Web Attack Forensics - Drone Alone

Day 15

Explore web attack forensics using Splunk.

- Web Attack Forensics

Web Attack Forensics

Overview

Room URL: https://tryhackme.com/room/webattackforensics-aoc2025-b4t7c1d5f8

Difficulty: Medium

Category: Forensics

Date Completed: 12/15/2025

Objectives

- Detect and analyze malicious web activity through Apache access and error logs
- Investigate OS-level attacker actions using Sysmon data
- Identify and decode suspicious or obfuscated attacker payloads
- Reconstruct the full attack chain using Splunk for Blue Team investigation

---

Table of Contents

Introduction

After deploying the target machine and connecting via the AttackBox, you'll navigate to the Splunk dashboard at `http://MACHINE_IP:8000` using the provided Blue Team credentials (Username: Blue, Password: Pass1234). This challenge guides you through the investigation of a command injection attack targeting a vulnerable CGI script (`hello.bat`), demonstrating how a Blue Teamer uses Splunk's powerful log analysis capabilities to trace an attacker's footsteps from initial web requests through post-exploitation reconnaissance. By examining web access logs, Apache error logs,

Sysmon process creation events, and encoded PowerShell payloads, you'll uncover the complete attack chain and identify the malicious executables the attacker attempted to deploy.

Key Investigation Steps:

- Analyzing HTTP requests for signs of command execution (`cmd.exe`, `PowerShell`, `Invoke-Expression`)
- Decoding Base64-encoded PowerShell commands to reveal attacker intent
- Inspecting Apache error logs for 500 "Internal Server Error" responses indicating backend exploitation
- Tracing parent-child process relationships via Sysmon to confirm successful command injection
- Identifying post-exploitation reconnaissance activity (`whoami` commands)
- Detecting encoded PowerShell payloads that may have evaded initial defenses

---

Walk Through

Step 1

Start the target machine and allow 3 minutes for full boot. Launch Firefox on the AttackBox and navigate to `http://MACHINE_IP:8000`. **Important:** Adjust the Splunk time range (e.g., "Last 7 days" or "All time") to ensure you capture all relevant events. A narrow time range may return "No results found."

Step 2

Execute the following Splunk query to search for HTTP requests containing command execution indicators: `index=windows_apache_access (cmd.exe OR powershell OR "powershell.exe" OR "Invoke-Expression") | table _time host clientip uri_path uri_query status` This query identifies **Command Injection attacks** where the attacker attempts to execute system commands through the vulnerable CGI script. Look for Base64-encoded strings in the results, then decode them using base64decode.org to understand the attacker's payload. **Answer to Question 1:** The reconnaissance executable file name is **`whoami.exe`**

Step 3

Query the Apache error logs for signs of malicious activity and execution failures: `index=windows_apache_error ("cmd.exe" OR "powershell" OR "Internal Server Error")` Select **View: Raw** from the dropdown above the Event display field. Look for 500 "Internal Server Error" responses, which indicate the attacker's input reached the backend and was processed (though it may have failed during execution). This confirms the attack **penetrated the web layer**.

Step 4

Query Sysmon logs to identify child processes spawned by Apache: `index=windows_sysmon ParentImage="*httpd.exe"` Select **View: Table** from the dropdown. Legitimate Apache should only spawn worker threads. If results show child processes like `cmd.exe` or `powershell.exe` with `ParentImage = C:\Apache24\bin\httpd.exe`, this is a **strong indicator of successful command injection**.

Answer to Question 2: The executable the attacker attempted to run through command injection is **`powershell.exe`**

Step 5

Search for evidence of post-exploitation reconnaissance: `index=windows_sysmon *cmd.exe* *whoami*` Attackers typically run `whoami` immediately after gaining code execution to determine which user account their malicious process is running under. Finding these events confirms the attacker's **post-exploitation reconnaissance phase** and proves the injected command executed successfully on the host.

Step 6

Search for encoded PowerShell commands, a common obfuscation technique: `index=windows_sysmon Image="*powershell.exe" (CommandLine="*enc*" OR CommandLine="*-EncodedCommand*" OR CommandLine="*Base64*")` If properly defended, this query should return **no results**, meaning the encoded payload never executed. If results appear, decode the Base64 string to inspect the attacker's true intent.

Lessons Learned

- **Splunk Log Analysis as a Detection Tool:** By chaining multiple Splunk queries across web access logs, error logs, and Sysmon process events, you can reconstruct a complete

attack timeline—from initial exploitation attempts through post-compromise reconnaissance—enabling rapid threat detection and response.

- **Process Lineage and Parent-Child Relationships:** Monitoring parent-child process relationships (particularly identifying system processes spawned by web servers like Apache/`httpd.exe`) is a critical detection method for command injection attacks, and Base64-encoded PowerShell commands should always be decoded and inspected as potential indicators of compromise.

Resources

TryHackMe

Splunk

Cheat Sheet