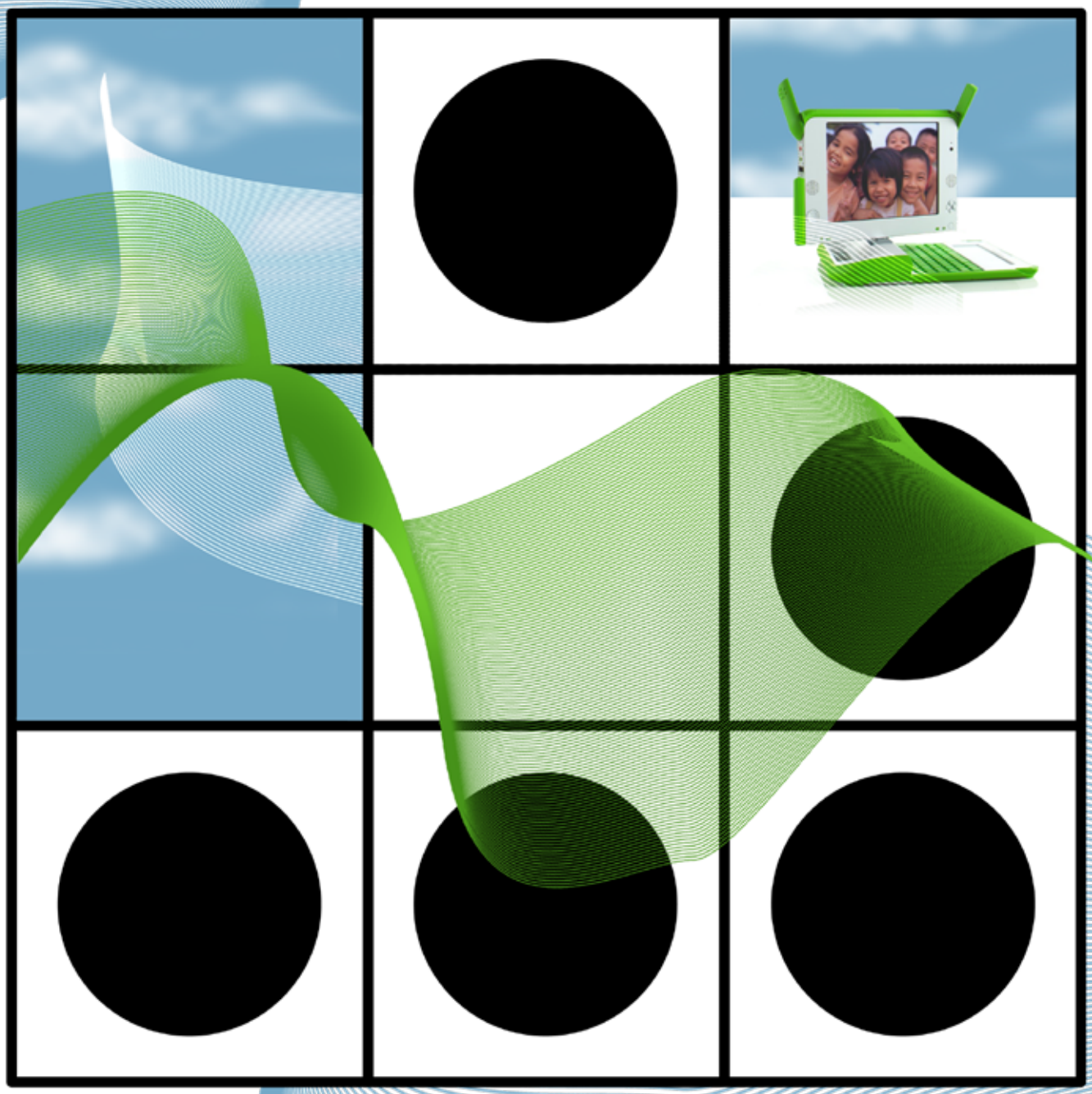


#3

hackhispano

electronic fanzine

Ezine #3

La tercera entrega de esta eZine se ha hecho esperar.quizás mas de la cuenta, pero la espera habrá merecido la pena.
Este ejemplar se plantea mas breve y con menos artículos, pero no por ello de menos calidad.
Todo el equipo que desarrolla esta ezine, se ha empleado a fondo para llevarla a cabo, y a los cuales quiero dedicar mi primer agradecimiento. También a todos nuestros lectores, que cada dia haceis que esto siga adelante.
En esta ezine se tratan temas de programación, seguridad, ética hacker, y novedades tecnológicas, las cuales han acontecido en este periodo.
Esperamos que disfrutéis la lectura, y os animéis a participar con nosotros.

HackHispano es una comunidad libre donde todo el mundo es bienvenido, donde nadie es extranjero, donde todos buscamos algo ydonde todos loofrecemos.
Nuestra comunidad no es más que un punto de encuentro para todos los que estáis perdidos en este cada vez más confuso mundo de la sobreinformación, donde encontrareis gente como vosotros que intentará ayudaros y donde seguro encontrareis alguienque precisa de vuestra ayuda.

Sed bienvenidos a HackHispano.

Indice:

OLPC
ONE LAPTOP PER CHILD

Desde el lanzamiento de este proyecto, he estado muy interesado por su idea revolucionaria y sus intenciones. En este artículo me propongo comentar todo lo relacionado con este proyecto, hablar de las XO, de Sugar, informacion, técnica y contratiempos del mismo, ademas de la competencia que ha recivido por parte de otras empresas.

¿ QUÉ ES OLPC ?

Una Laptop por Niño ó OLPC por sus siglas en ingles (One Laptop Per Child), es una asociación sin fines de lucro creada para desarrollar una laptop de 100 dólares.

Esta idea fue de Nicholas Negroponte, cientifico estadounidense del MIT, que con sus laptops de 100 dólares pretende disminuir la brecha tecnológica entre niños del primer mundo y los paises menos desarrollados.

Un poco de Historia...

La idea fue presentada en 2005 en Tunez, y luego en enero del 2006, en el Foro Económico Mundial de Davos, bajo OLPC, despues de esto, los lideres del mundo y corporaciones se han convertido en benefactores de este proyecto sin fines de lucro.

Algunos de los benefactores del proyecto son: Google, AMD, Red Hat, News Corp, Brighstar más otras empresas.

Pero cerca de tres años más tarde, solo 2000 niños en planes pilotos han recibido computadoras de OLPC. Hasta ahora mi país Uruguay, ha sido el pionero y uno de los pocos en hacer una orden de 100.000 laptops, las XO.

La meta de Negroponte es alcanzar 150 millones de usuarios para el termino del 2008. Un poco utópico desde mi punto de vista.

Desgraciadamente el ambicioso plan de Negroponte ha sido retrasado por el peso de su idea. Companias lucrativas no quicieron quedarse fuera de esta idea, e introducieron sus propias versiones o adaptaciones para este proyecto, y generar competencia. Posteriormente Hablaremos sobre ello y haremos algunas comparaciones.

Aun así, Negroponte se ve optimista, ya que sostiene que su meta no es vender laptops, " OLPC no es un negocio de venta de laptops, es un plan educativo "

Es valido aclarar, que desde su concepcion, el precio de las laptops ha dejado de ser de 100 dólares, ya que sus costos han aumentado, (costos de embalaje, etc) actualmente su precio es casi el doble, 199.5 dólares es su precio actual, lo cual ha echo declinar a algunos países.

LA COMPTENECIA:

Esto se puede resumir en dos nombres: Intel y Microsoft.

Los motivos por los cuales estas dos empresas casi hermanas, quieren introducirse en el mundo de las laptops de bajo costo, será explicado, con mi punto de vista más adelante, ahora citaré unas cuantas noticias, de portales de internet, haciendo referencia a OLPC y sus competidores:

"Classmate", la alternativa Intel al portátil de 100 dólares

Intel no está dispuesta a perder el suculento mercado que representan los países en desarrollo. Mientras AMD apuesta fuertemente por la iniciativa de Nicholas Negroponte (OLPC), Intel viene desarrollando su propia alternativa que destaca por su mayor potencia y prestaciones pero también más cara.

http://www2.noticiasdot.com/publicaciones/2006/1106/2111/noticias211106/noticias211106-371.htm

Intel intenta competir con OLPC en Argentina

Ya se vende en la Argentina una laptop para estudiantes. Se trata del modelo Classmate PC, fabricado por Intel, que de esta manera sacó ventaja frente al equipo que propicia Negroponte. Esteban Galuzzi, gerente General de Intel Cono Sur, explicó los detalles del equipo.»

http://hackingrioplatense.com.ar/foro/index.php?topic=410.0

Intel y Microsoft, contra el proyecto OLPC

El proyecto One Laptop Per Child está despertando el interés de los grandes de la informática mundial. Tanto Intel como Microsoft parecen haber firmado acuerdos de colaboración con la organización que lidera Nicholas Negroponte, pero ambas empresas juegan a dos bandas. Microsoft está vendiendo Windows y Office a 3 dólares en China --y puede que pronto en otros países-- e Intel no quiere dejar escapar su oportunidad con su portátil ClassMate, un competidor directo del XO de la OLPC

http://hackingrioplatense.com.ar/foro/index.php?topic=422.0

Comparacion, entre las laptops de bajo costo, actualmente en el mercado

Fabricante	Asus	Everex	Intel	OLPC
Modelo	Eee PC 701	Cloudbook CE1200V	ClassMate	XO-1
				
Web	http://es.asus.com	http://www.everex.com	http://www.intel.com/intel/worldahead/classmatepc/	http://www.laptop.org
Procesador	Intel Celeron-M ULV 353, 900 MHz	Via C7 ULV, 1,2 GHz	Intel Celeron M, 900 MHz	AMD Geode LX-700, 433 MHz
Memoria	512 / 1024 Mbytes DDR2	512 Mbytes	256 Mbytes DDR2	256 Mbytes DDR
Almacenamiento	SSD 8 Gbytes	Disco duro convencional 30 Gbytes	SSD 2 Gbytes	SSD 1 Gbyte
Pantalla	7"	7"	7"	7.5", con un modo especial para usarlo a la luz del día
Resolución	800x480	800x480	800x480	800x480
Webcam integrada	0,3 Mpixeles	1,2 Mpixeles	No	Sí, 640x480, 30 fps
WiFi	802.11b/g	802.11b/g	802.11b/g	802.11b/g
Fast Ethernet 10/100	Sí	Sí	Sí	No
Batería	5200 mAh	n.d.	5 células Li-Ión, no se especifica capacidad	5 células NiMH / 2-4 células LiFePO4
Lector de tarjetas de almacenamiento	MMC/SD	4 en 1	No	SD
Puertos	3 x USB 2.0, auriculares, VGA	2 x USB 2.0, DVI	2 x USB 2.0	3 x USB 2.0
Dimensiones (mm)	225x 165 x 35	n.d.	245 x 196 x 44	242 × 228 × 32
Peso (g)	890	n.d.	< 1450	1450
Sistema operativo	Xandros	gOS	Mandriva / Windows XP	Fedora (acaptado)
Disponibilidad	Ya disponible	15 de enero de 2008	Ya disponible	Ya disponible
Precio	399 dólares	400 dólares	No se vende al público	399 dólares a través del programa "Give One, Get One"

imagen original en: http://www.javipas.com/wp-content/tablaUMPCs.jpg

Haciendo una comparacion entre la Classmate y la XO, la XO pierde en procesador y RAM, aun así, veo más viable la opción, de la XO, por sus caracteristicas técnicas y su enfoque hacia los niños.

Y personalmente tengo mejor visto a AMD y a un SO GNU/Linux en una maquina que a Intel y Windows de Microsoft, ya que estas dos últimas, buscan la domesticación del usuario, y vamos.. que es por eso que han entrado a competir, ¡ Intel No fabrica Laptops ! y Microsoft, no quiere a un niño criado con un sistema GNU/Linux (Sugar, Red Hat) sino su Windows Adaptado.

Con respecto a la Asus EEE, otra opinion personal, seria interesante, verla implementada en un plan de educación secundaria, ya que es la más potente de las 3, y sigue estando dentro de la gama barata.

Caracteristicas Técnicas

Comencemos por el hardward que contiene la XO, y luego hablaremos del SO y caracteristicas de su seguridad.

HARDWARE:

Las XO tienen una pantalla simil LCD, la cual consume poca electricidad, Color y se le puede elegir para blanco y negro, con dimensiones de 7.5"

El diseño de la pantalla permite que se pueda leer en ella mientras el sol le da de frente.

- Pesa 1.5 Kg.
- Tiene un procesador AMD Geode LX-700@0.8W de 433 MHz
- Memoria DRAM: 256 MiB
- No usa disco duro, usa memoria flash de 1Gb.

Tiene un chip de red inalambrico con mayor rango de recepcion que los estandar, y utiliza un tipo de red mesh http://es.wikipedia.org/wiki/Red_inalámbrica_Mesh

Con respecto a la electicidad, tiene una bateria con estas caracterisitcas :

- Empaque sellado "duro/rígido"; removible por el usuario
- Tipo de empaque: Configuración de 4 celdas, 6V en serie
- Dos alternativas
- NiMH, con capacidad de 16.5 watt-hora
- LiFeP, con capacidad de 22 watt-hora

La forma de cargar la bateria puede ser usando una manija, ideal para lugares donde no se puede acceder a la electricidad.

Por muchos más detalles: http://wiki.laptop.org/go/Hardware_specification

SOFTWARE:

En tanto el tema de Software, las XO, usan un sistema operativo basado en Fedora, la rama gratuita de Red Hat, el sistema se llama Sugar y utliliza el núcleo Linux 2.6.22.

He podido probar el sistema, y este está completamente diseñado para los niños, su interface gráfica consta de pocos botones, lo cual lo hace todo muy sencillo e instintivo.

Algunos de sus contenidos para explorar:

- Un navegador web basado en Firefox.
- Un visor de documentos basado en Evince.
- Un lector de RSS Feeds.
- Aunque no viene incluido en la distribucion estandar, se puede instalar Opera, y utilizarlo.

Herramientas para la Expresión

- TamTam, un sintetizador y compositor de música.
- Un editor de texto basado en Abiwords
- Etoys, una herramienta Similar a LOGO, más información: http://laptop.org/OLPCEtoys.pdf
- Record, una herramienta que permite grabar sonido y video
- Un Diario

- VIM y Nano, como editores de texto

Herramientas para la Comunicación:

- Chat
- Video Chat (en desarrollo)
- Cliente para VoIP (en desarrollo)
- E-mail a a través de Gmail
- Un Cliente de correo nativo (en desarrollo)

Otras Herramientas:

- Una calculadora
- Un shell y un debugger
- Escritorio Remoto, para cotrolar la maquina a distancia

Juegos:

- Muchos juegos, de memoria y estrategia

Por más informacion y detalles de las aplicaciones y bibliotecas: http://wiki.laptop.org/go/Software_components

Su sistema de Seguridad:

Bitfost, es el sistema que utilizan las XO, tiene algunas caracteristicas interesantes, y llaman la atención algunas de ellas.

Caracteristicas:

No se requiere ninguna contraseñas para acceder a la computadora. Ni siquiera el root tiene contraseña. Lo cual me ha llamado mucho la atencion, pero tras una preguntas y averiguaciones, me dijeron que el root no lleva passwd ya que en el momento del desarrollo, seria una complicacion más que una ventaja. Despues cuando sean entregadas la institucion pertinente (Ministerio de educación, escuela, aula, maestro) puede implementar una.

Cualquier programa, al momento de ser instalado, requiere ciertos conjuntos de derechos, por ejemplo "acceder a la cámara", o "acceder a internet". El sistema realiza un seguimiento de estos derechos, y el programa es luego ejecutado en un ambiente en el cual sólo los recursos requeridos están disponibles. Esto está implementado mediante una máquina virtual contenedora completamente desarrollada.

Por defecto, el sistema no permite ciertas combinaciones de derechos; por ejemplo, un programa no tendría permisos para acceder a la cámara y acceder a internet a la vez. Cualquiera puede escribir y distribuir programas que requieran combinaciones de derechos permitidas. Los programas que requieran combinaciones de derechos no aprobadas necesitan una firma criptográfica de alguna autoridad. El usuario de la laptop puede utilizar el panel de seguridad incluído para otorgar derechos adicionales a cualquier aplicación.

Contratos anti-robo

Las laptops solicitan un nuevo "contrato" a un servidor de la red central una vez al día. Estos contratos llegan con una fecha de vencimiento (normalmente un mes), y la laptop deja de funcionar si todos sus contratos han expirado. Los contratos también pueden ser otorgados por un servidor en la escuela, o mediante un dispositivo USB. Aquellas laptops que hayan sido reportadas como robadas no pueden obtener un nuevo contrato.

El país que adquiere la laptop puede determinar si se utiliza este sistema de contratos, y seleccionar el tiempo de vencimiento de los mismos.

Microfono:

El micrófono y la cámara incluídos en la laptop se encuentran conectados directamente a un LED, así el usuario puede saber en todo momento si están encendidos. Esto no puede ser desabilitado por software.

Información extraida de la wikipedia

EMULANDO SUGAR:

Espero que toda esta introducción te haya entusiasmado como para probar el sistema, y por qué no, aportar al testeo y desarrollo de Sugar.

Pondre dos manuales, uno echo por mi, para GNU/Linux con Virtual Box, y otro para Windows, echo por un compañero de la facultad, Yasim Zeballos.

Las licencias de los mismos son Creative Commons.

La mia:

La de Yasim es:

Ante cualquier modificacion, contactarse con los autores respectivos, con Yasim en:

yasim (AT NO SPAM) adinet (dot) com (dot) uy

Y conmigo a a través de Hack Hispano o Hacking Rioplatense.

webmaster (AT NO SPAM) hackingrioplatense (dot) com (dot) ar

MANUAL PARA GNU/LINUX:

Lo primero que tenemos que hacer, es bajarnos las imagenes del sistema, son imagenes del disco duro. Las imagenes nos las podemos bajar de aqui:

http://dev.laptop.org/pub/virtualbox/

Yo elegi la última vercion, la 625 (En el momento que escribi el manual, era esa versión, ahora es la 666) Para bajarlo, NO vamos a usar al navegador, ya que si se llega a interrumpir la descarga, no hay vuelta atras. Para descargar la imagen, nos vamos a un terminal y escribimos:

```
$ wget http://dev.laptop.org/pub/virtualbox/
OLPC-625.zip
```

OLPC-625, es la última versión actualmente. En el caso de que tengamos que interrumpir la descarga, podemos volver agregando wget -c Más información:

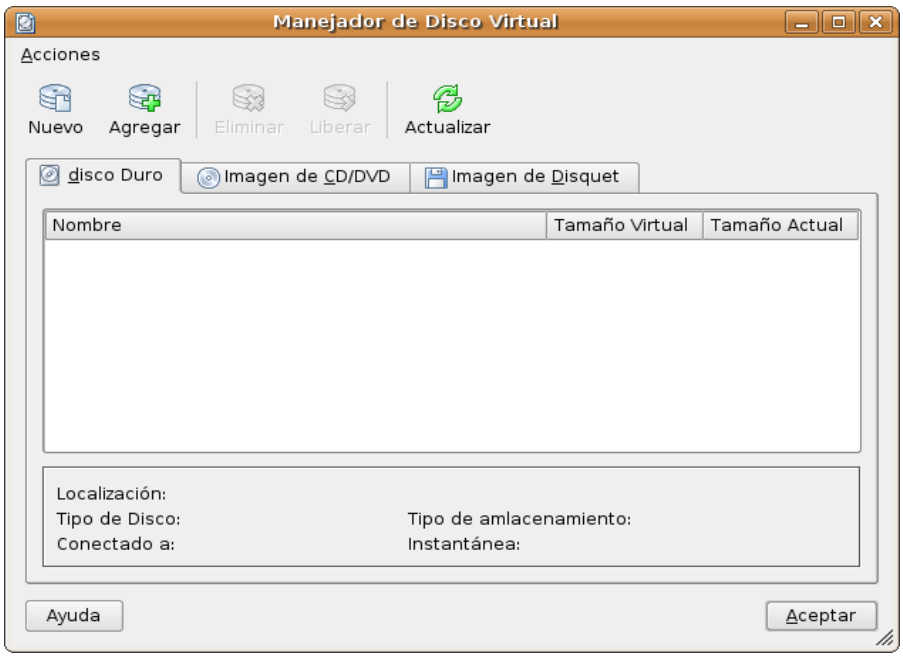
```
man wget
```

Una vez descargado el zip, extraemos el contenido del mismo.

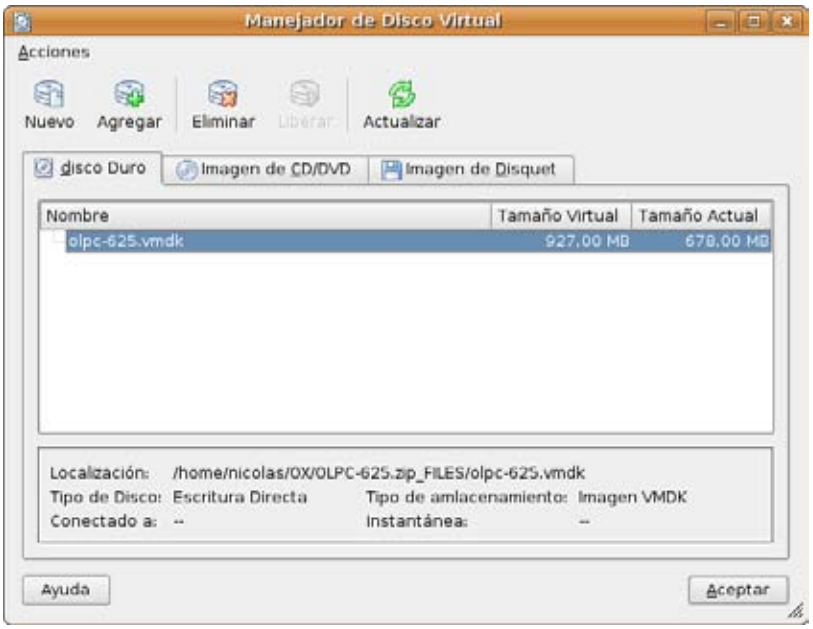
```
$ unzip OLPC-625.zip/OLPC-625.zip_FILES
$ ls olpc-625.vmdk OLPC-625.vmx
```

Ahora iniciamos virtualbox:

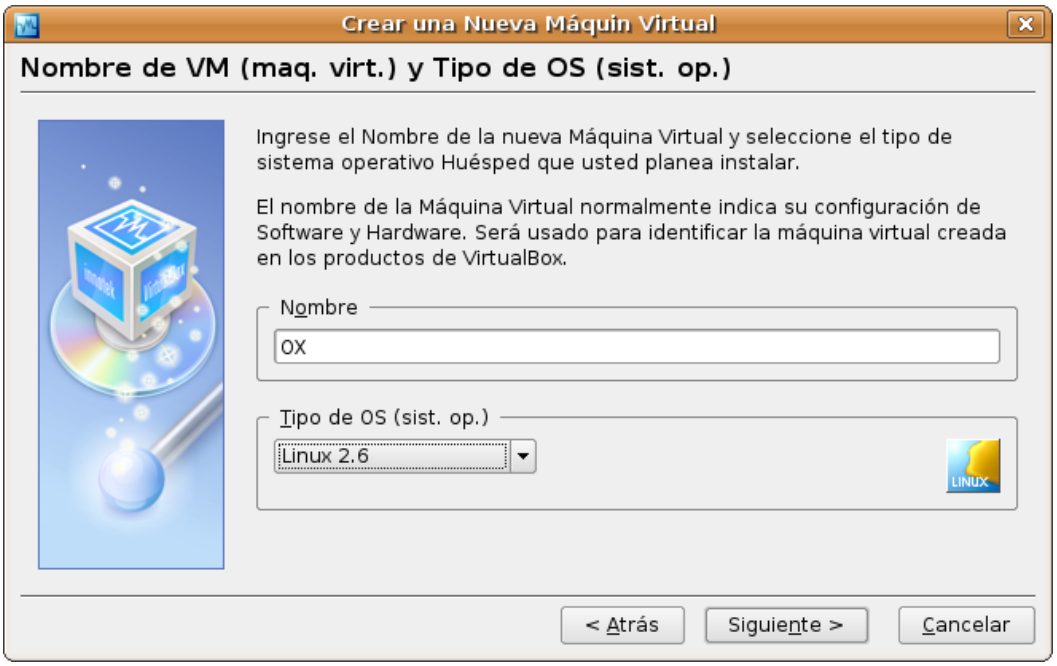
Ahora lo que tenemos que hacer, es agregar el disco duro vitual :) Archivo > Administrador de discos Duros



Clickeamos en agregar, y nos vamos a donde descomprimimos a la carpeta creada tras descomprimir el zip



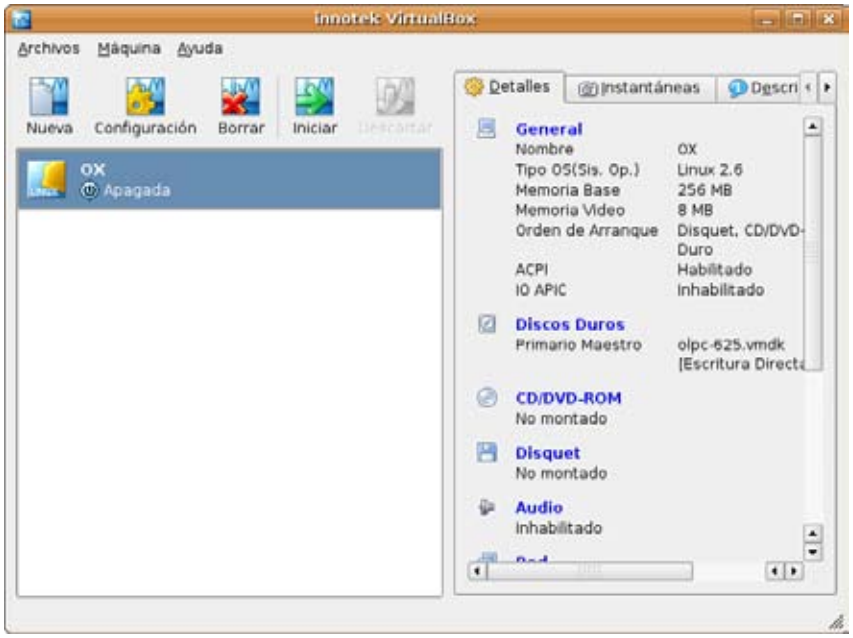
Selecionamos el archivo y Aceptar, Aceptar - facil, verdad ? Listo, ahora vamos a crear nuestro sistema ! Ponemos Nueva > Siguiente Ahora elegimos el nombre de nuestra maquina, y selecionamos el núcleo del sistema, Recordar que este sistema operativo es un sistema GNU/Linux basado en Fedora. Usaremos el kernel 2.6



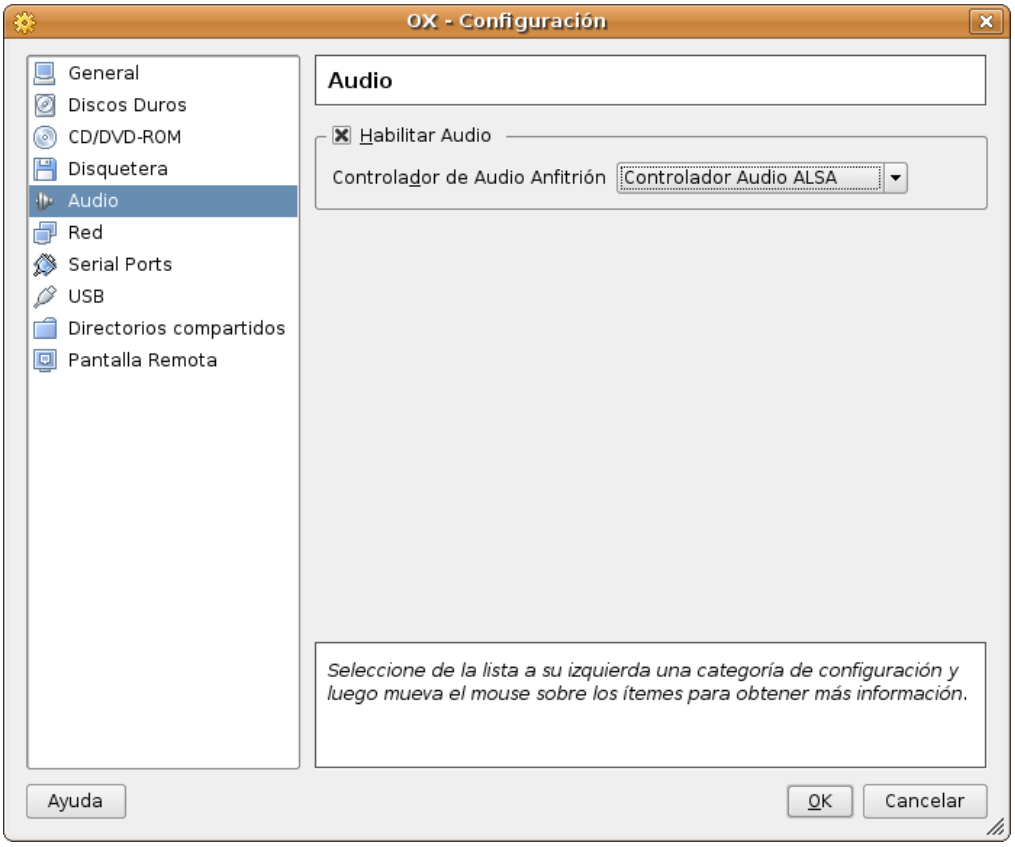
Y Siguiente.. Ahora elegiremos la memoria RAM que virtualizara, ponganle 512 Mb, aunque yo la anduve corriendo con 256 también :P

Siguiente.. Ahora selecionaremos la imagen del disco duro, se acuerdan ? lo que hicimos al prinicipio :D Selecionamos, Existente > Y clickeamos en Selecionar

Ahora Siguiente y ya casi Terminamos !!



Ahora tenemos que configurar unas pocas cosas ;) Nos vamos a configuración Dejamos todo como está en general y nos vamos a Sonido:



Seleccionamos el controlador de audio ALSA Y Listo, pueden configurar ustedes un poco más, todo el asunto de pen drives CDs y red, pero con esto ya es suficiente. Ahora solo queda clickear en iniciar y listo ! ;) Ya lo pueden probar.

Ahora el manual para Windows, escrito por Yasim :

FUNCIONAMIENTO DE LA OLPC EN WINDOWS XP.

Referencia rápida :

1) Bajarse el emulador de 'máquina'
http://www.h6.dion.ne.jp/~kazuw/qemu-win/qemu-0.9.0-windows.zip (Está en zip)

2) Descomprimirlo en c:\program files\Qemu (NO EN OTRO LADO, AHÍ SI O SI)

3) De las miles de imágenes que hay en http://olpc.download.redhat.com/olpc/streams/development/ Bajarse por ejemplo :

http://olpc.download.redhat.com/olpc/streams/development/LATEST-STABLE-BUILD/devel_ext3/olpc-redhat-stream-development-devel_ext3.img.bz2

(Fue elegida ARBITRARIAMENTE pero con criterio :D).

4) Infelizmente, no es un '.zip' sino un '.bz2' así que nos tenemos que bajar el programa que descomprima eso:
Vamos a http://www.bzip.org/downloads.html y descargamos el descomprensor. O en su defecto, buscamos algun descomprensor que aparte de ".zip" descomprima '.bz2'.

5) La infelicidad prosigue, no tiene entorno gráfico, por eso, cuando le hacemos dobleclick se nos cierra, esto no es que funciona incorrectamente, sino que tenemos que agregar varios pasos más.

5.1) Vamos a Inicio -> Ejecutar, y escribimos cmd.exe
5.2) Se nos tiene que abrir una ventana de 'DOS'

5.3) Vamos al directorio donde está el descomprensor que bajamos, (en el que también debería de estar la imagen) y escribimos :

```
bzip2.exe -k -d -v nombre_muy_largo_y_aburrido_de_escribir.img.bz2
```

osea

```
bzip2.exe -k -d -v olpc-redhat-stream-development-devel_ext3.img.bz2
```

6) Finalmente ponemos la imagen que acabamos de descomprimir (que pesará 1GB más o menos) en la carpeta que

contiene a qemu.exe, y escribimos

```
qemu.exe olpc-redhat-stream-development-devel_ext3.img
```

Debería de iniciarse...
7) Ahí seleccionamos OLPC for qemu target (Scaled)

Problemas que pueden surgir

Si dice que no encuentra el bios cuando tratamos de ejecutar "qemu.exe olpc-redhat...", debemos localizar el archivo "bios.bin", suponiendo que se encuentre en : D:\qemu-0.9.0-windows\bios.bin, para ejecutar todo debemos :

```
qemu.exe -L d:\qemu-0.9.0-windows olpc-redhat-stream-development-devel_ext3.img
```

Sugerencia Alternativa

1) Poner el qemu-0.9.0-windows en D:\qemu-0.9.0-windows
2) Poner la imagen de 1 GB en D:\qemu-0.9.0-windows\olpc-redhat-stream-development-devel_ext3.img
3) Hacer cmd.exe en Inicio -> Ejecutar.
4) En C:\Documents and Settings\fulano> poner

```
D:
```

5) En D:\ poner

```
cd qemu-0.9.0-windows
```

6) Ya en D:\qemu-0.9.0-windows> poner

```
qemu.exe -L d:\qemu-0.9.0-windows olpc-redhat-stream-development-devel_ext3.img
```

(Todo esto para aquellos que tengan "2 discos o más discos", si se tiene un disco solo hay que hacer todo trabajando en el C:\

Mejoras:

Si vemos que nos va muy lento, podemos setear más memoria ram al qemu, esto se hace con la opcion -m [RAM para agregarle]

Nuestro comando quedaría :

```
qemu.exe -L d:\qemu-0.9.0-windows -m 256 olpc-redhat-stream-development-devel_ext3.img
```

(256 hay que ponerlo si se tiene MAS que 256 MB RAM, osea, podemos poner el número que deseemos).

Podemos instalarle un 'acelerador' tal como lo indica:
http://wiki.laptop.org/go/Using_QEMU_on_Windows_XP#Troubleshooting
(para ir directamente a bajarse el acelerador :
http://fabrice.bellard.free.fr/qemu/kqemu-1.3.0pre11.tar.gz)

Los pasos están bastante bien, así que no los voy a explicar.

HOTKEYS :
Una vez 'atrapados dentro de la emulación' podemos salir con alt+tab
ctrl+alt+F1 nos lleva a la consola de comandos
ctrl+alt+F3 nos devuelve a el entorno gráfico.

Por último quisiera mostrarles los blogs de las escuelas, donde el proyecto se probo como plan piloto:

- Blog experimental de la escuela
  http://cardal-ceibal.blogspot.com/
- Blog de Cuarto año
  http://cuarto-cardal.blogspot.com/
- Blog de Quinto año
  http://cardal24-quinto.blogspot.com/
- Blog de Sexto año
  http://cardal24-sexto.blogspot.com/

Como nota final, espero que este artículo les haya echo conocer todos los aspectos o los más importantes del proyecto OLPC y las XO, este año, creo que aquí en Uruguay se van a distribuir a todos los niños de las escuelas públicas. Le tengo mucha fé a este proyecto y a sus intenciones, Arriba OLPC ! :)

Fuentes de la información:

- laptops.org
- wikipedia.org
- fing.edu.uy

Cypress

para Hack Hispano y Hacking Rioplatense hackingrioplatense.com.ar :)

Agradezco especialmente a Yasim por sus colaboraciones y correciones, él es parte de este articulo también.

MAQUINAS VIRTUALES EN NUESTRO PC

Una de las grandes desventajas de los sistemas operativos de la actualidad es la incompatibilidad de sus aplicaciones. Bien sea por motivos de hardware como puede ser el abismo existente entre las arquitecturas powerPc y las x86, o bien sea por motivos de software como ocurre en los entornos privativos de Windows, que muchas de sus librerías no han sido traducidas a entornos libres.

Aun salvando estas distancias, seguro que a alguno de nosotros nos ha ocurrido estar trabajando en un sistema Windows y tener que usar un entorno Unix Like para alguna cuestión.

Para esto dedicamos este articulo, para ver que opciones tenemos en uno de estos problemas Ejecutando varios ordenadores y/o sistemas operativos dentro de un mismo hardware de manera simultánea.

No vamos a hacer demasiado hincapié en la historia de esta tecnología, ya que puede recabarse en Internet o en cualquier libre de computación, pero si debemos hacer unos pequeños apuntes.

Una de las primeras ideas es lanzada por Vmware, el "Mware Virtual Plataform", que a comienzos de 1999 era capaz de virtualizar arquitecturas x86.

A partir de ahí, surge un mercado de tratar de unir las ventajas de todos los sistemas bajo una misma máquina.

Quizás uno de los entornos mas extendidos en el mercado sea el Vmware, proveniente de la casa EMC. Este software es capaz de ejecutarse en sistemas tan dispares como Windows, MacOS y Unix.

Una de las desventajas de este software es que es un software propietario, y necesitamos un licencia para poder correrlo en nuestros equipos. Lo mismo le va a ocurrir a Virtual PC, su homónimo de la casa Microsoft (que aunque no fue desarrollado por ellos si lo adquieron por una importante suma), y que también , como es tradición en la compañía de Bil Gates, es software privativo. La diferencia entre ambos es que Vmware esta ejecutando la emulación directamente sobre el hardware que nosotros poseemos, Virtual Pc lo que realiza es una traducción simultaneas de las operaciones requeridas, traduciéndolas.

Nosotros, sin embargo, siguiendo la filosofía hacker, nos vamos a servir de una herramienta libre, llamada Virtual Box. Como bien sabéis, no todo el monte es orégano, y en sus orígenes fueron comerciales. **VirtualBox** es un programa de virtualización creado por la empresa alemana de desarrollo de software innotek GmbH. El programa es comercial y propietario, si bien en enero de 2007, después de muchos años de desarrollo, se lanzó una versión limitada de VirtualBox bajo licencia GPL.

VirtualBox está disponible para su ejecución en sistemas Windows y Linux de 32-bits (aunque hay también una versión beta para MacOS X) y es capaz de virtualizar Windows, Linux (versión del núcleo 2.x), OS/2 Warp, OpenBSD y FreeBSD.

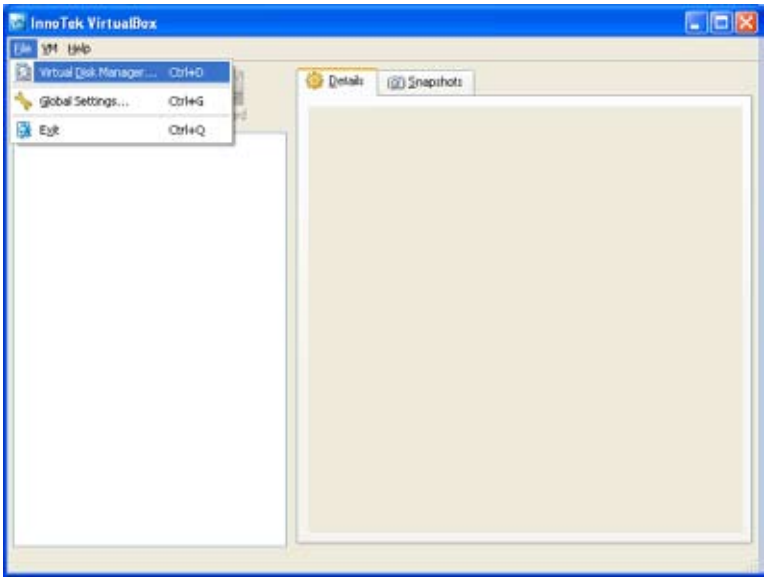
Lo primero que tenemos que hacer es descargarnos el paquete de su web oficial, para vamos a www.virtualbox.org y buscamos el link de descarga. En principio usaremos un entorno Windows como Host anfitrión.

Aquí debemos destacar otra virtud de VirtualBox, que es su reducido peso respecto a sus homólogos privativos. Apenas si pesa 10 Mb, con respecto a los más de 100 de sus competidores.

Una vez descargado en nuestro equipo , procederemos a la instalacion. Dicho ejecutable nos guiara paso a paso , eso si, en ingles , y nos ire preguntando cuestiones que vamos a explicar detalladamente.

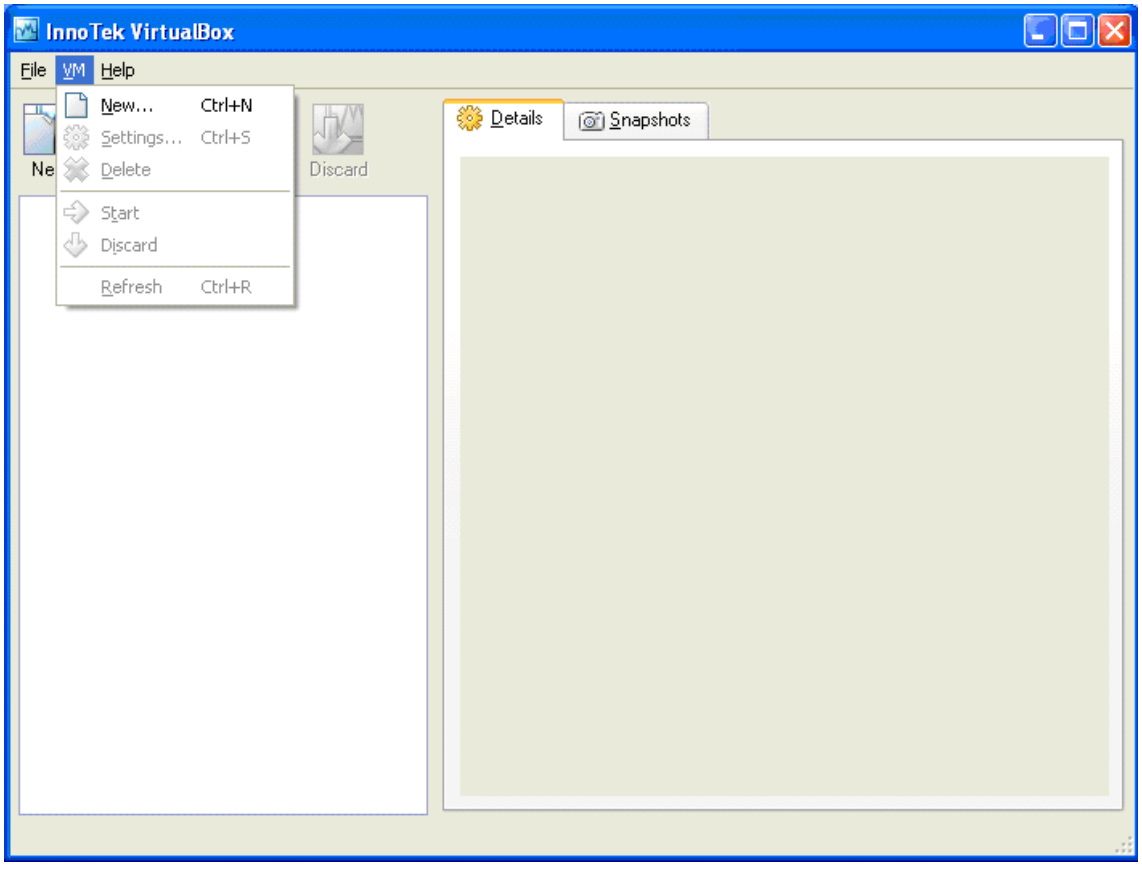
Este es el primer rotulo que se nos presenta en la instalación de Virtual Box, el cartel de presentación, en el que simplemente destacaremos la versiones que hemos usado para el articulo, y proseguiremos haciendo clic en siguiente.

Llegamos a un punto importante de la instalación del programa. En la parte final de la instalación nos aparece este cartel que nos avisa que este software no ha pasado la prueba del logotipo de Windows; esto sucede porque Windows verifica todo el hardware que hay instalado en el equipo, en principio para verificar el correcto funcionamiento entre el tandem hardware-software, y para evitar incompatibilidades. Ahora bien, si indagamos un poco mas podríamos encontrarnos patentes, marcas, acuerdos, licencias, que las pequeñas empresas de hard no pueden permitirse, y aunque son 100 % compatibles, no superan el criterio de Microsoft. También apuntamos otra cosa. Windows XP , nos lanza este aviso, pero nosotros continuaremos y no volverá a quejarse por ninguna otra cosa. Si quisiéramos realizar la prueba en Windows Vista, seria imposible, ya que Vista no concibe nada que no sea aprobado por la prueba del logotipo de Windows, es decir, si no esta firmado, no es posible instalarlo. Es como la antitesis del slogan que aparecía últimamente en TV, "si no es bueno para nosotros no es bueno para ti" .

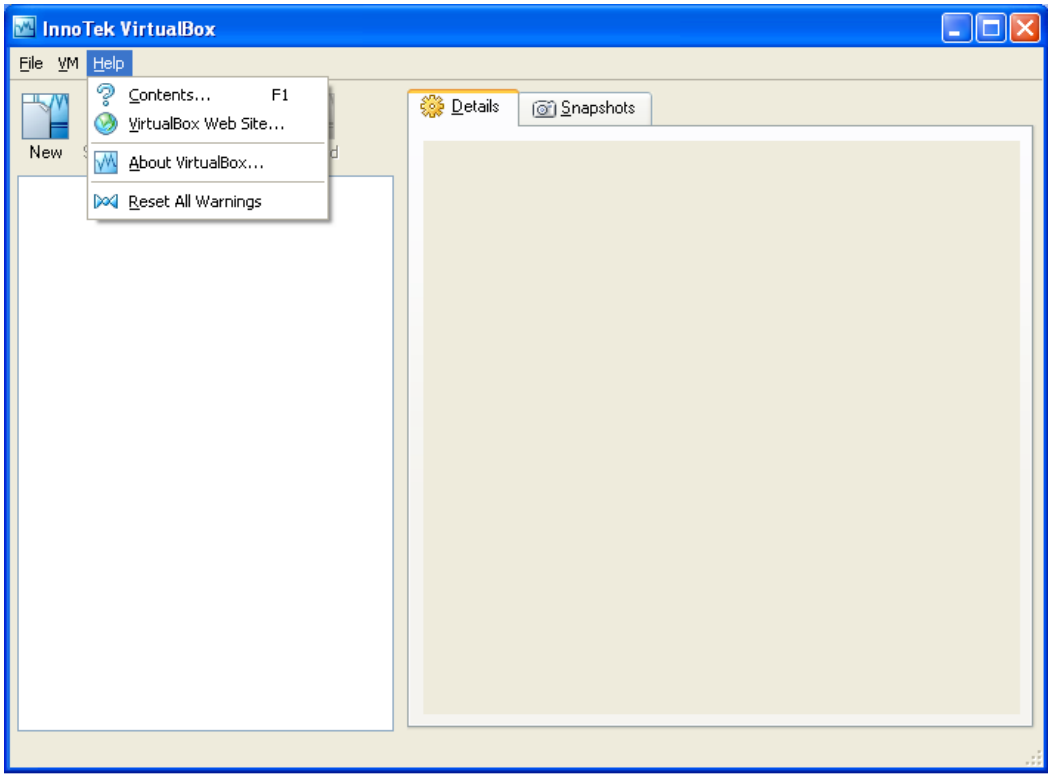


Veamos ahora las opciones que tiene nuestro programa. Dentro de los diferentes menús, podemos ver aquí las opciones más importantes. El menú File, nos va a dar la herramientas para configurar nuestro Virtual Box, tanto el programa en si como cada una de las maquinas virtuales por separado. Tambien podemos observar que existen una serie de atajos de teclado, que nos facilitaran el movernos mejor por el programa, y también por las diferentes maquinas que hayamos configurado. La primera de las opciones de este menú, nos permite gestionar los discos duros virtuales, es decir, el espacio de nuestro disco que cederemos para usar como si fuesen particiones reales.

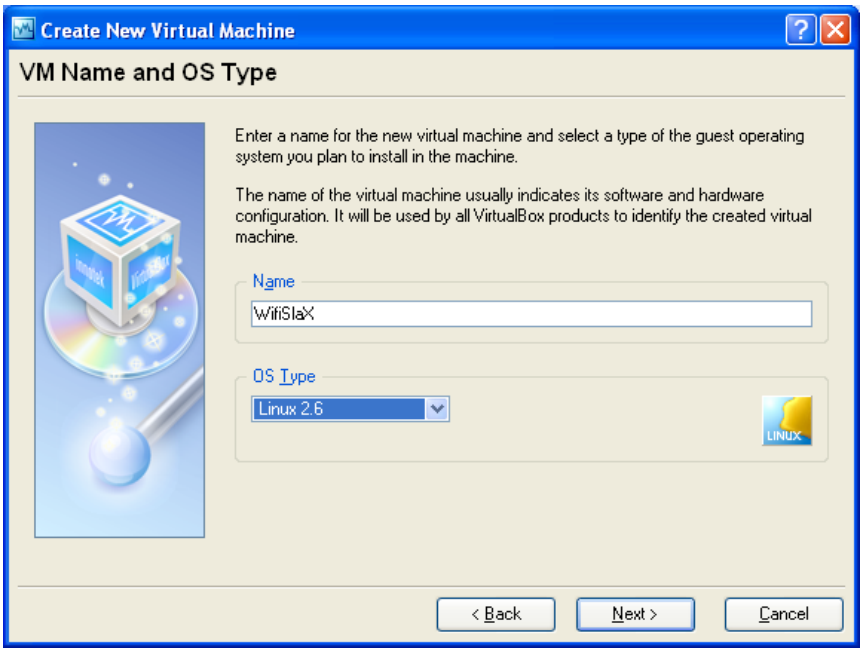
El siguiente menú es el VM (virtual machine o maquina virtual). Este menú nos va a permitir crear nuevas maquinas virtuales o editar y modificar las ya creadas por nosotros.



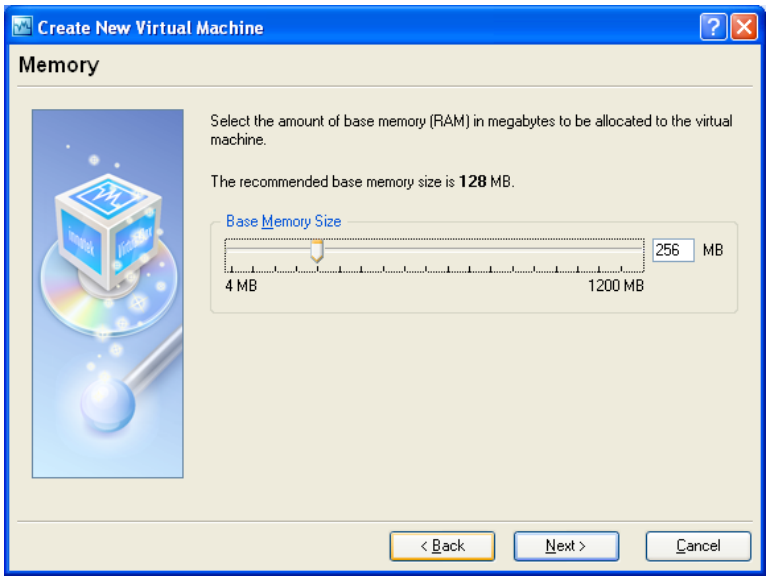
El ultimo de los menús es el Help, que nos mostrara las características de Virtual Box, con enlace directo a su web oficial , y con su propio menú de ayudas.



Bueno y sin más preámbulos vayamos a la parte interesante de este tutorial. Para crear una nueva maquina virtual vamos al menú VM y le decimos new machine o presionamos Ctrl+ N. Nos saldrá la siguiente pantalla. En ella especificaremos el nombre y el tipo de sistema que vamos a crear. En mi caso probare la distro WifiSlax, que tiene un kernel 2.6 .



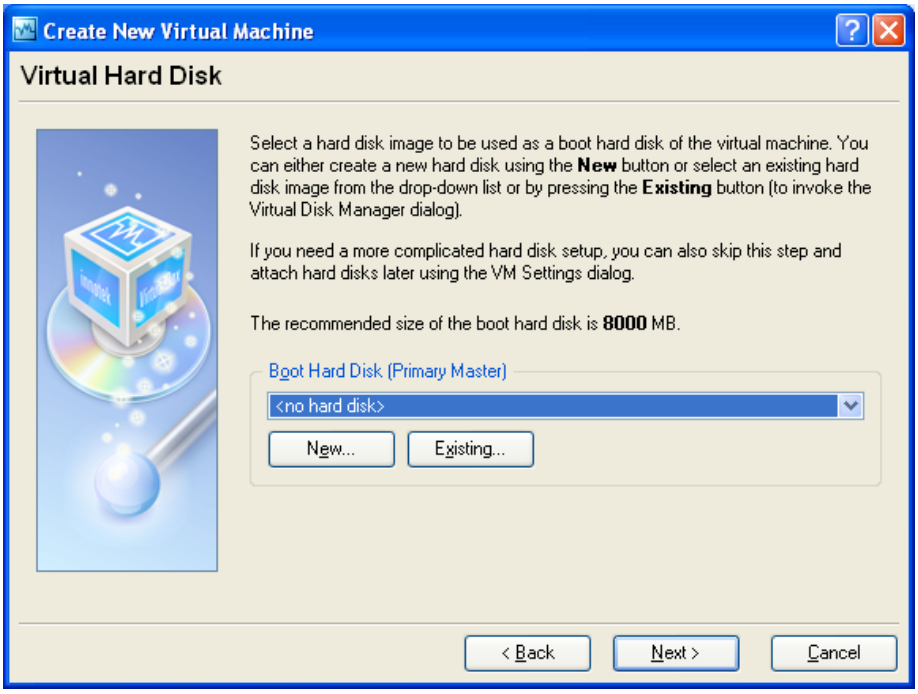
En la siguiente pantalla nos va a requerir que indiquemos la cantidad de memoria que queremos dedicar. Aquí ya va a depender de la configuración de nuestro PC. Yo en mi caso estoy usando un Celeron Hacer con 1'6 Ghz con 1 Gb de RAM, y dedicare 256 Mb.

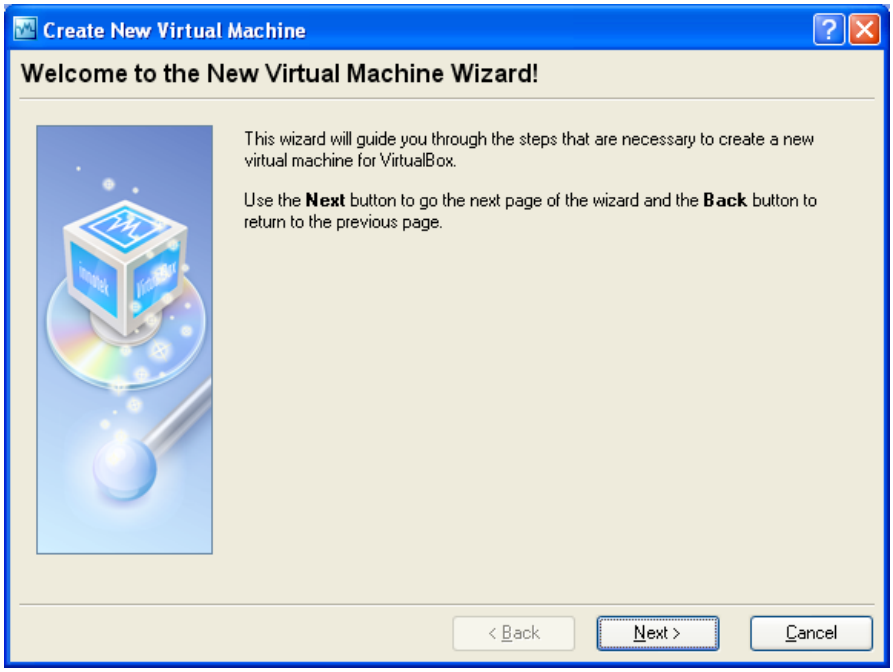


El Siguiente paso del asistente nos va a preguntar si queremos usar una partición virtual o si queremos que se ejecute desde el propio CD. En nuestro caso, vamos a decirle que crearemos una nueva partición virtual, para ello pincharemos el boton New.

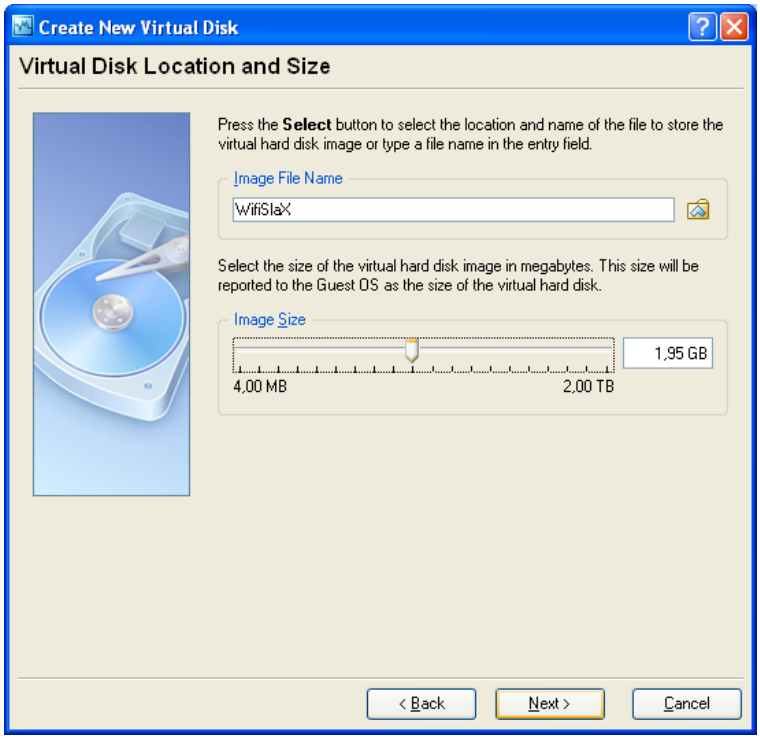
Tambien podriamos reutilizar una existente, o como hemos dicho, no usar ninguna.

Para el siguiente paso, necesitaremos hacer varios pasos.



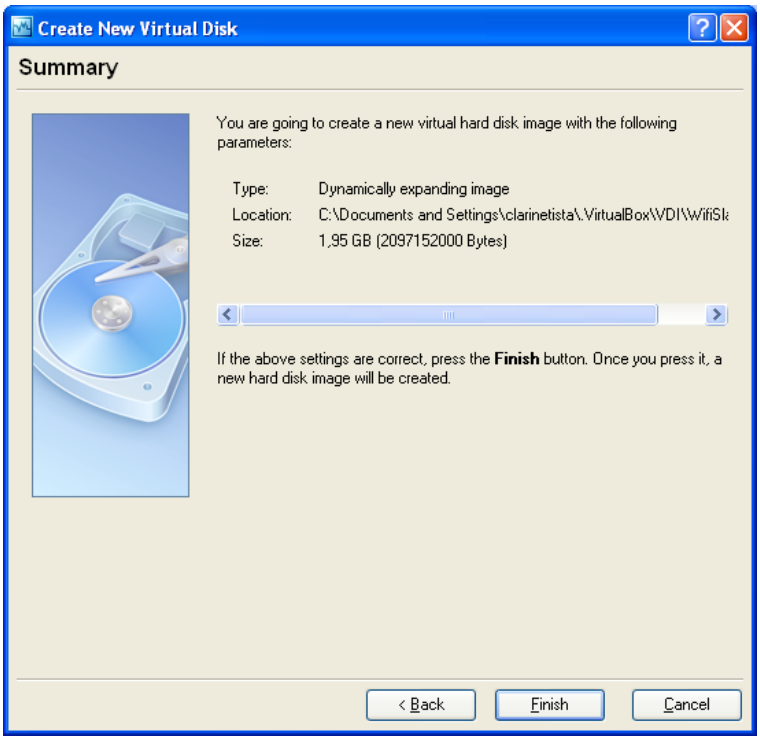


Primero deberemos verificar que tenemos suficiente espacio en el disco y que sistema utilizaremos, para saber cuanto espacio de disco debemos de asignar. Tras ello nos aparecerá la siguiente ventana, en la cual le pondremos nombre a nuestra partición (que nos será muy útil si queremos usar varios sistemas operativos con características comunes.)

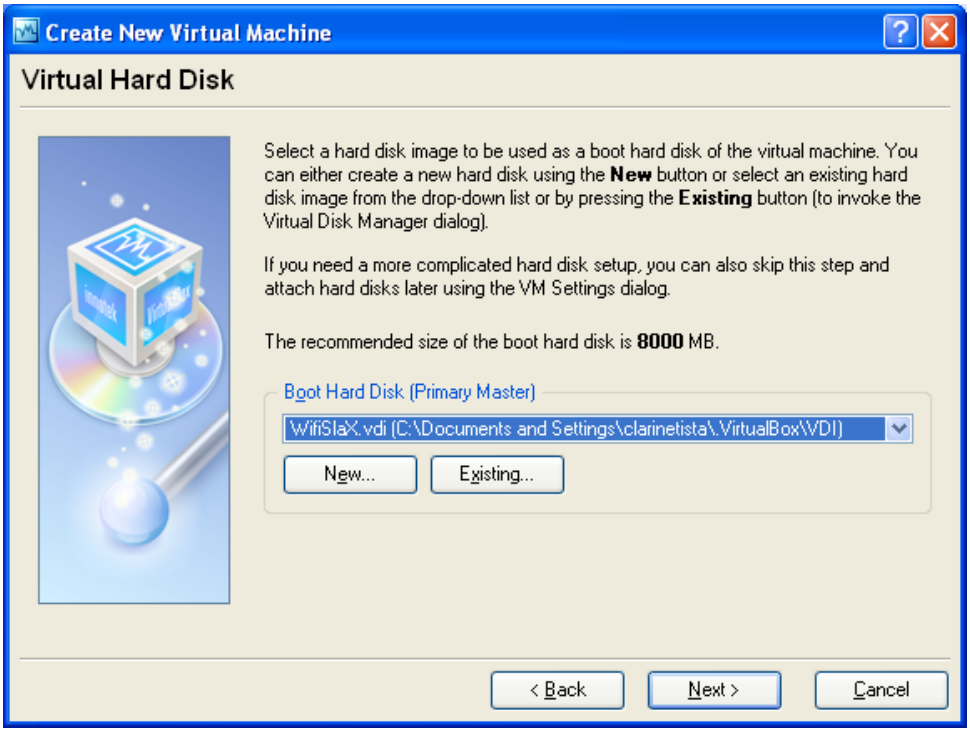


En mi caso voy a utilizar la distrubucion WifiSlax, especializada en auditorias y seguridad inalambrica, y le voy a asignar una partición virtual de 1'95 Gb.

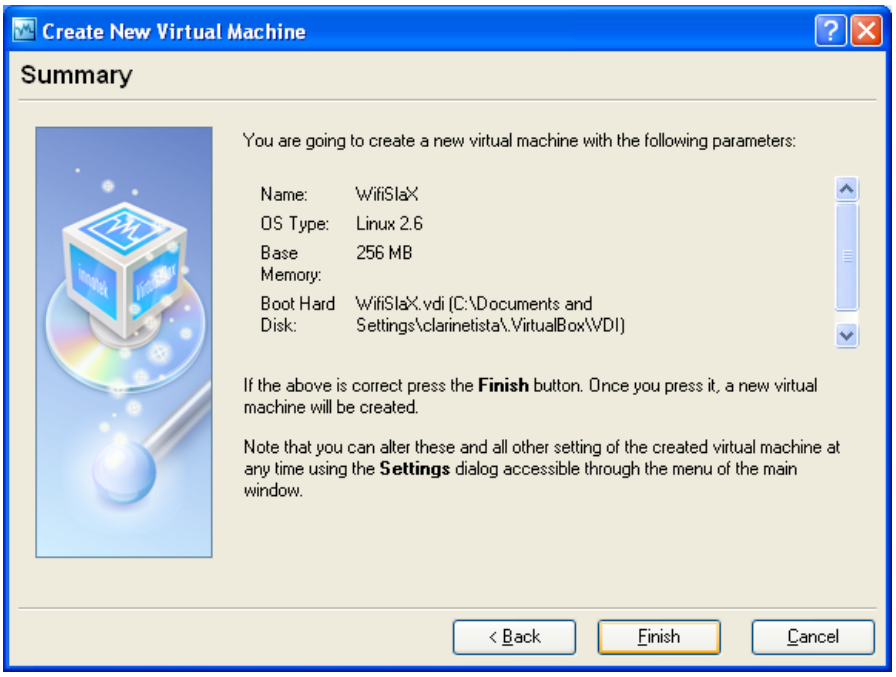
Tras ello no saldra una ventana resumen con las principales características de nuestra partición virtual. Cabe destacar la ubicación de las misma, que tanto en Windows como en entornos GNU/Linux, se guarda en la carpeta de los perfiles de usuarios. Tras ello pulsaremos en Finish para terminar el proceso.



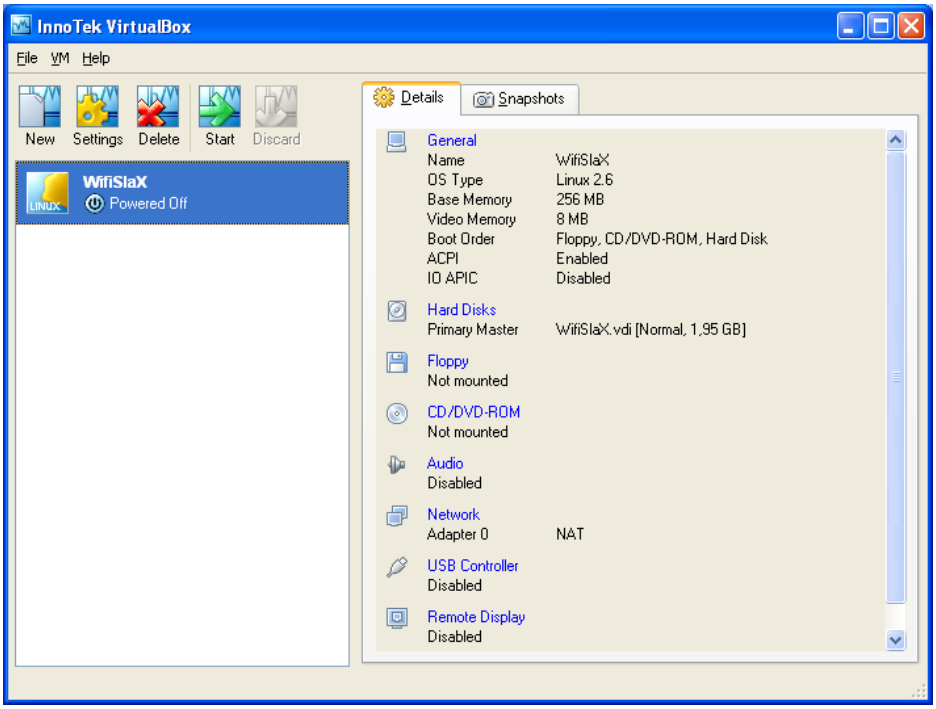
Una vez creada, volvemos al menú anterior de creación de nueva maquina, y le tenemos que decir que usaremos una exitente (la que acabamos de crear). Debemos de mencionar que la extensión por defecto de Virtual Box es vdi, por si alguna vez nos surge la duda .



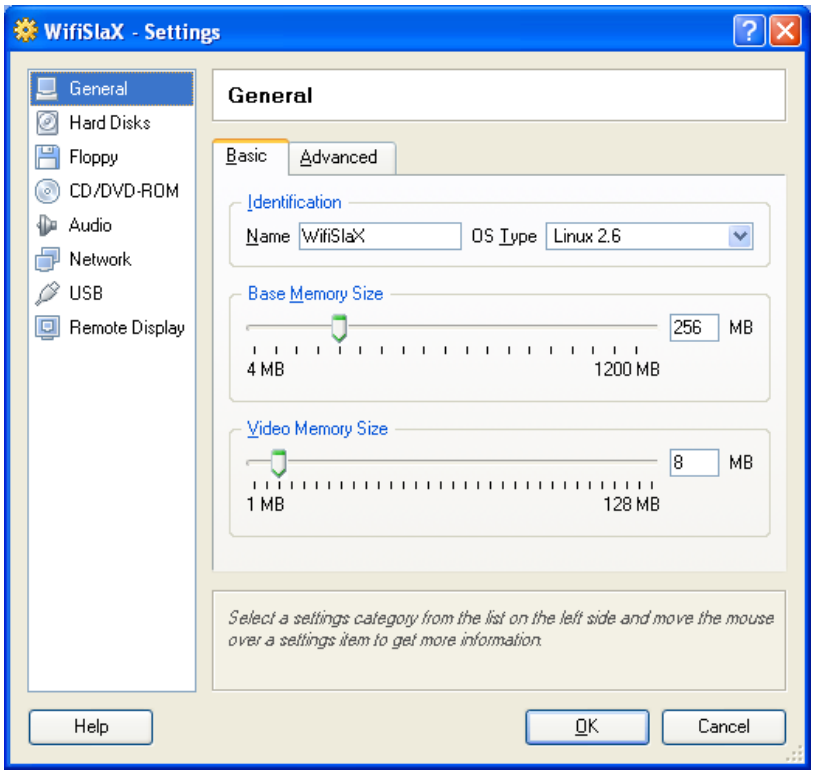
Pulsaremos siguiente y nos saldrá un resumen general de la maquina que hemos creado para que finalicemos estableciendo los cambios. Recordad que si queremos editar algo posteriormente lo podremos hacer desde el menú VM.



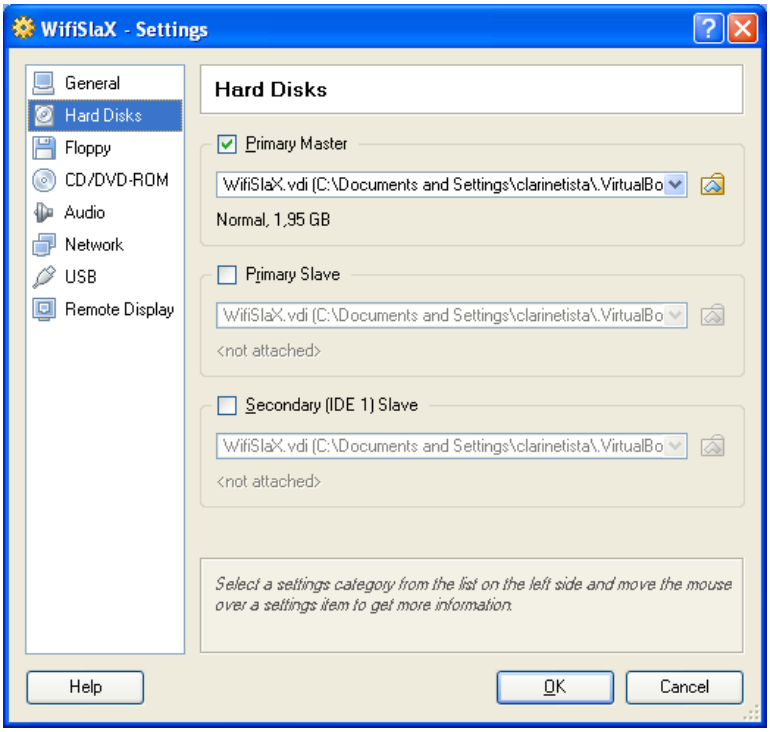
Aquí vemos como queda nuestro programa tras la nueva instalacion. Para arrancar el equipo virtual solo deberemos pulsar el boton Start. Si por el contrario quisiéramos borrar la maquina virtual existente, usaríamos el delete. El New es para repetir los pasos anteriores y crear una nueva VM, y el settings es para modificar las opciones de una existente.



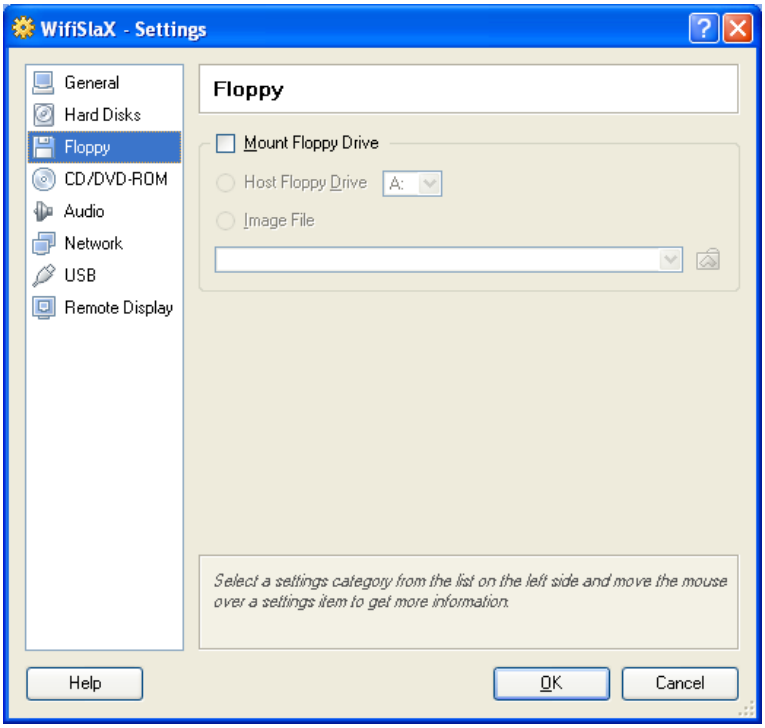
En este ultimo podemos ver las siguientes caracteristicas.



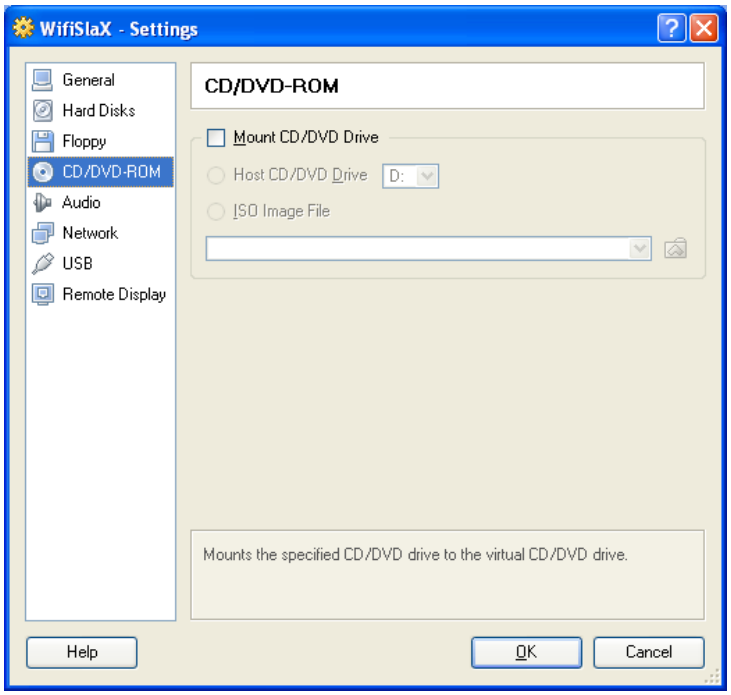
Hard Disk edita los disco duros, los cuales pueden establecerse como primario y secundario, mestro y esclavo, según la conveniencia.



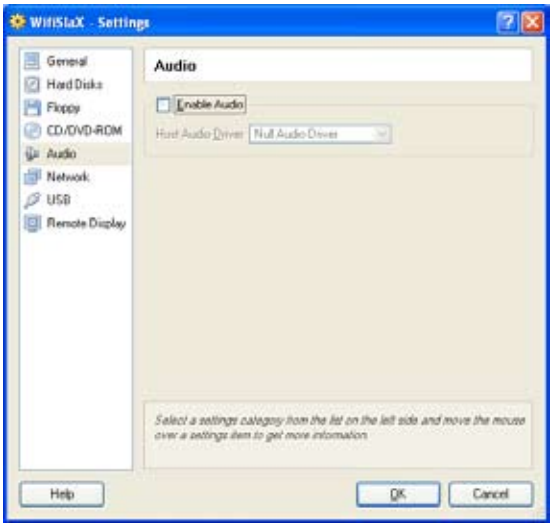
Floppy nos montaria la disquetera, aunque casi tiende a desaparecer .

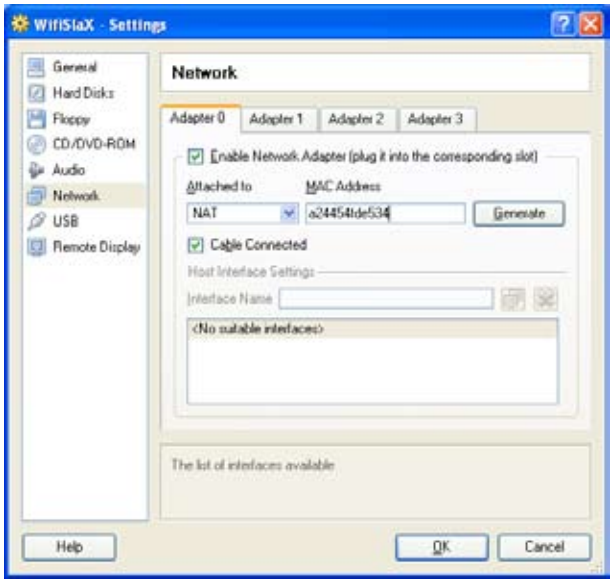


Lo mismo va a ocurrir con el CD/DVD, y tambien nos dejara usar una imagen iso.

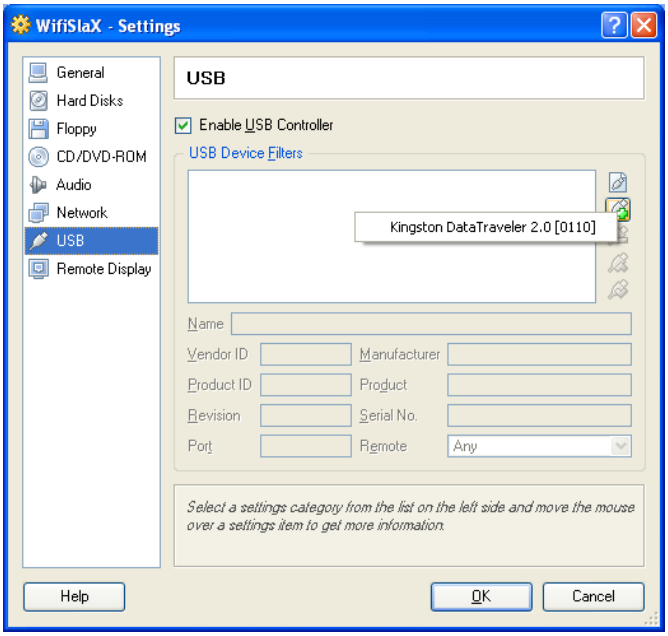


Tambien tenemos la opcion de configurar el driver de audio y la red. La red asigna un DHCP con una montado de NAT, con la MAC de nuestro equipo.



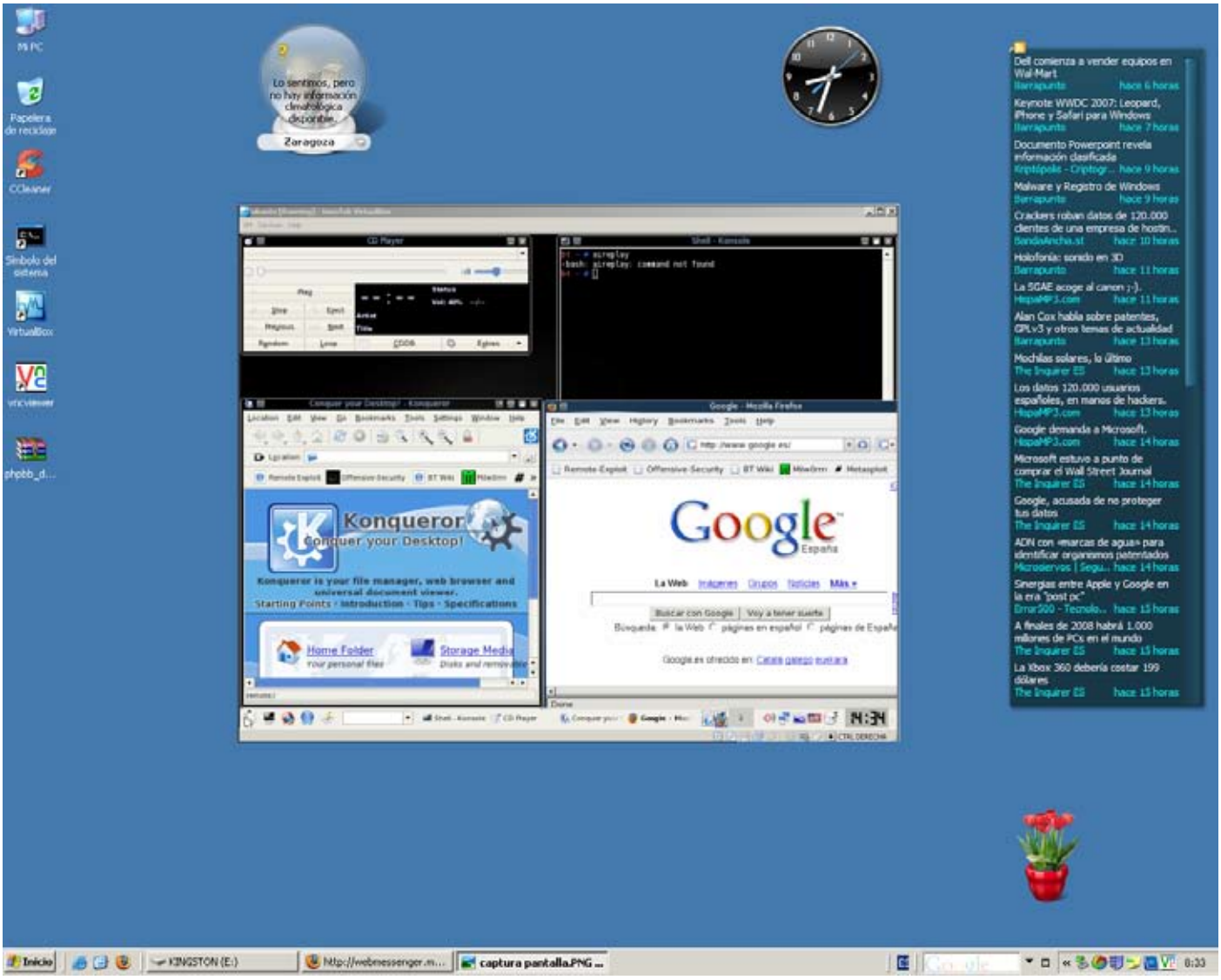


Otro punto de inflexión es la configuración de los dispositivos USB. Los mismos, los haremos del siguiente menú:



Si marcamos la pestaña de Enable USB Controller, vamos a permitir el uso de dispositivos USB en nuestra maquina, desde pendrives hasta impresoras. El propio programa detecta automáticamente cada dispositivo, y tan solo deberemos conectarlo y presionando el segundo de los botones de la derecha, nos dirá los que va reconociendo. En mi caso me ha detectado, como vemos en la imagen un pendrive marca Kingston. Si por un casual, no reconoce el dispositivo, deberemos rellenar cada uno de los campos que ahora aparecen deshabilitados manualmente. La ultima opcion es tan simple como un visor remoto, con varias opciones de configuración. No podria ser util en servidores Unix, como gestor de las X.

Por ultimo, quiero dejar una prueba de cómo funciona en vivo este programa



Espero que este articulo os sriva de ayuda, y os empieze a motivar a probar tecnologías de Software libre y para hacer pequeñas incursiones en sistemas operativos GNU/Linux y porque no decirlo en un futuro Unix o BSD .

Clarinetista

MODIFICAR CÓDIGO EN TIEMPO DE EJECUCIÓN. API WIN32+ENSAMBLADOR

En este artículo veremos cómo modificar el código de un programa en tiempo de ejecución.

Sabemos que cuando el sistema operativo crea un nuevo proceso, ya sea lanzado automáticamente o bien por parte del usuario, crea una serie de estructuras de datos para la gestión de éste, además de ello, debe especificarle al procesador del sistema en qué punto debe comenzar a ejecutar las instrucciones del programa que se va a ejecutar y en qué zona se van a almacenar los datos que necesita temporalmente para la ejecución. Para ello el procesador dispone de una serie de registros: CS (código), DS (datos), SS (pila), etc... (para más información sobre procesos del sistema, leer el artículo "Procesos del Sistema: Introducción a la API de Windows - III Parte")

Windows protege el segmento de código de un programa ante la escritura. Por defecto es "solo lectura". Es decir los bytes almacenados en el segmento de la memoria del proceso en cuestión solo se pueden leer.

Utilizando la función de la API VirtualProtect es posible asignar nuevos permisos a una página completa de memoria sobre el proceso actual. Existe a su vez otra función análoga VirtualProtectEx que realiza la misma función pero para gestionar los permisos en el espacio de memoria de otro proceso. Es bastante interesante para temas de inyección de código en el espacio de memoria de otro proceso, por ejemplo útil para muchos virus y otros programas con el fin de saltarse protecciones del sistema o para escalar privilegios en el sistema, o simplemente para modificar el normal funcionamiento del proceso en cuestión.

Como sabemos, la dirección de la siguiente instrucción a ejecutar está contenida en el registro EIP (32 bits) del procesador. Al principio de la ejecución del programa, apuntará a una dirección especificada por el registro CS. Cabe pensar que si dicho registro es de 32 bits, entonces virtualmente sólo se puede acceder a 4GB virtuales para los procesos, y así es, por tanto para no tener tal limitación se utiliza la denominada memoria virtual, para el acceso a cada uno de los distintos niveles dentro de la jerarquía de memoria (caché, memoria principal RAM, disco duro, etc..., que se hace traduciendo la dirección virtual del procesador, es decir la dirección del registro EIP, concatenada con una dirección contenida en una tabla de páginas, obteniéndose así una dirección física que es la que accede a la memoria correspondiente de la jerarquía). El acceso a cada uno de los distintos niveles de dicha jerarquía es totalmente diferente, y dependerá no sólo del nivel, sino del tipo de memoria dentro del nivel. Por ejemplo el acceso a una memoria caché de mapeado directo atacada por direcciones virtuales se suele implementar simplemente a partir de la dirección virtual, obteniendo a partir de ésta, la etiqueta, el índice y el desplazamiento adecuado para el acceso al dato correspondiente. Del mismo modo el acceso a una caché completamente asociativa, atacada por direcciones físicas, se hace concatenando la dirección virtual del procesador con una dirección contenida en una tabla de páginas. Y esto es totalmente distinto para el acceso a un dato en memoria principal o en disco (mayores tiempos de acceso y mayor capacidad), en donde es necesario incorporar buffers para el almacenamiento de datos y de incorporar puentes (chipset) para controlar las distintas transferencias. Por ejemplo el puente norte para el acceso a RAM.

Por tanto cabe pensar que si logramos en nuestro programa la obtención del puntero de instrucción (EIP), para así simplemente hacer algo como MOV [EIP], valor, pues ya tendríamos el problema resuelto. Pero resulta que el registro EIP no es accesible por código. Es decir el procesador no incorpora en su juego de instrucciones un modo de direccionamiento como el antes mencionado (MOV EIP, VALOR). Así que nos la ingeniaremos para inventar un pequeño truco, aprovechando la pila para obtener la dirección actual.

Como bien sabemos, cada vez que se hace una llamada, CALL a un procedimiento o una función (una subrutina en general), se almacena en la pila la dirección de retorno. Pues bien, si la subrutina a la que llama contiene en sus dos primeras instrucciones un POP REGISTRO, sacamos de la pila dicha dirección y la almacenamos en dicho REGISTRO. Posteriormente hemos de insertar nuevamente en la pila dicha dirección de retorno para que todo vaya correctamente (PUSH REGISTRO) tras el RET, que es quien verdaderamente saca de la pila la dirección de retorno y accede al registro EIP para continuar la ejecución del programa tras la llamada.

Teniendo ya en dicho REGISTRO la dirección de nuestro código (antes de la llamada), pues haciendo algo como MOV [REGISTRO+offset], VALOR, siendo el offset el desplazamiento hacia donde queremos escribir la nueva instrucción, supuestamente debería funcionar (porque si pertenece a un modo de direccionamiento del procesador).

Pero no tan deprisa, aunque parezca que ya hemos visto la luz, recordemos que nuestro querido Windows protege dicha zona de memoria de código, y aun debemos dar el permiso de escritura correspondiente, y aquí es donde entra la función VirtualProtect, que recibe como:

primer parámetro: la dirección cuyos permisos queremos modificar,
segundo parámetro: el número de páginas sobre los que tendrá efecto el cambio de permisos,
tercer parámetro: es una constante que indica el nuevo permiso,
y el cuarto parámetro: es un puntero a la dirección para guardar los permisos anteriores y poder restaurarlos más adelante.

Tras la llamada a VirtualProtect, si todo ha ido exitoso, habremos cambiado el permiso al que deseemos. Tras lo cual, haciendo la operación de MOV [REGISTRO+offset], VALOR, pondremos dicho VALOR en la dirección que queramos modificar, machacando literalmente la instrucción que exista.

Para llevar a cabo toda la teoría, (espero no haberme enrollado demasiado), he realizado un programa de ejemplo, implementado en Delphi, con el cual espero aclarar conceptos.

Describiendo el funcionamiento, tras la inicialización de los registros necesarios, punteros de pila y demás (PUSH EBP y MOV EBP,ESP), lo primero es llamar a la subrutina o procedimiento GetEIP. Haciendo ésto habremos metido en la pila la dirección de retorno (es un CALL). Cuando entre en la subrutina GetEIP, lo primero que haremos será sacar dicha dirección de retorno de la pila y guardarla en el registro EBX. Es importante no destrozar el contexto sobre el que se trabaja, ya que podríamos romper el buen funcionamiento del programa, y será necesario escoger un registro no utilizado, o bien, utilizar uno cualquiera y restaurarlo al final tal y como estaba. Yo he usado EBX, puesto que estaba inutilizado. Tras haber sacado la dirección de la pila, y guardandolo en EBX, no debemos olvidar volverlo a meter en la pila, para que tras finalizar la rutina se vuelva a la ejecución normal del código.

En dicha subrutina he aprovechado y he puesto en EBX el offset o desplazamiento sobre el que queremos modificar el código, aunque podríamos haberlo puesto fuera de dicha subrutina, pero es más correcto así, ya que será pasado como argumento a la funcion VirtualProtect.

A continuación llamamos a VirtualProtect, contenida en Kernel32.dll, pasándole como parámetro la dirección a la que queremos cambiar los permisos. Como el código del programa es pequeño, he puesto que solo modifique una página, y que de permisos de lectura y escritura (PAGE_READWRITE, constante de valor 4), y el último parámetro es el puntero sobre el cual se van a guardar los viejos permisos de dicha dirección.

Una vez ejecutada dicha función, ya tendremos desprotegida la página del segmento de código de nuestro programa, con lo cual, ya si podemos escribir directamente en dicha zona.

He puesto una serie de NOP's, (No OPeration, de código 90 en hexadecimal, cuya función es "ocupa un byte y no hacer nada"), que serán los bytes que modifiquemos, para dejarlo lo más claro posible. Por tanto en EBX+Offset se encuentra el primer

NOP.

Es importante tener en cuenta el detalle de que lo que se escribe debe tener el mismo tamaño que lo que había antes, si no, corremos el riesgo de una bonita excepción de desbordamiento jejeje.

El primer MOV BYTE PTR [EBX],184 sirve para sustituir el primer NOP (90), por un MOV EAX, ... (B8).

El segundo MOV DWORD PTR [EBX+1],00 sirve para especificar el operando fuente del MOV anterior. Observar que ahora es EBX+1, puesto que tras ejecutar el primer MOV ya habremos ocupado 1 byte, y la siguiente instrucción a escribir se encontrará en el byte siguiente. Además he especificado DWORD porque la instrucción MOV a utilizar va a mover un dato de 32 bits. (para poner EAX a 0: MOV EAX,0 será el resultado final).

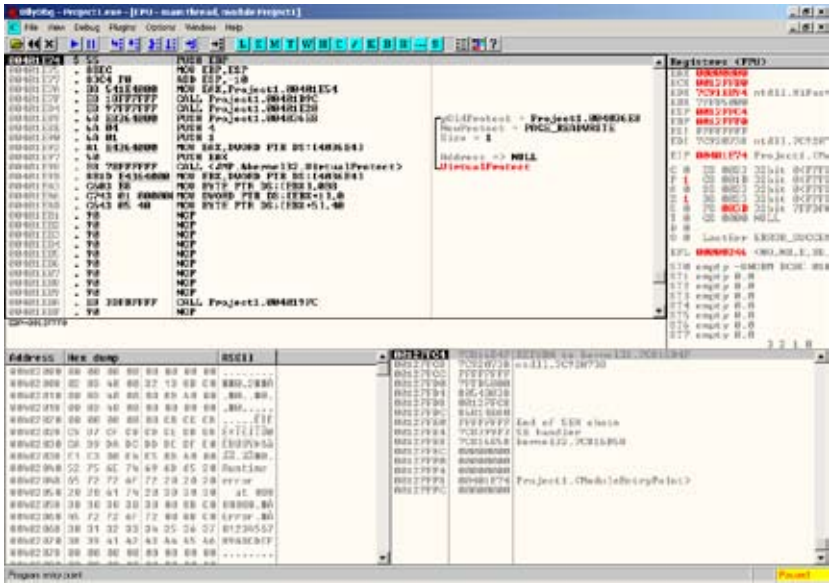
El tercer y último MOV BYTE PTR [EBX+5],64, ya corresponde a la última instrucción que vamos a insertar, corresponde a un INC EAX (incrementa EAX en una unidad). No olvidar que debemos poner EBX+5, puesto que los dos MOV anteriores (es decir, el MOV EAX,0, ocupa 5 bytes), y por tanto hemos de insertar la siguiente instrucción 5 bytes después.

Podríamos haber ido incrementando lo correspondiente el registro EBX tras cada MOV, pero bueno como son poquitas, pues por ahorrar instrucciones...jejeje.

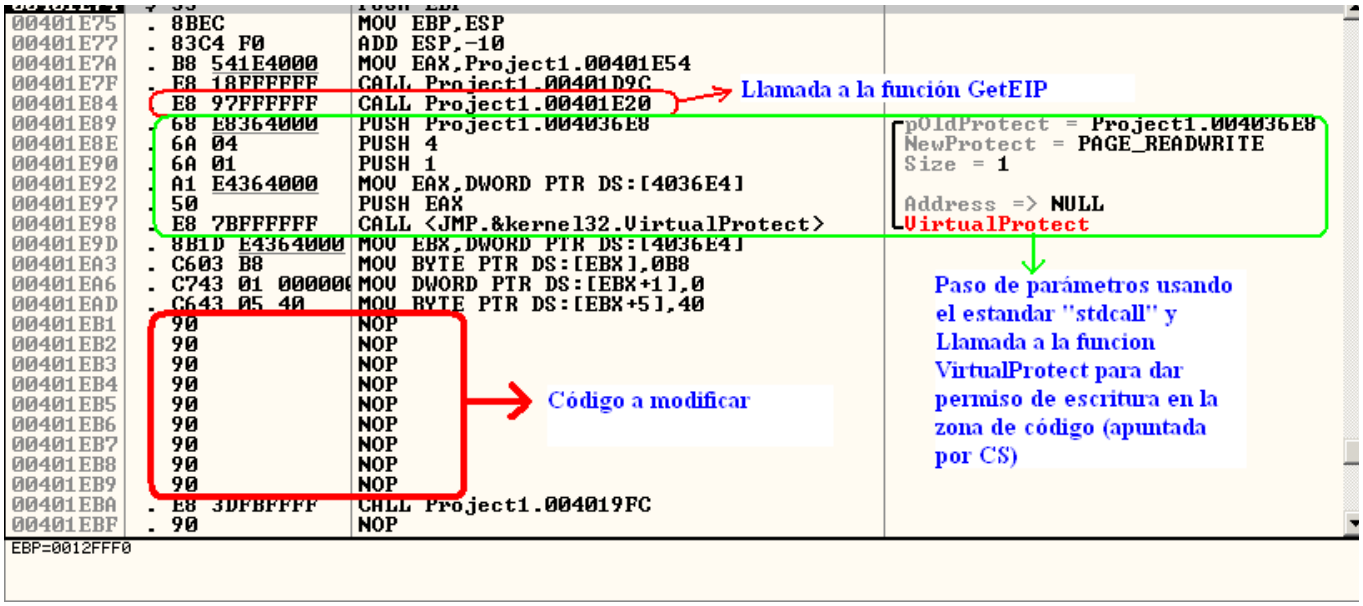
Sin más enrollarme, he aquí el código del programa:

```
program Project1;

function VirtualProtect(lpAddress: Pointer;
dwSize, flNewProtect: LongWord;
    lpflOldProtect:  Pointer):  LongBool;
stdcall;
external        'kernel32.dll'        name
'VirtualProtect';

var
  dir, vieja: Pointer;

procedure GetEIP;
begin
  asm
    pop ebx
    push ebx
    add ebx,28h //EBX+28 es la dirección a
machacar (los NOPs)
    mov dir,ebx
  end;
end;


begin
  asm
    call GetEIP
  end;
  VirtualProtect(dir, 1, 4, @vieja);

  asm
    mov ebx,divr
    mov byte ptr [ebx],184  //184 es B8 en
Hex: un MOV EAX,...
     mov dword ptr [ebx+1],00   //pon el
operando fuente al MOV anterior: MOV EAX,0
    mov byte ptr [ebx+5],64 //64 es 40 en
Hex: un INC EAX
    nop
    nop
    nop
    nop
    nop
    nop
    nop
    nop
    nop
  end;
end.
```

Compilado y enlazado con Delphi, aquí está el programa Project1.exe desensamblado con OllyDbg, un depurador para Windows. Como todos sabréis, tiene 4 secciones, la superior izquierda con el código desensamblado, la superior derecha para ver el estado de los registros y de las banderas, la inferior izquierda para ver el código hexadecimal y editarlo, y la inferior derecha que es la pila.



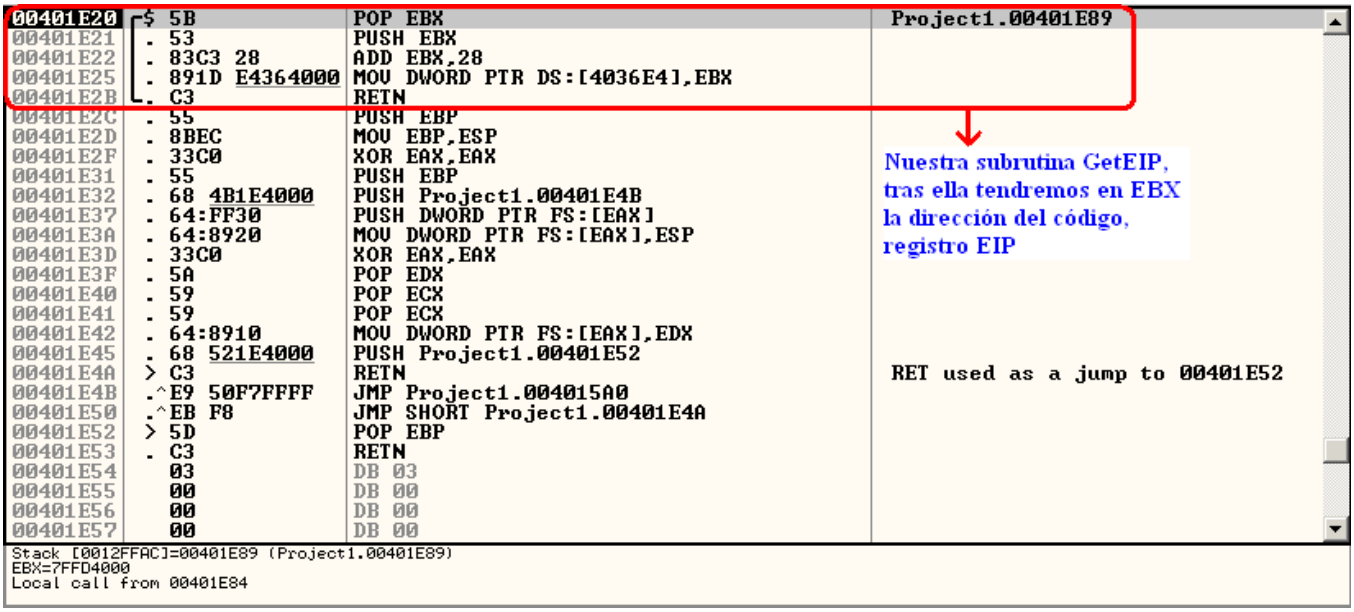
Aquí un resumen de cada una de las instrucciones del programa desensamblado:



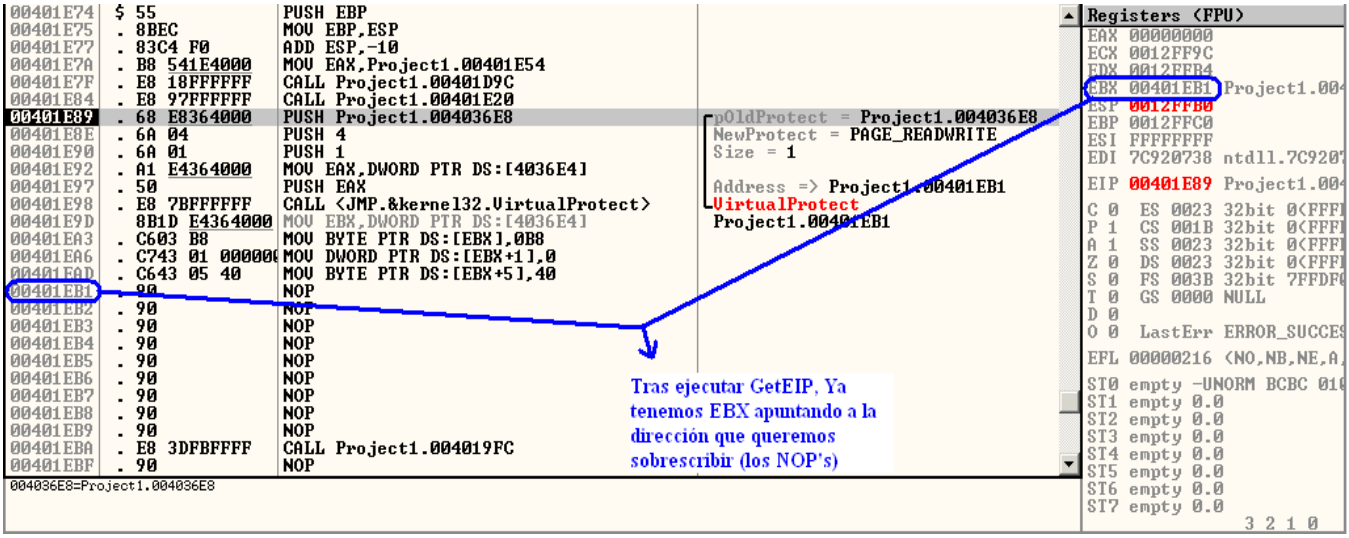
Los MOV justo anteriores a la lista de NOP's son los que sobrescribirán a éstos.

El primer y último CALL sirve para asignar y liberar respectivamente un Handle al proceso en cuestión. Como ya sabemos Windows maneja todo a través de un identificador o manejador. Esto ocurre en todos los procesos Windows.

Trazamos con F8 hasta el segundo CALL (para no entrar en el primero), y con F7 entraremos en el segundo CALL para observar la subrutina GetEIP:

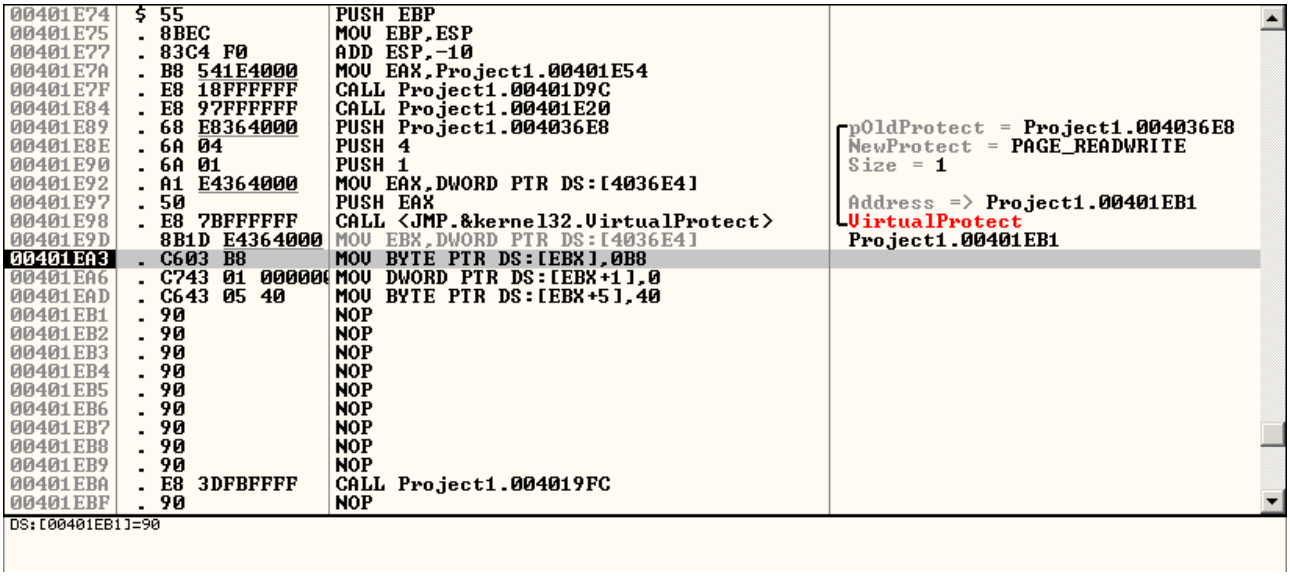


Observamos la Pila como inserta la dirección de retorno, tras ejecutar la subrutina (seguimos trazando con F8), tendremos en EBX dicha dirección, pero le habremos sumado el correspondiente desplazamiento hacia la dirección en la que vamos a sobreescribir, en este caso hacia el primer NOP:

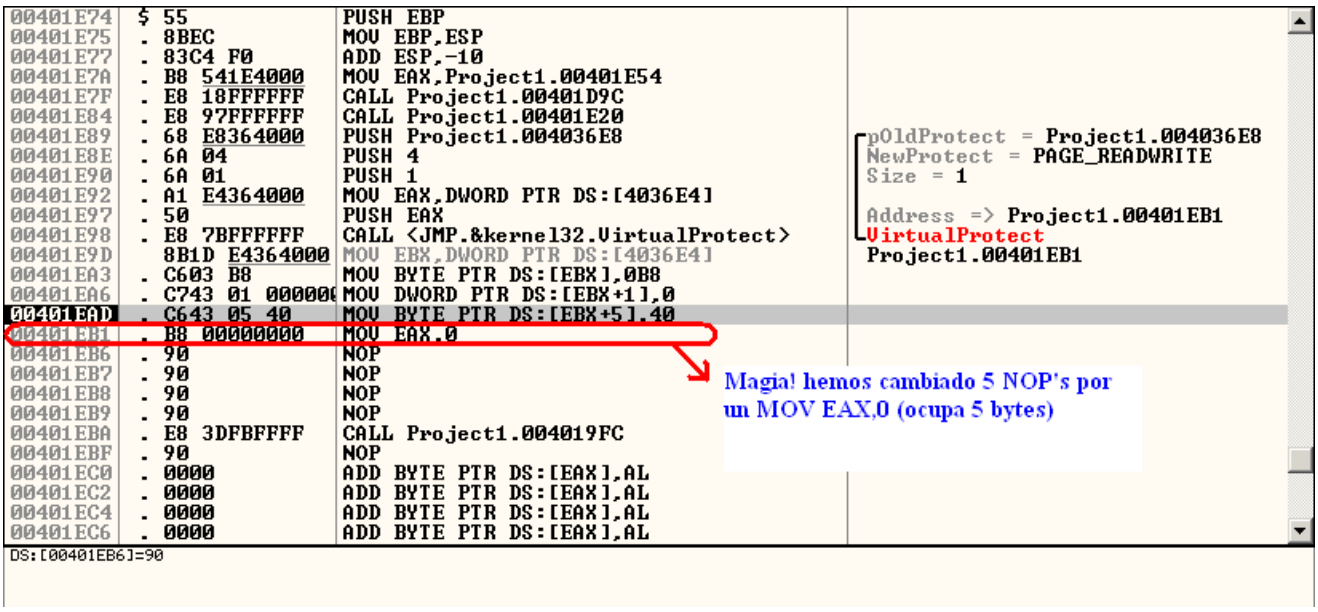


Ahora llamamos a VirtualProtect, para ello como hemos usado el estandar stdcall para las llamadas, los parámetros son pasados por pila de derecha a izquierda (se inserta en la pila del último parámetro al primero), de ahí la sucesión de los PUSH.

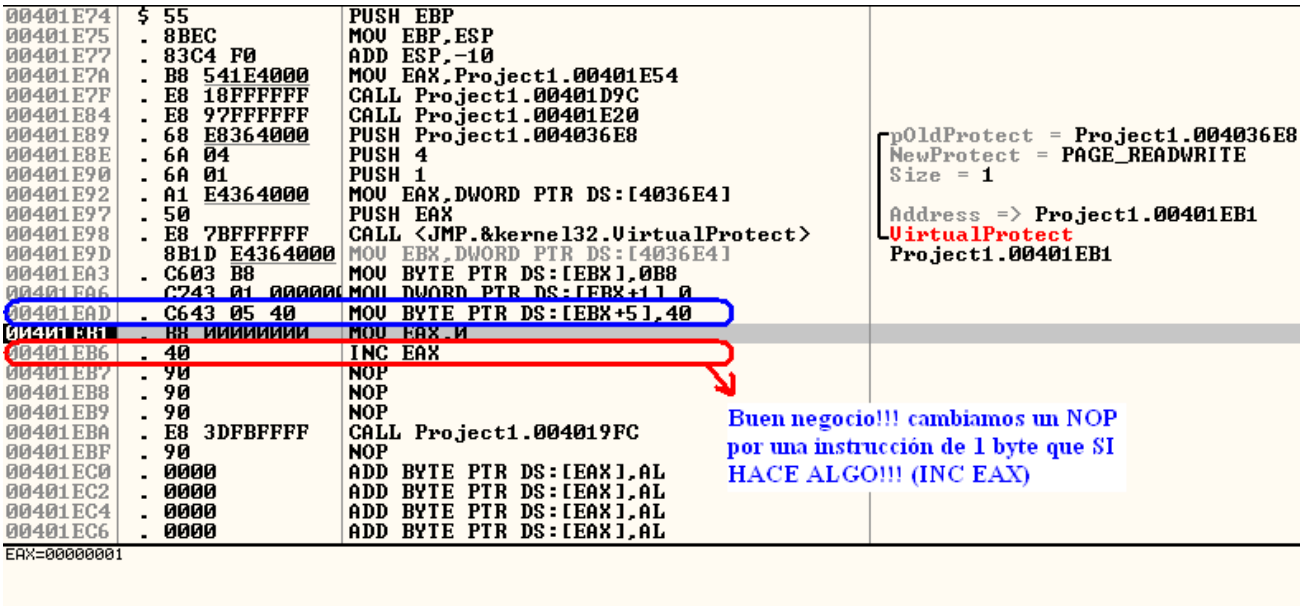
Seguimos trazando con F8, no hará falta entrar en la llamada a VirtualProtect, hasta situarnos en el primer MOV: MOV BYTE PTR DS:[EBX],0B8:



Y ahora es cuando observaremos los resultados (Observad los NOP's):



Por ultimo ejecutamos el último MOV que nos queda, para cambiar un NOP más por un INC EAX:



Hasta este punto, probablemente te estés preguntando ¿y para qué quiero yo poder hacer esto? pues bueno esta fricada dependerá del uso que tu le quieras dar... Desde simplemente volver loco a un Cracker para proteger tu código, es decir, que mientras esté depurando vea delante de sus narices como el código se va cambiando a medida que realiza una traza... Bonita debe ser la cara que se le quede... jejeje, hasta un simple uso como ejecutar instrucciones aleatoriamente... imaginad una función random, o simplemente usar las rutinas de servicio del DOS, para capturar por ejemplo la hora y fechas del sistema (con INT 21 y AH=2Ch) y que ejecute código en función de la hora de éste, perdón por poner un ejemplo tan friqui. Solo quería demostrar un ejemplo. Probablemente si lo hace se lleve a cabo más de una excepción o ¡¡¡quien sabe!!!, a lo mejor das con un superprograma que haga maravillas...

Lo curioso para añadir no es sólo la posibilidad que ofrece Windows a poder cambiar los permisos de la zona de código de nuestro proceso, sino de cualquier proceso en Windows. Esto es posible gracias a la función VirtualProtectEx, análoga a la comentada en este artículo, pero para procesos externos al nuestro.

Con lo cual, es posible acceder al espacio de memoria de otro proceso y cambiarle las instrucciones a nuestro antojo, logrando así que dicho programa (por ejemplo Internet Explorer, MSN, Notepad o Solitario) trabaje a nuestro gusto. O hablando en términos "hackers", no sólo cambiando sus instrucciones, sino inyectando las nuestras propias (de nuestro código de cosecha propia) para que se ejecute con los privilegios del programa inyectado. Útil por ejemplo para saltarse numerosas protecciones de firewalls y antivirus, o simplemente para ejecutar cualquier tipo de instrucción privilegiada y que si es accesible por dicho programa "víctima".

Samir Sabbagh Sequera (HySTD)

PROCESOS EN UN SISTEMA. INTRODUCIÓN A LA API DE WINDOWS (III PARTE)

En este capítulo, nos iniciaremos en el manejo de los procesos en Windows.

Primero describiremos brevemente el funcionamiento de un computador actual a nivel de hardware, y luego sabiendo lo que ocurre a este nivel, nos abstraeremos a un nivel superior, desde el punto de vista del Sistema Operativo.

Para empezar debemos saber exactamente qué es un proceso. Un proceso no es más que un programa en ejecución, y un programa es una secuencia de instrucciones que es capaz de ejecutar el computador por sí solas, de forma automática. Los programas se almacenan en memoria, de forma que el procesador (CPU) toma la primera instrucción de la memoria y la ejecuta, luego la siguiente y así sucesivamente hasta finalizar el algoritmo del programa. Evidentemente un sistema será más rentable cuando el tiempo de ejecución de un programa (desde que se inicia la primera instrucción hasta que finaliza la última) sea lo más pequeño posible. Para ello existen técnicas de optimización tanto hardware como software, por ejemplo a nivel hardware existen técnicas mediante las cuales el procesador es capaz de ejecutar varias instrucciones a la vez, (procesadores superescalares), que sea capaz de procesar múltiples datos a la vez (procesadores vectoriales, presentes sobre todo en sistemas para el procesamiento de imágenes, tarjetas gráficas, etc...), que el procesador sea capaz de ejecutar el mayor número de instrucciones en el tiempo (tenga mayor frecuencia de trabajo), que el procesador sea capaz de dividir las instrucciones en etapas de manera que en cada etapa se puede ejecutar varias etapas distintas de distintas instrucciones, sin que ello afecte a la dependencia de datos, técnica conocida como segmentación o pipeline, y un largo etc. Y técnicas software principalmente hay que destacar que el algoritmo del programa sea lo más eficiente posible, la reordenación por parte del compilador, de las instrucciones del programa en memoria para que no haya dependencias de datos, y muchas más. Pero todas estas técnicas de optimización no son tema de este capítulo.

En un computador, existen diferentes jerarquías de memoria, (registros del procesador, cachés, memoria principal o RAM, discos, etc...),. Esto es así para aumentar las prestaciones y la rentabilidad del sistema, ya que no sólo depende de cuantas instrucciones es capaz de ejecutar un procesador a la vez, o en cuantas etapas, sino, que también afecta el hecho de que el procesador tiene que ir a buscar a la memoria la instrucción, traérsela, y ejecutarla. Este tiempo de acceso a memoria, será más pequeño cuando se logre que desde que el procesador solicita una instrucción o un dato, hasta que se le es devuelto por la memoria, sea lo más pequeño posible, la latencia sea mínima (actualmente del orden de 4 nanosegundos). Por ello por ejemplo una memoria caché actual tiene menor tiempo de acceso que una SDRAM DDR2.

Dicho esto veamos de forma general y muy resumida, cómo funciona un computador.

El programa se encuentra en el disco. Se hace una llamada al sistema para que lo cargue en memoria principal (RAM, es decir un nivel inferior en la jerarquía de memoria) y comience a ejecutarlo. Cargarlo en memoria significa reservar espacio en ésta para ubicar la secuencia de instrucciones del programa, y reservar otro espacio distinto en la memoria para almacenar los datos que se vayan necesitando a lo largo de la ejecución de éste. El sistema operativo se encarga de realizar esta tarea de carga, para ello debe indicarle al procesador dónde se encuentra la primera instrucción a ejecutar, y a partir de dónde se guardarán los datos que se vayan necesitando. Para ello graba en un registro del procesador (registro **CS**, **segmento de código** para arquitecturas x86), la dirección dónde comienzan las instrucciones, y en otro registro (**DS**, **segmento de datos** para arquitecturas x86), para indicar a partir de donde se almacenarán los datos que se necesiten (por ejemplo las variables del programa, y todas las estructuras de datos que utilice durante la

ejecución), así como la pila de ejecución, (registro **SS**, **segmento de pila**), para indicar a partir de qué dirección se van a ir guardando los datos auxiliares y los parámetros de las funciones locales, como una pila de platos). Por cada proceso existe una pila, pero por cada llamada a una función existe un fragmento dentro de esa pila, denominado **StackFrame**, es decir, la pila en el estado actual de ejecución de dicha función. El acceso a dicho StackFrame siempre vendrá referenciado por un puntero almacenado en un registro del procesador (**SP, puntero de pila**), que apunta hacia el último elemento introducido en la pila. En arquitecturas x86, la pila no crece hacia arriba, como hemos indicado con el ejemplo de "pila de platos", sino que lo hace hacia abajo, hacia direcciones mas bajas.

Una vez cargado en memoria, el procesador accede a la caché (que suponemos vacía al principio, ya que el programa recordemos que está en la RAM), mediante la dirección virtual que genera, evidentemente ahí no están los datos ni las instrucciones, asi que se produce un fallo de acceso. Debe entonces acceder, traduciendo la dirección virtual a física, a la memoria principal, pero al ser ésta más lenta que la caché, se debe esperar cierto tiempo para que se logre traer los datos correctamente (Se produce una penalización de tiempo por no encontrarse en la caché). Evidentemente se puede pensar que ésto puede pasar con las siguientes instrucciones y retrasar por tanto la ejecución del programa, por ello, ya que se accede a la RAM, en vez de traer solamente el dato o la instrucción solicitada, se trae los consecutivos, ya que probablemente la siguiente instrucción a acceder sea la consecutiva a la anterior, es decir se trae un bloque completo de instrucciones y las almacena en la caché, de forma que la siguiente vez que se vaya acceder, la instrucción ahora si se encuentra en la caché y no habrá que esperar (no habrá penalización de tiempo). Reducir la penalización de tiempo debida al acceso a la memoria principal se consigue aumentando las prestaciones de la RAM. *Existen técnicas que permiten acceder en modo ráfaga (como pueden ser las antiguas BEDO), aunque ya en desuso, hoy en día existen las DRAM Síncronas que trabajan bajo una señal de reloj, como pueden ser las actuales DDR o DDR2. (Que transfieren el doble y el cuádruple de datos respectivamente en el mismo periodo de tiempo).*

Pero, ***¿qué ocurre si la caché ya está llena y no caben más bloques que se necesitan?*** Pues existe la política de reemplazo en la caché. Viene implementada a nivel de hardware, y existen 4 técnicas para el reemplazo, siendo las más utilizadas las **aleatorias** y la denominada **LRU**. La primera sustituye el bloque que se trae de memoria por un bloque aleatorio de la caché ya ocupado. Y la segunda sustituye aquel bloque que menos se ha usado recientemente. Otras políticas, que prácticamente no se usan son la **FIFO** (se sustituye el primer bloque que entró en la caché) y la **LFU** (se sustituye el bloque que menos veces ha sido accedido por el procesador)

Este proceso se sucede continuamente hasta finalizar el algoritmo completo del programa.

Visto de forma muy genérica y simplificada cómo funciona un computador, ahora nos centraremos desde el punto de vista del Sistema Operativo, es decir sin importarnos cómo está implementado el hardware, ni si el procesador es segmentado, superescalar, o si funciona a más de 4GHz, o si la memoria tiene un tiempo de acceso pequeño, o las instrucciones se encuentran o no en caché, o reordenadas de la mejor forma posible en memoria.

Windows, como todo sistema operativo provee de unas llamadas al sistema que hacen cargar o descargar un proceso de la memoria. Al estar tratando con sistemas multitareas, es decir es posible realizar más de una tarea al mismo tiempo, (los procesos presentes se cargan en memoria y cada proceso tiene un espacio distinto en memoria identificado por su segmento de datos y segmento de código distinto) no sólo carga el código en memoria, sino que se crean las estructuras de datos necesarias para poder gestionar todos los procesos presentes de la mejor forma posible, ya que el procesador sólo puede ejecutar una instrucción en un instante de tiempo (a excepción de los superescalares). Estas estructuras de datos son **la cola de procesos y**

el PCB (Process Control Block, es decir, bloque de control de procesos). Entre otros datos, en el PCB se almacena un identificador de proceso (**PID**, que es único para cada proceso), **usuario** al que pertenece, tiempo que lleva en ejecución, **prioridades**, etc...

Por tanto el sistema operativo debe repartir el tiempo de uso del procesador entre cada proceso, es decir, decide si durante 2 segundos el proceso A va a ocupar el procesador, durante el segundo siguiente va a ocuparlo el proceso B, y así sucesivamente. O bien si existen prioridades, decidir que proceso tiene mayor prioridad para ocupar el procesador. Al cambiar de proceso, se cambia todo el contexto, se escriben los registros CS, DS, SS, IP, etc, referentes al proceso a ejecutar por el procesador. Todo ésto se conoce como **planificación de procesos**. Existen muchas técnicas para implementar el **planificador de procesos** de un Sistema Operativo. Uno de los más conocidos es el "***Round Robin***", aunque no tiene en cuenta las prioridades, reparte equitativamente los tiempos de ocupación del procesador, dando mayor sensación de paralelismo. Es decir, se decide que un proceso no puede estar más de un periodo de tiempo ocupando el procesador, y si lo supera, se suspende y se le cede el turno al siguiente proceso de la cola, y así sucesivamente. El proceso que se suspendió al principio volverá a ocupar el procesador cuando todos los existentes en la cola de procesos hayan hecho uso ya de él.

Probablemente, te estés preguntando que si el sistema es multitarea, ***¿qué puede ocurrir si todos los procesos existentes no caben en la memoria principal?*** Pues efectivamente, hoy día la mayoría de los sistemas actuales suelen correr alrededor de unos 15 procesos como mínimo, cada uno de ellos con su espacio de memoria. Evidentemente si se ejecutan muchos más o simplemente uno sólo que ocupe mucha memoria, no dejaría espacio para seguir ubicando más procesos. Por tanto existe un mecanismo de virtualización que consiste en descargar un proceso de la memoria, ubicarlo temporalmente en el disco duro (que si es de mucha más capacidad que la memoria principal), dejando un hueco en la memoria principal (RAM), para ubicar otro proceso. Posteriormente el proceso descargado se vuelve a cargar en la memoria para continuar con su ejecución, por donde lo dejó. Este proceso de carga y descarga se sucede continuamente cuando existen muchos procesos en ejecución, cosa que ocurre muy a menudo. Evidentemente el tiempo de descargar, guardar en disco, leer del disco y volver a cargar es algo que consume bastante tiempo. El acceso al disco es muchísimo más lento que el acceso a la RAM, principalmente porque éste es mecánico y la memoria es eléctrica.

Este método de guardar temporalmente en el disco y volver a cargar, es decir de darle la cualidad al sistema de poder direccionar todo el espacio de memoria direccionable por el procesador, que depende del bus de direcciones de ése (si por ejemplo es de 32 lineas de dirección, pues direccionar hasta 4GB), se conoce como **Memoria Virtual**. Los procesos descargados se almacenan en una zona reservada en el disco y las transferencias necesarias entre memoria principal y el disco se realizan mediante un fichero especial denominado "**Fichero de paginación**" (**PageFile.sys** en Windows)

Entendiendo un poco la mecánica de funcionamiento de un computador actual y la misión del Sistema Operativo:

¿Existe algún mecanismo que me permita a mi personalmente ubicar un proceso en memoria? Evidentemente si. Cuando haces doble clic sobre un programa que se encuentra almacenado en el disco. O bien desde tu aplicación haciendo las llamadas correspondientes al sistema. (función **OpenProcess** de Window)

¿Puede un proceso invadir el espacio de direcciones de memoria de otro proceso?

Depende de la robustez del sistema operativo y de cómo gestione la memoria. En sistemas Windows ésto es posible mediante unas técnicas de inyección de código. (**DLL injection**). Usada frecuentemente por los virus y malwares actuales para saltarse las protecciones de los antivirus. Existen

varias formas de llevar a cabo esta técnica, las más frecuentes son creando hilos de ejecución remotos en otros procesos (**CreateRemoteThread**), y usando ganchos (**SetWindowsHookEx**).

A partir de Windows95/98, ya se incorpora un mecanismo de protección basado en anillos. Crear dos espacios virtuales distintos para procesos del sistema operativo y procesos de usuario. El primero, también conocido como "**modo kernel**", es el más privilegiado y puede acceder a cualquier espacio de direcciones de otro proceso, ya sea del modo kernel o de usuario. El segundo sólo permite acceder a espacios de direcciones en "**modo usuario**". Es decir un programa de usuario no puede acceder al modo kernel, pero si viceversa.

Para un procesador de 32 bits de dirección, en el que se pueden direccionar hasta 4GB de memoria virtual. El sistema operativo reserva 2GB para procesos de usuario (modo usuario o anillo3 "**ring3**"), y otros 2GB para procesos del sistema (modo kernel, o anillo0, "**ring0**").

Por ello, es por lo que actualmente, por ejemplo en un Windows2000 o XP, si estando en modo usuario se quiere realizar por ejemplo un programa para mandar información al puerto paralelo del sistema, o si se intenta cerrar un proceso del sistema, no se podrá. El mecanismo que permite realizar dicha acción es el "**driver**". El driver permite a un usuario acceder al modo kernel, para gestionar un dispositivo o un proceso en ring0, haciendo las respetivas llamadas al sistema y tratándolo como un fichero, por tanto el driver se crea, se abre, se llama, se cierra, usando las funciones de descriptores de fichero: **CreateFile( ), ReadFile( ), WriteFile( )**, etc...

La API de Windows, le pasa una estructura IRP al driver que es un paquete que contiene la información sobre las rutinas a ejecutar, por ejemplo: **Read(Klrp IRP);**

Por tanto un virus por ejemplo, para poder acceder a un dispositivo o un proceso del kernel, debe instalar un driver que le permita, mediante las llamadas al sistema (API), acceder a dichos recursos de ring0. Sin embargo un virus que simplemente quiere saltarse el antivirus o hacerse residente en memoria, sólo tiene que hacer las llamadas al sistema correspondientes para realizar dicha acción (DLL injection).

Por ejemplo una técnica bastante sencilla de inyección DLL consistiría en crear una DLL con el código a inyectar (el que queremos que se ejecute en el espacio de direcciones de otros proceso de usuario), y simplemente cargándola en aquellos procesos que usen **user32.dll** sería posible hacer prácticamente un virus "inmortal".

Windows permite realizar esta acción simplemente añadiendo la ruta de la DLL con el código a inyectar en la siguiente ruta del registro de windows:

```
HKEY_LOCAL_MACHINE\Software\Microsoft\
WindowsNT\CurrentVersion\Windows
```

Todos las rutas de DLL que se encuentren en la clave **AppInit_DLL**, serán cargadas en los procesos que usen **user32.dll**, es decir, carguen este módulo mediante una llamada al sistema: **LoadLibrary(),** que prácticamente serán todos los procesos de usuario.

Si comprobais dicha ruta observad que no haya ninguna ruta a una DLL sospechosa, puesto que si la hay, empezaría por pasar algún tipo de antivirus, ya que es bastante utilizado, sobre todo para Spywares.

El tema de DLL injection a nivel de programación lo comentaremos en otros artículos.

Muy por encima, conocemos a groso modo el funcionamiento tanto a nivel hardware como software de una "maquina que ejecuta programas simultáneamente por sí sola". Pero probablemente quieras gestionar procesos de usuario desde Windows, bajo tu aplicación. Esto es posible gracias a la API de Windows que nos facilita la labor para realizar las llamadas necesarias al sistema.

Hasta ahora hemos supuesto que los procesos en un sistema multitarea son independientes, pero puede ocurrir que exista dependencia entre ellos, o bien que un proceso existente lance otro proceso heredando sus atributos (proceso **padre** y proceso **hijo**), siendo el hijo una copia exacta del padre, eso sí cada uno de ellos tendrá su propio identificador (distinto PID), y espacios de memoria distintos. Ejemplos de ésto lo podemos encontrar en los procesos **daemons** (demonios) de Linux (como por ejemplo un servidor web funcionando, "Apache" por ejemplo), o los conocidos **svchost** de Windows. Esta técnica de "bifurcación", la podemos implementar, en vez de con varios procesos, mediante **hilos de ejecución** distintos (**Threads**), consiguiendo así que un proceso realice una tarea, como si fueran varios procesos distintos en ejecución trabajando en la misma tarea.

Para aclarar este importante concepto, pongamos un ejemplo: Supongamos que queremos realizar un programa para que encuentre los números primos del 1 al 1000000. Si lo realizamos mediante un algoritmo lineal, es decir, comprueba el 1, mira si es primo, y continua con el 2, y así sucesivamente hasta el 1000000... probablemente observemos una sensación de bloqueo en el sistema, debida a que el planifiador del Sistema Operativo conmuta entre nuestro proceso y los demás, consumiendo prácticamente el 100% de uso del procesador. Si ahora creamos por ejemplo un hilo de ejecución en nuestro programa para calcular los números primos, el resultado visto desde el punto de vista del Sistema Operativo serán dos hilos de ejecución diferentes, uno para el cálculo y otro el que ya existía en nuestro programa. Evidentemente el hilo encargado de realizar los cálculos consumirá recursos del sistema, pero el perteneciente a nuestra aplicación no se verá afectado (habrá desaparecido la sensación de bloqueo). No obstante creando un segundo hilo de ejecución no mejoramos, para este ejemplo, el rendimiento. Pero, si creamos varios hilos de ejecución de manera que cada hilo, que se comporta como un proceso distinto, realice una parte del cálculo, entonces habremos dividido el problema en subproblemas de menor tamaño, optimizando por tanto la velocidad.

Esta técnica de multihilos, se utiliza frecuentemente en aplicaciones gráficas (mientras un hilo se encarga de realizar los cálculos, el otro se encarga de atender las peticiones del usuario en la interfaz gráfica), y también es muy usada en aplicaciones web, sobre todo en Servidores.

A parte de ésto, comentar que por ejemplo no sólo se gestionan hilos a nivel del Sistema Operativo, sino también a nivel de procesador. Ejemplo de ello lo podemos encontrar en la familia de los Intel, en procesadores actuales, conocida como **Hyper-Threading**.

La forma en que el sistema permite comunicar dos procesos distintos, ya sean hilos, o varios procesos padre e hijo, es mediante tuberías (pipe "|"). De manera que se ejecutan estos procesos en el sistema, pero es posible redireccionar la salida estandar de uno a la entrada estandar de otro. Supongo que más de una vez lo habréis usado... para el caso de Windows por ejemplo podemos hacer lo siguiente desde una consola:

```
C\:>DIR|more => la salida estandar del comando
DIR, la introducimos en la entrada estandar de
MORE. (Dos procesos que se han comunicado entre
sí).
```

Para el caso de Linux, podemos poner por ejemplo:

```
$ ls-l /etc | grep "[0-9]"
```

Esto por ejemplo nos mostraría los archivos y directorios del directorio /etc que contienen al menos un dígito. Tenemos el mismo caso, dos procesos distintos que se ejecutan y la salida de uno se le pasa como entrada a otro.

Otra manera de comunicar dos procesos en ejecución consiste en el paso de mensajes.

En el capitulo anterior, vimos como manipular una ventana de otra aplicación a través de la nuestra, mediante el **paso de mensajes** entre aplicaciones.

Dentro del conjunto de los procesos que existen en Windows, podríamos hacer un subconjunto de aquellos procesos que se gestionan mediante ventanas. Son los más conocidos por los usuarios en entornos gráficos. (Aplicaciones de escritorio). En las que comúnmente existe un botón en la parte superior para poder cerrar todo el proceso, es decir cerrar la aplicación (descargar el proceso de memoria).

Recordar que en Windows, todo se controla y se gestiona mediante un puntero al objeto. Es decir, un Handle o manejador que nos permite manipularlo como queramos. Pues bien, es posible lanzar o cerrar un proceso, es decir cargarlo o descargarlo de la memoria, solicitar información sobre el proceso, etc... a partir del Handle de éste.

Las funciones de la API WIN32 más importantes, entre otras, para la gestión de procesos son: **CreateProcess(), OpenProcess(), CloseHandle().**

Estas funciones las trataremos desde el principio en el siguiente capítulo, ya que debido a su extensión (muchas estructuras de datos, bastantes parámetros, etc...), no queda espacio suficiente en este artículo para ello.

Espero haber aclarado los conceptos previos sobre procesos antes de iniciarnos a la escritura de código.

Samir Sabbagh Sequera (HySTD)

CONFIGURAR LA RED EN WINDOWS 2000/ XP POR CONSOLA:

Como muchos de vosotros sabéis, hay varías versiones de Windows XP, entre ellas, la Home, Profesional, TablectPC, MediaCenter... la base de todas es la misma, pero, de cara al usuario hay grandes diferencias.

Una de esas diferencia es la que provoco este artículo, la versión Home viene de serie capada para que solo puedas tener una configuración de red por tarjeta de red, lo que en cristiano significa que si eres de los que anda con el portátil de aquí para allá, pues a cada sitio que llegues y quieras conectarte en red, o en todas tienes la misma dirección, puerta de enlace… o tienes que cambiar manualmente tu la configuración...

MOTIVO DEL TEXTO

Bien, un conocido mió que está estudiando en la Universidad, resulta que se acaba de encontrar con esta contrariedad en su nuevo portátil, pues estos días se va a estudiar a la Universidad y no vuelve hasta la noche (está a vueltas con los exámenes), pero como le gusta aprovechar el tiempo (y la superconexión a Internet que tiene la Biblioteca de la Universidad) pues se lleva el portátil y utiliza la conexión de uno de los PCs de la Biblioteca (hay una zona Wifi, pero está algo capadilla)... y ahí está el problema, claro cuando vuelve a casa pues tiene que reprogramar la conexión a red para seguir con el trabajo en casa por la noche, ya se sabe el eMule no duerme ;-)

Debido a lo anterior, me llamo pidiéndome ayuda para no tener que desperdiciar su valioso tiempo de estudio en andar configurando la red cada vez que llega a un nuevo sitio, y bueno, decidí ayudarle...

Posibles soluciones

Bien, hay varías soluciones a su problema, la primera es que se baje alguno de los programas que con solo pulsar un botón te cambia la configuración de red... hay varios, aunque todos son similares. Otra opción es utilizar el Windows XP Pro (si no quieres salirte de Windows por supuesto)... Pero, hay otras, especialmente una que es en la que voy a centrar este pequeños apuntes... utilizar NETSH... porque algunas limitaciones solo son tal si no sabes utilizar las herramientas adecuadas, ya que en las utilidades del SO hay limitaciones que no existen si te bajas a la línea de comandos.

Bueno, también había la opción de configurar en su casa la red con unos parámetros similares a los utilizados en la Universidad, pero esa sería una solución válida para esta ocasión, normalmente no es un solución viable.

NETSH

Y, ¿que coño es NETSH? pues es el mini shell para configurar los servicios DHCP, servicios de red y enrutamiento y el servio RAS, entre otras cosas... por ahí veréis que muchos se refieren a el como NetWork Shell o su abreviatura NetShell, y es que eso es exactamente el NETSH, es el equivalente al CMD pero referido a la red... y es en el que se basan esos programas comentados antes... porque con el NETSH se puede cambiar una dirección IP, la puerta de enlace, agregar más direcciones de red a un mismo dispositivo de red...

Pero comencemos con el NETSH:

Para comenzar conozcamos nuestra configuración de red, para eso utilicemos el comando IPCONFIG

```
C:\>ipconfig

Configuración ID de Windows 2000

Ethernet adaptador Ethernet00:

Sufijo DNS específico de conexión:
Dirección IP.............:192.168.0.10
Máscara de subred..........:255.255.255.0
```

NOTA: Como muchos os habréis dado cuenta, los ejemplos los hago con Windows 2000, pero tranquilos, el XP no es más que una actualización menor del Windows 2000 (versión NT 5.1 frente a NT 5) con mejores capacidades para Multimedia y Juegos. Yo, aunque estoy probando el Windows Vista, sigo confiando en el Windows2000 para la mayor parte de mis horas de trabajo.

Antes de mostraros las ordenes NetSH, pues no sería mala idea de que guardarais la configuración de red que tengáis en este momento, para que así, si sois de los que no se conforman con leer, pues no tengáis nada de que preocuparos, ya que tenéis asegurado volver a la configuración tal como la tenéis en este momento. Primero os pongo la orden para guardar en un archivo la configuración actual. Después la necesaria para volver a la anterior configuración utilizando el archivo de salvaguarda que creasteis con la orden anterior:

```
netsh dump > configred.dmp

netsh exec configred.dmp
```

NOTA: No hace falta que lleve la extensión «dmp», pero es la que suele usar MS en la documentación técnica y yo suelo usarla. En lugar de «configreg» podéis poner el nombre que queráis.

Aparte de para guardar/recuperar una conexión de red, esas simples ordenes tienen muchos otros usos, desde configurar un equipo y usar esa configuración para el resto de los equipos de la red (solo habría que cambiar ligeramente la dirección de red, algo más rápido que repetir la configuración en muchos equipos), aprovechar algún fallido de seguridad del algún que otro Windows… pero esa es otra historia. Por cierto, los programitas comentados en "posibles soluciones" suelen basarse justamente en está pareja de ordenes…

Pero pongámonos manos a la obra. Ahora que ya tenemos los datos actuales de nuestra conexión, y a salvo nuestra actual configuración, pues vamos jugar con el NETSH. Para empezar, cambiar toda la configuración:

```
C:\>netsh
netsh>interface
interface>ip
interface ip>set address "ethernet00" static
10.100.0.10 255.0.0.0 10.0.0.1 1
Command Succesfully
netsh>quit
```

NOTA: El mensaje Command Succesfully puede cambiar, incluso en el mismo SO, por efecto de ServicePacks, actualizaciones..., en algunos te pregunta si quieres proceder (o Aceptar), en otros ... pero bueno en caso de que te pregunte le dices que si o Aceptar y listo.

NOTA2: el "1" final es obligatorio si se producen cambios en la puerta de enlace... sin el no funciona, tiene un porqué, pero escapa al motivo de este mini artículo.

Haced un IPConfig y veréis:

```
C:\>ipconfig

Configuración ID de Windows 2000

Ethernet adaptador Ethernet00:

Sufijo DNS específico de conexión:
Dirección IP.............:10.100.0.10
Máscara de subred.........:255.0.0.0
Puerta de enlace predeterminada...:10.0.0.1
```

Pero, seguramente, muchos de vosotros me diréis, "joder eso ya lo hago yo y sin tener que saberme ni un comando", es cierto, pero es que con NETSH se puede hacer muchas otras cosas, darme unas pocas líneas más y os lo demostrare...

Procedamos a agregar una dirección de red completa a ese adaptador (motivo del artículo), con su puerta de enlace y todo. Es muy similar al anterior, básicamente la diferencia radica que en lugar de asignar una configuración (con SET) se agregan nuevas configuraciones (con ADD), y además ahora no se utiliza el parámetro modificador static.

```
C:\>netsh
netsh>interface
interface>ip
interface ip>add address “ethernet00” 192.168.0.10
255.255.255.0 192.168.0.1 1
Command Succesfully
netsh>quit
```

Ahora hacer un nuevo IPCONFIG y veréis la diferencia:

```
C:\>ipconfig

Configuración ID de Windows 2000

Ethernet adaptador Ethernet00:

Sufijo DNS específico de conexión:
Dirección IP.............:10.100.0.10
Máscara de subred.........:255.0.0.0
Dirección IP.............:192.168.0.10
Máscara de subred.........:255.255.255.0
Puerta de enlace predeterminada...:10.0.0.1
```

También se podía haber agregado una segunda dirección de red, con su respectiva mascara de red pero sin puerta de enlace, en ese caso, la orden ADD sería:

```
add address “ethernet00” 192.168.0.10
255.255.255.0
```

Al igual que se pueden agregar nuevas direcciones, también se pueden borrar:

```
delete address “ethernet00” 10.100.0.10
```

Hacer el correspondiente IPConfig de turno y veréis:

```
C:\>ipconfig

Configuración ID de Windows 2000

Ethernet adaptador Ethernet00:

Sufijo DNS específico de conexión:
Dirección IP.............:192.168.0.10
Máscara de subred.........:255.255.255.0
Puerta de enlace predeterminada...:10.0.0.1
```

Pero claro borraste la dirección, pero no la puerta de enlace correspondiente a esa dirección, con lo cual ahora os presento la orden que además de la dirección borra la puerta de enlace:

```
delete address “ethernet00” 10.100.0.10
10.0.0.1
```

Si hacéis el IPCONFIG después de la orden anterior en lugar de la mostrada unas líneas más arriba os encontrarías con:

```
C:\>ipconfig

Configuración ID de Windows 2000

Ethernet adaptador Ethernet00:

Sufijo DNS específico de conexión:
Dirección IP.............:192.168.0.10
Máscara de subred.........:255.255.255.0
Puerta de enlace predeterminada...:192.168.0.1
```

Aunque no lo mostraba, el 192.168.0.1 estaba presente, aunque la predefinida era la otra, y ahí está el truco... y es que pasaba como en los Windows 9x, en donde podías tener muchísimas puertas de enlace, pero cuando preguntabas al sistema solo te decía la primera, aunque las otras también estaban activas.

Por cierto, la mayor parte de estas ordenes, aunque yo las lance en varías líneas, pues pueden lanzarse en una sola línea. Ejemplo:

```
netsh interface ip set address “ethernet00” static 192.168.0.10 255.255.255.0 192.168.0.1 1
```

Además de poder cambiar y agregar direcciones de red, pues NETSH nos permite muchísimas más cosas, realmente sirve para configurar todos los parámetros de red. Entre otras cosas, permite cambiar DNS, WINS… os pongo un ejemplo de agregar una DNS y una dirección WINS:

```
netsh interface ip set dns “ethernet00” static 88.42.202.56
netsh interface ip set wins “ ethernet00” static 192.168.0.46
```

Y alguno dirá “coño todo cojonudo pero, ¿y que pasa cuando hay que utilizar DHCP?”, pues NetSH también puede trabajar con DHCP. Las órdenes son iguales que las anteriores, pero en lugar de utilizar como parámetros direcciones, se utiliza el parámetro «dhcp», con lo que las órdenes quedarían de la siguiente forma:

```
netsh interface ip set address “ethernet00” dhcp
netsh interface ip set dns “ethernet00” dhcp
netsh interface ip set wins “ethernet00” dhcp
```

Llegados a este punto, solo haré un pequeño inciso más, yo a lo largo de todo el texto utilice “ethernet00” porque yo tengo más de una tarjeta de red. Cuando solo se tiene una tarjeta de red, lo que se suele usar es “Conexi¢n de Area local” (con Windows en español) o “Local Area Connection” (con Windows en inglés).

NETSH es esto y mucho más, tal vez otro día os muestre alguna característica más..., pero eso otro día, ya que lo aquí mostrado no es más que la punta del iceberg.

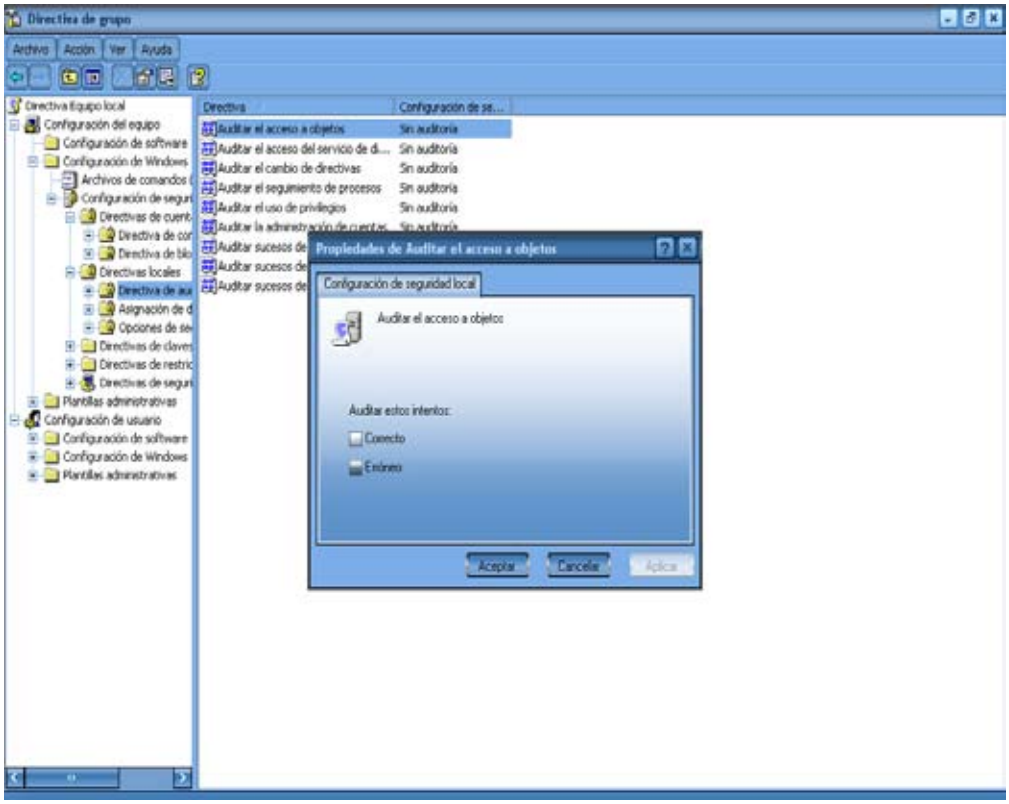
Un Saludo
iberhack@gmail.com

Iberhack

HACKERS, UNA CULTURA

Cuando mi amigo clarinetista me pidió que escribiera este articulo, no me dijo sobre que quería que escribiese. Leí las anteriores ezines en busca de una inspiración y tengo que reconocer que no se me ocurrió nada original e innovador. Tal vez tratar sobre las nuevas formas de ataque a redes basadas en encriptación o hablar de cómo evolucionan las formas de malware o la eminente profesionalización de técnicas de spammers.

Mi vida gira en torno a la informática. La gente no sabe que soy un hacker, no saben que soy experto en sistemas y redes y que he ayudado a empresas y particulares en problemas más o menos importantes en su lucha cotidiana contra el destrozo informático.



Los hackers estamos muy mal vistos en la sociedad. He leído mucho en foros de otras temáticas de cómo los admin se quejaban de que los hackers les destrozaban sus sites, como ellos decían como si hicieran grafitis. He tenido que fingir que no se de que me hablan y obviar el tema de su falsa sensación de estar bajo el uso de la verdad absoluta. También estoy harto de ser blanco de descalificaciones por medios de comunicación medio analfabetos que no sabrían diferenciar, el spam del spoofing. Sin embargo queridos lectores, esta época ya ha tenido lugar. Cualquiera que lea la historia del hacktivismo se dará cuenta que llueve sobre mojado.

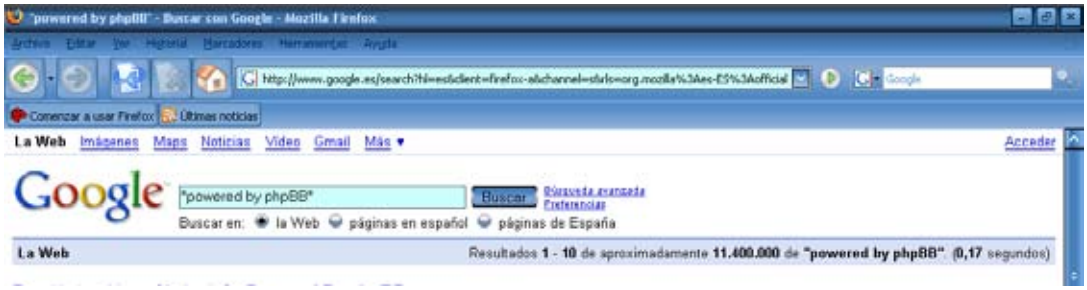
El problema ahora es que la difusión y expansión de la red de redes los exploits y las técnicas antes reservadas a auténticos hacker y gurús de la seguridad están al alcance de cualquier incauto que quiere jugar a ser dios. La mayoría de estos intrépidos prueba-herramientas tienen una edad corta y simplemente buscan el reconocimiento de una cultura que simplemente los repudia. Entonces ellos se refugian en una subcultura de individuos de similares características en las que ellos mismo se regodean de sus pseudo-logros. Hubo un tiempo que los hackers los llamaron lammers, pero esta etiqueta ya esta en desuso. El motivo del desuso es que ellos mismos se etiquetaban a si mismos y a otros de nivel inferior. ¿Un loco que le dice loco a otro loco deja de serlo automáticamente?

Sin embargo no debemos llevarnos a engaño. Estos personajes no son unos machateclados a secas. Algunos tienen formación académica y conocimientos avanzados de programación en diversos lenguajes y saben moverse bajo diferentes sistemas operativos. Me encontré en el IRC algunos de ellos con conocimientos casi brillantes.

Bien, vimos hasta aquí las similitudes, pero la diferencia es básica y a la vez fundamental. Ellos carecen de ética. Recuerdo una vez, en mis comienzos que revele una vulnerabilidad en un foro a un webmaster novato. El pobre no supo repararla y casi pierde su foro en el intento y tuve que darle consejo

en numerosas ocasiones.

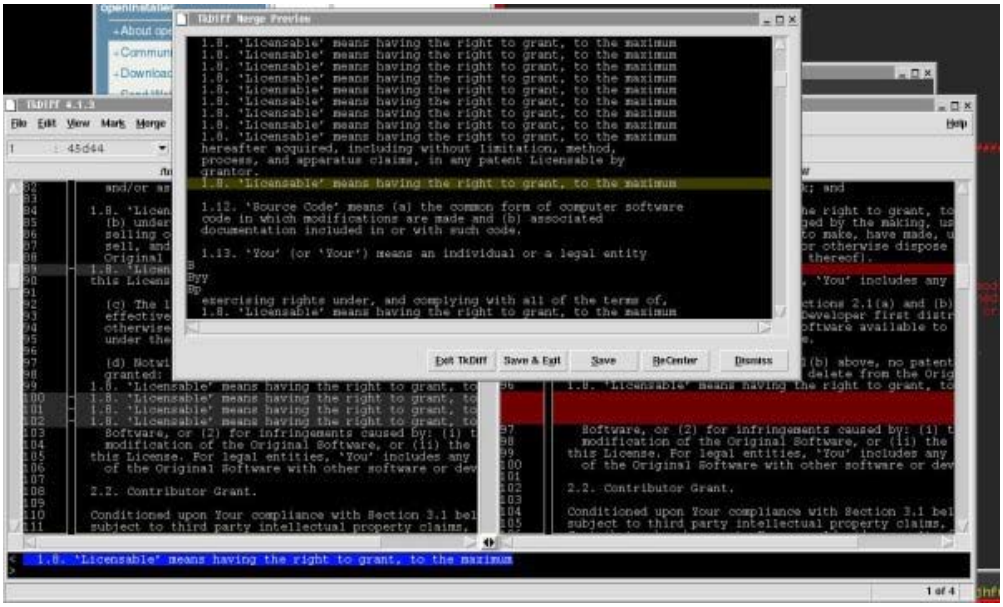
Ahora estos defaces como comúnmente son conocidos son totalmente indiscriminados. Estos símbolos de la protesta hacker se convirtieron en una forma de expresión y de lucha contra objetivos claramente marcados como pornografía infantil, violencia, odio, racismo, xenofobia . Ahora tan solo son búsquedas "powered by".



Pero no lleguemos a la conclusión errónea que quieren convencernos los que desconocen este mundo de culpar a toda una filosofía por la infamia de quien la desconoce. No culpare al budismo por el despropósito de un budista. No hare con esto una excepción.

Hace no demasiado, en un canal de IRC de hackers auténticos, me refiero a canales en los que se debate sobre filosofía del soft libre o si es bueno automatizar iptables en lugar de programarlo 100 % por citar dos ejemplos sencillos de comprender, se hablaba sobre el futuro de la filosofía hacker. El oscurantismo ya paso y estamos en una época de transición. Muchos hackers se convierten en auditores, empresarios y comerciales en general, abriendo las puertas a un mundo que nos odia porque no nos entiende. No entiende que gracias a gente así tenemos sistemas potentes seguros y fiables. Que tenemos acceso a un software gratuito y de calidad. También que nuestra privacidad es más controlada y que tenemos más medios para protegerla.

Yo no soy ningún gurú. Mis conocimientos no llegan a esos extremos, ya que no tengo un dominio absoluta de ninguna materia, ni de todas ellas. Pero si que soy hacker. Yo comulgo con la filosofía hacker, y además, trato de inculcarla a los que se interesan por ella. A pesar de que me siguen preguntando si se arreglar microondas y que mis colegas me preguntan si puedo encender la webcam de los vecinos con fines lujuriosos.



En mis horas de charlas con hackers noveles y amantes de la seguridad descubro que la magia no se ha perdido porque algunos magos descubran sus trucos. Siempre hay quien logra sorprenderte.

Dedicado a mis amigos de HackHispano y de CulturaHack.

Ronaldo

Staff:

Redactores:
- Cypress
- Samir Sabbagh Sequera (HySTD)
- Iberhack
- Clarinetista
- Ronaldo

Correción:
- Esk

Maquetación y Diseño:
- mystery-man

Dirección del Proyecto:
- Clarinetista