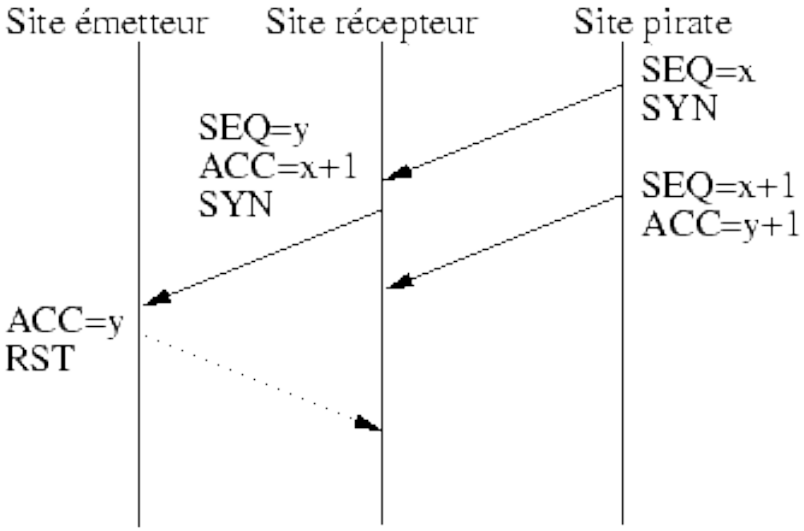
Techniques d'intrusions des réseaux



par :

Sophie LEVY

Fabien LE BLEVEC

Pierrick PREVERT

Ingénieurs 2000, Informatique et réseau 3ème année

Sous la tutelle d'Etienne DURIS,

professeur à l'Université de Marne la Vallée et à l'UFR Ingénieurs 2000.

Table des matières

1 Préambule

De nos jours, les attaques sur les réseaux sont traditions et même compétition. Les techniques découvertes sont nombreuses et celles non encore publiées doivent l'être tout autant. Il faut être réaliste et pragmatique devant ces menaces. Afin de protéger un réseau ou tout simplement ses données dans le sens noble du terme, il convient de bien connaître les failles d'un système (même potentiel) afin d'anticiper un maximum les attaques possibles. Comme une agence fournissant des coffres forts, ils convient de connaître comment les voleurs agissent !

2 Introduction

Ce document n'a pas la prétention de présenter toutes les attaques possibles, mais juste faire une présentation de certaines d'entres elle qui constitue de points d'entrée fréquents des attaques. plus communément, ces points d'entrée s'appelle des **failles**. mais qu'est ce qu'une faille. Pour paraphraser Mat Bishop, une faille est un comportement non prévu (ou non souhaitable) par le concepteur pouvant être provoqué par un tiers.

Nous présenterons d'un point de vue global et via quelques exemples 2 types de failles :

- Les failles de définitions de protocoles
- Les failles d'implémentation

Mais avant de lancer une attaque, un attaquant se renseigne le plus possible, comme un voleur le ferait. Nous illustrerons cela avant de commencer le vif du sujet.

3 La recherche d'informations

3.1 Présentation

3.2 Recherche passive

3.2.1 Whois

La base **Whois** est une base renseignée par les regitrars Internet. Elle renseigne sur l'appartenance d'une adresse IP ou de noms de domaines. **Whois** est normalisé par la RFC 3912.

Grâce à cette base, il est possible :

- Nom de domaine
- Registrar (organisme s'occupant de la gestion de nom de domaine)
- URL du serveur Whois de ce registrar
- URL du site du registrar
- Serveurs DNS (dans la base Internic)
- Date de dernière modification
- Adresse du propriétaire du nom de domaine
- Adresse de l'administrateur du nom de domaine
- Adresse du contact technique du nom de domaine
- Date de modification, création, et expiration du nom de domaine
- Serveurs DNS (dans la base du registrar)

Voici un exemple de récupération d'information complète : Dans un premier temps, nous allons utiliser la base **whois** pour récupérer des informations sur le nom de domaine :

```
Domain Name: BIENF.NET
Registrar: GANDI
Whois Server: whois.gandi.net
Referral URL: http://www.gandi.net
Name Server: NS7.GANDI.NET
Name Server: CUSTOM2.GANDI.NET
Status: ACTIVE
Updated Date: 21-mar-2005
Creation Date: 12-apr-2002
Expiration Date: 12-apr-2006

...

domain: BIENF.NET
owner-name: Fabien LE BLEVEC
owner-address: 6 rue Danville
owner-address: 75014
```

```
owner-address: PARIS
owner-address: France
owner-phone: +33.664829194
owner-e-mail: 2c5fd06f42ad97a8f432ce39df24c22a-578973@owner.gandi.net
admin-c: FLB8-GANDI
tech-c: FLB8-GANDI
bill-c: FLB8-GANDI
nserver: ns7.gandi.net 217.70.177.44
nserver: custom2.gandi.net 217.70.179.35
reg_created: 2002-04-12 07:12:05
expires: 2006-04-12 07:12:05
created: 2002-04-12 13:12:05
changed: 2005-03-21 22:09:51

person: Fabien LE BLEVEC
nic-hdl: FLB8-GANDI
address: 6 rue Danville
address: 75014
address: PARIS
address: France
phone: +33.664829194
e-mail: 6fccd2a0198d6aab99dcbbe317ae69e6-flb8@contact.gandi.net
lastupdated: 2004-05-26 09:25:43
```

Un simple **nslookup** permet de déterminer l'adresse IP de la machine cible.

```
#nslookup www.bienf.net
Server:         212.27.32.176
Address:        212.27.32.176#53

www.bienf.net   canonical name = gribouille.bienf.net.
Name:   gribouille.bienf.net
Address: 82.235.111.96
```

Nous voyons ici que nous avons récupérer l'adresse IP du serveur grâce à une simple requête DNS et à l'outil **nslookup** ...

Nous pouvons maintenant récupérer des informations sur la topologie Internet qui connecte cette adresse IP. Grâce aux bases gérées par les organismes d'attributions des adresses IP sur Internet, cela est très rapide. Par exemple, nous allons sur la base **whois** du RIPE (*www.ripe.net*) :

```
inetnum:         82.235.108.0 - 82.235.111.255
netname:         FR-PROXAD-ADSL
descr:           Proxad / Free SAS
descr:           Static pool (Freebox)
descr:           mts78-1 (cbv)
descr:           NCC#2005090519
country:         FR
admin-c:         ACP23-RIPE
tech-c:          TCP8-RIPE
status:          ASSIGNED PA "status:" definitions
remarks:         Spam/Abuse requests: mailto:abuse@proxad.net
```

```
mnt-by:          PROXAD-MNT
source:          RIPE # Filtered

role:            Administrative Contact for ProXad
address:         Free SAS / ProXad
address:         8, rue de la Ville L'Eveque
address:         75008 Paris
phone:           +33 1 73 50 20 00
fax-no:          +33 1 73 50 25 01
remarks:         trouble:      Information: http://www.proxad.net/
remarks:         trouble:      Spam/Abuse requests: mailto:abuse@proxad.net
admin-c:         RA999-RIPE
tech-c:          NH1184-RIPE
nic-hdl:         ACP23-RIPE
mnt-by:          PROXAD-MNT
source:          RIPE # Filtered
abuse-mailbox:   abuse@proxad.net

role:            Technical Contact for ProXad
address:         Free SAS / ProXad
address:         8, rue de la Ville L'Eveque
address:         75008 Paris
phone:           +33 1 73 50 20 00
fax-no:          +33 1 73 50 25 01
remarks:         trouble:      Information: http://www.proxad.net/
remarks:         trouble:      Spam/Abuse requests: mailto:abuse@proxad.net
admin-c:         RA999-RIPE
tech-c:          NH1184-RIPE
nic-hdl:         TCP8-RIPE
mnt-by:          PROXAD-MNT
source:          RIPE # Filtered
abuse-mailbox:   abuse@proxad.net

% Information related to '82.224.0.0/11AS12322'

route:           82.224.0.0/11
descr:           ProXad network / Free SAS
descr:           Paris, France
origin:          AS12322
mnt-by:          PROXAD-MNT
source:          RIPE # Filtered
```

Nous pouvons voir que le site est hébergé sur une connexion appartenant à *Proxad* (*alias Free Telecom*). Nous pouvons même voir que dans la description, les adresses IP de ce pool sont fixes (*descr: Static pool (Freebox)*).

3.2.2 Moteur de recherche

Les moteurs de recherches sont une aide précieuse dans la récupération d'informations. Ils procurent par exemple le moyen de connaître des noms de machines et d'en déduire une logique de noms afin de découvrir d'autres noms d'hôtes. Il est aussi possible de récupérer simplement des adresses e-mail qui sont souvent liés à de simple nom

d'utilisateur (comme par exemple ceux de l'université de Marne-La-Vallée).

3.2.3 Le sniffing

Cette technique demande d'être sur un réseau de type "broadcast" comme Ethernet afin de récupérer un trafic qui n'est pas à destination de la machine qui écoute. Il faut bine sur que le réseau Ethernet en question n'utilisent pas de switchs car sinon les trames récupérées ne seront que les nôtres ! Cependant une re-direction de flux permettrait de récupérer pendant un instant un trafic illégitime.

Le sniffing de trames (ou écoute de trames) peut permettre de récupérer des informations sensibles telles que des mots de passes ou logins.

Les outils les plus communs pour faire du sniffing sont **Tcpdump** et **Ethereal**.

3.3 Recherche active

3.3.1 Récupération d'informations DNS

La commande **host** permet de récupérer sur le serveur DNS laissant sortir trop d'informations au publique des informations sur la topologies d'un réseau. Il est probable par exemple que la commande sous Linux ***host -a -l bienf.net*** permet de lister toutes les hôtes d'un domaine.

La plupart du temps, les serveurs DNS sont bien configurés et les requêtes de transferts de zones sont filtrées aux hôtes légitimes (comme les serveurs DNS secondaires).

3.3.2 Le firewalking

Présentation

Le firewalking est un ensemble de techniques permettant de faire une cartographie du réseau et d'en déduire la présence des points de protection tels que les firewalls. Il est basé sur les techniques utilisant le champs TTL des datagrammes IP tels qu'utiliser par le très connu utilitaire **traceroute**.

Traceroute

Traceroute est l'utilitaire le plus célèbre et le plus simple pour découvrir la topologies d'un réseau. Son but initiale est d'aider à des fins de dépannage un administrateur réseau. Traceroute utilise le champs TTL et le comportement de la pile IP afin de découvrir les éléments vers l'hôte cible.

L'hôte source envoie des paquets (UDP ou ICMP en général) avec un TTL à 1. Passer le premier routeur, ce champs est décrémenté de 1 et un paquet *ICMP Time Exceeded* est

renvoyé à l'hôte. En incrémentant à chaque fois le TTL, on arrive à l'hôte cible qui répond soit par un *ICMP Echo reply* dans le cas de l'utilisation d'*ICMP*, soit d'un autre paquets d'acquittement ou autres.

```
# traceroute -l -v 193.110.152.241
traceroute to 193.110.152.241 (193.110.152.241), 30 hops max, 38 byte packets
 1  192.168.0.2 (192.168.0.2) 46 bytes to 192.168.0.3  0.956 ms (64)  0.793 ms
(64)  0.761 ms (64)
 2   82.235.111.254 (82.235.111.254) 46 bytes to 192.168.0.3  26.653 ms (63)
27.198 ms (63)  27.105 ms (63)
 3   213.228.8.254 (213.228.8.254) 36 bytes to 192.168.0.3  26.750 ms (252) *
27.013 ms (252)
   4     th1-6k-2-v802.intf.routers.proxad.net  (212.27.50.33)  36  bytes  to
192.168.0.3  26.801 ms (251)  26.649 ms (251) *
  5    GE15-0.nosta102.Paris.francetelecom.net  (193.252.160.153)  36  bytes  to
192.168.0.3  27.287 ms (249)  27.404 ms (249)  42.784 ms (249)
  6    pos11-0.ntsta302.Paris.francetelecom.net  (193.251.126.33)  36  bytes  to
192.168.0.3  27.190 ms (249)  27.440 ms (249)  27.076 ms (249)
  7    pos10-0.nrsta304.Paris.francetelecom.net  (193.252.161.77)  36  bytes  to
192.168.0.3  27.581 ms (248)  27.412 ms (248)  27.751 ms (248)
 8   pos0-0-0-0.ncidf304.NeuillySurMarne.francetelecom.net (193.252.103.173) 36
bytes to 192.168.0.3  27.929 ms (246)  27.934 ms (246)  37.886 ms (246)
 9  193.252.227.179 (193.252.227.179) 36 bytes to 192.168.0.3  28.375 ms (246)
27.979 ms (246)  28.019 ms (246)
10  * * *
11  * * *
12  * *
```

La commande **traceroute** nous montre que la cible semble être protégée par un élément du réseau. Nous voyons ici qu'au saut n° 10, un élément ne répond pas. En changeant le comportement de **traceroute** avec l'option -I, nous lui indiquons d'utiliser le protocole ICMP plutôt qu'UDP. Le résultat est le même. Nous pouvons donc en déduire que cet élément filtre le trafic réseau.

Il est relativement simple de se prémunir d'un simple **traceroute** en interdisant un champs TTL de 1 arrivé sur le réseau.

L'outils Hping

Hping permet de forger soit même ses propres paquets. Il est tout à fait possible de simuler le comportement de l'utilitaire **traceroute** en utilisant des paquets TCP au lieu de l'UDP ou de l'ICMP.

Il est ainsi possible passer un firewall qui bloquerai juste les requêtes ICMP ou UDP. Avec l'option -t , il est possible par exemple de modifier le champs TTL d'un paquet IP. Avec l'option -p, de spécifier le port TCP à utiliser , avec les flags voulu (-A : Ack, -S : Syn, -R : Reset, -F Fin).

Pour plus d'informations sur les options, nous vous conseillons le site officiel de Hping http://www.hping.org/.

3.3.3 Le scan de port

Un scan de ports permet :

- de déterminer les ports ouverts sur une machine
- de déterminer le système d'exploitation d'une machine (par le fingerprinter de sa pile TCP/IP)

L'outil de référence pour le scan de port est **Nmap**. Un exemple de scan avec les options par défaut pourrait être :

```
# nmap 192.168.0.1

Starting nmap 3.81 ( http://www.insecure.org/nmap/ ) at 2006-01-23 00:49 CET
Interesting ports on 192.168.0.1:
(The 1652 ports scanned but not shown below are in state: closed)
PORT     STATE SERVICE
21/tcp   open  ftp
25/tcp   open  smtp
53/tcp   open  domain
80/tcp   open  http
81/tcp   open  hosts2-ns
111/tcp  open  rpcbind
139/tcp  open  netbios-ssn
893/tcp  open  unknown
913/tcp  open  unknown
2049/tcp open  nfs
3128/tcp open  squid-http
MAC Address: 00:04:76:24:D0:DF (3 Com)

Nmap finished: 1 IP address (1 host up) scanned in 0.802 seconds
```

Nous pouvons utiliser l'option -O pour rechercher sur quel type d'OS, il y a en face :

```
# nmap -O 192.168.0.1

Starting nmap 3.81 ( http://www.insecure.org/nmap/ ) at 2006-01-23 00:50 CET
Interesting ports on 192.168.0.1:
(The 1652 ports scanned but not shown below are in state: closed)
PORT     STATE SERVICE
21/tcp   open  ftp
25/tcp   open  smtp
53/tcp   open  domain
80/tcp   open  http
81/tcp   open  hosts2-ns
111/tcp  open  rpcbind
139/tcp  open  netbios-ssn
893/tcp  open  unknown
913/tcp  open  unknown
2049/tcp open  nfs
3128/tcp open  squid-http
MAC Address: 00:04:76:24:D0:DF (3 Com)
Device type: general purpose
Running: Linux 2.4.X|2.5.X
```

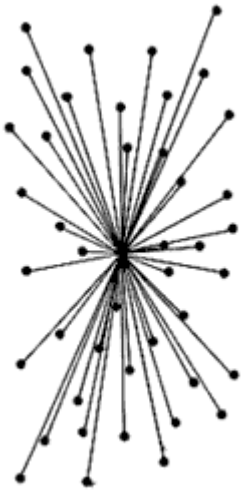
```
OS details: Linux 2.4.0 - 2.5.20
Uptime 63.475 days (since Sun Nov 20 13:26:20 2005)

Nmap finished: 1 IP address (1 host up) scanned in 3.035 seconds
```

Nmap utilise les réponses d'un scan pour rechercher une empreinte dans sa base de connaissance. Ainsi plus la cible cache ses informations, et plus **nmap** aura du mal à avoir une empreinte fiable pour la reconnaissance.

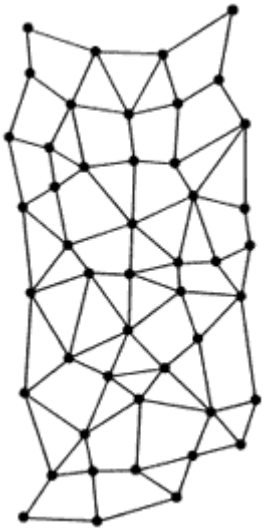
4 Les failles de définition de protocoles

4.1 Introduction

Historiquement, Internet a été développé autour de la notion d'une sécurité de l'infrastructure. Dès 1964, dans un contexte de guerre froide, Paul Baran **[1]**, un des principaux créateurs d'Internet, eût l'idée de créer un réseau sans noyau central (à l'époque l'architecture couramment employée était celle en étoile) car la destruction du noyau de ce réseau entraînait invariablement la chute de l'ensemble des connexions. Il inventa alors le "packet switching" (à l'époque intitulé par Baran le "hot-potato routing"), une toile sans point central où des blocs d'information se déplacent de "hop" en "hop", les matériels déterminant à chaque arrêt le chemin idéal. C'est de cette technologie que découle l'Internet tel qu'on le connait : en 1983, la partie militaire d'ARPANET devint MILNET et la partie restante resta nommée ARPANET jusqu'en 1989, pour s'intégrer définitivement dans l'Internet.



Réseau en étoile **[1]**



Réseau « toile » **[1]**

L'ensemble des protocoles développés pour ARPANET et donc Internet, le furent dans cette optique de la sécurité de l'infrastucture qui apparaissait, toujours dans ce contexte historique, comme la réelle menace. Cependant Internet s'est considérablement développé et s'est étendu au monde entier : près de 250 pays possèdent aujourd'hui leur propre TLD (Top Level Domain) **[2]**. Ainsi le problème de la centralisation a quasiment disparu et bien que persistent de nombreux commutateurs centraux le danger se situe maintenant au niveau des couches supérieures, notamment la couche réseau et, bien que ce ne soit pas le sujet de ce rapport, plus récemment la couche applicative.

Ce développement sans une conscience des implications de la sécurité à tous les niveaux soulève plusieurs problèmes. Notamment : comment deux systèmes qui souhaitent communiquer ensemble peuvent s'assurer de leur identité mais aussi comment peuvent-ils être sûrs que les données transmises n'ont pas été altérées par un tiers ? Actuellement, bien qu'un effort global est mis en oeuvre pour parer à ce type de problème avec le développement de SSL au niveau applicatif ou, au niveau réseau, de la conception d'IPv6, de DNSsec etc. qui intègrent dans les couches inférieures des mécanismes jusque là réservés aux couches applicatives, aucune définition des protocoles les plus largement utilisés ne permet de résoudre ces problèmes. Par conséquent, les protocoles actuels sont susceptibles d'être attaqués au niveau de leur définition.

Dans ce rapport trois protocoles courants, de couche OSI différentes, sont présentés : ARP, TCP et DNS.

4.2 ARP

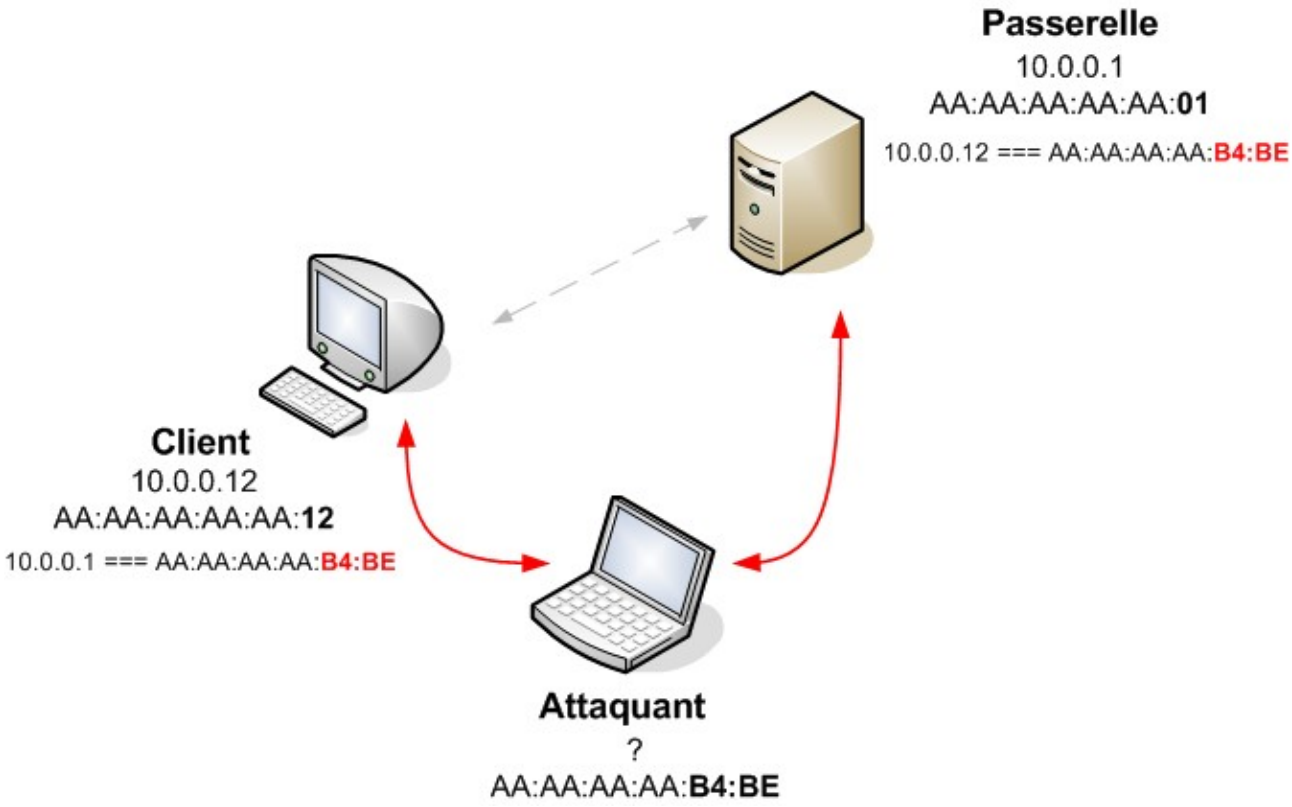
Le protocole ARP **[3]** implémente, dans un réseau ethernet, une solution de résolution des adresses IPs (Internet Protocol, 32 bits) en adresses MAC (Media Access Control, 48 bits) utilisées pour l'adressage physique. Dans un réseau IP sur de l'ethernet, toute machine souhaitant communiquer avec une autre doit connaître l'adresse MAC de la machine destinataire. Cette résolution s'effectue avec le protocole ARP. ARP est un protocole sans état qui fonctionne par requêtes envoyées en broadcast. Quand la machine destinatrice reçoit la requête ARP, elle répond à l'émetteur, le renseignant sur son adresse MAC. L'émetteur enregistre cette information dans un cache pendant un certain temps, redemandant régulièrement afin d'être sûr que l'autre machine reste accessible et est alors capable de communiquer directement avec la machine destinataire. Plusieurs problèmes sont soulevés par ce fonctionnement.

4.2.1 Principe du Spoofing ARP

Tout d'abord un problème mineur mais très difficilement corrigible : dans sa définition ARP est un protocole sans état. Une réponse reçue est donc toujours considérée comme faisant suite à une question, que ce soit le cas ou pas. Dès lors, il est possible d'envoyer une réponse à une machine cible en indiquant qu'une adresse IP choisie possède une adresse MAC particulière. La machine cible enregistrera donc dans son cache la réponse en supposant avoir posé la question préalablement.

Un scénario d'attaque classique est celui du "Man in the middle" (souvent abbrégé MiM) : un attaquant (C) utilise cette technique pour indiquer à deux machines (A et B) qu'il est l'une et l'autre : auprès de A il annonce qu'il est B, auprès de B il annonce qu'il est A. Il recevra alors l'intégralité du trafic destiné à l'une et à l'autre. De là les possibilités sont multiples, du simple sniffing à la modification des données qui transitent. Cette technique

utilisée contre une passerelle et un poste client est particulièrement efficace.



Spoofing ARP (Man in the Middle / MiM)

4.2.2 Les solutions

Des mesures ont été prises dans les différentes piles avant d'empêcher ces malversasions. Tous ces patchs par l'implémentation mettent en défaut le standard. Par exemple :

- S'assurer qu'une requête a été émise avant de traiter une réponse. Cependant, comme les demandes sont cycliques et que la temporisation par défaut est connue, cette approche n'apporte pas une avancée sécuritaire notable.
- Le kernel enverra une requête ARP pour tester si il y a un host à l'ancienne adresse MAC (fonctionnement du patch Antidote). Si tel est le cas, une alerte sera remontée indiquant qu'une IP dit posséder plusieurs adresses MAC et que cela est probablement une attaque de type spoofing ARP. Cette défense peut cependant être dépassée si l'hôte qui l'implémente ne possède pas d'entrée ARP pour l'host véritable. Auquel cas l'host véritable sera banni en tentant de répondre.

4.2.3 Conclusion

ARP n'est donc pas un protocole dont la définition permet de s'assurer que l'adresse ethernet renvoyée correspond bien à l'adresse IP demandée. Par conséquent, il est déconseillé de l'utiliser dans un environnement souhaitant être sécurisé et préférer ajouter des entrées statiques dans la table ARP. Une autre solution est d'utiliser un protocole de couche supérieur, comme SSL, pour permettre un mécanisme d'identification.

4.3 TCP

Sur Internet, TCP **[4]** est le protocole de transport, donc de niveau 4 sur le modèle OSI, principalement utilisé. Ce protocole qui fonctionne de manière connectée permet de s'assurer de la bonne réception des données, de gérer le flot de données, de fournir aux protocoles supérieurs (vraisemblablement IP) des segments de longueur différentes, mais aussi de multiplexer par le biais d'un mécanisme de "ports" différentes applications. C'est donc un protocole relativement complexe de par l'étendue du domaine qu'il couvre : l'envoi et la réception de données de manière sûre, en prenant en considération les contraintes de pertes éventuelles de données. Dans le cadre d'Internet, TCP repose sur IP et hérite donc des limitations de celui-ci : l'en-tête TCP apparaît en clair dans un datagramme IP.

Comme TCP est le protocole de prédilection pour le transport de données, de nombreux protocoles de niveau 7 l'utilisent. BGP (Border Gateway Protocol), par exemple, est le protocole de routage principalement utilisé sur Internet pour relier deux réseaux distants. L'échange d'information se fait via TCP : les routeurs ouvrent une connexion entre eux et restent connectés en permanence.

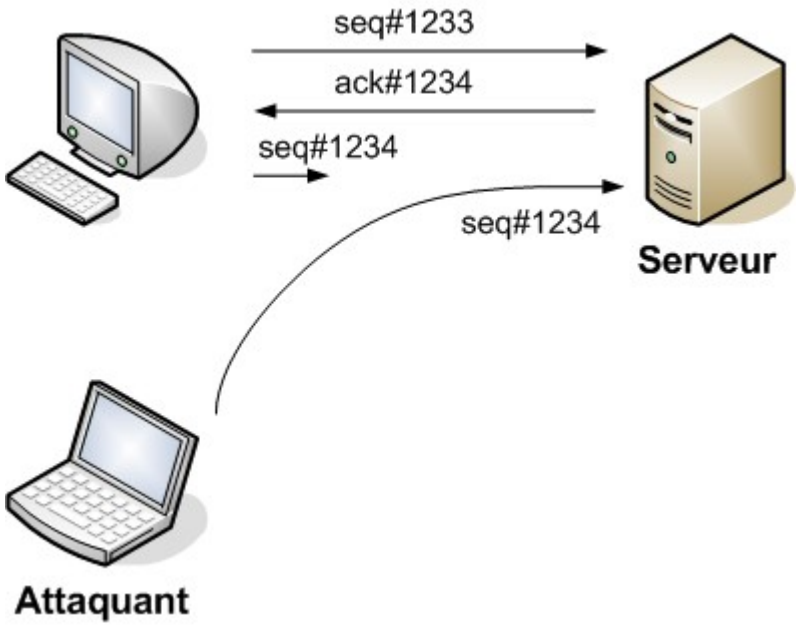
Pour pouvoir gérer efficacement la bonne réception des données, TCP possède un mécanisme numéro de séquence. Un numéro de séquence est un nombre codé sur 32 bits, choisi aléatoirement - bien qu'à l'origine et dans la spécification de TCP il était choisi de façon incrémentale. Chaque octet de donnée envoyé possède, selon la RFC, un numéro de séquence qui nécessite d'être acquitté. Si l'acquittement n'est pas reçu dans un intervalle de temps donné, il doit être ré-émis. De plus TCP fonctionne par fenêtre, qui indique le numéro d'octet que récepteur souhaite pouvoir recevoir sans accusé de réception.

4.3.1 Principe du Spoofing TCP

Le "spoofing TCP", ou "connection hijacking" consiste à envoyer des données dans une fenêtre TCP valide. La seule limitation pour un attaquant est de trouver un numéro de séquence valide tombant dans la fenêtre actuelle. Etant donnée la taille des fenêtres généralement utilisées, ce numéro se retrouve considérablement affaibli. Cependant, la probabilité de tomber dessus nécessite en essayant toutes les combinaisons (bruteforce)

un temps considérable, même avec les connexions actuelles.

Une première façon de trouver le numéro de séquence est, dans le cas d'un réseau local, de sniffer un lien entre deux machines. Pour se faire, si le réseau est switché, il peut être nécessaire d'utiliser une faille d'implémentation des switchs ou encore l'ARP spoofing préalablement indiqué. Une fois que le numéro de séquence est connu il devient très simple d'envoyer des segments qui seront acceptés par la machine cible.



Représentation d'un spoofing TCP

Une des limitations du Spoofing TCP vient du fonctionnement de celui-ci. Dans le cas d'une machine attaquante C, d'une machine cible A et d'un client B. Si C envoie un segment valide à A en se faisant passer pour B. A répondra alors à B qu'il a bien reçu les données qui lui ont été transmises. Si C n'a aucun moyen de stopper cet acquittement (pas dans le cas d'un MiM), B recevra cet acquittement et, s'apercevant qu'il y a une erreur, renverra un paquet ACK à A avec le numéro de séquence correct. A répondra alors à son tour, B répondra aussi etc. ceci provoquant une boucle (presque) sans fin : c'est un "ACK storm". La fin du « ACK storm » s'arrêtera avec le premier paquet ACK perdu et, comme TCP spécifie que les paquets d'acquittement ne doivent pas être retransmis, la boucle s'arrêtera à ce moment.

La plupart du temps un ACK storm, outre qu'il est "bruyant", n'est pas dérangeant pour un attaquant : il y a très peu de chance que le serveur ciblé se retrouve déconnecté sous le flot des ACKs de B. Cependant, pour rendre l'attaque plus propre, il est courant d'attaquer aussi la machine cliente avec un DoS (Denial of Service) afin que celle-ci cesse tout trafic sans que A en soit informé. Une technique similaire a notamment été utilisée par Kevin Mitnick pour le "hack" de 1994 qui l'a rendu célèbre. Cette technique était à l'époque connue et envisagée, mais aucune utilisation de celle-ci n'avait été constatée.

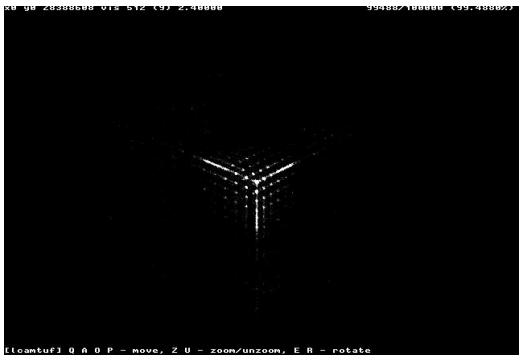
4.3.2 Le numéros de séquence comme dernier rempart

Les numéros de séquence, créés pour s'assurer du bon ordre de réception des paquets, se sont donc vus confier progressivement, entre leur création et leur utilisation à large échelle, un rôle sécuritaire : ils sont devenus le garant de l'intégrité de la connexion alors que leur définition ne le prévoyait pas. Ce problème qui est devenu une faille de définition du protocole - certains diront d'utilisation - a poussé les développeurs à corriger ces faiblesses au niveau de l'implémentation. Une des premières mesure a été prise au niveau des numéros de séquence utilisés pour l'initiation de la connexion (ISNs - Initial Sequence Numbers) : au lieu d'être choisis en se calquant sur un timer avec un incrément toutes les 4 microsecondes, comme précisé dans la RFC, ceux-ci ont désormais été choisi aléatoirement.

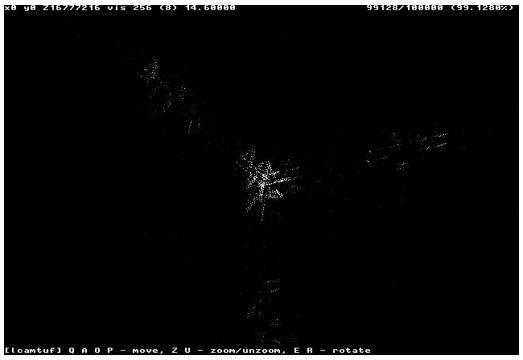
L'impossibilité technique, provoquée par la bande passante, de tester tous les numéros de séquence possible a donc été renforcée par le générateur de nombre pseudo-aléatoires employé pour créer l'ISN. Ensuite, comme certaines piles TCP/IP employaient des générateurs aléatoires faibles de nouveaux standards ont vu le jour, notamment la RFC1750, qui indique les considérations cryptographiques à prendre en compte dans le cas de la génération de nombres aléatoires.

Cependant, certaines faiblesses persistent. Notamment, on peut retrouver des faiblesses d'implémentations de ces recommandations. Il n'y a pas si longtemps, fin 2000, la génération des ISNs utilisée dans la pile TCP/IP de FreeBSD tombait lors d'un audit **[6]** de Pascal Bouchareine, aka kalou du groupe HERT **[7]**. Une autre étude, publiée un an plus tard par Michal Zalewski, aka lcamtuf, a permis de faire le point sur les générateurs employés, en utilisant l'espace des phases, une méthode d'analyse couramment utilisé en mécanique quantique ou dans l'étude des mathématiques du chaos. Cette étude dont quelques résultats "graphiques" sont présentés en dessous a permis de mettre à jour des vulnérabilités pour Cisco IOS, Win98 et WinNT. En 2002, soit un an après, Michal Zalewski a recommencé l'étude et commente l'évolution des résultats, notamment le patch fourni par Cisco **[8]**.

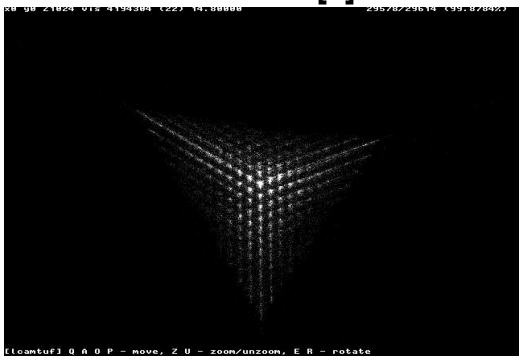
Les images présentées proviennent de l'ancienne étude. Les probabilités de prédiction des prochains identifiants étaient, en 2001, de 97% pour Windows NT, 100% pour Windows 98 SE, 20% pour Cisco IOS et de 3.05% pour OpenBSD.



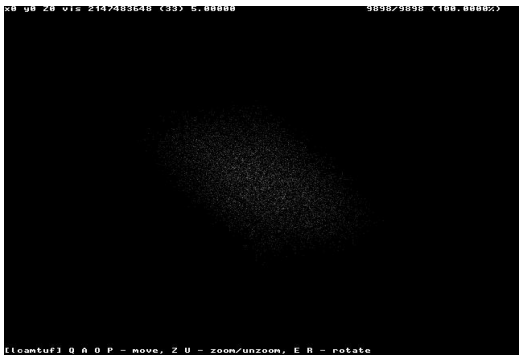
Windows 98 **[8]**



Windows NT **[8]**



Cisco IOS **[8]**



OpenBSD **[8]**

4.3.3 Les solutions

Pour circonscrire ces problèmes, la plupart des applications utilisent SSL dans les couches supérieures, pour compenser la faiblesse des couches inférieures du modèle OSI par un verrou mathématique. Cependant, il faut bien voir là que le problème n'est en aucun cas résolu, mais caché, et que la sécurité de TCP ne tient qu'à une limitation technique et une autre mathématique : tout d'abord, la bande passante n'est actuellement pas suffisante pour trouver un ISN de manière fiable en peu de temps, ensuite, les numéros de séquence ne sont protégés que par l'impossibilité de prévoir l'ISN suivant à partir d'un ou de plusieurs ISNs précédents - le principe même d'un générateur de nombres pseudo-aléatoires. Cependant, et bien que ce ne soit toujours pas une solution, les vulnérabilités de TCP peuvent être mitigées par l'utilisation d'IPsec et peuvent donc aussi l'être par l'arrivée d'IPv6 : le chiffrement et l'identification dans les couches inférieures empêchent l'insertion et la modification du contenu des datagrammes, donc des segments.

4.3.4 Conclusion

TCP, dans sa version actuelle ne permet pas de s'assurer une quelconque intégrité des données, si ce n'est par l'imprédictabilité supposée de l'ISN. Un segment n'est considéré

que valide que si le l'adresse IP source et destination correspondent bien (couche IP, niveau 3), mais aussi si le numéro de séquence correspond ainsi que le port source et destination - ces deux derniers peuvent facilement être déduits.

4.4 DNS

Le protocole DNS est utilisé pour la résolution de noms de domaine, c'est à dire effectuer la correspondance adresse IP / nom de domaine et inversement. Ce protocole est basé sur UDP et fonctionne suivant un principe de question-réponse. Quand un client souhaite résoudre une adresse IP ou un nom de domaine, il effectue une demande au serveur DNS configuré pour sa connexion. Si celui-ci possède la réponse dans son cache, il lui répond directement. En revanche, si ce n'est pas le cas, il demande, en suivant la hiérarchie imposée par le protocole à un autre serveur DNS.

Comme cette résolution de nom n'est pas instanée, les requêtes possèdent un numéro d'identifiant, codé sur 16 bits. Ainsi le serveur garde une trace des requêtes effectuées et sait en observant le numéro de la réponse reçue à quelle question elle correspond. Le mécanisme de réception d'une requête est simple : quand le serveur reçoit une réponse, souvent encapsulée dans UDP et IP, il analyse tout d'abord l'IP source et l'IP destination afin de s'assurer qu'elles correspondent bien puis passe le paquet à la couche UDP, qui vérifie que le port source et destination correspondent, qui à son tour passe le paquet à la couche DNS qui vérifie que l'identifiant correspond à une requête effectuée.

4.4.1 Principe du Spoofing DNS

Tout comme le spoofing TCP, l'impossibilité supposée du spoofing DNS repose sur l'imprédictabilité de l'identifiant de requête. Très tôt, en 1999, de nombreuses personnes dont raw du groupe ADM **[10]** se sont basés sur les travaux de SNI (Secure Network Inc.) et ont dénoncé une pléthore de moyens pour casser la sécurité apportée par cet identifiant. Là encore, le but premier de l'identifiant de requête DNS n'était pas d'apporter une sécurité : dans sa version proposée initialement, il était conseillé d'incrémenter l'identifiant à chaque requête. Cependant, tout comme TCP, DNS ne possède aucun mécanisme de protection, aussi la sécurité de ce protocole s'est décentralisée sur le seul point qui ne pouvait pas être prédit de "l'extérieur" : l'identifiant.

Le principe est le même : il s'agit de répondre avant le serveur a qui était adressée la question. De la même manière qu'avec TCP, si l'attaquant peut sniffer les requêtes DNS ou est capable de rendre injoignable le serveur a qui était adressée la question, il est alors très aisé de répondre avant le vrai serveur DNS en usurpant son identité et alors corrompre le cache du serveur DNS. Les conséquences d'une attaque DNS peuvent être désastreuses sur un serveur DNS très utilisé comme celui-ci enregistrera dans son cache une information erronée. Si les protocoles de couches supérieures ne vérifient pas

l'authenticité de l'hôte auquel ils se connectent, des données personnelles peuvent être volées.

4.4.2 Analyses de l'identifiant DNS

Se basant sur un fonctionnement "une requête reçue, une requête envoyée", BIND s'est rendu vulnérable à une attaque originale, basée sur le principe du "Paradoxe d'anniversaire", qui est nommée "L'attaque d'anniversaire" (Birthday Attack). Cette attaque est habituellement utilisée pour tenter de casser des systèmes de hashs. En envoyant "n" requêtes au serveur et autant de fausses réponses avec des identifiants différents, la probabilité statistique que l'identifiant d'une fausse réponse corresponde à celui d'une requête augmente considérablement. Ainsi, avec seulement 700 requêtes et donc 700 fausses réponses, il est possible d'atteindre un taux de réussite 99%.

$$Probability\ of\ Collision = 1 - \left(1 - \frac{1}{t}\right)^{\frac{n \times (n-1)}{2}}$$

Formule de calcul de probabilité d'une « attaque d'anniversaire » **[11]**

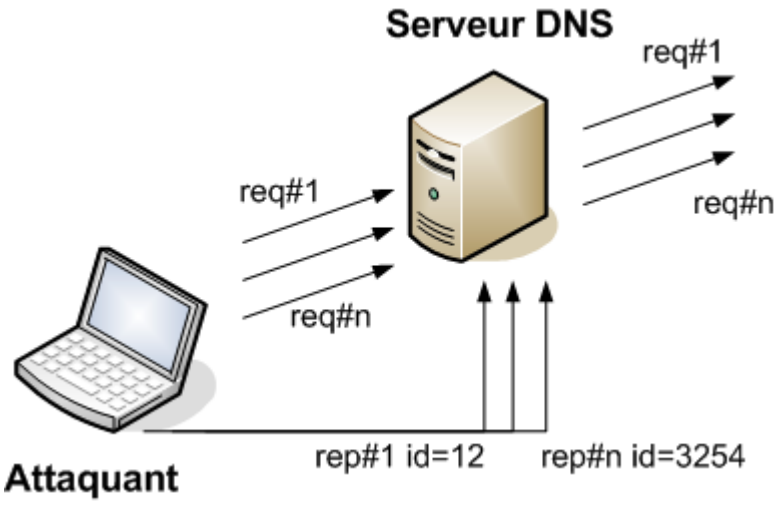
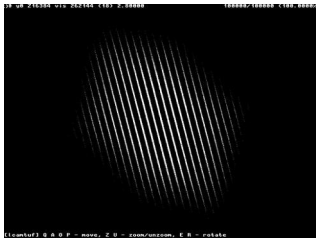
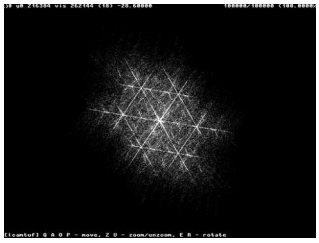


Schéma d'une attaque d'anniversaire

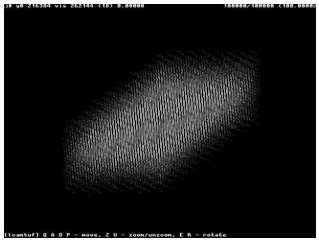
Aussi une analyse des phases, reprenant l'idée lancée par Zalewski, a été réalisée sur les générateurs de nombre pseudo-aléatoires utilisés pour générer les identifiants. Les résultats donnent BIND 8 prévisible à 100%, BIND 9 à 20% et djbdns, de Daniel Bernstein, prévisible à à 30%.



Bind 8 **[11]**



Bind 9 **[11]**



djbdns **[11]**

4.4.3 Solutions

BIND 9, jusqu'à récemment, n'intégrait aucun mécanisme de sélection d'un port source pour la requête, seulement un fonctionnement incrémental. En revanche, djbdns intègre depuis le début un mécanisme de sélection de port source aléatoire. Aussi, le problème se complexifie : il ne s'agit non plus de trouver simplement l'identifiant, mais l'identifiant ainsi que le port source utilisé par UDP - cette tâche est autrement plus difficile. Cependant, dans un réseau local où il est possible de sniffer les requêtes, le spoofing DNS devient très facile.

DNSsec est actuellement développé pour permettre de résoudre ce problème. Cependant, le problème étant complexe, les spécifications tardent et l'on risque d'attendre encore quelques années avant que DNSsec devienne le protocole de référence pour la résolution de noms.

4.5 Conclusion

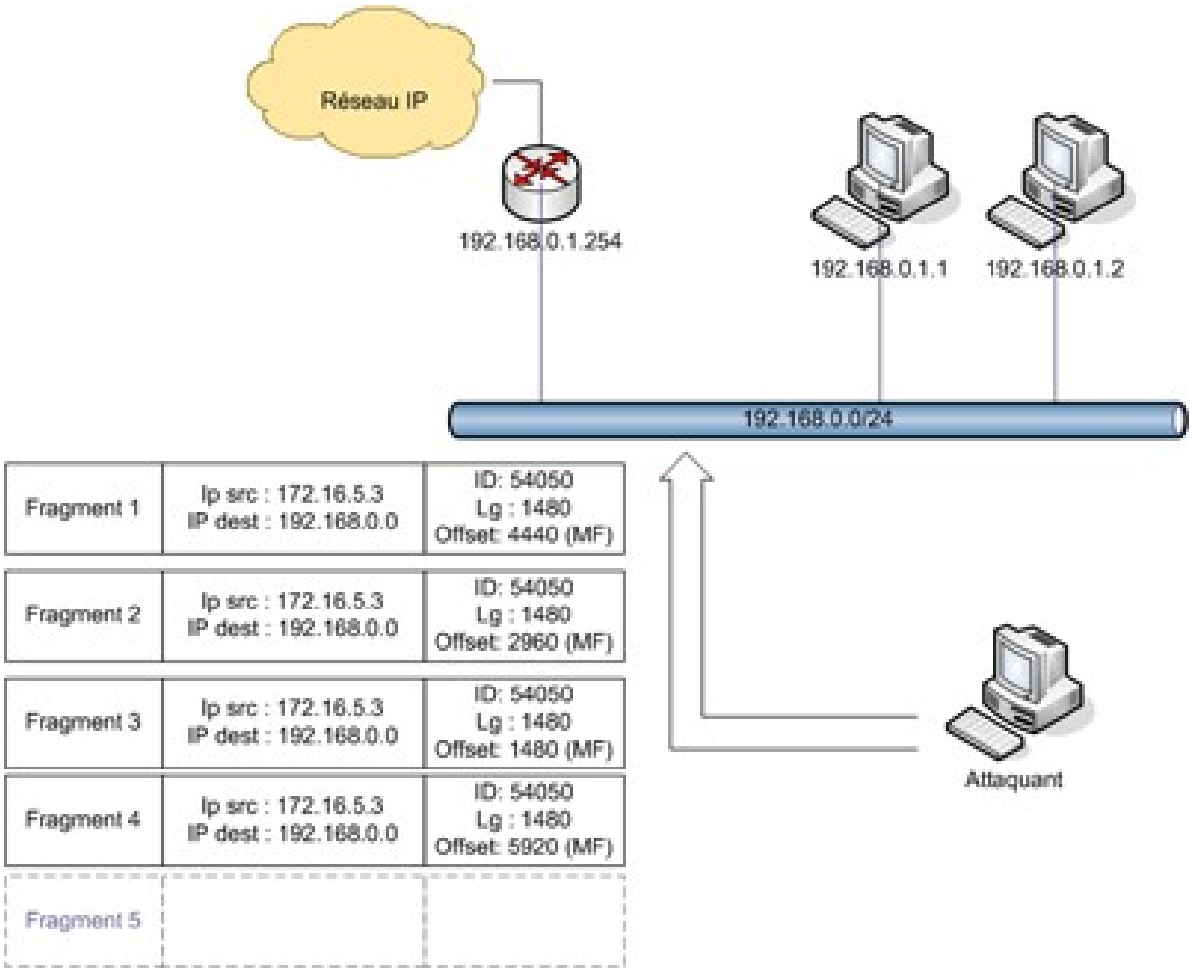
La plupart des protocoles actuels ne possèdent aucun mécanisme de sécurité. Les seuls mécanismes de sécurité sont ceux qui ont été apportés en se servant de champs qui originellement avaient une tout autre signification. De plus, de nombreux protocoles possédaient dès leur création des failles de spécification évidents : IP permet de notifier à un hôte la chaîne de routeur que les paquets qu'il envoie doivent emprunter (Option Source Routing). Fort heureusement, cette fonctionnalité n'est pas implémentée dans les piles TCP/IP modernes. De plus, ICMP (Internet Control Messages Protocol) a, avec ses nombreuses options dont l'option Redirect (type 5), apporté de nombreux problèmes : l'option Redirect permet d'indiquer à un hôte que le chemin que ses paquets emprunte n'est pas optimal et lui demande donc de rajouter dans sa table de routage l'entrée spécifiée dans le paquet. Aussi, le protocole SNMP (Simple Network Management Protocol) souffrait de multiples failles dans sa spécification, ce qui a donné naissance à

des évolutions de la spécification du protocole et de nouvelles versions de celui-ci.

L'ensemble des protocoles qui constituent Internet sont en actuellement en train d'être remaniés pour couper court aux nombreuses attaques qui les visent. Il est important de replacer ces vulnérabilités dans un contexte historique : à l'époque de la création d'ARPANET, et les nombreux protocoles utilisés pour Internet ont été créés à cette époque, la sécurité des données n'avait pas une aussi grande importance, seule la sécurité de l'infrastructure était importante. Le réseau n'était pas prévu pour s'étendre autant et les implications de confidentialité et d'identification précise d'un hôte n'étaient pas une réelle préoccupation mais aujourd'hui deviennent indispensable.

5 Les failles d'implémentation

5.1 Harcélement de routeur



Fragment 1	Ip src : 172.16.5.3 IP dest : 192.168.0.0	ID: 54050 Lg : 1480 Offset: 4440 (MF)
Fragment 2	Ip src : 172.16.5.3 IP dest : 192.168.0.0	ID: 54050 Lg : 1480 Offset: 2960 (MF)
Fragment 3	Ip src : 172.16.5.3 IP dest : 192.168.0.0	ID: 54050 Lg : 1480 Offset: 1480 (MF)
Fragment 4	Ip src : 172.16.5.3 IP dest : 192.168.0.0	ID: 54050 Lg : 1480 Offset: 5920 (MF)
Fragment 5		

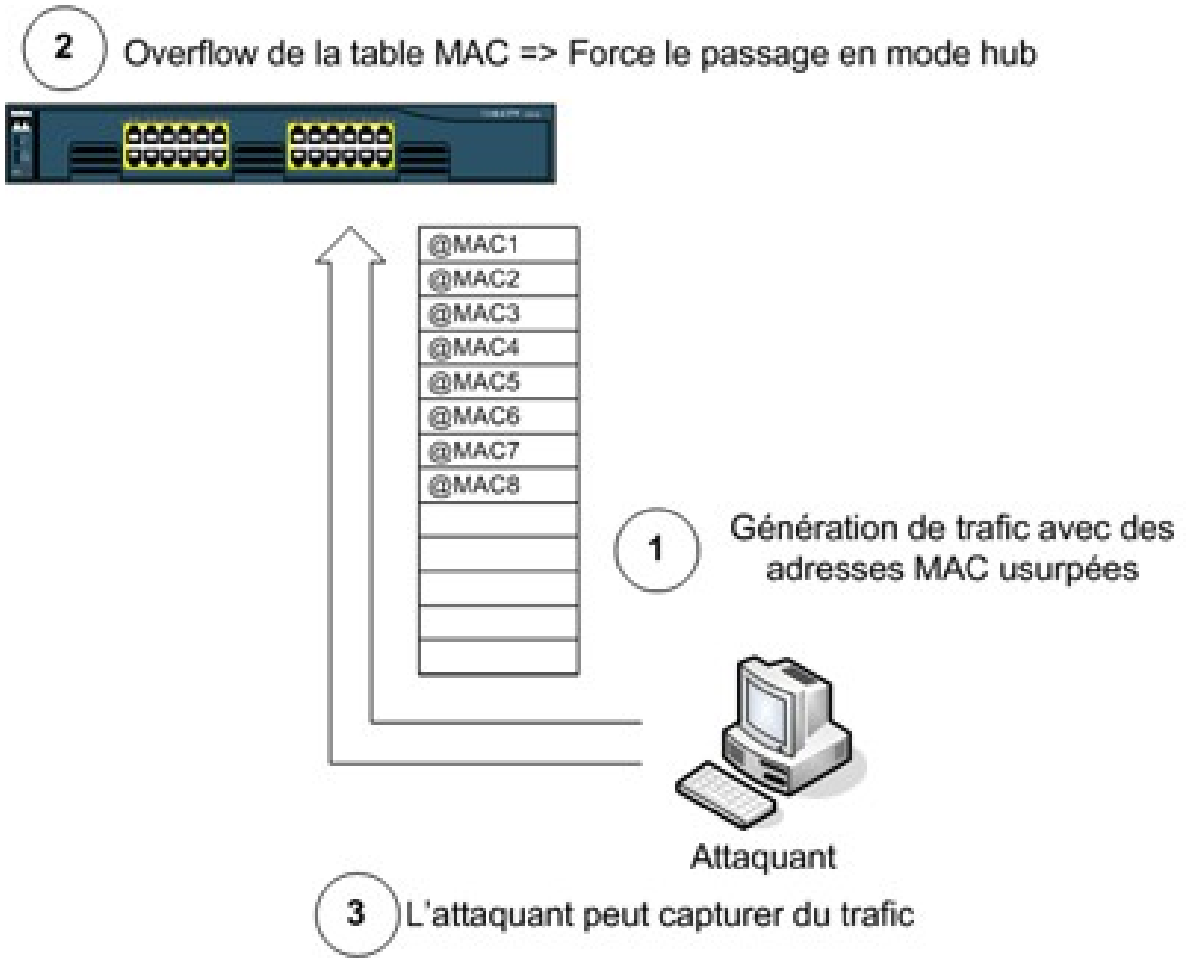
Le harcèlement de routeur est une technique de déni de service.

L'attaquant envoi sur un réseau broadcasté (192.168.0.0) un paquet TCP fragmenté sans jamais envoyer le fragment initial et le fragment final avec le même identifiant TCP et avec des offset cohérents. Chaque hôte va essayer de ré-assembler le paquet TCP dans son intégralité avant de l'envoyer aux couches supérieurs.

L'adresse IP 192.168.0.0 est particulière et seul quelques piles TCP/IP comme celle de routeur ou de BDS prennent le paquets pour eux. Les routeurs vont donc essayer de ré-assembler l'intégralité des fragments. Hors le fragment initial ne sera jamais envoyé et donc les données à ré-assembler vont rester en mémoire.

Si cette technique est utilisée en parallèle avec plusieurs numéros de séquence TCP, alors la mémoire du routeur va être saturée. La tâche principale du routeur va être à ce moment d'essayer de ré-assembler les paquets au détriment de sa fonction principale : Router des paquets. Le réseau serafortement ralenti.

5.2 Overflow de Table MAC



Cette technique permet de faire passer certains switchs en mode hub par une saturation de leur table MAC. En effet certains switchs pour se grader leur fonction principale de commutation, passe en mode hub en cas de surcharge. Ce passage peut être détourner afin de récupérer du trafic illégitime.

Le principe est très simple : l'attaquant génère de nombreux paquets en usurpant son adresse MAC et adresse IP. Ainsi le switch va accumuler dans sa table MAC de nombreuses entrées et au bout d'un moment va saturer. A ce moment soit le swicth se bloque, soit il passe en mode hub.

Cependant de nombreux switchs administrables ont une protection qui permet de

n'autoriser qu'une seule adresse MAC par port.

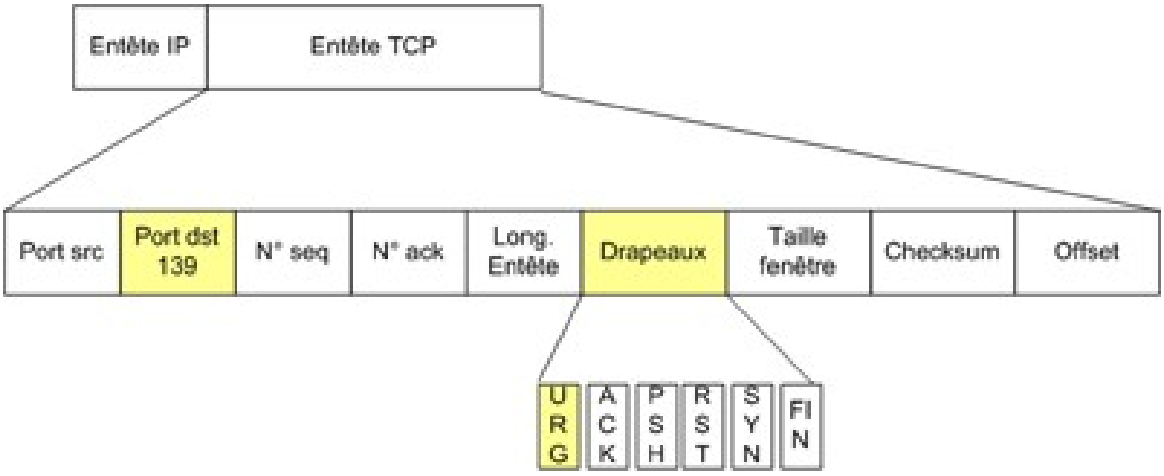
5.3 Windows Out Of Band Netbios

C'est une faille d'implémentation résolue depuis longtemps sur les Windows 95 et 98 mais son étude montre comment une faille peut se révéler un cauchemar pour un administrateur.

Le protocole Netbios utilisé pour le partage de fichiers Windows utilise principalement le port TCP 139.

Un défaut d'implémentation permettait très simplement de faire planter (par un écran bleu) les machines sous Microsoft Windows 9x. L'attaquant envoyé un simple paquet TCP sur le port 139 avec le flag TCP "URG". La pile TCP/IP remonte immédiatement le paquet à la couche applicative qui ne sait pas traité ce cas et fait planter l'ordinateur.

Cette technique montre qu'une erreur d'implémentation peut s'avérer fatale. Au vu de l'étendue du parc informatique sous Windows, cette faille d'implémentation a fait un malheur avant que le correctif soit diffusé en masse.

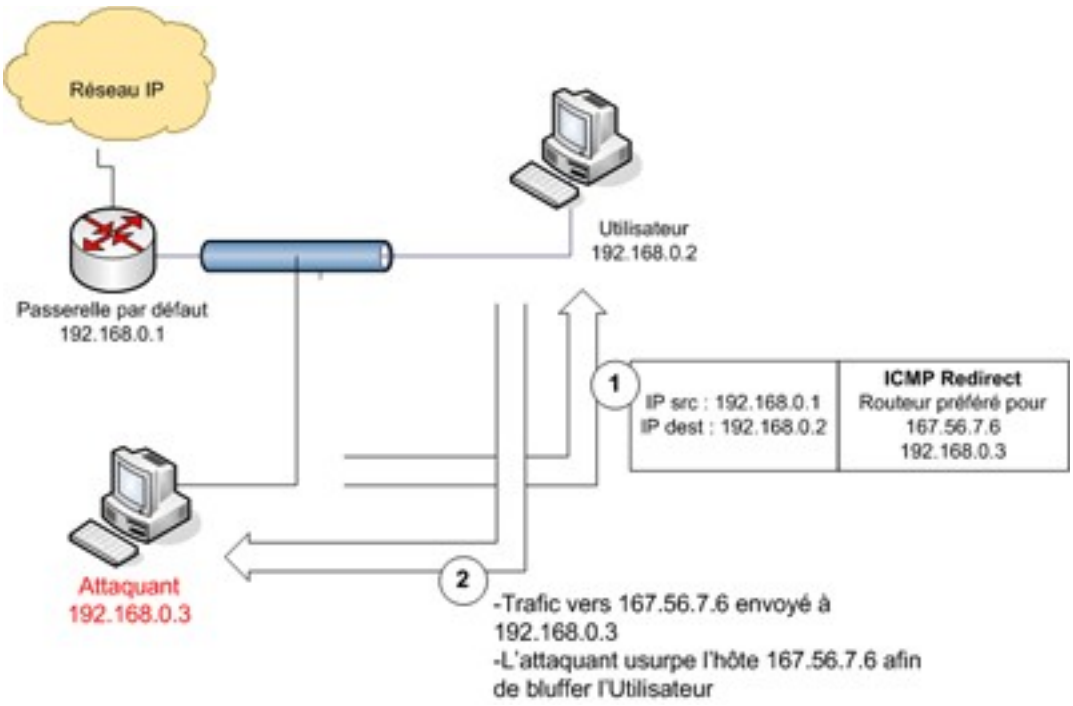


5.4 ICMP Redirect

Ce n'est pas réellement une faille d'implémentation car elle utilise un comportement classique d'une pile IP. Cette technique détourne l'utilisation classique des paquets ICMP Redirect afin de rediriger du trafic illégitime vers l'attaquant.

En écoutant le trafic sur le réseau, l'attaquant souhaite rediriger un flux IP vers soi à des fins d'usurpation d'un serveur distant. Ainsi il envoie un paquet ICMP Redirect en spécifiant que la route préférée pour accéder à l'hôte usurpée est 192.168.0.3. Une entrée dans la table de routage de la cible(192.168.0.2) détournera le flux IP en destination de 167.56.7.6 vers 192.168.0.3 (routeur préféré pour cette route).

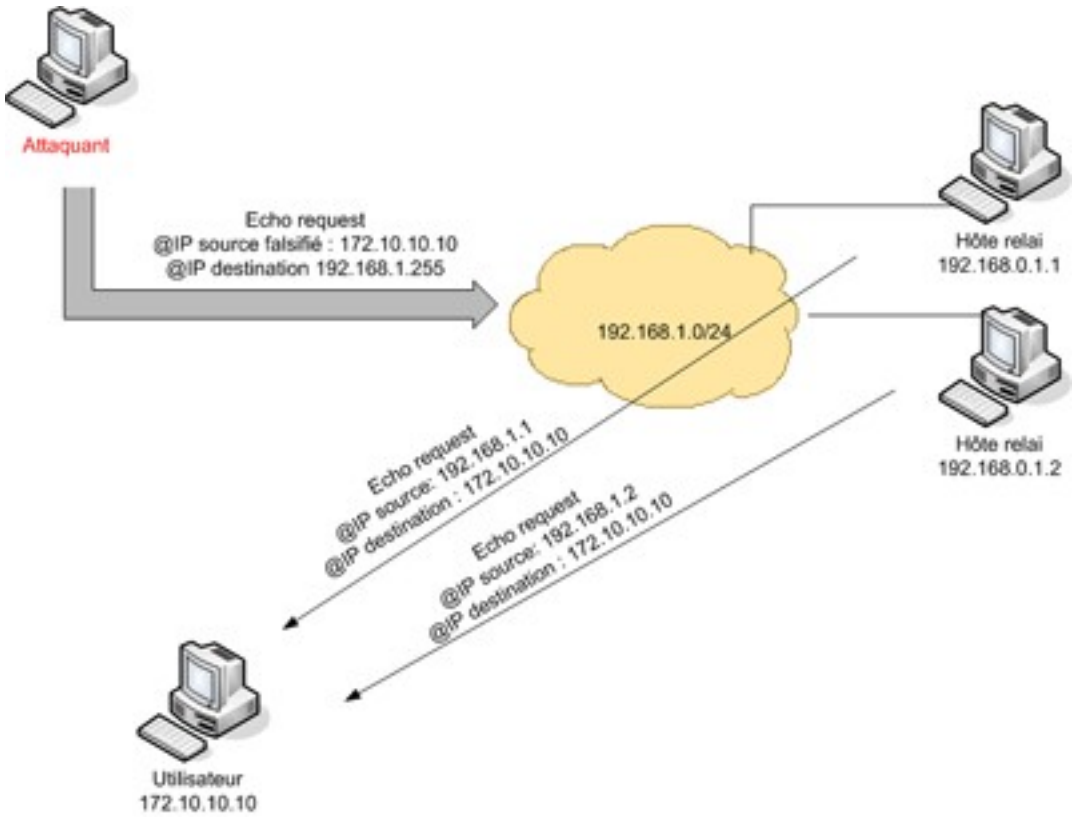
En simulant les services fournies par l'hôte usurpée (167.56.7.6), l'attaquant peut récupérer par exemple les logins et mots de passe d'un service web fourni par le serveur usurpé.



5.5 Smurf

Smurf est une technique ancienne qui fonctionne très souvent. C'est une technique de déni de service utilisant les réseaux "broadcast".

Le but est d'inonder de requêtes une cible afin de saturer sa bande passante.



Pour cela l'attaquant va envoyer avec une adresse IP usurpée (par celle de sa cible) un flux ICMP ou TCP à un réseau relais qui va répondra à la cible. Si le nombre de relais est important, l'attaque sera des plus importantes et la cible sera paralysé tout le long de l'attaque. Il est théoriquement impossible de se protéger d'une telle attaque si elle provient de nombreuses hôtes car elle doit être bloquer par le fournisseur d'accès (s'il n'es pas lui même saturé). Déni de Service distribué

Avec l'avènement de l'informatique et d'Internet, le nombre d'ordinateurs connectés devient de plus en plus impressionnant. Les ordinateurs mal protégés (de plus en plus nombreux) sont souvent utilisés afin d'y installer des virus ou chevaux de Troy permettant de prendre le contrôle à distance des ordinateurs.

Les attaquants en synchronisant cette population d'ordinateurs sous leur contrôle ont accès à une bande passante gigantesque (surtout avec l'avènement du haut débit !). En

combinant ces flux venant de partout à des techniques de dénis de services conventionnels, le flux engendré devient bloquant même pour un fournisseur d'accès.

Vu que le flux provient de milliers d'adresses IP différentes, il est inimaginable de filtre ce flux en amont au détriment des performances des routeurs des backbones. Le déni de service est dans tous les cas réussi !

6 Conclusion

La sécurité a pendant très longtemps été négligée par les utilisateurs, qu'ils soient professionnels ou particuliers, les développeurs ainsi que les auteurs des spécifications des protocoles. Depuis une dizaine d'années, une prise de conscience des risques et des implications de la sécurité commence à voir le jour, avec le nombre grandissant de systèmes d'information vitaux pour le fonctionnement des entreprises. Cependant, aujourd'hui encore de nombreux écueils sont hérités de spécifications désuettes et d'implémentations aléatoires, suivant l'interprétation et les contraintes techniques ou temporelles accordé à la réalisation de chacune.

Aujourd'hui la sécurité à un coût. Elle a un coût d'image – une entreprise piratée subit un déficit d'image auprès de ses clients – mais aussi un coût lié aux données, les systèmes d'information étant les garants de l'intégrité ainsi que de la non divulgation de certaines informations confidentielles stockées mais nécessitant d'être accédées.

Ainsi, étant donné le nombre de systèmes hétérogènes, de protocoles variés et d'implémentations différentes, il est nécessaire, en entreprise, non pas de voir la sécurité comme un risque pouvant être maîtrisé, mais comme un risque permanent. Aussi, en ayant conscience de ce risque, il ne s'agit plus de vouloir à tout prix s'en protéger mais d'effectuer le travail d'un DSSI : de la gestion de risque. La véritable question de la sécurité devient alors : « Cela me coûterait tant de sécuriser autant que cela est possible tel système et les données stockées ou transitant dessus en valent tant, quel risque suis-je disposé à prendre ? »

7 Références

[1] *Paul Baran and the Origins of the Internet* - http://www.rand.org/about/history/baran.html
[2] *Root-Zone Whois Information* - http://www.iana.org/cctld/cctld-whois.htm
[3] *RFC826, An Ethernet Address Resolution Protocol* - http://www.ietf.org/rfc/rfc826.txt
[4] *RFC793, Transmission Control Protocol* - http://www.ietf.org/rfc/rfc793.txt
[5] *RFC1750, Randomness Recommandations for Security* - http://www.ietf.org/rfc/rfc1750.txt
[6] *FreeBSD TCP/IP Spoofing* - http://security-protocols.com/textfiles/advisories/hert/hert.0003.freebsd.isn
[7] *HERT, Hacker Emergency Response Team* - http://www.hert.org/
[8] *Strange Attractors and TCP/IP Sequence Number Analysis* - http://lcamtuf.coredump.cx/oldtcp/tcpseq.html && http://lcamtuf.coredump.cx/newtcp/
[9] *Domain Name Servers, RFC1033,1034,1035,1037* - http://www.ietf.org/rfc/rfc[1033,1034,1035,1037].txt
[10] *DNS ID Hacking (and even more !!) with colors & images ;)* - http://adm.freelsd.net/ADM/ADMID.txt
[11] *DNS Cache Poisoning* - http://www.lurhq.com/cachepoisoning.html