INVESTIGATION REPORT ON UNAUTHORISED PROCESS DETECTION AND MALICIOUS IP BLOCKING

Executive Summary

This report details two cybersecurity operations performed to detect unauthorized activities and block malicious actors in a monitored environment. Tools such as Wazuh were configured and used to monitor process executions (specifically Netcat) and block malicious IP addresses attempting attacks (e.g., SQL Injection). Effective mitigation strategies and active responses were implemented to enhance the endpoint and network security.

Task 1

Detecting Unauthorized Processes (Netcat Usage)

This investigation report was carried out due to a recent security audit at 10ALYTICS-DC which revealed instances were;

engineers and third-party contractors used Netcat for debugging. While Netcat is a legitimate tool, its misuse poses a risk, as attackers frequently leverage it for unauthorized data exfiltration or

as a backdoor for remote access. The security team must implement a robust detection mechanism to ensure that only authorized processes are running within the environment.

Objectives

1. Implement Wazuh's command monitoring to track active processes.
2. Detect and log unauthorized Netcat executions.
3. Generate security alerts in Wazuh when suspicious activity occurs and Provide real-time visibility through the Wazuh dashboard

Setup and Configuration

- **Endpoint**: Ubuntu Server
- **Monitoring Tool**: Wazuh Agent

Processes on Investigation carried out

1. I configured the Wazuh command monitoring module on this endpoint to detect a running Netcat process.

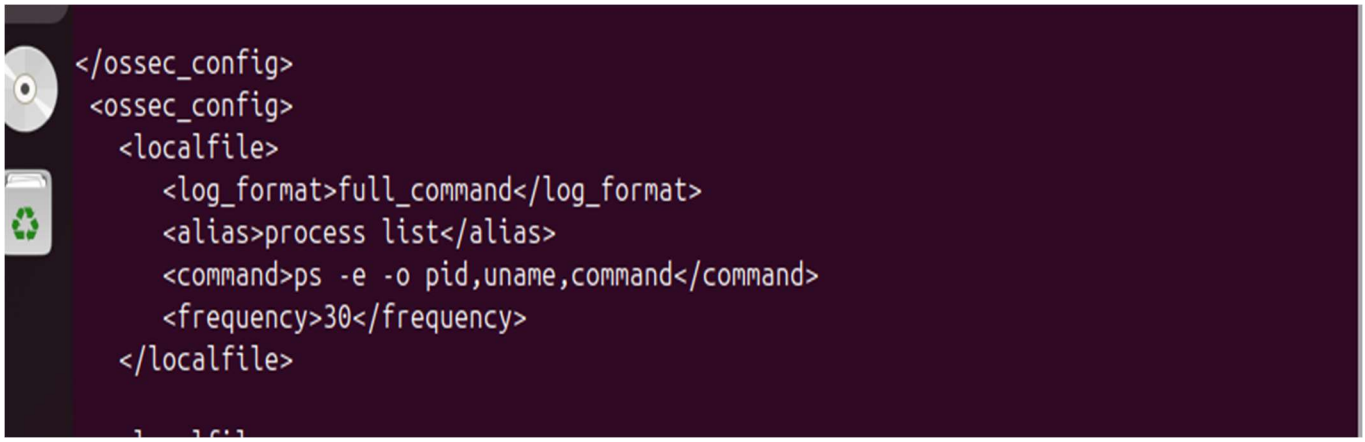
Configuration

- **Ubuntu endpoint**

I took the following steps to configure command monitoring and query a list of all running processes on the Ubuntu endpoint.

Step 1. Add the following configuration block to the Wazuh agent /var/ossec/etc/ossec.conf file. This allows to periodically get a list of running processes:

```
<ossec_config>
 <localfile>
  <log_format>full_command</log_format>
  <alias>process list</alias>
  <command>ps -e -o pid,uname,command</command>
  <frequency>30</frequency>
 </localfile>
</ossec_config>
```



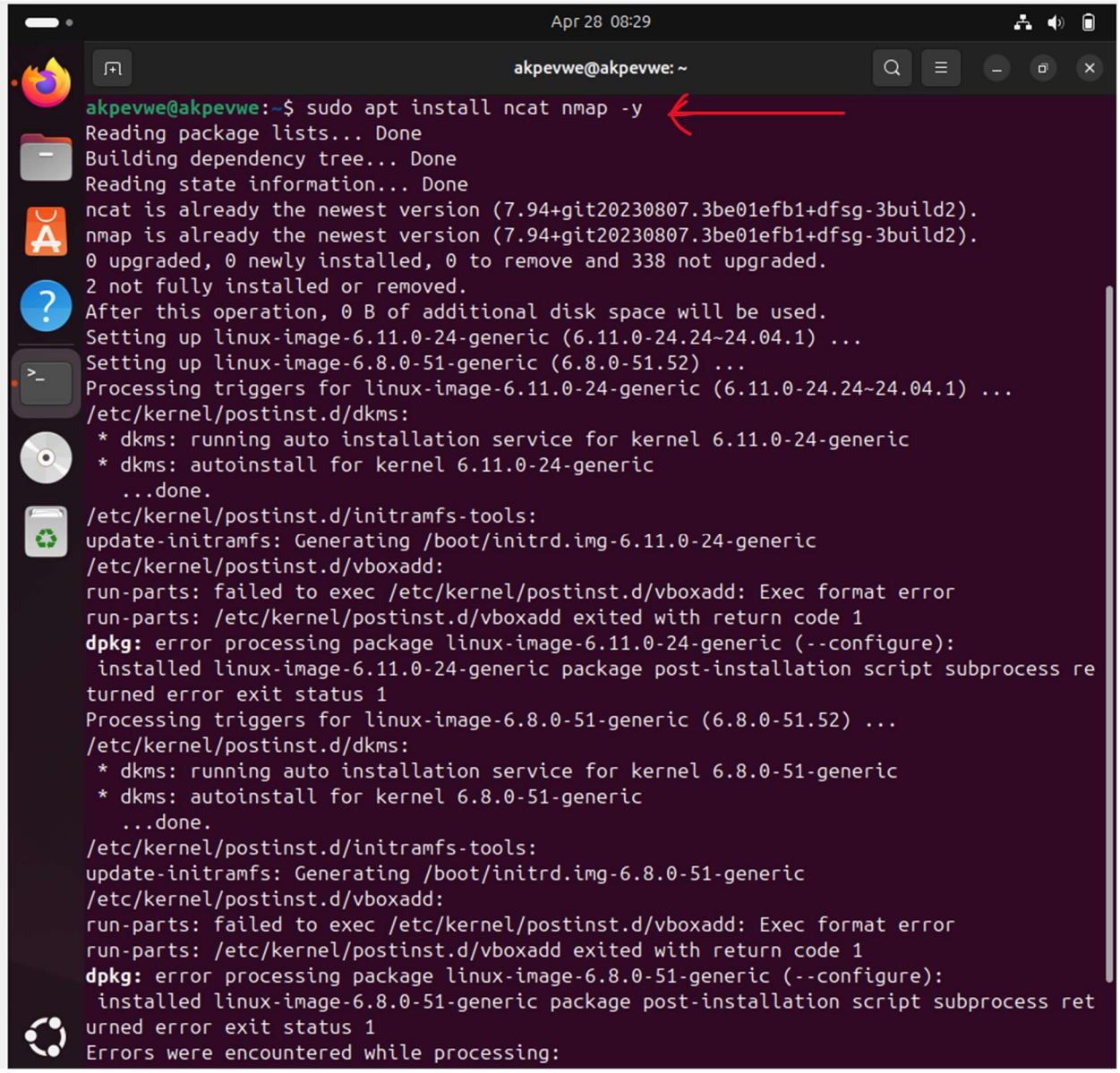
Description: This was added in my Ubuntu end point.

Step 2: Restart the Wazuh agent to apply the changes:

sudo systemctl restart wazuh-agent

Step 3: Installed Netcat and the required dependencies:

sudo apt install ncat nmap -y



Description: The sudo apt install ncat nmap -y was ran on the Ubuntu endpoint which is indicated with a red arrow.

Wazuh server:

I configured the following steps on the Wazuh server to create a rule that triggers every time the Netcat program launches.

Step 4: Add the following rules to the /var/ossec/etc/rules/local_rules.xml file on the Wazuh server:

```
<group name="ossec,">
  <rule id="100050" level="0">
    <if_sid>530</if_sid>
    <match>^ossec: output: 'process list'</match>
    <description>List of running processes.</description>
    <group>process_monitor,</group>
  </rule>

    <rule id="100051" level="7" ignore="900">
    <if_sid>100050</if_sid>
    <match>nc -l</match>
    <description>netcat listening for incoming connections.</description>
    <group>process_monitor,</group>
  </rule>
</group>
```

```
 GNU nano 2.9.8          /var/ossec/etc/rules/local_rules.xml


<group name="ossec,">
 <rule id="100050" level="0">
   <if_sid>530</if_sid>
   <match>^ossec: output: 'process list'</match>
   <description>List of running processes.</description>
   <group>process_monitor,</group>
  </rule>

<rule id="100051" level="7" ignore="900">
    <if_sid>100050</if_sid>
    <match>nc -l</match>
    <description>netcat listening for incoming connections.</description>
    <group>process_monitor,</group>
   </rule>
</group>
```

Description: The rules was inputted on wazuh.

Step 5: Restart the Wazuh manager to apply the changes:

sudo systemctl restart wazuh-manager

Attack emulation

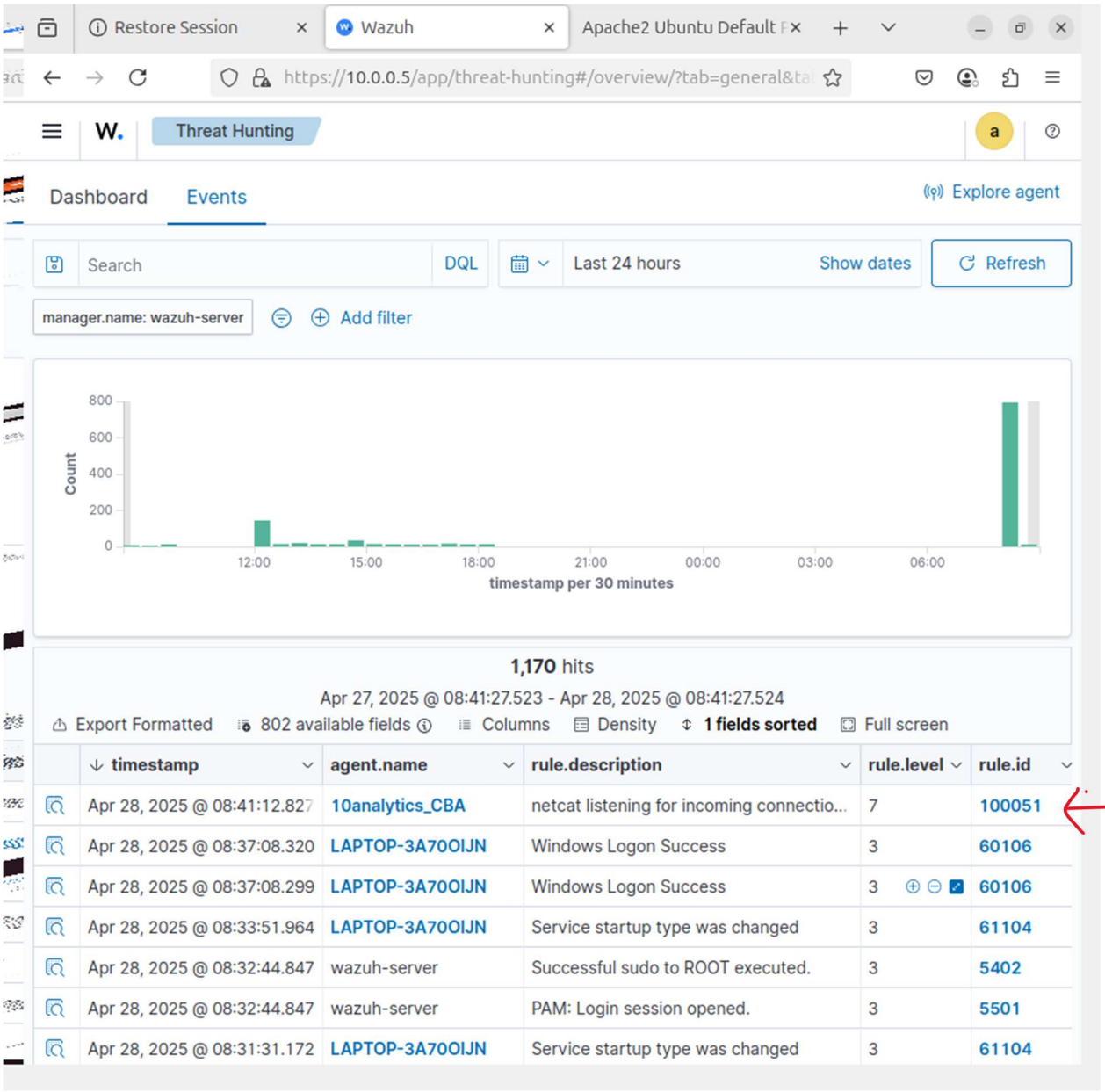
Step 6: On the monitored Ubuntu endpoint, run nc -l 8000 for 30 seconds.

```
run-parts: /etc/kernel/postinst.d/vboxadd exited with return code 1
dpkg: error processing package linux-image-6.8.0-51-generic (--configure):
 installed linux-image-6.8.0-51-generic package post-installation script subprocess ret
urned error exit status 1
Errors were encountered while processing:
 linux-image-6.11.0-24-generic
 linux-image-6.8.0-51-generic
E: Sub-process /usr/bin/dpkg returned an error code (1)
akpevwe@akpevwe:~$ sudo apt install ncat nmap-y^C
akpevwe@akpevwe:~$ nc -l 8000
```

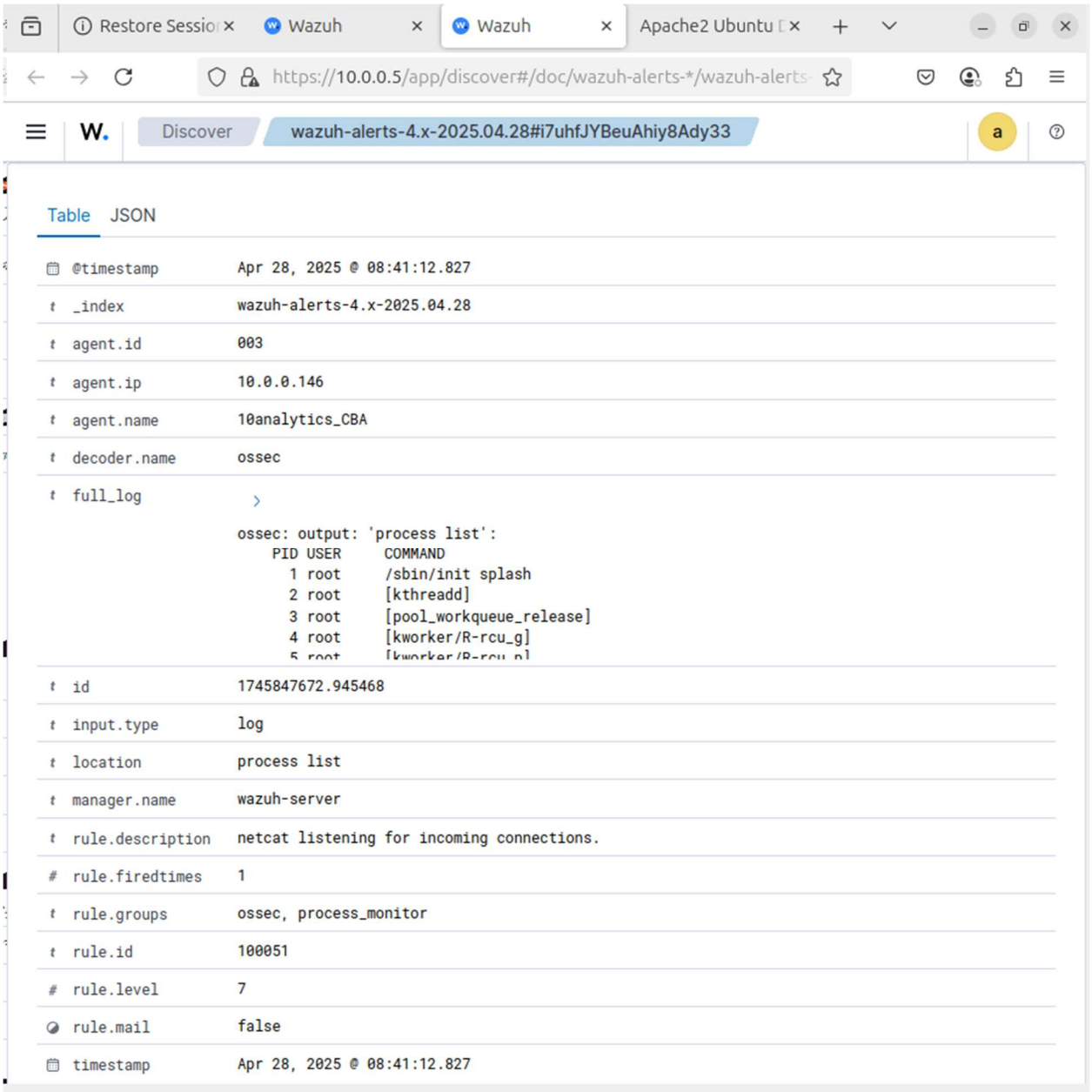
Description: The command nc -l 8000 was ran for 30 seconds in Ubuntu endpoint as indicated with a red arrow.

Step 7: Visualize the alerts

I visualized the alert data in the Wazuh dashboard by going to the Threat Hunting module to obtain rule.id:(100051)



Description: The rule.id:(100051) was visualized on the wazuh server as indicated with red arrow showing rule level 7 with the description, netcat listening for incoming connections.



Description: This shows the details of the alert rule id: 100051

Mitigation Strategies for Netcat Use at 10ALYTICS-DC

Following our recent security audit, we noticed that both internal engineers and third-party contractors have been using **Netcat** for debugging. While this is a legitimate tool, it's also widely used by attackers to create backdoors or move data out of the network unnoticed. To keep our environment secure, here's what is recommended

1. Set Clear Rules on When and How Netcat Can Be Used

- Define who's allowed to use Netcat, and make sure it's only being used for approved tasks.

- Add this to our internal security policy so there's no confusion.

- Require the use of more secure, logged tools for regular debugging wherever possible.

2. Limit Where Netcat Lives

- Remove Netcat from any machine that doesn't need it.

- Monitor systems for any new or renamed Netcat binaries being installed.

3. Detect Suspicious Activity

- Set up alerts for unusual Netcat usage, especially:

- If Netcat is used to listen on odd ports

- If it tries to connect to external IP addresses

4. Track Command Usage

- Enable tools that log the exact commands users run (e.g., auditd on Linux, Sysmon on Windows).

- This helps us trace exactly how Netcat (or any similar tool) was used if something suspicious happens.

5. Watch the Network

- Use intrusion detection tools (like Snort or Suricata) to scan for Netcat-style traffic.

- Monitor outbound connections that don't make sense like random devices talking to strange IPs outside our network.

6. Harden the Environment

- Use application control (like AppLocker or SELinux) to prevent unauthorized programs including Netcat from running.

- For debugging, consider isolated environments or containers so tools like Netcat don't put the whole system at risk.

7. Improve Processes & Awareness

- Encourage the team to use secure alternatives to Netcat, like SSH or socat with encryption.

- Run occasional training to show the risks of using tools like Netcat improperly.

- Simulate real-world scenarios with red team exercises to make sure our detection tools are working.

8. Promote Secure Alternatives

- Encourage use of secure, auditable tools like SSH, socat with TLS, or telnet within sandboxed environments (if absolutely necessary).

9. Log All Activity

- Centralize and retain logs of Netcat-related activity for forensic analysis.

By taking these steps, we can make sure that tools like Netcat are only used when truly needed and never become a way in for attackers.

Task 2

Blocking of Malicious threat Actor

This investigation was carried out because 10ALYTICS-DC has observed a spike in unauthorized login attempts against its web servers, originating from known blacklisted IP addresses. A security breach simulation revealed that attackers were probing for weaknesses in publicly exposed services. To prevent such threats, the security team must automate blocking of malicious IPs using Wazuh's Active Response feature.

Objectives

1. Establish a reputation-based IP blocking mechanism
2. Integrate Wazuh with external threat intelligence feeds.
3. Implement automated firewall rules to block malicious IPs dynamically and ensure compliance with industry best practices in incident response.

Processes on Investigation carried out

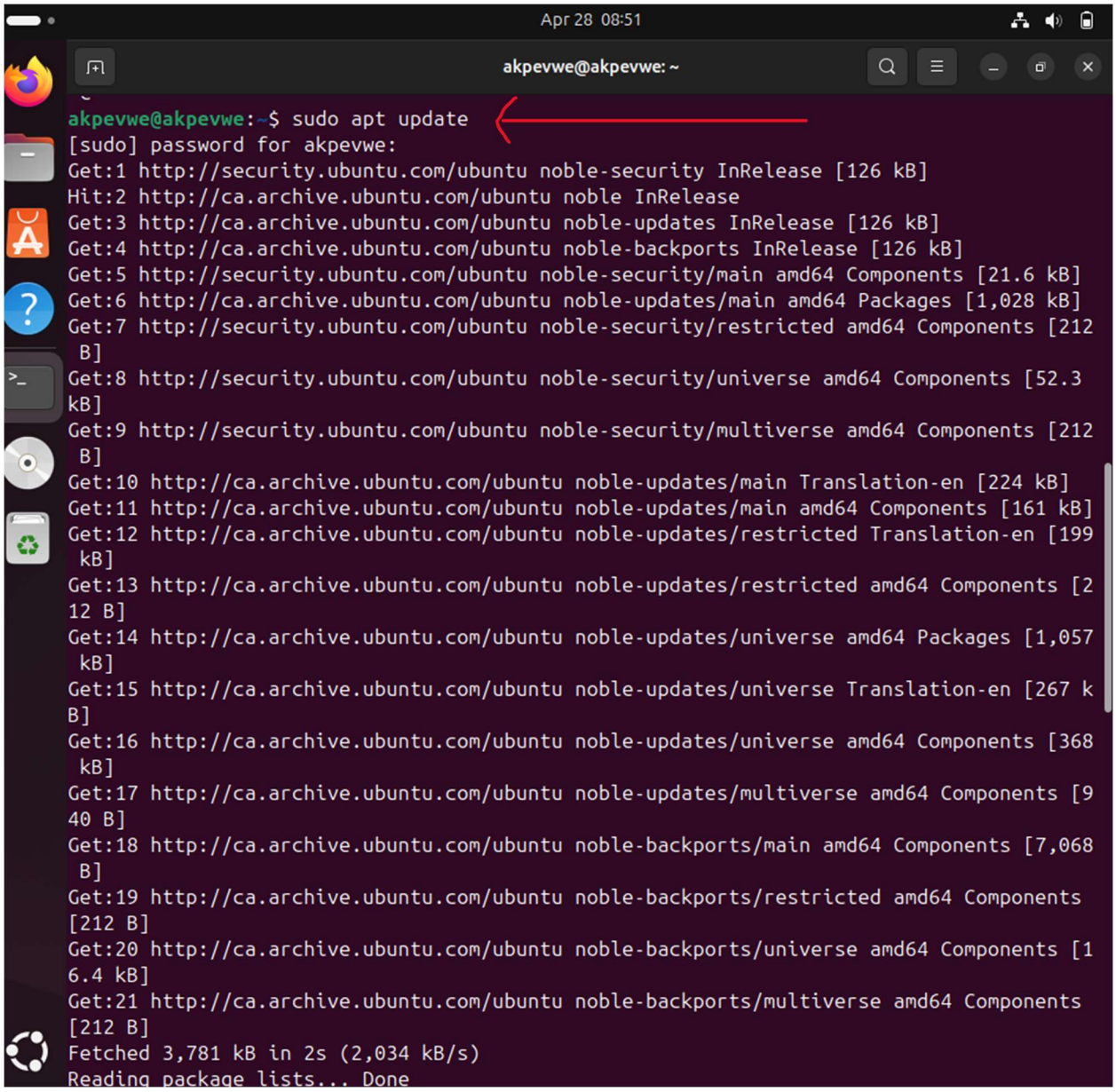
Configuration

Ubuntu endpoint

1. I Performed the following steps to install an Apache web server and monitor its logs with the Wazuh agent.

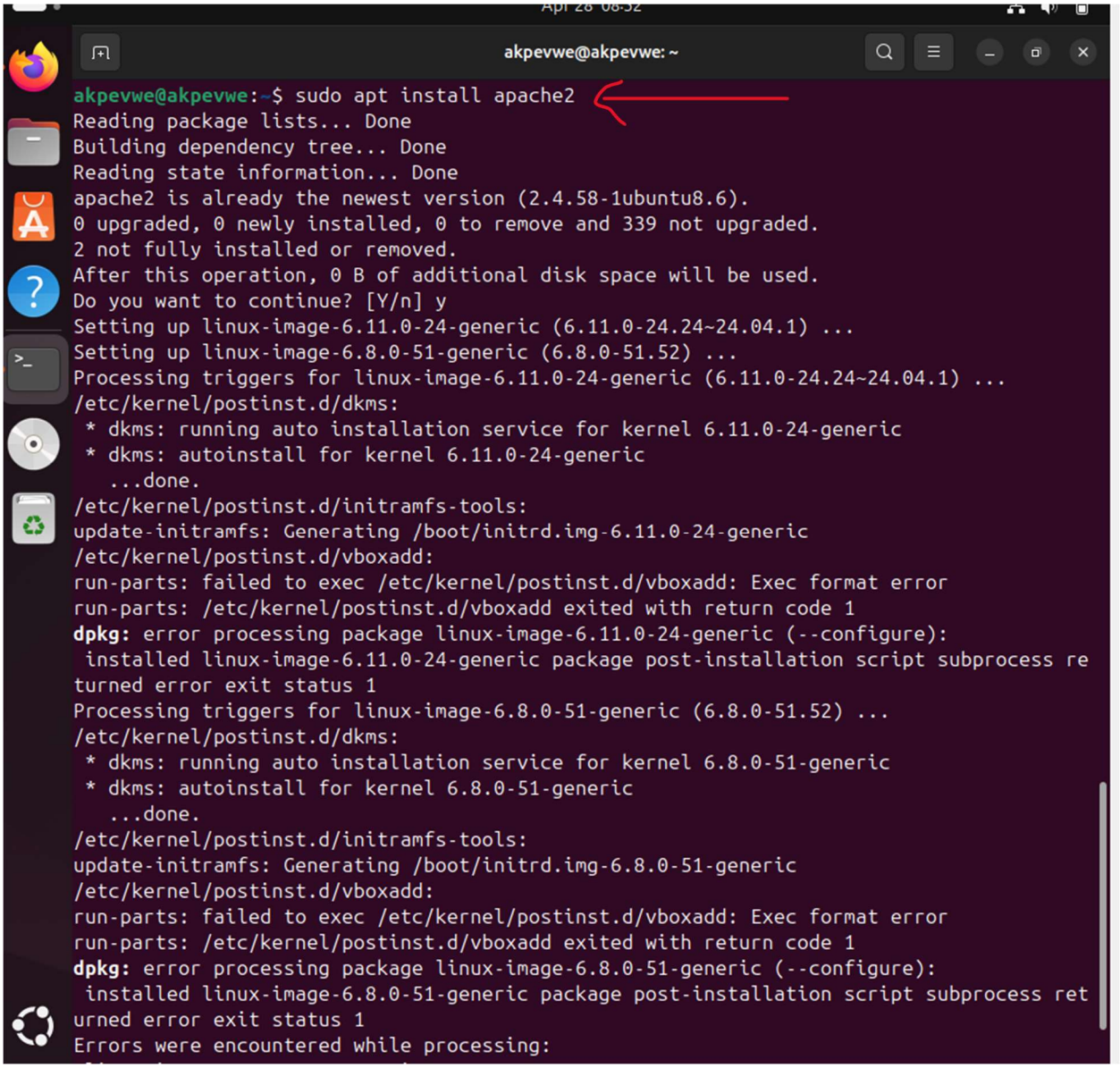
   Step 1: Update local packages and install the Apache web server:

- $ sudo apt update



Description: The sudo apt update command was done on the Ubuntu as indicated with a red arrow and all updates were done.

- $ sudo apt install apache2



Description: The command sudo apt install apache2 was ran on Ubuntu as indicated with a red arrow.

Step 2: If the firewall is enabled, modify the firewall to allow external access to web ports. Skip this step if the firewall is disabled:

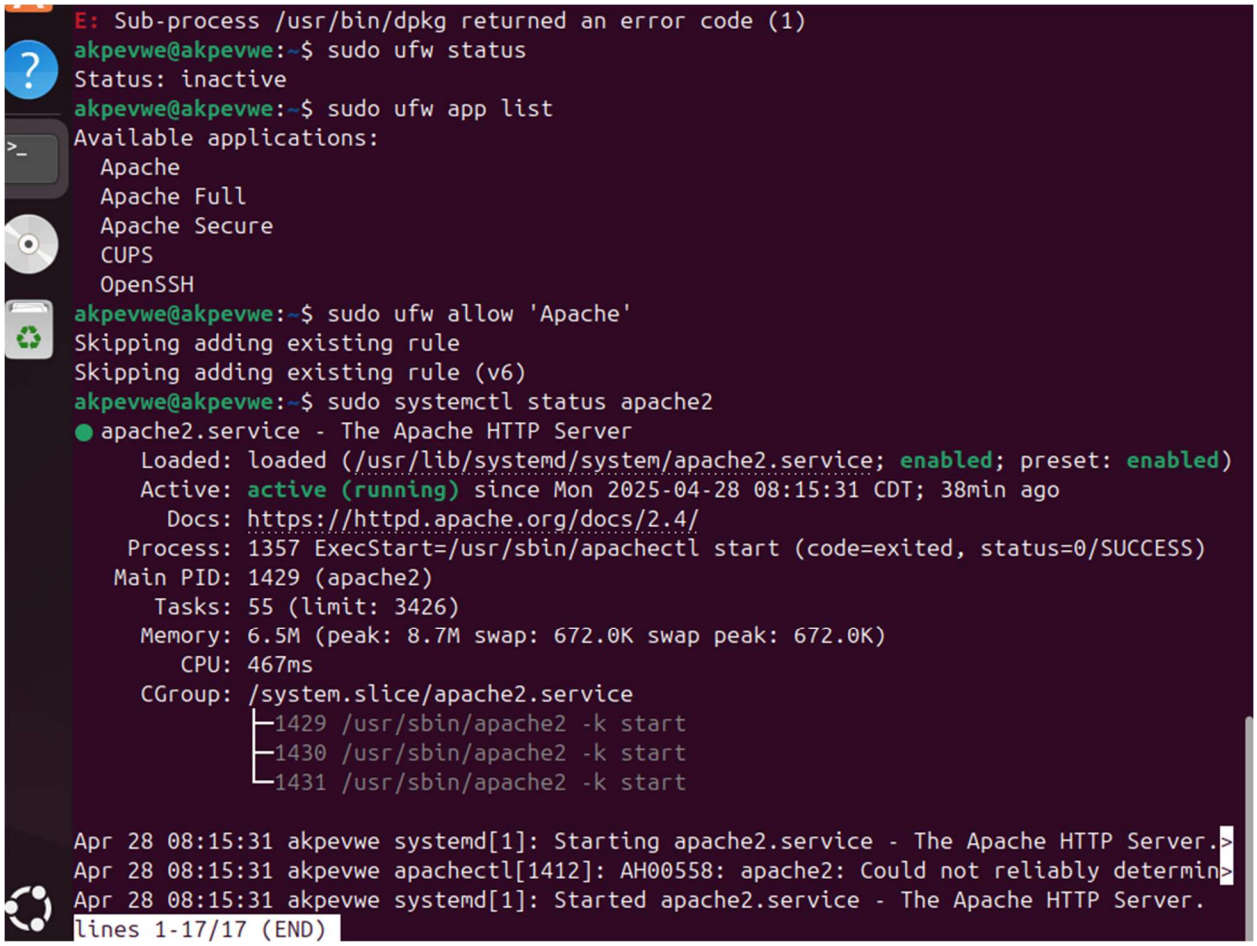
$ sudo ufw status

$ sudo ufw app list

$ sudo ufw allow 'Apache'

Step 3: Check the status of the Apache service to verify that the web server is running:

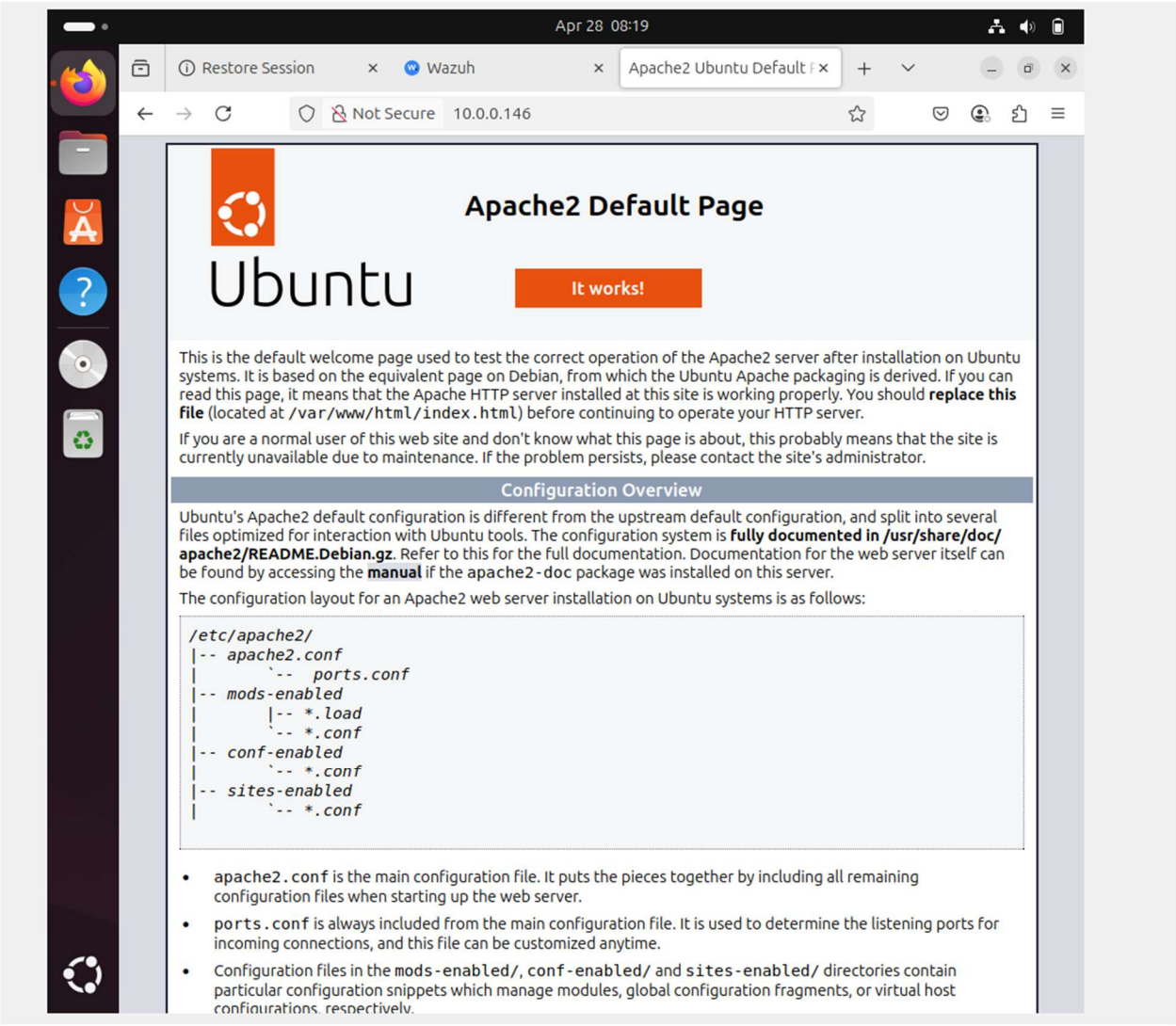
$ sudo systemctl status apache2



Description: This shows how step 2 and 3 were inputted on Ubuntu.

Step 4: Use the curl command or open http://<UBUNTU_IP> in a browser to view the Apache landing page and verify the installation:

$ curl http://10.0.0.146



Description: This show the Apache default page when the ubuntu IP address (10.0.0.146) was inputted

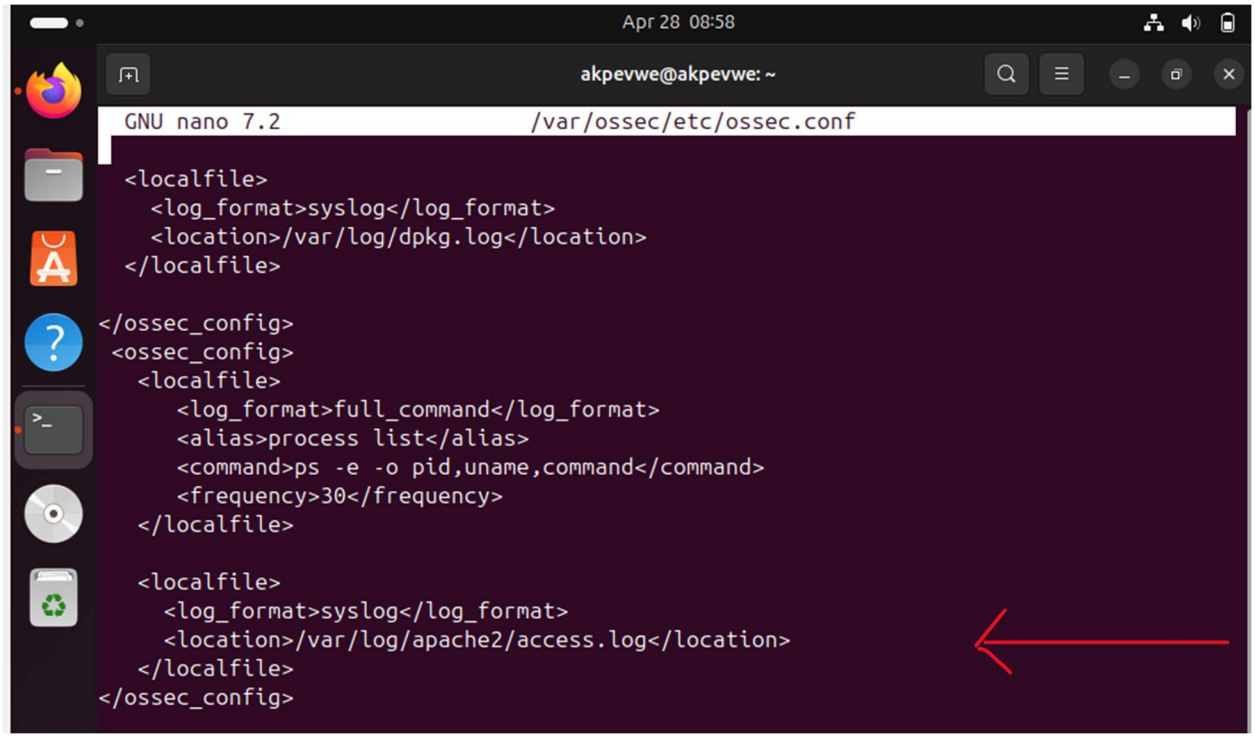
Step 5: Add the following to /var/ossec/etc/ossec.conf file to configure the Wazuh agent and monitor the Apache access logs:

<localfile>

<log_format>syslog</log_format>

<location>/var/log/apache2/access.log</location>

</localfile>



Description:The block indicated with a red arrow was added to the /var/ossec/etc/ossec.conf file on Ubuntu.

Step 6: Restart the Wazuh agent to apply the changes:

sudo systemctl restart wazuh-agent

- **Wazuh server**

I performed the following steps on the Wazuh server to add the IP address of the RHEL endpoint to a CDB list, and then configure rules and Active Response.

Step 7: Download the utilities and configure the CDB list

1. Install the wget utility to download the necessary artifacts using the command line interface:

   $ sudo yum update && sudo yum install -y wget

2. Download the Alienvault IP reputation database:

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

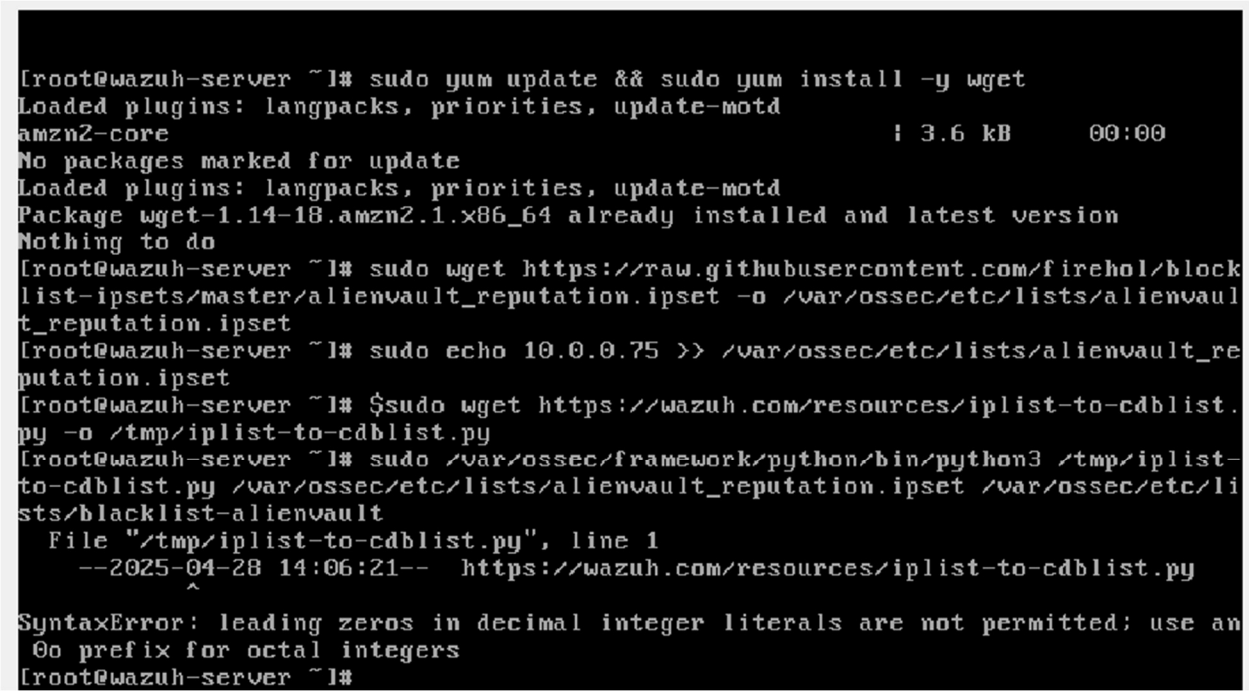
3.Append the IP address of the attacker endpoint to the IP reputation database.

Replace <ATTACKER_IP> with the RHEL IP address in the command below:

$ sudo echo "<ATTACKER_IP>" >> /var/ossec/etc/lists/alienvault_reputation.ipset

Download a script to convert from the .ipset format to the .cdb list format:

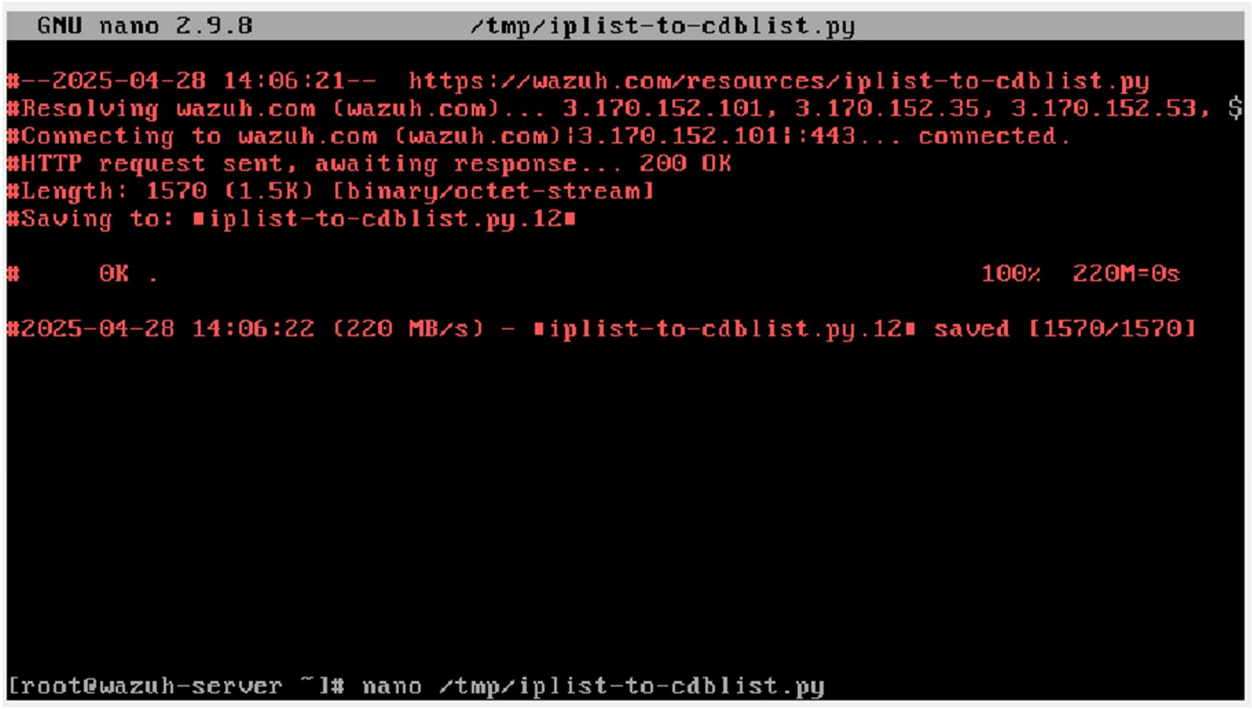
$ sudo wget https://wazuh.com/resources/iplist-to-cdblist.py -O /tmp/iplist-to-cdblist.py



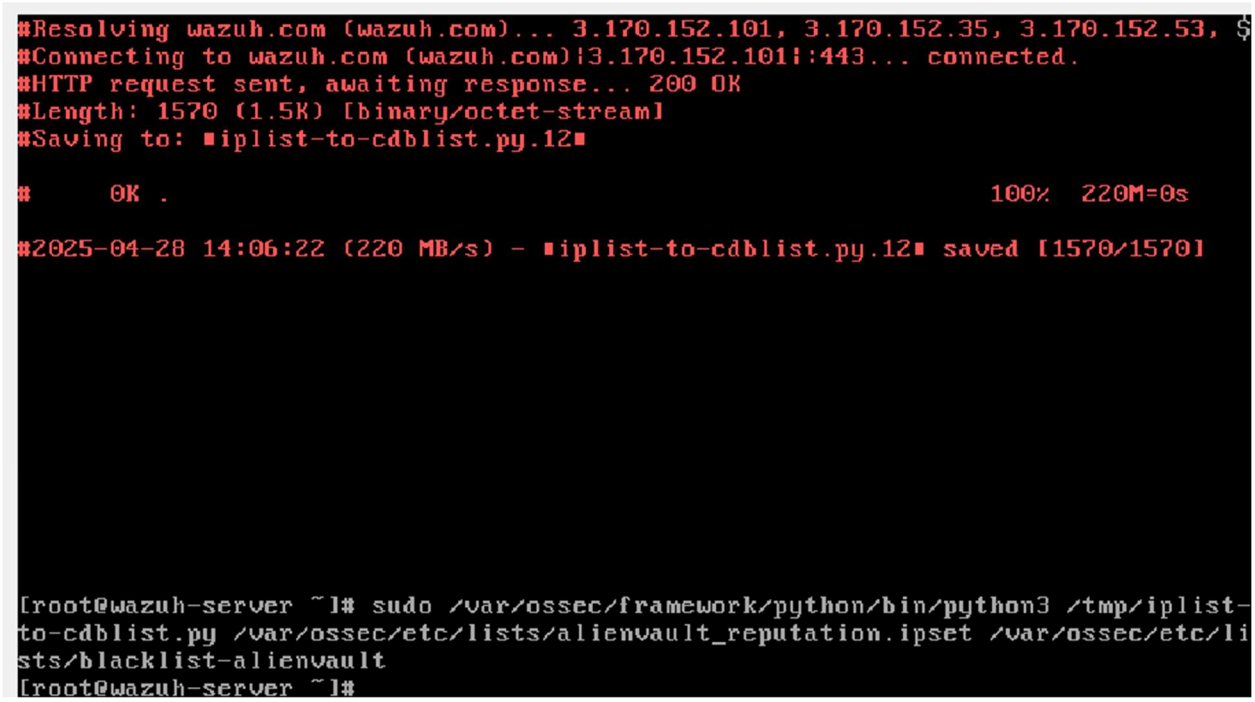
Description: This shows all commands on step 7 ran on wazuh

Step 8: Convert the alienvault_reputation.ipset file to a .cdb format using the previously downloaded script:

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

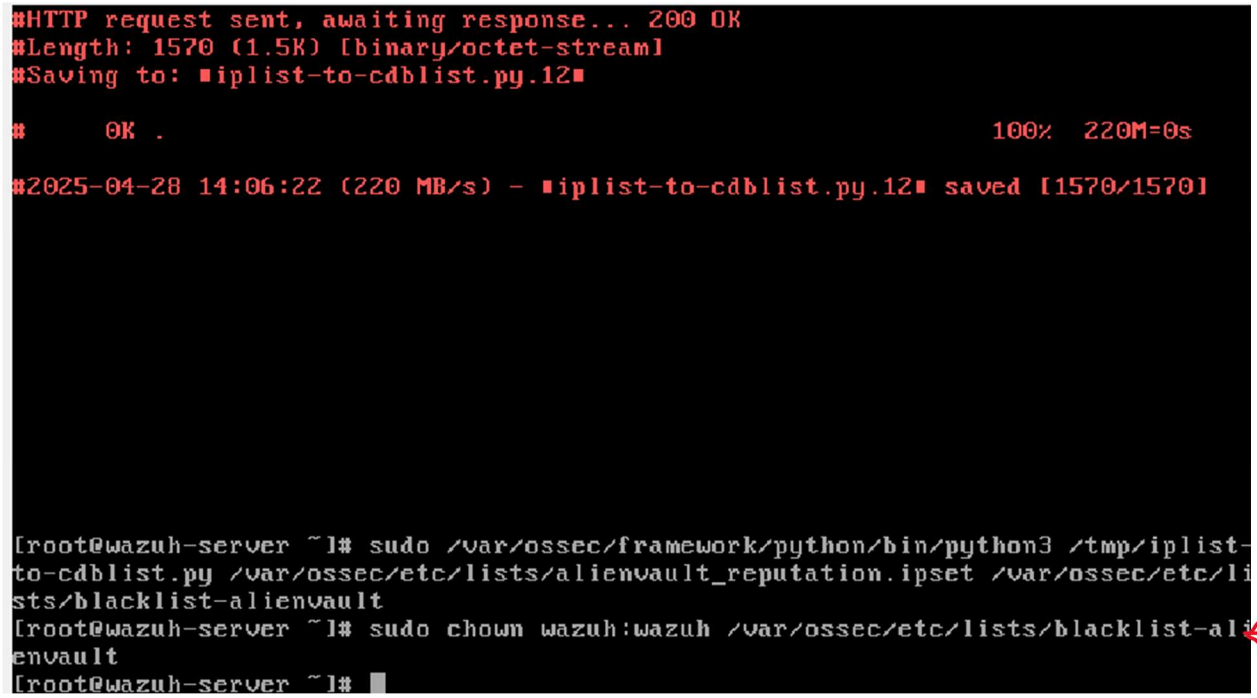


Description: I got an error when I used $ sudo /var/ossec/framework/python/bin/python3 /tmp/iplist-to-cdblist.py /var/ossec/etc/lists/alienvault_reputation.ipset /var/ossec/etc/lists/blacklist-alienvault. I had to open the file with nano /tmp/iplist-to-cdblist.py to open the file then I put # in front of each sentence to make it a string. I inserted the command again and it went through.



Step 9: Assign the right permissions and ownership to the generated file:

$ sudo chown wazuh:wazuh /var/ossec/etc/lists/blacklist-alienvault



Step 10: Configure the Active Response module to block the malicious IP address

Add a custom rule to trigger a Wazuh active response script. Do this in the Wazuh server /var/ossec/etc/rules/local_rules.xml custom ruleset file:

<group name="attack,">

<rule id="100100" level="10">

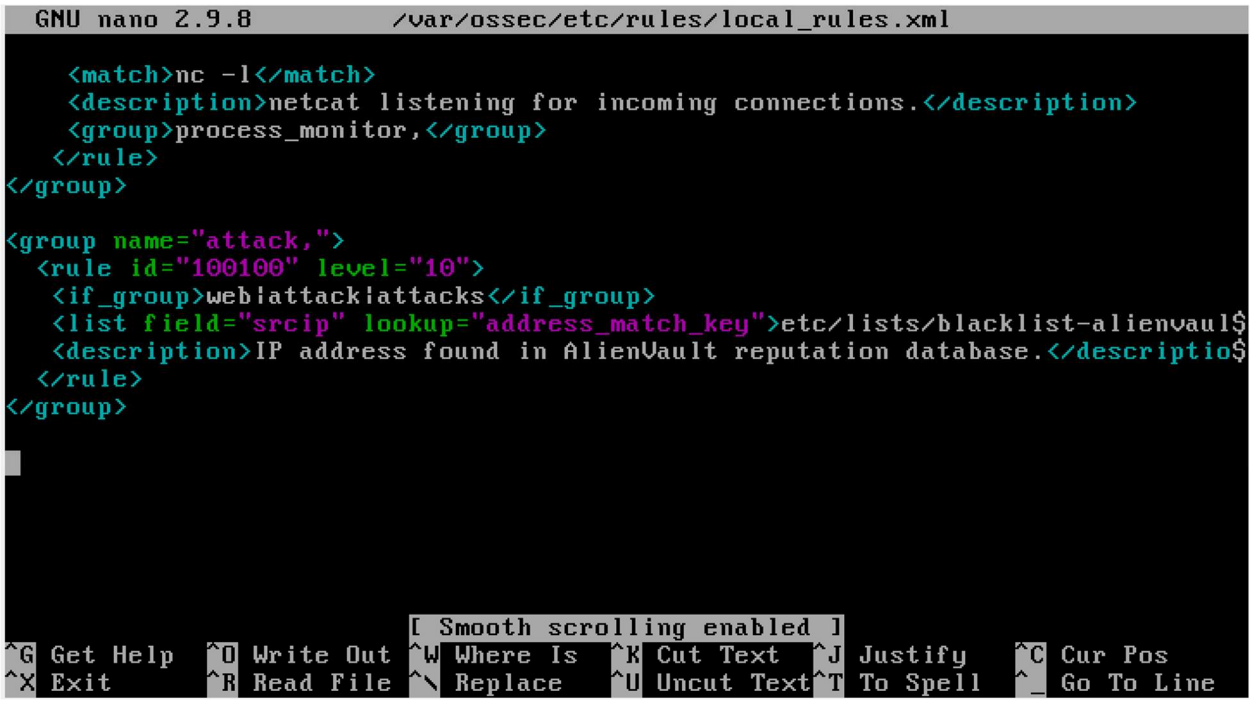
<if_group>web|attack|attacks</if_group>

<list field="srcip" lookup="address_match_key">etc/lists/blacklist-alienvault</list>

<description>IP address found in AlienVault reputation database.</description>

</rule>

</group>



Description: Rule inputted on wazuh

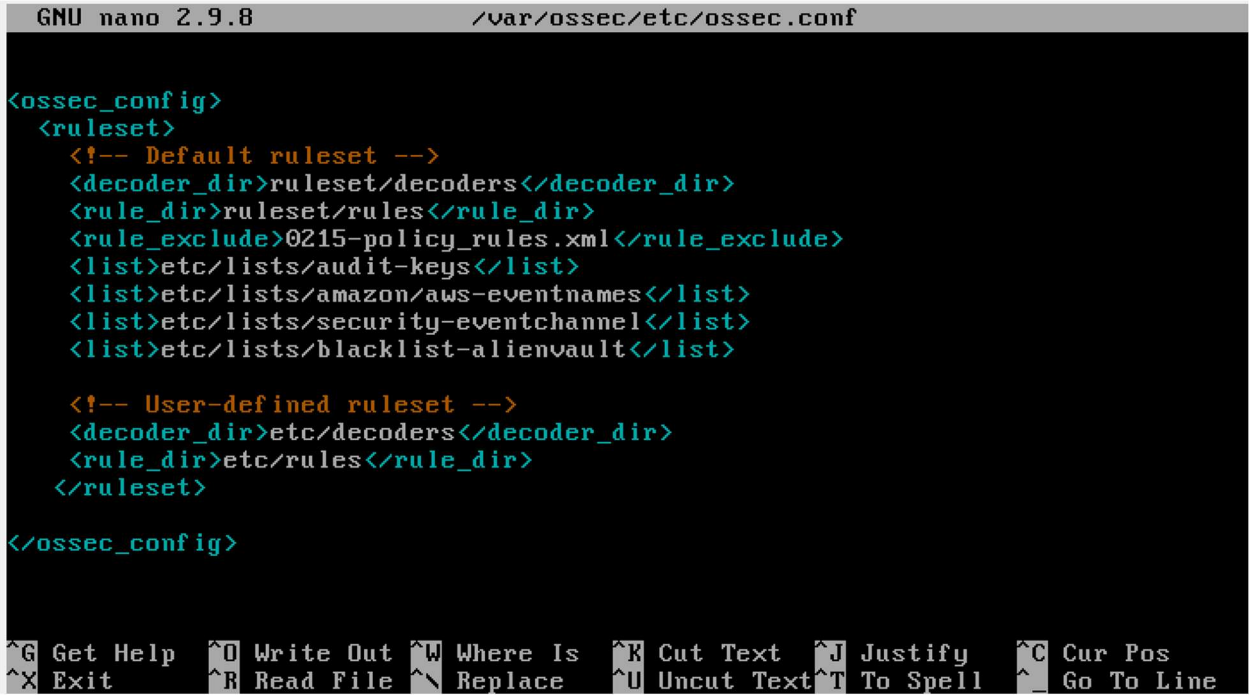
Step 11: Edit the Wazuh server /var/ossec/etc/ossec.conf configuration file and add the etc/lists/blacklist-alienvault list to the <ruleset> section:

```
<ossec_config>
 <ruleset>
  <!-- Default ruleset -->
 <decoder_dir>ruleset/decoders</decoder_dir>
  <rule_dir>ruleset/rules</rule_dir>
  <rule_exclude>0215-policy_rules.xml</rule_exclude>
   <list>etc/lists/audit-keys</list>
   <list>etc/lists/amazon/aws-eventnames</list>
   <list>etc/lists/security-eventchannel</list>
   <list>etc/lists/blacklist-alienvault</list>
    <!-- User-defined ruleset -->
  <decoder_dir>etc/decoders</decoder_dir>
```

< rule_dir>etc/rules</rule_dir>

</ruleset>

</ossec_config>



Description: Rule added on wazuh.

Step 12: Add the Active Response block to the Wazuh server /var/ossec/etc/ossec.conf file:

- **For the Ubuntu endpoint**

The firewall-drop command integrates with the Ubuntu local iptables firewall and drops incoming network connection from the attacker endpoint for 60 seconds:

<ossec_config>

<active-response>

<disabled>no</disabled>

<command>firewall-drop</command>

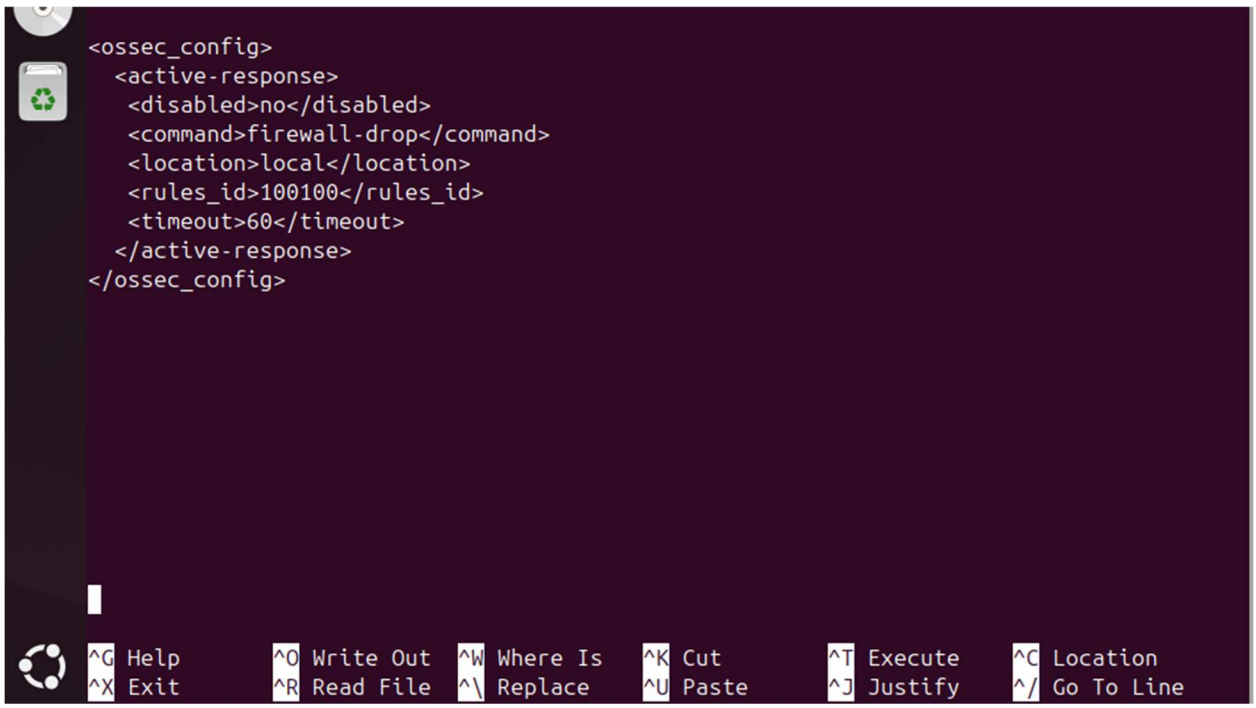
<location>local</location>

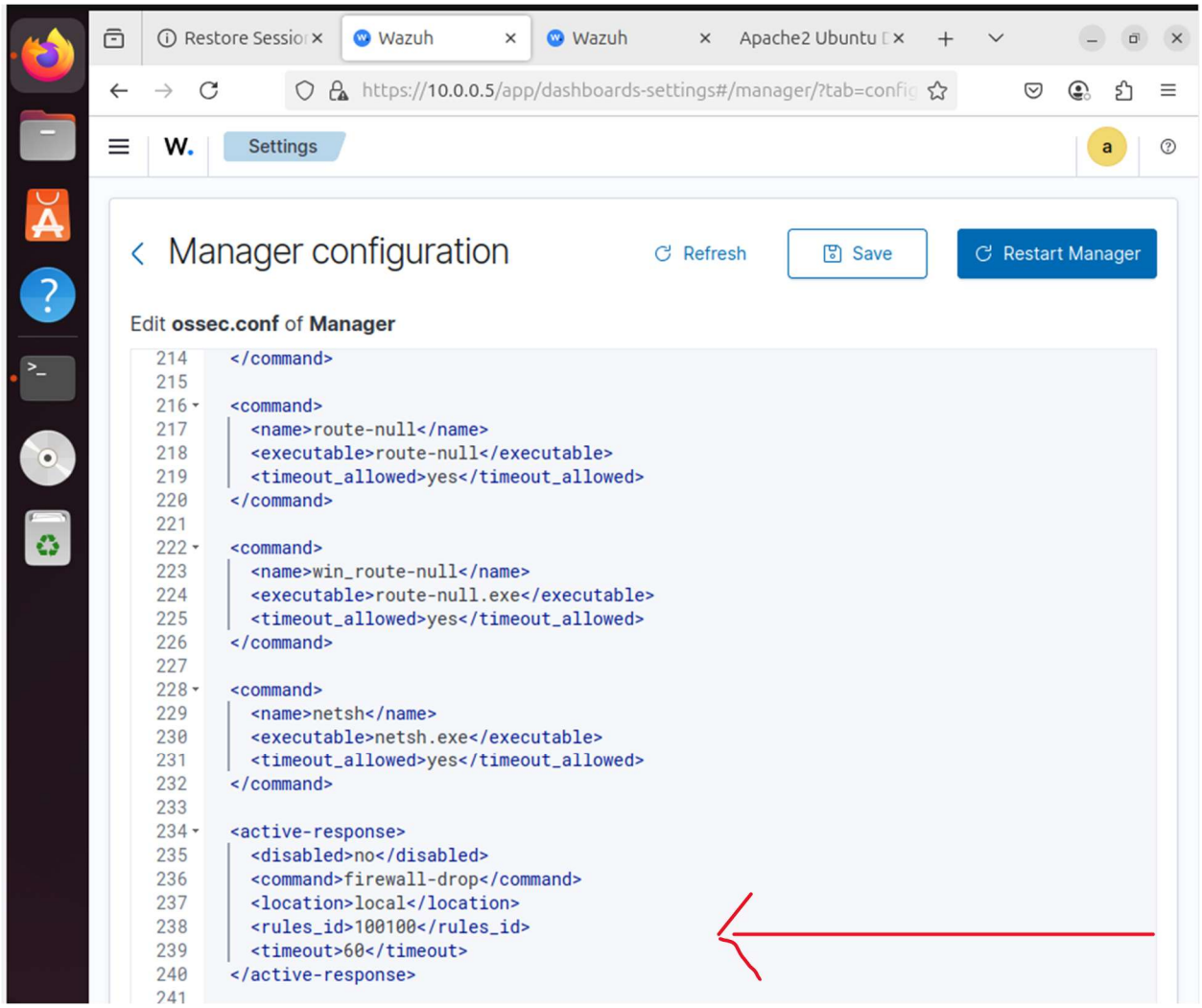
<rules_id>100100</rules_id>

<timeout>60</timeout>

</active-response>

</ossec_config>





Description: Active response on the web server

- Restart the Wazuh manager to apply the changes:

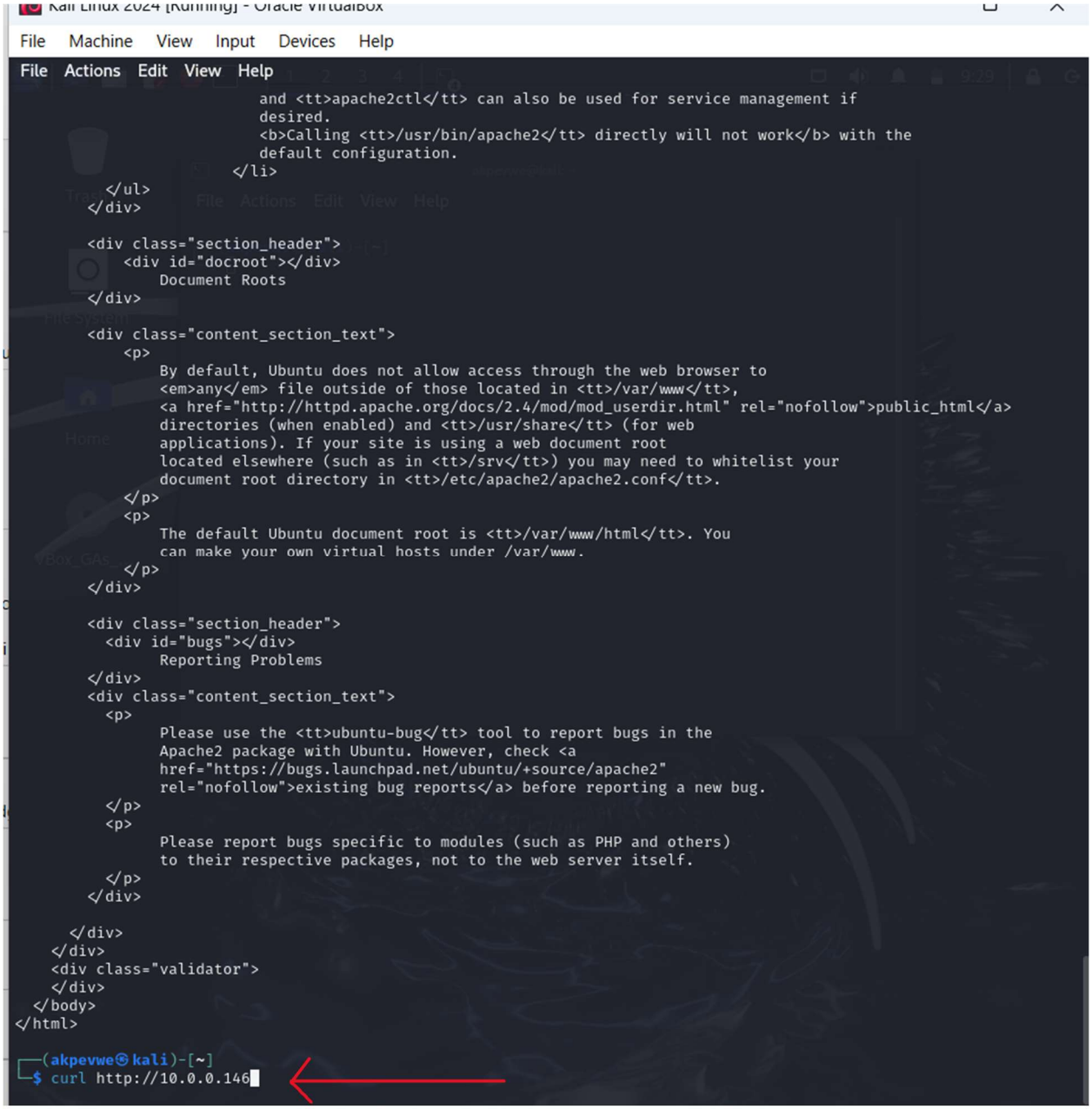
  $ sudo systemctl restart wazuh-manager

Step 13: Attack emulation

1. Access any of the web servers from the RHEL endpoint using the corresponding IP address. Replace <WEBSERVER_IP> with the appropriate value and execute the following command from the attacker endpoint:

   $ curl http://10.0.0.146

The attacker endpoint connects to the victim's web servers the first time. After the first connection, the Wazuh Active Response module temporarily blocks any successive connection to the web servers for 60 seconds.



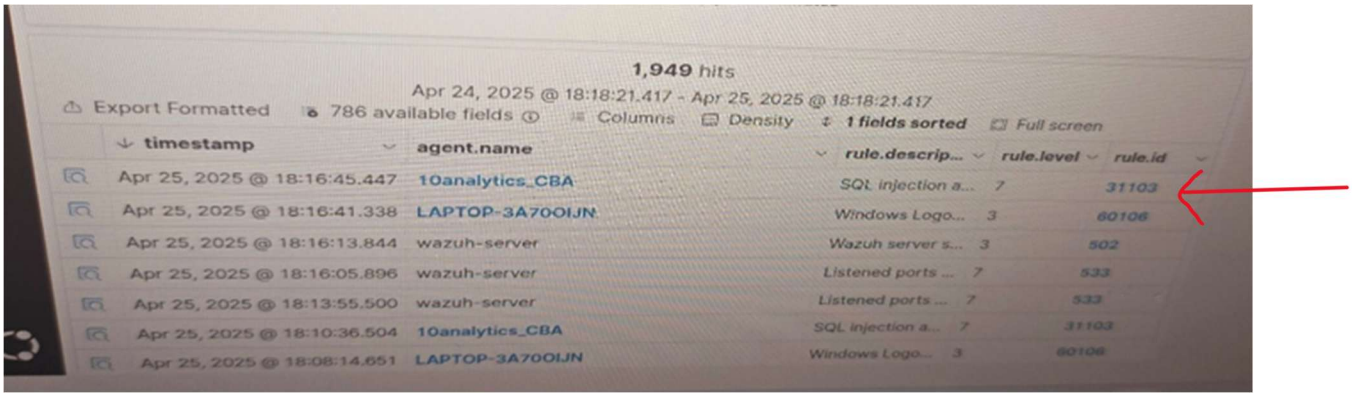
Description: curl http://10.0.0.146 was ran on kali and this was what spooled out

Step 14: Visualize the alerts

I visualize the alert data in the Wazuh dashboard. To do this, go to the **Threat Hunting** module and add the filters in the search bar to query the alerts.

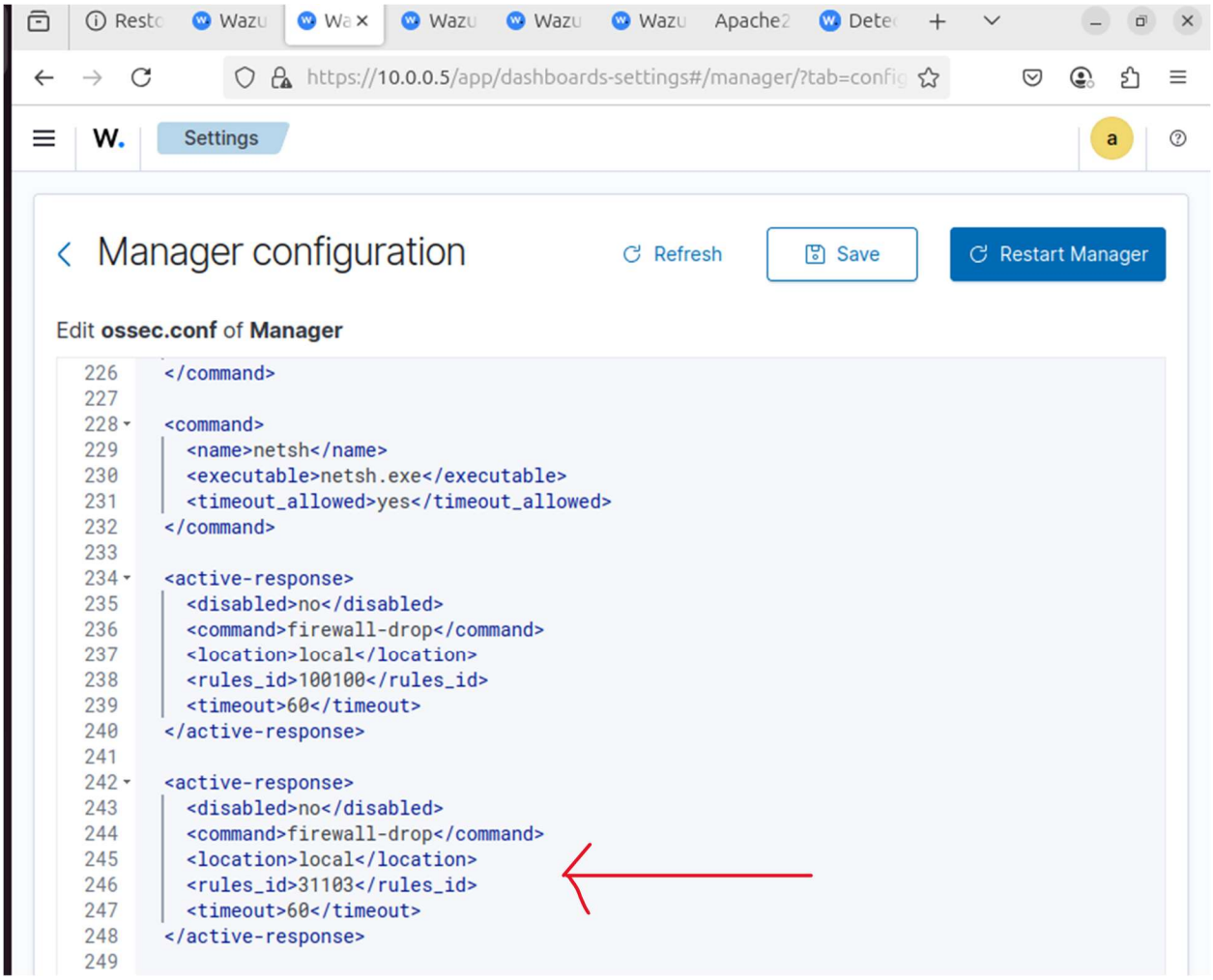
- I did an SQL injection attack to confirm blocking of the malicious ip

Using command curl -XGET "http://<UBUNTU_IP>/users/?id=SELECT+*+FROM+users";



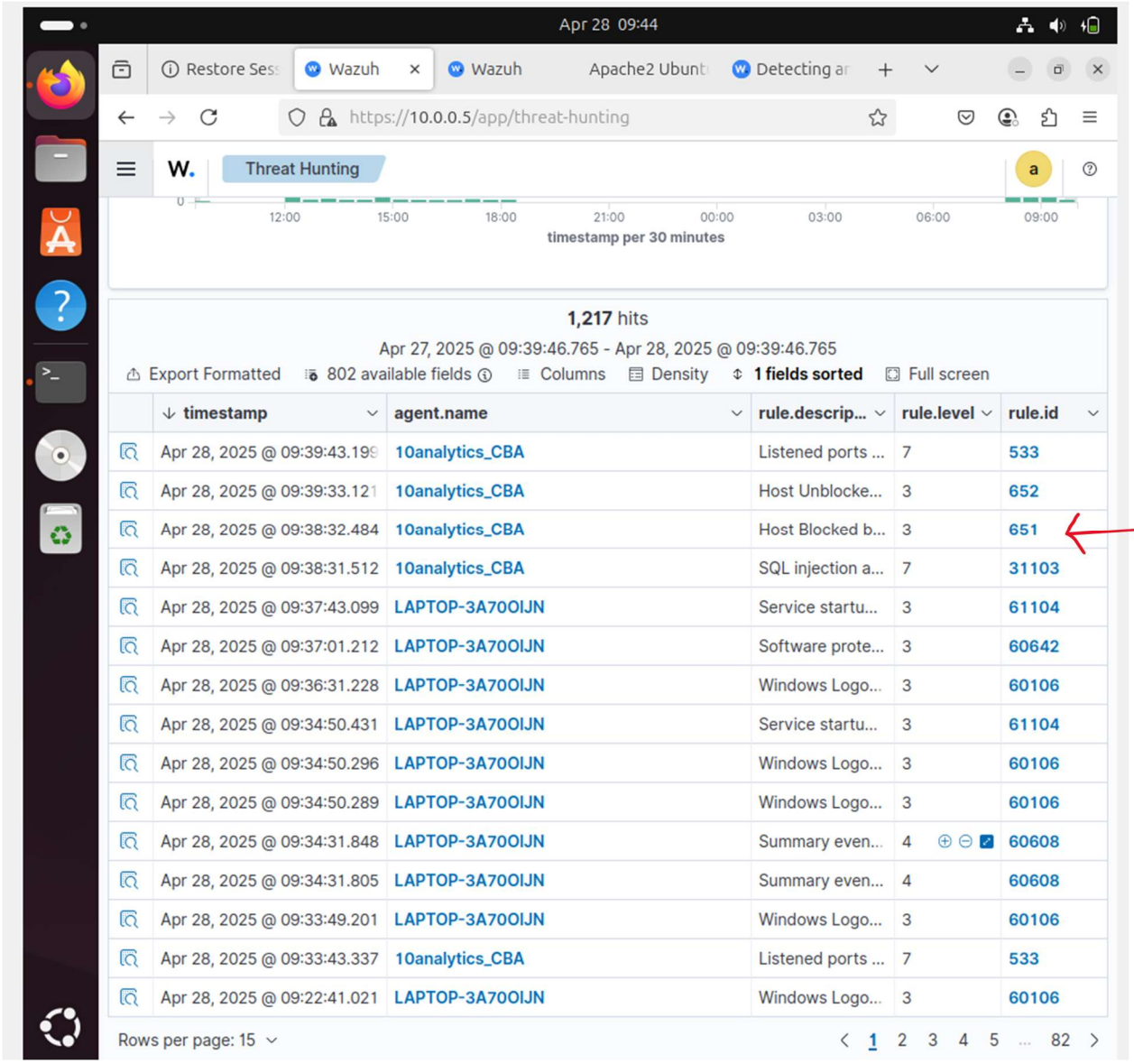
Description: This was my first visualisation of the SQL injection attack.

- Inserted 31103 in my active response on wazuh server and attacked again with my Kali



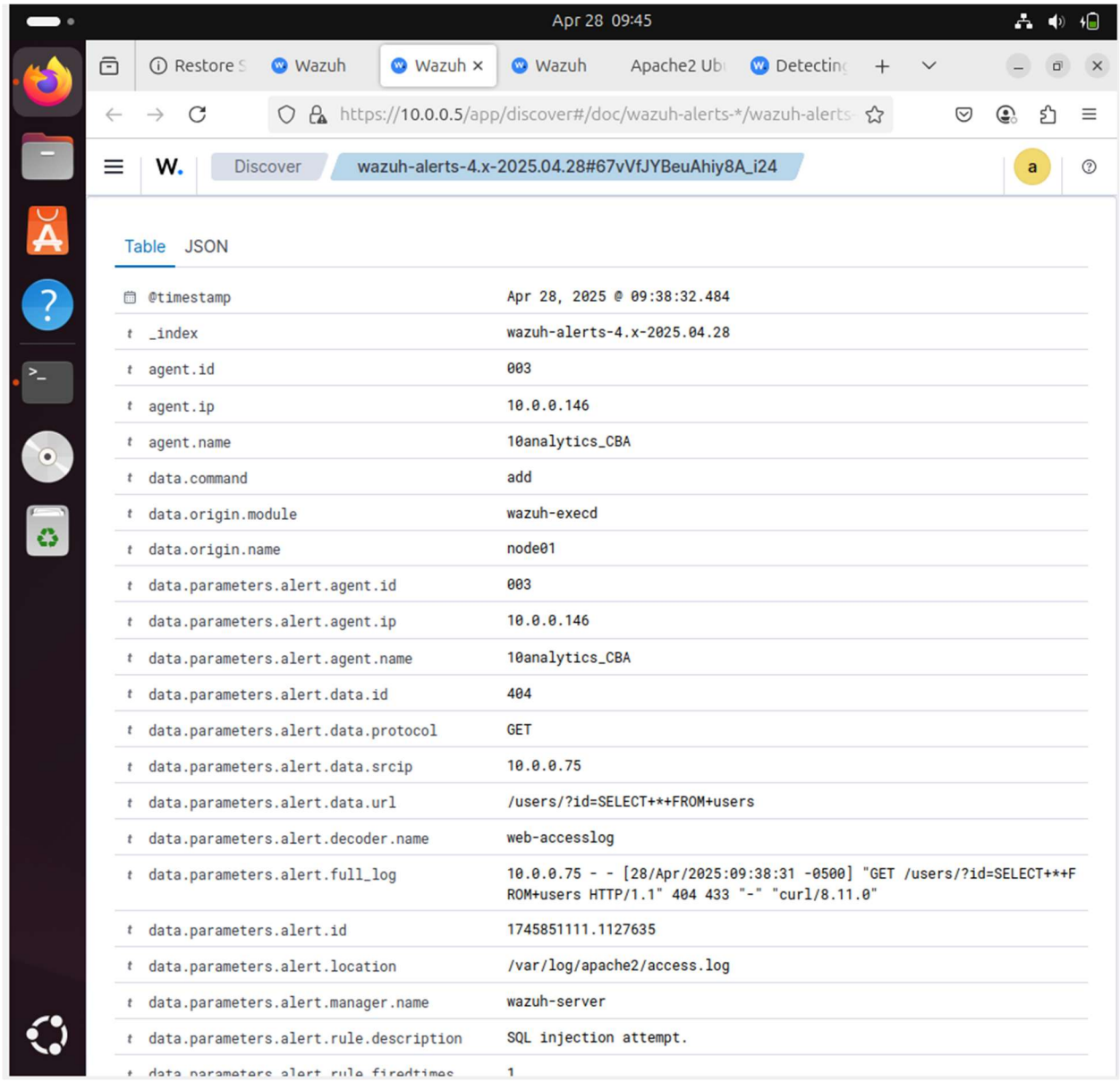
Description.This show when the rule id 31103 was added into the active response

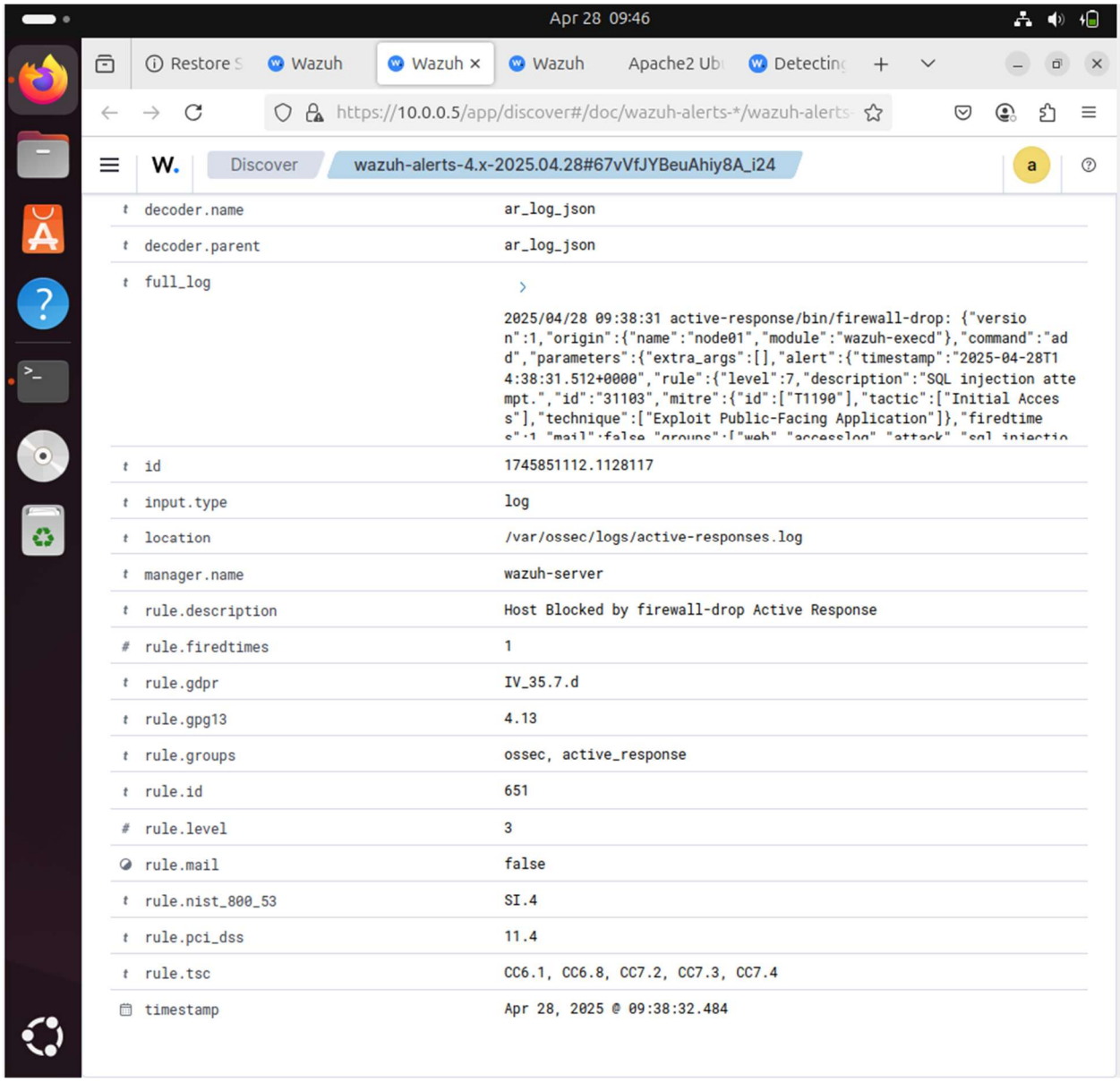
- I attacked with kali again and got these visualisations

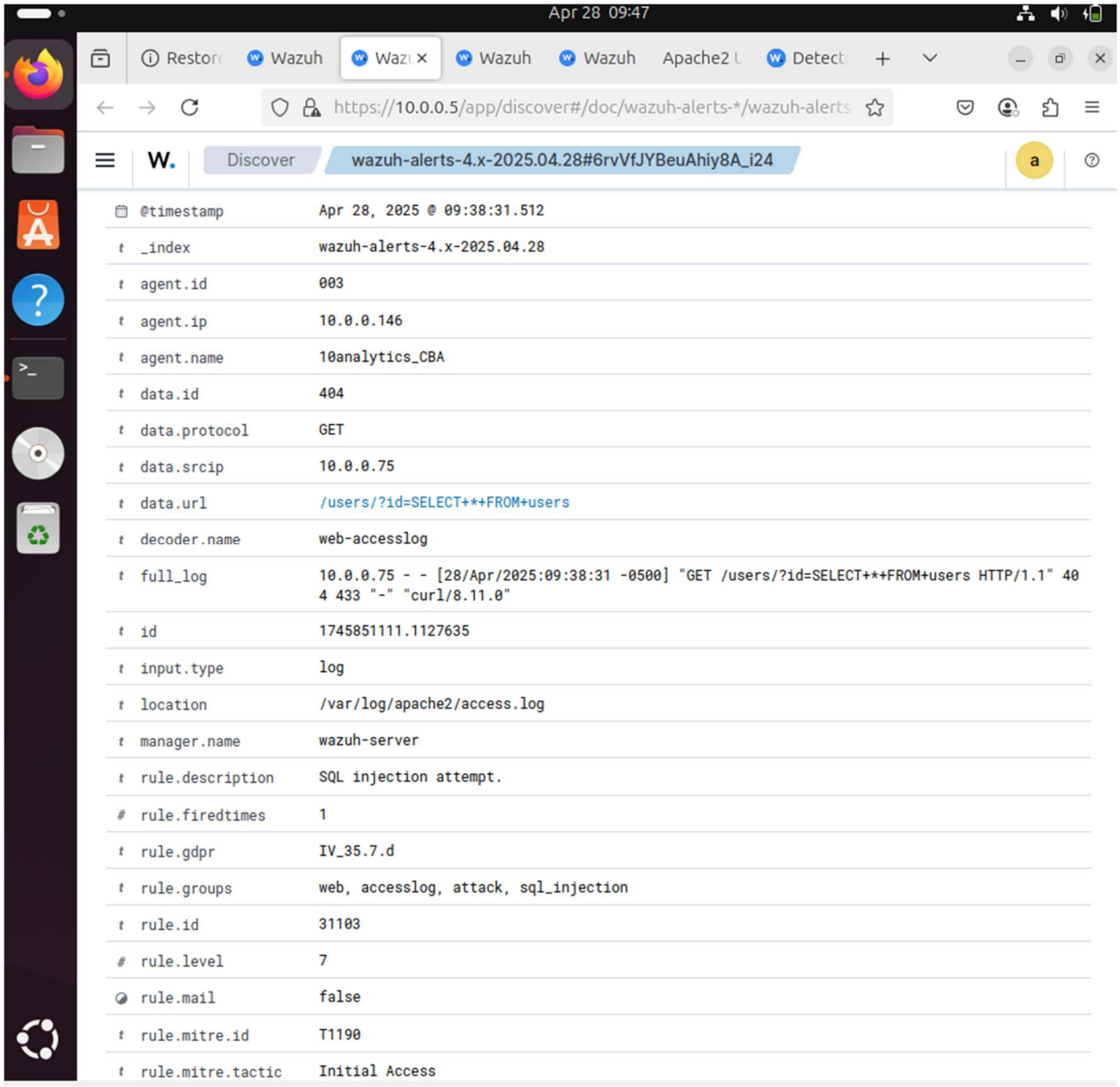


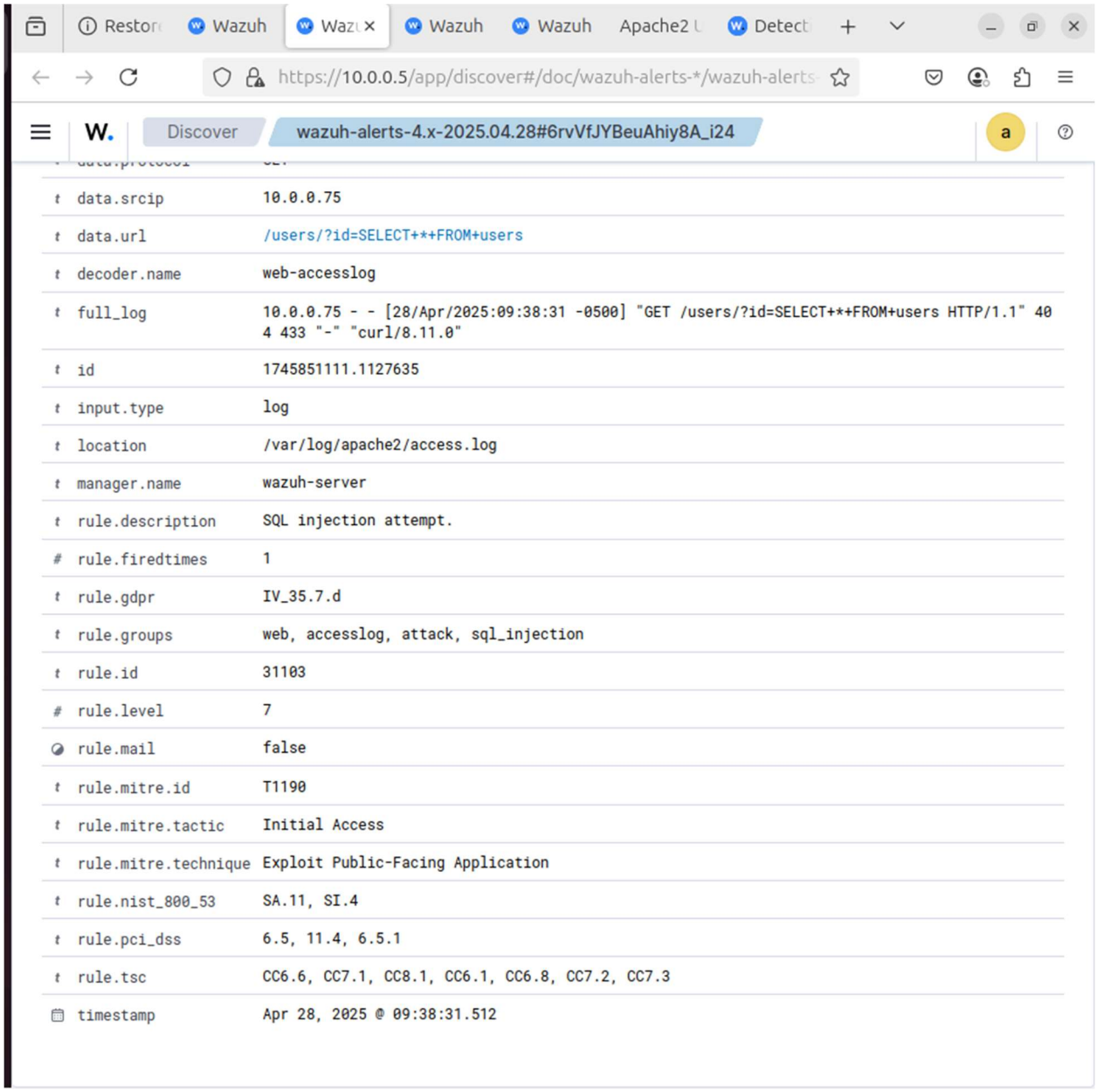
Description: This shows when the SQL injection attack was done again after the active response of 31103 was inputted, but this time it shows the host was blocked. Rule ID (651) shows that the malicious Ip address(10.0.0.75) was blocked when SQL injection was done.

Details of alerts

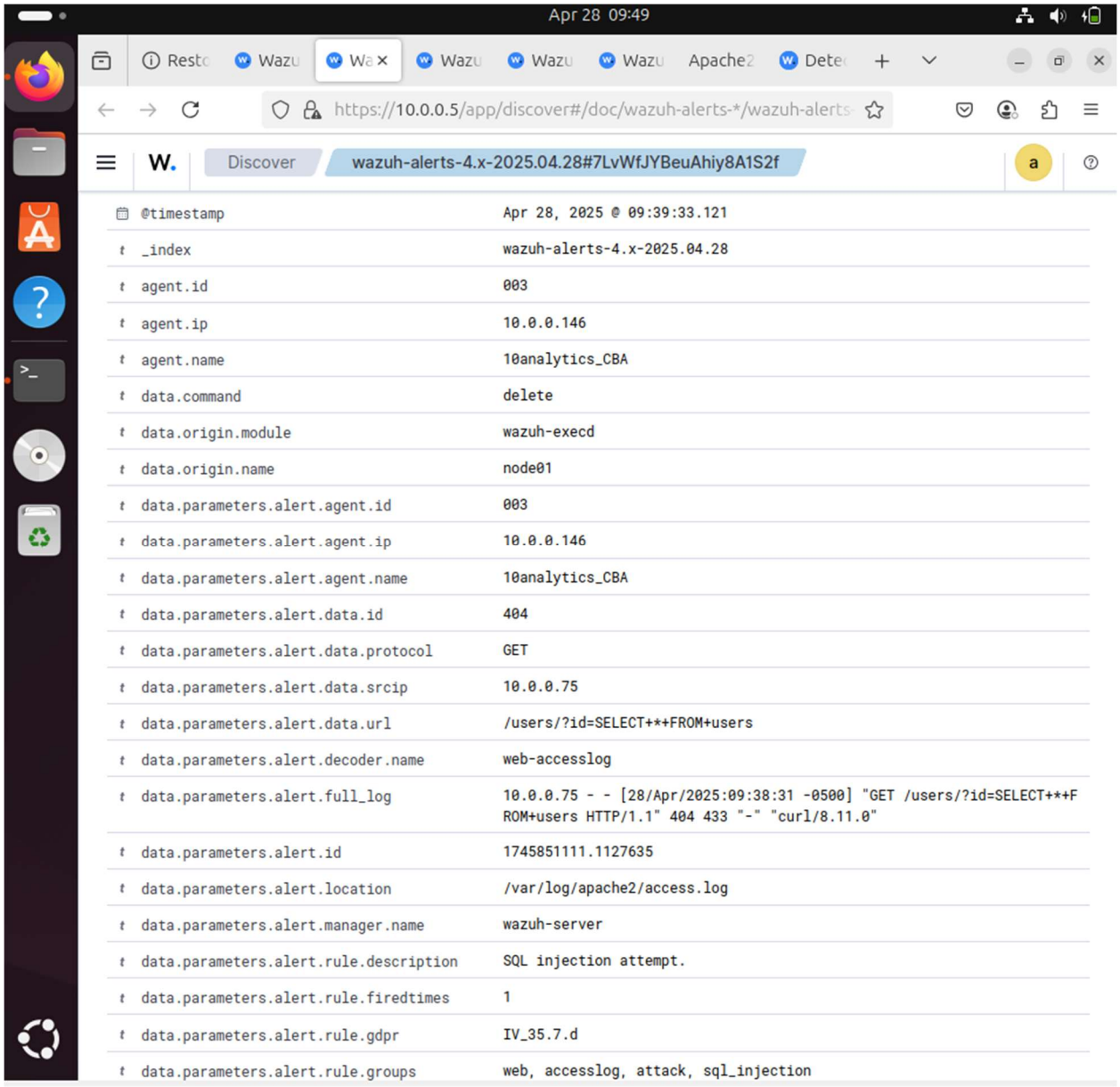

Apr 28 09:45
Restore S
Wazuh
Wazuh ×
Wazuh
Apache2 Ub
Detecting
https://10.0.0.5/app/discover#/doc/wazuh-alerts-*/wazuh-alerts-
W.
Discover
wazuh-alerts-4.x-2025.04.28#67vVfJYBeuAhiy8A_i24
a
Table
JSON
@timestamp Apr 28, 2025 @ 09:38:32.484
_index wazuh-alerts-4.x-2025.04.28
agent.id 003
agent.ip 10.0.0.146
agent.name 10analytics_CBA
data.command add
data.origin.module wazuh-execd
data.origin.name node01
data.parameters.alert.agent.id 003
data.parameters.alert.agent.ip 10.0.0.146
data.parameters.alert.agent.name 10analytics_CBA
data.parameters.alert.data.id 404
data.parameters.alert.data.protocol GET
data.parameters.alert.data.srcip 10.0.0.75
data.parameters.alert.data.url /users/?id=SELECT+*+FROM+users
data.parameters.alert.decoder.name web-accesslog
data.parameters.alert.full_log 10.0.0.75 - - [28/Apr/2025:09:38:31 -0500] "GET /users/?id=SELECT+*+FROM+users HTTP/1.1" 404 433 "-" "curl/8.11.0"
data.parameters.alert.id 1745851111.1127635
data.parameters.alert.location /var/log/apache2/access.log
data.parameters.alert.manager.name wazuh-server
data.parameters.alert.rule.description SQL injection attempt.
data.parameters.alert.rule.firedtimes 1

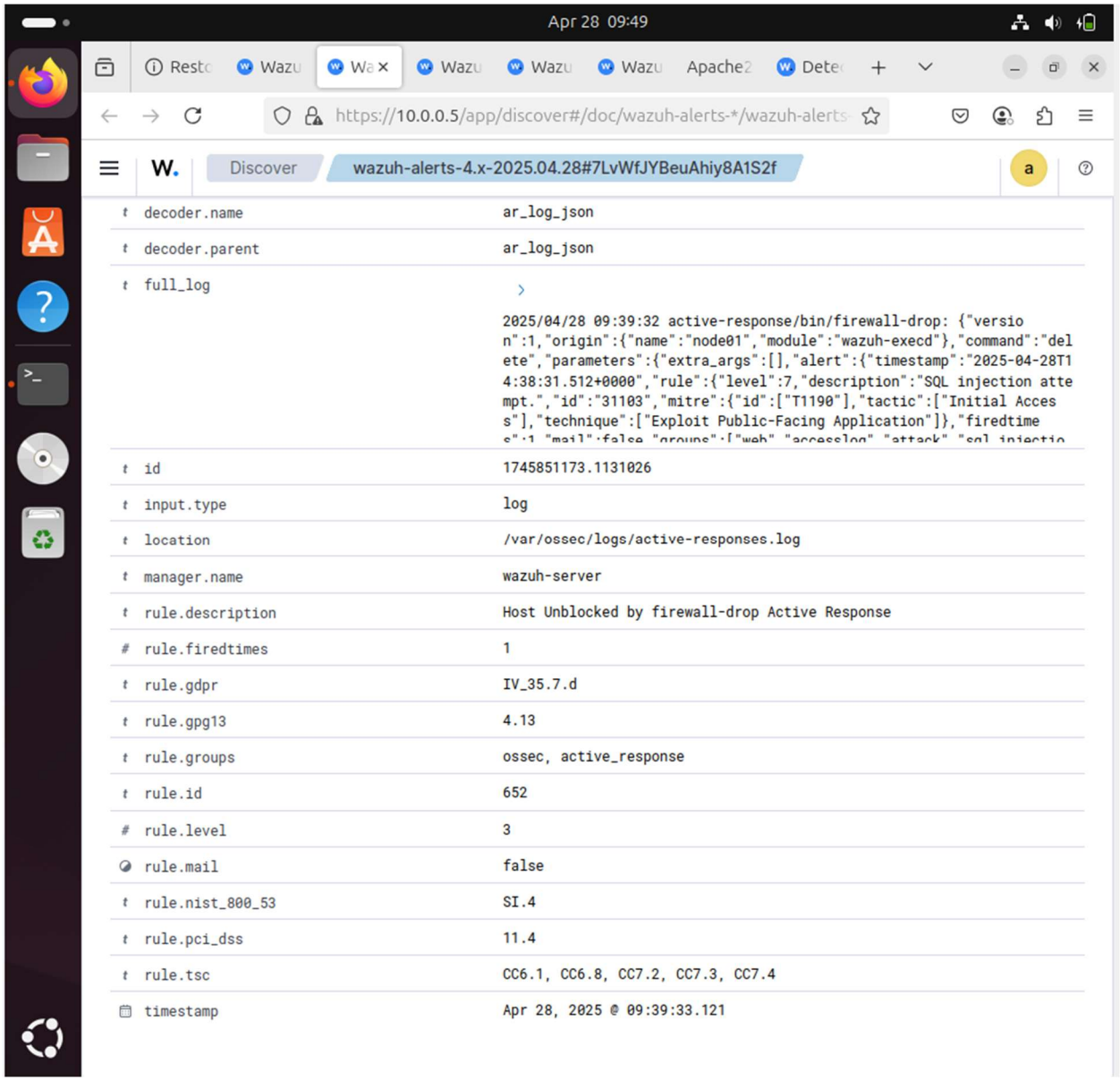

Apr 28 09:46
Restore S
Wazuh
Wazuh ×
Wazuh
Apache2 Ub
Detecting
https://10.0.0.5/app/discover#/doc/wazuh-alerts-*/wazuh-alerts-
Discover
wazuh-alerts-4.x-2025.04.28#67vVfJYBeuAhiy8A_i24
a
t decoder.name ar_log_json
t decoder.parent ar_log_json
t full_log
2025/04/28 09:38:31 active-response/bin/firewall-drop: {"version":1,"origin":{"name":"node01","module":"wazuh-execd"},"command":"add","parameters":{"extra_args":[],"alert":{"timestamp":"2025-04-28T14:38:31.512+0000","rule":{"level":7,"description":"SQL injection attempt.","id":"31103","mitre":{"id":["T1190"],"tactic":["Initial Access"],"technique":["Exploit Public-Facing Application"]},"firedtimes":1,"mail":false,"groups":["web","accesslog","attack","sql injectio
t id 1745851112.1128117
t input.type log
t location /var/ossec/logs/active-responses.log
t manager.name wazuh-server
t rule.description Host Blocked by firewall-drop Active Response
rule.firedtimes 1
t rule.gdpr IV_35.7.d
t rule.gpg13 4.13
t rule.groups ossec, active_response
t rule.id 651
rule.level 3
rule.mail false
t rule.nist_800_53 SI.4
t rule.pci_dss 11.4
t rule.tsc CC6.1, CC6.8, CC7.2, CC7.3, CC7.4
timestamp Apr 28, 2025 @ 09:38:32.484


Apr 28 09:47
Wazuh
Wazuh
Wazuh
Apache2
https://10.0.0.5/app/discover#/doc/wazuh-alerts-*/wazuh-alerts
Discover
wazuh-alerts-4.x-2025.04.28#6rvVfJYBeuAhiy8A_i24
@timestamp Apr 28, 2025 @ 09:38:31.512
_index wazuh-alerts-4.x-2025.04.28
agent.id 003
agent.ip 10.0.0.146
agent.name 10analytics_CBA
data.id 404
data.protocol GET
data.srcip 10.0.0.75
data.url /users/?id=SELECT+*+FROM+users
decoder.name web-accesslog
full_log 10.0.0.75 - - [28/Apr/2025:09:38:31 -0500] "GET /users/?id=SELECT+*+FROM+users HTTP/1.1" 404 433 "-" "curl/8.11.0"
id 1745851111.1127635
input.type log
location /var/log/apache2/access.log
manager.name wazuh-server
rule.description SQL injection attempt.
rule.firedtimes 1
rule.gdpr IV_35.7.d
rule.groups web, accesslog, attack, sql_injection
rule.id 31103
rule.level 7
rule.mail false
rule.mitre.id T1190
rule.mitre.tactic Initial Access



t	data.srcip	10.0.0.75
t	data.url	/users/?id=SELECT+*+FROM+users
t	decoder.name	web-accesslog
t	full_log	10.0.0.75 - - [28/Apr/2025:09:38:31 -0500] "GET /users/?id=SELECT+*+FROM+users HTTP/1.1" 404 433 "-" "curl/8.11.0"
t	id	1745851111.1127635
t	input.type	log
t	location	/var/log/apache2/access.log
t	manager.name	wazuh-server
t	rule.description	SQL injection attempt.
#	rule.firedtimes	1
t	rule.gdpr	IV_35.7.d
t	rule.groups	web, accesslog, attack, sql_injection
t	rule.id	31103
#	rule.level	7
◕	rule.mail	false
t	rule.mitre.id	T1190
t	rule.mitre.tactic	Initial Access
t	rule.mitre.technique	Exploit Public-Facing Application
t	rule.nist_800_53	SA.11, SI.4
t	rule.pci_dss	6.5, 11.4, 6.5.1
t	rule.tsc	CC6.6, CC7.1, CC8.1, CC6.1, CC6.8, CC7.2, CC7.3
🗓	timestamp	Apr 28, 2025 @ 09:38:31.512


Apr 28 09:49
https://10.0.0.5/app/discover#/doc/wazuh-alerts-*/wazuh-alerts-
Discover
wazuh-alerts-4.x-2025.04.28#7LvWfJYBeuAhiy8A1S2f
a
@timestamp Apr 28, 2025 @ 09:39:33.121
_index wazuh-alerts-4.x-2025.04.28
agent.id 003
agent.ip 10.0.0.146
agent.name 10analytics_CBA
data.command delete
data.origin.module wazuh-execd
data.origin.name node01
data.parameters.alert.agent.id 003
data.parameters.alert.agent.ip 10.0.0.146
data.parameters.alert.agent.name 10analytics_CBA
data.parameters.alert.data.id 404
data.parameters.alert.data.protocol GET
data.parameters.alert.data.srcip 10.0.0.75
data.parameters.alert.data.url /users/?id=SELECT+*+FROM+users
data.parameters.alert.decoder.name web-accesslog
data.parameters.alert.full_log 10.0.0.75 - - [28/Apr/2025:09:38:31 -0500] "GET /users/?id=SELECT+*+FROM+users HTTP/1.1" 404 433 "-" "curl/8.11.0"
data.parameters.alert.id 1745851111.1127635
data.parameters.alert.location /var/log/apache2/access.log
data.parameters.alert.manager.name wazuh-server
data.parameters.alert.rule.description SQL injection attempt.
data.parameters.alert.rule.firedtimes 1
data.parameters.alert.rule.gdpr IV_35.7.d
data.parameters.alert.rule.groups web, accesslog, attack, sql_injection



Response Actions Taken

- Detection and logging of suspicious web requests.
- Temporary blocking of malicious IP via firewall rules.
- Validation through Threat Hunting module in Wazuh dashboard.

- Confirmed effectiveness by observing blocked access post-attack.

Key Lessons Learned

- Proactive Monitoring: Periodic process listing and log monitoring can detect early-stage attacks.
- Active Response: Automatically blocking threat actors greatly limits the attack window.
- Security Hardening: Controlled use of potentially dangerous tools (like Netcat) and strict access policies are critical.
- Incident Response:
    - Identification and containment processes must be swift.
    - System recovery should be validated by thorough post-incident testing.
    - Continuous improvement of security controls is necessary.

Recommendations

Once an SQL injection attack is detected, it's essential to implement automated response actions to mitigate the threat. Here are some effective response actions to implement using Wazuh:

1. Identifying and accessing threats: Identifying potential security incidents through Security monitoring tools (SIEM, IDS/IPS, EDR, Firewalls),Log analysis and anomaly detection and User reports and SOC alerts

2. Containing the impact: Short-term Containment (Immediate Response)

- Isolating affected systems
- Blocking malicious IPs
- Disabling compromised accounts

Long-term Containment (Strategic Measures)

- Deploying security patches
- Changing compromised credentials Enhancing network segmentation

3. Investigating and eradicating threats:

- Removing the Root Cause:
- Deleting malware/backdoors
- Patching vulnerabilities
- Identifying attacker's tactics

Verification Steps:

- Running scans to ensure threats are removed
- Checking system logs for any reoccurrence

4. Recovering and restoring operations:

- Restoring affected systems:
- Rebuilding or reimaging systems
- Ensuring patches and updates are applied

Validating system integrity:

- Conducting penetration tests
- Continuous monitoring post-incident

5. Learning from the incident:

- Conducting a Root Cause Analysis (RCA)
- Identifying gaps in security controls
- Updating incident response procedures
- Improving staff training and awareness
- Enhancing SOC practices for future preparedness

6. Ongoing testing and evaluation:

- Develop a comprehensive incident response plan
- Regularly review and update plan
- Conduct regular training sessions
- Foster a culture of continuous improvement
- Conduct regular tabletop exercises

Conclusion

These investigations confirmed that proactive monitoring and automated responses using Wazuh effectively detect and block security threats. Unauthorized Netcat usage was successfully identified, and malicious IPs executing attacks were automatically blocked. Strengthening detection rules, automating responses, and enhancing user training are recommended to further improve security. Overall, the measures implemented will enhance10ALYTICS-DC ability in threat detection and response capabilities thereby reducing the risk of unauthorized access and potential data breaches.