Advanced Network Pen testing

Index

Manual Exploitation of System Vulnerabilities

Manual Exploitation Overview: Manual exploitation involves leveraging human skills and techniques to identify and exploit specific system or network vulnerabilities. Unlike automated tools, manual exploitation is highly customized and tailored to the unique vulnerabilities and environments.

Key Processes:

1. **Vulnerability Identification:**
    - **Code Review:** Analyzing source code to find logic flaws or vulnerabilities like buffer overflows.
    - **Configuration Review:** Examining system settings for potential weaknesses.
    - **Network Traffic Analysis:** Monitoring for anomalies indicating vulnerabilities.
    - **Reverse Engineering:** Disassembling binaries to understand and find weaknesses.
2. **Exploitation Techniques:**
    - **Proof of Concept (PoC):** Creating or adapting exploits to demonstrate vulnerability impacts.
    - **Manual Code Injection:** Crafting payloads to exploit identified weaknesses.
    - **Authentication Bypass:** Circumventing login controls to gain unauthorized access.
    - **Privilege Escalation:** Exploiting misconfigurations to gain higher-level access.
3. **Tools and Utilities:**
    - **Limited Tool Dependence:** Relying on expertise over automated tools.
    - **Scripting Languages:** Using Python or Bash to automate parts of the process.
4. **Post-Exploitation Activities:**
    - **Persistence:** Maintaining access by modifying settings or planting backdoors.
    - **Data Exfiltration:** Covertly extracting sensitive data.
    - **Covering Tracks:** Erasing evidence to avoid detection.
5. **Challenges and Considerations:**
    - **Complexity:** Requires in-depth knowledge and is resource-intensive.
    - **Detection:** Can be identified by monitoring and intrusion detection systems.
    - **Ethical and Legal Concerns:** Must adhere to guidelines and obtain permissions.

6. **Best Practices:**
    - **Skill Development:** Continually improve skills and knowledge.
    - **Documentation:** Keep detailed records of findings and actions.
    - **Collaboration:** Work with stakeholders to address vulnerabilities.

Post-Exploitation

Post-Exploitation Overview: Post-exploitation focuses on maintaining access, extracting information, and leveraging the compromised system to further compromise the network.

Objectives and Techniques:

1. **Establishing Persistence:**
    - **Techniques:** Backdoors, new user accounts, modified startup scripts.
2. **Data Exfiltration:**
    - **Techniques:** Copying files, stealing credentials, capturing screenshots, logging keystrokes.
3. **Privilege Escalation:**
    - **Techniques:** Exploiting misconfigurations or vulnerabilities for deeper access.
4. **Exploring and Exploiting Other Systems:**
    - **Techniques:** Using compromised credentials, exploiting trust relationships.
5. **Covering Tracks:**
    - **Techniques:** Deleting logs, modifying timestamps, altering audit trails.

Tools and Techniques:

- **Remote Access Tools (RATs):** For maintaining control.
- **Command and Control (C2) Frameworks:** For communication and data exchange.
- **Exfiltration Tools:** For secure data extraction.

Mitigation Strategies:

- **Continuous Monitoring:** For prompt detection.
- **Endpoint Detection and Response (EDR):** To monitor and respond to suspicious behavior.
- **Network Segmentation:** To limit lateral movement.

- **Patch Management:** Regularly apply updates.

Legal and Ethical Considerations:

- **Compliance:** Adhere to legal and regulatory requirements.
- **Ethical Guidelines:** Ensure assessments are authorized and consensual.

Privilege Escalation (Linux and Windows)

Overview: Privilege escalation involves gaining higher access levels on a system. This is critical for attackers to deepen their reach and impact.

Linux Techniques:

1. **Exploiting Sudo Misconfigurations:**
    - **Techniques:** Misconfigured sudoers file, environment variable manipulation.
2. **Exploiting Weak File Permissions:**
    - **Techniques:** Overwriting system binaries, editing configuration files.
3. **Kernel Exploitation:**
    - **Techniques:** Local privilege escalation vulnerabilities, vulnerable services.
4. **Exploiting SUID/SGID Programs:**
    - **Techniques:** Running SUID binaries, exploiting race conditions.

Windows Techniques:

1. **Exploiting Weak Service Permissions:**
    - **Techniques:** Service DLL hijacking, service executable replacement.
2. **Exploiting Unquoted Service Paths:**
    - **Techniques:** Placing malicious executables in directories with spaces.
3. **Exploiting Registry Permissions:**
    - **Techniques:** Modifying registry keys, adding new registry keys.
4. **Exploiting DLL Hijacking:**
    - **Techniques:** Placing malicious DLLs in search directories.

Mitigation Strategies:

- **Regular Patching:** To fix known vulnerabilities.
- **Least Privilege Principle:** Limit user and service permissions.

- **Monitoring and Auditing:** Watch for suspicious activities.
- **Security Tools:** Use IDS, EDR, and antivirus solutions.

Pivoting and Double Pivoting

Overview: Pivoting involves using a compromised system to access other network areas. Double pivoting extends this by using multiple compromised systems.

Pivoting Process:

1. **Initial Compromise:** Gain access to a system.
2. **Establishing Foothold:** Install tools for persistent access.
3. **Exploring the Network:** Scan for other systems and services.
4. **Exploiting Additional Systems:** Use vulnerabilities or weak credentials.
5. **Data Exfiltration or Further Compromise:** Achieve attack objectives.

Double Pivoting:

1. **Initial Compromise:** Gain access to an initial system.
2. **Pivoting to Intermediate Systems:** Access further systems in different segments.
3. **Further Exploration and Compromise:** Continue to compromise additional systems.
4. **Repeating the Process:** Use multiple pivot points to reach targets.

Mitigation Strategies:

- **Network Segmentation:** To limit lateral movement.
- **Monitoring and Detection:** To detect abnormal activities.
- **Least Privilege Principle:** To minimize impact.
- **Regular Patching:** To close vulnerabilities.

Resolution & Retesting

Resolution Process:

1. **Vulnerability Prioritization:** Based on severity and impact.
2. **Developing Remediation Plans:** Creating action plans to fix vulnerabilities.
3. **Implementing Fixes:** Applying patches and updates.
4. **Validation:** Ensuring fixes don’t introduce new issues.

5. **Documentation:** Recording the process and outcomes.

Retesting Process:

1. **Verification of Fixes:** Ensure vulnerabilities are mitigated.
2. **Testing Scope:** Focus on previously identified vulnerabilities.
3. **Testing Methodology:** Use similar methods as initial assessment.
4. **Documentation:** Record retesting results.
5. **Continuous Monitoring:** To detect new vulnerabilities.

Best Practices:

- **Timely Resolution:** Address vulnerabilities promptly.
- **Thorough Retesting:** Ensure effective mitigation.
- **Communication:** Keep open communication between teams.
- **Automation:** Use tools for scanning and testing.
- **Risk Management:** Integrate into broader cybersecurity strategy.

File Security

File security refers to the measures and practices implemented to protect files and their contents from unauthorized access, modification, deletion, or other malicious activities. It encompasses various strategies, technologies, and best practices aimed at safeguarding sensitive information stored in digital files. Here are key aspects and considerations of file security:

Key Aspects:

1. **Encryption:**
    - **Types:** File-level, Full disk encryption.
    - **Examples:** AES, RSA, BitLocker, FileVault.
2. **Access Control:**
    - **Types:** DAC, MAC, RBAC.
    - **Implementation:** ACLs, permissions, authentication mechanisms.
3. **Data Loss Prevention (DLP):**
    - **Features:** Content inspection, policy enforcement.

- **Examples:** Forcepoint DLP, Symantec DLP, McAfee DLP.

4. **Backup and Recovery:**
    - **Best Practices:** Automated backups, offsite storage, versioning.
5. **Auditing and Monitoring:**
    - **Features:** Logging, audit trails, real-time alerts.
    - **Tools:** SIEM systems, file integrity monitoring.
6. **Secure Transmission:**
    - **Protocols:** HTTPS, SFTP, FTPS.
    - **Encryption:** SSL/TLS certificates, strong algorithms.

Best Practices:

- **Regular Updates:** Keep systems and tools up-to-date.
- **User Education:** Train on file security and phishing awareness.
- **Compliance:** Adhere to regulations like GDPR, HIPAA.

Example: Practical Scenario

Scenario:

An attacker gains access to a Linux server via an unpatched vulnerability in a web application. They then use privilege escalation techniques to gain root access by exploiting a misconfigured sudoers file. With root access, the attacker installs a backdoor to maintain persistence. They then pivot to other systems in the network by exploiting weak SSH credentials and unpatched services. The attacker exfiltrates sensitive data and covers their tracks by deleting logs and modifying timestamps.

Resolution:

1. **Patch the web application vulnerability.**
2. **Fix the sudoers misconfiguration.**
3. **Implement stronger SSH credentials and multi-factor authentication.**
4. **Regularly update and patch all systems.**
5. **Monitor logs and network traffic for abnormal activities.**

By following these steps, the organization can mitigate the vulnerabilities, prevent future attacks, and enhance their overall security posture.