TP Threat Intel – Conception d’un outil de détection d’hameçonnage ciblé

IBRAHIM Maysaa AMMAR Ahlam TREVIDI Pierre BERJEAUD Erwann

Objectifs

Le but de ce TP est de développer une solution capable de détecter automatiquement et en temps réel des sites utilisés pour faire du hameçonnage ciblé (phishing en anglais).
L'hameçonnage ciblé consiste à créer un site internet ressemblant à un site utilisé par la cible afin de lui dérober des informations sensibles (mots de passe, données bancaires).

Secteur d'activités

Nous avons décidé de prendre les banques françaises comme secteur d'activités, c'est un secteur avec énormément de données sensibles.
Les identifiants de clients de différentes banques sont au moins autant importants que leurs numéros de cartes bancaires, si ce n'est plus.

Choix techniques

Nous avons décidé d'utiliser le langage Python (https://www.python.org/) pour sa simplicité et le fait que l'on puisse implémenter les différents services que l'on veut utiliser. Nous utiliserons Git pour le regroupement de code et le versioning de la solution.

Services utilisés

Pour cette solution, nous allons utiliser :

- certstream (https://certstream.calidog.io/) afin de récupérer en temps réel les informations sur les sites validant leur certificat.
- GeoIP (https://www.maxmind.com/fr/geoip-demo), permettant de trouver la géolocalisation d'une adresse IP.
- VirusTotal (https://www.virustotal.com/en/documentation/public-api/#scanning-urls) qui compare une URL avec une base de données de sites malveillants connus (les abuses).

Fonctionnement général

Tout d'abord, nous allons créer notre propre jeu de données correspondant aux noms de domaine des banques françaises nationales.
Ensuite, nous comparerons les noms de domaines retournés par certstream avec notre base de données en utilisant la distance de Levenshtein. Cet algorithme renvoie le nombre minimal de caractères à ajouter/remplacer/supprimer pour que les chaînes soient égales. Nous pensons prendre en valeur, pour renvoyer une alerte, au maximum la moitié de du nombre de caractères de la chaîne à analyser.
Si l'alerte est déclenchée, on passe au niveau de filtrage suivant, avec GeoIP. Comme nous nous intéressons seulement aux banques françaises, si le site n'est pas hébergé en France, nous déclenchons une nouvelle alerte.
Nous utiliserons ensuite VirusTotal afin de comparer ce site avec la base de données des sites malveillants connus.
A l'aide de ces 3 filtres, nous pouvons donner des degrés de malveillance de sites internet louches.

Si le nom de domaine ressemble à un nom dans notre base, mais qu'il est hébergé en France il a moins de chances d'être malveillant.

Pseudo-code

Tout d'abord, il nous faut développer une fonction permettant d'effectuer la distance de Levenshtein (nous n'avons pas encore regardé au niveau des librairies python).

entier Levenshtein(caractere chaine1[1..longueurChaine1, caractere chaine2[1..longueurChaine2])

Ensuite, une fonction permettant de récupérer et de comparer grâce à la fonction précédente l'url des sites envoyés par certstream (message et context étant les arguments de certstream, chaineATester sera une ligne du fichier à ouvrir dans le main).

```
caractere compareBDD(message, context, chaineATester)
    pour (ligne=0, nombreLignes(fichier), ligne++)
        val = Levenshtein(chaineATester, message['data']['source']['url'])
        si (val<=longueur(chaineATester)/2)
            return  message['data']['source']['url']
```

Ensuite, si l'url ressemble à une url de la base, on utilise GeoIP afin de savoir si le site est hébergé en France.

```
Boolean compareGeoIP (chaine)
    gir = gi.record_by_name(chaine)
    si  (gir['country_name'] == 'France')
        return false
    sinon
        return true
```

Ensuite, on regarde si l'url est dans la base de données de VirusTotal.

Boolean compareVirusTotal (chaine)

Retourne vrai si la chaine est dans la BDD et faux sinon.