

RAPPORT DE PROJET

Mise en place d'un IPS avec Fail2Ban sur CentOS 9

Réalisé par :
Name1 NOM1
Name2 NOM2

Table des matières

1	Introduction	2
2	Architecture et Prérequis	3
2.1	Environnement Technique	3
2.2	Préparation du Système	3
3	Mise en Œuvre	4
3.1	Configuration de la Protection SSH	4
3.2	Création d'une Prison (Jail) Personnalisée	4
3.2.1	1. Création du fichier de log simulé	4
3.2.2	2. Création du Filtre (Regex)	4
3.2.3	3. Activation de la Prison	5
4	Tests et Validation	6
4.1	Script de Simulation d'Attaque	6
4.2	Résultats du Test	6
5	Conclusion	8

Chapitre 1

Introduction

La sécurité des serveurs Linux est un enjeu critique dans l'administration système moderne. Les attaques par force brute (brute-force) constituent l'une des menaces les plus courantes, visant à deviner les mots de passe des services exposés comme SSH.

Ce projet a pour objectif de concevoir, configurer et valider un système de prévention d'intrusions (IPS) automatisé utilisant **Fail2Ban** sur une distribution **CentOS Stream 9**.

Le système sera capable de :

- Surveiller les journaux (logs) du système en temps réel.
- Déetecter les tentatives de connexion échouées.
- Bannir dynamiquement les adresses IP malveillantes via le pare-feu (Firewalld).
- Protéger à la fois le service SSH et une application personnalisée simulée.

Chapitre 2

Architecture et Prérequis

2.1 Environnement Technique

- Système d’exploitation : CentOS Stream 9
- Logiciel de sécurité : Fail2Ban v1.x
- Pare-feu : Firewalld (intégré par défaut dans CentOS)
- Services surveillés : Daemon SSH (sshd) et une application métier personnalisée.

2.2 Préparation du Système

Avant l’installation, le système a été mis à jour et le dépôt EPEL (Extra Packages for Enterprise Linux) a été ajouté, car Fail2Ban n’est pas inclus dans les dépôts par défaut.

```
1 sudo dnf install epel-release -y
2 sudo dnf install fail2ban fail2ban-firewalld -y
3 sudo systemctl enable --now fail2ban
```

Listing 2.1 – Installation des paquets nécessaires

```
[fail2ban@vbox ~]$ sudo systemctl stop fail2ban
[fail2ban@vbox ~]$ sudo nano /etc/fail2ban/jail.local
[fail2ban@vbox ~]$ sudo systemctl start fail2ban
[fail2ban@vbox ~]$ sudo systemctl status fail2ban
● fail2ban.service - Fail2Ban Service
    Loaded: loaded (/usr/lib/systemd/system/fail2ban.service; enabled; preset: disabled)
    Active: active (running) since Tue 2025-12-23 13:44:48 +01; 3s ago
      Docs: man:fail2ban(1)
   Process: 3364 ExecStartPre=/bin/mkdir -p /run/fail2ban (code=exited, status=0/SUCCESS)
   Main PID: 3365 (fail2ban-server)
     Tasks: 5 (limit: 22981)
    Memory: 11.6M (peak: 12.1M)
       CPU: 103ms
      CGroup: /system.slice/fail2ban.service
              └─3365 /usr/bin/python3 -s /usr/bin/fail2ban-server -xf start

Dec 23 13:44:48 vbox systemd[1]: Starting Fail2Ban Service...
Dec 23 13:44:48 vbox systemd[1]: Started Fail2Ban Service.
Dec 23 13:44:48 vbox fail2ban-server[3365]: Server ready

[fail2ban@vbox ~]$ █
```

FIGURE 2.1 – Vérification de l’état du service Fail2Ban après installation.

Chapitre 3

Mise en Œuvre

3.1 Configuration de la Protection SSH

Pour protéger le service SSH, nous avons créé un fichier de configuration locale `/etc/fail2ban/jail.local`. Une particularité de CentOS 9 est l'utilisation du backend `systemd` pour la lecture des logs SSH.

```
1 [DEFAULT]
2 bantime = 1h
3 maxretry = 3
4 findtime = 10m
5 ignoreip = 127.0.0.1/8 ::1
6
7 [sshd]
8 enabled = true
9 backend = systemd
10 mode = normal
```

Listing 3.1 – Configuration jail.local (Extrait SSH)

3.2 Crédation d'une Prison (Jail) Personnalisée

Afin de démontrer la flexibilité de Fail2Ban, nous avons simulé une application métier.

3.2.1 1. Crédation du fichier de log simulé

Nous avons créé un fichier vide qui recevra les logs de notre fausse application.

```
1 sudo touch /var/log/custom_app.log
2 sudo chmod 644 /var/log/custom_app.log
```

3.2.2 2. Crédation du Filtre (Regex)

Le filtre définit ce qu'est une "attaque". Nous avons configuré une expression régulière (Regex) dans `/etc/fail2ban/filter.d/custom-app.conf` pour détecter la chaîne "Failed login".

```
1 [Definition]
2 failregex = ^.*Failed login from <HOST>.*$
```

```
3 ignoreregex =
```

Listing 3.2 – Filtre personnalisé (custom-app.conf)

3.2.3 3. Activation de la Prison

Nous avons ajouté la configuration suivante à la fin de `jail.local` pour lier le filtre au fichier de log.

```
1 [custom-app]
2 enabled = true
3 filter = custom-app
4 logpath = /var/log/custom_app.log
5 maxretry = 3
6 backend = auto
```

Listing 3.3 – Ajout dans jail.local

Chapitre 4

Tests et Validation

Pour valider le fonctionnement, nous avons développé un script Bash simulant une attaque par force brute.

4.1 Script de Simulation d'Attaque

Ce script injecte 5 tentatives de connexion échouées provenant de l'IP fictive 192.0.2.100.

```
1 #!/bin/bash
2 LOG_FILE="/var/log/custom_app.log"
3 ATTACKER_IP="192.0.2.100"
4
5 echo "Simulation d'attaque depuis $ATTACKER_IP..."
6 for i in {1..5}
7 do
8     echo "$(date) Failed login from $ATTACKER_IP" | sudo tee -a $LOG_FILE
9     sleep 1
10 done
```

Listing 4.1 – Script simulate_attack.sh

4.2 Résultats du Test

Après l'exécution du script, nous avons vérifié que Fail2Ban a bien détecté l'attaque et banni l'IP.

```
[fail2ban@vbox ~]$ sudo fail2ban-client status custom-app
Status for the jail: custom-app
|- Filter
  |- Currently failed: 0
  |- Total failed:      0
    `-- File list:        /var/log/custom_app.log
|- Actions
  |- Currently banned: 1
  |- Total banned:     1
    '-- Banned IP list: 192.0.2.100
[fail2ban@vbox ~]$
```

FIGURE 4.1 – Preuve que l'IP 192.0.2.100 a été bannie par Fail2Ban.

Enfin, nous avons vérifié que la règle a été propagée au niveau du pare-feu Firewalld.

```
[fail2ban@vbox ~]$ sudo firewall-cmd --list-rich-rules
[sudo] password for fail2ban:
rule family="ipv4" source address="192.0.2.100" port port="0-65535" protocol="tcp"
reject type="icmp-port-unreachable"
[fail2ban@vbox ~]$
```

FIGURE 4.2 – Vérification de la règle Firewalld (Rich Rule).

Chapitre 5

Conclusion

Ce projet a permis de mettre en place une solution de sécurité robuste et automatisée sur CentOS 9. Grâce à Fail2Ban, le serveur est désormais capable de :

1. Déetecter proactivement les attaques par force brute sur SSH.
2. Surveiller des applications personnalisées via des filtres Regex.
3. Bloquer instantanément le trafic malveillant via Firewalld.

La validation par simulation d'attaque a confirmé l'efficacité des prisons configurées. Ce système constitue une première ligne de défense essentielle pour tout serveur exposé sur Internet.