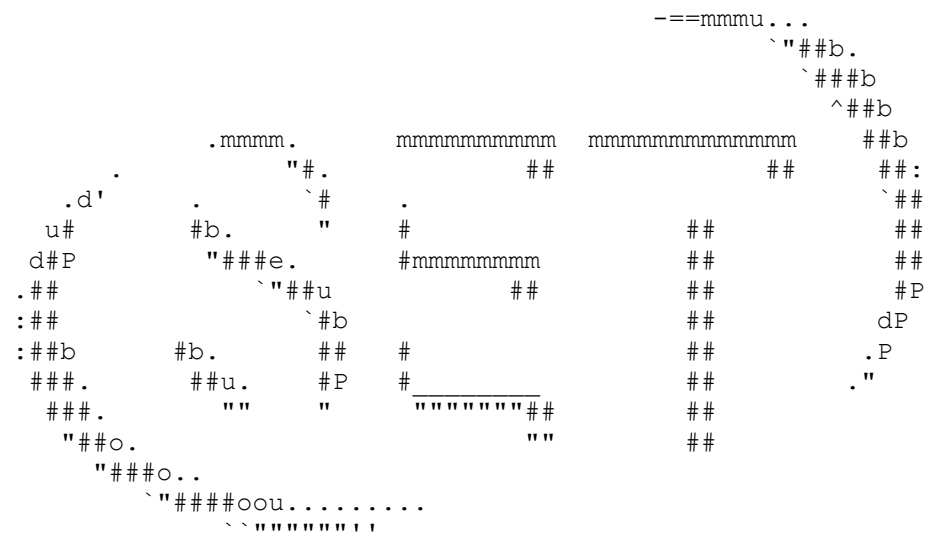


Saqueadores Edicion Tecnica

INFORMACION LIBRE PARA GENTE LIBRE

SET #35 - Octubre de 2008

"La caja musical sin cables no tiene ningún tipo de valor comercial. ¿Quién iba a pagar por un mensaje que no está siendo mandado a nadie en particular?"

Respuesta de David Sarnoff's Associates a la propuesta de invertir en radio.

[EDITORIAL]

SET Ezine

Disponible en:
http://www.set-ezine.org

Contacto:
<web@set-ezine.org>
<set-fw@bigfoot.com>

[AVISO]

[ADVERTENCIAS]

* La INFORMACION contenida en este ezine no refleja la opinion de nadie y se facilita con caracter de mero entretenimiento, todos los datos aqui presentes pueden ser erroneos, malintencionados, inexplicables o carentes de sentido.

El E-ZINE SET no se responsabiliza ni de la opinion ni de los contenidos de los articulos firmados y/o anonimos.

De aqui EN ADELANTE cualquier cosa que pase es responsabilidad

| vuestra. Protestas dirigirse a /dev/echo o al tlf. 806-666-000 |
|
| * La reproduccion de este ezine es LIBRE siempre que se respete la integridad del mismo. |
|
| * El E-ZINE SET se reserva el derecho de impresion y redistribucion de los materiales contenidos en este ezine de cualquier otro modo. Para cualquier informacion relacionada contactad con SET. |
|

-----[TABLA DE CONTENIDOS]----
----[SET 35]----

"El científico encuentra su recompensa en lo que Henri Poincare llama el placer de la comprensión, y no en las posibilidades de aplicación que cualquier descubrimiento pueda conllevar."

Albert Einstein

EOF

Phoenix e-zine definiria bastante bien el comportamiento que ultimamente ha adoptado esta revista electronica. Un numero sale a la luz y entonces todo parece morir. De pronto resurge de sus cenizas y tras una larga espera vuelve a la carga. Y el ciclo se repite incesantemente, una y otra vez...

Tranquil@s, estamos luchando por cambiar esta situacion, estamos renovando las fuerzas, buscando un equilibrio en el nivel y conservando el potencial con el que contamos.

Eres hacker y por eso estas aqui, buscas ampliar tu conocimiento con nuevas ideas subversivas y nuestra mision es proporcionartelas. Trabajamos por ti y para ti, pero todavia hay algo mas, ahora tu puedes formar parte de esto.

Llevas tiempo afirmando que si, que estas dispuesto a contribuir y pronto prepararas un articulo sobre algun tema underground. Llevas tiempo enganyandote a ti mism@ porque sabes que no lo haras; piensas que mientras alguien este dispuesto a hacer el trabajo por ti, el hacking seguira funcionando.

Haznos caso, si has descubierto una empresa vulnerable/xploiteable, si has crackeado aquella aplicacion cuyo algoritmo de cifrado tan seguro proclamaba ser, o si tu filosofia acerca del hacking posee nuevos conceptos que debamos sopesar, entonces no lo pienses dos veces, escribe tu experiencia y ayuda al mundo.

Miralo por otro lado, cuando tengas 10 anyos mas y una familia que mantener (si todavia no la tienes) e internet y la web sean unicamente una red social totalmente controlada por los devoradores de mentes, en ese instante podras decir:

- Mira hijo, antes las cosas las haciamos asi, y eramos los mejores.

- Los mejores?, por que erais los mejores?

- Ah claro! Tu no comprendes! Eramos hackers hijo, HACKERS.

Hasta el proximo numero,

El editor

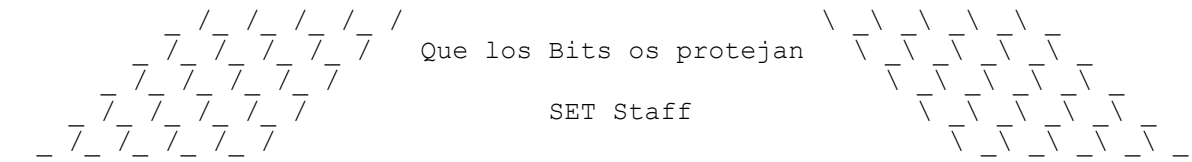
Comentario de madfran Y GRRL.

"El editor" no es el que, desde hace mucho, demasiado tiempo, escribía estas lineas. Cuando todo parecia terminar, cuando en el horizonte no se veia mas que una linea indefinida e inalcanzable, cuando parecia que lentamente las fuerzas se agotaban y tan solo quedaba sentarse frente al monitor esperando que este se extingiera por si solo, alguien ha pasado a nuestro lado y ha recogido el testigo. Estamos de nuevo en liza!

Este numero ha sido editado, maquetado y estructurado por un nuevo equipo de edicion. La aportacion de la vieja guardia tan solo ha sido el articulo sobre John The Ripper y el mantenimiento de la web que, entre parentesis, algunos dolores de cabeza da.

Ha sucedido lo que desde hace tiempo esperabamos. De la misma forma que unos rescoldos en el fondo de una hoguera de campo son capaces, inesperadamente, de provocar una llamarada brillante, nuestra set-ezine ha sido capaz de mantener el calor necesario para que alguien se acerque y se decida a empujar del carro.

Esperemos que este empujon tenga continuidad.



EOF

-[0x02]--
-[Crackear usando WINE]---
-[by FCA00000]---SET-35--

CRACKEAR usando...

```
==        == == ===  == =====
 \\      //  || ||\\ || |___
  \\ /\ //   || || \\|| |
   ==  ==    == ==  === ===== (wine)
```

```
-----------------------------------------------------------------------.-----------.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% by FCA00000
-----------------------------------------------------------------------·-----------·
```

Uno de los problemas que se encuentran los usuarios de Linux es que ocasionalmente necesitan usar algún programa disponible sólo para Windows. Aparte de usar 2 ordenadores o 2 particiones, la solución más cómoda es usar un programa que emule Windows. Esta solución existe desde hace tiempo y se llama "Wine".

```
&&&_____________________________&&&
||| ALGO Y MUCHO MÁS SOBRE WINE |||
&&&_____________________________&&&
```

Es un proyecto open-source y se puede obtener desde:

-> http://www.winehq.org

Instalarlo a partir de los fuentes es cosa de 10 minutos.

Wine actúa como una capa entre un programa de Windows y el sistema Linux. Por ejemplo, si el programa quiere dibujar un punto, la llamada Windows a la rutina de dibujar se intercepta y Wine invoca a su equivalente en X-Window. Otro ejemplo: si el programa abre un fichero, Wine llama a las correspondientes rutinas de Linux.

¿Qué pasa si no hay un equivalente? Pues se intenta apañar lo mejor posible.

Por ejemplo, el sistema de archivos: Windows permite que un fichero tenga 3 fechas: creación, modificación, y último acceso. Sin embargo Linux no usa la fecha de acceso. Para emularlo, guarda en un fichero interno, una lista con los ficheros, y su fecha de acceso. Cuando algún programa quiere saber la fecha de acceso, Wine la saca de este fichero interno.

Dado que Windows contiene muchas librerías con muchas funciones, traducir todas es una tarea enorme. Por eso Wine trabaja con 2 tipos de rutinas:

1 -> emuladas: alguien ha analizado la rutina original de Windows y la ha programado en Wine. Por ejemplo, abrir un fichero.

2 -> directa: se ejecuta 'tal cual', es decir, que llama a la rutina original. Esto tiene el inconveniente de que no se sabe exactamente lo que hace, por lo que puede que no funcione bien.

Vamos a ver un ejemplo. Supongamos un programa hecho para windows que consta del programa principal, y una librería que cifra datos.
Cuando lo ejecutamos en Wine, este programa llamará a:
-rutinas típicas de Windows: mostrar GUI, iniciar menús, pedir datos, obtener nombre de usuario... Todas estas rutinas han sido analizadas por la gente de Wine y están traducidas. Es decir, que las librerías originales han sido

reemplazadas por equivalentes en Wine.
-rutinas de la librería para cifrar datos. Como esta librería no es estándar de Windows, nadie se ha molestado en traducirla a Wine. Por tanto hay que ejecutarla directamente, sin interceptarla.

Vamos a ver un ejemplo de cómo se emulan las rutinas.

En windows existe una función llamada GetTempFileNameW que sirve para obtener un nombre único de fichero. Esto se usa cuando necesitas generar un fichero temporalmente. Cualquier programa puede invocar a esta función, que está incluida en la librería "kernel23.dll". El código fuente está en: "wine/dlls/kernel32/path.c" y se declara como:

```
o-------------------------------------------------------------o
|   UINT WINAPI GetTempFileNameW( LPCWSTR path, LPCWSTR prefix, |
|                                 UINT unique, LPWSTR buffer )  |
o-------------------------------------------------------------o
```

Es decir, que tiene 4 parámetros.

¿Cómo han sabido el nombre de la rutina, y el número de parámetros?

En este caso, es una rutina disponible para todo programador de Windows, por lo que su API (Application Programming Interface) está documentado en el SDK de Windows.

Los chicos de Wine no tienen acceso al código fuente de Windows, por lo que simplemente pueden imaginar cómo funciona internamente esta rutina. Así que lo han traducido a algo así:

1.- toma el nombre del directorio path
2.- si no acaba en '\' , añádela.
3.- si se ha pasado el argumento prefix , añádelo.
4.- toma un número aleatorio y conviértelo en un string. Añádelo.
5.- ahora tenemos un nombre de fichero.
6.- intenta abrir el fichero. Si ya existe, intenta otro número aleatorio hasta que funcione.

¿Es esto exactamente lo mismo que hace Windows en su código original?
Sólo lo pueden decir los programadores originales de Windows. Pero parece funcionar.

Esto es el mayor reto de la gente de Wine: quizás Windows hace algo parecido, pero no exactamente lo mismo.
Por otro lado, está la paradoja de que un bug de Windows quizás no exista en Wine, y viceversa.

Esto es lo que se hace para funciones emuladas, pero algunas no lo están; algunas por falta de tiempo, otras por falta de interés. ¿Para qué molestarse en emular una función que no se usa nunca?
Por ejemplo, la función InvalidateNLSCache de kernel23.dll , no-emulada en wine/dlls/kernel32/locale.c

Su código en Wine es simplemente:

```
o---------------------------------------o
|   BOOL WINAPI InvalidateNLSCache(void) |
|   {                                    |
|     FIXME("() stub\n");                |
|     return FALSE;                      |
|   }                                    |
o---------------------------------------o
```

O sea, que cuando un programa invoca a InvalidateNLSCache en Windows,

seguramente pasa algo.
Pero al ejecutarlo en Wine, no hace nada. Sólo muestra un error en la consola, y devuelve el valor False.

En el código fuente de Wine no se dan detalles de la razón de no emularla, pero seguramente es porque no afecta a la funcionalidad de Windows.

Cuando se compila Wine, hay unos ficheros que indican cuales librerías están completamente emuladas, cuales lo están parcialmente, y cuales son ejecutadas directamente.
Estos ficheros también indican cuales son las rutinas emuladas, y cuales no lo están.
Si una rutina no está emulada, lo normal es hacer que invoque a la original.
Esto se hace usando:

```
o-------------------------------------------------o
|   __wine_stub_InvalidateNLSCache(void) {        |
|      __wine_unimplemented("InvalidateNLSCache"); |
|   }                                             |
o-------------------------------------------------o
```

Por eso, Wine necesita que Windows esté instalado. Si un programa llama a una función emulada tal como GetTempFileNameW, llamará a la librería kernel32.dll de Wine.
Si no lo está, como en el caso de InvalidateNLSCache, entonces llamará a kernel32.dll original de Windows.

¿Cómo hacen los programadores de Wine para saber lo que hace Windows?

Lo primero es mirar la documentación del SDK de Windows. Luego hacen un programa que prueba la rutina una y otra vez, con distintos parámetros. Miran el resultado.
Entonces escriben el código para simularla. Prueban a ver si la rutina emulada actúa igual que la original.
Notar que no pueden desensamblar el código de Windows porque está prohibido. Tampoco pueden leer el código fuente original de Windows (en caso de que lo tengan) porque esto va en contra del copyright. Quizás a ti no te importe, pero si Microsoft encuentra parte de su código dentro de Wine, les puede meter en un gran apuro.

Después de esta larga introducción, vamos a ver cómo usarlo en nuestro provecho.
Mi objetivo es crackear un programa llamado VPOM1510-SB_1.0_Installer.exe , que es un emulador de teléfonos Symbian, pero esto no es importante.
Este programa tiene una limitación de 7 días de uso.

Así que tras compilar wine-1.1.4 e instalarlo, ejecuto:

```
$ wine VPOM1510-SB_1.0_Installer.exe
```

que me presenta el programa de instalación de VPOM1510-SB_1.0_Installer.exe
Notar que VPOM1510-SB_1.0_Installer.exe es un binario de Windows, lo que demuestra que Wine funciona. Tras unas cuantas pantallas de presentación, el programa se instala.

Al usar Linux, obviamente yo no tengo un disco C: pero Wine lo ha simulado en el directorio "$HOME/.wine/drive_c/" en particular, tengo:

```
$HOME/.wine/drive_c/Virtio/Softboards/VPOM1510-SB-SYMB/bin/vrelaunch.exe
```

O sea, que está instalado correctamente.

Al intentar ejecutarlo, me da el siguiente error:

```
o-------------------------------------------------------------------------o
```

```
|err:module:import_dll Library MFC42.DLL (which is needed by L"$HOME\\.wine\\ |
|drive_c\\Virtio\\Softboards\\VPOM1510-SB-SYMB\\bin\\vrelaunch.exe") not found|
o-------------------------------------------------------------------------------o
```

Esto es fácil de arreglar: busco "MFC42.DLL" en mi instalación de Windows y lo copio a:

```
   $HOME/.wine/drive_c/windows/system32/MFC42.DLL
```

lo intento de nuevo, y ahora arranca correctamente el programa vrelaunch.exe

Lo primero es que me presenta una pantalla para configurar VPOM1510-SB-SYMB pero esto no es lo importante. Una vez configurado intento ver cómo afecta la limitación de 7 días. Para ello adelanto la fecha de mi sistema y observo que VPOM1510-SB-SYMB dice que ha caducado, y sale. Cierro el programa, retraso la fecha, pero el programa detecta que ha expirado anteriormente.

No hay problema: borro $HOME/.wine/drive_c/ y arranco de nuevo el instalador:

```
   $wine VPOM1510-SB_1.0_Installer.exe
```

Una vez instalado, hago una copia de "$HOME/.wine/drive_c" y la llamo: "$HOME/.wine/drive_c_tras_instalacion".

```
o-_NOTA_----------------------------------------------------------------------o
|                                                                              |
| 1.- Esto es una de las cosas buenas de Wine. Me permitirá volver a una       |
|configuración anterior. Ya sé que hay herramientas de Windows que hacen       |
|lo mismo, pero es una cosa útil.                                              |
o------------------------------------------------------------------------------o
```

Así que arranco el programa instalado y cuando lo tengo configurado, hago otra copia en "$HOME/.wine/drive_c_tras_configuracion".
Comparo los directorios y veo que se han generado varios archivos nuevos. Uno es el de configuración, y otro se llama:

```
   "$HOME/.wine/drive_c/windows/system32/2222NPR11QW.oc"
```

Tras instalar el programa un par de veces en un entorno 'limpio', aprendo que este es un fichero que se crea en cuanto el programa se ejecuta por primera vez.

Si el programa expira y lo desinstalas, este fichero "2222NPR11QW.oc" no desaparece. Cuando re-instalo el programa, verifica este fichero, y sabe que anteriormente estaba instalado y caducado, por lo que rehúsa instalarse de nuevo. No es una protección muy sofisticada, pero algo es algo.

No sólo eso, sino que hay un archivo que ha cambiado, llamado:

```
   "$HOME/.wine/system.reg"
```

Este fichero es de tipo texto, así que lo cargo en mi editor favorito y veo que es el equivalente al registro de Windows. Esto dice mucho a favor de la gente de Wine. En el Windows original, el registro es un fichero binario accesible a través del API GetRegKey y similares. En Wine han traducido esta función en otra que hace lo mismo, usando un fichero de texto.

Así que puedo comparar los ficheros:

```
      $HOME/.wine/drive_c_tras_instalacion/system.reg
   y
      $HOME/.wine/drive_c_tras_configuracion/system.reg
```

Con esto veo que el instalador crea unas entradas en el registro de Windows, en

la clave: [Software\\Classes\\vdsp]

Al caducar o desinstalarse, estas entradas no desaparecen, por lo que una reinstalación posterior se queja de que había caducado anteriormente.

Por tanto ya sé en qué consiste la protección de re-instalación: un fichero y el registro. Así puedo instalarlo una y otra vez, sin más que borrar el fichero "2222NPR11QW.oc" y las entradas del registro.

```
&&&_____________________________&&&
||| A CRACKEAR SE HA DICHO |||
&&&_____________________________&&&
```

Ahora viene la parte de crackeo típico: en algún lugar debe de haber una verificación de fechas. Podría desensamblarlo con IDA o ejecutarlo paso a paso con OllyDBG, pero voy a usar el poder de Wine.

Según la documentación es posible poner un traceador usando la variable de entorno:

```
WINEDEBUG=warn+all
```

que pondrá el nivel máximo de traceado.

Así que ejecuto:

```
$env WINEDEBUG=warn+all wine vrelaunch.exe 2>trace.txt
```

y genera un listado de 500 líneas.

Esta traza incluye todos los avisos (warning) que suceden en Wine. Estos avisos se producen porque algo no funciona como se espera. Por ejemplo algún archivo que no existe, o alguna rutina que no se sabe si está bien emulada.
Estos avisos deben de haber sido previstos por el programador, usando la línea:

```
WARN( "algo va mal porque valor=%x\n", value );
```

Por ejemplo, la rutina "LocaleNameToLCID" incluye:

```
o-------------------------------------------------------------------------o
|   LCID WINAPI LocaleNameToLCID( LPCWSTR name, DWORD flags )             |
|   {                                                                     |
|      ...                                                                |
|      if (!locale_name.matches)                                          |
|         WARN( "locale %s not recognized, defaulting to English\n",      |
|                debugstr_w(name) );                                      |
|      ...                                                                |
|   }                                                                     |
o-------------------------------------------------------------------------o
```

es decir, que si el idioma (locale) elegido no es correcto, lo pone en inglés.
El programador no sabe si esto es adecuado o no; por eso avisa.

En esta misma rutina encontramos

```
o-----------------------------------------------------o
|   if (flags) FIXME( "unsupported flags %x\n", flags ); |
o-----------------------------------------------------o
```

o sea, que esta rutina funciona correctamente, excepto si se usa el parámetro "flags" . En este caso genera un error de tipo 'fixme', en lugar de 'warning'.

Aun existe otro tipo de notificación usada por los programadores:

```
o-------------------------------------------------------------o
| TRACE("setting %x (%s) to %s\n", lctype, debugstr_w(value),  |
|                                         debugstr_w(data) ); |
o-------------------------------------------------------------o
```

esto permite saber los valores que se le han mandado a esta rutina.
Dado que Wine es capaz de ejecutar cualquier programa de Windows, no es seguro que usen los parámetros correctos. Con este tipo de traza es posible analizar un programa en modo off-line, cuando ya se ha ejecutado.

Para que se muestre una traza más completa, se usa: "WINEDEBUG=trace+all".
Para verlo más claro, os diré que una de las líneas que sale en trace.txt es

```
   0009:trace:reg:NtOpenKey (0x10,L"Environment",f003f,0xbfc1bc68)
```

y viene del fichero "wine/dlls/ntdll/reg.c" en la línea:

```
o-------------------------------------------------------------------------------o
|  NTSTATUS WINAPI NtOpenKey( PHANDLE retkey, ACCESS_MASK access,               |
|                                             const OBJECT_ATTRIBUTES *attr ) |
|  {                                                                            |
|     ...                                                                       |
|     TRACE("(%p,%s,%x,%p)\n", attr->RootDirectory,                             |
|                              debugstr_us(attr->ObjectName),access,retkey );   |
|     ...                                                                       |
|  }                                                                            |
o-------------------------------------------------------------------------------o
```

o sea, que la función NtOpenKey muestra una traza de 4 argumentos:

```
   1) attr->RootDirectory
   2) debugstr_us(attr->ObjectName)
   3) access
   4) retkey
```

No sólo eso, sino que esta función NtOpenKey incluye al final la línea:

```
o----------------------------o
| TRACE("<- %p\n", *retkey); |
o----------------------------o
```

que imprimirá el valor que se va a devolver.

Por eso en la traza encontramos: 0009:trace:reg:NtOpenKey <- (nil)

Para completar, vemos que la traza incluye "0009:trace:reg:NtOpenKey" que significa que el nivel de debug es 'trace' , que el módulo es 'reg', y que la rutina invocada es 'NtOpenKey'.

Dado que Wine es un proyecto bastante grande, se divide en módulos.
Por ejemplo la función "NtOpenKey" está incluida en el módulo 'reg'.
Esto permite debugear sólo los módulos que te interesan.

En este caso, yo buscaba las funciones que sirven para saber la hora y fecha actual, que se obtienen en:

```
o-------------------------------------------------------------------o
|   NTSTATUS WINAPI NtQuerySystemTime( PLARGE_INTEGER Time )        |
|   {                                                               |
|      struct timeval now;                                          |
|                                                                   |
|      gettimeofday( &now, 0 );                                     |
```

```
|      Time->QuadPart = now.tv_sec * (ULONGLONG)TICKSPERSEC +       |
|                                              TICKS_1601_TO_1970; |
|      Time->QuadPart += now.tv_usec * 10;                         |
|      return STATUS_SUCCESS;                                      |
|   }                                                              |
o------------------------------------------------------------------o
```

Vaya, esta función no tiene un TRACE adecuado, así que yo se lo pongo:

```
o-----------------------------------o
| TRACE("<- %x\n", Time->QuadPart); |
o-----------------------------------o
```

Pongo en marcha el programa, y espero a que llegue a esta línea.
Observo que se llama desde:

   1 - KERNEL32.GetTimeZoneInformation
   2 - KERNEL32.GetSystemTimeAsFileTime

así que pongo otra traza ahí.

Ahora puedo usar el método típico de poner un breakpoint y ver dónde "vrelaunch.exe" llama a "NtQuerySystemTime" y calcula los 7 días, pero voy a seguir usando Wine.

La estrategia es que sea el propio Wine el que salte la limitación. Para ello, recordar que el límite es 7 días, así que tengo que modificar "NtQuerySystemTime" para que siempre me devuelva la misma fecha:

```
o-------------------------------------------------------------------------o
|   NTSTATUS WINAPI NtQuerySystemTime( PLARGE_INTEGER Time )              |
|   {                                                                     |
|      // 100 segundos después del año 1601                               |
|      Time->QuadPart = 100*(ULONGLONG)TICKSPERSEC + TICKS_1601_TO_1970; |
|      return STATUS_SUCCESS;                                             |
|   }                                                                     |
o-------------------------------------------------------------------------o
```

Las primeras pruebas demuestran que esto está mal, porque hace que el tiempo sea constante y descoordina todo lo que se base en alarmas, intervalos, y fechas de modificación. Así que hay que ser más inteligente:

```
o-------------------------------------------------------------------------------o
|   NTSTATUS WINAPI NtQuerySystemTime( PLARGE_INTEGER Time )                    |
|   {                                                                           |
|      struct timeval now;                                                      |
|                                                                               |
|      gettimeofday( &now, 0 );                                                 |
|      now.tv_sec %= 7*24*60*60;     //número de segundos que hay en 7 dias   |
|      Time->QuadPart = now.tv_sec * (ULONGLONG)TICKSPERSEC +                   |
|                                              TICKS_1601_TO_1970;              |
|      Time->QuadPart += now.tv_usec * 10;                                      |
|      return STATUS_SUCCESS;                                                   |
|   }                                                                           |
o-------------------------------------------------------------------------------o
```

o sea, que devuelve el tiempo correcto, pero siempre referido a la primera semana del año 1601. De otra manera: el tiempo avanza como siempre, pero el domingo vuelve a ser el lunes de la semana anterior. Por eso parece que nunca pasan más de 7 días.

Con esto se consigue engañar al programa para que nunca caduque.

No es el crack perfecto porque necesita Wine y puede fastidiar a otros

programas, pero a mí me sirve; sobre todo para ilustrar el poder de Wine.

```
&&&_____________________________&&&
||| Y AHORA LLEGO EL POSTRE |||
&&&_____________________________&&&
```

Como parece que ha quedado corto, voy a profundizar un poco más.

Cuando se desarrolla un emulador y se prueban programas, a veces fallan.
En este caso es necesario analizar si el programa original también falla.
Para ello Wine cuenta con un debuger llamado winedbg que actúa como cualquier debuger típico de windows, con el aliciente de que se integra en Wine.

Entre las opciones que más interés tiene, están:

   (*) "info share" -> para ver las librerías que están cargadas
   (*) "info maps"   -> para que muestre las zonas de memoria del programa
   (*) "info locals" -> para ver las variables locales

Por supuesto también son fundamentales las instrucciones para poner breakpoints.

Vamos a verlo con un ejemplo.

```
$winedbg vrelaunch.exe
```

nos deja en el nuevo prompt:

```
Wine-dbg>
```

que se detiene en cuanto el proceso está cargado.

Para continuar, hacemos:

```
>finish
```

y el programa empieza. Podemos detenerlo con Control-C.

Para ver los registros:

```
>info all-reg
Register dump:
CS:0073 SS:007b DS:007b ES:007b FS:0033 GS:003b
EIP:ffffe410 ESP:0033f998 EBP:0033fa40 EFLAGS:00200246(   - 00      - IZP1)
EAX:00000000 EBX:00000002 ECX:0033fa64 EDX:00000000
ESI:00000008 EDI:b7dbbff4
```

Para desensamblar código:

```
>disassemble 0xffffe410
0xffffe410: popl        %ebp
0xffffe411: popl        %edx
0xffffe412: popl        %ecx
```

Para ejecutar la siguiente instruccion:

```
>n
```

Y para las siguientes, pulsar ENTER.

Por ejemplo, se detiene en:

```
GetActiveWindow () at /home/curro/wine-1.1.4/dlls/user32/../../
```

```
include/wine/server.h:62
62          if (res) SetLastError( RtlNtStatusToDosError(res) );
```

Si ahora queremos poner un breakpoint, se hace:

```
   >break NtQuerySystemTime
   Breakpoint 1 at 0x7efcb170 NtQuerySystemTime
   [/home/curro/wine-1.1.4/dlls/ntdll/time.c:449] in ntdll
```

y hala, a esperar a que se dispare.

A partir de este momento obtenemos una traza de este tipo:

```
   =>1 0x7efcb170 stub_entry_point+0x4c(dll="ntdll.dll", name="")
       [/home/curro/wine-1.1.4/dlls/ntdll/time.c:449] in ntdll (0x0033e8a0)
     2 0x1000aed0 in vrelaunch (+0xaed0) (0x0033e8d0)
     3 0x1000b1c9 in vrelaunch (+0xb1c9) (0x0033e8f8)
     4 0x10029cf5 in vrelaunch (+0x29cf5) (0x0033e910)
```

y se puede desensamblar:

```
   >disassemble 0x1000aed0
```

A partir de este punto a mí me gusta usar un desensamblador off-line como IDA. Pero no resulta difícil ver que habría que parchear la rutina de "vrelaunch.exe" que está alrededor de 0x1000b1c9.

Más información en:

```
   -> http://www.winehq.org/site/docs/winedev-guide/wine-debugger
```

```
o-_NOTAS_-------------------------------------------------------------------o
|                                                                           |
| 1.- A todos aquellos que están interesados en pasarse a Linux pero        |
|     necesitan usar aplicaciones Windows, les recomiendo que usen Wine.    |
|                                                                           |
| 2.- Para los que quieren investigar sobre emulación de sistemas, que      |
|     miren el código fuente.                                               |
|                                                                           |
| 3.- Y para los que quieren aprender otra técnica sobre ingeniería         |
|     inversa, esta es una gran herramienta fácil de adaptar a tus          |
|     necesidades.                                                          |
o---------------------------------------------------------------------------o
```

EOF

Indice

-[3x01]----------
-[Certificaciones]----------
-[by Anay]----------

Existe toda una oferta en certificaciones ¿quien no se ha planteado nunca sacarse una? Y aun me arriesgo mas a apostar porque seguro que el 30% o mas de los que pasais por esta zine habeis cursado alguna con mejor o peor resultado.

Lo que realmente es cuestion de debate es si certificarse en alguna materia o software en concreto es o no realmente positivo para nosotros yo diria que depende.

Como anecdota contar lo que me paso durante las primeras clases de MCSA, muchos de mis compañeros eran gente de uno u otro modo habian acabado trabajando como informaticos, en alguna tienda o gran empresa de helpdesk, hablando con ellos me contaban que ellos realmente habian acabado la ESO y despues se pusieron a trabajar de lo que les iba saliendo. Puede que sea muy pesimista pero en esos casos no creo que sirva de nada que la gente invierta tales cantidades de dinero en hacer una certificiacion. En primer lugar tienes que tener muy claro porque has de hacer esa certificacion, un companyero mio me contaba que queria ser administrador de redes informaticas, apenas tenia experiencia en una gran empresa en el departamento de help desk y sus conocimientos sobre informatica se limitaban a lo que habia aprendido leyendo esos formulario de pregunta-respuesta que tienen a su disposicion, yo para esos casos recomiendo en vez de una certificacion el realizar otro tipo de cursos.

Si por el contrario ya has terminado tu ciclo o carrera y estas trabajando en una empresa y crees que tener esos titulos te abriran las puestas del CPD y de la consola de AD con mayores permisos, en ese caso, merece la pena certificarte. Todo conocimiento es bueno pero por experiencia os digo que hay que hacerlo desde un punto de vista con realismo, cuando llego el momento de comenzar los examenes muchos ni siquiera se sacaron el de XP (ahora tambien disponible el de Vista :)) ya que se creian que eso del examen de XP era instalar el sistema y drivers.

Sobre el tema del idioma: si, si y si lo mejor es hacer los examenes en ingles desde el principio. Soy consciente de que no todos tenemos el nivel de ingles que, segun el ministerio, deveriamos tener pero si yo he conseguido entender lo que preguntan cualquiera lo puede conseguir. Hay examenes ya traducidos que puedes estudiar y examinarte en ingles pero son los minimos y probablemente cuando lleges a un examen mas amplio si no te has familiarizado ya con todas esas palabras en ingles tienes muchas mas posibilidades de que se te haga cuesta arriba.

No seas pesimista con esto, una vez hayas echo las primeras 300 preguntas de test veras que las palabras son siempre las mismas y que ya apenas te cuesta esfuerzo entender todo lo que ponen, sin embargo las traducciones en muchisimos casos dejan bastante que desear y una pregunta mal traducida puede convertirse en una pregunta no acertada y necesitando un 700/1000 lo mejor es reducir esas preguntas fallidas cuanto mas mejor.

Una vez echos los pequenyos consejillos pasemos ha hablar sobre el como de las certificaciones.

Certificaciones de Microsoft.

Es la madre de todas las certificaciones, no porque sea la mejor sino porque es la que mas variedad de examenes te puede pedir y porque es la mas popularizada.

Hasta hace un anyo teniamos 2 certificaciones una para administradores y otra para ingenieros:

1 - MCSA
2 - MCSE.

No puedes conseguir la una sin pasar primero por la otra, te vendan lo que te vendan en las academias :P

[--- "MCSA" Administrador Certificado en Sistemas Microsoft ---]

Ante certificaciones de Microsoft relacionadas con sus sistemas esta sera la primera prueba que tendremos que vencer, consta de 4 examenes:

-> Windows XP (ahor atambien disponible Windows Vista).

-> Administracion y mantenimiento de entornos Windows Server 2003.

-> Implementacion, administracion y mantenimiento de redes con Windows Server 2003.

Hasta aqui lo obligatorio, una vez superados estos tres examenes puedes elegir el cuarto entre Isa Server o Exchange Server.

MCSA especializado en seguridad: SOn en total 5 examenes con el de ISA y el de Seguridad, propiamente dicho. La informacion oficial sobre la numeracion de los examenes la tienes aqui:

- http://www.microsoft.com/learning/mcp/mcsa/security/windowsserver2003.mspx

MCSA especializado en Mensajeria: Consta de los 3 examenes ya dichos y tan solo 1 mas, siendo este un poco mas corto, aqui tienes el enlace a las numeraciones y opciones:

- http://www.microsoft.com/learning/mcp/mcsa/messaging/windowsserver2003.mspx

Como apunte indicaros que tambien se pueden cursar con Windows 2000 Server, yo personalmente pienso que si vas ha hacer la titulacion completa no es util ya que es una tecnologia casi obsoleta y muy sencilla de dominar si saber acerca de 2003.

De todos modos aqui os dejo tambien su enlaces:

- Experto en seguridad Windows 2000:
  http://www.microsoft.com/learning/mcp/mcsa/security/windows2000.asp

- Experto en mensajeria Windows 2000:
  http://www.microsoft.com/learning/mcp/mcsa/messaging/windows2000.asp

Es muy importante que quienes hizieron estas certificaciones en el pasado sepan que no hace falta hacer todos los examenes de nuevo para actualizarse, solo tienen que hacer algunos de ellos.
Personalmente os recomiendo que os pongais en contacto con Prometric para que

os digan el centro certificador mas cercano a vosotros y asi poder preguntar alli mismo por vuestro caso. Estos son los telefonos de contacto en europa:

- http://www.prometric.com/Microsoft/EMEAIT.htm

[--- "MCSE" Ingeniero Certificado en Sistemas Microsoft ---]

Una vez que hemos superado el MCSA podemos comenzar con el MCSE, si es bien cierto que puedes hacer examenes de MCSE sin haber terminado los de MCSA o hacerlos salteados pero personalmente no recomiendo que hagais eso, si realmente quereis tener ambas certificaciones lo mejor es ir en orden ya que hay examenes sobre ingeniera de redes que es absurdo hacer si antes no has pasado por la administracion de Windows 2003. Una vez llegados a este nivel ya apenas os separan un par de examenes de nuestro titulo.

En esta parte nos enseñaran ya no solo a administrar y mantener redes sino que tambien aprenderas ha plantear las redes desde 0.

Especializado en seguridad - Aquí os dejo el enlace de oficial de los examenes y sus numeraciones que tendreis que cursar, estan todos incluidos los de MCSA que ya podriais descontarlos ;) :

- http://www.microsoft.com/learning/mcp/mcse/security/windowsserver2003.mspx

Especializado en mensajeria - Lo mismo que el anterior, aqui os dejo el enlace:

- http://www.microsoft.com/learning/mcp/mcse/messaging/windowsserver2003.asp

[--- ¿Academia o no academia? ---]

Existen academias que te pueden formar en las distintas certificaciones, las mas extendidas con las de Microsoft y Cisco estas academias suelen formarte para sacar los cursos basicos como son el MCSE, MCSA, CCNA... y para lo cual estan muy bien sobretodo para entrar en la dinamica de las certificaciones, irte acostumbrando al idioma y al estudio de test junto con clases al principio viene muy bien.

La respuesta a esa pregunta puede ser muy variada en primer lugar dependera de nuestra economia, no te he exagerado antes cuando he dicho que son caras y es que en una academia cursar un MCSE puede salirte por mas de 5000 euros lo cual ya es una cantidad de dinero que no todo el mundo puede pagar.
Tambien dependera, sobretodo, de la experiencia que tengan en lo que a la materia respecta, es decir hay examenes como Mantenimiento de Windows 2003 que si has estado trabajando un par de anyos y estas familiarizado con dicho entorno te va a resultar demasiado facil y posiblemente terminaras prescindiendo de las clases, siempre hay algo nuevo pero para eso estan los libros, que son tambien carisimos en todas ellas.

Una ayuda que podeis encontrar y que si suelen hacer mucho es pedir a tu empresa que te financie estos estudios, si la empresa cree que será probechoso para ellos seguramente accedan a pagartelos y sin duda alguna en ese caso asiste a clases ya puestos a pagar...

Tienes que tener en cuenta que las certificaciones no acaban donde parece, detrás del MCSE hay decenas de posibilidades mas, incluso puedes hacerte profesor si te sacas el Trainer o muchos otros examenes que lucen muy bonitos en el curriculum pero estos seguramente no te los cursarán en las academias, tendrás que acudir a un centro autorizado ha hacer el examen y nada mas. Asi que esta opcion, como ya os he dicho solo es valida para los primeros examenes.

[--- Otras certificaciones ---]

Me he explayado mucho mas en la Microsoft que en ninguna otra, ciertamente es la que conozco (luchando con ella estoy ahora mismo :P) pero existen cientos de certificaciones, casi tantas como empresas de informatica.
Para hacerte una idea de las posibilidades que existen y asi ver tu cual realmente te conviene tienes la web de Prometic, se podria decir que es la madre reguladora jeje.

Desde aqui podeis acceder a toda la informacion acerca de todas ellas:

- http://www.prometric.com/Candidates/default.htm

Ellos te ayudaran e informaran de todas las dudas que puedas tener.

[--- Sitios indispensables ---]

Cuando ya estes dentro de una certificacion necesitaras hacer cientos de preguntas ya que es lo que realmente va a ayudarte a la hora de pasar el examen.

Existen empresas alrededor de toda esta industria, por ejemplo tenemos a TestKing (http://www.testking.com/), la lider indiscutible en venta de test de pasados examenes, desde su web puedes comprar pdfs con cientos de preguntas de examenes pasados del que quieras, puedes bajarte de internet esos examenes, el problema esta en que comprando tendras las ultimas preguntas y los que te descargas fijate bien en que version te has bajado, ya que pueden ser preguntas de hace mas de 2 anyos y luego llevarte una desagradable sorpresa cuando estes ante el examen.

Otras empresas relacionadas son:

- http://www.pass4sure.com/
- http://www.actualtests.com/
- http://www.trandumper.com/

Pero como ya he dicho el lider indicutible es TestKing.

Si tu economia no da para mas o simplemente no tienes ganas de pagar mas dinero por tener ejemplos de examenes tienes otro sitio donde una inmensa comunidad colabora muy activamente para poder disponer de esos examenes y no en pdf sino en simulador como si de un examen real se tratara, esta comunidad la tienes en:

- http://www.examcollection.com

Para poder acceder a los examenes que tienes de descarga gratuita necesitaras el software de Visual CertExam Suite, es de pago pero como conseguirlo ya lo dejo a tu imaginacion ;)

Desde estos sitios tienes ya todas las herramientas que necesites para aprobar el examen que quieras de la titulacion que desees.

[--- Conclusion ---]

Espero que os haya sido de utilidad, he tratado mediante el ejemplo de Microsoft haceros una pequenya idea de como funcionan las certificaciones

y de donde poder encontrar recursos y ayuda para sacarlas. Ya no teneis excusa a estudiar toca :)

EOF

-[3x02]----------
-[La Cripta]----------
-[by blackngel]----------

HACK THE WORLD

by blackngel <blackngel1@gmail.com>
<black@set-ezine.org>

---[1 - Prologo

Hace tiempo que llevo pensando en una idea. No existe tiempo fisico ni dispongo de servidores suficientes como para llevar a cabo tal tarea. Es por ello que expongo aqui lo que se me ha ocurrido por si a alguien le pica la curiosidad y desea meterse en un nuevo proyecto.

---[2 - La Cripta

La idea de La Cripta se me ocurrio tras leer la novela de culto de Neal Stephenson "Criptonomicon". Escogi este nombre porque me parece el mas acertado.

El proyecto se basa en crear un espacio web (o paraiso de datos) donde uno pueda subir toda aquella informacion que desee (ficheros) quedando esta completamente ilegible despues de cerrar su sesion.

Seria mas bien como crear un agujero negro donde poder tirar todo aquello que deseamos esconder de miradas ajenas, con la posibilidad de recuperarlo en cualquier momento.

Que necesitariamos?

1 - Un servidor Apache. [1]
2 - PHP [2]
3 - MySQL [3]
4 - Truecrypt [4]

Como podria implementarse?

Pues bien, la interfaz de la pagina principal deberia ser igual de sencilla que el buscador de Google. Es decir lo minimo para que un usuario pueda iniciar su sesion y un cuadro de dialogo para que pueda subir sus archivos sin tener que utilizar ningun otro tipo de interactuacion.
Si el usuario quisiera recuperar alguno de sus ficheros, deberia acceder a otro

enlace que le facilitaria un listado de los mismos.

El problema?

El ESPACIO, como siempre. Lo normal seria proseguir la idea de un servidor de correo o hosting, asignando una cantidad fija de disco duro a cada usuario (por ejemplo 50Mb).

Cuando el usuario crea por primera vez su cuenta en "La Cripta", el servidor deberia a~adir automaticamente un nuevo volumen encriptado con la contrase~a del usuario mediante el software "Truecrypt". Ello es factible gracias a que la linea de comandos de truecrypt nos facilita la creacion de volumenes con una sola orden sin necesidad de interactuar directamente con el.

El resto de las veces, cuando el usuario inicie su sesion el volumen se montara para su uso. Al cierre de la sesion el volumen encriptado se desmontara.

Una de las ventajas es que no todos los usuarios se veran obligados a utilizar el mismo tipo de cifrado. Gracias a que Truecrypt dispone de varios algoritmos realmente potentes, esta eleccion podria ofrecersele al usuario en la creacion de su cuenta.

El nombre del volumen deberia asignarse en la base de datos de usuarios guardando asi la relacion con el mismo, su nombre deberia ser aleatorio. De este modo, en caso de intrusion en el servidor, no se ofreceria ninguna pista acerca de a quien pertenecen los volumenes.

Vale! Hasta aqui bien, pero hay dos datos que necesitamos tener en cuenta. Si alguiente puede facilitarmelos seria de gran ayuda:

1 - Numero de volumenes simultaneos que puede manejar Linux.
2 - Numero de volumenes simultaneos que puede manejar Truecrypt.

Si esto supusiera un limite, deberiamos controlar las entradas al servidor denegandolas cuando el tope haya sido alcanzado. Esto seria una perdida de eficiencia enorme cuando no se disponen de suficientes servidores.

Bueno, y ya no queda mucho por decir, el concepto ha sido expuesto lo mas claramente posible. Ahora solo queda que te pongas manos a la obra y que utilices un poco de tu imaginacion.

---[3 - Conclusion

Como has visto, esto no ha sido mas que una simple idea con posibilidad de desarrollo. Si estas dispuesto a emplear un poco de tu tiempo y, sobre todo, si tienes la gran amabilidad de prestar un servicio (deberia ser GRATUITO) a la sociedad, entonces puedes convertirte en alguien grande.

ETICA: Algunos ya se les habrá ocurrido y es cierto. Por que ofrecer un servicio de estas caracteristicas a elementos tales como: Terroristas, pedofilos, crackers, kiddies y un largo etc.

Vale, estamos jodidos, pero eso no es una forma de pensar evolutiva. Debemos aceptar que los buenos, aquellos que luchamos contra este tipo de comportamientos, tambien necesitamos escoder las armas con que combatimos a estas personas.

Al ser una idea, "La Cripta" necesita gente que piense en ella. Cualquier elemento que puedas aportar, ya sea intelectual como material, no dudes en hacerlo saber en <blackngel1@gmail.com>.

Y estate tranquilo, si consigues llevar a cabo este proyecto, a mi no

me habras robado la idea, pues para eso la he expuesto. El fruto del reconocimiento es que esto se convierta en realidad.

---[4 - Referencias

 [1] Apache
     http://www.apache.org

 [2] PHP
     http://www.php.net

 [3] MySQL
     http://www.mysql.com

 [4] Truecrypt
     http://www.truecrypt.org

EOF

-[3x03]----------
-[Syn vs Fin Scan]----------
-[by blackngel]----------

 HACK THE WORLD

 by blackngel <blackngel1@gmail.com>
              <black@set-ezine.org>

---[1 - Prologo

Aqui se presentara el motor basico de un escaner syn / fin y la diferencia en concepto que existe a la hora de obtener las respuestas e interpretarlas.

No es una maravilla lo que encontraras aqui, pero si te apetece leer un poco de codigo y conocer como funcionan este tipo de escaneos sin la necesidad de destripar Nmap [1] y evitar la agonia de la complejidad que presenta para un novato, entonces estas en el lugar adecuado.

---[2 - Motor Escaneo

Aqui tienes el motor principal para enviar paquetes syn o fin. Debes estar especialmente atento a la funcion "libnet_build_tcp()". En el quinto parametro de este metodo debes elegir si enviar la constante TH_SYN o TH_FIN segun el tipo de escaneo que desees realizar.

Para realizar esta tarea, puedes ayudarte del 3 parametro del motor "type",

segun el valor de esta variable deberias establecer el 'flag' correspondiente.

Tienes deberes. Adapta el codigo.

Requisitos:

 - LibPcap (version 0.8) [2]
 - LibNet (version 1.1 o superior) [3]

Recuerda, las opciones de compilacion para programas que utilicen estas
librerias son: "-lpcap -lnet".

```
#include <stdio.h>
#include <pthread.h>
#include <pcap.h>
#include <libnet.h>

#define SRCPORT 31313

libnet_t *lnet;
u_int32_t src_ip;

static int tscan;

int
scan_syn_fin(char *obj, int nports, int type)
{
   int c;
   u_int port;
   u_int iphdr = 1;
   u_long dst_ip;
   libnet_ptag_t tcp_pkt = 0;
   libnet_ptag_t ip_pkt = 0;
   u_int8_t *payload = NULL;
   u_int32_t payload_s = 0;
   char errbuf[PCAP_ERRBUF_SIZE];
   pthread_t th_snifftcp;

   /* El capturador de paquetes debe saber que tipo de escaneo realizamos */
   tscan = type;

   /* Iniciamos el contexto libnet y obtenemos la ip de origen */
   lnet = libnet_init(LIBNET_RAW4, device, errbuf);
   src_ip = libnet_get_ipaddr4(lnet);

   /* sniff_tcp() es una funcion que capturara las respuestas */
   pthread_create(&th_snifftcp, NULL, sniff_tcp, (void *)nports);

   sleep(1); /* Damos tiempo a que se inicie el capturador de paquetes */

   /* Resolvemos la direccion del host pasado como 1er argumento */
   dst_ip = libnet_name2addr4(lnet, (u_char *)obj, LIBNET_RESOLVE);
   if (dst_ip == -1) {
      printf("libnet_name2addr4(lnet) failed: %s\n", libnet_geterror(lnet));
      exit(1);
   }

   printf("   ***** Escaneando: %s *****\n", libnet_addr2name4(dst_ip, 0));

   for(port = 0; port < nports; port++){

      libnet_seed_prand(lnet); /* Preparar aleatoriedad en el contexto */
```

```
      tcp_pkt = libnet_build_tcp(SRCPORT, /* PUERTO ORIGEN */
      /* PUERTO DESTINO */       port,
      /* SEQ */                  libnet_get_prand(LIBNET_PRu32),
      /* ACK */                  libnet_get_prand(LIBNET_PRu32),
      /* TH_SYN o TH_FIN */      TH_SYN,
      /* TAMAÃ‘O VENTANA */       libnet_get_prand(LIBNET_PRu16),
      /* CHECKSUM */             0,
      /* URG */                  0,
      /* LONGITUD PAQUETE */     LIBNET_TCP_H + payload_s,
      /* DATOS */                (u_char *) payload,
      /* LONGITUD DATOS  */      payload_s,
      /* CONTEXTO */             lnet,
      /* MANEJADOR TCP*/         tcp_pkt);

      /* La cabecera IP solo debe configurarse la primera vez */
      if(iphdr){
         iphdr = 0;

         ip_pkt = libnet_build_ipv4(LIBNET_IPV4_H + LIBNET_TCP_H + payload_s,
         /* TOS */                  0,
         /* ID */                   0,
         /* FRAGMENTACION */        0,
         /* TTL */                  64,
         /* PROTOCOLO */            IPPROTO_TCP,
         /* CHECKSUM */             0,
         /* IP ORIGEN */            src_ip,
         /* IP DESTINO */           dst_ip,
         /* DATOS */                NULL,
         /* LONGITUD DATOS*/        0,
         /* CONTEXTO */             lnet,
         /* MANEJADOR IP */         ip_pkt);
      }

      c = libnet_write(lnet); /* Enviamos el paquete */
      usleep(50); /* nanosleep() */
   }
   libnet_clear_packet(lnet); /* Vaciamos el paquete */
   libnet_destroy(lnet); /* Cerramos el contexto */
}
```

```
 **------------------------------------------------------------------------**
```

Observaras que el motor es realmente sencillo:

- Se crea un hilo con una funcion que capturara los paquetes (se vera en la siguiente seccion).

- Un bucle 'for' que va desde 0 hasta el numero indicado en el segundo parametro de la funcion.

- Para cada puerto se crea un nuevo paquete configurando la cabecera IP solo la primera vez y la cabecera TCP en los sucesivos ciclos.

- Se manda un paquete cada 50 milisegundos.

---[3 - Respuesta

La funcion para establecer la interfaz de red en modo captura es muy sencilla y suele seguir siempre un mismo orden.

OJO: El programa que implemente tanto este como el codigo anterior, debe

mantener una variable global como 'char *device' que contenga la interfaz de red que utilizaras para realizar el escaneo.

```
/* Paquetes TCP destinados al puerto desde el que realizamos el escaneo */
#define FILTRO_SYNFIN "tcp and dst port 31313"

void *
sniff_tcp(void *var)
{
   int rc;
   char errbuf[PCAP_ERRBUF_SIZE];
   struct bpf_program filtro;
   bpf_u_int32 netp, maskp;
   pcap_t* snifftcp;

   snifftcp = pcap_open_live(device, 1024, 0, 0, errbuf);
   if (snifftcp == NULL) {
      asprintf(&cadena, "pcap_open_live(): %s\n",errbuf);
      salida(1);
      return -1;
   }

   if (pcap_lookupnet(device, &netp, &maskp, errbuf) == -1) {
      asprintf(&cadena, "Error en pcap_lookupnet(): %s\n", errbuf);
      salida(1);
      return -1;
   }

   if(pcap_compile(snifftcp, &filtro, FILTRO_SYNFIN, 0, netp) == -1){
      asprintf(&cadena,"Error compilando el filtro\n");
      salida(1);
      return -1;
   }
   if(pcap_setfilter(snifftcp, &filtro) == -1){
      asprintf(&cadena,"Error aplicando el filtro\n");
      salida(1);
      return -1;
   }

   rc = pcap_loop(snifftcp, (int)var, &leer_tcp, NULL);
}
```

Y ahora viene lo interesante del asunto:

Como funciona el escaneo SYN?

1 - Se envia un paquete SYN a un puerto del objetivo.
2 - Si se recibe un SYN/ACK, el puerto esta abierto.
3 - Si se recibe un RST, el puerto esta cerrado.

Como funciona el escaneo FIN?

1 - Se envia un paquete FIN a un puerto del objetivo.
2 - Si no se recibe "ninguna" respuesta, el puerto esta abierto.
3 - Si se recibe un RST el puerto esta cerrado.

El el primer caso, la operacion es muy facil, aquellos paquetes cuyo campo 'th_flags' de la cabecera TCP coincida con las constantes TH_SYN Y TH_ACK resultaran en un puerto abierto que extraeremos inmediatamente.

El segundo caso es un poco mas complejo de lo que parece, pero no tanto.
Me explico. En los puertos abiertos no llegara ninguna respuesta. La cuestion es, como interpretar esa "no respuesta"?

La solucion es recibir los paquetes con el flag RST activado y llevar un contador con el siguiente puerto que deberia responder otro RST comparandolo cuando este llegue para ver que coincide con el puerto.
Si un puerto esta abierto, este paquete no llegara, el contador no se incrementara y la proxima vez que recibamos una respuesta la comparacion nos alertara de que no coinciden.

Vamos a explicarlo con un ejemplo para que se entienda este embrollo. Imaginar que el puerto 4 esta abierto. Empezaremos con un contador llamado "nextp" a 0.

```
   * pe -> puerto escaneado

   static int nextp = 0;

   Escanear Puerto = 0 -> [RST] -> (nextp == pe) -> nextp = pe + 1;
   Escanear Puerto = 1 -> [RST] -> (nextp == pe) -> nextp = pe + 1;
   Escanear Puerto = 2 -> [RST] -> (nextp == pe) -> nextp = pe + 1;
   Escanear Puerto = 3 -> [RST] -> (nextp == pe) -> nextp = pe + 1;
   Escanear Puerto = 4 -> NO HAY RESPUESTA
   Escanear Puerto = 5 -> [RST] -> (nextp != puerto escaneado)
                                   ||
                                   ||-> PUERTO "nextp" ABIERTO.
                                        nextp = puerto escaneado + 1;
   Escanear Puerto = 6 -> [RST] -> (netxp == puerto escaneado) -> nextp += 1;
   ...
   ...
```

Tambien debemos tener en cuenta que puede haber varios puertos consecutivos abiertos. Para ocuparnos de este detalle utilizaremos un sencillo bucle while que ira marcando puertos abiertos mientras 'nextp' no vuelva a coincidir con el puerto escaneado. Es decir, si hay 3 puertos seguidos abiertos, el contador va a diferir en 3 por debajo del puerto esperado. Iremos incrementando este contador y marcando como abiertos los puertos en cada interaccion hasta que este se vuelva a igualarse con el puerto escaneado.

Creo que lo he liado mas de lo que realmente es. Aqui teneis el codigo. Mas vale un buen codigo que mil palabras :)

```
 **-------------------------------------------------------------------------**

struct hosts {
   char ip[16];
   int h_ports[65536];
} thost;

static int nextp = 0;

void
leer_tcp(u_char *useless, const struct pcap_pkthdr* pkthdr,const u_char* pkt)
{
   struct libnet_ipv4_hdr *iph;
   struct libnet_tcp_hdr *tcph;

   iph = (struct libnet_ipv4_hdr *)(pkt + LIBNET_ETH_H);
   tcph = (struct libnet_tcp_hdr *)(pkt + LIBNET_ETH_H + LIBNET_IPV4_H);

   switch (tscan) {

      /* SYN SCAN */
      case 1: {
```

```
         /* Podrias hacer: (tcph->th_flags & (TH_SYN|TH_ACK) ) */
         if ((tcph->th_flags & TH_SYN) && (tcph->th_flags & TH_ACK)) {
            thost.h_ports[ntohs(tcph->th_sport)] = 1;
         }
         break;
      }

      /* FIN SCAN */
      case 2: {
         if (tcph->th_flags & TH_RST) {
            if (nextp != ntohs(tcph->th_sport)) {
               while (nextp != ntohs(tcph->th_sport)) {
                  thost.h_ports[nextp] = 1;
                  nextp += 1;
               }
            }
            nextp = ntohs(tcph->th_sport) + 1;
         }
         break;
      }
   }
}
```

```
 **-------------------------------------------------------------------------**
```

Y esto es todo lo que te voy a proporcionar, en la estructura 'thost' han quedado almacenados los puertos abiertos. Estos se distinguen por los elementos del array h_ports[] que son iguales a 1.

Imprimirlos en pantalla es algo trivial. Pon tus manos sobre el teclado y crea tu propia interfaz que implemente estas funciones.

---[4 - Conclusion

Con esta forma de trabajar, realizar otro tipo de escaneos se vuelve mucho mas facil. Si echas un vistazo al README del programa "nmap" se te pueden ocurrir cosas bastante interesantes.

Cualquier duda o correccion, a <blackngel1@gmail.com>.

Hasta la proxima...

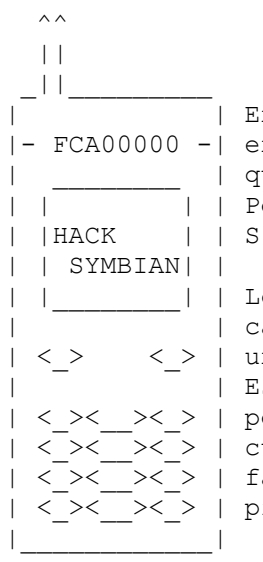
---[5 - Referencias

[1] Nmap
    http://www.insecure.org

[2] LibPcap
    http://sourceforge.net/projects/libpcap/

[3] LibNet
    http://libnet.sourceforge.net

EOF



En un número anterior conté cómo hacer un escalado de permisos en teléfonos Symbian. Aquello era válido para algunos modelos que usaban la versión S60v1, que eran mayoría en aquella época. Pero hace unos 2-3 años ha habido una nueva versión llamada S60v3, también conocida como Symbian 9.

Los que desconozcan cómo funciona, pensarán que no es un gran cambio. Nada más lejos de la realidad: en S60v3 se implementa un mecanismo de seguridad apropiado para teléfonos móviles. Es como cuando Windows 3.1 evolucionó hacia Windows NT para permitir permisos en el sistema de ficheros NTFS. Anteriormente, cualquier programa podía escribir en cualquier directorio y fastidiar sus datos. Es cierto que la memoria del kernel estaba protegida, pero era lo único.

Ahora, cada programa tiene una serie de permisos, organizados en 4 capas:

1 -> Común
2 -> Restringida
3 -> Sensible
4 -> Crítica.

Para que un programa pueda usar una funcionalidad, debe estar firmado con un certificado otorgado por Nokia o Symbian.

Por tanto existen 4 tipos de certificados:

-> El de tipo 'común' se distribuye gratuitamente. No da acceso a mucha funcionalidad, pero es gratis.

-> El de tipo 'restringido' se les da a algunas empresas. No es difícil conseguirlo, pero hay que pagar unos 200 euros, y otros 200 por programa.

-> El de tipo sensible se les da a pocas compañias. Cuesta mucho dinero y esfuerzo.

-> El crítico no se le da a nadie. Sólo lo tiene Nokia, Motorola, SonyEricsson, y alguna empresa más. Ni que decir tiene que es el más potente.

Esto hace que los programadores vulgares tienen el certificado 'común' y no puedan hacer programas que pongan en riesgo el sistema. La idea está bien: si tienes un certificado, tu aplicación tiene unas ciertas garantías, y puedes acceder a funcionalidad crítica.

Sin embargo esto conlleva que los usuarios no tienen control total sobre sus móviles. En particular no pueden ver todos los archivos.

Esto supone un obstáculo para la piratería de juegos. Voy a explicar porqué. Un juego incluye uno o más ficheros. Éstos se agrupan en un fichero de instalación, que se firma en la fábrica, y el usuario debe instalar. Supongamos que el fichero principal es "c:\sys\bin\juego.exe". Entonces sólo el programa de instalación puede escribir ficheros en el directorio "c:\sys\bin". Esto es importante.

Supongamos que cualquiera puede leer estos ficheros. Es más o menos cierto, pero no es necesario dar más detalles. Así que un cracker puede extraer estos ficheros y hacer un crack. Con más o menos esfuerzo, tiene "c:\sys\bin\juego_crk.exe"

Pero la pregunta es: ¿Cómo se puede meter de nuevo? La respuesta es, en general, NO. La única manera es usar un instalador.

No es del todo cierto: si el juego tiene únicamente permisos de la capa 'común', el cracker sí puede hacer un mini-instalador, usando su certificado 'común'. Pero si el juego original usa permisos de la capa restringida, entonces el cracker no tiene un certificado lo suficientemente potente para incluir "c:\sys\bin\juego_crk.exe".

Si bien es cierto que la mayoría de los juegos no necesitan permisos más allá de la capa 'común', los fabricantes se han dado cuenta de que usando permisos, pueden detener la piratería. Y parece funcionar bien.

Hasta ahora :-)

He descubierto una debilidad en este sistema!

Aunque he sido yo quien la ha descubierto y usado, he tenido ayuda de varias personas, así que esta es mi manera de agradecérselo a todos: voy a explicarlo como si fuera una conversación, que empiezará al publicar un mensaje en un foro sobre teléfonos móviles:

|---|

FCA00000: Hola. He visto que S60v3 está protegido. Yo tengo experiencia en móviles. ¿Alguien se anima a romperlo?

Vovan888: uff, el tema está complicado. El problema es que no podemos leer los archivos, así que no sabemos cómo funciona la protección.

FCA00000: bueno, mi SX1 usaba S60v1 y resulta que todos los programas estaban en una zona de memoria que empieza en 0x50000000.

Vovan888: ¿Cómo llegaste a esa conclusión? Quizás el mismo razonamiento sea válido.

FCA00000: en arquitecturas ARM, hay un registro llamado PC que te dice qué dirección de memoria se está ejecutando. Para llamar a rutinas del sistema, se pone PC=0x50000000+yyyyyyy , donde yyyyyyy es un valor distinto para cada rutina. Se podía volcar esa memoria en un fichero y analizarla.

Vovan888: mis pruebas indican que los programas de usuario se ejecutan en la dirección 0xF5000000+tttttt.

FCA00000: acabo de hacer un programa en mi S60v3 que vuelca 16 Kb a partir 0xF5000000 y efectivamente se asemeja al código binario de un programa. Parece ser que S60v3 llama a rutinas a partir de: 0xF8000000+zzzzz.

atzplzw: Toma, aquí tienes un programa que vuelca a partir de 0xF8000000. Yo lo he probado y he desensamblado el resultado. Son 16 Mg, y ciertamente parece código para procesadores ARM. De hecho hay algo que parecen ser nombres de ficheros.

FCA00000: correcto. Al principo de la memoria hay una lista de ficheros, con las direcciones en las que se almacenan. Es como un disco duro con acceso lineal: cada archivo está en una dirección de memoria. He logrado extraer cada fichero. Hay unos 1.500.

atzplzw: en el SDK de Symbian, que es gratis, hay un documento que explica cómo es la cabecera de cada fichero. Se llama RomHeader.

FCA00000: no sólo eso; la herramienta llamada petran permite mostrar dicha cabecera. El fichero más interesante se llama ekern.exe , que es el kernel.

Hex: yo lo he desensamblado y le he dado nombre a algunas de sus rutinas. En particular hay una que se llama -> Kern::DoCurrentThreadHasCapability.

FCA00000: en el SDK incluye un emulador. No es exactamente lo mismo poque funciona con código x86 en lugar de ARM, pero también existe "ekern.exe" y también incluye esa rutina.

Hex: ya entiendo cómo funciona: Un programa tiene 'capacidades', que se le definen cuando se compila, y se guardan en la cabecera del programa. Al hacer el instalador se verifica que tu certificado te permite incluir programas con esas capacidades. Si no, no puedes ni siquiera hacer un instalador.

FCA00000: no tengo un certificado que me otorgue capacidades altas. Para ver qué sucede, he hecho un programa.exe que las necesita, aunque no las tiene otorgadas. Lo he podido instalar, pero falla a la hora de ejecutar la rutina. Lo bueno es que el programa se inicia correctamente; sólo falla al ejecutar la rutina que carece de capacidades.

Hex: ¿Cual es el error que te da?

FCA00000: devuelve el valor KErrNoPermission. El error lo provoca el "ekern.exe". Lo he probado en el emulador poniendo un breakpoint, y he visto que las capacidades se extraen de la cabecera del fichero. Cuando el programa se carga en memoria antes de ejecutarlo, se inicializa una zona de memoria con esos mismos valores. Esta zona pertenece al kernel, no al programa.

ZoRn: Quizás puedas usar la misma técnica de tu artículo anteror, usando "ExecHandler::LockedInc".

FCA00000: ya lo he probado. Pero los de Symbian/Nokia ha cerrado esa puerta de un modo elegante: antes de escribir un dato, verifican que pertenece a la zona de usuario o de kernel.

ZoRn: ¿puedes cambiar ese trozo de código? Existe una debugger llamado MetroTRK que permite detener un programa para investigar.

FCA00000: he conseguido instalarlo y ver la zona de memoria de usuario, tanto el código de programa como los datos. Lamentablemente sólo puedo debuguear mi propio programa, no ekern.exe

ZoRn: eso es una limitación del propio MetroTRK . Yo lo he desensamblado y he visto que no permite ver memoria de código de otros programas. Como MetroTRK tiene capacidades críticas, estamos con el problema recursivo: si lo pudiéramos parchear, podríamos saltarnos las restricciones. Pero para parchear, estamos restringidos. Ojalá pudiéramos debuguear el mismo MetroTRK.

FCA00000: ¿Dices que MetroTRK sólo limita la memoria de código? Un programa tiene código y datos. Y es posible ver los datos de otros programas ! Comprobado !

ZoRn: ¿entonces puedes modificar la memoria del ekern.exe ?

FCA00000: dicho y hecho. Como ya sabes, ekern.exe mantiene en memoria una lista de los programas en ejecución. Además del nombre, tiempo de CPU, y memoria usada, se incluye las capacidades que tiene otorgadas. Inicialmente se sacan de la cabecera del fichero "programa.exe" pero el debuger MetroTRK me ha dejado modificarlas para ampliarlas. O sea que le he cambiado las capacidades, sobre la marcha. Ahora programa.exe es capaz de llamar a rutinas 'críticas'!

|---|

Para aquellos que les gustan los detalles técnicos, extraídos del modelo N80 con firmware 5.0719.02 - La memoria empieza en 0xF8000000 y ocupa 20 Mb.
Se pueden extraer los ficheros con la herramienta ROMTools hecha por mí. Genera unos 1.000 ficheros. La puedes encontrar en:

- http://FCA00000.googlepages.com/romtools-v0.2.2.rar

El fichero "ekern.exe" ocupa unos 250 Kb y se puede analizar con el desensamblador IDA. Genera unas 1.500 rutinas, que lamentablemente sólo se saben unos 400 nombres.

La rutina "Kern::DoCurrentThreadHasCapability" acaba llamando a "DProcess::DoHasCapability" que está en la dirección 0xF8047330.

El "ekern.exe" almacena todos sus datos en la zona que empieza en 0x60000000 y ocupa un 200 Kb.

La lista de procesos se guarda en una zona de memoria indicada por 0x60000148, por ejemplo 0x60004000.

Cada proceso tiene un número secuencial y usa una estructura de 0x40 datos.

Si programa.exe tiene número 0x123 entonces sus datos se almacenan en 0x60004000+0x40*0x123 = 0x600088C0. Las capacidades se encuentran en el byte 0x4 , es decir, 0x600088C4. Esto es un dato de 4 bytes, lo cual almacena 32 bits. Sólo hay 20 capacidades, por lo que se malgastan 12 bits.
Bueno, no es tanto. O sea, que cuando el programa no tiene capacidades, 0x600088C4 contiene el valor 0x00000000. Si el valor es 0xFFFFFFFF entonces el programa tiene todas las capacidades y es todopoderoso. Sólo hay que usar el MetroTRK para cambiarlo.

En otros modelos y otras versiones, estos números cambian.

Cuando pensabas que ya estábamos a punto de terminar, hay más información que necesito compartir. El emulador no sólo sirve para probar aplicaciones, sino que permite testear que se le han otorgado las capacidades que necesita. Supongamos que haces un programa que usa la rutina RFs:SetSubst(). Esta rutina necesita capacidades DiskAdmin que son de tipo sensible . Es decir, necesitas un certificado potente para poder usarla. Si no, tu programa fallará en el móvil.
Pero en el emulador es posible poner un parámetro para que simplemente avise, en lugar de fallar. Por tanto, el programa funcionará en el emulador.

Ojo, que esto está controlado por el parámetro de configuración llamado "PlatSecDiagnostics". Si vale On , entonces se queja pero no falla.

¿Dónde esta la diferencia con el móvil?

Por si no lo he contado, el emulador está generado a partir del mismo código que se usa en los móviles, pero compilado para PC (procesador x86) en vez de para móvil (arquitectura ARM).
O sea, que es más o menos el mismo código fuente, pero distinto código binario. Así que descompilo ekern.exe para PC y pongo un breakpoint en

"DProcess::DoHasCapability". Cuando se dispara, lo analizo paso a paso y me encuentro que, en función del valor de "PlatSecDiagnostics", salta a una rutina que yo llamo "log_missing_capabilities".

- Cuando PlatSecDiagnostics=On , después de saltar, devuelve True
- Cuando PlatSecDiagnostics=Off , no salta, y además devuelve False

Ahora es cuando viene lo bueno: el mísmo código existe en el "ekern.exe" de mi móvil. Sólo que en este caso, PlatSecDiagnostics se almacena en la dirección de memoria 0x64000148. Inicialmente vale 0x1E que es equivalente a False. O sea, que si no hay capacidades, fallará. Lo que tengo que hacer es transformarlo en True, que (avanzando acontecimientos) resulta ser el valor 0x10.

Así que arranco programa.exe que necesita capacidades de las que carece.
Luego lo debugueo para que MetroTRK me permita ver y modificar la memoria.
Voy a la dirección 0x64000148 y pongo 0x10 en vez de 0x1E .
Continuo la ejecución de mi programa, y observo con satisfacción que es capaz de invocar a "RFs:SetSubst()" sin problemas.
No sólo eso, sino que como es un parámetro global, cualquier programa adquiere todas las capacidades.

Es hora de iniciar un explorador de archivos y verificar que tengo acceso a "c:\sys\bin". Puedo leer y escribir con control total. Ahora ya puedo empezar a crear versiones crackeadas de los juegos ! Si eres de los afortunados posedores de un Nokia, seguro que has visto juegos crackeados por el grupo "BiNPDA", por ejemplo los de N-Gage. Ellos han usado esta vulnerabilidad.

Como no todo el mundo puede usar el debugger MetroTRK , elaboro un programa llamado hack_perms_s60v3 que hace exactamente lo mismo. Pero esto es tema para otro artículo.

Ni que decir tiene que la gente de Nokia no estaban muy contentos con esto: les costó mucho esfuerzo desarrollar un sistema de seguridad, y da rabia ver que está roto simplemente porque se les olvidó poner más restricciones en su propio debugger.

Reitero el aviso:

- Ten cuidado con lo que instalas en tu móvil.
  Puedes recibir sorpresas desagradables !!!!!

EOF

@.@----------@.@
 $ Protocolo MetroTRK $ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~{ by FCA00000 }
@.@----------@.@

En este mismo número he contado que es posible sobrepasar la seguridad de teléfonos Symbian usando el debuger MetroTRK. Este debuger se maneja desde un entorno gráfico, normalmente:

- Carbide.c++
- o
- MetroWorks.

Sin embargo, para facilitar que todos los usuarios puedan aprovechar esta vulnerabilidad, he hecho un programa que simula este entorno.

Para ello he necesitado destripar el protocolo usado. Esta es la historia de los acontecimientos.

Las herramientas necesarias son:

1.- Un móvil con Symbian S60v3, por ejemplo N78, E61, N95, 6220classic, ... Yo he usado un N80.

2.- Un ordenador. Yo he usado uno de color negro.

3.- Un cable USB de tipo CA-20 , que une ambos. También es de color negro.

4.- El programa MetroTRK para móviles, conocido como S60_App_TRK.sisx. He usado la versión 2.6

5.- Drivers de Nokia para tratar el puerto USB como si fuera un puerto serie.

6.- El programa Carbide.c++ para PC, versión 1.1 pro.

7.- Un sniffer de puerto serie, por ejemplo Serial Port Monitor de Eltima Software.

8.- Algo para hacer tus propios programas. Yo elegí Python, pero Java o C++ también valdrían.

Lo primero es poner en marcha todo el tinglado: instalar Carbide.c++ en el PC, MetroTRK en el móvil, verificar que el sniffer funciona.

La primera prueba consiste en usar Carbide.c++ en el PC para hacer un programa para el móvil. Existe un directorio con ejemplos que se pueden compilar sin mucho esfuerzo.

Usando este mismo entorno se transfiere el programa al móvil, y hay que aprender a poner breakpoints, leer la memoria, y saber cómo transferir archivos. El siguiente paso es sacar todas las trazas con el sniffer, y ponerse a analizarlas.

Es conveniente sacar varias trazas del mismo proceso. Para ello lo mejor es resetear el entorno completo, hacer una prueba, guardar la traza, y empezar reseteando otra vez. Con esto se consiguen trazas más o menos constantes. En particular esto permite discernir si se usa algún timestamp , dependiendo de la hora a la que se ejecuta la prueba.

A partir de ahora usaré el símbolo '>' para datos desde el PC al móvil, y '<' para la otra dirección. Cada dato es 1 byte en formato hexadecimal.

Veo que el primer comando que Carbide.c++ manda es:

```
> 7E 00 00 FF 7E
```

a lo que el móvil responde:

```
< 7E 80 00 00 7F 7E
```

El siguiente comando es:

```
> 7E 01 01 FD 7E
```

con respuesta

```
< 7E 80 01 00 7D 5E 7E
```

El tercero es:

```
> 7E 05 02 F8 7E
< 7E 80 02 00 7D 5E 00 4F 5F 01 00 00... 80 03 97 7E
```

Bueno, parece que todos los comandos empiezan y acaban con 0x7E . Tanto los que van, como los que vienen.

Si nos fijamos en el tercer byte, parece ser un número secuencial. En la primera trama vale 0x00 , en la segunda vale 0x01, y la tercera dice 0x02.

Más cosas: la respuesta del móvil tiene siempre el segundo byte a valor 0x80. El tercer byte es el mismo número secuencial que se le ha enviado.

Con esto ya puedo sacar las primeras conclusiones:

-> Cada mensaje va dentro de una trama con cabecera 0x7E y finalizada con 0x7E. Esto ayuda a separarlos.

-> El PC manda un mensaje con un identificador secuencial, y el móvil responde con el mismo número. Esto ayuda a saber si un mensaje se ha perdido.

-> El móvil responde con un código 0x80. El PC no parece tener un código asignado para cada pregunta.

Así que empiezo a hacer mi programa.

Simplemente abro el puerto serie y empiezo a mandar exactamente lo mismo que Carbide.c++ . Tras algunos apaños menores empiezo a recibir las mismas respuestas.

```
************************************
* ser = serial.Serial(7)   # COM8  *
* ser.write(chr(0x7E))              *
* ser.write(chr(0x00))              *
* ser.write(chr(0x00))              *
* ser.write(chr(0xFF))              *
* ser.write(chr(0x7E))              *
* s = ser.read(6)                   *
* print ord(s[0])                   *
* print ord(s[1])                   *
* print ord(s[2])                   *
* print ord(s[3])                   *
* print ord(s[4])                   *
```

```
   * print ord(s[5])                    *
   ************************************
```

Ahora tengo que averiguar el significado del resto de los datos. Sospecho que hay varios comandos, en función de lo que se haga desde el entorno gráfico.

Empiezo a hacer varias acciones en Carbide.c++ , tales como:

- -> Leer memoria
- -> Escribirla
- -> Poner breakpoint
- -> Quitarlo
- -> Detener programa
- -> Ejecutar 1 paso
- -> Leer registros
- -> Ponerlos

y veo que manda distintos comandos, identificados por el byte en la posición 1. Por ejemplo, leer memoria en la dirección "0x6404D1E0", resulta en el comando:

```
> 7E 10 03 05 01 00 64_04_D1_E0 00 00 00 00 00 00 00 01 CC 7E
```

Ya sabemos que:

- 7E es la señal de cabecera
- 10 es la orden para leer memoria
- 03 es el número secuencial
- 05 ni idea
- 01 ni idea
- 00 ni idea
- 64 04 D1 E0 es justamente la dirección que quiero leer
- 00 00 00 00 ni idea
- 00 00 00 01 puede que sea el valor 1 , pues quiero leer 1 byte
- CC ni idea
- 7E es la marca de fin

Ahora viene una técnica típica de análisis inverso: voy a mandar un comando ligeramente distinto al anterior. Para ello leo la memoria de la dirección "0x6404D1E4" , que es la misma que antes, 4 bytes más adelante, pero tomo la precaución de resetear el entorno, incluído el móvil. La razón es que quiero que el número secuencial se inicie de nuevo, para que sea 0x03, igual que antes.

Y veo que se manda el comando:

```
> 7E 10 03 05 01 00 64 04 D1 _E4_ 00 00 00 00 00 00 00 01 _C8_ 7E
```

o sea, igual que el comando anterior excepto 2 bytes que han cambiado:

- E4 en lugar de E0 . Normal, pues he leído "0x6404D1E4"
  en lugar de "0x6404D1E0"

- El penúltimo dato es ahora C8, y antes era CC

Notar que CC-C8= 0x4 . Es decir, que ha cambiado en la misma medida que la dirección de memoria.

¿Podría ser la dirección final que hay que leer? Bueno, yo me inclino más por un checksum. La razón es que este penúltimo dato está presente en todos los comandos, y me sorprendería si un protocolo no incluyera algún tipo de verificación.

Así que recupero el comando inicial:

```
   > 7E 00 00 FF 7E
```

y lo modifico ligeramente para mandar un checksum distinto:

```
   > 7E 00 00 F1 7E
```

y recibo:

```
   < 7E FF 07 05 F4 7E
```

que intuyo que significa que ha habido un error.

Bueno, al menos ya sé lo que pasa cuando el protocolo es incorrecto.
Fijaos bien que el comando inicial, eliminando el checksum y los 0x7E de cabecera y final, es: "00 00" y tiene checksum: FF

El segundo mensaje es:

```
   >7E 01 01 FD 7E
```

Si quito también el checksum y los 0x7E, queda: "01 01" que resulta en checksum: FD

El tercero es:

```
   > 7E 05 02 F8 7E
```

que contiene datos: "05 02" y checksum: F8

¿No veis algo aquí? Lo pondré más claro:

```
      00 + 00 + FF = FF
   y
      01 + 01 + FD = FF
   y
      05 + 02 + F8 = FF
```

es decir, que la suma de todos los datos (excepto cabecera y final) tiene que sumar 0xFF. Esto se denomina checksum de complemento: el penúltimo dato es el valor que hace que la suma de todos los bytes de la trama resulte 0xFF.

Lo verifico con otras tramas y parece ser correcto ... excepto en algunos casos.
La verdad es que esto que voy a explicar tiene sentido visto en retrospectiva, pero inicialmente me confundió.

La pregunta clave es:

¿Qué pasa si quiero mandar el dato 0x7E dentro de la trama?

Por ejemplo, si necesito leer la memoria en "0x7E7E7E7E". Si intento mandar manualmente el comando:

```
   > 7E 10 03 05 01 00 7E 7E 7E 7E 00 00 00 00 00 00 00 01 AD 7E
```

creerá que el segundo 0x7E (byte séptimo) indica el final de trama y resultaría la trama parcial:

```
   > 7E 10 03 05 01 00 7E
```

que es incorrecta (el checksum debería ser 00 , que en realidad no es el checksum)

Es decir, que el valor 0x7E no se puede mandar 'tal cual' porque el protocolo se hará un lío. Para eso se usa la técnica de 'escapar' los datos de control.

Para saber cómo se aplica este método, le digo a Carbide.c++ que quiero leer la memoria "0x7E7E7E7E" y resulta el comando:

> 7E 10 03 05 01 00 7D_5E 7D_5E 7D_5E 7D_5E 00 00 00 00 00 00 00 01 ED 7E

o sea, que manda "7D_5E" en vez de "7E".

¿Y si quiero mandar 0x7D? En este caso veo que se mandan como "7D_5D"

Entonces creo que ya sé cómo construir las tramas:

1.- Averiguar el comando, por ejemplo 0x10 para leer la memoria.
2.- Obtener un número secuencial. Ponerlo a continuación.
3.- Poner el resto de parámetros. Para leer la memoria, es necesario la dirección y el tamaño.
4.- Calcular el checksum, sumando todos los valores. Ponerlo al final.
5.- Sustituir 7D_5D en lugar de 7D, y 7D_5E en lugar de 7E
6.- Poner las cabeceras 7E y el finalizador 7E
7.- Mandarlo.
8.- Esperar respuesta.

Ahora queda saber cuales comandos se pueden mandar. Por ejemplo la trama:

> 7E _00_ 00 FF 7E

es la primera que se manda, por lo que podría ser una especie de saludo.

La siguiente es:

> 7E _01_ 01 FD 7E
que no sé lo que significa.

Así que, en mi propio programa, puedo tomar 2 decisiones:

a) mandarlo. No hace daño

b) no mandarlo, y ver qué pasa.

Pruebo la opción b) y parece que todo funciona bien. Posiblemente sea un comando para que Carbide.c++ sepa la versión de MetroTRK que está instalada.

Para aprovechar la vulnerabilidad que encontré en Symbian S60v3 lo que necesito es leer la memoria y escribirla, así que me voy a centrar en esos comandos. Más o menos sé que para leer la memoria uso el comando 0x10 y uno de los datos es la dirección de memoria, escrita como 4 bytes en big-endian.

Ahora lo que me interesa es saber qué datos me envía el teléfono. La respuesta es algo así como:

< 7E 80 03 00 01 00 10 12 34 56 78.....

Ya sé que:

- 7E es la cabecera
- 80 significa que todo ha ido bien
- 03 es el número secuencial de comando

Luego hay otros 4 bytes que indican la cantidad de bytes leídos. Y los datos siguientes son justamente el contenido de la memoria.

Nota: al igual que en los mensajes enviados, es necesario des-escapar los valores "7D_5E" y "7D_5D" .

Por ejemplo, esta trama recibida indica que:

- 0x6404D1E0 contiene valor 0x12
- 0x6404D1E1 contiene valor 0x34
- 0x6404D1E2 contiene valor 0x56
- 0x6404D1E3 contiene valor 0x78

Para saber el comando para escribir en la memoria, le digo a Carbide.c++ que modifique 0x6404D1E0 para el valor 0x33 y el comando es:

> 7E 11 04 08 00 04 64_04_D1_E0 00_00_00_00 00_00_00_01 33 ...

en el que se pueden ver los datos:

- 11 que sirve para escribir en memoria
- 64_04_D1_E0 que es la dirección de destino
- 00_00_00_00 que no sé lo que significa
- 00_00_00_01 para escribir 1 byte
- 33 , que es el valor del byte que quiero escribir

Fácil, ¿no? Ahora sólo tengo que generar este comando en mi programa. Recordar que la dirección a escribir es 0x64000148 con el valor 0x10, así que el comando es:

> 7E 11 04 08 00 04 64_00_01_48 00_00_00_00 00_00_00_01 10 ...

y ya está: simplemente con mandarle estos bytes al móvil, se abren todas las puertas. La protección de Symbian está rota.

Puedes encontrar el resultado buscando hack_perms_s60v3.py en foros sobre teléfonos móviles Nokia.

Una vez hecho el esqueleto, podría haberlo ampliado para hacer un entorno gráfico, o bien integrarlo con mi propio debuger, pero no lo hice porque Carbide.c++ ya es lo suficientemente bueno como para no necesitar mejoras.

Tras publicar este descubrimiento fui contactado por Ilfak Guilfanov, el creador del desensamblador IDA, para compartir ideas. El resultado es:

http://hexblog.com/2008/03/symbian_apptrk.html

Yo también pensé en integrarlo con el debuger GDB de linux, pero dado que no hay compiladores para Symbian que funcionen en Linux, no le veo ningún beneficio.

Más tarde encontré varios documentos y utilidades que habrían hecho mi tarea más fácil:

1.- En Carbide.c++ es posible ver las trazas que se le mandan a MetroTRK. No incluye la cabecera, pero explica los argumentos necesarios para cada comando. En particular explica los distintos códigos de error.

2.- Un documento que explica cómo funciona MetroTRK y cómo adaptarlo a otros procesadores y otros sistemas operativos.

3.- Desensamblé el propio MetroTRK en el móvil para ver qué otros comandos se pueden mandar. Casi todos se pueden probar desde Carbide.c++ pero alguno no, por ejemplo sacar el listado de librerías.

Este conocimiento fue compartido con gente a lo largo del mundo, y sirvió para que algunos crackers elaboraran un sistema para piratear juegos de N-Gage. No digo que esté bien ni mal; sólo lo quería mencionar.

Los fabricantes de MetroTRK, liderados por Nokia y Symbian, se apercibieron de

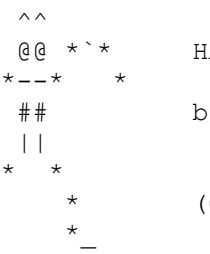
la situación e idearon un mecanismo para anular este truco en las nuevas versiones de firmware.

Este anti-hack consiste en instalar una versión limitada de MetroTRK, lo cual impide instalar la versión vulnerable. Obviamente no afecta a aquellos que ya la han aprovechado para abrir sus móviles, pero impide que nuevos usuarios tengan acceso total a sus propios teléfonos.

A día de hoy se siguen buscando nuevas vulnerabilidades para romper las nuevas protecciones impuestas. Y si una nueva puerta se abre, las empresas involucradas las cerrarán de nuevo.

Así una y otra vez...

EOF



HACK THE WORLD

by blackngel <blackngel1@gmail.com>
<black@set-ezine.org>

---[1 - Prologo

Estas sentand@ en el sofa de tu casa, tu cabeza vacila entre el monitor de television y la pantalla del portatil. Desechas el primero y abres el nuevo manual de criptografia que te acabas de descargar de tu repositorio habitual.

Lees pagina tras pagina, 10, 20, 30 y zas. Cierras la tapa del portatil y te dedicas nuevamente a ver esa telebasura que acabara destruyendo tu mente, esa carcoma que te consumira poco a poco hasta dejarte reducido en la mas profunda de las ineptitudes.

Pero, por que?

Muy facil, llevas toda tu vida leyendo teoria (sobre todo cuando hablamos de criptografia). Las representaciones graficas son un apoyo que te ayudan a avanzar; pero no es suficiente.

Lo que te falta es decir:

1 - Puedo hacerlo.
2 - Voy a hacerlo.

Pues hazlo...

---[2 - Introduccion

Vamos a meterle mano al codigo. Tomaremos de la mano un e-book titulado:

- Una introduccion a la Criptografia [1].

El libro es muy bueno. Con el podras empezar a entender la criptografia desde lo mas basico hasta los metodos y algoritmos mas complejos y actuales.
Todo se explica de forma educativa y con numerosos graficos. Pero hay algo que le falta, y esto no se le puede reprochar a este libro, pues muchos mas (y haberlos hailos) siguen el mismo camino. EL CODIGO.

El ejercicio que vamos a tratar en este texto versa precisamente de como implementar uno de los primeros ejemplos de criptografia basica que se presentan en el libro. Con ello conseguiremos ver por donde se mueve este mundo.

El objetivo es que cuando termines este articulo tengas el valor suficiente como para seguir aprendiendo a base de practica y esfuerzo.

Y ahora solo planteate lo siguiente:

```
if (TIEMPO_LECTURA_Y_OCIO > TIEMPO_PRACTICA_Y_EJERCICIO) {
   printf("\nTu jamas seras un hacker\n");
   exit(LIFE_ERROR);
}
else
   printf("\nEstas en el camino correcto\n");
...
...
...
```

---[3 - Sistema Hayhanen (Simple)

El sistema critografico que se presenta al principio del libro (deberias leerlo), es en realidad una version totalmente simplificada del utilizado por el espia ruso mencionado en el titulo de esta seccion.

Basicamente viene a hacer lo siguiente:

Se escoge un texto plano a cifrar en el que se sustituyen los espacios por puntos. Algo como:

```
ESTAMOS.ENTRANDO.EN.LA.NASA
```

Despues, con la ayuda de una contrase~a y las letras del abecedeario se genera una tabla tal como la siguiente:

```
tabla[3][10];

  1 2 3 4 5 6 7 8 9 0
  - - - - - - - - - -
0|    S E C R T O . A
1|B D F G H I J K L M
2|N ~ P Q U V W X Y Z
```

Es muy sencillo:

1 - Los dos primeros valores de la tabla quedan vacios.
2 - Se coloca la contrase~a sin repetir las letras.
3 - Se coloca el punto '.' que hace de sustituto al espacio.
3 - Se coloca el resto del abecedario por orden y sin repetir.

Se cambian las letras del mensaje por los valores correspondientes en la tabla:

```
E S T A M  O S . E N  T R A N  D  O . E N  . L  A . N  A S A
4 3 7 0 10 8 3 9 4 21 7 6 0 21 12 8 9 4 21 9 19 0 9 21 0 3 0
```

Y aqui la razon de porque los elementos 'tabla[0][1]' y 'tabla[0][2]' se dejan vacios. A la hora de descifrar el mensaje y leer esta secuencia de numeros, si nos encontramos un 1 o un 2 querra decir que debemos tener en cuenta la siguiente cifra. Y esta condicion siempre se cumple, pues ninguna letra de las que ocupan la primera fila poseen este valor individualmente.

Pero no hemos acabado. Ahora entra en juego el valor de la clave. Le damos un valor desde '1' hasta 'x' (la longitud de la clave) por orden alfabetico y teniendo en cuenta que las repeticiones tienen un valor superior segun se encuentren hacia la derecha. Veamoslo en este ejemplo:

```
   S E C R E T O
   6 2 1 5 3 7 4
```

Repetimos esta cadena de numeros hasta completar la longitud del mensaje y sumamos los valores uno a uno (descartando las decenas) de la siguiente manera:

```
      6215374621537462153746215374621537
      4370108394217602112894219190921030
   +  ----------------------------------
      0585472915744064265530424464542567 -> Cifrado
```

Y esta cadena es definitivamente nuestro mensaje cifrado. El proceso inverso para descifrar es sencillo:

```
   1 - Crear la tabla con la clave para descifrar.
   2 - Obtener los valores de la clave y restarselos al mensaje cifrado.
   3 - Buscar en la tabla los caracteres correspondientes a los valores
       que se~alan esta nueva cadena resultante.
```

Ej.: Si nos encontramos un '4' -> Accedemos a tabla[0][4] que es una 'E'

```
     Si nos encontramos un '2' -> Leemos el siguiente digito

                                  |-> Ej.: Si es un '1' -> tabla[2][1] = 'N'
```

Todo esto nos lo explica igual de bien e incluso muchisimo mejor el libro con el que contamos. Y nosotros podriamos pasar al siguiente tipo de cifrado pensando que esto es facil y que ya sabemos como hacerlo. Pero hay algo que en el 'hacking' se tiene muy en cuenta, y es lo siguiente:

```
   - AUN NO LO HAS DEMOSTRADO
```

---[4 - Prueba de Concepto

Este titulo suena a 'exploit' pero nada que se le parezca... Los que habeis leido hasta aqui (solo requiere unos minutos) ya sabeis de que va el tema. Asi que lo que viene a continuacion es el tipico chapuzon a la piscina del codigo fuente.

Pense, en un principio, colocar todo el programa de un golpe y explicarlo en los comentarios; pero alguno podria desistir a la primera de cambio. Es por ello que lo explicare por fases (procedimientos/funciones en este caso) y asi sera mas facil seguir la logica de los pasos que se explicaron en la seccion anterior.

Cuando juntes las piezas podras compilarlo con el comando super-secreto de toda la vida:

```
   $gcc bcrypt.c -o bcrypt
```

Empezamos por lo sencillo:

```
 **-------------------------------------------------------------------------**

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

```
#include <string.h>

int is_diferent(char, int);    /* COMPRUEBA REPETICIONES EN LA TABLA */
void crear_tabla(void);        /* GENERA LA TABLA 3 X 10             */
void imprimir_tabla(void);     /* IMPRIME TABLA PARA PRUEBAS         */
void leer_password(void);      /* GENERA VALORES PARA LA CONTRASE~A  */
void cifrar(void);             /* CIFRA MENSAJE                      */
void descifrar(void);          /* DESCIFRA MENSAJE                   */

char matrix[3][10];   /* TABLA PARA GENERAR CODIGOS                          */
int *valores_c;       /* VALORES CORRESPONDIENTES A LAS LETRAS EN LA TABLA */
int *valores_d;       /* VALORES CIFRADOS QUE VAN A SER DESCIFRADOS        */
int *pass_val;        /* VALORES CORRESPONDIENTES A LA CONTRASE~A          */
char *password;       /* CONTRASE~A                                        */
char *mensaje;        /* MENSAJE                                           */
static int len_pass;  /* LONGITUD CONTRASE~A                               */
static int len_msg;   /* LONGITUD MENSAJE                                  */
static int val = 0;   /* CANTIDAD DE VALORES EN '*valores_c'               */

char alfabeto[] = {'A','B','C','D','E','F','G','H','I','J','K','L','M','N',
                   '~','O','P','Q','R','S','T','U','V','W','X','Y','Z'};

 **--------------------------------------------------------------------------**

   1 - Archivos de cabecera.
   2 - Declaraciones de funciones.
   3 - Declaraciones de variables y matrices (*).

(*) No te preocupes, iras viendo que hacen las diferentes variables muy
    poco a poco en cada funcion. Cada una tiene una mision especifica.

 **--------------------------------------------------------------------------**

int main(int argc, char *argv[])
{
   int cifrado = 0;

   if (argc < 4) {
      fprintf(stderr, "\nUsage: ./bcrypt [-c|-d] message password\n");
      exit(0);
   }

   /* COMPROBAMOS ARGUMENTOS Y USO DEL PROGRAMA */
   if (strncmp(argv[1], "-c", 2) == 0)
      cifrado = 1;
   else if (strncmp(argv[1], "-d", 2) != 0) {
      fprintf(stderr, "\nUsage: ./bcrypt [-c|-d] message password\n");
      exit(0);
   }

   asprintf(&mensaje, "%s", argv[2]);  /* ALMACENAMOS MENSAJE  */
   len_msg = strlen(mensaje);          /* LONGITUD DEL MENSAJE */

   asprintf(&password, "%s", argv[3]); /* ALMACENAMOS CONTRASE~A     */
   len_pass = strlen(password);        /* LONGITUD DE LA CONTRASE~A  */

   if (len_pass >= 10) {               /* EMPEZAREMOS POR LO SENCILLO */
      printf("\nUtilice una contrase~a de 9 caracteres como maximo");
      exit(0);
   }

   leer_password();  /* CONSEGUIMOS LOS VALORES NUMERICOS DE LA CONTRASE~A   */
   crear_tabla();    /* GENERAMOS LA TABLA CORRESPONDIENTE A ESTA CONTRASE~A */
```

```
   if (cifrado)      /* CIFRAR O DESCIFRAR ? */
      cifrar();
   else
      descifrar();

   puts("\n\n");

   free(password);   /* DE VEZ EN CUANDO HAY QUE SACAR LA BASURA !!! */
   free(mensaje);
   free(pass_val);
   free(valores_c);
   free(valores_d);
   pass_val = NULL;
   valores_c = NULL;
   valores_d = NULL;
   password = NULL;
   mensaje = NULL;

   return 0;
}
```

 --

Lo mas importante:

   1 - Almacenamos mensaje y contrase~a.
   2 - Obtenemos valores de la contrase~a y generamos tabla
   3 - Llamamos a la funcion cifrar() o descifrar().

Continuemos:

 --

```
void leer_password(void) {

   int i, w;
   int t = 1;
   char *ptr;

   pass_val = (int *) calloc((len_pass), sizeof(int)); /* PEDIMOS MEMORIA */
   if (pass_val == NULL) {
      fprintf(stderr, "\nNo hay suficiente memoria\n");
      exit(-1);
   }

   ptr = password;                        /* PUNTERO TEMPORAL PARA TRABAJAR */
   for (i = 0; i < len_pass; i++) {
      for (w = 0; w < len_pass; w++) {    /* COMPARAMOS CADA CARACTER CON EL  */
         if (ptr[i] > ptr[w])             /* RESTO, SI ES MAYOR AUMENTAMOS SU */
            t += 1;                       /* VALOR EN 1                       */
      }
      for (w = 0; w < i; w++) {           /* SI ADEMAS HABIA ALGUN CARACTER   */
         if (ptr[i] == ptr[w])            /* IGUAL ANTES QUE ESTE, AUMENTAMOS */
            t += 1;                       /* SU VALOR EN 1                    */
      }
      pass_val[i] = t;                    /* GUARDAMOS EL VALOR EN LA MATRIZ  */
      t = 1;                              /* RESTAURAMOS EL VALOR MINIMO */
   }
}
```

 --

Este codigo seguramente podria mejorarse. Pero los 3 bucles muestra bien el
cometido:

1 - El bucle exterior recorre del primer al ultimo caracter de la contrase~a.
2 - El primer bucle interno establece un valor segun el orden alfabetico.
3 - El segundo bucle interno dice que si ya habia una letra igual antes
de la que se analiza, esta ultima tendra un valor superior en una unidad.

* Fijate bien en el segundo bucle para no llevarte a confusiones. Solo recorre los caracteres que hay antes del que se analiza. Es decir. Para la primera de la contrase~a el bucle no se ejecuta (no hay letras anteriores), para la segunda solo se ejecuta una iteracion y asi sucesivamente...

Ahora vamos con una de las partes mas interesantes y la base del algoritmo:

```
 **--------------------------------------------------------------------------**

void crear_tabla(void) {

   int i = 3;
   int j = 0;
   int z = 0;

   matrix[0][1] = ' ';     /* DOS PRIMEROS ELEMENTOS VACIOS, METEMOS        */
   matrix[0][2] = ' ';     /* ESPACIOS PARA IMPRIMIR LA TABLA CORRECTAMENTE */

   while (*password) {                 /* RECORREMOS LA CONTRASE~A Y MIRAMOS */
      if (is_diferent(*password, 0)) { /* SI EL CARACTER YA ESTA EN LA TABLA */

         if (i == 10) {                /* DE ULTIMO SE RELLENA EL ELEMENTO  */
            matrix[j][0] = *password++; /* '0' DE CADA FILA DE LA TABLA      */
            j += 1;                    /* RECUERDA: 1-2-3-4-5-6-7-8-9-0     */
            i = 1;                     /* Y PASAMOS A LA SIGUIENTE FILA     */
         }
         else {                        /* SINO                              */
            matrix[j][i] = *password++; /* ESCRIBIMOS EL CARACTER SIN MAS    */
            i += 1;
         }
      }
      else              /* SI EL CARACTER YA ESTABA GUARDADO EN LA TABLA */
         *password++;   /* PASAMOS DIRECTAMENTE AL SIGUIENTE             */
   }

   matrix[j][i] = '.';  /* DESPUES DE LA CONTRASE~A ALMACENAMOS EL PUNTO */
   i += 1;

   for (z = 0; z < 27; z++) {              /* Y AHORA METEMOS EL RESTO DE */
      if (is_diferent(alfabeto[z], 0)) {   /* LAS LETRAS DEL ALFABETO     */
         if (i == 10) {                    /* UTILIZANDO PRACTICAMENTE LA */
            matrix[j][0] = alfabeto[z];    /* MISMA TECNICA QUE ANTES.    */
            j += 1;
            i = 1;
         }
         else {
            matrix[j][i] = alfabeto[z];
            i += 1;
         }
      }
   }

   imprimir_tabla();     /* SOLO PARA LA ETAPA DE PRUEBAS */
}

 **--------------------------------------------------------------------------**
```

Entre los comentarios y el ejemplo anteriormente presentado tienes mas que de

sobra para entender lo que acabamos de hacer.

Lo siguiente queda bonito en el proceso de pruebas. Comprobaremos asi si las tablas se crean correctamente.

```
 **--------------------------------------------------------------------------**

void imprimir_tabla(void) {

   printf("\n1 2 3 4 5 6 7 8 9 0");
   printf("\n- - - - - - - - - -");

   printf("\n%c %c %c %c %c %c %c %c %c %c\n", matrix[0][1],
                                              matrix[0][2],
                                              matrix[0][3],
                                              matrix[0][4],
                                              matrix[0][5],
                                              matrix[0][6],
                                              matrix[0][7],
                                              matrix[0][8],
                                              matrix[0][9],
                                              matrix[0][0]);

   printf("\n%c %c %c %c %c %c %c %c %c %c\n", matrix[1][1],
                                              matrix[1][2],
                                              matrix[1][3],
                                              matrix[1][4],
                                              matrix[1][5],
                                              matrix[1][6],
                                              matrix[1][7],
                                              matrix[1][8],
                                              matrix[1][9],
                                              matrix[1][0]);

   printf("\n%c %c %c %c %c %c %c %c %c %c\n", matrix[2][1],
                                              matrix[2][2],
                                              matrix[2][3],
                                              matrix[2][4],
                                              matrix[2][5],
                                              matrix[2][6],
                                              matrix[2][7],
                                              matrix[2][8],
                                              matrix[2][9],
                                              matrix[2][0]);

   puts("\n");
}

 **--------------------------------------------------------------------------**
```

Bueno, lo anterior no requiere mas explicacion no? Bien ordenadita ademas para que podais seguir el ejemplo de la seccion anterior.

Ahora vamos a ver una porcion de codigo interesante. En realidad tiene dos cometidos, pero he utilizado la estructura para no repetir mas codigo. Sus funciones tienen que ver con el metodo anterior y el que le sigue, por eso lo he situado en este lugar.

```
 **--------------------------------------------------------------------------**

int is_diferent(char c, int op) {

   int i, j;
   static int x = 0;
```

```
   for (i = 0; i < 3; i++) {
      for (j = 0; j < 10; j++) {
        if (matrix[i][j] == c)       /* ENCONTRAMOS UN CARACTER EN LA TABLA  */
           if (op) {
              if (i == 0) {          /* SI ESTA EN LA PRIMERA FILA EL VALOR  */
                 valores_c[x] = j;   /* ES EL CORRESPONDIENTE A LA COLUMNA   */
                 x += 1; val += 1;
              }
              else {                 /* SI ESTA EN LA PRIMERA O SEGUNDA FILA */
                 valores_c[x] = i;   /* EL VALOR DE LA LETRA ES EL NUMERO DE */
                 x += 1; val += 1;   /* LA FILA SEGUIDO POR EL NUMERO DE LA  */
                 valores_c[x] = j;   /* COLUMNA                              */
                 x += 1; val += 1;
              }
           }
           else
              return 0; /* FALSE - HEMOS ENCONTRADO UN CARACTER IGUAL  */
      }
   }
   return 1; /* TRUE - SI LLEGAMOS AL FINAL DECIMOS QUE ES DIFERENTE   */
}
```

 --

Para aclararnos, vamos a separar las dos acciones de este metodo:

   1 - Cuando OP = 0:

       -> Comprueba si un caracter se encuentra ya almacenado dentro
          de la tabla ('matrix[][]'), asi nunca los repetiremos.

   2 - Cuando OP = 1:

       -> Buscamos los caracteres del mensaje a cifrar en la tabla y si los
          encontramos guardamos el valor correspondiente a su posicion.

          Ejemplo anterior: Si buscamos una 'E' la encontramos en la pirmera
                            fila y guardamos su numero de columna: '4'.

                            Si buscamos una 'N' la encontramos en la ultima
                            fila y guardamos su numero de fila y columna: '21'.

Y aqui una de las esperadas funciones. Se encarga de hacer la suma entre los
valores del mensaje a cifrar y la contrase~a e imprime el resultado por
pantalla.

 --

```
void cifrar(void) {

   int i;
   int tmp;

   valores_c = (int *) calloc((len_msg), sizeof(int));
   if (valores_c == NULL) {
      fprintf(stderr, "\nNo hay suficiente memoria\n");
      exit(-1);
   }

   for (i = 0; i < len_msg; i++) { /* SEGUNDA FUNCION DE is_diferent()        */
      is_diferent(mensaje[i], 1);  /* OBTENEMOS LOS VALORES NUMERICOS DEL MSG */
   }
```

```
   printf("\n-----BEGIN BLACK-CRYPT MSG-----\n");
   for (i = 0; i < val; i++) {
      tmp = valores_c[i] + pass_val[i % len_pass]; /* MODULO LONGITUD MSG */
      printf("%d", tmp % 10); /* IMPRIMIMOS SOLO LAS UNIDADES */

   }
   printf("\n------END BLACK-CRYPT MSG------\n");
}
```

Si hay algo a destacar en la funcion anterior es la utilizacion del 'operador modulo (%)'. Gracias a el nos evitamos crear una cadena igual de larga que el mensaje con los valores de la contrase~a repetidos; pues hace esta misma funcion, vulgarmente reinicia cada vez que alcanza el final.

Seguidamente nos vuelve a servir de ayuda para tomar solo las unidades en caso de que el resultado de la operacion anterior sea 10 o superior a esta cantidad. Nos quitamos de encima asi la necesidad de utilizar algo como:

```
   if (tmp >= 10)             /* SI LA SUMA ES SUPERIOR A 9   */
      printf("%d", tmp - 10); /* IMPRIMIMOS SOLO LAS UNIDADES */
   else                       /* EN CASO CONTRARIO            */
      printf("%d", tmp);      /* IMPRIMIMOS EL RESULTADO      */
```

Y ya por ultimo, como obtener el mensaje a partir de una cadena cifrada:

```
void descifrar(void) {

   int i, j;

   valores_c = (int *) calloc((len_msg), sizeof(int)); /* PEDIMOS MEMORIA */
   valores_d = (int *) calloc((len_msg), sizeof(int)); /* PUEDO REPETIR ? */
   if (valores_c == NULL || valores_d == NULL) {
      fprintf(stderr, "\nNo hay suficiente memoria\n");
      exit(-1);
   }

   for (i = 0; i < len_msg; i++) {    /* CUTRE - PASAMOS LA CADENA CIFRADA */
      valores_d[i] = mensaje[i] - 48; /* A UNA MATRIZ DE VALORES NUMERICOS */
   }

   for (i = 0; i < len_msg; i++) {   /* RESTAMOS A LA MATRIZ DE VALORES      */
      j = pass_val[i % len_pass];    /* LOS VALORES PROPIOS DE LA CONTRASE~A */
      if (j > valores_d[i])                    /* NOS ASEGURAMOS DE OBTENER  */
         valores_c[i] = valores_d[i] + 10 - j; /* SIEMPRE UN VALOR POSITIVO  */
      else
         valores_c[i] = valores_d[i] - j;
   }

   printf("\n-----BEGIN BLACK-DECRYPT MSG-----\n");
   for (i = 0; i < len_msg; i++) {
      if (valores_c[i] == 1) {                       /* SI ENCONTRAMOS UN '1' */
         printf("%c", matrix[1][valores_c[i + 1]]); /* TENEMOS EN CUENTA EL  */
         i += 1;                                    /* SIGUIENTE DIGITO      */
      }
      else if (valores_c[i] == 2) {                 /* SI ENCONTRAMOS UN '2' */
         printf("%c", matrix[2][valores_c[i + 1]]); /* TENEMOS EN CUENTA EL  */
         i += 1;                                    /* SIGUIENTE DIGITO      */
      }
      else                                          /* EN CASO CONTRARIO     */
          printf("%c", matrix[0][valores_c[i]]);    /* IMPRIMIMOS EL CARACTER*/
```

```
   }                                                    /* EN ESA POSICION       */
   printf("\n------END BLACK-DECRYPT MSG------\n");
}
```

```
 **-------------------------------------------------------------------------**
```

Pues nada, lo visto. Ya hemos explicado suficientemente el metodo. Los pasos son los contrarios al proceso de cifrado.

No ha sido tan dificil verdad?

ERRORES A CONSIDERAR:

Uno de los errores principales de este programa es que podria fallar de una forma catastrofica si utilizamos e~es en la contrase~a. No siempre tiene que ocurrir, pero no es recomendable usarlas, y la razon es la siguiente:

   -> Al no ser un caracter ASCII estandar, el programa lo interpretara en
      realidad como dos valores que son '-61' y '-111'. Esto provoca errores
      en la tabla de 3 x 10 y ya de entrada interpreta mal la longitud de
      la contrase~a pensando (que pensara un PC?) que es mayor.

Por otro lado, prueba a permitir longitudes de clave mas grandes y observa los resultados.

EN RESUMEN:

   - ANALIZA EL CODIGO
   - ESTUDIA EL CODIGO
   - Y ADAPTA EL CODIGO

---[5 - Algoritmos Basicos

Ahora mostrare algunos de los algoritmos mas basicos de la criptografia antigua. El motivo de situarlos aqui es muy sencillo, de haberlos colocado al principio, el articulo hubiera perdido todo su interes.

CIFRADO DEL CESAR

```
 **-------------------------------------------------------------------------**
```

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

int main(int argc, char *argv[]) {

   int i, tmp;
   int clave;
   int cifrado = 0;
   int longitud;
   char *mensaje;

   if (argc < 4) {
      fprintf(stderr, "\nUsage: ./cesarcrypt [-c|-d] mensaje clave\n");
      exit(0);
   }

   if (strncmp(argv[1], "-c", 2) == 0)
```

```
      cifrado = 1;
   else if (strncmp(argv[1], "-d", 2) != 0) {
      fprintf(stderr, "\nUsage: ./cesarcript [-c|-d] mensaje clave\n");
      exit(0);
   }

   mensaje = argv[2];
   clave = atoi(argv[3]);

   longitud = strlen(mensaje);

   if (cifrado) {
      printf("\n-----BEGIN CESAR-CRYPT MSG-----\n");
      for (i = 0; i < longitud; i++) {
         tmp = mensaje[i] + clave;
         printf("%c", tmp);
      }
      printf("\n------END CESAR-CRYPT MSG------\n");
   } else {
      printf("\n-----BEGIN CESAR-DECRYPT MSG-----\n");
      for (i = 0; i < longitud; i++) {
         tmp = mensaje[i] - clave;
         printf("%c", tmp);
      }
      printf("\n------END CESAR-DECRYPT MSG------\n");
   }

   puts("\n");

   return 0;
}
```

EXPLICACION: Lo de toda la vida, desplazamos las letras en el alfabeto tantas
posiciones como indique la clave y realizamos la sustitucion. En
este caso logramos lo mismo sumando o restando la clave a cada
uno de los caracteres ASCII que componen el mensaje.

CIFRADO XOR

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

int main(int argc, char *argv[])
{
   int car;
   FILE *fin, *fout;

   if (argc < 4)
      fprintf(stderr, "\nUsage: ./xorcript filein.txt fileout.txt clave\n");
      exit(0);
   }

   if ((fin = fopen(argv[1], "rb")) == NULL) {
      printf("\nNo se pudo abrir: %s.\n", argv[1]);
   }
   if ((fout = fopen(argv[2], "wb")) == NULL) {
      printf("\nNo se pudo abrir: %s.\n", argv[2]);
```

```
   }

   while ((car = getc(fin)) != EOF) {
      car ^= *argv[3]; /* MAGIA */
      putc(car, fout);
   }

   printf("\nLa operacion XOR se ha completado correctamente\n");
   printf("\nResultado XOR guardado en %s.\n", argv[2]);

   fclose(fin);
   fclose(fout);

   return 0;
}
```

```
 **--------------------------------------------------------------------------**
```

EXPLICACION: El cifrado mas simple y famoso de la historia. Un xor '^' podria definirse como una funcion asi:

```
   xor(texto_claro, clave);   /* CIFRAR    */
   xor(texto_cifrado, clave); /* DESCIFRAR */
```

Pero por suerte lo tenemos todo junto en un operador. Un mensaje cifrado con una clave mediante XOR se descifra ejecutando otro XOR con la misma clave. Es toda una funcion de cifrado en miniatura.

---[6 - Analisis de Frecuencias

En el libro ya mencionado anteriormente se expone un metodo acerca de como descifrar un texto que ha sido cifrado mendiante un algoritmo de sustitucion.

Se entiende muy clarito y se pueden seguir los pasos manualmente sin mayor problema. Pero nosotros tenemos otro objetivo:

   - AUTOMATIZAR UNA PARTE DE LA TAREA

Para ello trataremos de implementar en un lenguaje de programacion cualquiera la primera parte del criptoanalisis, que es precisamente el analisis de frecuencias para averiguar en que idioma esta escrito un texto cifrado por sustitucion.

En mi caso "C" para seguir con la tradicion.

Pero vamos al asunto. Copiare aqui el texto cifrado que se expone en el libro para que no tengas que acudir a el continuamente interrumpiendo la explicacion:

```
   HS2BHF7JT7207HS2B9C722SJ47JT72MP7BN77JMP7H92BS2926
   929J6SNMP72FMP7JS17N7B7H967J96SJ7NN7170FS9J9097J7H
   070SK9NS29TS0S2CP19J3HS2192170FT9J92SH922S8N7H926S
   8N729198H727JTN9K98H72BS292MP7H72HH7J9JSH72Q9BF9JH
   9QF097JT7N9D93MPF7J6SJ79H2FH7JBFSPJ90719J2SK90SN07
   F16N7BF29N7BSN09BFSJ3D93T918F7JMPF7JD9B7171SNF9BSJ
   H9B9N9982SNT937JH9B9N96FJT90S7H472TS07H9872TF9NPFJ
   07H991SNS292P6HFB9JT7872TF9B9J2909H919JS2P57T9J0SH
   9CN7JT737H1FN9NHH7JS07919N4PN9BS1SPJ19N7JB9H190S
```

(*) En el texto original las 'T's son 'e~es' pero las he sustituido para evitar el problema que ya hemos comentado. Puedo permitirme este lujo gracias a que es un cifrado por sustitucion monoalfabetico.

En otro caso podria destrozar por completo el mensaje.

Bien, lo que se nos pide es que analicemos la frecuencia de los caracteres que componen el texto y que comparemos los porcentajes resultantes con las tablas de frecuencia correspondientes a los idiomas:

- ESPA~OL, INGLES, FRANCES Y ALEMAN

Podrian ser tantos como quisieras, pero para el ejemplo que nos toca son mas que suficientes.

El codigo es bruto (todo en main(...)) y no esta depurado, pero aqui lo teneis. Esta es la implementacion correspondiente:

```
 **------------------------------------------------------------------------**

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

float car_val[26][2];           /* NUMERO DE APARICIONES DE CADA CARACTER   */
float freq[26][2];              /* FRECUENCIA DE APARICION DE CADA CARACTER */
float frecuencias[4][26][2] = { /* LAS FRECUENCIAS DE LOS 4 IDIOMAS         */

                        /* ESPA~OL */
                        { {69, 13.676}, {65, 12.529}, {79, 8.684}, {83, 7.980},
                          {82, 6.873}, {78, 6.712}, {73, 6.249}, {68, 5.856},
                          {76, 4.971}, {67, 4.679}, {84, 4.629}, {85, 3.934},
                          {77, 3.150}, {80, 2.505}, {66, 1.420}, {71, 1.006},
                          {89, 0.895}, {86, 0.895}, {81, 0.875}, {72, 0.704},
                          {70, 0.694}, {90, 0.523}, {74, 0.443}, {88, 0.221},
                          {87, 0.023}, {75, 0.004} },

                        /* INGLES */
                        { {69, 13.105}, {84, 10.468}, {65, 8.151}, {79, 7.995},
                          {78, 7.098}, {82, 6.832}, {73, 6.345}, {83, 6.101},
                          {72, 5.259}, {68, 3.788}, {76, 3.389}, {70, 2.924},
                          {67, 2.758}, {77, 2.536}, {85, 2.459}, {71, 1.994},
                          {89, 1.982}, {80, 1.982}, {87, 1.539}, {66, 1.440},
                          {86, 0.919}, {75, 0.420}, {88, 0.166}, {74, 0.132},
                          {81, 0.121}, {90, 0.077} },

                        /* FRANCES */
                        { {69, 17.564}, {65, 8.147}, {83, 8.013}, {73, 7.559},
                          {84, 7.353}, {78, 7.322}, {82, 6.291}, {85, 5.991},
                          {76, 5.783}, {79, 5.289}, {68, 4.125}, {67, 3.063},
                          {77, 2.990}, {80, 2.980}, {86, 1.557}, {81, 1.361},
                          {71, 1.051}, {70, 0.959}, {66, 0.876}, {72, 0.721},
                          {74, 0.598}, {88, 0.350}, {89, 0.116}, {90, 0.072},
                          {75, 0.041}, {87, 0.020} },

                        /* ALEMAN */
                        { {69, 16.693}, {78, 9.905}, {73, 7.812}, {83, 6.765},
                          {84, 6.742}, {82, 6.539}, {65, 6.506}, {68, 5.414},
                          {72, 4.064}, {85, 3.703}, {71, 3.647}, {77, 3.005},
                          {67, 2.837}, {76, 2.825}, {66, 2.566}, {79, 2.285},
                          {70, 2.044}, {75, 1.879}, {87, 1.396}, {86, 1.096},
                          {90, 1.002}, {80, 0.944}, {74, 0.191}, {81, 0.055},
                          {89, 0.032}, {88, 0.022} }

                        };

int main(int argc, char *argv[]) {
```

```
char car;       /* Caracteres leidos del archivo */
int bytes;      /* Tama~o del texto cifrado */
int tmp1, tmp2; /* Para el algoritmo de la burbuja, valores temporales */
int i = 0;
int j = 0;      /* Variables para bucles */
int w = 0;
float cdif;     /* Distancias entre frecuencias del texto cifrado */
float esp_freq, eng_freq, fra_freq, ale_freq;
FILE *fin;

if (argc < 2) {
   fprintf(stderr, "\nUsage: ./freq archivo_cifrado\n");
   exit(0);
}

/* UTILIZAMOS UN ARCHIVO CON TEXTO CIFRADO COMO ARGUMENTO */
fin = fopen(argv[1], "r");
if (fin == NULL) {
   fprintf(stderr, "\nError en la apertura del fichero\n");
   exit(0);
}

/* BUSCAMOS NUMEROS EN EL TEXTO Y CUANTAS VECES APARECEN */
for (i = 48; i < 58; i++) {
   car_val[w][0] = i;
   while ((car = fgetc(fin)) != EOF) {
      if ( car == (char)i)
         j += 1;
   }
   if (j) {
      car_val[w][1] = j;
      w += 1;
   }
   fseek(fin, 0, 0);
   j = 0;
}

/* BUSCAMOS LETRAS EN EL TEXTO Y CUANTAS VECES APARECEN  */
/* APROVECHAMOS PARA OBTENER EL TAMA~O DEL TEXTO CIFRADO */
int docount = 1;
for (i = 65; i < 91; i++) {
   car_val[w][0] = i;
   while ((car = fgetc(fin)) != EOF) {
      if (docount) {
         if (car != '\n' && car != '\r')
            bytes += 1;
      }
      if (car == (char)i)
         j += 1;
   }
   docount = 0;
   if (j) {
      car_val[w][1] = j;
      w += 1;
   }
   fseek(fin, 0, 0);
   j = 0;
}

fclose(fin);

/* AL FINAL, EN LA PRIMERA COLUMNA DE car_val[][] QUEDAN ALMACENADOS LOS  */
/* CARACTERES ENCONTRADOS Y EN LA SEGUNDA EL NUMERO DE VECES QUE APARECEN */
```

```
printf("\n-----------");
printf("\nAPARICIONES");
printf("\n-----------\n");

/* APLICAMOS EL ALGORITMO DE LA BURBUJA PARA ORDENAR DE MAYOR A MENOR */
for (j = 0; j < 26; j++) {
   for (i = j+1; i < 26; i++) {
      if (car_val[j][1] < car_val[i][1]) {
         tmp1 = car_val[j][0];
         tmp2 = car_val[j][1];
         car_val[j][0] = car_val[i][0];
         car_val[j][1] = car_val[i][1];
         car_val[i][0] = tmp1;
         car_val[i][1] = tmp2;
      }
   }
   if (car_val[j][1] > 0) {
      printf("\n- %c - tiene: %d apariciones", (int)car_val[j][0],
                                               (int)car_val[j][1]);
   }
}

printf("\n\nPulse [INTRO] para analizar frecuencias\n");
getchar();

printf("\n-----------");
printf("\nFRECUENCIAS");
printf("\n-----------\n");

/* UNA REGLA DE 3 SIMPLE PARA OBTENER LOS PORCENTAJES            */
/* SI EN 'LONGITUD FICHERO' HAY 70 CARACTERES '9' EN 100 HAY ... */
for (i = 0; i < 26; i++) {
   freq[i][0] = car_val[i][0];
   freq[i][1] = (car_val[i][1] * 100.0) / bytes;
   if (freq[i][1] > 0.0) {
      printf("\n- %c - tiene freq: %f", (int)freq[i][0], freq[i][1]);
   }
}

/* Y POR ULTIMO CALCULAMOS LA DISTANCIA ENTRE LAS FRECUENCIAS DEL TEXTO   */
/* CIFRADO Y LAS DE LAS TABLAS DE FRECUENCIA PARA CADA UNO DE LOS IDIOMAS */
for (j = 0; j < 4; j++) {
   for (i = 0; i < 26; i++) {

      /* FORMULA MAGICA: SUMA DE LOS CUADRADOS DE LAS DIFERENCIAS */
      if (freq[i][1] > 0.0){
         cdif = cdif +  pow((freq[i][1] - frecuencias[j][i][1]), 2);
      } else {
         cdif = cdif + pow((0.0 - frecuencias[j][i][1]), 2);
      }
   }
   switch (j) {            /* Esto se vuelve extremadamente ineficiente */
      case 0:              /* a cuantos mas idiomas sean analizados.    */
         esp_freq = cdif;  /* Se ha realizado de este modo para que sea */
         break;            /* didacticamente mas comprensible, pero el  */
      case 1:              /* 'modus operandi' debe ser rediseñado.     */
         eng_freq = cdif;
         break;
      case 2:
         fra_freq = cdif;
         break;
      case 3:
         ale_freq = cdif;
         break;
```

```
         default:
            break;
      }
      cdif = 0;
   }

   printf("\n\nDISTANCIA DE FRECUENCIAS");
   printf("\n------------------------");
   printf("\nESPA~OL: %f", esp_freq);
   printf("\nINGLES: %f", eng_freq);
   printf("\nFRANCES: %f", fra_freq);
   printf("\nALEMAN: %f", ale_freq);

   puts("\n\n");

   return 0;
}
```

```
 **-------------------------------------------------------------------------**
```

Observaras algo interesante si has leido el libro. Las frecuencias que nosotros obtenemos son las siguientes:

```
   ESPA~OL: 10.815223
   INGLES: 25.101366
   FRANCES: 39.845760
   ALEMAN: 24.572540
```

Las del libro son:

```
   ESPA~OL: 10.815
   INGLES: 25.108
   FRANCES: 39.837
   ALEMAN: 24.545
```

Vemos pues que acertamos de pleno para el espa~ol; pero existe una diferencia en los decimales de las frecuencias de los otros tres idiomas. Apenas son significativas pero:

- Hemos cometido un error o los autores hicieron las cuentas a mano?

Bueno, y a partir de aqui podeis seguir automatizando tareas, incluso con un poco de cabeza podriais lograr un codigo que os acercase practicamente a la solucion.

Fijate que todavia puedes:

- REALIZAR BUSQUEDAS DE BIGRAMAS

  (En espa~ol) -> ES EN EL DE LA OS UE AR RA RE ON ER AS ST AL AD TA CO OR

  (En ingles) -> TH HR IN ER AN RE ED ON ES ST EN AT TO NT HA ND OU EA NG
  AS OR TI IS ET IT AR TE SE IE OF HE

- REALIZAR BUSQUEDAS DE TRIGRAMAS

  (En espa~ol) -> QUE EST ARA ADO AQU CIO DEL NTE EDE OSA PER NEI IST SDE

  (En ingles) -> THE ING AND HER ERE ENT THA NTH WAS ETH FOR DTH HIS

Ya sabes... si te ha gustado tienes deberes. De todos modos siempre puedes (y debes) implementar lo hasta aqui descrito segun tu manera de ver las cosas.

Y ahora, dos apuntes. Si quieres aprender mas acerca del criptoanalisis practico de varios metodos de cifrado, te recomendaria echases un vistazo a la siguiente presentacion, de la mano de P. Caballero Gil [3]. La primera parte resultara interesantisima para aquellos que hayan gozado con la lectura de los "Relatos" de Edgar Allan Poe (libro que yo aprovechaba para leer en clase de matematicas };-D).

Y segundo, para los mas vagos, no ocultare que este trabajo ya ha sido realizado con anterioridad. Que esperabais? Podeis obtener una lectura interesantisima visitando el siguiente enlace [4].

En el texto mencionado se citan dos programas que han sido realizados hace tiempo para analizar el idioma en que esta escrito un texto. Son los siguientes:

- TEXTCAT [5] -> Identifica 76 idiomas. (1)

- LEXTEXT LANGUAGE IDENTIFIER [6] -> Identifica 260 idiomas. (2)

(1) TextCat esta escrito en 'perl', puedes bajarte el codigo fuente. Comprobaras asi que utiliza un metodo bastante diferente del que nosotros modestamente hemos diseñado.

(2) Este segundo programa va mucho mas alla, ya que nos proporciona un API mediante el cual podemos agregar a nuestros programas la capacidad de identificacion de lenguajes sin apenas realizar el mas minimo esfuerzo. En la misma web encontraras una buena documentacion acerca del SDK para realizar tus desarrollos. Con solo 7 funciones, el mundo a tus pies...

Si ahora se te ocurre preguntar porque diablos hemos llevado el esfuerzo de realizar el trabajo a mano, entonces es que no sabes en que mundo te has metido, y todavia necesitas recapacitar si te encuentras en el lugar idoneo.

Tu querias aprender, no es cierto?

---[7 - CrypTooL - CripTooLinux

Si lo que de verdad necesitas es una herramienta para probar algoritmos criptograficos, entonces lo que necesitas es CrypTool [2].

Para definirla tomare una breve descripcion de Kriptopolis[7]:

"CrypTool, un software destinado a facilitar el aprendizaje de la Criptologia, que viene siendo desarrollado desde 1998 por Bernhard Esslinger, aunque tras el proyecto estan Deutsche Bank, la Universidad de Siegen y TU Darmstadt.

Es software libre, pero de momento solo funciona bajo Windows, aunque ya se trabaja en una nueva version 2.0 basada en Java y que -por tanto- sera multiplataforma. No obstante, existe tambien CrypTooLinux, un port para Linux, basado en QT4...

El software esta disponible en ingles, aleman y polaco. Dispone de ayuda interactiva y abundantes presentaciones 3D sobre criptologia clasica y moderna, teoria de numeros, etc. Segun el autor no se necesitan grandes conocimientos matematicos ni criptograficos para utilizarlo.

Solo el documento de presentacion[8] (mas de 100 paginas, pdf) ya vale su peso en oro."

El mejor consejo: Descargatelo y pruebalo.

Y SI, has leido bien, es SOFTWARE LIBRE, que para empezar viene a decir que

puedes aburrirte a leer codigo fuente hasta el fin de los dias.

---[8 - Conclusion

Y hasta aqui te ofrezco de momento. Ya has visto que todo en este mundo se consigue paso a paso, es la causalidad (causa y efecto), primero el concepto y despues la realidad.

Toda idea puede convertirse en algo tangible, o al menos en algo que puedas sentir. Cuando seas capaz de sentir el software, cuando levantes la cabeza del monitor y te des cuenta que hay dos realidades diferentes que intentan fundirse en una y cuando te resulte practicamente imposible decidirte por una de ellas, entonces estaras preparad@ para ver mas alla.

Y recuerda:

- La seguridad de un sistema criptografico no debe depender del secreto del algoritmo. La seguridad solo debe depender del secreto de la clave.

---[9 - Referencias

[1] Una Introduccion a la Criptografia
http://www.criptored.upm.es/descarga/UnaIntroduccionCriptografia.zip

[2] CrypTool
http://www.cryptool.com

[3] Criptoanalisis
http://webpages.ull.es/users/cryptull/Cripto/Apuntes/Criptoanalisis.pdf

[4] Criptologia, Entropia y Evolucion
http://www.criptored.upm.es/guiateoria/gt_m720a.htm

[5] TextCat
http://odur.let.rug.nl/~vannoord/TextCat/

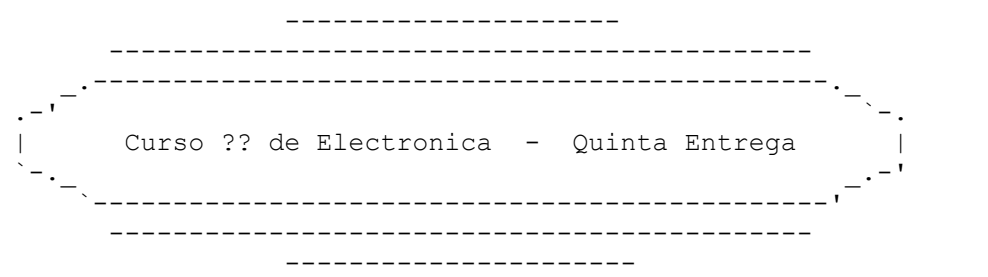
[6] Lextek Language Identifier
http://www.lextek.com/langid/

[7] Kriptopolis
http://www.kriptopolis.org

[8] Documento de Presentacion CrypTool
http://www.cryptool.com/downloads/CrypToolPresentation_1_4_10_en.pdf

EOF

-[0x07]--
-[Curso de electronica 05]---
-[by elotro]---SET-35--



Hola hola lectorcirijillos!!

Como se habran dado cuenta, no tengo mucho para hacer (bah,si tengo, pero soy vago), asi que me he dedicado a torturar sus mentes con esta funesta ciencia llamada electronica.
Espero q se hayan divertido (?) con el articulo anterior, y que hayan ensanblado algunos circuitos.
En esta humilde entrega... Amplificadores Operacionales

Comencemos...

Curso de Electronica - Quinta Entrega - Amplificador operacional 741
Indice de contenidos

#%
1 - Introduccion [aaaaaaadentro!!]
#%

El amplificador operacional (en adelante, op-amp) es un tipo de circuito integrado que se usa en un sinfin de operaciones, principalmente en operaciones analogicas, aunque tambien lo podemos encontrar en algunas aplicaciones digitales.

Porque c4raj0 es tan popular?
Es simple, estable, de facil uso y entendimiento, y por sobre todo, barato; de modo que se lo considera un componente electronico mas, en vez de una asociacion de componentes.

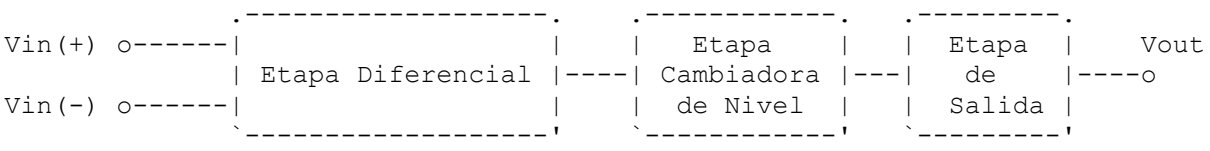
El objetivo de este articulo es que entendamos (si, yo tambien) como funciona un op-amp, y hagamos algunas practicas con el op-amp 741, que es uno de los mas usados por su caracteresticas y su relacion precio-calidad.

#%#%#%#%#%#%#%#%#%#%#%#%#%
2. 741 - Descripcion Basica
#%#%#%#%#%#%#%#%#%#%#%#%#%

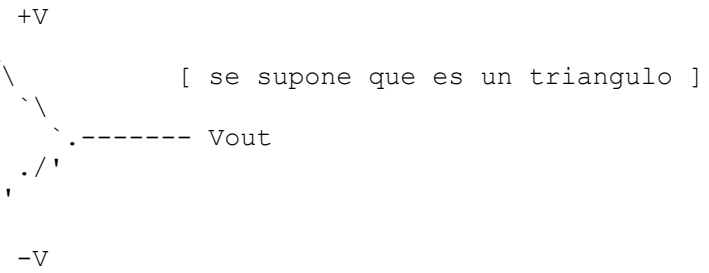
El op-amp 741 tiene dos terminales de entrada, Vin(+) y Vin(-); y un terminal de salida unico llamado Vout.

El 741 necesita dos tensiones de alimentacion +V y -V, que pueden variar entr 5 V y 18 V. En los circuitos que salgan en el articulo, vamos a trabajar con una fuente partida de 12 V, a menos de que indique otra cosa.

Los dos ultimos terminales se suelen demoninar como un 'ajuste de centrado' y sirven para compensar los pequenyos desequilibrios [que palabrota] internos del circuito que veremos mas adelante.
[nada es perfecto...]



Para hacer las cosas mas simples, en los esquemas no se suelen mostrar los terminales de alimentacion y ajuste de centrado, y solamente aparecen los pines de entrada y salida.
El simbolo de un amplificador operacional es algo como esto :



Para hacer las cosas faciles, podemos decir que un op-amp es un amplificador diferencial de alta ganancia [es lo mas facil q pude :)] acompanyado con etapas adicionales, para que podamos conseguir una salida que varie entre los valores positivo y negativo de la alimentacion.

[espero que se haya entendido algo]

Lo principal de un amplificador diferencial es que amplifica la diferencia de las tensiones aplicadas a los terminales de entrada Vin(+) y Vin(-) que se llaman 'Entrada No Inversora' y 'Entrada Inversora'.

La relacion entre las tensiones de la entrada y la salida es :

$$Vout = Ao * (Vin(+) - Vin(-))$$

donde Ao es el valor de la ganancia en bucle abierto (sin realimentacion o feedback). Esta ganancia es de muy alto valor, de 200.000 (doscientos mil) para el 741.

[unos 106 dB]

El amplificador operacional tiene unas caracteristicas bastante importantes que hay que tener en cuenta, que lo hacen un buen amplificador de tension. Posee una (impedancia de entrada) 2 MOhm y una impedancia de salida de unos 75 Ohm.

Tambien tenemos que tener en cuenta que el valor de la ganancia Ao se define para un cierto valor de tension de alimentacion, a una temperatura constante y para una carga determinada.
Si estas variables varian (chan!) la ganancia tambien lo hace.

- Temperatura : La ganancia baja a medida que aumenta la temperatura. El valor normal esta dado para unos 25§C.

- Frecuencia : A mayor frecuencia, menor ganancia. El valor normal se da para una baja frecuencia, de 5 a 10 Hz.

- Tension de Alimentacion : Cuando disminuye la tension, baja la ganancia (logico..) El valor normal espara una tension de +- (mas-menos) 15 V.

Y como si no fuera poco, la impedancia de entrada y de salida tambien varia junto con lo de arriba, pero sobre todo con la frecuencia. A mayor frecuencia, disminuye la de entrada y aumenta la de salida.

Pero esto solamente debemos tenerlo en cuenta en casos muy especiales, con tensiones muy [muy] pequenyas y con frecuencias del orden de varios MHz.

Entonces, para los fines practicos y para analizar los circuitos, podemos considerar que:

- La impedancia de entrada es infinita
- La impedancia de salida es 0 (cero)
- La ganancia es infinita

#%#%#%#%#%#%#%#%#%#%#%#%#%
3. Etapas Operacionales
#%#%#%#%#%#%#%#%#%#%#%#%#%

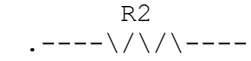
Supongo que a esta altura ya habras deducido porque este aparatito es un amplificador. Pero porque operacional??
Porque permite realizar operaciones matematicas.

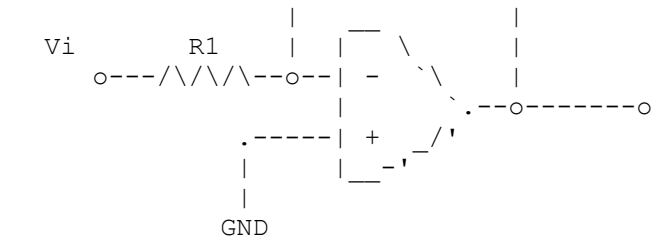
Vamos a ver algunas de las principales etapas operacionales o configuraciones, y su ecuacion correspondiente:

Recuerden que siempre lo que se esta amplificando es una tension.

[no indico la alimentacion para no
complicar las cosas, ya saben , +- 12V]

- Amplificador Inversor:
>>:>>:>>:>>:>>:>>:>>:>>:





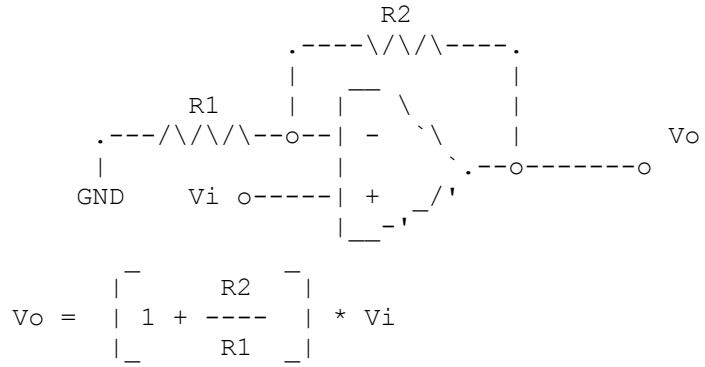
$$Vo = -\frac{R2}{R1} * Vi$$ [fijense el - del resultado]

Esta configuracion permite amplificar la senyal que se inyecta en la entrada, y a su vez invertirla. La ganancia de la etapa esta dada por 2 resistencias.

Por ejemplo, si a la entrada tenemos 1V y a la salida queremos tener -2V, lo que se haria seria colocar una R2 que sea del doble de R1.

Notese que si R2<R1, esta etapa puede atenuar la senyal, en vez de amplificarla.

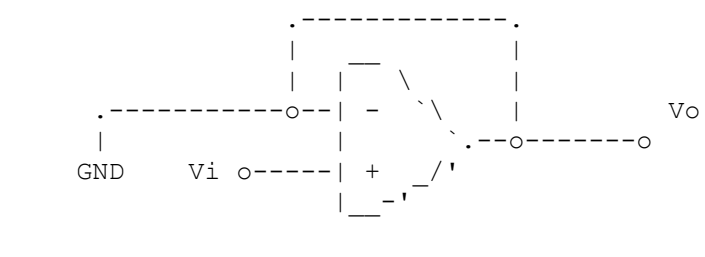
- Amplificador No Inversor:
>>:>>:>>:>>:>>:>>:>>:>>:>>:



$$Vo = \left(1 + \frac{R2}{R1}\right) * Vi$$

Como podemos apreciar, la salida de esta etapa no esta invertida con respecto a la entrada, y ademas la ganancia siempre sera 1 o mayor.

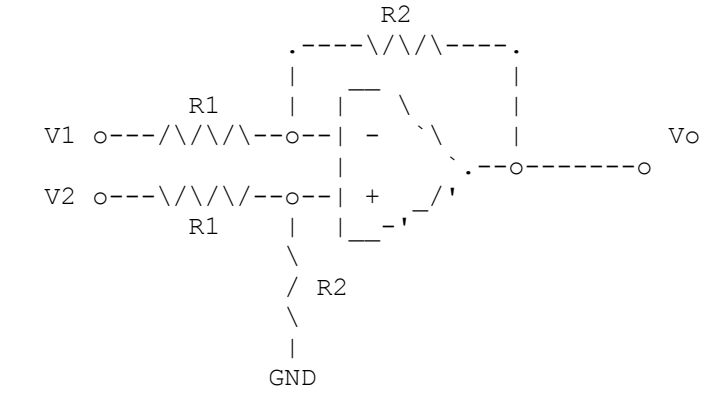
Existe un caso especial de esta configuracion, el llamado seguidor de tension o buffer.



$$Vo = \left(1 + \frac{R2}{R1}\right) * Vi$$ ==========> si R2 = 0, R1 = 0 : $Vo = (1 + 0) * Vi$, $Vo = Vi$

De que nos sirve esto?
Esta configuracion nos es util para adaptar un sensor o cualquier otro dispositivo.
Como la entrada tiene impedancia infinita, no se deriva corriente hacia el interior del integrado, de modo que no hay caidas de tension.

- Amplificador Restador
>>:>>:>>:>>:>>:>>:>>:>>

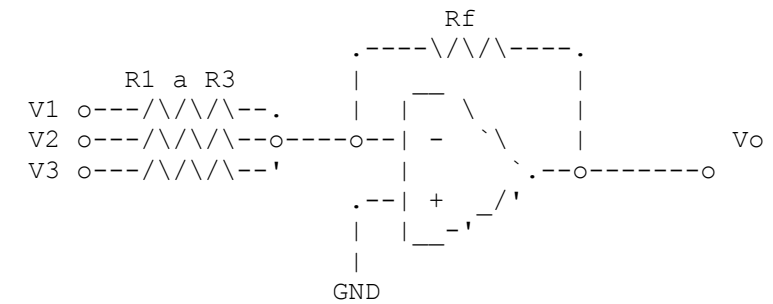


$$Vo = \frac{R2}{R1} * (V2 - V1) \quad \text{, siempre que ambas R2 sean iguales y ambas R1 sean iguales}$$

Bueno, no hay mucho que explicar aqui. Esta etapa nos permite restar 2 senyales y multiplicar el resultado por un valor establecido por el juego de resistencias.

[eo: la ecuacion final es mucho mas extensa,
pero raramente se utiliza, sino que se usa
la forma simplificada]

- Amplificador Sumador Inversor
>>:>>:>>:>>:>>:>>:>>:>>:>>:>>:>>



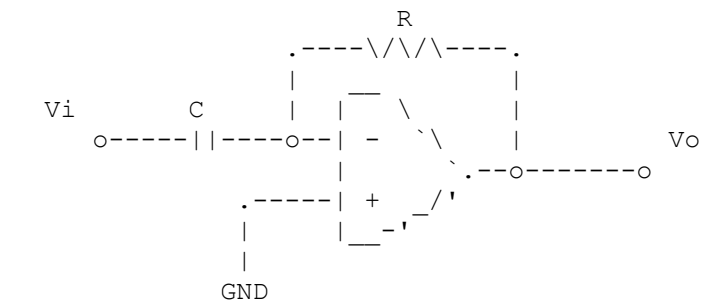
$$Vo = - \left| \frac{Rf}{R1} * V1 + \frac{Rf}{R2} * V2 + \frac{Rf}{R3} * V3 \right|$$

Aqui tenemos una amplificacion de cada senyal (V1, V2, V3) por un valor dado por el juego de resistencias. El valor no tiene porque ser el mismo,

ya que R1, R2 y R3 pueden ser diferentes.

Una vez amplificadas las senyales, se suman entre si y se invierten.

- Amplificador Derivador
>>:>>:>>:>>:>>:>>:>>:>>:



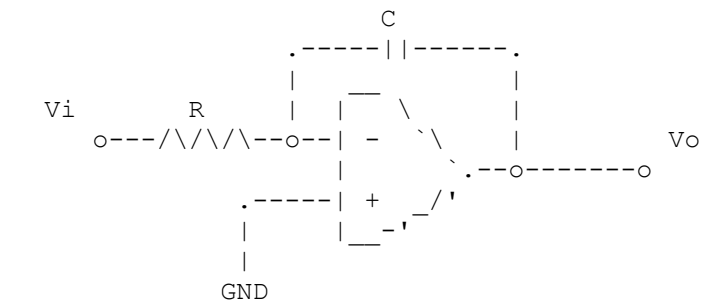
$$Vo = - R*C * \frac{dVi}{dt}$$

Basicamente, seria un filtro pasa altos.
Como pueden observar, la senyal es invertida y multiplicada por un valor que depende de R y C.
Ademas, tambies es multiplicada por un valor que depende de la variacion de la senyal de entrada con respecto a la variacion de tiempo.

O sea, que mientras mas rapido varie la senyal de entrada, mayor sera la amplificacion.

Como ya dije, un filtro pasa altos.

- Amplificador Integrador
>>:>>:>>:>>:>>:>>:>>:>>:

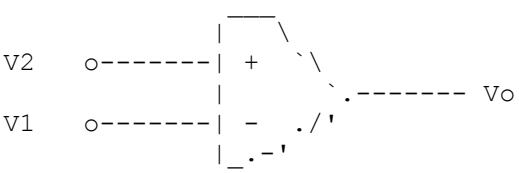


$$Vo = - \frac{1}{RC} \int Vi \, dt$$

Basicamente, un filtro pasa bajos.
La senyal de salida depende de la integral de la variacion del tiempo.
Mientras mas lento varie, mayor sera la integral.
Luego es invertida y multiplicada por un valor que depende de la inversa de R*C.

Como ya dije, un filtro pasa bajos.

- Comparador
>>:>>:>>:>>:>>

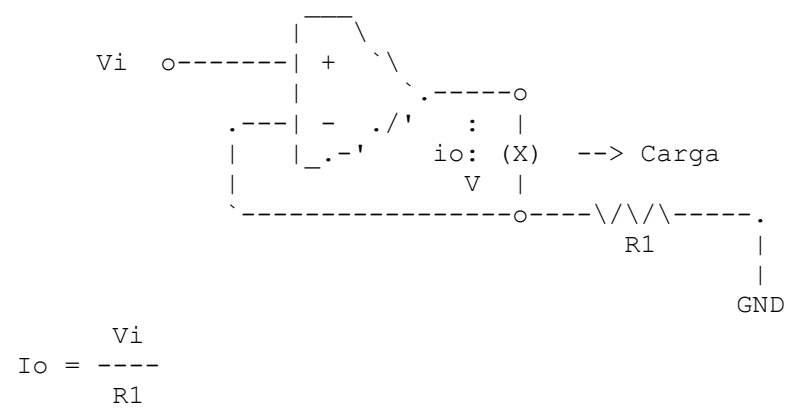


Vo = Vcc, si V1>V2

Vo = -Vcc, si V1<V2

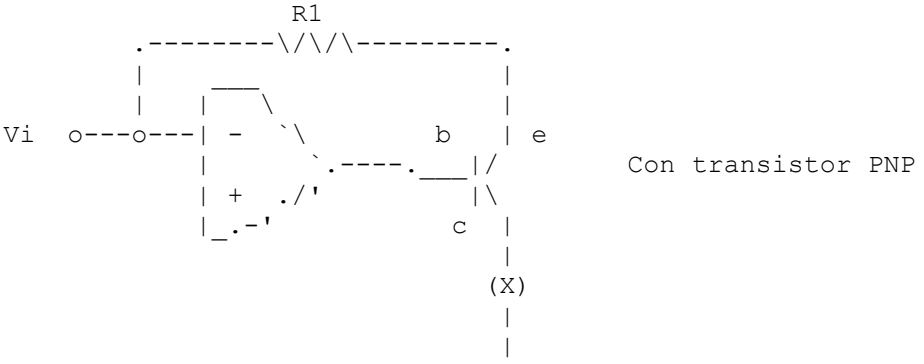
Que quieren que explique?

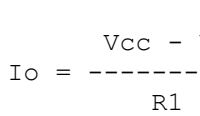
- Conversor V-I
>>:>>:>>:>>:>>:>>



$$Io = \frac{Vi}{R1}$$

Utilizaremos esta configuracion cuando la carga no necesite estar referida a masa.
Notese que la corriente que recibe la carga depende solamente de la tension, por lo que se puede decir que a tension constante, es una excelente fuente de corriente.

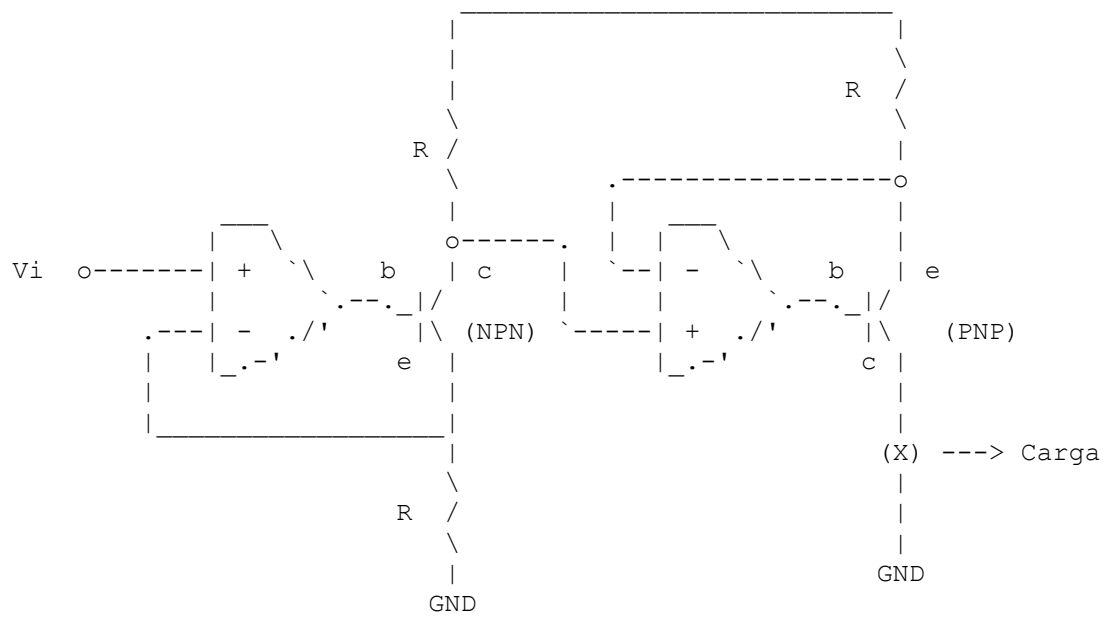




En este caso, con la ayuda de un transistor PNP, podemos utilizar el conversor V-I para una carga que necesite estar referida a masa.

Pero no todo es perfecto, en este caso, la corriente de salida es inversamente proporcional a la tension de entrada (en los casos anteriores era directamente proporcional).

Se puede solucionar de esta manera:



Entonces tenemos la chancha, los 20 y los chanchitos.
Es decir, un conversor v-i, con el cual se puede hacer una fuente de corriente, que entrega una corriente que depende de una tension y es independiente de la carga, y para festejar, la carga esta referida a masa y la corriente obtenida a la salida es directamente proporcional a la tension de entrada.
E independiente de la carga.

[gracias, gracias]

#%
4. Amplificadores de instrumentacion
#%

Normalmente las senyales provenientes de sensores tales como PTC, NTC, termocuplas, sensores de presion, humedad, velocidad, etc ; son de muy bajo valor. Ademas, como estos sensores se utilizan principalmente en la industria, las senyales vienen enmascaradas por ruido electrico producido por motores, variadores de velociad, electroimanes, etc.

Existe un parametro de los amplificadores operacionales que mide la inmunidad al ruido electrico, llamado Relacion de Rechazo del Modo Comun.

Usualmente se lo encuentra con la sigla CMRR [o CMMR, no me acuerdo]
[creo que cmrr esta bien]

$$CMRR = 20 \log \frac{dD}{dCmr}$$, expresado en dB.

Para el caso del 741, el CMRR tiene un valor de unos 70 dB, mientras que para operacionales dedicados especificamente a tareas de instrumentacion podemos encontrar valores de CMRR de 100 dB o mayores aun.

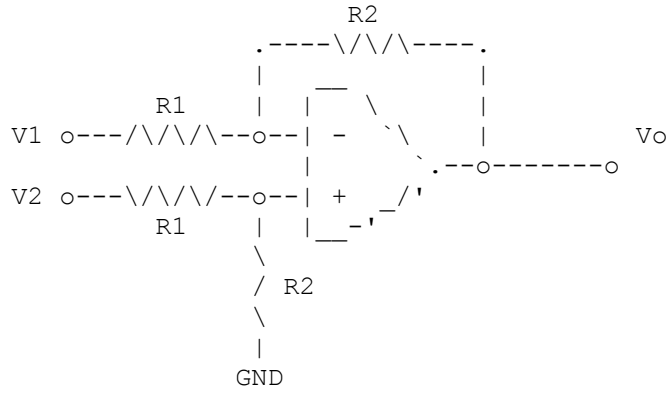
A la carga

Supongamos que necesitamos medir una tension de 1 a 10 mV (milivoltios, 1*10e-3) y el aparatejo de medicion tiene un rango de 1 a 10 volts.

Bue, facil, solamente hay que amplificarla 1000 veces.
Se puede hacer sin problema (si se complica mucho, se hace en 2 fases) pero aqui esta el quiz de la question.

Si por esas cosas de la vida, hay un pu70 1mV de ruido, vamos a tener un error del 10%. a la salida.
Demasiado para mi.

Mejor armemos un circuito restador.



Recordemos que: $Vo = \frac{R2}{R1} * (V2 - V1)$

Por supuesto, siendo ambas R2 del mismo valor, y lo mismo para ambas R1.

Volviendo un poco al pasado (creo q la segunda o tercera entrega), les mencione que las resistencias no eran perfectas

[ya se que se dice resistores, pero no tengo ganas]

Sino que contaban con una tolerancia o variacion del valor nominal, que es un defecto intrinceso del proceso de fabricacion.

Ahora tomemos los 2 conceptos:

Si V2=V1, Vo sera igual a cero [seguro]
Es sabido que el ruido electrico sera igual en ambas entradas, V1 y V2. De manera que si tenemos 1mV de ruido, habra 1mV de ruido en ambas entradas.

Que pasa...V2=V1, Vo=0

Entonces, esta etapa podria ser considerada como inmune al ruido.

Ahi es donde se encuentra el secreto de esta etapa amplificadora.
La inmunidad al ruido electrico.

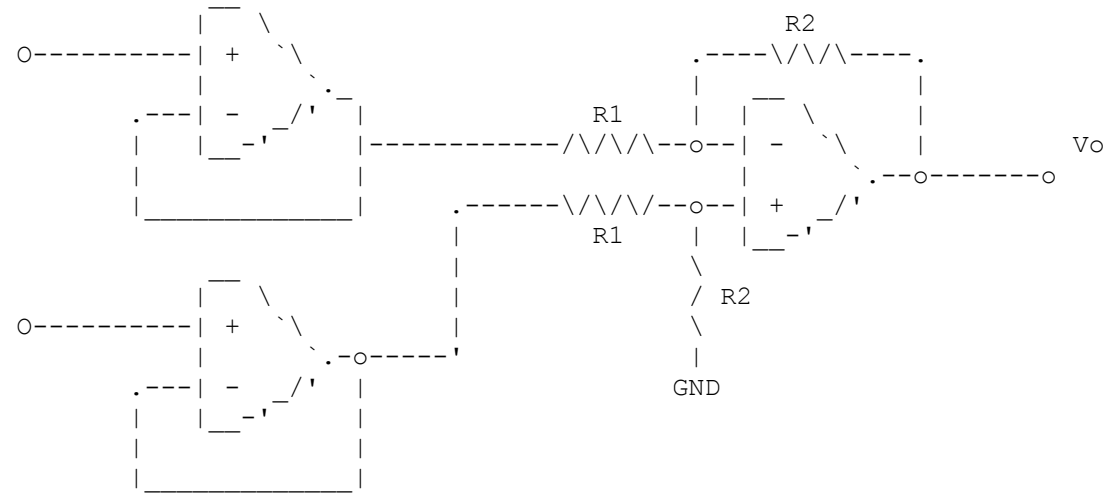
Puede ser usada de manera excelente para medir valores bajos, o
como [que se yo, un ejemplo] preamplificador para un microfono, para
una guitarra, para lo que se les cante los eggs.

Ya se que los estoy cansando con esta frase, pero nada es perfecto.
Si se fijan bien en la etapa amplificadora, la impedancia de entrada que
tenemos es muy [mucho] muy baja [bah, no tanto], de manera que la
grandiosa [ja!] caracteristica de la impedancia de entrada infinita
nos la jugamos al truco.

Ademas, el CMRR dependera de la tolerancia de las resistencias.
Y el CMRR y la ganancia de la etapa se afectan mutuamente.

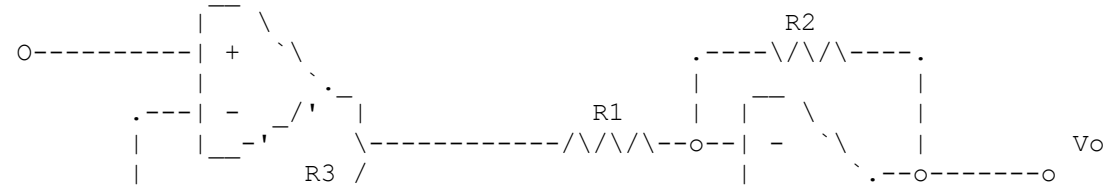
Lo mejor es dar una ganancia de 1 a la etapa [si, 1], para que el
CMRR no se vea afectado.
Y poner unas buenas resistencias de presicion. Al 2%, o 1%.

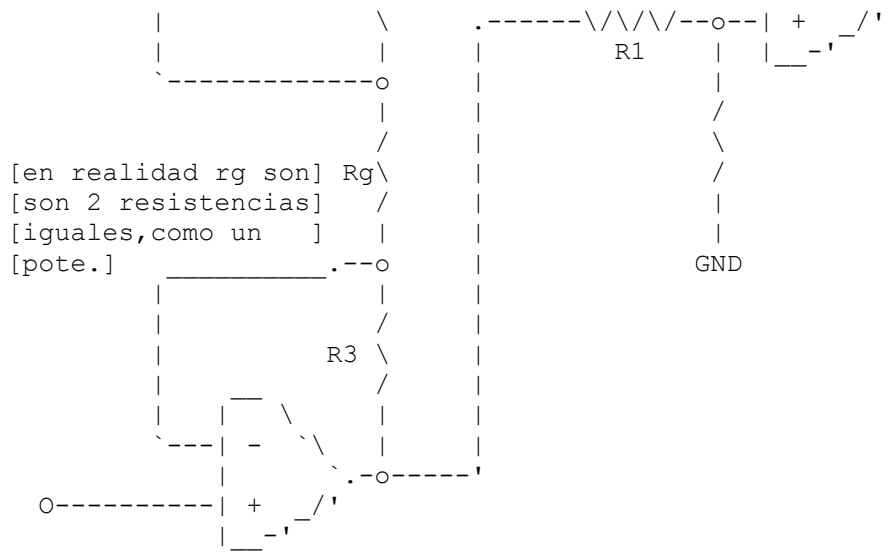
Sin pensar mucho, podemos resolver el problema de la baja impedancia d
entrada:



Si es que han leido el articulo, veran que a la entrada ahora hay 2
amplificadores no inversores, en este caso usamos la configuracion
especial que les comente, el buffer.
Ahora tenemos impedancia de entrada practicamente infinita.

Pero seguimos teniendo el problema con el CMRR y las tolerancias.
De la manera siguiente, podemos hacer que la ganancia sea independiente
de la etapa restadora:





Analicemos:

$$dD = \left| 1 + \frac{R3}{R1} \right|$$

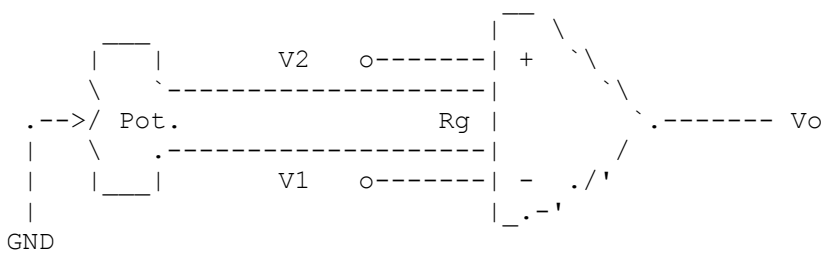
$$Ri = \frac{Rg}{2} \longrightarrow dD \left| 1 + \frac{2Rg}{Rg} \right|$$

[recuerden q son]
[2 resistencias]
[iguales]

Ahora el CMRR no depende de las resistencias de la etapa restadora, sino que la ganancia de la etapa depende de un ajuste externo.

Existen dispositivos comerciales que contienen en una sola capsula todo lo que mencione hasta ahora, el restador y los buffers, junto con las resistencias de la etapa restadora.
Todo en una sola pastilla.

Traen 2 pines que se llaman, justamente, Rg; que sirve para conectar un potenciometro (preset gralmente) con el cual se ajusta la ganancia de la etapa.

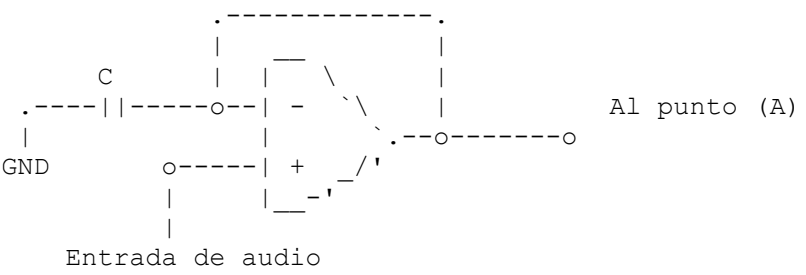


5. 0x43485550414D4520DC412050494A41 [el que se ponga a ver que es,
[circuitos utiles] no se enoje]

A todos nos gusta escuchar musica.
Algunos escucharan musica clasica.
Algunos escucharan heavy metal.
Algunos haran su propia musica.
A ellos me dirijo, presentandoles un mezclador.

Mezcladador de audio

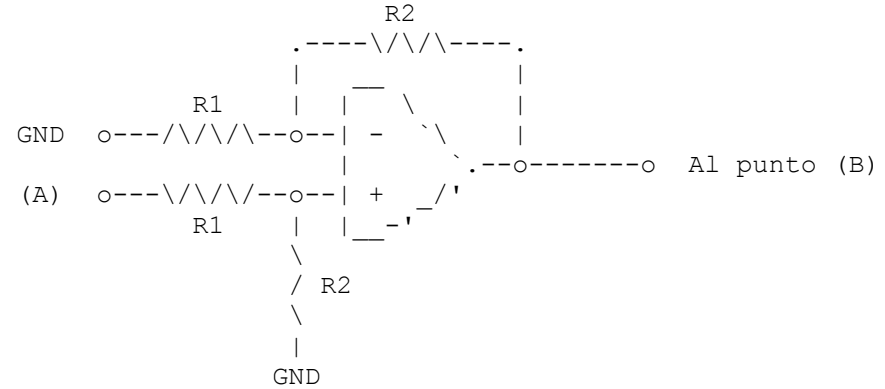
Hacer 1 de estos por cada canal de audio.



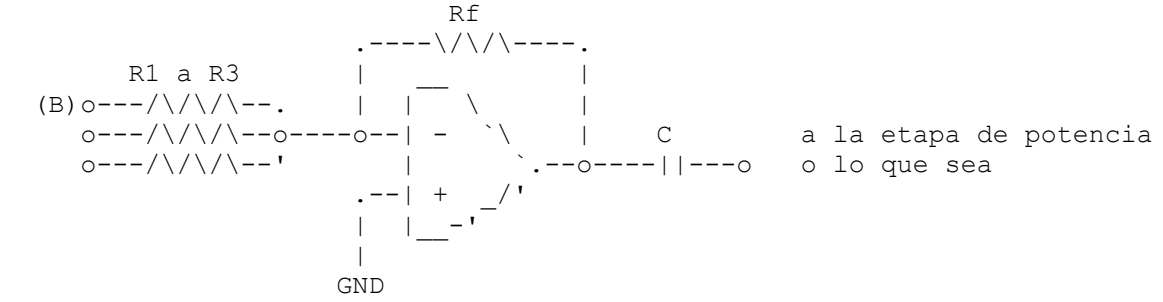
C = 10 uF

Tambien colocar uno de estos, por cada canal, anyadido a lo de arriba.

R2 = 100 K
R1 = 100 K



Y en esta etapa, colocar tantos puntos B como canales haya.
Osea, una resistencia mas.



R1-R3 = 10 K

Rf   = 100 K
C   = 10 uF

#%
6. Shutdown
#%

Bueno, esto ha sido todo por este pequenyo articulo.
El circuito queda a la experimentacion de cada uno, si quieren varien la ganancia, anyadan otras etapas, lo que uds quieran.

Nos veremos [o leeremos??] en otro articulo.

elotro
<elotro.ar@gmail.com>

##connection terminated
##shutdown

EOF

```
-[ 0x08 ]--------------------------------------------------------------------
-[ Curso de electronica 06 ]-------------------------------------------------
-[ by elotro ]----------------------------------------------------------SET-35--
```

```
@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%
 Electronica - Sexta Entrega
------------------------------------
 Disenyo Electronico
@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%
```

Y aqui vamos otra vez con la incanzable saga de articulos de electronica en SET.
Como pueden ver, en este articulo trataremos el apasionante [ja!] mundo del disenyo de circuitos electronicos, desde una mera idea hasta un dispositivo terminado, valiendonos de muchas herramientas informaticas.

Comencemos....


```
-.-.-.-.-.-.-.-.-.-.-
-.-.-.-.-.-.-.-.-.-.-
1.  Inicio
-.-.-.-.-.-.-.-.-.-.-
-.-.-.-.-.-.-.-.-.-.-
```

Recientemente me he dedicado a realizar algunos circuitos para vender a quien quiera [mejor postor], y me he dado cuenta de los trabajoso que es hacer cada una de las cosas a mano.

En esta 'guia' voy a tratar de dar una serie de consejos para que, a la hora de realizar un disenyo de un circuito electronico por nuestra cuenta, podamos usar algunas herramientas informaticas de facil acceso, que nos haran el trabajo mucho mas facil.

Pero primero, debemos tener una minima idea de que queremos hacer.
En mi caso, el circuito a disenyar sera un controlador y driver para motores paso a paso.

Un motor paso a paso es un tipo de motor muy especial, que se diferencia mucho de los motores electricos comunes que la mayoria de nosotros estamos habituados a ver.

Se utiliza mucho en robotica, y en informatica. Alguna vez te preguntaste como es que tu HD, tu floppy, tu etc., puede ubicar los datos de manera tan precisa?
Sencillo, lo hace con un motor paso a paso (PaP).

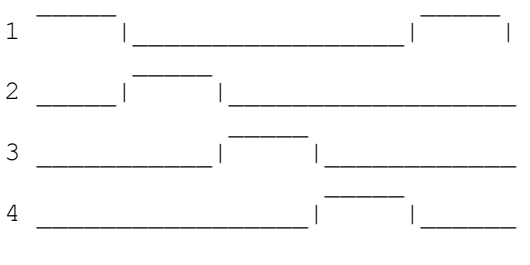
En escencia, un motor PaP es un motor que hace girar un eje central, pero

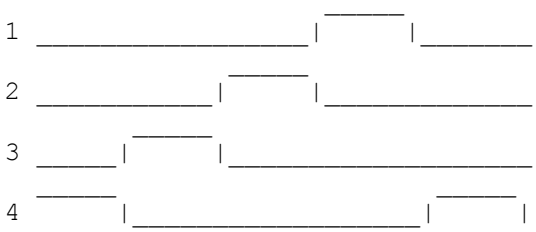
no de manera continua, sino que lo hace en forma de saltos (pasos).
Y para que esto ocurra, se necesita un circuito de control que envie los pulsos adecuados al motor.

-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-
-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-
2. Disenyando el circuito
-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-
-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-

Un motor PaP necesita que se le entreguen pulsos de una manera especial.
En mi caso, voy a usar motores PaP del tipo unipolar, que necesita alimentacion de un solo sentido.
Este tipo de motor cuenta con 4 pines para recibir los pulsos.

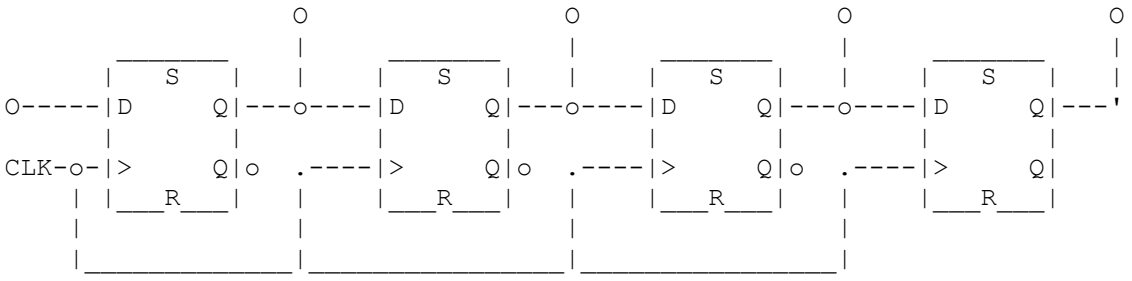
En este esquema podemos ver como deben ser los pulsos que debemos entregar para que el motor avance paso a paso en un sentido determinado. (supongamos en sentido horario)





Como pueden observar, los pulsos deben ser entregados en orden secuencial, para que el motor avance.

Bien, esto lo podriamos realizar con unos flip-flops encadenados, a manera de un registro de desplazamiento.



Bueno ahi tenemos 4 salidas, en orden secuencial.

Pero esta solucion no es la mejor. [que a mi se me ocurrio al menos]

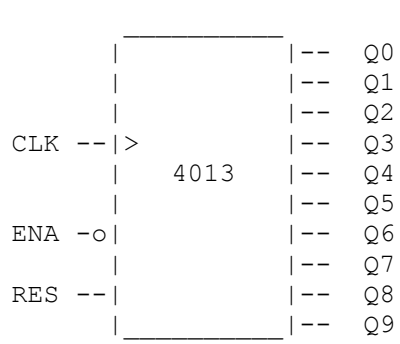
Cuando los biestables ponen un 1 en su salida, este no se reseta.
Si, ya se que podriamos activar el reset con la misma salida del flipflop,
pero igualmente, vamos a necesitar 2 integrados para generar los pulsos.

Un ejemplo de este tipo de integrado puede ser el CD4013, que es un
flipflop doble, tipo d, de 14 pines.

No se a ustedes, pero a mi, 2 integrados de 14 pines comienzan a asustarme.

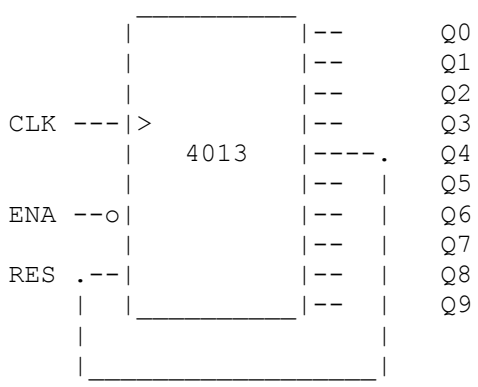
Etapa de generacion de pulsos

Pero por suerte, contamos con otro integrado que cumple las veces de
contador, el CD4017, que es un contador/divisor que cuenta con 10 salidas.



Como ven, contamos con salidas mas que suficientes, una entrada de reloj,
otra de habilitacion, y otra para el reseteo del contador.

En nuestro caso, no necesitamos contar hasta 10, sino hasta 4, de manera
que lo que haremos es que en vez de contar 5, vuelva a 1.



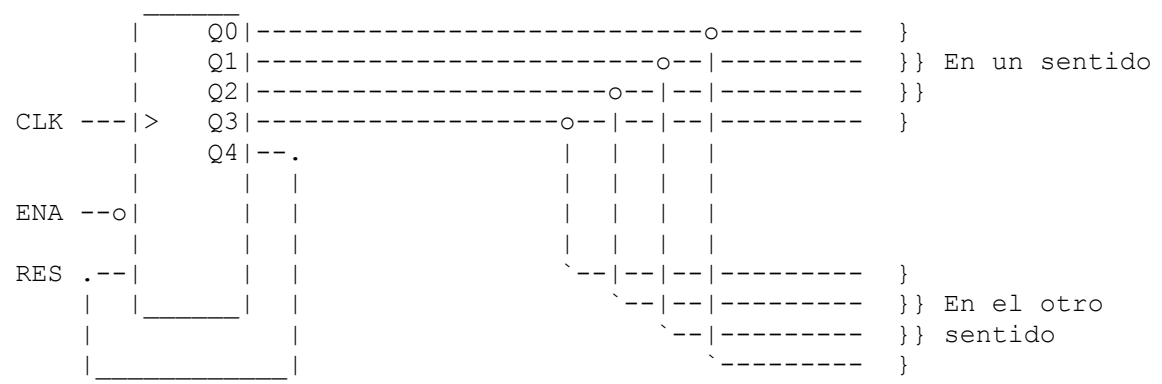
De esta manera, al activarse la salida Q4, tambien se activara el reset,
haciendo que el ciclo comienze nuevamente.

[noten que la primer salida es Q0,
  por eso debemos conectarlo en Q4]

Ok, ya tenemos solucionado el problema de como generar los pulsos, pero
si lo unico que hacemos es conectar las salidas directamente, solo
podremos tener un unico sentido de giro.

La forma mas facil de obtener el sentido opuesto, es cruzando las salidas,
esto es, colocando Q4 en el lugar de Q1, Q3 en el lugar de Q2, Q2 en el

lugar de... bah, me entienden.

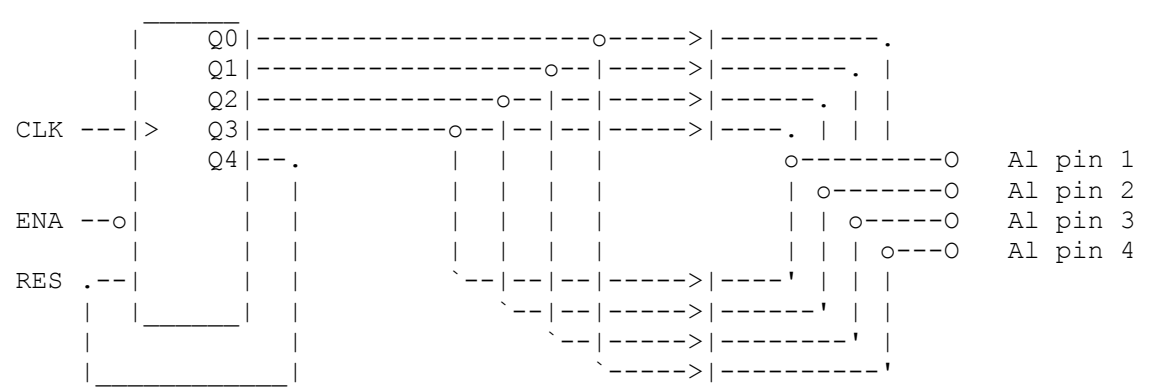


Ya tenemos solucionado como obtener los 2 sentidos de giro, pero es ahora cuando se nos presenta el problemon:

- Hay que convertir esas 8 lineas en 4.

Primero que nada, no podemos puentearlas, porque cuando una salida tenga una corriente entrando..., chau al integrado.

Supongo que podriamos ponerle diodos a cada salida, con lo que quedaria algo asi...



Y aunque este circuito parezca la panacea universal, tampoco es la solucion.

Explico:
A la salida del integrado tenemos unos 9V (siempre q esa sea la tension de alimentacion del integrado). El problema no esta en la ca¡da de tension que se origina en los diodos, sino en la corriente de salida de las salidas del integrado, que va a ser muy critica para poder accionar la etapa de potencia.

Ademas no contamos con ninguna manera de seleccionar el sentido de giro.

Asi que lo que tenemos que hacer es recurrir a unas fucking puertas AND, y a unas OR.

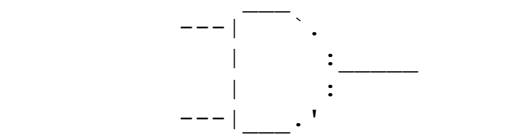
[3 integrados mas]

Veamos si me sale el ascii:

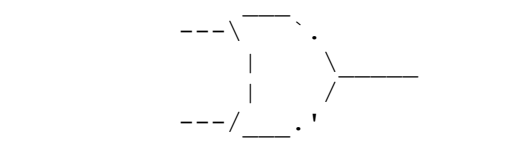
[no voy a dibujar las compuertas,
  porque se me va a ir un ascii de
  1000muchas000 lineas, y no se
  va a entender   ]

Mejor usemos una equivalencia:

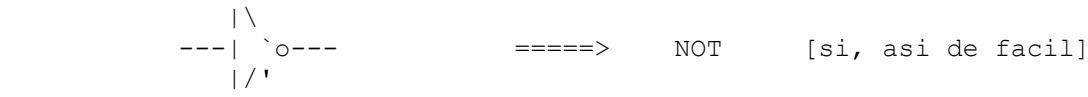
Puerta AND:

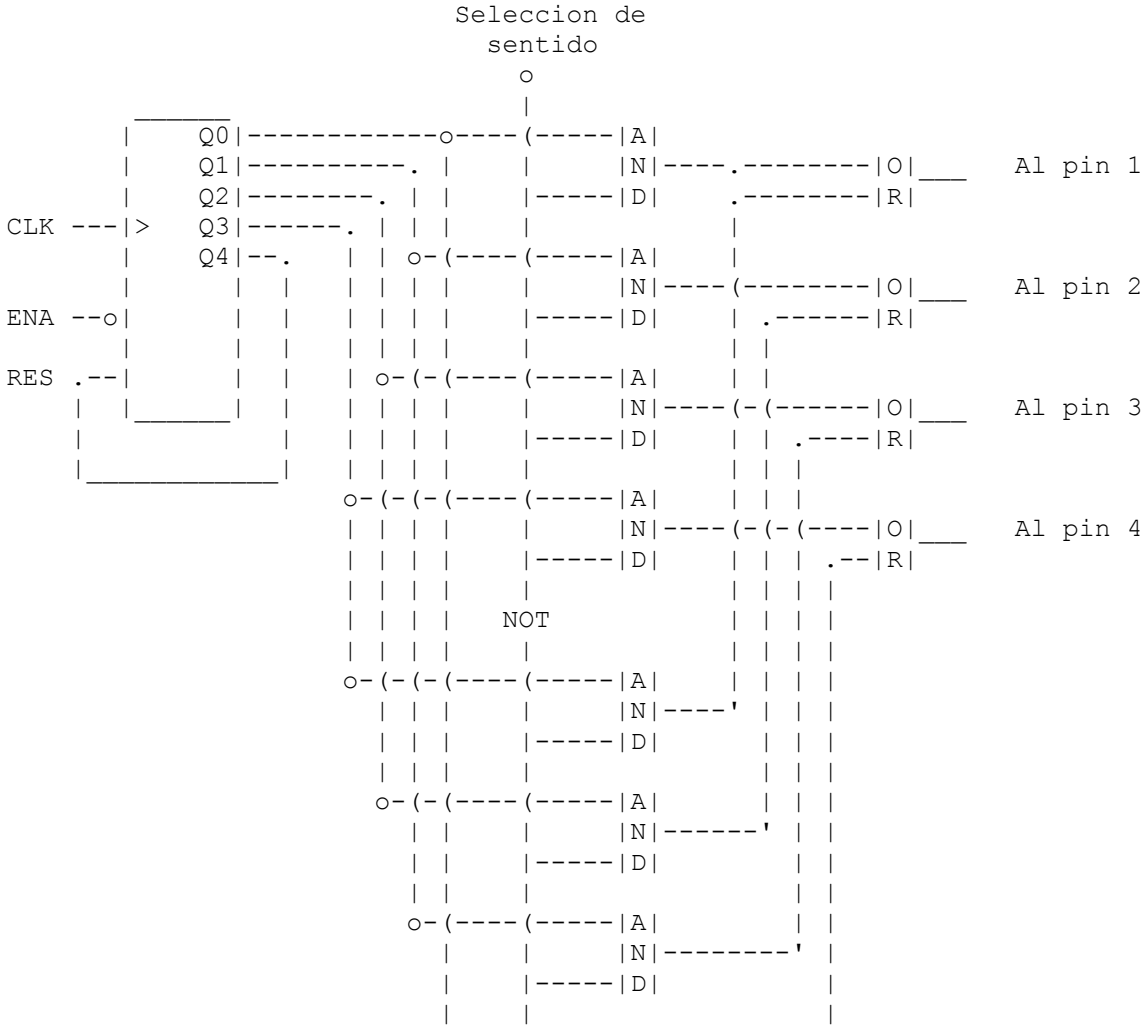


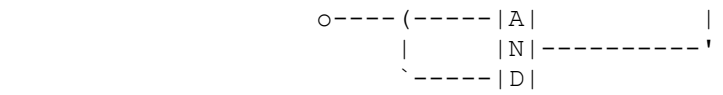
Puerta OR:



Puerta NOT:

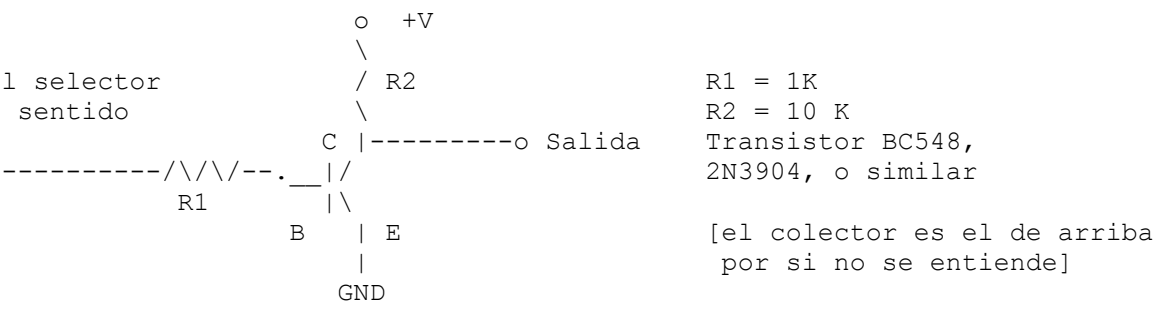






Y asi de simple es como podemos dar por terminada la etapa de generacion
de pulsos.

Notar la puerta NOT que esta entre las puertas AND.
Como no existe lugar sobre la tierra donde podamos comprar una NOT
sola, vamos a tener que fabricarla:



Y ahora si damos por finalizado.

Etapa de potencia

Hay que aclarar algo. Un motor PaP consume cerca de unos 500 mA, asi
que ni en p2 podemos hacerlo funcionar con la corriente que extraigamos
de las compuertas, porque vamos a tener unos 10 mA como maximo.
Si lo superamos, se quema.

Vamos a necesitar una etapa de potencia que nos permita manejar esa
corriente, y que al mismo tiempo no sea costosa ni complique demasiado
el disenyo del circuito.

Hay 2 opciones:

- Usar el integrado ULN3909, [creo q era ese, no estoy seguro]
  que es especialmente disenyado para esto, pero aca en Argentina
  se consigue a unos 15 U$S

  [notar que 1U$S = 3$ argentinos, 15U$S = 45$ argentinos,
    45$ argentinos = casi toda mi jornada de trabajo...]

- Usar unos transistores BD239,
  que tienen una potencia maxima de 8W. Si alimentamos con 9V, la
  maxima corriente que podremos tener sera de:

  [aprovechemos para repasar ley de ohm y joule]

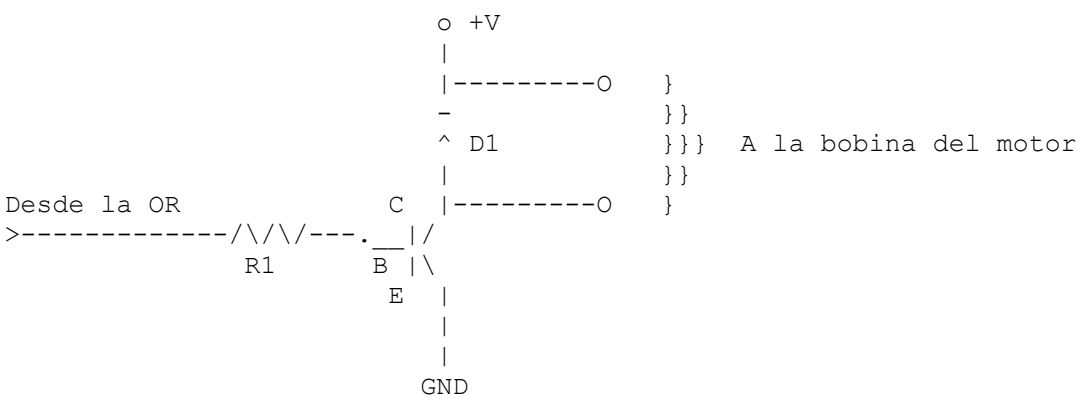
  8 W = 9 V * x A

  8 W
  ----- = x A ---------- > 0.8 A
  9 V

  La maxima corriente sera de 800mA, mas que suficiente para nosotros.

El disenyo final de la etapa quedaria algo mas o menos asi:

[notar que hay que hacer 4 etapas
  identicas, cada una conectada
  a una salida de las OR   ]



Todo esto con:

R1 = 680 Ohm
D1 = 1N4004, o superior
Transistor BD139 o similar

Ahora que ya tenemos la idea de como va a ser nuestro circuito, podemos
pasar a la etapa de simulacion, para ver si todo funcionara como
suponemos.

-.-
2.1   Software de simulacion
-.-

Dentro de mi escaso y burdo conocimiento, hay 3 software de simulacion
electronica.

- Livewire,
  producido por New Wave Concepts Ltd.,
  del que la ultima version conocida es la 1.11, del 28 de Octubre de 2004.
  Puedes visitar la web de sus desarrolladores en

  http://www.new-wave-concepts.com

  Si mal no recuerdo, hay una version de prueba para descargar que
  con un crack que anda en la web, se convertia en version full.
  O prueba a crackearlo tu mismo.

- Multisim,
  producido por Interactive Image Technologies Ltd.,
  del que la ultima version [que yo conozco], es la 7, del anyo 2003.
  Puedes visitar la web de sus desarolladores en

  http://www.no-tengo-la-menor-idea.mejor/busca%en+google

  Recuerdo haber bajado una demo con su correspondiente crack, y funciono,
  aunque no puedes abrir los archivos, y todo se ve como el reverendo orto

- Electronic WorkBench,
  producido por la misma empresa de arriba,
  del que la ultima version [que yo conozco, aunque se que hay una nueva

con su correspondiente trial] es la 5, del anyo no me acuerdo.

2.2 Livewire

Muy buen programa, con gran capacidad para simular circuitos digitales.
Cuenta con libreria interna de componentes, y instrumentos virtuales.
Ademas hay un software producido por la misma empresa que se llama
PCB Wizard, que se vincula automaticamente con Livewire y permite
realizar disenyos de circuitos impresos desde los esquemas de Livewire.

Pros:
- Excelente para circuitos digitales
- Facilidad de uso, muy intuitivo
- Aspecto visual

Contras:
- Libreria de componentes muy reducida
- No permite ampliar la libreria
- Un verdadero fiasco a la hora de simular circuitos analogicos.

2.3 Multisim

Excelente programa, el mejor de los 3. Basado en el motor de simulacion
de SPICE (legendario).
Se desempenya de manera impresionante en la simulacion analogica
y digital. Cuenta con osciloscopio totalmente personalizable y
analizador de espectro.
Hay paquetes de ampliado de librerias (pagos) y de vinculacion con
programas de CAD.
Cuenta con editor de simbolos y componentes.

Pros:
- Excelente en todos los aspectos de la simulacion
- Libreria ampliable

Contras:
- Dificil de aprender al principio
- Mucho espacio...

2.4 Electronic Workbench

MMMmmm...
Para la epoca de la version que yo probe (version 5, anyo 1994-1995)
debe haber sido un lujo.
Hoy superado ampliamente por Multisim.

[eo: aunque en la web del fabricante (de la cual no recuerdo el nombre)
hay una version nueva de Electronic Workbench. Habria que ver
que tal esta]

Pros:
- Pequenyo tamano
- Pequenyo uso de recursos

Contras:
- Simulacion no muy buena, aunque suficiente para cosas sencilla.

Pueden encontrar en la seccion 5. Data, un archivo uuencodeado que contiene el circuito del controlador PaP para livewire [tampoco voy a hacer todo por uds]. Consigan el soft, hagan en circuito en el soft que quieran, simulen y me cuentan cual les gusto mas.

-.-

3. Plaqueteando

-.-

Bueno, una vez que hemos dado por finalizado el disenyo del circuito, y lo hemos probado en el software de simulacion de nuetra preferencia, podemos pasar a la etapa de crear el circuito impreso para nuestro aparatejo.

Que es un circuito impreso??
Es una placa de algun material rigido, con material conductor en un lado (o ambos). Sobre este material conductor se trazan las conexiones entre los componentes de nuestro circuito, que van montados sobre perforaciones en la placa y luego soldados con una aleacion de esta¤o-plomo.

No se como andaran las cosas en otros paises, pero aqui en argentina se consiguen principalmente placas de 2 materiales:

- Pertinax: Una especie de acrilico, bastante resistente para el aspecto que tiene, fragil contra los golpes. Aisla del calor bastante bien. Absorbe humedad.
Malo para aplicaciones en alta frecuencia, porque actua como un capacitor.
Barato.

- Fibra de vidrio: Es basicamente, fibra de vidrio. Resistente, solido, no absorbe humedad, bueno en alta frecuencia (dentro de los limites normales). Muy buen aislante del calor y ruido.
No tan barato.

Como lo que estoy haciendo es un prototipo, y estoy en plan taca¤o, vamos a utilizar una placa de pertinax de 10x10cm, con cobre en un solo lado.

3.1 Software de CAD electronico

-.-

Recuerdo aun las viejas epocas de antanyo, en que mi 386SX/16 y yo nos peleabamos para poder dibujar una pu7a plaqueta en el afamado paintbrush.
No se lo recomiendo a nadie.

Por suerte, alguien oyo mis suplicas y contrato a algunos programadores que hicieran el trabajo por mi.
Este es el fruto de el CAD orientado a la electronica:

3.2 Eagle

-.-

Uno de los mejores programas que he visto en toda mi vida. Para mi, no es un programa de CAD, sino un IDE de disenyo electronico.
Cuenta con una vasta.., vasta..., vasta libreria de componentes.
Tiene un lenguaje de programacion propio que permite realizar la mayoria de las operaciones monotonas que generalmente hacemos cuando disenyamos un circuito impreso.

El editor de componentes es espectacular [sin palabras]
El autoruteado es muy eficiente e inteligente, sobre todo cuando el disenyo es de dos caras.

Trae drivers para los plotters mas de las marcas mas conocidas, e incluso para los taladros matriciales mas comunes.

Exporta en formate dxf y otro que no me acuerdo.

La empresa que hace esta bestia es CadSoft [se mataron la cabeza con el nombre,je,je].
Pueden descargar una demo [que esta en aleman generalmente] desde:

www.cadsoftusa.com

3.3   Express PCB
-.-

No es que sea malo, o trabaje para cadsoft, pero este programa no tiene comparacion frente a eagle.
Lo unico bueno es que una vez que hemos hecho el disenyo, podemos enviarlo por email a la empresa en usa, y ellos se encargan de hacer por nosotros el circuito impreso.

Busquenlo en internet.

-.-
4. Resultado
-.-

El circuito funciono a la perfeccion (despues de incontables horas frente al monitor haciendo los fucking pcb's) y no tengo nada mas para comentar porque es muy tarde y manyana tengo un examen..

-.-
5. Data
-.-

Aca les dejo el circuito del controlador pap, hecho con livewire y plaqueteado con su compadre, don pcb wizard. Enjoy....


```
begin 666 disenyo.rar
M4F%R(1H' #O0<P@ #0        #==7H @", +    "     ">!3LY      =
M,P,      $--5 E!3+Z0NZ#!V!:!(FBN41$-\^&'BWM@S#?<O5>_:9<\@SE7
M\1H70;;I@BWDJX=T@( J "$*  #4*0   JTBE&$=BCDY'34* "    !C=7)S
M;S<N;'9WYQA,1##LJX$AZI4T-"ZQM<ZL_8CBXLL,7C_PO;[V29TQQ#(X&,W<
MU'V^%A)"(RP@_5(!<.7:7 -9DNX].3<RHYO<,K0-S>RX2L++L1ES"92A:#M=
M@:4'37"H)_-D*^*-J_(C[/<(C^M;&EK(NDV?B->O9@&*+IL0?&62W.T$AJ8#
M^_B9)G/_-V96?,6XO*]S2LO_^U3):D16DPR_O6TEF8KBIX_F6"1&(?MFN4>-
MD+A\X.UP[4* 57C*X_H=1]/ZG$0@Z?MAAZU+VN=AIK]93KFX'6U>P_R5YR)5
MF"N$[C18]@OBIW.FHAK=@G+6Z#^A>L>U;2%@[>FIZES)$$.+&*GCJ$((MA[F
M##B9;1]=[0X71\]@TOMKT?W.SGPY)$_ZR80E;/T\]&B7"(0JVFHPE<Q TC$!
M)3.6.)+GC#:SW]1S;N%^QSVIHC!WEV_OYP)(XF*W(959:MN<P6H<0P-DR!E#
MHVGQ4^B :B:&,4&-Q4[NYD"-^0K>H!&RML>?SM^Y*P[10X.+J[>B4'_'A/R>
MY2('Q82H%A"!,_]$CQ1/^!_CMD#6VU$&1M4.,;M(G%O#SJOI=C#I8C&,=?FG
M1N1[2DE<^OC$9G+4D9'OA/5\KL'+4GG&=S[CQBF97A3;)!=P]0\[@JAI49[U
M <$$S%6#ZR N;4_-QND'%GG9'I.]Q$1LR_-PMM#F5@\ZW9P/ :QJ:ZPHVH(=
M)I2P#+3;* *NUH>-7%$&0C$ D1OGC4&X0OK?]&*@(_B/LVGAO;L-53;(R^YR
MCI;^XE=%RAF.G*<'*162 -/;=-USI$N5QR-C:@.MD>VT@3!EKK-> J!&7&;0
M/4"<G$7,*JA<^ [X&@;*_[7#=L'GK9+N_#.:U_?"[=EB!.LRL]27!5#G4+X
```

```
M]Z\/-^/.22":"=9Z@S;U$";]'_3&AYW-S-#\!6TJ:7EL9LOYSPEU)T+C)9"8
MU6XFDB4A4,F8&?C9XZ=BQ=W!C7!UL:;P>07=3MX<]#RNTBC:MRLD=O6ST],.
MWN!#=G))(.6,[5IA*#$,N<R>=:'ZH;.!=OBH<=ERZ9RU_9B&#^86Q'#?S^M=
MS#K^&D#(G&T6YVU-KT-@AMO?G3XQ;"RM[YE3%%W%_5#A:I<7.F&EFE[Q/U,
MVF.IU4"$X#=XX>.MHM_H8^1P8U?=*WA 7$'5@94!_I[8S#'4V]6*^@7Z%@U]
M7?R0Z*S5#JU9'G<&R 9/X&IRT8;A.#V;EQE95.)[\M=(988/_,Z_/!=]"1LY
MI&J,-EU=1+X]O]+-M2O->]J7V^VE2X@P[Z?%EUUQJ1QKT9)=5NC0?R)!<K%#
MQP:;?XC-MKG:^K,1-;9'IDU$#.$)%7Z:'H?57B"*=IUF*_T)X?CJP5WZ)S?N
MA(J87]\X#%5D8GJYG[XUEK<__H;*"&$5P#Q)K%C!:6S(R),"N8,#^4'6#E1U
M(C^U">$8I0)&77<?+72%B@<KOFN5LVR BFAB_A)/8_H[,0#1ZL3'@0#[I=4I
M7XYT2O"56M/V#V#BV5Q$7?1*6QQ4.(&0^7EWH'XB6^ ='?&O;B*GAKQKZY)X
M(.IU"* 3N*4U[5I$U^2IPKEU @6]*)O'BO48;'Q.].Q6P)L4%1O4NW6*2+@E
MQD0J1C5-E$^EGB,4_!.]Y($7:IVMMZUD#/^Z7F<NCK$\LG$E_EUDXH;SO:BL
M?*SS.40?#[B*"OYV(2H/M.>'3-<O0Y[F,Y"]X@((7F1+RMD;%!LPSUR :3.7
M7U(V?>6]?#3,\UWAW1J%3=3+.7XD*[:1DP_^>0*FJC75YZ!;O0ET],XW$2?:
M58(J;XP=;F:]H5O];SHD^37JW0E\<U/)KF] ]QE2W-M::@**M8U@_JH8=RRF
MQX(2H&!<(B&1HI0AD13SBU.C-I3XP1L?0.7)J:9ZS]0*"P@O6N<A*=7]?:)%
MS]4JVCG6>"848,&Y+2HB0%INUN+U.UQ:]YA,T2\SM1-H2+26P0Y:>D,0=E_X
MS=[9)@B M]U=^DY&-4?B](])*2*RGACRR:U%1.ZJ*FE%M;O2C[\7<TS^N"<9
M])R?WD'13L"08%3H&R+@P"X2G&&< 5/0-W<GB4!KW2^RP<G6$NOL"<'+!W5%
MB$_@7%J14M"IFP- #U9;VYX(A46XB2-+8T1^3"?F,=D@J (K*+2$#/J\>/H/
MM)+X..(&YI0I=?!IX#1[XBU$G0WK9Z]NA!K0CG7.#RU1"&).#+A%4)-;O&/J
MM-ZC-<O>K #P\-66TMT[S=7$S3*K] &TFFW.D(J-)W. XZ<USE6_QL=_P UG
M%]C_]^#782NQ0UU$(%I+TA&:[+4B^UGV6GUDP-=-OY!0"<?5X/$+X1^2N?S2
M'==[68O)R8J>=]DKA;X9G/K_5,-)-&VUZ!(W82)ZCF[F/T_;OG<1%<S%QEX-
MFY;X6_+0H<(KDA6QBZ_Y6.F$EH]#'*&$/<FO<0%GFD6HNJC1Y&S4(,@8=I(Z
M+3^\E!L)\H#D'E$';.+N:#J2!E1"5A&B^LGSKO+</ ?25Z&/#M>*YSCU0U.3
M3'2LL'5*,*+,N6X^V[O:?9NP3I&!$Z$.+[-;E)9\B372ICU20*,$V%HUS\ZW
M%_5DZ$]G7]2D U,KM*-1E5!$E/^#'*86M00 ^N92<FL& =(68'J[2=#5\_9;
M]<L'*W/_I<PI"G_JY*E:$3#'*QW>IHCVH][N;,%ACQ;'/,*]:[J87]?X9 0.
MS2_HB(_5A+NA63:H-2H6KI^Z7X/DY8H?)QU7UB &YVP<$/"]9#EL0UL*_TK2
MUF:;X"]M&A<<C_TV8UEV!6; )I>#,8#I 3\6NKUL$:VRY%X*D^?\+A)L]*BH
M_-JFX?/@))(+@&NZEMWQG70PBNC-/6JC2^UA_C*)KF'L(;Q=#N1.LS!(M%DJ
M$$ .N^GKAL'*</UT0>="*;->2+TU=94D+W=P9[=!GW;JM,6OYR:>S<KUVC40
MEL"L& \D2Z>B3]I( AH]S**$<(U-HO'Z1^F+\EX<POZ *[]33-JVL;RFB-G:
M-NL;'KB6!+>HF*4Q%29L\J'8]QTA9OI!G"2-YM,U-\H5(F5>BK?D\$.&VD*9
MNL!0IG,0+-3O":&M4V_H6T.FMCB(>8>>J_.#,R#,M]84^8G,3XC[XDGGH#X&
M.&5)+IH^L1\(' 2%A@4GFLE76[Z56SZOVM!D+] \=QEJ@K(G^.3X7/S@'?C
M*AN*+;DGB[ X"\JN.&O$[M6?0XF2=ZU*_<'2R<24Y!AS#?61SP]3 K]JJA(:
M*T'FAUT[ZMWD_>B(!1#2AA3_VD4T-W1&JM0(3MU,HJ>4D=-<E4E%0*5]8)$:
M#E7JHJ[*_J#*Z_<-&92>>\QV91H,GMGZQ%W'!DBXZ$@S)<PUB+>5#GY3:J3_
M9B-\VEEP7X'W,:"NC?L:1E$H1H9JU:;?DP7"=0<#9G$*!2.U4_>@T3-']W?C
M5SMP;$2@$(SG4ZP!I%'V.&9ZDJ]EH7FWW@ OXAG]JG_U C9=)" *@!L"
MXR, )NB_I8R(DY.1TU"@ @ 8W5R<V\W+G!C8H?Z-/">L#L(%HK,$C]]
M&5.'/X?T,'K2T(H5TERQ)-(L9]B"H.;!QA)+A9\S1+^."JF+'9AK:WF]VQZH
M>1#![CG![E<LY[FJ]1!$B1&HJ9.\ IG0_)'*LB+KCV/X5.99LW71%US61.QZ
M1QQ3)"D,';\V-&%*CZI(CMH'9N*]B [)OQ20^?%2,08)_ @R;YVGF1"$]B=W
MD7)3VF.&0G>@ EM\-TCVM![OYS#4KR6^6R<!KHWB\EOZ$I!RGGB5$X%:+7>,
MOY*PIW[#F=*Q;^TX?XPV-;#F5N]G !Z (GE+-UWQDM;\5^X@F"3CEX)_&G0'
MFSIZM@]IN,8-Q#4!KF,A!X\"Y;,7SN*_M/J.E(V-F@MI=.^0_=34W'CEUK"1
M#B6S:H,6=5,U!E?1I.[&4Y21N=30;NM9_OY'9F'"<X\A@>$L).RP&.;4/1SB
MXZ)^+5MB%4MC)9=LT].0#I?(WFQW3UO.:/K[HJ)JDOWVHS,=M]" PA< QO(2
M/C(/.1*$]NC'7EC.M#SDP1BP^<"AA_+X0N2$KLEJU)TAY&R]U8C:$S'SK)L5
M//A^@O#B74_M7&DZ='VHIEO9U:P@JO/1!T/P\P**3?R0O"!G0 8S?4PD0=EW
M5;\4I_WFYFZ:A+#-GT$F]G5L>?A[,/,.)*J0K&@9($B)Y/2'<3/)7UR3MT7R
M-Z#$1ZV\F@"DI9ZQ&S_:(6A6NNK%5^2_77/T8%I2EM94>C![+2TS(@%F'Q$7
M4M;4G2$T6_>^DO*E*9(YZ,#\LH?K$="2$:%1&@,?G%!GJ3-2_T;.V0(RF-BB
MWO8$I WW2(==?U/S:*=$*H!&,B9<#\F]_1%2+@;[)]BZ1#\/%_&:L\I$K,#C
M#K?-=5 EN;?<@+$8PE%84E7\U<35"N- *A8)$D K!CIQ4D1$WO>G^3R;21W0
MR&JXOPNRW9:;A,2PA"1D2"S8<*;X-/J;$<KNNPPM7! YO[Z[R<B9$C] XGN?
M*G]8^\_%NZR/"WVT9[?K& O^K8U8D [UM9_1/J=B]%+)4=]C?O'*+H@,5#%9
MWO)/].&I_QA\A6EZ% -EYC\E4=RA02"2@(1$9$C%97NQ8?O.A-?@HR>\HH*T
MIC6TUNX)<303WZPM3IP/5?#ABXY\^L,RR+TY>/%LA_6)=;IBFE#CCK(P?!@[
M!WVO?2=*#)AT'QAAE53V(/5[,4L?I1U'VJP@L&X?\*H7I2*?H L:B:?LM ]B
```

```
MH$)[3F@UN")Q<3@^<U]=]<J),V0;\(+/Q:;O_0/2E[6Q$'%)HDI9!+2( :"L
MBG*SU?95$!*/96RT_Z>:R("VHO<6%9/FKG1W">2!#CYFQ$/Z=*50.<@^GND#
M*L" 6HU1S3*&*K7#&[KN9\/O%5H&8M)>G6&0Z'W>%5M93[(V=4^OW3C/OH-^
MWT<#$$_IZ_*%"(@:-31F$T9;3CQR9Q$UH/ !4:FDSRF"%^ I11=NPT]=][ R
MQF4U\9@U,FN"31ZT3!0&RNZF355*Y0X)(]7BG&:%=;PABMJJC*TJ<,LFXSK1
M^-:89Q@$Z6C#>E.;B_"EFT:VCPA_4-E9U3?9E$A0T]!. K"*)-UF_VW9G'\>
M?();\U_-P#L[Q_B\HU&Z\&<(27O89@5:II@^M0B(B^-,779"F/SB$J!%"4WT
MIQ(KCT[#^POU@PDJ;)(4OZ)*\'9Y9BK<!W+QX=<B#WB@^QK4@U8>*B*XMM5P
M<*-1DE5;]DW]JQMU-&N?M2*UX'_49G*_E:7BN >Z*=!V/_ZH6Y?X)E/VOTAE
M2Z=L8BP>\R\E1^ZYM66""YH_[DRB->\9 OT65%'*K\A.Y_+\I9:F!I;@$6EN
MPRK75 ]CIMQ4G;F\R''B61^EVN3_ZI9*C$,W3E8F-!7)+!<!C.U7'._L(#M3
M.08U5T1052!@F3-ZF(S.)%(5UN<O4[#[Q+(S:BCP]:]9O-.ICBBZ3&W$8+;%
MCO6N8]$*WZ$=HV+-I\GJLDGE/#VV"AU#5"Z;4Q5K0%'FFNH+@S01='C8!*RN
M^_'Y+J\9V4R+;%O6NR/-^=O25%%*90D"&U>=1]-ZPC!6MUWE D;Q]X)[;K&V
MK1S8#NM,F<#K%8JM$<W]F:#;3(]L-,[KPD5%:#S%?DV-#:WB90%%+%<E?%!:
MVVR 4#Q.*O8B"^RC&B/)RUJM\PX\"I\BY))&=U;0"[\KK[@C)3.M;?="6'"B
MUZTJ6DX',$FBW^Q__)^MM#RL562*0KQ6$?X,<ZF&K[L3.=XHAX0#Y*8'MCOF
M^&T-QU(8.K?GNOE0J-E*IT" 73)^R\A+SVT.EGMHT@;<-=3GZERH01"=Z2#X
MY\GORIWL3?@Q17&B@UGLJ2(Q(W?&F$P*L8OWX,N_![CDJP-48^YDRU$M1,O&
MB3JGN4^7N^1<E)H6Z%)C)/.SG.*]2[?A8_STAU(NX-IG\4GR0S$["?])$45*
M&_(&^6TG(F&">B.HM>I?D!.CQGRZ-C#5BGLX9\V CT]%=?39F%K3CQ 3\XIQ
M0+, 5=O%"%A/6S RS8MCQNO:=%W#CD)KOXQIM>MQ3C!E1@PW3>G0.?Y.,Q8W
MI^7@RE7Q2;03POV6UM+EZ<QXRJ'^9)LP!>&7G ;N$HTI!-OY7S&S$DD0W.U%
M%'C[R3Z!UUG=+,Q?R1!W60*T>!'C%%5-2?9%E>YG>0D;(8EX"NJF7P5#@8/0
M,N7PT\%0[O_MR>[/R_*W_I[BI&[ LI^R3R=OTD;^NRZ7#WM:#@0H5QXV/3:\
M#Q<5J9N*JUCK2664"'4CH]P]!/.#645197\->3E2 ^5&I=7V.G_[%YN]<<9T
M-9Y*Q:]E\-T$_HZ)H']@]I1YRBYFRA*YS+-6=:+;2>';XK_'-ZC@B6O)P.:)
B"OG<?F$_!Z)%GIA#+$XV<YM0< "_B&?VJ?_4Q#U[ $ ' +_'
```

end

EOF

Si, sabemos que esta seccion es muyyy repetitiva (hasta repetimos este parrafo!), y que siempre decimos lo mismo, pero hay cosas que siempre teneis que tener en cuenta, por eso esta seccion de proyectos, peticiones, avisos y demas galimatias.

Como siempre os comentaremos varias cosas:

- Como colaborar en este ezine
- Nuestros articulos mas buscados
- Como escribir
- Nuestros mirrors
- En nuestro proximo numero
- Otros avisos

-[Como colaborar en este ezine]---

Si aun no te hemos convencido de que escribas en SET esperamos que lo hagas solo para que no te sigamos dando la paliza, ya sabes que puedes colaborar en multitud de tareas como por ejemplo haciendo mirrors de SET, graficos, enviando donativos (metalico/embutido/tangas de tu novia (limpios!!!)) tambien ahora aceptamos plutonio de contrabando ruso, pero con las preceptivas medidas de seguridad, ah, por cierto, enviarnos virus al correo no es sorprendente, es mas nos aburre bastante.

-[Nuestros articulos mas buscados]---------------------------------------

Articulos, articulos, conocimientos, datos!, comparte tus conocimientos con nosotros y nuestros lectores, buscamos articulos tecnicos, de opinion, serios, de humor, ... en realidad lo queremos todo y especialmente si es brillante. Tampoco es que tengas que deslumbrar a tu novia, que en ningun momento va a perder su tiempo en leernos, pero si tienes la mas minima idea o desvario de cualquier tipo, no te quedes pensando voy a hacerlo... hazlo!.

Tampoco queremos que te auto-juzges, deja que seamos nosotros los que digamos si es interesante o no.
Deja de perder el tiempo mirando el monitor como un memo y ponte a escribir YA!.

Como de costumbre las colaboraciones las enviais indistintamente aqui:

<set-fw@bigfoot.com>
<web@set-ezine.org>

Para que te hagas una idea, esto es lo que buscamos para nuestros proximos numeros... y ten claro que estamos abiertos a ideas nuevas....

- articulos legales: faltan derechos de autor! O tal vez sobran? No se protege merecidamente a los autores o tal vez inevntan excesivas trabas?

- sistemas operativos: hace tiempo que nadie destripa un sistema operativo en toda regla ¿alguien tiene a mano un AS400 o un Sperry Plus?

- Retro informatica. Has descubierto como entrar en la NASA con tu Spectrum 48+? somos todo ojos, y si no siempre puedes destripar el SO como curiosidad

- Programacion: cualquier lenguaje interesante, guias de inicio, o de seguimiento, no importa demasiado si el lenguaje es COBOL, ADA, RPG, Pascal, no importa si esta desfasado o si es lo ultimo de lo ultimo, lo importante es que se publique para que la informacion este a mano de todos,

eso si, No hagais todos manuales de C, procura sorpendernos con programacion inverosimil

- Chapuzing electronico: Has fabricado un aparato domotico para controlar la temperatura del piso de tu vecina? estamos interesados en saber como lo has hecho...

- Evaluacion de software de seguridad: os veo vagos, Nadie le busca las cosquillas a este software?

- Hacking, craking, virus, preaking, sobre todo cracking!

- SAP.. somos los unicos que gustan este juguete? Me parece que no, ya que hemos encontrado a alguien con conocimientos, pero: alguien da mas? Por cierto, en el proximo numero tal vez le dediquemos un articulo.

- ORACLE, MySQL, MsSQL... ¿Alguien levanta el dedo?

- LOTUS NOTES y sus bases de datos. Esta olvidado por el gran publico pero lo cierto es que muchas empresas importantes lo utilizan para organizar y distribuir la informacion internamente.

- Vuestras cronicas de puteo a usuarios desde vuestro puesto de admin...

- Usabilidad del software (acaso no es interesante el tema?, porque el software es tan incomodo?)

- Wireless. Otro tema que nos encanta. Los aeropuertos y las estaciones de tren en algunos paises europeos nos ofrecen amplias posibilidades de curiosear en lo que navega sobre las ondas magneticas. Nadie se ha dedicado a utilizar las horas tontas esperando un avion en rastrear el trafico wireless ?

- Finanzas anonimas en la red. Los grandes bancos empiezan a caer como moscas y los subrevivientes dudan mucho antes de conceder un credito. Habra empezado una nueva epoca para los usureros? Los podemos encontrar en la red?

- Finanzas a secas. Miles de blogs en la red y nadie fue capaz de ver lo que nos caia encima. Que ha pasado? Que los gobernantes sean unos incapaces ya lo sabemos, pero porque no somos lo bastante inteligentes como para propagar buena informacion en la red?

- Lo que tu quieras... que en principio tenga que ver con la informática

en fin, son los mismos intereses de los ultimos numeros....

Tardaremos en publicarlo, puede que no te respondamos a la primera (si, ahora siempre contestamos a la primera y rapido) pero deberias confiar, al ver nuestra historia, que SET saldra y que tu articulo vera la luz en unos pocos meses, salvo excepciones que las ha habido.

-[Como escribir]----------

Esperemos que no tengamos como explicar como se escribe, pero para que os podais guiar de unas pautas y normas de estilo (que por cierto, nadie cumple y nos vamos a poner serios con el tema), os exponemos aqui algunas cosillas a tener en cuenta.

SOBRE ESTILO EN EL TEXTO:

- No insulteis y tratar de no ofender a nadie, ya sabeis que a la minima salta la liebre, y SET paga los platos rotos

- Cuando vertais una opinion personal, sujeta a vuestra percepcion de las cosas, tratar de decir que es vuestra opinion, puede que no todo el mundo opine como vosotros, igual nisiquiera nosotros.

- No tenemos ni queremos normas a la hora de escribir, si te gusta mezclar tu articulo con bromas hazlo, si prefieres ser serio en vez de jocoso... adelante, Pero ten claro que SET tiene algunos gustos muy definidos: ¡Nos gusta el humor!, Mezcla tus articulos con bromas o comementarios, porque la verdad, para hacer una documentacion seria ya hay mucha gente en Internet.
  Ah!!!!, no llamar a las cosas correctamente, insultar gratuitamente a empresas, programas o personas NO ES HUMOR.

- Otra de las cosas que en SET nos gusta, es llamar las cosas por su nombre, por ejemplo, Microsoft se llama Microsoft, no mierdasoft, Microchof o cosas similares, deformar el nombre de las empresas quita mucho valor a los articulos, puesto que parecen hechos con prejuicios.

SOBRE NORMAS DE ESTILO

Tratad de respetar nuestras normas de estilo!. Son simples y nos facilitan mucho las tareas. Si los articulos los escribis pensando en estas reglas, sera mas facil tener lista antes SET y vuestro articulo tambien alcanzara antes al publico.

- 79 COLUMNAS (ni mas ni menos, bueno menos si.)

- Si quieres estar seguro que tu articulo se vea bien en cualquier terminal del mundo usa los 127 caracteres ASCII (exceptuando 0-31 y el 127 que son de control). Nosotros ya no corregiremos los que se salten esta regla y por tanto no nos hacemos responsables (de hecho ni de esto ni de nada) si vuestro texto es ilegible sobre una maquina con confiuracion extravagante.El hecho de escribirlo con el Edit de DOS no hace tu texto 100% compatible pero casi. Mucho cuidado con los disenyos en ascii que luego no se ven bien.

- Y como es natural, las faltas de ortografia bajan nota, medio punto por falta y las gordas uno entero.

  Ya tenemos bastante con corregir nuestras propias faltas.

- AHORRAROS EL ASCII ART NO NECESARIO, PORQUE CORRE SERIO RIESGO DE SER ELIMINADO (se que no estamos predicando con el ejemplo, pero el chollo se va a acabar).

- Por dios!, no utilizeis los tabuladores ni el retroceso, esta comprobado que nos levantan un fuerte dolor de cabeza cuando estamos maquetando este e-zine.

-[Nuestros mirrors]----------

NO HAY MIRRORS ESPERAMOS MIRRORS NUEVOS, LOS QUE ESTABAN YA NO ESTAN O NO ESTAN ACTUALIZADOS ¿A QUE ESPERAS A PONER SET EN TU WEB, NO TIENES ESPACIO?

-[En nuestro proximo numero]----------

Antes de que colapseis el buzon de correo preguntando cuando saldra SET 36 os respondo: Depende de ti y de tus colaboraciones.

En absoluto conocemos la fecha de salida del proximo numero, pero en un

esfuerzo por fijarnos una fecha objetivo pondremos... ya se verá, calcula entre 5 y 7 meses.

-[Otros avisos]----------

Esta vez, tampoco los hay.....

(no me cansaré de repetir las cuentas de correo)

<web@set-ezine.org>
<set-fw@bigfoot.com>

EOF

Hace apenas unos meses, en la revista @RROBA, hablamos de pasada sobre una versión de"John The Ripper" distribuida bajo un "live CD" como tantas otras versiones de linux. La característica de esta rama no oficial del famoso cracker es su capacidad de utilizar la potencia total de una maquina multicore o de poder distribuir de forma automática la tarea de crackear una hash por fuerza bruta. Nos podríamos preguntar porque los creadores de JTR no han soportado de forma activa esta rama de su criatura, razones puede que existan muchas pero lo único cierto es que si queremos hacer algo especial tenemos que andar sobre nuestras propias piernas. Hoy pretendemos contaros los problemas y posibles soluciones cuando se pretenden hacer cosas distintas a lo que nos tiene acostumbrada nuestra sociedad.

RESPUESTAS DE Solar Designer

En cualquier distribución de John The Ripper, existe en el directorio DOC un archivo llamado FAQ, ahí el creador de JTR nos dice, aparentemente, con claridad, porque no se soporta de forma activa una distribución de JTR hecha para una maquina multiprocesor o para correr bajo un proceso distribuido. Según sus propias palabras, simplemente no es necesario, ya que se puede fácilmente planear la distribución de las tareas de forma manual. Una de las maneras es lanzar john en diferentes sesiones atacando objetivos de longitud distinta. Esto se puede hacer fácilmente con el modo "incremental" y jugando con la configuración de los parámetros "MinLen" y "MaxLen". Aparentemente se pueden lanzar varias sesiones de john y diferenciar el avance a través del parámetro "--session"

No se a vosotros, pero a mi me ha dado dolor de cabeza solo imaginar la complejidad de la operación. Mejor dicho, es fácilmente explotable, si lo único que se desea es utilizar al máximo la potencia de una maquina multicore, pero si lo que nos interesa es atacar una hash difícil, lanzar la tarea y olvidarnos de la historia hasta que el objetivo haya sido alcanzado, no es precisamente el consejo de Solar el que nos puede ayudar y estamos seguros que nuestro amigo tiene la inteligencia suficiente para saberlo.

Es casi seguro que su problema reside en no verse involucrado en un ataque masivo en donde se haya utilizado su criatura en forma distribuida, ya que este tipo de cosas pueden acabar con una visita de la policía a tu casa, en el mejor de los casos o al menos esta es nuestra opinión. Pero dado que no podemos obtener respuestas fáciles y transparentes en ambientes públicos hay que plantear las preguntas en otros entornos donde podamos obtener otras respuestas.

RESPUESTAS DE SET EZINE

Como no estamos dispuestos a confiar en la primera respuesta, ya que somos escépticos profesionales, nos complace encontrar otras soluciones, lo que ocurre es que se requiere algún incentivo para que decidamos movernos y utilizar nuestras energías para conseguir un objetivo concreto, veamos como empezó todo.

El caso es que en un anónimo despacho, dentro de un flamante edificio de vidrio y cemento perteneciente a una de tantas corporaciones que velan por proveernos de servicios o cosas, vitales o inútiles, sonó un teléfono con insistencia. Nuestro viejo amigo Perico Viajero no tenia muchas prisas por responder, estaba enfrascado en responder a un e-mail idiota sin que se notara demasiado su opinión sobre el cretino que lo había escrito, cuando vió por el visor de donde procedía la llamada y se decidió a contestar. "¿Si?", "Oye, tenemos tu nuevo laptop", fue la rotunda respuesta. Era uno de los pocos dentro de la anónima sociedad, que recibía, por motivos ignotos, un cierto trato personalizado del departamento informático. "Vale, ya era hora. El que tengo empieza a tener

problemas de disco duro", contestó con soltura Viajero "Si no hicieras cosas raras con tu maquina, estas serian mas longevas. En dos días lo tendrás listo".

La nueva maquina que recibió fue un "dual core" de esos de ultima generación. Viajero se emocionó un poco al verlo, la posibilidad de utilizar la nueva capacidad de calculo mientras haraganeaba en los vestíbulos de los aeropuertos le despertó el apetito de volver a intentar nuevos ataques sobres hash que en un pasado habían resistido a sus ataques. Lo primero que se le ocurrió fue desenterrar una vieja hash de una maquina Win2000 que yacía en un despacho de un país de habla germánica. La clave pertenecía a una clave LM, que como todo el mundo debiera saber utiliza una clave de 14 byte, concatenada con ceros si no tiene esta longitud, convierte en mayúsculas todas las letras alfabéticas y dividida en dos mitades. Cada mitad es cifrada de forma separada mediante el algoritmo DES sin aplicar ningún tipo de salto. Todo ello parece ser hecho para facilitar la vida de los hackers mas tontos del mundo, pero los administradores, menos inteligentes intentan complicarles la vida usando caracteres que no existan en los teclados de tipo "QWERTY" de los mas normales o si simplemente se tiene a disposición un teclado configurado para escribir en esa lengua, es sumamente facil introducir un par de caracteres alemanes.

En el caso que nos ocupa la primera parte la clave indicaba que la password era una derivación de la palabra "administrator", pero intercalando números y caracteres especiales. Todo ello no es gran problema salvo si estos caracteres especiales son un tanto raros, tal como lengua alemana u otra cosa mas exótica, en este caso de poco sirven los múltiples diccionarios rainbow que existen en la red, solo un ataque por fuerza bruta, pero aplicando algo de inteligencia puede ser de alguna utilidad.

Viajero se planteó construir un ataque en base a "john" utilizando un modo "external" especifico para atacar cinco o seis caracteres en forma bruta. Calculaba que con los procesadores trabajando en paralelo podía atacar fácilmente cinco caracteres y si la cosa se complicaba debería plantearse un ataque conjunto con varias maquinas. Para ello pensó en utilizar la versión de "john" adaptada para trabajar en redes neuronales tipo MPI.

Todo muy bonito pero, la versión de john en cuestión solo viene distribuida bajo sus fuentes lo que significa que si se desea ejecutarlo hay que compilarlo y enlazarlo. Si ademas deseamos hacerlo bajo Windows, todo hay que hacerlo bajo este paraguas y tradicionalmente esto supone utilizar herramientas de pago. La documentación disponible no era extraordinaria pero parecía claro que era necesaria una versión de MPI instalada en la maquina donde se quería compilar. Estudiando un poco mas la web http://bindshell.net/tools/johntheripper/ vio que todas las pruebas habían sido hechas con la versión MPICH2-1.0.2 operando bajo Linux o MacOSX. Como su intención era Windows el escenario era radicalmente distinto, supuso que las cosas no iban a ser fáciles y el tiempo se encargó de darle razón.

A veces es sumamente útil cuando se emprende algo que supones va a ser complicado escribir sobre un papel cual es el objetivo. Después de reflexionar un rato, Viajero anotó algo sobre un papel y después leyó complacido lo siguiente. Compilar bajo cygwin "John The Ripper" versión MPI (MPICH2) para poderlo ejecutarlo sobre Win2000 y WinXP.

INSTALACION DE MPICH2 BAJO CYGWIN

Lo primero que hizo fue bajarse de la web del departamento "Argonne National Laboratory Group" de la Universidad de Chicago (www.mcs.anl.gov) el archivo comprimido mpich2-1.0.6.tar.gz, lo descomprimió en un directorio temporal y leyó en el README que le haría falta en un entorno linux, el compilador gcc, gfortran o g77, el compilador g++, python 2.2 o superior y como opción el compilador Fortran 90.

Después empezó con la laboriosa tarea de instalarse el cygwin con todos los anteriores requisitos. No es difícil ya que basta con bajarse de www.cygwin.com

el ejecutable setup.exe, lo mejor es colocarlo en un directorio accesorio y lanzarlo con una conexión a internet activa, ya que este pequeño programa lo que hace es proponer una instalación standard y va buscando los paquetes que le hemos indicado.

Es en la selección de los paquetes donde Viajero casi perdió la cabeza y los nervios. La tarea de seleccionar uno a uno y comprobar las dependencias no es de lo mas divertido del mundo, aunque cygwin presta una mano de forma activa. También hay que saber que en caso de dificultad recalcitrante siempre puedes elegir directamente los paquetes en www.cygwin.com/packages/

Una vez la primera parte de la tarea estaba terminada solo faltaba la segunda, o sea instalarse MPICH2. Bastaba con seguir cautelosamente las instrucciones que pudo leer en el archivo README que encontró al desempaquetar mpich2-1.0.6.tar.gz. No fue difícil, tan solo laborioso. Desempaqueto sobre un directorio de su elección "tar xfz mpich2.tar.gz" y se situó sobre el raíz "cd mpich2-1.0.6". Creó un directorio donde situaría posteriormente el paquete "mkdir /home/viajero/mpich2-install", después configuró el paquete "./configure --prefix=/home/viajero/mpich2-install 2>&1 | tee c.txt"
La ultima parte de la instrucción es todo un detalle por parte de la Universidad de Chicago, ya que, vaya todo bien o no, se crea un fichero "c.txt" donde hay copia de la ejecución. En caso de problemas Viajero sabia que podía enviarlo a "mpich2-maint@mcs.anl.gov" para ayudarle a desenredar la madeja.

Pero todo fue como una seda, lo cual es bastante raro, ya que como sabemos Viajero se encontraba bajo un cygwin y no sobre una distribución real de linux. Solo quedaba construir el paquete con "make |& tee m.txt" y tambien en este caso se crea un fichero "m.txt" que se puede enviar a sus creadores en caso de problemas. Finalmente se instalan los ejecutables y scripts en el directorio bin "make install 2>&1 | tee mi.txt" y finalmente se actualiza el PATH para que todos sepan donde están "PATH=/home/viajero/mpich2-install/bin:$PATH" , "export PATH"

Viajero temía algún problema en el momento de comprobar la instalación, pero tanto "which mpd" como "which mpiexec" dieron los resultados apetecidos. MPICH2 estaba funcionando correctamente bajo cygwin. Paso libre para compilar John The Ripper, versión MPI.

Aunque ahora se encuentran en la versión "http://bindshell.net/tools/johntheripper//john-1.7.2-bp17-mpi7.tar.gz" y fue ésta la que utilizó Viajero para hacer sus pruebas. Descarga sobre un directorio temporal dentro de cygwin y desempaquetado de forma standard da como resultado tres directorios y un fichero README. Del README rápidamente se dio cuenta que lo mejor era "pasar" de ,l. Se volvió hacia el directorio "src" e intentó lo clásico con "John The Ripper", un simple "make win32-cygwin-x86-sse2" debería rápidamente crear un ejecutable en el directorio "run", sin embargo lo que cosechó fue un mensaje de error diciendo que no se encontraba "mpicc", en concreto el mensaje era "make[1]: mpicc: Command not found". "mpicc" es es solo una utilidad creada especialmente para poder compilar en ambientes MPI. Una consulta rápida a la mail list (john-mpi-subscribe@bindshell.net) recibió la sorprendente respuesta de que lo mejor era que utilizara gcc en lugar de mpicc

Con un editor de lo mas normal, Viajero, substituyó en el fichero "Makefile" todas las ocurrencias "mpicc" por "gcc" y las cosas marcharon un poco mejor,... pero solo un poco. Ahora el mensaje fue "bench.c:38:17: mpi.h: No such file or directory". El mensaje no podía ser mas claro, "make" era incapaz de encontrar las librerías que pertenecen a la distribución de MPI. Aparentemente el problema es fácil de resolver, basta con añadir al PATH de cygwin la dirección donde se encuentran las fuentes que deseamos utilizar. Viajero deseaba hacerlo de forma inteligente, o sea que no tuviera que modificar PATH a cada arranque, y esto se consigue modificando el fichero "cygwin.bat" que se encuentra en directorio raíz de cygwin. Con añadir la linea "PATH=%PATH%:/dirección-deseada" al susodicho "bat" y a correr.

Digamos que correr no corrió mucho, ya que la siguiente compilación le dejo un amargo sabor de boca con un esóterico mensaje sobre una linea de uno de los includes. No era posible. Algo iba irremediablemente mal. Volvió a consultar la lista de distribución de bindshell que le respondieron rápidamente. "Si quieres compilar un programa basado en MPI bajo cygwin y que posteriormente corra bajo Windows, tienes que borrar la instalación de MPICH2 bajo cygwin e instalar la versión para Windows". Se dice mas pronto de lo que se hace y a Viajero le dio dolor de cabeza nada mas pensar en lo que se le venia encima, pero nada se termina si no se empieza, así que lo mejor en estos casos es dejarse de lamentarse y ponerse manos a la obra.

INSTALACION DE MPICH2 BAJO WinXP

Viajero empezó por el principio, o sea buscar las herramientas adecuadas en este caso el instalador. Está en
"http://www.mcs.anl.gov/research/projects/mpich2/downloads/index.php?s=downloads"
, elegir el fichero correcto, "mpich2-1.0.7-win32-ia32.msi" y en este caso copiarlo en un sitio que después se debe recordar y lanzarlo con derechos de administrador. El directorio por defecto donde se instala es
"C:\Program Files\MPICH2" y bajo ,l se encuentran "include", "lib", donde se encuentran "headers" y librerias y tambien el directorio "bin" donde están los ejecutables. Durante la instalación se le pidió una palabra de paso que en teoría después se debe reutilizar sobre el resto de maquinas donde se quiera monta el anillo de CPUs, pero de eso hablaremos mas tarde. Viajero se limitó a crear un password que no tuviera nada que ver con ,l ni con sus andanzas mas habituales y anotarla cuidadosamente.

Terminado esto, tuvo que desinstalar laboriosamente todo lo que se encontraba bajo cygwin para evitar interferencias entre las dos instalaciones sin olvidarse de todos los PATH habidos y por haber, que en esto, todo el entorno de Windows es sumamente quisquilloso y tiene una memoria prodigiosa e inoportuna que fastidia cuando menos te lo esperas. Finalmente todo quedó listo para la siguiente etapa. La compilación.

LA COMPILACION

Contento como niño con zapatos nuevos, Viajero volvió a probar con "make win32-cygwin-x86-sse2", con desesperación recibió de nuevo una catarata de mensajes que se podían reducir a que seguían sin encontrarse las librerías. Y sin embargo Viajero había modificado convenientemente el PATH de cygwin para que buscara los programas en donde habían sido instalados automáticamente por MPICH2 o sea en "C:/Program Files/MPICH2/include" ¿Que había podido fallar? Le costó bastante darse cuenta que estaba cometiendo un error de principiante. "make" no va a buscar las librerias y headers en función del contenido de las variables de entorno, hay que indicarse lo de forma explicita en el propio "Makefile". La razón no la conocía, pero bien pudiera ser que los programadores de esta utilidad se quisieran asegurar que las cosas estaban donde el programador desea y no donde le parece al Sistema Operativo el mejor sitio. Esto los utilizadores de Windows puede que no lo entiendan, pero los de Unix/Linux lo comprenden perfectamente.

Viajero modificó levemente el Makefile para que encontrara la joya de la corona. En este caso ademas había que acordarse que se estaba compilando bajo cygwin y por tanto las cosas están en sitios que aparentemente no son los mas evidentes. En este caso tuvo que cambiar la linea
"CFLAGS = -c -Wall -O3 -fomit-frame-pointer" por
"CFLAGS = -c -Wall -O3 -fomit-frame-pointer
          -I/cygdrive/c/Progra~1/MPICH2/include"
Véase con atención que los includes apuntan a
"/cygdrive/c/Progra~1/MPICH2/include" y no a
"Progra~1/MPICH2/include" ya que cygwin mapea todo lo que se encuentra bajo el disco principal a un dispositivo virtual llamado "/cygwin/c/". Es una de las

características y gracias de este simulador. A Viajero le salieron algunas canas hasta que no se dio cuenta de lo que pasaba.

Quien piense que hasta aquí llegaron las desventuras de Viajero, intuirá que nada mas lejos de la realidad, aunque solo sea porque el articulo le falta un buen trozo para terminar. El siguiente problema fue que "make" se lamentaba con amargura que no encontraba algunos componentes de las librerías. Vuelta a consultar a la lista de distribución para descubrir que las librerías no son las mismas si se quiere compilar desde cygwin utilizando una distribución MPICH2 instalada bajo cygwin que si esta instalada en Windows. Afortunadamente fue rápida la solución ya que le informaron de exactamente cuales eran las librerías a copiar en "/MPICH2/lib" y substituir la linea en Makefile que decía
"LDFLAGS = -s -lm" por otra donde ponía
"LDFLAGS = -s -lm -L/cygdrive/c/Progra~1/MPICH2/lib -lmpicxx -lfmpich2 -lmpi"
Aquí hay un pequeño detalle, mpicxx.h es un header y no una librería, pero el caso es que las quejas sobre las librerías inexistentes desaparecieron, solo para que aparecieran nuevos problemas.

El mensaje era siempre parecido
"john.o:john.c:(.text+0x33): undefined reference to `_MPI_Init'", o sea que había un procedimiento, variable o parámetro que se no encontraba en ninguna parte. Fueron duros días los que tuvo que sufrir Viajero, ya que _MP_Init esta claramente definido en "mpi.h" y sin embargo parecía ser ignorada durante la operación de enlazado. Fueron de nuevo los chicos que desarrollan MPICH2 que le identificaron el problema y le indicaron la solución. Es algo no ciertamente intuitivo y forma parte de la filosofía que siguen los enlazadores a la hora de unir todas las piezas precompiladas. Hay que acordarse que en el momento de enlazar, si el archivo "a.o" utiliza símbolos definidos en el archivo "b.o", hay que especificar "a.o" antes que "b.o". Si, definitivamente no es intuitivo, pero el caso es que cuando Viajero cambio la linea
"$(LD) $(JOHN_OBJS) -lkernel32 -o ../run/john.exe"
por la linea
"$(LD) $(JOHN_OBJS) -lkernel32 $(LDFLAGS) -o ../run/john.exe"
se acabaron sus problemas y encontró un magnifico fichero llamado john.exe bajo el directorio "run". Ya tenia el ejecutable, ahora no había mas que utilizarlo.

MPICH2, JOHN, DUAL CORE.

La instalación de MPICH2 bajo Windows no solo crea una serie de librerías y headers, también instala un servicio llamado "smpd" y un ejecutable llamado "mpiexec.exe" que sirve para lanzar el programa "MPIzado". De todas formas Viajero era la primera vez que utilizaba estos entornos y siempre los comienzos son difíciles, así que hechó mano de un par de utilidades que vienen con la distribución. "wmpiregister" sirve para configurar un anillo y como el primer paso de Viajero era solo aprovechar los dos procesadores de un "DUAL CORE" supuso que podía pasar de sus servicios. Después esta "wmpiregister" muy util para registrar un nombre y su password en el registro de Windows. De utilidad dudosa si eres el administrador de la maquina. El que tiene muchas mas posibilidades es "wmpiexec". Es simplemente un "front end" que hecha una mano. Viajero tuvo que especificar donde estaba el "john" que quería lanzar y las diversas opciones. Lo mas importante era tener presente que solo deseaba lanzar su programa en una sola maquina, para hacerlo tuvo que checkear "more options" y en la opción "host" poner "localonly".

Al lanzar el "john" mediante "execute" aparentemente todo funcionó correctamente, al comprobar mediante "Task Manager" la situacion de las dos CPUs, comprobó que ambas estaban al 100%. Pensó que ya habia encontrado la solución definitiva hasta que se le ocurrió parar el ataque, lo intentó, pero no lo consiguió, un extraño bug en el programa se lo impedía. Tuvo que pararlo con el "Task Manager" y rearrancarlo, después con la opción "shown" vio exactamente cuales eran las opciones y comandos que utilizaba MPI, después abrió una sesión "cmd" y desde allí recopió lo que había observado. Y desde allí si que podia pararlo, el problema es que no podía utilizar prácticamente la maquina ya que ésta consumía todos los recursos para "john-mpi". Viajero lo

que deseaba era que su maquina consumiera todos los tiempos muertos, no que lo matara con esperas desesperantes para abrir una vulgar hoja "excel".

La solución se encontraba en una de las múltiples opciones que tiene "mpiexec" y que consiste en fijar la prioridad de la sesión. Finalmente lanzo una sesión con la instrucción
"mpiexec.exe -hosts 1 localonly 2 -localonly -priority 1
-noprompt john.exe -session=pims -e=prepend-pims hash-pims.txt",
ya que tenia una serie de hash interesantes en el fichero hash-pims.txt y había creado un modo especial extendido con el nombre de "prepens-pims", para poder recuperar la sesión la había denominado "pims".

El resultado fue bastante agradable, ambos procesadores marcaban al 100%, pero cuando quería trabajar para su empresa, la carga se reducía automáticamente, con el único peaje de una ligera espera cuando cambiaba de tareas. Todo parecía funcionar correctamente y Viajero muy contento, interrumpió la sesión y construyó un fichero "bat" que contenía la siguiente instrucción
"mpiexec.exe -hosts 1 localonly 2 -localonly -priority 1 -noprompt john.exe
-restore=pims"
y que copió en su directorio "...\Programs\Startup". Seguro de su éxito no hizo mas pruebas.

A la mañana siguiente, cuando llego a su trabajo, después de hacer los falsos saludos de todos los días, conectó y encendió su ordenador, que después de hacer todas las comprobaciones y rutinas que Windows nos tiene acostumbrado, lanzó su especialmente construido fichero "bat",.... que en unos segundos se detuvo. Según el mensaje en pantalla no existían ni "john_rec.rec.0" ni tampoco "john_rec.rec.1". Evidentemente hay un bug en el programa. John crea por defecto un fichero "john.rec" donde se guardan los datos necesarios para recuperar una sesión interrumpida. En caso de que se especifique una sesión con un nombre en concreto, por ejemplo "pims", se crea un fichero llamado "pims.rec". Como en este caso había que recuperar dos sesiones, una para cada procesador, john había creado dos ficheros distintos denominados "pims.rec.o" y "pims.rec.1", pero después hacia caso de la opción "restore=pims" y seguía buscando sus amados "john.rec".

A Viajero le dolía la cabeza ya que mientras estaba buscando el problema una amable secretaria de esas que parecen puestas en este mundo solo para ser lo mas inoportunas e inútiles, le había estado dando la lata sobre un documento, evidentemente anodino, que debía ser firmado con urgencia. Por tanto ni se le ocurrió buscar el bug en los fuentes, simplemente cambio los nombres de los ficheros para que john los encontrara, este los descubrió a la primera. El contento y Viajero mas.

Solar Designer, al que hemos mencionada al principio del articulo podrá decir lo que quiera, pero lo cierto que un john "MPIzado" de esta forma se comió un par de hash en apenas dos días , cuando la experiencia de Viajero le decía que le hubiera costado el doble con el método clásico y todo ello sin tener que preocuparse ni de repartir tareas ni de hacer cálculos. Dos únicos inconvenientes. Probablemente debido a otro bug en el programa no se puede ver el avance del ataque y el otro es que la maquina funciona a plena potencia desde el momento en que se enciende y esto, en opinión de Viajero, no puede ser bueno par la salud del laptop ni para su longevidad. De todas formas la password alemana que fue el motivo de toda esta carrera contra los elementos cayó en cosa de una semana.

MPICH2, JOHN, EN RED

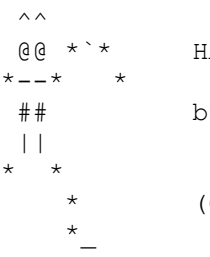
Perico Viajero tampoco deseaba freír la maquina que habían depositado a su cuidado. Puede que a otros no les guste que la inversión de sus escuálidos ahorros desaparezca en una nube de humo, poco virtual y bastante maloliente. Viajero pensó que otra opción era organizar un anillo de maquinas. Según la documentación de MPI esto es posible hacerlo si somos administradores del DOMAIN de la red, pues es ella la encargada de permitir la entrada en cada una

de las maquinas que la componen, en la practica esto no es cierto, basta con tener el mismo usuario registrado en las maquinas que deseamos utilizar para que estas se pongan a trabajar obedientemente. Viajero hizo la prueba con éxito con dos maquinas fijas, una con Windows 2000 y otra con WinXP. A continuación añadió un viejo laptop que ni siquiera tenia SSE2. Como anécdota, esta vieja reliquia se negaba a hacer su parte y Viajero tuvo que recompilar john con la option "any" y esta vez el viejo laptop se agregó obedientemente al anillo aportando su granito de potencia.

Por falta de espacio no podemos detallar los entresijos de la operación, que tampoco es sencilla, pero si demuestra que es posible infectar varias maquinas en una red y conseguir que se pongan a trabajar en tu honor para fines honestos o deshonestos.

www.set-ezine.org
web@set-ezine.org

EOF



HACK THE WORLD

by blackngel <blackngel1@gmail.com>
<black@set-ezine.org>

---[1 - Prologo

A quien le puede interesar espiar conversaciones? No encontrar respuesta a esta pregunta resulta francamente complicado. Que sea o no eticamente correcto es un tema tratado ya en demasia.

Los programas son criaturas extra~as, la mayoria de las veces complejas y otras tantas son obras demasiado (+++) personales. Cuando uno deja de ser un simple novato y comienza a interesarse en como funcionan las cosas e incluso decide meter su cabeza en el codigo, no todo es color de rosa.

Sniffar conversaciones realizadas a traves del software "Messenger" puede ser algo interesante. Y aqui no me refiero a la accion en si, sino al ambiciado: "como hacerlo?".

Entonces nos atrevemos, bajamos codigo y mas codigo, lo destripamos, adivinamos donde esta el comienzo y creemos descubrir una especie de estructura interna. Luego todo se complica. Sentencias sin sentido, procedimientos incomprensibles y un inmenso oceano donde no podemos hacer otra cosa mas que ahogarnos y desistir.

Y todo ello no es porque "el Programador" sea una entidad retorcida cuya unica intencion sea hacer su codigo indescifrable (puede serlo y puede llegar ser divertido [descifrarlo]). La razon es que cuando uno consigue una base lo suficientemente solida como para mantenerse por si misma, el programador no puede evitar la incesante necesidad de a~adir nuevas funciones, introducir nuevas variables, y reestructurar el codigo hasta tal punto que la idea original para el que fue creado se torne ilegible (a pesar de que siga funcionando igual o mejor que antes).

Que pretende este articulo? Facil: Como capturar conversaciones de "Messenger" con menos de 200 lineas de codigo (no contabilizando comentarios claro esta). El metodo que usaremos aqui, puede ser reutilizado para sniffar conversaciones irc, aim, yahoo, icq e incluso para capturar direcciones URL.

Plantar un sniffer en tu propio ordenador para capturar tus conversaciones puede parecer inutil excepto cuando lo situas en los PC's de tus empleados y esperas a saber a que se dedican realmente en horario de trabajo (espiar a tu novia es mas inmoral todavia); pero cuando entra en juego un ataque "Man In The Middle", entonces todo cambia. Esto no deberia requerir mas explicaciones...

---[2 - Introduccion

Requisitos:

 - LibPcap (version 0.8) [1]
 - LibNet (version 1.1 o superior) [2]

Compilar con:

 - $ gcc codigo.c -lpcap -lnet -o sniffmsn

Uso:

 - $ sudo ./sniffmsn dev

No se entrara en detalles acerca de la implementacion y argumentos de las funciones de la libreria libpcap, sin embargo, se ense~ara el cometido de cada una de ellas, pues para eso estamos aqui, para comentar el codigo.

Libnet podria ser evitado pero utilizaremos sus estructuras de cabecera para protocolos debido a que son verdaderamente intuitivas y facilitan nuestra tarea.

El metodo utilizado para crear el codigo que veremos en la siguiente seccion se basa en algo tan sencillo como interpretar las capturas de nuestro amigo: "Wireshark" (antes conocido como Ethereal). Con esto conseguiremos saber como funciona el protocolo "msn" y de aqui pasaremos a capturar los paquetes que pasen por nuestra interfaz de red para luego extraer la informacion que necesitamos.

Te parece que el formato de salida de Wireshark es criptico? Podria decirte que dejaras de leer esto, pero esta actitud ampliamente emulada no es etica; simplemente te dire que tu experiencia no es la suficiente y precisas de alguna que otra lectura previa. Acaba de leer este articulo y vuelve a el cuando hayas acumulado un conocimiento extra.

---[3 - Protocolo MSN

Si piensas que aqui se va dar una explicacion en profundidad acerca del citado protocolo, estas equivocado. Google es tu amigo pero aqui tienes una ayuda [3].

Para que nos entendamos, podriamos decir vulgarmente que este protocolo actua a traves de comandos. Algunos de ellos tales como: CAL, JOI, USR, XFR, MSG, y unos cuantos mas.

Es este ultimo el que en este momento nos conviene estudiar. Estamos a punto de destriparlo.

Para seguir lo que viene a continuacion deberias iniciar una sesion Msn con tu cliente favorito que podria ser: aMsn, Pidgin o cualquier otro. Antes de iniciar esta sesion abriras una pantalla de Wireshark como root y pondras a la escucha la interfaz de red que te conecta a internet. En las opciones de captura debes escribir lo siguiente:

   -> "tcp and src port 1863 or dst port 1863"

* 1863 es el puerto utilizado por los servidores de Messenger.

De este modo filtraremos solo los paquetes que nos interesan y evitaremos un monton de basura innecesaria.

Inicia entonces una conversacion con alguno de tus amig@s y cuando hayas intercambiado unas cuantas frases, despidete (te lo ordeno, aqui no estamos para chatear };-D) y finaliza la captura de Wireshark. Es conveniente que guardes los paquetes capturados en un archivo para que puedas utilizar este mismo modelo tantas veces como quieras y evitar repetir el proceso anterior.

Vamos al meollo del asunto:

En el campo INFO de varios paquetes capturados podremos observar algunos de los comandos mencionados anteriormente junto con algun otro tipo de datos que solo al lector avanzado interesan.

Bien, nuestro objetivo es buscar paquetes cuyo campo INFO contiene el comando "MSG" pero, alto ahi, no todos son interesantes y vamos a suprimirlos. Iremos situandonos encima de estos paquetes y desplegaremos la pesta~a de "MSN Messenger Service". En ella encontraremos toda la informacion que necesitamos. Aquellos cuyo campo "Content-Type:" sea diferente a "text/plain", podemos obviarlos, contienen informacion de control pero no mensajes legibles. Aquellos que lo contengan, son los nuestros.

Destriparemos ahora el resto de los campos:

Si nosotros enviamos el mensaje el comando se muestra como sigue:
MSG 13 A 165\r\n -> El comando, indicando el numero de caracteres incluyendo el mensaje y el resto de parametros a partir de aqui.

Si es nuestro amigo el que nos habla a nosotros, se vera asi:
MSG amigo@dominio.com I-Hack-The-World 165\r\n -> El comando, el e-mail y la frase o nick que le hace mas CooL.

MIME-Version: 1.0\r\n
Content-Type: text/plain; charset=UTF-8\r\n -> Version MIME, texto plano y estandar de codificacion Unicode

User-Agent: pidgin/2.4.3\r\n -> Puede estar o no! Software utilizado para la conexion.

X-MMS-IM-Format:FN=MS%20Sans%20Serif; EF=; CO=0; PF=0; RL=0\r\n
-> Si alguna vez te has preguntado como puede ser que la fuente y el color de la letra que tu eliges puede verse tambien en el PC de la persona con la que mantienes una charla, aqui tienes la respuesta. Interpreta los "%20" como espacios (deberias saberlo).

\r\n -> Retorno de carro y nueva linea adicional (ayuda, lo veremos).

eres un hacker? -> He aqui el mensaje tan esperado. El que nosotros hemos enviado o recibido.

Vale, hasta aqui tenemos bastante materia. Podrias cerrar este articulo y presumir delante de tus amig@s que capturas conversaciones mediante Wireshark. Si tienes un poco de cabeza y pretendes utilizarla, supondre que no perderas tu tiempo buscando frases entre la inmensa cantidad de paquetes almacenados y que estas dispuest@ a llegar mas alla. Como era? Ah si: "haztelo tu mismo".

Te suena?

---[4 - Codeando

Bueno, con lo anterior, la idea general del programa resulta mas que sencilla:

1 # - Capturar paquetes con libpcap.

```
 # 2 # - Filtrar los que vayan o provengan del puerto 1863 (msn).
 # 3 # - Copiar los datos del paquete a un buffer y diferenciar entre enviados y
        recibidos para guardar el e-mail/nick del receptor en caso necesario.
 # 4 # - Seleccionar aquellos que contengan el comando "MSG" y coincidan con la
        cadena "Content-Type: text/plain".
 # 5 # - Desplazarnos hasta la cadena del mensaje e imprimirla.
```

A partir de aqui se ira mostrando el codigo convenientemente comentado y respetando, ante todo, el orden de procesamiento anterior.

```
 **------------------------------------------------------------------------**

 // Archivos cabecera y definiciones

 #include <stdio.h>
 #include <pcap.h>
 #include <libnet.h>

 #define MSN_PORT 1863
 #define TCPHDR_LEN 0x20 /* Esto es interesante, algunas veces podrias
                            necesitar cambiar esto por "0x14" para que
                            funcione. Ello es debido a que diversas opciones
                            del protocolo TCP puden cambiar el tama~o de
                            la cabecera de 20 a 32 bytes
                         */
 #define FILTRO_MSN "tcp and src port 1863 or dst port 1863"

 // Declaraciones de funciones

 static char * get_ident(char *);
 void print_msg(char *, int, char *, char *);
 void handle_msg(char *, char, int);
 void read_msn(u_char *, const struct pcap_pkthdr *, const u_char *);

 // Funcion Principal

 int main(int argc, char *argv[])
 {
   int rc;
   char *device;
   char errbuf[PCAP_ERRBUF_SIZE];
   struct bpf_program filtro;
   bpf_u_int32 netp, maskp;
   pcap_t* sniffmsn;

   if (argc < 2)
      exit(0);

   device = argv[1]; // Unico parametro: interfaz de red. Ex.: eth1, ath0, etc.

   // Abrimos el dispositivo para captura en modo promiscuo
   sniffmsn = pcap_open_live(device, 1600, 1, 20, errbuf);
   if (sniffmsn == NULL) {
      fprintf(stderr, "pcap_open_live(): %s\n",errbuf);
      exit(-1);
   }
   // Obtenemos la direccion y la mascara de red de la interfaz
   if (pcap_lookupnet(device, &netp, &maskp, errbuf) == -1) {
      fprintf(stderr, "Error en pcap_lookupnet(): %s\n", errbuf);
      exit(-1);
   }
   // Creamos el filtro con las opciones anteriormente definidas
   if (pcap_compile(sniffmsn, &filtro, FILTRO_MSN, 0, netp) == -1) {
      fprintf(stderr, "Error compilando el filtro\n");
```

```
    exit(-1);
  }
  // Aplicamos el filtro a la interfaz
  if (pcap_setfilter(sniffmsn, &filtro) == -1) {
    fprintf(stderr, "Error aplicando el filtro\n");
    exit(-1);
  }
  // Iniciamos la captura de paquetes indefinidamente, el 3er parametro es la
  // funcion que se encarga de interpretar los paquetes, siempre tiene tres
  // parametros y no tiene valor de retorno.
  rc = pcap_loop(sniffmsn, -1, &read_msn, NULL);

  return 0;
}

// Funcion que interpreta los paquetes. Extraemos las cabeceras TCP e IP y
// controlamos el payload (carga o datos) que se pasara a la siguiente
// funcion diferenciando entre enviados y recibidos para leer los que sean
// realmente interesantes.

void
read_msn(u_char *useless, const struct pcap_pkthdr *pkthdr, const u_char *pkt)
{
  u_char *data;
  int len;

  struct libnet_ipv4_hdr *iph; // Cabecera IP
  struct libnet_tcp_hdr *tcph; // Cabecera TCP

  iph = (struct libnet_ipv4_hdr *)(pkt + LIBNET_ETH_H);
  tcph = (struct libnet_tcp_hdr *)(pkt + LIBNET_ETH_H + LIBNET_IPV4_H);

  // Los datos se alcanzan tras pasar las cabeceras: ethernet, ip y tcp.
  data = (u_char *)(pkt + LIBNET_ETH_H + LIBNET_IPV4_H + TCPHDR_LEN);
  len = ntohs(iph->ip_len);

  // Si el puerto origen es 1863, estamos recibiendo un mensaje.
  if (ntohs(tcph->th_sport) == MSN_PORT) {
    handle_msg(data, 'R', len); // Maneja los datos que anteriormente
                                // destripamos: comando, opciones y mensaje.
  }
  // Si el puerto destino es 1863, estamos enviando un mensaje
  else if (ntohs(tcph->th_dport) == MSN_PORT) {
    handle_msg(data, 'S', len);
  }
}

//

void
handle_msg(char *data, char dir, int dlen)
{
  char *pc, *pstart;
  char *email;
  char *nick;
  char *buf;

  // Creamos un buffer con la longitud del payload
  buf = (char *) calloc(dlen, sizeof(char));
  // Copiamos alli su contenido para manejarlo
  if (buf != NULL) {
    strncpy(buf, data, dlen);
  } else {
    fprintf(stderr, "\nNo hay suficiente memoria\n");
```

```
      exit(-1);
   }

   // Comprobamos que contenga el comando "MSG"
   if (strncmp(buf, "MSG", 3) == 0) {

      // Que su contenido sea texto plano y no datos de control
      if (strstr(buf, "Content-Type: text/plain") != NULL) {

         // Mensajes enviados
         if (dir == 'S') {
            // Nos situamos en el primer parametro del comando MSG
            pc = strchr(buf + 4, ' ');
            pc++;
            // Funcion que alcanza el mensaje y lo imprime. El ultimo parametro
            // es nulo porque no imprimimos el e-mail del emisor, nosotros.
            // Deberiamos hacerlo si utilizamos ataques MITM.
            print_msg(pc, dir, NULL, NULL);
         }

         // Mensajes recibidos
         else {
            // Lo mismo que antes pero esta vez colocamos un caracter nulo al
            // final de la direccion e-mail del receptor para poder manejar
            // este fragmento como una cadena.
            pstart = buf + 4;
            pc = strchr(pstart, ' ');
            if (pc != NULL)
               *pc = 0;
            email = get_ident(pstart); // Esta funcion, que sera descrita al
                                       // final del codigo, no es mas que un
                                       // peque~o administrador de memoria
                                       // dinamica, que reserva el espacio
                                       // suficiente para almacenar e-mail y
                                       // nick, evitando buffer's estaticos y
                                       // los consecuentes overflows.

            // Mismo procedimiento para almacenar el nick.
            pc++;
            pstart = pc;
            pc = strchr(pstart, ' ');
            if (pc != NULL)
               *pc = 0;
            nick = get_ident(pstart);

            pc++;
            print_msg(pc, dir, email, nick); // Imprimir mensaje
         }
      }
   }

   // Trabajar limpiamente
   free(buf);
   free(email);
   free(nick);
}

void
print_msg(char *pc, int dir, char *mail, char *nick)
{
 char *str, *str_end;
 int len;

 // Parametros validos para el comando "MSG"
```

```
 if (*pc == 'U' || *pc == 'N' || *pc == 'A') {
   str = strchr(pc, ' ');
   str++;
 }
 else {
   str = pc;
 }

 // Justo antes del primer retorno de carro se encuentra la longitud del
 // mensaje, lo almacenamos en la variable 'len'.
 str_end = strchr(str, '\r');
 *str_end = '\0'; // Ya saben, para manejar cadenas deben terminar en \0.
 len = atoi(str);
 str = str_end + 2;
 *(str + len) = '\0';

 // Muy facil, gracias al retorno de carro y nueva linea adicional que el
 // protocolo MSN nos facilita, el mensaje real siempre se encontrara despues
 // de dos retornos de carro y nueva linea consecutivos.
 str = strstr(str, "\r\n\r\n");
 str += 4;

 if (dir == 'S')
    printf("\nSEND MSG: %s\n", str); // Mensajes enviados
 else // Mensajes Recibidos
    printf("\nRECV MSG from (%s) [%s]: %s\n", mail, nick, str);

}

// Administrador de memoria dinamica para almacenar cadenas.

static char *
get_ident(char *ptr)
{
  char *buff;
  size_t bsize = 32; // Tama~o de buffer inicial
  int lenp = strlen(ptr);

  // Mientras la longitud de la cadena sea mayor que el tama~o de buffer,
  // lo vamos multiplicando por 2 hasta que la capacidad sea suficiente.
  while (lenp > bsize) {
     bsize *= 2;
  }

  buff = (char *) calloc(bsize, sizeof(char)); // Creamos el buffer con 0's

  if (buff != NULL) {
     strncpy(buff, ptr, bsize); // Copiamos la cadena al buffer
     return buff;
  } else {
     fprintf(stderr, "\nNo hay suficiente memoria\n");
     exit(-1);
  }
}

**------------------------------------------------------------------------**
```

---[5 - Conclusion

Espero que esto haya sido lo suficientemente interesante como para llegar hasta aqui. El codigo es muy basico, pues en todo instante evitamos irnos por las ramas centrandose unica y exclusivamente en la idea principal.

Aunque pueda parecer mentira, una de las partes mas interesantes del codigo es el peque~o administrador de memoria "get_ident(...)". Con el podemos aprender practicas de programacion bastante mas adecuadas en la actualidad. Debemos evitar caer en las tentaciones del programador perezoso y comportarnos de acuerdo al status que proclamamos.
"Programacion en Linux: Casos Practicos" [4], podria ser una buena lectura si pretendes que tu vision ante la programacion cambie de forma notable.

Cualquier duda podeis consultarla en <blackngel1@gmail.com>, leo el correo a diario, aunque solo sea para leer las ultimas de Hispasec [5].

---[6 - Referencias

[1] LibPcap
    http://sourceforge.net/projects/libpcap/

[2] LibNet
    http://libnet.sourceforge.net

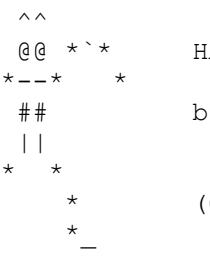
[3] Protocolo MSN
    http://65.23.158.196/nitz/nitz/protocolo_msn.pdf

[4] Programacion en Linux: Casos Prácticos - ISBN: 978-84-415-1839-1
    http://www.anayamultimedia.es

[5] Hispasec Sistemas
    http://www.hispasec.com

EOF

-[0x0C]--
-[Interfaces con SDL]---
-[by blackngel]---SET-35--



HACK THE WORLD

by blackngel <blackngel1@gmail.com>
<black@set-ezine.org>

---[1 - Prologo

Aqui comienza el camino, estas dispuesto a recorrerlo?

Si estas aburrido de escribir programas en modo consola... si has experimentado con "ncurses" y su potencia o gracia no te acaba de persuadir; pero todavia no quieres introducirte en el mundo de Glade, GTK, Borland C++ Builder o cosas por el estilo, entonces, y solo entonces puede que SDL sea lo que estas buscando.

SDL es una libreria pensada inicialmente para la creacion de videojuegos en 2D (puede ayudar en 3D junto con OpenGL). Pero este articulo no se centra en tal habilidad. Nosotros aprovecharemos esta libreria para crear interfaces graficas de usuario, mas conocidas como GUI.

Esta libreria nos dara una sensacion de Programacion Orientada a Eventos

(POE, oh Edgar... };-D)

---[2 - Introduccion

SDL [1], o Simple DirectMedia Layer es compatible con la mayoria de Sistemas Operativos, incluyendo Linux, Windows, MacOS, las variantes BSD, y muchos otros.
Desarrollaremos el codigo bajo Linux y mostraremos como compilar los programas correctamente, no obstante, portar todo lo aqui descrito a cualquier otro sistema acaba resultando en algo trivial.

Una de las caracteristicas mas importantes de SDL es su division en subsistemas tales como video, audio, eventos, cdrom, red, hilos (threads), manejo de texto y mas... varios de estos subsistemas forman parte de extensiones que han venido al rescate de la, a veces, arcaica base de SDL. Se explicaran en su momento y se indicaran las opciones de compilacion correspondientes.

Gracias a esta forma de trabajar, nosotros podemos elegir los que nos interesen y empezar a desarrollar nuestras aplicaciones inmediatamente.

Si necesitas un peque~o adelanto para ir cogiendo ideas, esto es para ti [2].

********** NO SUPRIMIRE EL MANEJO DE ERRORES EN EL CODIGO. MUCHOS LO
* IMPORTANTE * HACEN Y SOLO CONSIGUEN CREAR APTITUDES DE PROGRAMACION
********** PEREZOSAS, INCORRECTAS Y LA MAYOR DE LAS VECES PELIGROSAS.

---[3 - GUI's con SDL

Bien, empezaremos por a~adir a nuestro programa la cabecera principal:

```
#include "SDL/SDL.h"
```

Definiremos una superficie principal que representara la pantalla principal de la aplicacion durante la ejecucion del programa:

```
SDL_Surface *pantalla;
```

Ahora, lo principal es iniciar el sistema SDL en si. La siguiente funcion nos proporciona esta facilidad:

```
atexit(SDL_Quit);

if (SDL_Init(SDL_INIT_AUDIO|SDL_INIT_VIDEO) < 0) {
   fprintf(stderr, "Error al iniciar SDL: %s\n", SDL_GetError());
   exit(-1);
}
```

La primera llamada puede haber llamado tu atencion, pero es muy sencillo. La funcion 'atexit()' registra el nombre de las funciones que seran llamadas antes de la finalizacion del programa.

Como puedes observar, al iniciar el sistema basico tambien podemos arrancar otros subsistemas mediante el uso de constantes combinadas con un OR logico. Podemos utilizar otros como:

```
SDL_INIT_CDROM
SDL_INIT_TIMER
SDL_INIT_JOYSTICK
```

O iniciarlos todos con:

```
SDL_INIT_EVERYTHING
```

Si has olvidado iniciar un subsistema y no lo has indicado en esta funcion, todavia puedes iniciarlo de esta forma:

```
if (SDL_InitSubSystem(SDL_INIT_JOYSTICK) == -1) {
   fprintf(stderr, "No se puede iniciar el joystick %s\n", SDL_GetError());
   exit(1);
}
```

Podras detenerlos analogamente con la funcion 'SDL_QuitSubSystem()';

Establezcamos seguidamente el modo de video, que creara nuestra ventana principal con la resolucion y profundidad de color que indiquemos y ciertos atributos que explicaremos a continuacion:

```
pantalla = SDL_SetVideoMode(1024, 768, 24, SDL_ANYFORMAT | SDL_DOUBLEBUF);
if (pantalla == NULL){
   fprintf(stderr, "No se pudo iniciar el modo de pantalla: %s\n",
                   SDL_GetError());
   SDL_Quit();
   exit(1);
}
```

OPCIONES (flags):

```
SDL_SWSURFACE  -> Superficie en memoria del sistema.
SDL_HWSURFACE  -> Superficie en memoria de video.
SDL_ASYNCBLIT  -> Actualizacion asincrona.
SDL_ANYFORMAT  -> Fuerza el uso de los bpp de la surface actual. Hay que usarlo
                  cuando queramos crear la superficie en una ventana.
SDL_HWPALETTE  -> Da a SDL acceso exclusivo a la paleta de color.
SDL_DOUBLEBUF  -> Solo valido con SDL_HWSURFACE. Tecnica del "doble buffer".
SDL_FULLSCREEN -> Visualizacion a pantalla completa.
SDL_OPENGL     -> Crea un contexto OpenGL.
SDL_OPENGLBLIT -> Igual a la anterior, pero SDL hace el renderizado 2D.
SDL_RESIZABLE  -> La ventana puede cambiar de tama~o.
SDL_NOFRAME    -> Crea una ventana sin borde.
```

Las constantes anteriores puedes encontrarlas directamente en la cabecera correspondiente, no obstante, yo he cogido las descripciones utilizadas en este documento [3].

Existen otras 6 constantes, pero son de uso interno y no las podras establecer con la funcion anterior. Puedes estudiar mas en <SDL/SDL_video.h>.

Por ultimo, si quieres definir el titulo de la ventana de tu aplicacion, puedes hacerlo mediante:

```
SDL_WM_SetCaption("Hack The World", NULL);
```

El segundo parametro es el nombre de un icono opcional, puedes obtener estos valores con la analoga 'SDL_WM_GetCaption()' y establecer por separado un icono con 'SDL_WM_SetIcon(SDL_LoadBMP("/home/usuario/icono.bmp"), NULL)'.

Puedes minimizar la pantalla durante la ejecucion del programa mediante la funcion 'SDL_WM_IconifyWindow()'.

Si quieres ver como funciona tu primera ventana, puedes utilizar un codigo general de compilacion como este:

```
$ gcc prog.c -lSDL -o prog
```

o

```
$ gcc proc.c `sdl-config --cflags` `sdl-config --libs` -o prog
```

Vale, hasta aqui tenemos lo basico para iniciar el sistema y comenzar a cargar graficos (imagenes).

---[3.1 - Eventos

Esta es, sin duda alguna, la parte mas importante de una aplicacion creada con SDL. Se inicia siempre junto al sistema de video y es en resumen la parte del sistema que nos permite interactuar con la aplicacion, ya sea mediante el teclado, el raton, un joystick u otros relaciones mas directamente con el sistema operativo.

Como aperitivo mostrare el bucle principal (main loop) que yo suelo utilizar para controlar los eventos en mis aplicaciones. Luego dare una peque~a aclaracion sobre su estructura y funciones, dando paso, ya por ultimo, a las siguientes secciones que explicaran particularmente el caso de cada dispositivo de entrada.

```
**-------------------------------------------------------------------------**

   SDL_Event evento;
   int terminar = 0;

   while (!terminar) {

      SDL_WaitEvent(&evento);

      switch (evento.type) {
         case SDL_QUIT:
            terminar = 1;
            break;
         case SDL_MOUSEBUTTONDOWN:
            sonidos(1);
            /* printf("\nX=%d - Y=%d\n", evento.button.x, evento.button.y); */
            terminar = accion(evento.button.x, evento.button.y);
            break;
         case SDL_MOUSEMOTION:
            motion_cursor(evento.motion.x, evento.motion.y);
            break;
         case SDL_KEYDOWN:
            sonidos(2);
            mkeys(evento);
            break;
         default:
            break;
      }
   }

**-------------------------------------------------------------------------**
```

Empecemos por el principio. Lo mas basico es el bucle while que estara siempre iterando hasta que la variable 'terminar' tenga un valor positivo.

Seguidamente viene la funcion mas importante. No me queda mas remedio que explicar en este momento las funciones de captura de eventos que SDL nos facilita. Son 3:

- SDL_WaitEvent() -> Esta funcion espera por la llegada de un evento, ya sea

un click, una pulsacion de teclado, la peticion de cierre de la ventana u otro cualquiera. Se puede decir que esta funcion es 'bloqueante' pues nada ocurrira mientras un evento no se produzca y demos una respuesta al mismo.

- SDL_PollEvent() -> Muy parecida a la anterior, pero esta es mas usual en los videojuegos. En estos entornos, se precisa que la aplicacion siga realizando tareas en segundo plano aun a pesar de que no existan eventos a los que responder. Esta llamada no es bloqueante y ese es el motivo.

- SDL_PumpEvents() -> Esta funcion nos permitira acceder directamente al estado de un dispositivo para conocer si algun evento se ha producido en ese mismo instante. Su uso es menos habitual y puede que solo la veas ocasionalmente.

Lo que vemos a continuacion es una sentencia 'switch' que maneja el valor almacenado en el campo 'type' de la estructura 'SDL_Event'. De este modo sabremos exactamente que evento se ha producido. Puedes entonces introducir tantas sentencias 'case' como constantes hay definidas en la enumeracion 'SDL_EventType' declarada en el archivo de cabecera <SDL/SDL_events.h>

En este caso hacemos lo siguiente:

1 - Para 'SDL_QUIT', que resulta de hacer click en la [X] de la ventana, activamos la variable 'terminar' que provoca la salida inmediata del bucle 'while'.

2 - Para 'SDL_MOUSEBUTTONDOWN', que resulta de hacer click en cualquier region de la ventana, reproducimos un sonido (se vera en una seccion posterior), y ejecutamos una accion segun donde se haya clickado (se vera en la seccion 3.1.1).

3 - Para 'SDL_MOUSEMOTION', que resulta de haber movido el raton dentro de la ventana, nos comportamos practicamente como en el evento anterior pero sin reproducir sonido alguno.

4 - Para 'SDL_KEYDOWN', que resulta de presionar cualquier tecla, reproducimos un sonido y ejecutamos una funcion que se encarga de controlar que pulsacion hemos realizado exactamente y realizar un cometido segun corresponda.

Habras observado una sentencia 'printf()' convenientemente comentada. Pues bien; esta instruccion te sera practicamente imprescindible en la etapa de desarrollo de tus programas. Imprime las coordenadas donde has hecho click, y esto sera mas que necesario cuando quieras conocer la posicion de un objeto situado en pantalla y definir "regiones de clickado" de un modo veloz.

Pasemos ahora a describir las acciones de comportamiento que utilizaremos para cada dispositivo.

---[3.1.1 - Raton

Como acabas de ver hace tan solo unos instantes, cuando un evento 'click' se produce, las coordenadas de la pulsacion son almacenadas en el elemento 'button' de la union 'SDL_Event'.

He dicho que 'button' es un elemento y no una variable porque en realidad es otra estructura. Lo que es mas, menos uno, todos los elementos de 'SDL_Event' son estructuras que controlan todas las propiedades de cada evento.

Pero no nos vayamos del tema. Si sigues echando un vistazo a la estructura 'SDL_MouseButtonEvent', podras ver otros elementos, como cual de los dos botones del raton se ha pulsado, o si el boton pulsado esta bajando o subiendo, aparte de otras mas.

Una funcion tipica que ejecute acciones segun las coordenadas donde hayas pulsado suele tener la siguiente estructura:

```
 **-------------------------------------------------------------------------**

   int accion(Uint16 X, Uint16 Y)
   {
      /* BOTON 1 */
      if ((X >= 915 && Y >= 685) && (X <= 1024 && Y <= 718)) {
         funcion01(arg1, arg2, ...);
      }
      /* BOTON 2 */
      else if ((X >= 915 && Y >= 648) && (X <= 1024 && Y <= 682)) {
         funcion02(arg1, arg2, ...);
      }
      /* BOTON 3 */
      else if ((X >= 915 && Y >= 614) && (X <= 1024 && Y <= 646)) {
         funcion03(arg1, arg2, ...);
      }
      /* BOTON SALIR */
      else if ((X >= 915 && Y >= 578) && (X <= 1024 && Y <= 610)) {
         return 1;
      }
      return 0;
   }

 **-------------------------------------------------------------------------**
```

La funcion puede llegar a complicarse tanto como desees, pero al fin y al cabo siempre acaba teniendo una estructura similar.

Puedes pensar ahora para que necesitas una funcion que controle el movimiento del raton sobre la ventana. Pues es bastante facil, imaginate que tienes unas imagenes representando botones (se vera mas adelante), podrias desear que los botones se iluminen cada vez que pasas por encima de ellos.

Pero OJO: Debes prestar especial atencion a la hora de crear esta funcion.
Puede parecer invisible, pero esta funcion se ejecutara cada vez que el cursor del raton se mueva un solo pixel en cualquier direccion. Esto quiere decir que la funcion puede ejecutarse cientos de veces por segundo. Si los condicionales no estan bien definidos y tiene que recorrer mas de los que deberia, el rendimiento podria verse seriamente afectado.

No tendras problema alguno en desarrollar la tuya propia, pero quizas veamos algo mas adelante cuando tratemos con la representacion de botones o menus.

---[3.1.2 - Teclado

Para controlar las pulsaciones del teclado accederemos a un miembro de 'SDL_Event' llamado 'key' que es una estructura 'button' y que a su vez contiene otra estructura mas con el nombre 'SDL_keysym' cuyo elemento mas importante es el miembro 'sym' que, a pesar de que os entren unas ganas enormes de asesinarme, es una ultima estructura que define la totalidad de las constantes referentes a las posibles teclas pulsadas.

Segun la tecla pulsada tu podras hacer lo que creas conveniente. Yo mostrare

aqui un ejemplo que llama a una funcion 'escribir()' para cada tecla pulsada
cuyo objetivo es escribir el caracter correspondiente en la representacion
de una consola (o shell) que estudiaras en un capitulo posterior.

Recortare el codigo:

```
 **-------------------------------------------------------------------------**

   void mkeys(SDL_Event ev)
   {
      int shift = 0;
      int altgr = 0;

      if (ev.key.keysym.mod & (KMOD_LSHIFT|KMOD_RSHIFT))
         shift = 1;
      else if (ev.key.keysym.mod & (KMOD_LALT|KMOD_RALT))
         altgr = 1;

      switch (ev.key.keysym.sym){
         case SDLK_ESCAPE:
            terminar = 1;
            break;
         case SDLK_BACKSPACE:
            borrar();
            break;
         case SDLK_RETURN:
            ejecutar();
            break;
         case SDLK_TAB:
            escribir('\t', 1);
            break;
         case SDLK_SPACE:
            escribir(' ', 1);
            break;
         case SDLK_COMMA:
            if (shift)
               escribir(':', 1);
            else
               escribir('.', 1);
            break;
         case SDLK_a:
            if (shift)
               escribir('A', 1);
            else
               escribir('a', 1);
            break;
         case SDLK_b:
            if (shift)
               escribir('B', 1);
            else
               escribir('b', 1);
            break;
         ...
         ...
         ...
         case SDLK_1:
            if (shift)
               escribir('!', 1);
            else if (altgr)
               escribir('|', 1);
            else
               escribir('1', 1);
            break;
         case SDLK_2:
```

```
            if (shift)
               escribir('"', 1);
            else if (altgr)
               escribir('@', 1);
            else
               escribir('2', 1);
            break;
         ...
         ...
         ...
         default:
            break;
      }
   }
```

```
 **-------------------------------------------------------------------------**
```

Como ya he dicho, lo importante es comprender el procedimiento que seguimos para llevar a cabo nuestros objetivos. Posteriormente echaremos un vistazo a las funciones que en este caso he utilizado.

Es muy bueno observar el uso que hemos hecho del elemento 'mod' de la estructura 'SDL_keysym'. Siempre comprobamos si cada vez que pulsamos una tecla lo hacemos simultaneamente con SHIFT o ALT(GR).

---[3.2 - Imagenes

Lo principal es que conozcamos que es una SUPERFICIE. En realidad es una estructura que controla todos los pixeles de una imagen o region de la pantalla asi como su profundidad de color y otro tipo de propiedades. Nos sobrara con saber que se definen siempre igual que la superficie principal:

```
SDL_Surface *imagen;
```

Otra estructura basica que debemos estudiar es 'SDL_Rect', sirve para definir una region en la pantalla indicando sus coordenadas iniciales y el ancho/alto de la misma. Se acostumbra a utilizar asi:

```
SDL_Rect rect = (SDL_Rect) {300, 300, 100, 100};
```

Esto define una region que comienza en las posiciones x=300, y=300, teniendo una medida de 100 pixeles tanto para el ancho como para el alto.
Si los dos ultimos valores son iguales a 0, cuando dibujemos la superficie se igualaran automaticamente al ancho y alto de la misma.
Podemos tambien acceder a sus elementos individualmente de esta forma:

```
rect.x = 300;
rect.y = 300;
rect.w = 100;
rect.h = 100;
```

SDL nos proporciona una funcion basica para la carga de imagenes, que es conocida como:

```
imagen = SDL_LoadBMP("/home/usuario/file.bmp");
```

Devuelve siempre el valor NULL cuando no ha podido cargar el archivo. Pero esta funcion es pobre y no soporta otros formatos. Es aqui donde entran en juego las librerias auxiliares. En este caso SDL_image.

Antes de pasar a estudiar el uso de esta libreria debemos conocer algunas

funciones mas para el manejo basico de las superficies.

Por ejemplo, ahora que ya tenemos cargada una superficie en la variable 'imagen', la pregunta es: como dibujarla en pantalla?

'SDL_BlitSurface()' al rescate. Mostraremos como se usa y luego explicaremos sus argumentos:

```
SDL_BlitSurface(imagen, NULL, pantalla, &rect);
```

1 - La superficie que queremos dibujar.
2 - La porcion de la superficie que queremos dibujar (SDL_Rect). Un valor NULL dibuja la superficie completa.
3 - La superficie sobre la que dibujaremos.
4 - Las coordenadas donde imprimiremos la superficie a pintar (SDL_Rect).

No siempre desearemos dibujar nuestras superficies o imagenes directamente sobre la pantalla principal; pero quedas advertido, si realizas un 'blit' sobre cualquier otra superficie, luego estaras obligado a realizar el 'blit' de esta ultima sobre la pantalla principal.

Muy bien, si has llegado hasta aqui y has intentado dibujar una imagen propia sobre la ventana principal, te preguntaras porque esta no es visible y parece que la ejecucion de tu aplicacion ha fallado. No te precipites, aqui tienes la solucion:

```
SDL_Flip(pantalla);
```

Esta funcion provoca que la representacion grafica que has creado en memoria hasta el momento, sea volcada en pantalla. Podrias decir vulgarmente que estas "actualizando la pantalla" si te sientes mas comodo.

---[3.2.1 - SDL_image

La funcion mas importante que proporciona esta libreria es:

- SDL_Surface * IMG_Load(const char *file)

Puedes usarla tal y como lo has hecho con 'SDL_LoadBMP()'. Pero entonces, cual es la diferencia?

Una y muy grande. Soporta los siguiente formatos:

- BMP, PNM, XPM, LBM, PCX, GIF, JPG, PNG y TGA.

Puedes incluir esta libreria muy facilmente a~adiendo "-lSDL_image" a tus opciones de compilacion normales.

No hay mas que decir, creo que esta mas que claro que esta es la funcion que empezaras a utilizar a partir de ahora.

---[3.2.2 - Ventanas

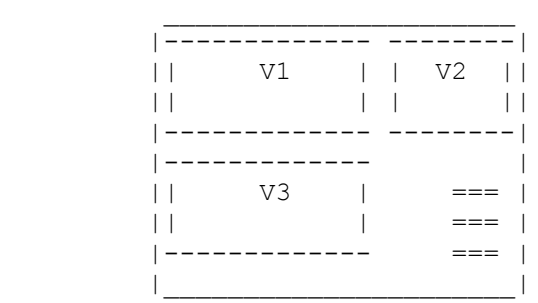
La cruda realidad es que SDL no comprende entidades como ventanas o botones. No tiene estructuras para manejar este tipo de objetos y eso supone un paso hacia atras para nosotros.

En SDL todo es apariencia, y como tal, nuestro objetivo es utilizar una imagen para darle la apariencia de una ventana y cierto comportamiento que se asemeje lo suficiente.

Lo normal es utilizar siempre una imagen en formato "PNG", dado que soporta las transparencias y ese es un aspecto muy util a la hora de crear una interfaz atractiva y eficiente. Las imagenes en este formato tiene una alta compresion y ocupan relativamente poco espacio. Un editor como "GIMP" puede ayudarte mucho en tareas como esta.

Encontrar una imagen con el aspecto de una ventana es tan facil como navegar durante unos minutos por la web o simplemente sacar una captura de pantalla de una ventana de tu PC y editarla posteriormente para borrar todo su contenido, dejando unicamente el marco de la misma.

Como aqui no vamos a ver la posibilidad de arrastrar una ventana por la pantalla, que podria tornarse en una tarea infernal (pues requiere el repintado continuo de todas las superficies por las que esta pasa), definiremos unas regiones en las que interactuaran nuestras ventanas individualmente. Lo logico seria situar las ventanas en cada una de las esquinas de la pantalla. Algo asi:



A mi me gusta utilizar dos peque~as funciones para cargar y mostrar en pantalla una ventana. Las expongo aqui y las explico a continuacion:

```
**-------------------------------------------------------------------------**

  SDL_Surface *v1;

  void carga_v1(int put)
  {
     v1 = IMG_Load(IMAGES"v1.png");
     if (!v1) {
        fprintf(stderr, "No se pudo cargar v1.png\n");
        SDL_Quit();
        exit(-1);
     }

     if (put)
        put_v1();
  }

  void put_v1(void)
  {
     SDL_Rect r1;

     r1 = (SDL_Rect) {0, 0, 705, 375};
     SDL_FillRect(pantalla, &r1, SDL_MapRGB(pantalla->format,0,0,0));
     r1 = (SDL_Rect) {0, 0, 0, 0};
     SDL_BlitSurface(v1, NULL, pantalla, &r1);
     SDL_Flip(pantalla);
  }

**-------------------------------------------------------------------------**
```

Te preguntaras por que no poner las dos funciones en una. El motivo es el siguiente. La primera vez que entras en el programa llamaras a la funcion

'carga_v1()' con un argumento positivo; esto cargara la imagen (ventana) y la mostrara en pantalla. Durante el resto de la ejecucucion del programa, cuando realices un 'blit' sobre esta ventana, llamaras simplemente a 'put_v1()', que borrara la region y repintara la ventana nuevamente con los cambios realizados. Solo llamaremos a 'carga_v1()' cuando querramos obtener una ventana limpia, sin modificaciones.

Bien, imaginate ahora que ya tenemos la imagen de la ventana cargada en la esquina superior izquierda de la ventana. Imaginate tambien que esa imagen tiene dibujados dos cuadrados o circunferencias (o lo que sea) representando los botones de cierre, minimizado, etc...

Tal como se ha explicado en una seccion anterior, podemos definir una "region de clickado" que realice una operacion al clickar en esta zona. Podriamos a~adir a la funcion 'accion()' algo como:

```
   /* BOTON "CERRAR" DE V1 */
   if ((X >= 642 && Y >= 6) && (X <= 662 && Y <= 26)) {
      if (is_v1)
         quitar_v1();
   }
```

Lo mas facil es que cuando hagamos click en este "boton", la ventana deje de ser visible ya sea dibujando un rectangulo negro encima, haciendo que vaya desapareciendo desplazandose hacia la izquierda, hacia arriba o incluso en diagonal. Nosotros, que somos asi de atrevidos vamos a optar por esta ultima eleccion:

```
 **-------------------------------------------------------------------------**

   void quitar_v1(void)
   {
      SDL_Rect pos;
      int x=0, y=0;

      while(x < 400 && y < 400){
         pos = (SDL_Rect) {0, 0, 705, 375};
         SDL_FillRect(pantalla, &pos, SDL_MapRGB(pantalla->format,0,0,0));
         pos = (SDL_Rect) {0-x, 0-y, v1->w, v1->h};
         SDL_BlitSurface(v1, NULL, pantalla, &pos);
         SDL_Flip(pantalla);
         x += 20;
         y += 20;
         SDL_Delay(20);
      }
      is_v1 = 0;
   }

 **-------------------------------------------------------------------------**
```

No te ciegues, la operacion no es nada complicada. Simplemente vamos restando 20 a las coordenadas iniciales (0,0) y redibujamos la ventana en cada vuelta. Entremedias dibujamos siempre un rectangulo negro que cubra toda la zona. Es esta secuencia de "quitar y poner" la que nos ofrece una sensacion real de movimiento. La ventana ira desapareciendo diagonalmente hasta perderse fuera de la region visible de la pantalla.

'SDL_Delay()' hace una funcion similar a la 'usleep()' que es detener el programa la cantidad de milisegundos especificados como unico argumento. Puedes incrementar esta cifra si deseas un movimiento mas lento o bajarla si deseas el efecto contrario.

Como puedes ver, utilizamos una ultima variable global 'is_v1' que usamos como si de un dato booleano se tratase y que nos permitira saber en que

estado se encuentra la ventana (abierta/cerrada o visible/oculta).

Deberias tener una lista de botones en la parte inferior de la ventana, o en aquel sitio que tu elijas, que te permita traer de nuevo las ventanas que has cerrado (como crear botones simulados sera el tema del siguiente apartado).

La funcion de regreso de una ventana puede parecerse a esto:

```
 **--------------------------------------------------------------------------**

   void traer_v1(void)
   {
      SDL_Rect pos;
      int x=0, y=0;

      while(x < 700 && y < 700){
         pos = (SDL_Rect) {0, 0, 705, 375};
         SDL_FillRect(pantalla, &pos, SDL_MapRGB(pantalla->format,0,0,0));
         if (x > -50 && y > 365)
            pos = (SDL_Rect) {0, 0, v1->w, v1->h};
         else
            pos = (SDL_Rect) {-710+x, -768+x, v1->w, v1->h};
         SDL_BlitSurface(v1, NULL, pantalla, &pos);
         SDL_Flip(pantalla);
         x += 20;
         y += 20;
         SDL_Delay(20);
      }
      is_v1 = 1;
   }

 **--------------------------------------------------------------------------**
```

La operacion es muy parecida. Recuerda tambien que las cifras van a depender siempre de tu caso en particular y que deberas ajustarlas en su momento.

La unica diferencia en esta ocasion es que es muy improbable que la ultima iteracion del bucle termine colocando la ventana en sus coordenadas exactas. Es por ello que controlamos cuando la ventana se acerca a esta posicion y acabamos por colocarla nosotros mismos en el lugar correspondiente.

---[3.2.3 - Botones

Para practicar con la representacion de botones seguiremos el penoso ASCII ART que utilizamos en el apartado anterior, es decir, que situaremos un boton en la esquina inferior derecha que abrira un menu cuando hagamos click en el.

Tambien veremos como crear un efecto de iluminado cuando pasemos por encima de cada uno de los botones. Esto les dara una sensacion de volumen y aumentara notablemente su credibilidad.

Los botones deben ser sin duda, junto con las fotografias xxx, las imagenes mas abundantes en la telara~a de la World Wide Web. Incluso si te molestas en buscar un poco en Google, encontraras unos peque~os programas que te permitiran crear, on-line, tus propios botones personalizados. Recuerda utilizar formato "PNG" preferentemente, si utilizas botones con esquinas redondas este requisito se vuelve cuasi imprescindible.

La primera funcion que debemos definir, se encarga de cargar las imagenes en las superficies correspondientes. Segun el argumento, se cargaran los botones normales o los que hayamos creado con efecto iluminado. Piensa que podrias

utilizar otro 'case' para cargar botones, por ejemplo, con efecto presionado.

```
 **------------------------------------------------------------------------**

   void carga_botones(int op)
   {
      switch (op) {
         case 0: {
            boton0 = IMG_Load(IMAGES"boton0.png");
            boton1 = IMG_Load(IMAGES"boton1.png");
            boton2 = IMG_Load(IMAGES"boton2.png");
            boton3 = IMG_Load(IMAGES"boton3.png");
            break;
         }
         case 1: {
            boton0 = IMG_Load(IMAGES"boton0_ilu.png");
            boton1 = IMG_Load(IMAGES"boton1_ilu.png");
            boton2 = IMG_Load(IMAGES"boton2_ilu.png");
            boton3 = IMG_Load(IMAGES"boton3_ilu.png");
            break;
         }
      }

      if (!boton0 || !boton1 || !boton2 || !boton3)
      {
         fprintf(stderr, "Don't load some button image\n");
         SDL_Quit();
         exit(-1);
      }
   }

 **------------------------------------------------------------------------**
```

Comprobamos al final que todas las imagenes han podido ser cargadas correctamente, evitamos asi caidas inesperadas.

La siguiente funcion se encarga de poner los botones que seran estaticos, es decir, los que no forman parte del menu interno y seran visibles durante toda la ejecucion. En nuestro caso solo pondremos uno en la esquina inferior derecha, pero podriamos poner tres en horizontal y hacer que cada uno abriera un menu diferente siguiendo los pasos que aqui explicaremos.

```
 **------------------------------------------------------------------------**

   void poner_botones(void)
   {
      SDL_Rect pos;
      int x=0, y=0;

      carga_botones(0);

      pos = (SDL_Rect) {910, 710, 0, 0};
      SDL_BlitSurface(boton0, NULL, pantalla, &pos);
      SDL_Flip(pantalla);
   }

 **------------------------------------------------------------------------**
```

Puedes ver como llamamos antes de nada a 'carga_botones()' con un valor de 0. Mientras no interactuemos con ellos, deben de estar en estado normal.

Para lograr recrear la apertura de un menu, primero tenemos que definir una "region de clikado" en torno a 'boton0'. Como dije antes, la instruccion 'printf()' que imprime las coordenadas puede servirte de mucho. Haz click

cerca de la esquina superior izquierda de 'boton0' anota la posicion en un papel y repite el proceso para su esquina inferior derecha. Hecho esto, vete a la funcion 'accion()' y a~ade algo como:

```
 **--------------------------------------------------------------------------**

   /* BOTON 0 */
   else if ((X >= 920 && Y >= 720) && (X <= 1024 && Y <= 768)) {
      if (is_menu)
         cerrar_menu();
      else
         abrir_menu();
   }


 **--------------------------------------------------------------------------**
```

Bien hecho, y ahora vamos directamente a definir las dos funciones tan esperadas.

```
 **--------------------------------------------------------------------------**

   void abrir_menu(void)
   {
      SDL_Rect rmenu;
      int x = 0;

      carga_botones(0);

      while(x < 250){

         if((1030 - x) >= 910){
            rmenu = (SDL_Rect) {745, 675, 300, 40};
            SDL_FillRect(pantalla, &rmenu, SDL_MapRGB(pantalla->format,0,0,0));
            rmenu = (SDL_Rect) {1030-x, 675, 0, 0};
            SDL_BlitSurface(boton1, NULL, pantalla, &rmenu);
         }
         if((1070 - x) >= 910){
            rmenu = (SDL_Rect) {745, 640, 300, 40};
            SDL_FillRect(pantalla, &rmenu, SDL_MapRGB(pantalla->format,0,0,0));
            rmenu = (SDL_Rect) {1070-x, 640, 0, 0};
            SDL_BlitSurface(boton2, NULL, pantalla, &rmenu);
         }
         if((1110 - x) >= 910){
            rmenu = (SDL_Rect) {745, 605, 300, 40};
            SDL_FillRect(pantalla, &rmenu, SDL_MapRGB(pantalla->format,0,0,0));
            rmenu = (SDL_Rect) {1110-x, 605, 0, 0};
            SDL_BlitSurface(boton3, NULL, pantalla, &rmenu);
         }

         SDL_Flip(pantalla);
         x += 10;
         SDL_Delay(20);
      }
      is_menu = 1;
   }

 **--------------------------------------------------------------------------**
```

El efecto que crea la funcion que acabamos de ver es realmente bonito. Los botones (1, 2 y 3) van apareciendo escalonados por la parte derecha de la ventana y se detienen finalmente para quedar situados en una vertical perfecta.

En realidad lo que ocurre es que empezamos dibujando estos botones fuera de la zona visible de la pantalla (lo hacemos ya de forma escalonada). Despues vamos incrementando 'x', cifra que se resta a las coordenadas inciales de cada boton. De esta forma los botones se van dibujando mas hacia la izquierda en cada pasada del bucle. 'SDL_Delay()' controla la velocidad y cada boton se detiene cuando su coordenada horizontal(rmenu.x) llega a 910.

No nos olvidamos de dibujar un rectangulo negro que borra cada boton de forma individual en cada iteracion del bucle. Con esto evitamos el rastro que irian dejando los botones en cada operacion de repintado.

Activamos por ultimo una variable global llamada 'is_menu' que indicara a la funcion 'accion()' que el menu ya ha sido abierto y el proximo click en 'boton0' conllevara el cierre del mismo.

Aqui la funcion analoga:

```
 **------------------------------------------------------------------------**

   void cerrar_menu(void)
   {
      SDL_Rect rmenu;
      int x = 0;

      carga_botones(0);

      while(x < 150){
         rmenu = (SDL_Rect) {745, 570, 300, 150};
         SDL_FillRect(pantalla, &rmenu, SDL_MapRGB(pantalla->format,0,0,0));

         rmenu = (SDL_Rect) {940+x, 675, 0, 0};
         SDL_BlitSurface(boton1, NULL, pantalla, &rmenu);
         rmenu = (SDL_Rect) {920+x, 640, 0, 0};
         SDL_BlitSurface(boton2, NULL, pantalla, &rmenu);
         rmenu = (SDL_Rect) {900+x, 605, 0, 0};
         SDL_BlitSurface(boton3, NULL, pantalla, &rmenu);

         SDL_Flip(pantalla);
         x += 10;
         SDL_Delay(20);
      }
      is_menu = 0;
   }

 **------------------------------------------------------------------------**
```

La diferencia principal es que aqui la variable 'x' se suma a las coordenadas en cada pasada para ir desplazando los botones hacia la derecha hasta que los mismos desaparezcan de la pantalla.

Desactivamos is_menu para dejar las cosas limpias.

Muy bien, muy bien. Ya queda poco. Lo prometido es deuda, ahora veremos como crear el efecto de iluminado de los botones. Veamos el primero de los metodos para comentarlo posteriormente.

```
 **------------------------------------------------------------------------**

   void motion_cursor(Uint16 X, Uint16 Y)
   {
      if (is_menu) {
         if ((X >= 915 && Y >= 685) && (X <= 1024 && Y <= 718))
            cambiar_estado(1, 1);
```

```
         else if ((X >= 915 && Y >= 648) && (X <= 1024 && Y <= 682))
            cambiar_estado(2, 1);
         else if ((X >= 915 && Y >= 614) && (X <= 1024 && Y <= 646))
            cambiar_estado(3, 1);
         else if (st_ilu) {
            cambiar_estado(0, 0);
            cambiar_estado(1, 0);
            cambiar_estado(2, 0);
            cambiar_estado(3, 0);
         }
      }
      else if ((X >= 920 && Y >= 720) && (X <= 1024 && Y <= 768))
         cambiar_estado(0, 1);
      else if (st_ilu) {
         cambiar_estado(0, 0);
      }
   }
```

 --

Esta es la esperada funcion que reacciona ante el movimiento del raton.

Pueden ocurrir varias cosas:

   (1) Si el menu esta abierto:

       (1.1) Si el puntero del raton se situa encima de alguno de los tres
             botones (1, 2 o 3), llama a la funcion cambiar_estado() que carga
             las imagenes iluminadas y redibuja el boton sobre el que esta el
             puntero.

       (1.2) En caso contrario, comprobamos si hay algun boton iluminado y los
             pasamos todos a modo normal.

   (2) Si el menu esta cerrado:

       (2.1) Si estamos encima de 'boton0', lo iluminamos (ya dije como).

       (2.2) En caso contrario lo apagamos :)

Y ya por fin la funcion que cambia el estado de un boton especifico:

 --

```
   void cambiar_estado(int b, int s)
   {
      SDL_Rect pos;

      switch (s) {
         case 0:{
            carga_botones(0);
            st_nor = 1;
            st_ilu = 0;
            break;
         }
         case 1:{
            carga_botones(1);
            st_ilu = 1;
            st_nor = 0;
            break;
         }
      }

      switch (b) {
```

```
         case 0:{
            pos = (SDL_Rect) {910, 710, 0, 0};
            SDL_BlitSurface(boton0, NULL, pantalla, &pos);
            break;
         }
         case 1:{
            pos = (SDL_Rect) {910, 675, 0, 0};
            SDL_BlitSurface(boton1, NULL, pantalla, &pos);
            break;
         }
         case 2:{
            pos = (SDL_Rect) {910, 640, 0, 0};
            SDL_BlitSurface(boton2, NULL, pantalla, &pos);
            break;
         }
         case 3:{
            pos = (SDL_Rect) {910, 605, 0, 0};
            SDL_BlitSurface(boton3, NULL, pantalla, &pos);
            break;
         }
      }
      SDL_Flip(pantalla);
   }
```

```
 **--------------------------------------------------------------------------**
```

Facil. El primer argumento indica a que boton se le quiere cambiar su estado. En el segundo que estado activaremos. Una sentencia 'switch()' para cada uno y todo arreglado.

---[3.3 - Audio

Realizar tareas con el subsistema de audio que proporciona directamente la libreria SDL es posible, pero puede convertirse en toda una haza~a digna de comentar.

Es por ello que la dejaremos a un lado centrandonos en la implementacion de la siguiente libreria auxiliar: SDL_mixer.

---[3.3.1 - SDL_mixer

Compilar un programa que utilice esta libreria es tan facil como a~adir el archivo de cabecera <SDL/SDL_mixer.h> y ayudarse de la opcion "-lSDL_mixer" a la hora de utilizar tu version de 'gcc' favorita.

Lo primero es lo primero, que es iniciar el sistema de sonido. Suelo utilizar algo como lo que veras a continuacion:

```
   if (Mix_OpenAudio(44100, AUDIO_S16, 2, 4096)) {
      fprintf(stderr, "No se puede iniciar SDL_mixer %s\n", Mix_GetError());
      SDL_Quit();
      exit(1);
   }
   atexit(Mix_CloseAudio);
```

Los argumentos son estos:

   1 - Frecuencia (en Hertzios) para reproducir un 'sample' (*).

   (*) Yo utilizo la calidad CD, otros posibles son:

```
          11025 -> Calidad telefono.
          22050 -> Calidad radio.

   2 - Formato del sample (revisa las constantes en <SDL_mixer.h>).
   3 - Numero de canales (1 = mono, 2 = estereo)
   4 - 'Chunksize', esto habitualmente siempre es 4096.
```

Utiliza Mix_CloseAudio() para detener el sistema.

Quizas te estes preguntando que significa eso de un "Chunk". Para que lo entiendas, podriamos decir escuetamente que es la estructura donde SDL_mixer almacena un sonido para poder trabajar con el.

Ahora expondre una funcion que me gusta utilizar para la reproduccion de sonidos y luego la explicare en detalle para que se entienda:

```
 **-------------------------------------------------------------------------**

   void sonidos(int op){

      u_int canal;
      Mix_Chunk *sonido;

      switch(op){
         case 1:{
            sonido = Mix_LoadWAV("/home/usuario/sonidos/sample1.wav");
            if(sonido == NULL){
               printf("No pude cargar sonido: %s\n", Mix_GetError());
               return;
            }
            canal = Mix_PlayChannel(-1, sonido, 0);
            break;
         }
         case 2:{
            sonido = Mix_LoadWAV("/home/usuario/sonidos/sample2.wav");
            if(sonido == NULL){
               printf("No pude cargar sonido: %s\n", Mix_GetError());
               return;
            }
            canal = Mix_PlayChannel(-1, sonido, 0);
            break;
         }
         ...
         ...
         ...
         default:
            break;
      }

      Mix_FreeChunk(sonido);
   }

 **-------------------------------------------------------------------------**
```

'MixLoadWAV()' es analoga a la funcion de carga de imagenes, carga el fichero indicado en su inco parametro y lo alamace en el chunk que hemos creado.

Comprobamos siempre que la funcion ha tenido exito y luego llamamos sin mas dilaciones a 'Mix_PlayChannel()' cuyos argumentos son los siguientes:

```
   1 - Numero de canal para reproducir el sonido (*).

       (*) Un valor de '-1' para seleccion automatica del canal.
```

2 - Sonido (chunk) a reproducir.

3 - Numero de veces que se repetira el sonido (*).

(*) Un valor de '-1' para reproducir indefinidamente.

Liberamos finalmente el chunk para mejorar el rendimiento.

Aqui tienes mas funciones, sacadas del archivo de cabecera de esta libreria. Dare una ligera explicacion de cada una:

- Mix_PlayChannelTimed() -> Igual que la anterior pero lleva un ultimo argumento que indica cuantos milisegundos se reproducira.

- Mix_FadeInChannel() y Mix_FadeInChannelTimed() ->

-> Analogas a las 2 anteriores pero va subiendo el volumen del sonido de modo gradual.

- Mix_Pause(channel), Mix_Resume(channel), Mix_HaltChannel(channel) ->

-> Mas que obvios: Pausar, reanudar y detener. 'channel' es un int.

- Mix_Playing(int channel) y Mix_Paused(int channel) ->

-> Funciones de consulta, con ellas obtienes el estado actual de un canal.

Y exiten muchisimas mas. Si te interesa crear un reproductor de sonidos o de musica queda a tu eleccion seguir estudiando este tema.

Y hablando de "musica", decir que SDL_mixer diferencia realmente entre la estructura de "un sonido" y de "una musica". Tanto que reserva un canal especial para la reproduccion de esta ultima.

Los formatos que reconoce son:

- WAV, MP3, MOD, S3M, IT, XM, Ogg Vorbis, VOC y MIDI

Todas las funciones para manejar una estructura del tipo 'Mix_Music *' son practicamente analogas a las que controlan "chunks". No te sera dificil investigar un poco por tu cuenta.

---[3.3.2 - CD-ROM

Para ejecutar las funciones que se ense~aran a continuacion debes incluir en tus programas la cabecera <SDL/SDL_cdrom.h>.

Existen dos estructuras para el control de un dispositivo CD-ROM.

'SDL_CD' controla el estado del dispositivo, el numero de pistas, la pista actual, el frame actual y un array de estructuras del tipo 'SDL_CDtrack'. Tambien tiene un identificador unico para cada uno de los dispositivos.

Como no me voy a poner a mostrar ejemplos sobre el uso de estos metodos. Indicare como siempre las funciones tal y como se pueden ver en la cabecera principal y un escueto comentario.

- int SDL_CDNumDrives(void) -> Devuelve el numero de dispositivos CD-ROM.

- const char * SDL_CDName(int drive) -> Devuelve el nombre de un dispositivo.

- SDL_CD * SDL_CDOpen(int drive) -> Abre un dispositivo de CD-ROM.

- CDstatus SDL_CDStatus(SDL_CD *cdrom) -> Devuelve el estado de un CD-ROM.

- int SDL_CDPlayTracks(SDL_CD *cdrom, int start_track, int start_frame,
  int ntracks, int nframes) ->

  -> Reproduce tantas pistas como se le indique
  en 'ntracks' empezando en 'star_track'.

- int SDL_CDPlay(SDL_CD *cdrom, int start, int length) ->

  -> Reproduce un CD-ROM desde el frame 'start'
  hasta la cantidad indicada por 'lenght'.

- int SDL_CDPause(SDL_CD *cdrom) -> Pausa la ejecucion de una pista.

- int SDL_CDResume(SDL_CD *cdrom) -> Continua la ejecucion de una pista.

- int SDL_CDStop(SDL_CD *cdrom) -> Detiene la ejecucion de una pista.

- int SDL_CDEject(SDL_CD *cdrom) -> Abre la bandeja del CD-ROM.

- SDL_CDClose(SDL_CD *cdrom) -> Cierra el dispositivo CD-ROM.

---[3.4 - Texto

Para hacer uso del texto en nuestras aplicaciones SDL utilizaremos otra de las librerias auxiliares. Se llama 'SDL_ttf' y la estudiaremos inmediatamente.

Deberas utilizar en tu programa el archivo de cabecera <SDL/SDL_ttf.h> y a~adir la opcion de compilacion "-lSDL_ttf".

Para iniciar el sistema de texto solemos utilizar algo asi:

```
if (TTF_Init() < 0) {
   fprintf(stderr, "No se pudo iniciar SDL_ttf: %s\n", SDL_GetError());
   SDL_Quit();
   exit(1);
}
atexit(TTF_Quit);
```

Con 'TTF_Quit()' logramos el resultado contrario.

Para poder escribir algo en pantalla, primero debemos abrir una fuente que tengamos guardada en un directorio e indicar que tama~o de letra deseamos utilizar. La funcion que viene a continuacion empaqueta estas acciones:

```
TTF_Font *fuente;

if ((fuente = TTF_OpenFont("FreeMono.ttf", 20)) == NULL){
   fprintf(stderr, "\nNo se puede abrir la fuente\n");
   exit(-1);
}
```

Como puedes ver, primero declaramos una variable especial que podremos utilizar posteriormente en las funciones de escritura.

Ahora toca definir el color. Seguimos el mismo procedimiento, declaramos una variable y le asignamos los atributos:

```
SDL_Color color;

color.r=0;   /* Rojo  */
color.g=255; /* Verde */
color.b=0;   /* Azul  */
```

Una de las funciones de escritura requiere otro segundo color que se utilizara como fondo del texto. No repetiremos el proceso pues se crea de la misma forma.

Bien, las funciones principales para escribir en pantalla son 3:

```
1. TTF_RenderText_Solid(TTF_Font *font, const char *text, SDL_Color fg);

2. TTF_RenderText_Shaded(TTF_Font *font, const char *text, SDL_Color fg,
                                                           SDL_color bg);

3.TTF_RenderText_Blended(TTF_Font *font, const char *text, SDL_Color fg);
```

Te preguntaras que las diferencia. Pues ademas de los ultimos argumentos que son obvios, difieren tanto en rendimiento como en calidad.
La primera es la que ofrece una calidad mas baja, no obstante precisa de un tiempo menor para ejecutar su cometido. La tercera es todo lo contrario. Nosotros siempre utilizaremos la ultima, pues la definicion es perfecta y en un PC normal no percibiras una merma de rendimiento visible.

Estas funciones no escriben directamente en pantalla, sino que devuelven una superficie que sera la que posteriormente se imprimira sobre la pantalla en la region que nosotros deseemos. Un uso habitual seria este:

```
**-------------------------------------------------------------------------**

SDL_Surface *texto;
SDL_Rect region = (SDL_Rect) {300, 300, 0, 0};

texto = TTF_RenderText_Blended(fuente, "HACK THE WORLD", color);

SDL_BlitSurface(texto, NULL, pantalla, &region);

SDL_FreeSurface(texto);

**-------------------------------------------------------------------------**
```

La ultima llamada, como habras adivinado, libera la superficie y por lo tanto la memoria que esta siendo utilizada. Es una forma de trabajar bastante eficiente.

Con esto deberia ser suficiente para que puedas poner texto aleatorio sobre la pantalla. No obstante, aqui tienes otras funciones que pueden ser de utilidad:

- TTF_OpenFontIndex() -> Igual a TTF_OpenFont(), pero ademas recibe un 3er parametro que es un indice para ficheros tff que contengan varios tipos de letra.

- TTF_SetFontStyle(TTF_Font *, int) y TTf_GetFontStyle(TTF_Font *).

  La primera necesita una constante como segundo parametro que puede ser una de estas:

```
TTF_STYLE_NORMAL    -> Normal
TTF_STYLE_BOLD      -> Negrita
TTF_STYLE_ITALIC    -> Cursiva
```

TTF_STYLE_UNDERLINE -> Subrayado

Puedes investigar mas sobre esta libreria leyendo el contenido de su cabecera.

---[3.5 - Varios

Quiero dejar claro que esta seccion "no es prescindible", aqui se explicaran algunos aspectos que normalmente no entran dentro de las funciones mas basicas de SDL, pero que pueden proporcionar ciertas habilidades a tu aplicacion que la convertira en mucho mas eficiente.

---[3.5.1 - Envoltorios

El nombre de esta peque~a seccion puede parecer un poco raro; pero en realidad es la mejor forma de definir a las funciones que SDL ha creado para poder llamar a los metodos normalmente definidos en tu libreria habitual: <stdio.h>.

La cuestion es que en el archivo de cabecera <SDL/SDL_stdinc.h> podemos encontrar metodos como:

```
SDL_calloc()
SDL_free()
SDL_putenv()
SDL_memset()
...
y asi con muchas otras.
```

Pero en realidad SDL hace siempre una comprobacion de este tipo:

```
#ifdef HAVE_PUTENV
#define SDL_putenv   putenv
#else
```

lo que viene a decir que siempre que la funcion exista ya en nuestro sistema operativo, sera utilizada aunque usemos el nombre proporcionado por SDL.

---[3.5.2 - SDL_strlcpy & SDL_strlcat

Esto es interesante, muy interesante.

Ya todos somos conocedores de los problemas que implica el uso de funciones tales como: 'strcpy()' y 'strcat()'. En multiples ocasiones pueden llevar a situaciones de desbordamiento de buffer indeseables y peligrosas.

Cuando has leido algo acerca de "buffer overflows" y los autores de estos articulos te han concienciado de que el uso de 'strncpy()' y 'strncat()' es la panacea y todos tus problemas han desaparecido, entonces ya se te permite darte de cabeza contra un muro.

El tema es el siguiente, y muchos deberian saberlo ya. Las funciones anteriores no comprueban que el ultimo caracter de la cadena a copiar sea un \0. Lo que es mas, aunque esta lo tuviera, podria llegar a no copiarse. Me explico:

```
Cadena Orignal: [H][O][L][A][\0] -> Longitud = 5

Si el invocador de la funcion comete un error y llama a la funcion
'strncpy() pensando que la longitud de la cadena "hola" es 4. Entonces
la cadena destino quedara cortada sin terminar nunca en un caracter \0.
```

```
   Cadena Destino: [H][O][L][A]
```

Esto puede dar lugar a implicaciones posteriores que conlleven a errores dificiles de detectar e incluso fallos de segmentacion.

Existen 2 funciones en los sistemas operativos de la familia BSD llamadas: 'strlcpy' y 'strlcat' que siempre terminan el buffer destino con un caracter nulo aunque ello conlleve la perdida de un caracter de la cadena original. Un sacrificio que merece la pena.

Como en un sistema Linux normal no encontraremos estas funciones por defecto, podemos evitar compilarlas junto con nuestro programa haciendo uso de las proporcionadas por SDL, que, al no encontrarlas, hara uso de su propia implementacion.

---[3.5.3 Threads (hilos)

Si llevas tiempo en esto de la programacion y las palabras 'thread' y 'proceso' te son conocidas de sobra. La interpretacion de las funciones que ahora vendran no te resultaran nada complicadas.

No te lleves la idea equivocada de que una aplicacion que utilice SDL debe utilizar las siguientes funciones obligatoriamente. Es mas, yo suelo utilizar directamente las que proporciona <pthread.h>. Pero si te gusta seguir la armonia con la nomenclatura de tu aplicacion, esto es para tu uso y disfrute.

Debes crear siempre un puntero a una estructura 'SDL_Thread' tal que asi:

```
SDL_Thread *hilo;
```

En este puntero recibiras el resultado de la siguiente funcion:

- SDL_Thread * SDL_CreateThread(int (*fn)(void *), void *data)
    - Primer argumento -> Una funcion que recibe un argumento 'void *'
    - Segundo argumento -> Un valor opcional que puedes pasar a la funcion.

Y aqui tienes otras:

- Uint32 SDL_ThreadID(void) -> Devuelve el identificador del hilo actual.
- Uint32 SDL_GetThreadID(SDL_Thread *thread) -> Devuelve el identificador del hilo especificado.
- SDL_WaitThread(SDL_Thread *thread, int *status) -> Espera la finalizacion de un hilo.
- SDL_KillThread(SDL_Thread *th) -> Fuerza la finalizacion de un hilo.

---[3.5.4 - Graficos Primitivos

Puede, quizas, que en algun momento necesites dibujar alguna que otra figura basica tal como un rectangulo, un circulo, una linea o tan siquiera un misero pixel.

Las funciones incluidas en el archivo de cabecera <SDL/SDL_gfxPrimitives.h> cubren mas que de sobra esta necesidad. Dare una breve rese~a acerca de la mayoria de ellas pero no mostrare sus argumentos, pues varian muy poco de una funcion a otra e inflarian esta guia de manera desproporcionada.

Cuando una de ellas te interese, te esforzaras en investigar como se ejecuta correctamente.

- pixelColor() -> Dibuja un pixel en una superficie.
- hlineColor() -> Dibuja una linea horizontal en una superficie.
- vlineColor() -> Dibuja una linea vertical en una superficie.
- rectangleColor() -> Dibuja un rectangulo una superficie.
- boxColor() -> Dibuja un cuadrado en una superficie.
- lineColor() -> Dibuja una linea en una superficie.
- circleColor() -> Dibuja un circulo en una superficie.
- ellipseColor() -> Dibuja una elipse en una superifice.
- pieColor() -> Dibuja ... algo relacionado con graficos de quesitos???
- trigonColor() -> Dibuja un triangulo en una superficie.
- polygonColor() -> Dibuja un poligono de 'n' lados en una superficie.
- bezierColor() -> Dibuja una linea curva entre dos puntos de control.
- characterColor() -> Dibuja un caracter en una superficie.
- stringColor() -> Dibuja una cadena de caracteres en una superficie.
- gfxPrimitivesSetFont() -> Establece una fuente para las funciones previas.

Y existen otras "variantes":

1 - Las que cambian el sufijo 'Color' por 'RGBA' requieren un parametro 'Alfa' que indicara el nivel de transparencia del color.

2 - Las que llevan el prefijo 'filled' que rellenan la figura o poligono con el color que se le indique.

3 - Otras que utilizan un prefijo como 'aa'.

---[3.6 - Efectos

A continuacion detallaremos el funcionamiento de dos funciones que nos proporciona la cabecera <SDL/SDL_rotozoom.h>. La primera de ellas nos servira para ampliar superficies o regiones de superficies y la segunda, como indica tambien su nombre, para rotarlas el angulo que le especifiquemos.

A~ade esto a las opciones de compilacion de tu programa: "-lSDL_gfx".

---[3.6.1 - Zoom

Como ya se ha dicho, esta funcion que se define como:

- SDL_Surface *zoomSurface(SDL_Surface * src, double zoomx,
  double zoomy, int smooth);

1 - Provoca el zoom de la superficie pasada com 1er argumento.
2 - Tanto como se indique en el 2do y 3er argumento (factores de escalado).
3 - El ultimo parametro suele ser 0. Si es '1' utiliza anti-aliasing.
4 - Devuelve una nueva superficie ya ampliada.

Podriamos utilizar una funcion bastante general como la que mostrare a continuacion:

```
 **-------------------------------------------------------------------------**

   void do_zoom(SDL_Surface *sup, int sx, int sy, int x, int y)
   {
      SDL_Surface *szoom;
      SDL_Rect pos;

      szoom = zoomSurface(sup, sx, sy, 0);
```

```
      pos = (SDL_Rect){x, y, 0, 0};
      SDL_BlitSurface(szoom, NULL, pantalla, &pos);
      SDL_Flip(pantalla);

      SDL_FreeSurface(szoom);
   }
```

 --

A esta funcion le indicamos la superficie que queremos ampliar, los factores de escalado (un valor de 2 doblara el tama~o de la imagen) y las nuevas coordenadas donde situar esta nueva superficie ampliada.

Imaginese ahora que usted tiene la imagen de un mapa y que quiere que se produzca la ampliacion de este. Si usted utiliza la funcion anterior, tendria que indicarle que la nueva superficie se dibuje sobre las mismas coordenadas en que esta la anterior, pero hay un error. Estas copiando completamente la nueva superficie y esta es el doble de ancho y de alto. Se saldra de la region esperada y podria solaparse encima de otros elementos que haya dibujado en su aplicacion.

La solucion es definir un nuevo 'SDL_Rect' que defina una region igual al tama~o completo de la imagen original. De esta forma la superficie ampliada quedara recortada y ocupara el mismo espacio que la anterior.

Pero aun no esta todo dicho. La solucion anterior no es muy practica, pues siempre estariamos ampliando la misma region de la imagen (esquina superior izquierda).
Lo habitual es ampliar una zona circundante a aquel lugar donde nosotros hemos "clickado".

Entonces ahora debemos comprender un nuevo aspecto. Si nosotros hemos hecho "click" en la posicion (x, y), el mismo punto en la superficie ampliada deberia ser mas o menos (x*2, y*2). Pero ese no debe ser el lugar donde empiece la imagen ampliada, pues hemos dicho que queremos una region circundante, por lo tanto tambien tenemos que mostrar una porcion del mapa anterior a esas coordenadas. Lo logico, entonces, seria utilizar unos puntos de inicio parecidos a estos ((x*2) - 100, (y*2) - 100). El ancho y el alto de esta nueva superficie dependera, como siempre, del tama~o de la imagen original.

Un ejemplo practico, pero particular y personal, podria ser el siguiente:

 --

```
   void do_zoom(SDL_Surface *sup)
   {
      SDL_Surface *szoom;
      SDL_Rect pos, region;

      szoom = zoomSurface(sup, 2, 2, 0);
      region = (SDL_Rect){(x*2)-50, (y*2)-100, 100, 100};
      pos = (SDL_Rect){350, 200, 0, 0};
      SDL_BlitSurface(szoom, NULL, pantalla, &pos);
      SDL_Flip(pantalla);

      SDL_FreeSurface(szoom);
   }
```

 --

La mejor forma de comprender como funciona esto, es que elabores tus propios ejemplos y los adaptes a tus necesidades y a las necesidades de tus imagenes.

---[3.6.2 - Rotaciones

Bien. Para esta seccion tenemos dos funciones muy similares a la que se acaba de presentar hace un momento. Aqui las tenemos:

```
   - SDL_Surface *rotozoomSurface(SDL_Surface * src, double angle,
                                  double zoom, int smooth);

   - SDL_Surface *rotozoomSurfaceXY(SDL_Surface * src, double angle,
                                    double zoomx, double zoomy, int smooth);
```

La diferencia entre las dos es bastante evidente. En la primera, el parametro 'zoom' es comun y se aplica proporcionalmente tanto al ancho como al alto de la imagen a rotar. Un valor de 1 dejara la imagen en su tama~o actual.
En la segunda, como puedes observar, los factores de escalado son individuales.

En principio la imagen rotara en sentido contrario a las agujas del reloj. Lo que quiere decir que un valor de 90 para el argumento 'angle' rotara la imagen hacia la derecha.

IMPORTANTE: No te olvides de poner un '1' o '1.0' al valor de 'zoom'. Un valor de 0 haria invisible tu nueva superficie rotada.

Algo mas hay que tener en cuenta. Y es que si vuelves a dibujar siempre la imagen rotada en las mismas coordenadas que la original, la imagen estara girando continuamente sobre una esquina. Seguramente no sea este el efecto que deseamos.

El siguiente codigo mostrar una funcion generica para rotar imagenes y como corregir la posicion en cada iteracion:

```
 **-------------------------------------------------------------------------**

   void do_rotozoom(SDL_Surface *sup, int x, int y,
                    float zoomx, float zoomy, float angle)
   {
      SDL_Surface *szoom;
      SDL_Rect pos;

      pos = {x, y, 0, 0};

      szoom = rotozoomSurfaceXY(src, angle, zoomx, zoomy, 0);

      /* Rotar siempre la imagen con respecto al centro */
      pos.x -= (szoom->w - sup->w) / 2;
      pos.y -= (szoom->h - sup->h) / 2;

      SDL_BlitSurface(szoom, NULL, screen, &pos);
      SDL_Flip(pantalla);

      SDL_FreeSurface(szoom);
   }

 **-------------------------------------------------------------------------**
```

Si la imagen a rotar no es un cuadrado perfecto, recuerda siempre borrar esta poniendo un rectangulo negro encima, asi no quedara solapada con la nueva pudiendose ver regiones de la anterior.

---[4 - Consola Personal

Con lo aprendido hasta este momento, nos atreveremos a crear la representacion de una consola o shell, bastante arcaica pero funcional.

¿Que utilidad tiene una consola si lo que queremos es precisamente crear una GUI para hacer las cosas mas faciles? Muy facil, imagina que estas creando una interfaz para una suite de hacking; seria interesante que una ventana de la pantalla fuera una consola para poder ejecutar comandos directamente en el sistema y capturar su salida para imprimirla en nuestro programa sin tener que minimizarlo y abrir una nueva shell (menos aun cuando este se ejecuta a pantalla completa).

Todavia mas. Crear la representacion de cuadros de texto (algo que seguramente ya te habras preguntado) es mas que una odisea en una aplicacion con SDL. Te invito a que lo intentes y me muestres tus resultados (si lo hicieras te recomendaria C++, donde puedes convertir realmente cada elemento de la GUI en un objeto).

Es por ello y mucho mas que necesitas algun modo de proporcionar informacion a tu aplicacion, ya sean direcciones IP, MAC, registros de agenda, cuentas bancarias o aquello que tu programa quiera consumir.

Si las razones no te convencen, entonces lee esto igual, aprenderas un poco mas de como se comportan los graficos e incluso quizas recibas algun que otro consejo.

<SDL_console.h> puede proporcionarte por si misma varias de las caracteristicas que se presentaran a continuacion; pero nosotros somos artesanos del codigo y vamos a hacer algo propio por una vez.

---[4.1 - Representacion

Antes de nada necesitamos una estructura que contenga las coordenadas de nuestro supuesto cursor (de momento sera invisible). La estructura podria ser la siguiente:

```
   struct position{
      int x;                /* columnas */
      int y;                /* lineas   */
   } post;
```

Definiremos tambien 2 variables "globales", una que contendra el comando que estamos escribiendo hasta un maximo de 64 caracteres y otra que llevara el contador de cuantos se han escrito hasta el momento:

```
   int ncar = 0;     // Contador de caracteres
   char comando[64]; // Nos cuidaremos bien de los desbordamientos
```

Ahora abriremos una fuente y definiremos su color:

```
   TTF_Font *fuente;
   SDL_Color fgcolor;

   if ((fuente = TTF_OpenFont("FreeMono.ttf", 20)) == NULL){
      fprintf(stderr, "\nNo se puede abrir la fuente\n");
      exit(-1);
   }

   fgcolor.r=0;
   fgcolor.g=255; // Verde
   fgcolor.b=0;
```

Vale, ¿que es una consola visualmente? Pues diriamos de forma vulgar que un simple rectangulo negro. Exacto, eso es lo que vamos a crear,

pero un rectangulo en medio de la pantalla quedaria bastante feo.
Seria mas agradable cargar la imagen de una ventana como se ha mostrado
en secciones anteriores y dibujar sobre esta la consola; lo que es mas,
si el interior de esa ventana es totalmente negro, ni siquiera haria
falta este ultimo paso.

Imaginemos entonces que la ventana se situa en las coordenadas (0, 385)
y que su tama~o en pixeles es de (z) por (y).
Podemos crear un rectangulo negro dentro de esta ventana de modo que el
marco quede bien ajustado:

```
   /* Asumire que "pantalla" es la 'SDL_Surface *' principal */

   rect = (SDL_Rect) {11, 420, 688, 328};
   SDL_FillRect(pantalla, &rect, SDL_MapRGB(pantalla->format,0,0,0));
```

Se observa que empezamos varios pixeles mas adentro para no "montarnos"
encima de los marcos de la ventana.

Que mas necesitamos? El Prompt, si, eso es importante, lo escribiremos
con:

```
   escribir('#', 0); // Esta funcion se explica en el siguiente apartado
                     // el 0 en el segundo argumento indica que no forma
                     // parte de un comando, es un caracter de control.
```

---[4.2 - Escritura y borrado

En esta primera funcion, controlaremos varias cosas.

   1 - Si el caracter pertenece a un comando (*).
   2 - Si el caracter es un tabulador (\t).
   3 - Si el caracter es una nueva linea (\n).
   4 - Si estamos al final de una linea, escribiremos en la siguiente.
   5 - Si alcanzamos la ultima linea, realizar scroll().

(*) Los caracteres se escribiran en la consola de forma indefinida, pero
    solo seran almacenados como parte del comando hasta alcanzar la longtiud
    de 64.

 --

```
   void escribir(char car, int is_cmd)
   {
      int tab = 0;
      int nline = 0;
      char line[2];

      snprintf(line, 2, "%c\0", car);

      // Si aun no se han superado los 64 caracteres y forma
      // parte de un comando, lo copiamos al buffer.
      if (ncar != 0 && ncar < 64 && is_cmd == 1) {
         SDL_strlcat(comando, line, 64);
      }

      // Si es una nueva linea bajamos el cursor aumentando
      // post.y, nos vamos al principio restaurando post.x
      if (strncmp(line, "\n", 1) == 0) {
         post.y += 20;
         post.x = 15;
         nline = 1; // Boolean
```

```
      }
      // Si es un tabulador escribimos 7 espacios
      else if (strncmp(line, "\t", 1) == 0) {
         texto = TTF_RenderText_Blended(fuente, "       ", fgcolor);
         tab = 1; // Boolean
      }
      else
         texto = TTF_RenderText_Blended(fuente, line, fgcolor);


      if(nline != 1){
         if (post.x >= 688) { // Si alcanzamos la ultima columna
            post.x = 15;      // empezamos en una linea nueva.
            post.y += 20;
         }
         if (post.y >= 715) { // Si alcanzamos la ultima linea
            scroll();         // realizamos un scroll de la consola
         }

         // Imprimimos el texto y liberamos la superficie
         rconsola = (SDL_Rect) {post.x, post.y, 0, 0};
         SDL_BlitSurface(texto, NULL, pantalla, &rconsola);
       SDL_FreeSurface(texto);

         if (tab == 1)
            post.x += 12 * 7; // 12 x tamanyo tabulacion
         else
            post.x += 12; // Adelantamos el cursor un caracter
      }

      ncar += 1; // Incrementamos el contador de caracteres
      SDL_Flip(pantalla); // Actualizamos la pantalla
   }
```

 --

Como siempre, la pregunta. Que ocurre visualmente al borrar un caracter?

Que desaparece (de Perogrullo). En nuestro caso, que un peque~o rectangulo negro lo tapa.

Aqui controlamos lo siguiente:

   1 - Si estamos al principio de la consola, no hacemos nada.
   2 - Si no hay caracteres escritos, no hacemos nada.
   3 - Si estamos al principio de una linea, siempre que esta
       sea continuacion de otra anterior, nos movemos al final
       de esta para borrar el ultimo caracter.

Nunca debemos olvidar anular el ultimo caracter del array 'comando[]'.

 --

```
   void borrar(void)
   {
      if (post.x >= 27) { // Cuidado de no borrar el PROMPT

         // Si estamos al principio de todo o en una linea en
         // la que no se ha escrito ningun caracter, salimos.
         if ((post.x == 27 && post.y == 430) || (ncar == 1))
            return;

         // Eliminamos el ultimo caracter del buffer
         comando[strlen(comando)-1] = '\0';
```

```
         // Movemos el cursor un caracter atras, dibujamos un cuadradito
         // negro para tapar el caracter a borrar y disminuimos el contador.
         post.x -= 12;
         rconsola = (SDL_Rect) {post.x, post.y, 13, 20};
         SDL_FillRect(pantalla, &rconsola, SDL_MapRGB(pantalla->format,0,0,0));
         ncar -= 1;
      }
      else if (post.y > 430) { // Si estamos al comienzo de una linea que es
         post.x = 699;         // continuacion de otra anterior, saltamos al
         post.y -= 20;         // final de esta para borrar el caracter que
         borrar();             // alli se encuentra.
      }
      SDL_Flip(pantalla); // Actualizamos la pantalla
   }
```

---[4.3 - Scroll

El metodo es sencillo. Solo debemos tener un poco de imaginacion e intentar comprender como funcionan los graficos en este sentido.

¿Que ocurre visualmente cuando el scroll entra en accion?

1 - La primera fila de la consola desaparece.
2 - El resto de las lineas se mueven una fila hacia arriba.
3 - Aparece una nueva linea vacia al final de la consola.

¿Como realizar estos simples pasos?

Los pasos 1 y 2 se ejecutan en una misma accion, pues al copiar todo el contenido de la consola excepto la primera linea, y copiarlo una fila mas arriba, esta se pierde automaticamente.

El paso 3 es evidente. Creamos un rectangulo negro que se superponga a esta ultima fila, que tras realizar el paso anterior seria igual a la penultima.

El codigo lo aclara todo:

```
   static void scroll(void)
   {
      SDL_Rect orig;

      // Copiamos toda la consola excepto la primera linea (la que se pierde)
      // y lo pegamos desde el origen de las coordenadas.
      orig = (SDL_Rect) {15, 450, 685, 285};
      rconsola = (SDL_Rect) {15, 430, 685, 285};
      SDL_BlitSurface(pantalla, &orig, pantalla, &rconsola);

      // Tapamos la ultima linea con un rectangulo negro
      rconsola = (SDL_Rect) {11, 715, 688, 20};
      SDL_FillRect(pantalla, &rconsola, SDL_MapRGB(pantalla->format,0,0,0));

      post.y -= 20; // Restauramos post.y para no salirnos de la consola
      post.x = 15;  // Nos vamos al principio de la linea
   }
```

---[4.4 - Ejecutar comandos

Esta funcion es muy simple, aunque puede llegar a ser peligrosa si ejecutamos el programa como ROOT (por ejemplo con sudo) y no somos conscientes de que estamos ejecutando comandos con este permiso.

El comando almacenado en 'comando[]' es ejecutado mediante 'popen()' y su salida es capturada para escribirla caracter a caracter directamente en nuestra consola.

Los comentarios del metodo son mas que suficientes.

```
 **-------------------------------------------------------------------------**

   void ejecutar(void)
   {
      char c;
      FILE *cmd;

      // Nueva linea y al principio.
      post.y += 20;
      post.x = 15;

      cmd = popen(comando, "r");       // Ejecuta comando
      while((c = fgetc(cmd)) != EOF){ // Lee la salida
         fflush(stdout);               // Evitar comportamientos extranyos
         escribir(c, 0);               // Escribimos la salida en la consola
      }
      pclose(cmd);                     // Cerramos el descriptor

      SDL_strlcpy(comando, "\0", 1);  // Vaciamos el comando
      ncar = 0;                        // Contador a 0
      escribir('#', 0);                // Escribir nuevo PROMPT
   }

 **-------------------------------------------------------------------------**
```

Te voy a mostrar un ejemplo real de la funcion que yo utilizo en uno de mis programas para dar entrada a un monton de informacion que este precisa para realizar sus operaciones. Tan solo echale un vistazo y quedate con la forma de operar:

```
 **-------------------------------------------------------------------------**

   void ejecutar(void){

      char c;
      char *tmp;
      int n, fp;
      int spc = 0;
      u_int32_t p[6];

      // FILE *cmd;

      post.y += 20;
      post.x = 15;

      if (comando[0] == '$') {
         tmp = comando + 1;
         free(thost);
         thost = (struct hosts *) calloc(1, sizeof(struct hosts));
         SDL_strlcpy(thost->h_addr, "\0", 1);
```

```
      SDL_strlcpy(thost->h_addr, tmp, 15);
      spc = 1;
   } else if (comando[0] == '@') {
      tmp = comando + 1;
      n = sscanf(tmp, "%02x:%02x:%02x:%02x:%02x:%02x",
                  &p[0],&p[1],&p[2],&p[3],&p[4],&p[5]);
      if (n != 6) {
         asprintf(&cadena, "MAC Address is not valid\n");
         salida(1);
      }
      for (n = 0 ; n < 6 ; n++)
         thost->trg_ha[n] = (u_int8_t)p[n];
      spc = 1;
   } else if(comando[0] == '%') {
      tmp = comando + 1;
      asprintf(&device, "%s", tmp);
      close_net();
      init_net(2);
      spc = 1;
   }

   if (strncmp(comando, "help", 4) == 0) {
      show_help();
      spc = 1;
   } else if (strncmp(comando, "clear",5) == 0) {
      clear_scr();
      spc = 1;
   } else if (strncmp(comando, "trace", 5) == 0) {
      trace_lines();
      put_mapa();
      spc = 1;
   } else if (strncmp(comando, "cleanmap", 8) == 0) {
      clean_gh();
      carga_mapa(1);
      spc = 1;
   } else if (strncmp(comando, "exit", 4) == 0) {
      terminar = 1;
      spc = 1;
   } else if (strncmp(comando, "smsn", 4) == 0) {
      sniff_msn();
      spc = 1;
   } else if (strncmp(comando, "surl", 4) == 0) {
      sniff_url();
      spc = 1;
   } else if (strncmp(comando, "fprint", 6) == 0) {
      fingerprint(thost->h_addr, 2);
      spc = 1;
   }

/* THIS FUNCTION IS VERY INSECURE, IT
 * PLAY COMMANDS AS ROOT. UNCOMMENT
 * THIS UNDER YOUR OWN RESPONSABILITY.

   if (!spc) {
       cmd = popen(comando, "r");       // Play command
      while ((c = fgetc(cmd)) != EOF) { // Read output command
         fflush(stdout);                // Flush the buffer
         escribir(c, 0);                // Print output to console window
      }
      pclose(cmd);                      // Close descriptor
   }
*/

   SDL_strlcpy(comando, "\0", 1);
```

```
      ncar = 0;
      escribir('#', 0);
   }
```

 --

Deja volar a tu imaginacion.

---[4.5 - Utiles

Te presentare aqui dos funciones muy sencillitas que puedes utilizar en aquellas aplicaciones en que utilices esta consola.

La primera de ellas es el conocido "clear screen" que borra todo el contenido de la consola:

 --

```
   void clear_scr(void)
   {
      SDL_Rect rscr;

      rscr = (SDL_Rect) {11, 420, 688, 328};
      SDL_FillRect(pantalla, &rscr, SDL_MapRGB(pantalla->format,0,0,0));

      post.x = 15;
      post.y = 430;
   }
```

 --

Menuda tonteria, diras. Y estas en lo cierto. Un rectangulo negro que tapa todo el contenido de la consola y un reinicio de coordenadas. Pensaras que esta funcion no te vuelve a proporcionar un 'PROMPT', cierto tambien; pero fijate, si la utilizas dentro de la funcion 'ejecutar()' activandola cuando 'strncmp(comando, "clear", 5) == 0' entonces ya tenemos cubierta nuestra necesidad.

Imaginate ahora. Has programado un escaner de puertos que ejecutas haciendo click en uno de los botones del menu y te gustaria que su resultado se imprimiera por consola. Entonces puedes utilizar esta llamada:

 --

```
   char *cadena;

   void salida(int is_end)
   {
      while (*cadena) {
         escribir(*cadena++, 0);
      }

      SDL_strlcpy(cadena, "\0", 1);
      SDL_strlcpy(comando, "\0", 1);

      if (is_end) {
         ncar = 0;
         escribir('#', 0);
      }
   }
```

 --

Cada vez que quieras imprimir una frase en la consola, puedes escribir en tu codigo esto:

```
asprintf(&cadena, "Hack The World by %s\n", "blackngel");
salida(1);
```

Queda a tu eleccion aqui volver a situar un prompt o dejar la linea preparada para imprimir a continuacion en otro momento.

---[5 - Conclusion

El objetivo de este articulo, que bien pudiera haberse mencionado al principio, era que pudieses crear tus propias interfaces graficas sin necesidad de vertelas con un entorno de desarrollo visual cuya potencia puede a veces desbordar tus capacidades.

Crear aplicaciones con el aspecto de un juego puede llegar a ser realmente estimulante. Es un mundo entero abierto a la imaginacion donde puedes elaborar desde interfaces con aspecto futurista y de ciencia-ficcion hasta lo mas exoterico que a ti se te pueda ocurrir.

Para terminar, un lugar donde encontraras todo o casi todo lo que necesitas para programar con SDL. Podras leer una rese~a acerca de lo que el futuro le reserva a SDL. Lo encontraras aqui [4].

Podeis contactar conmigo en <blackngel1@gmail.com>, los insultos seran bien recibidos siempre que sean lo bastante originales.

Open your eyes, open your mind.

---[6 - Referencias

[1] SDL
http://www.libsdl.org

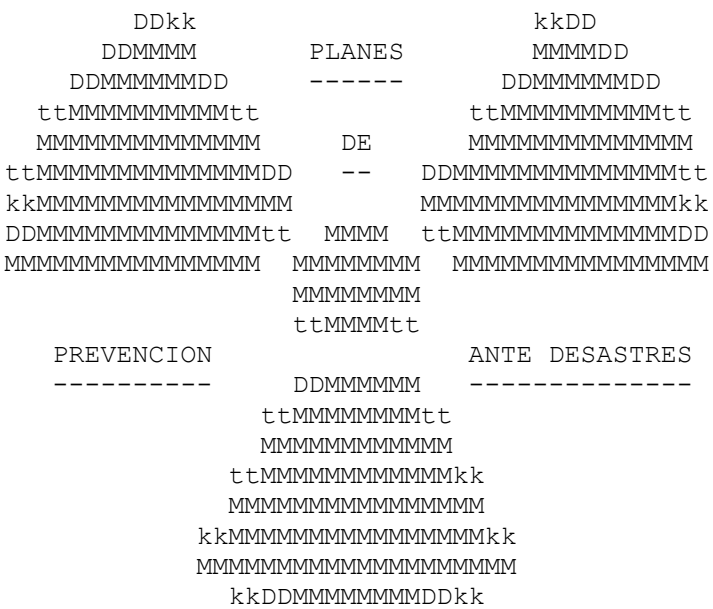
[2] Programacion con SDL
http://www.linux-magazine.es/issue/01/programacionSDL.pdf

[3] Programacion de videojuegos con SDL
http://www.agserrano.com/libros/sdl/%5Bebook%5DProgramacion%20de%20videojuegos%20con%20SDL.pdf

[4] Tutorial de libSDL para la programacion de videojuegos
http://softwarelibre.uca.es/tutorialSDL/TutorialSDL-30012008.pdf

EOF

-[0x0D]--
-[Planes de Prevencion ante Desastres]-----------------------------------
-[by Anay]---SET-35--



~~~~~~~
by Anay
~~~~~~~

{==============}
{ INTRODUCCION }
{==============}

Para quienes seguimos con asiduidad diaria los periodicos seguro que habremos leido de muchos "accidentes" naturales o no que han dejado desde edificios arrasados a barrios enteros sin luz pasando por ciudades destrozadas.
No quiero entrar a pensar o hablar sobre como hay que reaccionar en dichos casos a nivel humano donde se han podido producir desde accidentes hasta muertes sino que vamos a pensar un poco en el otro aspecto de estos incidentes.

Quiero haceros participes de la realidad y es que nadie esta a salvo a lo largo de su vida de que su negocio se queme, o se derrumbe ante un terremoto o que acabe off-line durante dias.

Bien, ¿que seria de las grandes empresas si no estuvieran preparadas para estos sucesos? Para que un Ike o un Katrina no destroze tus oficinas o pueda acabar con tus trabajadores es muy dificil prepararse pero si es posible estar operativo parcial o totalmente al día siguiente pudiendo seguir con tus operaciones casi de forma normal.

Voy a poner un ejemplo de lo que estoy hablando, uno que seguro podreis recordar de cuando se quemo el edificio Windsor en Madrid, la torre termino destruida y, de echo, ahora mismo apenas queda mas que un enorme agujero donde en su momento se alzaba orgullosa.
Dentro de dicha torre existian empresas las cuales muchas de ellas eran simplemente la parte organizativa de toda las ramas que puede tener una gran empresa ¿que habria sucedido si no estubieran preparados para volver a funcionar en horas o apenas un par de dias despues?

Sencillamente que las perdidas ante lo sucedido les habria llevado a la quiebra.

{==========}
{ CONCEPTOS }
{==========}

La Wikipedia define asi a los Planes de contingencia:

"Un Plan de contingencias es un instrumento de gestion para el buen gobierno de las Tecnologias de la Informacion y las Comunicaciones en el dominio del soporte y el desempeño (delivery and support, véase ITIL).

Dicho plan contiene las medidas tecnicas, humanas y organizativas necesarias para garantizar la continuidad del negocio y las operaciones de una companyia. Un plan de contingencias es un caso particular de plan de continuidad aplicado al departamento de informatica o tecnologias. Otros departamentos pueden tener planes de continuidad que persiguen el mismo objetivo desde otro punto de vista. No obstante, dada la importancia de las tecnologias en las organizaciones modernas, el plan de contingencias es el mas relevante."

En mi opinion es una definicion bastante acertada y es lo que vamos a tratar de desarrollar aqui, como establecer todas esas medidas de continuidad de negocio y para ello tenemos que definir los planes de contingencia y dividirlos en tres grupos ya que no todas las empresas tendran las mismas necesidades.

{==================================}
{ TIPOS DE PLANES DE CONTINGENCIAS }
{==================================}

La division mas acertada (si leeis otros documentos sobre planes de continuidad, hay miles en la red, esta division puede ser muy diferente) es la que he leido en Hispasec donde diferencian los tipos de planes asi:

1.- Sitios frios: sistemas con alta tolerancia a la indisponibilidad. La restauracion se puede realizar en 2 o 3 dias.

2.- Sitios templados: Sitios con tolerancia a 1 o 2 dias.

3.- Sitios calientes: son aquellos cuya tolerancia es muy limitada siendo de entre 4 y 24 horas.

Despues nombra a los sitios calientes tipo espejo, con esto quiere decir lugares cuya replicacion total esta siendo realizada en tiempo real, pero Hablaremos de estos sitios un poco mas adelante.

Ya tenemos catalogados los tipos de negocios y el tipo de contingencia que cada uno de ellos necesita. Ahora seguro que os abordaran cientos de preguntas, la que en mi experiencia mas suele tenerse es ¿y de que hago yo mi plan de contingencia? Un sitio frio puede ser perfectamente una tienda de reparacion de consolas, donde tan solo uno o dos pc's albergan una gran base de datos de clientes la cual para su negocio es vital, ellos ya saben que han de contemplar en su plan de contingencia: La base de datos, pero ¿que pasa con esas empresas que tienen mas de 20 departamentos? Vamos a tratar de focalizar que salvar primero y, por regal general, seran esos departamentos que generan mas ingresos. Siempre que una empresa invierte dinero en planes de continuidad es para reducir sus gastos de estar inoperativa por lo tanto los departamentos que dan esos ingresos tendran que ser especialmente revisados despues de estos tenemos que pensar en los departamentos raiz de los mismos, es decir, un departamento de ventas que genera al final de dia unas 100 ventas del producto X por mucho que sigan teniendo sus herramientas de venta si no esta respaldado por otro de contabilidad que haga la correcta gestion acabas por tener aun un problema mas grande, por supuesto no tenemos que olvidarnos de departamentos como direccion, alguien tiene que estar alli supervisando todo el correcto funcionamiento, etc,

etc... por lo tanto vamos a ver ejemplos de contingencia para los tres tipos de planes de una forma mas detallada.

{====================================}
{ EJEMPLOS DE PLANES DE CONTINGENCIAS }
{====================================}

{----- Sitios frios -----}

Un sitio frio como ya hemos dicho antes, no es un plan de contingencia de emergencia mayor, podemos aplicar aqui el ejemplo que he dado antes de la tienda de reparacion de consolas, esas que hay por todos lados donde te ponen el chip ;)

Dicha empresa consta de 3 trabajadores, tienes una base de datos muy completa donde alberga todos los datos de los clientes ya que sus chips dan garantia de por vida. La empresa consta de un PC con una base de datos MySQL y dos ordenadores que mediante un software lanzan consultas o meten datos en dicha base de datos. Uno de los ordenadores esta en la parte de cara al publico mientras que el otro esta dentro del almacen y el servidor en el mismo almacen.

Dentro de lo pequenya que es la infraestructura ya es lo suficientemente desarrollada como para hacer un pequenyo servidor, he querido ponerla ya un minimo de compleja.

En este caso necesitamos que nuestro plan no sea ni demasiado sofisticado ni demasiado caro, estamos hablando de una tienda que seguro su presupuesto para estas cosas es bastante limitado.

En los tres ejemplos vamos a seguir con la división de medidas que han puesto en la Wikipedia, es la division mas realista que he leido por la red, adaptada a este ejemplo:

Medidas tecnicas:

1.- Extintores de incendios: esto por ley deberia estar en cada negocio y aunque la ley no obligara te salvaria de problemas en un 80% de los casos ya que lo mas habitual es que se produzcan pequenyos incendios o por cortocircuito, colillas mal apagadas... Con esto quedaria solucionado.

2.- Detectores de humo: Un medida tambien, en mi opinion que deberia ser obligatoria en todo local y mas ahora que no se puede fumar en ningun sitio ;) Ligeramente mas cara pero en este caso con que hubiera uno en la zona de clientes y otro en la zona de almacen basta.

Medidas organizativas:

1.- Seguro de incendios: Bueno segun lo que estamos hablando aqui como ya he mencionado el incendio es la mayor probabilidad de acabar con nuestro negocio pero hay mas cosas que yo trataria de tener en cuenta segun como fuera el local, tendrian que hacer una buena inspeccion, se de muchos locales que estan situados bajo grandes edificios que en sus techos tienen enormes tuberias de desague, en ese caso contra inundaciones u otros eventos que se puedan ver que son posibilidad.

2.- Procedimiento de copia de respaldo: Este es el punto donde mas quiero hacer inciso. La copia de respaldo en este caso es vital, sin ella cualquier minima incidencia que suceda nos pondra en grave peligro de continuidad.

¿Que tipo de copia de respaldo necesita un negocio de este tipo? No nos vamos

a volver locos, hay cientos de software de backup gratuitos y bastante buenos, yo voy ha hacer refencia a uno que he usado y funciona de maravilla en red o local: Cobian Backup para una pequenya red como esta seria un software perfecto, podemos encontrarlo en su ultima version aqui:

- http://www.educ.umu.se/~cobian/cobianbackup.htm

Facil de instalar y tambien de interfaz lo suficientemente intuitiva para configurar las tareas.

Otra parte importante de este punto es: ¿que planteamiento de backups vamos a llevar?

Cuando queremos hacer un backup de una empresa este punto es el mas complicado, en todos los casos yo suelo seguir el mismo criterio:

* Disco para realizar backups diarios: Cada dia durante las horas en que no haya nadie en la oficina,para que no noten ralentizacion o la red ligeramente saturada, se hace un backup del dia, para que aun sea mas ligero con hacerlo incremental (para quienes no estan familiarizados con los software de backup los incrementales solo realizan backup de aquello que ha cambiado desde el ultimo que han realizado ahorrandonos tener que hacer backup de todos los datos y de este modo los backups tardan bastante menos en realizarse, esto puede ser muy util para grandes empresas donde sus backups son de tamanyos de Terabytes y no podemos consentir que duren demasiadas horas haciendose ya que, por regla general cuando un backup se esta haciendo la produccion de la empresa se nota resentida por el trafico de red).

Pero los incrementales tienen un riesgo y es que una vez que tienes dicha herramienta en muchos casos el usuario la usara para mas fines, es decir, recuperar archivos que hayan podido borrar por accidente. En dicho caso resulta extremadamente complicado para una persona que no esta familiarizada con la informatica en general y backups en particular, entender que ha de restaurar primero la ultima normal y despues las incrementales hasta el dia en que necesita dicho archivo.

Por esa razon yo suelo primero hacer una prueba de backup y si su tamanyo no es excesivo y veo que se realiza con normalidad en un periodo de 0-3 horas suelo dejarles en copias normales cada dia, asi ellos pueden realizar sus operaciones de recuperacion con mayor normalidad.

* Disco de backup semanal: Este disco es el que destino de mas contingencia, es decir, este disco de backup le configuro para que haga una copia normal (para los que no esteis familiarizados una copia normal es como una copia completa de toda la informacion que hay en las maquinas indicadas hasta el momento en que comienza a realizarse dicho backup).
Tambien se realiza durante un horario en que la tienda de chips este cerrada. Este disco es mas importante que el de diario ya que es el que queremos que salga de la tienda, para cuando suceda una contingencia si se han destruido el otro durante la misma tenemos en otro sitio totalmente distinto a este disco con todos los datos bien guardados.
Si, se que hay 1 semana de retraso, es decir que si este backup se hace los Miercoles y la contingencia sucede un martes tendremos en el un desfase de 4 dia laborables, pero es algo que hay que asumir, ningun usuario va a preocuparse de sacar y cambiar correctamente los discos de forma diaria, a la primera semana seguida que lo hagan se olvidaran asi que este metodo es mucho mas fiable y seguro de que lo haran. Ese pequenyo lapso de tiempo es algo que tenemos que asumir.

Detengamonos un segundo en el punto de ¿donde guardamos dicho disco? En grandes empresas, como luego veremos, tendran centro espejo o centros de contingencia preparados donde seguramente pueden llevar sus copias, pero en este caso la gente tendera a llevarselos a sus casas, yo NO LO RECOMIENDO, la casa es uno de los sitios mas peligrosos para llevar la precidada copia

de respaldo de tu negocio, en una casa pueden suceder mil accidentes y no veas como se multiplica esta posibilidad si en dicha casa hay ninyos :)
Yo siempre recomiendo tener una pequenya caja en un banco, donde se vayan dejado semana a semana dichos discos asi si hay varios empleados todos ellos pueden dirigirse a dicha caja y es una tarea compartida.

- Procedimiento de actuacion en caso de incendio: Este punto es muy importante a todos los niveles, cuando hay un incendio si a la gente no se les informa cual ha de ser su correcto comportamiento seguramente saldra corriendo de alli pudiendo aun causar peores consecuencias. Si todos salen de forma atropellada o por el lado incorrecto pueden desde chocarse con bomberos hasta tropezarse entre ellos. Eso lo veremos mas detenidamente en otro ejemplo en que esten ya en un edificio mas grande. En el caso de una pequenya tienda creo que lo mas importante es guardar la tranquilidad lo maximo posible y evacuar dicha tienda. Cuando hay un incendio olvida todo el material que haya y simplemente sal de alli con tranquilidad, recuerda que tu eres lo mas importante y estate tranquilo porque tenemos la copia de tu informacion, el resto del trabajo sera del seguro que ya hemos contratado antes ;)

En la Wikipedia tambien habla de las medida humanas, aunque yo creo que son un añadido que ya vienen incluidos en los anteriores puntos, es evidente que si contratas un sistema de backup alguien tendra que hacerse mas responsable del mismo y formarse un minimo de como usuarlo correctamente dia a dia, no es necesario que sepa hacer correccion de errores de las tares de backup (que siempre hay y muchos) pero si que sepa cambiar los discos, inventariarlos....

Me gustaria anyadir algo que en los puntos anteriormente comentados no se ha dicho, normalmente se pondria el punto de alquiler de equipos alternativos, estamos hablando de una pequenya tienda de barrio, si algo realmente grabe ocurre tendran el plan de contingencia sera realmente mas lento, hasta que pongan a punto la tienda de nuevo (siendo realistas asi sera) pueden pasar meses, asi que no vamos a pensar en otros centros pero si hay una cosilla que muchas veces pasan por alto y tambien es importante matizarlo, es una medida realmente util y barata una pequenya UPS configurable.
En nuestro ejemplo estamos hablando de bases de datos, todo el que haya trabajado con ellas sabra lo mal que le sienta un mal apagon a una base de datos sea SQL, MySQL, Oracle...
Una pequenya UPS configurable nos guardara el buen funcionamiento y el buen estado de nuestra maquina ya que si hay un apagon de la luz sin UPS simplemente el "servidor" se apagaria sin mas, sin embargo este pequenyo aparato que apenas valen 40 euros enchufado a nuestro servidor le dara corriente durante unos 10 minutos y bien configurado nos apagara la maquina correctamente sin que se produzcan daños a nivel de software por el mal apagado del sistema.

En un pequenyo punto que me cuesta de ver en la mayorias de tiendas a las que voy y que sin embargo seria lo primero que pondria en un sistema de contingencia.

{---- Sitios templados ----}

Como ya hemos comentado antes, un sitio templado son aquellos que tienen un poco menos de tolerancia a volver a su normal desarrollo del trabajo, vamos a poner un ejemplo tambien facil de ver:

La empresa es un buffet de abogados, esta ubicado en la principal zona de inversiones y negocios de la cuidad en un gran edificio de mas de 25 plantas. Su infraestructura consta de un servidor de datos donde se guardan miles de documentos de los disntintos casos que se manejan y recientemente han incluido un servidor de exchange (servidor de correo electronico) ya que han contratado a 2 jovenes informaticos y pueden administrarlo adecuadamente. La empresa tiene en total unos 50 trabajadores.

Bien, vamos a seguir las medidas que ya conocemos pero ahora su desarrollo es

dintinto esto ya es una oficina mas compleja, tiene mas usuarios y tambien dispondremos de mas presupuesto ;)

Medidas tecnicas:

1.- Extintores contra incendios: En este caso esto en concreto no sera algo de lo que devamos ocuparnos nosotros, estamos hablando de un gran edificio de mas de 25 plantas, el sistema de extincion del mismo es responsabilidad de quienes gestionan el edificio y por tanto son ellos los que han de poner los extintores en todos sus tipos (si, hay varios tipos de extintores) por todos los sitios tal como indica la ley, normalmente los encontraremos por las salidas de emergencias.

2.- Detectores de humo: Volvemos al caso anterior, esto deberia venir incluido con el edificio. Pero ante ambos puntos vamos a detenernos un segundo, pensemos que por primera vez disponemos de un CPD (sala de frio que alberga servidores y todo lo que tendremos que administrar, cableados, centralitas de salidas de llamadas....) el tema del CPD deveriamos gestionarlo nosotros, esta muy bien que dentro del mismo haya un detector de humo del edificio (es mas aseguraros de que haya uno) pero estamos hablando de una sala de frio la cual deberia estar totalmente cerrada a lo usuarios y gente externa y que gestionaremos nosotros.
Depende de como sea dicha sala tendremos que poner un sistema de extincion propio, unas bombonas de extincion automaticas o algo alternativo acorde a nuestro presupuesto, y si, esto tambien sera responsabilidad de los 2 jovenes informaticos y creo que es el punto que mas se tendran que pensar.
Ojito con que sistema de extincion os ponen que no sea excesivo, hoy como me encuentro alegre os voy a contar una anecdota para que veais cuan importante es esto, por cierto es veridica y ha pasado cientos de veces:

Existen empresas que cuando las contratas para que te analizen la sala de frio y te pongan un sistema acorde con la misma van de listos y siempre tienen a ponerte lo mas caro. La empresa que yo os comento tenia un CPD de 5 metros de ancho x 10 de largo, y la empresa de extincion les puso un sistema con sensores dentro de la misma y una bombona tambein dentro.
Dicha sala pese a su tamaño tenia 3 armarios repletitos de servidores, un buen dia vino un tecnico a revisar y terminar de instalar la bombona en dicha sala. Esa misma tarde 2 horas despues de que el tecnico abandonara el edificio un error en la placa del sistema de extincion hizo que la bombona se activara. Hasta aqui bien, el problema es cuando esa empresa habia dejado una bombona de gas alon que "exploto" a mas de 50 atmosferas de presion y dicho tecnico la habia dejado suelta, el unico punto que tenia de freno era el pequeño cable que la activaba.

El resultado de lo que os cuento fue una sala tecnica completamente destruida, los techos con la presion se calleron (estamos hablando de techos falsos) dicho gas en ese entorno es totlamentente ilegal lo que hace es "comerse" el oxigeno y haciendo el vacio (por decirlo de un modo) pues el incendio se apaga y, claro, a 50 atmosfecras una bombona de mas de 1 metro de alto estubo durante toda la descarga del gas dando bandazos por toda la sala llegando Incluso a volcar los armarios de racks con los fortisimos golpes que les dio.

Dicha empresa solo les quedo dar gracias a que dentro de el CPD no habia ningun tecnico en ese momento sino no habria salido de alli vivo o por algun golpe ya que todo en la sala quedo destruido, o por falta de oxigeno ya que la bombona salto sin dejar los 10 segundos que tenia de evacuacion o porque el techo se callo totamente con formas bastante peligrosas al recibir un impacto de 50 atmosferas no distribuido como debiera.

Espero que la historia os haya impresionado lo suficiente como para que os informeis bien de cual es el sistema de extincion adecuando para vuestro CPD, pensad que pasareis en el mas tiempo que fuera y ese sistema tiene que apagar incendios pero tambien salvaros a vosotros.

3.- Salidas de emergencia: Eso es responsabilidad total del edificio, lo que si deberiais hacer vosotros es revisar que las puertas se abren correctamente aunque sea una vez cada 15 dias, por millones de razones esas puertas se pueden bloquear con el peligro que eso conlleva. Y abrir y cerrar una puerta no son ni 2 minutos.

4.- Equipos informaticos de respaldo: Este es tambien un punto que hasta este caso no habiamos tenido.
Estamos hablando de un buffet de abogados con un presupuesto aun limitado y siendo un sitio templado por lo que yo creo que en este caso unos portatiles normalitos (de precio y de hardware) podrian valernos si compramos una buena serie todos del mismo modelo, podremos agilizar su preparacion con la creacion de unas imagenes con el sistema y software ya preparado.
Un software que sirve para dicho fin y lleva anyos siendo el lider de esto es el Norton Ghost, no es gratuito pero los que he probado que si lo son no me han dejado las cosas o no me han funcionado como debieran , este software cuesta alrededor de 70 dolares que en euros pues.. dividid :) Su funcionamiento lo teneis documentado en miles de sitios de la red ya que, como he dicho, es un software muy popular.

Con ese software en horas podeis dejar preparados unos 20 portatiles de backup o pcs de sobremesa si veis que para vosotros resulta mejor. Ni que decir tiene que dichos equipos deberan estar ubicados en otro emplazamiento totalmente alejado del edificio con las oficinas, vamos que si lograis encontrar algun sitio en la otra punta de la ciudad, mejor.

Medidas organizativas:

1.- Seguro de incendios: Lo ya comentado anteriormente, cuanto mejor seguro tengamos mas tranquilos dormiremos.

2.- Precontrato de equipos informaticos y ubicacion alternativa:
Vayamos por partes, acabamos de comprar 20 portatiles y hemos cargado su imagen con un ghost haciendolos casi 100% operativos (dicho el casi porque en un dominio pues el 100% no lo van a ser y menos todos con el mismo SID) por lo tanto lo del alquiler vamos a dejarlo para quienes su presupuesto haya dicho que no llega para hacer dicha compra.
En cualquier caso sean alquilados o no, el ghost nos ayudara a tener los equipos alquilados en 1 dia o 2 como mucho, ya preparados asi que esa parte sigue siendo imprescindible. Ubicacion alternantiva, esto ya si es mas importante, dependiendo de la empresa se necesita una ubicacion alternativa o no conozco algunas que con los portatiles se fueron y cada uno mientras se restauraba la normalidad trabajaba desde su casa.
No obstante hay cientos de empresas que se dedican a alquilar locales preparados con internet, CPD y todo tipo de servicios que necesites. Puedes alquilar una de estas mini oficinas y dejar alli a los usuarios operativos mientras tu vuelves a poner normalidad en las instalaciones originales.

3.- Procedimiento de copia de respaldo:
Ya tenemos mas presupuesto y mayor infraestructura pero pese a ello el tema del backup no cambia demasiado. En este caso aun habiendo dos tecnicos yo seguiria con la copia de diario y con la de semanal. Sacando la semanal como ya he explicado.
Puede que en este caso lo que yo cambiaria seria el programa de backup, he dicho un servidor exchange a proposito ya que no todos los sistemas de backup hacen respaldos de estos software (exchange, sql, iis...) correctamente, en estos casos recomiendo software de backups que tengas agentes especificos para estos programas tan vitales para nuestra empresa.
Uno que he usado durante anyos y que tiene un soporte ante incidencias bastante aceptable es el Symantec Backup Exec, dispone de agente especificos para

cualquier software que necesites (Windows, Linux o Unix, yo estoy poniendo ejemplos de Windows porque es mi especialidad, lo siento :)).
Symantec es de pago, y no es de los que se llamen especialmente baratos, el software cuesta un dinero, los paquetes van aparte (existen cientos tu puedes comprar los que necesites) y cada año has de renovar el contrato para tener soporte tecnico.
No, no es barato pero en infraestructuras que queremos hacer backups completos de las maquinas, en un entorno con dominios y programas de correo me parece una alternativa digna de pagar. Otro cambio que haria en un entorno asi es que anteriormente hemos hablado de discos y es que para una tienda de chips de consolas con discos USB para hacer los backups tienen mas que de sobra, pero esto ya es distinto aqui no nos valen dichos discos porque nos serian muy lentos y muy inseguros (todos sabemos que mueren con facilidad pasmosa) aqui ya tendriamos que pasarnos ha realizar los backup en unidades de cintas, son muy fiables, robustas y sobretodo notablemente mas rapidas. Una detalle mas que casi se me olvida, esta empresa tendra un volumen muchisimo mas alto de datos para respaldar, si hacemos backups normales puede que las horas de duracion de los backups se nos alarguen demasiado, en este caso ya contamos con personas cualificadas asi que en entornos asi ya cobra mucho mas sentido hacer backup incrementales al poderse gestionar bien.

4.- Procedimiento en caso de actuacion en caso de incendio:
En este caso el procedimiento es mucho mas amplio e importante que en el anterior, estamos hablando de un gran edificio donde a lo mejor estamos en la planta 20 trabajando, si no tenemos muy claro lo que cada uno ha de hacer podria tener consecuencias realmente grabes.
En primer lugar la evacuacion ha de ser ordenada, se han de utilizar unicamente las escaleras de emergencias, esta totalmente prohibido usar los ascensores ya que en caso de incendio hacen las veces de chimenea y nos asfixiariamos rapidamente.
La evacuacion ha de ser ordenada y con la mayor tranquilidad posible, hay que bajar por el lado de la derecha de las escaleras dejando el centro de la misma como via para bomberos, es muy importante seguir esto ya que si alguien se pone a correr por el lado de la izquierda los bomberos que suben pisos a velocidades de vertigo podrian chocarse con el y en un incendio el tiempo es crucial.
Tiene que existir un delegado para grupos, que se asegure de que todo su grupo o planta ha evacuado la misma esto siginifa revisar que no haya nadie en los banyos o salas de arhivos o algo asi que no haya podido oir la alarma y no se quede ahi.
Al delegado se le ha de reconocer o con un chaleco, una gorra o algo que los distinga, esta muy bien en esos casos que todos los delegados del edificio vayan con la mismas caracteristicas que los distingan ya que sera con ellos con quien hablen los responsables de seguridad sobre si ha quedado alguien en los pisos o ha pasado algo y a quienes diran lo que sucede y el tiempo que estaran sin poder regresar a sus trabajos. Tener tranquilidad, cuanto mas tranquilos esteis mas tranquilos estaran los de alrededor, por desgracia todos estos temas suelen llevarlo tambien los informaticos, ya que en casi ninguna empresa existen departamentos que hagan labores tan especificas asi que tendreis que asumir la mayor tranquilidad hasta que esteis todos a salvo en la calle en el punto de encuentro que os hayan indicado el personal de seguridad del edificio.

Como en ya he comentado en el punto anterior hay otras medidas indispensables, reitero el tema de la UPS, que en este caso sera mas potente y, claro, mas cara. Teniendo un buen backup, UPS para incidencias menores, y un buen sistema de extincion y detecion de incendios ya teneis lo suficiente como para estar tranquilos.

{---- Sitios calientes ----}

Por fin hemos llegado al ultimo grupo, este es el grupo mas delicado y que puede tener muchos perfiles. Puede ser una linea de negocio que lo que realizan

necesite funcionamiento constante, servicios de telefonia, una empresa de operaciones en bolsa... o puede que sea una gran central con miles de trabajadores.

Vamos a poner el caso de la empresa de bolsa que es el mas atipico que podemos encontrar.

En este caso la empresa se haya ubicada en el mismo edificio que la de abogados, en pleno corazon financiero de la ciudad en un edificio de mas de 30 plantas, dicha empresa tiene alrededor de un centenar de personas asi que ya no ocupan tan solo una planta.
Disponemos de un CPD mas grande donde albergan servidores desde SQl, Exchange, Datos a todos los servicios de telefonia y software de operaciones de bolsa que necesitan. La empresa cuenta con un equipo formado por 5 informaticos y su presupuesto para contingencias teniendo en cuenta la linea de negocio es mucho mas alto.

He puesto un ejemplo bastante complicado ya que al ser una empresa de bolsa estan totalmente reguladas en cuanto a contingencias por las normativas de la CNMV la cual es la entidad que regula estos tipos de empresas, pero tiene las caracteristicas que necesitamos para un sitio caliente ya que en el mundo de la bolsa los segundos son oro asi que el tiempo de respuesta ante contingencias ha de ser minimo.

Medidas tecnicas:

En este caso las tres primeras medidas son ya igual que las explicadas anteriormente, no hay muchas mas diferencia que contar, si hablaramos de una empresa que tiene un edificio completo para ella ya tendriamos personal que se dedicara en exclusiva a la seguridad ante incidentes de todo tipo, incluso los CPDS tendrian salas anexas con las bombonas de extincion, pero claro hablamos de empresas cuyos CPDs con de plantas completas con cientos de metros. Si tu caso es ese en que hay un departamento que gestiona todo eso, ponte de acuerdo con ellos, estudiad entre ambos como vais a asegurar vuestros CPDs y pedidles que os informen en todo momento de los cambios y novedades que hagan en este ambito, es muy importante que esteis coordinados ya que si pasa algo primero son las personas pero seguidamente recurriran a vosotros ya que el negocio ha de seguir y una mala coordinacion entre ambos puede dar al traste todo vuestro trabajo echo en contingencias informaticas.

En ese caso recuerdo que los tres primeros puntos ya los hemos comentado antes:

-> Extintores contra incendios.

-> Detectores de humo. Incluso en muchos casos anyadiria detectores de humedad para grandes CPDs.

-> Salidas de emergencia.

-> Equipos informaticos de respaldo.

Y volvemos a este punto. Ahora tenemos mucho mas presupuesto y no vamos a aranyarlo con portatiles o sobremesas, esto ya es otro mundo en tanto a planes de contingencias, necesitamos todo un centro 100% actualizado y preparado para acudir a el con la plantilla que direccion os haya senyalado y que puedan ponerse in situ a trabajar.

¿Dificil? Mas que dificil es costoso. Pero por el momento vamos a quedarnos con estos puntos de importancia:

* Seleccion de personal activo en contingencias: Esto lo decidira direccion o

en su defecto los responsables de cada departamento, quienes necesitan que sigan con su trabajo y quienes disfrutaran en su casa de unos dias festivos anyadidos :) Es un punto que te han de dejar muy claro desde el principio ya que en funcion de cuanto personal necesitan que siga trabajando asi sera el centro de contingencia. Puede que existan casos en que os digan que todos u otros casos en que los departamentos que generan ingresos iran al completo mientras que los administrativos llevaran a las personas mas importantes o de alto cargo de los mismos, eso ya es en funcion de cada empresa.

* Centro de contingencia: Sera en funcion de lo antes senyalado, aqui tendremos las maquinas, tantas como antes nos hayan pedido y por supuesto un CPD.

Pero sigamos adelante ya nos explayaremos mas en este punto, ahora solo quiero matizar que en este caso volvemos al tema del Ghost donde hemos podido, gracias a el, establecer un patron para cada departamento y establecer maquinas con el software que cada uno de ellos necesita, ya tenemos gran parte hecha.

Medida organizativas:

Igual que las tecnicas son igualmente aplicables las anteriores:

-> Seguro de incendios.

-> Precontrato de equipos en alquileres y ubicacion alternativa.
Acabamos de comentarlo.

-> Procedimiento de copia de respaldo.

Y esto es el punto mas importante a tratar. Hasta ahora hemos hablado de backups en cintas, discos y demas pero, pensemos un minuto, estamos hablando de tiempos de respuesta de minutos u horas, pero que no lleguen al dia completo sin estar operativos.
Si tratamos de hacer eso con backups sera imposible, habria que montar maquinas, restaurarlas y luego poner al dia todos esos cableados y softwares que lleve la empresa, no podemos, es imposible, hay que pensar en algo distinto.

Y ahora es cuando nos adentramos en el mundo de la virtualizacion :) necesitamos un centro cuyos servidores esten virtualizados y actualizandose casi en tiempo real con el centro de respaldo que hemos contratado ya unas lineas antes.

En este caso ya hablamos de servidores SAN o NAS dentro de los cuales tenemos el software de virtualizacion, aqui podemos debatir horas sobre cual es el mejor y cual es el peor, en mi caso cuando nos planteamos esta opcion decidimos decantarnos por el que actualmente es el lider indiscutible en virtualizacion VMWare con sus paquetes ESX. Los resultados que hasta el momento hemos obtenido son alucinantes, una gestion de los storage altisima, una usabilidad bestial yo sin duda creo que es el lider porque actualmente es la mejor opcion del mercado (lo siento por lo adeptos de Xen y de Virtual Center) si es bien cierto que se de grandes proyectos gubernamentales a lo largo del mundo que usan Xen pero VMWare una vez tengas el software es montarlo, montar los storage y a trabajar. Los famosos proyectos que vemos en las noticias de internet son proyectos de tamanyo monstruoso los cuales tienen equipos de personal informatico que programan un Xen perfectamente adaptado para ese entorno, yo estoy hablando de la realidad de una empresa normal donde no pueden hacer semejantes proyectos, a todos ellos les recomiendo VMWare porque es lo que necesitan en su entorno y los resultados que da son admirables.

Por cierto, NO, ni trabajo para Microsoft ni para VMWare, solo cuento sobre lo que realmente conozco y se que puedo recomendar no me atrevo a recomendar algo que ponen en Blogs de la red solo lo que puedo contar en primera persona, lo siento si alguien se siente molesto, no es mi intencion.

Bien ya tenemos nuestro entorno virtualizado ahora necesitamos otro software que sera el que interconecte nuestra red con la del centro de contingencia y nos haga las replicas del dominio y todas las maquinas virtualizadas en tiempo real.

Para dicho fin tambien hay dos grandes software, en este caso ya dejo a la libre eleccion de cada uno uno de ellos es Platespin, hasta hace unos anyos era el unico que existia y trabajaba muy estrechamente con la gente de VMWare ya que estos eran conocedores de que el uno necesitaba del otro, pero la virtualizacion es un negocio que se ha expandido tanto y ahora es tan usada que muchas empresas no han querido quedarse fuera, HP y otran han lanzado desde servidores preparados para esto, hasta software de gestion de consolas, storage y replicacion. He probado Platespin y lo he visto funcionar como para decir que funciona pero lo que mas he manejado y por el momento no tengo pega alguna de el es el software Datacore.

Bueno hagamos un resumen que nos estamos ya liando.

Centro de trabajo habitual:

- SAN o NAS con las maquinas virtualizadas.
- Servidor con el software de replica (Datacore).
- Usuarios con sus equipos de trabajo.

Centro de respaldo:

- SAN o NAS con las replicas de las maquinas virtualizadas.
- Servidor de software de gestion de replicacion (Datacore)
- Equipos preparados para que los usuarios puedan trabajar en ellos.

Eso es lo que tenemos que conseguir. Quiero matizar que montar una infraestructura asi es realmente complicado si no hay una persona que sepa hacer algo tan especifico, lo cual en la mayoria de empresas no hay, en esos casos lo mejor es contratar alguna que haga esto las cuales ahora mismo existen cientos. Empresas que unicamente se dedican ha hacer estas infraestructuras y posteriormente dan soporte. Ellos te crean la infraestructura y luego ya la virtualizacion de las maquinas reales, creacion de nuevas maquinas, gestion de los storage para cada una.... todo eso ya es cosa tuya :)

Bien hasta aqui ya tenemos creada una infraestructura que nos replica toda nuestra red en tiempo real, podemos dormir tranquilos. Nuestro centro se encuentra (esto es altamente recomendable y segun que tipo de negocio tenga la empresa es obligatorio) a mas de 25 Kilometros y si pasa algo grave alli esta todo preparado para funcionar. Hemos reducido el tiempo de respuesta a lo que nuestros usuarios tarden en llegar a la otra oficina.

Como ultimo apunte hemos dicho antes que los backups ya no nos valen, es cierto, como contingencia no nos sirven ya de nada pero en toda empresa existe un servidor de datos o de correo y nuestros usuarios muchas veces te pediran archivos o correos que ha podido borrar por accidente, existen metodos que vienen incluidos con los mismos sistemas de Microsoft para estos casos pero si os lo podeis permitir yo recomiendo un pequenyo backup cada dia aunque solo sirva para poderles recuperar ese archivo que necesitan de hace justo dos dias porque es cuando funcionaba la ecuacion bien. Creedme cuando os digo que eso os lo pediran casi a diario.

- Procedimiento de actuacion en caso de incendio.

- Contratacion de un servicio de auditoria de riesgas laborales.

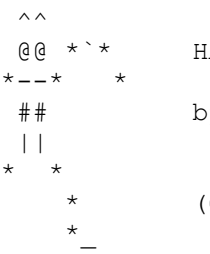
Hasta aqui los puntos, no nos vamos a olvidar de nuestras amigas las UPSs las cuales en estos casos deben ser bien potentes, pero, de nuevo si hay presupuesto, yo recomiendo poner un buen generador de energia. En esos grandes edificios no es dificil que la luz se vaya durante mas de 24 horas, adelantemonos a las electricas y si podeis poned estos generadores que con gasoleo os permitiran poder seguir trabajando con normalidad mientras ellos arreglan los problemas.

{==========}
{ CONCLUSIONES }
{==========}

Bueno hasta aqui hemos llegado, he tratado de hacer el documento lo mas ameno posible, por ello no he querido entrar demasiado en los temas de teoria que para eso teneis cientos de documentos en la red de muchas hojas cargadas de teorias de lo que deberia o no hacerse. En lugar de todo eso he tratado de poneros ejemplos que seguro os dan una mejor idea de que trata realmente un plan de contingencia, cuales son los riesgos reales y lo que puede pasar si lo descuidais.

Espero os haya dado buenas ideas y recordad que siempre es primero la persona y luego el negocio.

EOF



HACK THE WORLD

by blackngel <blackngel1@gmail.com>
<black@set-ezine.org>

---[1 - Prologo

Localizar una IP a lo largo del globo terraqueo para saber en que pais, ciudad, region, latitud o longitud se encuentra, no pasa de ser un mero juguete cuando se utiliza para buscar la direccion de alguien con el que chateamos y desconocemos su paradero. Pero cuando se implementa dentro del codigo de nuestro propio traceroute, entonces se convierte en un arma mucho mas poderosa.

En este articulo se ofreceran 3 aperitivos:

1 - El funcionamiento basico de GeoIP
2 - Un motor basico de traceroute (*)
3 - Exportar coordenadas a GoogleEarth

* Este motor ha sido extraido de un programa creado por mi y por lo tanto se expone aqui para que lo adapteis a vuestras necesidades.

---[2 - Introduccion

Primero la pregunta: que es GeoIP?

Pues son dos elementos:

1 - Una libreria
2 - Una base de datos

En realidad, GeoIP es un software de pago de la empresa MaxMind [1]. La suerte es que ofrecen al mismo tiempo una version libre denominada GeoLite City cuya unica diferencia es la cobertura. Aqui las estadisticas oficiales:

	GeoLite City	GeoIP City
Paises	99.3 %	99.8 %

```
|--------|---------------|------------ |
|Ciudades|     76 %      |    81 %     |
|_______|_____________|_____________|
```

Esta disponible como codigo abierto para los siguientes lenguajes:

```
C, PHP (Pure), Java, Perl (XS), Perl (Pure),
Apache, Python, MS COM, C#, Pascal, Ruby
```

Como aqui nos centraremos en el primero de los lenguajes, podras descargar las fuentes directamente desde aqui:

```
http://www.maxmind.com/download/geoip/api/c/GeoIP.tar.gz
```

Y la base de datos, que deberas instalar en "/usr/local/share/GeoIP/" deberia estar esperandote aqui:

```
http://www.maxmind.com/download/geoip/database/GeoLiteCity.dat.gz
```

Como trabajar con esta libreria es la cuestion pertinente a la siguiente seccion.

---[3 - Programar con GeoIP

Si tu objetivo es conocer todas las funciones que puede proporcionarte esta libreria, solo tienes que hacer una cosa muy facil, leerte los archivos de cabecera. Todas ellas tienen nombres intuitivos, sus parametros son evidentes y los valores que devuelven suelen almacenarse en estructuras bien definidas.

Aqui solo veremos la siguiente funcion:

```
GeoIPRecord * GeoIP_record_by_name(GeoIP *, const char *);
```

Las estructuras que observas se explicaran dentro de poco. Vulgarmente podriamos decir que el primer argumento es una base de datos abierta donde buscar toda la informacion del "host/ip" definido en el segundo.

Devuelve una estructura con toda la informacion que podamos necesitar acerca de su localizacion. Un valor 'NULL' en caso fallido.

Al fin y al cabo, todas las demas funciones devuelven diferentes valores segun sus argumentos. Algunas de ellas se encargan de devolver el codigo de un pais cuando este se le proporciona como argumento y viceversa. Otras son referentes a las ciudades, otras al codigo postal y asi...

Con la anterior lo obtenemos todo y de un solo golpe.

---[3.1 - Compilar

La cabecera principal para cualquier programa que desee utilizar la libreria GeoIp es: <GeoIP.h>. No obstante, nosotros utilizaremos en la directiva #include la cabecera <GeoIPCity.h> que incluye a si misma la anterior y nos facilitara el uso de funciones adicionales.

A la hora de la compilacion debemos linkar correctamente la libreria de esta forma:

```
$ gcc iplocate.c -lGeoIP -o iplocate
```

---[3.2 - El Codigo

GeoIP nos proporciona 2 estructuras basicas con las que interactuaremos en nuestro programa. Estas son:

```
   struct GeoIPTag
            |-> typedef: GeoIP

   struct GeoIPRecordTag
            |-> typedef: GeoIPRecord
```

Puedes consultar los archivos de cabecera para conocer todos y cada uno de sus campos. Aqui solo mostraremos los mas interesantes.

En la primera de las estructuras guardaremos el resultado de la apertura de la base de datos para poder consultarla a partir de ese momento tantas veces como queramos.
La segunda es un registro en el que se guardan los datos de una IP localizada. A nosotros nos interesaran los siguientes que son mas que intuitivos:

```
   char *country_name;
   char *city;
   float latitude;
   float longitude;
```

Podrian interesarte otros como:

```
   char *country_code;
   char *country_code3;
   char *region;
   char *postal_code;
   int dma_code;
   int area_code;
   int charset;
   char *continent_code;
```

Definiremos una variable global GeoIP para poder acceder a la base de datos desde cualquier funcion:

```
   GeoIP *gi;
```

Antes de empezar, todavia me gustaria hacer algo interesante y necesario. Crear una estructura global que contenga la informacion de un host y una variable que lleve la cuenta de cuantos hosts llevamos localizados y almacenados.
Sera util en posteriores secciones, cuando empiecen a encajar las piezas:

```
 **------------------------------------------------------------------------**

   struct host {
      char ip[16];            /* IP Address   */

      char country[25];       /* Country Location */
      char city[50];          /* City Location    */
      float lat;              /* Latitude         */
      float lon;              /* Longitude        */

   } geohost[256];

   int cnt_ghost = 0;

 **------------------------------------------------------------------------**
```

Veremos ahora como podemos abrir nuestra base de datos. Introduciremos la

sentencia en una funcion a parte. Por que? Simple, si este codigo formara parte de un programa mayor y necesitasemos realizar sucesivas busquedas, no resultaria nada aconsejable reabrir la misma base de datos continuamente.

```
 **------------------------------------------------------------------------**

   void load_geoip(void)
   {
      gi = GeoIP_open("/usr/local/share/GeoIP/GeoLiteCity.dat",
                      GEOIP_INDEX_CACHE);

      if (gi == NULL) {
         fprintf(stderr, "\nError: Don't open database\n");
         exit(1);
      }
   }

 **------------------------------------------------------------------------**
```

El primer argumento cae de cajon, el segundo no tanto, pero la explicacion no tardara. Podria ser una de estas opciones (extraido del README de GeoIP):

```
o--------------------------------------------------------------------------o
|GEOIP_STANDARD     -> Lee la base de datos desde el disco duro.           |
|                      RESULTADO: usa menos memoria.                       |
|                                                                          |
|GEOIP_MEMORY_CACHE -> Carga la base de datos en memoria.                  |
|                      RESULTADO: mejora el rendimiento pero usa mas memoria. |
|                                                                          |
|GEOIP_CHECK_CACHE  -> Comprueba si la base de datos ha sido actualizada,  |
|                      en tal caso la recarga.                             |
|                                                                          |
|GEOIP_INDEX_CACHE  -> Mantiene un indice de las busquedas mas recientes.  |
|                      RESULTADO: muy eficiente en bases de datos grandes  |
|                                                                          |
|GEOIP_MMAP_CACHE   -> Carga la base de datos en la memoria compartida.    |
o--------------------------------------------------------------------------o
```

Ahora viene lo que todos buscais, como conseguir la informacion referente a una direccion IP. Aqui el motor:

```
 **------------------------------------------------------------------------**

   void ip_location(char *host){

      int i, j;
      int find = 0;
      GeoIPRecord *gir;

      // Si alcanzamos el limite de hosts, salimos
      if (cnt_ghost == 256) {
         printf("\nToo many hosts to locate. Clean map first.\n");
         return;
      }

      // Si el HOST ya ha sido localizado, salimos
      for (i = 0; i < cnt_ghost; i++) {
         if (strcmp(geohost[i].ip, host) == 0)
            return;
      }

      // Buscamos el HOST en la base de datos
      if ((gir = GeoIP_record_by_name(gi, host)) == NULL) {
         return;
```

```
      }

      cnt_ghost += 1; // Hemos encontrado uno

      strncpy(geohost[cnt_ghost-1].ip, host, 15);
      strncpy(geohost[cnt_ghost-1].country, gir->country_name, 24);
      if (gir->city)
         strncpy(geohost[cnt_ghost-1].city, gir->city, 49);
      geohost[cnt_ghost-1].lat = gir->latitude;
      geohost[cnt_ghost-1].lon = gir->longitude;

      printf("\nIP localizada: %s", geohost[cnt_ghost-1].ip);
      printf("\nPais: %s", geohost[cnt_ghost-1].country);
      printf("\nCiudad: %s", geohost[cnt_ghost-1].city);
      printf("\nLatitud: %f", geohost[cnt_ghost-1].lat);
      printf("\nLongitud: %f", geohost[cnt_ghost-1].lon);
   }
```

Tus deberes son modificar la estructura "hosts" de la que deriva el array geohost[] para anyadir mas campos de la estructura GeoIPRecord y obtener asi mucha mas informacion acerca de una IP.

Utilizar estas funciones en un programa es algo trivial. Puede valerte algo como:

```
   int main(int argc, char *argv[])
   {
      if (argc < 2) {
         printf("\nUsage: ./iplocate x.x.x.x\n");
         exit(1);
      }

      load_geoip(); // Abrimos la base de datos

      ip_location(argv[1]); // Localizamos un host

      return 0; // Por algo devolvemos int
   }
```

Ya te habras dado cuenta de que con este programa solo podemos consultar la localizacion de un host y despues todo acaba. Con esto podrias suponer que la utilizacion de geohost[] es una perdida de memoria innecesaria.
Cuando leas las siguientes secciones cambiaras de opinion. De todas formas piensa que si integramos el codigo de las dos funciones principales en un programa interactivo, entonces el modo en que trabajamos se torna bastante eficiente.

NOTA: Si te gustan las listas enlazadas y te atreves con ellas, te animo a hacerlo. Puede ser un buen ejercicio de aprendizaje. Se vuelve util en el desarrollo de grandes aplicaciones donde establecer limites arbitrarios puede convertirse en nuestra perdicion.

Si necesitas mas ejemplos, el propio codigo fuente de GeoIP te los ofrece en la carpeta "test/".

---[4 - Traceroute

Aqui teneis un programa muy basico para trazar la ruta de un host.

Requisitos:

 - LibPcap (version 0.8) [2]
 - LibNet (version 1.1 o superior) [3]

Compilar con:

 - $ gcc codigo.c -lpcap -lnet -o broute

Uso:

 - $ ./broute x.x.x.x dev

 --

```
   #include <stdio.h>
   #include <string.h>
   #include <pcap.h>
   #include <libnet.h>

   #define FILTRO_ROUTE "icmp[0] = 0 or icmp[0] = 11"

   static u_int32_t src_ip;
   static u_int32_t dst_ip;
   static u_int16_t id;
   static int endsr = 0;
   static char *device;
   static libnet_t *lnet;

   static void
   read_response(u_char *useless, const struct pcap_pkthdr* pkthdr,
                 const u_char* pkt)
   {
      struct libnet_ethernet_hdr *ethh;
      struct libnet_ipv4_hdr *iph;
      struct libnet_icmpv4_hdr *icmph;

      ethh = (struct libnet_ethernet_hdr *)pkt;
      iph = (struct libnet_ipv4_hdr *)(pkt + LIBNET_ETH_H);
      icmph = (struct libnet_icmpv4_hdr *)(pkt + LIBNET_ETH_H + LIBNET_IPV4_H);

      /* AQUI LA RESPUESTA DEL OBJETIVO FINAL */
      if (iph->ip_src.s_addr == dst_ip) {

         /* ip_location(libnet_addr2name4(iph->ip_src.s_addr, 0)); */

         /* toGoogleEarth(); */

         printf("\nOBJETIVO ALCANZADO: %s\n",
                libnet_addr2name4(iph->ip_src.s_addr, 0));

         endsr = 1;
         pthread_exit(NULL);
      }

      /* AQUI LA RESPUESTA DE LOS HOSTS O ROUTERS INTERMEDIOS */
      else if (icmph->icmp_type == ICMP_TIMXCEED &&
               icmph->icmp_code == ICMP_TIMXCEED_INTRANS) {

         /* ip_location(libnet_addr2name4(iph->ip_src.s_addr, 0)); */
```

```
      printf("\nRespuesta desde: %s\n",
             libnet_addr2name4(iph->ip_src.s_addr, 0));
   }

}

void *
sniff_route(void *var)
{
   int rc;
   char errbuf[PCAP_ERRBUF_SIZE];
   struct bpf_program filter_code;
   bpf_u_int32 netp, maskp;
   pcap_t* sniff_route;

   sniff_route = pcap_open_live(device, 256, 0, 1000, errbuf);
   if (sniff_route == NULL){
      fprintf(stderr, "Error en pcap_open_live(): %s\n", errbuf);
      exit(-1);
   }

   if (pcap_lookupnet(device, &netp, &maskp, errbuf) == -1){
      fprintf(stderr, "Error en pcap_lookupnet(): %s\n", errbuf);
      exit(-1);
   }

   if (pcap_compile(sniff_route, &filter_code, FILTRO_ROUTE, 1, netp) == -1){
      fprintf(stderr, "Error en pcap_compile(): %s\n",
                      pcap_geterr(sniff_route));
      exit(-1);
   }

   if (pcap_setfilter(sniff_route, &filter_code) == -1){
      fprintf(stderr, "Error en pcap_setfilter(): %s\n",
                      pcap_geterr(sniff_route));
      exit(-1);
   }

   rc = pcap_loop(sniff_route, -1, &read_response, NULL);
}

int main(int argc, char *argv[])
{
   int c;
   int ttl = 1;
   u_int8_t *payload = NULL;
   u_int32_t payload_s = 0;
   libnet_ptag_t icmp_pkt = 0;
   libnet_ptag_t ip_pkt = 0;
   pthread_t th_sniffroute;

   if (argc < 3){
      fprintf(stderr, "\nUsage: ./broute x.x.x.x dev\n");
      exit(-1);
   }

   device = argv[2];

   lnet = libnet_init(LIBNET_RAW4, device, errbuf);
   if (lnet == NULL) {
      fprintf(stderr, "libnet_init failed: %s\n", errbuf);
      exit(-1);
   }
   src_ip = libnet_get_ipaddr4(lnet);
```

```
      pthread_create(&th_sniffroute, NULL, sniff_route, NULL);

      dst_ip = libnet_name2addr4(lnet, argv[1], LIBNET_RESOLVE);
      if (dst_ip == -1){
         fprintf(stderr, "IP de Destino erronea.\n");
         exit(-1);
      }

      sleep(1);
      while(endsr == 0){
         icmp_pkt = libnet_build_icmpv4_echo(ICMP_ECHO,
                                             0,
                                             0,
                                             id,
                                             0,
                                             payload,
                                             payload_s,
                                             lnet,
                                             icmp_pkt);
         if (icmp_pkt == -1){
            fprintf(stderr, "Error en Cabecera ICMP: %s\n",
                            libnet_geterror(lnet));
         }

    ip_pkt = libnet_build_ipv4(LIBNET_IPV4_H + LIBNET_ICMPV4_ECHO_H + payload_s,
                               0,
                               id,
                               0,
                               ttl,
                               IPPROTO_ICMP,
                               0,
                               src_ip,
                               dst_ip,
                               NULL,
                               0,
                               lnet,
                               ip_pkt);

         if (ip_pkt == -1){
            fprintf(stderr, "Error en Caberera IP: %s\n",
                            libnet_geterror(lnet));
            exit(-1);
         }

         c = libnet_write(lnet);
         usleep(1000);
         ttl += 1;
      }
      libnet_clear_packet(lnet);
      pthread_cancel(th_sniffroute);
      endsr = 0;
   }
```

```
 **-------------------------------------------------------------------------**
```

---[5 - Exportar a GoogleEarth

La siguiente funcion crea un archivo en formato xml (con extension .kml)
cuyas marcas seran interpretadas directamente en GoogleEarth estableciendo
puntos a lo largo del globo terraqueo, situando cada uno de los hosts que
haya ido localizando nuestra implementacion de GeoIP

Esta funcion debe conocer:

 1 - Una estructura en la que se hayan ido almacenando los hosts
     localizados mediante GeoIP.

 2 - Un contador con el total de las IP's encontradas (argumento).

```
  void toGoogleEarth(int cnt_ghost)
  {

     int i = 0;
     FILE *fge;

     fge = fopen("routeGE.kml", "w");

     fprintf(fge, "<?xml version=\"1.0\" encoding=\"UTF-8\"?>\n");
     fprintf(fge, "<kml xmlns=\"http://earth.google.com/kml/2.0\">\n");
     fprintf(fge, "<Folder>\n");
     fprintf(fge, "<name>Hack The World</name>\n");
     fprintf(fge, "<visibility>1</visibility>\n");

     for(i = 0; i < cnt_ghost; i++){
        fprintf(fge, "\n<Placemark>\n");
        fprintf(fge, " <name>%s</name>\n", geohost[i].ip);
        fprintf(fge, " <description><![CDATA[ IP:%s ]]></description>\n",
                     geohost[i].ip);
        fprintf(fge, " <description></description>\n");
        fprintf(fge, " <View>\n");
        fprintf(fge, " <longitude>%f</longitude>\n", geohost[i].lon);
        fprintf(fge, " <latitude>%f</latitude>\n", geohost[i].lat);
        fprintf(fge, " </View>\n");
        fprintf(fge, " <visibility>1</visibility>\n");
fprintf(fge, " <styleUrl>root://styleMaps#default?iconId=0x307 </styleUrl>\n");

  /*    ICONO PARA MARCAR LOS PUNTOS
        fprintf(fge, "<Style><icon>images/x.png</icon></Style>\n");
  */
        fprintf(fge, " <Point><coordinates>%f,%f,45</coordinates></Point>\n",
                     geohost[i].lon,geohost[i].lat);
        fprintf(fge, " </Placemark>\n");
     }

     fprintf(fge, "</Folder>\n");
     fprintf(fge, "</kml>");
     fclose(fge);

     printf("\nThe route has been printed to Google Earth format at \
\"routeGE.kml\" file\n");
  }
```

---[6 - Conclusion

Si tienes un poco de habilidad y has seguido hasta aqui todo lo descrito,
entonces estas en el camino correcto. Ahora estas preparado para unir las
piezas del puzzle y disfrutar de la satisfaccion que produce hacer las cosas
como "los hackers" mandan.

Si no conoces libNet, libPcap o si la programacion de threads (hilos) todavia esta fuera de tu alcance, entonces no te preocupes. Puedes centrarte unicamente en la seccion - 3 - y sacar el maximo provecho de lo alli expuesto; al fin y al cabo esa era la base del articulo.

En fin, algun@s ya saben, algun@s ya sabeis que... en fin... no podemos evitar ir un poco mas alla...

Siempre, siempre hay que ir un poco mas alla!!!

---[7 - Referencias

[1] MaxMind
    http://www.maxmind.com

[2] LibPcap
    http://sourceforge.net/projects/libpcap/

[3] LibNet
    http://libnet.sourceforge.net

EOF

PGP <http://www.pgpi.com>

Para los que utilizan comunicaciones seguras, aqui teneis las claves publicas de algunas de las personas que escriben en este vuestro ezine o que colaboran de una u otra forma.

<++> keys/grrl.asc

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: PGP 6.0.2

mQDNAzcEBEcAAAEGANGH6CWGRbnJz2tFxdngmteie/OF6UyVQijIY0w4LN0n7RQQ
TydWEQy+sy3ry4cSsW51pS7no3YvpWnqb135QJ+M11uLCyfPoBJZCcIAIQaWu7rH
PeCHckiAGZuCdKr0yVhIog2vxxjDK7Z0kp1h+tK1sJg2DY2PrSEJbrCbn1PRqqka
CZsXITcAcJQei55GZpRX/afn5sPqMUslOID00cW2BGGSjtihplxySDYbLwerP2mH
u01FBI/frDeskMiBjQAFEbQjR2FycnVsbyEgPGdhcnJ1bG9AZXh0ZXJtaW5hdG9y
Lm5ldD6JANUDBRA3BARH36w3rJDIgY0BAb5OBf91+aeDUkxauMoBTDVwpBivrrJ/
Y7tfiCXa7neZf9IUax64E+IaJCRbjoUH4XrPLNIkTapIapo/3JQngGQjgXK+n5pC
lKr1j6Ql+oQeIfBo5ISnNympJMm4gzjnKAX5vMOTSW5bQZHUSG+K8Yi5HcXPQkeS
YQfp2G1BK88LCmkSggeYklthABoYsN/ezzzPbZ7/JtC9qPK4O7Xmjpm//ni2E10V
GSGkrcnDf/SoAVdedn5xzUhHYsiQLEEnmEijwMs=
=iEkw
-----END PGP PUBLIC KEY BLOCK-----
<-->
```

```
Tipo Bits/Clave    Fecha      Identificador
pub   768/AEF6AC95 1999/04/11 madfran <madfran@nym.alias.net>

-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: 2.6.3ia

mQBtAzcQ8VIAAAEDAJuWBxdOxP81fhTJ29fVJ0NK/63dcn5D/vO+6EY0EHGHC42i
RF9gXnPUoSrlNfnfFnF9hZO0Ndb4ihX9RLaCrul8+FN97WYCqSonu2B23PpX7U0j
uSPFFqrNg0vDrvaslQAFEbQfbWFkZnJhbiA8bWFkZnJhbkBueW0uYWxpYXMubmV0
PokAdQMFEDcQ8VPNg0vDrvaslQEBHP0C/iX/mj59UX1uJlVmOZlqS4I6C4MtAwh3
7Dh5cSHY0N0WBRzSBKZD/O7rV0amh1iKkrZ827W6ncqXtzHOsQZfo183ivHOc3vM
N4q3EEzGJb9xseqQGA61Ap8R8rO37Q8kEQ==
=vagE
-----END PGP PUBLIC KEY BLOCK-----
```

blackngel

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v1.4.6 (GNU/Linux)

mQGiBEh9Mg4RBACTEsX6OVE4nVnHYELc+bsBTgcEs9/nWmETA+eGeySrx9qfJtxZ
NsBEn3HZlb1ioOBCzJc7Y+YAYeU45t4+pgFVfLUHfv/eL+6nUAii5rnmU7t0dfme
2WQRuiLbLhpdh33A4WfiXFMX0rR0yLFUETRYxx2lvkhV7nhzk0Rfp8Ob+wCg6YwN
UHhDiX7W6ZcGJTmd83t3nOUD/35ZWjorQllM4sHXp0Nt5C/EBzPaAhVA0ZlnBxoN
/7BxaPBQ3kbBVG8PcL2kiQC9Q7RsGMtddI1OBQihhxSbPNw+SMs/W4g2mfH/toO3
9e4PY+5eR8/l+/Yh+JuTFsiTH2fjft5e2CdDHWzwaUBysD7KdsI5wbCzGj8940Ls
izO9A/9OIJCn0mh0L2+W9X+AEU17en7uXoQuIpzmZqbnp0KOD1grsvNJTrruoOIS
0bZ2xNshelfCWYfrLUq78on7rrEDgi13KPKBkDnRjLMeYTdBjvYOiqTbx9uIyVxN
WysmpFpx7QuJyDk1y/R9Mu3RHMFL9sbJEwawaTDnxyMxHH7Zd7Q2YmxhY2tuZ2Vs
IGJlc28gKGhhY2sgdGhlIHdvcmxkKSA8YmxhY2tuZ2VsMUBnbWFpbC5jb20+iGAE
ExECACAFAkh9Mg4CGwMGCwkIBwMCBBUCCAMEFgIDAQIeAQIXgAAKCRBzF1yE6bll
DhGsAJwLkNbFzveTRcIMiVlDZfYkpw/BfACfd0cOWdbcxCVSuwGdqR2NWT6N6CK5
```

```
BA0ESH0y5RAQAPVhodzBKqfnN6PjGMiT91olyMeFnAutZvOr8bZFb3CiL8gbwnci
dPtrgFBoydtJ+1t2Dk7Ilqny+gxbCemu4PMPFfnZkDzZEMvUML85sNdTxqGbXsuf
DOjCng2zntWqB8t+LjtLgLZbzED3uOxbtJ6W8Kq3a87GAcg+SpDXwqR+ykDTa9kc
GytwsN7ICM7OsXHSzOwKhtZ5UpgNWKBxBaPO1BEv8AnY25ePN4ddgwqoixhhKQer
FoeHsuDu5i0Zj8zgJUD2hutrr7QIz55oRk0NUXx0ZTD75ofUArJuLQl2PJMZBcwB
rP9IMsGSTiNJJMICjtRzuLDrPN3gNRMqfs3fZBmD7WfpL2jjuHCRzZLUXiLk3Veq
3n3bTvckcphYhlo/8tM3apANUYV4j0yJYK9MqwPb++YZdniCe1KKthxewJuxZrER
ZPAhCtoZqpIv4jRO9aPC1o19BAXMSduFnBWfw8K1dO4PaCuRQkLsvDBud3NuFpUy
j6Uia8zAGi5to5rShyp0hPinoszKU714MmJJrVGKCa/vu5aTQrS+ucQflhnvdPDv
/U4IpMeM4sjrxEl5NYznKQxCr65qaxZYw9sJ/Ovchfh8Pml4fuNAEhk8ghGPlwCC
JaMTUwYY0Sj50RFXOg7c8ooIqOLmgna5nEug+EpBaVeyDYLIdy9tSXtrAAMFD/42
4D5/0hu16rp3kxA7CcrY7rAgapmD5zwB6WqbTkZ+j//2PsW70ZCtYymVV+fLGVXC
I982rvFfr0X7+mV5DZSwPLKnhAUH4TQiTc6jajt8WxqchEZ4rxG7OmcSSoEqbEBu
m3LhQllmko/P2n5SKQzQopGU/xCiDvrLSrRcqwtPpj7QiuYTMfkfWlxYZ63k4sev
4ifGXZcnJZScgKebxtkUUy3fP+XUFTC92mT1m6o5ElFhS0zQOt31p4K3lgj4kkd/
11/b7pcTiR1jG13PUIXyfWFpFEl4avB0mKBkgrinMQWvzMssiiUknu1LYeAYfT0D
dvnx0mANqn2XaQzErBSQwMb6s1xOujXC/FTYqLla9euqX1T+tEUioOPF9f4y8JN+
kqihqm9j+Z5NTPtHGq5vn1lXU0/bFjFXAdMH07OIqNSB6+tLd4MSwcX8C2eRX/bD
8hBK63aomzDC6YOGdvirWOwmwKsy748PzRZJq8blx2YiMhtizCo1kCbVs5anSuFg
cb6KlnkEhv1FxIk0ofOwIH/Zz9+3pIPe4jWhPcYY0hYA5m3kvvXVFETUBgdb8I8z
TSxBUAsbc+HcZdHNqw8sMV9yUbCWFocVav3cokKskM1DNjebfzNYGo1u50B7CfV6
VMaknglUJTKVfWGPGkT3C++g5OjI3fU0YPUl2t7Py4hJBBgRAgAJBQJIfTLlAhsM
AAoJEHMXXITpuWUOj7gAn2jsDvioxYAS4Iui5cbs8lkP9P81AKC+UdxGIthecpHx
g5spYWJrMWe1WA==
=KoEY
-----END PGP PUBLIC KEY BLOCK-----
```

kstor

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: PGPfreeware 7.0.3 for non-commercial use <http://www.pgp.com>

mQGiBD926+4RBACZI12ENJqFXFvw+7PWm5P2mFBWHrHP3x7MGOT0XcTV3h9/JmTf
tPMWk9q5Vpv+1UuZrLUgg9v75hD3YSGx/GdAPINjxwJxdqcgOxs6goGHM6q084kx
Oo6wRf2/E4uWXih9yqVwDly6g3Wx5bfvor+8qJGqEY9kQeNFsaeko1f+owCguWRd
6JLhCmRaqNS2XHh8J+yXjHUEAIddWuKpMA2l4v32FFhFPkORYtKF4hpCqP8eq35d
21fHUT0OXRC3+XptD1wc8IHoGMnhi8D6llRdHS7ZIWcUUgtfFtAX2BNT1crs8X4l
Xt/AjzGWPd+ITEqIis89Uc2f8RiJqgDX5ZeL610poWLp4CqsdExxRixOasYdKRRr
978BA/9jdXL/mjheQUMV19IYB4nPEXJ706qjqopA15U5Lgn7Ndp+ATi9A73wAcJN
2l7qSa4hDKHAJ3mvnoRuwcBhDX/EU4FNUd19rLG6NlGCG75e0wLSSrzfTp1k8Et6
Jb9UGRnTZLTANczEgrg368fhqC6GNDP0GP1Hqi2NQx+wnGSMW7QtTWFydGluIERp
IEx1emlvIChLU1RPUikgPGVrc3RvckB5YWhvby5jb20uYXI+iFkEExECABkFAj92
6+4ECwcDAgMVAgMDFgIBAh4BAheAAAoJED5cYxg2MpymJfoAniiv+zYKPuh3tgsm
M16kKIDMIfYDAJ926peylf68fK05XkP/OguJTXllabkBDQQ/duvyEAQAj6ACZkrh
+qKpBpJIDqNAntbpPgp8onMv0j7hEzLROSsjc3VuD3AxZADM9lrPcXM4t8M8DCCq
vcGS2rZbukKf9fQsn4NKnJlhqery6cNhEcQolrzi2D2f/PqAr5TxzgsFGqpLMeON
/g2V+iSrB1oq09CgiMCio7QqDYG/wgBZVEsAAwYD+wW0ARzU7meHe/Gg9JYp2hzn
lb9ieE/L7xQ5gfIRhIqnfJjFqmyzZkhlt/C+wFIq3G//TYM6STwkmRfUZ/0ZdLo+
406yrLiP+FBIEMm/WIzyiMWH6YxhUz9PN6HhcFJna50y4CRTQ5fFoksCaFd1hquQ
PZes1LI4MTYJ+cWaSpOWiEYEGBECAAYFAj926/IACgkQPlxjGDYynKZBzQCeJMjQ
1izVC11nVdN/Yz6nDs82CGwAn2V7opxfIynjKBxggv0/e88WJJSh
=FYUn
-----END PGP PUBLIC KEY BLOCK-----
```

elotro

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: PGPfreeware 5.0i for non-commercial use
```

```
mQCNAzr1734AAAEEAMKrsCLyeUS4ouBjltE/7ubE1i58tZem7xPVy5Vot9uW5rOA
OyZNM0zdlgEvW1xdxmsBdojLrkqEk8ZQXDx5zCn0wE/8CHhP3dewEIcYpcBv1/0a
wbxpG7r2c5AajGViceLtEVcT6p65ZnKW2c7DMH/GEbOtPaG6fSIPE8Z4w3KXAAUR
tBxlbG90cm8gPGVsb3Ryby5hckBnbWFpbC5jb20+iQCVAwUQOvXvfiIPE8Z4w3KX
AQEDwgQAtwrBv3To4QnN67jeNZSxjoZC2gAb7Yq4gueP20yfARRlKOSompGgwypI
Oy/qhgTxtdDkjdtvRk16cx8jjhgyXfSLOhJ787+IGmrxT/jWwxSMmuRNWmHBcavD
wQzLlpxEQBwdL/guZBNsMSMQr9FpBRkPDQSPGQC18OnGKNJMXf0=
=hgJM
-----END PGP PUBLIC KEY BLOCK-----
```

```
   ---------[ ULTIMA ]------------- ------------------------------------
  |                                |                                    |
   ---[ ULTIMA NOTA ]--------------                                     |
  |                                                                     |
  |  Derechos de lectura:                                               |
  |     (*)Libres                                                       |
  |                                                                     |
  |  Derechos de modificacion:                                          |
  |     Reservados                                                      |
  |                                                                     |
  |  Derechos de publicacion:                                           |
  |     Contactar con SET antes de utilizar material publicado en SET   |
  |                                                                     |
  | (*)Excepto personas que pretendan usarlo para empapelarnos, para    |
  | ellos 2505'34 Euros, que deberan ser ingresados previamente la cuenta|
  | corriente de SET, Si usted tiene dudas, tanto para empapelarnos o   |
  | de como pagar el importe, pongase en contacto con SET atraves de las |
  | direcciones a tal efecto habilitadas.                               |
   ---------------------------------------------------------------------
```

SET, - Saqueadores Edicion Tecnica -. Numero #35

EOF