, 2025 @ 11:27:39.547	nirav	sshd: brute force trying to get access to the system. Authenticatio...	10	5763
Jul 30, 2025 @ 11:03:45.460	nirav	sshd: brute force trying to get access to the system. Authenticatio...	10	5763
Jul 30, 2025 @ 11:02:01.340	nirav	sshd: brute force trying to get access to the system. Authenticatio...	10	5763
Jul 30, 2025 @ 11:00:27.195	nirav	sshd: brute force trying to get access to the system. Authenticatio...	10	5763
Jul 30, 2025 @ 10:58:05.030	nirav	sshd: brute force trying to get access to the system. Authenticatio...	10	5763
Jul 29, 2025 @ 18:37:48.652	nirav	sshd: brute force trying to get access to the system. Authenticatio...	10	5763
Jul 29, 2025 @ 18:36:41.821	nirav	sshd: brute force trying to get access to the system. Authenticatio...	10	5763
Jul 29, 2025 @ 18:34:47.548	nirav	sshd: brute force trying to get access to the system. Authenticatio...	10	5763
Jul 29, 2025 @ 16:49:27.624	nirav	sshd: brute force trying to get access to the system. Authenticatio...	10	5763
Jul 29, 2025 @ 16:47:19.172	nirav	sshd: brute force trying to get access to the system. Authenticatio...	10	5763

Figure 22: SSH Brute Force Attack

2. **SQL Injection Attack:** The SQL injection attempt has been detected all the attempts which were targeting web application and has the near real time detection.

```

(root@nirav)-[~/home/nirav]
# sqlmap -u "http://192.168.1.88/users/?id=1" --dump

[1] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program

[*] starting @ 06:53:39 /2025-07-30/

[06:53:39] [INFO] testing connection to the target URL
[06:53:39] [CRITICAL] page not found (404)
it is not recommended to continue in this kind of cases. Do you want to quit and make sure that everything is set up properly? [Y/n] n
[06:53:43] [INFO] testing if the target URL content is stable
[06:53:43] [INFO] target URL content is stable
[06:53:43] [INFO] testing if GET parameter 'id' is dynamic
[06:53:43] [WARNING] GET parameter 'id' does not appear to be dynamic
[06:53:43] [WARNING] heuristic (basic) test shows that GET parameter 'id' might not be injectable
[06:53:43] [INFO] testing for SQL injection on GET parameter 'id'
[06:53:43] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[06:53:43] [INFO] testing 'Boolean-based blind - Parameter replace (original value)'
[06:53:43] [INFO] testing 'MySQL >= 5.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXTRACTVALUE)'
[06:53:43] [INFO] testing 'PostgreSQL AND error-based - WHERE or HAVING clause'

```

Figure 23: SQL Map

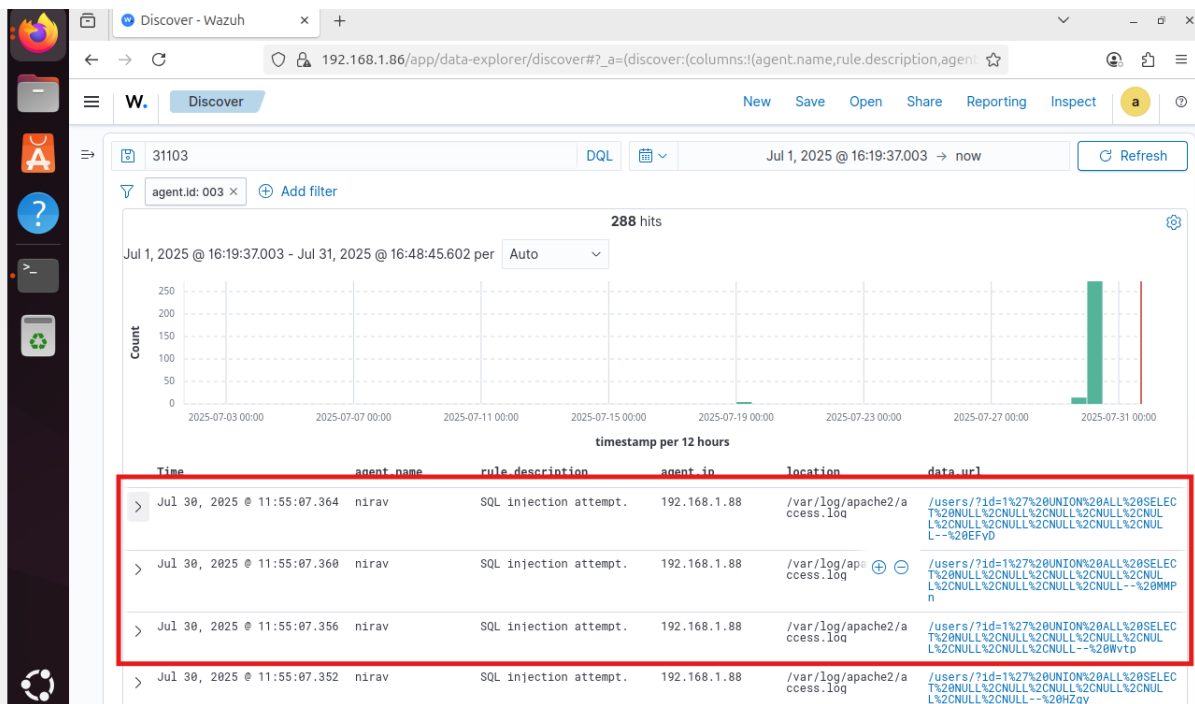


Figure 24: SQL Injection Attempt

3. **File Integrity Monitoring:** This usecase detects any creation of file or any modifications in the file system, using the checksum values for monitoring.

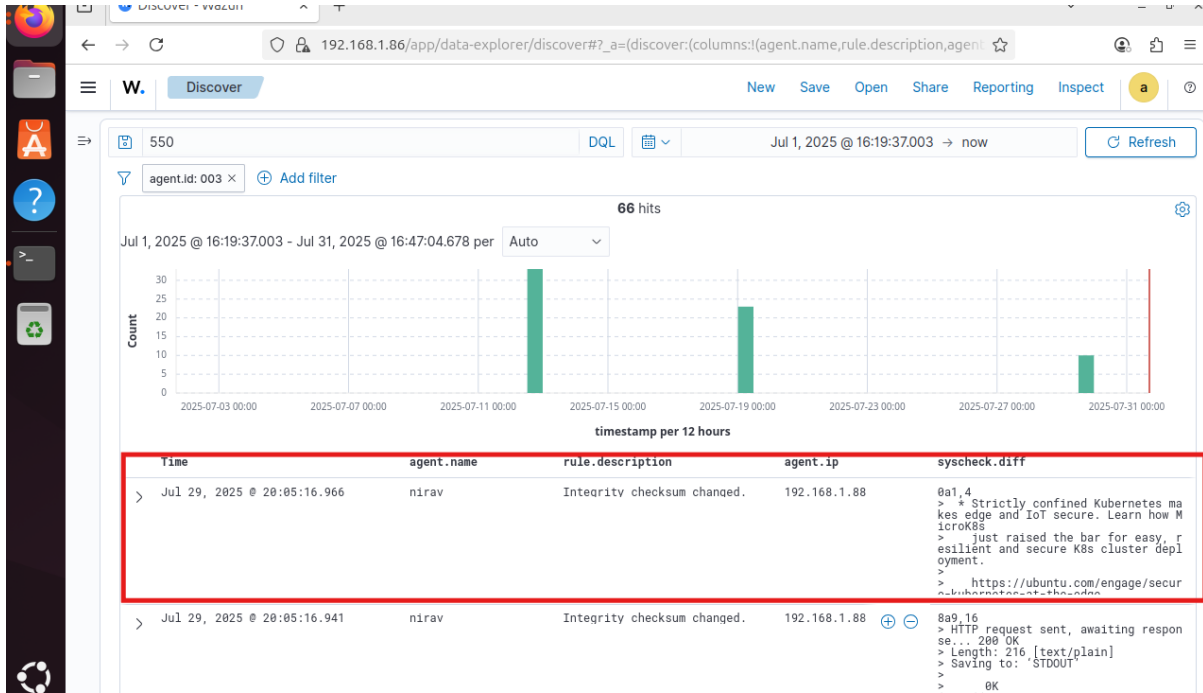


Figure 25: FIM

4. Detection of Shellshock Attack: This identifies the shellshock vulnerability in the infrastructure and detects it in near real time.

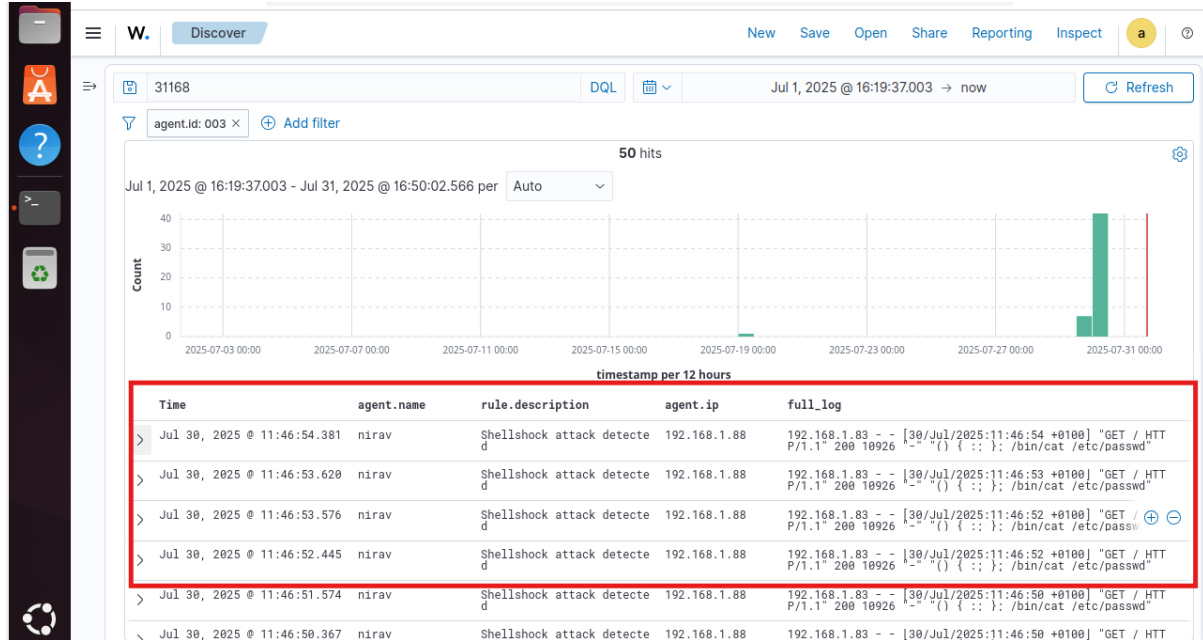


Figure 26: Shell Shock Attack detection

5. Detection of the Unauthorized Processess: This policy detects the unauthorized network tools such as netcat sessions which is on listening and is the indication of the reverse shell scripts.

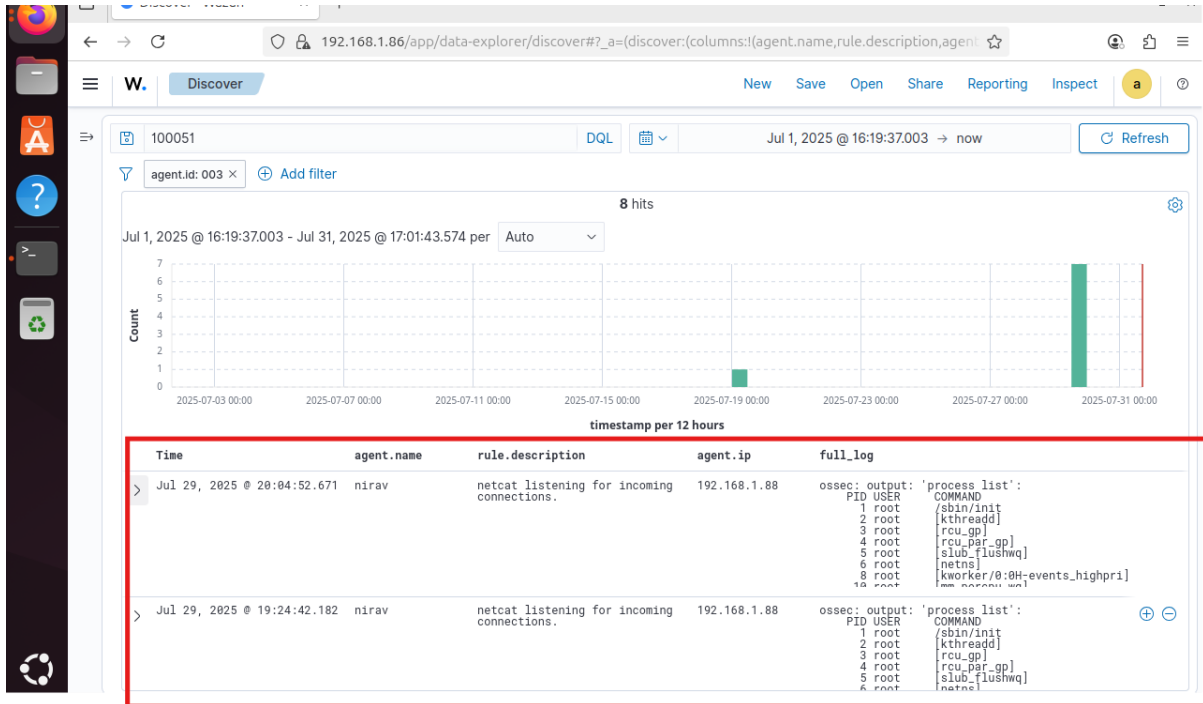


Figure 27: Unauthorized process

6. **Multiple Authentication Failures:** This rule triggers as there are lot of multiple authentication failures with in the short span of time in around 120 seconds.

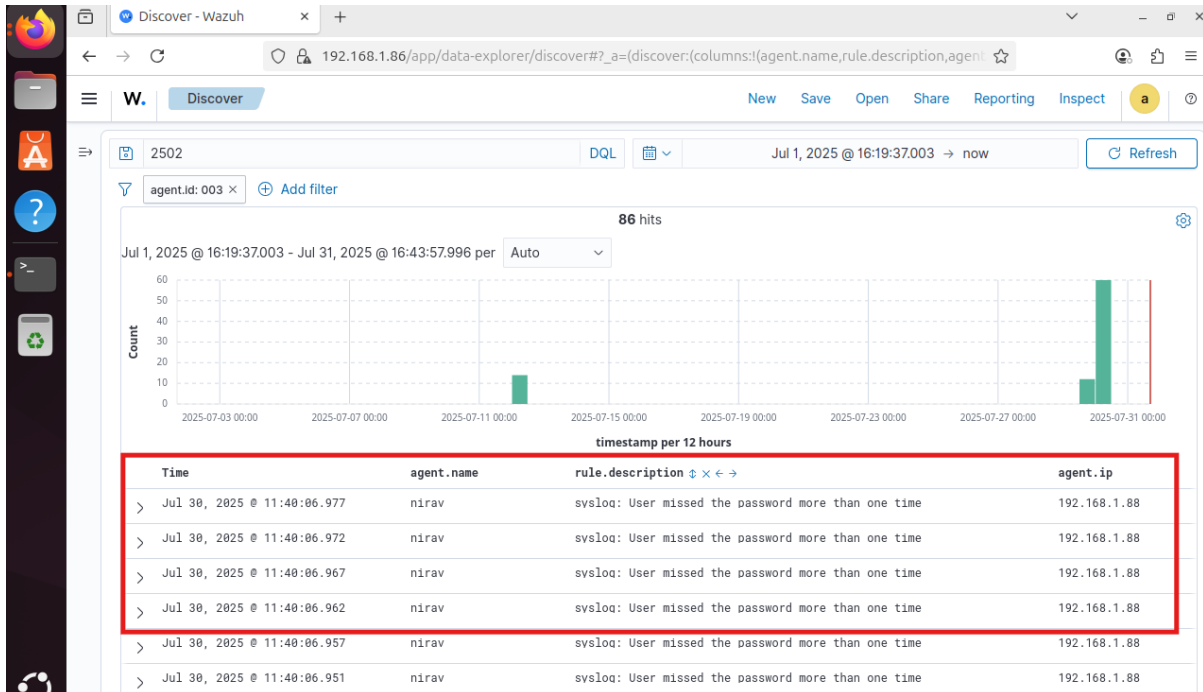


Figure 28: Multiple authentication Failure

7. **Non Existing User login attempts:** This rule is designed for detecting the adversaries reconnaissance techniques as they try to login using multiple usernames.

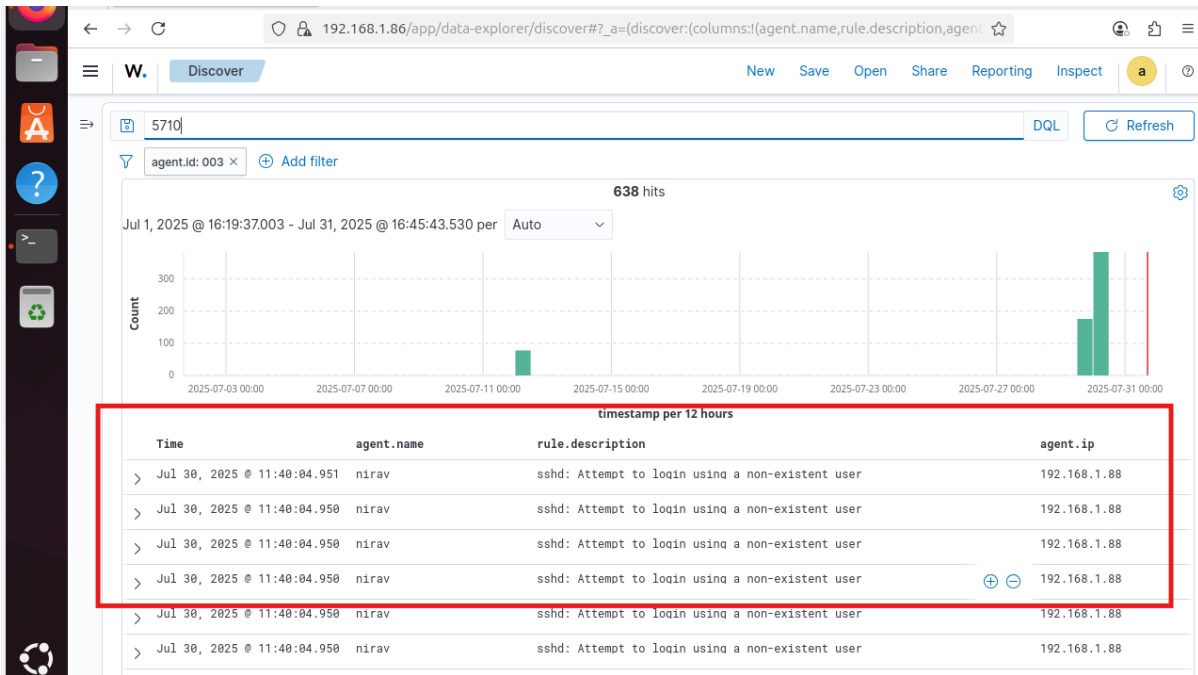


Figure 29: Non Existing User Login

8. **Successful login after the multiple failed attempts:** The rule triggers when there is successful login after the lots of failed login attempts in around time span of 240 seconds.

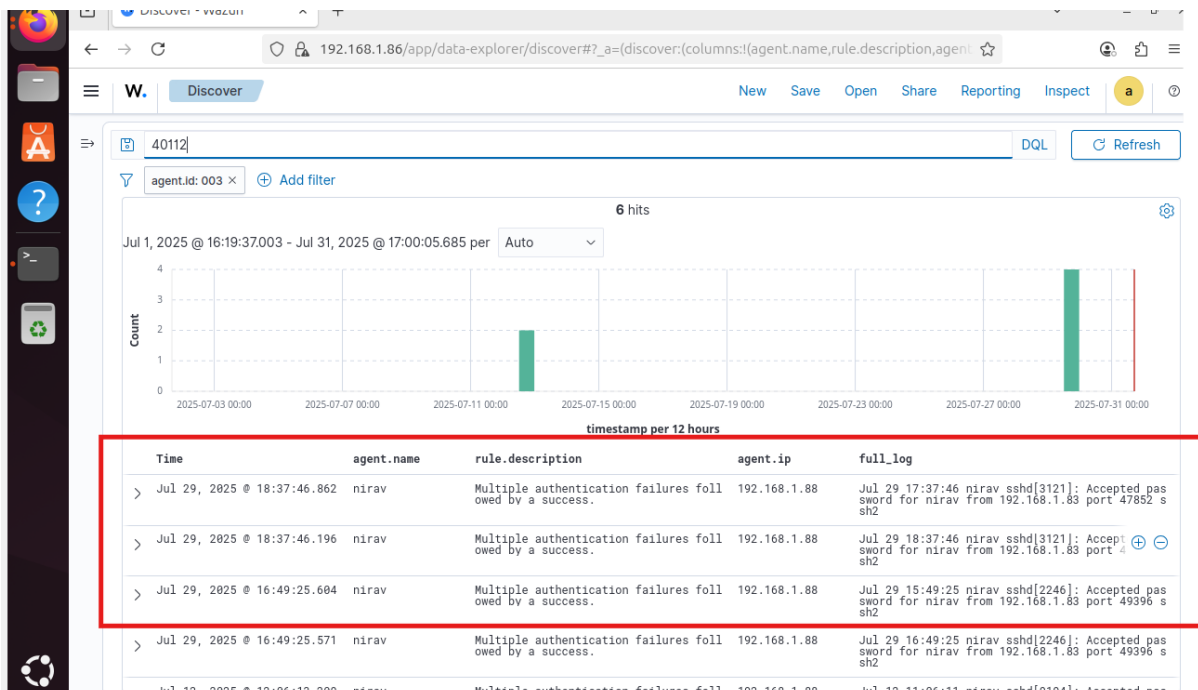


Figure 30: Successful Login after multiple failed Attempts

References

Spiffe.io. (2025). SPIFFE | Install the SPIRE Server. [online] Available at: <https://spiffe.io/docs/latest/deploying/install-server/> [Accessed 21 Jul. 2025].

Wazuh (n.d.). Deploying Wazuh agents on Linux systems - Wazuh agent. [online] documentation.wazuh.com. Available at: <https://documentation.wazuh.com/current/installation-guide/wazuh-agent/wazuh-agent-package-linux.html>.

Wazuh (2024). Installing the Wazuh server step by step - Wazuh server. [online] Wazuh.com. Available at: <https://documentation.wazuh.com/current/installation-guide/wazuh-server/step-by-step.html>.