Rapport de Projet : Analyse de logs FortiGate et génération automatisée de règles firewall

1. Introduction

Dans un contexte de sécurité réseau, il est essentiel de détecter les flux autorisés afin de restreindre les accès inutiles et limiter la surface d'attaque. Ce projet a consisté à développer un outil d'analyse de logs FortiGate en Python afin de générer dynamiquement des règles de firewall précises à partir d'un flux général autorisant tout (règle "ALL").

2. Objectifs du projet

- Extraire les flux réellement utilisés depuis des logs bruts.
- Identifier et catégoriser les flux selon leur protocole, leur destination et leur fréquence.
- Calculer des statistiques de trafic afin de définir un seuil pertinent d'intérêt.
- Générer automatiquement des règles FortiGate basées sur les flux identifiés.
- Séparer les flux généraux des flux HTTP/HTTPS publics pour un traitement spécifique.

3. Méthodologie

Étape 1 : Extraction des données de logs

- Parsing des lignes de log FortiGate au format brut.
- Utilisation des regex pour identifier : interfaces source/destination, IPs, ports, protocole, action, service...

Étape 2 : Filtrage et catégorisation

- Suppression des flux "deny".
- Classification des flux : généraux vs flux publics HTTP/HTTPS.
- Identification des sous-réseaux /24 pour le regroupement IP.

Étape 3 : Analyse statistique

- Calcul du nombre moyen de connexions par flux.
- Détermination du **75e percentile** pour filtrer les flux les plus fréquents.
- Création d'un seuil adaptatif à partir du percentile.

Étape 4 : Génération de règles FortiGate

- Création des objets d'adresse IP (source/destination).

- Mappage des ports vers les services.
- Génération de services personnalisés si nécessaire.
- Génération des règles "firewall policy" FortiGate.

4. Résultats

- Identification automatique des flux pertinents (plus de X flux analysés).
- Génération d'un fichier de règles prêtes à être importées dans FortiGate.
- Gestion des règles HTTP/HTTPS vers IPs publiques via une règle générique.
- Réduction significative du nombre de règles manuelles.

5. Défis rencontrés et solutions

Problème	**Solution apportée**
Flux peu fréquents et bruités	Calcul de percentile 75 pour fixer un seuil minimal
Multiplicité des ports non standard	Génération automatique de services personnalisés
Risque de redondance d'IP	Regroupement en sous-réseaux /24 ou /32 selon le contexte

6. Conclusion

Ce projet m'a permis de combiner compétences en Python, sécurité réseau, analyse de logs, et configuration de firewall. Grâce à ce script, il devient possible de créer une politique de filtrage dynamique, adaptée aux besoins réels du trafic, tout en réduisant l'exposition à des flux non nécessaires.

Gitlab : https://gitlab.com/ben9205303/analyse-log/-/blob/main/gui.py?ref_type=heads