

SSI - Laboratoire 1

Craquage de mots de passe

12.10.2011

IL-2012

Brahim Lahlou

Numa Trezzini

1. Résumé	III
2. Introduction	1
3. But	1
4. Mise en œuvre	1
4.1 Obtention des empreintes des mots de passe	1
4.1.1 Question 1	2
4.2 John The Ripper	2
4.2.1 Question 2	3
4.3 Rainbow Tables	4
4.3.1 Question 3	5
4.4 Question 5	5
5. Conclusion.....	6
6. Références.....	7
7. Liste des symboles de référence	7
8. Liste des figures.....	7

1. Résumé

Pour ce rapport, nous avons exploré et testé les manières basiques de craquer des mots de passe. Nous avons ainsi essayé de craquer des mots de passe premièrement à l'aide d'une attaque dictionnaire et force brute et deuxièmement une attaque par tables rainbow. Nous avons testé l'efficacité de ces méthodes et les avons ensuite comparées.

2. Introduction

Dans le monde informatique, l'une des principales protections contre les intrusions et autres tentatives d'accès frauduleuses à des machines tierces sont les mots de passe. Ils jouent un rôle premier dans la sécurité des systèmes d'information. Pour pouvoir faire usage de ces mots de passe, il faut bien que l'ordinateur puisse les stocker, sous une forme ou une autre, afin de les vérifier. L'idée d'exploiter cette contrainte à des fins malveillantes ne date pas d'hier, et les techniques y relatives sont, si ce n'est plus nombreuses, au moins plus complexes. Les techniques de protection se sont elles aussi développées, mais le « bon sens » de l'utilisateur est toujours un facteur primordial dans la sécurité de sa machine : les mots de passe doivent être « forts », c'est-à-dire résister, le plus longtemps possible, aux attaques et au décryptage.

Afin de créer des mots de passe forts, il convient de connaître les mécanismes de craquage sous-jacents. Ceci nous permettra de comprendre les risques et ainsi d'anticiper les menaces. Nous allons en premier lieu présenter le but de ce laboratoire, avant de fournir une explication sur la façon de le réaliser. Nous répondrons ensuite aux questions posées, analyserons les réponses et nous terminerons par une conclusion sur les méthodes de craquage.

3. But

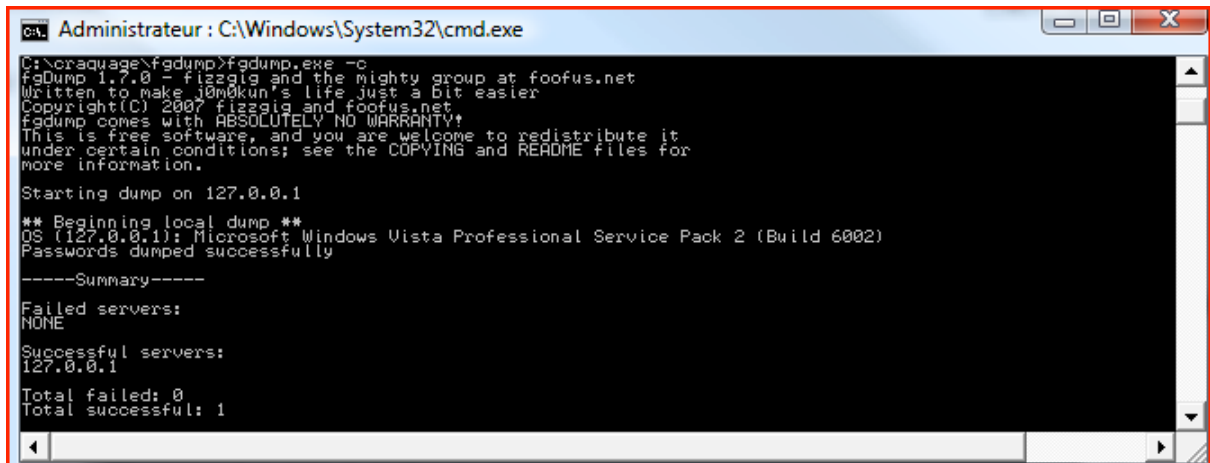
Le but de ce laboratoire est de tester nous-mêmes quelques-unes des techniques de craquage de mots de passe. Par l'intermédiaire du programme **John The Ripper (JtR)** (plus loin dans le rapport), nous allons tester une attaque par « dictionnaire » et une attaque par « force brute ». Le programme **rainbowcrack** nous donnera un aperçu de la façon d'utiliser des tables rainbow (tables arc-en-ciel) pour trouver des mots de passe.

4. Mise en œuvre

Les mots de passe sont stockés sur l'ordinateur sous forme de chaîne de caractères hashée (cryptée), afin d'assurer une première protection : les mots de passe ne sont pas stockés « en clair » et ne sont donc pas lisibles par l'humain. De plus, ils sont situés dans des emplacements protégés du système. L'opération de hachage est irréversible.

4.1 Obtention des empreintes des mots de passe

La première étape consiste donc à récupérer ces empreintes. Selon l'OS utilisé, la méthode sera différente, car les protections ne sont pas pareilles : dans notre cas, nous avons utilisé **pwdump** pour Windows. Pour réaliser l'ensemble du laboratoire, deux comptes fictifs seront créés sur nos machines : Paul et Roger, avec comme mots de passe *sawrin* et *amazing*, respectivement.



```
c:\craquage>fgdump>fgdump.exe -c
fgDump 1.7.0 - f1zzag and the mighty group at foofus.net
Written to make j0n0kun's life just a bit easier
Copyright(C) 2007 f1zzag and foofus.net
fgdump comes with ABSOLUTELY NO WARRANTY!
This is free software, and you are welcome to redistribute it
under certain conditions; see the COPYING and README files for
more information.

Starting dump on 127.0.0.1

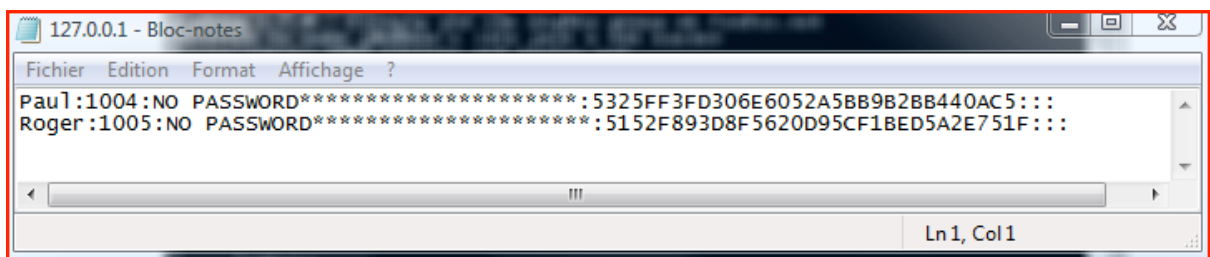
** Beginning local dump **
OS (127.0.0.1): Microsoft Windows Vista Professional Service Pack 2 (Build 6002)
Passwords dumped successfully

-----Summary-----
Failed servers:
NONE
Successful servers:
127.0.0.1
Total failed: 0
Total successful: 1
```

Figure 1 - Exécution de fgdump.exe

4.1.1 Question 1

- Quel est l'OS utilisé :
L'OS utilisé pour réaliser ce laboratoire est Windows Vista Professional SP2 (build 6002).
- Votre fichier contient-il des empreintes de type LM, NTLM, MD5 ou autre :
Notre fichier contient des empreintes de type NTLM.
D'une part parce que Windows Vista n'utilise que des empreintes de ce type. D'autre part, on remarque en observant la figure 2 que chaque compte se voit attribuer un numéro identificateur et deux suites de 32 chiffres hexadécimaux représentant le mot de passe hashé. La première suite correspond à l'empreinte LM et la seconde à NTLM.
Et dans notre cas, la première suite de chiffres est vide, ce qui indique l'utilisation de NTLM.



```
Paul:1004:NO PASSWORD*****:5325FF3FD306E6052A5BB9B2BB440AC5:::
Roger:1005:NO PASSWORD*****:5152F893D8F5620D95CF1BED5A2E751F:::
```

Figure 2 - Fichier 127.0.0.1.pwdump

4.2 John The Ripper

JtR fonctionne selon plusieurs méthodes, détaillées ci-dessous. JtR est normalement capable, avec assez de temps à disposition, de trouver n'importe quel mot de passe. Pour réaliser la comparaison avec l'empreinte, JtR hashé d'abord le mot de passe en cours de traitement selon la méthode requise par l'algorithme utilisé par le système hôte.

```
C:\craquage\laboCraquageJohnNTLM\john-multipatch>john.exe --format=NT C:\craquage\fgdump\127.0.0.1.pwdump
Loaded 2 password hashes with no different salts (NT MD4 [TridgeMD4])
amazing (Roger)
sawrin (Paul)
guesses: 2 time: 0:00:03:30 (3) c/s: 1750K trying: sawrin
```

Figure 3- Exécution de john.exe

4.2.1 Question 2

- Comment JtR fonctionne-t-il ? Par dictionnaire ? Par force brute ? Autre ? Dans quel ordre ? (Donner des explications sur chaque méthode)

JtR effectue, dans l'ordre, les actions suivantes : attaque par mots de passe simple (« single crack » [JTR]), par attaque dictionnaire et finalement par « force brute ».

1. Attaque par mots de passe simple: JtR commence par tester des mots de passe dérivés des informations disponibles sur l'utilisateur (nom, prénom, username, etc.). Dans la même lancée, JtR teste les mots de passe les plus couramment rencontrés (plus de détails sur ce sujet à la question suivante). Cette étape du craquage contient aussi plusieurs modifications des mots de passe de base, avec notamment l'ajout de préfixes et suffixes, le remplacement de lettres par des chiffres ou la duplication des mots.
2. Attaque dictionnaire : L'attaque dictionnaire consiste à tester comme mot de passe tous ceux d'une liste constituée à partir d'un dictionnaire. Les mots peuvent simplement être listés dans un fichier texte. Cette étape peut être relativement longue, surtout si le programme teste des dictionnaires de plusieurs langues. De plus, les modifications indiquées au point précédent sont aussi réalisées à cette étape.
3. Attaque par « force brute » : Il s'agit de tester toutes les combinaisons possibles en fonction du nombre de caractères et du jeu défini¹. Par exemple, un mot de passe de 8 minuscules possède 26^8 (≈ 208 milliards !) combinaisons possibles. Evidemment, plus le nombre de caractères permis augmente, plus le nombre de combinaisons augmente. Les mots de passe ne sont cependant pas toujours limités dans les caractères à utiliser. Ainsi, le fichier *dumb32.conf* de JtR nous dit : « Generic implementation of "dumb" exhaustive search of FULL Unicode and an arbitrary charset. Default is to try *all* allocated characters (there's 109070 of them). Even if a fast format can exhaust two characters in one hour, three characters would take 12 years... » [JTR].

Cette méthode permet de trouver sans faille un mot de passe, mais peut prendre un temps virtuellement infini. La force brute se doit donc d'être un dernier recours.

- Que contient le fichier **password.lst**?

Ce fichier contient, comme il l'indique lui-même, les mots de passe les plus communs sur des systèmes UNIX : « This list is based on passwords most commonly seen on a set of Unix systems in mid-1990's [...]. It has been revised to also include common website passwords from public lists » [JTR]. Ces mots de passe sont de plus triés pour que leur ordre

¹ Le jeu de caractères peut être par exemple les minuscules, les majuscules ou les chiffres. Il est aussi possible d'inclure les caractères asiatiques et accentués

corresponde au nombre d'occurrences décroissantes. Un mot de passe statistiquement plus souvent utilisé sera donc testé en priorité.

- Pourquoi le mot de passe de 7 caractères a-t-il été trouvé plus rapidement que celui de 6?

Le mot de 7 caractères est un mot du dictionnaire anglais (*amazing*), tandis que l'autre ne l'est pas (*sawrin*). Le second n'apparaît pas non plus dans *password.lst*, il faut donc le trouver avec de la force brute. Etant donné que cette méthode est la dernière utilisée, et surtout la plus longue, trouver un mot de passe avec plus de caractères mais « connu² » sera plus rapide, car effectué avant. Il aurait été éventuellement possible de trouver le second mot de passe plus rapidement avec de l'ingénierie sociale, permettant de mettre en place un dictionnaire personnalisé³.

- Votre mot de passe est-il facilement craquable avec John? ☺

Nos mots de passe ne sont ni des mots de passe simples/communs, ni des mots du dictionnaire. JtR se voit donc contraint de passer à la force brute pour tester nos mots de passe. En plusieurs heures de décryptage, JtR n'a produit aucun résultat. Nos mots de passe sont donc relativement fiables. Pour des questions de confidentialité, nous ne souhaitons pas donner les résultats du dump et de la tentative de craquage de nos mots de passe personnels.

4.3 Rainbow Tables

Les rainbow tables (tables arc-en-ciel) sont des tables de mots de passe hashés préconstruites. Il n'est ainsi pas nécessaire de calculer les empreintes. Il suffit de la comparer avec son entrée dans la table pour avoir une correspondance et trouver le mot de passe. Nous pouvons constater grâce à la figure suivante que le temps mis par rainbowcrack pour trouver les deux mots de passe est environ 5 fois plus rapide que avec JtR (43.27 sec contre 3min30 avec JtR).

```
C:\craquage\rainbowcrack>rcrack.exe ntlm*.rt -l C:\craquage\fgdump\127.0.0.1-2.pwdump
ntlm_loweralpha#1-7_0_7000x8000000_oxid#000.rt:
128000000 bytes read, disk access time: 4.54 s
verifying the file...
searching for 2 hashes...
plaintext of 5152f893d8f5620d95cf1bed5a2e751f is amazing
plaintext of 5325ff3fd306e6052a5bb9b2bb440ac5 is sawrin
cryptanalysis time: 43.27 s

statistics
-----
plaintext found:      2 of 2 (100.00%)
total disk access time: 4.54 s
total cryptanalysis time: 43.27 s
total chain walk step: 8124691
total false alarm:    7939
total chain walk step due to false alarm: 38471936

result
-----
5152f893d8f5620d95cf1bed5a2e751f  amazing  hex:616d617a696e67
5325ff3fd306e6052a5bb9b2bb440ac5  sawrin   hex:73617772696e
```

Figure 4 - Exécution de rcrack.exe

² Dans le sens de « apparaissant dans un dictionnaire »

³ Selon nos recherches, « sawrin » est un type/marque de sous-vêtement féminin...

4.3.1 Question 3

- Pour quelle raison l'utilisation de sel rend les attaques par les tables "rainbow" inefficaces ?

Ajouter un sel à un mot de passe modifie l'empreinte générée par les algorithmes de cryptage. Saler les mots de passe oblige donc les tables rainbow à multiplier le nombre d'empreintes possibles pour chaque mot de passe. Créer des tables rainbow prend déjà beaucoup de temps, mais multiplier les empreintes avec un sel d'un octet (par exemple) multiplierait par 2^8 le nombre de tables rainbow. Parcourir, mais surtout créer et stocker autant de tables serait particulièrement long et fastidieux.

- Malgré cette inefficacité, expliquer quels critères les tables devraient remplir pour être en mesure de craquer des empreintes avec un sel ?

Il suffirait de créer autant de tables qu'il existe de sels différents, contenant chacune l'empreinte spécifique à un sel.

- Considérons une table de 2 GB permettant de craquer des empreintes sans sel. Un sel de 4 bits est ajouté à chacun des mots de passe. Estimer la taille de la nouvelle table permettant de craquer les mots de passe avec le sel. (Détailer les calculs)

Avec un sel de 4 bits, nous pouvons en produire $2^4 = 16$ différents. Si une empreinte différente doit être générée pour chaque sel, nous aurons donc 16 tables de 2GB. La taille totale sera donc $16 \times 2\text{GB} = 32\text{GB}$.

4.4 Question 5

	John the Ripper	Rainbowcrack	Explications
Temps de craquage	-	+	Si c'est un mot du dictionnaire alors JtR est rapide, autrement rainbowcrack est plus efficace.
Temps de préparation avant craquage	+	-	Il faut générer les table pour rainbowcrack alors que pour JtR, il n'y pas de besoin de préparation, à part le dump des empreintes.
Craquage sur tout les OS	+	-	JtR est multiplateforme mais rainbowcrack aura des problèmes avec les OS qui utilisent des sels pour les mots de passes hachés.

5. Conclusion

Au cours de ce laboratoire, nous avons pu expérimenter avec les systèmes de mots de passe et leur cryptage. Nous avons aussi eu un aperçu de la façon de les craquer, et ce grâce à deux techniques distinctes : le craquage par essai, à l'aide de John The Ripper et le craquage par comparaison à l'aide de rainbowcrack. Les méthodologies utilisées par ces deux programmes ont été analysées et comparées. Nous avons ainsi pu tester quelques façons de craquer des mots de passe, ainsi que leurs avantages et désavantages.

Connaitre l'emploi par la machine des mots de passe est un élément essentiel à la compréhension de la sécurité des systèmes informatiques. En effet, les mots de passe sont le plus grand rempart contre les intrusions indues. Nous avons pu constater que le choix du mot de passe est un facteur très important dans la sécurité. En effet, en fonction de sa complexité et du nombre et diversité de caractères utilisés, le craquage sera plus ardu. Les systèmes et les algorithmes se chargent, eux, de pallier aux faiblesses face aux programmes de craquage, en introduisant notamment un sel. C'est cependant à l'utilisateur qu'appartient le choix du « niveau » de protection des ses données avec le choix d'un mot de passe adapté.

6. Références

1. Collectif, Sources *John The Ripper*, [en ligne] <http://www.openwall.com/john/>
2. Mast Patrick, *SSI – Craquage de mots de passe*, 2011, HEIG-VD
3. Buchs Christian, *Cours Sécurité des Systèmes d'Information*, 2011, HEIG-VD

Dernière consultation des sites : 21.10.2011

7. Liste des symboles de référence

- [JTR] : Sources John The Ripper

8. Liste des figures

- Figure 1 : Résultat de l'exécution de pwdump sur notre système de test
- Figure 2 : Contenu du fichier créé par pwdump
- Figure 3 : Résultat de l'exécution de JtR
- Figure 4 : Résultat de l'exécution de rainbowcrack