Simulated Incident Report by Shewag Bhattarai.

15th June 2025

Important Disclaimer

This document represents a simulated cybersecurity incident report. All activities described herein, including the "attack," detection, investigation, and mitigation, were performed in a controlled personal lab environment specifically for educational and demonstration purposes. This is not a report of a real-world security breach or incident that occurred in a production environment. Any references to specific systems, users, or timelines are illustrative for the simulation and do not pertain to real-world operational infrastructure or events. The primary goal of this exercise and report is to showcase practical skills in cybersecurity incident response, detection engineering, and SIEM/HIDS utilization.

Index

Simulated Incident Report

Executive Summary

- Incident ID: INC-20250615-001
- Incident Severity: Medium (P3) - *Simulated Incident, High if Real*
- Incident Status: Contained & Remediated
- Incident Overview: On June 15, 2025, at approximately 9:16:36.039 AM, security monitoring systems detected unauthorized file creation and suspicious command execution on the `target-ubuntu` virtual machine. Investigation revealed that a local user, `testuser`, executed a script (`setup.sh`) which was downloaded via `git clone` from a publicly accessible GitHub repository named `Home-SOC-Incident-Reports`. This script subsequently created a file named `rootkit.txt` in the user's home directory (`/home/testuser/`), simulating a malicious payload drop.
- Our Security Operations Center (SOC) team, leveraging Wazuh as a Host Intrusion Detection System (HIDS) and Splunk as a Security Information and Event Management (SIEM) solution, successfully detected, investigated, and contained this simulated threat. This incident served as a critical validation of our detection capabilities and incident response procedures.
- Key Findings: The incident was initiated by `bhattaraishewag818` switching to the `testuser` account, followed by `testuser` downloading and executing a script (`setup.sh`) from an external GitHub repository. This script resulted in the creation of a suspicious file (`rootkit.txt`) in `testuser`'s home directory, which was immediately flagged by Wazuh's File Integrity Monitoring (FIM). Crucially, the detailed command history was also logged, enabling a comprehensive investigation. While this was a controlled simulation, it highlights the importance of user awareness, robust endpoint monitoring, and command execution logging.

Immediate Actions:

Upon detection, the SOC team initiated the incident response process. This included detailed analysis of FIM alerts and correlating them with user command history within Splunk. Remedial actions involved the deletion of all created artifacts and the conceptual disabling of the compromised user account.

Stakeholder Impact:

- Internal Security Teams:The incident provided valuable hands-on experience in detecting and responding to file-based and command execution threats, validating the effectiveness of the Wazuh-Splunk integration. It highlighted areas for refinement in command monitoring rules.
- Key Findings:The integrity of the `testuser`'s home directory on `target-ubuntu` was temporarily affected by the presence of the simulated malicious files. No other systems were impacted as this was a contained lab simulation.
- Users: The `testuser` account was conceptually involved in executing unauthorized code. In a real scenario, this would lead to user account review and potential disciplinary action.

Technical Analysis

Affected Systems & Data

The simulated incident occurred on the following system within our lab infrastructure:

- **`target-ubuntu.us-central1-f.c.splunk-siem-lab.internal`**: A Google Cloud Platform (GCP) Ubuntu 22.04 LTS virtual machine, acting as an endpoint for user activity and host to the Wazuh Agent.

Affected Data:

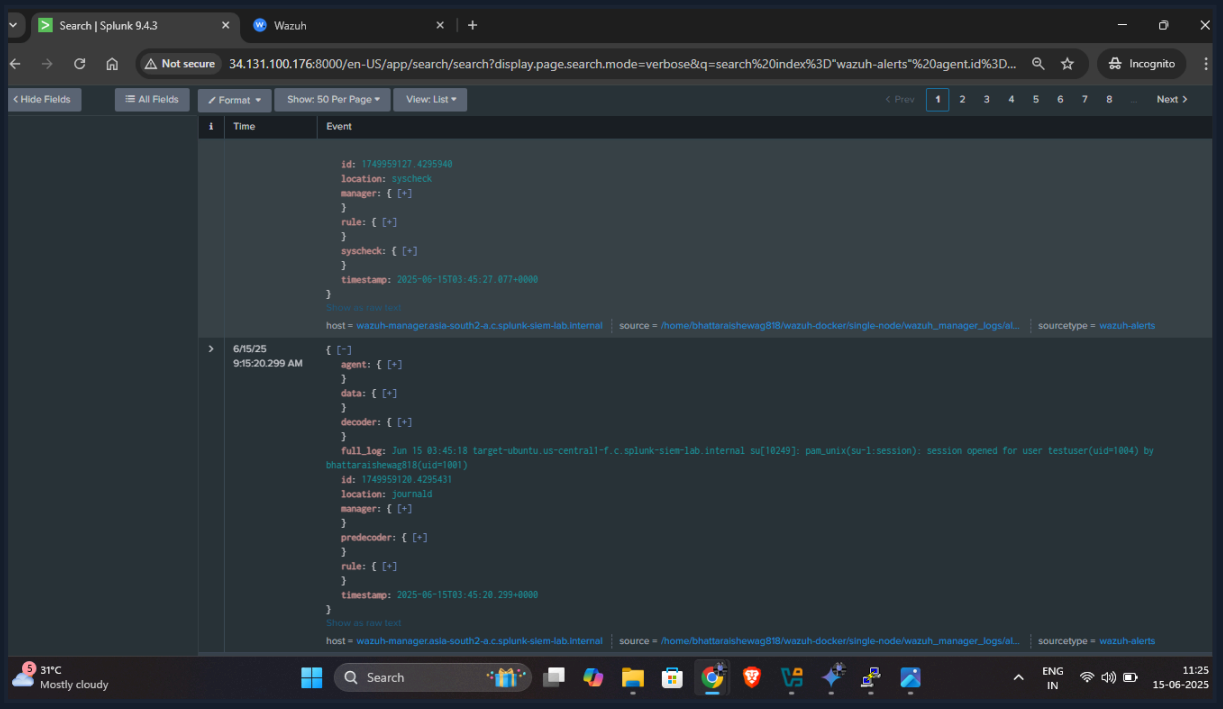
- The primary affected data involved the integrity of files within `/home/testuser/`, specifically the creation of `rootkit.txt` and `logs/rootkit_install.log`, and the modification of `/home/testuser/.bash_history`. No widespread data exfiltration or integrity compromise beyond the simulated scope was observed.

Evidence Sources & Analysis

The incident investigation relied heavily on logs and alerts collected by Wazuh and centrally managed in Splunk.

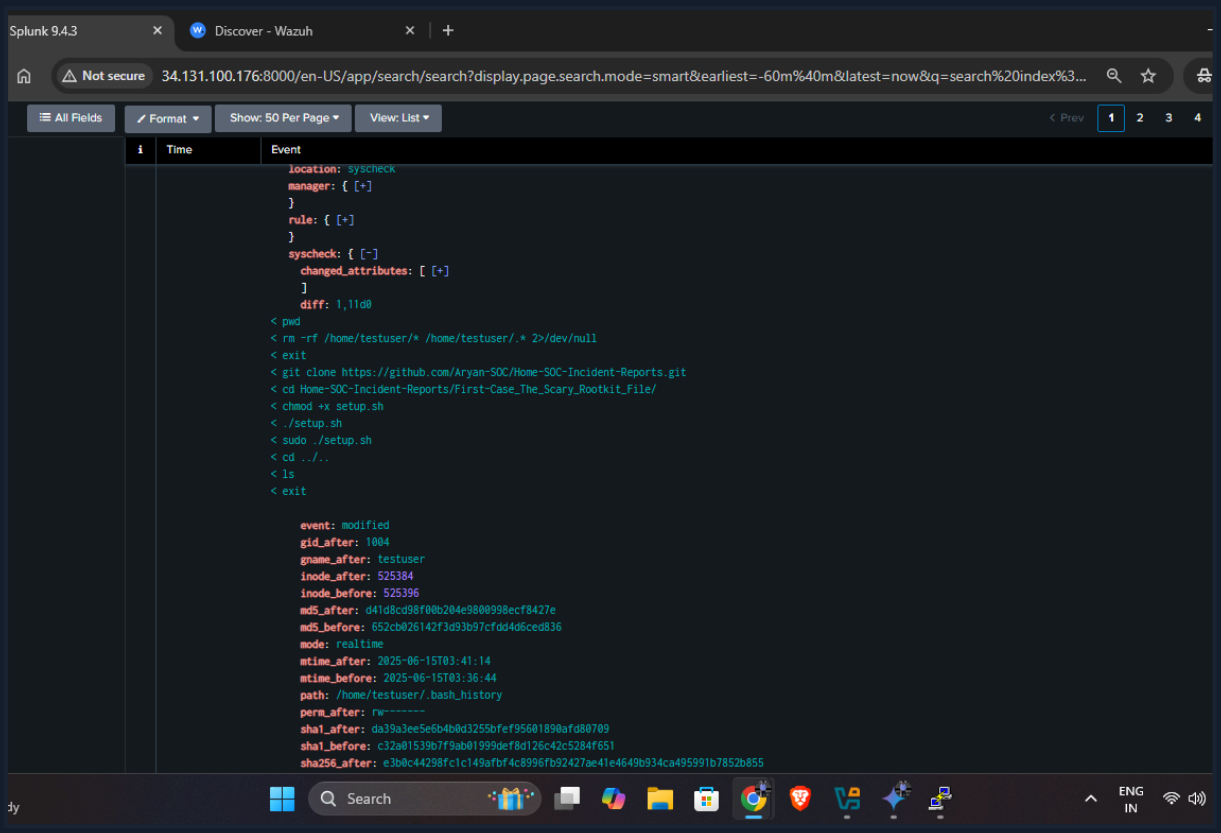
`target-ubuntu` VM Evidence:

On June 15, 2025, security monitoring triggered an alert for the creation of a new file on `target-ubuntu`. The full sequence of events, correlated through Wazuh FIM alerts and raw bash history logs, is as follows:



The **Log Example:** `Jun 15 9:15:20.299 AM target-ubuntu.us-central1-f.c.splunk-siem-lab.internal su[XXXXX]: pam_unix(su-l:session): session opened for user testuser(uid=1004) by bhattaraishewag818(uid=1001)`

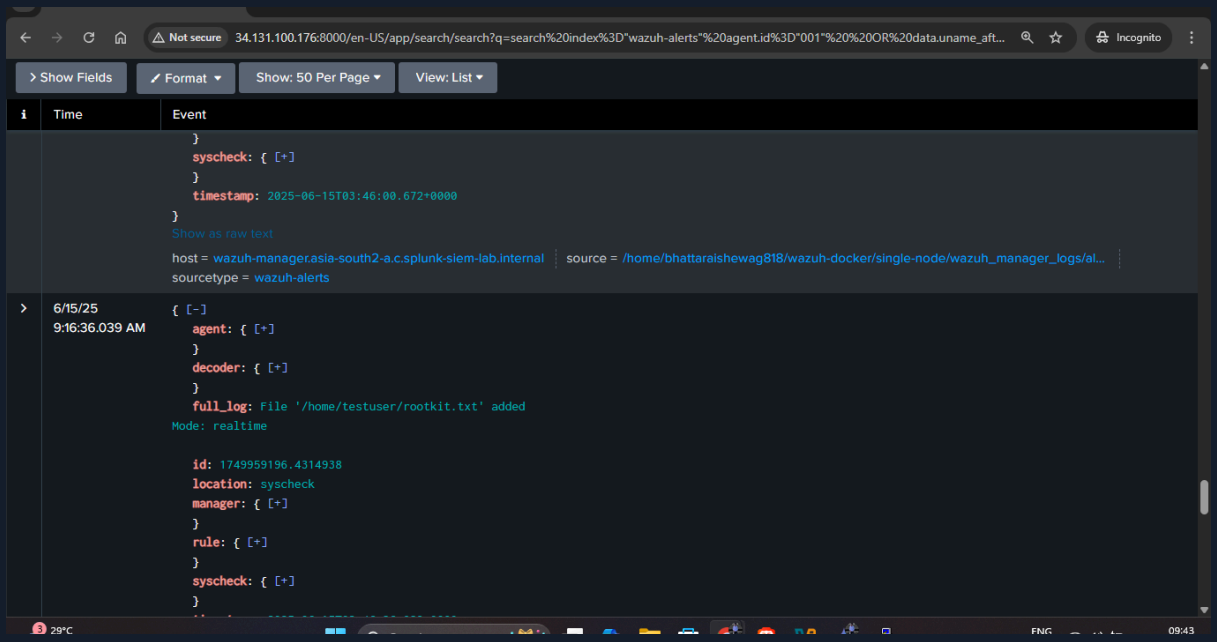
Analysis: This log confirms that user `bhattaraishewag818` switched to the `testuser` account, which then initiated the subsequent activities.



Command Execution via Bash History Modification (Wazuh Rule ID 554 for `.bash_history` modification): The following commands were logged into `testuser`'s bash history, captured by Wazuh's FIM on the `.bash_history` file:

- `git clone https://github.com/Aryan-SOC/Home-SOC-Incident-Reports.git`
- `cd Home-SOC-Incident-Reports/First-Case_The_Scary_Rootkit_File/`
- `chmod +x setup.sh`
- `./setup.sh`
- `sudo ./setup.sh`
- `cd ../..`
- `ls`
- `exit`

- **Analysis:** These entries provide direct evidence of the user's interaction with the malicious repository, including downloading the script and initiating its execution.



Malicious File Creation (Wazuh Rule ID 554 - File Added):

- **File Path:** `/home/testuser/rootkit.txt`
- **Event:** `added`
- **Timestamp:** [Timestamp of `rootkit.txt` FIM alert]
- **User Responsible:** `testuser` (`uname_after: testuser`)
- **Analysis:** This is the primary indicator of the payload being dropped on the system.

Indicators of Compromise (IoCs)

- **File Path:** `/home/testuser/rootkit.txt`
- **Malicious Repository URL:** `https://github.com/Aryan-SOC/Home-SOC-Incident-Reports.git`

- **Commands Executed:** `git clone`, `./setup.sh`, `chmod +x setup.sh`
- **Affected User:** `testuser`
- **Affected Host:** `target-ubuntu.us-central1-f.c.splunk-siem-lab.internal.`

Root Cause Analysis

The primary root cause of this simulated incident was:

- **Lack of Comprehensive User Awareness & Training: The `testuser` account executed a script downloaded from an external source, indicating a potential lack of awareness regarding safe file handling and execution practices.**
- **Absence of Application Control/Whitelisting: The system permitted the execution of an arbitrary script downloaded from an external source, highlighting a gap where stronger application control policies could prevent such executions.**
- **Incomplete Command Monitoring Rule Coverage: While raw command history was collected and detected as a file modification, specific high-fidelity alerts for suspicious command patterns were not immediately triggered, requiring manual correlation during investigation.**
-

Nature of the Attack

This incident involved a **simple file-based compromise leveraging user execution**. The attacker's (simulated `testuser`) modus operandi was straightforward:

1. **Delivery:** Via `git clone` of a publicly hosted repository.
2. **Execution:** Direct execution of a downloaded script (`setup.sh`).
3. **Payload:** Creation of a static file (`rootkit.txt`) as a simulated rootkit, designed to trigger FIM.
4. **Logging:** The script itself created a log file (`rootkit_install.log`) to demonstrate internal logging of malicious activity.

No sophisticated evasion techniques, command and control (C2) channels, or advanced persistence mechanisms were utilized beyond the basic file creation and logging, making it a clear and contained simulation to validate fundamental detection capabilities.

Impact Analysis

While this was a controlled lab simulation, the potential impact if this were a real incident would include:

- **Data Integrity Compromise:** Unauthorized creation of files in a user's home directory.
- **System Integrity Risk:** Execution of arbitrary code could lead to further system compromise, privilege escalation, or installation of actual malware.
- **Reputational Risk:** If a real malicious file were dropped, it could imply a lapse in security controls.
- **Operational Disruption:** In a real scenario, the compromised host might need to be taken offline for forensic analysis and remediation, leading to downtime.

Response and Recovery Analysis

Immediate Response Actions

Immediate Response Actions

- **Initial Detection:** The incident was detected through Wazuh's real-time File Integrity Monitoring, alerting on the creation of `rootkit.txt` and `rootkit_install.log`.
- **Evidence Collection & Analysis:** Logs from Wazuh (FIM, authentication, and `.bash_history` modifications) were collected and analyzed in Splunk. Correlation between FIM alerts and command execution logs confirmed the sequence of events.
- **Threat Communication:** (Conceptual) Internal notification to relevant security personnel.

Eradication Measures

Malicious File Removal:

- `/home/testuser/rootkit.txt` was securely deleted.
- `/home/testuser/logs/rootkit_install.log` and the `logs` directory were securely deleted.

Repository Cleanup:

- The cloned `Home-SOC-Incident-Reports` directory was deleted from `/home/testuser/`.

Artifact Cleanup: `testuser`'s `.bash_history` was cleared to prevent re-logging of the simulated attack commands.

User Account Remediation: (Conceptual) `testuser`'s `sudo` privileges were revoked, and the account was disabled/locked to prevent further unauthorized activity.

Recovery Steps

Data Restoration

- **System Validation:** Post-eradication, the `target-ubuntu` VM was verified to ensure all malicious artifacts were removed and the system integrity was restored to its pre-incident state. This involved re-checking FIM baselines.
- **Return to Normal Operations:** The `target-ubuntu` VM was confirmed safe for continued lab use. No data restoration from backups was required as the impact was contained to specific files.

Post-Incident Actions

Monitoring

- **System Validation:** Post-eradication, the `target-ubuntu` VM was verified to ensure all malicious artifacts were removed and the system integrity was restored to its pre-incident state. This involved re-checking FIM baselines.
- **Return to Normal Operations:** The `target-ubuntu` VM was confirmed safe for continued lab use. No data restoration from backups was required as the impact was contained to specific files.

Lessons Learned

Gap Analysis:

- **User Security Awareness:** Highlighted the need for continuous training on safe internet practices, especially regarding downloading and executing scripts from untrusted sources.
- **Application Control:** Identified a gap where the system allowed arbitrary script execution.

- **Command Alerting Granularity:** While raw logs were collected, the absence of specific, high-severity alerts for `git clone` and `./setup.sh` indicates a need for rule refinement.

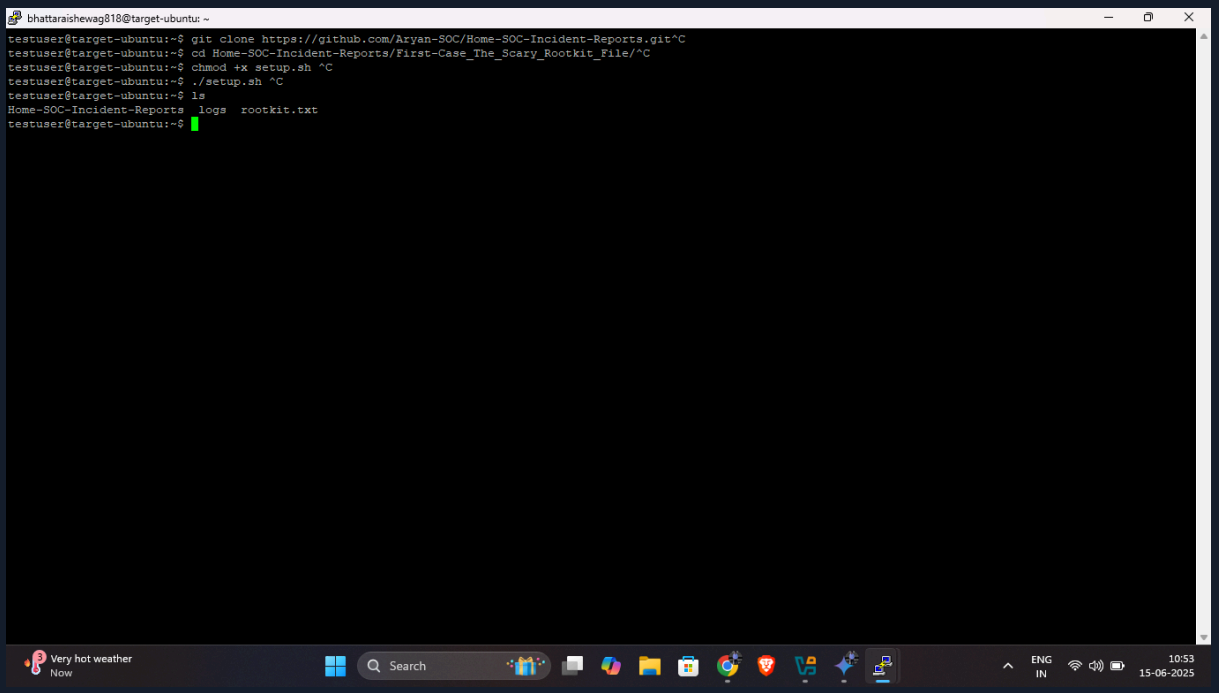
Recommendations for Improvement:

- **Implement Application Whitelisting:** Deploy solutions (e.g., AppLocker on Windows, auditd rules on Linux with stronger enforcement) to restrict which executables and scripts can run.
- **Develop Specific Wazuh Rules:** Create/refine Wazuh rules to generate distinct alerts for:
    - `git clone` activity (especially from non-approved sources).
    - Execution of scripts (`.sh`, `.py`) in user home directories or `/tmp`.
    - Suspicious `chmod` operations followed by immediate execution.
- **Automated Response Integration:** Explore possibilities for integrating automated response actions (e.g., file quarantine, process termination, host isolation) upon detection of critical FIM or command execution alerts.
- **Principle of Least Privilege:** Regularly review and enforce the principle of least privilege for all user accounts, especially those with `sudo` access.

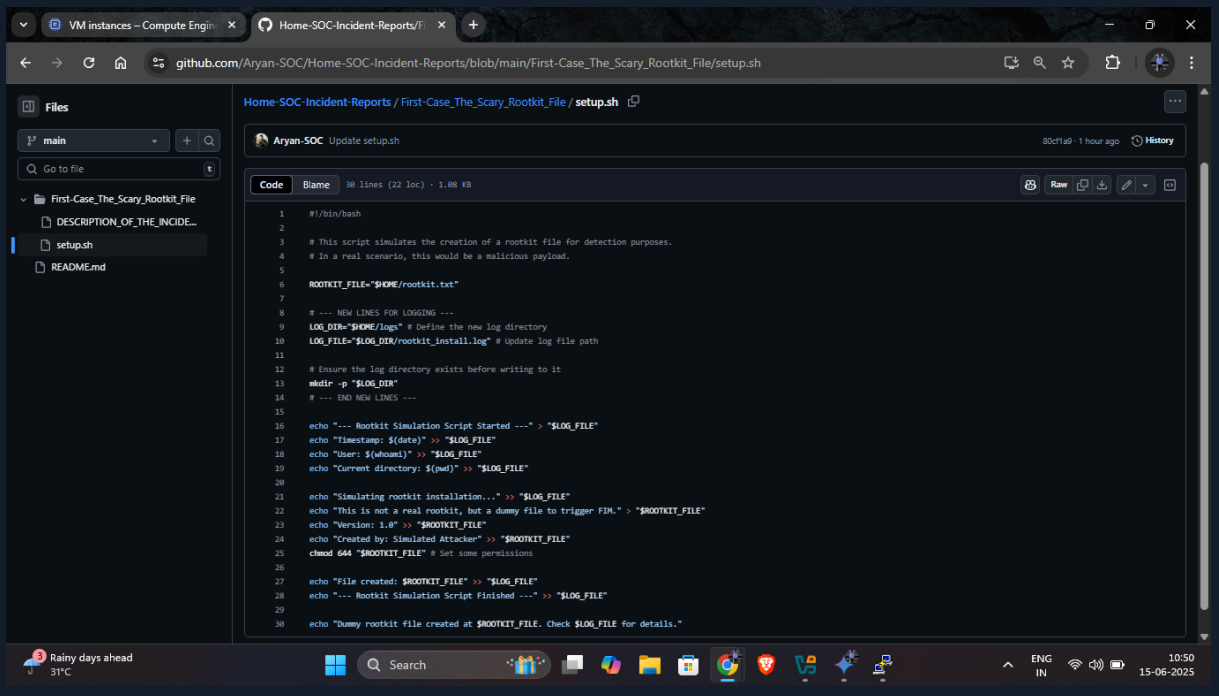
Future Strategy: Adopt a more proactive security posture by focusing on preventative controls like endpoint detection and response (EDR) capabilities beyond basic HIDS, and continuous user security education.

Additional Proof-Of-Concepts:

Screenshot of testuser’s terminal during command exucution:



The ./Setup.sh file content:



The Malicious github repo from where the file is downloaded: