Assessment of Wazuh's Threat Detection Capabilities

Hassan Hijazi
hasanridahijazii@gmail.com
June 2025

Abstract

In today's complex digital environments, cybersecurity is a major challenge for organizations. Tools such as Wazuh, CrowdStrike Falcon and Microsoft Defender XDR are used to detect and respond to threats. Wazuh is a cost-effective open-source platform that combines Security Information and Event Management (SIEM) and Endpoint Detection and Response (EDR) functionalities. It is widely used in small to medium enterprises and universities. However, it remains unclear whether Wazuh is reliable for detecting all commonly encountered real-world attacks.

This research aims to evaluate Wazuh's effectiveness in detecting threats generated by tools such as Metasploit. To conduct this evaluation, Kali Linux, Windows 8, and Wazuh virtual machines were set up using Virtualbox. Several tests targeting Windows 8 were conducted, including credential harvesting, lateral movement, reverse shells, and post-exploitation. The analysis showed that while some activities were detected, others were missed or only partially detected. To address these gaps, new detection rules were developed to improve coverage for attacks such as reverse shells and LLMNR poisoning. The results highlight the strengths, limitations, and possible improvements of Wazuh.

1 Introduction

In recent years, cybersecurity has become a major concern for organizations. As digital systems become more complex, the security risks increase.

Today, organizations need to protect many parts of their systems, such as endpoints, cloud services, virtual machines, and more, but it is not possible to manage all of these manually. This is why tools like Security Information and Event Management **(SIEM)** and Endpoint Detection and Response **(EDR)** are now used. These tools play a critical role with continuous monitoring, threat detection, and incident response across endpoints and networks.

Wazuh is an open-source platform that works as both a SIEM and an EDR. It offers a cost-effective alternative to commercial SIEM/EDR platforms, while providing flexibility and transparency that are valuable to organizations aiming to customize their security monitoring.

Wazuh uses predefined XML-based rules to find suspicious actions, such as brute-force attacks, failed login attempts, running dangerous scripts, and unauthorized access to sensitive files. When Wazuh detects something suspicious, it can send alerts and even take action on its own if it's configured to do so.

However, this raises an important question:

> Do these rules align with real-world attack scenarios, and are they capable of detecting well-known and commonly used attack techniques?

No system can guarantee complete security. For that purpose, the main goal of this research is to evaluate how well Wazuh performs in detecting real-world threats and to improve its detection rules if needed. This research aims to benefit Wazuh users and contributes to the developing process of this open source tool.

The remainder of this paper is organized as follows. In Section 2, related work on Wazuh's detection capabilities was reviewed. Section 3 described the methodology, including the architecture of the Wazuh platform and the experimental setup. Section 4 presented the simulated attack scenarios that were executed to test Wazuh's detection rules. Section 5 provided the results and analysis of Wazuh's responses to these attacks. In Section 6, new detection rules that were developed to enhance Wazuh's capabilities were introduced. Finally, Section 7 concluded the paper and discussed possible directions for future work.

2 Related Work

Several studies have evaluated Wazuh's capabilities in detecting cyber threats. In [1], the authors used various attack scenarios, such as brute-force logins, file tampering, and malware downloads, to assess its ability to protect cloud environments from cyberattacks. The results demonstrated that Wazuh is a reliable tool for real-time detection and monitoring and was able to detect different types of attacks in cloud systems.

In another study [2], the authors evaluated Wazuh's ability to detect well-known attacks on web servers. Wazuh success-

fully detected these attacks using its built-in rule set, generating alerts such as "multiple web server 400 error codes from the same source IP". Their study showed that Wazuh is effective at identifying intrusion attempts, particularly in web server environments.

While prior studies have focused on simulating basic suspicious actions, such as repeated login attempts, malware downloads, or system file modifications, this work takes a different approach. Rather than simulating simple actions, full network-based identity attacks were conducted using well-known tools such as **Responder** [3], **Metasploit** [4], **Nmap** [5], and **John the Ripper** [6]. These tools enable attackers to perform actions such as scanning for open services, capturing credentials, and cracking password hashes. This allows for a more comprehensive evaluation of Wazuh's ability to detect and respond to real-world attacks, both at the system level and across the network, using its built-in default rule set.

3 Methodology

3.1 Wazuh Architecture and Detection Rules

Wazuh is a comprehensive open-source security platform that combines SIEM and EDR functionalities. Its architecture consists of three main components: **Wazuh agent**, **Wazuh server (or manager)**, and **Wazuh dashboard**.

Wazuh agent is installed on endpoints such as laptops or servers, where it collects data including system logs, file changes, and other events related to security. This data is transmitted to Wazuh manager, which analyzes it using a set of predefined rules. If suspicious activity is detected on an endpoint, the Wazuh manager triggers an alert (e.g., Figure 6).

Each alert is assigned a corresponding **MITRE ATT&CK** technique. "The MITRE ATT&CK (Adversarial Tactics, Techniques, and Common Knowledge) Framework provides a rich and actionable repository of adversarial tactics, techniques, and procedures" [7]. This helps the analyst understand the possible attack scenario behind the alert and guides an appropriate response to the potential risk. The alerts are stored in a JSON format so that they can be queried quickly.

Wazuh dashboard serves as a web interface that provides an overview of the monitored systems, including alerts and agent statuses, enabling effective management and monitoring.

Wazuh's detection capabilities rely on a set of predefined XML-based rules to detect suspicious activity. These rules define patterns and conditions that trigger alerts when matched by incoming log data. Each rule specifies a rule ID, description, severity level, and conditions such as the source log, frequency, or grouping. These rules can be found in the directory: /var/ossec/etc/rules/

3.2 System Setup

The project setup consists of three main components:

- **Wazuh Environment** – a virtual machine configured as the Wazuh manager to monitor and analyze security data from connected agents.

```
<rule id="140101" level="12">
  <if_group>authentication_success</if_group>
  <user negate="yes">wazuh|root</user>
  <description>Unexpected user successfully logged to the system.</description>
</rule>
```

Figure 1: "This rule triggers when a user different from root or wazuh successfully logs in to the system." [8].

- **Windows 8 Virtual Machine** – has the Wazuh agent installed, enabling communication with the Wazuh manager for monitoring and analysis.
- **Kali Linux Virtual Machine** – includes penetration testing tools such as Metasploit and Nmap, and is used to conduct simulated attacks on the Windows 8 machine to assess and analyze security responses.

The three virtual machines (VMs) are hosted using VirtualBox and configured using a **host-only network**. This setup ensures that all VMs are on the same isolated network.
This isolation is with the purpose of not affecting other systems, and simplifying internal scanning. It facilitates Nmap's ability to detect connected devices and open ports on the Windows 8 VM. It also enables Metasploit to exploit internal vulnerabilities and Responder to apply credential capture attacks while bypassing external firewalls and NAT (Network Address Translation), which usually block unexpected incoming traffic from outside sources.

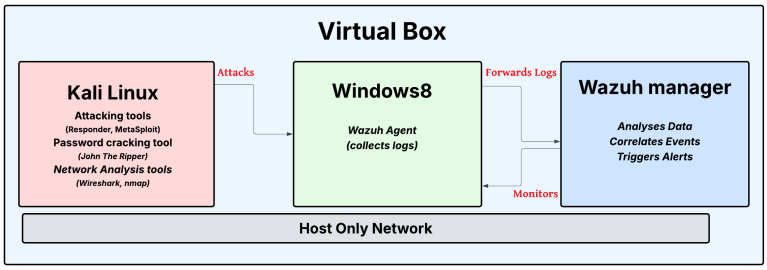

Figure 2: The testbed environment

4 Simulated Attack Scenarios and Testing

This section presents the different attacks and activities used to evaluate Wazuh's detection rule.

4.1 LLMNR Poisoning and NTLMv2 Hash Capture

Sharing files is a common practice in organizations to enable easy access to documents across the network. Users can access shared files by entering a network path in File Explorer, such as `\\admin\interns\hassan.pdf`.

To resolve the hostname (`admin`), the system first uses The Domain Name System protocol (**DNS**). If DNS resolution fails, Windows falls back to protocols like Link-Local Multicast Name Resolution (**LLMNR**) [9], which attempt to

resolve hostnames by sending multicast requests over the local network.

In this scenario, LLMNR sent a request to the local network asking, "Who is `admin`?". The attacker machine, Kali, running the Responder tool, listened for this request. Responder spoofed a reply, pretending to be the `admin` host, and sent back Kali's own IP address.

Assuming the response is trusted, Windows initiated a TCP connection to Kali on port 445, followed by a Server Message Block (**SMB**)[10] negotiation request. Since Kali is acting as an SMB server, it selected an SMB version and specified NT LAN Manager version 2 (**NTLMv2**)[11] as the authentication mechanism. It then sent a random challenge (nonce) to Windows, requesting authentication. Windows responded by computing and sending an **NTLMv2 challenge-response hash**, which is based on the user's password and the challenge. While the actual password is not transmitted to the attacker, this **hash was cracked** using the cracking tool John the Ripper. This Attack is described in Figure 3.

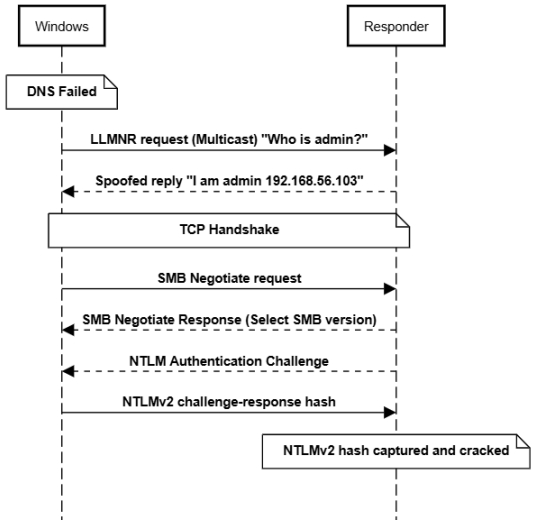

Figure 3: Responder attack network flow

4.2 Remote Login using the Discovered Credentials

After obtaining valid Windows user credentials, a remote login attempt was performed using the `rdesktop` tool, an open-source client for Microsoft Remote Desktop Protocol (RDP), which allows users to connect to Windows systems over the network.

This login succeeded as expected since the credentials were correct. Another attempt was performed using incorrect credentials, which failed as expected.

The purpose of this step was not to exploit the system further, but to evaluate whether such login attempts would trigger any alerts in Wazuh, and whether Wazuh could distinguish between legitimate and potentially suspicious logins, even when performed using valid credentials.

4.3 SMB-based Lateral movement

"Lateral movement is a technique used by cyber attackers use to navigate through a compromised network or system. They move from one host or device to another in search of valuable data, higher privileges, or critical assets. Unlike initial access, which involves breaking into a system, lateral movement is about expanding control within the environment"[12].

In real-world scenarios, lateral movement is preceded by the Credential Access phase.

One common method to achieve Credential Access is through an SMB brute-force attack. Since SMB requires authentication, the attacker systematically tests multiple username/password combinations against the SMB service (TCP port 445) in order to identify valid credentials for the Administrator account.

After performing a network scan using Nmap, it was observed that port 445 (SMB) was open on the target Windows system. The attack then was conducted using Metasploit. The first step involved establishing a TCP connection to the SMB port, followed by systematically attempting password combinations to discover valid credentials.

Once valid credentials were obtained, lateral movement was performed using Metasploit **PsExec** module, which emulates the behavior of the legitimate PsExec tool. This module allows remote command execution over SMB by connecting to the target's SMB service (port 445), uploading a service executable to the ADMIN$ share, creating and starting a remote service, and establishing a channel to execute commands on the target system.

A Meterpreter payload was then uploaded and executed, resulting in a reverse shell. Reverse shell is a technique where the victim's machine initiates a connection to the attacker's machine, allowing the attacker to remotely control the system. This established a Meterpreter session, granting the attacker full remote access to the target system. The attack is illustrated in Figure 4.

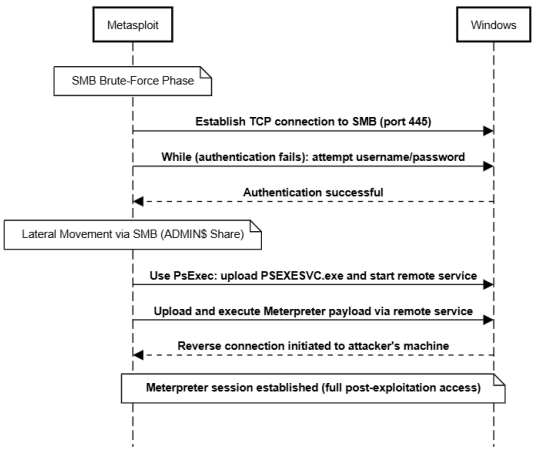

Figure 4: Lateral movement with reverse shell

4.4 Phishing-Based Reverse Shell

In the attack described in Section 4.3, the attack chain was all conducted by the attacker with zero user interaction. There is another common scenario that relies on phishing. The attacker can generate a reverse shell payload and paste it in an executable, then this executable can be dumped in a website or email. The victim can download and execute this file without knowing that it is a malware, and this execution can lead to a reverse shell.

In this scenario, a malicious Windows executable embedding a reverse TCP payload was generated using `msfvenom`, a command-line tool that is part of the Metasploit Framework, used to create custom payloads and encode them into various executable formats.

The executable was downloaded by the Windows user through the web browser and subsequently executed. Upon execution, the payload initiated a reverse TCP connection from the Windows system to the attacker's Kali machine on port 4444.

Simultaneously, Metasploit's `multi/handler` module was configured to listen on port 4444 for incoming connections. As a result, a `meterpreter` session was successfully established, granting the attacker full remote access to the victim machine. The Attack is described in Figure 5.

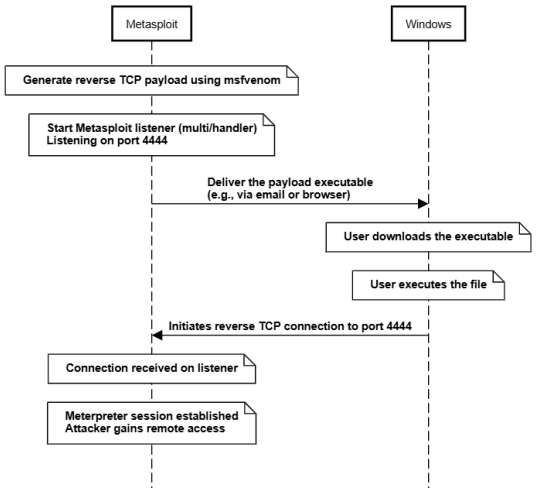

Figure 5: Phishing-based Reverse Shell

4.5 Post-Exploitation

A Meterpreter session typically inherits the privileges of the user who executed the payload.

Two attempts were made, in separate scenarios, to create a new user, delete an existing user, and change a user's group from the Meterpreter session established on the Kali machine.

- **Scenario 1**: The payload was executed by a **standard user**. As expected, an attempt to create a new user and add them to the local users group failed due to insufficient privileges.
- **Scenario 2**: The payload was executed by a **Windows administrator**. This resulted in a successful creation of a new user account with administrative privileges through the Meterpreter session.

5 Results and Analysis

This section presents how Wazuh responded to the attack scenarios described in Section 4 and provides an analysis of the results.

LLMNR Poisoning and NTLMv2 Hash Capture. Wazuh's initial response to the LLMNR poisoning and NTLMv2 hash capture attack described in Section 4.1 was entirely absent; no alerts were generated. This reveals a significant gap in Wazuh's default configuration and built-in detection rules.

However, during the first phase of the improvement process (Section 6), Wazuh began generating an alert when the attack was simulated. This occurred after installing Sysmon (System Monitor) [13] on the Windows system. Sysmon is a Windows system monitoring tool from Microsoft's Sysinternals Suite that provides detailed event logs such as process creation, network connections, and file modifications. This suggests that Wazuh includes built-in detection rules that leverage Sysmon data.

Nevertheless, the alert that was generated was neither accurate nor reliable for this attack scenario. The alert message indicated "Windows System process activity over SMB port — Possible suspicious access to Windows admin shares" and was mapped to the MITRE ATT&CK technique T1021.002 (Lateral Movement)[14].

While "Windows System process activity over SMB port" is technically correct and can be considered part of overall detection, in this case the attacker was not attempting to access ADMIN$ shares nor conducting lateral movement. The actual activity involved only LLMNR poisoning to capture credentials.

An alert for T1021.002 should only be triggered when an attacker intentionally uses valid credentials to initiate an SMB session to another system. An example is the attack described in Section 4.3.

In contrast, the SMB connection observed in this attack was automatically initiated by Windows as a fallback mechanism following LLMNR poisoning, without any lateral movement action by the attacker.

Remote Login Activity.
Wazuh's initial response to the remote login activity described in Section 4.2 was effective. It generated alerts for both successful and unsuccessful login attempts as shown in Figures 6 and 7.

However, a successful remote login alone does not indicate whether the source host is authorized. To enhance security, it is recommended to add an additional layer that raises a higher-severity alert when such logins originate from unauthorized hosts.

agent.n...	rule...	rule.mitre.id	rule.description
windows8	92653	T1021.001 T1078.002	User: WORKGROUP\hacker logged using Remote Desktop Connection (RDP) from ip:192.168.56.103.

Figure 6: Successful remote login

agent.name	rule.id	rule.mitre.id	rule.description
windows8	60122	T1531	Logon Failure - Unknown user or bad password

Figure 7: Failed remote login

SMB-based Lateral movement

Wazuh's response to the SMB-based Lateral movement described in Section 4.3 was highly effective.

As illustrated in Figure 8, Wazuh detected and generated alerts for each individual failed login attempt, as well as for the overall brute-force attack. In addition, an alert indicating possible lateral movement was triggered, suggesting that an attacker might be attempting to gain access to the system via SMB.

agent.name	rule.mitre....	rule.mitre.tac...	rule.description	rule.level	rule.id
windaubeHuitte	T1021.002	Lateral Movement	Windows System process activity over SMB port - Possible suspicious access to Windows admin shares	3	92106
windaubeHuitte	T1021.002	Lateral Movement	Windows System process activity over SMB port - Possible suspicious access to Windows admin shares	3	92106
windaubeHuitte	T1110	Credential Access	Multiple Windows Logon Failures	10	60204
windaubeHuitte	T1531	Impact	Logon Failure - Unknown user or bad password	5	60122
windaubeHuitte	T1531	Impact	Logon Failure - Unknown user or bad password	5	60122
windaubeHuitte	T1531	Impact	Logon Failure - Unknown user or bad password	5	60122

Figure 8: Brute-Force detection

For the lateral movement performed using PsExec, the detection was good enough for experts to understand the nature of the attack, but it was not complete.

As illustrated in Figure 9, the alerts "Successful Remote Logon Detected - User: Administrator" and "New Windows Service Created" are clear indicators of lateral movement via PsExec. However, no alert was generated for the reverse shell connection.

agent.name	rule.mitre.id	rule.mitre.tactic	rule.description	rule.level	rule.id
windaubeHuitte	T1059.001	Execution	Powershell process spawned powershell instance	4	92027
windaubeHuitte	T1070.004	Defense Evasion	Powershell was used to delete files or directories	3	92021
windaubeHuitte	T1059.003	Execution	Windows command prompt started by an abnormal process	4	92052
windaubeHuitte	T1543.003	Persistence, Privilege Escalation	New Windows Service Created	5	61138
windaubeHuitte	T1550.002 T1	Defense Evasion, Lateral Movement, P...	Successful Remote Logon Detected - User:\Administrator - ..	6	92657

Figure 9: Lateral movement

Phishing-Based Reverse Shell

Wazuh's initial response to the phishing-based Reverse Shell, as described in Section 4.4 was partially effective. Once the executable was downloaded from the browser, the following alert was raised as shown in Figure 10.

agent.name	rule.mitre....	rule.mitre.tactic	rule.description	rule.level	rule.id
windaubeHuitte	T1105	Command and Control	Executable file dropped in folder commonly used by malware	15	92213

Figure 10: Executable downloaded

However, the second phase of the attack, which involves executing the file and the outbound connection triggered by the payload, was not detected at all. This indicates a gap in detection that should be improved.

Post-Exploitation Actions.

Wazuh's initial response to the post-exploitation activities described in Section 4.5 was effective in both scenarios.

The alerts triggered by actions performed with standard user privileges are presented in Figure 11. The alerts triggered by actions performed with administrator privileges are presented in Figure 12.

agent.name	rule.mitre.id	rule.mitre.ta...	rule.description	rule.level	rule.id
windaubeHuitte	T1087	Discovery	Discovery activity executed	3	92031
windaubeHuitte	T1087	Discovery	Discovery activity executed	3	92031
windaubeHuitte	T1087	Discovery	A net.exe account discovery command was initiated	3	92039

Figure 11: As standard user

agent.name	rule.mitr...	rule.mitre.tactic	rule.description	rule.level	rule.id
windaubeHuitte	T1484	Defense Evasion, Privilege Escalation	Domain Users Group Changed	5	60160
windaubeHuitte	T1098 T1531	Persistence, Impact	User account disabled or deleted	8	60111
windaubeHuitte	T1484	Defense Evasion, Privilege Escalation	Administrators Group Changed	12	60154

Figure 12: With administrator privileges

6 Contributions: Enhanced Detection Rules

To begin addressing the gaps, Sysmon (System Monitor) was installed on Windows. Its logs were configured to be collected by the Wazuh agent and forwarded to the Wazuh manager.

The strategy followed was to detect each individual activity first, and then correlate them into a new XML-based detection rule that shows the possibility of a specific risk. While the detection rules were implemented in XML within the `local_rules.xml` file (`/var/ossec/etc/rules/local_rules.xml`), the mentioned pseudo-code examples illustrate the core detection logic in a simplified format for clarity.

LLMNR Poisoning Detection

For the first attack (Section 4.1), the detection was based on three steps. The first step identifies LLMNR requests, observed via Sysmon Event ID 3 on UDP port 5355. To reduce noise, an alert with the description "LLMNR request (UDP port 5355)" is generated only during frequent LLMNR activity.

The second step identifies SMB connection attempt from the victim to the attacker. The built-in rule that was previously triggered was used with adaptations to improve accuracy. The rule was enhanced to include the target IP address and mapped to the MITRE ATT&CK technique T1557.001 (LLMNR/NBT-NS Poisoning and SMB Relay).

Finally, a correlation rule was introduced to trigger a "Possible Man-in-the-Middle attack" alert when both the LLMNR request and the SMB connection occur within 10 seconds.

The detection logic for LLMNR poisoning is shown below:

```
# Rule 1: LLMNR Detection
if ( sysmon_event_id == 3 && destinationPort
    == 5355 &&  frequency >= 5 within 1
    second )
    return "LLMNR request";

# Rule 2: SMB Connection Detection
if ( sysmon_event_id == 3 && destinationPort
    == 445 && frequency >= 1 within 1 second
    && image == "^System$" )
    return "SMB connection with <ip_attacker
        >";

# Correlation Rule for MITM Attack
if ( LLMNR_request &&
    SMB_connection_with_ip_attacker &&
    time_between_events <= 10 seconds )
    return "Possible Man-in-the-Middle attack
        ";
```

After adding these rules, Wazuh successfully generated the following alert (Figure 13) when the attack was simulated.

rule.mitre.id	rule.description	rule.level	rule.id
T1557.001	possible LLMNR-based Man-in-the-Middle attack with credential capture over SMB	15	111128
T1557.001	possible LLMNR-based Man-in-the-Middle attack with credential capture over SMB	15	111128
T1557.001	SMB connection with: fe80:0:0:0:1661:4922:3508:8f0b	10	111125
T1557.001	LLMNR requests (UDP port 5355)	10	111123

Figure 13: LLMNR poisoning detection

Reverse Shell Detection

For the reverse shell established via payload execution (Section 4.4), detection was based on three steps.

The first step identifies an execution of an executable file with any of these extensions (exe|scr |com |msi |bat |cmd |ps1 |hta) that are commonly used to deliver or execute reverse shell payloads. Sysmon Event ID 1 was used to extract the image name (full executable path). The detection logic identifies executions where the image name does not belong to a predefined list of trusted Windows executables (e.g., svchost.exe, winlogon.exe).

The second stage identifies an outbound network connections from suspicious executables. Sysmon Event ID 3 was used to verify that the image corresponds to a non-trusted executable.

Finally, a correlation rule was introduced: when such a process execution is followed, within 20 seconds, by a corresponding outbound connection, a possible reverse shell is identified and an alert is raised. The MITRE ATT&CK techniques assigned to this rule were T1204.002 (User Execution: Malicious File) and T1071 (Application Layer Protocol).

The detection logic for reverse shell detection is shown below:

```
# Suspicious EXE Execution Detection
if ( sysmon_event_id == 1 && image matches
    ".(exe|scr|com|msi|bat|cmd|ps1|hta)" &&
    not in known_safe_processes )
    return "Suspicious executable executed";

# Suspicious Outbound Connection by EXE
if ( sysmon_event_id == 3 && image matches
    ".(exe|scr|com|msi|bat|cmd|ps1|hta)" &&
    not in known safe processes )
    return "Suspicious executable creates
        outbound connection"

# Correlation Rule: Possible Reverse Shell
if ( Suspicious executable executed &&
    Suspicious executable creates connection
    && time_between_events <= 20 seconds )
    return "Possible reverse shell:
        suspicious executable executed, then
        creates outbound connection";
}
```

After adding these rules, Wazuh successfully generated the following alert (Figure 14) when the attack was simulated.

rule.mitre.id	rule.description
T1204.002 T1071	Possible reverse shell: C:\\Users\\admin\\Downloads\\shell.exe launched with outbound connection

Figure 14: Reverse Shell detection

Unauthorized remote login detection

First, a file was created at `/var/ossec/etc/lists/`, containing the IP addresses of trusted hosts that are authorized to perform remote logins to the endpoint.

The detection begins by identifying successful remote desktop logins. This is done by checking for Windows Security Event ID 4624 with a logon type of 10, which corresponds to RemoteInteractive logins (e.g., Remote Desktop Protocol).

Next, the source IP address of the login attempt is checked against the predefined list of trusted remote hosts. If the IP address is not present in the list, the login is considered unauthorized.

If both conditions are satisfied — a remote login and the source host is untrusted — an alert is raised with a high severity level. The MITRE ATT&CK techniques mapped to this rule are T1021.001 (Remote Services: Remote Desktop Protocol) and T1078.002 (Valid Accounts: Domain Accounts).

The detection logic for unauthorized remote login is shown below:

```
# Unauthorized Remote Login Detection
if ( windows_event_id == 4624 &&
     logon_type == 10 &&  # RemoteInteractive
     ip_address not in
         trusted_remote_hosts_list )
    return "Unauthorized remote login from
        external IP";
```

After adding this rule, Wazuh successfully generated the following alert (Figure 15) when a remote login was initiated from an unauthorized host.

rule.mitre.id	rule.mitre.tactic	rule.description
T1021.001 T1078.002	Lateral Movement, Defense Evasion, Persiste..	Unauthorized remote login from ip:192.168.56.103.
T1021.001 T1078.002	Lateral Movement, Defense Evasion, Persiste..	Unauthorized remote login from ip:192.168.56.103.

Figure 15: Unauthorized Remote Login Detection

7 Conclusion

This paper presented real-world network-based attack simulations against a target machine, utilizing offensive tools such as Metasploit, to evaluate the threat detection capabilities of Wazuh.

The experiments showed that certain activities, such as remote login attempts, were successfully detected. Others, such as reverse shell execution, were not detected. Some attack phases, including lateral movement, were only partially detected, while techniques like LLMNR poisoning triggered alerts that were unreliable or imprecise.

To address these gaps, new XML-based detection rules were developed for Wazuh. Custom rules were written to detect malicious payload executions that establish reverse connections, as well as LLMNR poisoning activities leading to credential capture and potential cracking.

It was also observed that many advanced attack techniques, including reverse shells and lateral movement, are primarily network-based and require visibility into endpoint network connections. As Wazuh's default configuration lacks sufficient monitoring data for this, it is strongly recommended to deploy Sysmon alongside the Wazuh agent on all monitored endpoints.

Overall, the results showed that Wazuh's default rules only partially align with real-world attack scenarios, but detection can be significantly improved through targeted rule enhancements.

Future work could focus on refining the proposed detection rules to increase robustness and broaden coverage. Integrating automated response mechanisms, such as terminating suspicious connections upon detection, could further enhance the defensive capabilities of the system.

References

[1] Alde Alanda, H.A Mooduto, Ronal Hadi.
Real-time Defense Against Cyber Threats: Analyzing Wazuh's Effectiveness in Server Monitoring.
ResearchGate

[2] Stefan Stanković, Slavko Gajin, Ranko Petrović.
A Review of Wazuh Tool Capabilities for Detecting Attacks Based on Log Analysis.
https://www.etran.rs/.../068-RTI2.6.pdf

[3] Cynet Security.
LLMNR & NBT-NS Poisoning and Credential Access Using Responder.
https://www.cynet.com/...llmnr-nbt-ns-poisoning-and-credential-access-using-responder

[4] Dimitris.
A step-by-step guide to the Metasploit Framework.
https://www.hackthebox.com/blog/metasploit-tutorial

[5] Michael Buckbee.
How to Use Nmap: Commands and Tutorial Guide.
https://www.varonis.com/blog/nmap-commands

[6] Techtarget Editorial.
How to Use the John the Ripper Password Cracker.
https://www.techtarget.com/...john-the-ripper-password-cracker

[7] C. Dimitriadis, M. Angelopoulou, G. Mantas, E. Bekiaris.
Assessing MITRE ATT&CK Risk Using a Cyber-Security Culture Framework.
ResearchGate

[8] Wazuh Documentation. *Ruleset XML Syntax*.
https://documentation.wazuh.com/.../rules.htmlrules-rule

[9] J. Aboba et al.
Link-Local Multicast Name Resolution (LLMNR).
https://datatracker.ietf.org/doc/html/rfc4795

[10] Microsoft.
Server Message Block (SMB) Protocol.
https://learn.microsoft.com/en-us/windows/win32/fileio/microsoft-smb-protocol-and-cifs-protocol-overview

[11] Microsoft.
NTLM v2 Authentication.
https://learn.microsoft.com/.../microsoft-ntlm

[12] Palo Alto Networks.
What is Lateral Movement?.
https://www.paloaltonetworks.com/cyberpedia/what-is-lateral-movement

[13] Microsoft.
Sysmon - Windows System Monitor.
https://learn.microsoft.com/en-us/sysinternals/downloads/sysmon

[14] MITRE ATT&CK.
Remote Services: SMB/Windows Admin$ Shares (T1021.002).
https://attack.mitre.org/techniques/T1021/002/