#siem #wazuh #soar #thehive

- **Name**: Iskander Nafikov
- **E-mail**: i.nafikov@innopolis.university
- **GitHub**: https://github.com/iskanred

LAB4 SIEM

Description

This lab is designed to introduce students to security solutions, specifically a SIEM. In this lab, students can use any SIEM of choice; regardless, a solid recommendation is to use the open source security platform Wazuh as this provides a fleet of capabilities at no cost.

In this lab, students will interact with additional tools such as virustotal, YARA, osquery, SOAR and also gain experience with SIEM log analysis, vulnerability detection and more.

Part A

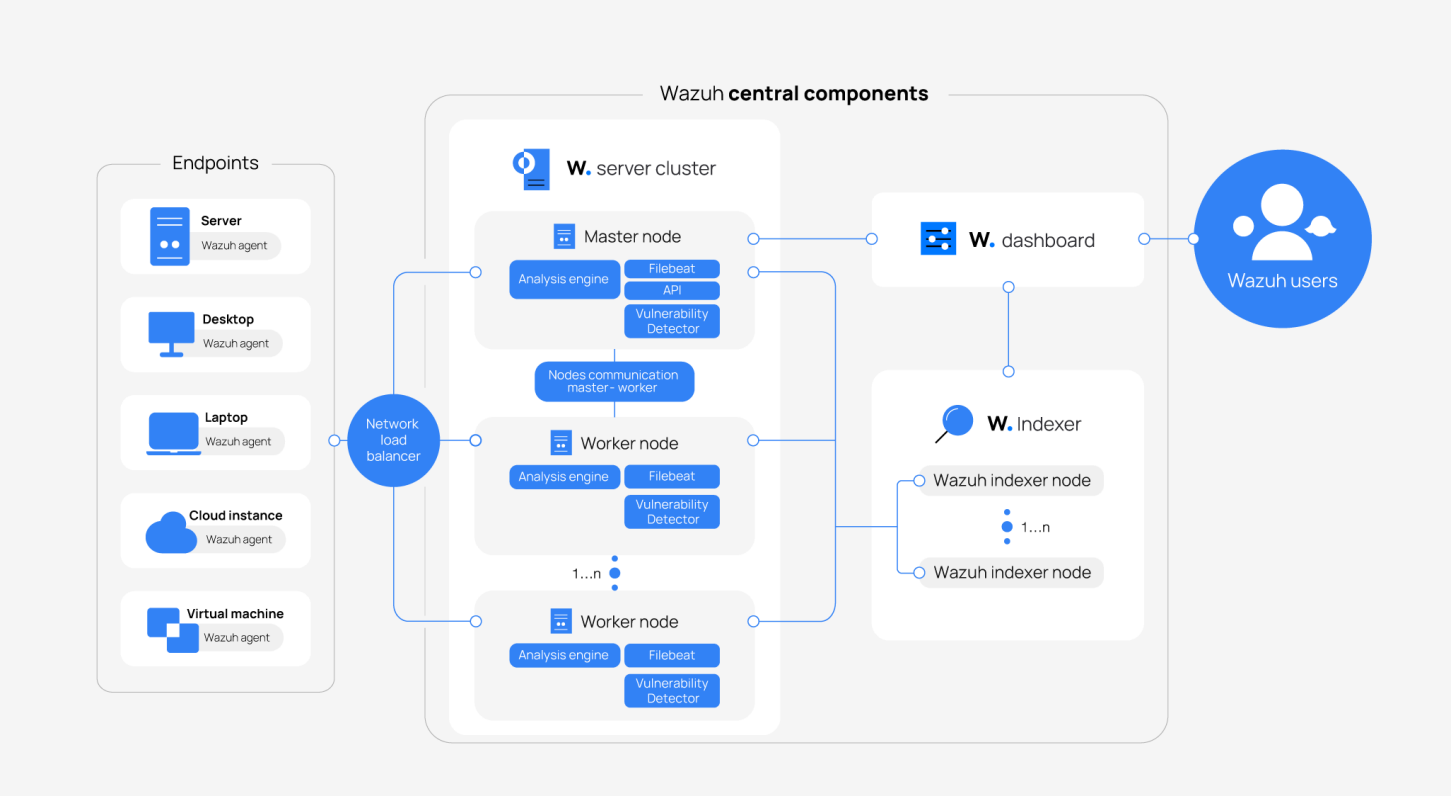
Task 1 - Introduction

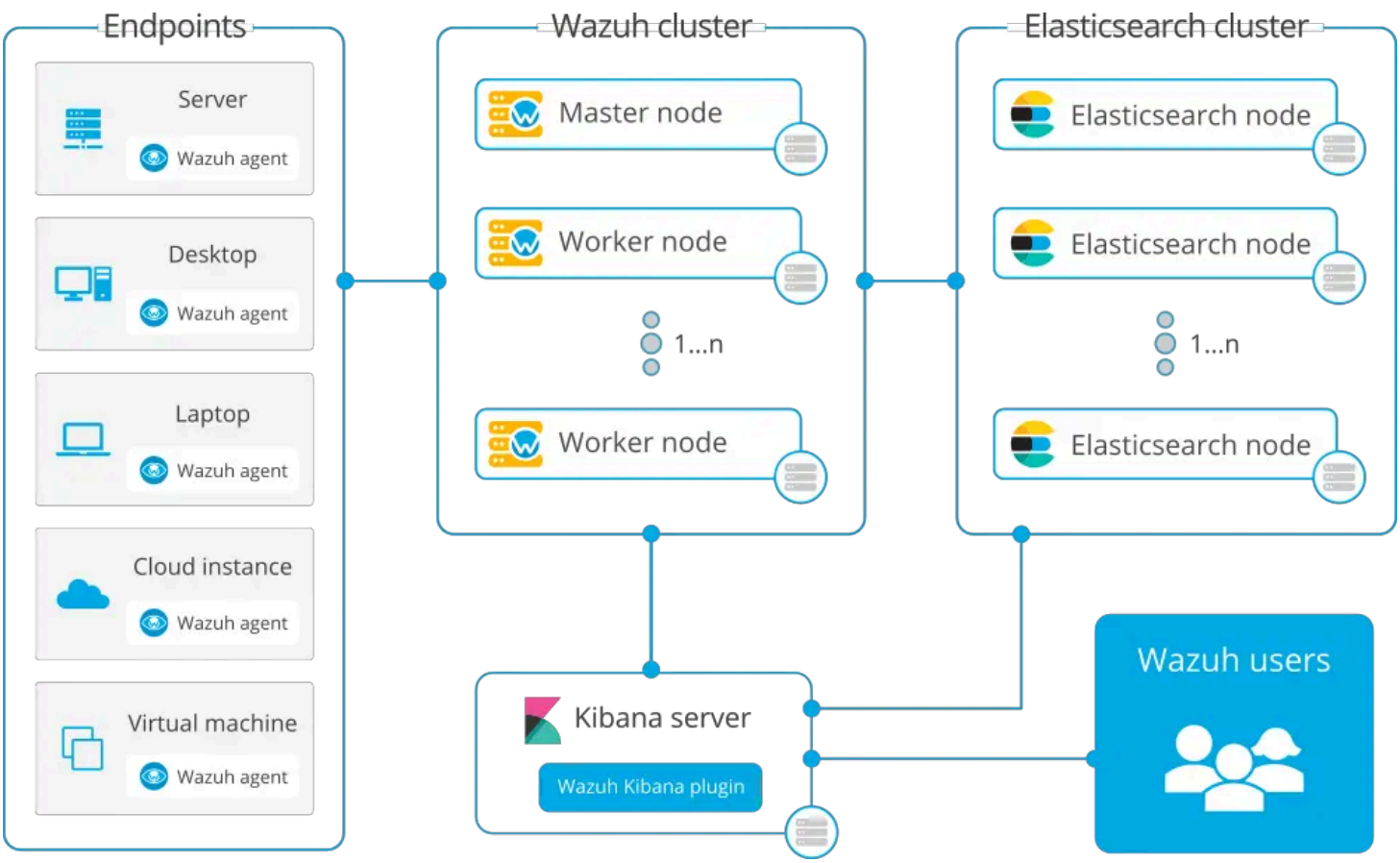
a.

Task description

a. Give a brief explanation of the architecture of your SIEM solution

- Here is the pictures





- Below is the description:
  - **Wazuh Agents**: These lightweight agents are installed on the endpoints that need to be monitored. They collect security event data, such as logs, and system information, perform file integrity checks, and monitor for vulnerabilities. The agents then send this data to the Wazuh manager.
  - **Wazuh Manager**: This central component aggregates and analyzes the data received from the agents. It processes alerts, applies rules for threat detection, and generates security notifications.

The Wazuh manager also communicates with the Wazuh API and the Wazuh dashboard.

- **Wazuh Indexer**: Wazuh utilizes indexers for storing and indexing the log data collected from agents. It enables powerful search capabilities and facilitates the real-time querying of security logs and events.
- **Wazuh Dashboard**: This is a visualization tool that is used to create dashboards and reports based on the data stored in Wazuh Indexers. It provides a user-friendly interface for visualizing security data, allowing users to analyze and correlate events effectively.
- **Wazuh API**: The API allows for integration with other tools and automation workflows. It provides programmatic access to the Wazuh manager's functionality, enabling users to query alerts, manage agents, and retrieve logs.

- You may also notice that ELK stack can be used instead of Wazuh Indexers and Wazuh Dashboard.

b.

> **Task description**
>
> b. Provide 3 advantages of open source solutions and how do these vendors actually make money?

- **Advantages**:
  1. **Cost Savings**:
     - Open source software is **typically** (but not always!) free to use, which can significantly reduce the costs associated with software licensing. Organizations can allocate their budgets towards other critical areas, such as infrastructure, training, or personnel.
  2. **Flexibility and Customization**:
     - Open source solutions allow users to access and modify (not always!) the source code, enabling organizations to customize the software to fit their specific needs. This flexibility allows for the development of unique features, integrations, and functionality that may not be available in proprietary software.
  3. **Community Support and Innovation**:
     - Open source projects often have vibrant communities that contribute to their development, maintenance, and support. This collaborative environment can lead to rapid innovation, as many developers contribute ideas, report issues, and create enhancements. Additionally, community forums and user groups provide valuable resources for troubleshooting and best practices.
- **How do vendors make money?**
  - ❗ Firstly, Open Source does not immediately mean free software (according to Richard Stallman), so companies may make code of their product open source while keeping it proprietary. Therefore companies can sell their software that is open-source directly to customers.
  - **Creation for their own sake**. Some big corporations may need some software that they develop and make open-source after to make the technology developing further. In such a case this

software is not their primary goal to earn money but rather a tool that they make available to the community.

- **Consulting and Professional Services**. Open source vendors may offer consulting services, training, and implementation assistance to help organizations effectively deploy and utilize their software.
- **Hosted Solutions**. Some open source vendors provide cloud-hosted versions of their software as a service (SaaS). Customers pay for the convenience of using the software without the need for local installations or maintenance.
- **Donations**: Open source vendors can generate revenue through donations by allowing individuals and organizations to contribute financially to the projects they use. This enables communities to support ongoing development, maintenance, and feature enhancement.

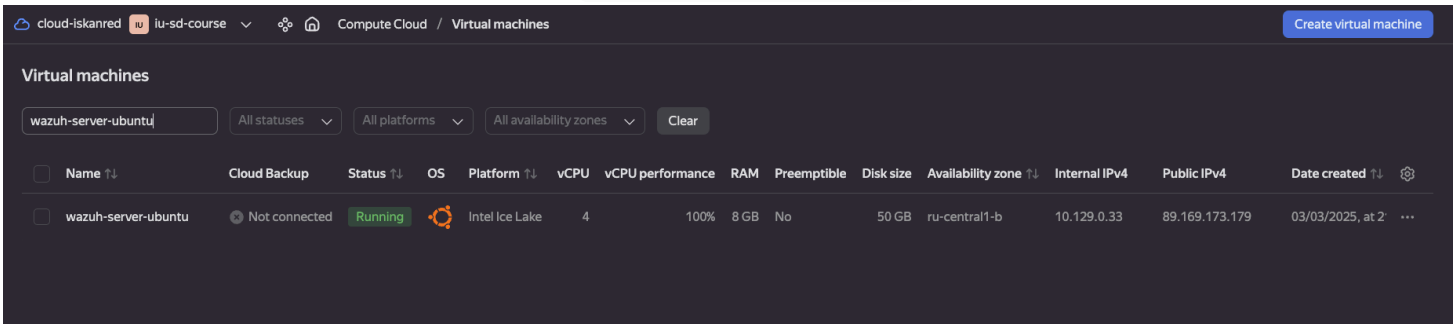
Task 2 - Setup infrastructure

a.

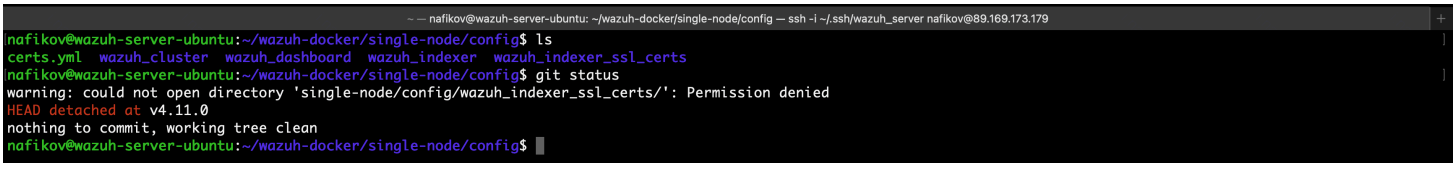
> **Task description**
>
> a. Configure a SIEM solution with 3(or more) unique devices. e.g Windows, Linux and a Network device. Can you view log data from each connected device? If yes show this.

Wazuh server

- Firstly, I setup Wazuh server-side: manager, indexer, dashboard on the cloud machine using Docker
- Below is a configuration of my cloud VM `wazuh-server-ubuntu` on Yandex Cloud



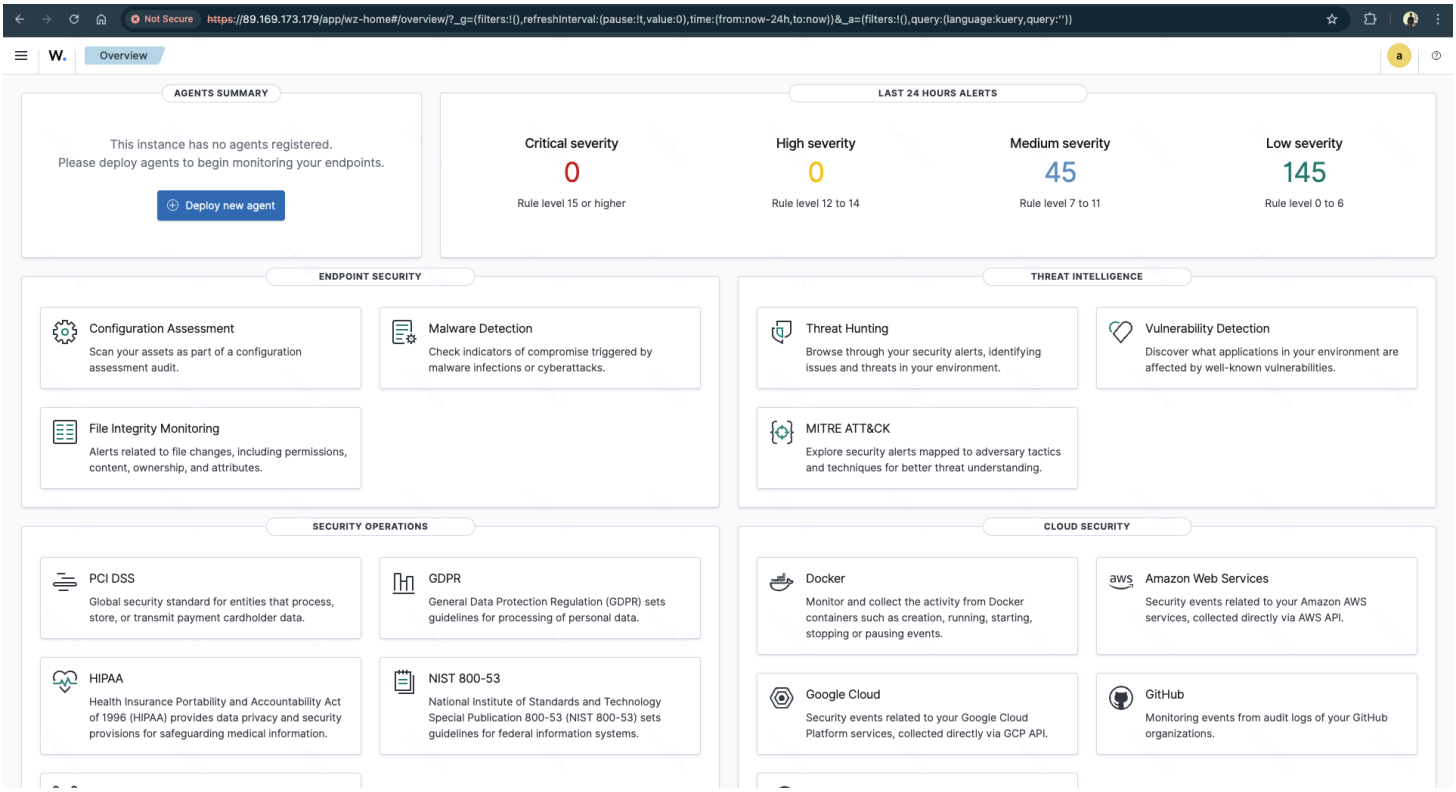
- IP = `89.169.173.179`, local user is `nafikov`, and hostname is `wazuh-server-ubuntu`
- I installed Docker there and cloned the official `wazuh-docker` Git repo: https://github.com/wazuh/wazuh-docker/tree/main/single-node



- Following the instruction in `single-node/README.md` I generated TLS/SSL certificates and finally ran the compose file

```
~ — nafikov@wazuh-server-ubuntu: ~/wazuh-docker/single-node/config — ssh -i ~/.ssh/wazuh_server nafikov@89.169.173.179
nafikov@wazuh-server-ubuntu:~/wazuh-docker/single-node/config$ docker compose ls
NAME                STATUS              CONFIG FILES
single-node         running(3)          /home/nafikov/wazuh-docker/single-node/docker-compose.yml
nafikov@wazuh-server-ubuntu:~/wazuh-docker/single-node/config$ docker ps
CONTAINER ID   IMAGE                          COMMAND                  CREATED          STATUS          PORTS
                                                                                                      NAMES
cb9268748949   wazuh/wazuh-dashboard:4.11.0   "/entrypoint.sh"         12 minutes ago   Up 12 minutes   443/tcp, 0.0.0.0:443->5601/tcp, [::]:443->5601/tcp
                                                                                                      single-node-wazuh.dashboard-1
84ef0dd0b071   wazuh/wazuh-manager:4.11.0     "/init"                  12 minutes ago   Up 12 minutes   0.0.0.0:1514-1515->1514-1515/tcp, [::]:1514-1515->1514-1515/tcp, 0.
0.0.0:514->514/udp, [::]:514->514/udp, 0.0.0.0:55000->55000/tcp, [::]:55000->55000/tcp, 1516/tcp   single-node-wazuh.manager-1
a22d2b05f5a1   wazuh/wazuh-indexer:4.11.0     "/entrypoint.sh open…"   12 minutes ago   Up 12 minutes   0.0.0.0:9200->9200/tcp, [::]:9200->9200/tcp
                                                                                                      single-node-wazuh.indexer-1
nafikov@wazuh-server-ubuntu:~/wazuh-docker/single-node/config$
```

- Finally, I was able to access the dashboard:

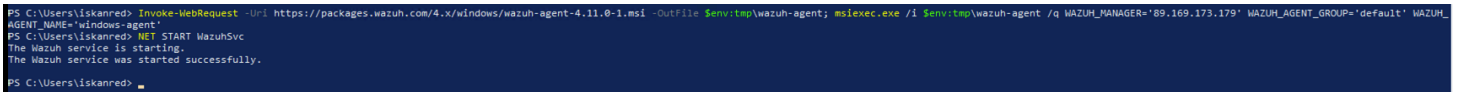


Wazuh agents

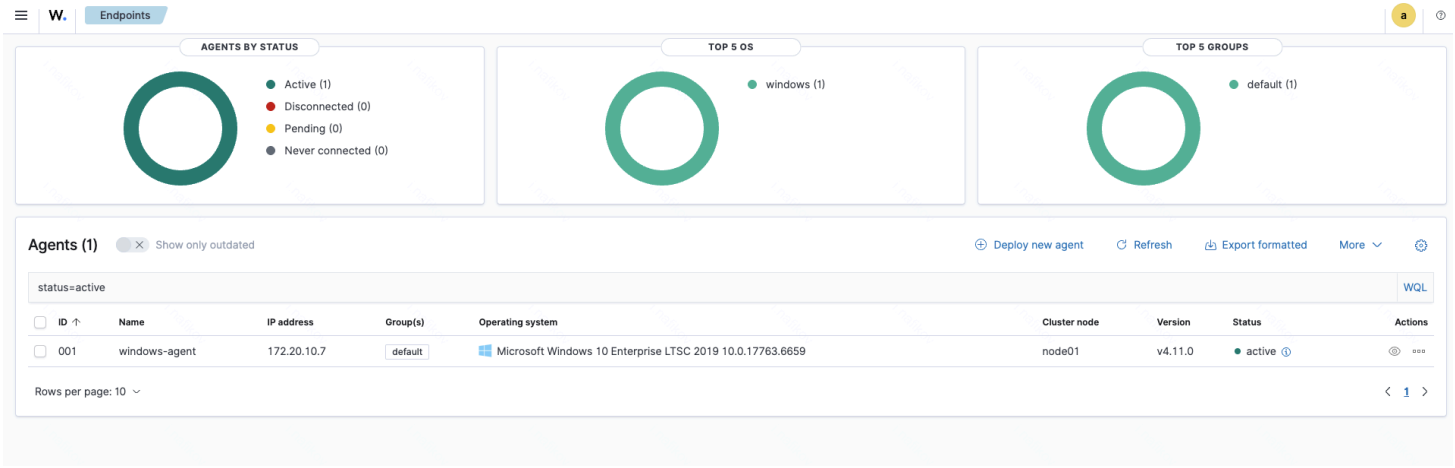
Windows

- First, I installed an agent on my **Windows machine** using the PowerShell script:

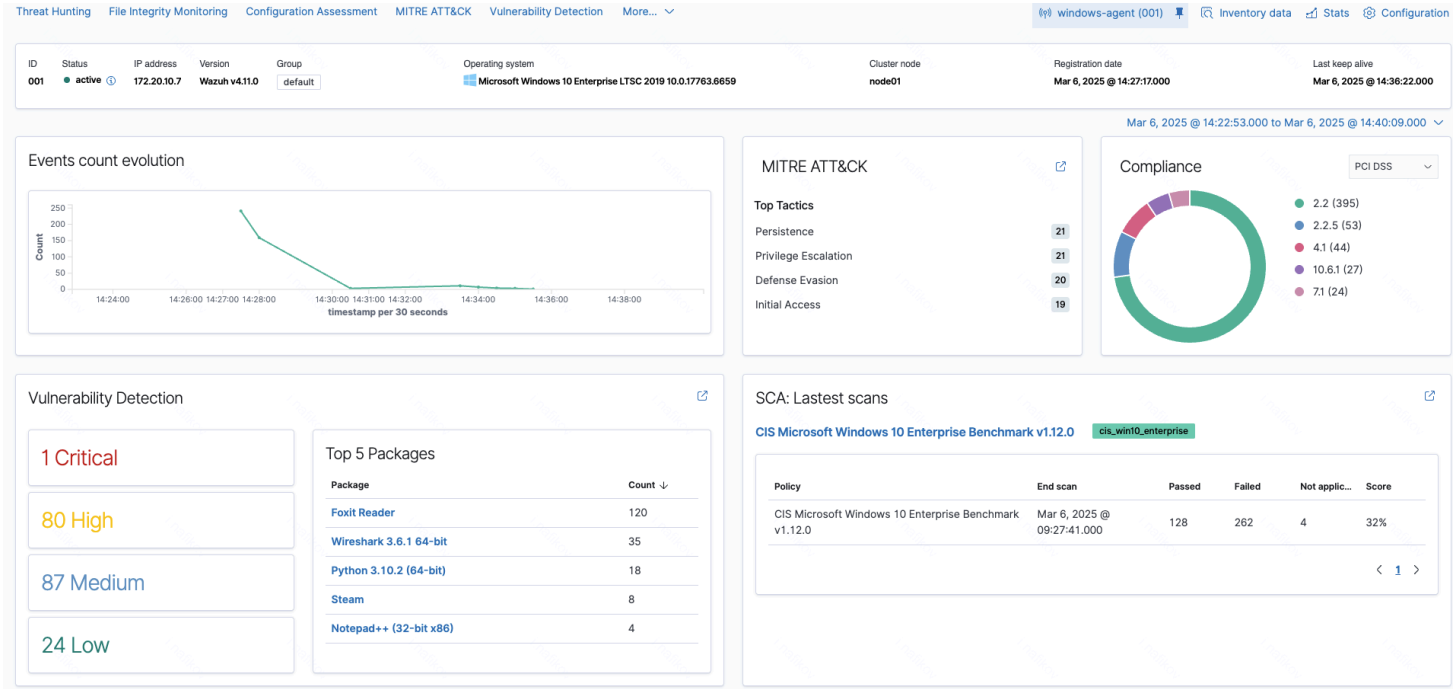
```
Invoke-WebRequest -Uri https://packages.wazuh.com/4.x/windows/wazuh-agent-4.11.0-1.msi -OutFile $env:tmp\wazuh-agent
msiexec.exe /i $env:tmp\wazuh-agent /q WAZUH_MANAGER='89.169.173.179' WAZUH_AGENT_GROUP='default' WAZUH_AGENT_NAME='windows-agent'
```



- Finally I could see my first endpoint in Wazuh



- We can see its state from the perspective of Wazuh



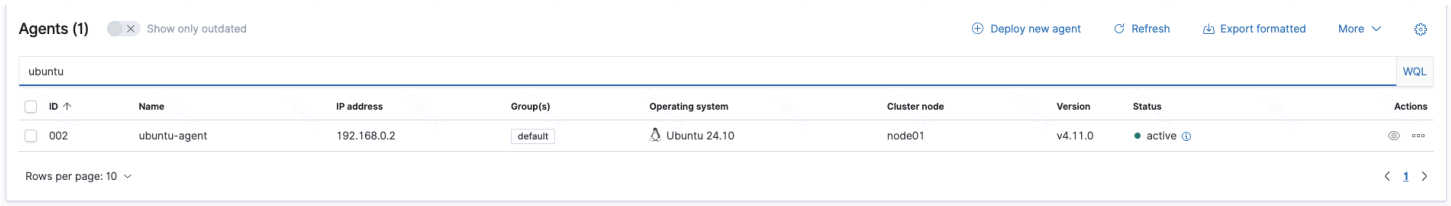
Ubuntu

- Now let's integrate Wazuh agent to my **virtual Ubuntu Cloud machine**:

```
wget https://packages.wazuh.com/4.x/apt/pool/main/w/wazuh-agent/wazuh-agent_4.11.0-1_amd64.deb && \
sudo WAZUH_MANAGER='89.169.173.179' WAZUH_AGENT_GROUP='default' WAZUH_AGENT_NAME='ubuntu-agent' dpkg -i ./wazuh-agent_4.11.0-1_amd64.deb
```

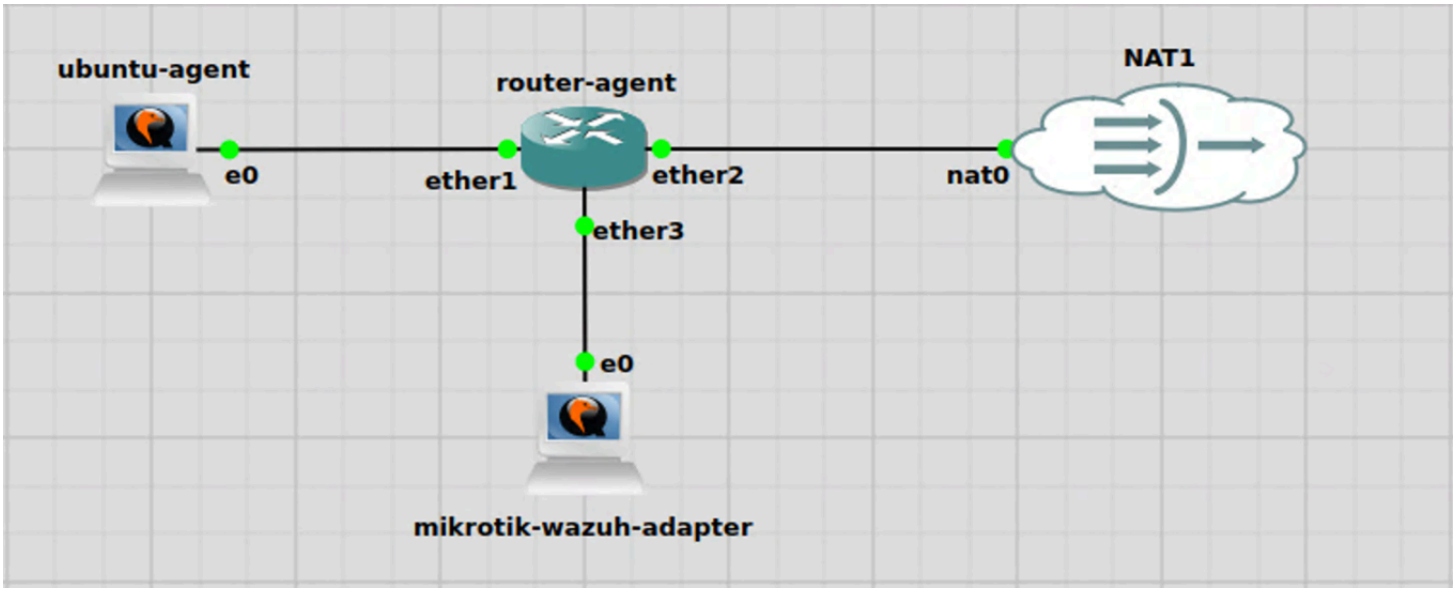
```
sudo systemctl daemon-reload
sudo systemctl enable wazuh-agent
sudo systemctl start wazuh-agent
```

```
ubuntu@ubuntu-cloud:~$ sudo wget https://packages.wazuh.com/4.x/apt/pool/main/w/wazuh-agent/wazuh-agent_4.11.0-1_amd64.debwget https://packages.wa
zuh.com/4.x/apt/pool/main/w/wazuh-agent/wazuh-agent_4.11.0-1_amd64.deb && sudo WAZUH_MANAGER='89.169.173.179' WAZUH_AGENT_GROUP='default' WAZUH_AG
ENT_NAME='ubuntu-agent' dpkg -i ./wazuh-agent_4.11.0-1_amd64.deb
--2025-03-06 15:26:43--  https://packages.wazuh.com/4.x/apt/pool/main/w/wazuh-agent/wazuh-agent_4.11.0-1_amd64.deb
Resolving packages.wazuh.com (packages.wazuh.com)... 3.164.68.88, 3.164.68.26, 3.164.68.41, ...
Connecting to packages.wazuh.com (packages.wazuh.com)|3.164.68.88|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 11078824 (11M) [application/vnd.debian.binary-package]
Saving to: 'wazuh-agent_4.11.0-1_amd64.deb'

wazuh-agent_4.11.0- 100%[===================>]  10.57M   117KB/s    in 92s

2025-03-06 15:28:16 (118 KB/s) - 'wazuh-agent_4.11.0-1_amd64.deb' saved [11078824/11078824]

Selecting previously unselected package wazuh-agent.
(Reading database ... 76019 files and directories currently installed.)
Preparing to unpack .../wazuh-agent_4.11.0-1_amd64.deb ...
Unpacking wazuh-agent (4.11.0-1) ...
Setting up wazuh-agent (4.11.0-1) ...
ubuntu@ubuntu-cloud:~$ sudo systemctl daemon-reload
sudo systemctl enable wazuh-agent
sudo systemctl start wazuh-agent
Created symlink '/etc/systemd/system/multi-user.target.wants/wazuh-agent.service' → '/usr/lib/systemd/system/wazuh-agent.service'.
ubuntu@ubuntu-cloud:~$ sudo systemctl status wazuh-agent.service
● wazuh-agent.service - Wazuh agent
     Loaded: loaded (/usr/lib/systemd/system/wazuh-agent.service; enabled; pres>
     Active: active (running) since Thu 2025-03-06 15:30:31 UTC; 21s ago
 Invocation: 361d8092740841f0ae29364ad4714d66
    Process: 2914 ExecStart=/usr/bin/env /var/ossec/bin/wazuh-control start (co>
      Tasks: 22 (limit: 1058)
     Memory: 131.3M (peak: 131.9M)
        CPU: 3.918s
     CGroup: /system.slice/wazuh-agent.service
             ├─3079 /bin/sh active-response/bin/restart.sh agent
```



Mikrotik Router

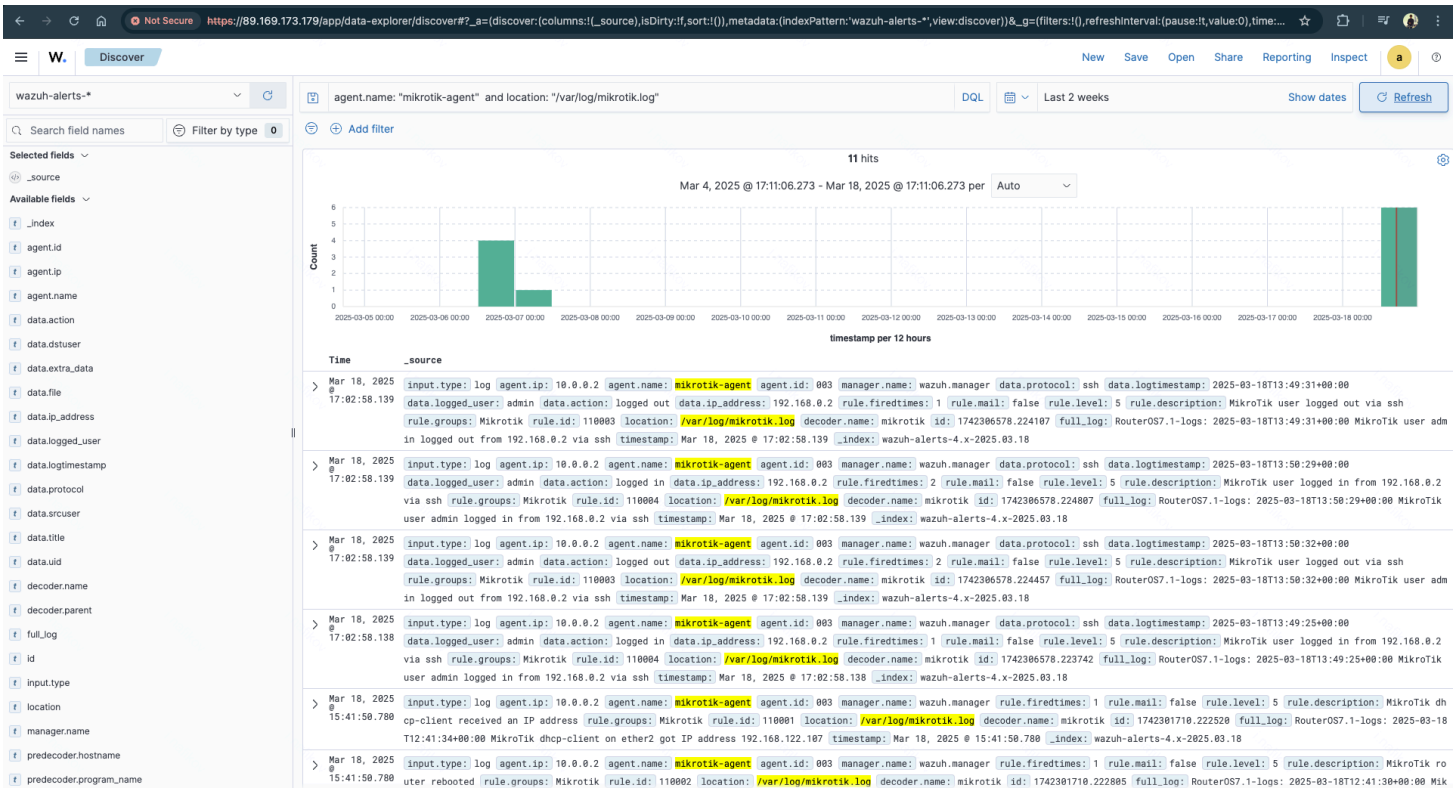
- Finally, let's connect my **virtual Mirkotik Router**
- Using this official article I configured transferring logs from the router to the Wazuh server via another Ubuntu agent (`mikrotik-wazuh-adapter`).
- By the way I found some mistake in this article and created an issue on GitHub for wazuh-documentation
- Therefore, I got the following network structure:



- We have `ubuntu-agent` , `router-agent` , and `miktorik-wazuh-adapter` .
- The following log events will be sent from the router:
  - `MikroTik dhcp-client received an IP address`

- `MikroTik router rebooted`
- `MikroTik user logged out via {PROTOCOL}`
- `MikroTik user logged in from {IP_ADDRESS} via {PROTOCOL}`

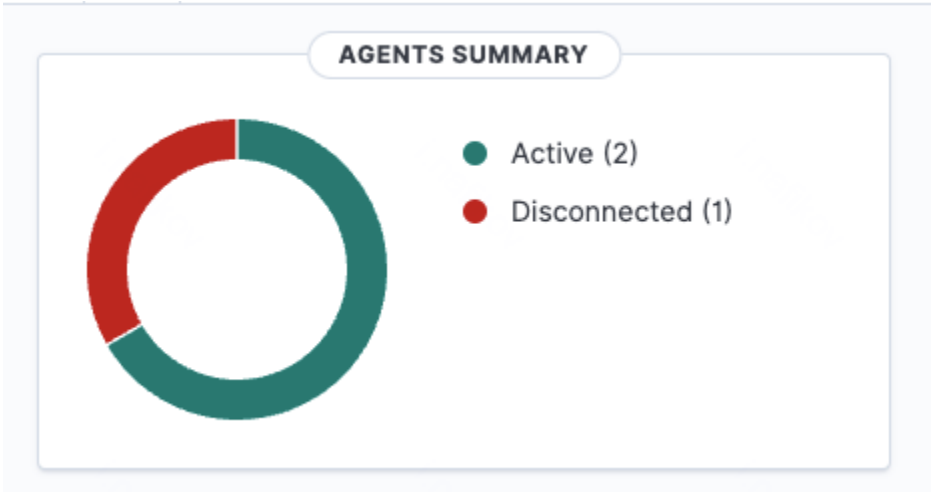
- Let's finally check the logs in Wazuh



- We see that sending logs actually works

⚠️ Disclaimer

- Unfortunately, I have **two systems** (Windows & Ubuntu) **on the same machine**, so I need to disable Windows system (therefore making this endpoint inactive). I cannot use virtualisation for 3 devices since I have only 4Gb of memory on my device 😢
- So, I decided to keep only these two systems and connect Windows only when necessary.

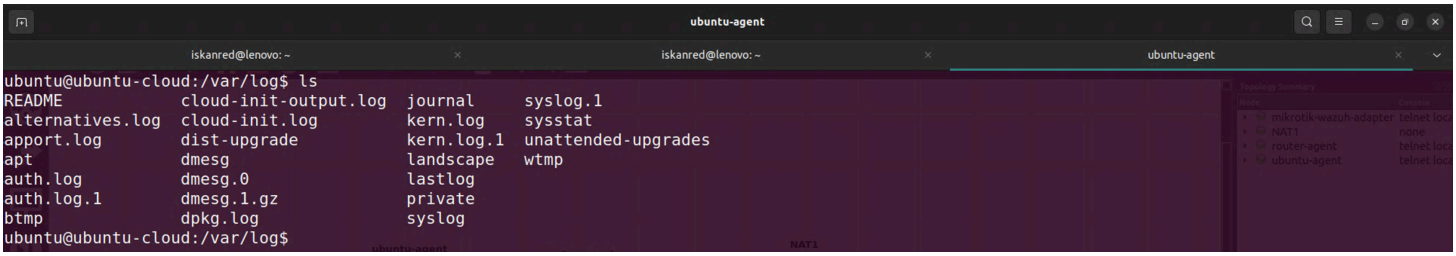


b.

🖊 Task description

b. Why specifically are you able to view these logs i.e select two visible logs, explain these logs, and explain why and how you are able to view it on the SIEM.

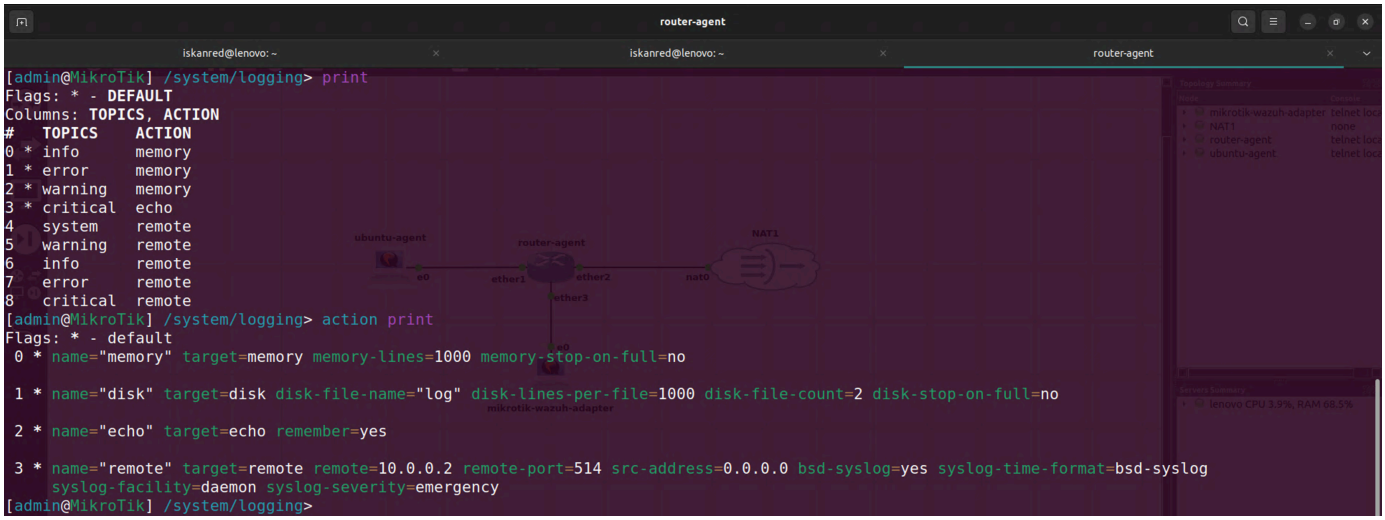
- I am able to view these logs because they were sent to the Wazuh Manager Server using `wazuh-agent` software. These logs are scrapped using `rsyslog`. For example there are some log files on `ubunut-agent`:



- Let's check two different logs and try to explain them

`MikroTik user logged in from {IP_ADDRESS} via {PROTOCOL}`

- **Explanation**: This log tells that someone logged in to the router's system via some protocol from some IP address.
- **Why and how I am able to view it**:
  - Mikrotik logs are sent from the `router-agent` to the `mikrotik-wazuh-adapter`.



  - Then `mikrotik-wazuh-adapter` accumulates these logs in the file `/var/log/mikrotik.log` with the help of `rsyslog`.

```
mikrotik-wazuh-adapter
iskanred@lenovo: ~ | iskanred@lenovo: ~ | ubuntu-agent | mikrotik-wazuh-adapter
ubuntu@ubuntu-cloud:~$ cd /var/log
ubuntu@ubuntu-cloud:/var/log$ ls
README            cloud-init-output.log  dpkg.log      private
alternatives.log  cloud-init.log         journal       syslog
apport.log        dist-upgrade           kern.log      sysstat
apt               dmesg                  landscape     unattended-upgrades
auth.log          dmesg.0                lastlog       wtmp
btmp              dmesg.1.gz             mikrotik.log
ubuntu@ubuntu-cloud:/var/log$ cat mikrotik.log
2025-03-06T19:44:59+00:00 MikroTik log action changed by x11-ubuntu-linux-x86-64-rv:136.0/web:admin@192.168.122.1 (/system logging action set "" b
sd-syslog=yes remote=10.0.0.2 remote-port=514 src-address=0.0.0.0 syslog-facility=daemon syslog-severity=emergency target=remote)
2025-03-06T19:45:00+00:00 MikroTik log action changed by x11-ubuntu-linux-x86-64-rv:136.0/web:admin@192.168.122.1 (/system logging action set "" b
sd-syslog=yes remote=10.0.0.2 remote-port=514 src-address=0.0.0.0 syslog-facility=daemon syslog-severity=emergency target=remote)
2025-03-06T20:45:03+00:00 MikroTik user admin logged out from 192.168.122.1 via web
2025-03-06T20:45:03+00:00 MikroTik user admin logged out from 192.168.122.1 via web
2025-03-06T20:54:11+00:00 MikroTik user admin logged out via local
2025-03-06T20:54:15+00:00 MikroTik user admin logged in via local
2025-03-06T20:54:54+00:00 MikroTik user admin logged in from 192.168.122.1 via web
2025-03-06T20:55:06+00:00 MikroTik user admin logged out from 192.168.122.1 via web
2025-03-06T21:02:13+00:00 MikroTik ip service changed by console:admin@ttyS0/action:7 (/ip service set ssh disabled=no)
2025-03-06T21:02:53+00:00 MikroTik user admin logged in from 192.168.122.1 via ssh
2025-03-06T21:13:38+00:00 MikroTik address added by ssh:admin@192.168.122.1 (*4 = /ip address add address=1.0.0.0/21 interface=ether1)
2025-03-06T21:15:25+00:00 MikroTik address removed by ssh:admin@192.168.122.1/action:8 (/ip address remove *4)
2025-03-10T02:37:35+00:00 MikroTik router was rebooted without proper shutdown
2025-03-10T02:37:35+00:00 MikroTik lo link up
2025-03-10T02:37:39+00:00 MikroTik dhcp-client on ether2 got IP address 192.168.122.107
ubuntu@ubuntu-cloud:/var/log$
```

- Afterwards `wazuh-agent` software for Ubuntu sends this logs to the Wazuh Manager Server.

```
mikrotik-wazuh-adapter
iskanred@lenovo: ~ | iskanred@lenovo: ~ | mikrotik-wazuh-adapter
ubuntu@ubuntu-cloud:~$ systemctl status wazuh-agent.service
● wazuh-agent.service - Wazuh agent
     Loaded: loaded (/usr/lib/systemd/system/wazuh-agent.service; enabled; pres>
     Active: active (running) since Tue 2025-03-18 10:30:15 UTC; 3h 46min ago
 Invocation: 79ed47322eb14d2fa6fe8be274a1da01
    Process: 669 ExecStart=/usr/bin/env /var/ossec/bin/wazuh-control start (cod>
      Tasks: 31 (limit: 1110)
     Memory: 326.9M (peak: 332.5M)
        CPU: 27.954s
     CGroup: /system.slice/wazuh-agent.service
             ├─883 /var/ossec/bin/wazuh-execd
             ├─921 /var/ossec/bin/wazuh-agentd
             ├─952 /var/ossec/bin/wazuh-syscheckd
             ├─966 /var/ossec/bin/wazuh-logcollector
             └─984 /var/ossec/bin/wazuh-modulesd

Notice: journal has been rotated since unit was started, output may be incomple>
lines 1-16/16 (END)
```

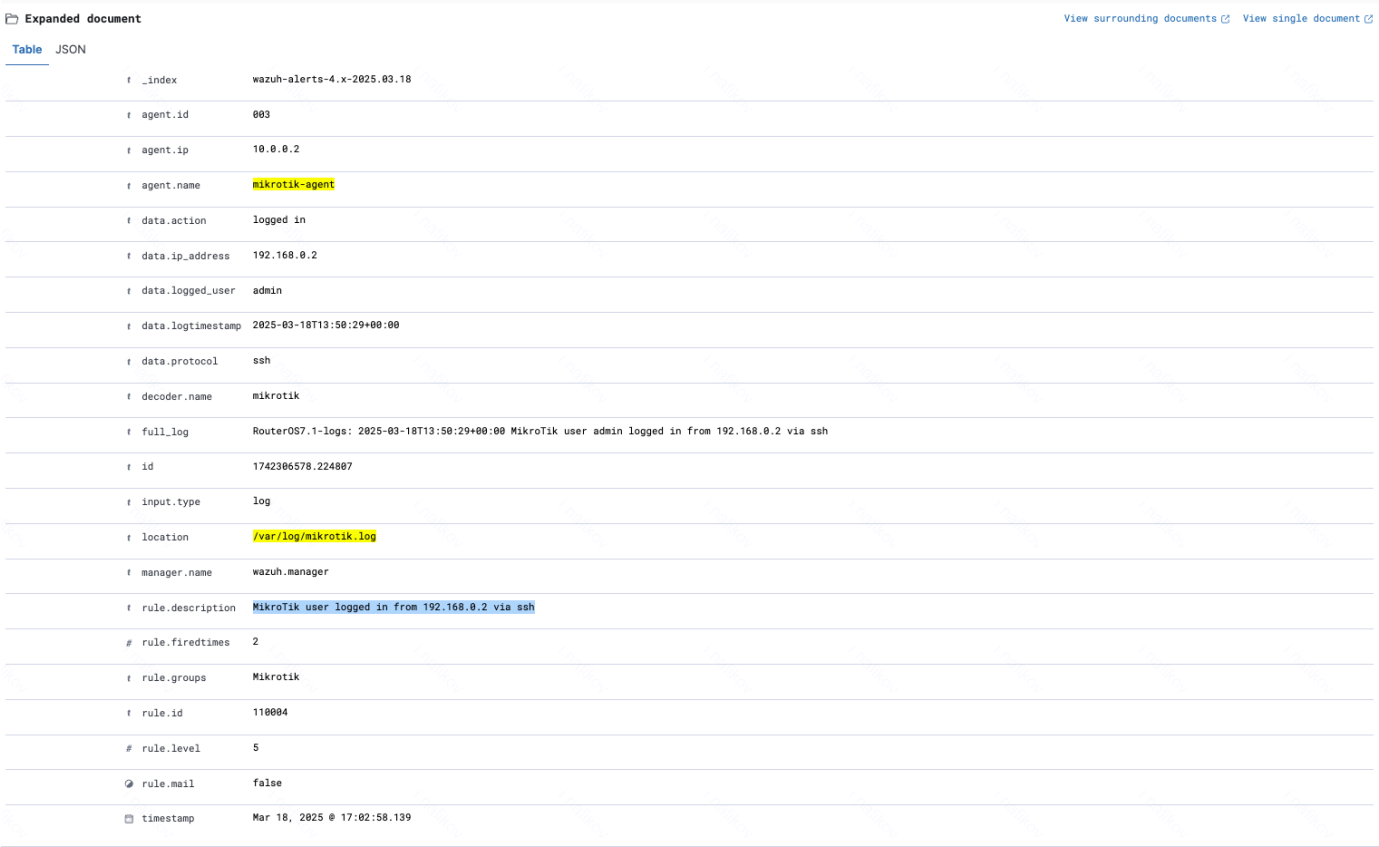
- Wazuh Manager applies custom `mirkotik decoders` to parse necessary fields from the log message

```
~ — nafikov@wazuh-server-ubuntu: ~/wazuh-docker/single-node — ssh -i .ssh/wazuh_server nafikov@89.169.173.179
nafikov@wazuh-server-ubuntu:~/wazuh-docker/single-node$ docker exec single-node-wazuh.manager-1 cat /var/ossec/etc/decoders/mikrotik_decoders.xml
<decoder name="mikrotik">
  <prematch>^RouterOS7.1-logs: </prematch>
</decoder>

<decoder name="mikrotik1">
  <parent>mikrotik</parent>
  <regex type="pcre2">\S+ (\d\d\d\d-\d\d-\d\d+T\d\d:\d\d:\d\d+\+\d\d\:\d\d) MikroTik user (\S+) (.*?) from (\d+.\d+.\d+.\d+) via (\w+)</regex>
  <order>logtimestamp, logged_user, action, ip_address, protocol</order>
</decoder>

<decoder name="mikrotik1">
  <parent>mikrotik</parent>
  <regex type="pcre2">\S+ (\d\d\d\d-\d\d-\d\d+T\d\d:\d\d:\d\d+\+\d\d\:\d\d) MikroTik dhcp-client on (\S+) (.*?) address (\d+\.\d+\.\d+\.\d+)</regex>
  <order>logtimestamp, interface, action, ip_address</order>
</decoder>

<decoder name="mikrotik1">
  <parent>mikrotik</parent>
  <regex type="pcre2">\S+ (\d\d\d\d-\d\d-\d\d+T\d\d:\d\d:\d\d+\+\d\d\:\d\d) MikroTik router (\S+)</regex>
  <order>logtimestamp, action</order>
</decoder>

nafikov@wazuh-server-ubuntu:~/wazuh-docker/single-node$
```

- After decoding Wazuh Manager applied custom `mikrotik rules` to transform raw logs to the Wazuh events

```
~ — nafikov@wazuh-server-ubuntu: ~/wazuh-docker/single-node — ssh -i .ssh/wazuh_server nafikov@89.169.173.179
nafikov@wazuh-server-ubuntu:~/wazuh-docker/single-node$ docker exec single-node-wazuh.manager-1 cat /var/ossec/etc/rules/mikrotik_rules.xml
<group name="Mikrotik,">

  <rule id="110000" level="0">
    <decoded_as>mikrotik</decoded_as>
    <description>Mikrotik-Event</description>
  </rule>


  <rule id="110001" level="5">
    <if_sid>110000</if_sid>
    <match>dhcp-client on ether</match>
    <description>MikroTik dhcp-client received an IP address $(ip_address)</description>
  </rule>

  <rule id="110002" level="5">
    <if_sid>110000</if_sid>
    <match>rebooted</match>
    <description>MikroTik router rebooted</description>
  </rule>

  <rule id="110003" level="5">
    <if_sid>110000</if_sid>
    <match>logged out from</match>
    <description>MikroTik user logged out via $(protocol)</description>
  </rule>

  <rule id="110004" level="5">
    <if_sid>110000</if_sid>
    <match>logged in from</match>
    <description>MikroTik user logged in from $(ip_address) via $(protocol)</description>
  </rule>

</group>nafikov@wazuh-server-ubuntu:~/wazuh-docker/single-node$
```

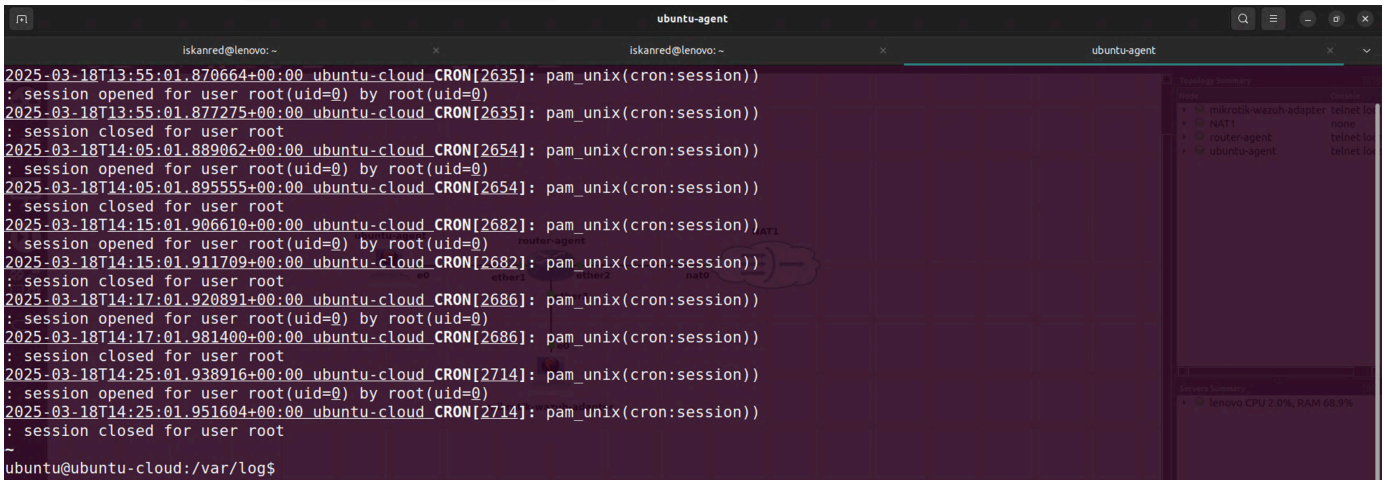
- Finally, these logs become visible as Wazuh triggered rules



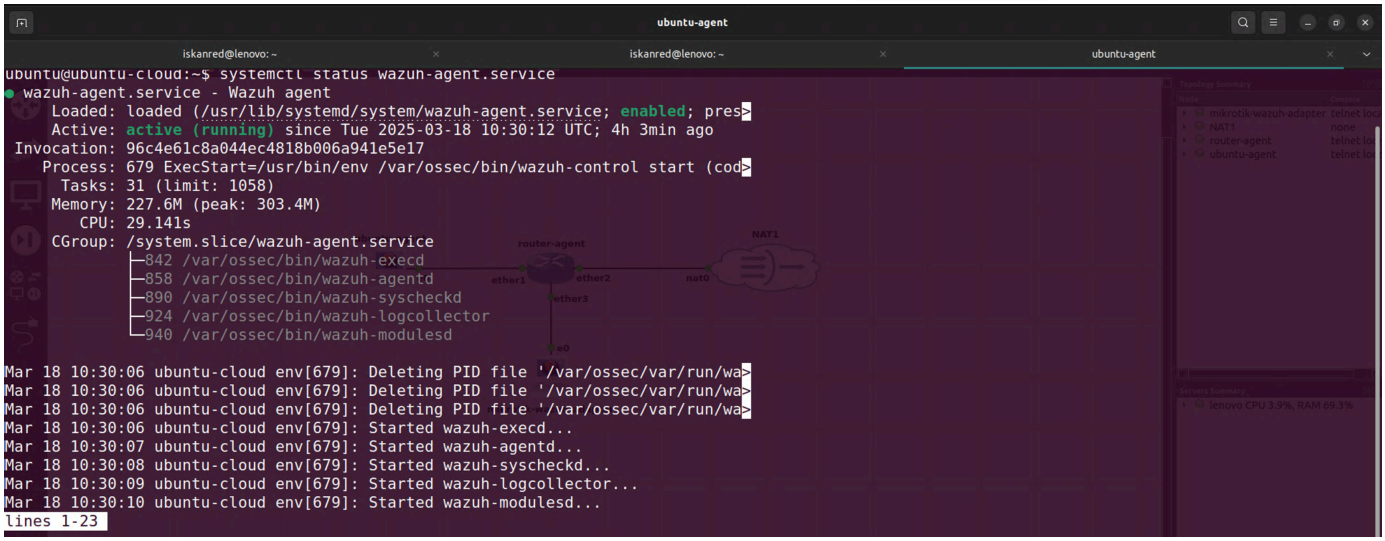
`PAM: Login session opened.`

- **Explanation**: This log tells that someone logged in to the `ubuntu-agent` 's system successfully.
- **Why and how I am able to view it**:

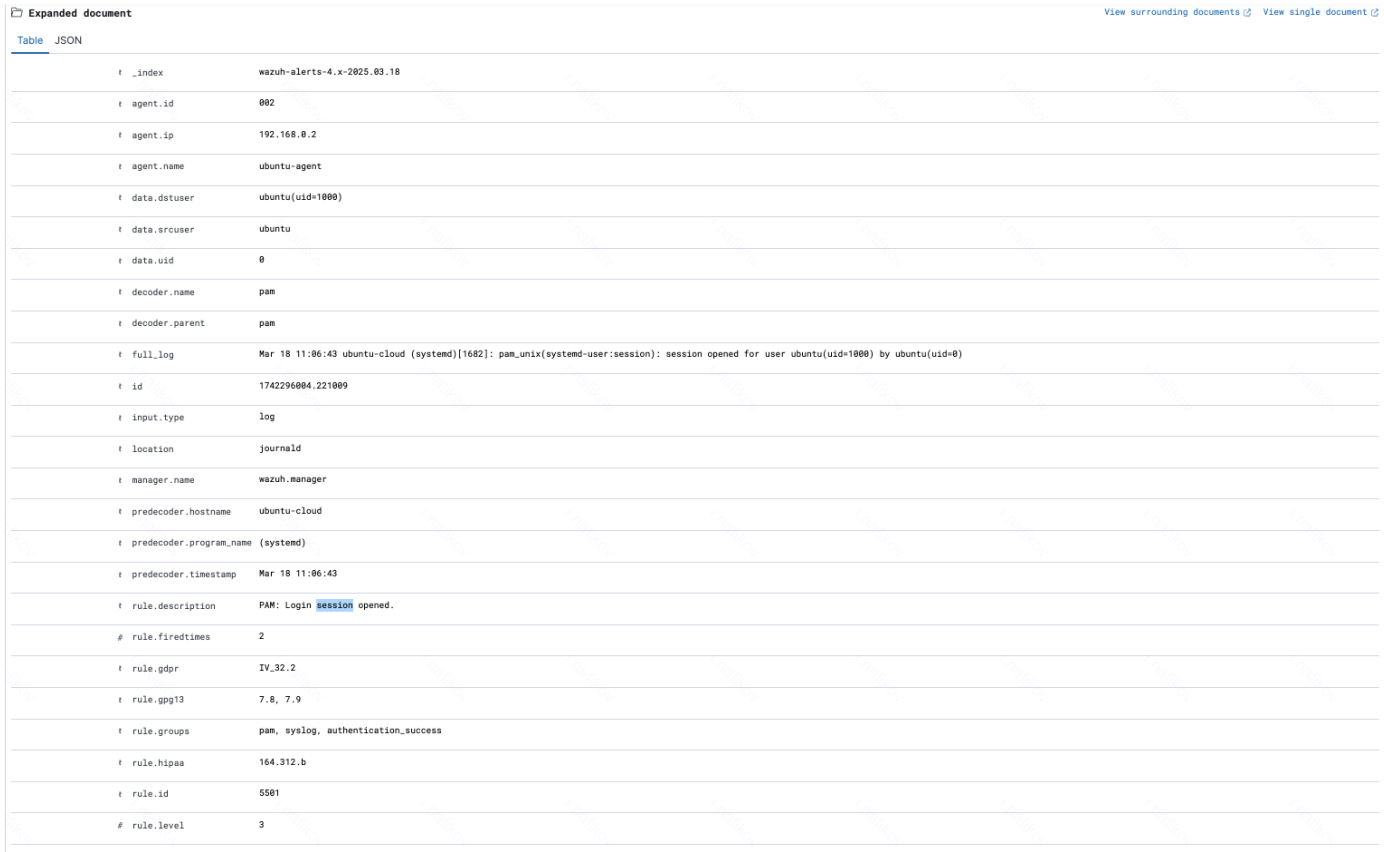
- Logs come from `/var/log/auth.logs`



- Then they are sent to the Wazuh Manager Server using the `wazug-agent` software



- Then they are parsed using default decoders and default rules are applied
- Finally, we can see them in the Wazuh Dashboard interface



Task 3 - Use cases

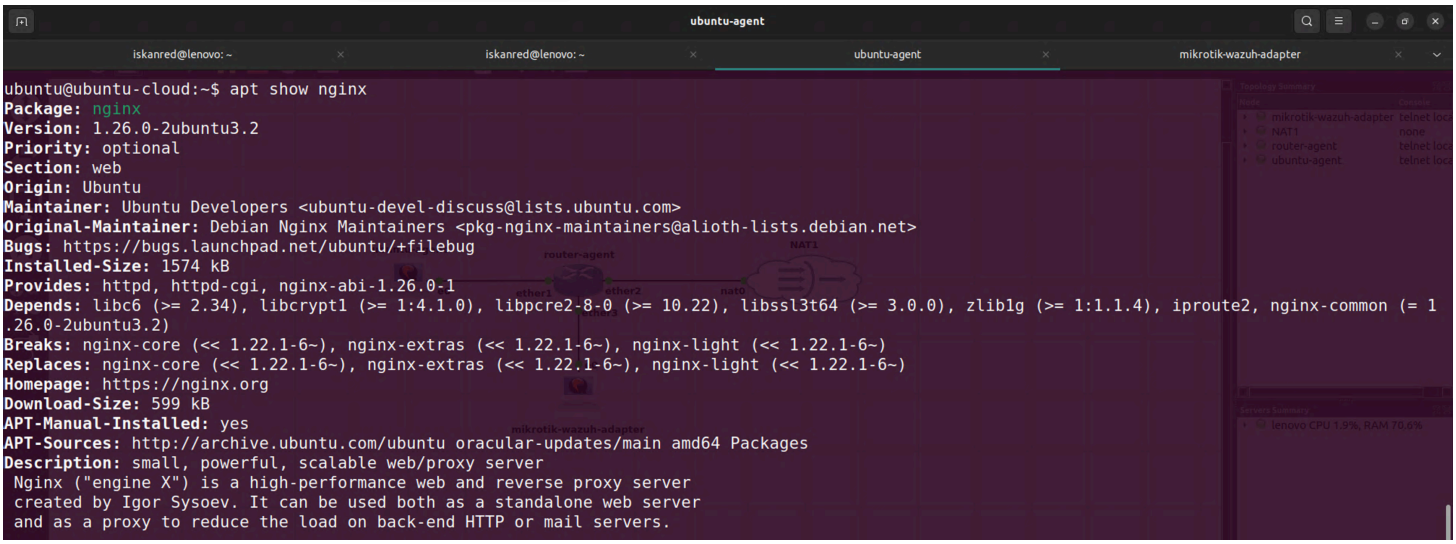
a.

> **Task description**
>
> a. Demonstrate how to block malicious IP addresses from accessing web resources on a web server. To do this, you will set up your web servers on select endpoints within your infrastructure, and try to access them from an external endpoint.

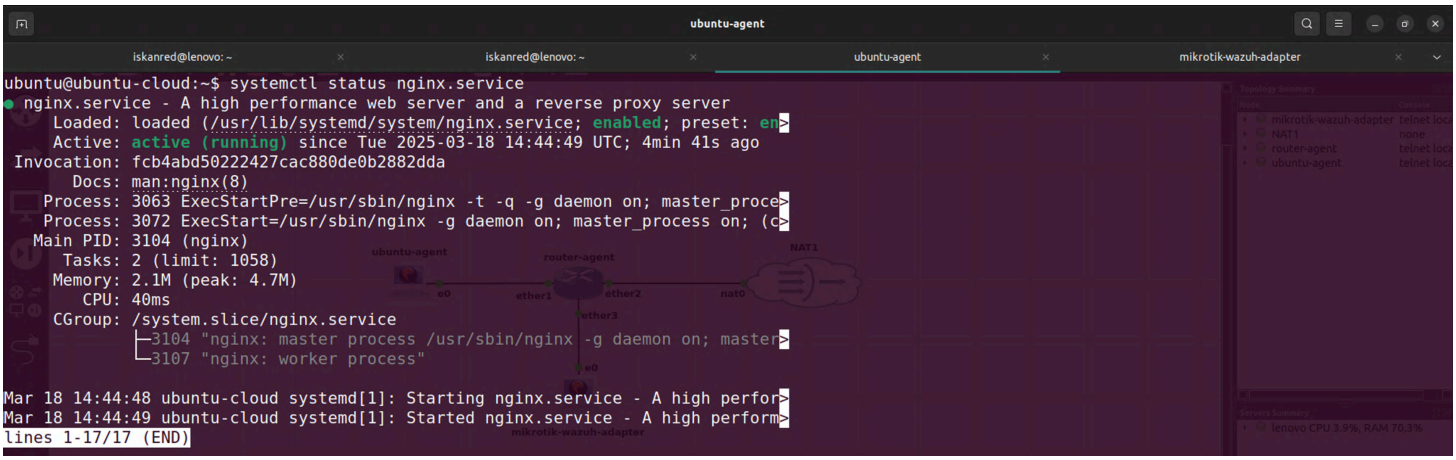
I used the following PoC guide from the official Wazuh website.

Set up web server

- I decided to use `ubuntu-agent` machine as a web server
- I installed NGINX on the `ubuntu-agent` machine



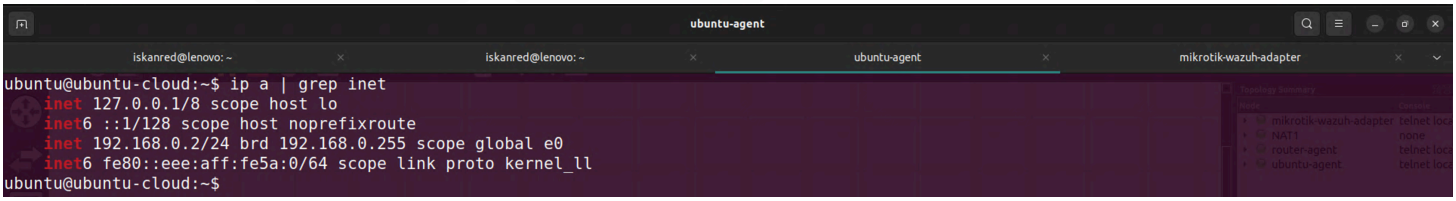
- And checked if it works



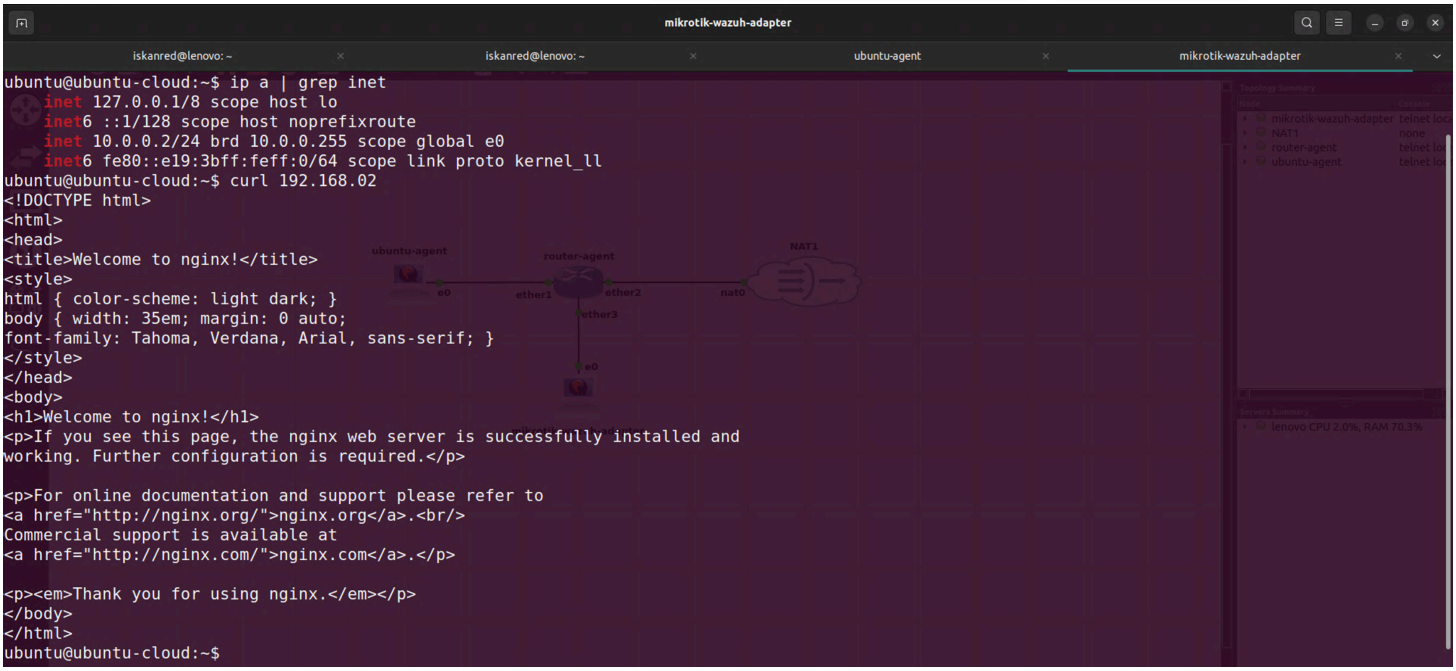
- Finally, I was able to access NGINX main page on 80 port

Access web server from malicious IP address

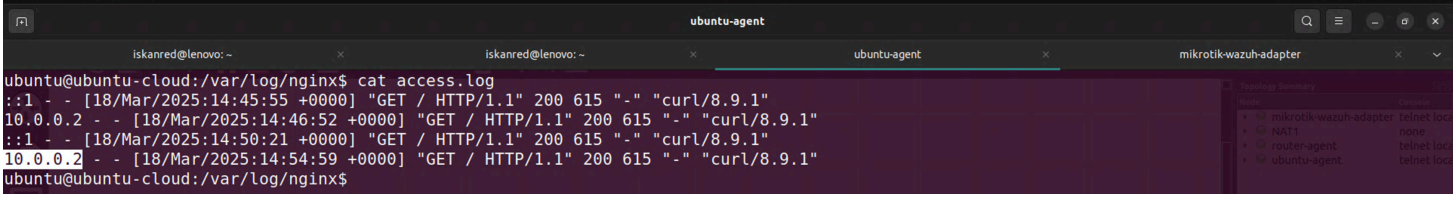
- I decided to make `mikrotik-wazuh-adapter` machine malicious because it should not check my `ubuntu-agent` machine that is connected to the same router. Adapter's main task is just to transfer logs from the `router-agent` to the Wazuh Manager Server.
- IP address of the `ubuntu-agent` is `192.168.0.2/24`



- I was able to access the web server from the `mikrotik-wazuh-adapter` (`10.0.0.2/24`)



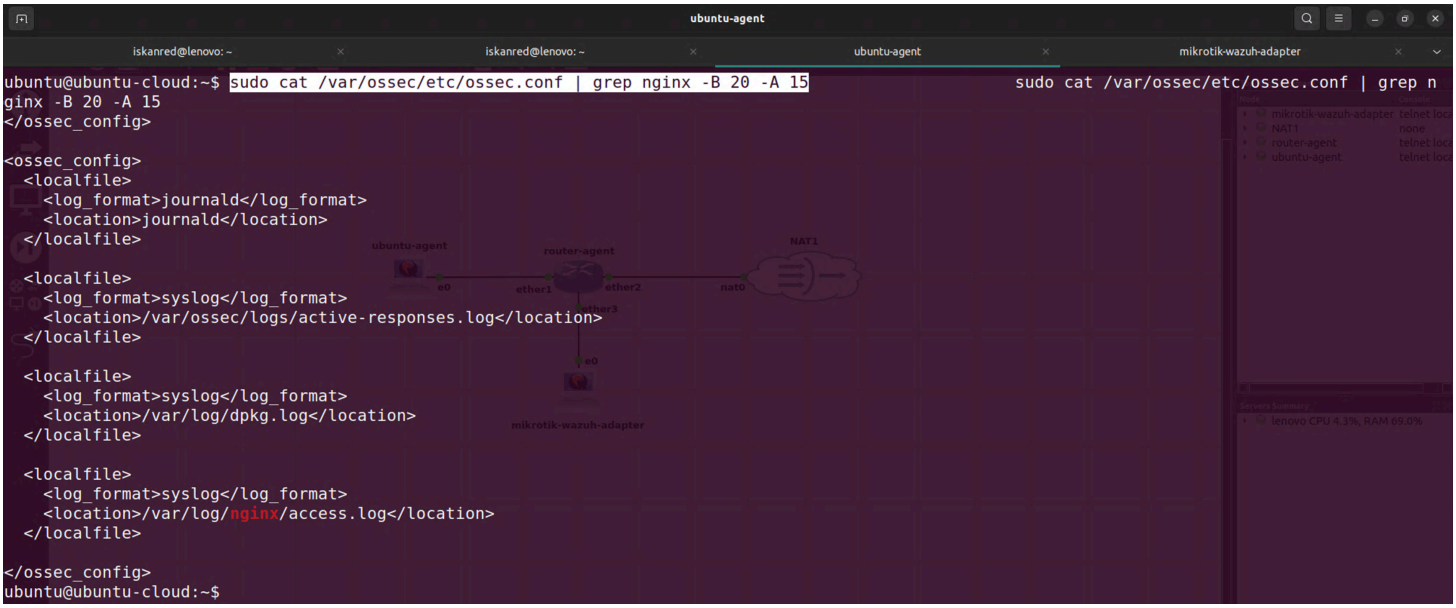
- And we can see in the NGINX access logs that `10.0.0.2` accessed the web server on `ubuntu-agent` machine



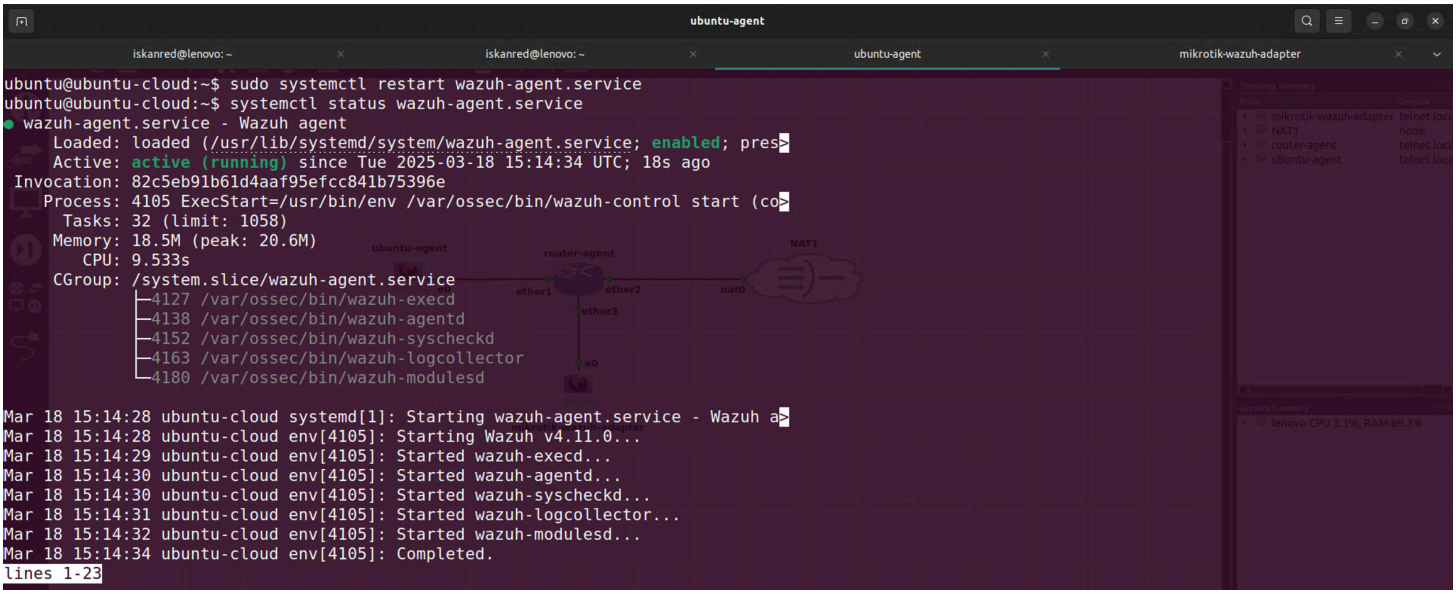
Send NGINX access logs

- To block IP to access the web server we firstly need to recognise it, so Wazuh should have NGINX accesslogs
- To configure NGINX access logs to be sent to the Wazuh Manger Server I added `/var/log/nginx/access.log` as a local source of logs in the `var/ossec/etc/ossec.conf`

```
  <localfile>
    <log_format>syslog</log_format>
    <location>/var/log/nginx/access.log</location>
  </localfile>
```



- Then I need to restart the `wazuh-agent` service



Block malicious IP address

- After things done I connected back to the Wazuh Manager Server

```
~ — nafikov@wazuh-server-ubuntu: ~ — ssh -i .ssh/wazuh_server nafikov@89.169.173.179
nafikov@wazuh-server-ubuntu:~$ docker exec -it single-node-wazuh.manager-1 bash
bash-5.2#
```

- I installed `wget` using `yum` package manager

```
~ — nafikov@wazuh-server-ubuntu: ~ — ssh -i .ssh/wazuh_server nafikov@89.169.173.179
bash-5.2# yum update
Last metadata expiration check: 1:32:06 ago on Tue Mar 18 14:03:03 2025.
Dependencies resolved.
Nothing to do.
Complete!
bash-5.2# yum install -y wget
Last metadata expiration check: 1:32:16 ago on Tue Mar 18 14:03:03 2025.
Dependencies resolved.
================================================================================
 Package                 Architecture          Version                  Repository                Size
================================================================================
Installing:
 wget                    x86_64                1.21.3-1.amzn2023.0.4    amazonlinux               779 k
Installing dependencies:
 gnutls                  x86_64                3.8.0-381.amzn2023.0.7   amazonlinux               1.1 M
 libmetalink             x86_64                0.1.3-14.amzn2023.0.2    amazonlinux                31 k
 nettle                  x86_64                3.8-1.amzn2023.0.2       amazonlinux               417 k

Transaction Summary
================================================================================
Install  4 Packages

Total download size: 2.3 M
Installed size: 7.2 M
Downloading Packages:
(1/4): libmetalink-0.1.3-14.amzn2023.0.2.x86_64.rpm      95 kB/s |  31 kB     00:00
(2/4): gnutls-3.8.0-381.amzn2023.0.7.x86_64.rpm         2.2 MB/s | 1.1 MB     00:00
(3/4): nettle-3.8-1.amzn2023.0.2.x86_64.rpm              799 kB/s | 417 kB     00:00
(4/4): wget-1.21.3-1.amzn2023.0.4.x86_64.rpm             3.7 MB/s | 779 kB     00:00
--------------------------------------------------------------------------------
Total                                                    1.7 MB/s | 2.3 MB     00:01
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                                        1/1
  Installing       : nettle-3.8-1.amzn2023.0.2.x86_64                       1/4
  Installing       : gnutls-3.8.0-381.amzn2023.0.7.x86_64                   2/4
  Installing       : libmetalink-0.1.3-14.amzn2023.0.2.x86_64               3/4
  Installing       : wget-1.21.3-1.amzn2023.0.4.x86_64                      4/4
  Running scriptlet: wget-1.21.3-1.amzn2023.0.4.x86_64                      4/4
  Verifying        : gnutls-3.8.0-381.amzn2023.0.7.x86_64                   1/4
  Verifying        : libmetalink-0.1.3-14.amzn2023.0.2.x86_64               2/4
  Verifying        : nettle-3.8-1.amzn2023.0.2.x86_64                       3/4
  Verifying        : wget-1.21.3-1.amzn2023.0.4.x86_64                      4/4

Installed:
  gnutls-3.8.0-381.amzn2023.0.7.x86_64      libmetalink-0.1.3-14.amzn2023.0.2.x86_64      nettle-3.8-1.amzn2023.0.2.x86_64      wget-1.21.3-1.amzn2023.0.4.x86_64

Complete!
bash-5.2#
```

- I downloaded [AlienVault IP reputation database](#) from GitHub:

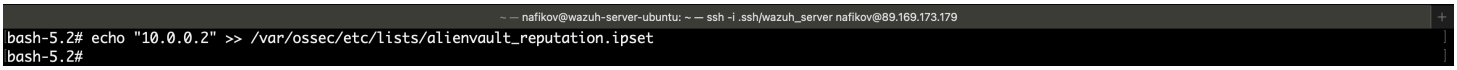
```
wget https://raw.githubusercontent.com/firehol/blocklist-ipsets/master/alienvault_reputation.ipset -O /var/ossec/etc/lists/alienvault_reputation.ipset
```

```
~ — nafikov@wazuh-server-ubuntu: ~ — ssh -i .ssh/wazuh_server nafikov@89.169.173.179
bash-5.2# wget https://raw.githubusercontent.com/firehol/blocklist-ipsets/master/alienvault_reputation.ipset -O /var/ossec/etc/lists/alienvault_reputation.ipset
--2025-03-18 15:40:43--  https://raw.githubusercontent.com/firehol/blocklist-ipsets/master/alienvault_reputation.ipset
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.111.133, 185.199.108.133, 185.199.110.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.111.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 9495 (9.3K) [text/plain]
Saving to: '/var/ossec/etc/lists/alienvault_reputation.ipset'

/var/ossec/etc/lists/alienvault_reputation.ipse 100%[=========================================================================>]   9.27K  --.-KB/s    in 0.003s

2025-03-18 15:40:44 (2.67 MB/s) - '/var/ossec/etc/lists/alienvault_reputation.ipset' saved [9495/9495]
```

- Then I appended `mikrotik-wazuh-adapter`'s IP address to the downloaded instance of the database

```
mikrotik-wazuh-adapter
iskanred@lenovo: ~ | iskanred@lenovo: ~ | ubuntu-agent | mikrotik-wazuh-adapter
ubuntu@ubuntu-cloud:/var/log$ ip a | grep inet
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host noprefixroute
    inet 10.0.0.2/24 brd 10.0.0.255 scope global e0
    inet6 fe80::e19:3bff:feff:0/64 scope link proto kernel_ll
ubuntu@ubuntu-cloud:/var/log$
```

```
echo "10.0.0.2" >> /var/ossec/etc/lists/alienvault_reputation.ipset
```



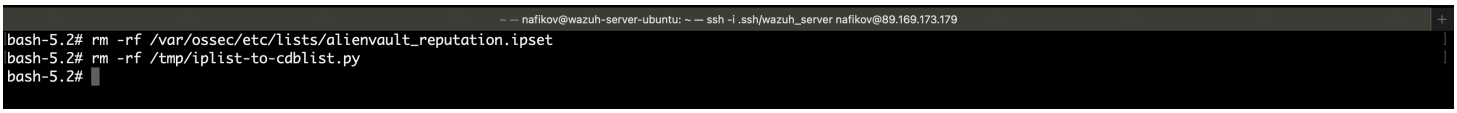
- Afterwards, I downloaded a script to convert the database from `.ipset` to the `.cdb` format from the Wazuh official website.

```
wget https://wazuh.com/resources/iplist-to-cdblist.py -O /tmp/iplist-to-cdblist.py
```

- And converted the database instance using the script

```
/var/ossec/framework/python/bin/python3 /tmp/iplist-to-cdblist.py
/var/ossec/etc/lists/alienvault_reputation.ipset /var/ossec/etc/lists/blacklist-
alienvault
```

- Also, I removed the original IPSET database and the script since they are unnecessary now



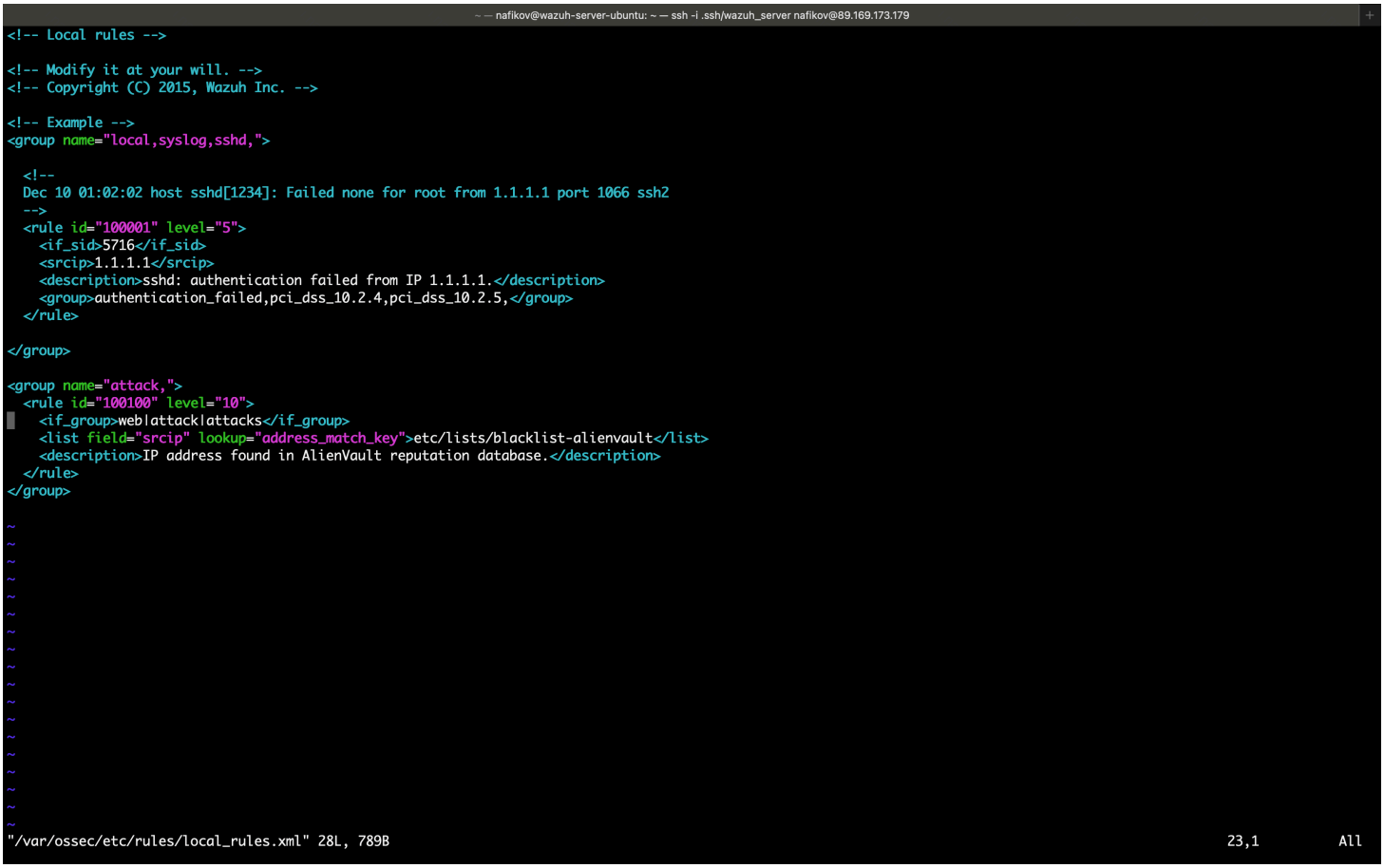
- Then I assigned the right permissions to the generated file for the `wazuh` user

- I added a new custom rule to trigger Wazuh active response script.

```
vim /var/ossec/etc/rules/local_rules.xml
```

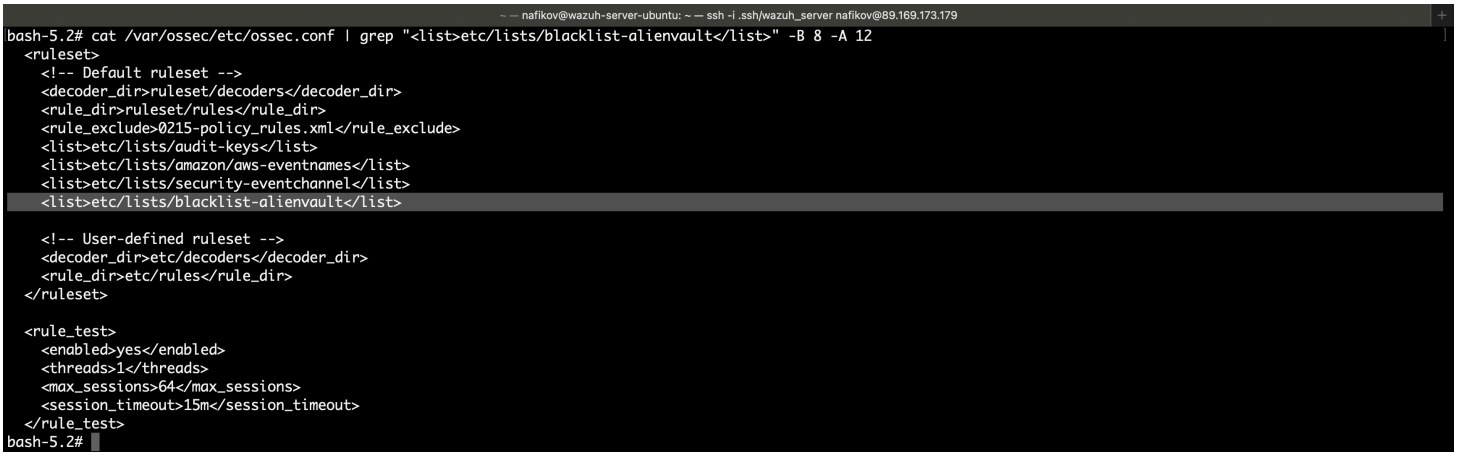
```
<group name="attack,">
  <rule id="100100" level="10">
    <if_group>web|attack|attacks</if_group>
    <list field="srcip" lookup="address_match_key">etc/lists/blacklist-
alienvault</list>
    <description>IP address found in AlienVault reputation database.</description>
```

```
    </rule>
  </group>
```



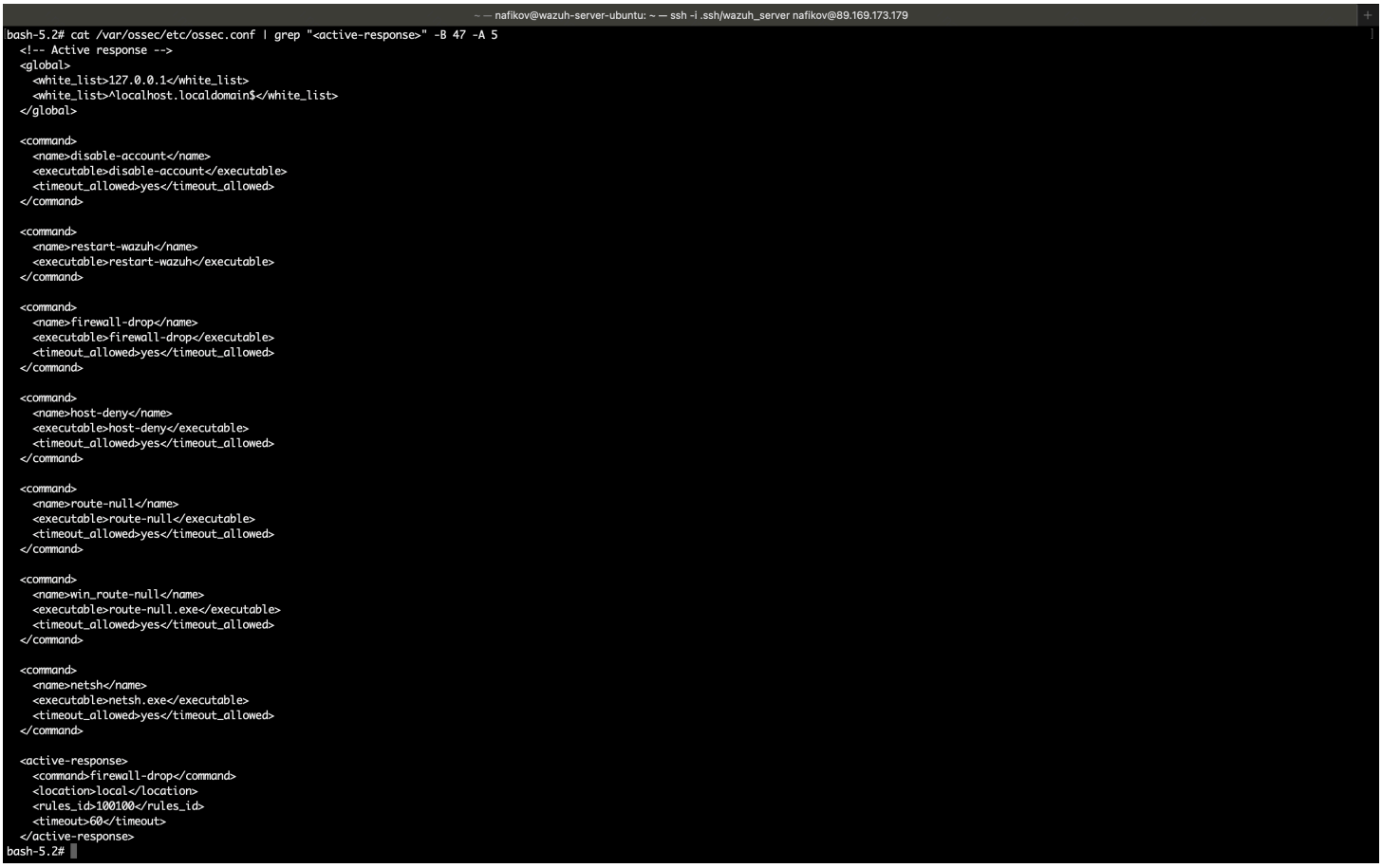
- I added `etc/lists/blacklist-alienvault` **list** entity from this rule to the default ruleset inside the `/var/ossec/etc/ossec.conf`

```
  <ruleset>
  ...
    <list>etc/lists/blacklist-alienvault
  ...
  </ruleset
```

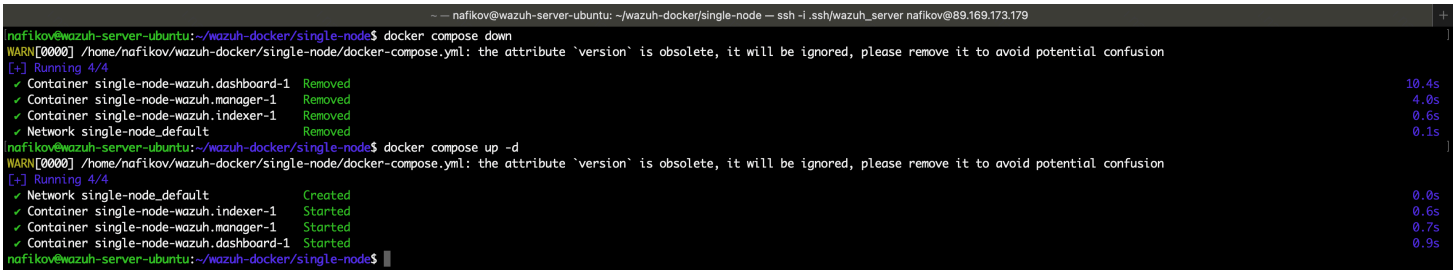


- Finally, I added Active Reponse block to the Wazuh server `/var/ossec/etc/ossec.conf` file that performs `firewall-drop` which integrates with the Ubuntu local `iptables` firewall and drop incoming network connection from the attacker endpoint for the given timeout:

```
<ossec_config>
...
<active-response>
  <command>firewall-drop</command>  <!-- Drop via iptables firewall -->
  <location>local</location>
  <rules_id>100100</rules_id>
  <timeout>60</timeout>             <!-- Block attacker's traffic for 60 seconds -
->
</active-response>
...
</ossec_config>
```

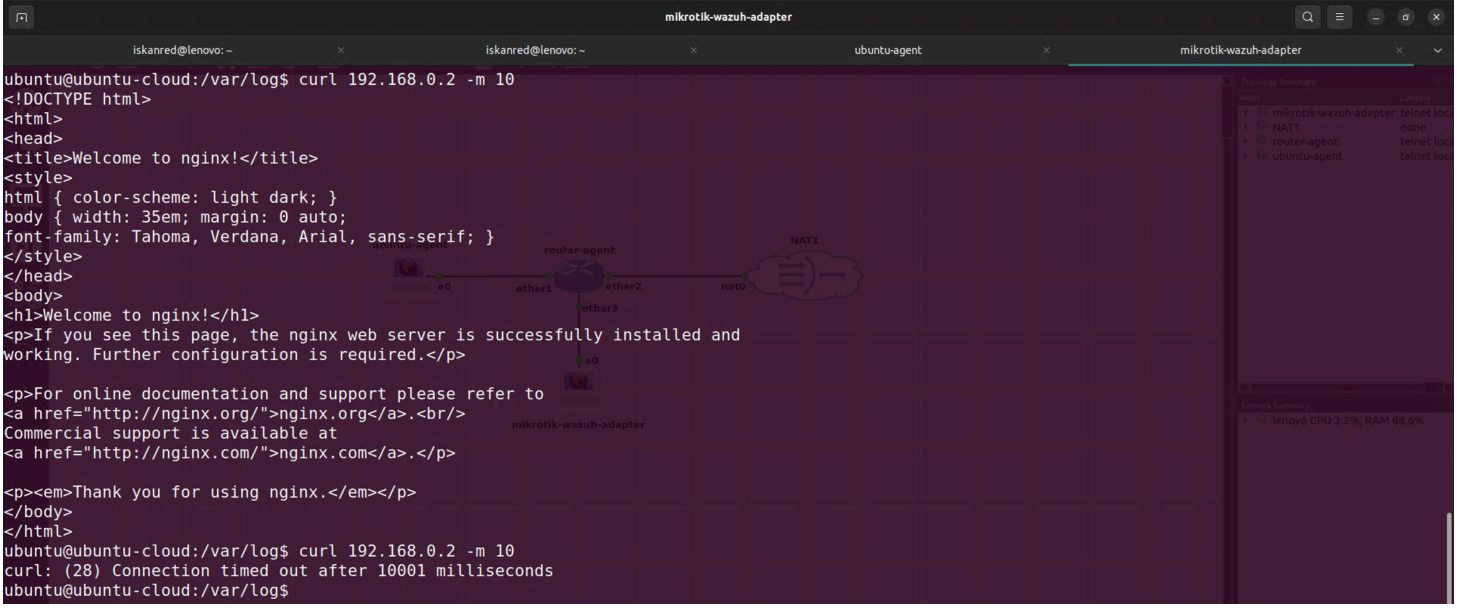


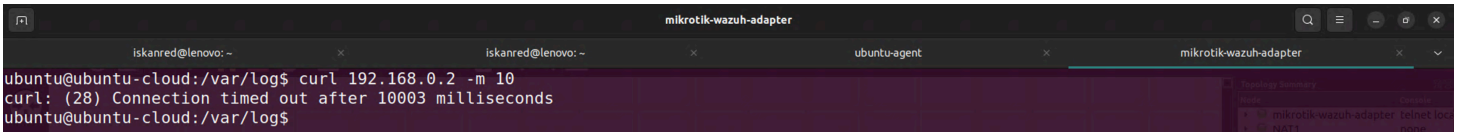
- And restarted the Wazuh



Relusts

- Let's check if things were done correctly
- Firstly, we see that now `mirkotik-wazuh-adapter` has a 60 seconds timeout after accessing the web server on `ubuntu-agent` machine.

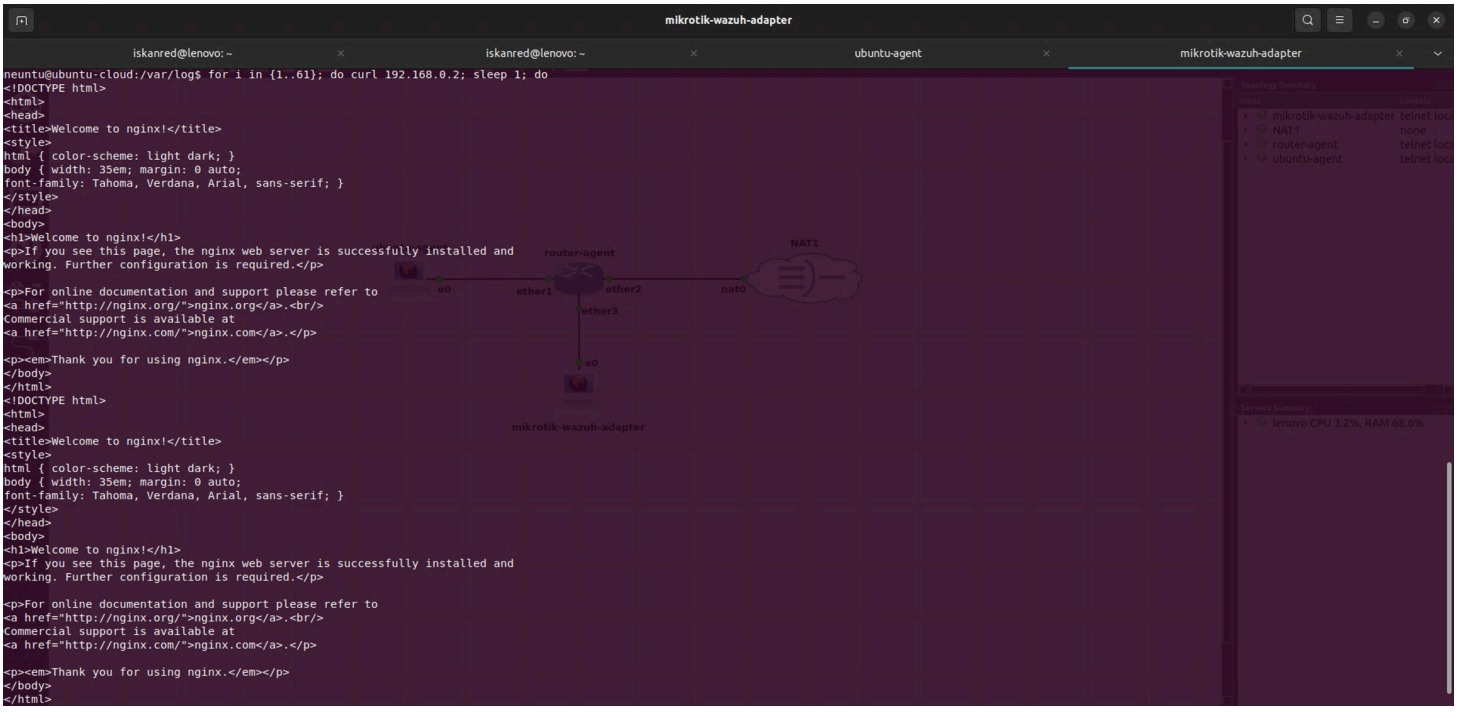




- We can prove it using some script. The script below make requests every second for 61 seconds.

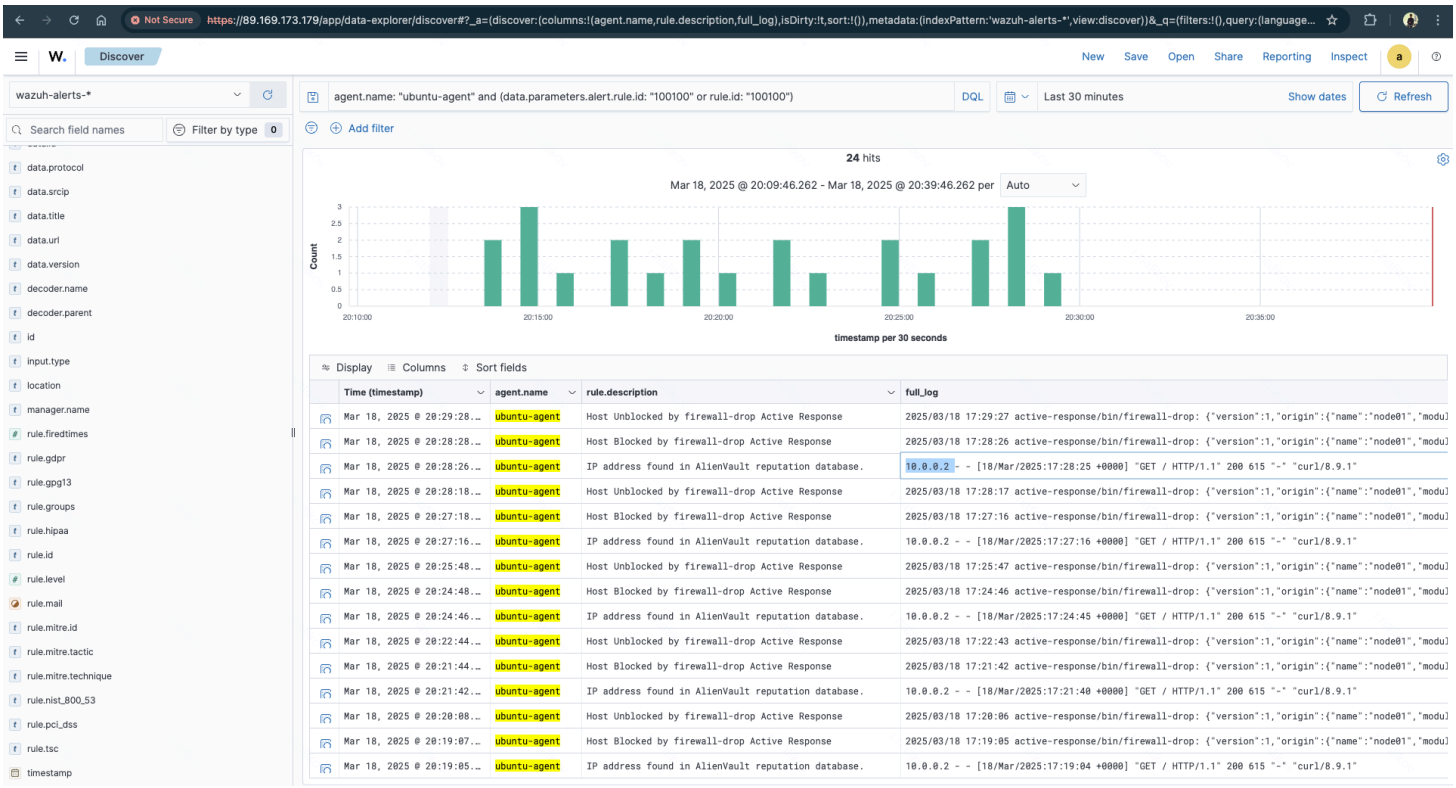
```
for i in {1..61}; do curl 192.168.0.2; sleep 1; done
```

- Running this script we can see that now an attacker can make at maximum $1\ \mathrm{RPM}$ (request per minute).



- We see that in 61 seconds `curl` was successfully executed only 2 times: the 1st one, and the one after 60 seconds.

- In addition, now we can see the rules were triggered many times



- We can see that our `ubuntu-agent` successfully sends logs to the Wazuh Manager Server which detects IP address `10.0.0.2` which is allegedly from AlienVault reputation database and blocks requests from this address for 60 seconds. In a minute it unblocks this IP address.

b.

> **Task description**
>
> b. Simulate a brute force attack against your infrastructure and demonstrate how you would detect the attack on each of the devices within your infrastructure. Are you able to detect the attack? If not, ensure you are able to.

- For simulating a brute force attack I used hydra tool
- Let's start with trying to brute force `ubuntu-agent` and `mikrotik-agent` via SSH protocol making login requests using different credentials.

New rule for Mikrotik

- But before I need to configure another rule for `mikrotik-agent` : **login failure**
- Let's add the following new decoders to the `var/ossec/etc/decoders/mikrotik_decoders.xml` :

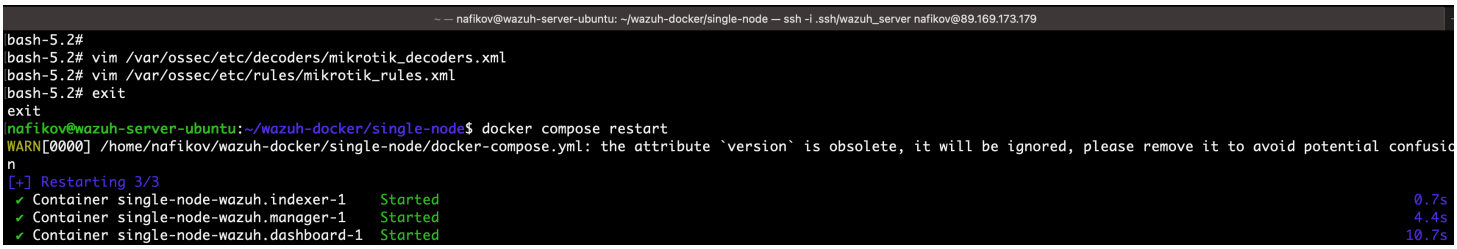
```
<decoder name="mikrotik1">
  <parent>mikrotik</parent>
  <regex type="pcre2">\S+ (\d\d\d\d-\d\d-\d\d+T\d\d:\d\d:\d\d+\+\d\d\:\d\d)
MikroTik (.*) for user (\S+) from (\d+\.\d+\.\d+\.\d+) via (\w+)</regex>
  <order>logtimestamp, action, possible_user, ip_address, protocol</order>
```

```
</decoder>

<decoder name="mikrotik1">
  <parent>mikrotik</parent>
  <regex type="pcre2">\S+ (\d\d\d\d-\d\d-\d\d+T\d\d:\d\d:\d\d+\+\d\d\:\d\d)
MikroTik (.*) message repeated (\d+) times: \[\s*(.*) for user (\S+) from
(\d+\.\d+\.\d+\.\d+) via (\w+)\s*\]</regex>
  <order>logtimestamp, action_message, repeat_times, action, possible_user,
ip_address, protocol</order>
</decoder>
```

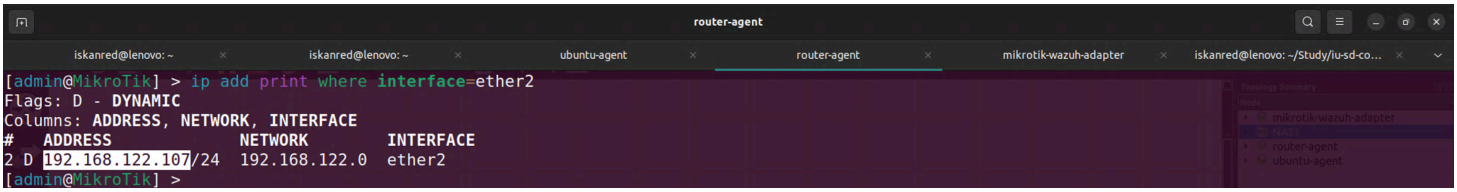
- And new rules to the `var/ossec/etc/rules/mikrotik_rules.xml`

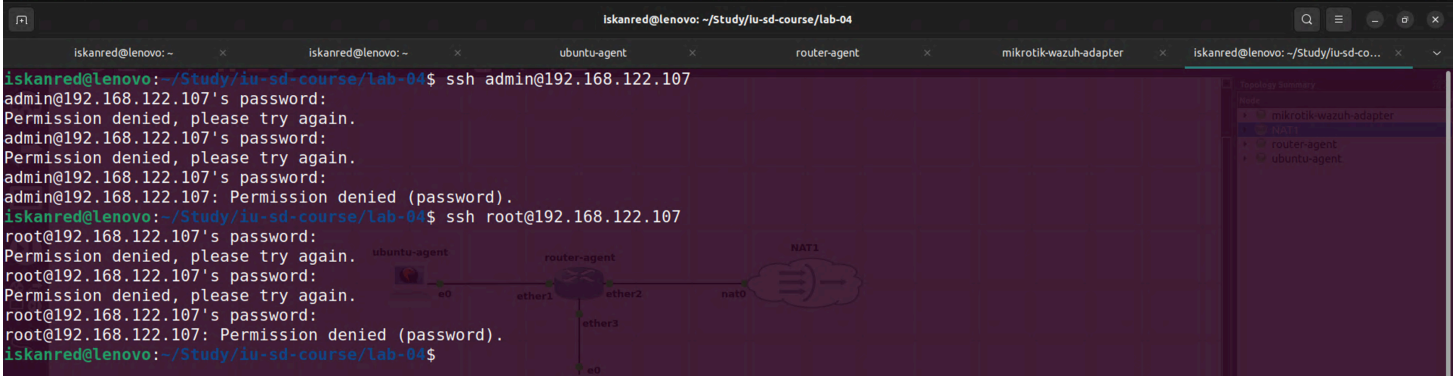
```
  <rule id="110005" level="5">
    <if_sid>110000</if_sid>
    <match>login failure</match>
    <description>Someone tried to login Mikrotik user $(possible_user) from
$(ip_address) via $(protocol)</description>
  </rule>

  <rule id="110006" level="5">
    <if_sid>110000</if_sid>
    <field name="action_message">login</field>
    <match>failure</match>
    <description>Someone tried to login Mikrotik user $(possible_user) from
$(ip_address) via $(protocol) $(repeat_times) times</description>
  </rule>
```

- And reboot the **wazuh-manager**

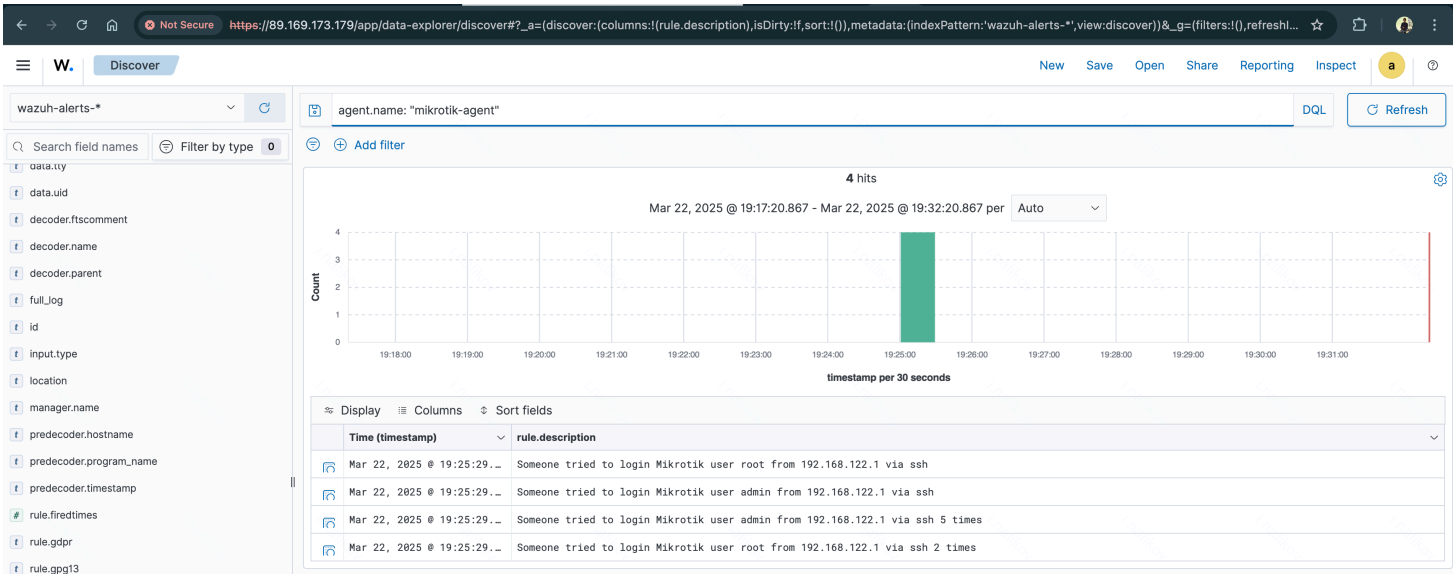


- Now let's try to fail login to the router via SSH from my host machine:



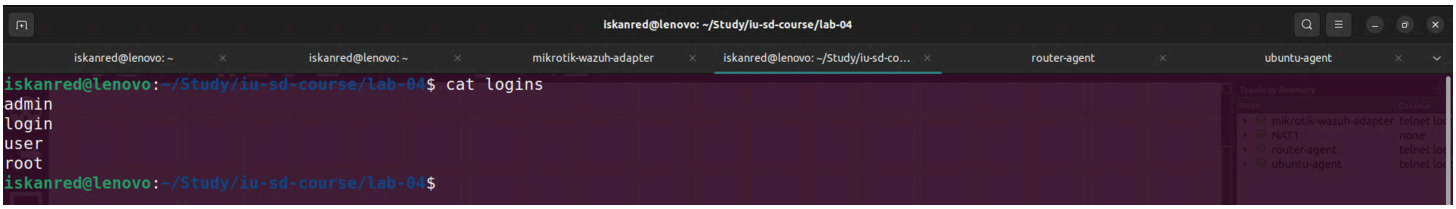


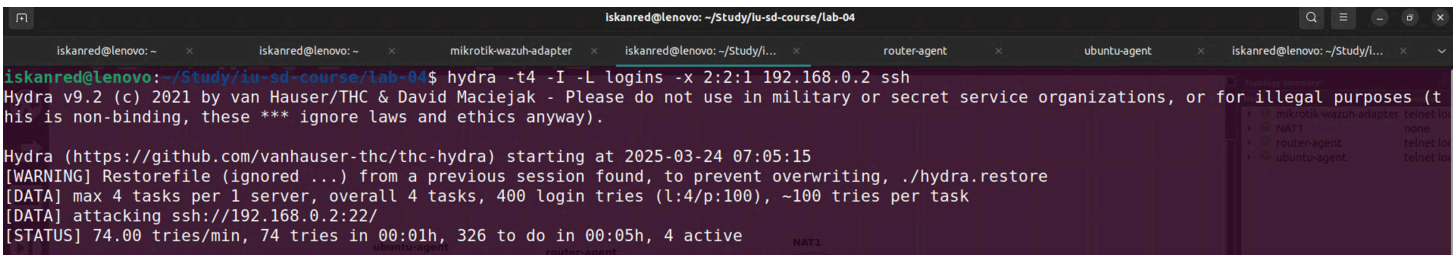
- Finally, we can see these logs on the Wazuh dashboard



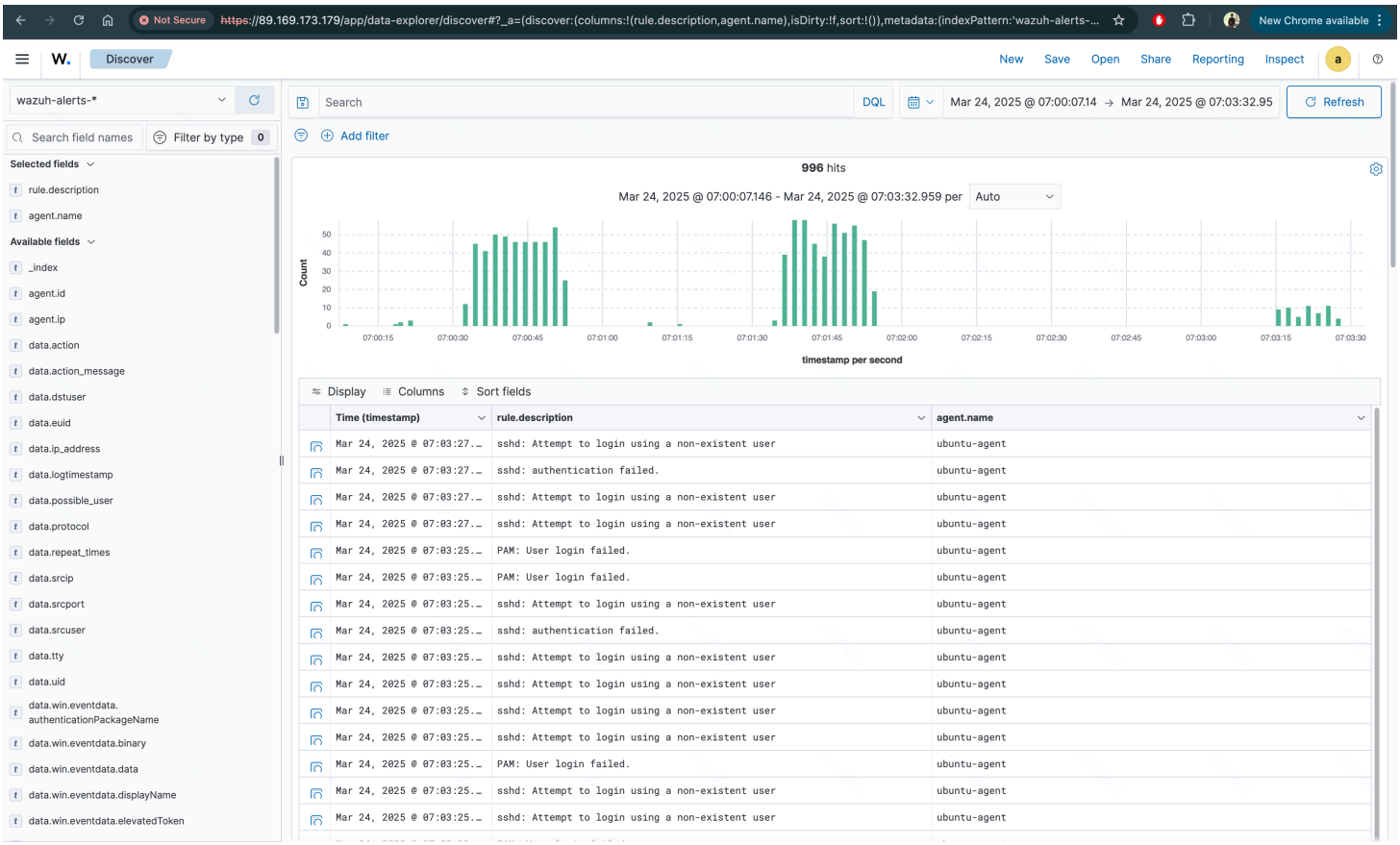
Brute force attack on Ubuntu

- I configured `hydra` to brute force SSH service with some logins provided and passwords of length 2 containing digits only



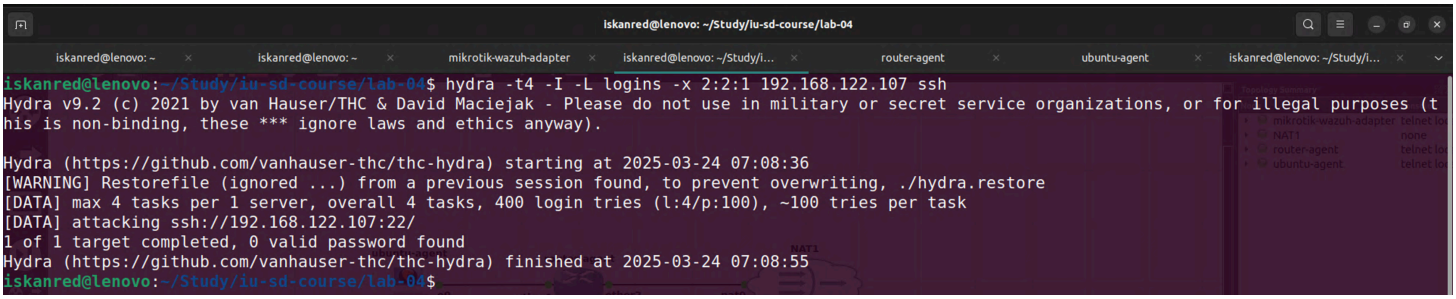


- So I can immediately see these login tries in Wazuh:

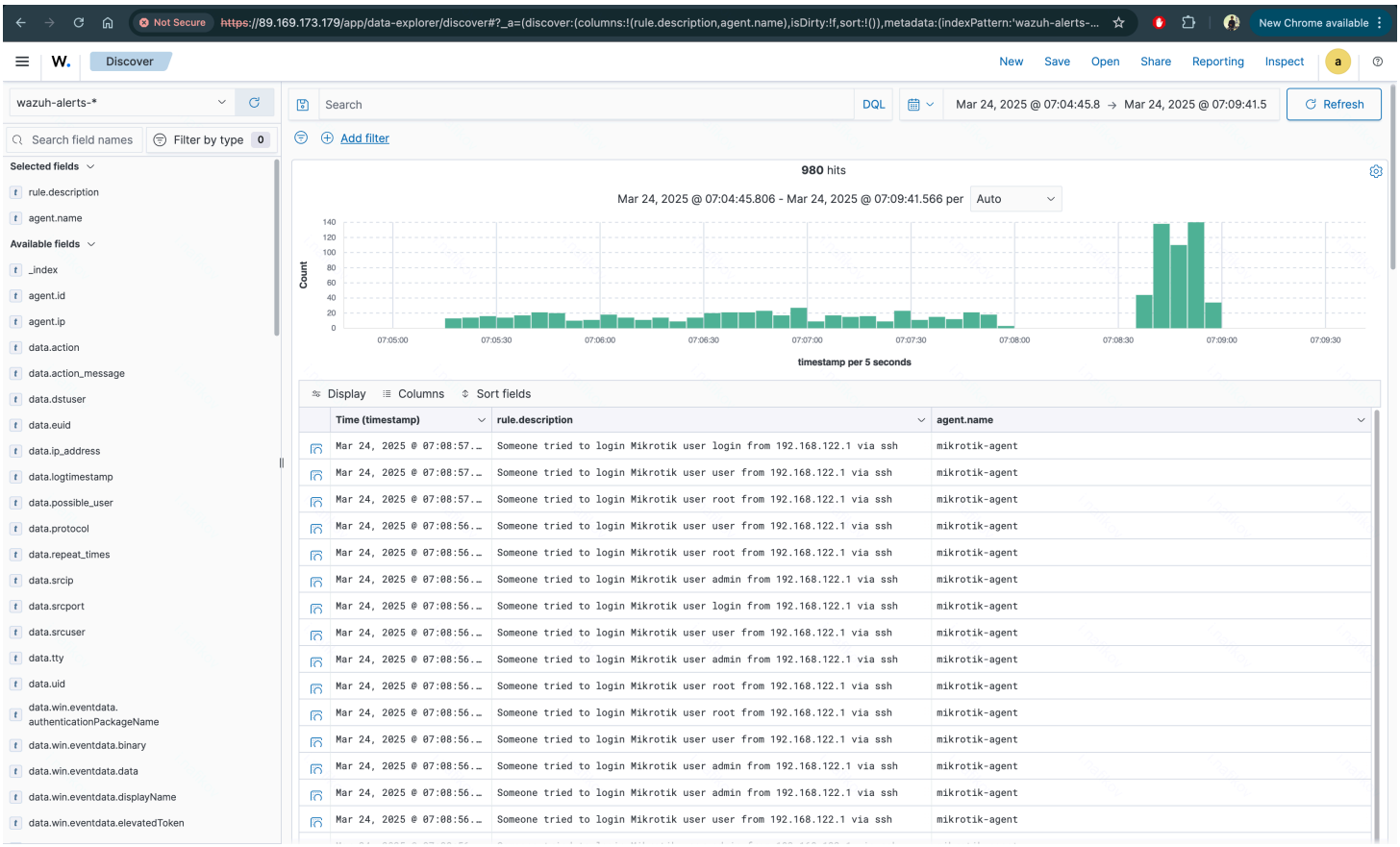


Brute force attack on Mikrotik

- Then I did the same to attack the router which was much faster

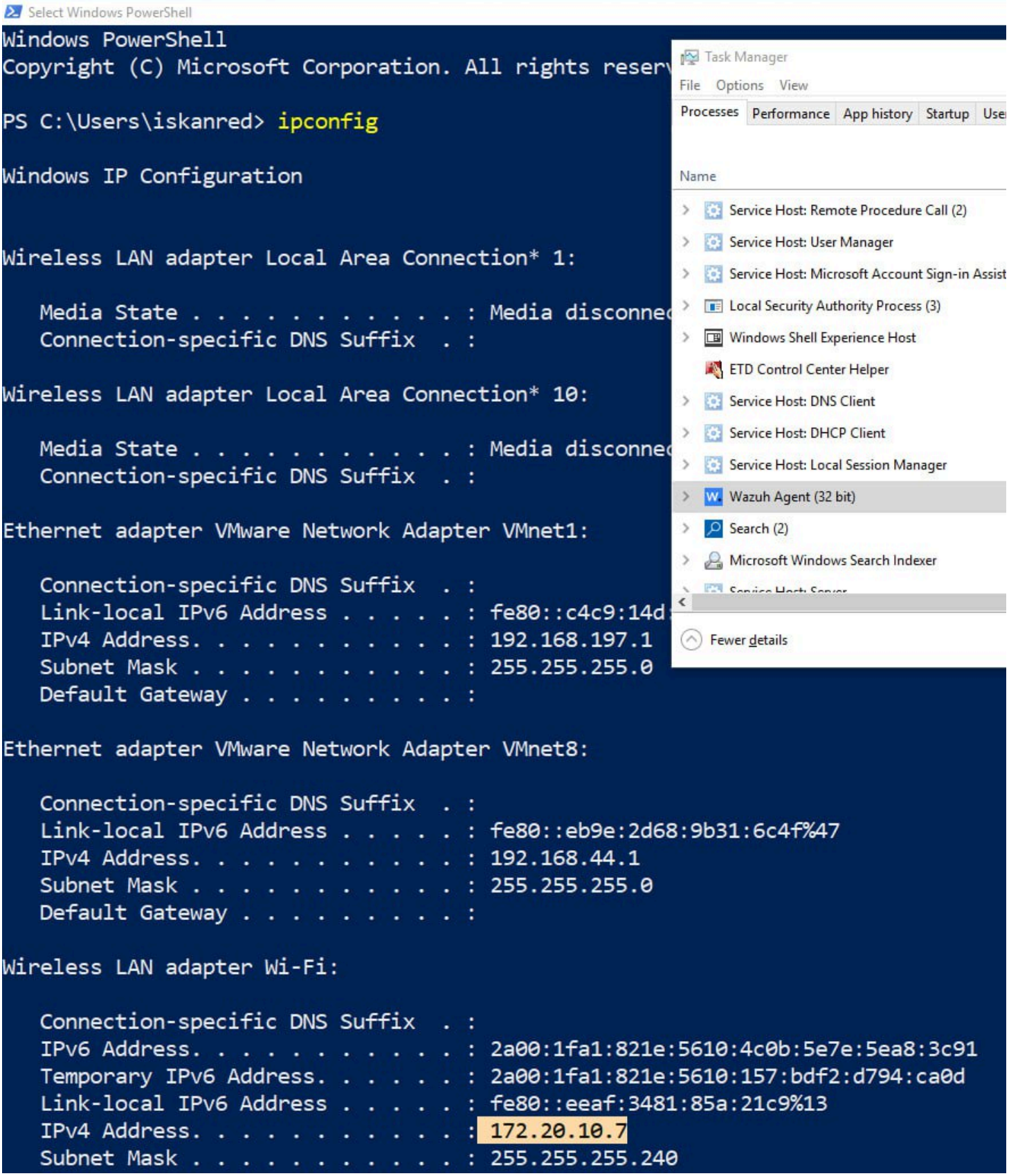


- And again in Wazuh I was able to monitor the attack:

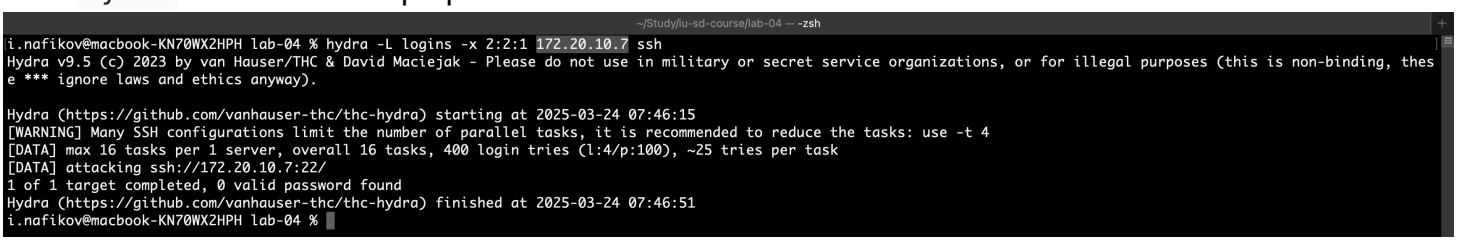


Brute force attack on Windows

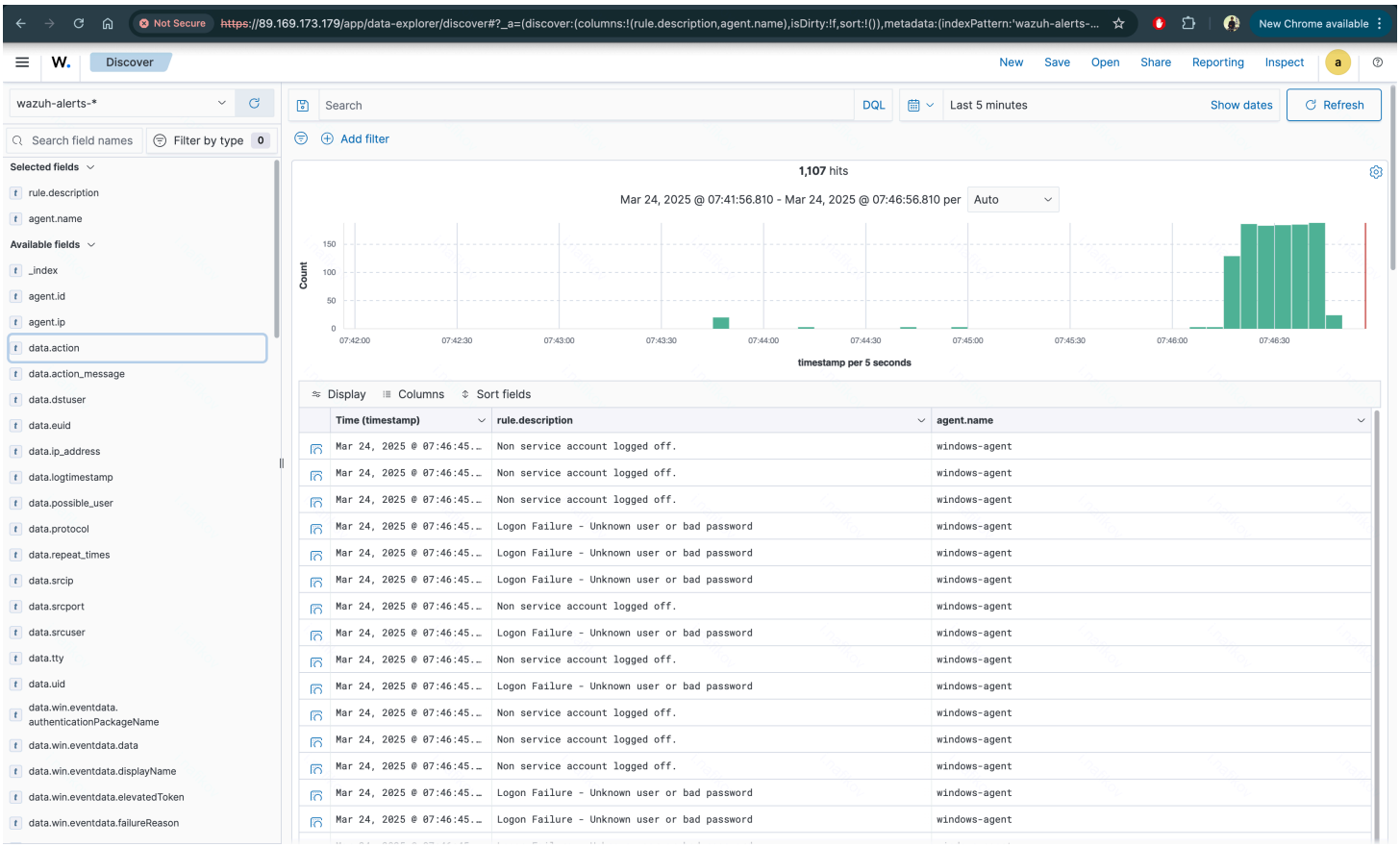
- I did exactly the same to the `windows-agent`



- I ran `hydra` from another laptop inside the same LAN



- And again I could easily notice the attack in the Wazuh Discovery



Part B

Task 5 - SOC integrations

a.

> **Task description**
>
> a. Integrate the SIEM with a case management system of your choice? e.g theHive. Show that you are able to automatically open cases from SIEM alerts.

- I selected TheHive from StrangeBee as an Incident Response Platform and Case Management System.

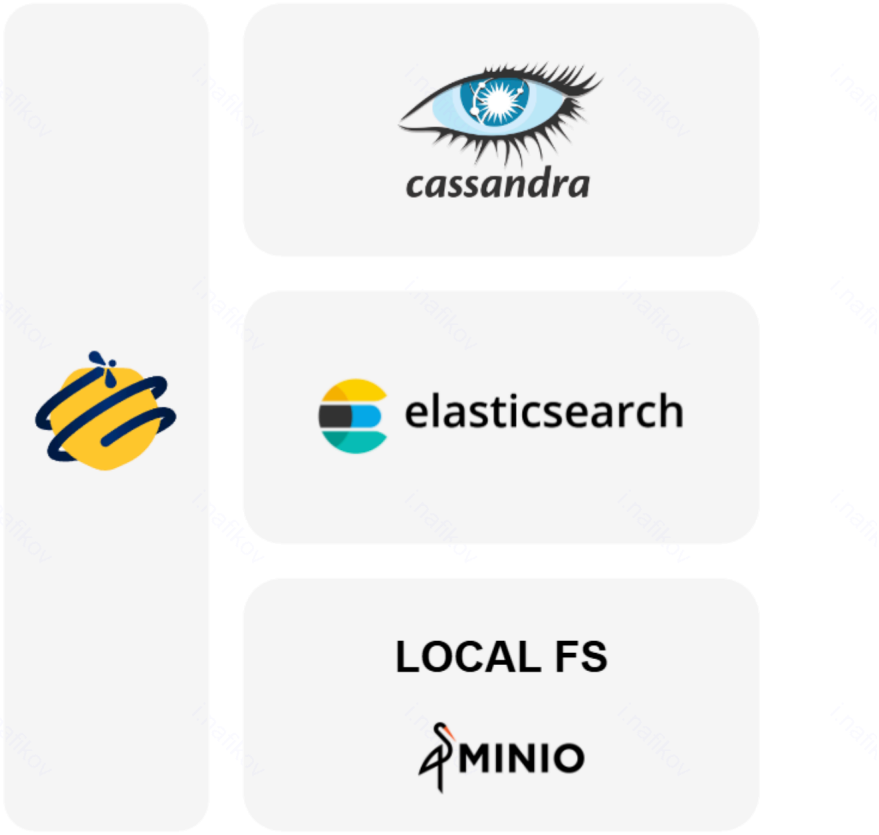
> **TheHive** is an open-source incident response platform designed to assist security teams in managing and responding to security incidents effectively. It provides a collaborative environment where analysts can create, track, and investigate security cases using a structured workflow. TheHive allows for the integration of various security tools, enabling teams to enrich incident data, collaborate in real-time, and maintain documentation for audits. It is particularly suited for

> organizations looking to enhance their incident response capabilities and improve overall threat management.

- TheHive architecture stack is modern and powerful

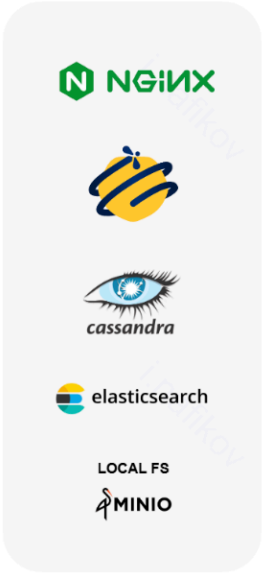
  The essential components of TheHive's setup include:

  - Apache Cassandra for robust data storage, with support for version 4.x.
  - Elasticsearch, serving as a powerful indexing engine, with support for versions 7.x and 8.x.
  - A file storage solution, which can be the local filesystem of the server hosting the application for standalone setups or, NFS or S3 MINIO for clustered environments.



Deploying TheHive

- I decided to deploy TheHive as a standalone server instance which consists of the 5 main components

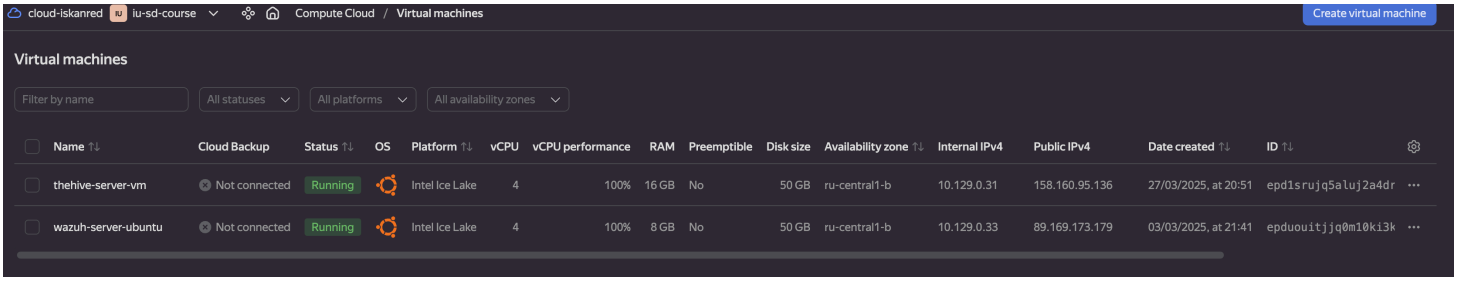


A standalone server setup involves installing all necessary components on a single server:

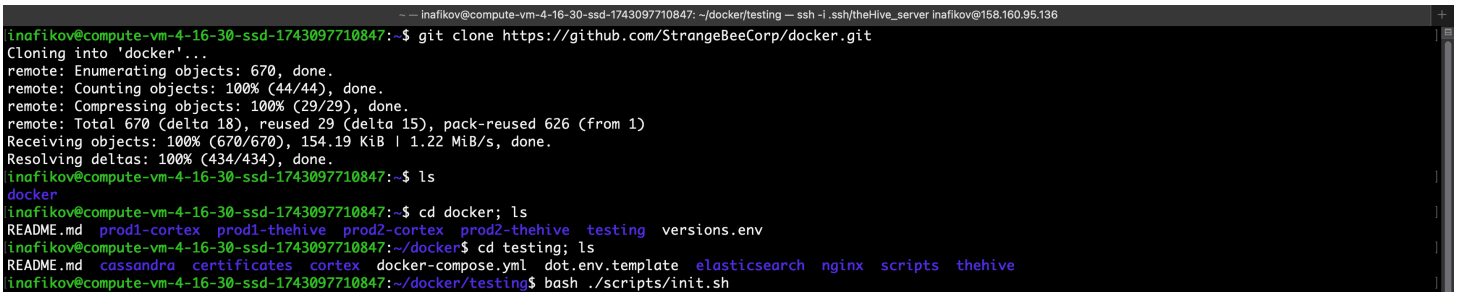
- Cassandra
- Elasticsearch
- File storage on the local filesystem (or MinIO if desired)
- TheHive
- Optional NGINX for managing HTTPS communications

For detailed installation instructions, refer to the step-by-step installation guide.

- First, I deployed to a new VM on Yandex Cloud with higher resources since `theHive` is quite resource-demanding



- Then using the official instruction I deployed TheHive on this VM using Docker: https://github.com/StrangeBeeCorp/docker

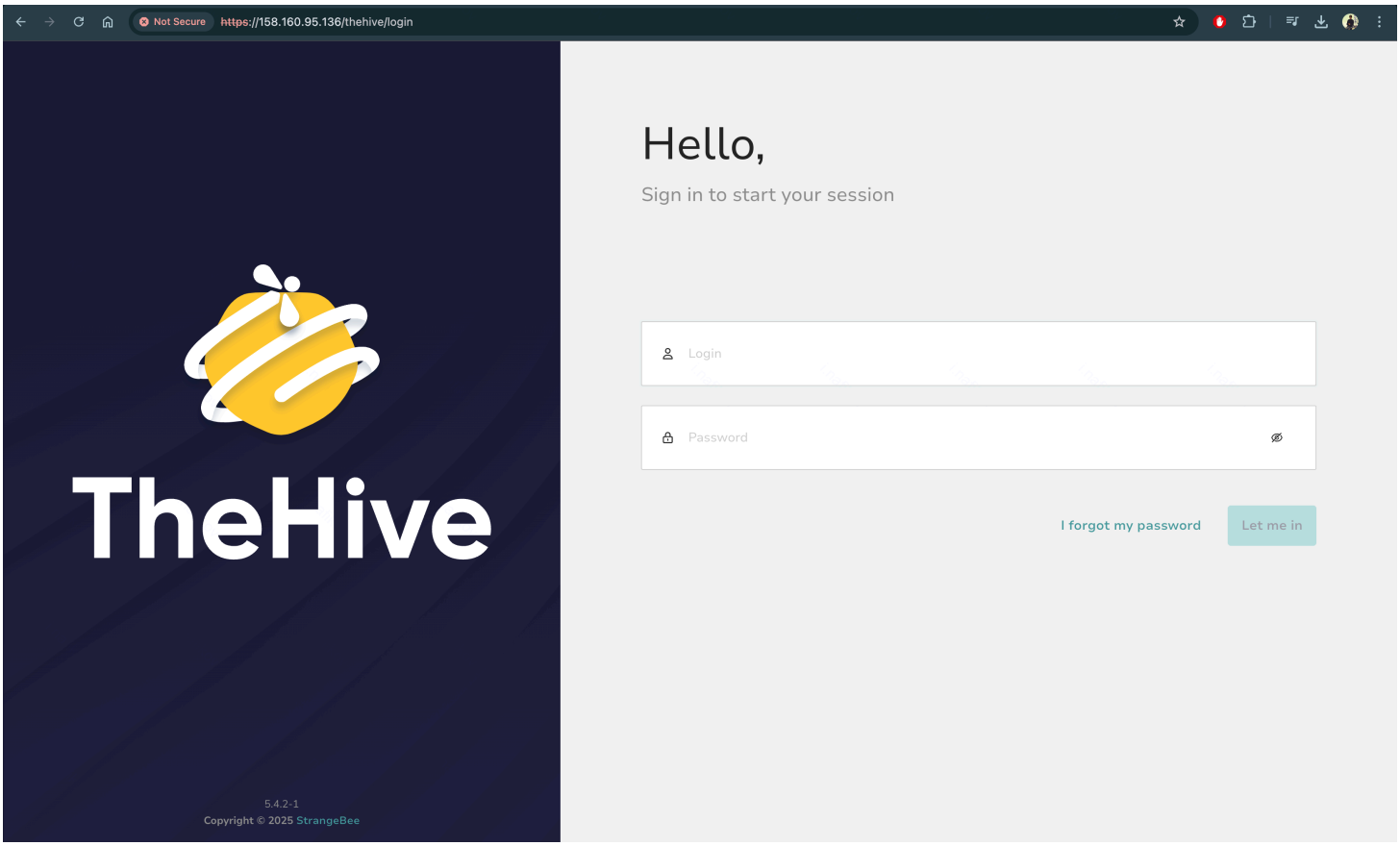


```
[✔] Updated file permissions for: ./cortex/config/logback.xml
[✔] Updated file permissions for: ./cortex/config/secret.conf.template
[✔] Updated file permissions for: ./cortex/neurons/.gitkeep
[✔] Updated file permissions for: ./elasticsearch/logs/.gitkeep
[✔] Updated file permissions for: ./elasticsearch/data/.gitkeep
[✔] Updated file permissions for: ./nginx/certs/.gitkeep
[✔] Updated file permissions for: ./nginx/templates/.gitkeep
[✔] Updated file permissions for: ./nginx/templates/default.conf.template
[✔] Updated file permissions for: ./thehive/logs/.gitkeep
[✔] Updated file permissions for: ./thehive/config/application.conf
[✔] Updated file permissions for: ./thehive/config/logback.xml
[✔] Updated file permissions for: ./thehive/config/secret.conf.template
[✔] Updated file permissions for: ./thehive/data/files/.gitkeep
[✔] Updated file permissions for: ./scripts/test_init_cortex.sh
[✔] Updated file permissions for: ./scripts/output.sh
[✔] Updated file permissions for: ./scripts/init.sh
[✔] Updated file permissions for: ./scripts/test_init_thehive.sh
[✔] Updated file permissions for: ./scripts/generate_certs.sh
[✔] Updated file permissions for: ./scripts/check_permissions.sh
[✔] Updated file permissions for: ./scripts/reset.sh
[✔] Updated file permissions for: ./scripts/restore.sh
[✔] Updated file permissions for: ./scripts/backup.sh
[✔] Updated file permissions for: ./scripts/test_init_applications.sh
[✔] Permissions have been updated for files and directories.
[*] Define the hostname used to connect to this server
Server Name (default: compute-vm-4-16-30-ssd-1743097710847 ): the-hive-server
[*] No custom certificate found.
[*] If you want to provide your custom certificate for the Reverse Proxy, please copy the following files in the ./certificates directory:
    * ./certificates/server.crt
    * ./certificates/server.key
    * ./certificates/ca.pem

[✔] Generating self-signed certificate...
..+....+..+...+....+...+..+.............................+..+.+..+............+.....+.+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*...+.............
..+.......+..........+.+...+.....+.......+..........+...+.+...+...+.........+..........+...+....+...+...+...........+..........+........+...+...+.+.....+.+++++++++++++
++++++++++++++++++++++++++++++++++++++++++++++++++*.+..........+...+.....+......+...+.....+.+..+...+....+...........+..........+.....+++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++
.+..............+..+....+.........+..++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*.+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
*.+.......+.....+..+....+......+..+.+.....+......+.+........+.+.........+.....+....+...+..+.+........+.+.........+..+...........+.+..........+....+.....+.........+......+
...+.....+......+.......+.....+...+.+...+..........+.+..........+..............+....+.........+..+.......+...+.........+.....+...+..+.+......+..........+....+...+...+...
.....+.+...+.................+.+..+...........+............+..........+....+...+..+.+.......+.....+....+..+...+......+...+......+....+..+....+.....+....+..+++++++++++++++
++++++++++++++++++++++++++++++++++++++++++++++++++
-----
[✔] Self-signed certificate generated for compute-vm-4-16-30-ssd-1743097710847.
[✔] Initialisation completed.
[*] Run the following command to start applications:
        $ docker compose up

inafikov@compute-vm-4-16-30-ssd-1743097710847:~/docker/testing$
```

```
~ — inafikov@compute-vm-4-16-30-ssd-1743097710847: ~/docker/testing — ssh -i .ssh/theHive_server inafikov@158.160.95.136
inafikov@compute-vm-4-16-30-ssd-1743097710847:~/docker/testing$ docker compose up -d
[+] Running 37/49
 ⠧ nginx [⣿⣿⣿⣿⣿⣿⣿] 70.12MB / 72.93MB Pulling                                                                                23.5s
 ⠧ cassandra [⣿⣿⣿⣿⣿⣿⣿⣿⣿⣿⣿] 130.4MB / 155.9MB Pulling                                                                        23.5s
 ⠧ thehive [⣿⣿⣿⣿⣿⣿⣿⣿⣿⣿⣿] 610.2MB / 616.3MB Pulling                                                                         23.5s
 ⠧ cortex [⣿⣿⣿⣿⣿⣿⣿]   429MB / 432.7MB Pulling                                                                               23.5s
 ⠧ elasticsearch [⣿⣿⣿⣿ ⣿    ] Pulling                                                                                       23.5s
```

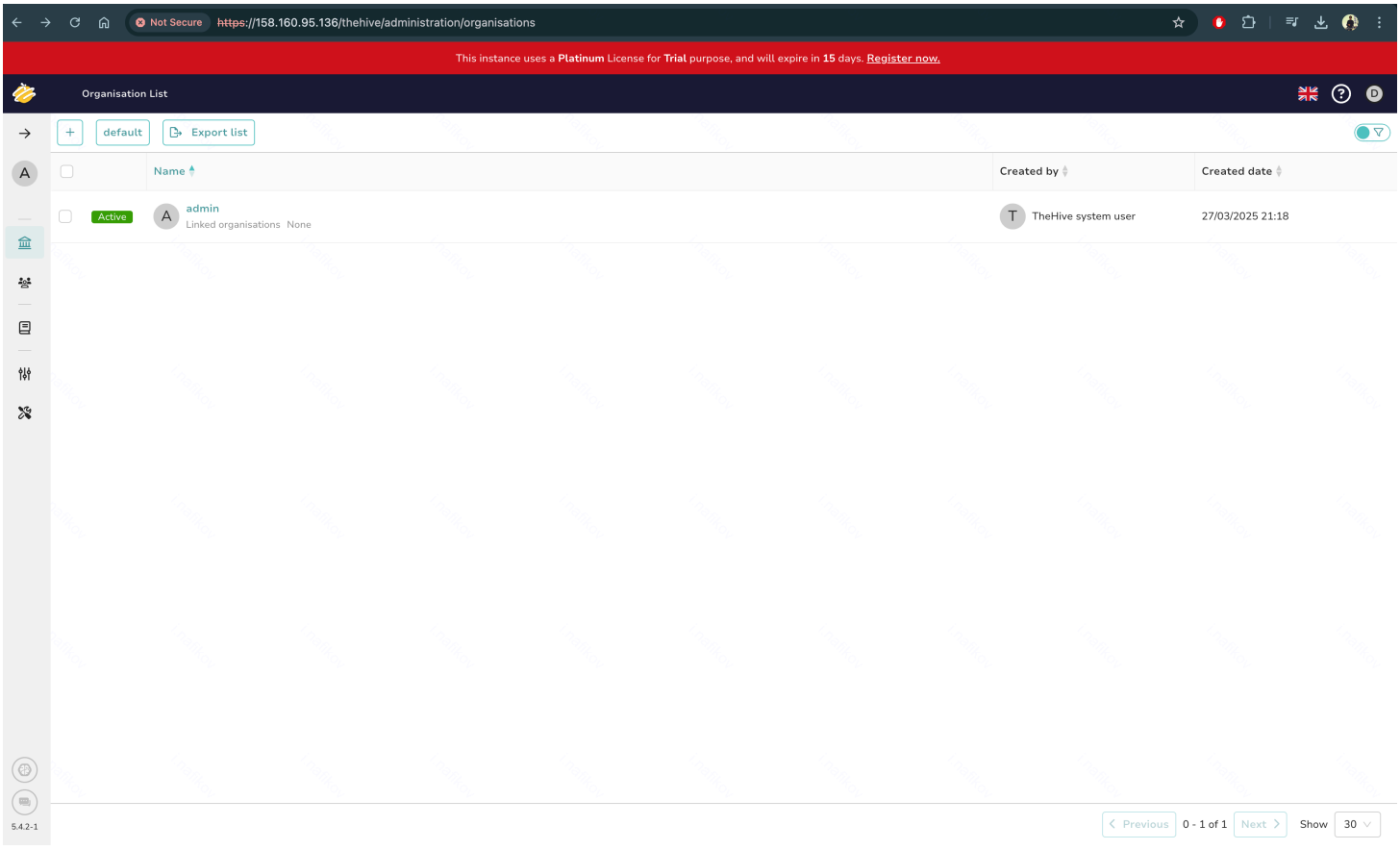
```
~ — inafikov@compute-vm-4-16-30-ssd-1743097710847: ~/docker/testing — ssh -i .ssh/theHive_server inafikov@158.160.95.136
inafikov@compute-vm-4-16-30-ssd-1743097710847:~/docker/testing$ docker compose up -d
[+] Running 49/49
 ✔ nginx Pulled                                                                                                          105.1s
 ✔ cassandra Pulled                                                                                                      147.9s
 ✔ thehive Pulled                                                                                                        103.9s
 ✔ cortex Pulled                                                                                                          76.8s
 ✔ elasticsearch Pulled                                                                                                  170.9s

[+] Running 6/6
 ✔ Network testing_thehive-cortex-network  Created                                                                         7.9s
 ✔ Container cassandra                     Healthy                                                                       127.9s
 ✔ Container elasticsearch                 Healthy                                                                        73.9s
 ✔ Container cortex                        Started                                                                        22.4s
 ✔ Container thehive                       Started                                                                        76.3s
 ✔ Container nginx                         Started                                                                        76.4s
inafikov@compute-vm-4-16-30-ssd-1743097710847:~/docker/testing$ docker ps
CONTAINER ID   IMAGE                          COMMAND                  CREATED              STATUS                             PORTS                                                NAMES
cd415dabc915   nginx:1.27.2                   "/docker-entrypoint.…"   About a minute ago   Up 39 seconds                      80/tcp, 0.0.0.0:443->443/tcp, [::]:443->443/tcp      nginx
f0b38531551f   strangebee/thehive:5.4.2-1     "/opt/thehive/entryp…"   About a minute ago   Up 39 seconds (health: starting)   0.0.0.0:9000->9000/tcp                               thehive
d91410648f18   thehiveproject/cortex:3.1.8-1  "/opt/cortex/entrypo…"   About a minute ago   Up About a minute (healthy)        0.0.0.0:9001->9001/tcp                               cortex
82bc2d561449   cassandra:4.1.7                "docker-entrypoint.s…"   2 minutes ago        Up About a minute (healthy)        7000-7001/tcp, 7199/tcp, 9042/tcp, 9160/tcp          cassandra
80499323e7ef   elasticsearch:7.17.24          "/bin/tini -- /usr/l…"   2 minutes ago        Up About a minute (healthy)        9200/tcp, 9300/tcp                                   elasticsear
ch
inafikov@compute-vm-4-16-30-ssd-1743097710847:~/docker/testing$
```

- Afterwards, I was able to connect to my TheHive instance by the machine's IP address



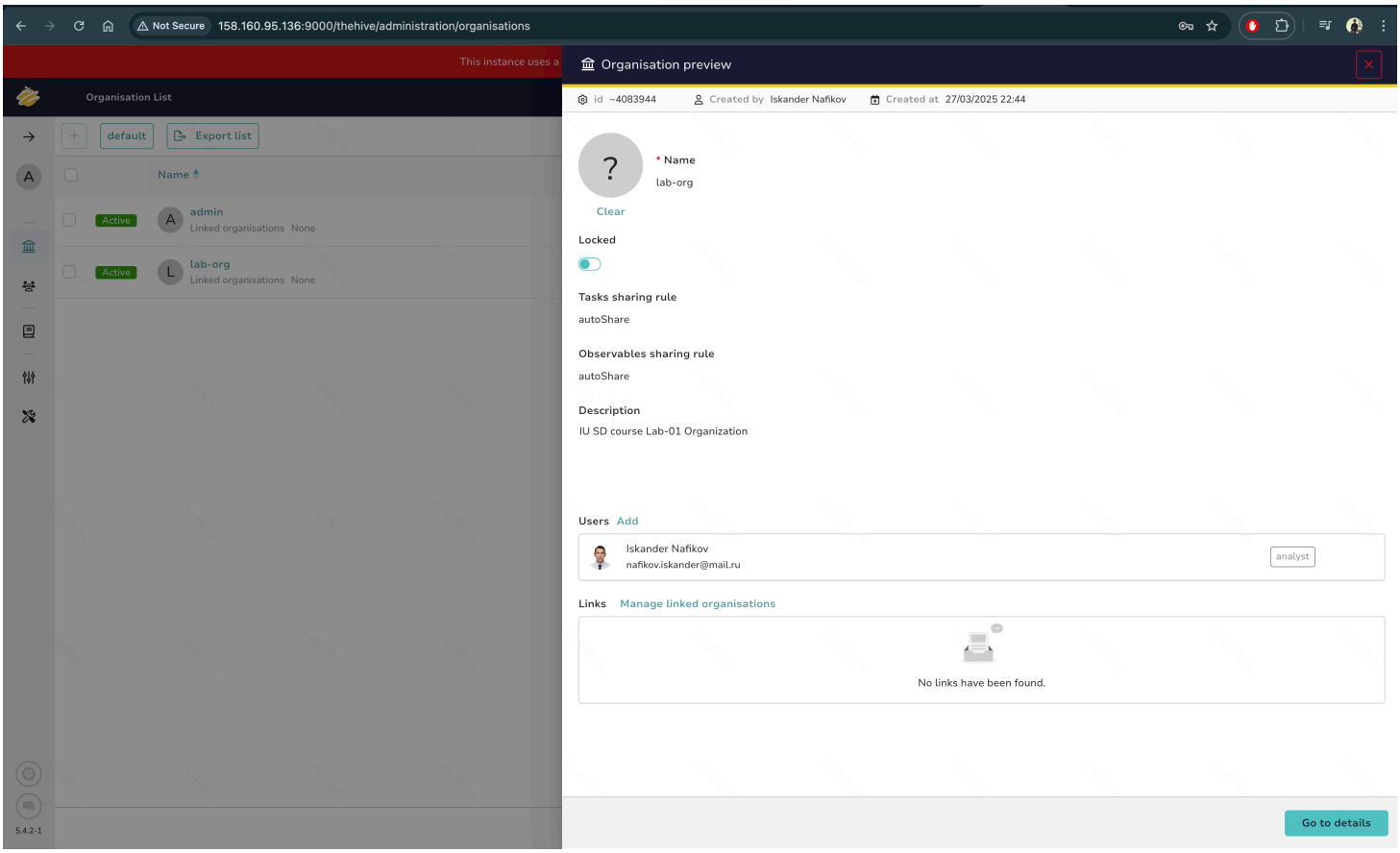
TheHive Configuration

- I logged in using default credentials as an admin

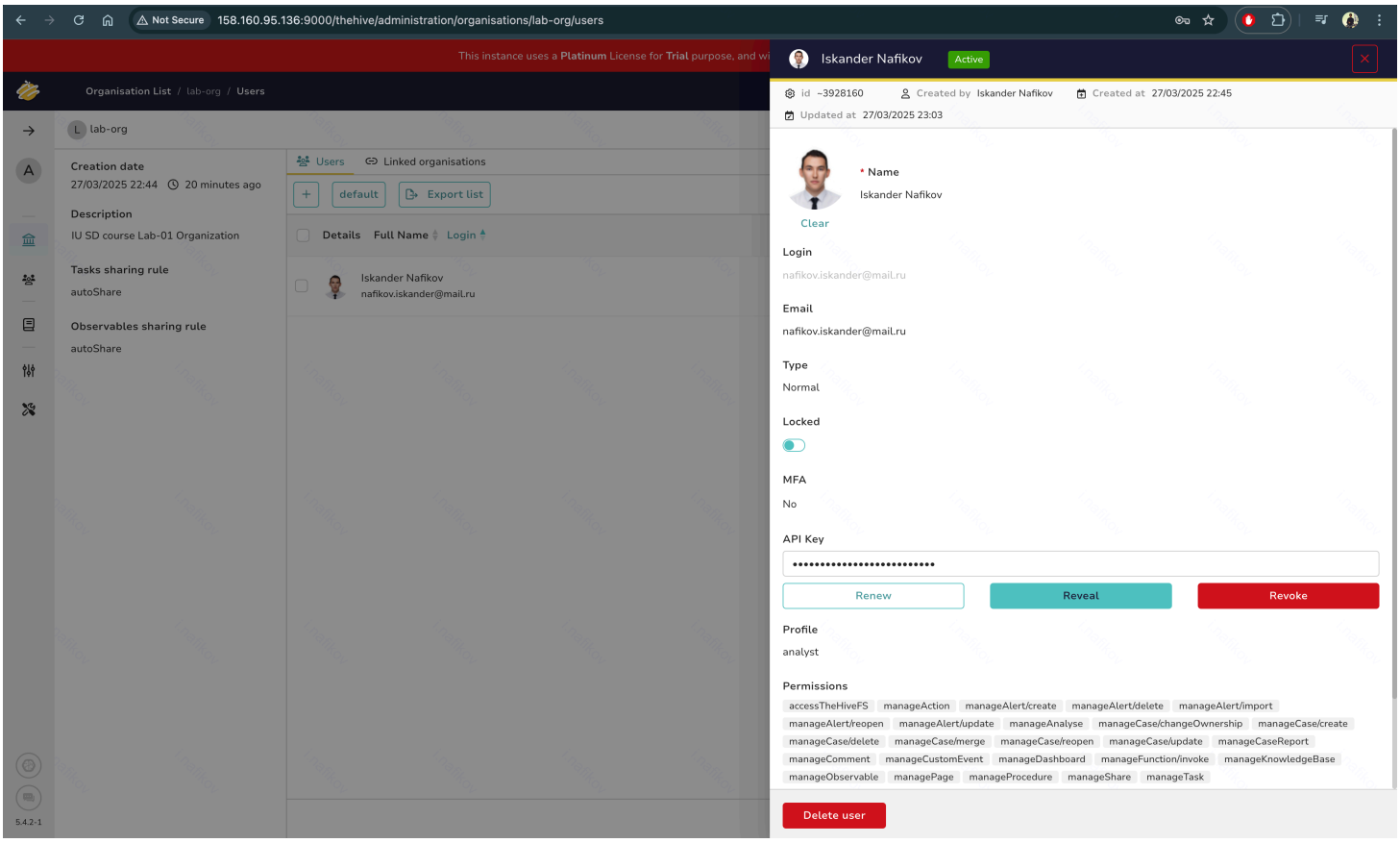


- Here we can see the `admin` organisation with the single `admin` user

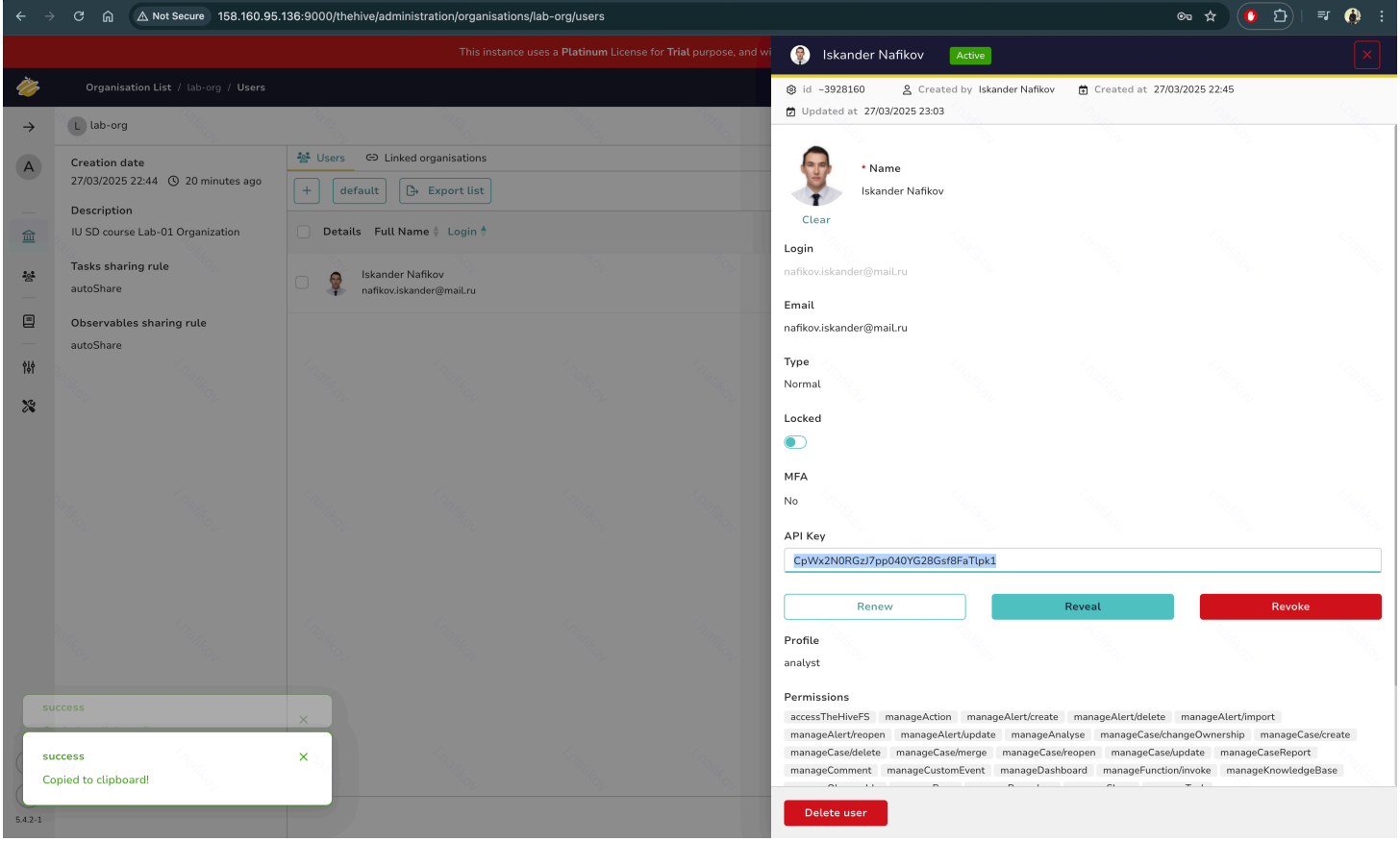
- However, I created a new separate organisation `lab-org`



- And a new user inside this organisation with the role `analyst` which gives rights to write/read alerts inside the organisation using UI and REST API

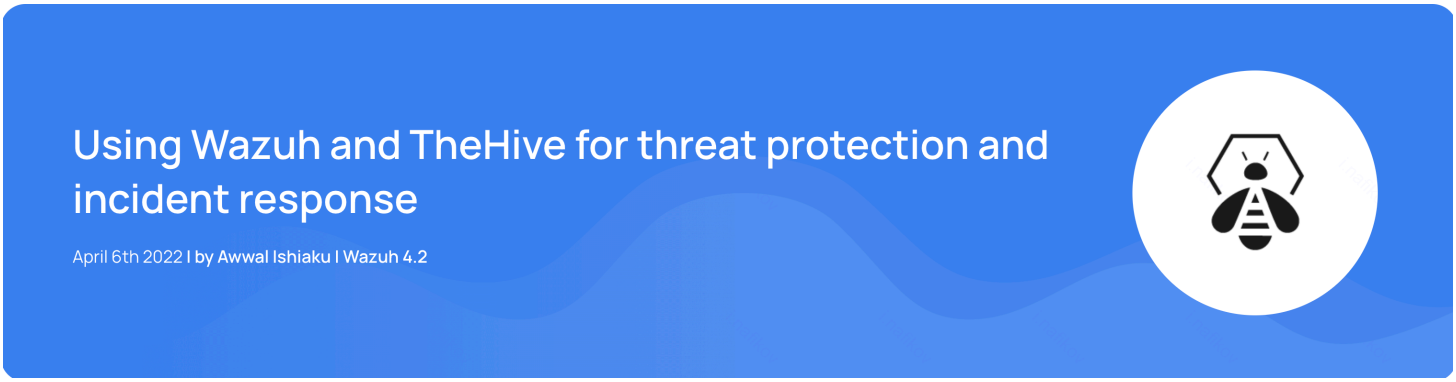


- I created an API key to support integration of Wazuh -> TheHive since Wazuh will call TheHive's API to create an alert when Wazuh rule is triggered



Integration with Wazuh

- To integrate Wazuh with TheHive I used the following instruction from Wazuh blog written by our student and TA, **Awwal Ishiaku**



- I installed `thehive4py` Python module on my Wazuh manager

```
/var/ossec/framework/python/bin/pip3 install thehive4py==1.8.1
```

```
~ — inafikov@compute-vm-4-16-30-ssd-1743097710847: ~/docker/testing — ssh -i .ssh/theHive_server inafikov@158.160.95.136   ~ — nafikov@wazuh-server-ubuntu: ~ — ssh -i .ssh/wazuh_server nafikov@89.169.173.179
nafikov@wazuh-server-ubuntu:~$ docker exec -it single-node-wazuh.manager-1 bash
bash-5.2# /var/ossec/framework/python/bin/pip3 install thehive4py==1.8.1
Collecting thehive4py==1.8.1
  Downloading thehive4py-1.8.1-py3-none-any.whl.metadata (4.3 kB)
Requirement already satisfied: future in /var/ossec/framework/python/lib/python3.10/site-packages (from thehive4py==1.8.1) (0.18.3)
Requirement already satisfied: requests in /var/ossec/framework/python/lib/python3.10/site-packages (from thehive4py==1.8.1) (2.32.2)
Collecting python-magic (from thehive4py==1.8.1)
  Downloading python_magic-0.4.27-py2.py3-none-any.whl.metadata (5.8 kB)
Requirement already satisfied: charset-normalizer<4,>=2 in /var/ossec/framework/python/lib/python3.10/site-packages (from requests->thehive4py==1.8.1) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in /var/ossec/framework/python/lib/python3.10/site-packages (from requests->thehive4py==1.8.1) (3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in /var/ossec/framework/python/lib/python3.10/site-packages (from requests->thehive4py==1.8.1) (2.2.2)
Requirement already satisfied: certifi>=2017.4.17 in /var/ossec/framework/python/lib/python3.10/site-packages (from requests->thehive4py==1.8.1) (2024.7.4)
Downloading thehive4py-1.8.1-py3-none-any.whl (32 kB)
Downloading python_magic-0.4.27-py2.py3-none-any.whl (13 kB)
Installing collected packages: python-magic, thehive4py
Successfully installed python-magic-0.4.27 thehive4py-1.8.1
WARNING: Running pip as the 'root' user can result in broken permissions and conflicting behaviour with the system package manager. It is recommended to use a virtual environment instead: h
ttps://pip.pypa.io/warnings/venv

[notice] A new release of pip is available: 23.3.2 -> 25.0.1
[notice] To update, run: /var/ossec/framework/python/bin/python3.10 -m pip install --upgrade pip
bash-5.2#
```

- Then I created a Python in script `/var/ossec/integrations/custom-w2thive.py` to send alerts from Wazuh to TheHive

```
~ — inafikov@compute-vm-4-16-30-ssd-1743097710847: ~/docker/testing — ssh -i .ssh/theHive_server inafikov@158.160.95.136   ~ — nafikov@wazuh-server-ubuntu: ~ — ssh -i .ssh/wazuh_server nafikov@89.169.173.179
bash-5.2# vim /var/ossec/integrations/custom-w2thive.py
bash-5.2# head vim /var/ossec/integrations/custom-w2thive.py
head: cannot open 'vim' for reading: No such file or directory
==> /var/ossec/integrations/custom-w2thive.py <==
#!/var/ossec/framework/python/bin/python3
import json
import sys
import os
import re
import logging
import uuid
from thehive4py.api import TheHiveApi
from thehive4py.models import Alert, AlertArtifact

bash-5.2# tail /var/ossec/integrations/custom-w2thive.py

if __name__ == "__main__":

    try:
        logger.debug('debug mode') # if debug enabled
        # Main function
        main(sys.argv)

    except Exception:
        logger.exception('EGOR')
bash-5.2#
```

- Then I created a bash script that will run the Python script properly

```
~ — inafikov@compute-vm-4-16-30-ssd-1743097710847: ~/docker/testing — ssh -i .ssh/theHive_server inafikov@158.160.95.136   ~ — nafikov@wazuh-server-ubuntu: ~ — ssh -i .ssh/wazuh_server nafikov@89.169.173.179
# Copyright (C) 2015-2020, Wazuh Inc.
# Created by Wazuh, Inc. <info@wazuh.com>.
# This program is free software; you can redistribute it and/or modify it under the terms of GP>

WPYTHON_BIN="framework/python/bin/python3"

SCRIPT_PATH_NAME="$0"

DIR_NAME="$(cd $(dirname ${SCRIPT_PATH_NAME}); pwd -P)"
bash-5.2# tail /var/ossec/integrations/custom-w2thive
        if [ -z "${WAZUH_PATH}" ]; then
            WAZUH_PATH="$(cd ${DIR_NAME}/..; pwd)"
        fi

    PYTHON_SCRIPT="${DIR_NAME}/${SCRIPT_NAME}.py"
    ;;
esac


${WAZUH_PATH}/${WPYTHON_BIN} ${PYTHON_SCRIPT} $@
bash-5.2#
```

- I gave the proper permissions to these scripts for Wazuh to be able run them

```
~ — inafikov@compute-vm-4-16-30-ssd-1743097710847: ~/docker/testing — ssh -i .ssh/theHive_server inafikov@158.160.95.136   ~ — nafikov@wazuh-server-ubuntu: ~ — ssh -i .ssh/wazuh_server nafikov@89.169.173.179
bash-5.2# chmod 755 /var/ossec/integrations/custom-w2thive.py
bash-5.2# chmod 755 /var/ossec/integrations/custom-w2thive
bash-5.2# chown root:ossec /var/ossec/integrations/custom-w2thive.py
chown: invalid group: 'root:ossec'
bash-5.2# chown root:wazuh /var/ossec/integrations/custom-w2thive.py
bash-5.2# sudo chown root:wazuh var/ossec/integrations/custom-w2thive
bash: sudo: command not found
bash-5.2# chown root:wazuh var/ossec/integrations/custom-w2thive
bash-5.2#
```

- Afterwards, to allow Wazuh to run the integration script, I added the `integration` block to the manager configuration file located at the `/var/ossec/etc/ossec.conf`. I inserted the IP address for TheHive server along with the API key that was generated earlier.
- However, since I deployed Wazuh as a Docker container I used mounted volume and to modify Wazuh Manager config

```
nafikov@wazuh-server-ubuntu:~/wazuh-docker/single-node/config/wazuh_cluster$ vim wazuh_manager.conf
nafikov@wazuh-server-ubuntu:~/wazuh-docker/single-node/config/wazuh_cluster$ cat wazuh_manager.conf | grep "<integration>" -A 7
  <integration>
    <name>custom-w2thive</name>
    <hook_url>http://158.160.95.136:9000/thehive</hook_url>
    <api_key>CpWx2N0RGzJ7pp040YG28Gsf8FaTlpk1</api_key>
    <alert_format>json</alert_format>
  </integration>

</ossec_config>
nafikov@wazuh-server-ubuntu:~/wazuh-docker/single-node/config/wazuh_cluster$
```

- And restarted Wazuh Manager

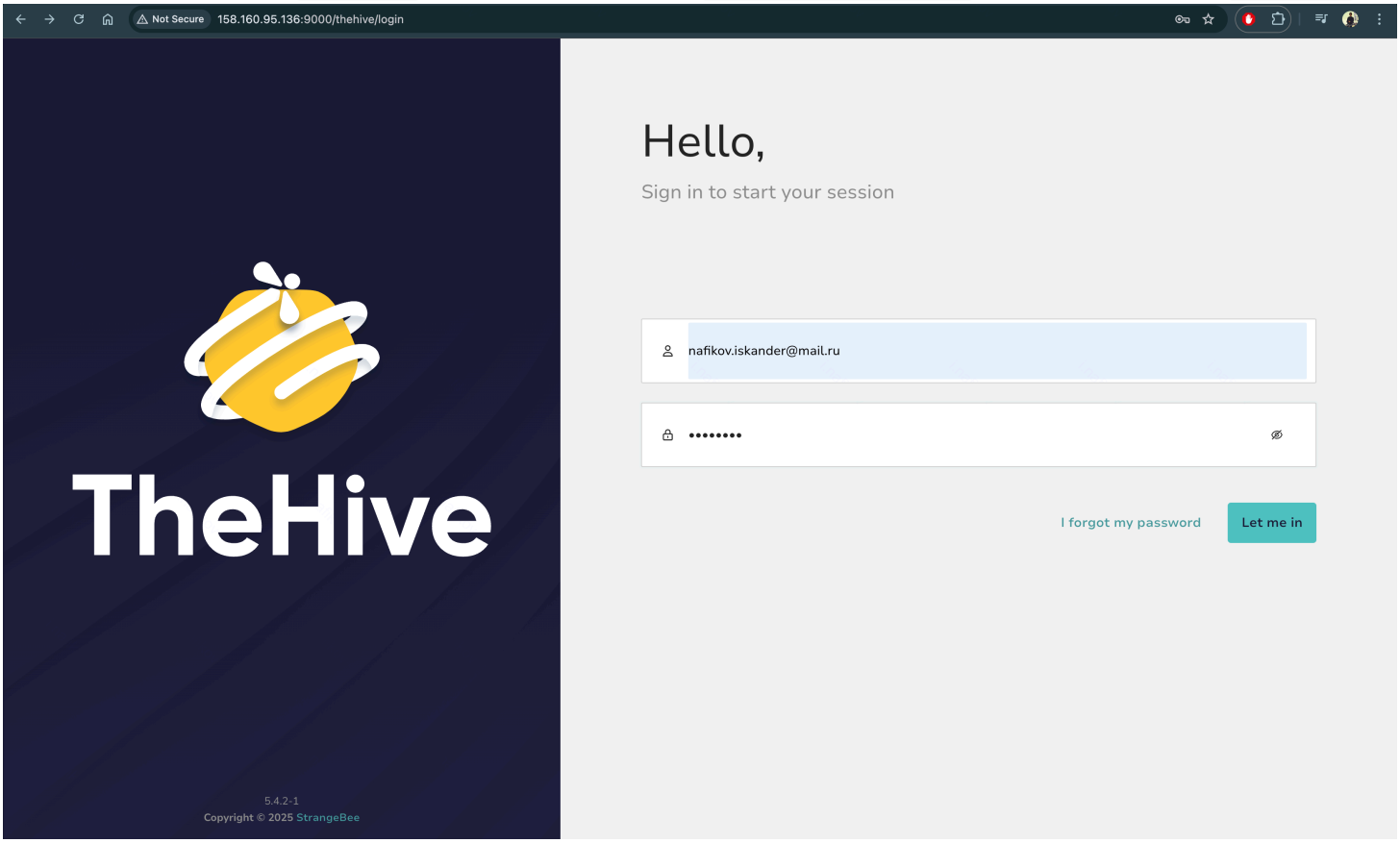
```
nafikov@wazuh-server-ubuntu:~/wazuh-docker/single-node/config/wazuh_cluster$ docker compose restart wazuh.manager
WARN[0000] /home/nafikov/wazuh-docker/single-node/docker-compose.yml: the attribute `version` is obsolete, it will be ignored, please remove it to avoid potential confusi
on
[+] Restarting 2/2
 ✔ Container single-node-wazuh.manager-1    Started                                                                                                    4.1s
 ✔ Container single-node-wazuh.dashboard-1  Started                                                                                                   10.8s
nafikov@wazuh-server-ubuntu:~/wazuh-docker/single-node/config/wazuh_cluster$
```

- Finally, the file changes were applied to the `/var/ossec/etc/ossec.conf` inside the container

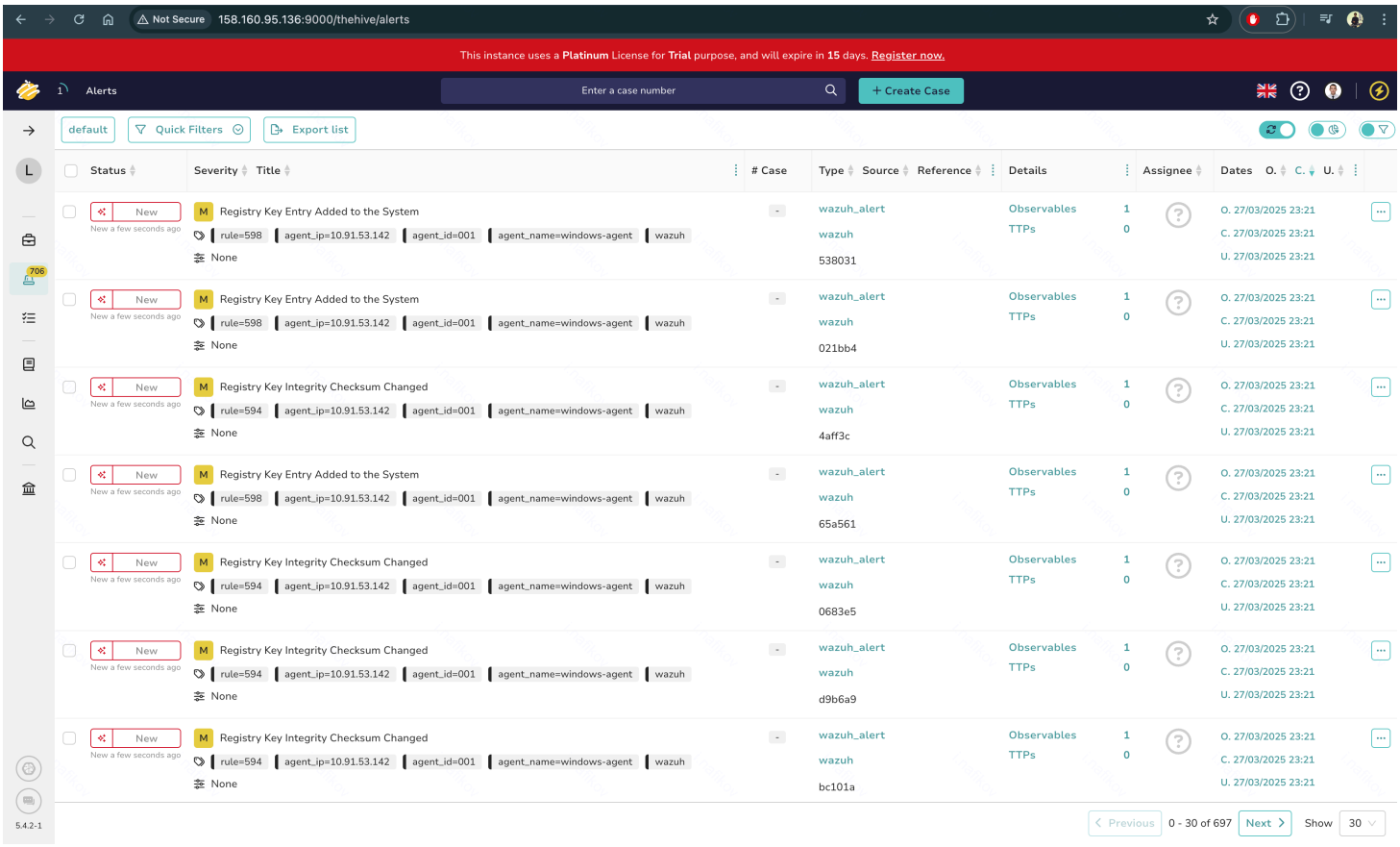
```
nafikov@wazuh-server-ubuntu:~/wazuh-docker/single-node/config/wazuh_cluster$ docker exec single-node-wazuh.manager-1 cat /var/ossec/etc/ossec.conf | grep "<integration>"
-A 7
  <integration>
    <name>custom-w2thive</name>
    <hook_url>http://158.160.95.136:9000/thehive</hook_url>
    <api_key>CpWx2N0RGzJ7pp040YG28Gsf8FaTlpk1</api_key>
    <alert_format>json</alert_format>
  </integration>

</ossec_config>
nafikov@wazuh-server-ubuntu:~/wazuh-docker/single-node/config/wazuh_cluster$
```

Alerts analysis & Incident Response

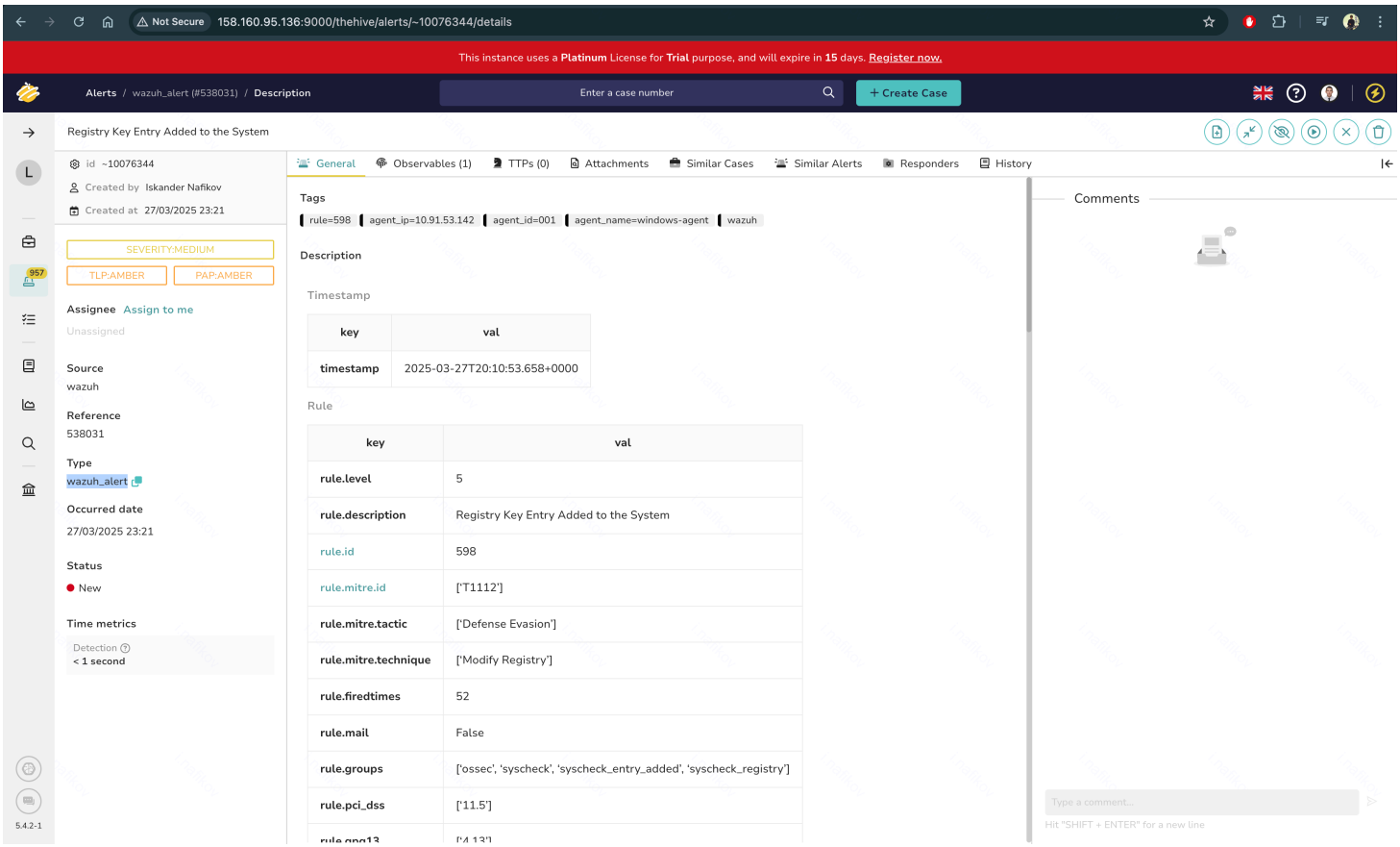
- So, I logged in to the `lab-org` organisation as the `Iskander Nafikov` user with the role `analyst`



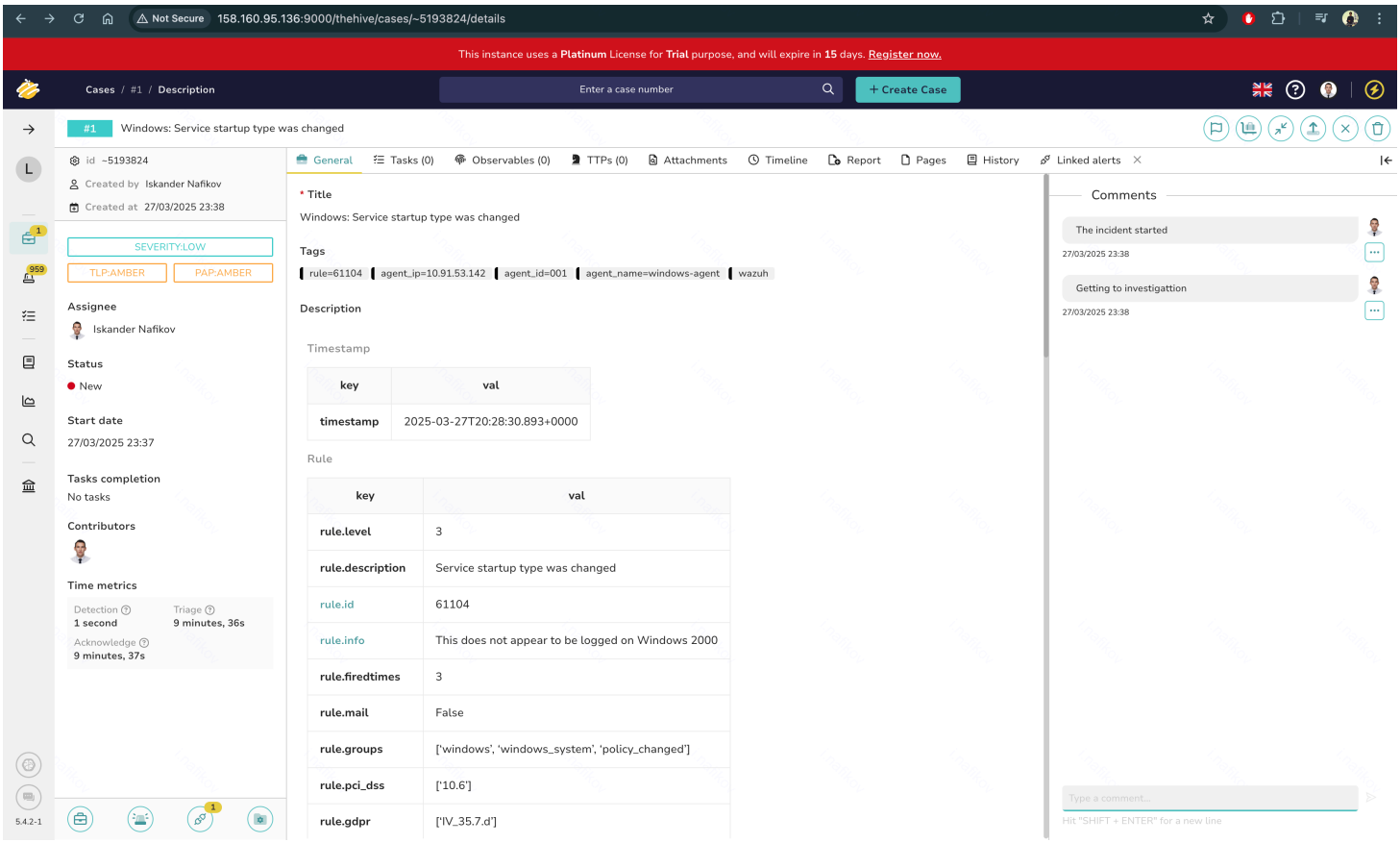
- Finally, I was able to check the alerts inside TheHive coming from Wazuh



- We see that these alerts came right from Wazuh in the same format



- I even created a new case with low severity from some alert



- Then I closed it quickly emulating the case was false positive