Red Team Engagement - Technical Documentation for DEPI Project

Team :

1-**Mahmoud Mosbah (Team Leader)**

**2-Mohamed Magdy**

**3-Ahmed Ali**

**4-Youssef Rabie**

**5-Ayah Elbaz**

**6-Mohamed Mohaseb**

**7-Mahmoud Galal**

# 1. Introduction

Red Teaming is a simulated attack approach that mimics real-world adversaries to evaluate an organization's security posture. It tests detection capabilities, response time, and resilience across technical, human, and process layers. Unlike standard penetration testing, Red Teaming is stealthy, long-term, and mission-oriented.

# 2. Red Team vs Penetration Testing

- Red Team: Goal-driven, stealthy, simulates real attacker behavior, often without prior announcement.  
- Penetration Testing: Limited in scope and time, focuses on identifying known vulnerabilities, often conducted with system owner's knowledge.

# 3. Red Teaming Phases

1. Reconnaissance: Passive and active information gathering (OSINT, Nmap, WhatWeb).  
2. Vulnerability Analysis: Analyze results from scans, identify software versions, exposed services.  
3. Exploitation: Use known CVEs or misconfigurations to gain initial access.  
4. Lateral Movement: Use credentials or exploits to move deeper into the network.  
5. Objective Completion: Exfiltrate data, obtain flags, or simulate impact.  
6. Reporting: Provide technical and executive-level summaries of findings and mitigations.

# 4. Engagement Overview

Target: Legacy Linux server at Company X running a vulnerable CMS.  
Hostname: mrrobot.companyx.internal  
IP Address (identified later): 192.168.1.32  
Environment: WordPress running on Apache with weak configurations.  
Objective: Simulate attacker behavior to access sensitive files such as key-1-of-3.txt.

# 5. Reconnaissance & Scanning

- Initial sweep using Nmap to find live hosts:  
 Command: nmap 192.168.1.32/24  
- Stealth scan with evasion techniques:  
 Command: nmap -sS -Pn -T2 --scan-delay 1s --max-retries 1 -f -D RND:10 --source-port 53 192.168.1.32  
- Discovered open ports: 80 (HTTP), 443 (HTTPS)  
- Web technology fingerprinting using WhatWeb:  
 Command: whatweb 192.168.1.32 -v -a 4  
- Discovered: Apache server, WordPress CMS

# 6. Directory Enumeration

- Visited http://192.168.1.32 manually  
- Used Dirsearch tool to find hidden or protected directories:  
 Command: dirsearch -u http://192.168.1.32  
- Discovered: /wp-login.php, /robots.txt  
- robots.txt exposed:  
 fsocity.dic (wordlist for brute-force)  
 key-1-of-3.txt (first flag)

# 7. Brute Force Attack

- Target: WordPress login page at /wp-login.php  
- Used fsocity.dic as a wordlist to attempt password guessing.  
- Performed login brute force using Burp Suite's Intruder tool.  
- Steps in Burp Suite:  
 1. Captured a login POST request with invalid credentials.  
 2. Sent it to Intruder and configured payload positions for username and password.  
 3. Loaded fsocity.dic into the password payload list.  
 4. Observed responses and filtered using response length and messages.  
- Detected valid username 'Elliot' from message: 'The password you entered for the username Elliot is incorrect.'  
- Discovered valid password via Intruder with status code 302 (redirect on success).

# 8. Reverse Shell

- Uploaded PHP reverse shell using vulnerable plugin or file upload form.  
- Started Netcat listener on attacker's machine:  
 Command: nc -lvnp 4444  
- Triggered shell via HTTP GET request to malicious file.  
- Gained shell access on the target server with web server privileges.

# 9. MD5 Cracking

- Found password hash in file: password.raw-md5  
- Hash: c3fcd3d76192e4007dfb496cca67e13b  
- Used CrackStation online tool to crack the hash.  
- Cracked password: abcdefghijklmnopqrstuvwxyz  
- Switched to user 'robot':  
 Command: su robot

# 10. Privilege Escalation

- Searched for SUID binaries:  
 Command: find / -type f -perm -4000 -ls 2>/dev/null  
- Found SUID-enabled Nmap binary.  
- Launched Nmap interactive shell:  
 Command: nmap --interactive  
 Then typed: !sh  
- Got root shell, used:  
 Command: whoami (confirmed root)  
- Read final flag:  
 Command: cat /root/key-3-of-3.txt

# 11. Covering Tracks

- Clear shell command history:  
 Commands: history -c, unset HISTFILE, rm ~/.bash\_history  
- Remove logs:  
 Files: /var/log/auth.log, /var/log/syslog, /var/log/apache2/access.log  
- Use sed to sanitize logs:  
 Command: sed -i '/attacker\_ip/d' /var/log/auth.log  
- Delete malicious scripts:  
 Command: rm /var/www/html/revshell.php  
- Remove cron jobs:  
 Command: crontab -r  
- Mask IP using VPN or proxychains

# 12. Defense Strategies

- Update WordPress core, plugins, and themes regularly.  
- Enforce strong password policies and hash storage (e.g., bcrypt).  
- Disable directory listing (Options -Indexes).  
- Secure file upload functionality: allow only specific file types, no execute permissions.  
- Audit SUID files, minimize sudo usage.  
- Monitor with Fail2Ban, auditd; enable SELinux or AppArmor.

# 12. Defense Strategies

- Update WordPress core, plugins, and themes regularly.  
- Enforce strong password policies and hash storage (e.g., bcrypt).  
- Disable directory listing (Options -Indexes).  
- Secure file upload functionality: allow only specific file types, no execute permissions.  
- Audit SUID files, minimize sudo usage.  
- Monitor with Fail2Ban, auditd; enable SELinux or AppArmor.