TP1

Chiffrement symétrique

Compte rendu

Le TP donnera lieu à la rédaction d'un petit fichier texte contenant les réponses aux questions ainsi que d'éventuels résultats de commandes. Pour cela utiliser un éditeur quelconque et classer les réponses par numéro et utilisez le copier-coller si besoin. Le but est d'obtenir un texte court permettant de vérifier vos acquis.

Le fichier portera votre nom suivi du numéro de TP en l'occurrence 1, exemple : martin1.txt

Le fichier sera envoyé comme devoir sur e-campus.

Environnement de travail

Pour ce premier tp qui ne nécessite pas de droits root vous pouvez lancer toutes les commandes depuis un terminal sous votre session linux normale.

Toutefois ce premier Tp est aussi l'occasion d'utiliser les VM mises à disposition pour les TP de sécurité.

1 Utilisation des VM

Le service est disponible sur l'URL suivante :

https://guacamole.pgip.universite-paris-saclay.fr/

C'est un serveur guacamole encapsulé dans une connexion https.

Pour la connexion sur le serveur utiliser les identifiants Université Paris Saclay, (prenom.nom)

Le site donne accès à votre machine virtuelle qui est une version MINT 20 dérivée d'Ubuntu.

Pour se connecter sur la machine virtuelle le compte est mm20, mot de passe guacamole.

Pour avoir accès au presse papier de la VM utliser CTRL-Alt-Shift

Pour les tp avec droit sous root utiliser la commande sudo su qui lance un shell sous root.

L'enseignant a aussi accès à vos VM soit par ssh soit avec l'outil de surveillance veyon.

La sortie de session normale depuis le système est la bonne méthode pour terminer la session. En fin de TP

La fermeture du navigateur ou de l'onglet provoque la fermeture de la session.

Veillez à sauvegarder votre travail en dehors de la VM car une nouvelle VM est mise en place pour chaque TP, ceci de manière à repartir sur une configuration propre.

2 Algorithme de hash

Voici une commande qui liste les algorithme supportés par openssl :

```
#openssl dgst -list
```

A l'aide de la commande time mesurer le temps de calcul pour un hash du fichier *boot*vmlinuz entre un hash blake2512 et sha3-512

Question 1

Quel est le plus rapide, quel ordre de grandeur de la différence ?

On se propose de vérifier la détection du changement du fichier source, pour cela faire :

```
#cp /boot/vmlinuz modif

# echo 1 >>modif
```

Recalculer le hash du fichier modifié et comparer avec le fichier original (à algorithme égal bien évidemment)

Question 2

Quelle conclusion sur les deux résultats?

On va maintenant mettre en évidence les collisons dans md5, pour ce faire on récupère les deux blocs bien connus :

```
d131dd02c5e6eec4693d9a0698aff95c2fcab58712467eab4004583eb8fb7f89
55ad340609f4b30283e488832571415a085125e8f7cdc99fd91dbdf280373c5b
d8823e3156348f5bae6dacd436c919c6dd53e2b487da03fd02396306d248cda0
e99f33420f577ee8ce54b67080a80d1ec69821bcb6a8839396f9652b6ff72a70
```

and

```
d131dd02c5e6eec4693d9a0698aff95c2fcab50712467eab4004583eb8fb7f89
55ad340609f4b30283e4888325f1415a085125e8f7cdc99fd91dbd7280373c5b
d8823e3156348f5bae6dacd436c919c6dd53e23487da03fd02396306d248cda0
e99f33420f577ee8ce54b6708028 0d1ec69821bcb6a8839396f965ab6ff72a70
```

Cette représentation est en hexadécimal, il faut convertir en binaire, par exemple avec l'éditeur binaire jeex.

Pour installer cet éditeur :

```
#apt-get install jeex
```

Pour convertir avec jeex, ouvrir un nouveau fichier puis faire un copié-collé des valeurs dans l'éditeur.

Si on part d'un fichier texte contenant le texte l'éditeur prend un octet par caractère au lieu de lire en hexadécimal.

Supprimer les lignes, puis sauver dans un fichier. Les deux fichiers doivent faire exactement 128 octets.

Vérifier que les deux fichiers sont différents avec cmp

Question 3

Quel est le rang de la première différence ?

Question 4

Vérifier la collision, quelle est sa valeur ?

Question 5

On souhaite vérifier que la concaténation avec ces deux blocs génère aussi une collision, donner un exemple de commandes pour faire la démonstration.

3 Chiffrement symétrique

Utiliser la commande openssl pour chiffrer avec un algorithme symétrique. Pour avoir la liste des possibilités :

```
#openssl enc –help
```

Pour chiffrer dans la suite on utilisera l'algorithme le plus classique chiffrement aes, par exemple avec aes-256-cbc.

Les paramètres du chiffrement sont affichés avec l'option -p.

Le nombre d'itération pour la dérivation de la clé à partir du mot de passe donné par l'utilisateur est spécifié par l'option -iter

l'option -salt qui permet d'ajouter un sel aléatoire pour la génération de la clé est par défaut.

Example de chiffrement

```
$ openssl enc –aes-256-cbc –salt -iter 10 -p  –in /etc/passwd –out passwd.c
```

Pour déchiffrer il faut bien spécifier les même option pour dériver la clé du mot de passe

```
$ openssl enc –d –aes-256-cbc -iter 10 -p –in passwd.c
```

Question 6

Pourquoi 2 chiffrements avec le même mot de passe donnent deux fichiers différents ?

Question 7

Remplacer l'option salt par nosalt, quel est le résultat sur deux chiffrement du même fichier avec le même passwd?

Préparer deux fichiers contenant exactement les caractères suivant :

F1 : 12345678abcdefgh

F2 : abcdefgh12345678

Chiffrer ces deux fichiers en des-ecb et nosalt.

Question 8

Comparer les deux fichiers chiffrés.Que constatez-vous ?

Question 9

Donner une explication, en particulier pour le choix des options de chiffrement.