

Cassage et durcissement des mots de passe

Première partie : Windows

Denis Ducamp / Hervé Schauer Consultants

Mars 2002

Cet article aborde un certain nombre de problèmes au sujet des mots de passe et leurs applications dans le milieu Windows. Un second article présentera cela dans le milieu Unix. À chaque fois, les aspects système, applications et réseau sont décrits. **Introduction**

Le présent article est composé de 7 parties :

1. Les notions de chiffrement et de stockage d'un mot de passe : les choses à savoir tout au long de ces deux articles.
2. Les méthodes de cassage : parce que, pour savoir protéger ses mots de passe, il faut savoir comment les pirates s'y prennent pour les casser.
3. La localisation des empreintes sous Windows : pour savoir où l'attention des administrateurs doit se porter.
4. Les différents algorithmes de chiffrement sous Windows : comment, sur les systèmes et les réseaux Windows, les mots de passe sont chiffrés.
5. Les logiciels de cassage contre Windows : transformer en protection les meilleurs logiciels, avant que les pirates ne les emploient pour vous attaquer.
6. La protection des mots de passe sous Windows : comment protéger ses mots de passe aux niveaux système, applications et réseau.
7. Le durcissement des mots de passe sous Windows : comment améliorer la qualité des mots de passe choisis par l'utilisateur.

1 Les notions de chiffrement et de stockage d'un mot de passe

1.1 Stockage d'un mot de passe

Le mot de passe choisi par l'utilisateur est haché et le résultat est enregistré dans la base des mots de passe à côté du login correspondant. Le mot de passe haché est appelé empreinte, haché, ou *hash* en anglais.

Le terme "chiffré" couramment utilisé est un abus de langage. La fonction utilisée est en fait une fonction de hachage dont une propriété est d'être à sens unique (*one way*), afin que la donnée en clair entrée à l'origine ne soit pas récupérable à partir du résultat. Certaines fonctions de hachage utilisées sont fondées sur des fonctions de chiffrement, d'où l'abus puisqu'une fonction de chiffrement a pour objectif de rendre inintelligible le résultat à qui ne possède pas le secret (la clé) nécessaire.

1.2 Chiffrement d'un mot de passe

Comme vu précédemment, le résultat doit être non-réversible. Si le mot de passe est chiffré avec une clé secrète, alors le mot de passe peut être obtenu en déchiffrant le chiffré par quiconque connaissant la clé. Ce type de sécurité, nommé sécurité par l'obscurité, n'est pas viable car à court terme, les développeurs possèdent une porte dérobée et à moyen terme, le niveau de sécurité est précaire puisque tout secret est découvert ou dévoilé toujours plus tôt que prévu.

Pour obtenir un résultat non-réversible, le mot de passe sert comme clé pour chiffrer une chaîne de caractères constante. Dans ce cas, si l’empreinte est volée, alors il faut deviner le mot de passe, s’en servir comme clé afin de déchiffrer l’empreinte et obtenir la chaîne connue.

La solution est donc d’utiliser le mot de passe comme clé pour chiffrer une chaîne connue mais deux utilisateurs avec le même mot de passe possèdent alors la même empreinte. Une solution est d’utiliser un diversifiant (graine) afin que les deux empreintes soient différentes. Cette graine (ou piment) est utilisée comme paramètre afin de changer le comportement de la fonction utilisée, mais la même graine doit toujours donner le même résultat.

La graine est enregistrée en clair dans l’empreinte. La connaissance de celle-ci n’est pas une vulnérabilité et est nécessaire au processus de vérification du mot de passe.

Il est en revanche important que la même graine ne soit pas utilisée par plusieurs utilisateurs, un pirate n’ayant à calculer qu’une seule fois un mot de passe pour essayer de casser plusieurs empreinte simultanément.

1.3 Vérification d’un mot de passe

Pour vérifier l’authentification de l’utilisateur, le serveur a accès au mot de passe en clair fourni par l’utilisateur et à l’empreinte enregistrée dans la base des mots de passe. L’empreinte n’étant pas déchiffrable, la comparaison des mots de passe en clair est impossible. Le serveur doit donc hacher le mot de passe du client et vérifier que le résultat correspond à l’empreinte sauvegardée.

Pour hacher le mot de passe, le serveur récupère la graine et l’utilise pour calculer le résultat qui est comparé à l’empreinte enregistrée.

2 Les méthodes de cassage

Avant de voir les différentes méthodes de chiffrement, nous allons nous attarder sur les méthodes de cassage utilisées, celles-ci étant découpées en trois catégories : ingénierie sociale, dictionnaires et force brute.

2.1 Ingénierie sociale

L’ingénierie sociale est l’utilisation de données personnelles en relation avec le propriétaire du compte. Pour commencer, la première donnée est tout simplement le login : sur plusieurs dizaines de comptes, il est rare qu’il n’y en ait pas au moins un qui ait son login comme mot de passe.

Les données suivantes sont simplement le nom et le prénom de l’utilisateur. Cette information est généralement enregistrée dans la base des utilisateurs et donc librement accessible.

Ensuite, il s’agit d’utiliser des informations qui ne sont pas dans la base des utilisateurs, ce qui demande d’obtenir d’une autre façon ces données personnelles. Celles-ci sont d’abord les noms des proches de l’utilisateur. De bons exemples sont pour une femme les prénoms de ses enfants alors que pour un homme, ce sont les prénoms de sa femme, de sa secrétaire, de sa maîtresse (ho !). Les passions et le métier de l’utilisateur sont également une bonne source de mots de passe possibles.

Enfin, comme aujourd’hui chacun doit se rappeler d’un grand nombre de données, certaines sont souvent réutilisées dans d’autres cadres. Ainsi, numéro de sécurité sociale, immatriculation, téléphone, adresse, date de naissance, etc. sont régulièrement choisis comme mot de passe. Des personnes utilisent même leur code de carte bleue comme mot de passe... Bien sûr, personne ne connaît le code de quelqu’un d’autre, mais le pirate peut essayer les dix mille

combinaisons possibles pour obtenir les quatre chiffres magiques, qui risquent de ne correspondre qu'au digicode de l'immeuble de l'utilisateur.

Nous ne parlerons pas ici du cas où le pirate se fait passer pour un administrateur système ou un supérieur et demande le mot de passe et le compte de l'utilisateur pour soi-disant effectuer des réparations. Il va sans dire que non seulement d'un point de vue technique, cela ne demande aucune compétence, mais qu'en plus, aucun administrateur n'a besoin de vous demander votre mot de passe, quel que soit le problème auquel il a affaire.

2.2 Dictionnaires

La source suivante de mots de passe possibles est constituée des dictionnaires. Il existe de nombreux types de dictionnaires et il est donc important de les utiliser dans un ordre optimal. En général, les mots courants de la langue natale de l'utilisateur sont la meilleure source, surtout dans le cas de personnes expatriées. Ensuite, les prénoms et noms de même origine fonctionnent bien.

Les dictionnaires spécialisés dans les passions et le métier de l'utilisateur sont d'une aide précieuse. Par exemple, un chimiste peut utiliser le nom d'une molécule, en pensant qu'ils sont peu nombreux à le connaître, mais celui-ci est forcément référencé dans un dictionnaire.

D'autres dictionnaires avec des noms de personnages et d'acteurs sont de bonnes sources. Imaginez-vous un jeudi matin alors que vous souhaitez vous connecter à votre serveur et celui-ci vous force à changer de mot de passe... comme la veille, vous êtes allé voir le dernier James Bond qui vient de sortir, il y a de fortes chances que vous utilisiez bond007 comme mot de passe.

Usuellement, les derniers dictionnaires sont ceux contenant des vocabulaires propres à des livres, films, séries, jeux, etc. Plusieurs sites sur Internet regroupent de nombreuses listes de mots de toutes origines comme <http://ftp.ox.ac.uk/pub/wordlists/> et <http://ftp.cerias.purdue.edu/pub/dict/wordlists/>

Finalement il est possible de générer un dictionnaire à partir de la liste des mots de passe déjà cassés.

2.2.1 Transformations de dictionnaires

Il est ensuite utile de réessayer tous les mots testés, après les avoir modifiés afin de découvrir des mots de passe plus complexes. Tout d'abord, il s'agit de mettre zéro, une ou toutes les lettres en majuscules, puis de les renverser ou de les dupliquer.

Une transformation souvent utilisée est de suffixer ou préfixer le mot avec un chiffre ou un caractère de ponctuation, le chiffre 1 ajouté après un mot étant la transformation la plus usuelle. Enfin, il arrive aussi régulièrement que l'utilisateur utilise la transformation "h4x0r" c'est-à-dire qu'il substitue une ou plusieurs lettres par un chiffre ou un caractère qui y ressemble, comme i et l par 1 ou |, e par 3, a par 4 ou @, s et z par 5, t par 7, o par 0, etc.

2.2.2 Dictionnaires pré-calculés

Dans le cas où la méthode de chiffrement qui est attaquée n'utilise pas de graine, les dictionnaires pré-calculés sont exploitables. Puisque chaque mot de passe ne peut être chiffré que d'une seule façon, alors l'attaquant chiffre tous ses dictionnaires qu'il enregistre dans une base de données. L'avantage de cette technique est qu'il suffit alors de rechercher le mot de passe haché pour voir immédiatement s'il est cassé.

Les deux contraintes sont d'une part une phase de préparation qui est longue comparée au temps gagné ultérieurement lors du cassage à proprement parler et d'autre part, une utilisation importante de mémoire de masse puisque les

dictionnaires pré-calculés sont incompressibles contrairement aux dictionnaires standards.

Une méthode pour concilier gain de temps et économie d'espace disque, est de n'enregistrer qu'une image du mot de passe haché dans la base. La recherche dans la base donne alors de nombreux mots de passe potentiels qu'il faut alors chiffrer pour savoir si le mot de passe est cassé. La phase de préparation prend autant de temps et la phase de cassage est bien plus longue, mais le temps gagné par rapport à la méthode classique vaut la dépense réduite en espace disque.

2.3 Force brute

La force brute, également appelée attaque exhaustive, est la dernière méthode générale à utiliser lorsque toutes les autres ont échoué. Il s'agit de tester les unes après les autres toutes les combinaisons possibles d'un ensemble de caractères. Cela peut être les lettres minuscules, puis les minuscules et les majuscules, puis les minuscules et les chiffres, etc. Il est facilement compréhensible que suivant les algorithmes de chiffrement, cette méthode n'est pas réalisable dans un temps raisonnable si le mot de passe est suffisamment solide.

Une variante de cette technique est la force brute intelligente. Cet adjectif est contraire au principe de la méthode mais parfaitement compréhensible : il s'agit de générer des mots de passe potentiels qui ressemblent aux mots de passe qui ont déjà été cassés. Pour cela, des statistiques sont récoltées : longueurs, suites de caractères, etc. et utilisées lors de la génération "aléatoire" des mots de passe.

Une utilisation possible de cette méthode est d'effectuer des statistiques sur un dictionnaire afin de générer des mots qui ressemblent aux mots de la langue vivante correspondante.

3 La localisation des empreintes sous Windows

Une empreinte est le résultat du hachage d'un mot de passe, donc une transformation par un algorithme non-réversible. Cette opération enregistre dans le système une "image" du mot de passe et empêche quiconque d'accéder au mot de passe en clair.

Au niveau du système, les mots de passe Windows NT sont enregistrés dans une partie de la base de registres (base de données binaire renfermant toutes les données de configuration du système et des applications) nommée SAM. Celle-ci se trouve dans le fichier `$windir$\system32\config\sam` mais une copie de sauvegarde par "`rdisk /s`" plus ou moins à jour se trouve dans le fichier `$windir$\repair\sam_` et éventuellement sur la disquette de réparation. Une entrée utilisateur de la SAM est constituée :

- du RID = le numéro d'identification unique de l'utilisateur dans le domaine ou sur le système,
- de son nom de connexion,
- de son nom complet,
- de commentaires,
- de son répertoire principal et
- de ses empreintes LanMan et NTLM.

Les empreintes ne sont en fait pas enregistrées telles quelles dans la SAM mais obscurcies par du chiffrement DES. Chaque moitié est chiffrée avec des valeurs calculées à partir de fonctions du RID de l'utilisateur. L'objectif de ce chiffrement est d'empêcher deux utilisateurs ayant le même mot de passe d'avoir une même entrée dans le fichier. Ceci ne protège pas de la faiblesse de l'algorithme de calcul des empreintes, toute disquette de sauvegarde doit impérativement être hors de portée de personnes non sûres.

Voir **Figure 1** : *Stockage d'un mot de passe dans la SAM*

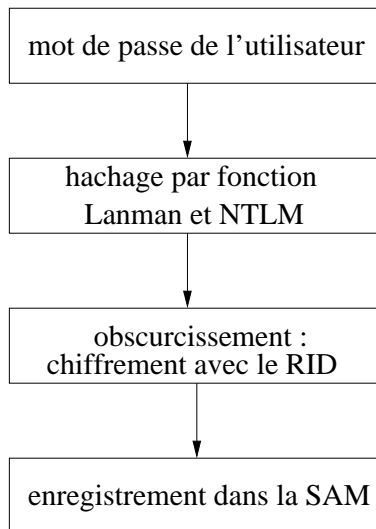


FIG. 1 – Stockage d'un mot de passe dans la SAM

Pour plus d'informations, consultez les sources du programme "Offline NT Password & Registry Editor" <<http://home.eunet.no/~pnordahl/ntpasswd/>> par Petter Nordahl-Hagen.

Sur les contrôleurs de domaines Windows 2000 et XP, les mots de passe sont enregistrés suivant les mêmes algorithmes, mais dans l'Active Directory et non plus dans la SAM.

Les fichiers *.pwd contenant les comptes et mots de passe des utilisateurs déclarés auprès de *frontpage* sont aussi extrêmement sensibles. En effet, il suffit de changer le nom d'un script de *frontpage* dans un navigateur par celui d'un des fichiers d'authentification pour en obtenir à distance le contenu. Nous verrons plus tard comment protéger ces fichiers.

Avertissements :

- De nombreux logiciels clients sous Windows (messagerie, web, etc.) sauvegardent les mots de passe des utilisateurs de façon réversible et accessible à tous dans la base de registres ou dans des fichiers utilisateurs.
- Les problèmes engendrés par ces mots de passe, et notamment leurs résolutions, dépassent le cadre de cet article.

4 Les différents algorithmes de chiffrement sous Windows

L'algorithme utilisé dépend fortement du contexte, suivant que l'authentification se passe au niveau système ou au niveau réseau.

4.1 Les différents algorithmes de chiffrement sous Windows NT

Deux algorithmes sont utilisés pour enregistrer de façon non-réversible les mots de passe des utilisateurs déclarés au niveau du système : LanMan et NTLM.

L'algorithme LanMan a été imposé par le protocole SMB utilisé dans les services réseau Windows. Le hachage repose aussi sur le chiffrement d'une chaîne connue par une clé générée à partir du mot de passe.

Ici, le mot de passe est tout d'abord tronqué à 14 caractères. S'il n'est pas assez long alors il est complété avec des

caractères nuls. Puis il est mis en majuscules et divisé en deux parties de 7 caractères. Chaque partie est indépendamment utilisée comme clé de chiffrement DES à 56 bits pour chiffrer la chaîne "KGS !@#%". Les deux résultats de 8 octets chacun sont concaténés pour donner l'empreinte LanMan de 16 caractères.

Les vulnérabilités de cet algorithme sont multiples. Tout d'abord les attaques sont limitées à 2 mots de 7 caractères et aucune différence n'est faite entre majuscules et minuscules. Les tests sont alors limités à $2 \times 69^7 = 1,49E13$ combinaisons au lieu des $96^{14} = 5,65E27$ auxquelles s'attend l'utilisateur, rendant aujourd'hui possible une recherche exhaustive en quelques mois sur n'importe quelle machine personnelle.

De plus, aucune graine n'étant utilisée, les attaques par dictionnaire pré-calculé sont réalisables. Enfin, une comparaison de la seconde partie de l'empreinte par rapport à une certaine constante montre si le mot de passe a une longueur maximale de 7 caractères.

L'algorithme NTLM a été introduit dans Windows NT 3.5 côté postes serveurs et dans Windows Millenium côté postes clients. Le mot de passe est tout d'abord limité à 128 caractères puis passé en Unicode, c'est-à-dire qu'après chaque caractère un caractère nul est inséré. Cette chaîne est passée à la fonction de hachage MD4 pour donner l'empreinte NTLM de 16 caractères.

Ce nouvel algorithme accepte donc des mots de passe de 128 caractères, mais surtout différencie majuscules et minuscules et supporte bien plus de caractères comme les lettres accentuées. Malheureusement, ici non plus, aucune graine n'est utilisée pour diversifier les empreintes.

Des problèmes ont rapidement été trouvés sur la fonction MD4, entraînant la création de MD5 et la découverte d'autres problèmes dans MD4. Néanmoins, attaquer un haché NTLM demande bien plus de temps que d'effectuer une recherche exhaustive sur l'empreinte LanMan. Il est de plus suspecté que le fait qu'un octet sur deux soit nul rende le haché MD4 plus vulnérable à la cryptanalyse, même s'il n'y pas aujourd'hui de façon d'exploiter cela. En fait, comme nous le verrons plus tard, il est des cas où la seule connaissance de l'empreinte suffit à s'authentifier à la place de l'utilisateur, le cassage du mot de passe étant alors un luxe.

Voir **Figure 2** : *Authentification locale*

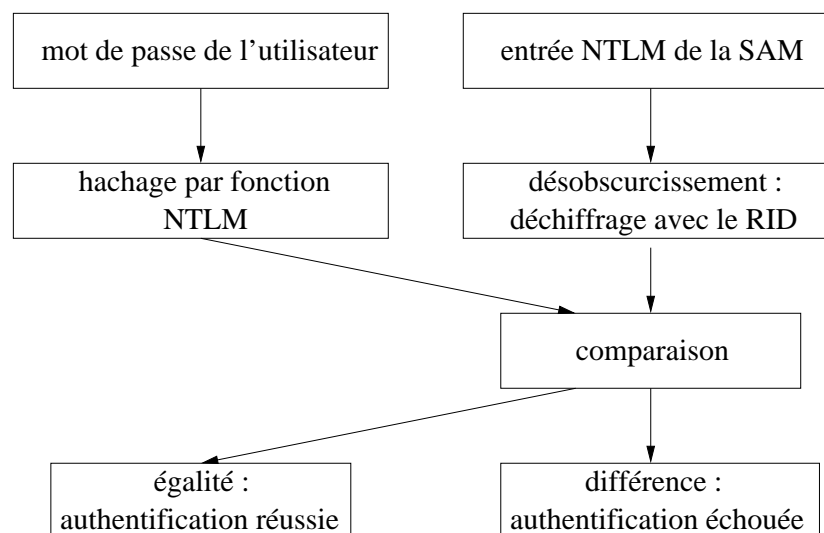


FIG. 2 – Authentification locale

Vocabulaire : j'ai choisi les termes LanMan et NTLM par habitude. Dans la littérature les termes respectifs LM et NT ou ASCII et Unicode ou CaseInsensitivePassword et CaseSensitivePassword sont également lisibles.

4.2 Les différents algorithmes de chiffrement sur un réseau Windows

Sur le réseau, l'algorithme utilisé dépend fortement du protocole applicatif utilisé entre le client et le serveur. Dans le cas d'un réseau Windows, l'authentification est fondée sur un protocole de type défi/réponse.

Le principe du défi/réponse est utilisé lorsque le mot de passe ne doit pas passer en clair sur le réseau. Le chiffrement de la session entière ou l'authentification par un système de clef publique/clef privée sont d'autres moyens mais ces méthodes ne sont généralement pas utilisées sur un réseau Windows. Un protocole défi/réponse est composé d'un échange de données entre le serveur et le client, afin que le client prouve au serveur qu'il connaît le mot de passe de l'utilisateur sans envoyer ce mot de passe en clair sur le réseau. Pour cela, à la connexion, le serveur envoie un défi au client. Le client utilise ce défi et le mot de passe entré par l'utilisateur pour calculer une réponse qu'il retourne au serveur. De son côté, le serveur a effectué la même opération avec le défi expédié au client et l'image du mot de passe contenu dans la base des utilisateurs. Il compare alors les résultats et considère que le mot de passe saisi par l'utilisateur est correct lorsque les résultats sont identiques.

Ce défi doit être unique à chaque connexion, de sorte que quelqu'un qui écoute le réseau ne puisse pas rejouer les données d'authentification capturées, c'est à dire mener une attaque par replay.

Dans le cas d'un réseau Windows, le défi est une chaîne aléatoire de 8 octets. Le serveur ne possédant que des empreintes du mot de passe de l'utilisateur, le client calcule une empreinte du mot de passe fourni par l'utilisateur. Cette empreinte de 16 octets est complétée de 5 octets nuls et divisée en 3 chaînes de 7 octets chacune. Chacune des 3 chaînes est indépendamment utilisée comme clé de chiffrement pour chiffrer le défi en DES. Les 3 chaînes résultats de 8 octets chacune sont concaténées pour former la réponse.

Voir **Figure 3** : *Authentification distante*

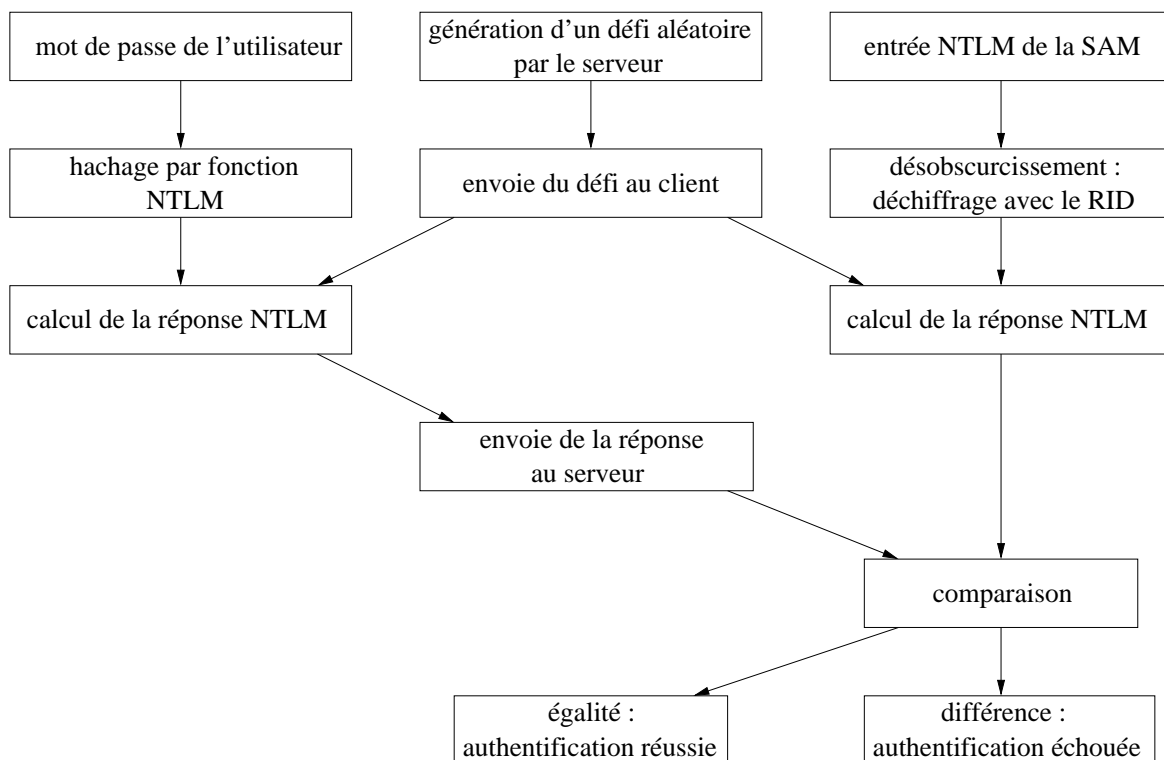


FIG. 3 – Authentification distante

Si ce protocole permet que le mot de passe ne circule pas en clair et que des données d'authentification capturées ne sont pas rejouables, il possède quand même quelques faiblesses importantes à connaître.

Tout d'abord, le serveur n'ayant pas accès au mot de passe en clair puisqu'il ne possède que des images non-réversibles du mot de passe, le client doit donc se mettre au même niveau que le serveur en calculant les empreintes du mot de passe fourni. Ainsi, le vol d'un fichier de mots de passe fournit la connaissance des empreintes, ce qui suffit à s'authentifier avec l'aide d'un client modifié. C'est pour cela qu'il est dit que les fichiers de mots de passe sous Windows contiennent des équivalents de mots de passe en clair, car même si le mot de passe ne peut pas être déduit immédiatement, l'authentification à la place de n'importe quel compte est possible. Il est à noter que ces systèmes de défi/réponse fondés sur les mots de passe sont vulnérables à ce type d'attaque si le système ne chiffre pas les empreintes.

Il est laissé comme exercice au lecteur d'écrire un *patch* à *smbclient* de quelques lignes considérant toute entrée d'un mot de passe de 32 caractères hexadécimaux comme une empreinte NTLM et l'utilisant avec le défi reçu du serveur pour calculer la réponse attendue.

Le calcul de la réponse ne dépendant que du mot de passe de l'utilisateur, la facilité de cassage du mot de passe à partir de la capture du défi et de la réponse ne dépend que de la force du mot de passe. Par exemple un mot de passe comme "toto" est trouvé en un dixième de seconde avec tout logiciel de cassage performant.

Le principe du défi/réponse impose que le défi soit unique à chaque connexion pour empêcher le rejeu. Bien sûr, des problèmes de mise en oeuvre existent comme sous Windows 95 où un nouveau défi n'est généré qu'une fois toutes les 30 minutes, laissant alors en moyenne 15 minutes à un pirate pour se reconnecter au même serveur Windows 95 avec les mêmes données d'authentification que celles qu'il vient de capturer sur le réseau.

Un dernier problème est que les postes clients Windows 9x ne possèdent pas les fonctions de calcul des empreintes NTLM, ce qui signifie que c'est l'empreinte LanMan qui est utilisée pour s'authentifier. Dans ce cas, un pirate qui écoute le réseau s'attaque au résultat LanMan avec un bon taux de succès si le mot de passe est relativement faible. Si les deux protagonistes possèdent les fonctions de calcul des deux types d'empreintes, le client utilise les deux pour calculer et envoi deux résultats à partir du défi du serveur. Cela est considéré comme une vulnérabilité puisque le pirate, en s'attaquant tout d'abord au résultat LanMan, bien plus faible, puis à l'autre résultat, obtient le mot de passe exact, c'est à dire avec la casse exacte. Le serveur ne vérifie que le résultat NTLM car l'empreinte NTLM est bien plus fidèle au mot de passe en clair.

Pour se protéger de certains outils s'attaquant à l'authentification réseau LanMan et NTLM, Microsoft a mis en oeuvre NTLMv2. Il s'agit en fait de l'algorithme HMAC-MD5 (voir la rfc2104) avec en prime erreur de mise en oeuvre... Normalement lorsque la longueur de la clé est supérieure à 64 bits alors le MD5 de la clé doit être utilisé comme clé. La version de Microsoft tronque simplement la clé aux 64 premiers bits.

Lorsqu'un utilisateur se connecte localement à un poste d'un domaine, le poste se connecte à l'un des contrôleurs du domaine sur le partage IPC\$ en utilisant le mot de passe saisi pour vérifier son authentification. Si la connexion réussit alors le poste accepte l'authentification de l'utilisateur. Pour d'autres informations sur l'authentification sous Windows, voir dans ce numéro l'article de Jean-Baptiste Marchand "*Modèle de sécurité du système Windows*".

5 Les logiciels de cassage contre Windows

De nombreux logiciels de cassage de mots de passe sont utilisés contre les empreintes Windows. Voici les plus usités ainsi que les plus originaux.

5.1 L0phtCrack

L0phtCrack est assurément le logiciel le plus connu pour casser les mots de passe Windows. Il s'agit à l'origine d'un shareware pour Windows écrit par Mudge du groupe L0pht et aujourd'hui accessible sur le site <<http://www.atstake.com/research/lc3/>>.

Il supporte les chiffrements LanMan et NTLM avec ou sans défi/réponse. Par conséquent, il est capable de s'attaquer aussi bien à un fichier de mots de passe récupéré localement ou à distance via SMB, que de s'attaquer aux mots de passe capturés sur le réseau sous forme de défi/réponse grâce à un renifleur intégré.

Lors de sa version 2.52, *L0phtCrack* possédait la mise en oeuvre LanMan la plus rapide mais *John the Ripper* l'ayant détrôné avec ses versions 1.6.x-dev, *L0phtCrack* possède désormais une fonctionnalité de cassage distribué depuis sa version 3.0.

Seules les sources de la version 1.5 sont disponibles, celles-ci compilant aussi sous Unix. Si la mise en oeuvre de cette ancienne version est très loin d'être optimale, elle a pour avantage de rendre plus intelligibles les différents algorithmes en se référant directement à des sources qui fonctionnent.

5.2 Password Appraiser

Password Appraiser est un logiciel commercial édité par Quakenbush Consulting, Inc. et est disponible sur <<http://www.quakenbush.com/>>, une version de démonstration est téléchargeable.

Sa particularité est de travailler à partir d'un dictionnaire pré-calculé. La version libre est fournie avec une petite base et la version commerciale avec un CDROM.

S'il ne supporte que le chiffrement LanMan, il faut se méfier d'une fonction activée par défaut nommée "*Internet Query*". Celle-ci consiste à envoyer au site web pour interrogation de la base centrale les empreintes qui n'ont pas pu être cassées.

De plus, ce logiciel considère comme sûr tout mot de passe absent de sa base. Cependant, comme il manque certains mots de passe usuellement cassés en moins de quelques heures, l'administrateur ressent un faux sentiment de sécurité.

5.3 c2myazz

Ce vieux logiciel Open Source pour MSDOS ne possède aujourd'hui ni auteur ni homepage. Il attaque activement les connexions à des partages SMB pour en récupérer les mots de passe en clair.

Lors d'une connexion, le client envoie au serveur la liste des différentes versions du protocole SMB (dialectes) qu'il connaît et normalement le serveur répond avec le dialecte le plus évolué connu des deux. Dans notre cas, l'expression "différents dialectes" signifie "fonctions de hachage de mot de passe" utilisables. Quant au "dialecte plus évolué" il s'agit en fait de la fonction la moins sensible aux attaques. Si besoin est, la réponse du serveur est accompagnée du défi à utiliser lors de l'authentification.

Le principe de l'attaque est très simple : le programme écoute le réseau en attendant qu'un client se connecte à un serveur. Lorsque le client a envoyé à un serveur la liste des dialectes qu'il connaît alors le programme demande au client d'envoyer son mot de passe en clair en répondant plus rapidement que le serveur.

Cette attaque nommée "*downgrade*" possède une variante où un pirate fait en sorte qu'un client se connecte sur un programme relais et ce dernier se connecte alors au serveur. Les données échangées entre le client et le serveur transitent toutes par le relais, le pirate change la demande du serveur de hachage du mot de passe par une demande d'envoi en clair. Lors de cette attaque de l'intercepteur (*man in the middle* en anglais), le pirate n'a plus qu'à enregistrer les mots

de passe transitant en clair. La possibilité de capturer en direct les échanges de fichiers et celle de leurs modifications au fil des transferts est alors une cerise sur le gâteau.

5.4 Crack

Crack est le père des logiciels de cassage moderne, contenant depuis très longtemps toutes les fonctions de génération de mots de passe aujourd'hui classiques. Sa version actuelle ne supporte que les mots de passe pour Unix, mais un patch existe pour casser les empreintes LanMan.

Ne pas confondre la version patchée de *Crack* avec *NTCrack*, logiciel moins sophistiqué mais dont les sources sont plus compréhensibles.

5.5 John the Ripper

John the Ripper est actuellement le logiciel de cassage le plus évolué, aussi bien en termes d'algorithmes de chiffrement supportés, d'algorithmes de génération de mots de passe mis en oeuvre, que d'architectures processeur supportées.

S'il n'existe aucune interface graphique, une version Windows est disponible et supporte comme toutes les autres l'algorithme LanMan. S'il est bien moins sexy que *L0phtCrack*, *John the Ripper* est, lorsqu'il est utilisé avec intelligence, bien plus puissant.

Nous reviendrons sur ce logiciel dans le prochain article, en expliquant tout ce qui fait sa force.

5.6 Autres logiciels de cassage contre Windows

Côté utilisateurs il existe deux vecteurs d'attaque.

Sous les postes Windows clients, de Windows 3.x à Windows Millenium, à chaque fois qu'un utilisateur local se connecte sur un serveur distant alors ses données d'authentification (login, serveur et partage distants) sont enregistrées dans un fichier **.pwl* dans le répertoire système du poste client. Jusqu'à la première version de Windows 95, il suffit simplement de décoder le fichier pour obtenir les données. À partir de Windows 95 OSR1, ce fichier est chiffré en RC4 avec le mot de passe d'authentification locale de l'utilisateur qu'il suffit de casser pour obtenir toutes ces données en clair. Plusieurs logiciels existent pour effectuer cela, comme *pwlcrack* et *pwltools*.

Du côté des logiciels, ceux ayant besoin du mot de passe de l'utilisateur pour accéder à un serveur (messagerie, web, etc.) sauvegardent les mots de passe des utilisateurs de façon codée pour les réutiliser sans ennuyer l'utilisateur à les lui redemander... Ces données sont enregistrées par exemples dans un fichier nommé *preferences.js* par Navigator et dans une partie de la base de registres par *MSIE*.

6 La protection des mots de passe sous Windows

Après avoir vu les méthodes classiques d'attaque voyons comment les contrer, tout d'abord en protégeant les mots de passe puis en les durcissant. La protection des mots de passe doit être réalisée à plusieurs niveaux : dans le système, sur le réseau et au niveau applicatif.

6.1 La protection des mots de passe au niveau du système

Au niveau système, une seule possibilité s'offre à nous : le chiffrement de la SAM. Cela est réalisable sous Windows NT avec une fonctionnalité optionnelle livrée depuis le service pack 2 de Windows NT 4 : *syskey*. Sur tous les systèmes Windows 2000, cette fonctionnalité est activée par défaut.

syskey effectue le chiffrement des empreintes grâce à une clé générée à partir d'un mot de passe choisi par l'administrateur. Ce mot de passe est nécessaire au démarrage du système afin que le programme *lsass.exe* puisse accéder normalement aux informations d'authentification. Ce mot de passe est soit stocké sur une disquette, soit caché dans la partition système, soit demandé à la console à chaque démarrage du système, celui-ci ne pouvant terminer son initialisation tant que le bon mot de passe n'aura pas été entré.

Si sur le disque les empreintes sont chiffrées, elles sont toujours accessibles déchiffrées à tous les utilisateurs possédant le droit de débogage (tous les développeurs et administrateurs). Le programme *pwdump2* et son successeur récupèrent la SAM locale grâce à une "*DLL Injection*" sur le processus *lsass.exe*. Il s'agit de conduire le programme visé (*lsass.exe*) à appeler une fonction située dans une bibliothèque désignée par *pwdump2*. Dans le cas présent, cette fonction est celle qui permet à *lsass.exe* d'obtenir en mémoire la SAM en clair.

Un défaut de mise en oeuvre a été trouvé mi-décembre 1999 par Bindview. Après analyse de SAM chiffrées, il est apparu que les empreintes des utilisateurs étaient chiffrées en RC4. RC4 est un chiffrement fondé sur le XOR des données en clair avec une chaîne de codons (suite de nombres aléatoires). Cet algorithme est solide tant qu'un flux ne sert qu'à chiffrer un seul texte en clair. Or, Microsoft a utilisé le même flux pour chiffrer les deux empreintes LanMan et NTLM d'un utilisateur, ainsi que son historique.

Cette faille permet donc l'attaque des empreintes par force brute sans pour autant connaître la clé de chiffrement utilisée par le système : choisir un mot de passe potentiel, le chiffrer en LanMan et en NTLM, l'obscurcir tel qu'il le serait dans la SAM, puis effectuer un XOR entre ces deux empreintes calculées. Effectuer un XOR entre les empreintes chiffrées et comparer les résultats : si les résultats sont identiques, alors le mot de passe choisi est celui de l'utilisateur. Si cette attaque est bien plus lente que celle effectuée normalement sur l'empreinte LanMan, puisque ces calculs doivent être effectués pour chaque utilisateur, avec une base de plusieurs dizaines d'utilisateurs, elle donne un mot de passe assez rapidement.

De plus, Bindview a trouvé que la clé de chiffrement RC4 est fondée sur le numéro de l'utilisateur (pour être précis le MD5 de la concaténation du mot de passe *syskey* sur 128 bits et de 4 octets du RID), ce qui accélère encore l'attaque précédente...

Depuis le service pack 2 de Windows 2000 et sur Windows XP, la sauvegarde des mots de passe au format LanMan dans la SAM est désactivable grâce à la clé de registre NoLMHash. Il faut alors s'assurer que tous les clients sont compatibles NTLMv1 : voir l'article "*How to Disable LM Authentication on Windows NT*" à l'URL <http://support.microsoft.com/support/kb/articles/q299/6/56.asp>

6.2 La protection des mots de passe sur un réseau Windows : pourquoi ?

S'il est important de protéger ses mots de passe sur le réseau, il est aussi important de savoir pourquoi, afin ne pas confondre moyen et but. La principale raison est simplement le grand nombre de renifleurs spécialisés dans l'écoute et l'extraction des mots de passe qui passent en clair sur les réseaux locaux ou distants.

Aujourd'hui encore, de très nombreux protocoles envoient leurs mots de passe en clair sur le réseau, les plus utilisés étant FTP, Telnet, POP, IMAP, SMTP, etc. De nombreux programmes, sous Unix comme sous Windows, arrivent très facilement à construire des bases d'authentification à partir de l'écoute de ces flux.

Plusieurs programmes récupèrent les couples défi/réponse SMB : *readsmb2.c* et *L0phtCrack 2.x*.

Le programme aujourd'hui le plus connu est *dsniff* <<http://naughty.monkey.org/~dugsong/dsniff/>> développé par Dug Song. Il capture les mots de passe en clair (ou obscurcis) dans plus de 30 protocoles : FTP, Telnet, SMTP, HTTP, POP, poppass, NNTP, IMAP, SNMP, LDAP, Rlogin, RIP, OSPF, NFS, YP/NIS, SOCKS, X11, CVS, IRC, AIM, ICQ, Napster, PostgreSQL, Meeting Maker, Citrix ICA, Symantec, NAI Sniffer, Microsoft SMB, Oracle SQL*Net, Sybase et Microsoft SQL auth info.

La manière la plus classique de se protéger au niveau réseau contre l'écoute est en général d'utiliser des commutateurs (*switches*) à la place des répéteurs (*hubs*) afin que seuls l'expéditeur et le destinataire puissent avoir accès à une donnée sur le réseau. *dsniff* contient plusieurs autres outils afin d'aider à l'interception des informations dont *arpspoof* qui permet d'écouter même sur un réseau commuté. De plus, *dsniff* contient une documentation très complète et une FAQ expliquant de nombreuses vulnérabilités, les attaques et les façons de s'en protéger.

dsniff méritant à lui seul un article, il est fortement conseillé d'aller lire ses documentations, des traductions françaises étant disponibles : cela vaut réellement le temps investi.

6.3 La protection des mots de passe sur un réseau Windows : SMB

Sur un réseau Windows, le principal protocole à surveiller est SMB. Le durcissement nécessaire à la protection de ses mots de passe requiert "seulement" la modification de chaque serveur et de chaque poste client.

Il faut tout d'abord interdire sur chaque client l'envoi du mot de passe en clair sur le réseau, la clé de registre dépendant de la version du système :

- Windows 95 et Windows 98 :
[HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\ VxD\VNETSUP]
"EnablePlainTextPassword"=dword:00000000
- Windows NT (par défaut depuis le service pack 3 de Windows NT 4) :
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Rdr\Parameters]
"EnablePlainTextPassword"=dword:00000000
- Windows 2000 (par défaut) :
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\LanmanWorkStation\Parameters]
"EnablePlainTextPassword"=dword:00000000

Il faut ensuite, sur chaque serveur, limiter le chiffrement le plus faible à NTLMv1, ou NTLMv2 à partir du service pack 4 de Windows NT 4, voir l'article "*How to Disable LM Authentication on Windows NT*" à l'URL <<http://support.microsoft.com/support/kb/articles/q147/7/06.asp>> pour les différentes explications de mise en oeuvre.

6.4 La protection des mots de passe sur un réseau Windows : extensions Microsoft

Microsoft est tellement content de son système de défi/réponse NTLM, qu'il l'a porté dans plusieurs protocoles ouverts comme HTTP, IMAP, LDAP, NNTP et POP3, rendant alors ses serveurs incompatibles avec nombre de clients d'éditeurs concurrents lorsque l'administrateur du serveur rend obligatoire cette option.

L'avantage principal est bien sûr que le mot de passe n'est plus envoyé en clair sur le réseau lorsque ces protocoles sont utilisés, cela ne fonctionnant bien sûr qu'entre un client et un serveur édités par Microsoft.

Comme nous l'avons vu avec l'authentification SMB, la base d'authentification du serveur contient des équivalents de mots de passe en clair, utilisables par tout pirate avec l'aide d'un client modifié pour s'authentifier à la place de n'importe qui. Cette extension NTLM possède exactement le même problème. Elle procure donc un faux sentiment de sécurité auprès des utilisateurs et des administrateurs.

De plus, si cela protège les mots de passe solides d'être volés sur le réseau, les mots de passe simples sont très facilement cassés à partir de couples défi/réponse capturés.

Il existe plusieurs ressources sur Internet pour effectuer de l'authentification NTLM ou "jouer" avec :

- *fetchmail* <<http://www.tuxedo.org/~esr/fetchmail/>> est un programme client de récupération de messages électroniques via POP, IMAP, etc. qui sait s'authentifier auprès d'un serveur NTLM.
- *NTLM Authentication Scheme for HTTP* <<http://www.innovation.ch/java/ntlm.html>> et *libntlm-0.21* <<ftp://ftp.visi.com/users/grante/ntlm/>> sont deux bibliothèques d'authentification NTLM.
- *NTLM Authorization Proxy Server* <<http://www.geocities.com/rozmanov/ntlm/>> et *The SMBProxy Tool* <<http://www.cqure.net/tools02.html>> sont deux relais qui changent au vol l'authentification de l'utilisateur par celle trouvée dans une base SAM.

6.5 La protection des mots de passe applicatifs sous Windows

La seule application Windows abordée aujourd'hui est *frontpage*. Il est nécessaire de protéger les fichiers **.pwd* par des ACL (listes de contrôle d'accès) ne donnant accès à ces fichiers qu'aux administrateurs et au système, et en aucun cas aux comptes utilisés par IIS.

Dans le prochain article, plusieurs applications seront étudiées. Généralement utilisées aussi bien sous Windows que sous Unix, les solutions mettront en place aussi bien de l'authentification forte que du chiffrement fort, parfois avec de l'encapsulation.

7 Le durcissement des mots de passe

Le durcissement des mots de passe des utilisateurs est la fonctionnalité qui n'accepte un nouveau mot de passe qu'après s'être assuré qu'il suit un certain nombre de règles. Ce durcissement est donc toujours lié au système.

Cela assure que les mots de passe choisis ne sont pas trop simples et qu'ils résistent plus longtemps contre les programmes de cassage de mots de passe.

Dans le prochain article, quelques statistiques vous seront données sur un grand nombre de mots de passe cassés ainsi que sur les temps de cassage maximaux pour les différents types d'algorithmes. Si vous n'êtes pas encore convaincus de l'utilité du durcissement des mots de passe, vous le serez alors.

7.1 Le durcissement des mots de passe sous Windows

Le durcissement sous Windows est réalisé grâce à une bibliothèque optionnelle livrée depuis le service pack 2 de Windows NT 4 : *passfilt.dll*. L'installation est manuelle via le changement d'une clé de registre :

```
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Lsa]
"FPNWCLNT"=reg_multi_sz:PASSFILT
```

Les deux caractéristiques de *passfilt.dll* sont : trop basique et non configurable. En effet, les deux règles à suivre sont :

- la longueur minimale est de six caractères
- au moins deux caractères parmi deux groupes suivants : les majuscules, les chiffres et les caractères non alphanumériques.

Un exemple comme **Bonjour1** est donc accepté par le système puisqu'il fait 8 caractères et comporte une majuscule et un chiffre, malheureusement n'importe quel programme moderne de cassage trouve ce mot de passe en moins d'un dixième de seconde.

Il est donc conseillé d'utiliser une mise en oeuvre d'un fournisseur tiers. Bien sûr, celle-ci est payante, mais elle est de bien meilleure qualité et configurable suivant vos besoins. Dans tous les cas, seuls les changements via une connexion réseau sont renforcés et jamais ceux réalisés par l'administrateur via l'outil de gestion des utilisateurs.

7.2 Quelques règles de constitution

Si vous installez un programme de durcissement de mots de passe sur un système, vous serez confronté au problème de gestion des utilisateurs qui ne comprennent pas pourquoi leurs nouveaux mots de passe sont systématiquement rejetés. Il est donc important avant toute chose d'éduquer vos utilisateurs, la sécurité d'un système reposant sur la force des mots de passe utilisés.

Quelques règles simples de constitution, même si elles ne suffisent pas à elles seules à assurer la force du mot de passe, aident souvent à passer le cap du programme de durcissement.

La particularité des environnements Windows est la présence de l'empreinte LanMan. Sa faiblesse intrinsèque force à choisir en fait 2 mots de passe costauds de 7 caractères chacun et de jouer sur ce stratagème pour résister aux programmes de cassage. Chaque "sous-mot de passe" doit donc comporter 7 caractères dont au moins 1 chiffre et un caractère non-alphanumérique, et passer individuellement le cap du programme de durcissement. Une règle simple est que le caractère non-alphanumérique et le chiffre ne doivent pas être placés en premier ou dernier caractère et ne doivent pas remplacer une lettre d'un mot existant ou s'insérer entre deux lettres.

En se référant aux méthodes de cassage, aucun mot de passe ne doit faire référence à une information relative à l'utilisateur, à son entourage, à ses passions, à son métier, à son entreprise, etc. Aucun ne doit être bâti à partir d'un mot existant qui se trouve dans un dictionnaire général, de noms propres, spécialisé, etc.

Une méthode simple est en général de choisir deux mots sans relation et de les combiner en insérant des chiffres et des caractères non-alphanumériques. Malheureusement, à cause de l'algorithme LanMan, l'utilisateur risque de se retrouver avec des mots de passe extrêmement simples à casser.

Il est donc possible de combiner des parties de mots entre elles, en faisant bien attention que ceux-ci ne forment pas par hasard un mot existant. Ce rôle incombe au programme de durcissement qui à partir du nouveau mot de passe en clair effectue en une fraction de seconde tous les calculs demandant généralement plusieurs mois à un programme de cassage. Cela évite que l'utilisateur malchanceux ne tombe par hasard sur un mot d'origine étrangère pour lequel un simple "i" aurait été remplacé par le chiffre "1". Toute ressemblance avec une situation vécue serait totalement fortuite !

L'écriture de mots en phonétique est aussi une solution qui aide, tout comme utiliser les premières lettres de vers, phrases ou expressions, agrémentées bien sûr de quelques chiffres et caractères non-alphanumériques.

Conclusion

J'espère vous avoir sensibilisé sur le rôle des mots de passe dans un environnement connecté et convaincu sur le fait qu'il est important que les utilisateurs y apportent toutes leurs attentions et les administrateurs, toutes leurs protections.

En attendant la seconde partie traitant de l'environnement Unix, bien du plaisir vous est souhaité dans l'application de certains de ces conseils. Mais il est important de ne pas confondre moyens et buts : n'appliquez un conseil que pour parer à un problème auquel vous avez affaire et non pas pour le plaisir de vous dire que vous avez fait quelque chose.

Denis Ducamp

Denis.Ducamp@groar.org - <http://www.groar.org>

Denis.Ducamp@hsc.fr - <http://www.hsc.fr>

Denis Ducamp est consultant en sécurité informatique chez Hervé Schauer Consultants et spécialisé dans la sécurité systèmes et réseaux. Cet article ainsi que sa prochaine suite ont été écrits à partir d'une présentation nommée "*crackage et durcissement des mots de passe*" développée chez HSC et publiquement accessible sur <http://www.hsc.fr/ressources/presentations/mdp2/>.

\$Id : part1.sgml,v 1.21 2002/03/12 11 :02 :19 ducamp Exp \$