SIEM'S: HOME LAB

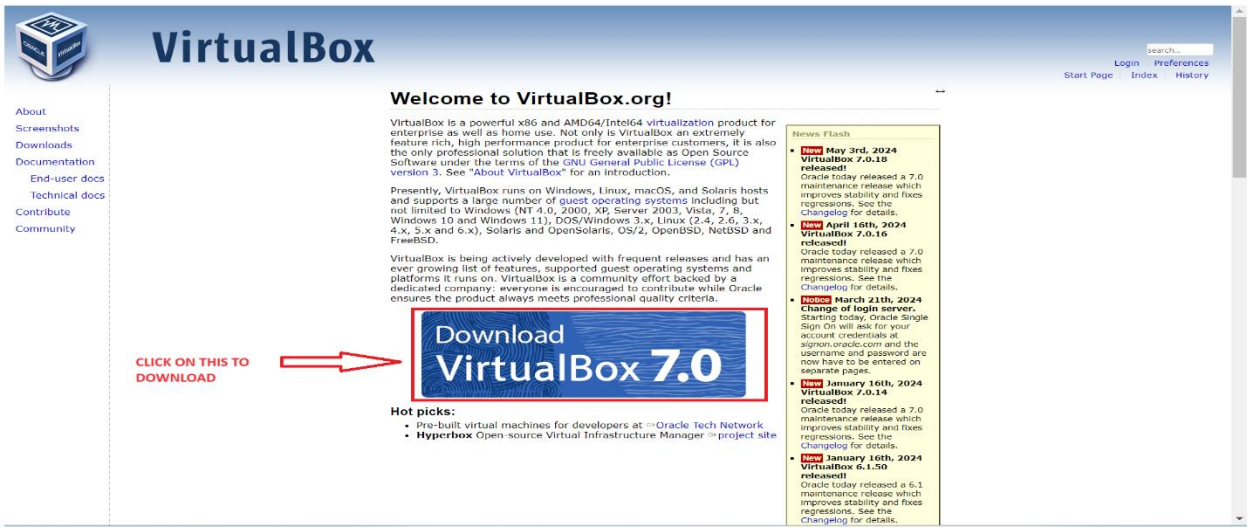
2110040122@klh.edu.in
BOBBA PANDURANGA
CYBER SECURITY

Table of Contents

1. INSTALLING VIRTUAL MACHINE

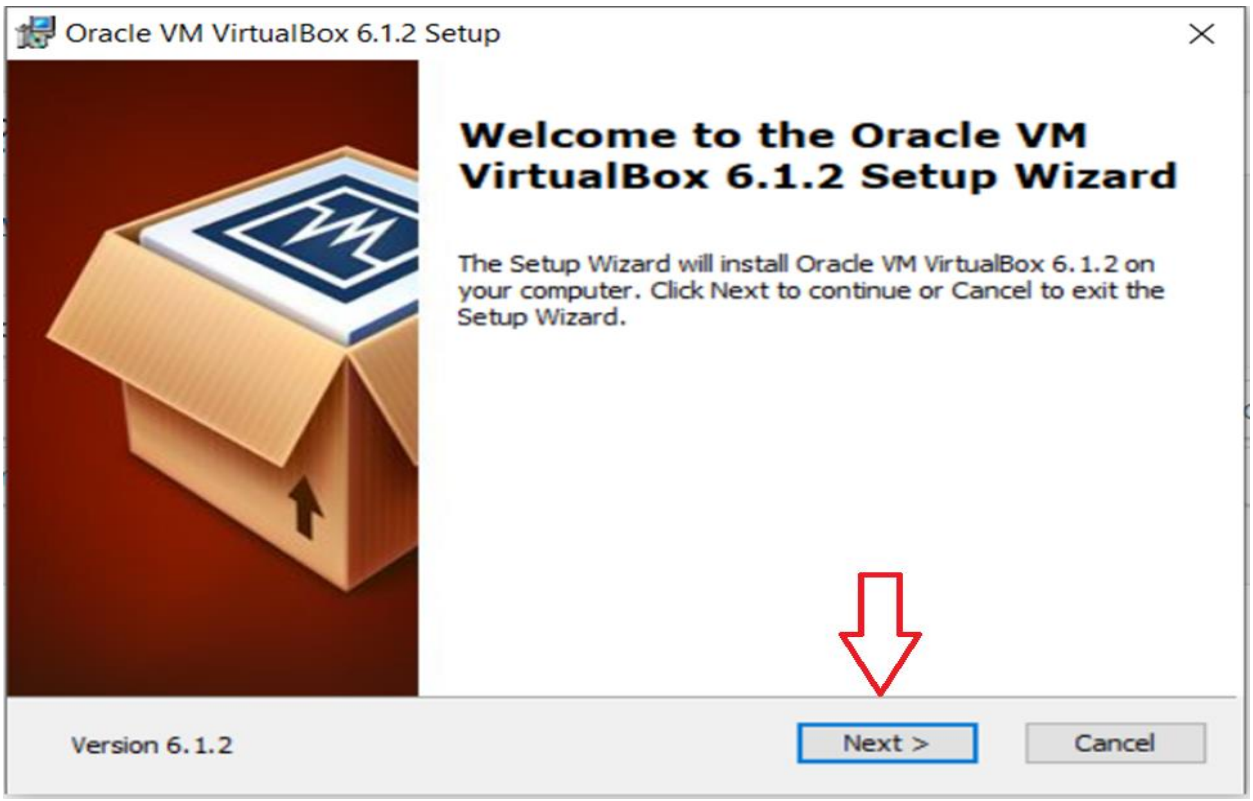
STEP 1: Go To The Virtual Machine Official Website: https://www.virtualbox.org/

STEP 2: Download The Virtual Machine By Clicking On Download Button

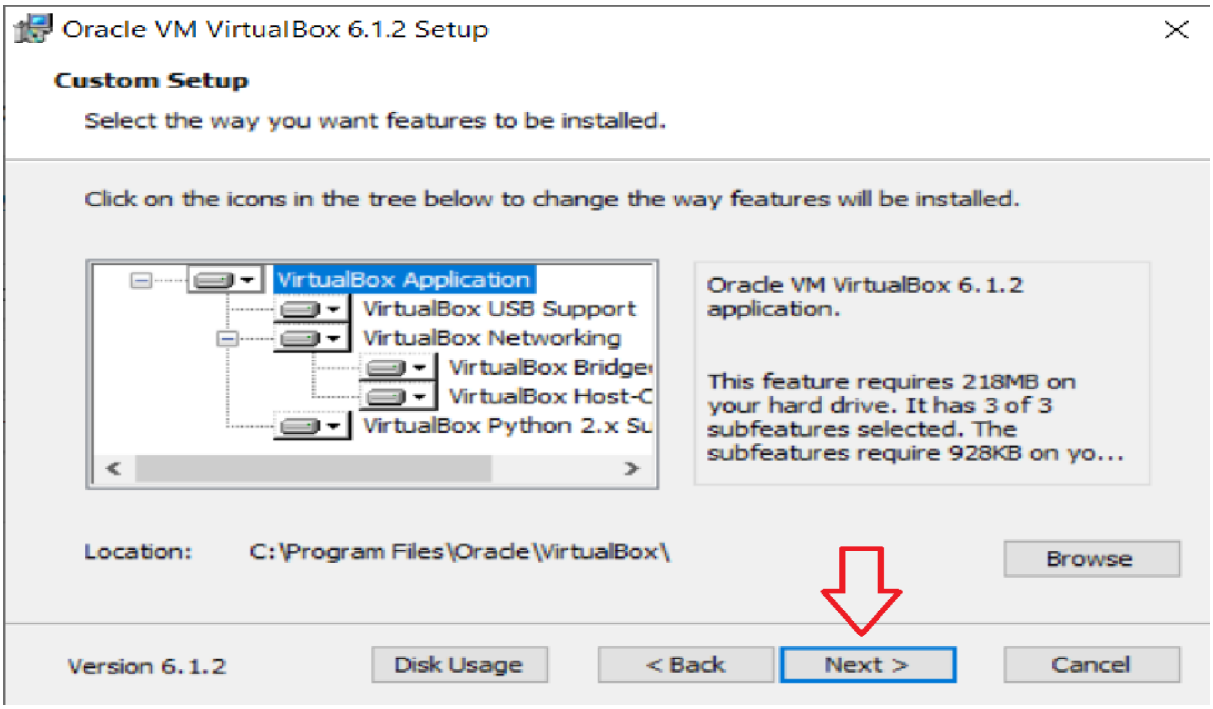


STEP 3: Select Virtualbox As Per Your OS

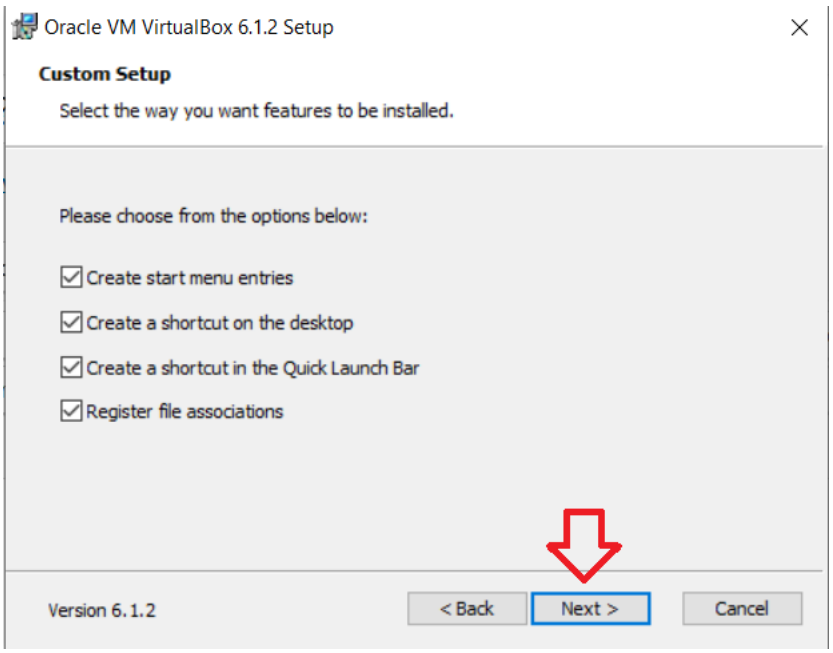
STEP 4: Install The Virtual Box



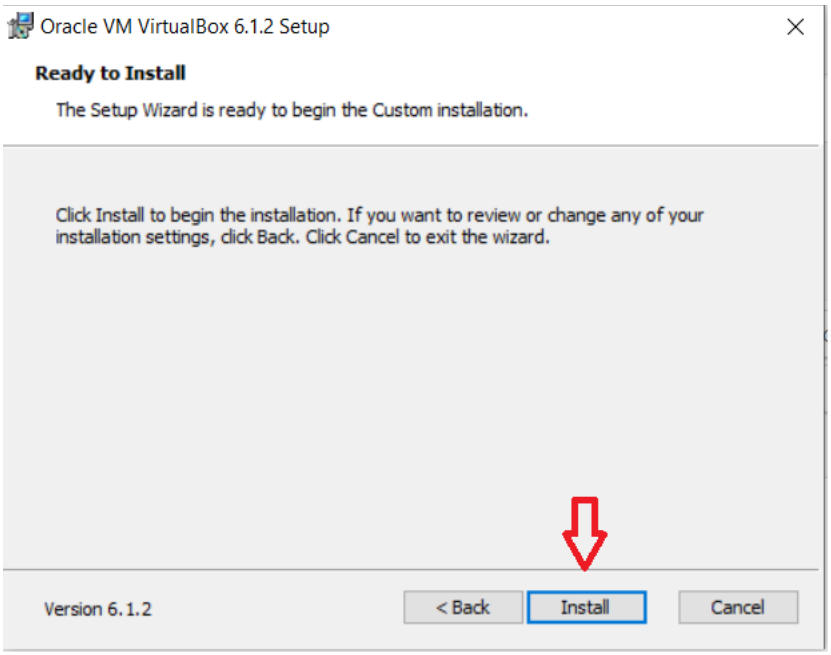
STEP 5: Select Installation Location



STEP 6: Creating Entries and Shortcuts:



STEP 7**:** Ready To Install:

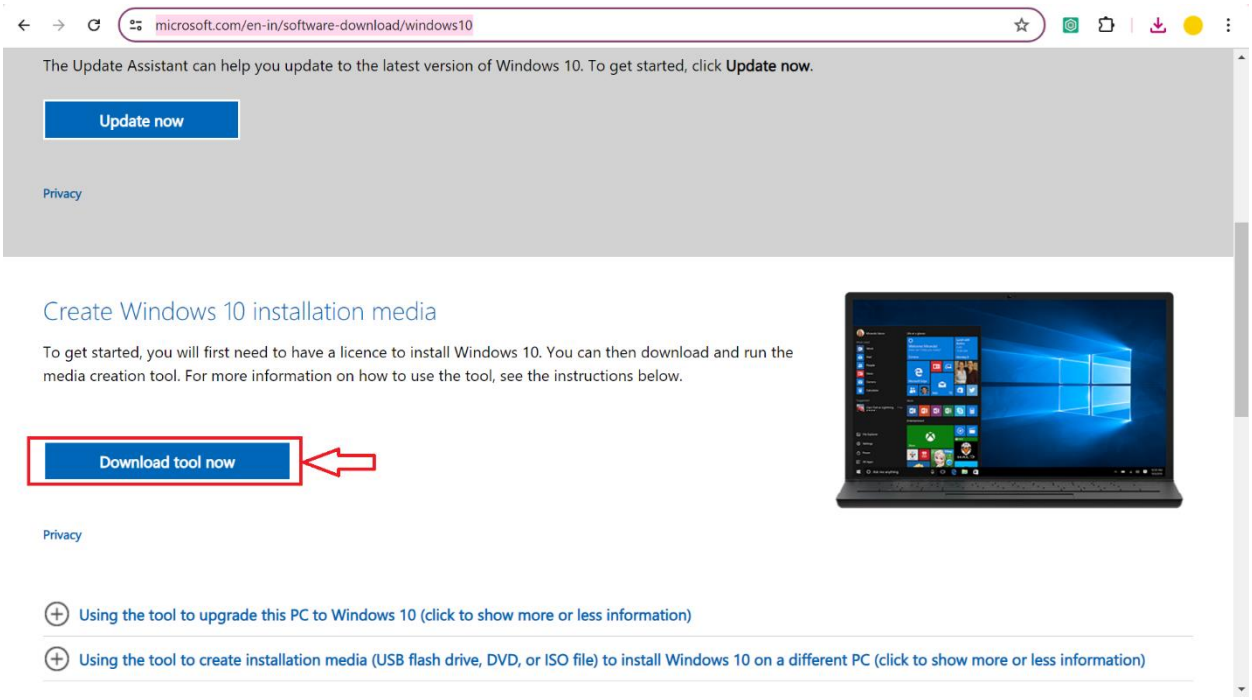


STEP 8: Finished Installation

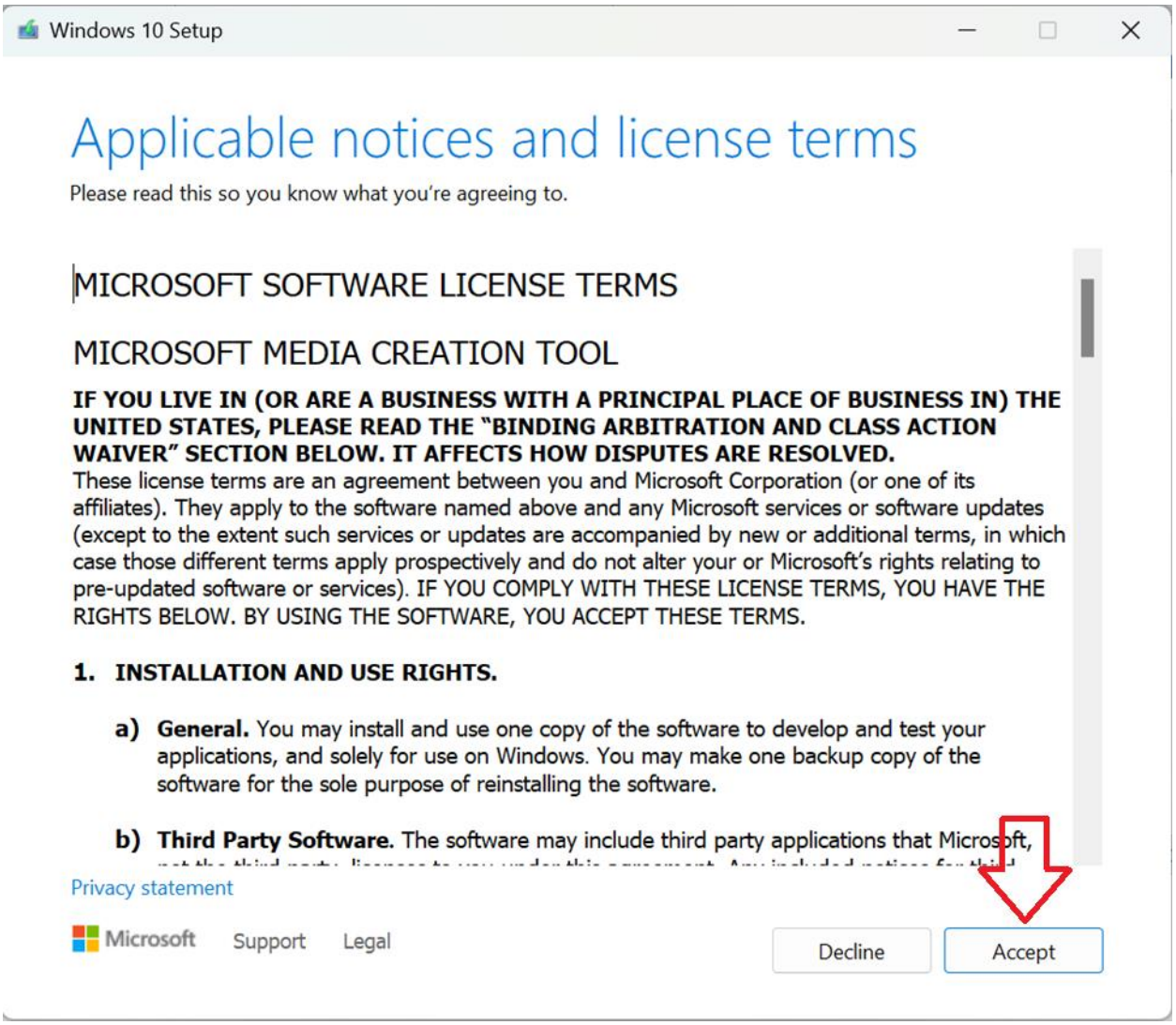
1.1 INSTALING WIN 10 PRO IN VIRTUAL MACHINE

STEP 1: GO TO WINDOWS OFFICIAL WEBSITE: https://www.microsoft.com/en-in/software-download/windows10

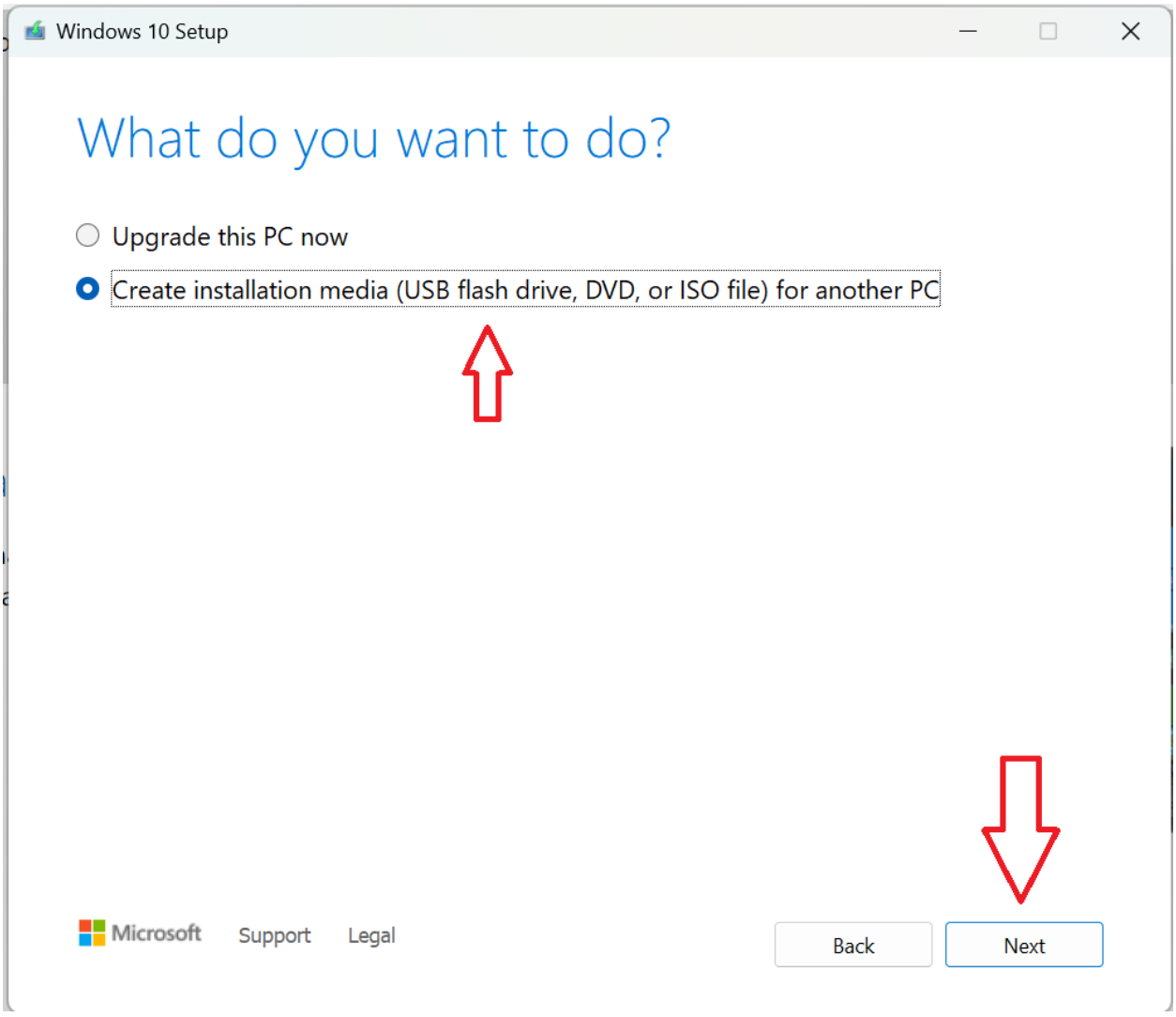
STEP 2: DOWNLOAD THE TOOL.



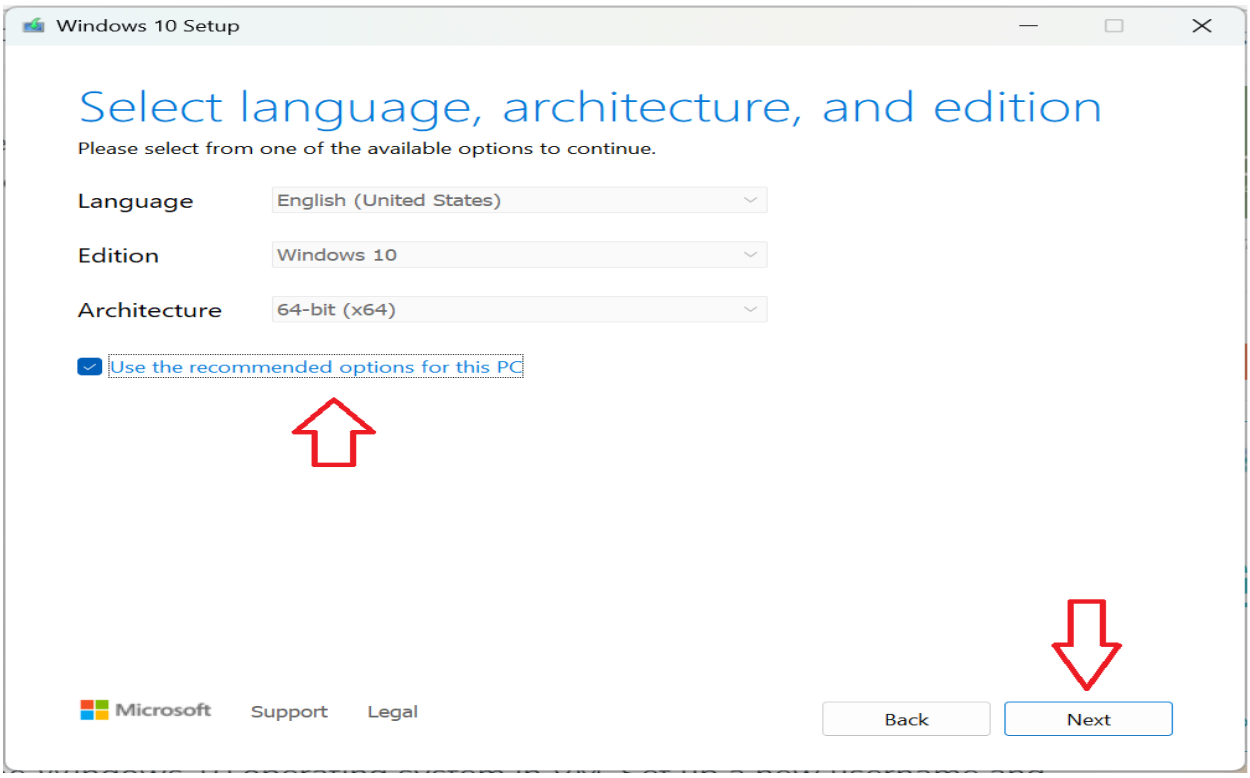
STEP 3: GO TO DOWNLOADS AND FIND THE TOOL AND RUN IT.



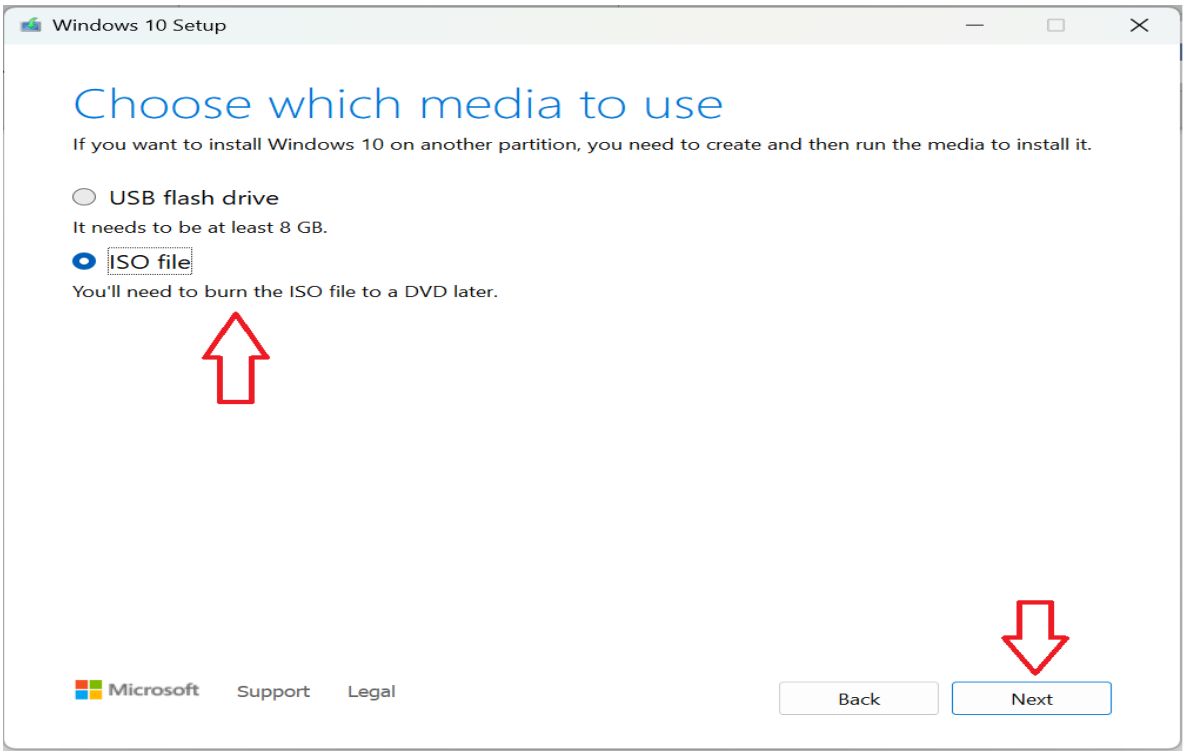
STEP 4: SELECT THE CREATE INSTALLATION MEDIA (ISO) AND THEN CLICK ON NEXT



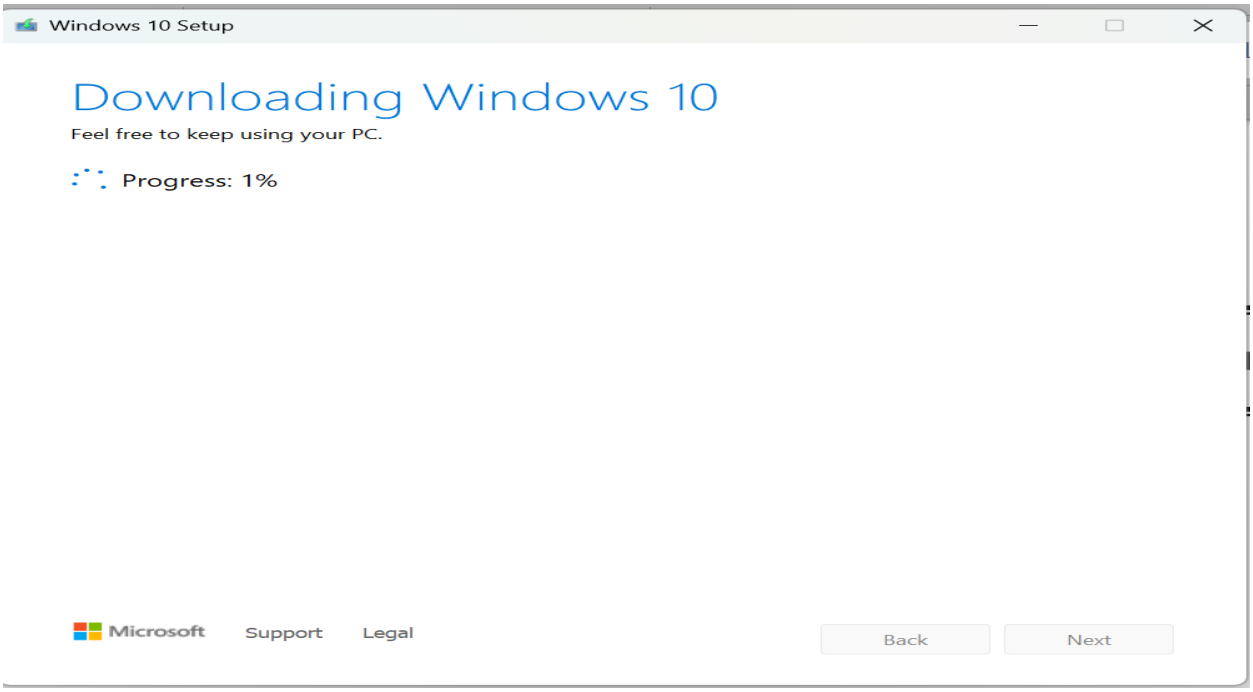
STEP 5: SELECT THE OPTIONS AS SHOWN BELOW



STEP 6: SELECT THE ISO FILE AND CLICK ON NEXT

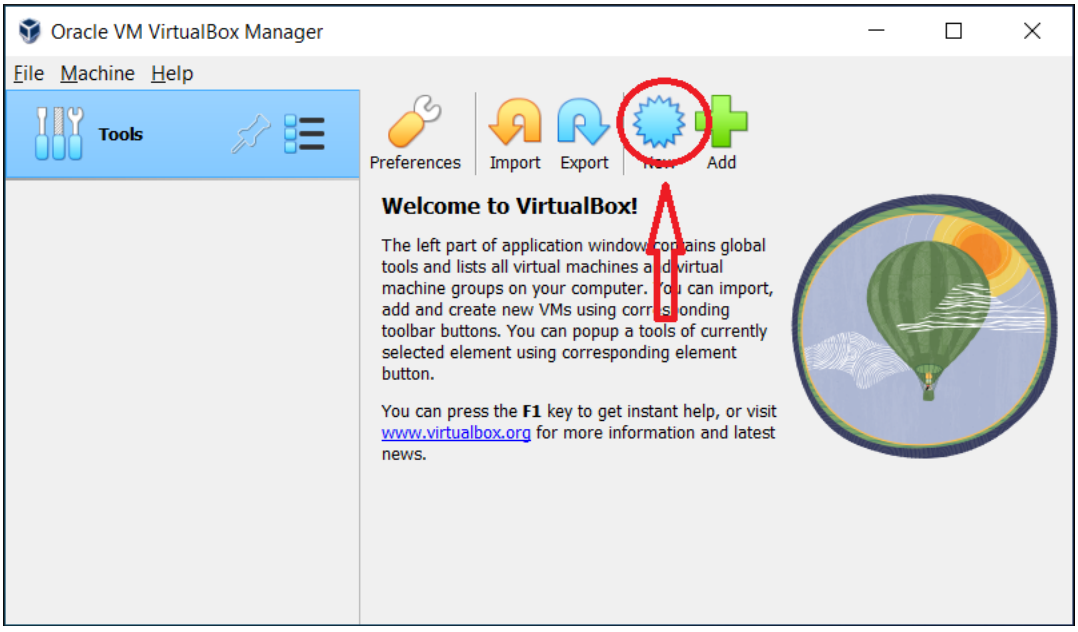


STEP 7: SELECT THE FILE DESTINATION AND CLICK ON NEXT WIN 10 FILE WILL START DOWNLOADING



INSTALING THE WIN 10 IN VIRTUALBOX

STEP 1: OPEN THE VIRTUALBOX AND SELECT THE NEW OPTION



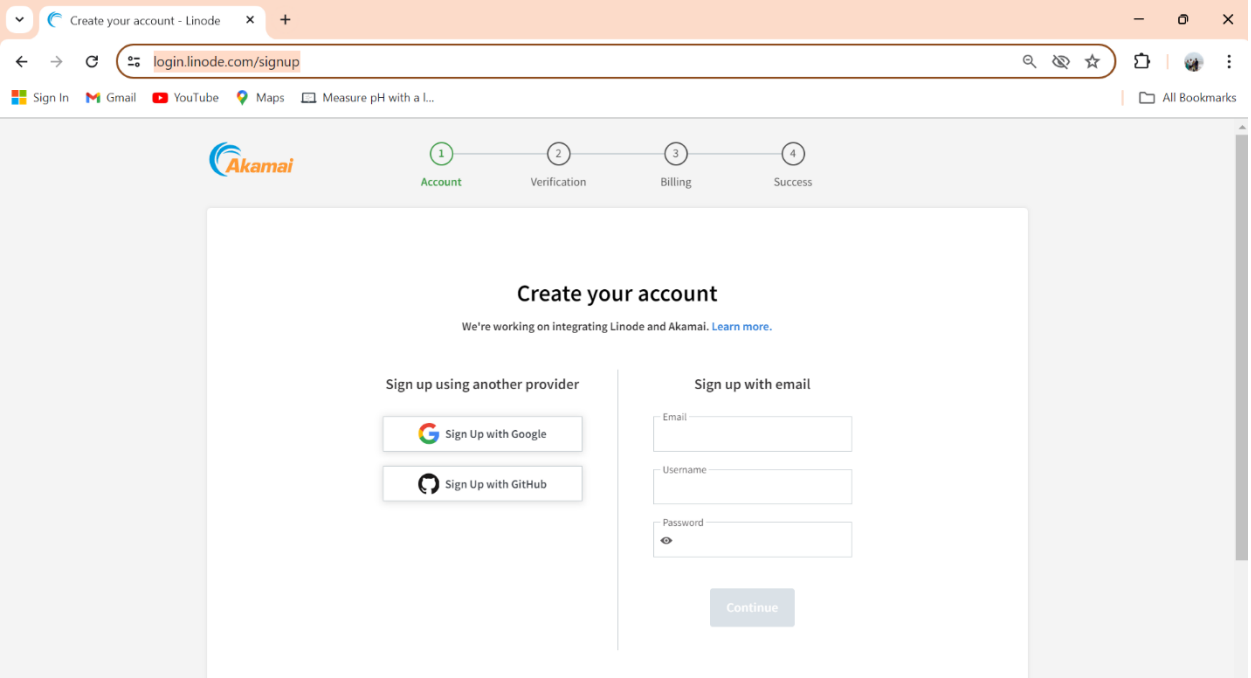
STEP 2: SELECT THE WIN 10 ISO FILE AND CLICK ON FINISH

STEP 3: That is Windows 10 has been installed on the Virtual Machine. You can now open the Windows 10 operating system in VM. Set up a new username and password and use Windows 10 in VM for various purposes

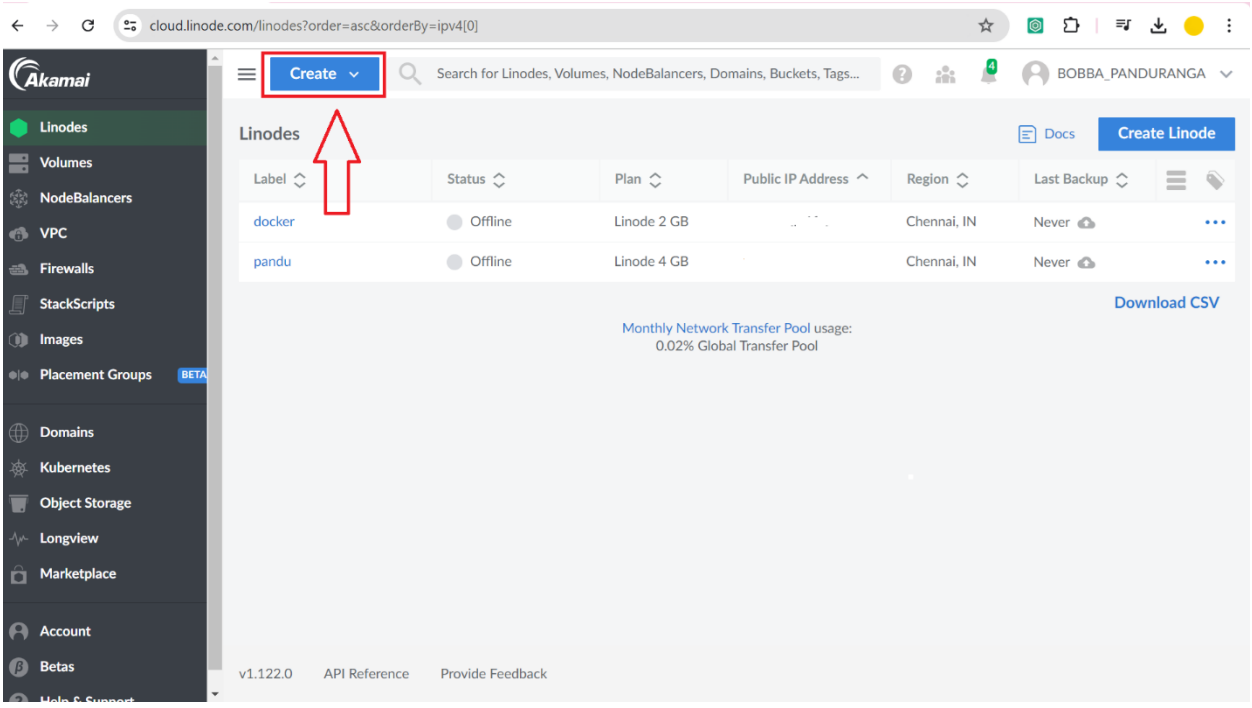
.

STAGE 3 : CREATING ubuntu SERVER AND DOCKER ON akamai cloud manger

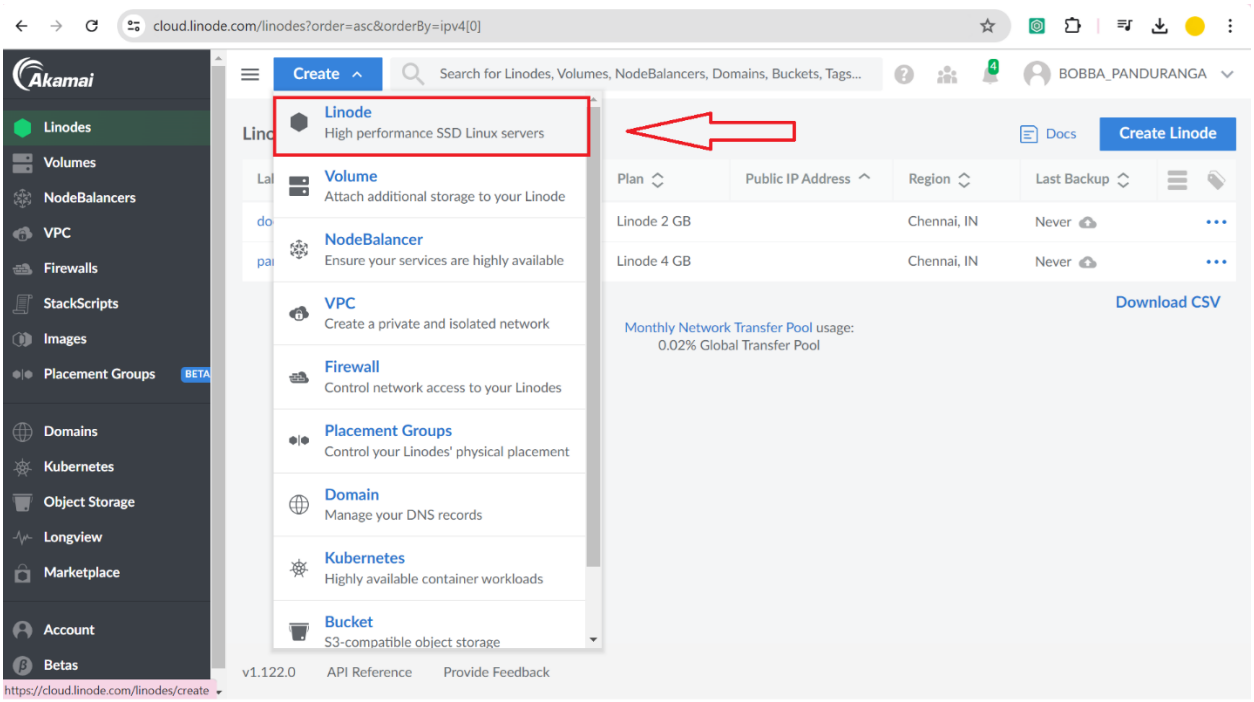
STEP 1: Go to This Website Https://Login.Linode.Com/Signup And Signup First And Create Your Account



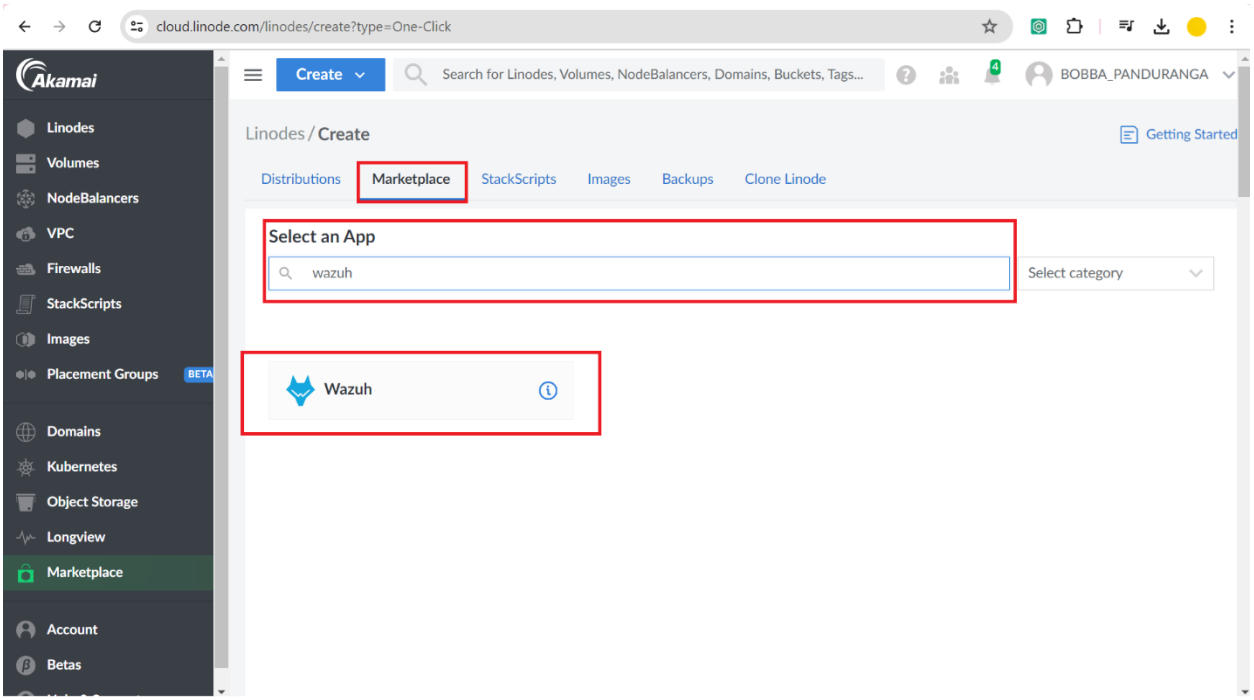
STEP 2: Click On Create To Create The Ubuntu Server



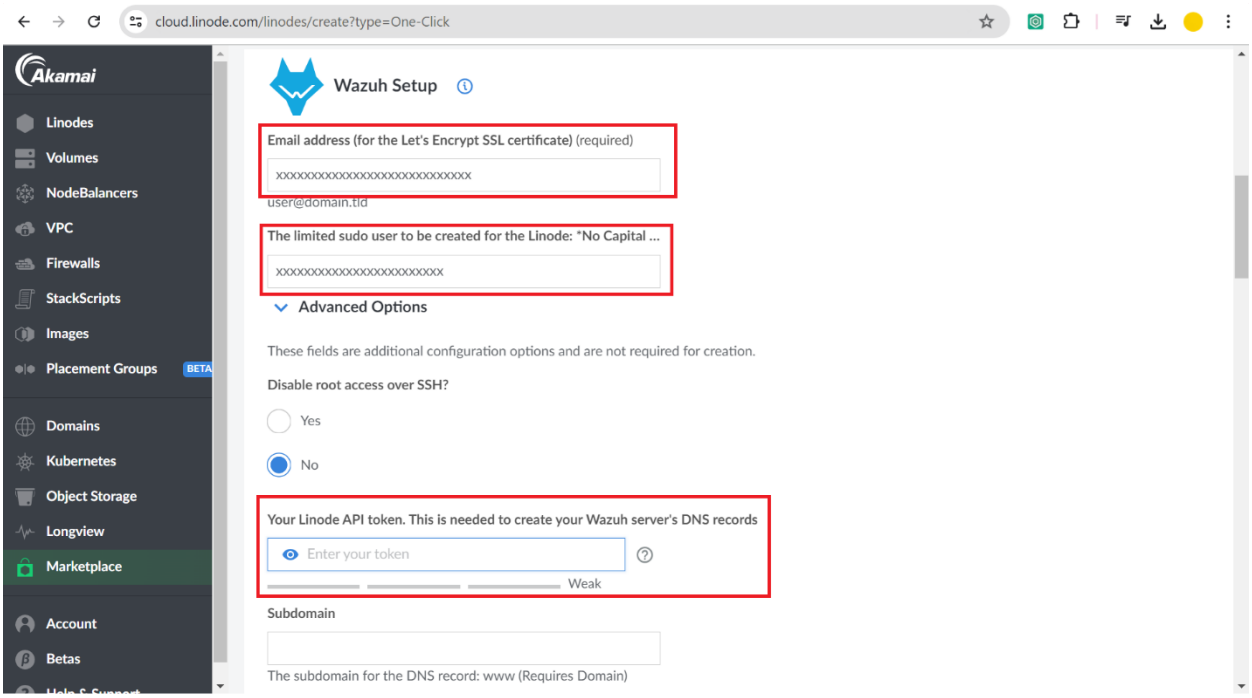
STEP 3: Click On Linode



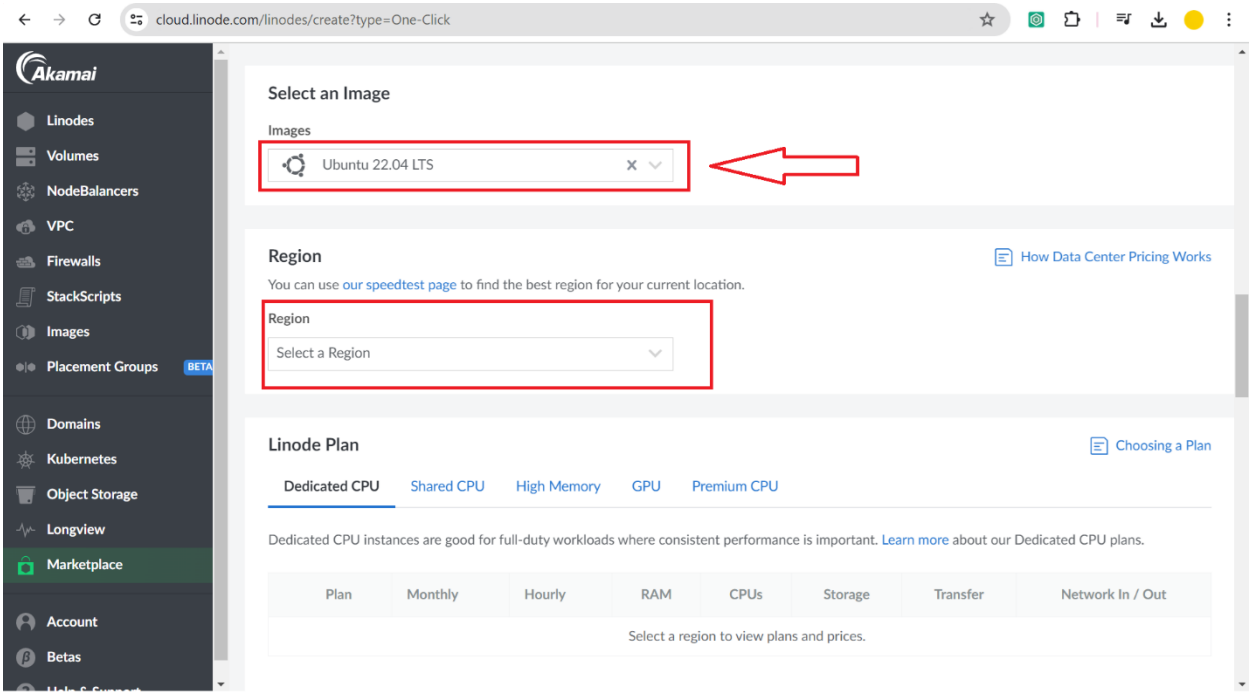
STEP 4: Select The Marketplace → Search Wazuh On Select An App →Select Wazuh:



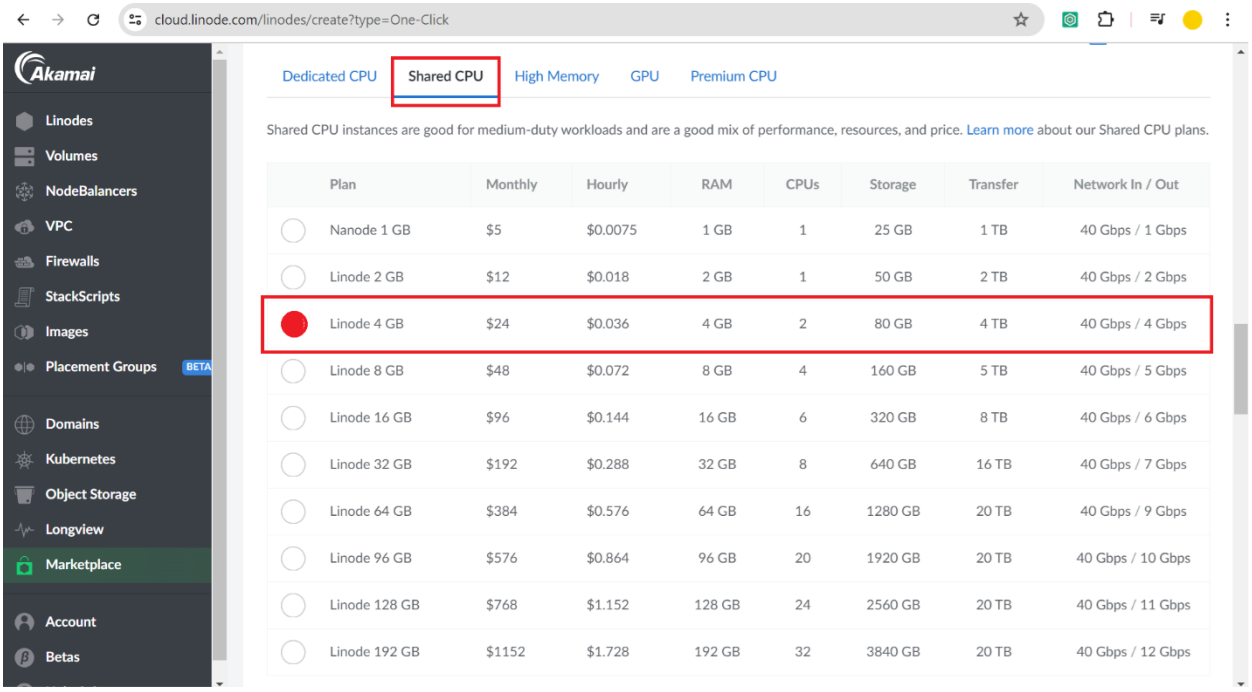
STEP 5: Scroll Down And Fill The Details As Shown Below



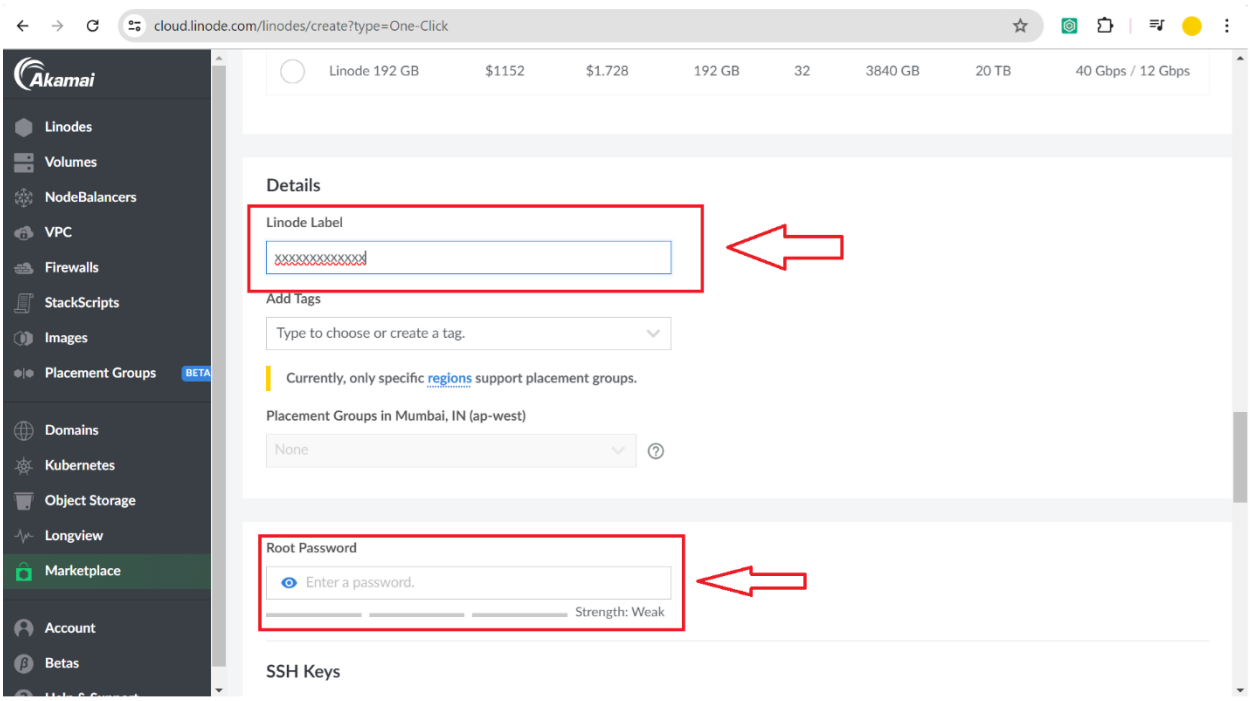
STEP 6: Select The Image File Make Sure That You Should Select The Ubuntu 22.04 LTS Only And Select Your Nearest Region



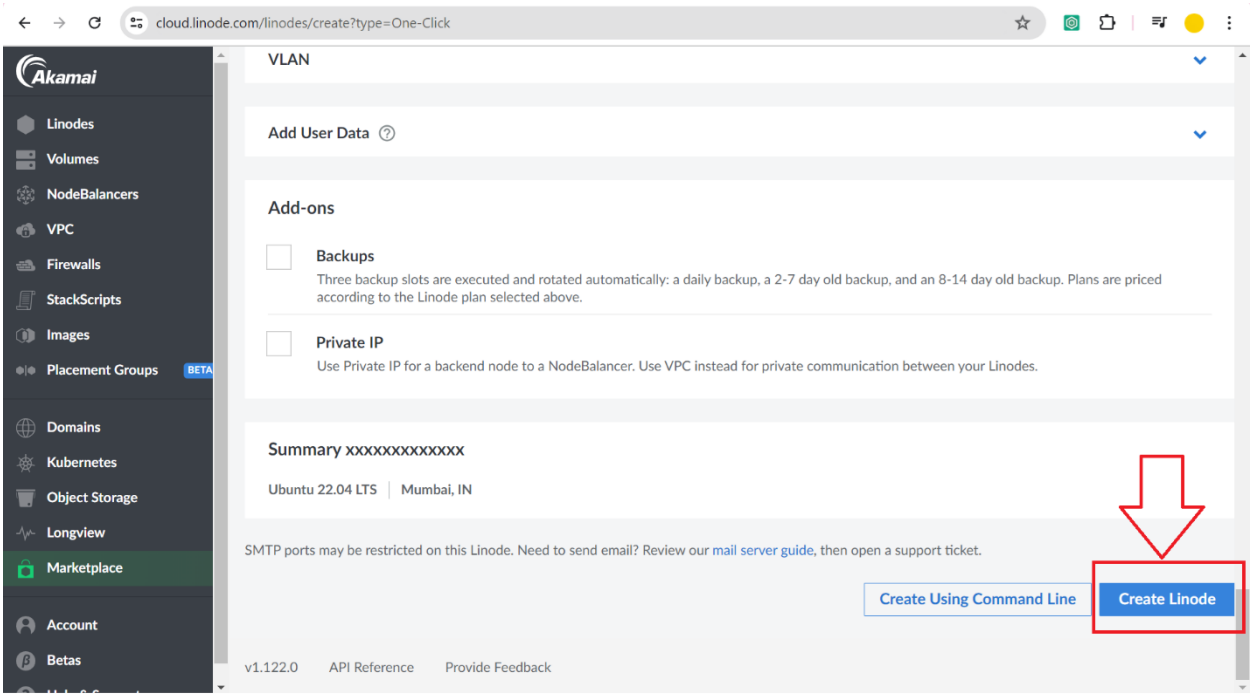
STEP 7::Select The Shared Cpu And Select The Linode 4 GB Plane As Shown Below



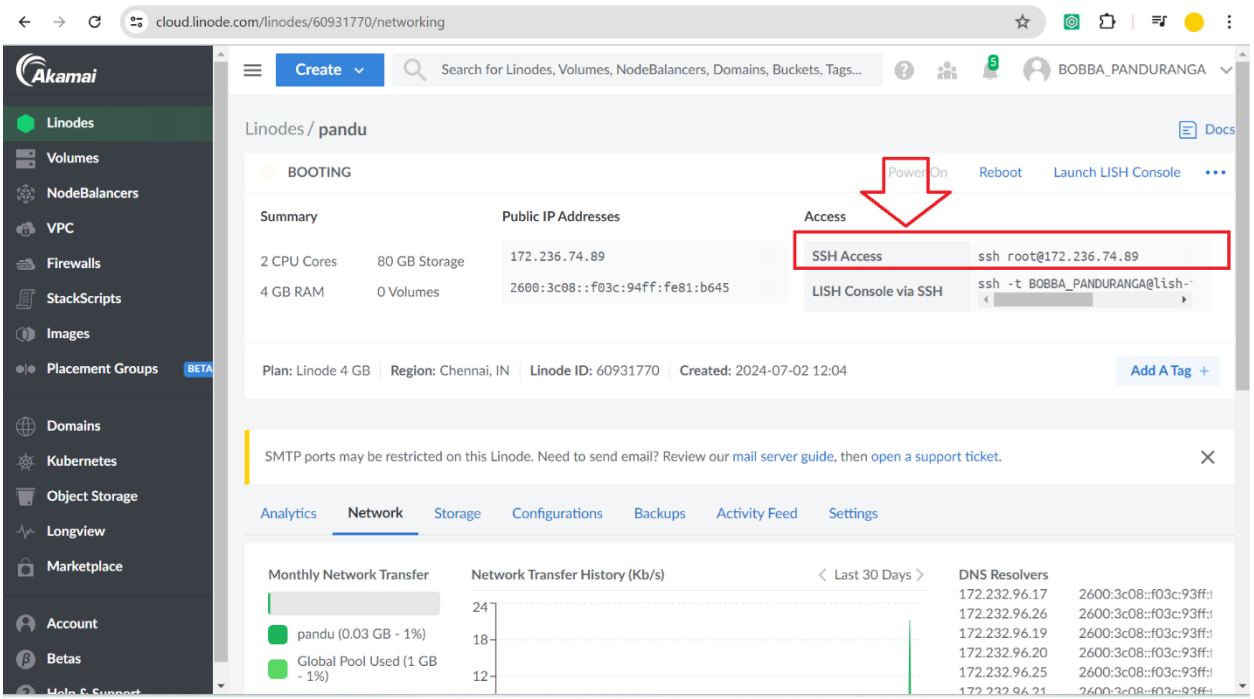
STEP 8: Now Fill The Details Linode Label As You Like And Give The Root Password As You Like Make Sure That You Remember That Root Password Again You Cant Able To Change



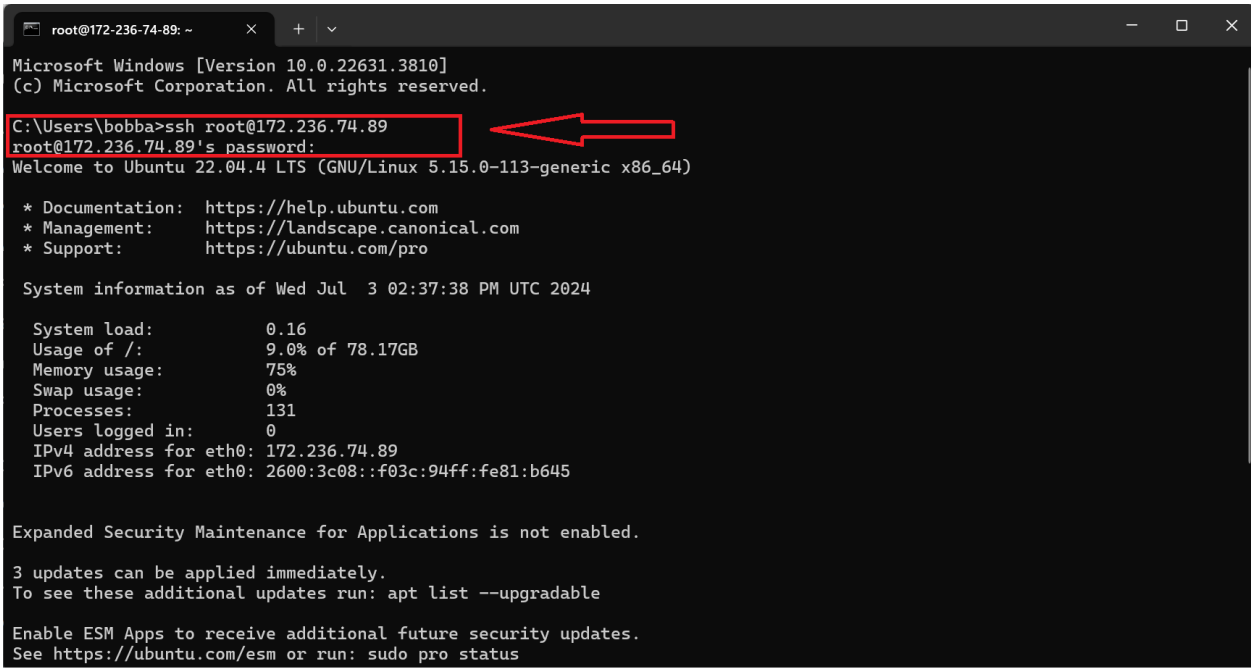
STEP 9: Click On Create The Linode Now You Are Good To Go



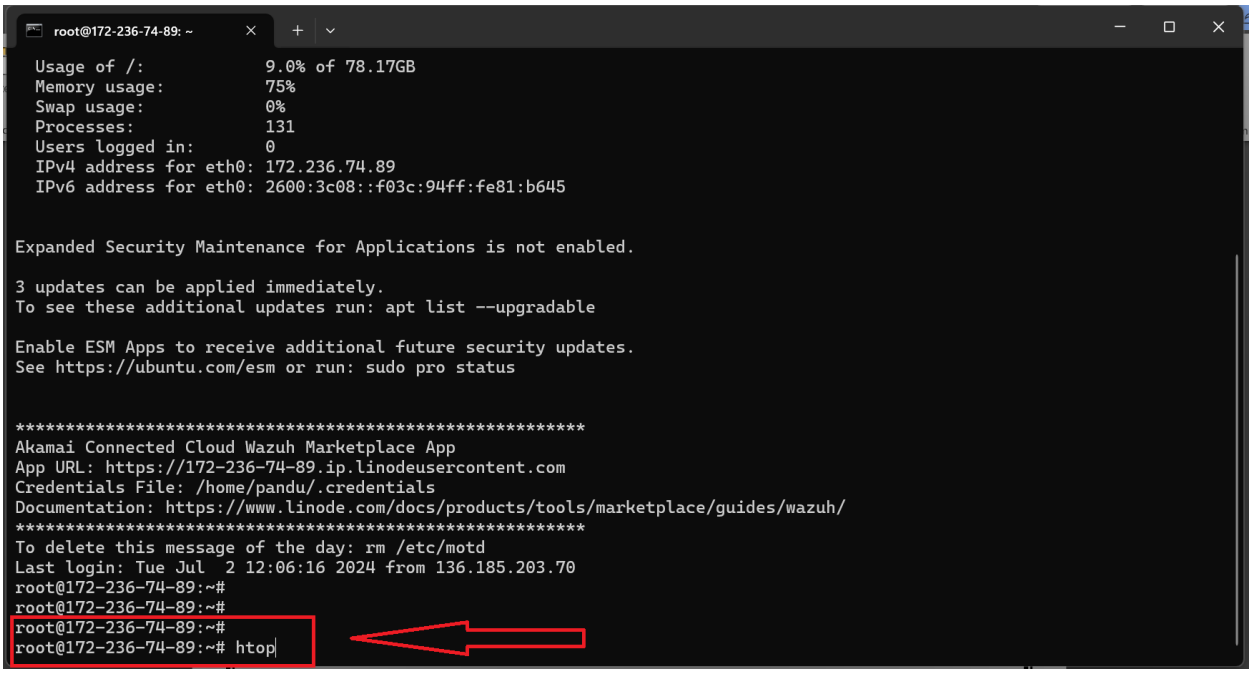
STEP 10: After creating the server now copy the SSH ACCESS CODE

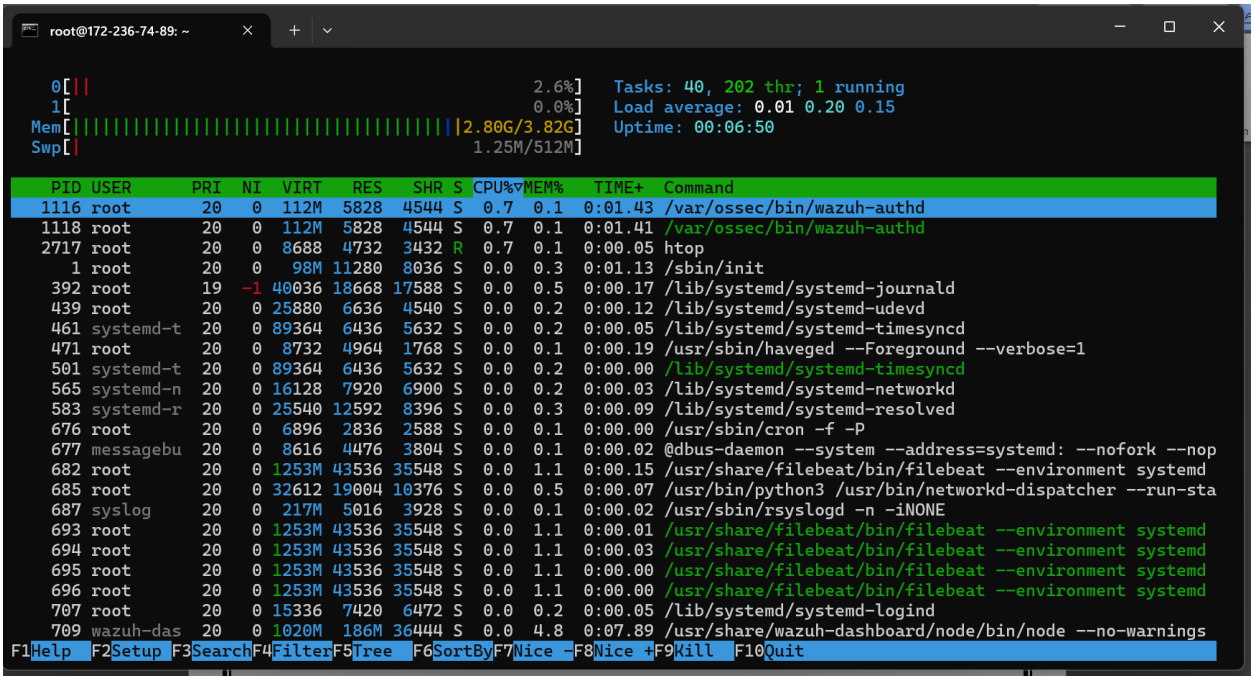


STEP 11: GO TO CMD IN WINDOWS PAST THE SSH ACCESS CODE And Enter The Password As You Given Before



STE:P12 to see the internal process such as install





- Use the htpasswd command to reset the password for the Wazuh web interface. Replace admin with the appropriate username if different.

bash
Copy code
htpasswd -c /var/ossec/api/configuration/auth/user admin

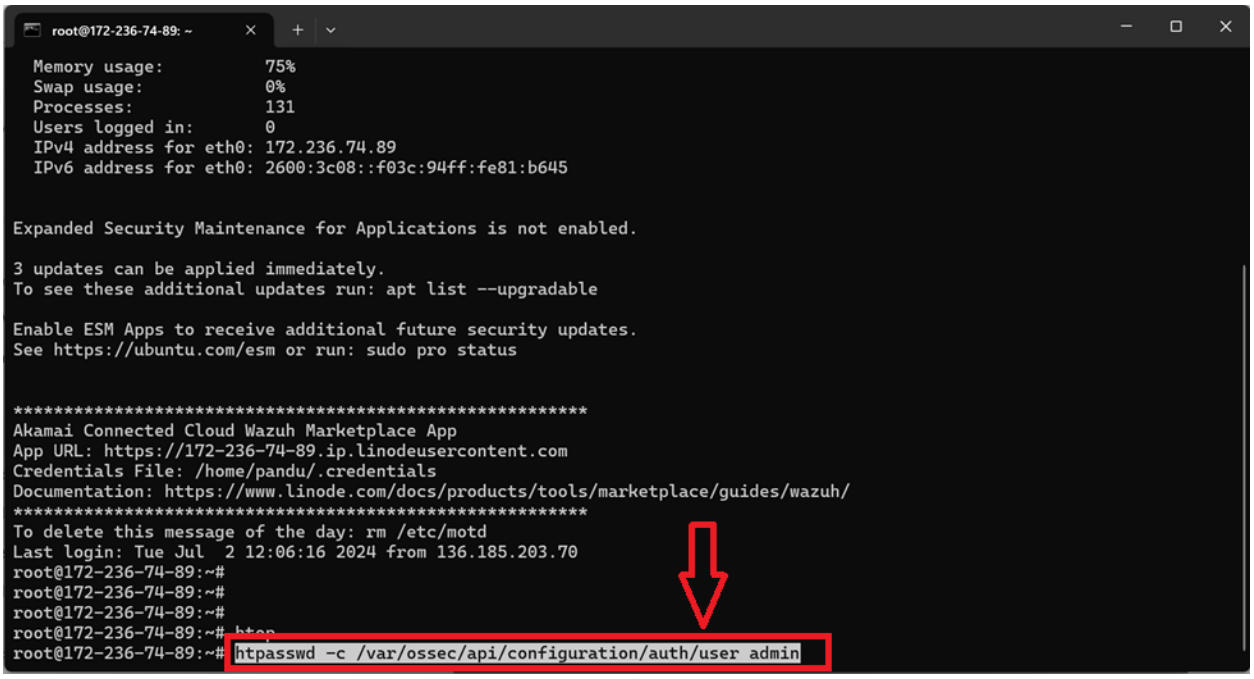
- You will be prompted to enter and confirm the new password.

Restart Wazuh API:

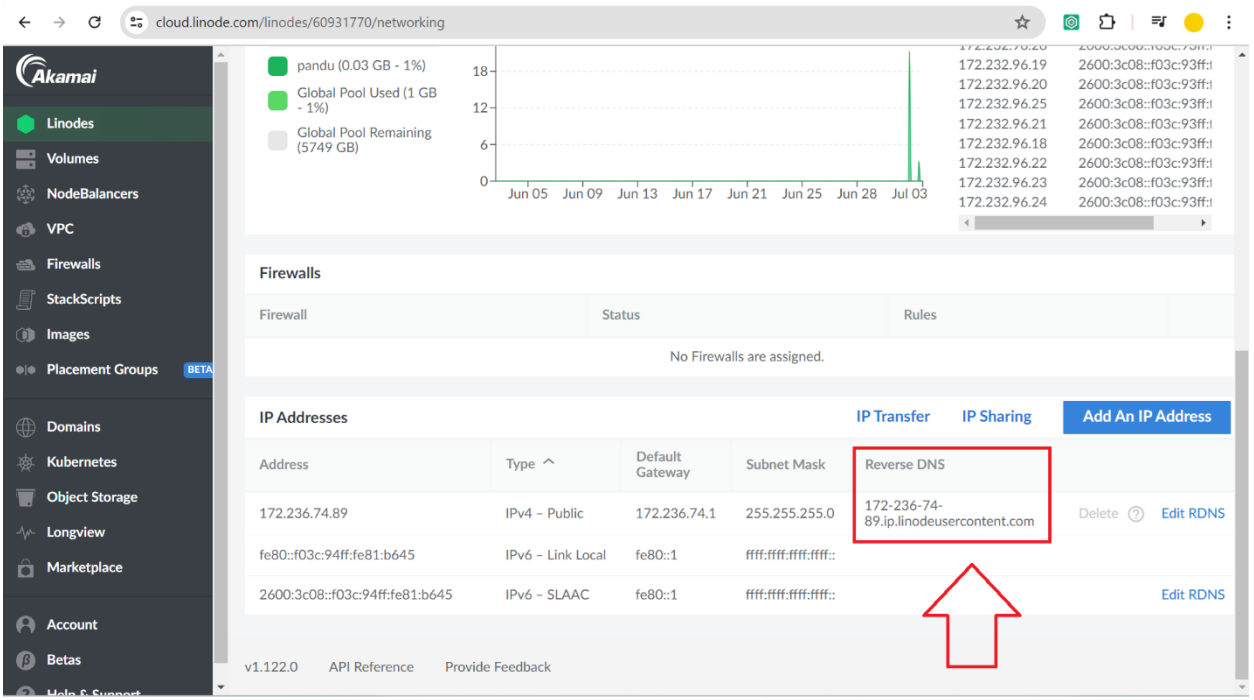
- After resetting the password, restart the Wazuh API service to apply the changes.

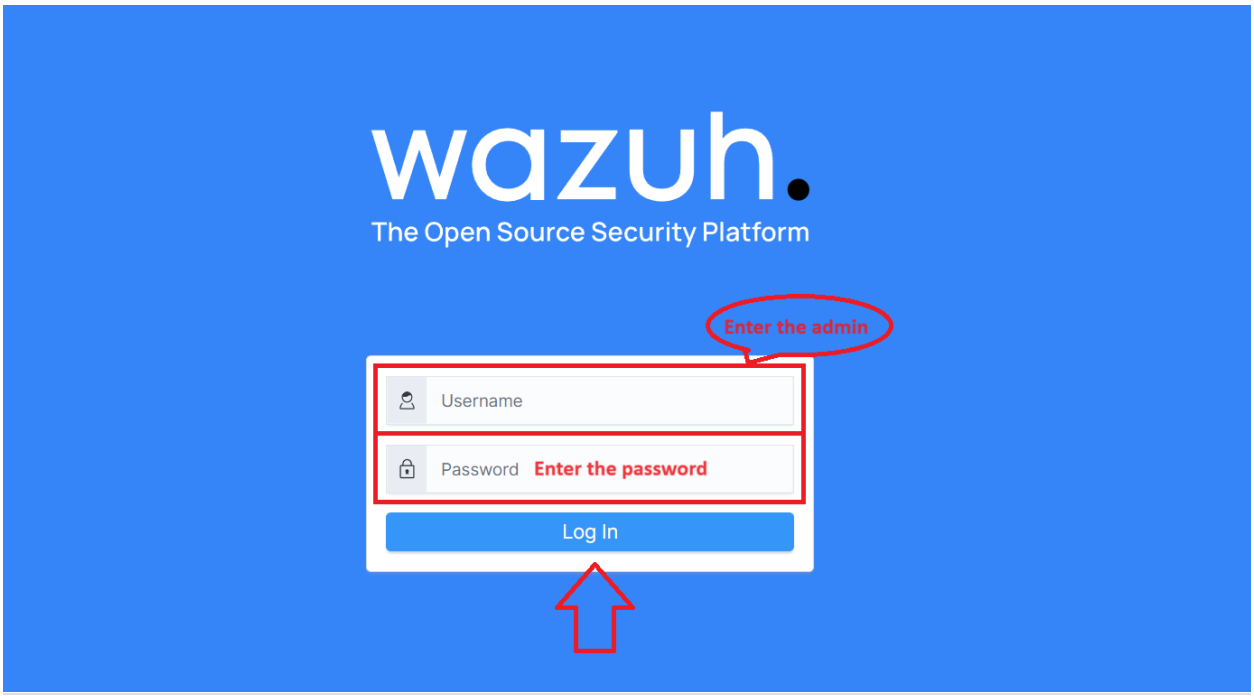
bash
Copy code
systemctl restart wazuh-api

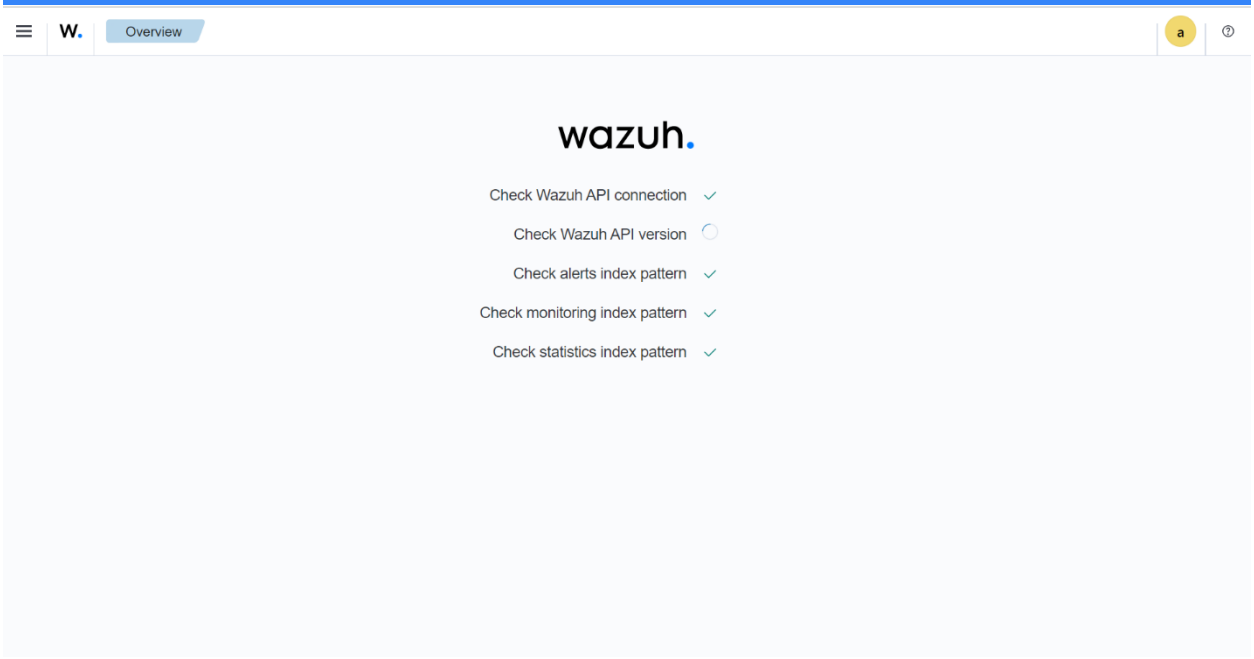
STEP 13: If Want To Change The Passwaord Of Wazuh Login Tou Can Do By Apply Ing The Following Comand

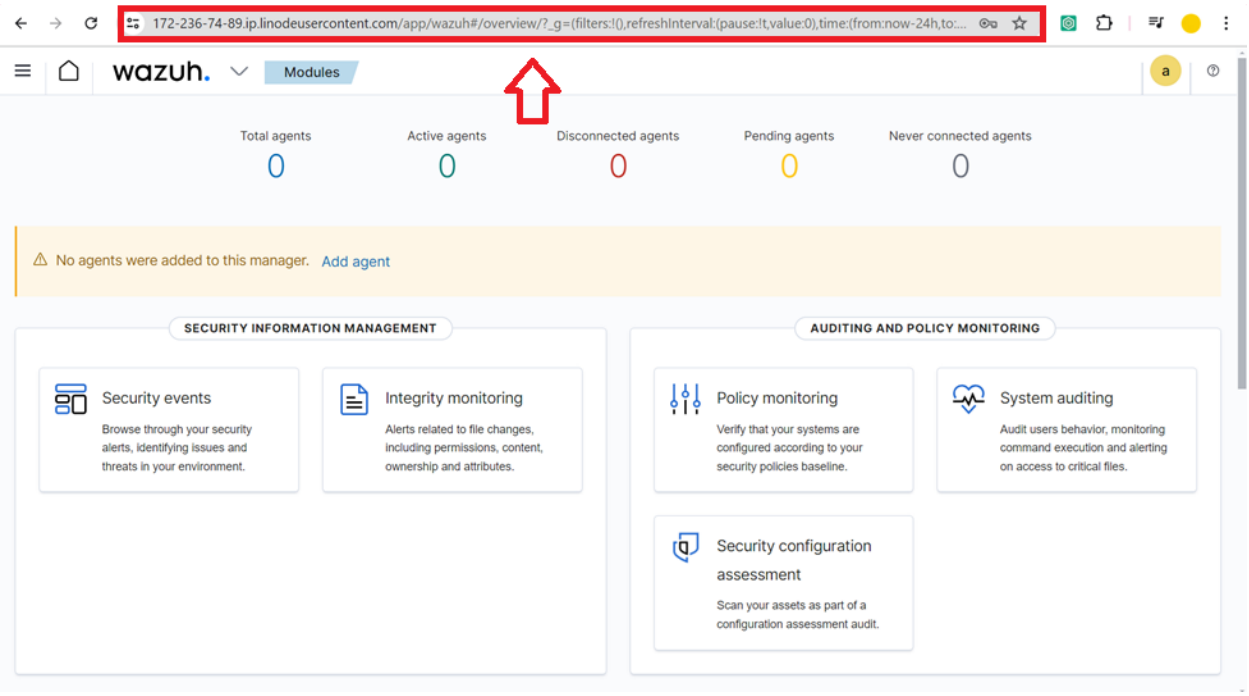


STEP 14: Copy The Ip As I Shown Below And Past It In New Tab Url Box




wazuh.
The Open Source Security Platform
Enter the admin
Username
Password
Enter the password
Log In

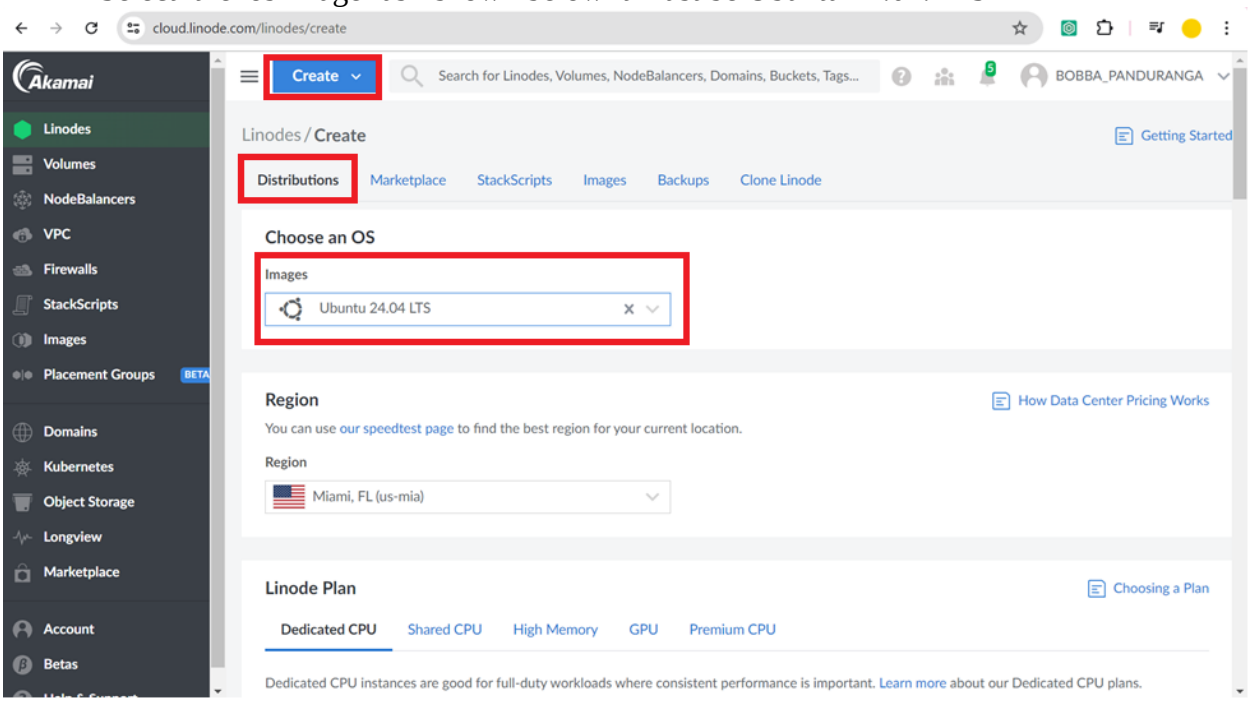

W.
Overview
a
wazuh.
Check Wazuh API connection
Check Wazuh API version
Check alerts index pattern
Check monitoring index pattern
Check statistics index pattern



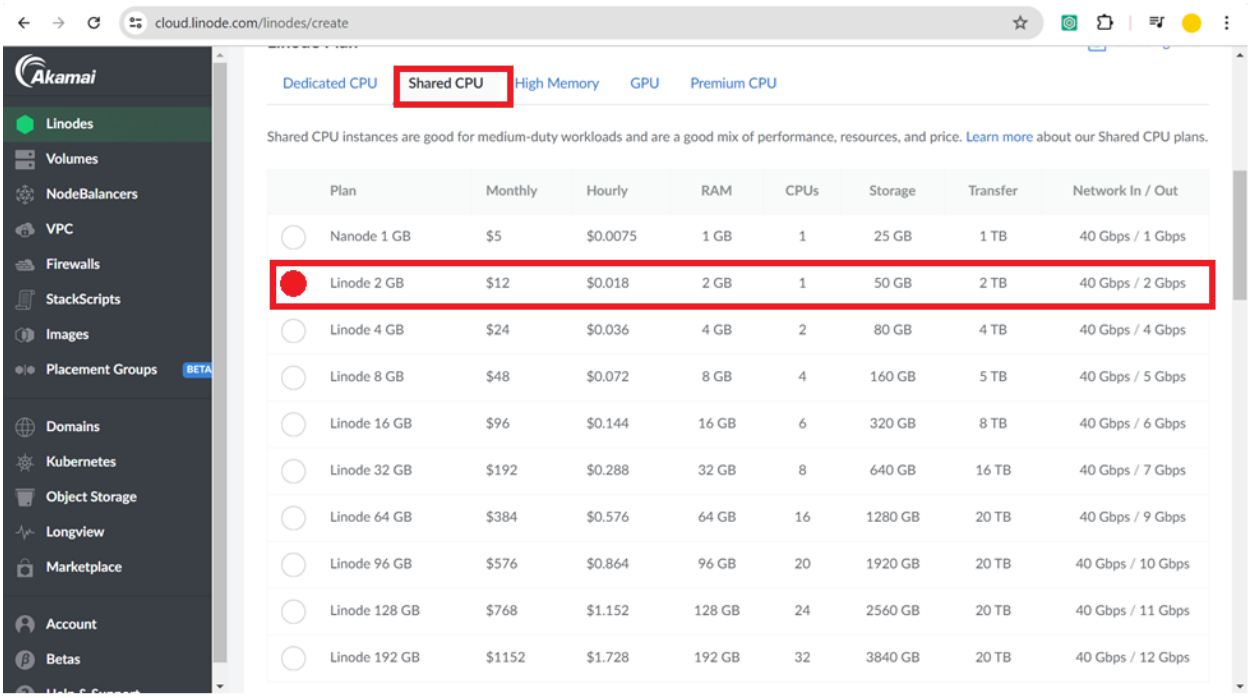
Creating the dock server:

STEP 1: Create the new server by following the below commands

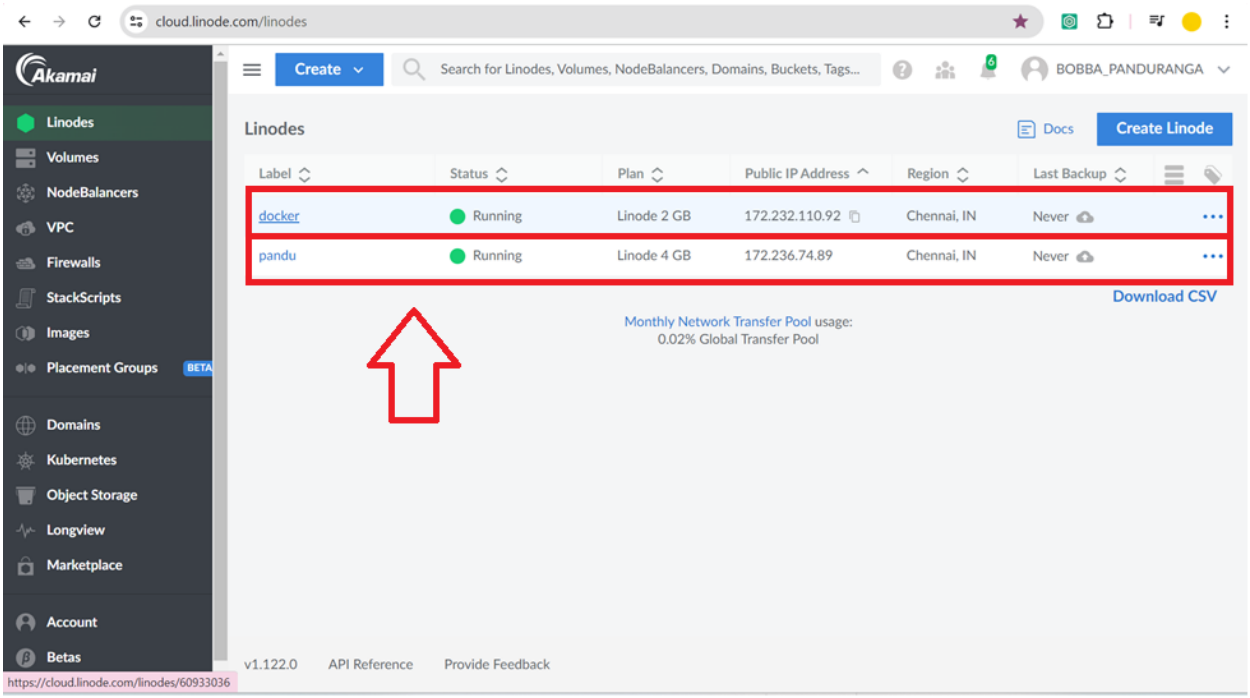
- Click on the create
- Select the distributions option
- Select the os imager as I shown below it must be Ubuntu 24.04.LTS



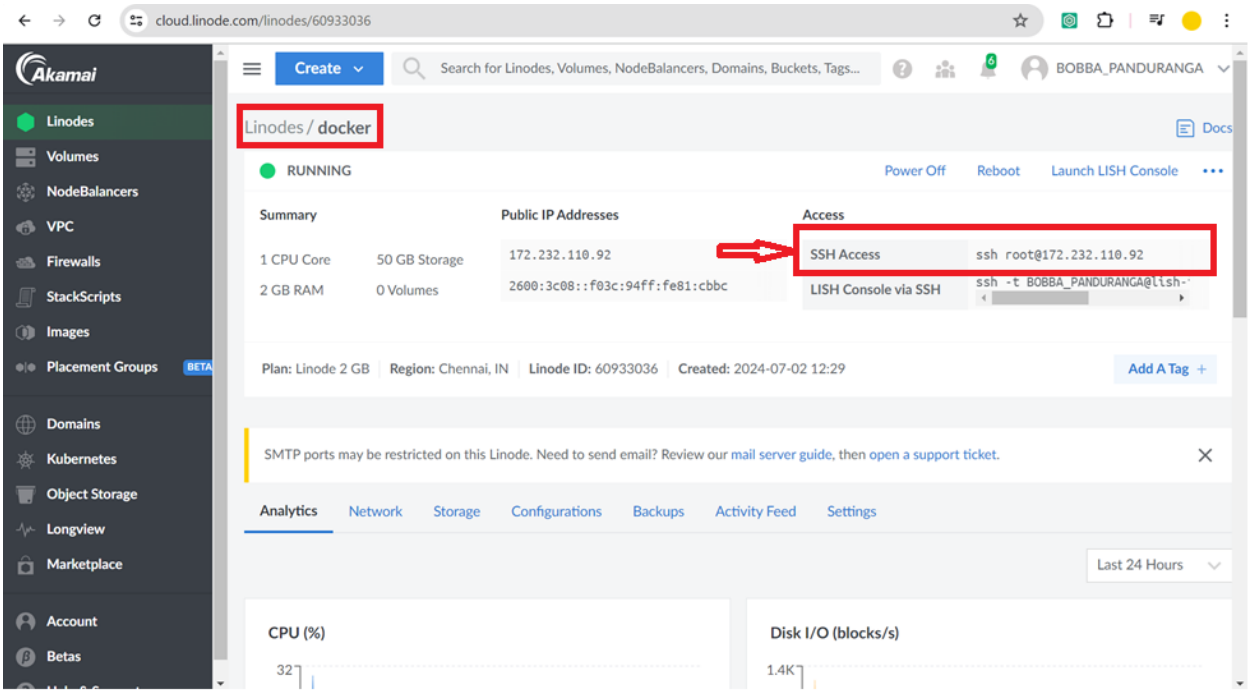
STEP 2: :Select The Shared CPU And Select The 2gb Version And Click On Create Linode



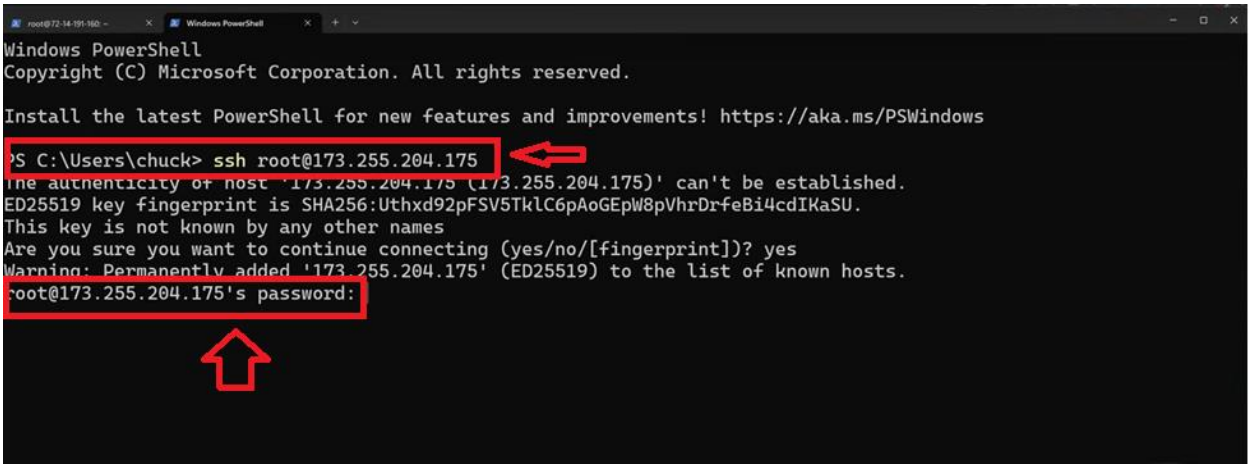
STEP 3: Now select the docker and open it



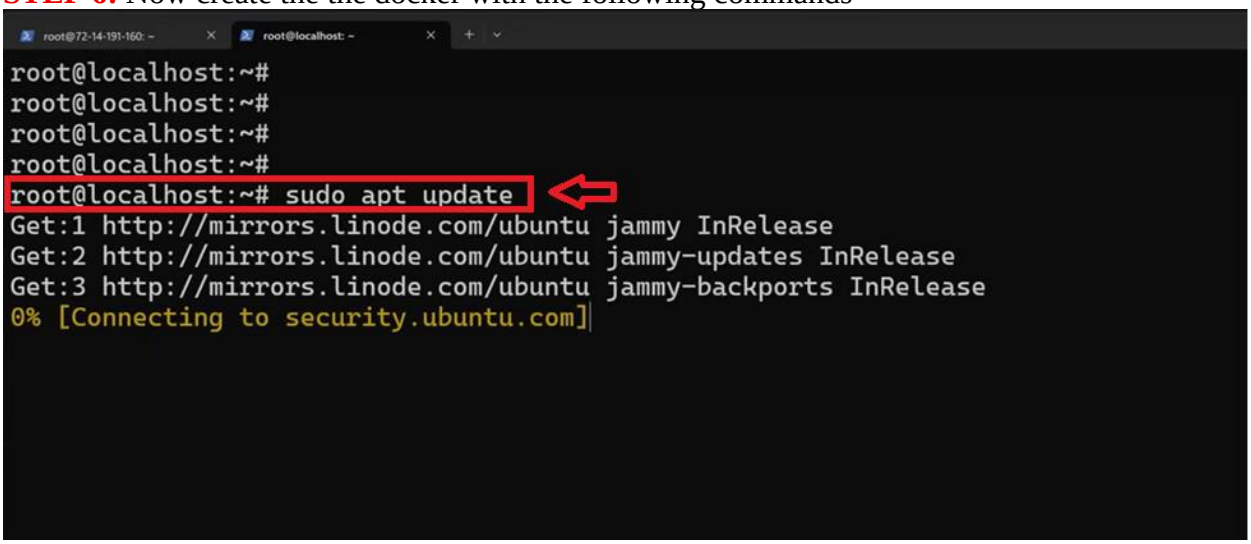
STEP 4: Now Copy The Ssh Access



STEP 5: Past It In Cmd (Regular Mode) Enter The Password As Well



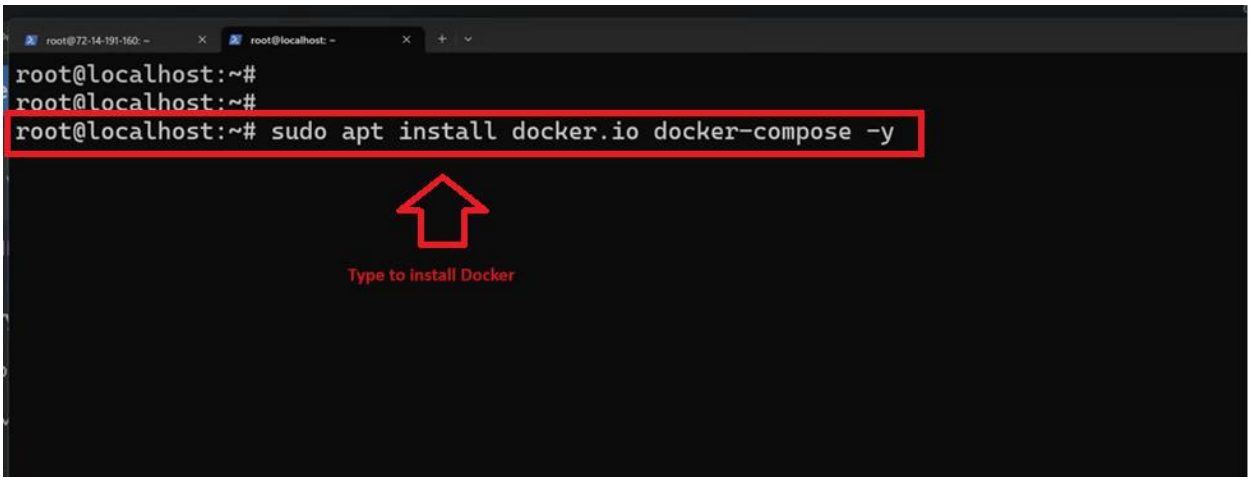
STEP 6: Now create the the docker with the following commands



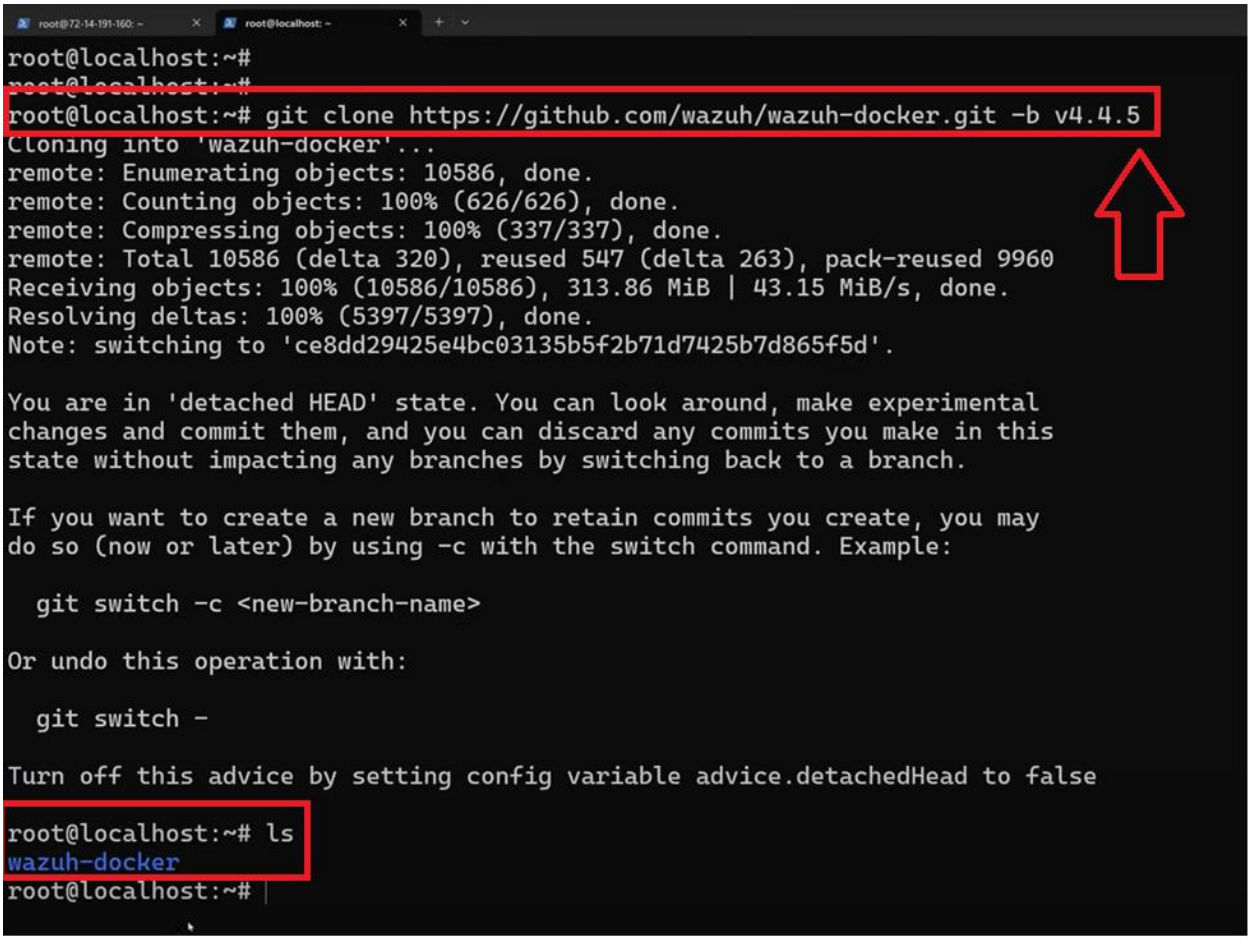
- Must visit this link to learn more about :

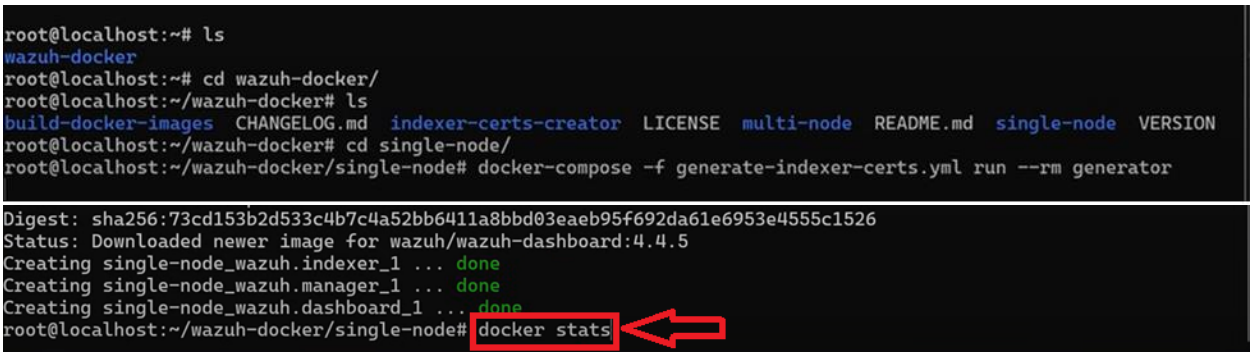
- https://documentation.wazuh.com/current/deployment-options/docker/wazuh-container.html

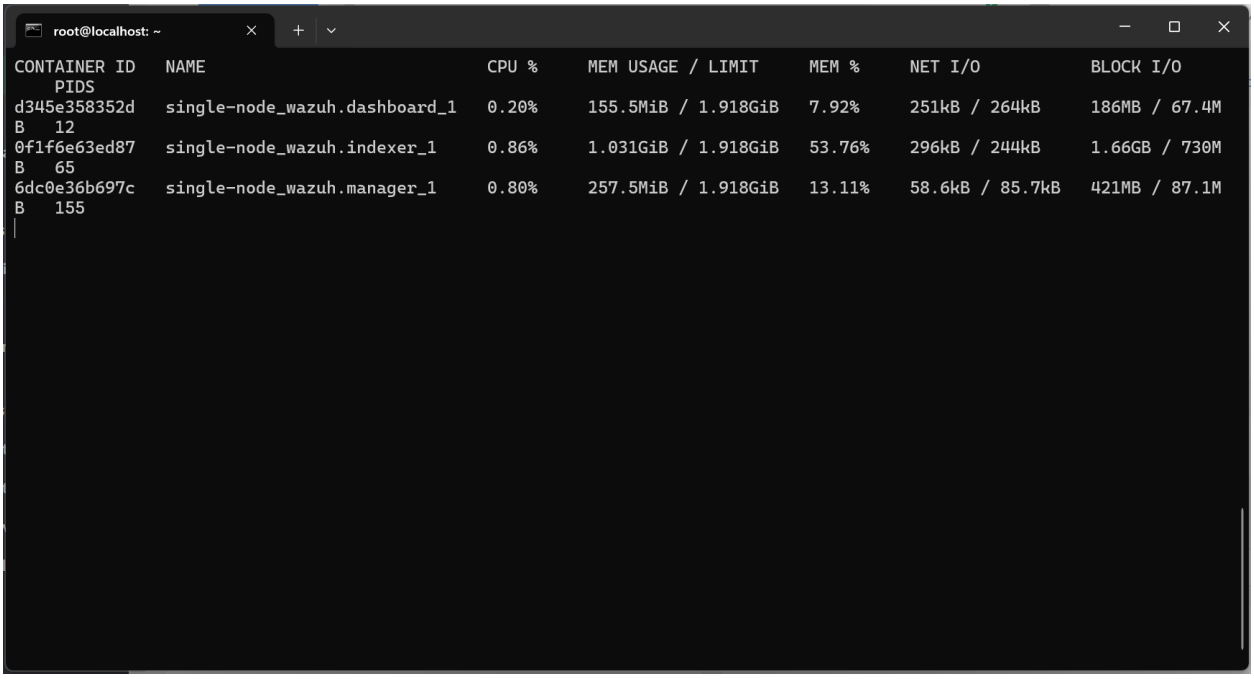
STEP 7: Type The Following Command To Install Docker In Your Server



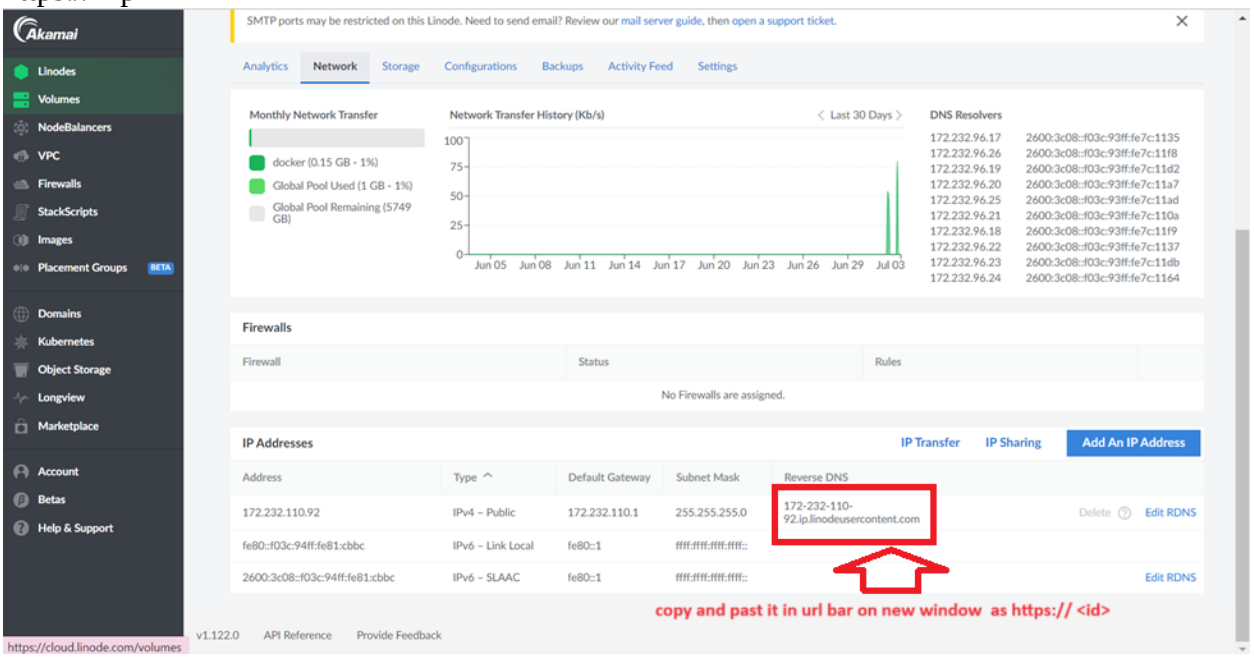
STEP 8 : Clone The Docker By Typing The Clone Command As Shown Below

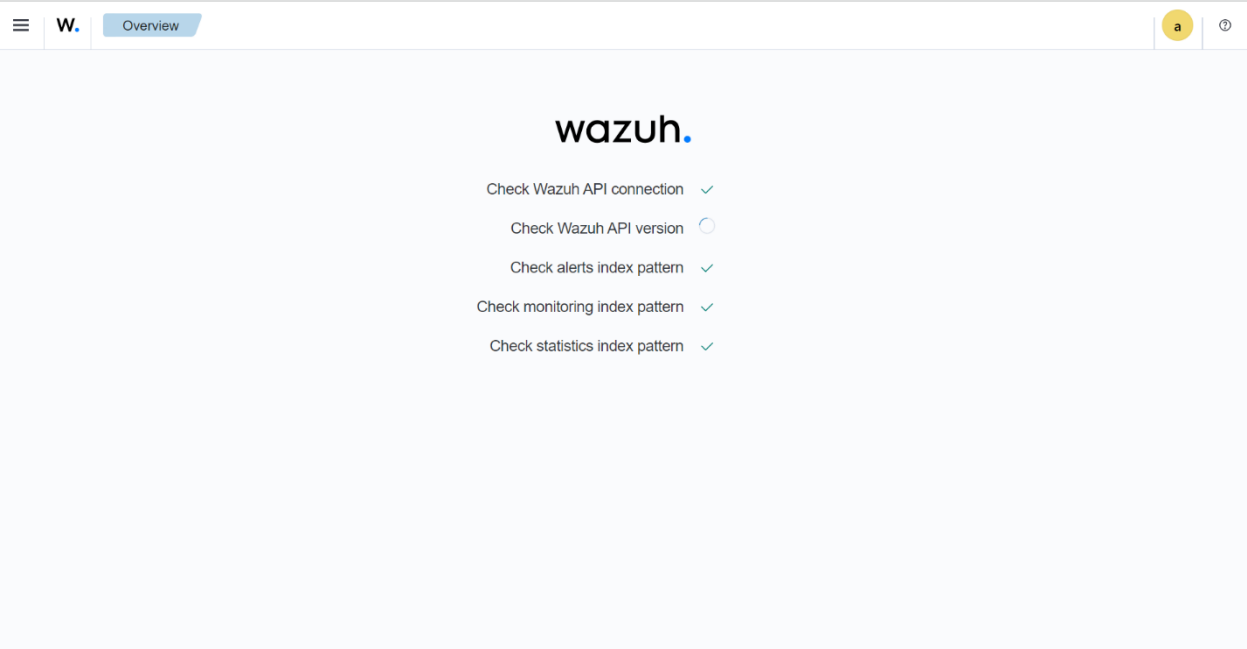




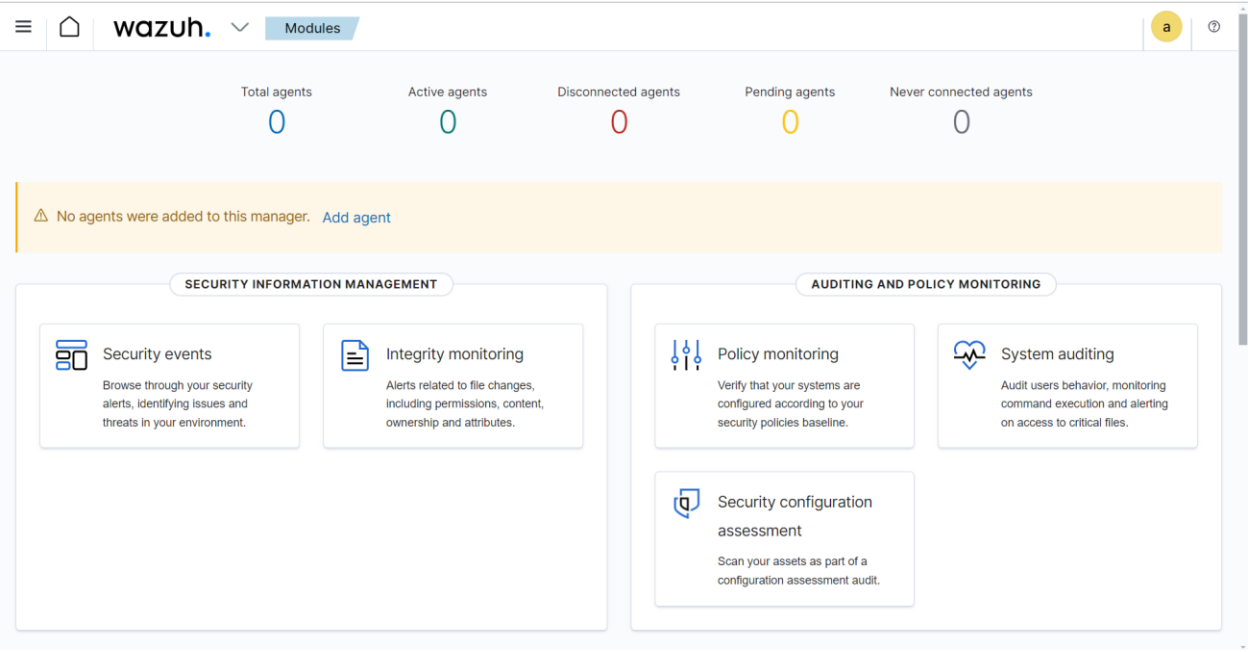


STEP 9: Now Done The Docker Is Installed In Server Now Copy The <Ip> In New Tab As https://<Ip>





Now We Have To Add The Pc To Our Sever

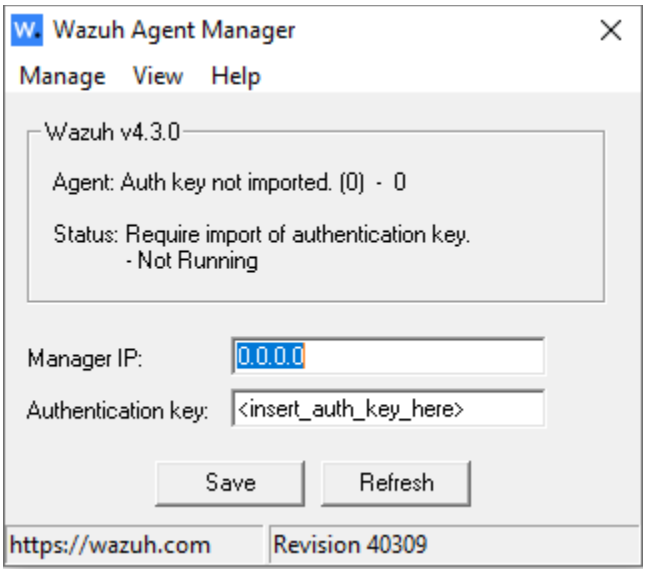


STEP 1: Download The Wazuh Agent From Below Link

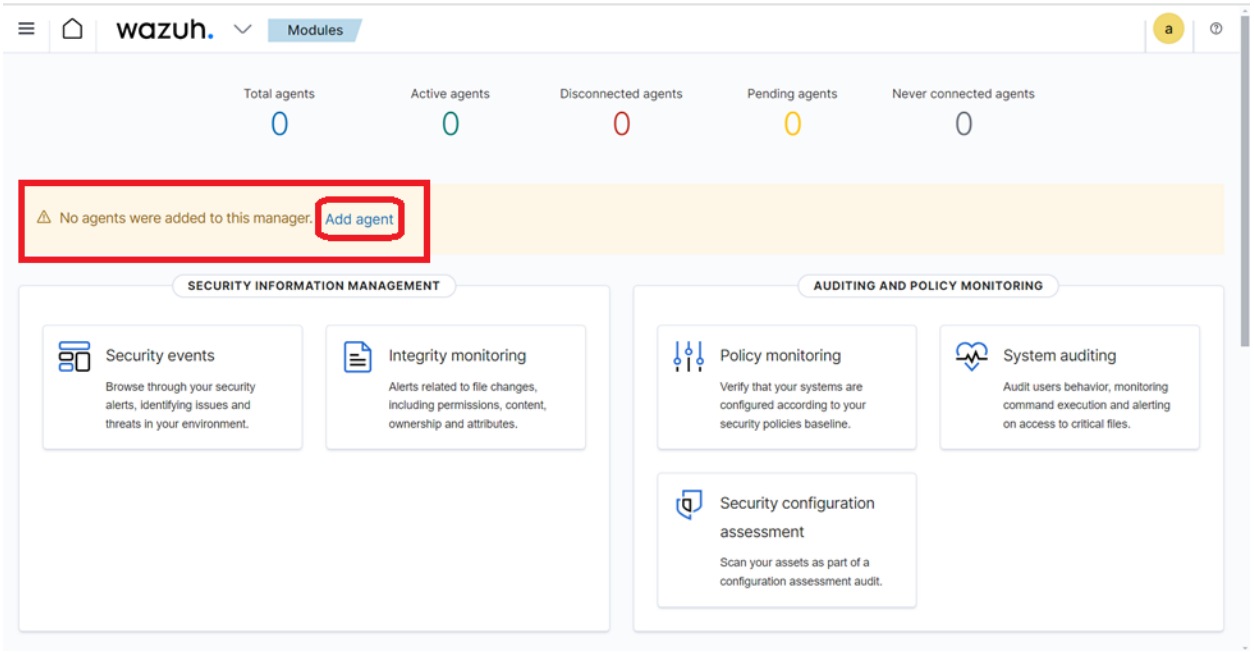
https://documentation.wazuh.com/4.7/installation-guide/wazuh-agent/index.html

STEP 2: Install The Wazuh-Agent And Type The IP That You Have Pasted In URL Bar

STEP 3: After Click On Save And Go To Manage Option And Click On Start Option



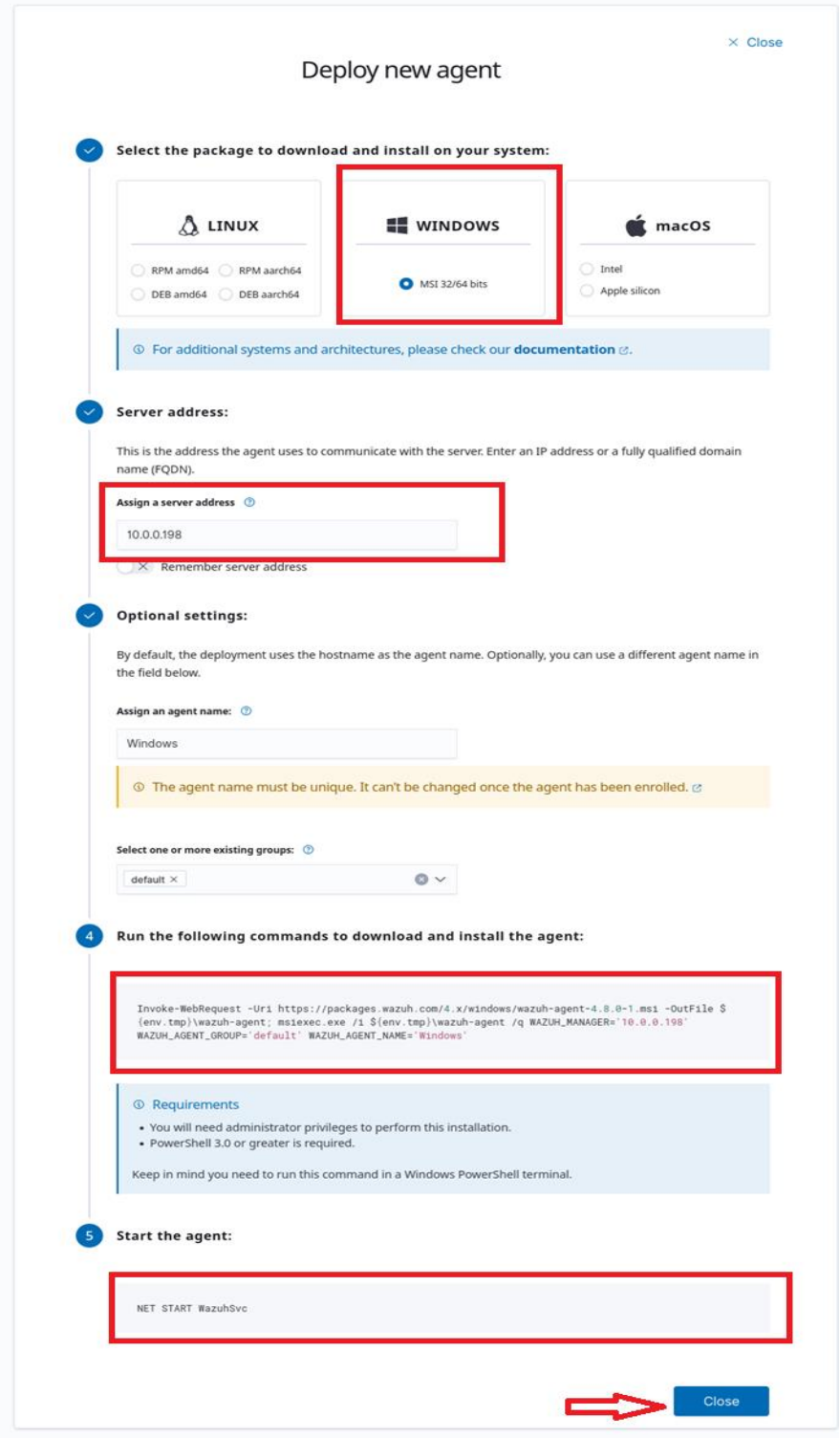
STEP 4: Now Click On The Agent Option



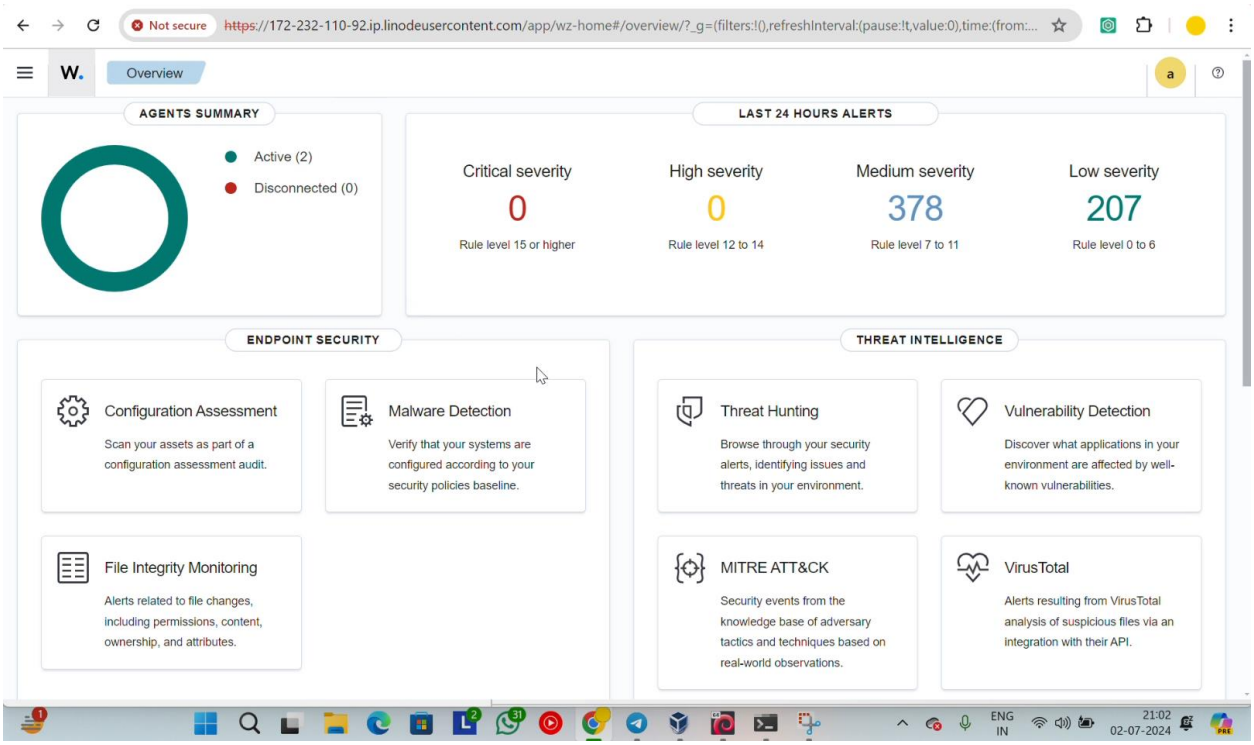
STEP 5: Enter The IP As You Enter In URL Bar

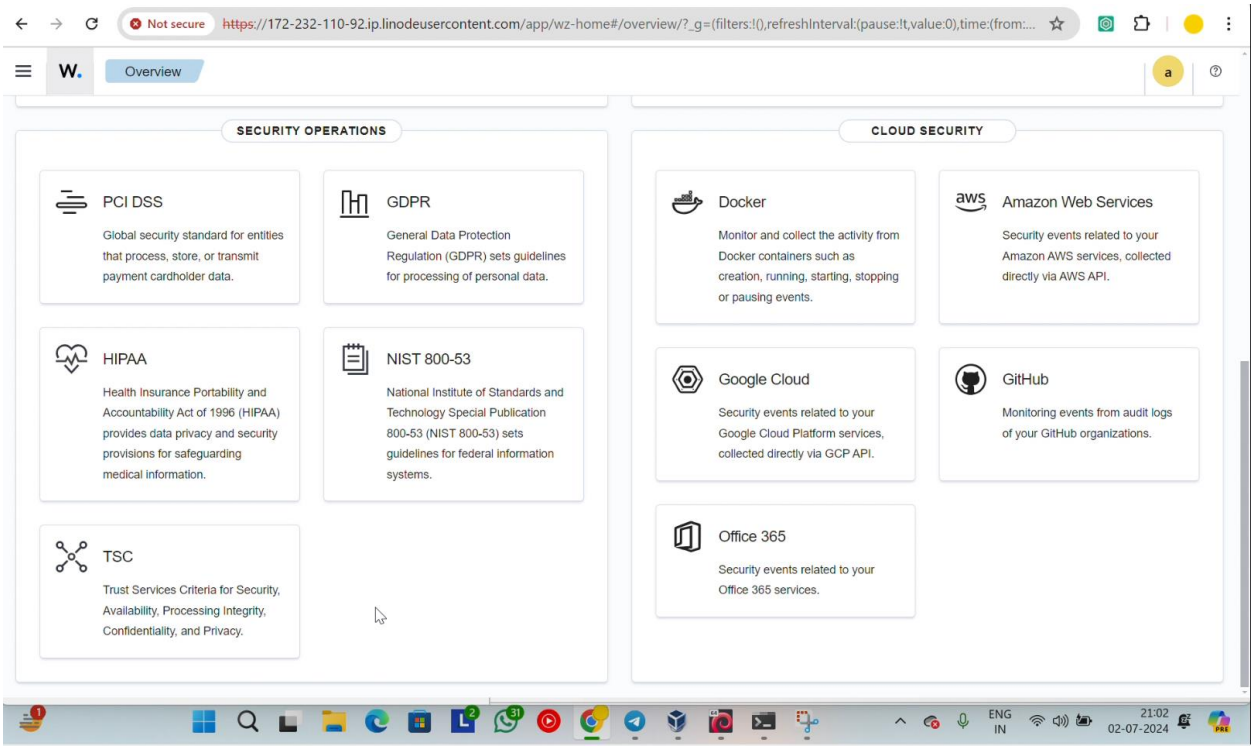
STEP 6: Copy The Command As You Got And Go To Windows Powershell (Admin) And Copy Them

STEP 7: Click On The Close Button As Shown Below

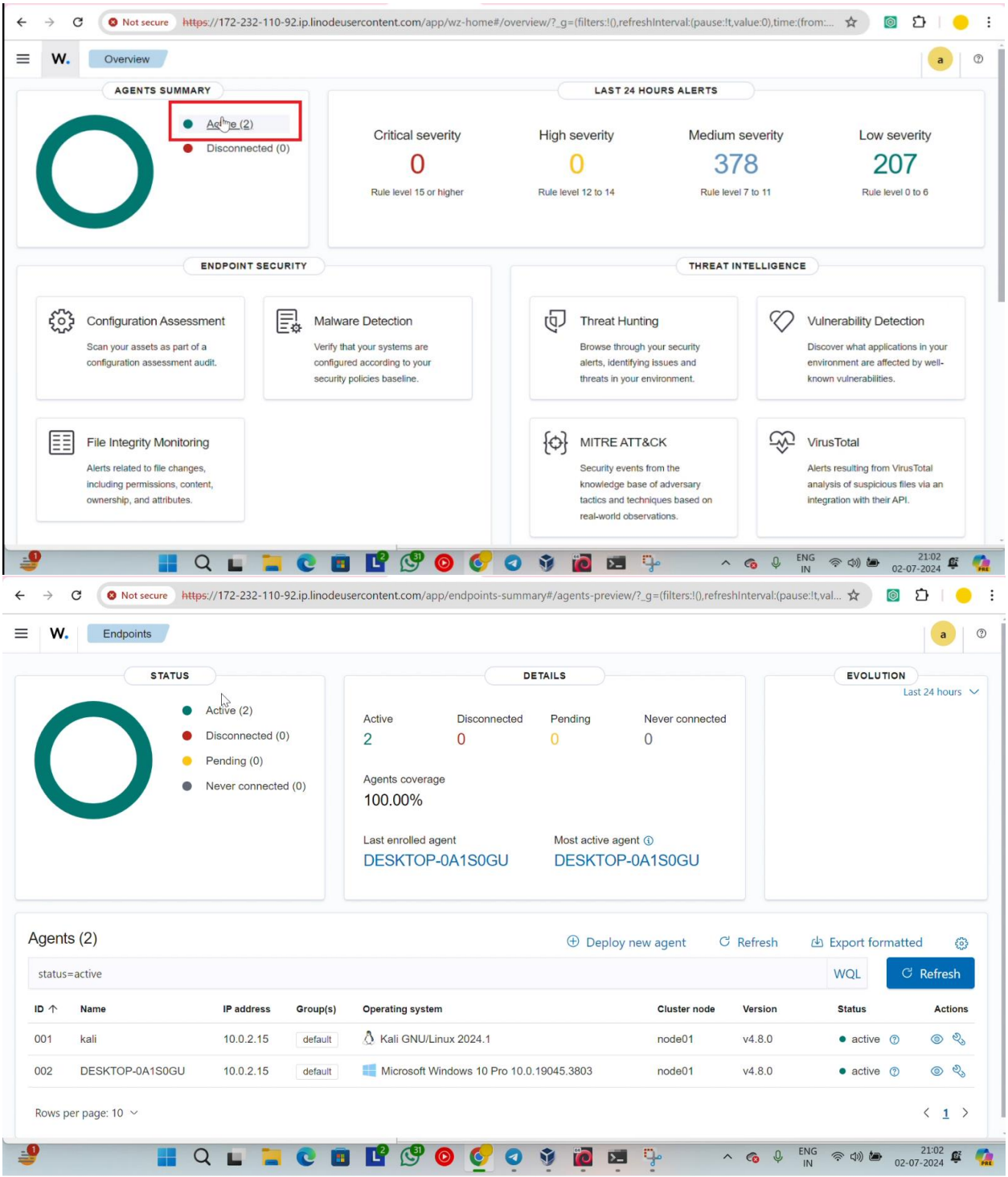


STEP 8: Now You Will See Your Dashboard





STEP 9: click on the Active



2. Malware detection Permalink to this headline

Malware, short for malicious software, refers to any software specifically designed to harm or exploit computer systems, networks, or users. It is created with the intention of gaining unauthorized access, causing damage, stealing sensitive information, or performing other malicious activities on a target system. There are various types of malware, each with specific functions and infection methods. Some common types of malware include viruses, worms, ransomware, botnets, spyware, trojans, and rootkits.

Malware detection is crucial for safeguarding computer systems and networks from cyber threats. It helps identify and mitigate malicious software that can cause a data breach, system compromise, and financial loss.

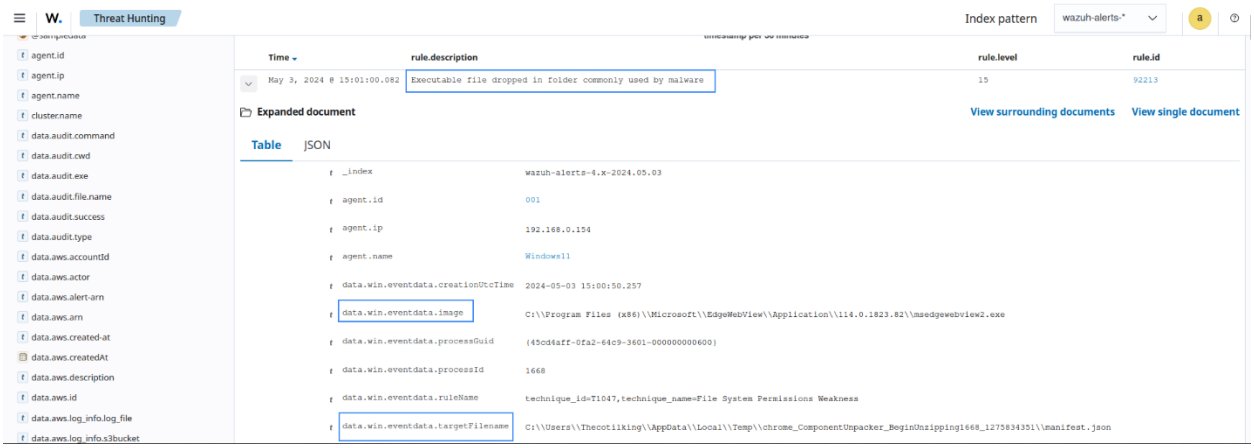
Wazuh for malware detectionPermalink to this headline

Traditional methods, which rely solely on signature-based detections, have limitations and fail to capture new threats. Signature-based approaches struggle with detecting zero-day attacks, polymorphic malware, and other evasion techniques employed by threat actors. As a result, organizations are at risk of undetected breaches and data exfiltration. Wazuh empowers organizations to detect and respond to sophisticated and evasive threats effectively. Wazuh encompasses different modules that identify malware properties, activities, network connections, and more.

Detecting malicious activities with threat detection rulesPermalink to this headline

Wazuh has threat detection rules that enable behavior-based malware detection. Instead of relying solely on predefined signatures, Wazuh focuses on monitoring and analyzing abnormal behavior exhibited by malware. This allows Wazuh to detect known and previously unknown threats. This way, Wazuh provides a proactive and adaptable defense

against cyber threats. Wazuh has out-of-the-box rulesets that are specifically designed to trigger alerts for recognized malware patterns, providing a quick response to potential security incidents. For example, the image below shows an alert with rule ID 92213 triggered when an executable is dropped in a folder commonly used by malware. This alert prompts security teams to begin the investigation and remediation process.



Wazuh allows users to create custom rules for more flexibility in detection, empowering them to focus on relevant activities, and optimizing malware detection. Wazuh decodes and organizes logs from monitored endpoints into fields, which can then be utilized to create custom rules for alerting when malicious activity is detected.

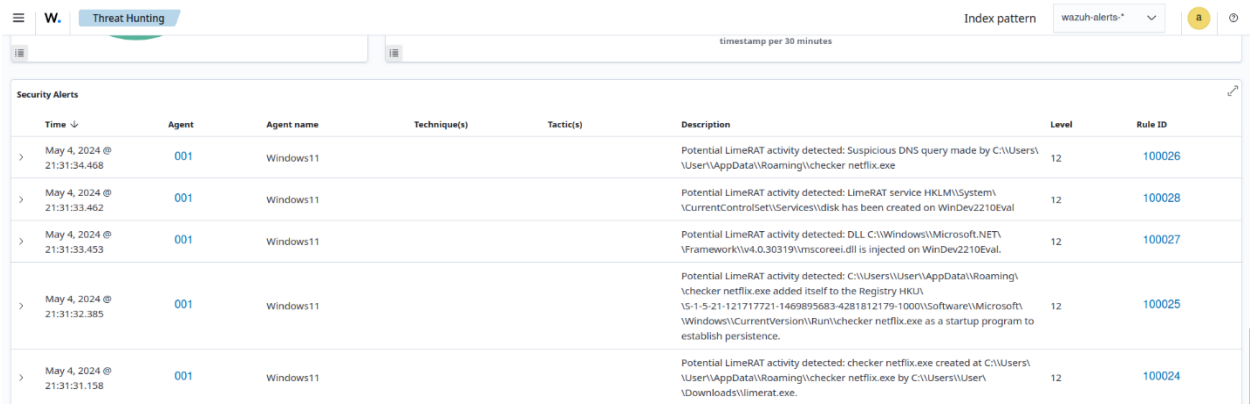
Wazuh rules use multiple fields that denote indicators of compromise (IOCs) to reduce false positives and detect known malware based on specific behaviors. These rules can connect related malware activities, such as intrusion, privilege escalation, lateral movement, obfuscation, and exfiltration for comprehensive detection.

Below is an example of some Wazuh custom rules created to alert on malicious activities of the LimeRAT malware:

```
<group name="lime_rat,sysmon,">

  <!-- Rogue create netflix.exe creation -->
  <rule id="100024" level="12">
    <if_sid>61613</if_sid>
    <field name="win.eventdata.image" type="pcre2">\.exe</field>
    <field name="win.eventdata.targetFilename" type="pcre2">(?i)[c-z]:\\\\Users\\\\.+\\\\AppData\\\\Roaming\\\\checker netflix\.exe</field>
    <description>Potential LimeRAT activity detected: checker netflix.exe created at $(win.eventdata.targetFilename) by $(win.eventdata.image).</description>
    <mitre>
      <id>T1036</id>
    </mitre>
  </rule>

  <!-- Registry key creation for persistence -->
  <rule id="100025" level="12">
    <if_group>sysmon</if_group>
    <field name="win.eventdata.details" type="pcre2">(?i)[c-z]:\\\\Users\\\\.+\\\\AppData\\\\Roaming\\\\checker netflix\.exe</field>
    <field name="win.eventdata.targetObject" type="pcre2" >HKU\\\\.+\\\\Software\\\\Microsoft\\\\Windows\\\\CurrentVersion\\\\Run\\\\checker netflix\.exe</field>
    <field name="win.eventdata.eventType" type="pcre2" >^SetValue$</field>
    <description>Potential LimeRAT activity detected:  $(win.eventdata.details) added itself to the Registry as a startup program $(win.eventdata.targetObject) to establish persistence.</description>
    <mitre>
      <id>T1547.001</id>
    </mitre>
  </rule>

  <!-- Network activity detection -->
  <rule id="100026" level="12">
    <if_sid>61605</if_sid>
```

```
  <field name="win.eventdata.image" type="pcre2">(?i)[c-z]:\\\\Users\\\\.+\\\\AppData\\\\Roaming\\\\checker netflix\.exe</field>
  <description>Potential LimeRAT activity detected: Suspicious DNS query made by $(win.eventdata.image).</description>
  <mitre>
    <id>T1572</id>
  </mitre>
 </rule>


 <!-- LimeRAT service creation -->
 <rule id="100028" level="12">
  <if_sid>61614</if_sid>
  <field name="win.eventdata.targetObject" type="pcre2" >HKLM\\\\System\\\\CurrentControlSet\\\\Services\\\\disk</field>
  <field name="win.eventdata.eventType" type="pcre2" >^CreateKey$</field>
  <description>Potential LimeRAT activity detected: LimeRAT service $(win.eventdata.targetObject) has been created on $(win.system.computer).</description>
    <mitre>
  <id>T1543.003</id>
    </mitre>
 </rule>

</group>
```

These rules create alerts that are visible in the **Threat Hunting** module on the Wazuh dashboard.



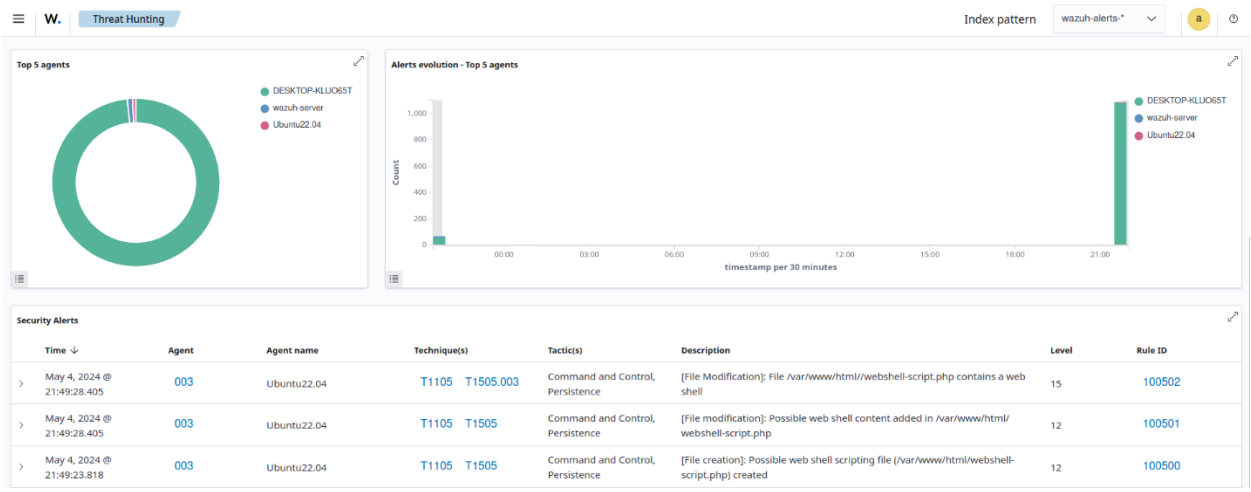
Refer to the blog post on LimeRat detection and response with Wazuh for the full configuration.

Wazuh identifies behavior indicative of malware, it generates real-time alerts and notifications, enabling security teams to respond swiftly and mitigate potential risks before they escalate.

Leveraging file integrity monitoring for detecting malware activityPermalink to this headline

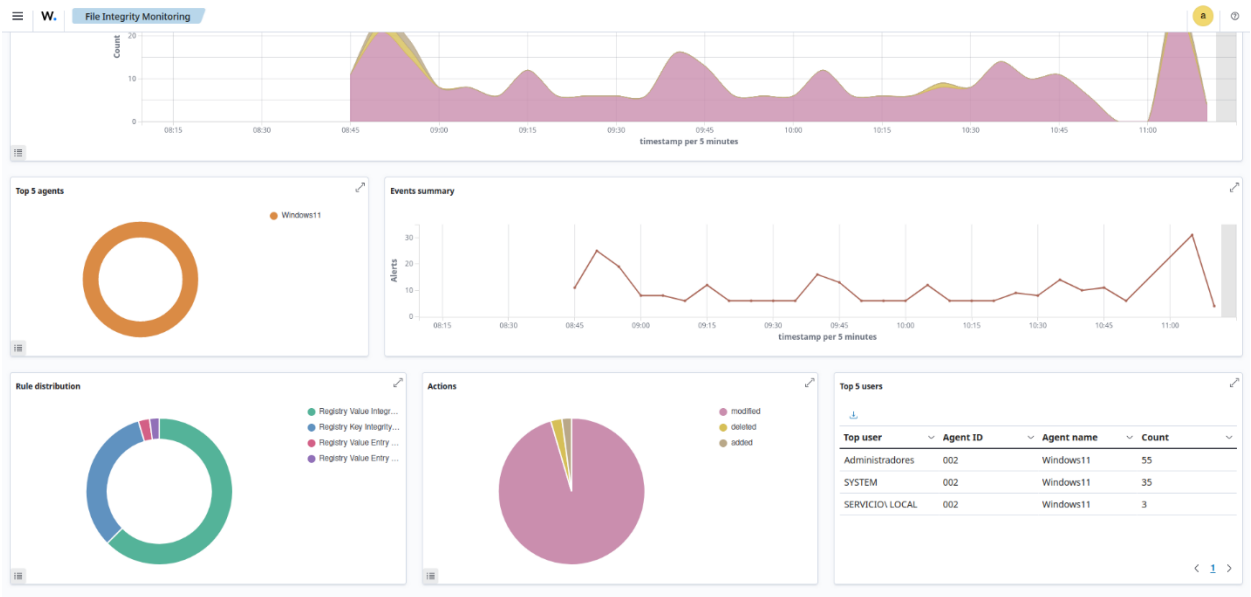
File Integrity Monitoring (FIM) is a valuable component in malware detection. Wazuh provides FIM capabilities to monitor and detect changes to files and directories on monitored endpoints. These changes include creation, modification, or deletion. While FIM provides essential insights, combining it with other capabilities and integrations further enhances its effectiveness for malware detection. Wazuh allows security teams to create custom rules based on FIM events, enabling targeted malware detection. These customizable rules correlate FIM events with specific indicators of compromises such as suspicious file extensions, code snippets, or known malware signatures.

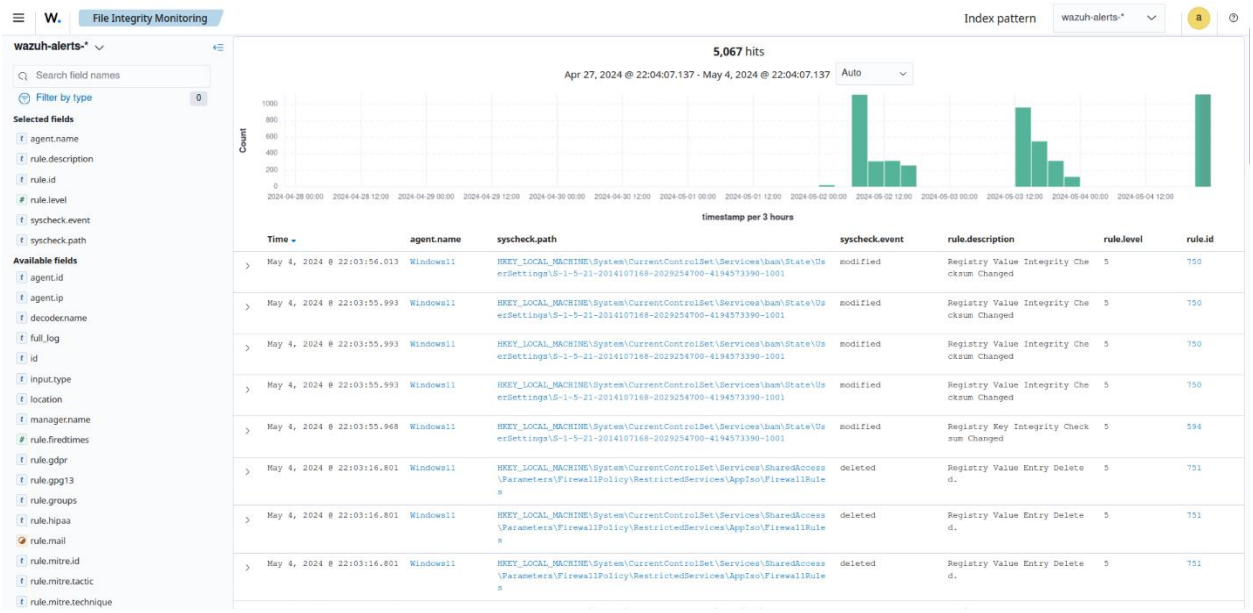
The image below shows an alert when a web shell creates or modifies a file on a web server.



Malware frequently targets Windows Registry to achieve malicious objectives, such as establishing persistence and performing other malicious actions. The Wazuh File Integrity Monitoring (FIM) module includes Windows Registry monitoring that monitors commonly targeted registry paths to detect modifications. When changes occur, the FIM module triggers real-time alerts, empowering security teams to swiftly identify and respond to suspicious registry key manipulation.

The images below display the Wazuh FIM module dashboard and events of Windows Registry modifications.





Unveiling stealthy threats with rootkit detectionPermalink to this headline

Rootkits are malicious software designed to conceal the presence of malware on an endpoint by manipulating operating system functions such as altering system calls or modifying kernel data structures. Wazuh has a Rootcheck module that periodically scans the monitored endpoint to detect rootkits both at the kernel and the user space level. The rootcheck identifies and alerts potential rootkit activity. By analyzing system behavior and comparing it to known rootkit patterns, Wazuh promptly detects rootkit-related patterns and raises alerts for further investigation.

Below, we show an example of an alert generated by the Wazuh Rootcheck module when it detects an anomaly in the filesystem:

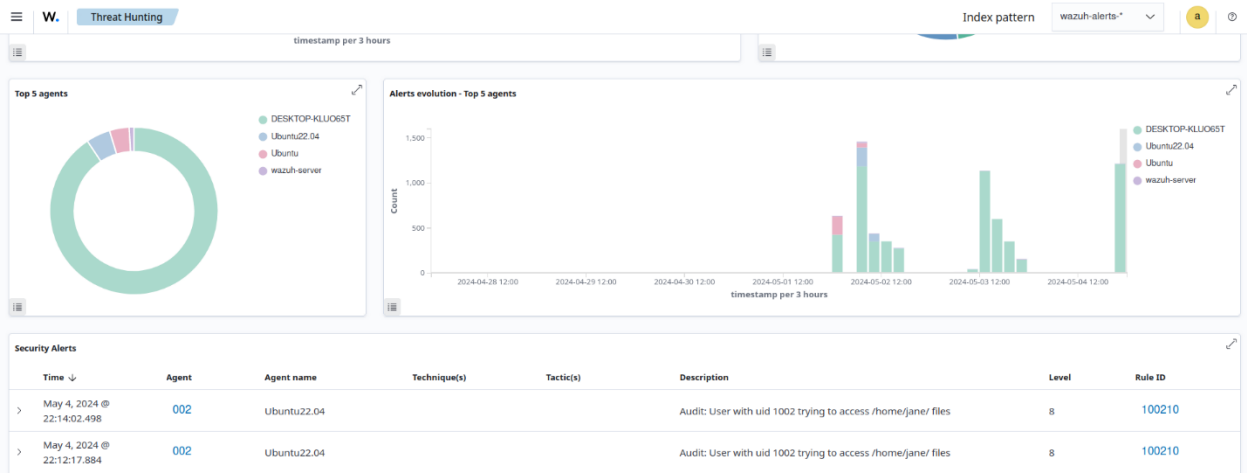
```
** Alert 1668497750.1838326: - ossec,rootcheck,pci_dss_10.6.1,gdpr_IV_35.7.d,
2022 Nov 15 09:35:50 (Ubuntu) any->rootcheck
Rule: 510 (level 7) -> 'Host-based anomaly detection event (rootcheck).'
Rootkit 't0rn' detected by the presence of file '/usr/bin/.t0rn'.
title: Rootkit 't0rn' detected by the presence of file '/usr/bin/.t0rn'.
```

While Wazuh continues to enhance its rootkit behavior detection capabilities, the Command monitoring module can also be configured to monitor command-line activities across endpoints, enabling the detection of malicious commands and malware activities. This module provides organizations with a comprehensive approach to uncovering hidden threats and safeguarding their systems effectively.

Monitoring system calls for malware and anomaly detection Permalink to this headline

Wazuh monitors system calls on Linux endpoints to bolster malware detection and aid in anomaly detection. Wazuh utilizes the Linux Audit system to monitor system calls.

System call monitoring in combination with Wazuh File Integrity Monitoring (FIM) and threat intelligence integration enhances malware detection. It captures security-relevant events like file access, command execution, and privilege escalation, providing real-time insights into potential security incidents. This comprehensive approach strengthens organizations' cybersecurity resilience. In the image below, you can visualize the alerts for privilege abuse on the Wazuh dashboard for Ubuntu Linux 22.04.

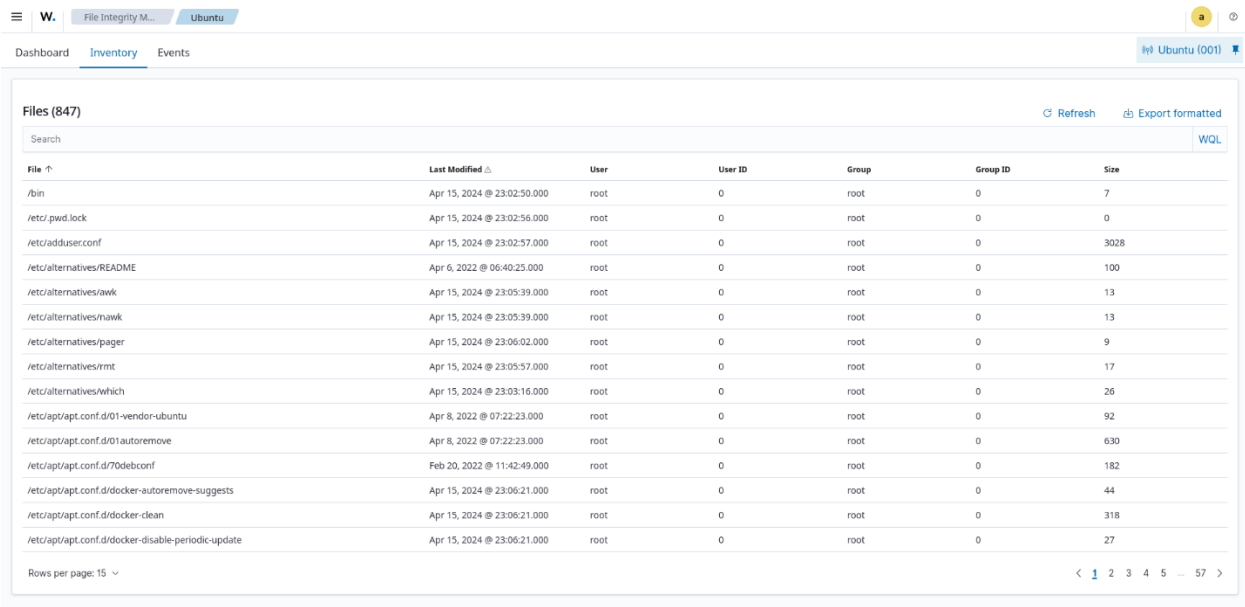


Wazuh empowers security teams to leverage the audit rules provided by Auditd. Creating custom rules based on system call events enhances malware detection efforts and strengthens overall cybersecurity resilience.

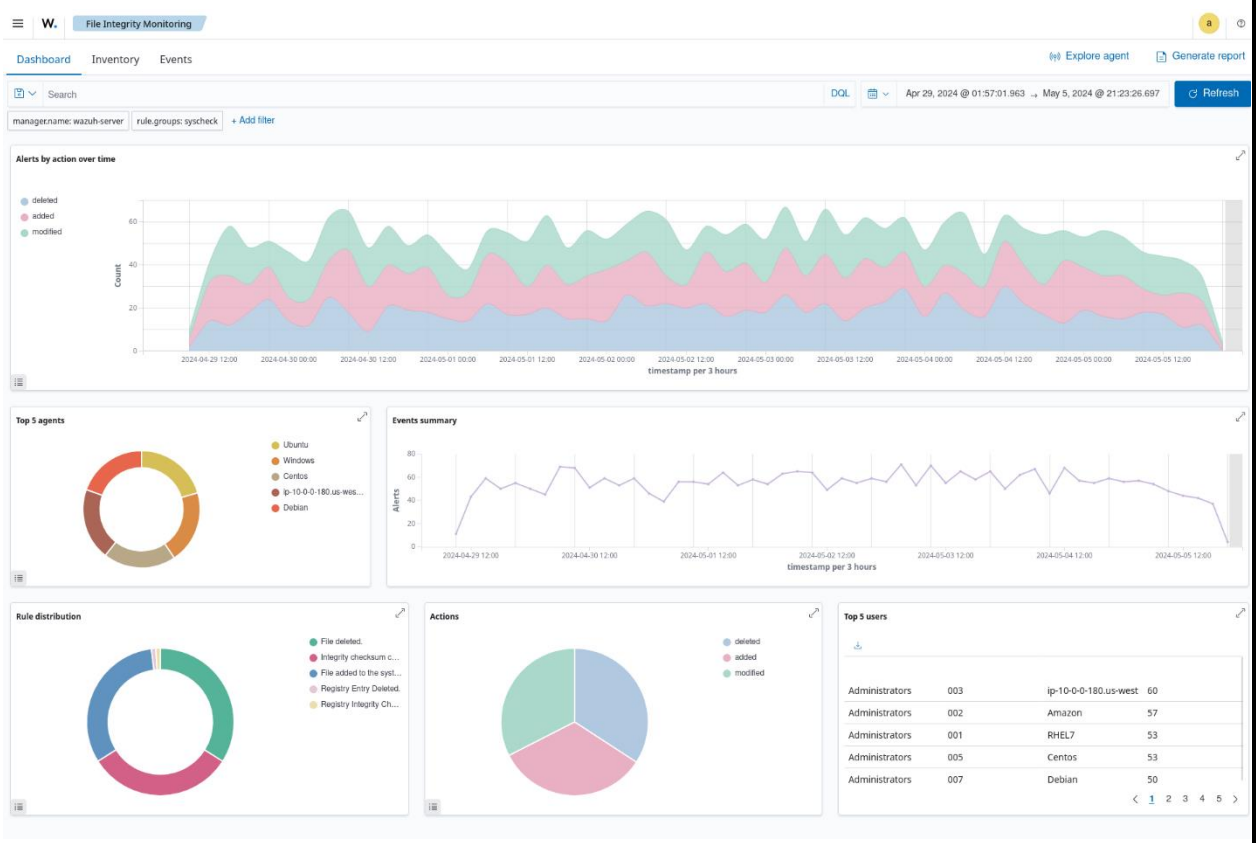
File Integrity Monitoring

You can find a dedicated **File Integrity Monitoring** module in the Wazuh dashboard where all file integrity events triggered from monitored endpoints are reported. This increases visibility as it provides valuable information on the status of monitored directories and their potential impact on the security posture. The Wazuh FIM dashboard has three different sections to view FIM analysis results; **Inventory, Dashboard, and Events**.

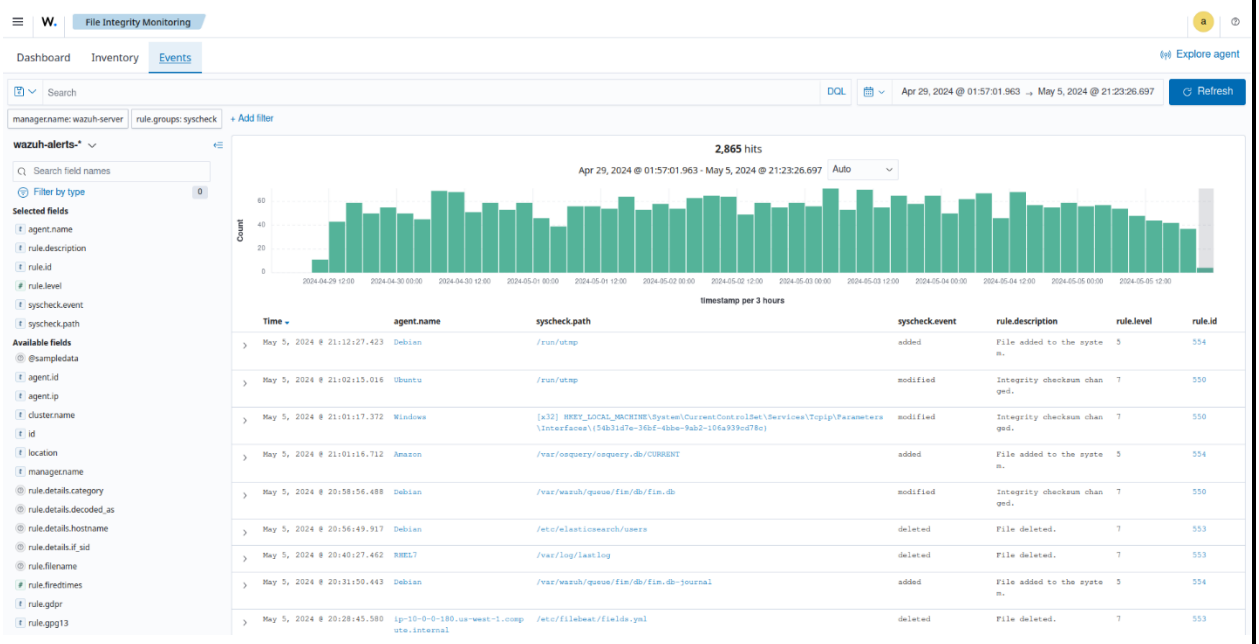
1. The **Inventory** section displays a list of all files that the FIM module has indexed. Each file has entry information including the filename, last modification date, user, user ID, group, and file size.



2. The **Dashboard** section shows an overview of the events triggered by the FIM module for all monitored endpoints. You can also streamline it to show the events for a selected monitored endpoint.



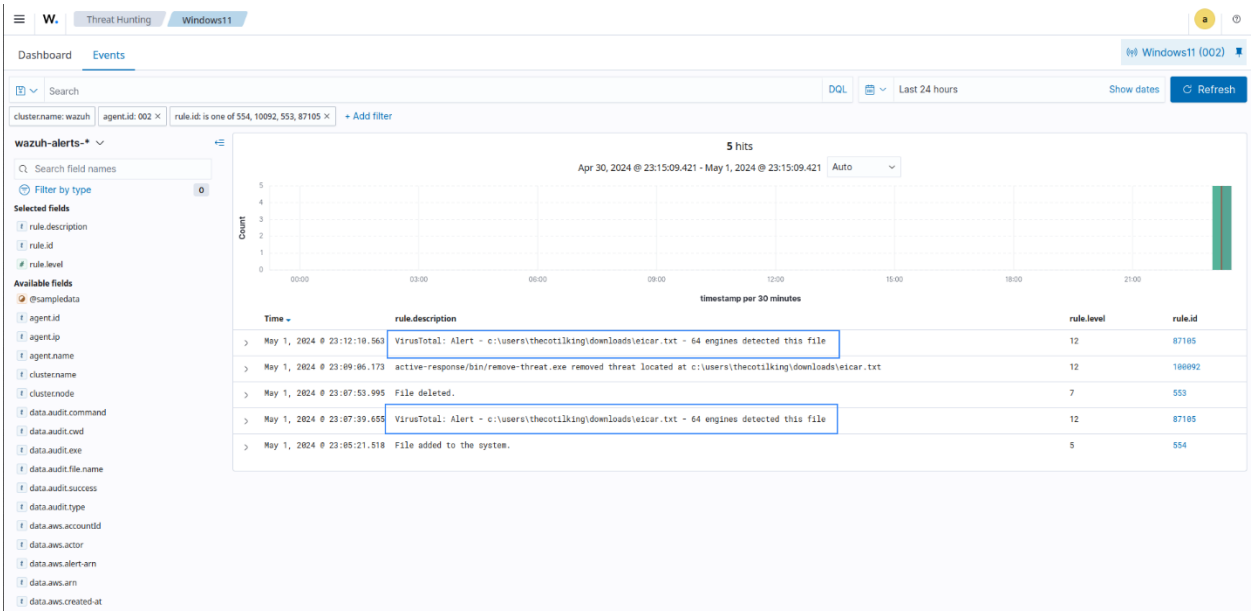
3. The **Events** section shows the alerts triggered by the FIM module. It displays details such as the agent name, the file path of the monitored file, the type of FIM event, a description of the alert, and the rule level of each alert.



Below are common use cases the Wazuh FIM module would assist you in monitoring within your environment.

Enhancing malware detection with threat intelligence integrationPermalink to this headline

Users can boost their malware detection capabilities by integrating with threat intelligence sources. These intelligence feeds enrich the Wazuh knowledge base with additional up-to-date information on known malicious IP addresses, domains, URLs, and other indicators of compromise. Examples of threat intelligence sources Wazuh can integrate with include VirusTotal, MISP, and more.

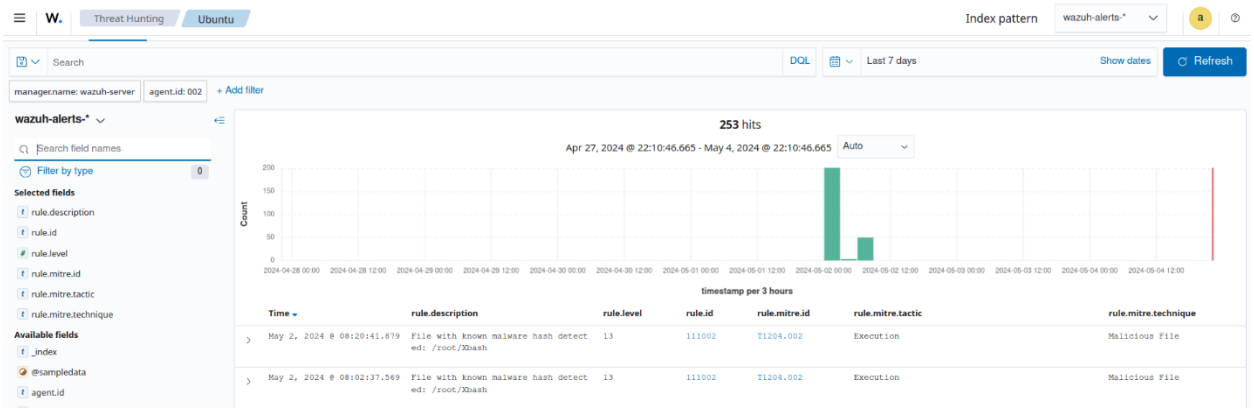


Wazuh proactively identifies malicious files by comparing the identified IOCs with the information stored in the CDB lists (constant databases). These lists can store known malware indicators of compromise (IOCs) including file hashes, IP addresses, and domain names.

You can customize entries in either key:value or key: format for tailored detection, an example of such is seen below. A CBD list containing known MD5 malware hashes of the Mirai and Xbash malware is used for detection:

```
e0ec2cd43f71c80d42cd7b0f17802c73:mirai
55142f1d393c5ba7405239f232a6c059:Xbash
```

Upon detection, these alerts are observed within the **Threat Hunting** module of the Wazuh dashboard, as seen below.



Refer to the Use case: Detecting malware using file hashes in a CDB list for full configurations.

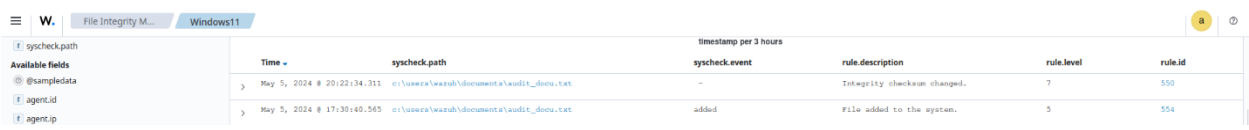
Monitoring file integrityPermalink to this headline

Modifications to configuration files and file attributes are frequent occurrences within endpoints in an IT infrastructure. However, if not validated, there may be unauthorized and inadvertent changes that could affect the behavior of the endpoints or the applications running in them. The Wazuh FIM module runs periodic scans on specific files and directories to detect file changes in real time. It scans the designated files to create a baseline of the current state. It checks for file modifications by comparing checksums and attribute values to the baseline, generating alerts if discrepancies are found.

The Wazuh FIM module supports various configuration options that enable effective monitoring of assets:

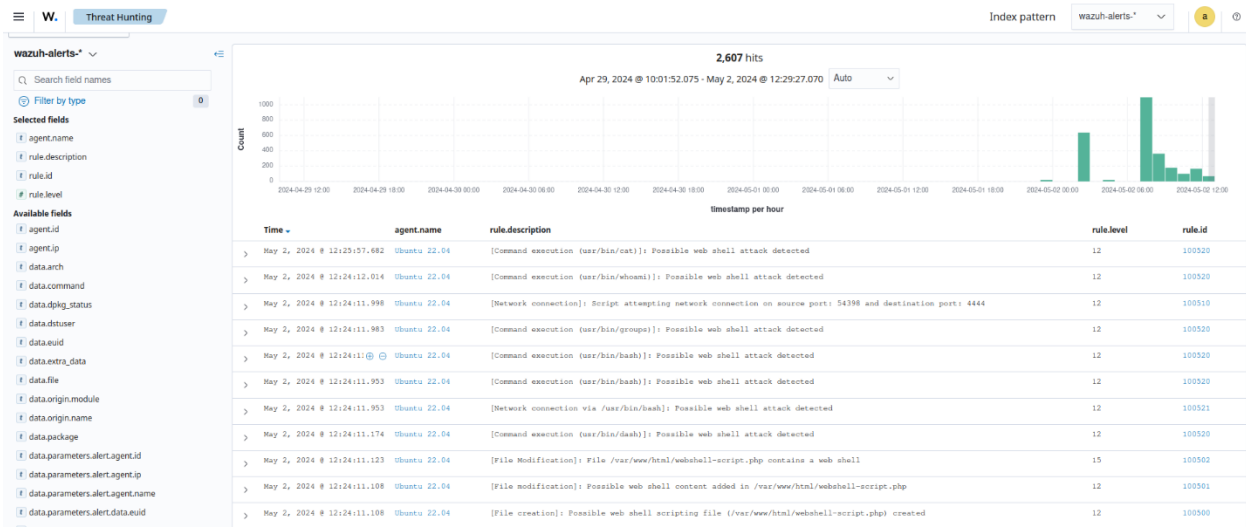
- **Real-time monitoring:** The FIM module provides a realtime attribute that enables continuous monitoring of specified directories. This feature is particularly useful for monitoring critical directories and tracking changes immediately after they occur. Wazuh allows you to specify the directories or files in the monitored endpoints that would be reported in real-time if file changes occur.
- **Scheduled monitoring:** The frequency option in the Wazuh FIM module allows users to customize the scheduling of each FIM scan performed in your monitored endpoints. The default scan interval for the FIM module is 12 hours (43200 seconds) and can be customized on each endpoint. Alternatively, scans can be scheduled using the scan_time and the scan_day options. These options help users to set up FIM scans outside business hours or during holidays.
- **Who-data monitoring:** Wazuh captures advanced insights into file changes using the who-data functionality. This functionality uses audit tools like the Linux Audit subsystem and Microsoft Windows SACL to determine important information about the detected file changes. The who-data monitoring functionality allows the FIM module to obtain information on when the change event occurred, who or what made the change, and what content was changed. This is useful in maintaining accountability and validating if changes made to monitored files or directories were authorized and performed using approved processes.

Below is an example of an alert generated when a monitored file is changed on a Windows endpoint.



In alert fields, the *who-data* metadata shows that the user wazuh added the word Hello to the audit_docu.txt file using the Notepad.exe process.

- Reporting changes in file values: The FIM module provides a report_changes attribute that records and reports the exact content changed in a text file to the Wazuh server. The attribute enables the Wazuh agent to make copies of monitored files to a private location on each endpoint for further review. This monitoring option is helpful when users want to initiate specific responses when file changes in monitored directories match the behavior of known malicious activities. For example, the alert below indicates when Wazuh detects the creation of a web shell scripting file webshell-script.php in a monitored directory.



- **Recording file attributes:** Users can configure the FIM module to record specific attributes of a monitored file. Wazuh supports various file attributes that users can

use to specify the file metadata that the FIM module will record or ignore. For example, this monitoring option would be useful when users want to record only the SHA-256 hash of a configuration file, excluding other hash types.

Detecting and responding to malwarePermalink to this headline

The Wazuh FIM module integrates with other Wazuh capabilities and third-party threat intelligence solutions to create a comprehensive security monitoring environment. This is imperative to enhance malware detection and response capabilities, ensuring robust defense against cyber threats.

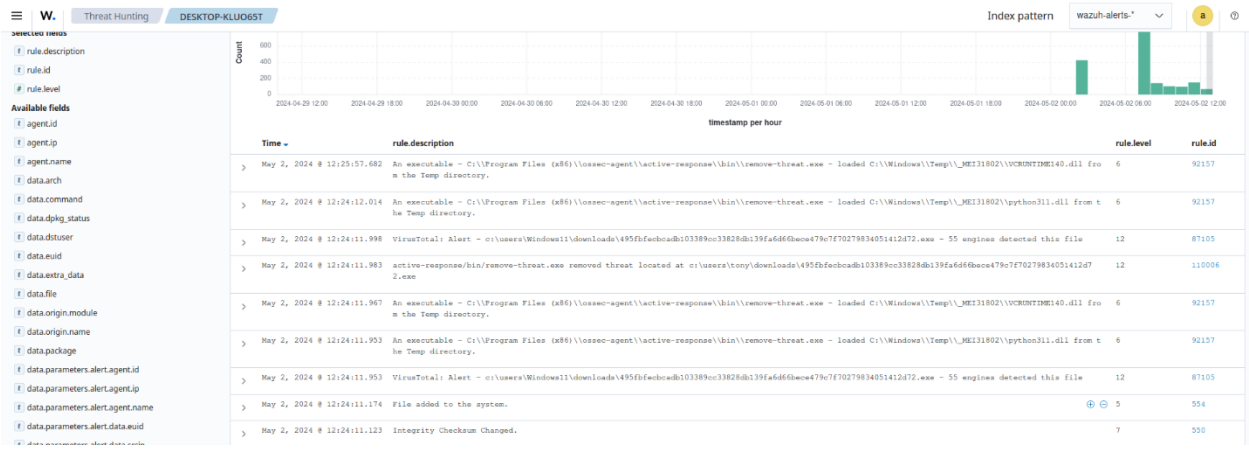
The Wazuh FIM module supports various integrations, including but not limited to:

- **File integrity monitoring and YARA**: By combining the Wazuh FIM module and the YARA tool, it is possible to detect malware when suspicious file additions or modifications are identified. The YARA rule files contain samples of malware indicators that are downloaded to the monitored endpoints. When the FIM module detects a change in the monitored file or directory, it executes a YARA scan using a script to determine if it is malware. If the YARA rule finds a match with a file, it will send the scan results to the Wazuh server for decoding and alerting. This would be reported according to the custom rule and decoder configurations configured on the Wazuh server. Check this documentation for more information on how to integrate the Wazuh FIM module with YARA.
- **File integrity monitoring and VirusTotal**: The Wazuh Integrator module connects to external APIs and alerting tools such as VirusTotal. The VirusTotal integration uses the VirusTotal API to detect malicious file hashes within the files and directories monitored by the FIM module. Once enabled, when FIM generates alerts, Wazuh initiates the VirusTotal integration to extract the hash value associated

with the flagged file from the alert. The VirusTotal API is then used to compare these hashes against its scanning engines for potentially malicious content.

- **File integrity monitoring and active response**: The Wazuh active response module automatically responds to threats identified in a timely manner. This combination enables the FIM module to not only detect but also respond to malicious activities. You can configure active response scripts to execute when the FIM module detects file changes in your monitored environment. Additionally, it also generates alerts for the response performed. This reduces the Mean Time To Respond (MTTR) as malicious changes detected are remediated in a timely manner.

  In the image below Wazuh triggers when a file is added to the monitored endpoint. The VirusTotal API scans the file and identifies it as malicious content on 55 engines. Then the Wazuh active response module acts immediately to remove the threat from the monitored endpoint.



- **File integrity monitoring and CDB list**: Wazuh FIM module also detects malicious files by checking the presence of known malware signatures when combined with CDB lists (constant database). CDB lists are used to store known malware indicators of Compromise (IOCs) such as file hashes, IP addresses, and domain names. When CDB lists are created, Wazuh checks if field values from FIM alerts

such as file hash match the keys stored in the CDB lists. If matched, it generates an alert and response based on how you configure your custom rule.

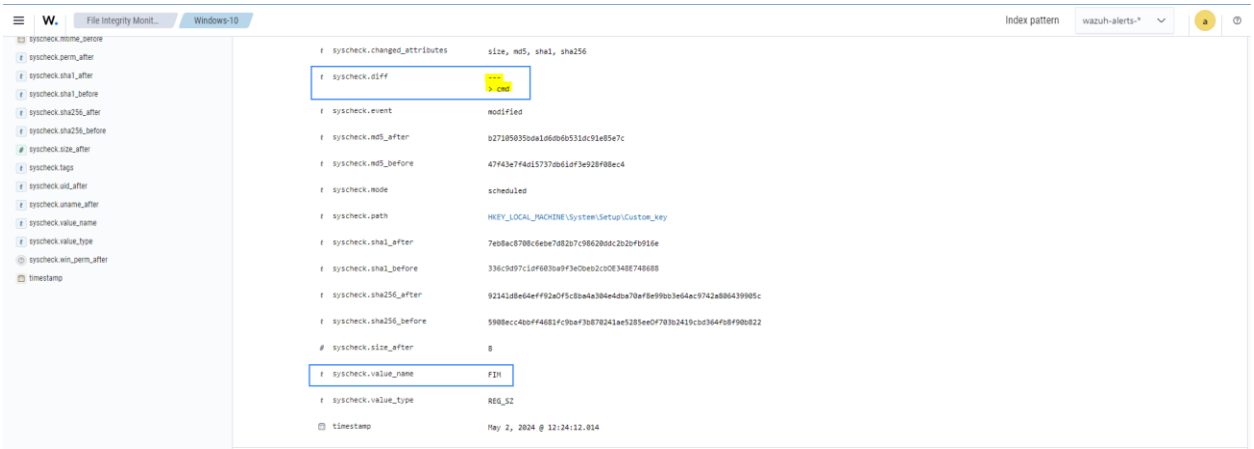
Monitoring Windows RegistryPermalink to this headline

The Wazuh FIM module periodically scans Windows Registry entries, stores its checksums and attributes in a local database, and alerts when changes in registry values are detected. This would keep users informed about registry modifications resulting from user activities or software installations whether malicious or not.

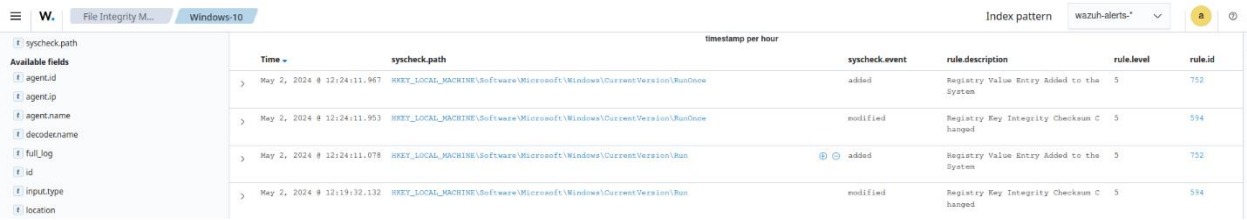
You can configure the Wazuh open source FIM module to monitor Windows Registry values using various configuration options. The report_changes attribute in the windows_registry option provides a granular breakdown of modification detected in the monitored Windows Registry value. You can configure which Windows Registry attributes the module would record or ignore. For example, you can choose to record the check_sha1sum attribute and ignore the check_md5sum attribute, if your CDB list only contains SHA1 hashes of malicious files.

The image below shows the event of a modified Windows registry value in a monitored endpoint.

The alert when expanded shows the modified field.



Threat actors maintain persistence by commonly adding programs for their malicious activities to the *Run* and *RunOnce* keys in the Registry. Additionally, Wazuh detects any suspicious programs added to the startup registry keys. This allows you to take appropriate action to remove them before they cause harm to your system.

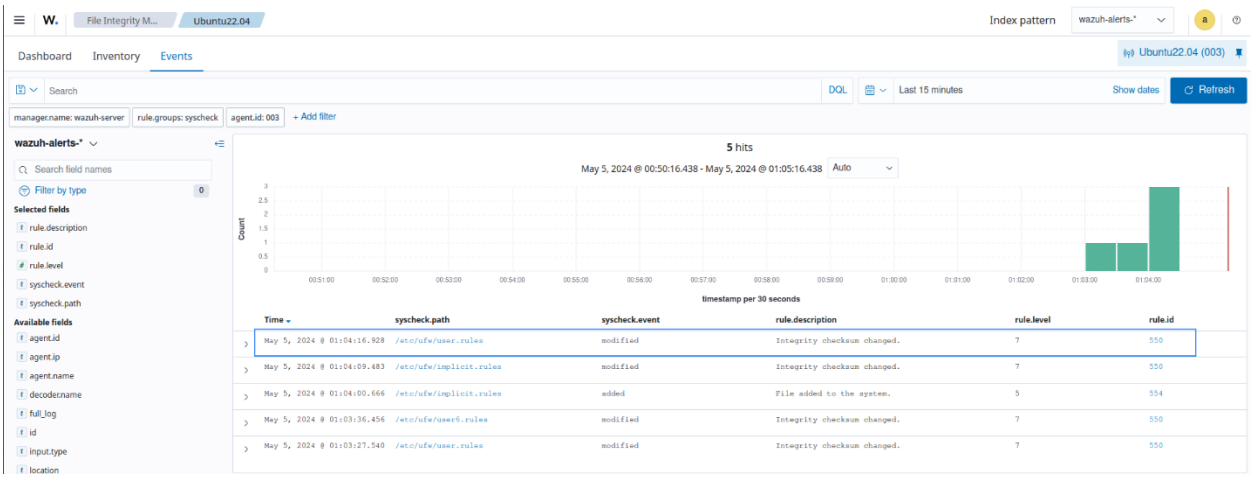


Meeting regulatory compliance[Permalink to this headline]

Meeting regulatory compliance requirements is an important consideration for organizations in various industries. File integrity monitoring is a requirement for achieving compliance with regulations such as PCI DSS, SOX, HIPAA, NIST SP 800-53, among others.

You can customize the Wazuh FIM module to monitor specific files and directories where your organization's sensitive and confidential data are stored. Wazuh provides a comprehensive report that outlines the changes made to the files and directories being monitored. This feature is particularly useful for ensuring compliance with various regulatory standards.

For example, organizations can meet the CM-3 Configuration change control requirement in NIST SP 800-53 standard by using Wazuh. The control requires organizations to protect information at rest and monitor configuration changes in their infrastructure. The image below shows an event generated when the permissions for Uncomplicated Firewall (UFW) rule files are modified on a monitored endpoint.



Log data analysisPermalink to this headline

Log data analysis is a crucial process that involves examining and extracting valuable insights from log files created by different systems, applications, or devices. These logs contain records of events that provide useful information for troubleshooting, security analysis and monitoring, and optimizing performance. Log data analysis is an essential practice that contributes to a secure, efficient, and reliable IT ecosystem.

Wazuh collects, analyzes, and stores logs from endpoints, network devices, and applications. The Wazuh agent, running on a monitored endpoint collects and forwards system and application logs to the Wazuh server for analysis. Additionally, you can send log messages to the Wazuh server via syslog or third-party API integrations.

Log data collectionPermalink to this headline

Wazuh collects logs from a wide range of sources, enabling comprehensive monitoring of various aspects of your IT environment. You can check our documentation on Log data collection to understand better how Wazuh collects and analyzes logs from monitored endpoints. Some of the common log sources supported by Wazuh include:

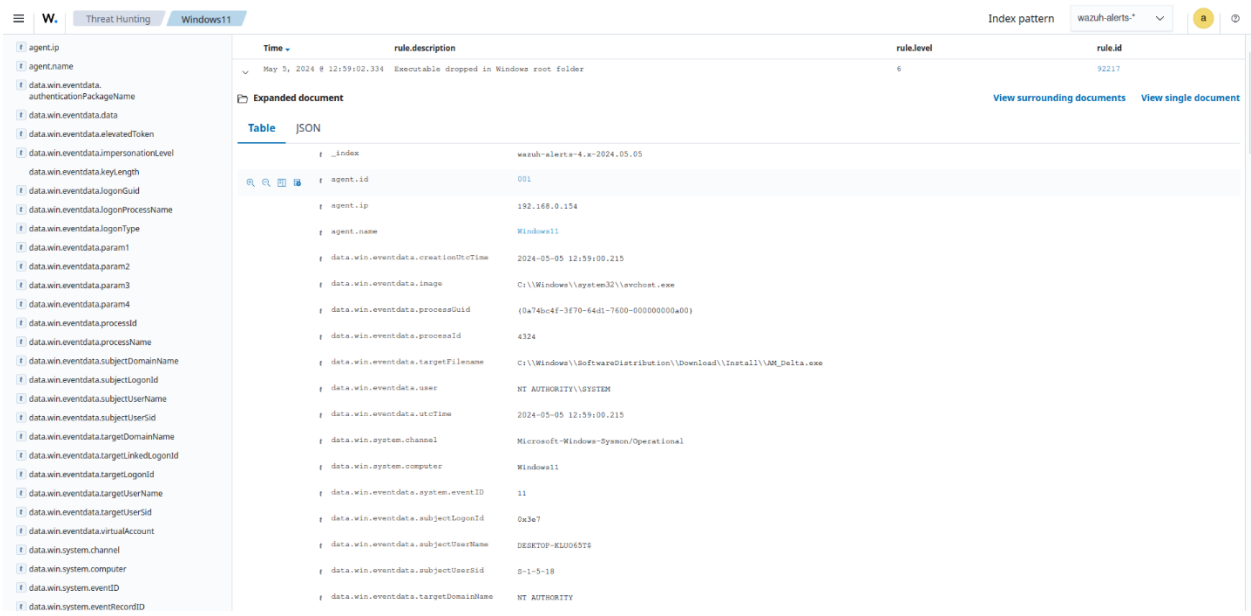
- **Operating system logs**: Wazuh collects logs from several operating systems, including Linux, Windows, and macOS.

  Wazuh can collect syslog, auditd, application logs, and others from Linux endpoints.

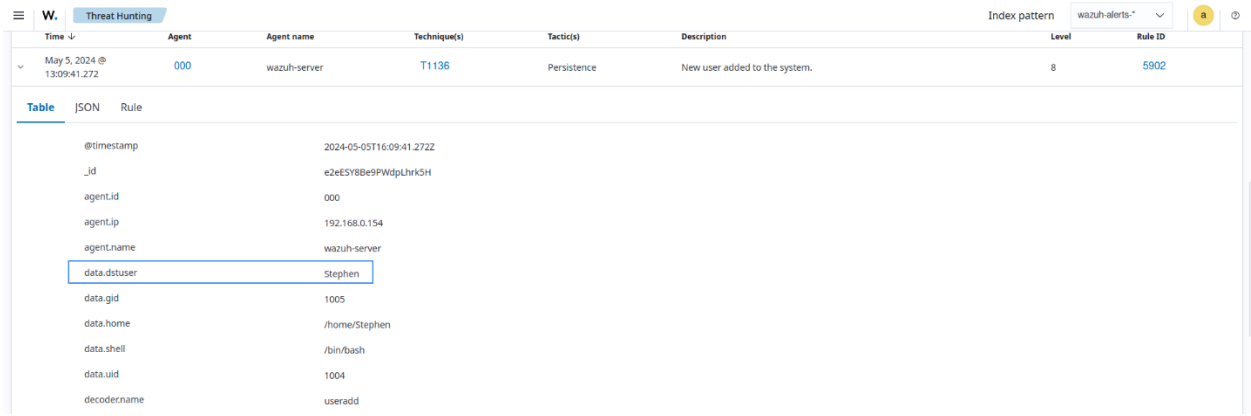
  Wazuh collects logs on Windows endpoints using the Windows event channel and Windows event log format. By default, the Wazuh agent monitors the System, Application, and Security Windows event channels on Windows endpoints. The Wazuh agent offers the flexibility to configure and monitor other Windows event channels.

  Wazuh utilizes the unified logging system (ULS) to collect logs on macOS endpoints. The macOS ULS centralizes the management and storage of logs across all the system levels.

  The image below shows an event collected from the Microsoft-Windows-Sysmon/Operational event channel on a Windows endpoint.

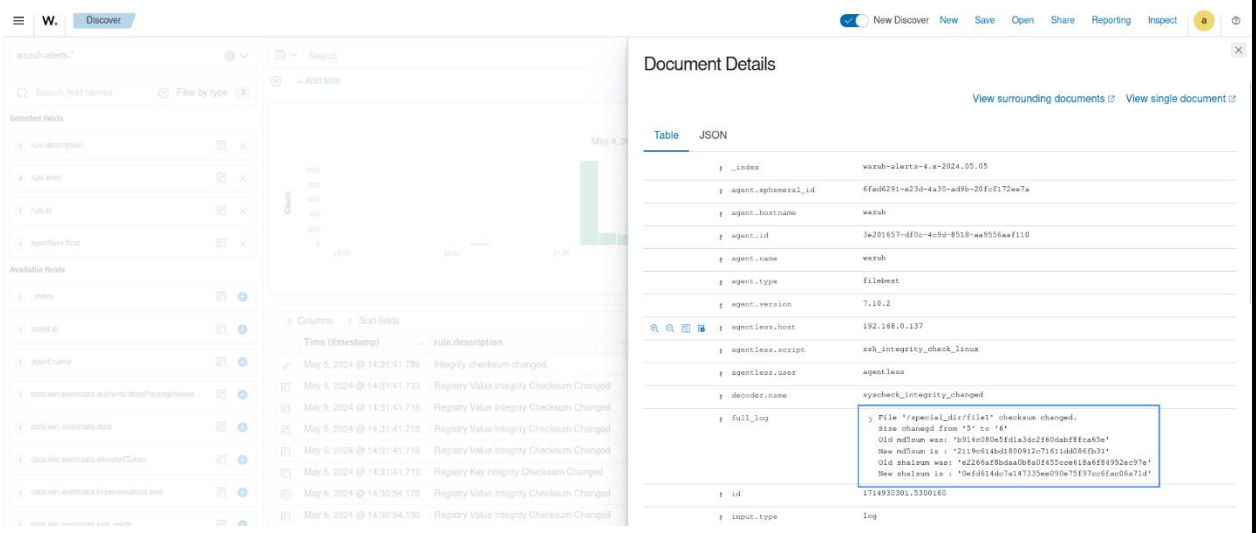


- **Syslog events**: Wazuh gathers logs from syslog-enabled devices, encompassing a wide array of sources including Linux/Unix systems and network devices that do not support agent installation. The image below shows an alert triggered when a new user is created on the Linux endpoint and the log is forwarded to the Wazuh server via rsyslog.

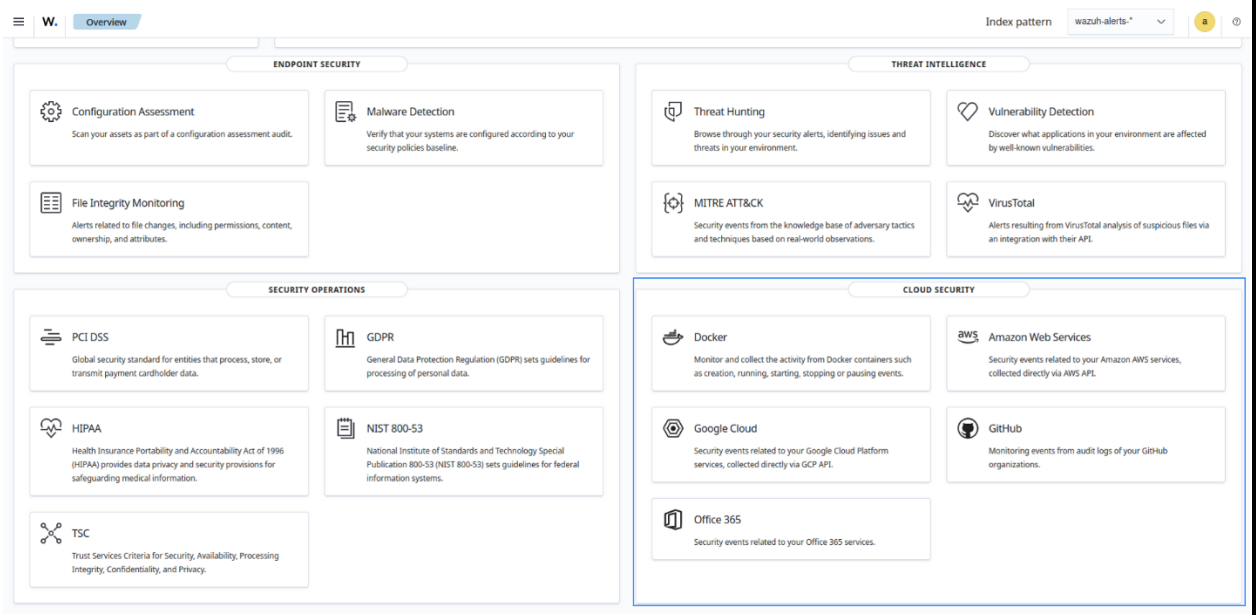


- **Agentless monitoring**: The Wazuh agentless monitoring module monitors endpoints that don't support agent installation. It requires an SSH connection between the endpoint and the Wazuh server. The Wazuh agentless monitoring module monitors

files, directories, or configurations and runs commands on the endpoint. The image below is an alert from an agentless device on the Wazuh dashboard.



- **Cloud provider logs**: Wazuh integrates with cloud providers like AWS, Azure, Google Cloud, and Office 365 to collect logs from cloud services such as EC2 instances, S3 buckets, Azure VMs, and more. The image below shows the **CLOUD SECURITY** section in the Wazuh dashboard.



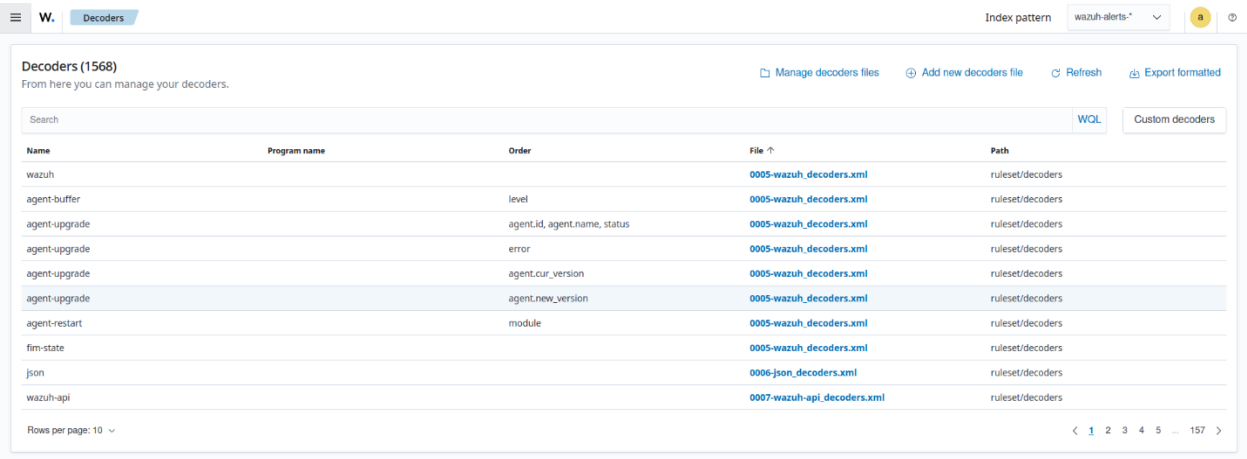
- **Custom logs**: You can configure Wazuh to collect and parse logs from several applications and third-party security tools like VirusTotal, Windows Defender, and ClamAV. The image below shows an alert of a log from VirusTotal processed by the Wazuh server.

Rules and decodersPermalink to this headline

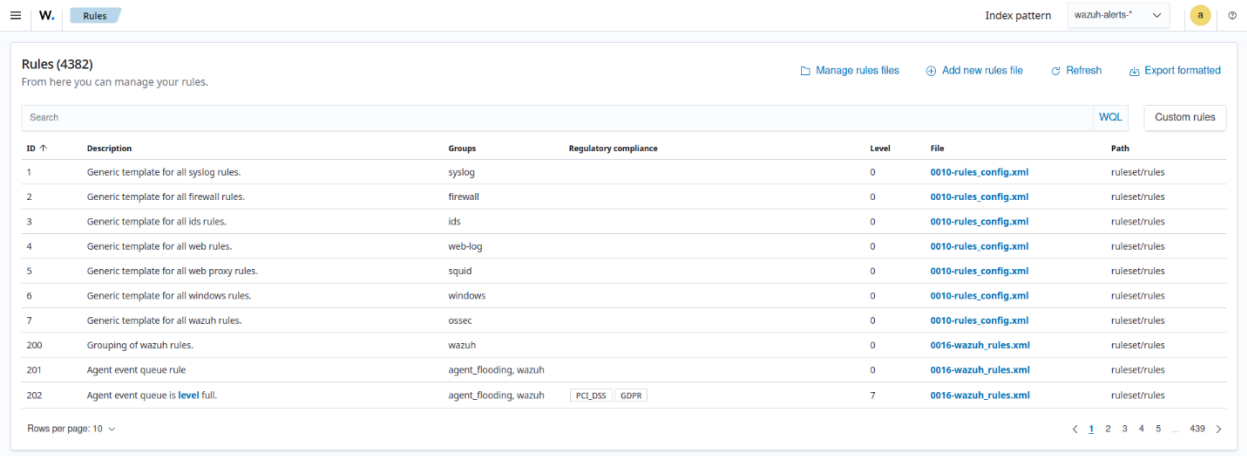
Wazuh rules and decoders are core components in log data analysis and threat detection and response. Wazuh provides a powerful platform for log data analysis, allowing organizations to enhance their security posture by promptly detecting and responding to potential security threats.

Wazuh decoders are responsible for parsing and normalizing log data collected from various sources. Decoders are essential for converting the raw log data in several formats into a unified and structured format that Wazuh can process effectively. Wazuh has pre-built decoders for common log formats such as syslog, Windows event channel, macOS ULS, and more. Additionally, Wazuh allows you to define custom decoders for parsing logs from specific applications or devices with unique log formats. By using decoders, Wazuh can efficiently interpret log data and extract relevant information, such as timestamps, log levels, source IP addresses, user names, and more. As shown below, you can view Wazuh

out-of-the-box and custom decoders on the **Server management** > **Decoders** of the Wazuh dashboard.



Wazuh ruleset detects security events and anomalies in log data. These rules are written in a specific format and they trigger alerts when certain conditions are met. The rules are defined based on certain criteria like log fields, values, or patterns to match specific log entries that may indicate security threats. Wazuh provides a wide range of pre-built rules covering common security use cases. Additionally, administrators can create custom rules tailored to their specific environment and security requirements. The **Server management** category of the Wazuh dashboard lets you view the default and custom **Rules**.



For example, the rule below includes a match field used to define the pattern that the rule looks for. The rule also has a level field that specifies the priority of the resulting alert. Additionally, rules enrich events with technique identifiers from the MITRE ATT&CK framework and map them to regulatory compliance controls.

```
<rule id="5715" level="3">
  <if_sid>5700</if_sid>
  <match>^Accepted|authenticated.$</match>
  <description>sshd: authentication success.</description>
  <mitre>
    <id>T1078</id>
    <id>T1021</id>
  </mitre>

<group>authentication_success,gdpr_IV_32.2,gpg13_7.1,gpg13_7.2,hipaa_164.312.b,nist_800_53_AU.14,nist_800_53_AC.7,pci_dss_10.2.5,tsc_CC6.8,tsc_CC7.2,tsc_CC7.3,</group>
</rule>
```

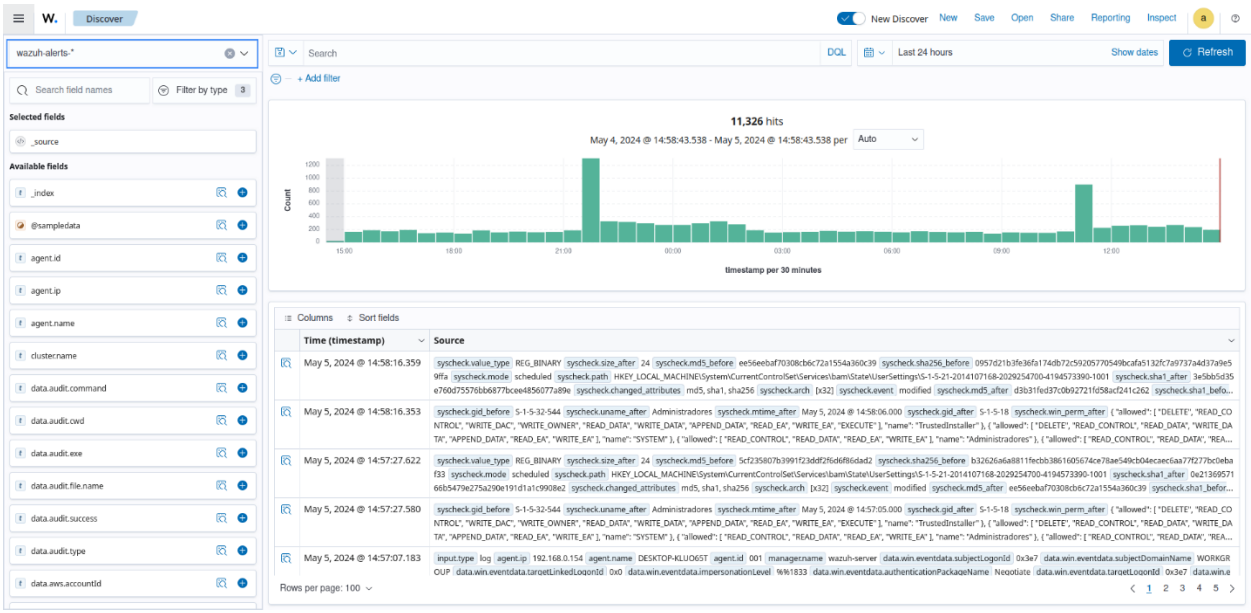
Log data indexing and storagePermalink to this headline

The Wazuh indexer is a highly scalable, distributed real-time search and analytics engine. The Wazuh indexer is critical in log analysis as it stores and indexes alerts generated by the Wazuh server. These alerts are stored as JSON documents.

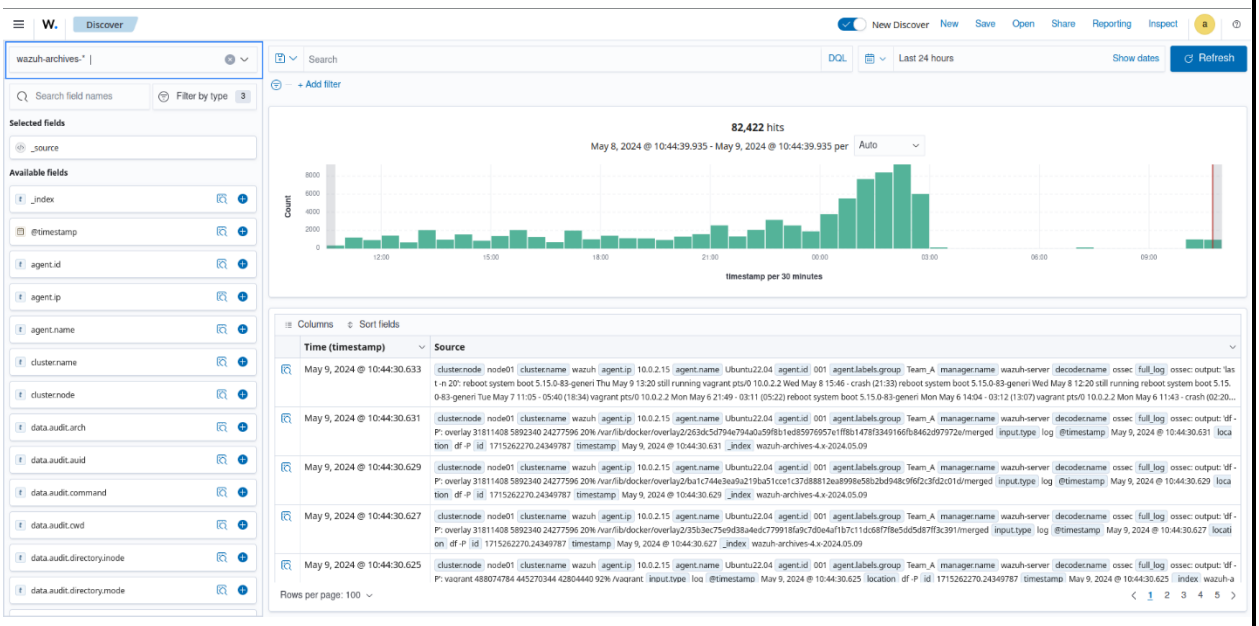
The Wazuh indexer guarantees redundancy by storing the JSON documents across several containers called shards and distributing the shards across multiple nodes. This implementation prevents downtime when hardware failures or cyber-attacks occur and increases query capacity as nodes are added to a cluster.

Wazuh uses four indices to store several event types:

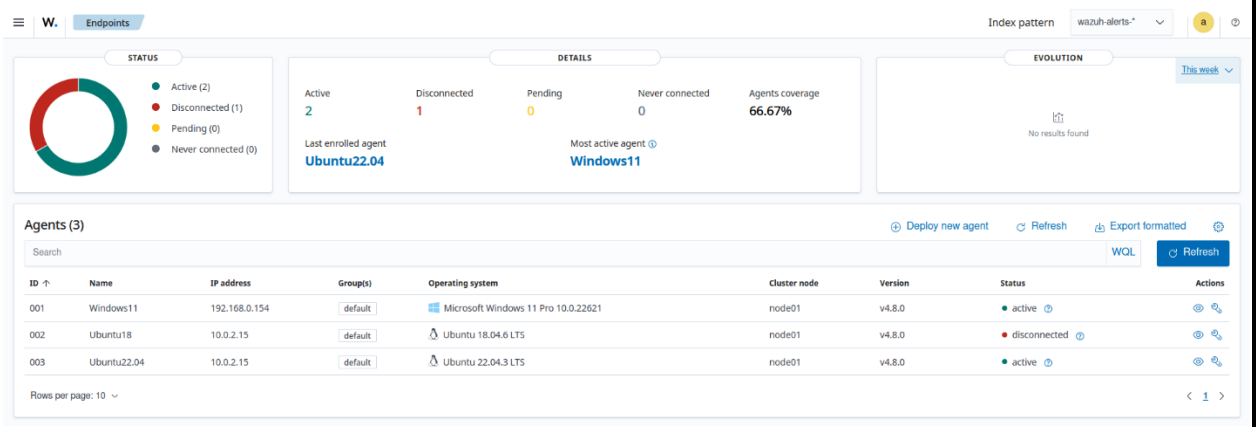
- **wazuh-alerts** stores alerts generated by the Wazuh server when an event triggers a rule with high enough priority. The image below shows alerts in the **Discover** module of the Wazuh dashboard. The index pattern is set to wazuh-alerts-* by default.



- **wazuh-archives** index stores all events received from the Wazuh server regardless of whether they trigger an alert. The Wazuh archives use this index to enable log retention and querying capabilities that offer deeper insight into events happening within monitored endpoints. Wazuh archives are disabled by default because of the huge storage requirements needed to store all the logs. The image below shows archived events in the **Discover** section of Wazuh dashboard with the index pattern set to wazuh-archives-*.

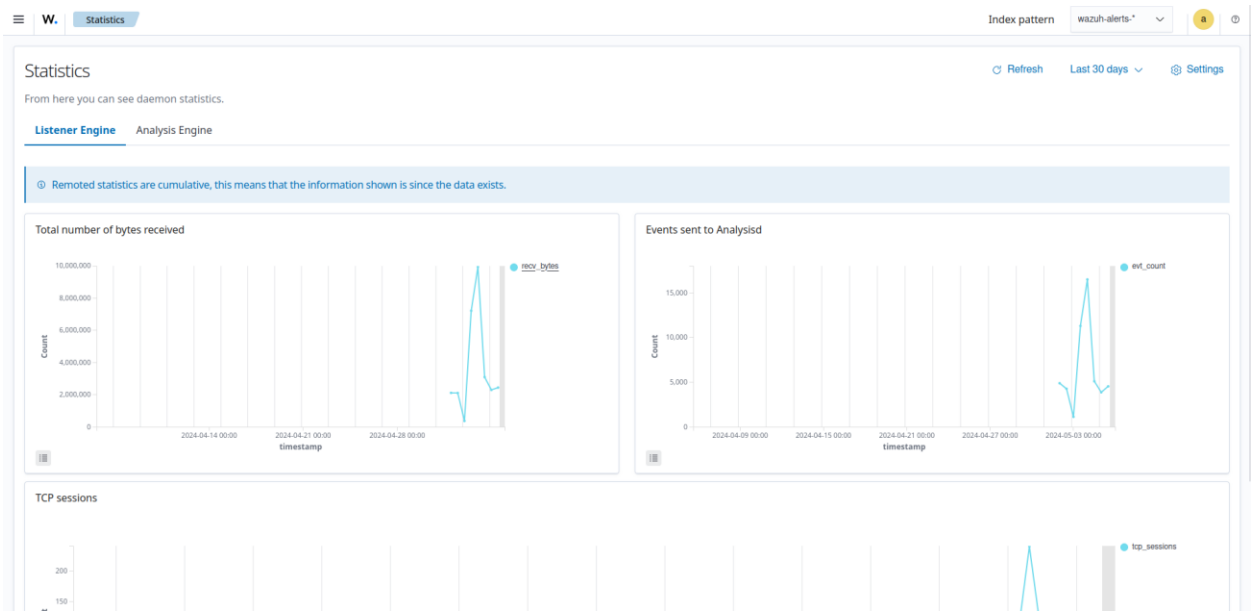


- **wazuh-monitoring** index stores data about the state of Wazuh agents over a period of time. The state of the agent could be Active, Disconnected, or Never connected. This information is very useful in tracking Wazuh agents that are not reporting to the dashboard for several reasons that need investigation. The image below shows the connection status of the agents on the Wazuh dashboard. The agent information as shown in the image is collected from the wazuh-monitoring index.



- **wazuh-statistics** index stores performance data related to the Wazuh server. This information is critical to ensuring the Wazuh server performs optimally with the

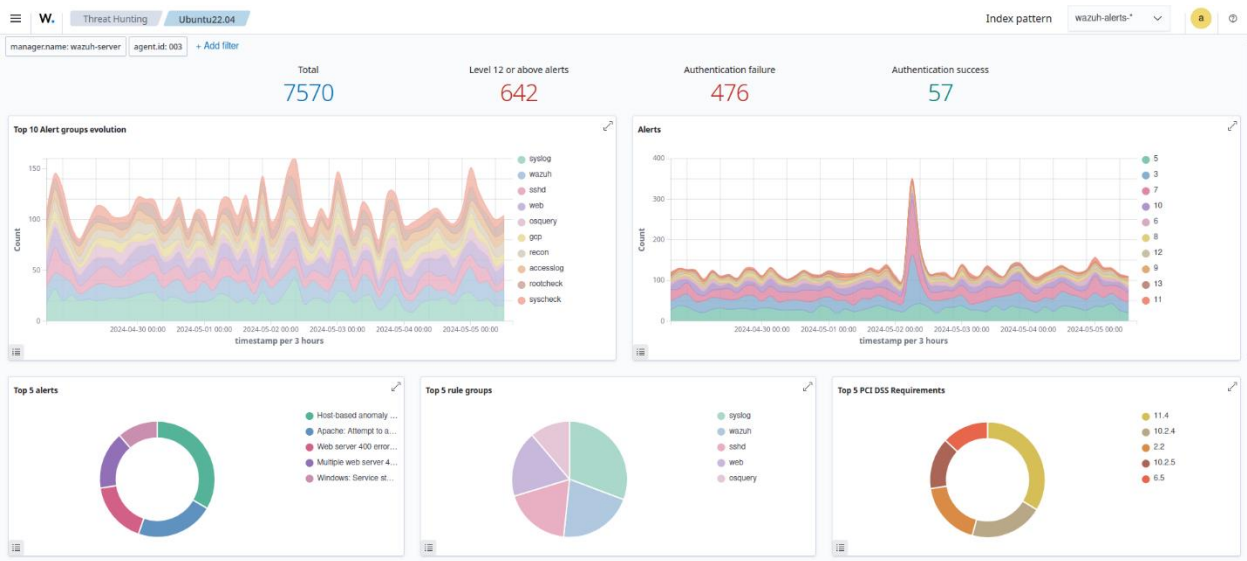
available computing resources. The image below shows performance-related events on the Wazuh dashboard.



Log data querying and visualizationPermalink to this headline

The Wazuh dashboard offers log data querying and visualization capabilities. You can leverage the dashboard's intuitive interface to conduct complex searches and queries to extract meaningful insights from the log data collected by Wazuh.

Wazuh provides a set of predefined dashboards and visualizations out of the box, specifically tailored to security monitoring and compliance use cases. These dashboards provide insight into common security events such as failed logins, malware detection, and system anomalies. You can further customize these dashboards to suit your specific needs and requirements. Below is a sample image of the **Security event** dashboard showing several interesting information like **Top 5 PCI DSS Requirements**, **Top 5 alerts**, and **Alert groups evolution**.



The Wazuh dashboard enables users to explore log entries in real time, apply various filters, and drill down into specific events or time ranges. This flexibility allows security analysts to identify trends, anomalies, and potential security incidents within their environment.

Wazuh allows users to create customized dashboards that display key performance indicators, security metrics, and real-time monitoring of critical systems and applications. Users can assemble multiple visualizations, such as pie charts, line graphs, and heat maps, onto a single dashboard, providing a holistic view of their infrastructure's security posture. The following blog posts detailed how to query and create custom dashboards:

- Monitoring macOS resources with Wazuh
- Monitoring Linux resources with Wazuh
- Monitoring Windows resources with Performance Counters

Vulnerability detectionPermalink to this headline

Software vulnerabilities are weaknesses in code that can allow attackers to gain access to or manipulate the behavior of an application. Vulnerable software applications are

commonly targeted by attackers to compromise endpoints and gain a persistent presence on targeted networks.

Vulnerability detection is the process of identifying these flaws before they are discovered and exploited by attackers. The goal of vulnerability detection is to identify vulnerabilities so that remediation can be carried out to prevent successful attacks.

The Wazuh agent uses the Syscollector module to collect inventory details from the monitored endpoint. It sends the collected data to the Wazuh server. Within the Wazuh server, the Vulnerability Detection module correlates the software inventory data with vulnerability content documents to detect vulnerable software on the monitored endpoint.

Wazuh detects vulnerable applications, generating risk reports, using our Cyber Threat Intelligence (CTI) platform. In this platform, we aggregate vulnerability data from diverse sources like operating system vendors and vulnerability databases, consolidating it into a unified, reliable repository. The process involves standardizing the varied formats into a common structure. Additionally, we maintain the integrity of our vulnerability data by doing the following.

- Rectifying format inconsistencies like version errors and typos.
- Completing missing information.
- Incorporating new cybersecurity vulnerabilities.

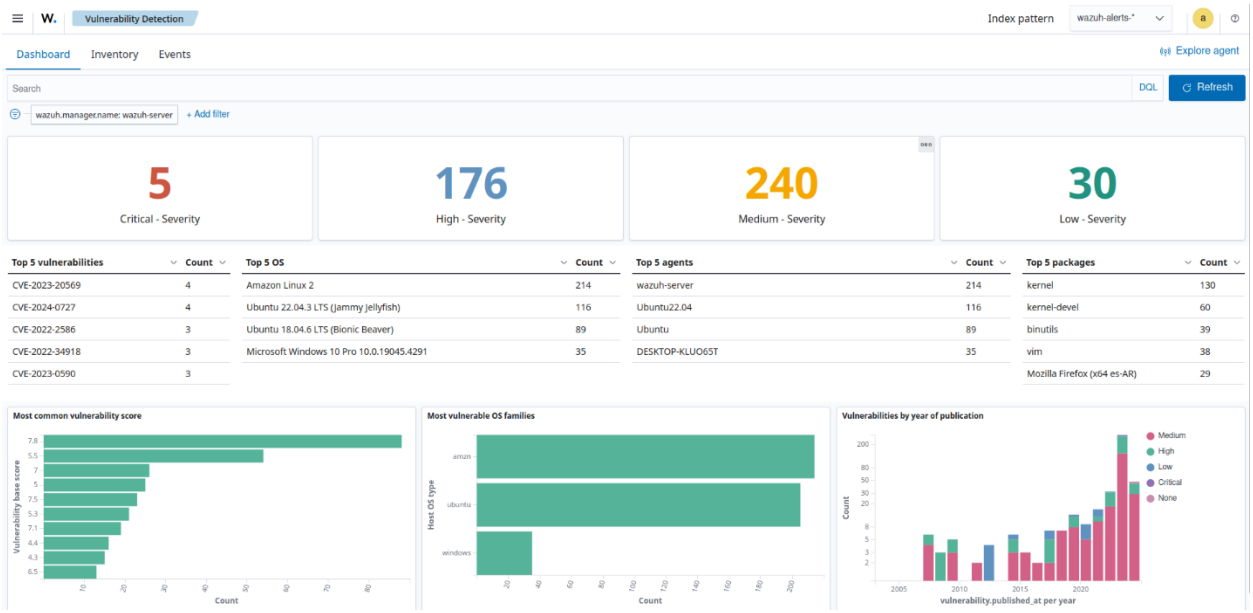
Subsequently, we merge this content, uploading the compiled documents to a cloud server. Finally, we publish these documents to our CTI API.

Relying on the Wazuh CTI, the Vulnerability Detection module supports a variety of operating systems, such as Windows, CentOS, Red Hat Enterprise Linux, Ubuntu, Debian, Amazon Linux, Arch Linux, and macOS operating systems, and applications.

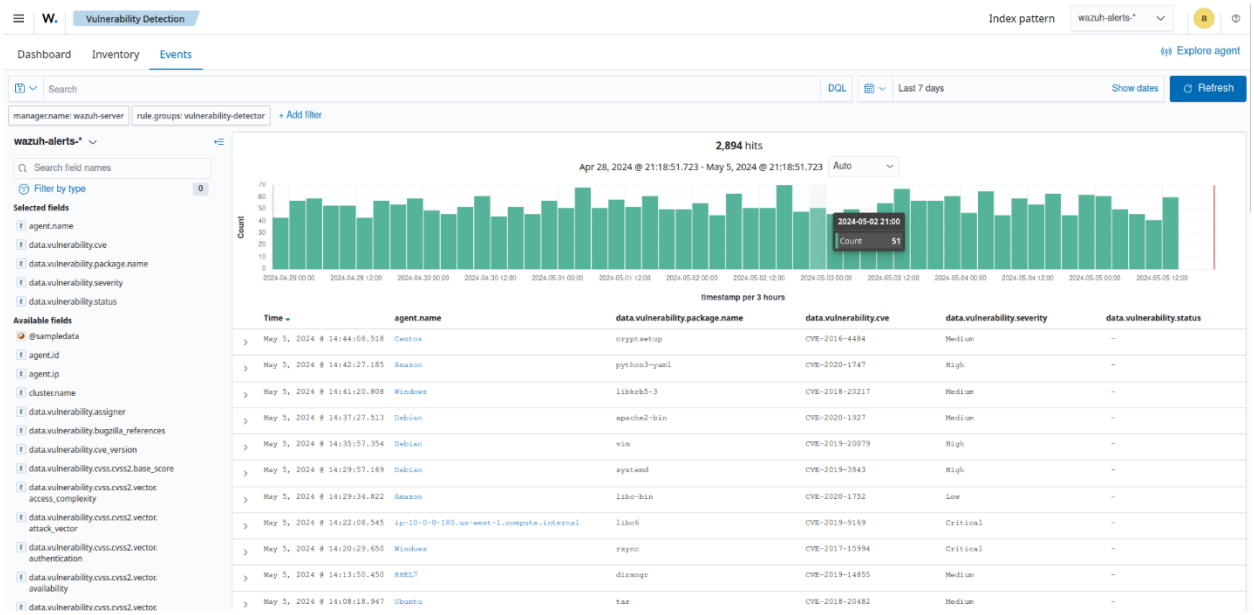
Achieve comprehensive visibility[Permalink to this headline]

The Vulnerability Detection module generates alerts for vulnerabilities discovered on the operating system and applications installed on the monitored endpoint. It correlates the software inventory collected by the Wazuh agent with the vulnerability content documents and displays the alert generated on the Wazuh dashboard. This provides a clear and comprehensive view of vulnerabilities identified in all monitored endpoints, allowing you to view, analyze and fix vulnerabilities.

The vulnerability detection dashboard shows the frequency of occurrences in different categories such as package name, operating system, agent name, vulnerability ID, and alert severity. This allows analysts to direct their focus appropriately.



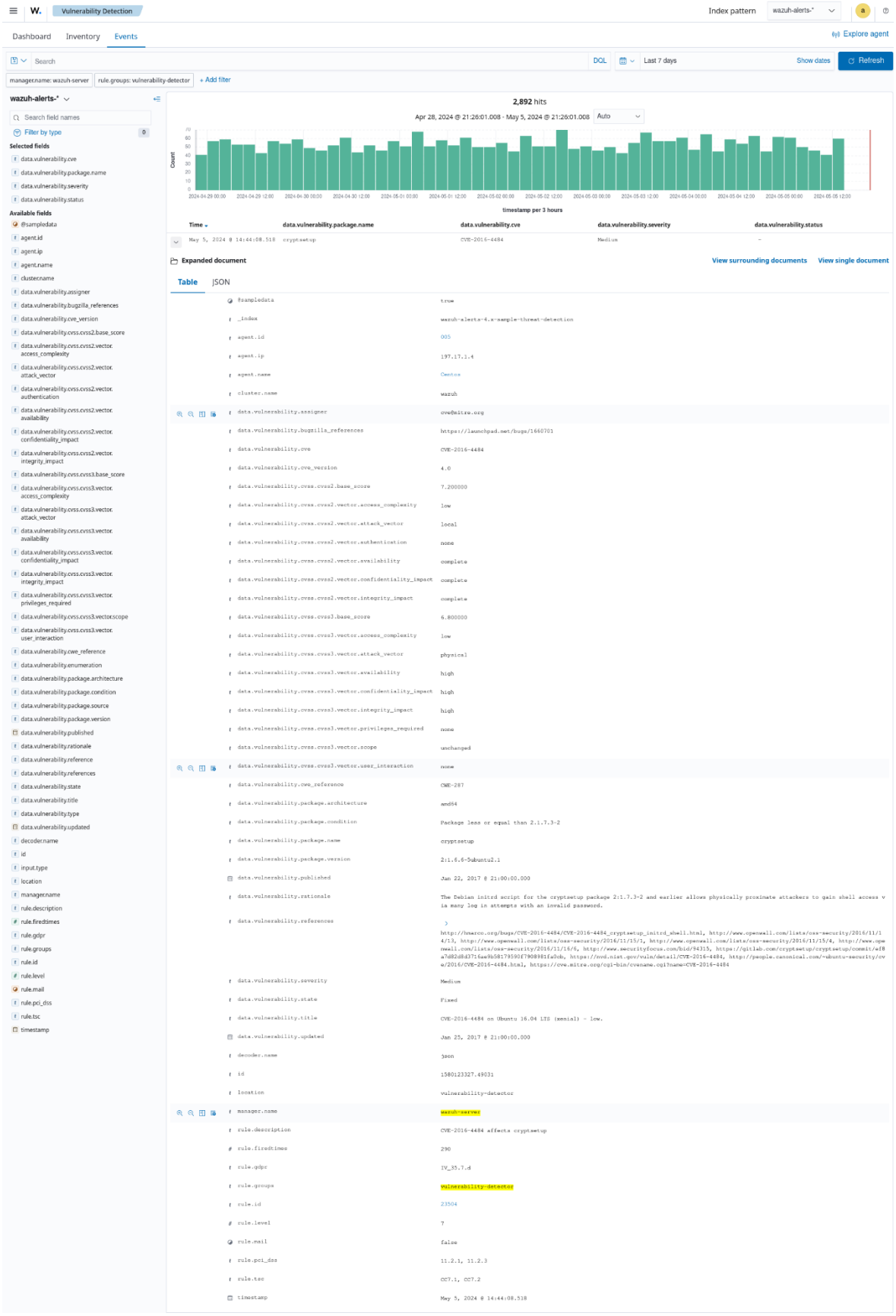
You can view the alerts generated on the dashboard when new vulnerabilities are discovered.



The alerts generated on the dashboard could also be a result of remediation activities. The image below shows alerts generated after an upgrade or an uninstallation of a package resolved a vulnerability.

Obtain actionable intelligence from vulnerability alerts[Permalink to this headline]

Wazuh vulnerability alerts contain relevant information about the identified vulnerability which can help users understand and decide on remediation steps. You can see an example of a vulnerability detection alert below:


Vulnerability Detection
Index pattern
wazuh-alerts-*
Dashboard
Inventory
Events
Explore agent
Search
DQL
Last 7 days
Show dates
Refresh
manager.name: wazuh-server
rule.groups: vulnerability-detector
+ Add filter
wazuh-alerts-*
Search field names
Filter by type
Selected fields
data.vulnerability.cve
data.vulnerability.package.name
data.vulnerability.severity
data.vulnerability.status
Available fields
2,892 hits
Apr 28, 2024 @ 21:26:01.008 - May 5, 2024 @ 21:26:01.008
Auto
Count
timestamp per 3 hours
Time
data.vulnerability.package.name
data.vulnerability.cve
data.vulnerability.severity
data.vulnerability.status
May 5, 2024 @ 14:44:08.518
cryptsetup
CVE-2016-4484
Medium
Expanded document
View surrounding documents
View single document
Table
JSON
_index
wazuh-alerts-4.x-sample-threat-detection
agent.id
005
agent.ip
197.17.1.4
agent.name
Centos
cluster.name
wazuh
data.vulnerability.assigner
cve@mitre.org
data.vulnerability.bugzilla_references
https://launchpad.net/bugs/1660701
data.vulnerability.cve
CVE-2016-4484
data.vulnerability.cve_version
4.0
data.vulnerability.cvss.cvss2.base_score
7.200000
data.vulnerability.cvss.cvss2.vector.access_complexity
low
data.vulnerability.cvss.cvss2.vector.attack_vector
local
data.vulnerability.cvss.cvss2.vector.authentication
none
data.vulnerability.cvss.cvss2.vector.availability
complete
data.vulnerability.cvss.cvss2.vector.confidentiality_impact
complete
data.vulnerability.cvss.cvss2.vector.integrity_impact
complete
data.vulnerability.cvss.cvss3.base_score
6.800000
data.vulnerability.cvss.cvss3.vector.access_complexity
low
data.vulnerability.cvss.cvss3.vector.attack_vector
physical
data.vulnerability.cvss.cvss3.vector.availability
high
data.vulnerability.cvss.cvss3.vector.confidentiality_impact
high
data.vulnerability.cvss.cvss3.vector.integrity_impact
high
data.vulnerability.cvss.cvss3.vector.privileges_required
none
data.vulnerability.cvss.cvss3.vector.scope
unchanged
data.vulnerability.cvss.cvss3.vector.user_interaction
none
data.vulnerability.cwe_reference
CWE-287
data.vulnerability.package.architecture
amd64
data.vulnerability.package.condition
Package less or equal than 2.1.7.3-2
data.vulnerability.package.name
cryptsetup
data.vulnerability.package.version
2:1.6.6-5ubuntu2.1
data.vulnerability.published
Jan 22, 2017 @ 21:00:00.000
data.vulnerability.rationale
The Debian initrd script for the cryptsetup package 2:1.7.3-2 and earlier allows physically proximate attackers to gain shell access v
ia many log in attempts with an invalid password.
data.vulnerability.references
http://bmarco.org/bugs/CVE-2016-4484/CVE-2016-4484_cryptsetup_initrd_shell.html, http://www.openwall.com/lists/oss-security/2016/11/1
4/13, http://www.openwall.com/lists/oss-security/2016/11/15/1, http://www.openwall.com/lists/oss-security/2016/11/15/4, http://www.ope
nwall.com/lists/oss-security/2016/11/16/4, http://www.securityfocus.com/bid/94315, https://gitlab.com/cryptsetup/cryptsetup/commit/ef8
a7d82d8d3716ae9b5817959f7908981fa0db, https://nvd.nist.gov/vuln/detail/CVE-2016-4484, http://people.canonical.com/~ubuntu-security/cv
e/2016/CVE-2016-4484.html, https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2016-4484
data.vulnerability.severity
Medium
data.vulnerability.state
Fixed
data.vulnerability.title
CVE-2016-4484 on Ubuntu 16.04 LTS (xenial) - low.
data.vulnerability.updated
Jan 25, 2017 @ 21:00:00.000
decoder.name
json
id
1580123327.49031
location
vulnerability-detector
manager.name
wazuh-server
rule.description
CVE-2016-4484 affects cryptsetup
rule.firedtimes
290
rule.gdpr
IV_35.7.d
rule.groups
vulnerability-detector
rule.id
23504
rule.level
7
rule.mail
false
rule.pci_dss
11.2.1, 11.2.3
rule.tsc
CC7.1, CC7.2
timestamp
May 5, 2024 @ 14:44:08.518

```
{
 "_index": "wazuh-alerts-4.x-sample-threat-detection",
 "_id": "e2ffSY8Be9PWdpLhA_nt",
 "_version": 1,
 "_score": null,
 "_source": {
  "predecoder": {},
  "cluster": {
   "name": "wazuh"
  },
  "agent": {
   "ip": "197.17.1.4",
   "name": "Centos",
   "id": "005"
  },
  "manager": {
   "name": "wazuh-server"
  },
  "data": {
   "vulnerability": {
    "severity": "Medium",
    "package": {
     "condition": "Package less or equal than 2.1.7.3-2",
     "name": "cryptsetup",
     "version": "2:1.6.6-5ubuntu2.1",
     "architecture": "amd64"
    },
    "references": [
     "http://hmarco.org/bugs/CVE-2016-4484/CVE-2016-4484_cryptsetup_initrd_shell.html",
     "http://www.openwall.com/lists/oss-security/2016/11/14/13",
     "http://www.openwall.com/lists/oss-security/2016/11/15/1",
     "http://www.openwall.com/lists/oss-security/2016/11/15/4",
     "http://www.openwall.com/lists/oss-security/2016/11/16/6",
     "http://www.securityfocus.com/bid/94315",
```

```
"https://gitlab.com/cryptsetup/cryptsetup/commit/ef8a7d82d8d3716ae9b58179590f7908981fa0cb",
        "https://nvd.nist.gov/vuln/detail/CVE-2016-4484",
        "http://people.canonical.com/~ubuntu-security/cve/2016/CVE-2016-4484.html",
        "https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2016-4484"
      ],
      "cve_version": "4.0",
      "assigner": "cve@mitre.org",
      "published": "2017-01-23",
      "cwe_reference": "CWE-287",
      "title": "CVE-2016-4484 on Ubuntu 16.04 LTS (xenial) - low.",
      "rationale": "The Debian initrd script for the cryptsetup package 2:1.7.3-2 and earlier allows physically proximate attackers to gain shell access via many log in attempts with an invalid password.",
      "cve": "CVE-2016-4484",
      "state": "Fixed",
      "bugzilla_references": [
        "https://launchpad.net/bugs/1660701"
      ],
      "cvss": {
        "cvss2": {
          "base_score": "7.200000",
          "vector": {
            "integrity_impact": "complete",
            "confidentiality_impact": "complete",
            "availability": "complete",
            "attack_vector": "local",
            "access_complexity": "low",
            "authentication": "none"
          }
        },
        "cvss3": {
          "base_score": "6.800000",
          "vector": {
            "user_interaction": "none",
```

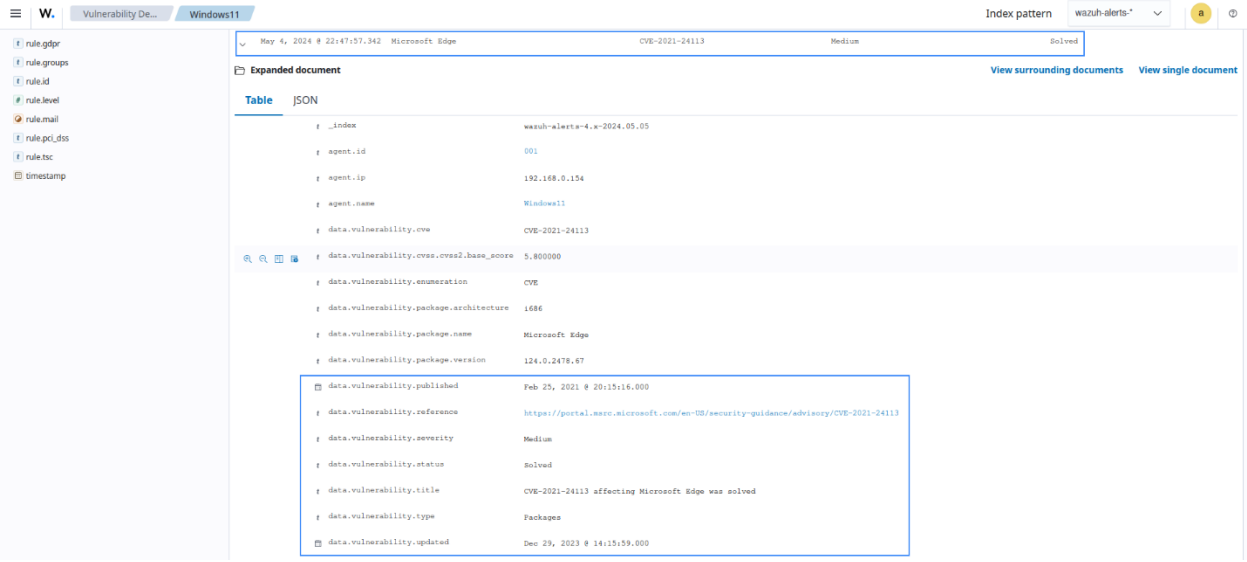
```
          "integrity_impact": "high",
          "scope": "unchanged",
          "confidentiality_impact": "high",
          "availability": "high",
          "attack_vector": "physical",
          "access_complexity": "low",
          "privileges_required": "none"
        }
      }
    },
    "updated": "2017-01-26"
  }
},
"@sampledata": true,
"rule": {
  "firedtimes": 290,
  "mail": false,
  "level": 7,
  "pci_dss": [
    "11.2.1",
    "11.2.3"
  ],
  "tsc": [
    "CC7.1",
    "CC7.2"
  ],
  "description": "CVE-2016-4484 affects cryptsetup",
  "groups": [
    "vulnerability-detector"
  ],
  "id": "23504",
  "gdpr": [
    "IV_35.7.d"
  ]
},
"location": "vulnerability-detector",
```

```
  "id": "1580123327.49031",
  "decoder": {
    "name": "json"
  },
  "timestamp": "2024-05-05T17:44:08.518+0000"
},
"fields": {
  "data.vulnerability.published": [
    "2017-01-23T00:00:00.000Z"
  ],
  "data.vulnerability.updated": [
    "2017-01-26T00:00:00.000Z"
  ],
  "timestamp": [
    "2024-05-05T17:44:08.518Z"
  ]
},
"highlight": {
  "manager.name": [
    "@opensearch-dashboards-highlighted-field@wazuh-server@/opensearch-dashboards-highlighted-field@"
  ],
  "rule.groups": [
    "@opensearch-dashboards-highlighted-field@vulnerability-detector@/opensearch-dashboards-highlighted-field@"
  ]
},
"sort": [
  1714931048518
]
}
```


As you can see above, the alert contains key information about the detected vulnerability. This information includes the CVE information, reference links for further research, and a description that provides a concise explanation of the vulnerability.

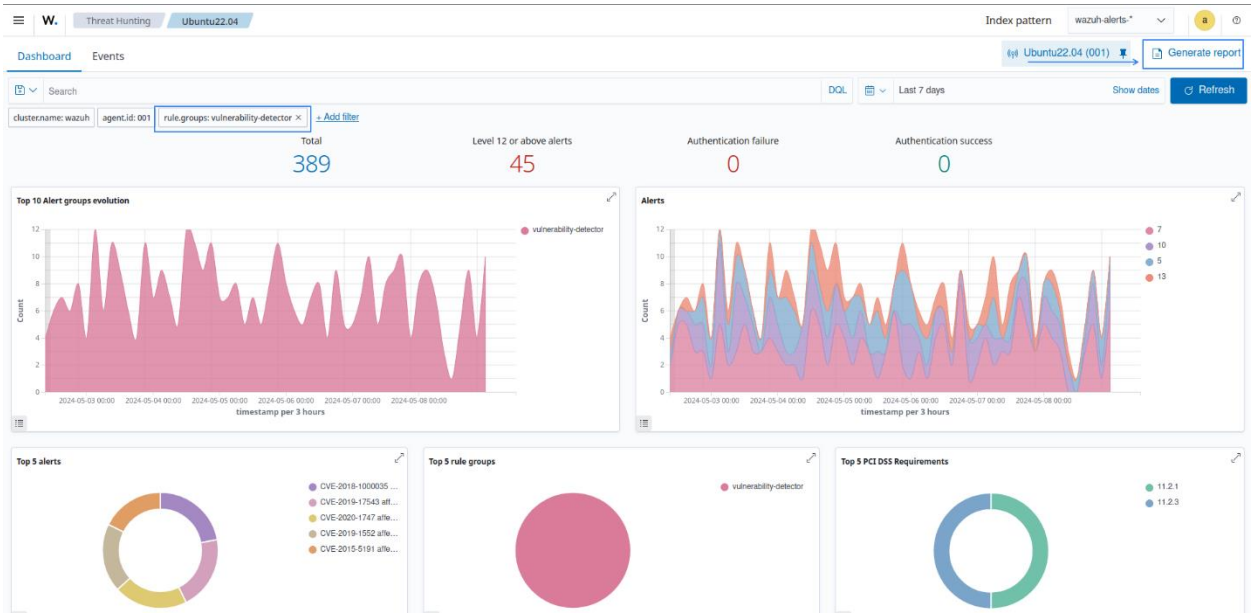
Track vulnerability remediationPermalink to this headline

The Wazuh Vulnerability Detection module also allows you to confirm when a vulnerability has been remediated. This feature detects when a patch or software upgrade resolves a previously detected vulnerability. The feature is enabled using the hotfixes option and is available for Windows endpoints.



Use vulnerability reports to identify critical security issuesPermalink to this headline

Wazuh provides users with the ability to download a report that contains security events related to discovered and resolved vulnerabilities. This feature allows users to identify endpoints with unresolved vulnerabilities and keep track of remediation activities.



Incident responsePermalink to this headline

A security incident refers to any adverse event or activity that risks or threatens the confidentiality, integrity, or availability of digital assets, networks, data, or resources. Such incidents include unauthorized access, data breaches, malware infections, denial-of-service attacks, and any other activities that compromise the security posture of an organization's information technology environment.

The goal of incident response is to effectively handle a security incident and restore normal business operations as quickly as possible. As organizations' digital assets continuously grow, managing incidents manually becomes increasingly challenging, hence the need for automation.

Automated incident response involves automatic actions taken when responding to security incidents. These actions can include isolating compromised endpoints, blocking malicious IP addresses, quarantining infected devices, or disabling compromised user accounts. By

1prevent or minimize the impact of incidents, and efficiently handle a large volume of security events.

Wazuh Active Response modulePermalink to this headline

The Wazuh Active Response module allows users to run automated actions when incidents are detected on endpoints. This improves an organization's incident response processes, enabling security teams to take immediate and automated actions to counter detected threats.

You can also configure the actions to be either stateless or stateful. Stateless active responses are one-time actions while stateful responses revert their actions after some time.

Default active response actionsPermalink to this headline

Out-of-the-box scripts are available on every operating system that runs the Wazuh agents. Some of the default active response scripts include:

Name of script	Description
disable-account	Disables a user account
firewall-drop	Adds an IP address to the iptables deny list.
firewalld-drop	Adds an IP address to the firewalld drop list.
restart.sh	Restarts the Wazuh agent or server.
netsh.exe	Blocks an IP address using netsh.

Custom active response actionsPermalink to this headline

One of the benefits of the Wazuh Active Response module is its adaptability. Wazuh allows security teams to create custom active response actions in any programming language, tailoring them to their specific needs. This ensures that when a threat is detected, the response can be customized to align with the organization's requirements.

Automating incident response with WazuhPermalink to this headline

To leverage the Wazuh Active Response module, you need to configure the action to be carried out when a specific event occurs on a monitored endpoint. For example, you can configure the Wazuh Active Response module to delete a malicious executable from an infected endpoint. In the examples that follow, we show how the Wazuh Active Response module handles different incidents.

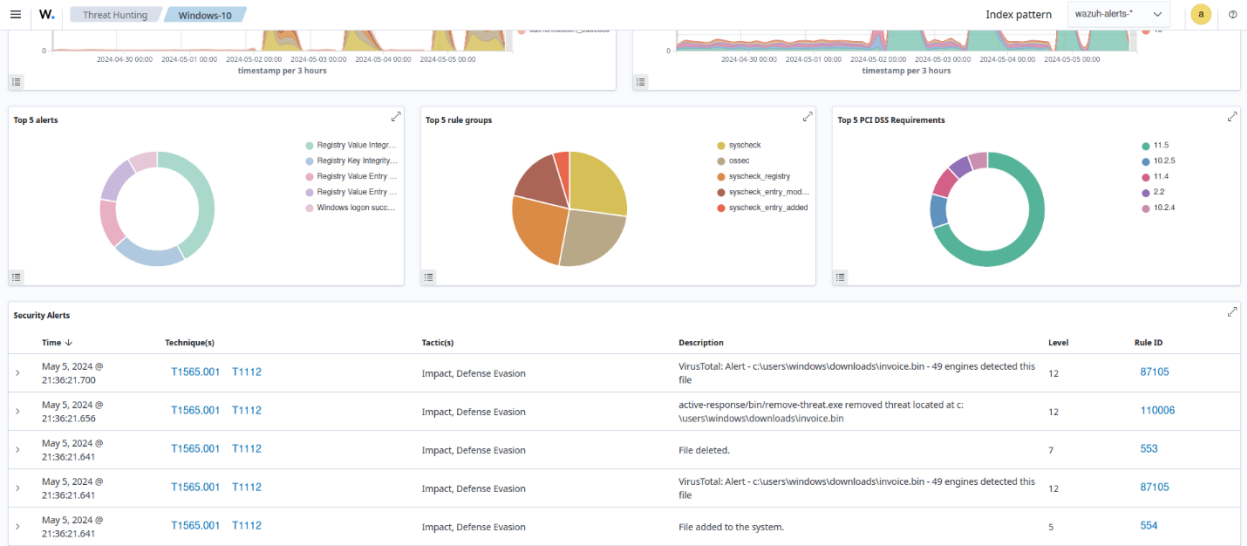
Removing malwarePermalink to this headline

You can use the Wazuh Active Response module in conjunction with the File Integrity Monitoring module and VirusTotal integration to detect and remove malicious files from an endpoint.

The image below shows the following activities:

1. Rule ID 554 is fired when a file is added to the Downloads directory which is monitored with the Wazuh File Integrity Monitoring module.
2. Rule ID 87105 triggers when Wazuh extracts the file hash, requests data about the file hash from the VirusTotal database via its API, and receives a malicious file response.

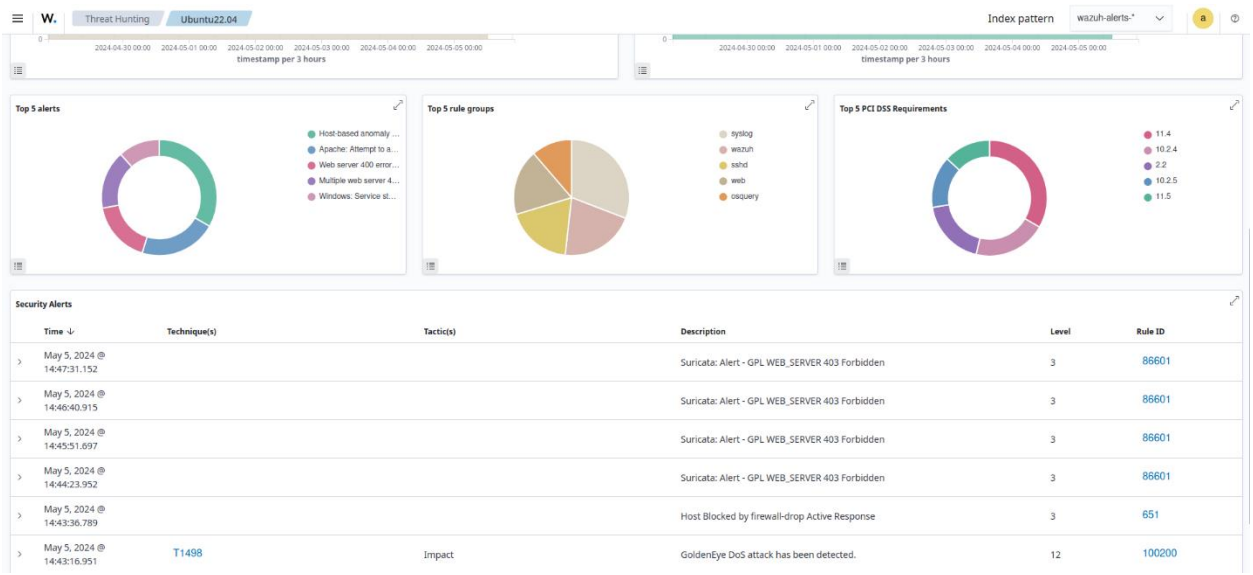
3. Rule ID 553 is fired when a file is deleted from the Downloads directory which is monitored with the Wazuh File Integrity Monitoring module.
4. Rule ID 110006 is fired when the Wazuh Active Response module deletes the malicious file from the endpoint.



In this scenario, the Wazuh Active Response module automatically removes the malicious file, reducing the time between threat detection and mitigation.

Responding to DoS attacksPermalink to this headline

The primary goal of a DoS attack is to render the target inaccessible to legitimate users, causing a denial of service. In the image below, we show how the Wazuh Active Response module blocks malicious IP addresses performing a DoS against a web server on an Ubuntu endpoint.

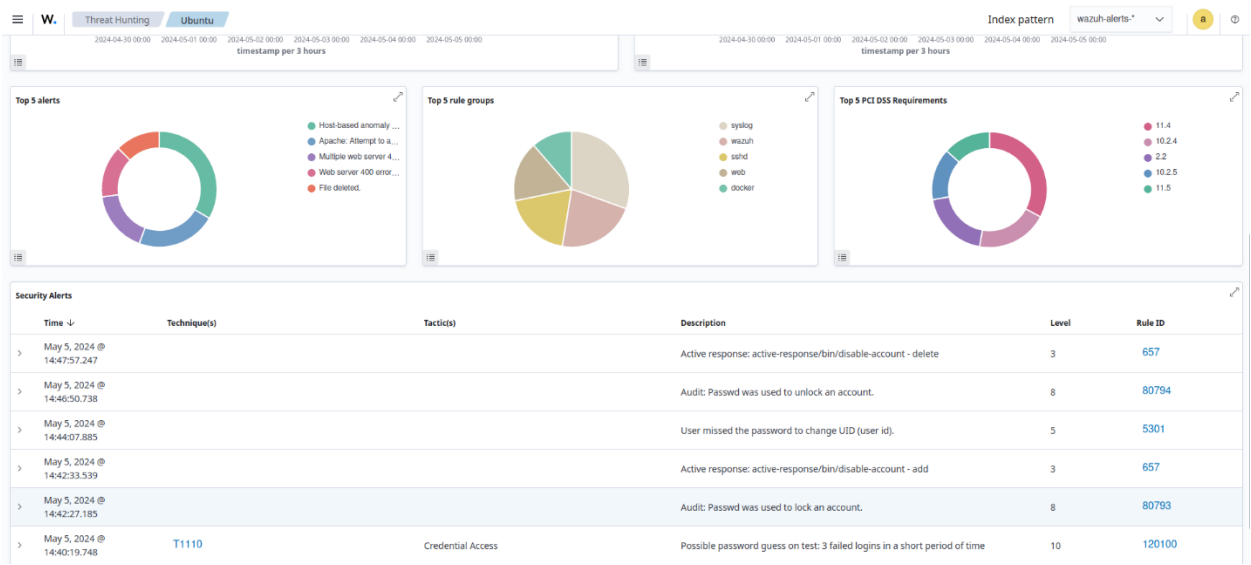


In this case, the Wazuh Active Response module automatically blocks the malicious hosts from causing a DoS attack on the web server. Thereby ensuring the availability of the web server to the authorized users.

Disabling a user account after a brute-force attackPermalink to this headline

Account lockout is a security measure used to defend against brute force attacks by limiting the number of login attempts a user can make within a specified time. We use the Wazuh Active Response module to disable the user account whose password is being guessed by an attacker.

In the image below, the Wazuh Active Response module disables the account on a Linux endpoint and re-enables it again after 5 minutes.



In this scenario, when an attacker tries to guess a user's password repeatedly and fails, the account becomes temporarily inaccessible. This impedes attackers who rely on brute-force methods to guess user account passwords.

By utilizing the Wazuh Active Response module, security teams can automate responses to different incidents. Thereby ensuring efficient incident response and a more resilient cybersecurity posture.