

Conversation with Gemini



png2pdf (4)
PDF

IR Environment Setup

The project must demonstrate successful installation and configuration of Wazuh SIEM platform on Parrot OS (VirtualBox) with evidence of basic agent deployment, log collection configuration from Parrot OS, and creation of 3 custom alert rules for security events. Configuration of Wireshark on Parrot OS must be demonstrated with proper capture filters. Setup must include Volatility framework in Parrot OS with proper memory analysis configuration. System logging must be configured to collect both Parrot OS and macOS logs with evidence of successful ingestion into Wazuh dashboard. All configurations must include proper documentation and evidence of functionality.

i don't have screenshot of parrot os, but i did it in parrots utm. use mock data wherever you need to. Create a detailed report.

Analysis
Analysis

- Query successful

Okay, I can create a detailed report based on the provided information and the requirements, incorporating mock data where necessary.

Incident Response Environment Setup Report

Introduction

This report details the setup of an Incident Response (IR) environment using Parrot OS (in UTM), focusing on the Wazuh SIEM platform, Wireshark, and the Volatility framework. The setup includes configuring these tools for log collection, alert generation, network analysis, and memory analysis. While the original setup was on Parrot OS in UTM, this report will assume a similar setup and provide mock data examples where screenshots are unavailable.

1. Wazuh SIEM Platform Setup

1.1 Wazuh Installation and Configuration on Parrot OS (UTM)

- **Mock Data/Steps:**

- Assumed successful installation of Wazuh manager and agent on Parrot OS in UTM.
- Installation steps generally involve adding the Wazuh repository, updating the system, and installing the `wazuh-manager` and `wazuh-agent` packages.
- Example commands: Bash

```
# Add Wazuh repository (Mock)
```

- `echo "deb https://packages.wazuh.com/4.x/apt/stable main" | sudo tee /etc/apt/sources.list.d/wazuh.list`
- `curl -s https://packages.wazuh.com/key/GPG-KEY-WAZUH | sudo apt-key add -`
-
- ```
Update system and install Wazuh manager (Mock)
```
- `sudo apt-get update`
- `sudo apt-get install -y wazuh-manager`
- 
- ```
# Install Wazuh agent (Mock - on Parrot OS)
```
- `sudo apt-get install -y wazuh-agent`
-
- ```
Configure agent to connect to manager (Mock)
```
- `sudo /var/ossec/bin/agent-control -m <Wazuh_Manager_IP> -a <Agent_Name>`
- 

- **Configuration Evidence (Mock):**

- Wazuh manager and agent services are running (check with `systemctl status wazuh-manager` and `systemctl status wazuh-agent`).

- Agent successfully enrolled with the manager (check agent logs in `/var/ossec/logs/ossec.log`).

## 1.2 Log Collection Configuration from Parrot OS

- **Mock Data/Steps:**

- Wazuh agent configured to collect system logs (e.g., `/var/log/auth.log`, `/var/log/syslog`).
- File integrity monitoring (FIM) configured to monitor critical files and directories (e.g., `/etc/passwd`, `/etc/shadow`, `/usr/bin`).
- Example configuration in `/var/ossec/etc/ossec.conf` (Mock): XML

```
<ossec_config>
○ <localfile>
○ <log_format>syslog</log_format>
○ <location>/var/log/auth.log</location>
○ </localfile>
○ <ossec_config>
○ <syscheck>
○ <directories check_all="yes">/etc,/usr/bin</
directories>
○ </syscheck>
○ </ossec_config>
○
```

- **Configuration Evidence (Mock):**

- Wazuh dashboard shows events from `/var/log/auth.log` and FIM alerts for monitored files.

## 1.3 Creation of 3 Custom Alert Rules for Security Events

- **Mock Data/Steps:**

- Custom rules created in `/var/ossec/etc/rules/local_rules.xml` to detect specific security events.
- **Rule 1: Detect Multiple Failed SSH Logins:** XML

```
<group name="syslog,authentication_failed,">
○ <rule id="100001" level="8">
○ <if_sid>5712</if_sid> <same_source_ip />
○ <frequency>5</frequency>
○ <timeframe>60</timeframe>
```

- <description>Multiple SSH failed login attempts from same source IP.</description>
- </rule>
- </group>
- 

- **Rule 2: Detect Changes to Critical System Files: XML**

- <group name="syscheck,">
- <rule id="100002" level="10">
- <category>ossec</category>
- <decoded\_as>syscheck</decoded\_as>
- <info type="File added" />
- <id>100002</id>
- <pci\_dss>11.5</pci\_dss>
- <hipaa>164.312.c.2</hipaa>
- <description>File added to critical system directory</description>
- <location>/etc,/usr/bin,/bin,/sbin,/boot</location>
- </rule>
- </group>
- 

- **Rule 3: Detect Sudo Command Usage XML**

- <group name="local,syslog,">
- <rule id="100003" level="5">
- <decoded\_as>pam</decoded\_as>
- <match>COMMAND=sudo</match>
- <description>User executed sudo command</description>
- </rule>
- </group>
-

- **Configuration Evidence (Mock):**
  - Wazuh dashboard generates alerts when the defined events occur (e.g., multiple failed SSH logins, changes to `/etc/passwd`).

## 2. Wireshark Configuration on Parrot OS (UTM)

- **Mock Data/Steps:**
  - Wireshark installed on Parrot OS (`sudo apt-get install wireshark`).
  - Capture filters configured to capture specific traffic (e.g., SSH traffic on port 22, HTTP traffic on port 80).
  - Example capture filters:
    - `tcp port 22` (for SSH)
    - `tcp port 80 or tcp port 443` (for HTTP/HTTPS)
    - `host <Target_IP>` (to capture traffic to/from a specific host)
- **Configuration Evidence (Mock):**
  - Wireshark captures network traffic based on the configured filters.
  - Example: Wireshark capture showing SSH handshake when a user connects to the Parrot OS.

## 3. Volatility Framework Setup on Parrot OS (UTM)

- **Mock Data/Steps:**
  - Volatility framework installed on Parrot OS (installation from source or using `apt` if available).
  - Dependencies installed (e.g., `python-dev`, `distorm3`).
  - Memory image acquired from a target system (for testing purposes, a mock memory dump file can be used or a live acquisition from a test VM). For example, using `sudo dd if=/dev/mem of=memory.dump bs=1M`.
  - Volatility profile created or obtained for the target system's OS (e.g., Windows 10, Linux kernel version).
- **Configuration Evidence (Mock):**
  - Volatility commands executed successfully on the memory image.
  - Example: Bash

```
Get image info (Mock)
```

- `volatility3 -f memory.dump windows.info`

- `# List processes (Mock)`

- `volatility3 -f memory.dump windows.pslist`

- Output shows information about the memory image (OS version, processes running, etc.).

## 4. System Logging Configuration

### 4.1 Parrot OS Logging Configuration

- **Mock Data/Steps:**
  - Parrot OS system logs configured to be collected by the Wazuh agent. This was covered in section 1.2.
  - Relevant logs include:
    - `/var/log/auth.log` (authentication logs)
    - `/var/log/syslog` (system logs)
    - `/var/log/wtmp` (login records)
- **Configuration Evidence (Mock):**
  - Wazuh dashboard shows events from these log files.

### 4.2 macOS Logging Configuration

- **Mock Data/Steps:**
  - Since direct agent installation on macOS might have complexities, we'll use a log forwarding mechanism (e.g., Syslog) to send macOS logs to the Wazuh manager.
  - macOS Syslog configured to forward logs to the Wazuh manager's IP address and port (e.g., 514).
  - Wazuh manager configured to receive Syslog messages. This involves configuring the `<remote>` block in `/var/ossec/etc/ossec.conf`.
  - Example `/etc/syslog.conf` on macOS (Mock): `*.* @<Wazuh_Manager_IP>:514`
  - 
  - Example `/var/ossec/etc/ossec.conf` on Wazuh manager (Mock):  
XML

```
<ossec_config>
 <remote>
 <connection>syslog</connection>
 <port>514</port>
 <listen>0.0.0.0</listen>
 <allowed-ips><macOS_IP></allowed-ips>
 </remote>
</ossec_config>
```
  -
- **Configuration Evidence (Mock):**

- Wazuh dashboard shows macOS logs (e.g., authentication logs, system events).

## Documentation

All configurations are documented with step-by-step instructions and configuration files. (Since screenshots are unavailable, detailed text descriptions and mock examples are provided above.)

## Conclusion

The Incident Response environment is successfully set up with Wazuh for SIEM, Wireshark for network analysis, and Volatility for memory analysis. Parrot OS and macOS logs are collected and analyzed within the Wazuh dashboard, providing a centralized platform for security monitoring and incident response. The custom alert rules enhance the system's ability to detect specific security threats.

Sources and related content

[png2pdf \(4\)](#)



[PDF](#)

Have Gemini write a document or code that you can edit

Try Canvas