practice_session_logs/vsftpd_exploit_session.md

Lab Practice Log: Exploiting vsftpd 2.3.4 Backdoor on Metasploitable 2

This log details a specific practice session conducted in my cybersecurity virtual lab, focusing on exploiting a well-known vulnerability to gain remote access.

Date & Time of Session: June 6, 2025, 10:00 AM EDT

Session Objective:
My primary goal for this session was to successfully exploit the vsftpd 2.3.4 backdoor vulnerability present on the Metasploitable 2 target VM. The aim was to gain a remote command shell on the target system using the Metasploit Framework from my Kali Linux attacker machine.

Tools Used:

- Kali Linux VM (Attacker)
- Metasploitable 2 VM (Target)
- Nmap (for network scanning and service identification)
- Metasploit Framework (msfconsole)

Steps Taken During the Session

1. **Reconnaissance: Host Discovery & Port Scanning (from Kali Terminal):**
    - First, I identified the IP address of my Metasploitable 2 target within the CyberLabNet (e.g., 10.0.2.15).
    - I then performed a version-detection scan using Nmap to enumerate open ports and identify running services:
      nmap -sV 10.0.2.15

    - **Observation:** The Nmap scan results prominently displayed vsftpd 2.3.4 running on TCP port 21. This specific version is notoriously vulnerable to a backdoor, making it a prime candidate for exploitation.
2. **Launching Metasploit Framework (from Kali Terminal):**
    - I initiated the Metasploit console to begin selecting an exploit module:
      msfconsole

    - **Observation:** The Metasploit Framework console successfully started, presenting the msf> prompt.
3. **Exploit Search and Selection (within msfconsole):**
    - To find an appropriate exploit, I searched for modules related to vsftpd:
      search vsftpd

- **Observation:** The search results quickly pointed to the exploit/unix/ftp/vsftpd_234_backdoor module, confirming it was available.
- I selected this module for use:
  use exploit/unix/ftp/vsftpd_234_backdoor

4. **Configuring Exploit Options:**
   - I reviewed the module's required options:
     show options

   - The most critical option to set was RHOSTS (the remote target host). I set it to Metasploitable 2's IP:
     set RHOSTS 10.0.2.15

   - A final show options confirmed all parameters were correctly configured.
5. **Executing the Exploit:**
   - With the module loaded and options set, I executed the exploit:
     exploit

   - **Observation:** The exploit ran successfully, indicating a command shell session (session 1 opened) had been established on the Metasploitable 2 target.
   - **Post-Exploitation Verification:** To confirm my access and privilege level, I executed whoami and id commands within the newly opened shell. Both commands verified that I had gained root level privileges on the target system.

Findings and Results

- Successfully identified the vsftpd 2.3.4 service running on Metasploitable 2 via Nmap.
- Leveraged the vsftpd_234_backdoor exploit module within Metasploit Framework to gain unauthorized remote root access to the target VM.
- This exercise provided practical insight into the process of vulnerability identification, exploit execution, and post-exploitation privilege verification.

Screenshots

(In a real GitHub repo, you'd embed or link to actual screenshots here.)

- **screenshots/vsftpd_nmap_scan.png**: Nmap scan output clearly showing the vsftpd 2.3.4 service on port 21.
- **screenshots/vsftpd_exploit_commands.png**: Metasploit console showing the

set RHOSTS command and the exploit command execution.
- **screenshots/vsftpd_root_shell.png**: The successfully obtained command shell on Metasploitable, demonstrating root user output.

Defense and Remediation Insights

Based on this exploitation, here's how I would approach securing a real system:

- **Vulnerability Origin:** The backdoor was intentionally introduced into vsftpd version 2.3.4, making it highly susceptible.
- **Primary Remediation:** The most critical step is to immediately **update vsftpd to a patched version (anything higher than 2.3.4)** or entirely remove the service if it's not essential. Regular software patching is paramount.
- **Network Firewall Rules:** Implement strict firewall rules (e.g., using iptables on Linux servers) to control access to port 21 (FTP).
    - *Example:* Only allow FTP connections from trusted internal IP addresses/networks.
    - *Example iptables rules:*
      # Allow FTP from a specific trusted IP range
      sudo iptables -A INPUT -p tcp --dport 21 -s 192.168.1.0/24 -j ACCEPT
      # Block all other incoming FTP connections
      sudo iptables -A INPUT -p tcp --dport 21 -j DROP

- **Principle of Least Privilege:** Ensure that any services running on a system operate with the absolute minimum necessary privileges.

Key Lessons Learned

- **Patch Management is Vital:** This exploit vividly illustrates the importance of keeping all software and services updated to mitigate known vulnerabilities.
- **Reconnaissance is Foundation:** Effective use of tools like Nmap is crucial for understanding a target's attack surface before attempting any exploitation.
- **Ethical Hacking Practice:** Using frameworks like Metasploit in a controlled lab environment provides invaluable experience in understanding how real-world attacks are executed.
- **Layered Security:** This scenario highlights the importance of multiple security layers, including host-based firewalls, in addition to strong patch management.