DOCUMENTATION
2ARC

Nom du projet	FirewallARC
Type de document	Documentation technique
Date	05/06/2018
Auteurs	AUFFRET Guillaume EVEN Alexandre FERNANDO Kevin HARSCOUET Josselin LE ROUX Malo

Table des matières

Généralités sur le projet

Expression de besoin

Notre démonstrateur a pour but de valider les principaux aspects techniques du projet Firewall IP. Il nous était demandé de créer un pare feu sans contraintes de langage ou de système d'exploitation, capable de filtrer et bloquer des paquets réseaux d'après un ensemble de règles défini par l'utilisateur.

Pour simplifier le développement du démonstrateur, le travail à été décomposé en 2 parties :

- Démonstration du fonctionnement du filtre par la lecture de fichiers de capture
- Démonstration du fonctionnement du pare-feu sur le réseau réel

Cahier des charges

Programmes :

- Création d'un programme pour lire les fichiers .cap testant la logique du filtre
- Création d'un programme compatible avec un système d'exploitation existant (au choix), gérant les paquets entrants et appliquant les filtres d'après un ensemble de règles

Règles :

- Gestion des règles facile pour l'utilisateur
- Filtres basés sur les headers des paquets (ports, protocoles, ip…)
- Application de filtres sur les paquets : accepté ou refusé

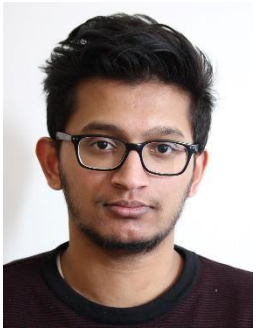
Contraintes techniques :

- Choix libre du langage et du système d'exploitation
- Choix de la librairie permettant la lecture des fichiers de capture
- Utilisation interdite de solutions déjà existantes
- Capacité à gérer une large gamme de paquets

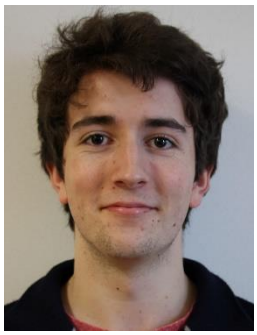
Equipe en charge du projet

AUFFRET
Guillaume

EVEN
Alexandre



FERNANDO
Kevin



HARSCOUET
Josselin

LE ROUX
Malo

Solutions techniques

OS et langage

Nous avons choisi Windows pour le démonstrateur, car ce système d'exploitation est le plus repandu. Pour cette version de démonstration, seul cet OS permet de faire fonctionner le pare feu. Cette caractéristique pourra être revue si besoin.

Concernant le langage, nous avons choisi le Python, pour simplifier la production et la maintenance du pare-feu, ainsi que son grand nombre de librairies et sa documentation complète. C'est également un langage que l'équipe maitrise bien.

Lecture des fichiers de capture

Pour la lecture des fichiers de captures, nous avons choisi Scapy. Une fonction de cette librairie permet la lecture des fichiers « .cap ». Cependant, plus le fichier est volumineux, plus cette fonction est lente, pouvant provoquer des temps de chargement de quelques secondes à l'ouverture d'un fichier. Il serait donc judicieux de ne pas intégrer cette fonctionnalité dans la version définitive du projet.

Gestion des paquets

La gestion des paquets peut être segmentée en 3 parties : la décomposition des paquets, l'application des filtres et l'envoi ou non des paquets vers le userland.

La décomposition des paquets et l'envoi vers le userland se fait avec la librairie pydivert. Elle ouvre une session à la réception d'un paquet, puis le décompose en fonction du protocole qui lui est associé. Pour ce démonstrateur, les protocoles pris en charge par le programme sont ICMP, UDP et TCP. Une fois cette décomposition effectuée, il est envoyé vers le userland si le filtre le permet.

L'application des filtres se fait sur 5 critères : l'adresse IP source, l'adresse IP de destination, le protocole, le port source et le port de destination. Ces différents critères peuvent être appliqués un par un (refuser tous les paquets utilisant le protocole TCP), comme en groupe (refuser les paquets dont l'adresse IP source correspond à l'adresse XXX.XXX.XXX.XXX et dont le protocole est UDP).

Les données sur les filtres du pare-feu se trouvent dans le fichier FirewallARC.conf. Ce fichier peut être modifié directement par l'utilisateur, mais il est conseillé de passer par l'invite de commande ou l'interface pour éviter toute faute de rédaction. Le paterne est le suivant :

Propriété1: value1, propriété2: value2…\n

Une propriété intéressante de cette rédaction est son adaptabilité. Il est possible d'ajouter des propriétés et des valeurs non prévues à l'origine, permettant un maintien plus simple, ainsi qu'une amélioration du script sans avoir à revoir en profondeur l'architecture du programme.

Interaction utilisateur / machine

2 outils permettent à l'utilisateur d'interagir avec le pare-feu : l'invite de commande du programme et l'interface graphique.

L'invite de commande permet de lancer le pare-feu, l'interface, ainsi que toutes les fonctionnalités intégrées au projet. L'ensemble des commandes se trouvent dans le manuel de l'invite de commande, accessible avec la commande **man**.

Attention : Il est nécessaire d'ouvrir le programme en administrateur, sans quoi le pare-feu ne fonctionnera pas.

Parmi ces fonctionnalités, il est possible depuis l'invite de commande de lire un fichier de capture, de bannir un hostname (exemple : « ban rennes-atalante.fr ») ainsi que de le débloquer, de stopper les services (firewall ou interface), d'ajouter ou de supprimer les règles, ou encore de voir tous les paquets traités depuis le démarrage du pare-feu.

Nous avons également intégré une interface afin de permettre un accès plus simple aux fonctionnalités du programme. Elle permet de voir le flux de paquets en direct, leur état (bloqué ou non), de lire les fichiers de capture, de modifier les règles et d'enregistrer une capture (au format texte).

Installation

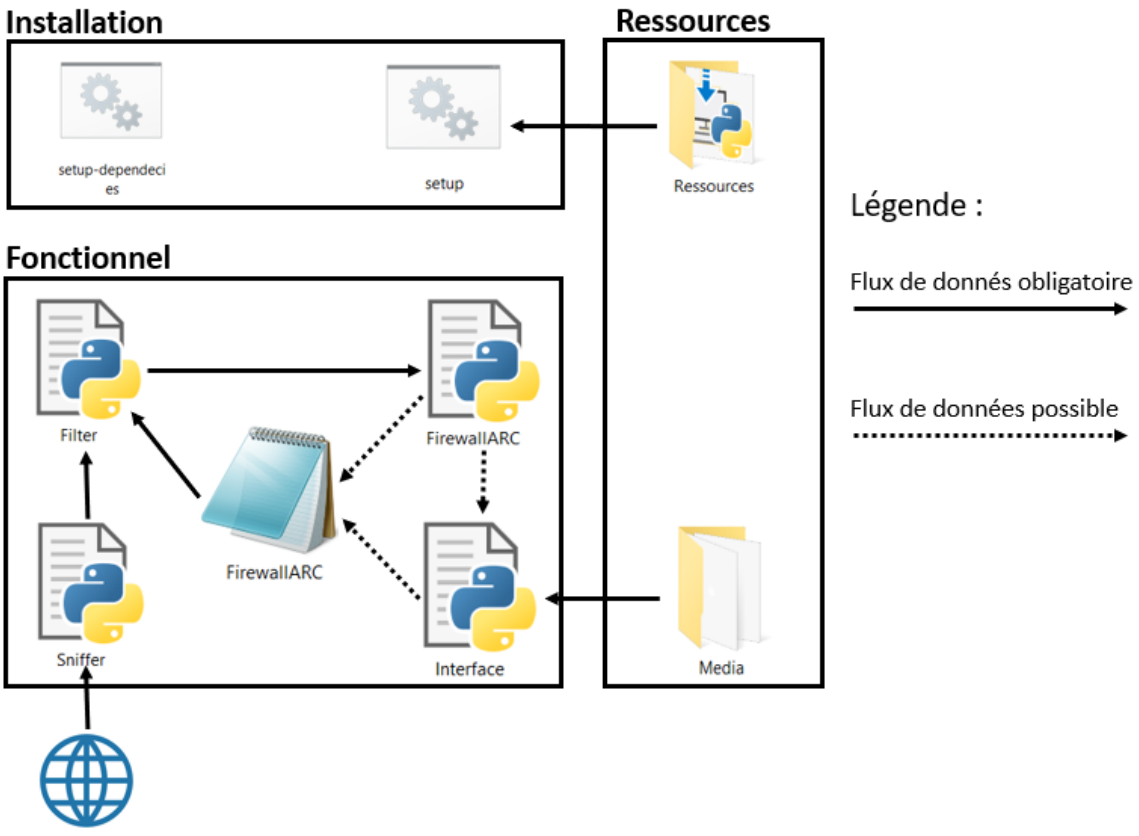
Pour simplifier l'utilisation, nous avons ajouté des scripts. Le premier (setup.bat) permet d'installer python 3.6 tandis que le deuxième (setup-dependencies.bat) installe les librairies complémentaires en plus de créer un raccourci sur le bureau. Pour connaitre la démarche complète d'installation, suivre le manuel d'utilisation, dans la partie « installation ».

Complément technique

Liste des fichiers du projet

Nom du fichier	Description
FirewallARC.py	- Gestion global du programme - Lien entre les différents script python - Gestion de l'invité de commande
Filter.py	- Chargement des informations contenu dans FirewallARC.conf - Application des filtres sur les paquets - Gestion les paquets
Sniffer.py	- Réception et décomposition des paquets - Envoi des paquets vers le userland si le filtre le permet
Interface.py	- Gestion complète de l'interface graphique - Affichage du flux de paquets
Setup.bat	- Première phase d'installation du pare-feu : installation de python
setup-dependecies.bat	- Deuxième phase d'installation du pare-feu : installation des librairies
FirewallARC.conf	- Sauvegarde des règles du pare-feu
Media	- Dossier contenant les images et la police de l'interface
Ressource	- Dossier contenant l'installateur python

Architecture globale



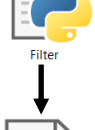
Séquences d'exécution

Traitement d'un paquet

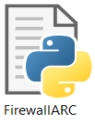
Internet : envoi des paquets vers l'ordinateur cible.

Sniffer.py : reçoit les paquets et les traite pour permettre au filtre d'analyser le paquet. Envoie les paquets vers le userland en fonction des résultats du filtre.

Filter.py : gère et filtre les paquets d'après les règles contenues dans le fichier de configuration FirewallARC.conf (voir schéma ci-dessus).



FirewallARC.py : gère les différents programmes, fait le lien entre l'interface et les paquets si besoin (si l'interface est allumée), permet la gestion globale du pare-feu et des fonctionnalités de l'invite de commande.



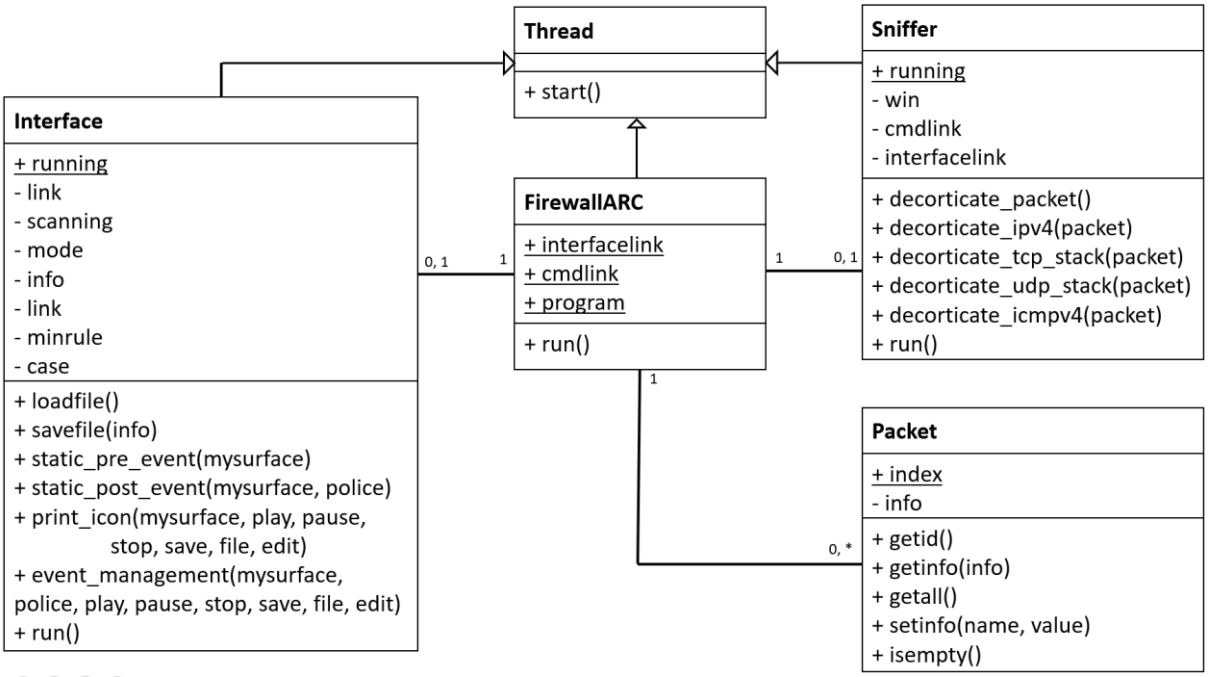
Interface.py : permet une gestion plus simple du pare-feu, peut modifier la configuration des règles et affiche en direct le flux de paquets. Les images et polices de l'interface se trouvent dans le dossier « Media ».

Installation du pare-feu

Setup.bat : première phase de l'installation. Installe Python, contenu dans le dossier Ressources.

Setup-dependecies.bat : deuxième phase de l'installation. Installe les librairies nécessaires au fonctionnement du pare-feu.

Diagramme de classe



Projections pour la version final

Ce démonstrateur à permis de se faire une bonne idée du volume de travail nécessaire à la réalisation d'un pare-feu. Il a également permis de valider et invalider les solutions techniques envisagées durant la phase de recherche initiale. La version définitive comportera les améliorations suivantes :

- Disponibilité sur tous les systèmes d'exploitation.
- Ajout et amélioration des filtres (adresse mac, plus de protocoles…).
- Exécution en tâche de fond, sans intervention de l'utilisateur.
- Amélioration de l'interface graphique.
- Suppression de la fonctionnalité de lecture de fichier de capture.
- Ajout de fonctionnalités à la demande du client.

Annexe

- Manuel pydivert : https://pythonhosted.org/pydivert/
- Manuel Scapy : http://scapy.readthedocs.io/en/latest/index.html
- Manuel pygame : https://www.pygame.org/docs/