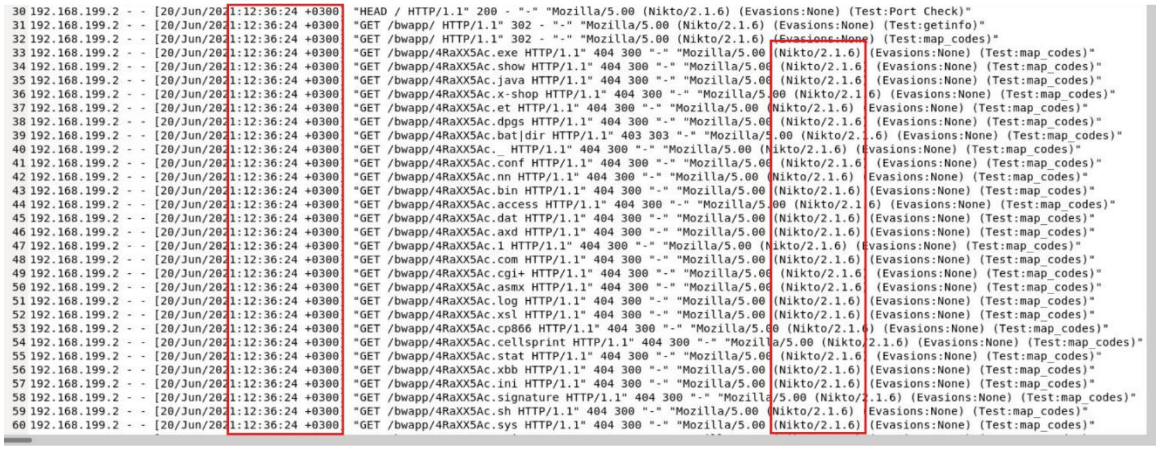
Investigate Web Attack [Let's Defend – Write-up]

We detected some web attacks and need to do deep investigation.
Challenge File: /root/Desktop/ChallengeFile/access.log

| Start investigation

Q1: Which automated scan tool did attacker use for web reconnaissance?

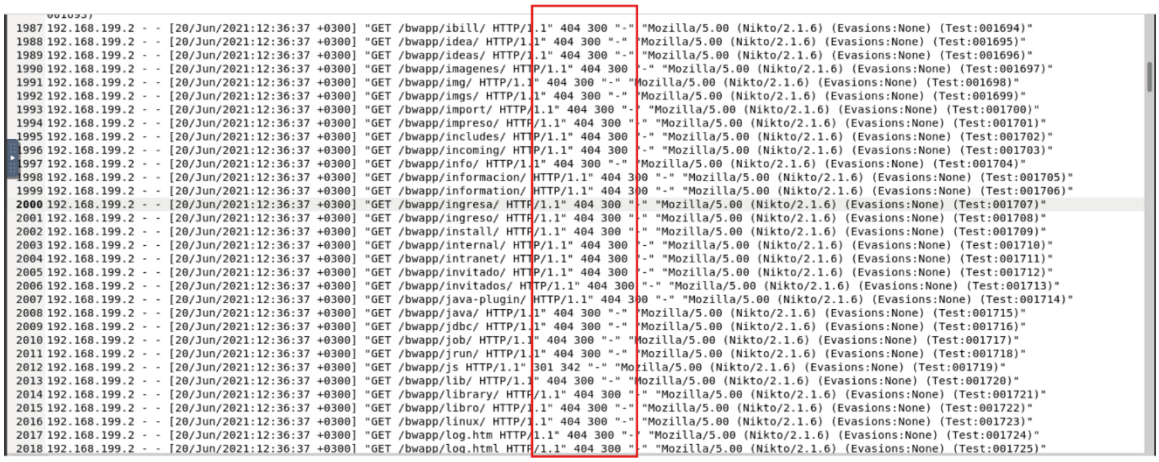


During the initial phase of the attack, the attacker performed an automated scan using Nikto/2.1.6 to identify potential vulnerabilities. The evidence for this is explicitly visible in the User-Agent string of the HTTP request headers. The logs show a rapid burst of requests—all occurring at the same second (12:36:24)—targeting various file extensions such as .exe, .conf, .log, and .ini.

Answer: Nikto**.**

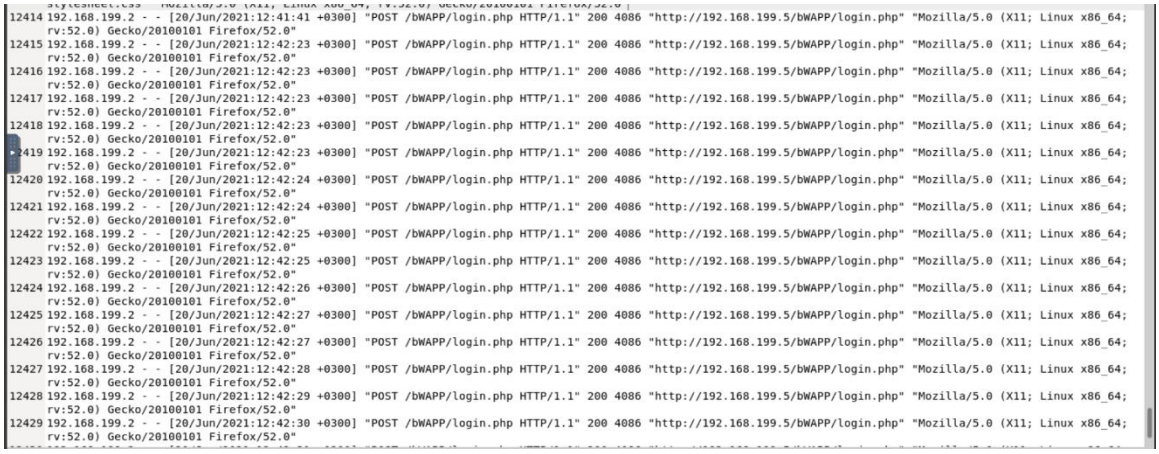
Q2: After web reconnaissance activity, which technique did attacker use for directory listing discovery?

After the initial reconnaissance, the attacker used Directory Brute Forcing to discover the site's structure. This is evident from the high frequency of requests for common or random file paths that returned 404 errors with a uniform size of 300 bytes, showing an automated attempt to guess existing directories.



Answer: Directory brute force.

Q3: What is the third attack type after directory listing discovery?



Following the directory discovery, the attacker initiated a Login Brute Force attack targeting the /bWAPP/login.php endpoint via the POST method. The logs provide clear evidence of this through a rapid succession of requests. Every request resulted in an identical HTTP 200 OK response with a constant size of 4086 bytes, confirming the use of an automated tool to guess credentials.

Answer: brute force.

Q4: Is the third attack successful?

The Login Brute Force attack was successful. Evidence in the logs at 12:50:10 shows a shift from failed attempts (200 OK, 4086 bytes) to an HTTP 302 Redirect, followed by a successful GET request to /bWAPP/portal.php (23369 bytes). This redirect confirms the attacker successfully authenticated and gained access to the user portal.

Answer: yes.

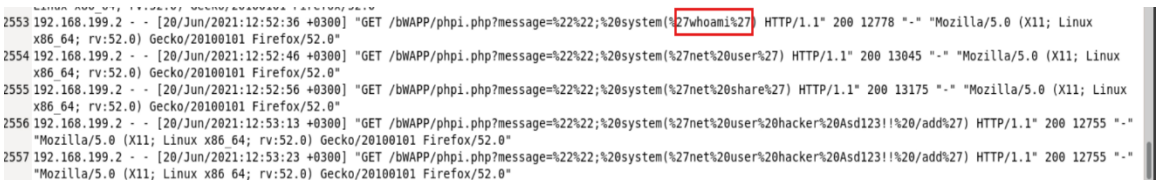
Q5: What is the name of fourth attack?

After gaining unauthorized access, the attacker exploited the /bWAPP/phpi.php page by performing a PHP Code Injection attack through the message parameter. By injecting PHP system() functions into the URL, the attacker executed various OS commands between 12:52:36 and 12:53:23 to perform system enumeration, including whoami, net user, and net share. The attack culminated in a critical impact where the attacker attempted to establish persistence by creating a new administrative account using the command net user hacker Asd123!! /add.

Answer: code injection.

Q6: What is the first payload for 4th attack?

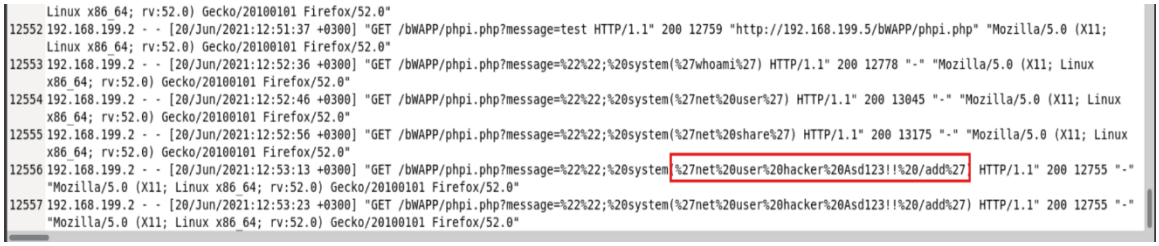
The first command injection payload is this:



Answer: whoami.

Q7: Is there any persistency clue for the victim machine in the log file? If yes, what is the related payload?

The logs confirm a persistency attempt on the victim machine through the creation of a new user account. Using the PHP Code Injection vulnerability at 12:53:13, the attacker executed the payload %27net%20user%20hacker%20Asd123!!%20/add%27. This command leverages the Windows net user utility to add a local user named "hacker" with the password "Asd123!!", ensuring the attacker can maintain long-term, administrative access to the system even after the initial session is terminated.



Answer: %27net%20user%20hacker%20asd123!!%20/add%27

Investigation Summary

The forensic analysis of the access.log file reveals a systematic multi-stage attack starting with **automated reconnaissance** using the **Nikto** scanner to identify server vulnerabilities. Following this, the attacker employed **Directory Brute Forcing** to map the application's structure.

The attack escalated to a successful **Login Brute Force** against the /bWAPP/login.php endpoint, which granted the attacker unauthorized access to the portal. Once authenticated, the attacker exploited a **PHP Code Injection** vulnerability in the phpi.php page to execute system commands. The incident concluded with a high-risk **persistence attempt**, where the attacker used the net user command to create a backdoor administrative account. This sequence demonstrates a complete compromise of the web application, moving from discovery to full system access and persistence.