UNIVERSITÉ DE RENNES 1

Pour le 06/01/23

Compte rendu de CC SRIO

Partie Pratique

Noémie Le Det, Thibaud Lequertier, William Babin, Yelli Coulibaly, Zoé Costan

esir
ECOLE SUPERIEURE
D'INGENIEURS DE RENNES

Durant ce CC, nous nous positionnons en tant qu'attaquants.

Obtention de secret d'architecture

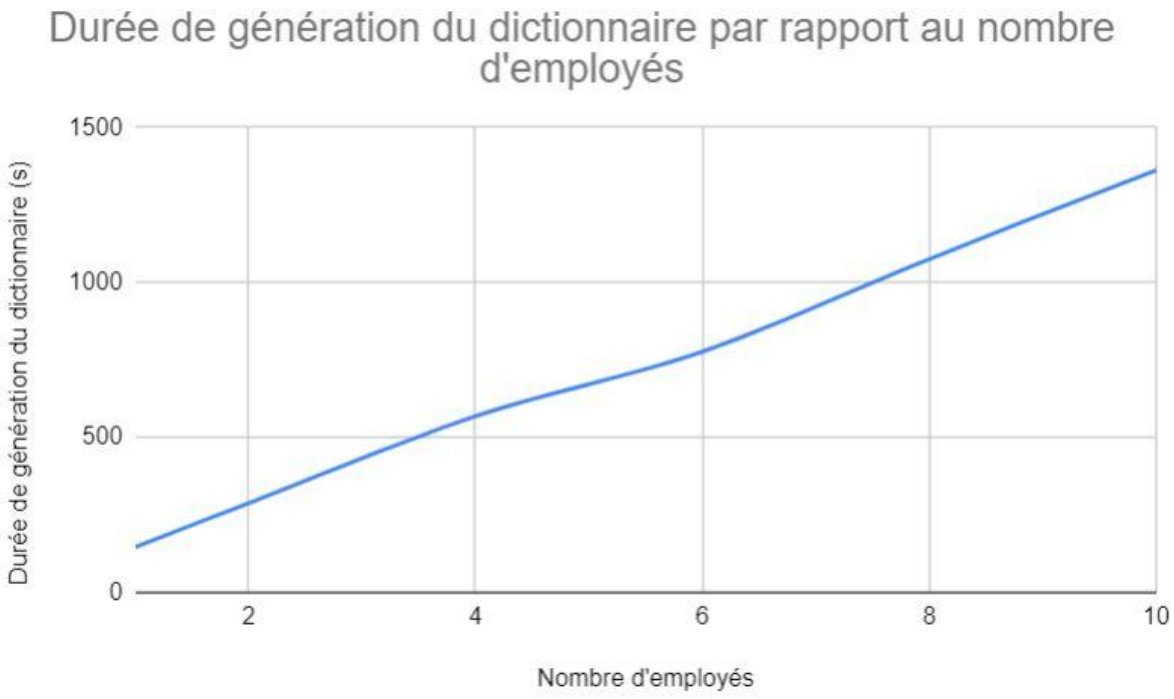
On a tout d'abord créé un fichier JSON contenant chaque dossier contenant les users et ses droits associés. Pour chaque dossier, nous regardons les droits des utilisateurs, nous cherchons celui qui possède le plus de droits. Pour cela, nous avons compté le nombre de '-' pour chaque droit. Celui possédant le moins de '-' aura donc le plus de droits.

Gestion de la base de donnée

Le premier code peut engendrer un dépassement de la taille du tableau et ainsi l'écriture sur un espace mémoire non réservé pour ce tableau. Ces codes sont ainsi exposés au risque de BufferOverFlow. Pour éviter ce problème, il faut alors ajouter un assert. Cette action permettra à l'utilisateur de ne pas écrire en mémoire à un autre endroit que celui fourni. De plus, dans le second code, on peut voir que le tableau "days" est un pointeur de pointeur, il faut enlever un * pour que sa fonction soit la bonne.

Génération de dictionnaire et force brute

Nous avons premièrement généré un script calculant toutes les permutations possibles entre les différents anniversaires, les noms et les dates d'embauche des employés, qui écrit le résultat de toutes ces permutations dans un dictionnaire nommé permutations.txt. Notre script ne prend pas seulement en compte les permutations à 6 éléments, mais aussi celles à 1, 2, 3, 4 et 5 éléments. Ce script a mis du temps à s'exécuter, car notre code n'est malheureusement pas optimisé. En effet, notre programme utilise six boucles for imbriquées, ce qui relève d'une complexité en n^6. Ainsi, nous avons pu réaliser un diagramme des performances de notre programme sur le PC de Noémie. Les chiffres sont alors divisés par 100 à cause du nombre très important de permutations réalisées. Nous observons en moyenne un temps de traitement de 145 secondes par personne sur le PC de Noémie. La durée de conversion du fichier de permutation en fichier crypté est de 3 secondes pour 10 personnes.



Déchiffrement d'un point d'accès et nettoyage des traces :

L'objectif ici va être de récupérer le Handshake du réseau SRIOEYE1 afin de pouvoir récupérer la clé wifi de ce dernier à l'aide du dictionnaire permutée dans la partie précédente.

1 - Handshake + bruteforce

Récupérer handshake :

Passage en mode monitor : airmon-ng start wlan0
Écoute du trafic alentour : airodump-ng wlan0mon => récupère le BSSID et le channel de SRIOEYE1
Écoute sur channel : iwconfig wlan0mon channel <num_channel>
Capture pour récupérer le handshake : airodump-ng –bssid <bssid> -c <num_channel> -w log wlan0mon
Forcer la déconnexion de tout le monde : aireplay-ng -0 0 -a <bssid> wlan0mon

Brute Force :

aircrack-ng log-01.cap -w dico.txt

2 - Supprimer ses traces

rm /var/sys/syslog ou alors history -c

Ainsi, grâce au dictionnaire calculé précédemment et le fichier log1.cap, en utilisant la commande décrite plus haut, on peut trouver la clé du wifi SRIOEYE1. Cependant, après avoir testé toutes les permutations possibles, on n'a pas su trouver la clé.

Analyse du tas d'exécution :

Dans cette partie, un informateur nous fournit le tas d'exécutions de certains programmes sensibles et nous demande d'effectuer une analyse.

Les informations fournies sont stockées dans un fichier tas.txt.

Premièrement, nous avons écrit le script : script.sh qui va contenir 2 fonctions :

- getSizeMax() qui affiche la rangée mémoire avec la plus grosse taille
- zoneBuffer() qui affichent les zones qui sont susceptibles à un buffer overflow et non protégées pour les processus fils.