Rappel Pentest – Etapes

M1-SI

Juillet 2025

Plan général

Plan

Introduction au Pentest

Objectif

Simuler une attaque réelle afin de détecter des failles de sécurité.

- Identifier les vulnérabilités
- Évaluer leur impact
- Recommander des remédiations

Attention

Un pentest nécessite un cadre légal strict : autorisation écrite et périmètre défini.

Plan

Étapes d'un Pentest

Objectif

Comprendre les différentes phases d'un test d'intrusion.

1. Reconnaissance
2. Scan et énumération
3. Exploitation
4. Post-Exploitation
5. Rapport et remédiation

Attention

Chaque étape est cruciale pour assurer la pertinence et la qualité du test.

1. Reconnaissance – OSINT

- **whois** : obtenir des infos sur le propriétaire d'un domaine (nom, e-mail, registrar, etc.).
- **DNS** :
  - Obtenir les enregistrements A, MX, TXT, NS, etc.
  - Identifier les serveurs mails ou services tiers (ex. G Suite, Outlook).
- **Google Hacking (Google Dorks)** :
  - Requêtes avancées pour trouver des fichiers ou infos sensibles.
  - Ex. : `filetype:xls inurl:/admin site:target.com`

Outils

whois, dig, Google Dorks

1. Reconnaissance – Analyse passive

- **Shodan** :
  - Moteur de recherche d'appareils connectés à Internet.
  - Ex. : trouver une caméra exposée ou un port RDP ouvert.
- **theHarvester** :
  - Récupération automatique d'e-mails, sous-domaines, hôtes publics.
  - Couvre de nombreuses sources : Google, Bing, LinkedIn, etc.
- **Recon-ng** :
  - Framework modulaire pour automatiser la collecte d'OSINT.
  - Interface en ligne de commande proche de Metasploit.

Outils

Shodan, theHarvester, Recon-ng

2. Scan et Énumération – Introduction

Objectif

Identifier les services exposés, les ports ouverts et les points d'entrée exploitables.

- Étape active : la cible peut détecter l'activité.
- Objectif : dresser un inventaire des services accessibles depuis l'extérieur.
- Permet d'orienter la phase d'exploitation vers les vulnérabilités réellement présentes.

Attention

Cette phase peut générer des alertes SIEM (logs, IDS, etc.).

2.1 Scan de ports et services

- **Nmap** :
  - Scan TCP/UDP, détection d'OS, version des services.
  - Ex. : `nmap -sS -sV -O -T4 target.com`
- **Masscan** : extrêmement rapide pour des scans à grande échelle.
- **Objectif** :
  - Identifier les ports ouverts (22, 80, 443, etc.)
  - Associer chaque port à un service et sa version.

Outils

Nmap, Masscan, Unicornscan

2.2 Énumération de services

- **Web** :
  - Brute-force de répertoires : `dirb`, `gobuster`
  - Détection de CMS : `whatweb`, `wpscan`
- **SMB / LDAP / FTP** :
  - `enum4linux` pour les partages Samba
  - Tentatives de connexion anonymes
  - Extraction d'utilisateurs (LDAP Bind)
- **Bannières et versions** :
  - Analyse des headers, des pages de login, des messages d'erreur.

Outils

enum4linux, gobuster, whatweb, nikto

3. Exploitation – Introduction

Objectif

Tirer parti des vulnérabilités identifiées pour obtenir un accès non autorisé à la cible.

- Exploitation basée sur les résultats des scans (services, versions, pages Web, etc.)
- Cette phase permet de valider l'impact d'une faille et d'atteindre des objectifs (ex. : accès admin, exécution de commandes).
- Peut utiliser des failles connues (CVE) ou des attaques sur mesure.

Attention

Risque de déstabilisation du système si l'exploit est mal maîtrisé.

3.1 Exploitation – Vulnérabilités Web

- **Injection SQL (SQLi)** :
  - Manipuler une requête SQL via un paramètre d'URL ou formulaire.
  - Permet d'extraire, modifier ou supprimer des données.
- **Cross-Site Scripting (XSS)** :
  - Insertion de code JavaScript dans une page Web vulnérable.
  - Vole les cookies, redirige, injecte des scripts malveillants.
- **Local File Inclusion (LFI)** :
  - Inclusion de fichiers locaux via une faille dans les paramètres d'URL.
  - Ex. : afficher '/etc/passwd' sur un serveur Linux.

Outils

BurpSuite, SQLmap, Firefox DevTools

3.2 Exploitation – Vulnérabilités système

- **Buffer Overflow** :
  - Débordement de mémoire causant une exécution de code arbitraire.
  - Nécessite une analyse du binaire et du comportement mémoire.
- **Exploitation de CVE connues** :
  - Recherche de failles publiques exploitables (CVE, exploit-db, GitHub).
  - Exemple : EternalBlue (MS17-010) pour Windows ou Log4Shell (CVE-2021-44228).
- **Failles de configuration** :
  - Services mal configurés (ex. : accès SSH root sans mot de passe, base MySQL sans restriction).

Outils

Metasploit, searchsploit, Python, GitHub Exploits

4. Post-Exploitation – Introduction

Objectif

Exploiter l'accès obtenu pour renforcer le contrôle, collecter des données et progresser dans le réseau.

- Phase critique pour mesurer l'impact réel de la compromission.
- Peut impliquer : escalade de privilèges, extraction de secrets, persistance, pivot.
- Objectif final : démontrer jusqu'où un attaquant peut aller.

Note

Cette phase doit rester contrôlée : ne pas affecter la stabilité ou l'intégrité des systèmes.

4.1 Escalade de privilèges

- **But** : passer d'un utilisateur simple à un compte administrateur/root.
- **Méthodes classiques** :
  - Exploitation de failles locales (CVE locales, services SUID, scripts mal protégés).
  - Mauvaises permissions sur des fichiers critiques.
  - Analyse des tâches cron ou services Windows.
- **Outils** :
  - `linpeas.sh`, `winPEAS.exe`, `sudo -l`

Outils

linPEAS, winPEAS, GTFOBins, sudo, PowerUp.ps1

4.2 Extraction et reconnaissance interne

- **Dump de mots de passe** :
  - `mimikatz` sous Windows : extraction des mots de passe en mémoire.
  - Dump de hash (`/etc/shadow`, SAM) et bruteforce offline.
- **Reconnaissance réseau interne** :
  - Identifier d'autres machines et services internes accessibles.
  - Ex. : ping sweep, scan de ports en interne, analyse des partages SMB.

Outils

Mimikatz, crackmapexec, Responder, BloodHound

4.3 Maintien de l'accès et mouvements latéraux

- **Backdoors et shells inversés** :
  - Ajout d'un compte admin, backdoor sur service légitime.
  - Reverse shell via cron, services persistants.
- **Mouvements latéraux** :
  - Se déplacer d'une machine à une autre dans le réseau interne.
  - Utiliser des identifiants volés ou relayer des sessions.

Outils

Netcat, Meterpreter, PsExec, SSH pivot, socks proxies

5. Remédiation – Propositions de correction

- **Corrections à court terme** :
  - Désactivation des services vulnérables non utilisés.
  - Correctifs logiciels, changements de mots de passe.
- **Mesures à moyen/long terme** :
  - Segmentation réseau, principe du moindre privilège.
  - Surveillance accrue, durcissement système (hardening).
  - Campagnes de sensibilisation utilisateur.
- **Test de remédiation** :
  - Vérification que les vulnérabilités corrigées ne sont plus exploitables.
  - Parfois appelé « re-test ».

But final

Réduire significativement la surface d'attaque de l'organisation.

Plan

Conclusion

- Le pentest est un processus rigoureux et structuré visant à évaluer la sécurité d'un système dans des conditions réalistes.
- Chaque étape – de la reconnaissance au rapport – a un rôle clé dans l'identification, l'exploitation et la remédiation des vulnérabilités.
- La valeur d'un pentest repose autant sur sa technicité que sur la clarté de ses livrables.

À retenir

Un bon pentest ne se limite pas à « pirater » un système, il aide à mieux le protéger.

« *Tester pour mieux défendre.* »