



**Air University**

---

# **Removable Medium Detection Using Wazuh**

*A Practical Security Implementation Report*

---

**Prepared By:**  
Rabya Hassan

May 18, 2026

# Table of Contents

|                                                           |    |
|-----------------------------------------------------------|----|
| 1. Introduction .....                                     | 2  |
| 2. Objectives .....                                       | 3  |
| 3. Background and Technology Overview .....               | 3  |
| 3.1 Wazuh SIEM .....                                      | 3  |
| 3.2 Removable Media as a Threat Vector .....              | 3  |
| 3.3 Windows Event ID 6416 .....                           | 3  |
| 4. Implementation .....                                   | 4  |
| 4.1 Wazuh Server Installation .....                       | 4  |
| 4.2 Retrieving Admin Credentials .....                    | 4  |
| 4.3 Accessing the Dashboard via Browser .....             | 5  |
| 4.4 Installing the Wazuh Agent on Windows .....           | 5  |
| 4.5 Configuring Windows Audit Policy for PnP Events ..... | 6  |
| 4.6 Adding the Custom Detection Rule .....                | 7  |
| 4.7 Restarting the Wazuh Manager .....                    | 7  |
| 4.8 Verifying Alert Generation .....                      | 7  |
| 5. Results and Analysis .....                             | 9  |
| 6. Security Implications and Recommendations .....        | 9  |
| Device Allowlisting .....                                 | 9  |
| Automated Response .....                                  | 10 |
| Centralized Log Retention .....                           | 10 |
| Integration with Asset Management .....                   | 10 |
| 7. Conclusion .....                                       | 10 |
| 8. References .....                                       | 10 |

## 1. Introduction

USB and removable media devices present one of the most persistent and underestimated threat vectors in modern enterprise environments. Whether through accidental data exfiltration, deliberate insider threats, or malware propagation, unmonitored removable media can silently undermine an organization's security posture.

This report documents the practical implementation of removable medium detection using Wazuh — an open-source, enterprise-grade Security Information and Event Management (SIEM) platform. By leveraging Wazuh's agent-based architecture, custom detection rules, and Windows

event log integration, this project establishes real-time alerting for USB device connection events on monitored endpoints.

The implementation follows a structured workflow covering server installation, agent deployment, policy configuration, custom rule authoring, and alert verification. All steps are documented with supporting screenshots and technical explanation.

## 2. Objectives

- Deploy Wazuh SIEM on an Ubuntu-based server using the official installation script.
- Configure authenticated access to the Wazuh web dashboard.
- Install and register a Wazuh agent on a Windows 11 endpoint.
- Configure Windows Local Security Policy to enable Plug and Play (PnP) audit events.
- Author a custom Wazuh detection rule to trigger alerts on USB device connection (Event ID 6416).
- Restart the Wazuh manager and verify that alerts fire correctly when a USB device is inserted.

## 3. Background and Technology Overview

### 3.1 Wazuh SIEM

Wazuh is a free and open-source security platform that provides unified SIEM and Extended Detection and Response (XDR) capabilities. It collects, parses, and analyzes log data from distributed endpoints through a lightweight agent model. The central Wazuh manager processes all incoming events against a rule set and generates alerts when suspicious or noteworthy activity is detected.

Wazuh supports integration with OpenSearch (Wazuh Indexer) and includes a built-in web interface for visualizing events, configuring agents, and generating reports. Its rule engine is XMLbased, allowing administrators to write fine-grained detection logic matched against decoded event fields.

### 3.2 Removable Media as a Threat Vector

Removable media — including USB flash drives, external hard disks, and optical media — introduce several risks in networked environments: introduction of malware, unauthorized data extraction, and bypassing of network-layer security controls. Monitoring and alerting on device connection events is a fundamental control required by security frameworks such as NIST SP 800-53, ISO 27001, and PCI DSS.

### 3.3 Windows Event ID 6416

Windows generates Event ID 6416 ("A new external device was recognized by the system") under the Microsoft-Windows-Kernel-PnP/Operational log channel whenever a plug-and-play device is connected. This event includes the device's friendly name, instance ID, and service name —

sufficient information to identify a USB mass storage device and form the basis of Wazuh's detection rule.

## 4. Implementation

The implementation was executed across seven sequential phases, each building upon the previous. The following sections document each phase with supporting screenshots and technical commentary.

### 4.1 Wazuh Server Installation

The Wazuh All-in-One installation assistant was used to deploy the Wazuh manager, indexer, and dashboard components on an Ubuntu host. The installation script was downloaded from the official Wazuh package repository and executed with elevated privileges. The `--ignore-check` flag was used to bypass the recommended OS compatibility warning during the lab setup, as the Ubuntu version in use was not listed among the officially recommended distributions but is functionally compatible.

The installer automatically configures all required services, generates TLS certificates, and produces the initial admin credentials upon successful completion. The web interface is exposed on port 443.

```
root@hasnat-VMware-Virtual-Platform: /home/hasnat# curl -sO https://packages.wazuh.com/4.7/wazuh-install.sh
root@hasnat-VMware-Virtual-Platform: /home/hasnat# sudo bash wazuh-install.sh -a
17/05/2026 16:30:30 INFO: Starting Wazuh installation assistant. Wazuh version: 4.7.5
17/05/2026 16:30:50 INFO: Verbose logging redirected to /var/log/wazuh-install.log
17/05/2026 16:30:50 ERROR: The recommended systems are: Red Hat Enterprise Linux 7, 8, 9; CentOS 7, 8; Amazon Linux 2; Ubuntu 16.04, 18
.04, 20.04, 22.04. The current system does not match this list. Use -i|--ignore-check to skip this check.
root@hasnat-VMware-Virtual-Platform: /home/hasnat# ^C
root@hasnat-VMware-Virtual-Platform: /home/hasnat# sudo bash wazuh-install.sh -a --ignore-check
17/05/2026 16:32:54 INFO: Starting Wazuh installation assistant. Wazuh version: 4.7.5
17/05/2026 16:32:54 INFO: Verbose logging redirected to /var/log/wazuh-install.log
17/05/2026 16:33:02 INFO: --- Dependencies ---
17/05/2026 16:33:02 INFO: Installing gawk.
17/05/2026 16:33:09 WARNING: Hardware and system checks ignored.
17/05/2026 16:33:09 INFO: Wazuh web interface port will be 443.
17/05/2026 16:33:15 INFO: --- Dependencies ---
17/05/2026 16:33:15 INFO: Installing apt-transport-https.
17/05/2026 16:33:36 INFO: Wazuh repository added.
17/05/2026 16:33:36 INFO: --- Configuration files ---
17/05/2026 16:33:36 INFO: Generating configuration files.
17/05/2026 16:33:38 INFO: Created wazuh-install-files.tar. It contains the Wazuh cluster key, certificates, and passwords necessary for
installation.
17/05/2026 16:33:38 INFO: --- Wazuh indexer ---
17/05/2026 16:33:38 INFO: Starting Wazuh indexer installation.
17/05/2026 16:49:11 INFO: Wazuh indexer installation finished.
17/05/2026 16:49:11 INFO: Wazuh indexer post-install configuration finished
```

Figure 1: Wazuh installation via the official bash install script on Ubuntu

### 4.2 Retrieving Admin Credentials

Upon completion of the installation, the Wazuh dashboard credentials were displayed in the terminal output. The auto-generated admin password is unique to each deployment and must be retained for initial login. These credentials provide full administrative access to the Wazuh web interface and should be changed or securely stored after the initial login.

```
17/05/2026 17:03:32 INFO: You can access the web interface https://<wazuh-dashboard-ip>:443
User: admin
Password: t4cKqic4CHE9fibtzkXQL.+47*WRShWN
17/05/2026 17:03:32 INFO: --- Dependencies ---
17/05/2026 17:03:32 INFO: Removing gawk.
17/05/2026 17:03:43 INFO: Installation finished.
```

Figure 2: Auto-generated admin credentials displayed at end of installation

### 4.3 Accessing the Dashboard via Browser

With the server running, the Wazuh web interface was accessed by navigating to the server's local IP address (<https://192.168.19.128>) in a browser. The login page confirms successful installation and dashboard availability. The admin username and the auto-generated password were used to authenticate.

The browser displays a "Not secure" warning due to the use of a self-signed TLS certificate, which is expected in a local lab environment. In production deployments, certificates from a trusted Certificate Authority should be used.



Figure 3: Wazuh dashboard login page accessed via HTTPS on the local network

### 4.4 Installing the Wazuh Agent on Windows

A Wazuh agent (v4.7.5) was deployed on the Windows 11 endpoint using a PowerShell one-liner. The Invoke-WebRequest command downloaded the MSI installer directly from the Wazuh package server, and msixexec silently installed it with the manager IP address, registration server, and agent name specified as parameters. After installation, the Wazuh service was started with the NET START WazuhSvc command.

```
PS C:\WINDOWS\system32> Invoke-WebRequest -Uri https://packages.wazuh.com/4.x/windows/wazuh-agent-4.7.5-1.msi -OutFile $(env:tmp)\wazuh-agent; msixexec.exe /i $(env:tmp)\wazuh-agent /q MAZUH_MANAGER='192.168.19.128' MAZUH_AGENT_NAME='windows-pc' MAZUH_REGISTRATION_SERVER='192.168.19.128'
PS C:\WINDOWS\system32> NET START WazuhSvc
The Wazuh service is starting.
The Wazuh service was started successfully.
```

Figure 4: PowerShell command used to install and start the Wazuh agent on Windows 11

Following agent installation, the Wazuh dashboard confirmed that the agent named "windows-pc" had successfully registered and was in an active state, as shown in the Agents panel. 100% agent coverage was achieved with one active agent on Microsoft Windows 11 Pro.

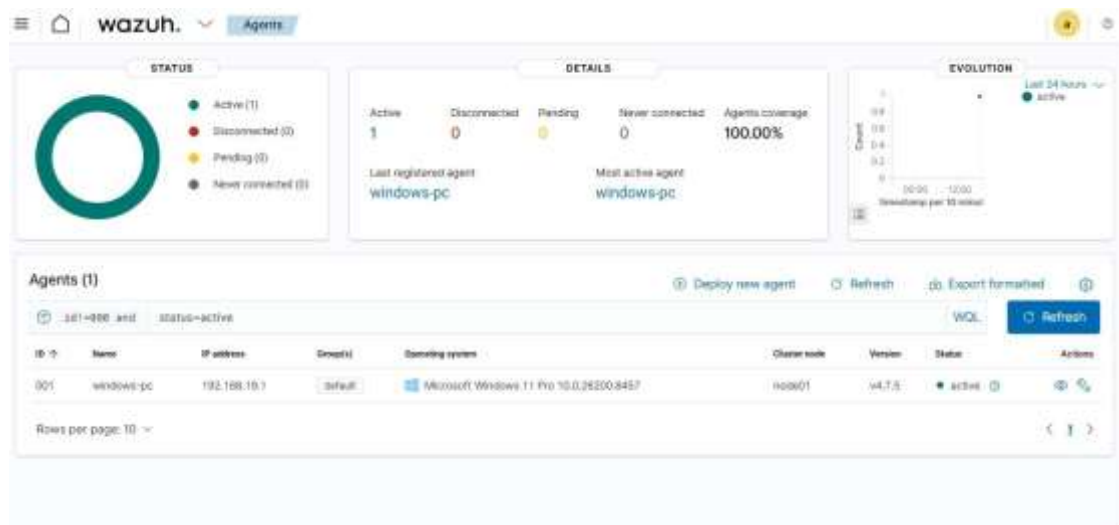


Figure 5: Wazuh Agents panel showing windows-pc as active with 100% coverage

## 4.5 Configuring Windows Audit Policy for PnP Events

By default, Windows does not generate Event ID 6416 unless PnP audit logging is explicitly enabled in the Local Security Policy. This was accomplished by navigating to:

*Local Security Policy → Advanced Audit Policy Configuration → System Audit Policies → Detailed Tracking → Audit PnP Activity*

The policy was set to log Success events. This configuration instructs Windows to write an event to the Security log every time a new plug-and-play device — including USB mass storage — is recognized by the system.

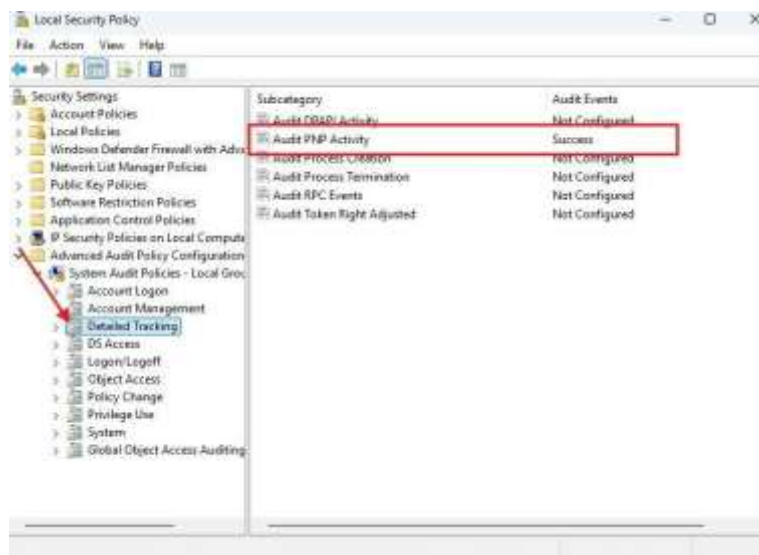
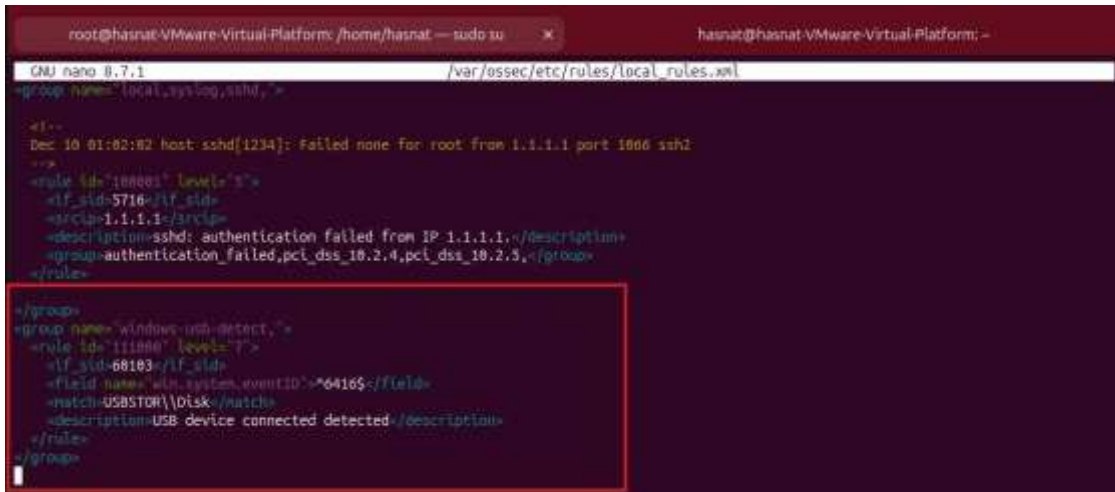


Figure 6: Local Security Policy configured to audit PnP Activity (Success)

## 4.6 Adding the Custom Detection Rule

With the Windows endpoint generating PnP audit events, a custom detection rule was authored in the Wazuh manager's local rules file (`/var/ossec/etc/rules/local_rules.xml`). The rule was structured as follows:

The rule belongs to a group named "windows-usb-detect" and carries rule ID 111000 with severity level 7. It triggers on events that match Wazuh's built-in Windows event decoder (decoder ID 60103), filter specifically for Event ID 6416 using a regex field match, and further constrain the match to events where the device service name contains "USBSTOR\Disk" — confirming it is a USB mass storage device rather than another PnP-compatible peripheral.



```
root@hasnat-VMware-Virtual-Platform: /home/hasnat — sudo su
GNU nano 8.7.1 /var/ossec/etc/rules/local_rules.xml
<group name="local_syslog_sshd">
  <!--
  Dec 10 01:02:02 host: sshd[1234]: Failed none for root from 1.1.1.1 port 1000 ssh2
  -->
  <rule id="100001" level="5">
    <if_sid=5710>/if_sid>
    <src_ip=1.1.1.1>/src_ip>
    <description=sshd: authentication failed from IP 1.1.1.1.</description>
    <group=authentication_failed,pci_dss_10.2.4,pci_dss_10.2.5,</group>
  </rule>
</group>
<group name="windows-usb-detect">
  <rule id="111000" level="7">
    <if_sid=60103>/if_sid>
    <field name="win.system.eventid">"6416"</field>
    <match=USBSTOR\\Disk>/match>
    <description=USB device connected detected.</description>
  </rule>
</group>
```

Figure 7: Custom rule added in `local_rules.xml` — rule ID 111000 detects USB connection via Event ID 6416

## 4.7 Restarting the Wazuh Manager

Custom rules are not loaded until the Wazuh manager service is restarted. The restart was performed using `systemctl` on the Ubuntu server. The command `sudo systemctl restart wazuh-manager` applied the updated rule configuration and brought the detection logic into effect.



```
root@hasnat-VMware-Virtual-Platform: /home/hasnat# sudo nano /var/ossec/etc/rules/local_rules.xml
root@hasnat-VMware-Virtual-Platform: /home/hasnat# sudo systemctl restart wazuh-manager
root@hasnat-VMware-Virtual-Platform: /home/hasnat#
```

Figure 8: Wazuh manager restarted via `systemctl` to apply the new custom rule

## 4.8 Verifying Alert Generation

To confirm end-to-end detection, a SanDisk Cruzer Blade USB flash drive was physically inserted into the monitored Windows 11 machine. The Wazuh dashboard was then used to verify that an alert was generated.

Navigation to the Security Events module was performed through Modules → Security Events, as illustrated below.



Figure 9: Wazuh Modules panel — Security Events selected for alert review

The Security Events dashboard was filtered using rule ID 111000 to isolate the USB detection alert from other events. The filter was applied in the DQL search bar.



Figure 10: Rule ID 111000 entered in the Security Events DQL filter

The alert was confirmed in the event log. The event table showed a single hit with the description "USB device connected detected" for the agent "windows-pc" at the timestamp of the USB insertion. Expanding the event record revealed all relevant field data, including:

- win.system.eventID: 6416
- win.eventdata.FriendlyName: SanDisk Cruzer Blade USB Device
- win.eventdata.Description: USB Mass Storage Device
- win.eventdata.Service: USBSTOR
- rule.id: 111000 / rule.level: 7
- location: WinEvent:Microsoft-Windows-Kernel-PnP/Operational

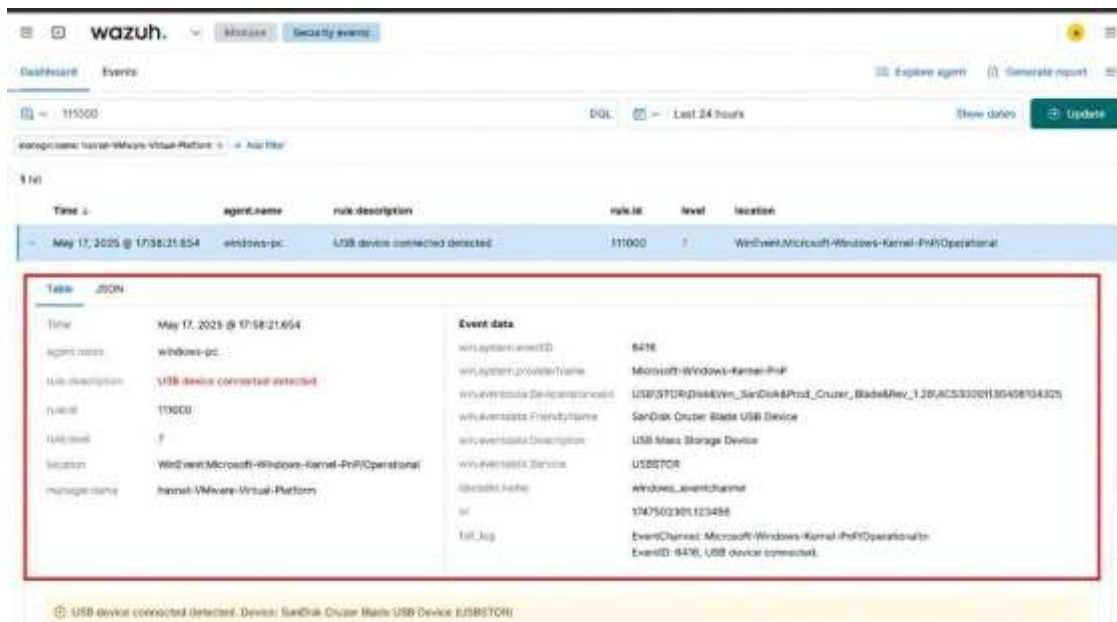


Figure 11: Wazuh alert — USB device connected detected (SanDisk Cruzer Blade, rule ID 111000)

## 5. Results and Analysis

The implementation successfully demonstrated end-to-end USB device detection using the Wazuh SIEM platform. Key outcomes are summarized below:

Detection latency was minimal — the alert appeared in the Security Events dashboard within seconds of physical USB insertion. The custom rule accurately identified the specific device type (USB mass storage via USBSTOR service) while avoiding false positives from other PnP devices such as keyboards or mice.

The event data enrichment provided by Wazuh's Windows event channel decoder proved highly informative. The alert record contained the full device instance ID, friendly name, service type, and originating log channel, giving a responder all the context needed to investigate the event without querying the endpoint directly.

The alert severity was set to level 7 (medium-high), appropriate for a security-relevant but not immediately critical event. In a production environment, this could be tuned — for example, raising the severity for devices not on an approved device list, or correlating multiple events to detect repeated unauthorized access attempts.

## 6. Security Implications and Recommendations

While this implementation demonstrates the detection capability, its value is realized only when integrated into a broader security operations workflow. The following recommendations are offered for organizations looking to operationalize this detection:

### Device Allowlisting

A companion rule set should maintain a list of approved device instance IDs or vendor IDs. Any device not on the allowlist should trigger a higher-severity alert requiring immediate investigation.

## Automated Response

Wazuh's active response module can be configured to automatically execute a script on the agent when rule 111000 fires — for example, disabling the USB port, isolating the endpoint, or notifying a SOC analyst via email or ticket.

## Centralized Log Retention

All Wazuh alerts should be retained in the indexer for a minimum of 90 days to support forensic investigations. Long-term archival to cold storage (S3 or similar) ensures data availability for compliance audits.

## Integration with Asset Management

Cross-referencing the device instance ID and agent name against an asset management database enables immediate identification of the physical machine and its owner, accelerating the incident response process.

## 7. Conclusion

This project demonstrated that Wazuh's open-source SIEM capabilities, when properly configured, provide robust and actionable detection of USB removable media events. The implementation required no third-party agents or commercial licenses — only Wazuh, Windows audit policy, and a carefully crafted XML detection rule.

The seven-step workflow — from server installation through alert confirmation — highlights the accessibility of enterprise-grade security monitoring for educational and organizational use. The framework can be extended to detect other removable media types, enforce USB port controls, and generate compliance reports aligned with standards such as ISO 27001 and NIST SP 80053 (AC-19: Access Control for Mobile Devices).

Air University's Aerospace & Aviation Campus security infrastructure can benefit significantly from adopting such monitoring capabilities, given the sensitivity of aviation-related data and systems. Continuous visibility into physical access vectors is a necessary complement to network-layer defenses.

## 8. References

1. Wazuh, Inc. (2024). Wazuh Documentation — Installation Guide. <https://documentation.wazuh.com/current/installation-guide/>
2. Wazuh, Inc. (2024). Wazuh Agent Deployment for Windows. <https://documentation.wazuh.com/current/installation-guide/wazuh-agent/wazuh-agentpackage-windows.html>
3. Microsoft. (2023). Audit PnP Activity — Windows Security Event 6416. <https://docs.microsoft.com/en-us/windows/security/threat-protection/auditing/event-6416>

4. NIST. (2020). NIST Special Publication 800-53 Rev. 5 — Security and Privacy Controls for Information Systems and Organizations. National Institute of Standards and Technology.
5. ISO/IEC 27001:2022. Information Security Management Systems — Requirements. International Organization for Standardization.
6. Wazuh, Inc. (2024). Custom Rules and Decoders.  
<https://documentation.wazuh.com/current/user-manual/ruleset/custom.html>