Tests d'intrusion, méthodologie

Introduction

Ce papier n'est qu'un guide, et n'a pas la prétention de parler de toutes les techniques d'intrusion. En fait, c'est plus une introduction aux méthodes d'intrusion, destinée aux débutants ayant un *background* minimum d' *UNIX* et des réseaux. Peut-être qu'un prochain guide de ce style parlera des techniques et méthodes avancées, et d'autres protocoles que l'on peut exploiter pour obtenir des informations, et s'introduire dans des réseaux informatiques.

Décharge

L'auteur ne peut être tenu pour responsable des dégâts résultants de la mauvaise utilisation des connaissances et techniques décrites dans ce guide.

Récupération d'informations, 1ère reconnaissance

But

Pour évaluer le niveau de sécurité d'un réseau, il faut d'abord le connaître. La première étape est donc d'avoir une carte du réseau ciblé pour l'auditer.

Logiciels utilisés

Pour se faire, voici une liste des logiciels utiles:

- *Nmap* (http://www.insecure.org/nmap/), un *scanneur* de ports ;
- *Siphon* (http://siphon.datanerds.net/), un *mappeur* de réseaux passif (sans connexion) ;
- *XScan* (http://www.martnet.com/~johnny/exploits/X/xscan.c) ;
- *XSpy* (http://www.acm.vt.edu/~jmaxwell/programs/xspy/xspy.html).

Méthode

Identifier les sous-réseaux

Théorie:

La première chose à faire est de déterminer quels sont les sous-réseaux appartenant à la cible. Si l'audit à réalisé est "autorisé", la liste des réseaux peut être communiquée par les instances impliquées dans celui-ci. Sinon, on peut interroger les sites qui recensent les noms de domaines. Pour l'Europe: http://www.ripe.net, pour les Etats-Unis: http://www.arin.net/, et pour la planète entière: http://www.iana.net . Une autre méthode consiste en l'interrogation des serveurs de nom de la cible, si le transfert de zone vous est autorisé.

Pratique:

L'utilitaire en ligne de commande **whois** peut également être utilisé pour obtenir des informations sur le réseau ciblé. Les informations que l'on peut obtenir sont par exemple, les serveurs *DNS*, le nom de la personne responsable, un numéro de téléphone, une adresse *e-mail*, ainsi qu'une description du réseau. Exemple (l'adresse 213.36.127.5 est celle de *www.chez.com*):

```
prompt$ whois -h whois.ripe.net 213.36.127.5
% This is the RIPE Whois server.
% The objects are in RPSL format.
% Please visit http://www.ripe.net/rpsl for more information.
% Rights restricted by copyright.
% See http://www.ripe.net/ripencc/pub-services/db/copyright.html

inetnum:      213.36.120.0 - 213.36.127.255
netname:      LIBERTYSURF-4
descr:        Liberty Surf is a free Internet Service Provider
descr:        based in France
```

```
country:      FR
admin-c:      LTAD1-RIPE
admin-c:      ED1197-RIPE
tech-c:       LTNO1-RIPE
status:       ASSIGNED PA
remarks:      All abuse requests MUST be sent to 'abuse@libertysurf.fr'
remarks:      and the logs must include the timezone and GMT offset.
mnt-by:       MNT-LIBERTYSURFTELECOM
changed:      b.grange@libertysurf.fr 20010416
changed:      b.grange@libertysurf.fr 20010503
source:       RIPE

route:        213.36.96.0/19
descr:        Praxitel / LibertySurf
origin:       AS12876
mnt-by:       MNT-LIBERTYSURFTELECOM
changed:      b.grange@libertysurf.fr 20000120
changed:      b.grange@libertysurf.fr 20010215
source:       RIPE

role:         LIBERTYSURF TELECOM ABUSE DEPARTMENT
address:      Service Juridique Libertysurf
address:      10 rue Fructidor, 75834 Paris CEDEX 17, France
phone:        +33 1 41 66 46 36
fax-no:       +33 1 41 66 46 90
e-mail:       abuse@libertysurf.fr
trouble:      ***
trouble:      *** ALL ABUSE REQUESTS MUST BE SENT TO abuse@libertysurf.fr ***
trouble:      ***
trouble:      *** Les requisitions judiciaires doivent etre
trouble:      *** envoyees a cette adresse a l'attention du
trouble:      *** service juridique.
trouble:      ***
admin-c:      ED1197-RIPE
tech-c:       RL8839-RIPE
nic-hdl:      LTAD1-RIPE
mnt-by:       MNT-LIBERTYSURFTELECOM
changed:      b.grange@libertysurf.fr 20010503
source:       RIPE

role:         LIBERTYSURF TELECOM NETWORK OPS
address:      Libertysurf Telecom, Paris
phone:        +33 1 45 08 23 28
fax-no:       +33 1 45 08 25 29
e-mail:       noc@libertysurf.net
trouble:      Questions and problem reports ... noc@libertysurf.net
trouble:      ALL ABUSE REQUESTS MUST BE SENT TO abuse@libertysurf.fr
admin-c:      ED1197-RIPE
tech-c:       BG34
nic-hdl:      LTNO1-RIPE
mnt-by:       MNT-LIBERTYSURFTELECOM
changed:      b.grange@libertysurf.fr 20010419
source:       RIPE

person:       Eric Denoyer
address:      LibertySurf Telecom
address:      10, rue Fructidor
address:      75834 Paris CEDEX 17
phone:        +33 1 41 66 77 00
fax-no:       +33 1 41 66 77 67
e-mail:       eric.denoyer@libertysurf.fr
nic-hdl:      ED1197-RIPE
```

```
mnt-by:       MNT-LIBERTYSURFTELECOM
changed:      b.grange@libertysurf.fr 19991109
changed:      b.grange@libertysurf.fr 20010120
changed:      b.grange@libertysurf.fr 20010215
source:       RIPE
```

La commande **host** permet d'obtenir la liste des machines enregistrées dans les serveurs de noms de la cible:

```
prompt$ host -l chez.com
Server failed: Query refused
```

Dans ce cas, vous n'avez pas le droit d'obtenir cette liste, mais si vous aviez été à l'intérieur du réseau de domaine *chez.com*, vous l'auriez probablement obtenue. On peut également obtenir énormémement d'informations sur la cible en interrogeant les moteurs de recherche, et spécialement les moteurs de recherche qui *scannent* les *newsgroups* (*usenet*) spécialisés.

Scanner le réseau

Théorie:

Une fois les sous-réseaux connus, il reste à les *scanner*. Le *scan* de réseau est une étape qui permet de savoir quelles sont les adresses *IP* valides, quels sont les ports ouverts sur ces machines (ouverts, fermés ou filtrés), et également l' *OS* et sa version. Pour cela on utilise *nmap*. Ce programme permet de *scanner* des réseaux complets, en spécifiant par exemple l'adresse réseau et son masque. L' *OS* est identifié en analysant la réponse qu'est donnée à la connexion *TCP* à un port ouvert, puis à un port fermé. Ce n'est pas une méthode sûre à cent pourcent, mais elle est quand même très efficace. Il existe différents type de *scan*, le plus couramment utilisé est le *SYN scan*. Il est furtif, car il n'établi pas une connexion *TCP* complète. Mais si un *NIDS logge* les paquets *TCP* avec le *flag SYN*, il sera repéré. Une autre méthode pour découvrir la topologie d'un réseau est l'utilisation d'un *mappeur* de réseau passif, comme *Siphon*. Il suffit d'être *root* sur sa machine, et de lancer l'exécutable. Il ne permettra malheureusement que de découvrir la topologie réseau du brin physique sur lequel votre machine se trouve. Si il y a de nombreuses machines *UNIX*, on peut également *scanner* le réseau pour les connexions *X* possibles depuis d'autres machines. Ceci permet, entre autres, de capturer les écrans des machines vulnérables, de capturer la frappe de l'utilisateur, ou même de regarder les fenêtres de la victime en temps réel.

Pratique:

Nmap: Pour *scanner* un réseau du type: *192.168.0.0/24*, l'adresse réseau étant donc *192.168.0*, et ce sous-réseau possède jusqu'à 255 machines (le masque étant sur 24 bits). L'adresse *IP* précédente n'étant pas forcément de classe C, puisque le masque peut être spécifié, grâce aux extensions *CIDR*, en restant conforme aux standards proposés (voir *RFC1517, RFC1518, RFC1519*).

```
prompt# nmap -sS -sU -O -oN trace.log 192.168.0.1-254

Starting nmap V. 2.54BETA25 ( www.insecure.org/nmap/ )
Interesting ports on agathon (192.168.0.2):
(The 3108 ports scanned but not shown below are in state: closed)
Port       State       Service
22/tcp     open        ssh
111/tcp    open        sunrpc
111/udp    open        sunrpc
137/udp    open        netbios-ns
138/udp    open        netbios-dgm
139/tcp    open        netbios-ssn
514/udp    open        syslog
518/udp    open        ntalk
919/udp    open        unknown
```

```
988/udp    open        unknown
991/udp    open        unknown
1003/udp   open        unknown
1007/udp   open        unknown
1011/udp   open        unknown
1019/tcp   open        unknown
1020/tcp   open        unknown
1021/tcp   open        unknown
1021/udp   open        unknown
1022/tcp   open        unknown
1023/tcp   open        unknown
2049/tcp   open        nfs
2049/udp   open        nfs

Remote operating system guess: FreeBSD 4.3

Nmap run completed -- 1 IP address (1 host up) scanned in 24 seconds
```

La sortie de ce programme ne montre que le résultat du *scan* d'une machine. Le paramètre *-sS* spécifie le type de *scan* (ici un *SYN scan*), le paramètre *-sU* spécifie de *scanner* également les ports *UDP*, l'option *-O* permet de tenter l'identification de l' *OS* de la machine ciblée, le paramètre *-oN* sauvegarde la sortie de l'exécution de la commande dans un fichier *trace.log*, et le dernier paramètre spécifie qu'il faut *scanner* de cette manière toutes les machines du réseau 192.168.0. Le script fourni dans l'annexe A permet de *scanner* tous les réseaux spécifiés dans un fichier fourni an paramètre.

Siphon:

```
prompt# siphon -v -i xl0 -o siphon.log

        [ The Siphon Project: The Passive Network Mapping Tool ]
                [ Copyright (c) 2000 Subterrain Security Group ]


Running on: 'freebsd.domain.org' running FreeBSD 4.3-RELEASE on a(n) i386

Using Device: xl0

Host                            Port    TTL             DF
Operating System

192.168.77.171          139             128             ON              2238
192.168.38.8            21              255             ON              Solaris
2.6 - 2.7
192.168.132.127         139             128             ON              2238
10.208.175.71           80              59              ON              16D0
192.168.39.202          139             128             ON              2238
10.208.175.70           80              59              ON              16D0
192.168.112.112         139             128             ON              2238
192.168.12.10           139             128             ON              2238
192.168.132.230         631             60              OFF             16D0
192.168.38.8            7               255             ON              Solaris
2.6 - 2.7
192.168.38.8            13              255             ON              Solaris
2.6 - 2.7
192.168.38.8            19              255             ON              Solaris
2.6 - 2.7
```

Le programme n'a pas tourné très longtemps pour cet exemple. Mais plus longtemps il trounera, plus vous aurez d'informations. A l'aide de la sortie de ce programme, vous pouvez connaître l' *OS* d'une machine, ainsi que les ports qui sont ouverts sur cette machine, et même quelles sont les

machines ayant le droit de se connecter à ces ports. Cette dernière information est très importante, puiqu'elle permet de savoir pour quelle machine, ou quel réseau, vous allez devoir vous faire passer pour contourner les protections basées sur l'adresse *IP* (telles que les *TCP Wrappers*, ou bien les *firewalls*).

XScan & XSpy: Le premier programme permet de trouver les machines vulnérables (celles dont les utilisateurs ont probablement exécuté la commande **xhost +**), et la deuxième permet de capturer la frappe au clavier de ces machines. Pour *scanner* le réseau 192.168.1/24.

```
prompt$ ./xscan 192.168.1 -s
xscan v1.1      X11 connection scanner     xygorf 1997
scanning 192.168.1.1 to 192.168.1.1 on port 6000
Trying 192.168.1.1...Invalid argument
Trying 192.168.1.2...Invalid argument
Trying 192.168.1.3...Invalid argument
...
Trying 192.168.1.41...Invalid argument
Trying 192.168.1.42...Invalid argument
Trying 192.168.1.43...Socket is already connected
Trying 192.168.1.44...Connect Timeout
Trying 192.168.1.45...Invalid argument
```

La ligne contenant *"Socket is already connected"* signifie que cette machine est vulnérable. Si la connexion *X* utilise les *MIT MAGIC COOKIES*, il faut d'abord récupérer la clé en *sniffant* (avec *DSniff*, par exemple).

```
prompt$ ./xspy -display vulnerable_host:0
ls
who
rlogin -l user01 host passwd
```

Un programme livré en standard avec *X* est **xwd**. Il permet de capturer le *DISPLAY* des machines vulnérables.

```
prompt$ /usr/X11R6/bin/xwd -display vulnerable_host:0 -root -out file.dmp
prompt$ /usr/X11R6/bin/xwud -in file.dmp
```

Recherche d'informations complémentaires

Une fois le *scan* du réseau terminé, une quantité d'informations est à extraire des fichiers de *log* s. Par exemple, on peut rechercher le serveur *DNS* en regardant la ou les machines qui ont le port 53 ouvert. De même pour le serveur de courriels, en regardant la ou les machines qui ont le port 25 ouvert. On peut également explorer les partages *NFS* (port 2049) et *NetBIOS* (port 139) pour acquérir d'autres informations, telles qu'une liste de comptes valides (en cherchant le partage **/var/mail**, gérant le courriel des utilisateurs), et que des fichiers récupérés dans ces partages. Le service *finger* permet également d'obtenir des comptes valides. C'est pour cette raison que bien souvent, l'accès à ce service est restreint par les *TCP Wrappers*, et qu'il faut avoir recours, soit à l'*IP Spoofing*, soit au *finger forwarding* pour obtenir ces informations. Le service *sendmail* permet également l'obtention de comptes valides par *brute force*, si les commandes *SMTP EXPN* et *VRFY* n'ont pas été désactivées.

Identifier les failles potentielles

But

Une fois la topologie du réseau cible bien connue (*OS*, ports ouverts, nombre de machines accessibles depuis l'extérieur du réseau), il reste a identifier les failles qui seront utilisées pour pénétrer dans le réseau. Ceci se fait en exploitant les failles distantes.

Logiciels utilisés

Il n'y a pas vraiment de logiciel plus utile qu'un autre. On peut quand même citer des *scanneurs* de vulnérabilités:

- *Nessus* (http://www.nessus.org/) ;
- *SAINT* (http://www.wwdsi.com/saint/).

Mais l'utilisation de ces logiciels n'est pas conseillée si la discrétion est de rigueur, puisque ces logiciels testent des failles bien connues des *NIDS*. La méthode la plus utile une fois le réseau bien *mappé* est l'interrogation de sites qui maintiennent une base de données des vulnérabilités découvertes, en recherchant des informations sur les services auxquels on peut accéder depuis l'extérieur du réseau cible:

- *SecurityFocus*: http://www.securityfocus.com ;
- *SecuriTeam*: http://www.securiteam.com/exploits/ ;
- *LSD*: http://www.lsd-pl.net/vulnerabilities.html ;
- *Uberhax0r*: http://www.uberhax0r.net/exploits/os/ ;
- *Technotronic*: ftp://ftp.technotronic.com/unix/ ;
- *Insecure*: http://www.insecure.org/sploits.html ;
- *LSD*: http://www.lsd-pl.net/vulnerabilities.html ;
- *Hack.co.za*: http://www.hack.co.za/ ;
- *Google*: http://www.google.com/ ;
- Sur *IRC*.

Méthode

Théorie:

Rechercher dans les sites précédents les éventuelles vulnérabilités des services accessibles depuis le réseau extérieur, et faire la liste des machines étant susceptibles d'être exploitées par ces failles. Sur des *channels IRC* de l' *underground*, on peut également obtenir des informations sur des vulnérabilités non-encore divulgées, à condition de connaître des personnes intéressantes.

Pratique:

Cette étape est très importante, une bonne organisation est nécessaire. On peut par exemple noter dans un fichier texte les vulnérabilitées potentielles concernant les machines du réseau. Il faudra ensuite vérifier que ces possibles trous de sécurité sont fondés. Exemple: Sur le site http://www.securityfocus.com/ :

- Dans la partie gauche, cliquer sur *Vulnerabilities* ;
- Choisir ensuite le type de recherche en cliquant sur un des onglets de la partie centrale (*By vendor*, *By title*, *By keyword* ...) ;
- En prenant par exemple *By keyword*, et en entrant comme mot clé *solaris*, on obtient une

liste de vulnérabilités découvertes pour cet *OS*.

Eliminer les failles non-fondées

But

Toutes les failles trouvées concernant les *OS* et les services du réseau à auditer dans l'étape précédente ne sont pas fondées, c'est-à-dire qu'il faut vérifier que ces services ou programmes sont réellement vulnérables. Cette étape permet d'éliminer une grande partie des fausses alertes. A la fin de cette étape, seules les failles exploitables doivent subsister.

Logiciels utilisés

Ici non-plus il n'y a pas vraiment de logiciels pour cela. Il faut tester les vulnérabilités à l'aide de petits programmes, appelés *exploit* s, sur les machines potentiellement vulnérables. C'est une étape plus ou moins longue, selon que le réseau est composé de nombreuses machines avec de nombreux *OS* différents, ou de nombreuses versions de ces *OS*. Les sites *SecurityFocus*, *SecuriTeam* et *Google* peuvent être intérrogés pour trouver ces programmes, ainsi que les autres sites listés dans la section précédente. On peut rechercher dans les *channels IRC* dans le monde de l' *underground*, pour trouver des *exploits* non divulgués. Mais il faut se méfier des programmes écris par d'autres *hacker* s, il vaut mieux regarder le source de près.

Méthode

Chaque vulnérabilité a sa propre technique d'exploitation. C'est pour cette raison qu'aucun programme pour automatiser ce genre de chose n'existe encore aujourd'hui. Il peut exister, par contre, des bibliothèques d' *exploits*, qu'il faut remettre à jour en permanence, dès qu'une nouvelle vulnérabilité est découverte. Nous allons faire un exemple avec le service *snmp*, de *Solaris 7 & 8*. Il est à noter que c'est un exemple de vulnérabilité réseau. Un plus grand nombre de vulnérabilités locales existe pour les systèmes *Solaris*.

Théorie:

Si une machine exécute le service *snmpXdmid* de chez *Sun*, et que c'est une version vulnérable à un débordement de tampon, il faut trouver un *exploit* pour cette version du *daemon snmp*.

Pratique:

Une fois l' *exploit* trouvé (par exemple sur le site de *SecurityFocus*), il suffit de compiler le programme, et de l'exécuter contre la machine à tester. Si vous obtenez un *shell root*, c'est que la machine est vulnérable. Il faut alors *patcher* la vulnérabilité, ou bien utiliser un *workaround* en attendant que celui-ci soit disponible. L'exemple précédent est donc à répéter pour toutes les vulnérabilités potentielles découvertes pendant l'étape de recherche de failles.

Intrusion

But

Maintenant, il faut s'introduire dans le réseau cible pour continuer la collecte d'informations. Cette étape se fait par exemple en exploitant les faiblesses des commandes *R** de *Berkeley*, ou en exploitant une faille de sécurité réseau (cf. exemple précédent). Si aucune des techniques

précédentes ne marchent, il reste la possibilité d'une intrusion par *brute force*, pour cela il faut trouver des comptes valides sur le réseau, probablement obtenus dans l'étape de reconnaissance, ou également par *brute force*, en exploitant les faiblesses de configuration de *sendmail* par exemple (qui permet d'obtenir des comptes valides).

Logiciels utilisés

- *NetCat* (rechercher sur *Google*) ;
- *Find Rlogin* (programme dans l'annexe A) ;
- Exploitation des faiblesses de *Sendmail* et utilisation des programmes standards *UNIX* tels que *finger* et *rusers* pour trouver des comptes valides ;
- Des sources d' *exploit* s trouvées sur *Internet*.

Méthode

Théorie:

Dans notre base de données d'informations sur le réseau, il ne reste plus maintenant que les machines qui ont des services vulnérables. Il suffit donc d'exploiter leurs faiblesses afin d'acquerir un accès *shell* au réseau de la cible. Il y a de fortes chances que cet accès soit directement un accès *root*. Si une liste de comptes valides a été obtenue, on peut utiliser **rlogin** pour se connecter au réseau en utilisant soit un mot de passe nul (pas de mot de passe du tout), soit le même mot de passe que le nom de *login*, soit espérer que la personne à qui appartient le compte a mis un + dans son fichier **~/.rhosts**. Le programme *Find RLogin* a été developpé pour accomplir cette tâche. Bien sur, si le service *rlogin* n'est pas accessible depuis l'extérieur du réseau, vous pouvez avoir recours au *spoofing*, et à l'utilitaire *NetCat* pour contourner cela.

Pratique:

Le programme *Find RLogin* prends comme paramètre un fichier contenant une liste de comptes (un par ligne). Il existe différentes méthodes pour obtenir une liste de comptes valides. La plus triviale est de trouver le partage *NFS* gérant le courrier des utilitateurs (comme expliqué dans l'étape de reconnaissance). Par *brute force*, on peut trouver des mots de passe, lorsque que l'on a obtenu une liste de comptes valides (par exemple à l'aide de *sendmail*, voir section plus haut). Un autre exemple, il suffit de demander par téléphone en se faisant passer pour un administrateur du réseau aux utilisateurs eux-mêmes. Une liste de numéros de téléphone a pu être obtenue dans la première étape (ceci s'appelle du *social engineering*). Utilisation de *Find RLogin*: \p prompt$ ./find_rlogin target_host accounts_list Trying login: root Trying login: user01 Trying login: user02 Trying login: bill \P Un fichier **logins.found** est créé, il contient les mots de passe des comptes trouvés, ou bien les comptes qui possèdent un + dans leur fichier **~/.rhosts**.

Récupération d'informations, 2ème reconnaissance

But

Une fois que l'on est dans le réseau, c'est encore une étape d'observation et de reconnaissance. Il faut savoir quels sont les serveurs de fichiers, de *backup*, de *log* s, *NIS*, et autres serveurs importants. Il faut aussi savoir quelles sont les machines des administrateurs et le nom de *login* de ceux-ci. Compromettre le compte d'un administrateur n'est pas chose aisée (en théorie), mais une fois l'un de ces comptes acquis, de nombreuses portes peuvent s'ouvrir.

Logiciels utilisés

Ce sont en général les outils standards qui sont utilisés dans cette seconde étape de reconnaissance. Des outils peuvent être développés pour cette tâche, mais la méthode manuelle est tout aussi efficace. On peut quand même installer un *sniffeur* (si vous avez obtenu un accès *root* dans l'étape d'intrusion), pour voler d'autres comptes, et obtenir d'autres informations sur les machines du réseau sur lequel on se trouve, mais il vaut mieux d'abord observer silencieusement, et savoir dans quelle mesure notre présence peut être repérée sur le réseau (si un *HIDS*, ou un *NIDS* est installé):

- *Snoop* ou *TcpDump* ;
- *Siphon*.

Méthode

Théorie:

La première chose à savoir est si le réseau cible utilise les *NIS* pour gérer les comptes des utilisateurs. Si oui, il suffit ensuite de récupérer les *NIS map* s, qui regorgent d'informations susceptibles de décrir le réseau de A à Z, mais surtout d'obtenir la liste complète des comptes utilisateurs, ainsi que la chaîne cryptée de leur mot de passe (les *maps* importantes sont: **passwd** et **netgroup**).

Pratique:

Une fois introduit dans le réseau, il suffit de taper la commande **domainname** pour obtenir le nom du domaine *NIS* si il y en a un. La commande **ypwhich** donne le nom du serveur *NIS*, faisant de celui-ci une cible. Pour connaître le nom de *login* du ou des administrateurs, il suffit de taper la commande **rusers**, puis d'observer le résultat. Les machines sur lesquelles l'utilisateur *root* est présent, ainsi qu'un autre nom d'utilitateur peut indiquer que celui-ci est le compte utilisateur d'un administrateur. Une fois le compte de l'administrateur trouvé, il suffit de regarder sur quelles machines il est connecté, faisant de ces machines des cibles (car potentiellement les machines d'administration, donc certainement moins restreintes aux niveaux des accès aux machines utilisant les *TCP Wrappers* ou un *firewall*). **ypcat**:

```
prompt$ domainname nis_domain
prompt$ ypwhich nis_server
prompt$ ypcat passwd > passwd_file
prompt$ ypcat -k netgroup > netgroup_file
```

Elévation de privilèges, 3ème reconnaissance

But

Le but est d'acquérir des droits *root* sur un segment *ethernet* où de nombreux mots de passe et autres données importantes circulent. Une fois *root* sur une telle machine, il suffit d'être patient, pendant la phase de *sniffing* du réseau. Une autre chose à accomplir est le *crackage* de comptes utilisateurs, le fichier de mots de passe crypté ayant été récupéré à l'étape précédente.

Logiciels utilisés

- *DSniff* (http://www.monkey.org/~dugsong/dsniff/), une suite d'outils pour *sniffer* ;
- *Snoop*, installé avec *Solaris*, ou *TcpDump* ;
- *John the Ripper* (rechercher le lien sur *Google*), *crackeur* de mots de passe ;

- Des *exploits* locaux.

Méthode

Théorie:

La première chose est de devenir *root* sur la machine compromise, si ce n'est pas déjà fait. Il suffit de rechercher une vulnérabilité locale sur celle-ci. En connaissant son *OS* et sa version, on peut trouver des *exploits* locaux pour accroître ses privilèges. Si aucun n'est trouvé pour cette machine, on peut soit trouver une autre machine avec un autre *OS* ou une autre version de celui-ci, soit chercher une vulnérabilité dans un logiciel tiers. Une fois *root*, on peut *sniffer* les informations. On peut également accroître ses privilèges en volant le compte d'un administrateur en *crackant* son mot de passe.

Pratique:

Devenir *root*:

```
prompt$ uname -sr SunOS 5.8
```

Recherche d'un *exploit* local pour cette version de *Solaris* sur *SecurityFocus*. On trouve une vulnérabilité dans la bibliothèque *libsldap*, avec un *exploit*.

```
prompt$ gcc exploit.c -o exploit
prompt$ ./exploit
prompt# id uid=0(root) gid=0(root)
```

Sniffer (*DSniff*, *Snoop*, *Siphon*): *DSniff* ne capture que les mots de passe. Il reconnait un grand nombre de protocoles parmis ceux-ci: *Rlogin*, *Telnet*, *snmp*, *NetBIOS*.

```
prompt# dsniff dsniff: listening on xl0

-----------------
08/08/01 11:07:57 udp host1.domain.fr.56030 -> host2.domain.fr.161 (snmp)
[version 1]
password

-----------------
07/17/01 16:02:03 tcp host84.domain.fr.1023 -> host10.domain.fr.513 (rlogin)
[root:root]
password
login
password
ls
who
```

Cracker les mots de passe: *John the Ripper*: Pour récupérer le fichier de mots de passe cryptés, sur la machine compromise:

```
prompt$ ypcat passwd > passwd_file
```

Il suffit ensuite de lancer le *cracker* en mode rapide sur ce fichier de mots de passe:

```
prompt$ john -single passwd_file
```

Compromission des serveurs

But

Maintenant qu'une machine du réseau nous appartient, il faut tenter de contrôler le réseau complet. A l'aide des informations récupérées précédemment, il suffit successivement de devenir *root* sur chacunes des machines les plus importantes (analyser le fichier **netgroup**).

Logiciels utilisés

Les logiciels utilisés ici sont les mêmes que pour l'étape d'intrusion sur le réseau.

Méthode

Théorie:

Grâce aux étapes de reconnaissance, la topologie du réseau est maintenant bien connue. Toutes les machines qui gèrent le réseau sont identifiées, toutes leurs failles également. Il ne reste plus qu'à continuer l'infection comme dans l'étape précédent, jusqu'à ce que le réseau soit entièrement contrôlé. On sait également, grâce à *Siphon*, quelle machine à le droit d'accéder à quelle machine (ou quel réseau à le droit d'accéder à quel réseau), et on peut tenter l' *IP Spoofing* pour accéder aux machines les plus sécurisées. Cette méthode peut parraître compliquée, mais il n'en est rien, nous allons le montrer tout de suite.

Pratique:

Le réseau 192.168.0 est autorisé à accéder au réseau 192.168.1. Nous sommes sur le réseau 192.168.10. Il faut que nous nous fassions passer pour une machine du réseau 192.168.0 (le plus facile, si l'on se trouve sur le même brin physique, est de trouver une adresse *IP* libre du réseau 192.168.0). Un routeur fait la jonction entre ces trois réseaux, et il supporte le *source routing* (c'est pour cela qu'il faut desactiver cette option, pour des raisons de sécurité). Il a donc trois interfaces réseau, 192.168.0.1, 192.168.1.1, et 192.168.10.1.

```
prompt# ifconfig vr0
vr0: flags=8843 mtu 1500
        inet 192.168.10.2 netmask 0xffffff00 broadcast 192.168.10.255
        ether 00:50:ba:a1:53:c7
        media: 100baseTX  status: no carrier
        supported media: autoselect 100baseTX  100baseTX 10baseT/UTP
10baseT/UTP none
prompt# ifconfig vr0 inet 192.168.0.25 netmask 0xffffffff add
prompt# ifconfig vr0
vr0: flags=8843 mtu 1500
        inet 192.168.10.2 netmask 0xffffff00 broadcast 192.168.10.255
        inet 192.168.0.25 netmask 0xffffffff broadcast 192.168.0.25
        ether 00:50:ba:a1:53:c7
        media: 100baseTX  status: no carrier
        supported media: autoselect 100baseTX  100baseTX 10baseT/UTP
10baseT/UTP none
```

Nous avons configuré un alias réseau sur notre interface en prenant une adresse *IP* appartenant au réseau privilégié. Il suffit maintenant de se connecter au réseau 192.168.1 en prenant comme adresse source cet alias. Pour cela, il faut utiliser une propriété du protocole *IP* qui est la possibilité de choisir la route à emprunter (le *source routing*). Sous *FreeBSD*, il faut autoriser l'envoie et la réception de paquets *IP* utilisant cette capacité, comme ceci (sous *FreeBSD 4.3*):

```
prompt# sysctl -w net.inet.ip.sourceroute=1
prompt# sysctl -w net.inet.ip.accept_sourceroute=1
prompt# sysctl -w net.inet.ip.forwarding=1
```

Autoriser ceci uniquement pendant votre utilisation de l' *IP Spoofing*, car vous pourriez vous exposer vous-même à des tentatives de *spoofing*. On peut ensuite utiliser l'utilitaire *NetCat* pour effectuer une connexion *telnet* en précisant la route à emprunter.

```
prompt# netcat -n -v -t -s 192.168.0.25 -g 192.168.10.2 -g 192.168.10.1
192.168.1.20 23
```

Cette commande permet de se connecter en *telnet* à la machine 20 du réseau 192.168.1 en prenant comme adresse source 192.168.0.25 (adresse faisant partie d'un réseau autorisé à se connecter au réseau 192.168.1), en passant d'abord par notre adresse *IP* (192.168.10.2: nous sommes devenu un routeur, car nous acceptons l' *IP forwarding*), puis en passant par le routeur qui possède 3 interfaces (192.168.10.1), puis on peut accéder à la cible (192.168.1.20). Puisque l'adresse 192.168.0.25 n'est pas utilisée, il n'y aura aucun conflit. Si tout va bien, vous êtes connecté dans le réseau 192.168.1. Il est à noter qu'il y a des restrictions à cette technique d'usurpation d'identité: il faut qu'il y ait moins de 8 routeurs entre la source et la destination. De plus, si vous vous trouvez sur le même brin physique que la machine pour laquelle vous voulez vous faire passer, il y aura un conflit d'adresse, lors de la réception des données, c'est-à-dire que 2 machines (vous, et celle pour qui vous voulez vous faire passer) recevront les données. Donc 2 machines répondront. Par contre, sur des brins physiques différents, il n'y aura qu'une machine d'une *IP* donnée sur ce brin, donc une seule machine répondra. Cet exemple est un exemple trivial, mais le principe fonctionne dans des réseaux plus complexe que celui-ci.

Fin du test, nettoyage

But

Le réseau est maintenant entièrement maîtrisé, ou tout ce qui à pu être tenté pour en arrivé là est terminé. Il faut nettoyer toutes les traces, tous les fichiers créés ou modifiés pour que le réseau soit de nouveau propre. Tous les problèmes de sécurité rencontrés ont été notés, un rapport doit maintenant être établi.

Logiciels utilisés

Un outil de nettoyage peut être développé pour accomplir cette tâche, mais le nettoyage à la main est également aisé, à condition d'avoir bien noté chacune des modifications que l'on a apporté au réseau.

Méthode

Suivre les notes prises pendant les étapes précédentes, dans le sens inverse.

Conclusion

C'est la fin de cette introduction aux tests d'intrusion, toutes les techniques et méthodes n'ont pas été abordées, mais un autre guide comme celui-ci sera peut-être écrit prochainement, traîtant d'aspects plus complexes que ceux évoqués ici. Pour rapporter une erreur, ou ajouter un commentaire, n'hésitez pas: webmaster@gomor.org.