**TP Iptables — Thierry Dupré**

**Groupe :** Cinquin Andy / Gauthier Mathis.

Pour réaliser ce TP nous avons préparé deux VMs l’une dans le but d’être serveur Web avec un service comme Apache ou Nginx (dans notre cas nous avons choisis Apache). Et la seconde dans l’objectif d’être le serveur de base de données avec un service comme MySQL, MariaDB (nous avons choisi MariaDB)…

Chaque VM possède deux interfaces réseau, les premières ayant pour objectif de communiqués avec l’extérieur donc d’être connectées à Internet. Les secondes, ont-elles pour objectif de faire communiquer les deux machines entre elles dans un LAN.

**6.1 Step 1 : Secure that stuff :**

* Quels ports allons-nous autoriser sur l’interface réseau WAN de chaque VM et pourquoi ?
* Nous allons autoriser les ports **22**, **53**, **80**, **443** sur la machine Web :
  + Le port **22** pour nous permettre d’avoir une connexion SSH avec cette machine.
  + Les ports **80** et **443** pour que notre serveur Web puisse répondre aux requêtes HTTP / HTTPS, cela permettra de consulter notre site depuis un navigateur Web.
  + Le port **53**, port utilisé par les services DNS, ce qui permettra à notre serveur de résoudre le nom de domaine de notre site Web si celui-ci en a un. Ainsi que de pouvoir être client pour demander à d’autres serveurs DNS de résoudre des noms de domaine qu’il ne connait pas.
* Nous allons autoriser les ports **22**, **3306** sur la machine Bdd :
  + Le port **22** ici aussi pour nous permettre d’avoir une connexion SSH avec cette machine.
  + Si ce serveur n’est pas dans le même réseau LAN que la machine Web alors on autorisera le port **3306** pour que le serveur Bdd puisse recevoir et répondre aux requêtes liées au service MariaDB.
* Quelle(s) règle(s) iptables allons-nous provisionner sur l’interface LAN de chaque VM ? Pourquoi ?
* Nous allons autoriser le port **22** en entrée sur les deux machines ainsi qu’en sortie, ce qui permet d’avoir un tunnel SSH entre les deux machines pour que les échanges puissent se faire de façon sécurisée.
* Ainsi que les ports nécessaires au fonctionnement de nos services :
  + Exemple : le port **3306** si notre serveur MariaDB n’est accessible que depuis le LAN par le serveur Web.
* On va vouloir interdire l’accès à notre plateforme aux machines de l’hébergeur Digital Ocean (et donc à tous leurs blocs), comment faire ?
  + On récupère leurs blocs d’adresse IP avec le site <https://ipinfo.io/AS14061>

Puis on sélectionne uniquement les IP avec la regex suivante en remplaçant la chaîne par uniquement le premier groupe de sélection :

« **^([0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.0)(.\*)$** »

Puis on drop ses groupes d’IP, dans nos règles IPtables.

* Quelle solution allons-nous utiliser pour rendre ces règles persistantes (i.e. résistantes au reboot) ?
  + Une solution pour rendre nos règles persistantes est d’installer le paquet « **iptables-persistent** » puis de sauvegarder nos règles dans un fichier avec la commande : « **iptables-save > nom\_fichier ».**

Et recharger nos règles à chaque reboot avec la commande :

« **iptables-restore < nom\_fichier** » en faisant en sorte que celle-ci s’exécute automatiquement au reboot de la machine.

**6.2 Step 2 : « Parsage » des logs et exports BDD** (*Cf : le script*)

* On va vouloir parser les logs Apache2 de façon à extraire :
  + Les différentes adresses IP visitant notre site, ainsi que la fréquence de ces visites (i.e. le nombre de requêtes effectuées par ces Ips).
  + Les URLs visitées le plus souvent.
  + On va vouloir que nos scripts écrivent leurs résultats dans différents fichiers dans notre répertoire utilisateur (ACLs), au format CSV.

Pour parser les logs selon les IPs qui ont visité notre site et la fréquence à laquelle ces adresses visitent notre site. Nous préparons un fichier « csv » ensuite nous explorons notre fichier de log Apache2 préalablement créer pendant la configuration de ce service. Puis avec la commande « awk » on ne garde que les adresses IPs, ensuite avec la commande « sort » nous pouvons réordonnées de façon à obtenir un fichier plus lisible à la fin. Et avec la commande « uniq » nous pouvons également supprimer les lignes dupliquées dans le but de n’en retrouver qu’une, cela nous permet également de compter les occurrences de ces lignes.

Pour parser les logs, pour récupérer les URLs les plus visités on utilise également la commande « awk » afin de récupérer les URLs, avec « egrep » pour sortir les lignes contenant uniquement des URLs. On utilise la commande « sort » pour les trier. Et avec la commande « sed » en utilisant une regex on sélectionne tous les espaces en début de chaîne pour supprimer les espaces excessifs, ensuite avec un deuxième « sed » on remplace les espaces entres les informations par des virgules dans le but d’obtenir le format utilisé pour un fichier csv.

Pour extraire au format CSV de notre base de données le nombre d’articles et le nombre d’utilisateurs.

**6.3 Step 3 : Ban all the lamers !** (*Cf : le script*)

Bloquer les utilisateurs “malveillants” :

* Les utilisateurs effectuant un certain nombre de requêtes dans un intervalle de temps donné (« DoS »).
* Les utilisateurs visitant un peu trop souvent une URL précise (au hasard, page de login – « Brute Force »).
* Les petits rigolos essayant de se connecter en SSH sur notre serveur Web.
* On va vouloir les bannir via iptables, pendant un intervalle de temps donné (on prendra 1h dans notre cas).

Pour cela on peut utiliser le service « fail2ban » qui nous permet de faire plusieurs configurations permettant de définir des règles, par service, par port, etc. De vérifier grâce aux logs de nos services (par exemple nos logs Apache2), si les règles sont bien respectées.

On peut en effet définir le nombre de tentatives maximum de connexion, ainsi que l’intervalle de temps dans lesquelles ces essais peuvent être effectués, ainsi que le temps de bannissement.

Si jamais ces règles ne sont pas respectées alors l’utilitaire pourra ajouter une règle temporaire à IPtables qui pourra « DROP » l’IP qui a essayé de forcer notre serveur pendant la durée que nous aurions prédéfinie.