IUT Charlemagne
Département Informatique

LP ASRALL
11 février 2015

Chiffrement, crackage et mot de passe

L'objectif est de se familiariser avec les algorithmes de chiffrement, le stockage des mots de passe sous forme « chiffré », et de tester leur robustesse en essayant de les cracker avec des attaques à base de dictionnaire..

1. Les algorithmes de chiffrement

Le chiffrement est au cœur de la sécurité informatique. C'est pourquoi il est nécessaire de le comprendre et d'en appréhender toutes ses facettes. Cette partie concernera l'implémentation d'un algorithme dans un langage de votre choix.

1.1. Implémenter dans le langage qu'il vous plaira Diffie Hellman. Vérifier son fonctionnement.

2. Hash de mots de passe

Les mots de passe sont stockés sous forme « chiffrés » (un hash en réalité). Dans la vérification d'un mot de passe (au login d'un utilisateur), le mot de passe fourni par l'utilisateur est chiffré et c'est la chaîne chiffré qui est comparée.

Ainsi, l'administrateur (root) n'a pas accès au mot de passe en clair. L'inconvénient est qu'il ne peut pas en vérifier la robustesse.

2.1. Les mots de passes encryptés sont dans le fichier /etc/shadow. Pourquoi ne sont-il pas dans /etc/passwd ?

2.2. Coder dans le langage de votre choix un algorithme permettant de générer le hash d'un mot de passe en SHA 512 pour l'inclure directement dans /etc/shadow.

La chaine doit être de la forme 6<salt>$<encrypted> par exemple.

La fonction crypt() sera utilisée.

Vérifiez en la conformité avec votre mot de passe (que le résultat obtenu est bien identique à celui du fichier /etc/shadow)

3. Cracker des mots de passe

John the ripper est un utilitaire pour cracker les mots de passe. Il se base sur des dictionnaires et des règles. Il est utilisé pour les pirates, mais aussi par les services informatiques afin de vérifier la robustesse des mots de passe. Vous allez essayer de cracker des mots de passe.

3.1. Installation et utilisation de jtr sur les fichiers /etc/passwd et /etc/shadow

Installez JTR (john the ripper) et utiliser le sur les fichiers systèmes. Quelles sont les manipulations à faire pour lui faire comprendre de travailler sur les fichiers en question ?

3.2. Utilisation sur un fichier

Utilisez votre programme précédent pour générer un fichier de la forme :

login:6<salt>$<hash>

Utilisez jTP sur ce fichier avec divers mots de passe simples ('toto', etc...). Vérifiez qu'il les trouve.

3.3. Intégrer une nouvelle règle

JTR effectue des transformations sur les mots des dictionnaires. Créer un nouveau dictionnaire contenant uniquement le mot « secret » et mettez dans le fichier de mot de passe le mot s3cr3t (en remplacant les « e » par des « 3 »). Vérifier qu'il ne le trouve pas (en utilisant votre dictionnaire), puis rajouter la règle de transformation adéquat pour qu'il le trouve.

4. Utilisation de pam_cracklib

Tester la robustesse des mots de passe des utilisateurs (par rapport à des dictionnaires) au moment du changement par ce dernier est plus efficace que de les tester à postériori.

En effet, c'est le seul moment ou root a accès en clair au mot de passe de l'utilisateur.

Nous allons mettre en place cracklib, qui est un système de vérification de robustesse des mots de passe. Il s'inclue dans PAM.

4.1. Installation de pam-cracklib

Regarder la configuration pam pour passwd, familiarisez vous avec pam. Installez la lib pam cracklib

4.2. modification de la configuration de cracklib

Augmentez la longueur des mot de passe à minimum 10 caractères, et tester un changement de mot de passe d'un utilisateur.

4.3. Rajouter un dictionnaire à cracklib

Choisir un mot de passe valide.

Ajouter un dictionnaire à pam cracklib contenant des mots nouveaux de votre choix dont le mot de passe valide testé précédemment. Vérifiez au prochain changement de mot de passe qu'ils sont bien pris en compte.

5. Chiffrer et déchiffrer avec RSA (via openssl)

Dans cette partie, nous allons chiffrer et déchiffrer des fichiers avec du chiffrement asymétrique. Pour cela, vous allez créer une bi-clef RSA, puis l'utiliser pour chiffrer et déchiffrer un fichier. Tout sera basé sur openssl, à éventuellement installer sur votre poste.

5.1. Générer des bi-clefs

En utilisant la commande openssl genrsa, génerez une bi-clef de la longueur qu'il vous plaira, dont la clef secret sera protégée par une mot de passe (passphrase). N'oubliez pas d'en sortir la clef publique car par défaut tout est dans le même fichier.

5.2. Chiffrer un fichier (de quelques octets) avec cette clef publique et déchiffrez le avec la clef privée.

5.3. Retenter l'opération avec un fichier de grande taille. Que se passe-t-il ? Remédiez en intégrant un chiffrement symétrique, toujours avec l'utilitaire openssl.

Le chiffrement n'est pas possible car trop couteux. Il faut « mixer » du chiffrement symétrique (par exemple AES) pour le contenu du fichier, avec une clef généré aléatoirement qui sera chiffrée avec RSA.

On ajoutera les commandes de chiffrement/déchiffrement symétrique suivantes :

6. Bibliographie

Perl, bash, john the ripper, openssl, pam, cracklib...