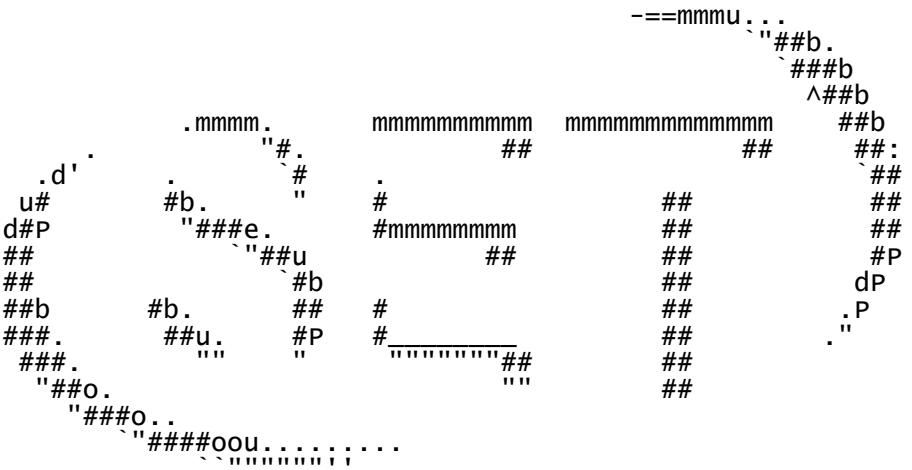


Saqueadores Edicion Tecnica

INFORMACION LIBRE PARA GENTE LIBRE

SET #33 - Octubre de 2006

EDITORIAL

SET Ezine

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

Mirrors:
http://salteadores.tsx.org
http://www.zine-store.com.ar
http://www.hackemate.com.ar/ezines/set/
(¡con version online!, desgraciadamente no muy actualizado)

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 33]----

"Los ordenadores del futuro no pesaran más de 1,5 toneladas"
POPULAR MECHANICS (adelantando el vertiginoso avance científico), 1949.

EOF

-[0x01]--
-[Editorial]---
-[by SET Staff]---SET-33--

Parece que la cosa se enfria ¨no?, ultimamente tengo la sensacion que todo este mundo lleno de curiosidad y de curiosos se enfria poco a poco, La cosa ya no es como antes -y no quiero ponerme en tono paternal cuando tus padres te cuentan sus batallitas- pero sinceramente me parece una realidad, ahora existen muchos tecnicos de seguridad, formación reglada.... tengo la sensacion, que poco a poco la gente se va "jubilando" y van quedando poca gente, hace mucho que no se nada de mucha gente que era realmente buena, ni de muchos grandes proyectos que en su dia se estaban preparando.

Muchas veces pienso sobre esto, y sinceramente, creo que simple y llanamente, el hacking ya no esta de moda... enterrado entre las tendencias urbanas de hoy en dia, esperando a que de nuevo alguien lo vuelva a poner de moda, solo así me explico, la tranquilidad que se respira en este mundillo, que, a decir verdad, no soy nadie para tirar la primera piedra, nosotros tambien estamos tranquilos, pero ahi os dejo eso...

****

Bien. Aqui hay varios que editan y no todos tienen el mismo punto de vista. Es cierto que el viejo hacking ha cambiado, pero es que tambien han cambiado las circunstancias. Antes se podia entrar en un sevidor importante y lo maximo que te pasaba era que te borraban la web donde habias acumulado algunas majaderias. Poco a poco las cosas han cambiado y ahora puedes recibir la visita de la policia, no precisamente para desearte un buen dia, por bajarte esas peliculas que tanto te gustan.

Es cierto que el dinero ha introducido sus negras manos en este mundillo. Estoy seguro que existen buenos hackers, el problema que estan buscando la forma de financiar la hipoteca que han contraido y no se les ha ocurrido otra cosa que vender sus servicios al mejor postor. La avalancha de spam que todos recibimos es una de las consecuencias.

Yo soy optimista. Todo tiene sus ciclos y despues del invierno viene la primavera.

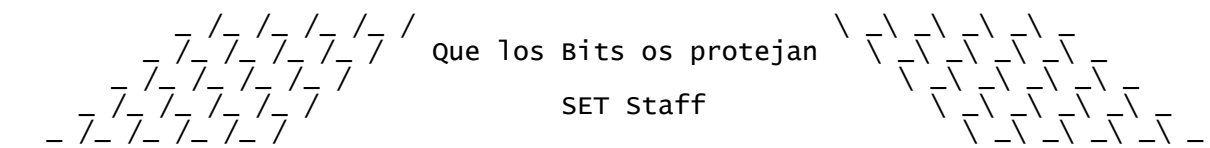
****

Yo tampoco lo esperaba ;-) pero hemos vuelto a parir otro numero mas... lleno de cosas para que todos los criticos de SET os podais divertir durante un tiempo y lleno de cosas para los que gustan de la curiosidad puedan saciarla aunque sea un poco...

Os dejo ya de contar batallitas de abuelo y os dejo con SET 33...

Hasta el proximo numero,

El editor



EOF

auto_crack

En este articulo se cuenta una metodologia para aumentar la eficiencia en el proceso de averiguacion adecuada de modificaciones de programas.

Vamos, que te ensenyo un metodo rapido para crackear !

Se explica como se puede aplicar a varios tipos de programas y de procesadores.
Durante el articulo se repetiran varias veces los mismos conceptos para que quede claro que se pueden desarrollar variaciones sobre el mismo tema.

Intento no poner muchos listados en ensamblador porque reconozco que son pesados de seguir y considero que lo importante en este caso es la teoria.

El principio basico del crackeo es modificar un programa para que, en un sitio concreto, haga algo distinto a lo que el programador ha disenyado originalmente.

Normalmente el proposito es evitar la proteccion de copyright, y su segundo uso es conseguir acceso a funcionalidad oculta, de uso interno.
Una utilidad mas inocente es conseguir mayor poder en un juego: mas vidas, mas potencia de armas, mas dinero, ...

La metodologia tipica es ejecutar el programa, hacer que llegue hasta la rutina que muestra un resultado "indeseable" e intentar seguir inversamente el programa hasta averiguar porque ha llegado hasta alli, en vez de la rutina "deseable".
Por ejemplo, si el programa pide una clave de acceso, se pone un punto de interrupcion en la rutina que muestra el mensaje "Clave erronea" o similar.
Entonces se busca si hay otro mensaje "Clave aceptada" y se intenta averiguar porque ha llegado a un sitio, y no al otro.
Usualmente habra una rutina que calcula la validez de la clave, que devolvera el valor "cierto" o "falso". Entonces el crackeador modifica esta rutina para que siempre devuelva "cierto".

Por si no lo sabias, la ley prohibe el uso de tales tecnicas, debido al perjuicio que le supone al fabricante del programa.
Y yo, como programador, estoy de acuerdo con ello. La pirateria es un delito y debe ser castigado. No uses nunca programas ilegales.

Por otro lado, yo argumento una justificacion banal para la modificacion de programas, sobre todo juegos: a veces son tan dificiles que no le sacas todo el partido al dinero que has pagado.
Es por esto que no tengo ningun reparo en alterar juegos para conseguir mas facilidad de uso, por ejemplo consiguiendo vidas infinitas.
La teoria es la misma: supongamos que cuando pierdes una vida, el programa muestra un mensaje. Entonces se pone un punto de interrupcion en dicho mensaje; cuando se llega a este punto, buscar la variable que contiene el valor del numero de vidas disponibles, y evitar que se decremente.

Esta tarea puede ser simple o extremadamente complicada.
Frecuentemente el proceso consiste en 2 etapas:
1.1-desensamblar el programa
1.2-poner puntos de interrupcion

y luego:
2.1-parchear el programa
2.2-ejecutarlo
2.3-jugar hasta que se pierde una vida
2.4-si efectivamente se decrementa, volver al punto 2.1

La parte segunda puede necesitar muchas pruebas, y algo de suerte.

El metodo propuesto por mi consiste en automatizar 2.1 y, si es posible, 2.2-2.4

Supongamos que al principio del juego disponemos de 3 vidas. Cuando perdemos una, este valor se decrementa hasta 2, obviamente.

Ahora se trata de capturar periodicamente toda la memoria usada por el juego
y compararla con la copia tomada anteriormente. Si un valor ha pasado del
valor 3 al valor 2, sabemos que en medio se ha ejecutado la rutina
decrementadora.

El metodo depende del sistema debuggeado.

Para empezar usare el programa calc.exe de la calculadora en Windows-NT , y
el debugger OllyDbg de Oleh Yuschuk en:
http://home.t-online.de/home/OllyDbg

La calculadora tiene 4 modos de valores: Hexadecimal, Decimal, Octal,
y Binario.
Por defecto arranca en Decimal, pero a mi me gustaria que arrancara en Hex.
Si, ya se que lo puedo hacer pulsando F5, pero prefiero ahorrarme este paso.
Y el objetivo es ensenyaros, leches !

Asi que pongo en marcha el debugger, y arranco la calculadora.
El segmento de datos esta cargado en la direccion 01014000 asi que le digo
a OllyDbg que haga una copia: Backup->Save data to file
y la llamo calc_01014000_dec.mem

Sigo ejecutando el programa, y ahora cambio el modo a Hexadecimal. En este
momento hago una segunda copia.
La llamo calc_01014000_hex.mem

Comparo los dos archivos. En total hay 73 diferencias.

Por otro lado, desensamblo el programa y veo que el menu "Hexadecimal" tiene
valor ID=0x0132, mientras que el de "Decimal" tiene ID=0x0133.

Asi, reinicio OllyDbg y cargo calc.exe , aunque sin iniciarlo.
Ahora, le digo a OllyDbg que se detenga cuando una direccion cualquiera
de memoria pase del valor 0x0133 a 0x0132.
Inicio la calculadora (modo Decimal), y pulso F5 para que pase a modo Hex.
Habilmente OllyDbg se detiene en loc_1002ECB . Compruebo que cualquier
cambio Hex<->Dec<->Octal aterriza en esta rutina.
Hago otra copia de la memoria y veo que tambien han cambiado las direcciones

0000001C: 0A -> 10
00000088: 0A -> 10
00000298: 0A -> 10

Con lo que deduzco que hay una variable interna (que esta en 0x0298) que
sabe cual es el modo.

Los distintos modos son:

modo	tecla	ID de menu	ID de modo
Hex	F5	[ID=0132h]	10
Decimal	F6	[ID=0133h]	0A
Octal	F7	[ID=0134h]	08
Binary	F8	[ID=0135h]	02

Fijaros que el modo indica exactamente el numero de bits usados en el
calculo: Binary=2 , Decimal=0xA=10d, ...

Hasta aqui, el metodo tradicional de tracear.

Recordar que el programa original arranca en Decimal, por lo que F7 pasa
a Octal, y aterriza en loc_1002ECB . En este momento la direccion 0x0298
contiene el valor antiguo: 0x0A, que significa "modo Decimal".
Al finalizar esta rutina (en 01004299) la direccion 0x0298 contiene el
valor nuevo: 0x08, que significa "modo Octal". Por ahora es consistente.

Ahora es cuando intento que calc.exe entre directamente en modo Hexadecimal.
Para ello pongo en marcha mi metodo. Recordar que no estoy interesado en
saber donde tengo que parchear, sino que lo que intento es que se crackee
automaticamente.

Para ello parcheo cada byte (bueno, no todos; solo los que son sospechosos)
y ejecuto el programa de nuevo.
?Que define que un byte es sospechoso? Pues que contiene el valor 0x0A .
Si, en el fondo este es un ataque de fuerza bruta.

El byte 0x0A aparece 132 veces dentro de calc.exe
Hago un programa que:
-tome el fichero original calc.exe
-busca 0x0A (significa "modo Decimal")
-lo cambia por 0x10 (significa "modo Hexadecimal")
-almacena la direccion que ha sido parcheada
-inicia OllyDbg, que carga calc.exe
-pone un breakpoint en loc_1002ECB (significa cambio de un modo hacia otro)
-comienza la ejecucion
-simula la pulsacion de F7 para pasar a modo Octal
-si, tras 2 segundos, se detiene en loc_1002ECB y 0x0298 contiene 0x0A (valor Decimal), entonces significa que el valor al inicio es "Decimal", con lo que el parche no ha hecho nada
 -En este caso detiene el programa y vuelve al paso primero, parcheando una direccion distinta
-si 0x0298 contiene 0x10, entonces es que el parche ha funcionado.

Con lo que no habia contado es que alguno de dichos "0x0A" modificados por "0x10" son instrucciones del procesador, por lo que el programa parcheado a veces es in-ejecutable y peta. Bueno, no pasa nada. Simplemente esta instruccion no es la que hay que parchear.

En este caso he tenido suerte: tras 3 ejecuciones, el parche funciona !
Para los que quieren saberlo todo: la rutina adecuada es
01002A7   push 0Ah

Por supuesto este metodo tiene varios inconvenientes: si el programa tiene un chechsum, el cambio puede que provoque que el program no se pueda cargar. Aunque, visto recursivamente, es posible usar esta misma tecnica para eliminar la rutina de chequeo, por supuesto.

Si el programa esta comprimido, es imposible parchearlo con mi programa parcheador, pero OllyDbg tiene una funcionalidad que permite esperar a que el programa esta descomprimido en memoria, y puede hacer un programa ejecutable a partir de la memoria.

El programa parcheador debo hacerlo en un lenguaje de programacion que permita:
-buscar datos en un fichero
-modificarlos
-iniciar otro programa (el debugger)
-simular pulsacion de teclas
-esperar
-"entender" lo que el otro programa muestra.
-detener el programa ejecutado

Tanto en lenguaje C/C++ como VisualBasic es posible hacer los 5 primeros pasos.
Lo que es mas dificil es que entiendan lo que esta mostrando el otro programa.
La unica solucion que se me ocurre es tomar capturas de pantalla, y ver si ciertos puntos estan pintados de un color u otro.

Pero hay otra herramienta mejor: un testeador de aplicaciones. Es un programa que simula teclas y el movimiento del raton, y es capaz de saber si un control (checkbox, textLine, dialogo, menu, ...) esta activo o no, y su valor.

Yo he trabajado con Rational Robot Test Manager. Una herramienta realmente potente con gran versatilidad.

Arranco OllyDbg y pongo los puntos de interrupcion necesarios en loc_1002ECB.
Cuando lo cargue de nuevo, los recordara.
Tambien abro la ventana "Watch expressions" y hago que me diga el valor en la direccion 0x0298 .

Este es el programa, excepto algunas lineas de inicializacion de variables y las validaciones :

```
fin=false
intentos=0
do
 File.Copy("calc.exe", "calc_parcheado.exe")
 parcheado = File.Open("calc_parcheado.exe")
```

```
 for saltados = 0 to intentos
  parcheado.Search(0x0A) // salta las N primeras occurencias de 0x0A
 next
 parcheado.Replace(0x10) // modifica la N+1
 parcheado.Close
 intentos=intentos+1
 if intentos>=132 then fin=true  // solo aparece 132 veces el byte 0x0A
 olly = Process.Execute("OllyDbg calc_parcheado.exe")
 olly.SendKey(F9)   // OllyDbg arranca el programa debuggeado
 calc = Process.FindWithName("calc_parcheado.exe") // poner foco en la calculadora
 calc.SendKey(F7)   // pasa a modo Octal
 Wait(2)          // espero un poquito. Confio en que haya activado el breakpoint
 encontrado_10 = olly.Window("Watch expressions").FindString("[0x0298]=0x10")
 if encontrado_10 = true then   // exito! Mostrar un mensaje, y detenerse aqui
   Message("El valor de 0x0298 contiene 0x10. Parche encontrado !")
   fin=true
 end if
 encontrado_0A = olly.Window("Watch expressions").FindString("[0x0298]=0x0A")
 if encontrado_0A = true then continue // mala suerte. Estamos como antes.
                                       // Seguir con otro intento
 if encontrado_0A = false then continue // vaya: no encuentra ni 0x10 ni 0x0A
                                        // Probablemente ha petado
 olly.Close    // termina el proceso, y seguimos intentando
while (NOT fin)
Message("Parche NO encontrado :-( ")
```

Facil, ?eh? Y muy potente.

----------S

Esto vale para cualquier sistema que incluya un debugger paso a paso, o algo similar, como un traceador de rutinas.

Casi desde el principio de la informatica, los sistemas han sido cada vez mas potentes, pero el software estaba escrito para procesadores menos versatiles. Por eso era frecuente que tuvieran que emular otros sistemas "menores". Con el paso del tiempo, se consiguio una gran cantidad de sistemas que emulaban otros. Mi favorito es el emulador de ZX-Spectrum para PC, que me permite jugar a todos esos juegos desarrollados hace 20 anyos pero igual de adictivos ahora que antes.

Lo bueno de estos sistemas es que el codigo fuente suele estar disponible, y es relativamente facil de adaptar a otras necesidades.

Tomemos el caso del juego WestBank, hecho por Dinamic. El juego consiste en que eres el vigilante de un banco en el antiguo oeste. Hay 3 puertas, y en cualquier momento puede entrar un bandido o un cliente. A los bandidos se les dispara, y a los clientes se les deja entrar para que ingresen su dinero. Si te equivocas, pierdes una vida.
El juego es muy simple, pero yo no consigo pasar la tercera fase: los bandidos me disparan a mi demasiado rapido, y pierdo las 3 vidas enseguida.

El programa ocupa 40 Kb, de los cuales el 70% son graficos. El resto es codigo en ensamblador del Z-80, que IDA es capaz de desensamblar.
No solo eso, sino que existen multiples emuladores. Yo he elegido el ASpectrum, programado por Santiago Romero.
Lo he elegido porque el codigo fuente esta disponible y me resulta facil de entender.
Desde aqui doy las gracias a su autor y te invito a echarle un vistazo:
http://aspectrum.sourceforge.net

En un Z80 existen 8 registros de 8 bits: A, B, C, D, E, F.
Hay tambien otros registros de 16 bits acceso indexado (HL), para la pila (SP), las interrupciones (I/R), la direccion de ejecucion (PC).
A partir de aqui el emulador lee las instrucciones, y hace lo que le dicen.
Por ejemplo,
LD A, C
hace que A reciba el valor de C , asi que el emulador (hecho en lenguaje C)
hace int8 A, B, C, D, E, F;
.....
A=C;

Para simular la memoria, usa un area de 48 Kb paginada. Para no complicar la explicacion, supongamos que se llama:
int8 Z80mem[48*1024];

Para poner un valor en una direccion de memoria el Z-80 usa
LD (HL), A
por lo que el emulador hace
Z80mem[HL]=A;

Y eso si, el emulador tiene un bucle que decodifica cada instruccion, y la ejecuta. Asi continuamente.
Este bucle esta en aspectrum-0.1.8/main.c y llama a Z80Run , que usa switch (opcode)
para identificar cada instruccion que debe ejecutar.

Manos a la obra: inicio el emulador, y cargo el WestBank. Pongo el emulador a velocidad del 10% para poder controlar mejor los tiempos.
Empiezo teniendo 3 vidas. Justo antes de perder 1 vida guardo el juego, y otra vez tras perderla.
Comparo los ficheros y veo que el dato 0x03 pasa a ser 0x02 aproximadamente en 7 direcciones distintas. Parece que se guarda en varias variables, quizas algunas de ellas son solo temporales.

Pero no se la instruccion exacta que lo decrementa. Suponer que el numero de vidas (3) esta almacenado en el registro A. Entonces hay varias posibilidades:

```
------- opcion 1: comparando cada valor --------
CP A, 3
JR Z pon_2
CP A, 2
JR Z pon_1
; obligatoriamente A=1. Tras decrementar, vale 0, es decir, fin de la partida
JR fin_patrida
pon2:
LD A, 2
JR sal_rutina
pon_1:
LD A, 1
JR sal_rutina


------- opcion 2: decrementando --------
SUB A, 1
CP A, 0
JR Z fin_partida
JR sal_rutina

------- opcion 3: usando una tabla indirecta --------
LD HL, tabla
ADD HL, A
LD A, (HL)
JR sal_rutina
tabla: DB 3, 2, 1
```

Por supuesto existen variaciones:
-el registro puede ser B, C,...
-puede usar directamente una direccion de memoria: SUB (HL), 1
-puede que el valor este multiplicado por 10, o por 100 ...
-puede ser que use RET en lugar de saltos relativos
-quizas incremente en vez de decrementar. Cuando llega a 3, salta a fin_partida

En fin, las posibilidades son infinitas. Lo unico de lo que estoy seguro es que se guarda en alguna direccion de memoria.

Para usar el crackeador automatico sigo este procedimiento:
-no toco ninguna tecla durante el juego. Tarde o temprano aparecera un bandido que me matara.
-cuando tengo 3 vidas, hace una copia de la memoria: mem3
-cuando tengo 2, hago otra copia: mem2
-cuando solo me queda 1, hago otra copia: mem1

Dado que estoy usando un emulador, puedo reproducir la partida cuantas veces quiera: no existe el factor de aleatoriedad.
Llamo t3, t2, y t1 los momentos en los que he hecho las copias de memoria.

Por supuesto t3<t2<t1
Denoto tI=0 al momento en el que el emulador inicia el programa, y por tanto ha ejecutado 0 instrucciones.
Entonces t3 es el momento en el que ha ejecutado i3 instrucciones.
Analogamente t2 es el momento en el que ha ejecutado i2 instrucciones.
Lo mismo para i1.

Claro que i3<i2<i1

Visto desde el punto de vista inverso: altero el emulador para que cuando haya ejecutado i3 instrucciones, haga una copia. Lo mismo en i2 y i1.

Sean dif[x] todas las direcciones de memoria que cambian entre i3-i2 y tambien entre i2-i1.
Sea val3[x] el valor de esa direccion en el momento i3. Lo mismo para i2 y i1.

Por ejemplo:
dif[10]=0xFCA0 , lo que significa que la memoria 0xFCA0 contiene valores distintos en i3, i2, y i1.
Supongamos
val3[10]=0x33
val2[10]=0x22
val1[10]=0x11

Lo que quiere decir que:
-en el momento t3, la direcion 0xFCA0 contiene el valor 0x33
-en el momento t2, la direcion 0xFCA0 contiene el valor 0x22
-en el momento t1, la direcion 0xFCA0 contiene el valor 0x11

Ahora viene el punto clave.

Modifico el emulador para que cuando llega la instruccion i2, tome una de esas variables dif[x] y le ponga de nuevo el valor que tenia en el momento i3. O sea, que el bucle procesador de instrucciones hace

```
if(numero_instrucciones_ejecutadas==i2)
 {
 // el antiguo valor deberia ser val2[dif[x]], pero no lo compruebo
 Z80mem[dif[intento_x]]=val3[dif[intento_x]];
 }
```

Para comprobar que efectivamente he cambiado la variable que contiene el numero de vidas, pongo una verificacion en el momento i1 ; si Z80mem[dif[x]] vale lo que en la ejecucion inicial (sin parchear), entonces es que el parche no es el adecuado.

En cambio, si ha sido alterado solo 1 vez (en contraposicion a 2 veces) entonces es correcto:

```
if(numero_instrucciones_ejecutadas==i1)
 {
 if( Z80mem[dif[intento_x]] == val1[dif[intento_x]] )
   parche_inadecuado();
 if( Z80mem[dif[intento_x]] == val2[dif[intento_x]] )
   parche_correcto();
 }
```

Ahora inicio de nuevo el programa, con la primera iteracion: intento_x = 1
Si aterriza en
parche_inadecuado()
entonces lo intento de nuevo:

```
 intento_x=intento_x+1;
 carga_programa_en_emulador()
 inicia_programa_cargado()
```

Pero si llega hasta
parche_correcto()
entonces ha encontrado el parche!

El unico esfuerzo que tengo que hacer es las 3 capturas iniciales, modificar el emulador para definir i3, i2, i1, la lista dif   y decir cuantos elementos tiene esta lista, equivalente al numero de intentos.

Para el programa WestBank, dif[] contiene 450 elementos. O sea, que 450

variables cambian de valor en i3, i2, i1. Consecuentemente tengo que parchear y ejecutar el programa 450 veces, a lo maximo.

El emulador es capaz de ejecutar instrucciones simuladas hasta 4 veces mas rapido que el ZX-Spectrum original asi que las pruebas se realizan en un periquete.

Cada partida dura unos 20 segundos. O sea que en 2 horas y media deberia haber encontrado el parche.
En un 20% de los intentos, el hecho de parchear la memoria hace que el programa pete. No hay problema; simplemente no es el parche adecuado.

Pero tengo relativa suerte, y en una hora ya encuentra el parche correcto.
La direccion de memoria resulta ser 0xD474 y el valor inicial es 3, como cabria haber supuesto.

Ahora resulta facil poner un punto de interrupcion para saber donde se cambia el valor:

```
 if( Z80mem[0xD474] == 0x02 )
   rutina_correcta();
```

E inicio el programa de nuevo.
La rutina que cambia este dato es

```
sub_A372:
....
LD HL, 0xD474
....
LD B, (HL)
....
SUB B, 1
....
LD (HL), B
....
```

Asi que es evidente lo que hay que hacer: eliminando la instruccion
SUB B, 1
ya no reduce el numero de vidas. Fantastico, a jugaaaar!

Este procedimiento sirve para cualquier programa. El segundo dia que lo pruebo consigo vidas infinitas para el Babaliba, y luego para en Tranz-Am.
La verdad es que ha sido mas interesante desarrollar el parcheador que jugar, porque pierde parte del aliciente saber que puedes jugar sin riesgo todo el tiempo que quieras.

He de decir que tambien me ha ayudado el hecho de que este emulador incluye un debugger manual, facilmente modificable para poner puntos de interrupcion y consulta de la memoria.

Por supuesto IDA es capaz de desensamblar codigo del procesador Z80, lo que contribuye a entender el flujo de programa.

Lo mismo se puede usar para el emulador MAME, que permite jugar a muchos juegos antiguos.
En este caso el punto clave para mirar es la rutina
cpu_emulate(int cycles)
que incluye
switch(op)
para analizar la instruccion adecuada.

En este caso la complicacion viene porque no todos usan el procesador Z80, y yo en particular no entiendo codigo ensamblador para procesadores Motorola o Hitachi.

El modelo de memoria tambien es diferente, y realmente no tengo interes en aprenderlo.
Ya hay suficientes juegos disponibles (o migrados) para el Spectrum.
Y yo tampoco tengo mucho tiempo para perderlo jugando.

Este metodo es facil de aplicar pero no siempre es adecuado.
Un ejemplo que no funciona es cuando lo que pretendo es eliminar una proteccion.

Aqui no hay una variable que cambie de 3 a 2, luego de 2 a 1, y luego de 1 a 0.
Por el contrario, en este caso hay una rutina que hace alguna operacion con la clave: si el resultado es satisfactorio, salta a una rutina y el juego empieza.
Si la validacion falla, en general muestra un mensaje y vuelve a solicitar la clave.

Analizando la secuencia, tenemos varios hitos:
tiempo=t0, rutina=r0: se muestra el mensaje solicitando la clave
tiempo=t1, rutina=r1: se ha escrito la ultima letra de la clave, y se pulsa ENTER
tiempo=t2, rutina=r2: esta rutina devuelve el resultado dependiendo de si la clave es correcta
tiempo=t3a, rutina=r3a: si no lo es, pide clave de nuevo, posiblemente saltando a r0 (que llamare tambien r4) en el tiempo=t4a
tiempo=t3b, rutina=r3b: si es correcta, empieza el juego

Obviamente:
t0<t1<t2
t2<t3a
t2<t3b

Usando el emulador podemos seguirle la pista al programa.
Para identificar las posibles rutinas confio en que se usara la instruccion
CALL sub_xxxx
para llamar a una rutina, y
RET
para volver al punto desde donde se ha llamado.

Esto me permite sacar un listado de todas las rutinas existentes.

Arranco el juego en el emulador, espero a que suceda en momento=t0 , y en la rutina=r0 hago el primer volcado de memoria: mem0
Lo mismo en t1, obteniendo la copia mem1
Empiezo a tracear todas las rutinas hasta que llega de nuevo a r0 en el momento t4a

Listo las rutinas sucedidas entre medio, y obtengo 120 distintas.
Alguna tiene que ser la rutina r2 buscada, por narices.

Supongamos que la rutina suma las 4 letras de la clave, y la compara con el valor 0xFC.
O sea, que hace algo asi:

```
r2:
.... ; la clave (4 letras) introducida esta en (HL)
LD A, 0
LD C, 4    ; procesara 4 letras
no_final:
ADD A, (HL)  ; suma cada una al resultado anterior
INC HL
DEC C
JR NZ, no_final
compara:     ; al final el resultado es A=clave[0]+clave[1]+clave[2]+clave[3]
CMP A, 0xFC      ; se compara con este valor
JR Z, correcta
incorrecta:
LD A, 1          ; si es incorrecta, el resultado sera A=0
RET              ; y retorna
correcta:
LD A, 0         ; si es incorrecta, el resultado sera A=1
RET ;
```

Pero recordar que no sabemos donde esta esta rutina.
Como se ve, el punto que marca la diferencia esta en la instruccion
JR Z, correcta

De hecho, la mayoria de los "cracks" que encuentras por ahi lo que hacen es eliminar la condicion (Z) para cambiar esta instruccion por
JR correcta
con lo cual parece que la clave es siempre correcta, aunque no lo sea.

Esto es lo que hara mi crackeador automatico: buscar cada instruccion
JR Z, xxxx
y sustituirla por
JR xxxx

y ver si aterriza en la rutina=r3b

Si no es asi, modifico otras occurrencia y lo intento de nuevo.

Esto plantea varios problemas:
El primero es: ?donde esta r3b?
En otras palabras, hay que encontrar una rutina que se ejecute cuando el juego esta en plena accion.
La manera de encontrar tal rutina deberia ser facil:
-desensamblo todo el programa para encontrar la lista de rutinas: lista_total
  No tiene que ser 100% perfecta
-ejecuto el programa desde el principio hasta el momento t4a, obteniendo (en el emulador) las rutinas ejecutadas: lista4a
-elimina de lista_total todas las rutinas que estan en lista4a, obteniendo lista_no_ejecutadas
-en general, lista_no_ejecutadas contiene muchas mas rutinas que lista4a.
  Aproximadamente 90% de las rutinas en lista_total estan en lista_no_ejecutadas pero no en lista4a
-poner puntos de interrupcion en todas estas rutinas lista_no_ejecutadas .

para mejorar resultados, yo pongo otra restriccion: si paso por 100 rutinas de lista_no_ejecutadas , entonces entiendo que estoy ejecutando el juego, con lo cual he encontrado el parche adecuado!

El segundo problema es que puede que algunos de esos cambios altere demasiado el programa, y provoque el reseteo del juego. Esto se resuelve poniendo un punto de interrupcion en la rutina de reset (direccion 0x0000). Si llego alli, es porque ha petado, y sigo con el siguiente intento.

Tambien es conveniente poner un limite: si al cabo de 2 minutos (o 2 millones de instrucciones ejecutadas) no he llegado a lista_no_ejecutadas ni a r4 , entonces entiendo que me he metido en un bucle infinito, y en este caso voy al siguiente intento.

El tercero de los problemas a resolver es que la instruccion puede ser una variante de
JR Z, xxxx

por ejemplo:

```
JR NZ, xxxx ; usa el dato inverso: 1 en vez de 0
JR C, xxxx  ; usa otro flag: Carry en vez de Zero
RET Z       ; en lugar de saltar, retorna
JP Z, xxxx  ; salto absoluto en lugar de salto relativo
CALL Z       ; llama a otra rutina, en vez de salto corto
```

Asi que mi crackeador tiene que ser capaz de parchear todas estas variantes, tomando la decision inversa de la que tomaria el programa sin parchear.

en pseudo-codigo:

```
direcciones_parcheadas = NULL ;
for(intento_x=0;intento_x<120;intento_x++)
 {
 carga_juego();
 juego_ejecutandose=1;
 programa_parcheado=false;
 numero_instrucciones_ejecutadas=0;
 inicia_juego();
}
```

y dentro del emulador:

```
bucle_continuo:

numero_instrucciones_ejecutadas++;

if(instruccion IN (JR, RET, JP, CALL))
 { // solo parchea si no ha sido previamente parcheado, y cada vez parchea
   // una direccion diferente
```

```
 if(programa_parcheado==false       &&       direccion_ejecutandose       NOT       IN
direcciones_parcheadas)
  {
  programa_parcheado=true;
  direccion_ejecutandose >> direcciones_parcheadas;
  switch(condicion)
    {
    case NZ: condicion=Z  ; break;
    case Z:  condicion=NZ ; break;
    case NC: condicion=C  ; break;
    case C:  condicion=NC ; break;
  }
 }

if(direccion_ejecutandose IN lista_no_ejecutadas)
 {
 juego_ejecutandose++;
 if(juego_ejecutandose>100)
  {
  mensaje "Parche encontrado en intento=" + intento_x ;
  exit(0);
  }
 }

if(direccion_ejecutandose==r4 && programa_parcheado=true)
 {          // si pide la clave por segunda vez ....
 exit(1);  // salta al siguiente intento
 }

if(direccion_ejecutandose==0x0000)
 {          // si el programa se ha reseteado ...
 exit(1);  // salta al siguiente intento
 }

if(numero_instrucciones_ejecutadas>=2000000)
 {          // si el programa tarda demasiado ...
 exit(1);  // salta al siguiente intento
 }
```

Un programa que me gustaria desproteger es DonQuijote2. En teoria cuando acabas la primera fase te dice una clave que te permite continuar con la segunda fase.
Yo no he acabado el primero, pero me gustaria echarle un ojeada al segundo. Pongo en marcha mi auto-crackeador y tras algunos apanyos en el emulador, al cabo de un par de horas me encuentra la rutina que permite jugar.

No he encontrado la clave, pero al menos se como saltarmela !!

--- Claves encadenadas ---
Como se ha visto en el parrafo anterior, esta tecnica reemplaza solo una unica instruccion.
Sin embargo es posible que algunas protecciones anti-copia consistan en varias rutinas.
Supongamos la siguiente rutina de verificacion de clave que comprueba que contiene 8 caracteres numericos y su suma es igual a 40:

```
verifica_clave(char *clave)
{
if(strlen(clave)<>8)
 return -1;
for(i=0;i<8;i++)
 if(clave[i]<'0' || clave[i]>'9')
  return -2;
suma=0;
for(i=0;i<8;i++)
 suma=clave[i]-'0';
if(suma<>40)
 return -3;

return 0;
}
```

Para esta rutina de proteccion son validas las siguientes claves:

55555555
23455678
00088888
pero no son validas:
00000000000 porque no tiene 8 caracteres
abcdefgh   porque contiene caracteres no-numericos
11111111   porque no suma 40

Para romper la rutina de verificacion hay que parchearla en 3
sitios (desconocidos a priori):
-comprobacion de longitud=8 . La llamo rutina_8
-comprobacion de numericos. La llamo rutina_09
-comprobacion de suma=40. La llamo rutina_40

O sea, que hay que aplicar el metodo 3 veces, una por encima de la otra.
En vez de usar
direcciones_parcheadas[]

hay que combinar
direcciones_parcheadas_1[]
direcciones_parcheadas_2[]
direcciones_parcheadas_3[]

Ahora usare las variables
programa_parcheado_en_rutina_1
programa_parcheado_en_rutina_2
programa_parcheado_en_rutina_3
para parchear las 3 rutinas.

Y la rutina de parcheado es

```
if(instruccion in (JR, RET, JP, CALL))
 {
 if(programa_parcheado_en_rutina_1==false    &&    direccion_ejecutandose    NOT    IN
direcciones_parcheadas_1)
  {
  programa_parcheado_en_rutina_1=true;
  direccion_ejecutandose >> direcciones_parcheadas_1;
  condicion=invierte(condicion);
  }
 if(programa_parcheado_en_rutina_2==false    &&    direccion_ejecutandose    NOT    IN
direcciones_parcheadas_2)
  {
  programa_parcheado_en_rutina_2=true;
  direccion_ejecutandose >> direcciones_parcheadas_2;
  condicion=invierte(condicion);
  }
 if(programa_parcheado_en_rutina_3==false    &&    direccion_ejecutandose    NOT    IN
direcciones_parcheadas_3)
  {
  programa_parcheado_en_rutina_3=true;
  direccion_ejecutandose >> direcciones_parcheadas_3;
  condicion=invierte(condicion);
  }
 }
```

Si hay suerte, obtendremos
direcciones_parcheadas_1[intento_1_x]=rutina_8
direcciones_parcheadas_2[intento_2_x]=rutina_09
direcciones_parcheadas_3[intento_3_x]=rutina_40

Al apilar estas 3 condiciones, ahora no hay que hacer solo 120 intentos,
sino 120*120*120=1728000, lo cual puede llevar bastante tiempo.

No solo eso, sino que la tecnica asume que hay 3 protecciones. Si hay mas,
no las encontrara.

Lo que necesitamos es una manera de saber que nos estamos acercando a la
solucion: un indicador de que vamos por el buen camino.
Dejame explicar otro par de ejemplos y luego atacaremos este enigma.

Hasta ahora he roto la protecciones de programas funcionando dentro de un emulador. La ventaja es que puedo ejecutarlo paso a paso, y ademas puedo volver a la situacion inicial: la ejecucion del programa es exactamente la misma puesto que las variables se restauran desde sus valores originales. Esto se llama determinismo.

Esto puede no ser siempre cierto. Supongamos un programa (en Windows) que cada vez que pruebas la clave, escribe en el registro que lo has intentado otra vez. Si lo intentas mas de 10 veces, se desinstala.

Si intentamos aplicar mi tecnica de auto-crackeo, probablemente tardemos mas de 10 intentos, y habria que reinstalarlo de nuevo.

La solucion pasa por dare cuenta de que el programa almacena este dato en el registro, por ejemplo usando RegMon, de www.sysinternals.com
Despues de cada ejecucion del programa, usamos un programa que deje el registro tal como estaba anteriormente.

Otra proteccion "persistente" puede consistir en hacer uso de un fichero contador. En este caso Filemon puede decirnos cuales ficheros resultan modificados. Para dejar el sistema como estaba inicialmente la solucion mas adecuada es instalar la aplicacion en otra unidad de disco, y restaurarla completamente usando PartitionMagic o GhostInstall.

Tambien es posible que el programa sea una version especial para nosotros que solo puede ser activada en los siguientes 3 dias a su adquisicion. Esto se puede identificar por el hecho de que haga llamadas al kernel para obtener la hora. La manera de solucionarlo es cambiar la hora del sistema previamente a cada ejecucion.

Por supuesto, el uso de programas emuladores de sistemas, tales como VMWare, WINE, CoLinux o PowerWindows permite disponer de un sistema facilmente interceptable, y que se puede restaurar en cualquier momento.

Uno de los mayores problemas de sistemas que usan carga de librerias dinamicas es que, a no ser que el sistema sea exactamente igual en cada ejecucion, estas librerias puede que se carguen en direcciones distintas.

En intento_3 puede que MSVCRT este en 78000000
mientras que en intento_4 puede estar en 77F80000

Esto sucederia si entre intento_3 y intento_4 hubieramos ejecutado cualquier otro programa.

Por eso es utilisima la solucion de WINE que permite hacer una copia total de la memoria del sistema Windows que esta ejecutandose.
De paso, esto evita las protecciones de tipo aspack que descomprimen el programa en memoria antes de ejecutarlo. Lo dificil es hacer un parche que modifique el programa comprimido en disco, pero este es otro tema.

El debugger que uso para seguir el flujo del programa se llama OllyDbg.
Lo malo es que no es de codigo libre, asi que no puedo hacerle las modificaciones que quiero. Lo que hago es mandarle simulaciones de pulsaciones de teclas para que reaccione automaticamente.

Posiblemente una buena alternativa seria usar el debugger gdb , pero es un programa muy grande para mi. Tiene tantisimas opciones que todavia no he averiguado como automatizar tareas. Seguro que esta bien documentado y existen excelentes manuales, pero no me he metido a fondo.

Una ventaja adicional de este auto-crackeador es que no es demasiado intrusivo.
Bueno, al menos no mas que el debugger que use.
Para parchear las instrucciones he usado el metodo de decirle a debuger que invierta sobre la marcha el resultado de la condicion:
Z (Zero) lo convierto en NZ (no-Zero)
C (Carry) lo convierto en NZ (no-Carry)
y similarmente con otras instrucciones.

Otra posibilidad es parchear el fichero que contiene el ejecutable. Este procedimiento es totalmente no-intrusivo, pero es dificil saber que el programa efectivamente ha aceptado la clave.

Antes ha quedado en el tintero la tecnica para encontrar algo que nos diga que efectivamente nos estamos acercando a la rutina que comprueba la clave.

La pregunta es en cierto sentido similar a ?Como encuentras algo que has perdido?
La respuesta es: buscando en los sitios donde no has buscado antes.

Voy a desarrollar esta perogrullada:
Los sitios en los que ya hemos buscado son lista4a, de acuerdo con la denominacion anterior.

Se busca una rutina cualquiera de lista_no_ejecutadas, por la que no hemos pasado antes.
Con un analizador se miran las rutinas llamadas por ella, y las que ella llama.
Con esto se obtiene otro conjunto: lista5.
Aplicar de nuevo el analizador hasta que no encontremos mas rutinas que enlacen con ellas. Cada vez lista5 crece un poco mas, hasta que cubre todas sus referencias internas.

Ahora se elimina lista5 de lista_no_ejecutadas.
Si todavia quedan elementos, se aplica de nuevo para obtener lista6.
Y luego lista7, lista8, ...

Tecnicamente, lo que asi obtenemos es una particion del conjunto de rutinas.

Recordar que estos conjuntos disjuntos se obtienen sobre el codigo desensamblado, ya que no se pueden ejecutar estas rutinas pues no sabemos el punto de entrada. Este es el punto clave que impide un analisis automatizado.

Si en uno de estos conjuntos encontramos alguna instruccion sospechosa, miramos
con mas detalle. Son candidatas:
-aquellas rutinas que muestran un mensaje
-los datos que indiquen algo relacionado con una clave, por ejemplo el
  texto "Clave correcta"
-los que escriben un fichero, suenan una musica, o ponen timers
-las que reservan mucha memoria

Bueno, ya se que este parrafo no ha quedado muy detallado, pero es que depende de cada programa en concreto.
Para eso tenemos un cerebro: para pensar !

En otros articulos he contado que para mi movil he hecho una especie de traceador que, cada vez que se llama a una rutina del sistema operativo, me dice la rutina destino, la origen, y los datos, tanto antes de ejecutarse, como despues.
Este traceador es la herramienta que uso para todas las investigaciones que llevo a cabo sobre el funcionamiento del movil.
Una aplicacion de Symbian de tamanyo pequenyo-medio puede ocupar unos 16 Kb, contiene 150 rutinas, y llama a 200 rutinas del Sistema Operativo.
Yo puedo interceptar cada una de esas llamadas, y hacer una copia exacta de la memoria en cualquier momento.

Existe un metodo llamado JTAG para conectar el movil a un dispositivo hardware
para debuggear paso a paso el programa, pero yo no lo he probado.
Que yo sepa, no existe ningun emulador de ARM que permita ejecutar en un PC un programa de Symbian. Programas como
t32marm, de http://www.lauterbach.com
Keil uVision, de Keil Elektronik GmbH
necesitan el conector JTAG mencionado, por lo que la ejecucion se realiza directamente sobre el sistema real, no en el PC.
Pero usando el mismo fundamento que el emulador ASpectrum, me he hecho yo mismo un emulador para ARM, capaz de simular mi movil Siemens-SX1.

Y claro, no he podido evitar la tentacion de usar la tecnica de auto-crackeado para intentar saltarme las claves de un programa llamado 7Seas , de
www.5pro-software.com

El fichero 7Seas.app ocupa 126 Kb , contiene 200 rutinas, y llama a 300 rutinas del sistema operativo.

La proteccion del programa consiste en que deja jugar las 2 primeras pantallas.
Despues, te pide una clave (que depende del IMEI del movil) numerica de 10 digitos, y si es correcta te permite seguir jugando.

Parece perfecto para aplicar mi tecnica, ?no?
En marcha: lo primero que tengo que hacer es identificar una rutina que se llame justo despues de introducir la clave.
Esto lo hago activando mi tracer justo antes de pulsar la tecla ENTER: veo que se llama a la rutina
HandleKeyEvent
con el codigo de la tecla ENTER. Esta sera la rutina r0.

Asi que hago que vuelque toda la memoria en un fichero mem0
A partir de aqui dejo que el movil siga ejecutando el programa, para que mi traceador recoja las rutinas. En total capturo 180 rutinas ejecutadas.
La ultima la llamo r3a, que indica que la clave es incorrecta.
Vuelco esta traza en un fichero.
Pongo en marcha mi emulador, cargo mem0 , y dejo que se ejecute el programa (simulado) en el PC.
Observo que la traza generada en el PC coincide con la generada en el movil.
Bueno, no al principio, porque mi emulador no es perfecto, pero al cabo de un tiempo lo consigo arreglar y al final obtengo la trazas que ya es igual a la obtenida en el movil.

El programa victima contiene 2000 sitios donde se compara una variable con otra.
De todos ellos, solo 300 han sido ejecutados entre r0 y r3a.

Alguna de ellas es la que hay que cambiar.

Por tanto tengo que hacer 300 intentos. Si aterrizo en r3a, es porque no he encontrado la clave, y lo intento de nuevo parcheando otra direccion.

Pero todavia me queda averiguar alguna rutina que se ejecute en el caso de que la clave es correcta.
Para ello, pongo el traceador desde el principio del programa, haciendo que capture las rutinas solo la primera vez:

```
org traceador:
for(i=0;i<num_rutinas_capturadas;i++)
 if(rutina_actual==rutinas_capturadas[i]
  return 0;
rutinas_capturadas[num_rutinas_capturadas++]=rutina_actual;
return 1;
```

Desde que inicio el programa hasta que introduzco la clave erronea, capturo en total 190 rutinas.

Como he indicado, esta aplicacion referencia 300 rutinas del sistema operativo.
Por tanto hay 300-190=110 rutinas que no se han ejecutado: alguna de ellas tiene que ser la que se ejecute cuando la clave es correcta.
Por eso lista_no_ejecutadas[] tiene 110 elementos. Al igual que el caso del Spectrum, no me conformo con que 1 de ellas se ejecute: para considerar que he tenido exito debe saltar al menos a 5 de ellas.

Pongo en marcha el crackeador automatico, que parcheara una por una las 300 comparaciones, invirtiendo su resultado.
Para invertir la instruccion CMP uso la instruccion CMN , y viceversa.
En otras palabras: si
CMP R0, #0
devuelve "cierto", entonces
CMN R0, #0
devuelve "falso"

Por tanto, el bucle encargado de parchear hace:

```
if(instruccion in ( CMP, CMN ))
 {
```

```
 if(programa_parcheado==false       &&       direccion_ejecutandose       NOT       IN
direcciones_parcheadas)
  {
  programa_parcheado=true;
  direccion_ejecutandose >> direcciones_parcheadas ;
  switch(instruccion)
    {
    case CMP: instruccion=CMN ; break;
    case CMN: instruccion=CMP ; break;
  }
 }
```

Tendre que ejecutar el programa 300 veces (como maximo) pero eso no es problema: cada ejecucion tarda menos de 10 segundos, por lo que en media hora obtengo la adecuada instruccion
CMP
que tengo que parchear.

Es cierto que he tenido que desensamblar el programa y tracearlo, pero el resto del proceso ha sido automatico. De hecho, ha costado menos tiempo encontrar la clave, que desarrollar las herramientas que lo han hecho posible.

Ahora, a jugar libremente al 7Seas !

Recientemente he descubierto que los moviles Symbian llevan un debugger incorporado!
Las instrucciones estan en
Symbian_OS_Debugging_Cpp_Applications_v1_0_en.zip
que indican que hay que arrancar en el PC el debugger gdb y en el movil se instala un programa para comunicar el PC con el debugger interno.
Pero una de dos: o yo soy muy torpe, o no funciona correctamente.
Lo primero es que el comunicador no funciona. Lo arranco y siempre da un error.
Lo he probado con varios moviles y no hay manera de que inicie correctamente.
Podria desensamblarlo para intentar ver donde esta el error, pero no se si podre arreglarlo.

Lo segundo es que gdb tampoco encuentra el fichero de configuracion.
Seguramente me falta alguna libreria, pero no se cual.

Pero no desespero: en las instrucciones me dice que usa las funciones de la clase RDebug. No dice mas, aunque encuentro que el fichero del SDK llamado e32svr.h enuncia los metodos.
Tras muchas pruebas llego a la conclusion de que es posible ejecutar un programa paso a paso:
-abrir una sesion de debug con RDebug::Open(16, 16, 16, 0x50000000);
-encontrar el proceso del programa victima usando RThread().Id();
-darnos privilegios mediante RDebug::SupervisorMode(true);
-poner uno o mas puntos de interrupcion con RDebug::SetBreakPoint(id, aAddress);
-ver si ha llegado a uno de estos puntos, con
                    RDebug::GetException(SDebugInfo& ,TRequestStatus&);
-ver la memoria con RDebug::ReadMemory(id, aAddr, aPtr, aLength);
-ver variables haciendo uso de la funcion RDebug::GetRegister(id, Rx, aValue);
-modificarlas con RDebug::SetRegister(id, Rx, aValue);
-continuar la ejecucion con RDebug::Continue(const TThreadId aId);

Pero esto es la tecnica tradicional de debugeado paso a paso. Lo que pretendo en este articulo es usar el auto-crackeo.

Muy bien: el juego elegido es Xonix, que consiste en un tablero del cual hay que ir recortando trozos hasta dejarlo en el 70%. Lo malo es que dentro del tablero hay enemigos. si chocas con ellos pierdes una vida.
Se empieza disponiendo de 5 vidas.

Podria jugarlo en el simulador pero el objetivo es ver si soy capaz de auto-crackearlo desde el propio movil.

El programa se llama xonix.app y ocupa 40 Kb ; unas 7000 lineas en lenguaje ensamblador. El dato "5" aparece unas 200 veces. Intentare sustituirlo por "9".

Para seguir con la misma tecnica debo encontrar un modo de:
-parchear el programa
-empezar a jugar

-ver si empieza con 9 vidas.
-si no, sigue intentandolo.

Para parchearlo uso las rutinas de Symbian de escritura de ficheros:

```
parchea(int intento_x)
{
 filep.Open(fileSession, _L("xonix.bak"), EFileRead)); // abro el original
 Replace(fileSession, "xonix.app", EFileWrite);  // genero el parcheado
 RFileWriteStream outputFileStream(_L("xonix.app")); // empiezo a escribir

 result=filep.Read(aValue);   // voy leyendo bytes
 if(result!=KErrNone)        // si llega al final ...
   {
   CleanupStack::PopAndDestroy(); // cierra ficheros
   return;
   }
 if(aValue==5 && contador==intento_x)
  aValue=9;   // si el valor es "5", escribo "9". La posicion es distinta
              // en cada intento
 outputFileStream << aValue ; // escribo el valor leido (o el modificado)
}
```

Para ejecutar el programa parcheado uso:

```
carga_parcheado()
{
 CApaCommandLine* commandLine = CApaCommandLine::NewLC();
 commandLine->SetLibraryNameL( _L("xonix.app") );
 commandLine->SetCommandL( EApaCommandRun );
 EikDll::StartAppL(*commandLine);
}
```

Ahora tengo que hacer que el programa empiece a jugar. Eso lo hago mandandole simulaciones de teclas pulsadas.
Primero la tecla Menu, y luego Select. Ambas son la tecla de codigo EKeyEnter

```
inicia_partida()
{
 TApaTask ATask3 = aList.FindApp(  _L("xonix.app") ); // busca la aplicacion
 ATask3.SendKey(EKeyEnter,0); // Menu
 ATask3.SendKey(EKeyEnter,0); // Select
}
```

Para ver si tengo 9 vidas solo se me ocurre un metodo: capturar la imagen de la pantalla y ver si en las coordenadas adecuadas esta el numero "5" u otro distinto.
Para simplificarlo, dire que el dibujo del "5" tiene un pixel negro en la posicion (10, 178)

```
Boolean aparece_5()
{
 CWsScreenDevice* cWsScreenDevice = new CWsScreenDevice(wSession);
 TPoint* mypoint = new TPoint(10, 178); // defino el punto
 TRgb* aColor=new TRgb();              // almacenara el color del punto
 cWsScreenDevice->GetPixel(*aColor,*mypoint); // lo saco de la pantalla
 if(aColor==KRgbBlack)       // si es de color negro...
  return true;               // entonces el parche no es el adecuado
 else
  return false;
}
```

Ahora es el momento de combinarlo todo junto:

```
for(intento_x=0, encontrado=false;intento_x<200 && !encontrado; intento_x++ )
 {
 parchea(intento_x);
 carga_parcheado();
 inicia_partida();
 espera(1); // espera un segundo para que la pantalla se refresque.
 if(aparece_5()==true)
  ATask3.KillTask(); // terminarlo, porque el parche no es correcto
 else
  {
  encontrado;
```

```
  Mensaje("Parche encontrado !!!");
  }
 }
```

Aunque lo maximo a probar son 200 intentos, y no me sorprendo cuando se encuentra el valor correcto al cabo de 150 ejecuciones, que tardan unos 30 minutos. Desde luego, mas rapido que el proceso manual de desensamblar/modificar/transferir/jugar/probar de nuevo.

El punto correcto es la rutina
10002BF0   MOV R3, #5
que se codifica como
05 30 A0 E3

Pero esto no es lo importante :-)

Como veis, los ataques de fuerza bruta no solo se pueden aplicar sobre los datos: tambien son utiles cuando alteran el flujo del programa.
Aunque parece que esta tecnica precisa muchos requisitos (emulacion, copia exacta de memoria, traceo paso a paso) lo cierto es que se cumplen la mayor parte de las veces.
Ademas, es aplicable sobre casi cualquier sistema operativo, y cada procesador.

Lo cierto es que esta tecnica se aprecia verdaderamente con el uso.
Leer lo que yo he hecho no es comparable con aplicarlo "es caliente" por lo que te animo a probar, probar, y compartir los resultados.

EOF

Indice

-[3x01]--
-[Cajeros]--
-[by elotro]---

  ****
  CAJEROS AUTOMATICOS   por elotro - elotro.ar@gmail.com
  ****

  Este es un pequenio articulo que puede ser aburrido para algunos, y para
  otros no. Queda en tus manos la responsabilidad de juzgar el contenido
  del mismo.

  Alguna [o mucha] informacion se ha obtenido de:

  - Revista Compumagazine, Edicion de abril de 1995. Anyo VIII, numero 81.
  - Internet, principalmente www.google.com [a que no se lo inmaginaban..]

Todos los que esten esperando una especie de 'How to steal a lot of money',
seguramente se lamentaran, porque lo que aqui trato de exponer no es nada mas
que la historia de los fraudes mas famosos que se realizaron en torno a los
cajeros automaticos. [por lo menos los que fueron de este lado del charco]

La historia transcurre en el anyo 1994. Algunos tal vez recordaran que vieron
en las noticias [muchas veces], los robos que ocurrian en los cajeros
automaticos de Francia y Argentina. A finales del mismo anyo, las empresas
fabricantes de cajeros contrataron a un experto en este tipo de fraudes. Por
supuesto, nunca se probo que habia cometido un delito, pero tampoco se probo
que no los habia cometido.

Ese mismo mes, algunos usuarios de los cajeros argentinos comenzaron a ser
victimas de un tipo de asalto denominado 'pesca'. Consistia en un trozo de
placa radiografica, previamente destenyda con cloro u otro producto parecido.
La placa era seccionada en trozos de unos 95mm x 54mm, el tamanio aproximado
de una tarjeta de debito/credito comun. El rectangulo de placa era doblado
para que quedaran delimitadas 3 secciones de 5mm, 5mm y 85mm.

```
        __ 5mm
5mm    |_____________________________ 85mm
```

El sobrante de la placa (los dobleces) era utilizado para extraer la tarjeta
del cajero.

El delincuente que operaba en el cajero, colocaba la placa rectangular dentro
del orificio destinado a introducir la tarjeta de credito, y se retiraba a
un lugar proximo al cajero para esperar a su victima. Cuando alguien intentaba
realizar una operacion, debia introducir la tarjeta de credito. En el momento
que la tarjeta ingresaba al compartimiento de lectura/escritura, los censores
del lector detectaban la presencia de un elemento extranio (la tarjeta y la
placa), y el cajero quedaba fuera de servicio reteniendo la tarjeta.

En ese mismo instante, una persona se acercaba al que introdujo la tarjeta
y le hacia saber que el cajero estaba funcionando mal, y que hiciera la prueba
de vuelta, colocando la clave otra vez, para tratar de recuperarla.

      [La gente realmente era tan idiota de colocar su clave frente
          a la mirada de un desconocido. Esto sucedia realmente,
                       aunque no lo creas.]

La persona obviamente no recibia ninguna respuesta de parte del cajero, y se

retiraba, esperando recuperar su tarjeta cuando la sucursal abriera, ignorando que la tarjeta seria retirada del cajero, tirando de la placa radiografica. Esta tecnica parece mas fantasiosa que la tecnica de atar un hilo a una moneda para poder hablar gratis en las cabinas telefonicas. Pero realmente funcionaba.

Para neutralizar este modo de robo, la firma fabricante de los cajeros [que no voy a nombrar para no ser boton (1)], hizo llegar a Argentina disenio de cajero que incluia un censor que emitia una senial a la central de computos del banco propietario del cajero. El censor emitia la senial cuando detectaba un objeto extranio en la ranura de las tarjetas de credito. De este sencillo [y estupido] modo se frustro la operacion de 'pesca' de tarjetas.

(1) Boton: En argentina y otros paises de sudamerica, es aquella persona que delata a otro.

El censor en cuestion era un diodo led infrarrojo que emitia luz infrarroja a un fotodiodo que mantenia constante la tension en la entrada + de un operacional, que mantenia un flipflop en estado alto. Si la intensidad de la luz infrarroja disminuia, el operacional ponia la salida en un estado bajo y activaba el circuito de llamada a la central de computos. Como puedes ver, el circuito era una estupidez que no incrementaba el precio de fabricacion del cajero en mas de 10 U$S (dependiendo del tipo de circuito que avisaba a la central de computos)

Pero lo que nunca se difundio fue la utilidad que se le dio a estas tarjetas 'pescadas' para vaciar un determinado modelo de cajero automatico, sin que se llegara a registrar la operacion.

El cajero en cuestion poseia cajones que guardaban separadamente los billetes de distintos valores y tipo de moneda. De estos recipientes se extraian los billetes hasta completar el monto de la extraccion que se indicara. Los billetes eran transportado por una banda hasta el exterior, pasando por distintos censores que registraban el tipo de moneda (dolar, peso), el valor del billete (1,2,5,10,20,50,100) y un censor final que registraba la extraccion del billete. Este censor estaba junto a la ranura de extraccion y registraba tambien que el billete era extraido por el usuario y que no se encontrara atascado en el compartimiento de salida. Al finalizar este proceso se registraba la operacion y se comunicaba a la central de computos.

El procedimiento que se hacia para vaciar los cajeros todabia sigue siendo un misterio para la justicia. Hay especulaciones que el metodo era la anulacion del ultimo censor, de modo que nunca se registrara la extraccion del billete y la operacion se repetia hasta vaciar el cajero, que tampoco registraba la extraccion pues el censor nunca se entero que el billete habia salido del cajero.

Los hechos se repitieron por un mes y medio y nunca mas se repitieron. Los investigadores sospechan que los que cometian el delito eran extranjeros (posiblemente de USA), y que poseian un dispositivo que anulaba los censores. Tambien se especula con la hipotesis de que los ladrones poseian un lector/grabador de tarjetas magneticas y que sabian el algoritmo de codificacion para la extracciones en cajeros. Los bancos en esa epoca tenian pesimos sistemas de seguridad y reconocieron todos sus errores, principalmente uno en la lectura de las tarjetas magneticas, que producia un desbordamiento del contador de billetes y de esta forma el contador nunca se enteraba que el billete habia salido.
Algunos bancos llegaron a ofrecer recompensas para aquel que diera alguna informacion para averiguar el 'modus operandi' de los ladrones.

Para que veas la incompetencia de la policia, y que esto paso realmente, aqui tienes un extracto de SET 22, seccion 0x02:

----------SET 22----------

---{ OTRO GENIO EN LA CARCEL }----------

Un frances de 36 a~os ha sido detenido por crear una tarjeta de credito con la que conseguia que cualquier cajero de banco le proporcionara la cantidad de dinero que este deseara. Lo que al staff de SET le llama mas la atencion es que el sujeto, no la utilizo, solo pretendia venderla a los bancos para demostrarles que su sistema tiene fallos. El juicio se ha celebrado recientemente

y Serge Humpich no ha salido muy mal parado para lo que le pedian los bancos que son los autenticos ladrones en esta y las demas historias.

[O sea, es lo mismo estafarles que no, vas a la carcel igual, cuando lo que deberian de hacer es besarte los pies, pero bueno, que se le va a hacer.....]

[Para que luego digan que no _era_ posible... Ed.]

----------SET 22----------

Se cree que los que cometieron los asaltos eran personas que conocieron el metodo de Serge Humpich y lo practicaron sin discresion .

(o sea, eran unos script-kiddies)

Al cabo de un tiempo la mayoria de los bancos dejaron de insistir con la pesquisa, porque los costos economicos y la reputacion de los bancos era algo demasiado importante como para perseguir a ladrones que nunca tuvieron contacto con un argentino y que estaban en el extranjero. Simplemente aprovecharon la experiencia para considerarla en la fabricacion de los siguientes cajeros.

Soluciones al primer metodo de seguridad implementado por los bancos

Esta solucion es simple pero puede ser algo [muy] dificil de realizar porque no creo que haya un cajero que este tan oculto en el edificio de modo que nadie vea lo que estamos haciendo.

La solucion reside en conectar un fotodiodo a un frecuencimetro y meter el fotodiodo en la ranura de la tarjeta de credito, de modo que no bloqueemos la luz que recibe el fotodiodo del cajero.

El fotodiodo dejara pasar la corriente a la frecuencia que tenga el diodo del cajero, y esta frecuencia nos sera informada por el frecuencimetro.

Finalmente la solucion reside en montar un multivibrador astable con un 555 o un par de transistores, y calibrar la frecuencia de oscilacion a la misma frecuencia del diodo del cajero.

Luego el circuito se esconde en un lugar del cajero o un lugar cercano y se introduce el diodo infrarrojo dentro de la ranura de las tarjetas, procurando que el diodo quede lo mas cercano posible al fotodiodo del cajero (casi siempre esta montado en la parte de abajo). De este modo el fotodiodo siempre recibira luz infrarroja a la misma frecuencia que la que emitia el diodo original y nunca detectara la presencia del 'pescador'.

Disclaimer: Nadie tiene responsabilidad sobre tus acciones (tal vez ni siquiera tu).

-[3x02]----------
-[David y Goliat]----------
-[by seguratas]----------

David vs. Goliat
(o como luchar contra las empresas)

Introduccion

Hoy en dia, las megaestructuras empresariales son las duenas de la economia. Es una cuestion de economia de escala, cuanto m s grande seas m s barato ser s capaz de producir, mejor podr s distribuir y m s eficientemente podr s gestionar tus capitales.

El Sr. Galbraith ya aventuraba estas estructuras y sus implicaciones en sus an lisis de la economia y de sus mecanismos. Y esto, aunque parezca que no tiene conexion con nuestra vida diaria, la tiene y mucha.

Cualquier empresa que se convierta en una superestructura, sin duda

obtiene los beneficios de la misma: beneficios, ahorro de costes, poder de presion sobre el mercado, etc. Pero tambien obtiene los defectos. Estos defectos no son tales para las superestructuras, si no que son meros accidentes a mejorar con el tiempo.

Los defectos se traducen para el usuario de a pie (nosotros) en:

Peor servicio de atencion al cliente
Imposibilidad para el cliente final de hablar con alguien que realmente solucione o pueda solucionar el problema
Choque frontal con el formulario de respuestas tipo del servicio de atencion al cliente.

Quien no ha querido quejarse a una compania, por ejemplo, por una lavadora recien comprada que no funciona, y no ha sido capaz de pasar mas all de una senorita (eso si, muy amable) que nos dice que se pondr n en contacto con nosotros?

Si se ponen en contacto con nosotros, ser despues de mucho tiempo y habitualmente para decirnos que la culpa es nuestra y, de forma sistem tica, sin ninguna solucion. Muchas veces, la contestacion es tan generica que no solo no nos arregla el problema, si no que nos toma por imbeciles.

Este caso se puede aplicar a casi cualquier tipo de empresas: cajas de ahorro, bancos, distribuidoras, luz electrica, servicios, grandes centros comerciales, etc.

El problema no se soluciona porque, habitualmente, las personas que quieren y pueden solucionar el problema nunca llegan a ser conscientes de que existe.

Vamos a ver t cticas para desmontar estas estructuras y utilizarlas para lo que a nosotros nos interesa: conseguir que nos arreglen el problema que no conseguimos solucionar.

Este articulo esta enfocado al ejemplo pr ctico. A muchas personas les gusta teorizar y hablar eternamente sobre el estado de la vida, particularmente, creo que el movimiento se demuestra andando y de poco sirve disertar sobre el movimiento si uno no sabe moverse.

Con el sistema que se explica, en 3 horas podemos generar los suficientes problemas como para que nos solucionen el nuestro en muy poco tiempo.

Vamos a suponer una empresa llamada "Seguridad Incorporated ", sin hacer referencia a nadie especificamente.

Tecnicas de hacking

Cualquier buen comienzo de un ataque empieza con la recopilacion de informacion sobre el objetivo. En el caso de las empresas, es prioritario conseguir informacion sobre los directivos de la misma, asi como de su presidente, a los comites que asiste, las organizaciones de las que es miembro, etc.

Tambien necesitamos el nombre del ariete, que b sicamente es el que nos va a solucionar el problema y al que vamos a poner de referencia. Esta persona es independiente que sea de un nivel jer rquico alto o no. Habitualmente suele ser el responsable del servicio de atencion al cliente, un coordinador del mismo, o el responsable del Dpto. de quejas.

Obtener este nombre es relativamente f cil, se llama al Dpto. de atencion al cliente y se identifica como primer paso a la operadora con la que se est hablando (nombre, apellidos, fecha y hora). Se pide hablar con el responsable del servicio, como es normal la persona que le atiende le dir que no es posible, sin embargo se le advierte (y que quede muy claro) que se van a enviar quejas por carta escrita al director de la compania y que el nombre que figurar es el de la persona con la que se va a hablar.

Lo habitual es que te pasen con un responsable y, como antes, es necesario identificarle con nombre y apellidos. En caso de no conseguirlo, no hay problema, en los emails y cartas nos referiremos al operador con nombres y apellidos y al responsable del mismo nombr ndole con el cargo generico.

Una vez presentada la queja oralmente, ya podemos empezar.

Recopilacion de informacion

Internet es nuestro aliado. Primero identificaremos a los miembros del consejo de administracion y luego a todos los miembros que podamos de la escala jer rquica inferior. Utilizando Internet es relativamente f cil. Hay que identificarlos con nombres y apellidos.

Una vez tenemos los nombres y apellidos, hemos de apuntar el cargo. Da igual que hayan cambiado de presidente o de puesto, lo importante es identificarlos.

Hemos de saber si la empresa tiene presencia en Internet, e identificar el sitio Web que la empresa posee. Si la empresa tiene presencia internacional, mejor que mejor: identifiquemos todos los nombre de la misma que podamos, asi como sus sitios Web.

Con posterioridad, hay que identificar al menos una direccion de email v lida en Internet. Esto es muy f cil si sabemos como buscar. La forma m s eficaz de hacerlo es en http://groups.google.com/. Supongamos que el dominio es "seguridad.es", buscariamos por "@seguridad.es" (esta tecnica vale con cualquier dominio)

Cuando encontremos direcciones del tipo susana.garcia@seguridad.es , viendo el cuerpo del mensaje comprobaremos que corresponde a Susana Garcia (incluso viene lo que probablemente sea su login de usuario en red, pero esto es otra cuestion). Asi podemos ver como se construyen las direcciones de correo y hacer las nuestras.

Se pueden ver otras como marta.gomez@seguridad.es, joseantonio.XXXXXX@seguridad.es, y un largo etcetera. Parece claro que usan el nombre y el primer apellido separado por un punto.

Por otro lado necesitaremos todas las direcciones de email de la p gina Web. Basta con que accedamos a la seccion de Contacto, y de alli podemos habitualmente obtener alguna.

La direccion

En m£ltiples ocasiones, la direccion no corrige un problema endemico en la compania, simplemente, porque no lo conoce. Utilizar los cauces ordinarios no permite hacer conocer a la direccion el problema, ya que los filtros internos evitan que esta informacion llegue a su destino.

Por ello, no nos queda m s que tomar el camino m s corto. Le vamos a mandar directamente nosotros la informacion, y nos vamos a asegurar de que nuestro ariete se va a preocupar de solucionarnos los problemas lo antes posible.

La parte m s f cil es imaginar la direccion de email de los responsables de la empresa. Mirando Internet, r pidamente vemos que algunos miembros de la direccion de Seguridad Incorporated son:

Luisa Maria XXXX
Directora General

Hector XXXX
Responsable de Marketing

Ramon XXXX
Director de Zona

Jorge XXXX
Director Adjunto a la Presidencia

...
...

Lo cual corresponde a:

luisamaria.XXXX@seguridad.es
hector.XXXX@seguridad.es
ramon.XXXX@seguridad.es
jorge.XXXX@seguridad.es
..

Si a estos emails les sumamos los obtenidos en la Web, conseguimos un buen punado de direcciones interesantes.

Accediendo a la directiva

Ahora viene la parte por la cual realmente nos van a prestar atencion, ya que no olvidemos que los directores generales y otros de ese nivel jer rquico tienen secretarias, y ellas habitualmente hacen de filtro y les quitan muchos emails inconvenientes porque sus labores tienen una orientacion distinta a la de solucionar problemas de esta indole.

Una manera de conseguir que, tanto la estructura jer rquica como sus colaboradores inmediatos, pasen al superior la informacion del problema, es instarles a ello recalcando, y demostrando que se puede hacer, que en caso contrario se le va a enviar dicha informacion de forma personal y por vias no interceptables.
Esto es: a su domicilio particular.

Esta informacion es p£blica y est disponible en Internet, basta con saber buscarla. Si buscamos el domicilio social de la empresa, nos dar una pista sobre la ciudad de residencia de las personas que componen la direccion de la compania.

Hace tiempo hice un informe de seguridad que identificaba este peligro: en resumen, que los datos de las personas relevantes de todas las empresas (en realidad de muchisima gente) estaban disponibles p£blicamente en Internet, y que habia que sacarlos de las bases de datos p£blicas ya que con el tiempo trascenderia y seria imposible sacarlos.

Ese momento ya ha llegado, y las bases de datos de Infobel han sido crackeadas y exportadas a texto a las news y en la famosa "mula". Esto significa que, aunque ahora alguien desee desaparecer de los registros de Infobel, Telefonica (www.paginasblancas.com, www.qdq.com, www.paginasmarillas.es) u otros, ya no es posible, es tarde. Una solucion evidente es cambiar de domicilio, pero resulta un tanto costosa.

Con estas bases de datos podemos localizar f cilmente los domicilios particulares de la "direccion" de la empresa (asi como sus telefonos particulares).

Anatomia de un ataque

Lo siguiente que hemos de hacer es redactar una carta/email muy correctos:

- Expresamos claramente nuestro problema

- Identificamos a nuestro ariete (la persona que sea) y explicamos que el mismo senala a la direccion como responsable del problema.

- Dejamos muy claro que vamos a enviar esta carta al director de la empresa (o a los directores) "a las siguientes direcciones", y detallamos estas.

- Especificamos que en esa carta al director incluiremos copia del email/carta presente, junto con la lista de las personas a las que ha sido enviada.

Se envia este email con copia oculta (sin destinatario visible) a todas las personas/emails identificados: tanto los personales (incluido el ariete) como los emails genericos (de la empresa y los relacionados con la direccion, aunque sean de otras empresas *1).

Posteriormente se envia la carta al director por correo ordinario, en los mismos terminos y con las copias citadas.

Ya solo queda esperar.

De los 3 casos en que no he tenido m s remedio que usar esta t ctica,

en menos de 48 horas me han solucionado el problema. Cierto es que requiere un poco de pr ctica y soltura, pero es realmente eficaz.
En casos de companias con representacion internacional (o que la espanola sea una filial), es todavia m s eficaz mandar la queja a la compania matriz (con la misma t ctica): la respuesta es todavia mas r pida.

(*1) Hay ocasiones en que encontramos emails relacionados, ya que el director general de turno puede haber estado haciendo una ponencia en alg£n tipo de seminario, universidad u otras. Es muy eficaz enviar la queja a los companeros de seminario, ya que habitualmente mantienen el contacto y terminar n transmitiendoselo al interesado, con la consecuente "mofa".

-[3x03]----------
-[DOS y economia]----------
-[by seguratas]----------

Nuevas Amenazas
el viejo DoS en las estructuras financieras actuales

Introduccion

El DoS (Denegacion de Servicio) es un viejo conocido de la seguridad. Tradicionalmente ha sido empleado por los hackers como medio de venganza o persuasion a las companias u otras personas.

Sin embargo, las nuevas mafias que vienen (junto con las empresas) han aprendido que Internet es un medio que ahorra costes estructurales a la vez que promueve y mejora el servicio a los usuarios.

El escenario actual: todo es cuestion de dinero.

Las companias actuales, ven en Internet un sistema ideal. Por un lado les ahorra costes estructurales, ya que evita que una compania contrate lineas Frame Relay o punto a punto con sus sucursales. Pero esto no es m s que la punta del iceberg. El correo les evita costosas comunicaciones, la web les evita la impresion de costosos cat logos de productos a la par que les abre nuevas vias de comunicacion y servicio con los clientes (banca electronica, etc).

Pero la cosa no acaba ahi, si pensamos en la telefonia IP, el correo en el movil (blackberry y similares) y un largo etcetera.

Pero esto tampoco acaba aqui, Internet permite trabajar m s r pido y dar mejor servicio a los clientes (menos gente atiende a m s clientes), ahorro de costos en contratacion/compra de oficinas, mejor sistema de comunicacion, expansion internacional casi gratuita, y asi podriamos seguir muchisimo tiempo m s.

Ignorar el problema no lo arregla.

Las mafias se est n dando cuenta del mismo "negocio"... y piensan...si antes extorsion bamos a los del barrio/ciudad/pais y necesit bamos X matones, ahora con 1 nos vale para extorsionar a todo el mundo.

Esto genera la aparicion de pequenas mafias que ya han empezado a hacer sus estragos. Mafias uni o bipersonales extorsionan a grandes entidades financieras o son f cilmente "contratables" para extorsionar a otras companias.

Quienes son los objetivos inminentes? Es evidente... donde esta la pasta? En los bancos.

Los bancos lo saben pero no lo arreglan, no por que no sepan, sino porque, como es tradicional, lo arreglar n cuando sea demasiado tarde (para que vas a invertir unas horas en arreglar o paliar un problema futuro si todavia no lo tienes?).

Los DoS que vienen son potentes, son fulminantes y tienen mala solucion (al menos de momento). Vamos a ver algunos de los que vienen y su manera de solucionarlos.

El viejo DoS

El viejo DoS no es muy efectivo hoy en dia, ya que la mayor parte de la electronica de red incorpora controles anti DoS. Evidentemente algunos siguen y seguir n funcionando.

Vamos a ver con un ligero detalle tecnico los antiguos DoS (los m s conocidos) y los nuevos DoS que van a venir.

Syn Flood.

Cuando se establece una conexion TCP, en realidad se establece una conexion a tres vias. El computador origen envia un paquete TCP con el byte SYN activado, el computador destino le devuelve un SYN+ACK y la conexion se completa con un ACK.

Despues del £ltimo paquete, la conexion TCP se ha establecido. Ahora bien, si t£ le envias al computador destino solo un SYN, el te devuelve un SYN+ACK, pero si no le contestas con un ACK, el computador se quedar esperando un tiempo (por defecto antes era bastante largo) hasta que le llegue.

Bastaba con hacerse un programa muy simple que enviase millones de TCP (estamos hablando de bytes, por lo que imaginaros la cantidad de paquetes que se pueden enviar por segundo) con el bit de SYN activado, falsificando la direccion IP origen y eligiendo una origen que, o bien no exista, o que este sin responder (si no, el computador falsificado, cuando recibiese un paquete SYN+ACK que no habia emitido el, mandaria un RST y la conexion se descartaria).

Este sistema terminaba llenando la tabla de conexiones, y el proceso que tenia el socket en escucha cascaba r pidamente.

El mercado reacciono r pido, se aumentaron el n£mero de conexiones que las tablas podian almacenar, se redujeron los tiempos de espera para los paquetes ACK y la electronica de red metio patrones de deteccion de DoS en curso bas ndose en los ISN (Internet Sequence Number) de los ataques (claro que esto £ltimo se ha basado en el secretismo para que les funcione, eso se les ha ido al garete).

Hoy en dia, todas las redes (o casi todas) son responsables de meter controles antispoofing. Es decir, si t£ est s en una red de tu ISP, pongamos por ejemplo a Telefonica, los paquetes que salgan de la misma no podr ser de una direccion IP origen que no sea asignada a ellos. Incluso con los ADSL este control es din mico por IP.

Las variantes son casi infinitas, pero creo que he descrito el concepto.

Smurfing

Quien no ha hecho un ping a un computador para ver si esta vivo? Y que pasa si hacemos un ping a una direccion de broadcast? Que nos contestan todos los equipos que pertenecen a dicha red. Es decir, tenemos un efecto multiplicador. De esta forma, mandando unos bytes, obtenemos varios Kb de tr fico. Si hacemos spoofing de la direccion IP que deseamos "tumbar" y mandamos millones de paquetes ICMP a millones de direcciones de broadcasting, el resultado es evidente.

Como dije antes, los controles antispoofing y otros especificos del smurf, han restado eficacia a este DoS.

Chargen-Echo

Existen varios servicios por defecto en todos los sistemas operativos. Uno es el servicio de ECHO (no, este no usa el ICMP echo), estamos hablando del UDP al puerto 7. Este servicio devuelve todo lo que se le envia.

El DoS es evidente, haciendo spoofing del computador al que queremos tumbar, le mandamos un paquete UDP con puerto origen 7 y destino el 7 al otro ordenador. La comunicacion entra en un loop infinito y las m quinas se caen.

Los controles antispoofing y varias modificaciones en todas las pilas del TCP de todos los sistemas operativos mundiales, acabaron con este DoS.

El chargen (puerto 19) es similar. Cuando uno se conecta al mismo, devuelve un chorro de caracteres para hacer tests de conectividad. Etc.

En fin, la lista es interminable, pero creo que para muestra ya hemos tenido un boton. Ahora lo importante.

Viejos DoS, nuevas estructuras

Agotamiento de recursos

Tradicionalmente, no se muy bien por que, los administradores prefieren poner todos los recursos de la m quina a disposicion de los servicios para los cuales est n puestas. Es decir, si es un servicio WEB y el mismo ejecuta con el usuario "webuser", no tiene limites de consumo de recursos.

Por otro lado, cuando nosotros abrimos una conexion, por ejemplo web, en realidad destinamos muy poca memoria a abrirla y mantenerla abierta. Si hacemos un programita que abra 5 sockets contra un web y vemos que memoria consume, veremos que es menos de 1 Kb por socket.
Sin embargo, cada proceso que el servidor web hace fork (o thread), ocupa varios Kb (logico, ha de mantener una copia del software que va a atender la peticion en memoria). Tenemos nuevamente un proceso multiplicador. Por cada Kb que yo gasto, el servidor gasta X. Imaginemos que puedo abrir (que no hay problema) 4000 sockets simult neos contra el servidor web...

El DoS est servido. Es muy f cil, lo £nico un pelin m s complicado es la logistica. Se trata de abrir tantas conexiones como sea posible, desde diferentes ubicaciones (si no, el administrador bloquear la IP), y mantenerlas abiertas tanto tiempo como sea posible.

El efecto es devastador en las infraestructuras actuales. Las m quinas, que no tienen control de recursos por usuarios (o lo que es lo mismo, por servicios), no pueden aguantar la carga y devoran recursos hasta que la m quina queda inutilizada. Habitualmente requieren intervencion manual.

He citado el servicio Web, pero este DoS funciona contra servicios de email, ftp, web y todos los que esten basados en conexiones TCP.

En otro articulo m s tecnico entrare en detalles de como hacerlo desde diferentes ubicaciones sin que sea rastreable y pondre codigo fuente ejemplo para que se vea el concepto (que los "malos" ya est n utilizando)

La solucion? Mala, la £nica aproximacion viable es limitar los recursos de la m quina, de tal forma que el servicio quede bloqueado pero la m quina no. En cuanto el DoS cesa, el servicio se restaura autom ticamente (si est bien configurado). Otra manera de paliar el problema es limitar el n£mero de conexiones al limite razonable que pueda aguantar la m quina sin caerse.

Flood a los DNS

Se est empezando a poner de moda la distribucion de codigo fuente que implementa un virus/troyano con ataque incluido al servicio DNS.

Cualquier empresa de Internet basa su servicio en el DNS (Domain Name service), entiendo que sobra explicar que significa el acronimo DNS y sus implicaciones. Como quiera que sea, si una entidad financiera no tiene DNS, no tiene servicio, y por tanto esta fuera del mercado.

Las mafias se han dado cuenta de esto y han disenado ataques que b sicamente anulan este servicio. Como? Muy f cil. Lanzan miles de consultas por segundo a los DNS de la entidad financiera, con nombres aleatorios e inexistentes. Con ello logran colapsar y derribar el servicio DNS.

Hasta ahora, los ataques de este tipo que he visto generan paquetes con el mismo identificador de Query (n£mero de peticion aleatorio DNS, para cachear las contestaciones y que no puedan ser mezcladas/falsificadas/confundidas con otras queries), pero es cuestion de tiempo que el identificador se genere aleatorio tambien.

Las soluciones a este tipo de ataques son muy limitadas, al menos de momento. Si no se usa el DNS y se opta por resolver el nombre de las m quinas de la entidad financiera en local (haria falta la distribucion de un software especifico que modificase el \windows\system32\drivers\etc\hosts a los usuarios inexpertos), la entidad financiera perder la flexibilidad que le otorga el DNS.

Sin embargo, esta solucion se perfila como la £nica viable. Claro que, para ser realmente eficaz, ha de ser implantada antes de sufrir el ataque, ya que si no, despues ser terriblemente costoso.

El correo no aguantar

No hay nada peor para un correo que recibir miles de correos al minuto para destinatarios que no existen de origenes que tampoco existen. No solamente se recibe un correo (habitualmente de pocos bytes) sino que encima la m quina trata de devolverlo a un origen inexistente, lo que genera una cascada de correos salientes.

El sistema de correo, cuando recibe un correo para un destinatario inexistente, ha de verificar toda su lista y luego enviarle un mensaje de error. Si esto se repite, digamos 1000 veces al minuto, el colapso de la m quina es inevitable.

Si estos correos vienen de diferentes IPs, con emails origenes aleatorios y destinatarios aleatorios, el colapso es inevitable y la solucion a este tipo de ataque es muy mala. Evidentemente la t ctica utilizada es similar a la del DNS, un monton de "zombies" con su propio sistema de SMTP que envian correos cada minuto. Conseguir que un virus/troyano "persuada" a 50.000 pc's en internet, es una tarea algo menos que "trivial". He visto incluso variantes que consultan cada 30 minutos los registros MX de los destinos atacados, lo cual invalida soluciones como cambiar los servidores de correo y actualizar/propagar los nuevos registros MX.

La £nica solucion viable que veo de momento, pasa por identificar explicitamente a las X entidades bancarias con las que haya contacto, los X ISP mayores del pais, asi como las X empresas con las que se mantienen contacto, y limitar la posibilidad de abrir sockets a la m quina de correo solamente desde los registros MX de estas empresas, lo cual ayudar a paliar un ataque de este tipo, pero indudablemente generar muchas incomodidades.

Conclusion

El panorama que viene se torna oscuro, y solamente el m s preparado sobrevivir . La toma de decisiones a tiempo ser la que marque la diferencia entre la entidad que lentamente iniciar su lento declinar en este mundo online que viene, y la que prevalecer como lider indiscutible en el mundo online.

No olvidemos que la indisponibilidad de los sistemas ser la clave para la garantia del servicio.... y que es Internet para las empresas sino un medio de ahorrar costes en el servicio?

seguratas@telefonica.net

EOF

Acelerando moviles

------- Presentacion y agradecimientos ------------

En este articulo se explica el modo de incrementar la velocidad del procesador del telefono movil Siemens-SX1.
Tambien se explica el proceso usado para conseguirlo, y se dan pautas para intentar acelerar otros modelos de moviles.

Advertencia: esta accion puede danyar irreversiblemente tu telefono.
No me hagas en ningun modo responsable de lo que pueda suceder.

Como sucede en la mayoria de los casos, he necesitado algo de ayuda.
Agradezco a vovan888 sus ideas, apoyo, y sus explicaciones sobre el funcionamiento de OMAP. Sin el este articulo no habria sido posible.

Tambien doy gracias a Abhejit por dejarme enredar con su Nokia-N70.
Posiblemente si hubiera sabido las perrerias que iba a hacer, no me lo habria permitido.

-------- Herramientas ----------

Como habras deducido de otros articulos escritos por mi, he investigado el funcionamiento de varios moviles, siendo el mas reciente el modelo Siemens-SX1.

Este movil tiene sistema operativo Symbian, que si bien no es de codigo abierto, es posible hacer programas en C++ merced a los cuales yo he analizado las rutinas del kernel para intentar entender lo que hacen.

Esta investigacion se hace con la ayuda de un desensamblador.
Yo he usado IDA porque es el unico que permite desensamblar codigo de procesador ARM. Tambien me he hecho herramientas ad-hoc para tareas rutinarias que IDA no puede hacer.

Aviso: este articulo contiene bastante codigo ensamblador. No es complicado, pero hay que prestar atencion.
En general, las instrucciones MOV y LDR hacen lo mismo: asignan un valor a un registro, o leen/escriben un dato de/hacia la memoria.

Para saber mas sobre el conjunto de instrucciones ARM recomiendo
"Atmel Corporation. ARM7TDMI (Tm) Datasheet"
descargable desde
http://www.atmel.com

------- Empezando el viaje -------------

En cualquier movil Symbian hay un fichero
Z:/System/Libs/ekern.exe

que contiene la base del kernel, es decir, rutinas que ejecutan la parte mas importante del sistema, en un modo protegido que permite el acceso a todos los recursos.

Esta complementado por
Z:/System/Libs/EUser.dll

que contiene rutinas invocables por cualquier programa de usuario, que llaman al kernel cuando necesitan ejecutar algo con privilegios superiores.

O sea, que para acceder a los recursos protegidos hay que pasar a traves de EUser hasta ekern.

Ejemplos de esto es el acceso a la lista de tareas, pantalla, teclado, memoria, timers, scheduler, ...

No todas las rutinas del kernel son accesibles desde EUser, claro. La mayoria de ellas son para consumo privado del kernel.

Para saber mas sobre esto, consultar el documento
"Crossing the Userland" en www.symbian.com

Si de verdad te interesa el tema, debes mirar este documento: contiene no solo una explicacion completa, sino tambien la lista de funciones que saltan de modo usuario hacia modo protegido.

------- Uso del kernel ---------------
En cuanto se inicia el movil la primera rutina que se ejecuta en ekern es:
_E32Startup

que llama a:

-ImpDma::Init1
-namesOMAP1509
-ImpHal::Idle
-PP::KernCSLocked
-K::NextId
-TMessagePool::Allocate
-PowerEmergency

Todas estas rutinas usan varias zonas de memoria:
0x4000???? : memoria RAM rapida usada por el kernel
0x50?????? : memoria ROM, conteniendo los programas ejecutables
0x58?????? : memoria compartida entre el kernel y los dispositivos
0x8000???? : memoria estatica usada por el kernel
0xFFFE???? : mirror de 0x5800???? mapeado en memoria virtual

Por ejemplo, la rutina
Hal::TotalRomInBytes
devuelve el tamanyo de la ROM. Para averiguarlo, lee la direccion 0x4000000C
usando las instrucciones
```
MOV     R3, #0x40000000
LDR     R0, [R3,#0xC]
RET
```

Otro ejemplo:
ImpHal::SystemTime
devuelve la hora del sistema. Para averiguarlo, lee el dato desde la
direccion 0x80000968 usando las instrucciones
```
LDR     R3, =0x80000968
LDR     R0, [R3]
```

y periodicamente
THelen::ReadTimer32K_TCR
ha puesto este valor leyendolo desde el reloj interno:
```
LDR     R3, =0x58007000
STR     R0, [R3]
LDR     R3, =0x80000968
STR     R0, [R3]
```

Existen mas de 2.000 variables usadas de este modo. Algunas son punteros
a zonas de memoria conteniendo mas variables, con lo que es evidente
que su numero y tamanyo es mucho mayor.

Algunas rutinas documentadas en el SDK de Symbian usan estas variables.
Por ejemplo
User::Language(void)
devuelve el lenguaje en el que esta ejecutandose el sistema operativo.

Esta funcion se puede llamar desde cualquier programa.
Entonces se llamara a la rutina del kernel
ExecHandler::Language(void)
que hace
```
LDR     R3, =0x800003FC
LDR     R0, [R3]
RET
```

Es decir, el lenguaje usado se almacena en la memoria de direccion
0x800003FC

Notar que al ser una memoria privada perteneciente al kernel, un programa
de usuario no puede acceder a ella directamente:

```
int *mem, valor;
p=0x800003FC;
valor=*p;
```

Este programa falla porque no tiene permisos para leer la memoria del kernel.

Es necesario llamar a User::xxx y pasar a traves de un ExecHandler:xxx
para que lo haga por nosotros.

En el SX1 es posible modificar la ROM. Asi puedo modificar la rutina
ExecHandler::Language(void)
para que haga otras cosas mas.

?Te parece complicado? Usa un programa tal como FExplorer o SystemExplorer, y transfiere estos archivos al PC y desensamblalos con IDA. Veras que es mas simple de lo que parece.

-------- Zonas de Memoria ----------------------
Algunas de estas variables del kernel se usan para transmitir informacion a los dispositivos.
Por ejemplo he desensamblado la rutina
THelen::SetMmcReg(unsigned int addr, unsigned short value)
que como su nombre indica, sirve para poner un registro de intercambio con el modulo de gestion de la tarjeta MMC.

El desensamblado es:
```
ADD   R0, R0, #0x58006800
STRH  R1, [R0]
```

que simplemente pone el argumento value (R1) en la direccion
0x58006800+addr , siendo addr lo mismo que R0

Esto da una pista de que la direccion 0x58006800 (y siguientes) tienen que ver con la memoria MMC.

Similarmente encuentro otras direcciones:

```
dma:             5800F800
cam:             58005800
tci:             5800EC00
wire:            58002000
wireTx:          58002018
WireClk:         58002010
mmc:             58006800
config:          5800D000
lcd:             5800E000
GigaGpio:        5800A000
Linear2Physical: 58002800
```

?Porque estas direcciones y no otras?
La respuesta se encuentra en el manual de ARM, en concreto el documento
"OMAP5910 Dual-Core Processor. Memory Interface Traffic Controller. Reference Guide"
disponible en www.ti.com

Yo he abierto mi movil y justo en el centro de la placa hay un chip grandote con las letras "OMAP"
No solo eso, sino que en los manuales tecnicos
"Repair Documentation" = Manual_L25e_SX1_V10.pdf
y Diagrams_SX1_MMI.pdf
detalla que el microprocesador es del tipo OMAP310-D3000 de la familia del Texas Instruments OMAP1510.

Como iba diciendo, el manual del OMAP enumera las direcciones usadas para los dispositivos:
```
Camera IF     FFFB6800
MMC           FFFB7800
mWire         FFFB3000
Teclado       FFFB5000
....
```
Una lista completa con 500 variables se encuentra en el fichero
ioomap5910.h incluido en el software IAR Systems 2000

A poco que seas un poco avispado te habras dado cuenta de que los numeros son parecidos:
58000000+addr en SX1 = FFFB0000+1000+addr en OMAP

Esto me ayuda mucho para localizar los dispositivos.

Rebuscando entre todos los documentos me encuentro
"OMAP5910 Dual-Core Processor MPU Subsystem Reference Guide"
y tambien
"OMAP5910 Dual-Core Processor Timer Reference Guide"
http://www-s.ti.com/sc/techlit/spru682

y el que mas llama mi atencion es
"OMAP5910 Dual-Core Processor Clock Generation and System Reset Management. Reference Guide"
http://www-s.ti.com/sc/techlit/spru678

En mitad de este documento dice:

The CLKM1 controls the clock distribution and idle modes of the MPU subsystem, plus the associated private and public peripherals (see Figure 5).

Con detalle explica en 4.1 que la velocidad del procesador esta determinada por la frecuencia del reloj, que es una variable DPLL1 almacenada en una direccion de memoria, y se puede modificar.
Me encuentro en la tabla 4.1.2 con estos datos:

nombre	inicio	fin	tam	acceso
MPU_CLKM (clock control)	FFFECE00	FFFECEFF	256 bytes	32 R/W
DPLL1	FFFECF00	FFFECFFF	256 bytes	32 R/W

El parrafo mas interesante es la imagen 4 en la que dice que DPLL1 se usa como multiplicador de CLKM1, el cual marca la frecuencia de la MPU (procesador central) y los perifericos.
Brevemente: el reloj proporciona una senyal de 12 Mhz llamada CLKIN. Entonces se multiplica por DPLL1 y resulta CLKM1, que sera la frecuencia a la que se ponga el procesador.

Para hacer que mi movil vaya mas rapido, podria poner otro reloj con una frecuencia mayor, aunque yo con el hardware no me llevo muy bien, pero al parecer tambien seria posible cambiar la frecuencea, alterando este multiplicador.

Pero necesito un poco mas de verificacion para confirmar que mi movil me dejara modificar esta variable.

Segun los calculos anteriores deduzco que en symbian estas variables se corresponden con el area
MPU_CLKM = 58000000+EE00
DPLL1   = 58000000+EF00

La memoria 0x5800EE?? correspondiente a MPU_CLKM veo que se usa en la rutina
THelen::SetCLKMReg(unsigned int, unsigned int)
que tiene un nombre (CLKM) consistente con el dato que me ha dicho el manual OMAP.

Por otro lado, la memoria 0x5800EF00 veo que se usa en la rutina
THelen::GetDPLLFrequency
llamada desde
Variant::ProcessorClockInKHz

(Estos nombres de rutinas los he obtenido del listado encontrado por casualidad: 02072004-K1_O2vA201.symbol
Puedes encontrarlo en www.oslik.ru o www.siemens-mobiles.org)

Recapitulando, tengo 2 rutinas que tienen que ver con la velocidad del micro, que usan las direcciones de memoria 5800EE00 y 5800EF00

Claramente el manual me dice que para establecer la velocidad tengo que dividirla en dos partes: el multiplicador, y la frecuencia.
No todos los bits cuentan, asi que hay que hacer uso de mascaras para operar sobre los bits adecuados. Y tampoco todos los multiplicadores valen.

Se empieza por el par (0x0005, 0x2210) que significa 48 Mhz.
Para aumentar en 24 Mhz (para obtener 72 Mhz), se le suma el par (0x0, 0x100)
Esto es valido hasta 119 Mhz.
A partir de aqui, la base es (0x010A, 0x3510)
Esto es valido hasta 167 Mhz.
A partir de aqui, la base es (0x010F, 0x3710)

Por si te has perdido, esta es una lista parcial:

```
velocidad: MPU_CLKM, DPLL1
24   Mhz:   0x0005,  0x2110
48   Mhz:   0x0005,  0x2210
72   Mhz:   0x0005,  0x2310
96   Mhz:   0x0005,  0x2410
120  Mhz:   0x010A,  0x3510
132  Mhz:   0x010A,  0x3590
144  Mhz:   0x010A,  0x3610
150  Mhz:   0x010A,  0x3cb0
156  Mhz:   0x010A,  0x3D30
162  Mhz:   0x010A,  0x3DB0
168  Mhz:   0x010F,  0x3710
174  Mhz:   0x010F,  0x3EB0
180  Mhz:   0x010F,  0x3790
186  Mhz:   0x010F,  0x3FB0
192  Mhz:   0x010F,  0x3810
204  Mhz:   0x020F,  0x3890
216  Mhz:   0x020F,  0x3910
```

Mi SX1 funciona a 120 Mhz , por lo que los datos deberian ser:
MPU_CLKM=0x010A
DPLL1=0x3510

Pero cuando leo la memoria descubro que en realidad son
MPU_CLKM=0x010E (en 5800EE00)
DPLL1=0x3A33   (en 5800EF00)

Lo cierto es que son equivalentes segun la imagen 5 del documento SPRU678.

En este documento se cuentan otras muchas variables que permiten jugar con
la velocidad de los perifericos, comunicaciones, ... pero para acelerar
el procesador principal vale con usar MPU_CLKM y DPLL1.

Por supuesto, si consigues entender bien el documento puedes aprovechar
toda la potencia, ademas de que te haras merecedor de toda mi admiracion
porque yo apenas he entendido la mitad.

-------- Intrepidez ---------------
Ahora, para cambiar on-the-fly la velocidad del procesador solo
tendria que usar otro par, por ejemplo
132 Mhz: 0x010A, 0x3590

Dicho y hecho: parcheo la rutina
ExecHandler::Language(void)

para que haga:

```
if(R7==0xFCA00000)
 {
 mem[MPU_CLKM]=0x010A;
 mem[DPLL1]=0x3590;
 }
```

o el correspondiente en ensamblador:

```
LDR R6, =0xFCA00000    ; valor indicando que quiero cambiar la velocidad
CMP R7, R6             ; flag de entrada. Si vale distinto que R6 entonces
BNE no_FCA0            ; sal
LDR R6, =0x5800EE00    ; si es igual, vamos a modificar la direccion MPU_CLKM
LDR R7, =0x010A        ; poniendo este valor
STR R7, [R6]           ; Lo ponemos en memoria
LDR R6, =0x5800EF00    ; Analogamente con  DPLL1
LDR R7, =0x3590
STR R7, [R6]
no_FCA0:
RET                      ; fin de la rutina
```

Y hago un programilla en Symbian/C++ que haga

```
asm("MOV R7, 0xFCA00000");  // preparar el flag
User::Language(void);     // llamar al modulo de usuario que acabara llamando
                          // al modulo del kernel ExecHandler::Language
```

Lo ejecuto, y aunque aparentemente no mejora mucho, lo mido con un benchmark

y efectivamente va un 10% mas rapido !!!

Hago otras pruebas, y descubro que la velocidad maxima es 192 Mhz.
Esto supone un incremento en un porcentaje del 60%

Si le pido mas velocidad, el telefono se vuelve inestable y se cuelga.

Asi que ahora los juegos son mas rapidos, y las aplicaciones mas eficientes.
Sospecho que los ingenieros de Siemens no se habian imaginado nunca que
alguien consiguiera hacer esto.

---------- Extrapolando ----------
Todos los moviles basados en Symbian deben tener una rutina que dice la
velocidad a la que esta corriendo el procesador. Por tanto sera util saber
como es esta rutina, para asi intentar encontrarla en otros modelos.

Aunque la rutina no tiene que ser exactamente igual, es posible que se
parezca bastante.
Por ejemplo, las llamadas a otras rutinas seguro que no estan en las
mismas direcciones. En el SX1 la rutina
stub_THelen::GetDPLLFrequency(unsigned int)
esta en
0x5002D708
mientras que en el Nokia-N70 estara en otra direccion.

Los registros puede que tambien sean distintos. Aunque el SX1 usa el
registro R3 en la instruccion
LDR   R3, =0x10624DD3
es posible que el Nokia N70 use el registro R5 para hacer lo mismo.

Incluso es posible que el codigo sea ligeramente distinto.

De todos modos. este es el desensamblado de la rutina que dice la velocidad
del microprocesador.

```
; Variant::ProcessorClockInKHz_void_
STMFD   SP!, {R4,LR}          ; almacena registros
LDR     R0, =0x5800EF00       ; memoria con el dato DPLL1
BL      THelen::GetDPLLFrequency(unsigned int) ; lee el dato
LDR     R3, =0x10624DD3       ; multiplicador
UMULL   R2, R3, R0, R3        ; multiplica DPLL1
MOV     R4, R3,LSR#6          ; divide entre 64
LDR     R0, =0x5800EE00           ; memoria conteniendo MPU_CLKM
BL      stub_THelen__GetCLKMReg_unsigned_int_ ; lee el dato (multiplicador)
MOV     R0, R0,LSR#4                      ; divide entre 16
AND     R0, R0, #3                ; usa solo los 2 bits menos significativos
CMP     R0, #2                    ; si vale 2 entonces
BEQ     vale2                   ; salta a vale2
BGT     vale3               ; si es mayor (o sea, 3) , entonces salta a vale3
CMP     R0, #1            ; si vale 1 entonces
BEQ     vale1           ; salta a vale1
B       salir           ; si no, sale
vale3:
CMP     R0, #3        ; mira si vale 3
BNE     salir         ; si no, sale
CMP     R4, #0        ; compara la frecuencia con 0
ADDLT   R3, R4, #7    ; si es menor (o sea, negativo) entonces R3=R4+7
MOVGE   R3, R4        ; pero si es mayor, hace R3=R4
MOV     R4, R3,ASR#3  ; lo divide entre 8
B       salir
vale2:
CMP     R4, #0        ; compara la frecuencia con 0
ADDLT   R3, R4, #3    ; si es menor (o sea, negativo) entonces R3=R4+3
MOVGE   R3, R4        ; pero si es mayor, hace R3=R4
MOV     R4, R3,ASR#2  ; lo divide entre 4
B       salir
vale1:
ADD     R3, R4, R4,LSR#31 ; toma el bit alto
MOV     R4, R3,ASR#1      ; lo divide entre 2
salir:
MOV     R0, R4            ; el resultado siempre se devuelve en R0
LDMFD   SP!, {R4,PC}      ; recupera registros, y retorna
```

Ahora podrias buscar una rutina similar en tu movil.

--------- Primeras coincidencias ----------------------------
La rutina
THelen::GetDPLLFrequency(unsigned int)

esta en el SX1 en 0x50005590 y hace

```
STMFD   SP!, {R4,LR}
BL      THelen__Register16_uint_
MOV     R0, R0,LSL#16
MOV     R1, R0,LSR#16
LDR     R0, =0xB71B00 ; 12.000.000 en decimal
TST     R1, #0x10
.....
```

Asi que busco el dato 0xB71B00 en la memoria del Nokia-N70
Al cabo de un rato de investigacion me encuentro con esto:

```
org 5000B8D0
STMFD   SP!, {LR}
LDR     R0, =0x580E1130
BL      THelen__Register32_uint_
TST     R0, #1
.....
LDR     R2, =0x800005E4
LDR     R3, =0xB71B00 ; 12.000.000 en decimal
STR     R3, [R2]
.....
```

que lo que hace es poner el dato 12.000.000 en la direccion 0x800005E4
Recordar que las direcciones 0x8000???? son direcciones estaticas usadas
por el kernel; no es la memoria compartida con los dispositivos.
Si cambio este dato, el movil *dice* que esta ejecutandose a otra
velocidad, pero no es cierto que este funcionando a dicha velocidad.

Lo que hay que hacer es buscar donde se ubica esta memoria compartida.

Parece que voy por el buen camino.

---------- Apostando sobre seguro ------------------
He encontrado que la rutina Arm::IrqDispatch(void)
esta:
-en la direccion 50007B1C en el SX1
-en 5000D5FC en el Nokia N70
por tanto he podido trazar paralelismos entre ambos moviles.

En el SX1 la rutina es:

```
50007B1C                STMFD   SP!, {R4-R6,LR}
50007B20                LDR     R0, =0x5800EB00
50007B24                BL      THelen::IrqPending(unsigned int)
50007B28                MOV     R5, R0
```

mientras que en el N70 es:

```
5000D5FC                STMFD   SP!, {R4,R5,LR}
5000D600                MOV     R5, #0
5000D604                LDR     R0, =0x580ECB00
5000D608                BL      THelen::IrqPending(unsigned int)
```

Las diferencias son:
-el SX1 almacena R4, R5 y R6 , mientras que el N70 almacena R4 y R5
-el SX1 hace R5=R0 despues de mirar si hay IrqPending , mientras que R7 lo
  hace antes de mirar
-el SX1 sabe que hay una Irq pendiente mirando la memoria 0x5800EB00,
  mientras que el N70 mira en 0x580ECB00

Por tanto parece que
0x5800EB00 en SX1 = 0x580ECB00 en N70

Para confirmarlo, busco otras similitudes entre rutinas, y he encontrado
otra coincidencia en la rutina ImpPic::Init1(void)

En el SX1 esta en 0x5000722C :

```
STMFD   SP!, {R4,R5,LR}
```

```
SUB     SP, SP, #8
MOV     R4, #0
LDR     R5, =Interrupt_level_table
MOV     R3, R4,LSL#2
ADD     R0, R3, #0xEB00
ADD     R0, R0, #0x58000000
LDR     R1, [R5,R3]
BL      THelen::SetIntLevel(unsigned int, unsigned int)
....

En el N70 esta en 0x5000CF68 :
STMFD   SP!, {R4,R5,LR}
SUB     SP, SP, #8
MOV     R4, #0
LDR     R5, =Interrupt_level_table
MOV     R3, R4,LSL#2
ADD     R0, R3, #0x58000000
ADD     R0, R0, #0xEC000
ADD     R0, R0, #0xB00
LDR     R1, [R5,R3]
BL      THelen::SetIntLevel(unsigned int, unsigned int)

....

!Son exactamente iguales, excepto que usan distinta direccion para
almacenar SetIntLevel !

Asi que comparando estas rutinas (y ImpPic::Init3 ) llego
a la conclusion que:
En el SX1:
MPU_CLKM = 5800EE00
DPLL1    = 5800EF00

En el N70:
MPU_CLKM = 580B1C10
DPLL1    = 580B1E10

Ahora la pregunta es ?Como hago para meter los datos que yo quiera en
estas posiciones de memoria?
En el SX1 es facil porque es posible parchear la memoria, pero en el Nokia
no se como hacerlo.

---------- La puerta trasera ----------------
Como he dicho antes, esta zona de memoria esta protegida y solo se puede
escribir desde el kernel mismo.

El fundamento es que si cualquier programa pudiera escribir, seria muy facil
que un error de programacion escribiera un dato erroneo, quizas corrompiendo
el sistema completo del telefono.
Los unicos programas en los que se confia son el kernel mismo, y los
programas en los que confia el kernel.
Por ejemplo, los drivers.
Un driver es un modulo que realmente necesita acceder al hardware y a la
memoria, sin restricciones.

En general estan desarrollados por el propio fabricante del telefono o de
los dispositivos.
Existen unos cuantos drivers, y son ficheros con extension  *.ldd , que
significa "Loadable Device Driver".
Por ejemplo, si un programa de usuario necesita acceder al puerto
serie (o la camara, o el puerto infra-rojos, o BlueTooth, ...) entonces
necesita cargar el driver, y mandarle comandos para que haga lo que el
usuario quiera: mandar un dato por el puerto, capturar una foto, imprimir
por infra-rojos, ...

// Primero, el programa Symbian carga el driver:
User::LoadLogicalDevice(_L("eComm"));  // carga ecomm.ldd

// luego se asigna a un dispositivo
RDevice dev_comm;
r=dev_comm.Open(_L("eComm"),EOwnerProcess); // se usa el nombre interno, no
                                            // el nombre del fichero

// despues se une a un canal
RBusLogicalChannel log_chann;
```

```
log_chann=RBusLogicalChannel::DoCreate("eComm", ... );

// y ya podemos mandarle peticiones:
log_chann->DoRequest(int, TRequestStatus &, void *, void *);

// o tambien
log_chann->DoControl(int);

// y la funcion mas versatil
log_chann->DoControl(int, void *, void *);
```

Lo malo es que no hay ningun driver que permita escribir en una direccion cualquiera de memoria; todos verifican antes que la direccion no es peligrosa.

Pero esto no va a detenerme. Cojo uno de los modulos mas simples: edrm.ldd , que ocupa 944 bytes.
Inicialmente esta en la unidad Z: de solo lectura
Z:/System/Libs/edrm.ldd

asi que lo copio a C: , que es de lectura y escritura.

Compruebo con deleite que cuando arranco el movil, el archivo edrm.ldd usado es el de C: y no el de Z:
Esto es un fallo de seguridad: el kernel prefiere ejecutar las aplicaciones ajenas (en C:) en lugar de las propias !

Asi que desensamblo edrm.ldd , y en la rutina
DLddChanDrm::DoControl(int comando, void *fuente, void *destino)
meto este parche:

```
if(comando==0xFCA00000)
 {
 Mem::Copy(destino, fuente, 4);
 }
```

Lo recompilo, y lo meto en
C:/System/Libs/edrm.ldd

Este parche me permitira escribir cualquier dato en cualquir direccion de memoria; lo unico es que hay que tener cuidado de que la memoria de destino se valida y no corrompa datos criticos.

y hago un programilla que incluya estas lineas:

```
log_chann=RBusLogicalChannel::DoCreate("eDrm", ... );
TInt val_MPU_CLKM=0x010A;
TInt *fuente=&(val_MPU_CLKM);
TInt *destino= TInt *(0x580B1C10);
log_chann->DoControl(val_MPU_CLKM, fuente, destino);

TInt valor_DPLL1=0x3590;
*fuente=&(valor_DPLL1);
*destino= TInt *(0x580B1E10);
log_chann->DoControl(val_DPLL1, fuente, destino);
```

Con gran excitacion transfiero esta aplicacion al movil N-70, la ejecuto, y ahora el movil va a otra velocidad !

No ha sido tan complicado. ?eh?

------- Resultados -----------
En el Siemens-SX1 la velocidad estandar es 120 Mhz, y es posible poner cualquier velocidad entre 48 y 192 Mhz.
En el Nokia-N70 la velocidad estandar es de 220 Mhz y yo lo he conseguido poner a 280 Mhz. Velocidades superiores provocan que se vuelva inestable.

Tambien es posible bajar la velocidad. De este modo se ahorra bateria.
La unica desventaja es que las aplicaciones van mas despacio. Lo que yo hago es usar habitualmente velocidad baja, y cuando quiero jugar o surfear, lo pongo mas rapido. Sin abusar, que no quiero que se me queme el movil.
Mencionar que las comunicaciones de voz y datos no resultan alteradas en absoluto.

Como veis, el mundo de los moviles sigue dando provechosos frutos para aquellos que se arriesgan y tienen ganas de trastear.
EOF

Crack Symbian

Para seguir con el tema de symbian, voy a explicar como se pueden modificar aplicaciones programadas en este sistema operativo.
Esto es valido para cualquier telefono Symbian, no solo el SX1.
Esto incluye n-gage, Sendo, y la mayoria de los Nokias de nueva generacion.
Oficialmente existen 10.000 aplicaciones disponibles, pero a decir verdad yo no me creo esta cifra; si hay mas de 1.000, me daria por contento.

Las herramientas son:
-un movil Symbian
-una aplicacion que deseemos modificar.
-desensamblador IDA o similar.
-es bueno tener a mano el SDK de symbian. Al menos la documentacion.

Las aplicaciones generalmente se descargan desde la pagina web del proveedor, y tienen extension *.sis
Hay otro tipo de aplicaciones hechas en java que se proporcionan como un fichero *.jar y otro *.jad . Estos son programas en java, y no son objetivo de este articulo, ya que los explique en un numero anterior de SET.

Lo primero es transferir la aplicacion hacia el movil:
-por el puerto infrarrojos. Esto es lo que yo uso.
-mediante BlueTooth.
-copiandola en la tarjeta MMC usando un lector/escritor en el PC

Este ultimo metodo no es confortable en los Nokia, porque la MMC esta debajo de la bateria, y para sacarla hay que desmontar el movil. En el SX1 esta en un lateral y se puede extraer incluso con el movil encendido.

A continuacion se inicia una aplicacion interna al movil llamada instalador.
Esta aplicacion se llama
Z:\System\Libs\InstEng.dll
que sigue las instrucciones que le dice el fichero *.sis

Para ver el contenido de este fichero podemos usar la aplicacion SISTool by Alezz, del Z-Team.
Por ejemplo, vamos a analizar la aplicacion FExplorer1.15 , que creo que todo el mundo la tiene, puesto que resulta extremadamente util.
Sirve para manejar ficheros: mover, renombrar, transmitir por BlueTooth, ...
Los ficheros incluidos en el instalador son:

1) License.txt
2) !:\system\apps\FExplorer\FExplorer.aif
3) !:\system\apps\FExplorer\FExplorer.app
4) !:\system\apps\FExplorer\FExplorer.rsc
5) !:\system\apps\FExplorer\FExplorer.mbm
6) !:\system\apps\FExplorer\EXETEST.EXE
7) !:\system\apps\FExplorer\keys.txt

Usando SISTool se ve que el primer fichero tiene un indicador que dice que debe mostrarse durante la instalacion. Ademas presenta las opciones OK y NO para seguir con la instalacion o cancelarla.
El programa instalador se encarga de la tarea de presentar este texto, hacer la pregunta, y recoger la respuesta.
No es posible solicitar ningun numero de serie, ni nada que se le parezca.

Los otros ficheros que empiezan por !:\ indican que se le presentara al usuario la posibilidad de instalarlo en distintas unidades de disco.
Esto suele ser:
E: para la tarjeta MMC, que es lo que yo voy a usar.
C: para la memoria interna no-volatil.

En nuestro ejemplo, no pide la unidad de destino 6 veces, sino que con la primera vez le vale.
Existen otras posibilidades. Por ejemplo, una linea del tipo
C:\system\data\FExplorer.dat
siempre se guardara en la unidad C: . Esto es usado comunmente para ficheros de configuracion, claves de autorizacion, y control de uso. Mas tarde veremos un ejemplo, paciencia.

Asi que tras responder (3 veces !) a las confirmaciones de instalar la aplicacion, la tenemos en
e:\system\apps\FExplorer\FExplorer.app

El fichero FExplorer.aif se puede abrir con la aplicacion SISTool by Alezz.
Vemos que contiene
AIF ID = 0x383A0010
App ID = 0xF17B1F10
1 string = "FExplorer"
2 iconos, con identificadores 000 (el dibujo) y 001 (la mascara)

Todo esto sirve para que el menu principal sea capaz de presentar un icono, y el titulo de la aplicacion.

El fichero FExplorer.mbm se puede abrir con la aplicacion MBMTool by Alezz.
Vemos que contiene 22 dibujos, de los cuales la mitad son iconos, y la otra mitad son las mascaras.

El fichero FExplorer.rsc se puede abrir con la aplicacion RSCTool by Alezz.
Si todavia no te has convencido de que este tipo es un maestro, no se a que esperas.
Vemos que contiene 313 objetos, aglutinados en 92 grupos.
Los objetos son de 2 tipos:
-numeros enteros de 4 bytes.
-cadenas de caracteres, que van precedidas por un valor con su longitud.

Existen otros tipos, pero lo normal es encontrarse solo con estos dos.
Este fichero suele contener las palabras y frases que se usan en la aplicacion.
Por ejemplo, si hay un menu con la palabra "Comprimir Memoria", es casi seguro que esta en este fichero.
Es frecuente encontrarse que una aplicacion incluye varios ficheros FExplorer.r01, FExplorer.r02, ... FExplorer.rXX
Cada fichero contiene las frases en un idioma distinto.
El fichero *.r01 se usa cuando instalas la aplicacion en Ingles.
El r02 sirve para Frances, r04 para espanyol, el r56 para Gujarati, del que puedes obtener mas informacion en
http://en.wikipedia.org/wiki/Gujarati_language
Por tanto el fichero *.sis incluye muchos lenguajes, pero solo uno de ellos se instala en el movil.

Lo bueno es que este fichero es facil de modificar. Hay gente que ha traducido aplicaciones ajenas para adaptarlas a su propio idioma. Si bien esto no se puede considerar como un acto de crackear el programa, lo cierto es que es mejor pedirle permiso al autor. Quizas se alegre e incluya la traduccion en la siguiente version.
Mas interesante es saber que la forma de acceder a uno de tales ficheros desde un programa Symbian es usar la funcion RResourceFile::OpenL

Para leer los datos dentro de este fichero se usa la clase TResourceReader::Read_xxx

Para acelerar la carga de menus, Symbian definio un fichero de extension *.rss
que se compila y se mete en el *.rsc y se lee de un golpe usando la funcion CEikEdwin::ConstructFromResourceL(menu_ID);
Cada opcion dentro del menu tiene un identificador unico.

Este es un ejemplo de FCA00000.rss

```
RESOURCE MENU_PANE r_FCA00000_menu
    {
    items =
        {
        MENU_ITEM {command = 0x0102;  txt = "HacerAlgo";},
        MENU_ITEM {command = 0x0103;  txt = "OtraCosa";},
        MENU_ITEM {command = 0x0104;  txt = "MuchoMas";},
        MENU_ITEM {command = EAknSoftkeyExit;   txt = "Exit";}
        };
    }
```

Hay otro detalle digno de mencionar: cada aplicacion tambien tiene un identificador unico. En teoria hay que solicitarselo a la companyia Symbian,

pero es mas comun inventarse uno propio. Si coincide con el de alguna otra aplicacion, mala suerte.
En particular mi aplicacion tiene el identificador 0xFCA00000

Para obtener el identificador de cada menu simplemente hay que combinar ambos:
0xFCA00000 + 0x0102 = 0xFCA00102
Por lo tanto en el fichero FCA00000.rsc aparecen los bytes
02 01 0A FC
que son la representacion en little-indian de 0xFCA00102

En el programa habra una intruccion del tipo:
```
if(menu_elegido==0xFCA00102)
 {
 ......
 }
```

Este sera un buen punto de inicio para empezar a seguir el flujo del programa.

Los programas Symbian solo funcionan en procesadores ARM, que usan un lenguaje ensamblador mas simple que el que encuentras en un ordenador PC.
Para desensamblar los programas, la mejor herramienta que yo conozco es el IDA-Disassembler. Conoce las instrucciones ARM y saca un listado muy limpio.
Por ejemplo me sorprendio que la secuencia de instrucciones
```
MOV R0, 0x77
ADD R0, 0x100
```

la muestra como
```
MOV R0, 0x177
```

haciendo que el listado sea mas comprimido que lo que sacan otros desensambladores. Por favor soporta a sus creadores y compralo: es una herramiente magnifica.

Una cosa a recordar es que los programas siempre aparecen desensamblados a partir de la direccion 0x10000000.
Cuando se cargan en memoria se ejecutaran en cualquier otra direccion de memoria, normalmente 0xFFF?0000 para los primeros 16 programas que se ejecuten.

Como creo haber dicho en otro articulo, symbian es un sistema operativo en el que los programas completan el link en tiempo de carga.
Lo explico: cuando compilas el programa que llama a una rutina del sistema operativo, no sabes a priori en que direccion de memoria esta dicha rutina. Solo sabes el nombre.
Asi que el programa compilado contiene una referencia a un indicador fijo. Cuando quieras ejecutar el programa, el sistema operativo lo carga en memoria, y arranca un cargador (second-phase loader) que ajusta las direcciones de las rutinas usadas.

Vamos a verlo con un ejemplo:
Este es un trozo de un program escrito en lenguage C/C++ para symbian:
```
RFile myFile;
myFile.Replace(fileSession, "c:\\myfile", EFileWrite);
```

se traduce en codigo ensamblador en algo asi:
```
LDR     R1, [r0+X]      ; en algun sitio existe una referencia a fileSession
LDR     R2, =p_C_myFile ; puntero a un puntero al String "c:\\myfile"
MOV     R3, #0x200      ; EFileWrite es una constante que vale 0x200
BL      stub_RFile_Replace ; rutina que saltara a RFile::Replace, la cual
                           ; necesita 3 argumentos: R1, R2, y R3
......
p_C_myFile DCD C_myFile      ; puntero al String "c:\\myfile"
C_myFile   DCA "c:\\myfile"  ; String "c:\\myfile"

......
stub_RFile_Replace:
LDR     R12, =p_Replace__5RFileR3RFsRC7TDesC16Ui ; direccion que apunta
                                                 ; a la rutina destino
LDR     R12, [R12] ; carga el dato de dicha direccion
BX      R12        ; y salta
......
p_Replace__5RFileR3RFsRC7TDesC16Ui   DCD Replace__5RFileR3RFsRC7TDesC16Ui
Replace__5RFileR3RFsRC7TDesC16Ui     IMPORT Replace__5RFileR3RFsRC7TDesC16Ui
```

.....

por pasos:
-primero se preparan los 3 argumentos necesarios para stub_RFile_Replace
-luego se llama al stub
-el stub lee en Replace__5RFileR3RFsRC7TDesC16Ui la rutina a la que
 debe saltar
-este dato ha sido rellenado por el Sistema Operativo cuando ha
 cargado en memoria el programa

Visto de otro modo, la rutina stub_RFile_Replace es equivalente a:
```
int (void *)lista_funciones[];
int R12 = lista_funciones[index_p_Replace__5RFileR3RFsRC7TDesC16Ui];
(void *)funcion = *(R12);
call funcion(R1, R2, R3);
```

Si no lo has entendido bien, te recomiendo que consultes algun otro
documento en el que explique la carga dinamica de programas; es una
tecnica muy comun en sistemas operativos de disco.

Por eso es posible saber a cuales funciones llama un programa. Esto lo
aprovecha IDA y muestra algo asi:

```
10003C40 MOV  R0, R5
10003C44 BL   stub_DriveAndPath__C10TParseBase ; TParseBase::stub_DriveAndPath(void)
10003C48 ADD  R3, SP, #0x18
```

Aunque no sepamos lo que hace este programa en la rutina en 10003C40, sabemos
que llama a TParseBase::DriveAndPath(void), que es una funcion que retorna
la unidad y la ruta en la que esta instalado el programa.
A partir de aqui empieza la tarea de investigacion.

Como ya es habitual, vamos a trabajar en un caso real.

En el movil SX1 de Siemens es posible actualizar uno mismo el firmware que
viene incluido de serie. No esta al alcance de cualquiera poder alterarlo
para que haga las cosas que uno quiere, pero hay un grupo llamado Z-TeAm
que ha hecho realmente muchos cambios.
En un telefono Symbian existe las unidades:
- Z: que contiene los programas. Es de solo lectura. Ocupa 16 Mb.
- E: que esta en la tarjeta MMC. Obviamente aqui se pueden copiar programas
- C: tambien es de lectura y escritura. Ocupa 4 Mb.
- A: que es de solo lectura. Teoricamente. Ocupa 4 Mb. Hay 1Mb libre

Aqui es donde esta lo bueno: el Z-TeAm ha conseguido que sea posible meter
programas en esta ultima unidad A: , aprovechando el espacio libre.
Pero no se puede escribir desde el propio telefono, sino desde un ordenador.
Basicamente uno tiene que crear una estructura de directorios similar a la
que quiere que vaya a A: , y convertirlo en un mega-fichero de 4 Mb.
Luego se mete en el movil.
La unidad A: contiene unos cuantos programas que no son imprescindibles.
Ademas se pueden meter en la memoria MMC.
Pero en A: queda espacio de sobra para instalar unos cuantos programas.

Uno de los programas mas utiles es el FExplorer 1.15 mencionado
anteriormente.
Asi que seria recomendable tenerlo en la unidad A para que siempre estuviera
accesible aunque cambie la tarjeta MMC.
El problema es que FExplorer se empenya en buscar el archivo de configuracion
en la misma unidad en la que esta instalado. Obviamente no le gusta que
este en A: porque no puede sobre-escribirlo.

La solucion es enganyarlo para que lo lea desde C: y no desde A: .

El archivo de configuracion se llama FEgen.ini
Cargamos FEXplorer.app en el desensamblador IDA y al cabo de 3 minutos
produce el codigo desensamblado.

La palabra "FEgen" la encuentro en:
```
100195B8 aFegen          unicode 5, <FEgen>,0
```

y un poco mas abajo, encuentro
```
100195D0 a_ini           unicode 4, <.ini>,0
```

busco referencias a estas direcciones:
```
10003C90 off_10003C90    DCD aFegen
```

que a su vez viene desde:
```
10003C74   ADD     R4, R4, #8
10003C78   ADD     R0, SP, #0x10
10003C7C   LDR     R1, =aFegen
10003C80   BL      s__7TPtrC16PCUs ; TPtrC16::TPtrC16(ushort const *)
10003C84   MOV     R0, R4
```

O sea, que llama a la funcion TPtrC16::TPtrC16 que segun el manual de
Symbian sirve para iniciar una cadena. No es de extranyar que esta rutina
tan comun se llame en 350 sitios distintos a lo largo del programa.

Un poco mas adelante hace:
```
10003CC0   LDR   R1, =a_ini
10003CC4   BL    s__7TPtrC16PCUs ; TPtrC16::TPtrC16(ushort const *)
10003CC8   MOV   R0, R4
10003CCC   MOV   R1, SP
10003CD0   BL    s_Append__6TDes16RC7TDesC16 ; TDes16::Append(TDesC16 const &)
```

que sirve para concatenar una cadena a otra. A mi me parece que vamos en
la direccion adecuada.
Ahora falta ver donde le ha antepuesto el nombre del directorio.
Para ello investigo el codigo antes de 10003C74 y me encuentro con:

```
10003C44   BL      stub_DriveAndPath__C10TParseBase
....
10003C58   BL      s_Copy__6TDes16RC7TDesC16
```

y la documentacion del API me confirma que TParseBase::DriveAndPath me
devuelve una cadena conteniendo la unidad y el directorio.

Por eso deduzco que el codigo original es algo asi:
```
String unidad_y_directorio;
String ruta_completa;

unidad_y_directorio = TParseBase::DriveAndPath();
ruta_completa.Copy(unidad_y_directorio);
ruta_completa.Append(String("FEgen");
ruta_completa.Append(String(".ini"));
```

Ahora, para conseguir eliminar la unidad de disco de la ruta, tengo que hacer
```
unidad_y_directorio = "C:\" + TParseBase::Path();
```

es decir, que no llame a TParseBase::DriveAndPath() ,
sino a TParseBase::Path();
lamentablemente no existe la rutina TParseBase::Path();

Pero la solucion mas rapida es hacer que no use en absoluto el nombre
del disco donde esta la aplicacion:

```
String unidad_y_directorio;
String ruta_completa;

unidad_y_directorio = TParseBase::DriveAndPath();
//ruta_completa.Copy(unidad_y_directorio);  // <--comentar esta linea
ruta_completa.Append(String("c:\FE");  // el archivo es c:\FE en vez de FEgen
ruta_completa.Append(String(".ini"));
```

En otras palabras: no uso la ruta original de la aplicacion, sino que
el archivo al final sera
c:\FE.ini

?Porque no he usado c:\FEgen ? Pues porque "FEgen" ocupa 5 caracteres, y por
tanto "c:\FEgen" ocuparia 8, lo que me obligaria a hacer hueco en el programa.
Recordar que estoy modificando el archivo binario, no el codigo fuente.
Cuando se altera un binario, no es facil hacer hueco en medio del programa.

Los cambios son:
```
.....
10003C58   NOP ; inicialmente era    BL s_Copy__6TDes16RC7TDesC16
.....
100195B8 aFegen   unicode 5, <c:\FE>,0   ; inicialmente era <FEgen>
```

.....

Para los que esten interesados en codigo binario, la instruccion
BL s_Copy__6TDes16RC7TDesC16
tiene codigo
394800EB

y la instruccion
NOP
tiene codigo
0000A0E1

Para probarlo, modifico FExplorer.app y lo transfiero al movil. Inicio la aplicacion y confirmo que ha creado el fichero c:\FE.ini
Y esta todo listo para meterlo en la unidad A: tal como queria conseguir.

Como veis, no es mucho mas distinto que desensamblar programas para PC.

Al principio he comentado la estructura de un fichero .sis de instalacion.
Ahora me gustaria anyadir que hay algunas posibilidades mas complejas.
Una de ellas es que se pueden instalar distintos ficheros dependiendo de:
-fabricante: Ericsson, Motorola, Nokia, Panasonic, Psion, ...
-modelo
-velocidad de la CPU
-memoria
-numero de teclas
-numero de luces LED
-tamanyo de pantalla
-lenguajes disponibles
....

Otra funcionalidad es que se puede ejecutar la aplicacion tras la instalacion.
Esto es util cuando el programador desea:
-crear una serie de directorios
-verificar que existe un cierto fichero
-solicitar un codigo de proteccion o numero de serie
-conectarse a una pagina web para obtener algun otro dato
-enviar un SMS para informar al programador
.....
y en caso de que alguna condicion no sea correcta, se devuelve un codigo de error al instalador, que automaticamente desinstala el programa.

La mayoria de los programas contienen unicamente:
-la aplicacion principal XXX.app
-el icono y nombre del programa en XXX.aif
-los dibujos que usa XXX.mbm
-los sonidos XXX.mid
-los mensajes en XXX.rsc , XXX.r01, XXX.r02 , ...
-archivo de configuracion XXX.ini o XXX.cfg
-ayuda o notas en XXX.hlp o XXX.txt

Algunos programas mas complejos incluyen:
-librerias XXX.dll
-programas servidores XXX.exe para tareas criticas
-reconocedores XXX.mdl
-bases de datos XXX.db o XXX.dat

Son particularmente utiles los ficheros XXX.mdl que se ejecutan cuando se inicia el movil. Si estas pensando en crear un antivirus (o un virus) , un protector de pantalla (o un sniffer de teclas), o algo que se parezca a un LKM (Loadable Kernel Module) debes mirar en esta direccion.

Dado que este articulo explica como modificar aplicaciones Symbian, voy a explicar otra tecnica.
Por si no ha quedado claro lo repito otra vez: cada programa, tal como esta en el disco (MMC o C:), contiene solo referencias a rutinas del sistema operativo. Cuando se carga en memoria, esas referencias se convierten en las direcciones especificas de las rutinas. Estas direcciones son distintas segun el modelo de movil, e incluso segun la version del Sistema Operativo.

Por ejemplo, para el Siemens-SX1 existen mas de 20 versiones distintas de sistema operativo, segun el idioma con que han sido fabricados: la version 15 en Espanyol y la version 15 en Ruso tienen distintas direcciones

de rutinas.

Por cada rutina externa que es invocada desde el programa, debe de haber un "lanzador" hacia ella. Eso se conoce como Stub.

Aqui hay un par de ejemplos:

```
....
;--------------------------------------------------------------------------
stub_CCoeControl::SizeChanged(void)
 LDR     R12, =SizeChanged__11CCoeControl
 LDR     R12, [R12]
 BX      R12
 off_1001F674  DCD SizeChanged__11CCoeControl ; CCoeControl::SizeChanged(void)
;--------------------------------------------------------------------------
stub_CCoeControl::PositionChanged(void)
 LDR     R12, =PositionChanged__11CCoeControl
 LDR     R12, [R12]
 BX      R12
 off_1001F684                    DCD          PositionChanged__11CCoeControl          ;
CCoeControl::PositionChanged(void)
;--------------------------------------------------------------------------
....
```

Si un programa llama a 200 rutinas externas (creeme, 200 es un numero normal) entonces hay 200 stubs.

Vamos a analizar uno cualquiera de estos stubs.
-Siempre usa 3 instrucciones y 1 dato ; cada uno de estos elementos ocupa 4 bytes, lo cual da 16 bytes
-todos hacen lo mismo; simplemente cambia la direccion desde la que obtienen la direccion de la rutina de destino. (Si, es una direccion que apunta a otra direccion)
-usan el registro R12, sin importarles lo que hubiera antes. Esto parece indicar que R12 es realmente de usar y tirar.

Si recordais otro articulo anterior para el Siemens S45, yo contaba que seria util hacer un traceador para saber cuales rutinas eran llamadas por un programa. Vamos a ver si tambien lo puedo hacer para Symbian.

El objetivo es definir una rutina que se llame cada vez que pase algo interesante, y que diga de donde viene, a donde va, y que datos usa.
?Como se define "sucede algo interesante"? Pues muy facil: cualquier llamada al sistema operativo es interesante.

Puedo interceptar cada uno de los stubs: en lugar de

```
stub_XXX(void)
 LDR     R12, =XXX
 LDR     R12, [R12]
 BX      R12
 off_XXX4  DCD XXX ; XXX
```

lo transformo en

```
stub_XXX(void)
 LDR     R12, =XXX
 LDR     R12, [R12]
 B       mi_debugger  ; <---- esta es la linea que cambio
 off_XXX4  DCD XXX ; XXX
```

Hay que tener cuidado en varios aspectos:
1-la rutina   mi_debugger no puede cambiar ningun registro
2-debe finalizar con   BX R12   , como en el original
3-el programa original debe modificarse para incluir esta rutina debugeadora.
  Por tanto hay que buscar un hueco libre en algun lugar del programa; por ejemplo un string que no se use.
4-debido a la arquitectura ARM, la rutina mi_debugger debe estar en el mismo segmento de memoria que stub_XXX

La primera restriccion es facil de cumplir: solo hay que guardar temporalmente en la pila los registros que necesitemos usar, y al final hay que recuperar los valores originales.
Lo segundo es trivial.
La tercera condicion obliga a usar un debugger pequenyo. Pero hay un truco: podemos saltar a cualquier otra rutina. Lo malo es que hay que

actuar manteniendo la cuarta regla.

Ahora, olvidate por un momento de que los programas se cargan en direcciones de memoria virtuales: suponte que existe una direccion fija (llamada mi_debugger_grande) y que cualquier programa puede saltar a ella.

(Aviso: el siguiente parrafo solo lo entenderas si sabes de arquitectura ARM)
Para saltar a esta rutina podemos usar la instruccion:
- B (branch) , pero solo permite saltar al mismo segmento de memoria, es decir, esta limitado a saltos de maximo 0xFFFF bytes
- BL (branch location), que admite saltos largos pero modifica el registro LR
- BX (branch extended), que necesita un registro, en la forma BX Rb

Usare este tercer modo. Ahora necesito un registro Rb que pueda corromper sin alterar el programa, por ejemplo R11 porque no he visto ningun programa que lo use. Es cierto que lo usa el kernel a veces, pero no interfiere.

Entonces queda:
```
mi_debugger:
 LDR   R11, =mi_debugger_grande
 BX    R11
 off_XXX3  DCD mi_debugger_grande
```

Y ahora es cuando viene la sorpresa: de verdad existe tal direccion de memoria ! De hecho existen muchas, por ejemplo 0x85740000.
La explicacion vendra en un parrafo posterior.

Por tanto queda (en pseudo-codigo)
```
org 0x85740000
mi_debugger_grande:
 guarda_registros
 dump_registros
 dump_direccion_origen
 dump_direccion_destino
 restaura_registros
 BX R12
```

Si te interesa de verdad este debugger, sigue leyendo.

Para guardar los registros usare la pila. Seguro que modificare los registros R1, R2, ... R7 asi que los puedo almacenar todos con una unica instruccion
```
STMFD   SP!, {R1-R7}   ; equivalente a  push R1, push R2, ... push R7
```

Decido que para almacenar los datos interesantes usare la direccion de memoria 0x85750000 y siguientes:
```
LDR R7, =0x85750000
```

El primer dato de esta zona sera un contador que me dice cuantos datos he guardado, y me sirve para saber donde meter el siguiente.
```
LDR R6, [R7]
```

Ahora debo incrementarlo. Cada vez que se invoca al debugger grabare 8 datos de 4 bytes. Asi que el contador se incrementa en 32 cada vez.
En hexadecimal 32 vale 0x20
```
ADD R6, #0x20
```

y lo vuelvo a meter en 0x85750000
```
STR R6, [R7]
```

Ahora debo apuntar al primer byte que esta libre:
```
ADD R7, R6
```

y empiezo a meter los datos; primero los argumentos R1, R2, R3 enviados a la rutina:
```
STR R1, [R7+#0x00]
STR R2, [R7+#0x04]
STR R3, [R7+#0x08]
```

El registro R0 tambien es util
```
STR R0, [R7+#0x0C]
```

Ahora la direccion origen. En Symbian, las rutinas se llaman con la instruccion
```
BL stub_XXX
```

que hace que el registro LR almacene la direccion a la que se debe retornar.
Por eso lo puedo "dumpear" haciendo
STR LR, [R7+#0x10]

La direccion destino es justamente R12
STR R12, [R7+#0x14]

Y ahora me queda hueco para otros 2 registros: uno es la pila
SP (Stack pointer).
Pero para obtener el valor original debo substraer los 7 datos R1-R7 que he
metido al principio, que ocupan 4 bytes cada uno
MOV R1, SP
SUB R1, #0x07*4
STR R1, [R7+#0x18]

y el otro es el registro PC, que contiene los flags (N, Z, V, I).
El resto de los flags no interesan, pero aun asi los guardo tambien
STR PC, [R7+#0x1C]

recupero los registros de la pila:
LDMFD   SP!, {R1-R7}   ; equivalente a pop R7, pop R6, ... pop R2, pop R1

y salto a donde debo saltar
BX R12

Para poner todo el tinglado en marcha necesito varios programas:
-Uno en ensamblador ARM con las 20 instrucciones anteriores. Lo compilo
 con GCC-ARM y me genera un mini-fichero de 20*4 bytes.
 Al resultado lo llamo 0x85740000.bin
-Otro en el PC que transfiera este fichero al movil. Dado que tengo
 infrarrojos en el movil y en el ordenador, uso el IrFtp de Windows.
-Otro en el movil que cargue 0x85740000.bin en la direccion 0x85740000 .
 Este programa lo hago en Symbian. Lo explicare en un parrafo posterior
-El programa victima, adecuadamente parcheado
-para eso me hago un programa parcheador. Basicamente:
 -busca
     LDR   R12, [R12]   ; bytes 00C09CE5
     BX   R12   ; bytes 1CFF2FE1
 -y los substituye por
     LDR   R12, [R12]   ; bytes 00C09CE5
     B   mi_debugger ; bytes xyzt00EA ,donde xyzt depende de donde puedo
                                            meter mi_debugger
 -busca un hueco libre donde escribir el lanzador
   mi_debugger:
    LDR   R11, =0x85740000
    BX   R11
    off_XXX3 DCD 0x85740000
 -como ocupa 3 instrucciones, necesito 3*4 bytes. Estos 12 bytes los localizo
  yo a mano. Normalmente lo que hago es machacar una zona del programa que
  me parezca que no se usa.

-Otro programa que vuelca la traza generada a partir de 0x85750000
-Para transferirlo al PC uso de nuevo el IrFTP
-Hago otro programa para visualizar estos datos. Las rutinas "destino" son
 del sistema operativo, por lo que puedo darles nombre. Lo unico que
 necesito es saber la direccion donde estan almacenadas cada una.
 Las rutinas "origen" son del programa victima, y obviamente no puedo saber
 sus nombres. En esto consiste el arte del desensamblado, ?no?

Este es el programa Symbian que mete el fichero 0x85740000.bin en la
direccion 0x85740000

```
RFs fileSession;
TInt open_result, direccion, valor;
TBuf8<4> four_bytes;
_LIT(x85740000_bin,"c:\\0x85740000.bin");

fileSession.Connect(); // conecta con el servidor de archivos
open_result=filep.Open(fileSession, x85740000_bin, EFileRead);
                                     // abre el fichero 0x85740000.bin

if(open_result==KErrNone) // si todo ha ido bien ...
{
```

```
 direccion = 0x85740000;
 for(i=0;i<20*4;i+=4) // bucle de 20 instrucciones
  {
  filep.Read(four_bytes); // cada instruccion ocupa 4 bytes, que
                                  //   meto en un array
  valor=four_bytes[0]%256 // y lo meto en un TInt, que ocupa 4 bytes
      + (four_bytes[1]%256)*256
      + (four_bytes[2]%256)*256*256
      + (four_bytes[3]%256)*256*256*256 ;
  Mem::Copy(&valor, direccion, sizeof(TInt)); // lo meto en memoria
  direccion+=4; // y salto a la siguiente direccion
  }
}
```

El programa inverso que mete las tramas en un fichero es similar, usando como origen la direccion 0x85750000 .

Como ejemplo, voy a modificar el programa Spectrian V1.51 by White Cloud.
Esta aplicacion es un emulador de ZX-Spectrum para telefonos moviles.
Lo puedes obtener desde www.whitecloudsoftware.com
Se compone de varios ficheros, y el programa se llama Spectrian.app

Al ser un programa comercial, necesita un numero de serie, o de lo contrario caduca a los 30 dias. Simplemente por diversion, voy a intentar eliminar esta limitacion. Una vez mas te emplazo a pagar por el software que uses.

Primero hay que ver como actua el programa: lo arranco, y veo que si no esta registrado, aparece un mensaje indicando el modo de registrarlo (pagando, claro).
Este mensaje permanece durante 3 segundos, que se que hay que transformarlo en milisegundos, dando la cifra 3000000 , o sea, 0x2DC6C0.

Desensamblo con IDA, y busco los sitios donde se usa el dato 0x2DC6C0.
Lo encuentro en la rutina 1000262C.
Cambio el valor a 1 segundo con mi editor hexadecimal, transfiero el programa parcheado, y ahora efectivamente el mensaje aparece solo durante 1 segundo.
Pero con esto no consigo eliminarlo.

Asi que voy a usar mi debugger.

Esta aplicacion usa (10020EB8-1001EBB8)/4 = 0x8C0 stubs, desde

```
1001EBB8 ; CBase::newL(unsigned int)
1001EBB8         LDR     R12, =newL__5CBaseUi
1001EBBC         LDR     R12, [R12]
1001EBC0         BX      R12
1001EBC4 off_1001EBC4    DCD newL__5CBaseUi
.....
hasta
.....
10020EB8  ; PlpVariant::GetMachineIdL(TBuf<128> &)
10020EB8         LDR     R12, =GetMachineIdL__10PlpVariantRt4TBuf1i128
10020EBC         LDR     R12, [R12]
10020EC0         BX      R12
10020EC4 off_10020EC4    DCD GetMachineIdL__10PlpVariantRt4TBuf1i128
```

Parcheo estas 2240 rutinas haciendo que salten a la direccion
libre 0x10022550

```
org 0x10022550
mi_debugger:
 LDR   R11, =mi_debugger_grande
 BX    R11
 off_XXX3  DCD mi_debugger_grande
 mi_debugger_grande  DCD 85740000
```

Y en la direccion 0x85740000 coloco mi debugger.
Pongo en marcha el programa victima, y una vez que ha salido el mensaje diciendo que no esta registrado, vuelco la memoria a partir de 0x85750000, que es donde mi debugger ha escrito la traza de las rutinas interceptadas.

Por ejemplo, una de tales trazas me dice

```
 R0=00401628  R1=004037C4 R2=00000004 R3=80000368
R12=502801F0  R4=00000000 LR=FFFD3934 SP=00403844
```

?Como se interpreta esto?
Lo primero es el dato LR=FFFD3934 que indica la direccion del programa desde el que se ha llamado a la rutina debugeada.
El desensamblador IDA cree que todos los programas se ejecutan a partir de la direccion 0x10000000.
Esto no es del todo correcto, pues el cargador de Symbian se encarga de buscar una direccion de memoria libre en el propio telefono. Debido al sistema de memoria paginada, esto suele ser 0xFFF?0000 , en nuestro caso 0xFFFD0000
Por tanto
LR=FFFD3934
equivale a la direccion 0x10003934 en el desensamblado.
Vamos a mirar el codigo en esa direccion:

```
10003928    MOV     R2, #4
1000392C    BL      sub_1001EE78
10003930    ADD     R0, SP, #0x148
10003934    BL      sub_10020EB8  <---- aqui lo hemos interceptado
10003938    MOV     R4, #0
1000393C    ADD     R5, SP, #0x40
```

Y tambien
```
10020EB8  ; PlpVariant::GetMachineIdL(TBuf<128> &)
10020EB8          LDR     R12, =GetMachineIdL__10PlpVariantRt4TBuf1i128
10020EBC          LDR     R12, [R12]
10020EC0          BX      R12
10020EC4 off_10020EC4     DCD GetMachineIdL__10PlpVariantRt4TBuf1i128
```

Combinando con
R12=502801F0
esto quiere decir que
la rutina
PlpVariant::GetMachineIdL(TBuf<128> &)
esta (en mi modelo de movil) en la direccion 0x502801F0 y Spectrian la invoca desde 10003934 .
En otros modelos de moviles, estara en otra direccion.
Mas tarde explicare una consecuencia importante de esto.

Siguiendo con la explicacion de la traza, la rutina
PlpVariant::GetMachineIdL
necesita un argumento
TBuf<128> &
que combinado con
R1=004037C4
significa que la variable usada como argumento esta en la direccion
004037C4
Si tuviera un debugger en condiciones podria ver el valor de esta variable.
Lo importante es saber que el segmento
0x00400000
contiene los datos del programa en ejecucion, es decir, toda la zona de variables completa.

Asimismo, R0=00401628 indica que la zona de variables locales a esta rutina esta en la direccion 00401628.

El valor
R2=00000004
es consistente con la linea
```
10003928    MOV     R2, #4
```

Por hacer un pequenyo resumen:
-valores de   0x0040???? indican variables del programa
-valores de   0x50?????? indican direcciones de rutinas del sistema operativo
-valores de   0xFFF????? rutinas del programa, equivalente a 0x100????? en IDA

Ya que tenemos las trampas preparadas, vamos a ver que atrapamos.

En el programa encuentro la palabra "RegCode2" que parece bastante sugestiva:
```
1002254C aRegcode2        unicode 8 , <RegCode2>,0
```

Esto es tipico de Symbian: cada palabra es en realidad una estructura de datos:

-el primero (ocupa 4 bytes) es la longitud de la palabra
-el segundo es la palabra, donde cada letra ocupa 2 bytes en formato unicode
-el ultimo (2 bytes) es 00 00 , indicando el fin de cadena

A lo que iba: IDA me dice que esta palabra esta referenciada desde
```
10003384  LDR   R1, =Regcode2
10003388  ADD   R0, SP, #0x28
1000338C  SUB   R0, R0, #4
10003390  BL    Compare__C7TDesC16RC7TDesC16 ; TDesC16::Compare(TDesC16 const &)
10003394  CMP   R0, #0
10003398  BNE   loc_100033B8
```

O sea, que en
[SP, #0x28]
hay un puntero, y se compara con la palabra "Regcode2"
Si no son iguales, salta a loc_100033B8

Si son iguales, se hace
```
...
100033AC    BL      Val__6TLex16Rl ; TLex16::Val(long &)
100033B0    LDR     R3, [SP,#0x59C+var_59C]
```

que sirve para transformar una cadena del tipo "1234" en su valor
numerico 1234.

Bueno, parece que esta es una buena rutina para investigar.
Asi que busco en mis trazas donde se ha llamado a esta rutina
TLex16::Val(long &)

A partir de ahi, con el desensamblado en una ventana, y el traceado en otra,
me hago una idea de lo que hace el programa, y dispongo de la posibilidad de
analizarlo off-line.

Para completar este parrafo, dire que hay un fichero
Spectrian.prf
que contiene la cadena
Regcode2=xxxxxxx

y la rutina previamente analizada se encarga de leerla usando
TFileText::Read(TDes16 &)

Mas tarde, se llama a
PlpVariant::GetMachineIdL(TBuf<128>
para averiguar el numero de serie unico del movil.

Luego se calcula un codigo y se compara con xxxxxxx usando
TDesC16::Compare(TDesC16 const &)const

Si la comparacion no es correcta, entonces se hace:
```
10000A30     MOV R5, #0
```
mientras que si el programa esta satisfactoriamente registrado entonces
```
10000A24     MOV R5, #1
```
saltando en ambos casos a
sub_10003824
donde se verifica este valor.

Si es distinto de 0, entonces no aparece la pantalla de registro.

Resulta trivial parchear el programa para que evite esta ultima comparacion,
y siempre se piense que el progama esta registrado.
Ahora, a jugar al Saimazoom !

Otro programa muy util para Symbian es TomTomCityMaps.
Contiene planos de varias ciudades del mundo. Existe otras versiones
TomTomMobile y TomTomNavigator pero ocupan mucha mas memoria.

El problema con CityMaps es que no funciona en el SX1.
Puedes iniciar el programa, pero se queda en la pantalla de inicio.
Cuando pulsas una tecla da un error, aunque no finaliza el programa.

No tengo ni idea de si podre arreglarlo, pero al menos lo intentare.

La estrategia es la misma: interceptar todas las rutinas para ver lo que hacen.
Cuento con la ventaja de que el programa funciona en un movil Nokia, por lo que puedo instalar el programa parcheado en ambos. Si una rutina devuelve un resultado totalmente diferente (mas alla de lo razonable) entonces habra que investigar en detalle dicha rutina.

El programa llama a (100624D0-1005F7D4)/4 = 0x0B3F rutinas del sistema operativo, y hay 12 bytes libres en 100634A0, suficientes para poner mi_debugger.

Arranco el programa, y en cuanto da el mensaje de error, vuelco la memoria a partir de 0x85750000.

Una de las ultimas rutinas ejecutadas ha sido:

```
 R0=0040F838  R1=00407304 R2=100059BE R3=80000368
R12=5015A6DC  R4=00000000 LR=FFF548F0 SP=0040873C
```

O sea, que se llama desde FFF548F0 (equivalente en IDA a 100048F0) a la rutina en 5015A6DC.
Segun IDA, la rutina es CListBoxData::Reserved_2(void)

Esto es un poco raro. El codigo desensamblado que precede a 100048D8 no contiene llamadas a rutinas relacionadas con la clase CListBoxData , sino que son llamadas a AVKON , por lo que me extranya que se llame a una rutina que, ademas, tiene el extranyo nombre "Reserved_2".

El desensamblado es:

```
100048DC       MOV     R1, R0
100048E0       MOV     R6, #0
100048E4       ADD     R4, SP, #0x2C
100048E8       MOV     R0, R4
100048EC       LDR     R2, =(loc_100059BC+2)
100048F0       BL      sub_10060F34 ; CListBoxData::Reserved_2(void)
100048F4       MOV     R0, R4
100048F8       BL      sub_100617F4 ; TRect::Width
100048FC       ADD     R1, SP, #0x1C
10004900       STR     R0, [SP,#0x98+var_7C]
10004904       STR     R6, [R1,#4]
10004908       MOV     R0, R5
1000490C       BL      sub_10061804 ; TSize::TSize
```

Tambien hay que fijarse en que justo antes de llamar la rutina en 100048F0 se establecen los argumentos R0, R1 y R2, lo cual no es consistente con que CListBoxData::Reserved_2(void)
no acepte argumentos.

Esto me hace sospechar que IDA no ha desensamblado bien la rutina.
La manera de comprobarlo es mirar la rutina que hay en R12=5015A6DC

Lo he mencionado en otro articulo, pero merece la pena repetirlo: yo tengo un listado completo de las direcciones de las rutinas del mi movil Siemens-SX1. No tengo el codigo fuente de Symbian, pero se el nombre de cada rutina, y con eso y un poco de imaginacion me hago una idea de su uso.

En particular miro R12=5015A6DC y descubro que es
CPeriodic::Start(TTimeIntervalMicroSeconds32, TTimeIntervalMicroSeconds32, TCallBack)
y no CListBoxData::Reserved_2(void) como IDA me habia dicho.

Esto tiene mejor pinta. Esta rutina admite 3 argumentos, y el tercero es la direccion de una rutina que se llamara cuando transcurra el tiempo establecido.

En mi caso esta rutina CallBack se define en

```
100048EC       LDR     R2, =(loc_100059BC+2)
```

El codigo

```
100059BC     MVN     R7, #0
100059C0     LDR     R1, [R5,#0xA0]
100059C4     ADD     R2, R4, R4,LSL#1
100059C8     RSB     R2, R4, R2,LSL#3
```

parece ciertamente una rutina, pero hay algo intrigante:
?Porque se llama a loc_100059BC+2 ?
Esto significa saltar en medio de la instruccion
```
MVN     R7, #0
```
y eso no es posible.
O bien salta a 100059BC, o a 100059C0. Nunca a 100059BC+2

En la arquitectura ARM, las instrucciones estan siempre en direcciones multiplo de 4. Al menos eso es lo que dice el manual de ARM.

Lo que me extranya es que funcione en un Nokia. Tras posteriores investigaciones concluyo que el Nokia siempre redondea las direcciones hacia el multiplo de 4.
Pero el SX1 no lo hace.

Asi que el cambio es evidente: en 100048EC debe apuntar a loc_100059BC en vez de loc_100059BC+2

Hago el cambio, y el programa ahora funciona ! Ya puedo ir a Valencia sin temor a perderme por sus calles.

Como he dicho antes, cada modelo de movil tiene las rutinas del sistema operativo en direcciones de memoria distintos. La carga dinamica de programas convierte las direcciones indexadas en direcciones reales, y IDA es capaz de interpretarlas, aunque a veces se equivoque.

Mi listado del SX1 incluye mas de 30.000 rutinas del sistema operativo, aunque posiblemente menos del 80% son usadas por algun programa.

En el movil existe una tabla que contiene las direcciones de todas las rutinas.
En el SX1 esta tabla esta en 0x50F00000 y llega hasta 0x50F25000 , haciendo un total de 40.000 rutinas.

En el Nokia N70, van desde 0x500FF000 hasta 0x50130000, unas 50.000 rutinas.

Si cambiamos en la ROM la direccion de esta rutina, podemos hacer que salte a una rutina propia. En los moviles Siemens es posible actualizar la memoria uno mismo. En los Nokia supongo que tambien, aunque creo que hace falta un cable especial.

Sea la funcion
CCoeControl::SetDimmed(int)
que en mi SX1 esta en la direccion 0x5065C93C, referenciada desde 0x50F16340

que sirve para ocultar(1) o mostrar(0) un menu.

Sin duda, se llamara haciendo
```
LDR     R12, =SetDimmed__11CCoeControli
LDR     R12, [R12]
BX      R12
off_10000F1C    DCD SetDimmed__11CCoeControli
```

Para saber donde esta ubicada esta rutina en el Nokia N70, hago un programa en Symbian que oculta el menu. Lo parcheo con mi debugger e intercepto la llamada a CCoeControl::SetDimmed(int)
Miro el registro R12, que contiene la direccion real de la rutina, y averiguo que esta en 0x50787148

Supongamos que queremos que nunca se oculten los menus en ninguna aplicacion. Obviamente puedo parchear la rutina 5065C93C (o 50787148 en el N70), pero es mas flexible hacer:
-hacer que 0x50F16340 no apunte a 0x5065C93C, sino a 0x5071BE90
-buscar un espacio libre en la ROM, por ejemplo en 0x5071BE90
-desensamblar la rutina en 5065C93C
-modificarla a mi gusto, es decir, haciendo que no considere el primer argumento, sino que siempre muestre el menu
-compilarla usando el GCC-ARM o el compilador Kiel
-meter el codigo generado en la direccion 0x5071BE90

Esto me permite modificar el sistema operativo a mi gusto, siempre que sea capaz de entender lo que hace la rutina parcheada, claro.

Uno de los problemas mencionados es que IDA no es siempre capaz de entender correctamente las llamadas a rutinas externas.

Un programa Symbian de tipo APP contiene una cabecera llamada E32ImageHeader o TRomImageHeader definido en el SDK en e32rom.h , definiendo los parametros del programa, tales como version, fecha de construccion, tamanyo de la pila, numero de librerias usadas, y nombres de tales librerias.

Por cada libreria hay una lista de las funciones usadas dentro de ella.

Por ejemplo, hago un programa prueba.app que llama a
TFindLogicalChannel::Next(TFullName& aResult)
que segun la documentacion del SDK, esta declarado en
e32std.h
y tengo que linkar con la libreria
euser.lib
que usara
euser.dll

Por eso IDA muestra:
; Segment type: Externs
....
; Imports from EUSER[100039e5].DLL
....
IMPORT Next__19TFindLogicalChannelRt4TBuf1i256
....

Vemos que incluye las palabras TFindLogicalChannel y Next pero ?que son todos esos numeros?
Pues son los tipos de argumentos. Por ejemplo, una longitud de TBuf se llama "i1".

Asi que empezando por el final:
TBuf1i256
quiere decir que el argumento recibido es
(TBuf<256> &)

que es lo mismo que
TFullName& aResult

Luego, Rt4 es un puntero a TBuf .

Despues, _19T significa que es un metodo publico de la clase TFindLogicalChannel.

En realidad esto se puede deducir de los ficheros *.h del SDK.

Como IDA no es capaz de entender los argumentos ni los datos de salida, presenta el nombre largo, y eres tu el que tiene que interpretarlos.

Ahora bien, la palabra "TFindLogicalChannel" no esta dentro de prueba.app , asi que la pregunta es ?de donde los saca IDA ?

Si miras el programa prueba.app en binario veras que
Next__19TFindLogicalChannelRt4TBuf1i256
esta representado por los bytes
0xFC020000
que pasados a little-indian son
0x000002FC
o sea 764 en decimal.

Ahora vas al directorio del SDK donde tengas
euser.lib
que en mi caso es
C:\Symbian\6.1\Siemens\SX1\bin\epoc32\release\armi\urel
y haces
ar -xv euser.lib ds00764.o

el cual extrae de la libreria euser.lib el fichero ds00764.o
Dentro de el encuentras la palabra
Next__19TFindLogicalChannelRt4TBuf1i256
como era de esperar.

Asi que IDA debe tener en algun sitio una lista que le dice que

la liberia euser.lib contiene la rutina 0x000002FC y se llama
Next__19TFindLogicalChannelRt4TBuf1i256

Y algunas de las rutinas no contienen el nombre correcto. Por eso se confunde con alguna rutinas.
Lamentablemente no he encontrado esta lista en IDA, asi que supongo que esta comprimido.

Pero lo que si puedo hacer es ayudarle:
1-dejo que desensamble el programa (quizas con algun error)
2-lo exporto a un fichero de texto
3-extraigo la zona de rutina importadas
4-miro las librerias, ej. user.dll
5-miro las funciones, ej. 0x000002FC
6-lo paso a decimal: 764
7-extraigo la funcion 764 de user.lib usando "ar"
8-saco el nombre de la rutina desde el fichero extraido ds00764.o
9-saco los nombres de los argumentos con la utilidad dumpbin
10-los meto en un fichero prueba.idc
11-le digo a IDA que re-lea los nombres de las rutinas desde prueba.idc

Para los pasos 7-9 sigo las instrucciones de Mika Raento en
www.cs.helsinki.fi/u/mraento/symbian/reverse.html
que basicamente explica como obtener los nombres de los argumentos (proceso conocido como "demangling").

Tambien uso este procedimiento para analizar el listado producido por mi_debugger. Recordar que
R12 vale la rutina Symbian que es invocada
LR es la rutina invocante

Mencionar que el SDK incluido en IDA 4.9 permite arreglar estos fallos y regenerar el archivo que contiene la lista de funciones externas.
Pero esto lo dejo para otro articulo en el que contare cosas sobre IDA.

?Que se puede hacer con esta metodologia?
Uno de los parches que mas me ha gustado hacer es cambiar la unidad Z: por C:
Me explico: las aplicaciones internas del movil estan en el disco Z:, que es de solo lectura (se pueden cambiar en el SX1 usando un program especial y re-escribiendo la flash, pero es un rollo hacer esto).
Estas aplicaciones usan ficheros con datos que estan tambien en Z:
Por ejemplo, la aplicacion del reloj (Clock.app) usa unos dibujos que estan en
Z:\System\Data\Clock.mbm

Si no te gustan estos dibujos, no puedes poner otros.

Pero si consiguiera que busque
C:\System\Data\Clock.mbm
en vez de
Z:\System\Data\Clock.mbm
entonces podria cambiar los dibujos, ya que C: es de lectura y escritura.

?Como se sabe donde buscar?
En principio, la aplicacion Clock.app es la que decide donde estan los dibujos que va a usar. Normalmente se calcula el directorio donde ella misma esta instalada (Z:\System\Data\Clock.app) y los busca en su mismo directorio.

Otras veces, sin embargo, incluye la ruta completa, es decir, que
Z:\System\Data\Clock.app
contiene la palabra
Z:\System\Data\Clock.mbm

Es este caso solo hay que parchear la memoria flash usando la unidad C: , es decir, en el fichero Z:\System\Data\Clock.app hay que reemplazar
Z:\System\Data\Clock.mbm
por
C:\System\Data\Clock.mbm

Esto es lo que hacen muchos de los parches. Lo malo es que sirve solo para este programa Clock.app y este fichero Clock.mbm en concreto.

Seria mas flexible hacer un metodo generico.

La manera de abrir un archivo es llamando a la rutina
RFile::Open(RFs &, TDesC16 const &, unsigned int)
donde el segundo argumento es el nombre del fichero, incluyendo la unidad de disco y el directorio.

Esta rutina esta en 0x50195C1C, referenciada desde 0x50F03C7C
Asi que solo tengo que parchear esta funcion:
-si el nombre del fichero comienza por "Z:" entonces:
 -sustituyelo, poniendo C: al principio
 -busca
 -si no lo encuentra, busca el original (en Z:) . Devuelve el resultado
 -si lo encuentra, devuelve el resultado (exito)
-si no, llama a la rutina original sin cambiar nada

En lenguage C++/ASM :

```
org 0x50F03C7C
call mi_RFile::Open       // el original hace call 0x50195C1C

mi_RFile::Open(RFs & mi_RFs, TDesC16 const & mi_archivo, unsigned int mi_flag)
{
TDesC16 & mi_archivo_en_C;
if(mi_archivo.StartsWith("Z:"))
 {
 mi_archivo_en_C=mi_archivo.Replace("Z:", "C:");
 result = original_RFile::Open(mi_RFs, mi_archivo_en_C, mi_flag); // en 0x50195C1C
 if(result<>KErrNone)
   return original_RFile::Open(mi_RFs, mi_archivo, mi_flag);
 else
  return result;
 }
else
 {
 return original_RFile::Open(mi_RFs, mi_archivo, mi_flag);
 }
}
```

Obviamente esta idea no es nueva: cualquier rootkit de PC en Linux o Windows hace esto.

No solo he aumentado la flexibilidad; ahora un monton de ficheros no se obtienen desde Z: sino desde C: , lo cual libera espacio que puedo usar para mis propias rutinas.
Es mas, puedo extender esta rutina para que me diga cuales son los ficheros que se intentan abrir.
Cuando cualquier aplicacion abra un fichero, yo lo sabre. Un sniffer para moviles similar a Filemon (www.sysinternals.com).

Hay una aplicacion de pago llamada SystemExplorer by Justek (www.justek.us) que permite trabajar con ficheros. La diferencia con FExplorer es que puede mostrar el contenido de ficheros con formato de base de datos.

Por ejemplo el fichero c:\system\data\cdbv2.dat es una base de datos.

Este fabuloso programa (ayuda a sus autores !) tiene una proteccion para que solo se pueda usar durante 15 dias. Luego caduca.

Gracias al parche anterior averiguo que uno de los ficheros que abre es C:\system\data\lm2004.dat

Desensamblo el programa SystemExplorer.app y veo que efectivamente hace algo con este fichero.

Asi que cierro el programa, borro el fichero lm2004.dat , lo inicio de nuevo, y veo con satisfaccion que me permite usarlo durante otros 15 dias. Ha sido demasiado facil crackearlo.

Como hemos visto, modificar programas Symbian no es mas complicado que modificar programas de Windows.
Por supuesto no hay tanta documentacion, pero la tecnica de interceptar rutinas funciona bastante bien.
Mi agradecimiento a entusiastas como "18+2", Eric Bustarret, MacKam, Mika Raento, y el Z-TeAm que han hecho accesibles estos conocimientos.

Y por supuesto, tantos y tantos programadores de Symbian repartidos por todo el mundo. Ellos son los que hacen que cada dia sea mejor que el anterior.

EOF

```
  _____ _____ _____ _____ _____ _____ _____ _____ _____ _____ _____ __
 /                                                                     \
|     * * * * *   C U R S O   D E   E L E C T R O N I C A   * * * * *   |
 \ _____ _____ _____ _____ _____ _____ _____ _____ _____ _____ _____ __ /
```

Hola lectores (y lectoras :)) de SET ! Este es el primero de una larga serie de articulos que espero que los ayuden a iniciarse en el mundo de las electronica. Como siempre, si tienes dudas, consejos o insultos; manda un mail a: elotro.ar@gmail.com [leo todos los mails]

Muchos veran el indice y se van a preguntar :
- Muy lindo todo..Pero como hago para transimitir en FM o por la TV ?
- O como se hace para construir circuitos que me sirvan en vez de leer toda esta parafernalia de estupideces electronicas ?

Si eres uno de estos, tal vez escriba unos articulos dedicados especialmente al disenio y montaje de circuitos [igual podrias leer esto, para luego venir a enseniarme]

OJO : Como la mayoria de los articulos, no contiene e~es, asi que van a tener que usar la inmaginacion para leer las palabras...

Desde ya pido perdon por que los esquematicos estan en .gif, lo que a muchos les va a parecer una estupidez, siendo que hay millones de lugares de donde bajar el PsPice por lo menos. Yo tengo un 386 SX con Windows 3.11, y seria una utopia tener el PsPice o el Livewire en mi maquina.

INDICE DE CONTENIDOS - PARTE I - LABORATORIO BASICO DE ELECTRONICA

1. Un poco de (mal) humor

La escena transcurre en la casa de un 'hacker', (llamemoslo Saludos), que esta leyendo este articulo con un amigo , llamemoslo Migos. (para el que no sepa o no se acuerde, leer SET 21, set inbox(0x0B), mensajes 0x02 y 0x03)

Saludos: - -Pero si a mi no me gusta la electronica !
Migos: - Acaso no eres 'hacker'?

S- Si, si soy, soy 'eleet', ayer estaba aburrido y me robe algunos archivos de la NASA y tengo un tatuaje que dice 31337 (digna confesion de un lamer).

M- Dime, los hackers con que trabajan?
S- Yo trabajo con Windows (ver nota1)
M- No idiota! Con que maquina trabajas?
S- Pues trabajo con una Pestium 4 que corre a 33mhz con quiticientos megas de ram y un disco duro de veiteyquince gigas.
(otra confesion de alguien que no sabe de lo que habla. A mi me pasa a veces)

M- Aja, Pentium 4 a 33mhz... XDDDDD. Abre el gabinete y dime que ves...
S- Mmm..veo unas cajitas negras, unas callecitas verdes y otras cosas chiquitas que no se que son...

M- Pues esas cajitas negras son circuitos integrados, esas callecitas verdes forman un circuito impreso y las cosas chiquitas que ves son capacitores, resistencias, transistores, filtros, cristales de cuarzo, etc.

S- Y eso a mi que me importa?
M- Es que en el fondo (bien en el fondo) todo lo que tu haces tiene que ver con la electronica. Cuando prendes tu computadora, utilizas electronica, cuando escuchas a tu heladera funcionar, utiliza electronica, casi todo lo que haces, en el fondo (bien en el fondo), tiene que ver con la electronica

S- Ah....

Nota 1 : Lamentablemente, yo tambien me tengo que aguantar un Windows en mi maquina, porque las otras personas que utilizan la pc en la que escribo, son 'Windows Fans', ademas un 386 SX sirve para poco mas que eso...

[tengo una copia en cd de suse linux, pero no tengo lectora de cd en la pc]
:(

2. Introduccion a la electronica

Dentro de esta introduccion conoceras la pricipal funcion de un laboratorio de electronica, que es proporcionar un lugar de trabajo donde tu, aficionado o profesional, puedas diseniar tus circuitos electronicos, probar otros, o reparar equipos ya construidos. Seria ideal disponer de mucha instrumentacion para poder realizar las medidas con un maximo de precision, pero esto seria imposible [a menos que seas el rey midas], teniendo en cuenta la cantidad de instrumental diferente que se ha construido.

No obstante, la mayoria de las practicas que veras en el curso las podras realizar con un tester, un soldador y una placa de insercion de componentes, o protoboard.

3. Laboratorio basico de electronica

Un laboratorio de electronica se compone principalmente de una mesa de trabajo, aparatos de medida y las herramientas auxiliares.

* Aparatos de medida: Se utilizan para la puesta a punto de los diferentes circuitos. Ejemplos de estos son: tester, osciloscopio, analizador de espectro, frecuencimetro.

Cuando se utiliza cualquiera de estos, cabe hacerse dos preguntas:

- Cual es el punto optimo de funcionamiento ?
- Como llegar a el ?

3.1 Primeros pasos

La primera pregunta debe responderla el disenador del circuito, dando cuantos datos sean necesarios para encontrarlo. Asi, debe dar la tension(1), corriente(2), frecuencia(3) y todos los demas datos a los que deba ajustarse el circuito.

[Los datos numerados son variables en un circuito electronico. No estan detallados en una seccion aparte ya que muchos tienen alguna nocion de que significan. Para los que se estan iniciando en el tema, la energia electrica es un movimiento de electrones. Para analizar los datos compararemos la energia electrica con el agua de una represa :
La tension es equivalente a la cantidad de agua, se mide en Volts (V)
La corriente es equivalente a la presion del agua, o sea la intensidad de la corriente. Se mide en Amperes o amperios (A)
La frecuencia no tiene un equivalente, pero para que se den una idea aproximada, es la cantidad de veces en un segundo que un dato cambia de un estado alto a otro alto de signo contrario, o a uno nulo.
Se mide en Hertz (Hz)]

La segunda pregunta solo se puede contestar de una manera: con los instrumentos de medida. No te asustes por la lista que veras mas adelante, para el 90% de los casos solo sera necesario un buen tester.
No cabe duda, que mientras mas completo sea tu instrumental, mejor llegaras al punto optimo y mas perfectos ajustes podras realizar.

[Recomendado: Espera las proximas entregas para ver como puedes realizar por ti mismo muchos instrumentos de medida]

3.2 Herramientas

Igualmente es importante el papel de las herramientas, ya que de nada serviria tener el mejor instrumental, si no dispones de un destornillador para variar una sencilla resistencia ajustable, o si no dispusieras de un soldador para unir cables o soldar componentes.

3.3 Laboratorio basico

Con un tester, un soldador y otras simples herramientas, se pueden realizar la mayoria de los montajes y medidas accesibles a nuestros bolsillos.
No necesitaremos otras cosas muy especiales hasta entregas muy avanzadas, en las que entraremos en temas que te interesaran dentro del mundo de la electronica. Estos temas son temas muy antiguos, pero que a la mayoria le interesaran, tales como el montajes de boxes telefonicas [que supongo que sabes que en nuestros dias no tienen ningun uso util]

3.3.1 Instrumentos indispensables

Los instrumentos que no te deben faltar si de verdad quieres ingresar profundamente en el mundo de la electronica son:

- Un polimetro o tester del modelo analogico o digital.

- Una fuente de alimentacion regulable, que servira para alimentar con corriente electrica los circuitos que montes.
  [hay una de muy buena calidad al final del articulo]

- Osciloscopio : No compres uno, porque son MUY caros. Pero todo en esta vida tiene solucion : hay muchos programas en internet que cumplen la misma funcion mediante la entrada de senial de la placa de sonido de la PC . [aunque la mayoria son para windows] . La calidad de un osciloscopio real y uno virtual no se puede comparar, porque el real ganaria por goleada. Aunque el virtual es mas barato..

- Generador de senial : Igual que el anterior, hay muchos programas que hacen lo mismo y que tambien funcionan por la placa de sonido.

- Frecuencimetro digital : Puedes construir el que vendra en las proximas entregas o conseguir el programa que cumple la misma funcion.

Con estos pocos instrumentos y programas ya tienes un buen laboratorio para trabajar con equipos de baja frecuencia, como amplificadores, reguladores, circuitos de control, microcontroladores, etc.

3.3.2 Banco de trabajo

No creo que sea necesario explicar como conseguir un lugar para trabajar con equipos electronicos, pero nunca esta de mas dar informacion, y es sabido que la informacion no ocupa lugar. [salvo en mi disco de 120 megas que se va consumiendo poco a poco]

Si no quieres construir el banco porque no quieres gastar dinero o porque no tienes tiempo [o porque eres un poco flojo], simplemente busca una mesa que tenga unos cuantos enchufes a su alrededor, y trabajo terminado.

Mi recomendacion es un banco dividido en 2 partes: una elevada del resto, donde colocar los instrumentos de medida para que sean de facil lectura; y otra debajo de esta donde colocar los equipos y circuitos a estudiar.

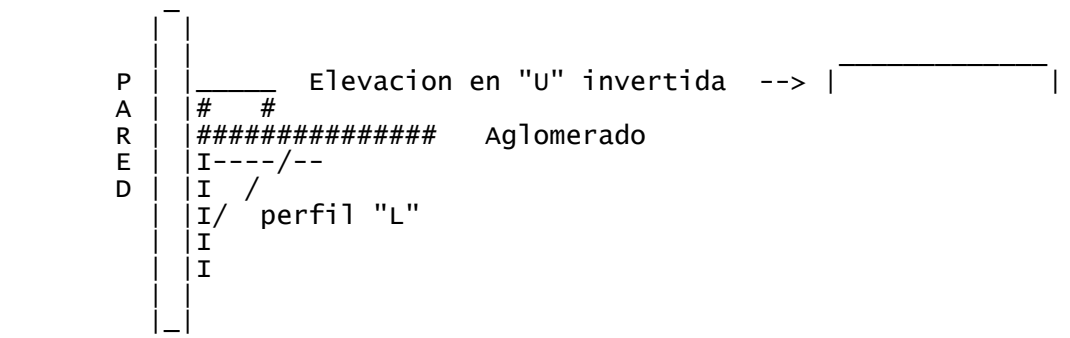
Para construirlo, primero consigues en cualquier carpinteria o aserradero una plancha de material aglomerado (que es resistente y barato sobre todo), de mas o menos 1 m x 70 cm, y otra de 1.30m x 15cm. Si no consigues estas medidas, simplemente compras unas de tamanios mas grandes

Luego, vas a una ferreteria o tienda de abarrotes [no se como se llama a esta clase de comercios en espania], y consigues unos perfiles en forma de "L", les haces 2 agujeros de cada lado, atornillas los perfiles a la plancha de aglomerado mas grande, y luego los colocas en una pared con la ayuda de pernos plasticos de sujecion (tacos).

La elevacion la realizas ensamblando el aglomerado en forma de "U" invertida, y luego la sujetas al banco con cola de carpintero o algun buen pegamento [para que no gastes dinero en tornillos o clavos.]

Conviene instalar unos ganchos a un costado del banco, para sujetar toda la marania de cables que puedas llegar a tener.

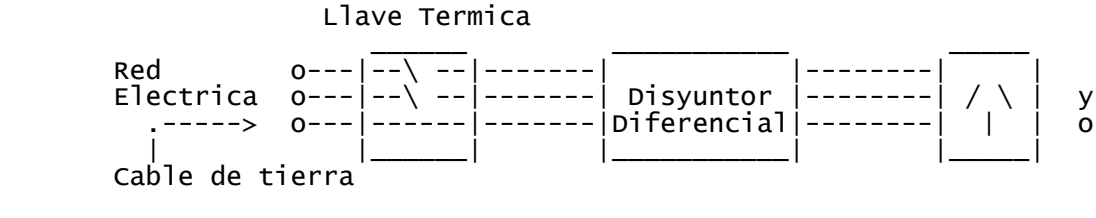
El modelo final del banco sera algo asi :



3.3.3 Instalacion electrica

La instalacion electrica del laboratorio de electronica es uno de los puntos mas importantes de todos, ya que una buena instalacion puede salvarte la vida, en caso de que recibas una descarga electrica de alto voltaje, como sucede en el caso de los tubos de rayos catodicos de un televisor o monitor de computadora.

Una vez que tengas un banco o lugar fijo para trabajar, haz una instalacion sencilla y barata, pero segura y de calidad como la de la figura:



Es importante, aunque no indspensable, que el cable 'vivo' de la red electrica este conectado al borne derecho del enchufe o toma. Para saber cual es haz lo siguiente: necesitaras un destornillador neon o 'buscapolo', o en su defecto el tester. Si actuas con el buscapolo, colocas tu dedo pulgar en

el borne metalico del cabo y tocas con la punta los cables hasta que alguno haga prender el neon del destornillador, si es asi, ese es el cable 'vivo'.
Si actuas con el tester, primero colocas el selector de posiciones en corriente alterna (dira AC o tendra el simbolo ~), en el mayor valor que haya. Tomas la punta negra del tester [supongo que habras leido el manual y sabras como se conecta, si no es asi, la punta negra va al terminal indicado con 'COM' o el que tenga color negro], y la agarras con tu mano derecha, y con la otra pruebas los cables hasta que obtengas alguna medicion en el display del tester.

ES MUY IMPORTANTE QUE NO TOQUES NINGUNA OTRA PARTE METALICA DEL BUSCAPOLO O QUE TOMES LA PUNTA DE PRUEBA CON TU MANO DERECHA, YA QUE EN CASO DE DESCARGA ELECTRICA ACCIDENTAL, LA CORRIENTE CIRUCULARA POR EL LADO OPUESTO A TU CORAZON Y NO PROVOCARA LA FIBRILACION O PARO DE TU CORAZON.

El cable de tierra (si existe) normalmente estara marcado con una cubierta verde, con una linea amarilla.

[sucede en casi todas partes, ya que se acordo por convecion internacional]

3.3.4 Toma de tierra

La toma de tierra es muy importante en la instalacion electrica. Si no dispones de ella, no hay problema.
Consigue una rejilla metalica (mejor si es de cobre o bronce) de unos 30 cm por 30cm, y la entierras en un recipiente de bronce, aluminio o cobre; relleno con carbonilla. Luego entierras el recipiente en la tierra (es mejor si la mantienes humeda. La rejilla debe ser conectada con un cable de buena seccion (1 o 2mm) a los bornes de tierra de cada uno de los tomacorrientes.
Si llegas a manejar circuitos integrados CMOS, o similares, es recomendable que tu mismo te conectes a tierra mediante una pulsera de cobre o plata.

3.3.4 Sistemas de proteccion

Te aconsejo que la llave termica sea de unos 10 o 15 amperes, lo que es suficiente para un laboratorio tipo. Tambien es muy importante la instalacion del disyuntor diferencial, que corta el suministro de energia en caso de descarga electrica.
Si llegas a manipular el interior de un televisor o monitor de PC, es muy recomendable que conectes a este con un transformador de aislamiento con relacion 1:1 (iguales bobinas en cada nucleo), que ademas de reducir el peligro, permite la utilizacion de algunos instrumentos que solo se pueden utilizar de este modo.

3.3.5 Complementos

Los complemenos practicos que puedes instalar son:

- Una serie de ganchos dispuestos a lo largo del banco de trabajo, que ayudara a recoger todos los cables y latiguillos.

- Una instalacion de antenas colectivas de TV, AM y FM (puedes utilizar la misma de tu casa)

- Un autotransformador regulable, que te permitira comprobar como funcionan los equipos a diferentes tensiones.

4. El soldador, la primera herramienta

La herramienta que proporciona el calor necesario para fundir el hilo de soldar (estanio), es el soldador, que juega un papel muy importante en el laboratorio de electronica. Un soldador se compone de los siguientes elementos

- Mango : Permite agarrar al soldador sin quemarte la mano (el soldador tiene unos 400 §C de temperatura). Los mejores soldadores en este sentido son los que tienen mango de madera, que conduce el calor debilmente.

- Resistencia interna : Es la resistencia encargada de producir el calentamiento de la punta del soldador.

- Punta de soldar : Es una punta que trasnfiere el calor al hilo de soldar.

Generalmente, las puntas metalicas son mas baratas, pero son de menor calidad. Si quieres una buena punta (no es muy cara tampoco), las mejores son las llamadas 'ceramicas'. estas son de color blanco y pueden durar 5 anios si sabes cuidarlas.

Esta de mas decir que evites todo contacto del cable del soldador con el cuerpo metalico o la punta, ya que la cubierta aislante se derretira y puede provocar un cortocircuito (cosa que no queremos)

4.1 Tipos de soldadores

En el mercado electronico hay varios tipos de soldadores. Los mas comunes son estos que listo a continuacion:

- Tipo recto o lapiz : Tiene una forma alargada que depende que la potencia del soldador. La temperatura normal que alcanza es de unos 400 grados.

- Tipo recto con regulacion de temperatura : Es muy similar al anterior, pero incorpora una caja de mando con el objetivo de regular la temperatura del soldador. Las temperaturas muy altas pueden daniar a muchos componentes.

- Tipo recto de baja tension : Similar al primero, pero incorpora un transformador que reduce la tension a unos 24 voltios.

- Tipo pistola : Tiene una forma como la que indica el nombre, es muy voluminoso y de una potencia muy elevada.

4.2 Eleccion del modelo adecuado

El mejor modelo de soldador que puedes elegir es el primero de la lista, es decir, el tipo 'lapiz'. Aunque te parezca mejor el modelo con regulador de temperatura, no compres ese....porque en proximas entregas aprenderas a construir tu propio regulador.
Los soldadores tambien se diferencian segun la potencia que posean. Se clasifican en :

- Baja potencia : Menor que 30 vatios (watt)
- Mediana potencia : Entre 30 y 60 w
- Alta potencia : 60 w en adelante

La potencia recomendada para tu soldador, esta entre los 40 y 50 W, ya que no son modelos muy caros y se desempenian de manera excelente.
Los soldadores de alta potencia son muy caros e inutiles, porque el unico uso que tienen es soldar chasis metalicos, cosa que se puede hacer con tornillos o algo similar.

4.3 Puntas

Como ya dije, las mejores puntas para soldador son las puntas ceramicas, que se calientan mas rapido y tienen una vida util mas larga.
Tambien se pueden encontrar putas con formas curvas, algunas mas anchas y de mayor seccion, pero la punta que sera de mayor utilidad es la de soldadura DIL (dual in line), que permite soldar/desoldar circuitos integrados que vengan en este formato.

4.4 Control de la soldadura

Una buena soldadura se realiza con una temperatura que oscila entre los 300 y 400 grados centigrados. Para que se pueda elegir la temperatura de trabajo del soldador, existe un tipo de soldador que viene con un control de temperatura externo, que regula desde el apagado del soldador hasta la maxima potencia que alcance.
Como ya dije, en una proxima entrega tendras el circuito de un regulador de temperatura de muy buena calidad para tu soldador.

4.5 Mantenimiento

Para que tu soldador tenga una vida util larga [y para que no desperdicies el dinero y tengas para tomar unas cervezas], es muy aconsejable tener un soporte para el soldador, que evite que quemes la superficie del banco de

trabajo, maltrates la punta del soldador, y quemes la cubierta del cable de alimentacion.
  Lo puedes fabricar tu, arrollando un alambre de acero en forma de espiral y colocandolo sobre un soporte, de modo que la punta quede suspendida en el aire. Esto ayuda a que la punta no toque superficies que puedan corroerla, y tambien a que el soldador se ventile por conveccion del aire, preveniendo una ruptura de la resistencia interna por un sobrecalentamiento.
  Si sucede esto, es muy facil conseguir repuestos para la resistencia y soldarla en el interior del soldador.

5. Tester o polimetro

  Este es sin duda el instrumento mas util y el mas utilizado en un taller de electronica. Existen dos tipos: analogico y digital, cada uno con sus ventajas y desventajas. Normalmente pueden medir tensiones en continua y alterna, intensidades en continua y alterna, y resistencias.

5.1 Tester analogico

  Este modelo se compone de un selector de escalas y un galvanometro (es la aguja con escala). Normalmente el aparato viene con su bateria incluida y los bornes o latiguillos para realizar las mediciones.
  Las ventajas de este modelo es que es un aparato regularmente barato :), y casi siempre mide tensiones mayores que los testers digitales.
  Como ya debes saber, el borne rojo corresponde al positvo, y el negro al de negativo o masa.

5.2 Medida de resistencias

  Teniendo en cuenta que el funcionamiento como voltimetro y amperimetro es demasiado sencillo como para explicarlo, voy a ir directamente al apartado de medida de resistencia, es decir que se utiliza el tester como ohmetro.

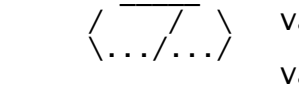
  [mira el manual de tu tester para ver las funciones que tenga. Te indico que para medir voltajes, el instrumento se coloca en paralelo con el circuito; y para medir intensidad usandolo como amperimetro, se debe abrir el circuito y colocarlo]

5.3.1 Escalas de medida

  Seguramente tu aparato tiene una escala de medida que oscila entre los 0 ohm y los 2000 Kohm (K"1000). Seria imposible de leer esto en una sola escala, asi que los fabricantes dividen los margenes en saltos de 10 o 20. Por ejemplo :

  1O, 1OO, 1K, 1OK, 1OOOK (puede llegar a mas)
       o tambien
  2O, 2OO, 2K, 2OK, 2OOOK

  La escala indica la resistencia maxima que puede medir en cada posicion.
  Para leer la medida, debes multiplicar el numero que indica la aguja, por la escala en que estas midiendo. Es mas facil de entender con un ejemplo :



  Valor en galvanometro : 5O   Escala : 2K

  Valor real " Valor galvanometro * nro de ceros de la escala

  Valor real " 5O * 1OOO (2K"2OOO , son 3 ceros, o sea 1OOO)

  Valor real " 5OOOO ohm " 5O K

5.3.2 Puesta a cero

  Cada vez que cambias de escala, o realizas la primera medida, debes realizar una puesta a cero. Esto se realiza conectando las 2 puntas entre si, y actuando sobre el potenciometro que tendra un rotulo que dice "Ohm adj" o "Set zero". Debes regular esto hasta que, manteniendo las 2 puntas conectadas, la escala indique el O.
  La puesta a cero es una medida de seguridad que evita el gasto inutil de

la bateria del tester.

5.3.3 Realizacion de la medida

Al medir una resistencia en un circuito, debes desconectarla del mismo, para que no interfieran otras que puedan estar conectadas con esta. Tambien debes desconectar la alimentacion del circuito.

Tambien debes tener la precaucion de no tocar las dos patas de la resistencia con la piel (puedes tocar una), para que la resistencia de tu cuerpo no interfiera en la medida.

5.3.4 Medidas de continuidad

Algunos modelos tienen incorporado un zumbador que produce un pitido cuando se comprueba continuidad (2 o mas puntos unidos).

Si este no es tu caso, puedes comprobar continuidad colocando la escala de tu tester el el mayor valor y efectuar la prueba. Si la lectura es 0 o cercana al 0, hay continuidad. Si no es asi, se debe leer una resistencia muy alta o infinita.

5.3 Prueba de diodos y transistores

Todos los procedimientos de medida son tambien aplicables y de una gran utilidad para la comprobacion de diodos rectificadores y transistores. En el caso de los diodos, debe existir una resistencia minima entre sus terminales, cuando la corriente lo atravieza desde el anodo al catodo (polarizacion directa); y otra resistencia muy alta cuando es al contrario.

Para el caso de los transistores, el procedimiento es muy similar, siempre teniendo en cuenta si el transistor en NPN o PNP (ya aprenderas que es esto). Para realizar las medidas se toman referencias entre base y emisor, y base y colector. Hay algunos modelos que disponen de un zocalo exclusivo para esta funcion.

Siempre hay que tener en cuenta la polaridad de las puntas del tester, respetando la serigrafia del aparato y las puntas.

5.4 Medidas de tension

Normalmente, todos los testers disponen de una funcion que lo hace trabajar como voltimero. Casi siempre las escalas son en escalas de 10 o 20, igual que la escala de resistencias.

La polaridad solo deber ser tenida en cuenta cuando midas corriente continua (DC), porque si inviertes la polaridad en un tester analogico, casi siempre el galvanometro se arruinara. Este efecto no ocurre en los testers digitales, poruqe el instrumento seniala si la polaridad es correcta o no.

5.5 Resistencia interna

Un factor importante que determina la calidad del tester, es la resistencia interna que posee, porque cuando es usado como voltimetro, pueden haber medidas con un margen de error muy grande si la resistencia es inadecuada con el tipo de medicion.

La resistencia interna generalmente esta expresada en ohmios por voltio. (Ohm x V), o sea que la magnitud varia segun la escala que selecciones. Si vas comprendiendo bien, mientras mas alta es la escala de medida, mas alta es la resistencia. Por lo tanto en las medidas de las escaas mayores se puede apreciar un ligero margen de error.

Mientras mas alta sea la resistencia del tester, mas precisa sera la medicion.

5.6 Medidas de intensidad

El otro conjunto de medidas que resta, es el de la corriente electrica, tanto en continua como en alterna. La realizacion de la medida es muy similar a las otras, salvo que debes tener en cuenta que para medir corrientes, debes abrir el circuito electrico e intercalar el instrumento.

Tambien debes prestar atencion a las indicaciones de los fusibles del tester, que normalmente oscila entre uno de 200 o 400 mA (miliamperios), y 10 o 20 Amperes. No debes sobrepasar la corriente que indica este ultimo, porque casi siempre el tester se arruina y no hay remedio para eso. Ademas,

la punta positiva del tester debera estar hacia el punto mas positivo del circuito. No debes tener en cuenta esto si la medida es en AC.

5.7 Variantes

Es recomendable que la posicion del tester al realizar la medida, sea una posicion cuasihorizontal, para que la gravadad no afecte sobre la aguja y resortes del galvanometro, y asi obtener medidas erroneas.

5.8 Eleccion del tester

Con un tester que cumpla las funciones de voltimetro, ohmetro y amperimetro, es mas que suficiente. Aunque ten en cuenta que tambien hay algunos que miden decibles, y dentro de los digitales, estan los que miden algunas frecuencias y tambien miden temperatura y capacidad en los condensadores o capacitores.

5.9 Consejos para la medida

En esta tabla estan las escalas recomendadas que debes tener en cuenta cuando adquieras un tester.

ESCALA	Minimo	Maximo
VCC	0.5 V	1000 V
VCA	2 V	1000 V
ICC	0.1 mA	10/20 A
ICA	10 mA	5/10 A
OHM	100 ohm	>= 100 Kohm
CAPACIDAD	150 nF	>= 30 UF
FRECUENCIA	-	500 Hz

5.10 Escalas no lineales

Los mejores testers son aquellos que tienen las escalas en una progresion de potencias de 10, o progresiones de 20. Para hacerte entender mejor, te dire que las potencias de 10 van en una secuencia de: 10 - 100 - 1000. Las progresiones de 20 siguen la secuencia 20 - 200 - 2000.

Queda en tu decision elegir cual prefieres, pero no es recomendable que compres uno que no cumpla con estas normas, porque el margen de error de las medidas puede ser muy grande.

6. Tester digital

Dentro de la gama de modelos y variedades de testers, hay algunos que tienen un funcionamiento totalmente digital. Esta clase de instrumentos permite que las medidas sean mas precisas. Son un poco mas caros que los testers analogicos, pero su mayor variedad de funciones hace valer su precio.

6.1 Principio de funcionamiento

El funcionamiento de estos testers es totalmente diferente al de los analogicos. Para no comlicarte con formulas y matematica, solo sabremos que estos testers realizan la medida comparando la magnitud que se mide con otra referencia exacta que contiene en su interior.

Aunque los principios de funcionamiento varian de un modelo a otro, describo el funcionamiento mas comun, para que te hagas una idea aproximada.

6.2 Medidas de tension

Supongamos que el tester tiene internamente una tension de referencia de exactamente 10 V.

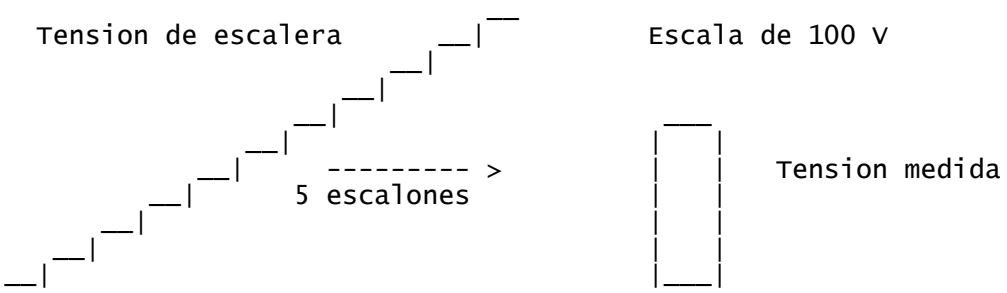
Un circuito incluido dentro del aparato genera una "tension en escalera", de forma que comienza en 0 V y acaba en 10 V. Supongamos que por cada escalon representa un incremento de 0,1 V, por lo tanto existiran 100 'escalones'.

La tension a medir se compara constantemente con la tension en escalera, de forma que mientras sea mayor esta ultima, el circuito productor de la misma va aniadiendo escalones. Cada vez que se incrementa la escalera con un escalon, otro circuito del tester va contando 'un paso'.

En el momento en que la tension de la escalera iguala o supera a la que proviene del exterior, el contador deja de sumar pasos y en ese momento, la cantidad que contiene en su memoria se transporta al indicador digital. Si por ejemplo, ha llegado ha contar 56 de los 100 pasos, la tension por supuesto sera de 5.6 V.

Si la escala es muy grande, una vez que se ha llegado al numero entero de escalones, hay 2 opciones: incrementar el numero de escalones, o incluir sub-escalones.

Todo este proceso solo toma unas decimas de segundo, o incluso menos; por eso es que la medicion con un tester digital es mas precisa y rapida que la que se realiza con uno analogico.



Si el tester es de 2 1/2 digitos, tendria una indicacion como esta :

6.3 Medidas de intensidad

Entre las ventajas de los testers digitales, esta la de poder invertir la polaridad de la medicion sin daniar el instrumento. O sea que para medir tensiones continuas, se pueden colocar la punta positiva del tester en el polo positivo de la tension, o viceversa, sin alterar el instrumento o la medicion. Normalmente permiten una medicion de corrientes mas altas que los testers analogicos.

6.4 Modelo mas adecuado

Uno de los factores de mas peso a la hora de elegir un tester, sin duda es el numero de digitos del display. Tambien es el factor que mas tiene que ver con el precio del aparato.

Como saber cuantos digitos son necesarios ? Para dar a entender este concepto, tomemos como ejemplo la tension de la red electrica (aqui en argentina es de 220 V, no cuanto sera alla en la madre tierra). Si medimos la tension con un aparato de 5 digitos, la medicion podria ser de 218.96 V. Con uno de 3 digitos la lectura seria de 219 V.

Cual de las medidas es la mas util? En la inmensa mayoria de los casos, sera suficiente con la lectura de 3 digitos, mientras que la de 5 digitos nos aporta una informacion suplementaria, que en la practica puede elimenarse. Por lo tanto, el exceso de digitos no tiene ninguna utilidad practica.

Es conveniente el empleo de menos de 3 digitos ? Depende de los valores de fondo de escala que deseen medirse. Con dos digitos, la maxima lectura que se puede obtener es 99, lo que sera insuficiente en la mayoria de los casos.

Puesto que con 2 digitos solo se pueden obtener 100 lecturas distintas (00 a 99), la precision general del instrumento (independiente de la circuiteria), sera del 1%. En cualquiera de los casos, y suponiendo que la precision de los circuitos del instrumento es buena, las cifras de precision son mas que suficientes.

6.5 Otras soluciones

Parece que se crea un gran hueco [gran] entre los display de 2 digitos y los de 3. Para cubrir este hueco se disenio un tipo de display que incorpora un primer digito, que puede tomar los valores de 0 o 1, de forma que si pone delante de un indicador de 2 digitos, pueden hacerse lecturas desde 000 hasta 199, es decir 200 lecturas diferentes, que son el doble de las de antes. Este tipo de display, se conoce como display de 2 1/2 digitos.

Asi, vemos que con la inclusion de este digito se amplia notablemente el margen y precision de la medida, ya que con 2 1/2 digitos se consigue un valor de fondo de escala de 199, con una precision de 0,5 %

6.6 Seleccion automatica

Muchos testers emplean un sistema de seleccion automatica de la escala. Este sistema cuida el instrumento de medir una magnitud en una escala que no es la de la magnitud misma, por ejemplo querer medir una tension en la escala de las resistencias.

Este sistema se hace mediante el empleo de comparadores, que se realizan con amplificadores operacionales. Puedes encontrar el esquematico de los comparadores en el archivo 'tester.gif'.

7. Elementos auxiliares

No creo que haga falta detenerse a explicar las herramientas que faltan para completar nuestro laboratorio de electronica, que son las herramientas de utilizacion electronica, asi que solo las nombrare.

- Alicates de corte
- Alicates de pelar
- Alicate punta recta
- Alicate punta en angulo
- Destornilladores
- Taladro miniatura para circuitos impresos

8. Cables de conexion

Las conexiones de los paparatos de medida o circuitos con el exterior, se realiza mediante cables especiales, cada uno con su tipo de conector apropiado. El sistema mas usado es el de la clasica 'banana', con una clavija tipo cocodrilo en una de sus puntas.

El cable, puede ser un solo hilo rigido, o tener muchos hilos finos y de una cubierta plastica que sirva de aislamiento.

Los aparatos tales como los osciloscopios y generadores de senial tienen sus conectores de entrada y salida tipo BNC hembra, por lo que a este conector solo se conectan BNC macho.

Los testers casi siempre vienen con sus propios conectores y puntas de medida. Algunos tambien traen unas pinzas de cocodrilo para adaptarle.

8.1 Tipos

El hilo o cable de conexion es el componente mas simple que hay, y se encuentra en casi todos los equipos electronicos. Se pueden realizar todas las conexiones que se deseen, siempre que la seccion del hilo soporte la corriente que lo atraviese.

Para poder distinguir los 2 tipos, diremos que el hilo es aquel cable sin aislante, y el cable es un hilo rigido o muchos hilos finos, cubierto con una cubierta plastica que sirve de aislante.

Existe tambien un tipo de cable llamado 'mallado' o 'apatallado', que recibe este nombre porque contiene una malla metalica que sirve de filtro frente a las perturbaciones electricas del exterior. Se usa para conexion de equipos de sonido e instrumentos de medida.

9. Simbologia electronica

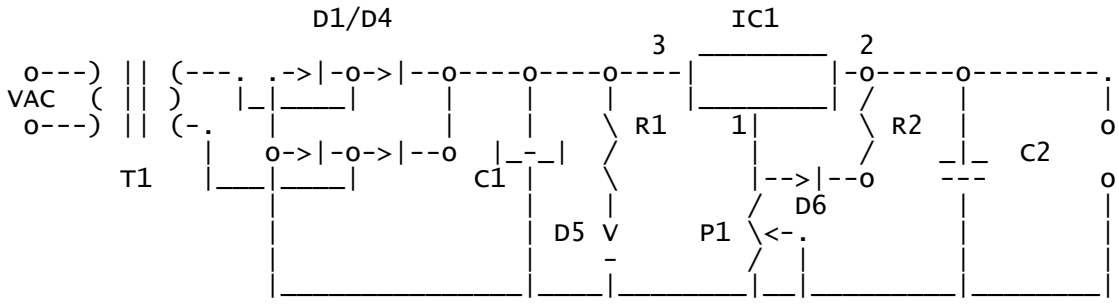
Para que te sea mas facil identificar los simbolos, los podras encontrar en 'simbolos.gif'. [tambien para no tener que dibujar en ascii, porque dibujo muy mal :)]

10. Montajes

Para aprender a construir circuitos impresos, lee el articulo 'Montaje de circuitos electronicos', de IMC68000, en SET 22, articulo 0x08.

Fuente de alimentacion regulable

Si no entiendes el ascii, no te preocupes. Mira el archivo 'fte317.gif' para un esquematico como Dios manda.



Los componentes son :

T1 : Tranformador de acuerdo a la tension de red de el lugar donde vivas.
Voltaje de salida maximo recomendado : 30 V
Corriente de salida maxima [maxima, no recomendada]: 1.5 A
D1/D4 : Diodos 1N4004
C1 : Capacitor electrolitico 4700 uF (o 2x2200 uF), 50 o 63 V
R1 : Resistencia 1000 ohm (1Kohm) 10%
R2 : 240 ohm 10%
D5 : Diodo LED comun (no compres de los intermitentes, porque perjudica la continuidad de la corriente)
D6 : Diodo 1N4004
P1 : Potenciometro 5 K Ohm (debes usar este valor solamente)
C2 : Capacitor de tantalio 33uF (o valor mas cercano)
IC1 : Regulador LM317T

Puedes encontrar un circuito casi identico a este en
http://www.hut.fi/Misc/Electronics

Proxima entrega :

- Introduccion a los semiconductores
- Diodos
- Transistores - Tipos
- Diac / Triac
- El transistor como amplificador
- Osciladores a transistor
- El transistor en el campo digital

EOF

```
-[ 0x07 ]--------------------------------------------------------------------
-[ Curso de Electronica (II) ]-----------------------------------------------
-[ by elotro ]----------------------------------------------------------SET-33--

               @%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%
               % Curso De Electronica  -  Segunda Entrega @
               @%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%

            Buenos dias/tardes/noches a l@s lector@s de SET ! !

 Espero que hayan disfrutado la primera parte del curso, y que hayan
 aprendido nuevas cosas del mundo de la electronica. Si es asi, disfruten
 este articulo como todos los otros, y si no es asi, enviame todas tus
 dudas/sugerencias/criticas/cualquiercosa, a elotro.ar@gmail.com

 Dentro de la segunda entrega del curso de electronica, abordaremos el tema
 de los semiconductores, su funcionamiento, tipos y aplicaciones en el mundo
 de la electronica analogica y digital.

 He notado [y me han dicho] que no dibujo bien en ascii,
 asi que tienes total libertad de pedirme los diagramas de este y/o
 otros articulos mios, en forma grafica.
 [bah, no me dijieron eso. Me dijieron directamente que dibujo mal]

 Los formatos preferidos son .gif .jpg y .rle.
 Otros formatos pueden tardar muuuuucho en llegar.


 Temas de la segunda entrega :

     1.  Conductores, aislantes, semiconductores y superconductores.
     2.  El diodo semiconductor
     3.  Transistores bipolares
     4.  Transistores unipolares
     5.  El tiristor
     6.  El triac
     7.  Circuitos integrados
     8.  El transistor como amplificador
     9.  Osciladores a transistores
    10.  El transistor en el campo digital
    11.  Montajes practicos
    11.3.  Baliza electronica


 Si no te interesa el proceso que sucede en el interior de un semiconductor,
puedes saltear el capitulo 1.

  [tambien podrias leerlo, y decirme que te parece, para motivarme a seguir
   escribiendo y mejorando cada dia mas la calidad de los articulos ]

  Comencemos...


  @%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%
  % 1 - Conductores, aislantes, semiconductores y superconductores      @
  @%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%

  La estructura cristalina de los metales contiene muchos electrones libres,
de modo que todos los metales son conductores de la electricidad. La
resistencia de los conductores no es la misma, de modo que la plata o el
cobre presentan menor resistencia electrica que el mercurio o el hierro.

  Una particularidad de los metales, es que sus resistencia electrica es
menor mientras menor sea su temperatura. Esto tiene una explicacion fisica,
que se manifiesta en que las moleculas de un compuesto se agitan mas mientras
mayor es la temperatura. Si la temperatura es muy baja, las moleculas no
presentan agitacion y los electrones tienen que realizar menos esfuerzo para
moverse a traves del conductor.

  [los electrones no toman anabolicos]

  Teniendo en cuenta esto, se han desarrollado unos compuestos, en su mayoria
de tipo ceramicos, que al someterse a temperaturas de unos -220 §C, presentan
una resistencia nula o casi nula. Son los llamados superconductores.
```

[puedes averiguar mas sobre esto buscando 'condensado de einstein-bose' en alguna publicacion cientifica o en internet]

[como en SET siempre hay espacio para el humor, contare un viejo chiste sobre los superconductores :
Que le dijo un superconductor a otro ?....
'Tengo frio, no resisto mas.']

La mayoria de los no metales son buenos aislantes, porque contienen pocos electrones libres, y por lo tanto, una resistencia electrica muy alta.

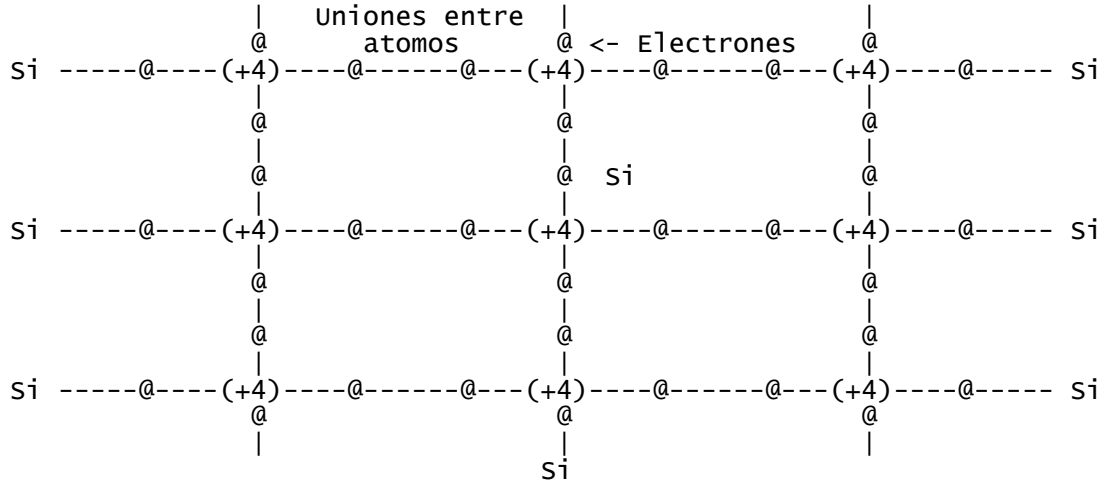
Semiconductores [condutor de auto borracho = semiconductor]

En un conductor normal, la corriente electrica se manifiesta mediante el movimiento de las cargas negativas (electrones), mientras que un semiconductor intervienenen el movimiento de los electrones y el de las cargas positivas (huecos).

Ademas, a los semiconductores se les introducen atomos de otros elementos, denominados impurezas, de forma que la corriente se deba a los electrones o a los huecos, dependiendo de la clase de impureza introducida.
El Germanio(Ge) y el Silicio(Si) constituyen los principales elementos en la construccion de semiconductores.

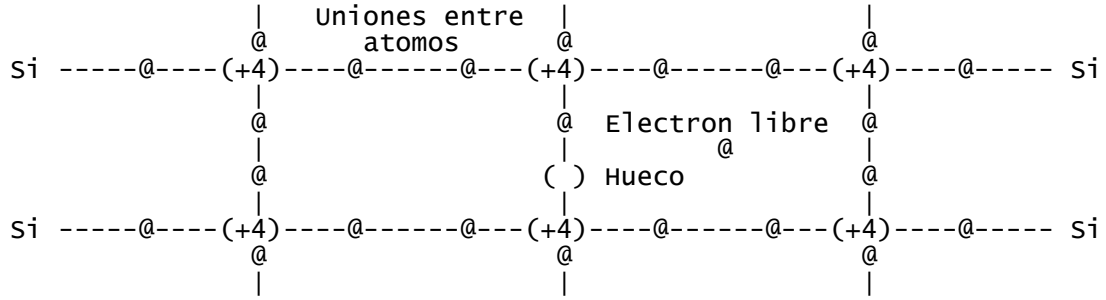
Ahora llega el momento que preguntas :
Como es la estructura molecular de un semiconductor de silicio ?

[por supuesto que nunca harias esa pregunta]



En un semiconductor de este tipo no existen huecos ni electrones libres

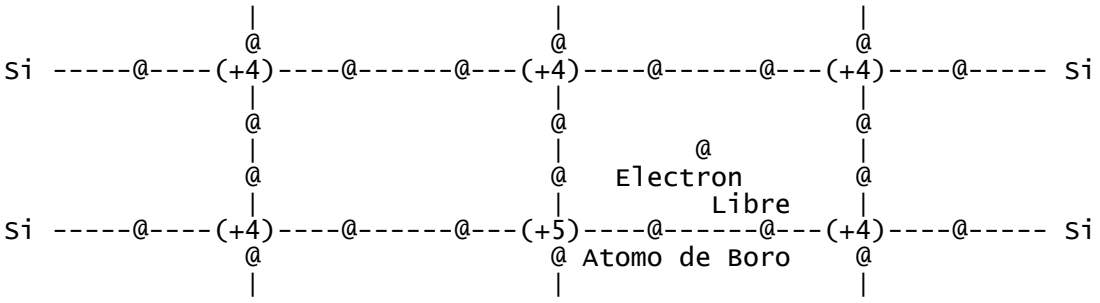
Los +4 simbolizan la carga de los nucleos y las @ son los electrones, unidos debilmente a estos. La fuerza que los mantiene unidos es que cada electron esta unido por lo menos a un nucleo. A bajas temperaturas esta es la estructura que toma el silicio.
En cambio, a temperatura ambiente (20-25§C), algunas de las uniones se rompen, dejando huecos y electrones libres.



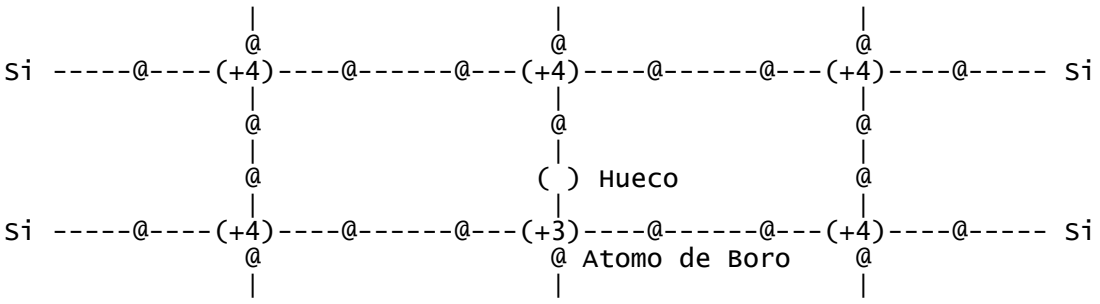
Cuando el electron se retira y deja un hueco, le resulta muy facil al electron vecino ocupar ese hueco, dejando otro y siguiendo el proceso. De esta manera es como el huecon contribuye al paso de la corriente.

Cuando se aniaden atomos de otro elemento, el semiconductor se transforma en un semiconductor impuro. Las impurezas generalmente son de arsenico o boro. Si los atomos de las impurezas donadoras contienen 5 o mas electrones libres, el atomo posee carga negativa, y se denomina una impureza de tipo 'N'. Si los electrones son 3 o menos (pueden ser 0), el atomo tiene carga positiva y la impureza se denomina de tipo 'P'.

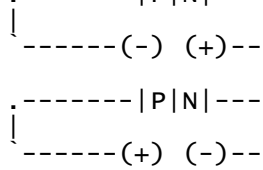
Semiconductor Tipo N



Semiconductor Tipo P



Uniones semiconductoras :



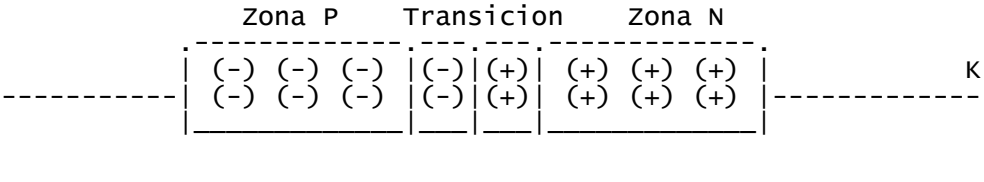
Polarizada directamente

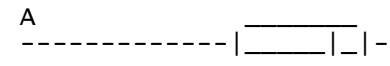
Polarizada inversamente

[es aburrido pero interesante no?]

2 - El diodo semiconductor

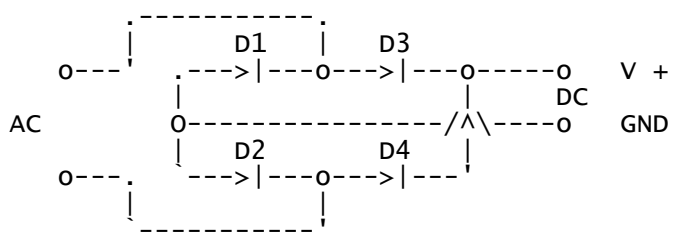
Un diodo semiconductor se compone de una union P-N, a la que se le aniaden terminales metalicos en cada uno de sus extremos y una capsula para alojarlo. El terminal que corresponde al anodo es el de la union P y el catodo es la union N.





El catodo se marca con una linea cercana al terminal.

El diodo permite el paso de corriente solamente cuando se conecta el polo positivo en el anodo, y el positivo en el catodo. Esta propiedad puede ser usada para trasnformar la corriente alterna en continua. Esto se denomina rectificacion.



Cuales son los tipos de diodos mas comunes ?

Diodo Rectificador : Es el tipo mas comun, que se usa en etapas rectificadoras o conmutacion en circuitos de CC o baja frecuencia. Existen diferentes presentaciones y capsulas, dependiendo de la potencia que deben disipar. Cualquier diodo rectificador se caracteriza por los siguientes factores.

- Corriente maxima directa (If)
- Tension derecta maxima (Vd) a una If determinada
- Tension maxima de pico (Vrwm)
- Tension inversa maxima de pico (Vrrm)
- Corriente maxima de pico (Ifsm)
- Corriente inversa maxima de pico, a una determinada Vrrm
- Potencia total (Pt)

OJO :
Estas caracteristicas difieren mucho de modelo a modelo, y deben ser tenidas en cuenta a la hora del disenio de circuitos.

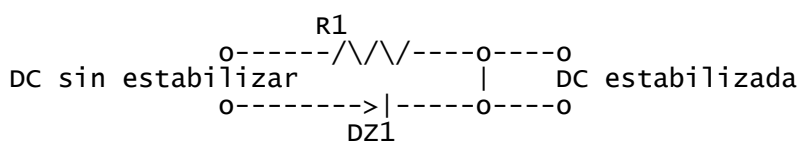
Diodos de senial : Se utilizan mayormente en circuitos transmisores o receptores de ondas de radio, o en circuitos digitales.

Diodos de alta frecuencia : Se emplean en circuitos que trabajan con frecuencias mayores a 1Mhz.

Diodos de conmutacion : Se caracterizan por ser capaces de trabajar con seniales digitales. Lo fundamental en estos diodos es el 'tiempo de recuperacion inverso' (TRI) que expresa el tiempo en que la union P-N transporta la carga electrica. Se consideran rapidos los diodos con un TRI de 500 nanosegundos en los modelos de media potencia, y de 5 nanosegundos en los de baja potencia.

Diodos zener : Se usan para estabilizar tension, usando una caracteristica de los semiconductores que se polarizan inversamente. Normalmente cuando se encuentran polarizados de esta manera [el catodo en el polo +], no permiten el paso de la corriente, o permiten el paso de una pequenia parte, pero al alcanzar una tension denominada 'tension zener', se produce un aumento de la cantidad de corriente, de tal forma que esta diferencia de potencial se mantiene constante.

Como se hace esto ?
Observa el circuito que sigue :



Diodos varicap : Utilizan la propiedad de capacitancia (un capacitor) de la union P-N para comportarse como un capacitor al polarizarlos de forma inversa. Se usan casi siempre en transmision y recepcion de radiofrecuencia.

Diodos LED : Son diodos que emiten luz, que puede ser visible o infrarroja.

Fotodiodos : Son diodos que reciben luz, visible o infrarroja.

% 3 - Transistores bipolares

El transistor es un semiconductor que puede gobernar a voluntad la intensidad de la corriente que circula en dos de sus tres terminales, a traves de una corriente mas baja aplicada al tercer terminal. Las dos primeras se llaman Emisor y Colector, el tercero se llama Base.
El efecto producido es una amplificacion de la corriente que puede ser usada en seniales de DC, radio, sonido, etc.

La palabra transistor tiene su origen de otras dos : TRANSfreresISTOR, que describe su aplicacion de transferencia de resistencia. Se desarrollo en 1948 por Shockley, Bardeen y Brattain; que realizaron investigaciones sobre los fenomenos electricos en la superficie de los semiconductores. Los 3 cientificos recibieron el Nobel de Fisica en 1956.

El funcionamiento interno del transistor puede ser descrito a partir de los conceptos del capitulo del diodo [si no lo leiste, ya estas subiendo]. A diferencia del diodo, el transistor tiene dos uniones semiconductoras, separadas por una capa de material. Suponiendo que se dispone de una estructura formada por dos zonas tipo N, sobre la que se diluye un material con exceso de electrones, como fosforo o arsenico; y entre ellos existe una delgada capa de material tipo P, con un elemento que tenga falta de electrones, tal como indio o boro.

El conjunto forma dos uniones : una N-P, y otra P-N; produciendose entre las 3 zonas movimientos de electrones, similares a los que se forman en el diodo.

Si ahora se aplica una tension exterior a la primera union N-P (emisor y base) con el negativo aplicado al emisor y el positivo a la base, se producira una circulacion de corriente entre las dos regiones. Aplicando una segunda tension a la union P-N (base y colector) se consigue que la primera tension sea atraida por la diferencia de potencial del colector. Suponiendo que la corriente es I, es 100 veces la de base Ib y esta es de 5mA, entonces I es de 500mA. Aqui se origina una amplificacion que se obtiene por el colector del transistor. El factor de amplificacion generalmente se denomina beta (B).

Como ya debes suponer, este es el funcionamiento de un transistor NPN. Los transistores PNP se construyen de la misma manera, pero se deben utilizar tensiones contrarias a las que se utilizan con los NPN.

% 4 - Transistores unipolares

Dentro de la familia de los transistores unipolares podemos diferenciar 4 tipos generales :

- Transistores FET
- Transistores MOS
- Transistores VMOS
- Transistores UJT (uniunion)

Transistores FET (Field effect transistor, transistor de efecto de campo)

Estos transistores realizan la funcion de control de la corriente, que es comun a todos los transistores, mediante una tension aplicada en uno de sus terminales.

Estan constituidos por una zona semiconductora tipo P o N que une dos de sus tres terminales, denominados Fuente (Source) y Drenador (Drain).
Sobre esta region existe otra de signo opuesto, conectada al tercer terminal

o puerta, formandose entre ambas una union P-N o N-P.

Todo el conjunto anterior esta realizado sobre un semiconductor del mismo signo que la puerta, que forma otra union con el canal y esta conectado electricamente al terminal que hace las veces de puerta. Cuando se aplica una tension entre el drenador y la fuente, una corriente circulara por el canal. Si ahora se aplica otra tension a la puerta, de forma que se polarizen inversamente las uniones P-N, se producira un estrechamiento del canal, aumentando la resistencia del mismo y variando en consecuencia la intensidad que circulaba por el.

La corriente de puerta sera muy debil por el hecho de ser una union polarizada de forma inversa.
Entonces, se observa que es posible variar la corriente que circula por el transistor con una tension variable de control, sin que sea necesario absorber corriente de ella.
Este efecto es muy similar alque produce una valvula de vacio triodo y puede ser empleado para amplificar.

Transistores MOS [nada que ver con el MacOS]

Otro transistor de efecto de campo el el denominado MOS o MOSFET, nombres que son formados con las iniciales de los elementos que lo componen :

- Una pelicula metalica (M)
- Un aislante de oxido de silicio (O)
- Una region semiconductora (S)

Se fabrican partiendo de un semiconductor tipo P sobre el que se esparcen dos regiones tipo N para formar la fuente y el drenador.
Sobre la superficie de esta estructura se coloca una fina capa de oxido de silicio (SiO_2), que es muy aislante. Sobre ella se coloca una capa metalica que actua como puerta.

Entre la fuente y el drenador existe un canal similar al tipo FET, cuya anchura o resistencia se controla mediante la tension de puerta, comportandose como un transistor FET.
Estos transistores producen una amplificaion de mayor poder que los FET, y se pueden montar en una disposicion similar a los transistores bipolares, es decir, Fuente comun, Puerta comun y Drenador comun; aunque la primera y la ultima son las mas utilizadas.

Transistores VMOS

Luego de la aparicion de los transistores MOS, nacio una nueva tecnologia : los transistores VMOS. El nombre nace a partir de la estructura geometrica de las regiones semiconductoras.
El disenio de un transistor VMOS se diferencia de un MOS convecional en el que la fuente, la puerta y el drenador se encuentran en la zona superficial del componente.

Los transistores VMOS tienen la estructura de una forma analoga a una piramide invertida grabada sobre el semiconductor.
Los empleos mas frecuentes son en transistores de potencia y/o conmutacion, realizando funciones de interruptor, gracias a la baja resistencia interna que poseen.

Transistores UJT

El nombre de transistor puede ser aplicado a cualquier otro semiconductor cuya resistencia pueda ser controlada con ayuda de una tension o corriente en uno de sus terminales.

Este es el caso de los transistores UJT (unijunction transistor). Estos dispositivos estan construidos internamente por una barra de un material semiconductor tipo N (generalmente silicio), de cuyos extremos se toman dos contactos que se denominan bases. (B1 y B2)

En un punto intermedio de la barra, se conecta un tercer terminal, de forma que el contacto establecido forme una union semiconductora, o union P-N, correspondiendo la zona P al nuevo contacto y la zona N a la barra.
Este terminal se denomina emisor y el contacto se comporta de forma

analoga a un diodo rectifcador.

Entre el emisor y cada una de las bases, la barra de silicio presenta una cierta resistencia electrica, simbolizadas generalmente como Rb1 y Rb2, siendo su suma igual a la resistencia electrica entre sus extremos (RT=Rb1+Rb2).
Para un transistor cualquiera, la relacion entre Rb1 y RT sera fija. Este cociente viene dado por el fabricante y se simboliza como n=Rb1/RT

Funcionamiento del UJT

Cuando entre las bases se aplica una cierta tension, por ejemplo, 15 V, por la barra semiconductora circula una cierta corriente que depende del modelo de transistor empleado.
La tension a lo largo de la barra variara desde 0V en B1 hasta 15V en B2, siendo la tension correspondiente al emisor n veces la aplicada entre bases.

Si por ejemplo, n tiene un valor de 0.6, la tension del emisor sera de :

0.6 * 15 = 9V

Puesto que el contacto de la barra puede verse como el catodo de un diodo, y sobre tal catodo hay una tension de 9V, mientras que al terminal emisor se le aplique una tension superior a esos 9V, el diodo no entrara en conduccion y por el emisor no circulara ninguna corriente. En esta situacion, se dice que el transistor esta en corte.

Cuando la tension exterior del emisor supera la tension del diodo interno (9.6V p ej) por dicho terminal comienza a circular corriente, comportandose el conjunto emisor-base como un diodo semiconductor normal, con lo que la tension pasa a ser de un valor muy bajo. Mientras dure esta transicion, se dice que el transistor esta en la zona de resistencia negativa, debido a que un aumento de la corriente le corresponde una disminucion de la tension.

Una vez que el transistor se ha estabilizado, la situacion permanece estable diciendose que se encuentra saturado. Esta situacion se mantiene hasta que la corriente de emisor baja a un valor para que el transistor vuelva a entrar en estado de corte.

5 - El tirirstor

El tiristor es un componente diseniado para realiar una funcion interruptora o una rectificacion conrtolada.
Su forma de trabajo es similar a la de un diodo, ya que unicamente permite el paso de la corriente en un unico sentido de circulacion, pero sin embargo, el tiristor se diferencia de este porque su conduccion esta regulada por uno de los cuatro electrodos que posee.

Estructura del tiristor

Esta formado por cuatro regiones p-n-p-n, formando la primera el anodo, la ultima el catodo y la region que esta en contacto con el se llama puerta.
La funcion de la puerta es la de disparo o puesta en funcionamiento del componente.

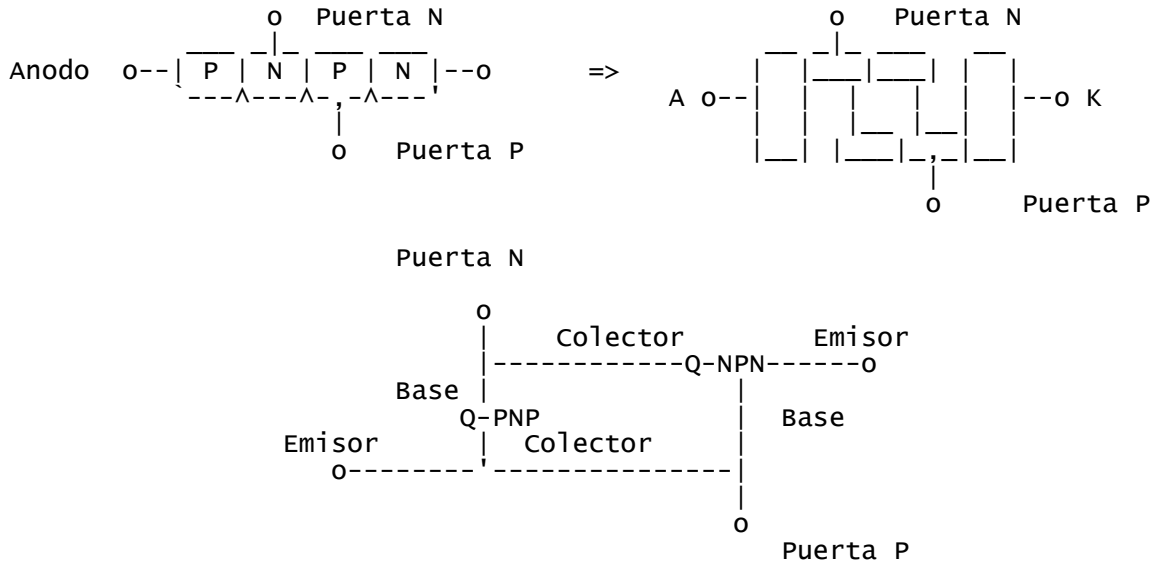
La puerta puede dividirse en dos partes, formando cada una de ellas un transistor. De esta manera, existe un transistor pnp constituido por el anodo y las dos regiones siguientes ; junto con otro npn formado por el catodo junto con las otras dos regiones.

Los transistores estan unidos en dos zonas :

- La base del pnp con el colector del npn
- El colector del pnp con la base del npn y al electrodo llamado puerta.

El circuito obtenido forma una estructura fuertemente realimentada ya que cualquier senial que se aplique por la puerta sera amplificada por el colector del npn, alcanzando la base del pnp y siendo amplificada otra vez por el colector de este.

En este caso, el componente entra en estado de saturacion y podra circular una corriente electrica entre el emisor pnp que forma el anodo, y el emisor npn que forma el catodo.



[No dibujo el transistor para no tener que hacer un ascii de 15 o 20 lineas. En su lugar, coloco una letra Q, que es la que se utiliza para senializar los transistores en los diagramas esquematicos]

Disparo de un tiristor

De todo lo que explique arriba, podras deducir que la entrada en conduccion del tiristor depende de la senial que se aplique en la puerta, pero no de la permanencia de la senial, porque la realimentacion del componente lo mantiene en estado de conduccion permanente.
Entonces, la senial se puede suprimir sin quitar al componente del estado de conduccion.

Las formas mas comunes de disparo de un tiristor son :

- Tension : Al aumentar la tension colector-emisor de un transistor se puede llegar a la ruptura por avalancha del mismo. En este momento se llega a una situacion similar a la comentada por la realimentacion interna del tiristor.

- Variacion rapida de la tension : Si la tension anodo-catodo varia bruscamente se produce una transmision de la variacion hacia el interior del componente, haciendo que entre en estado de disparo.

- Temperatura : El efecto de la temperatura sobre un transistor es el de aumentar la corriente de deriva del colector. En el momento que se alcance la corriente necesaria para iniciar la regeneracion, el tiristor entra en estado de disparo.

- Disparo por la senial de puerta : Esta es la forma mas comun de disparo, que es la que esta explicada arriba.

- Luz : En el caso de los fototiristores se produce un disparo con la luz incidente (similar a las fotoresistencias o fotodiodos)

Control de la corriente

Observen que a pesar de que los tiristores tienen caracterisiticas muy similares con los transistores, en el caso del control de la corriente, las diferencias son muy grandes.
Mientras que un transistor esta controlado por su base, en un tiristor no existe ningun control sobre la misma despues del momento inicial del disparo.

Entonces es preciso definir algun procedimiento de bloqueo del tiristor de forma que pueda estar controlado por cualquiera de los mecanismos de disparo.
Este procedimiento consiste en aplicar una tension inversa entre anodo y catodo, con el negativo sobre el anodo y el positivo sobre el catodo.

De esta manera el tiristor pasa a un periodo de tiempo denominado 'tiempo de bloqueo' o 'run-off-time'. La tension inversa podra ser desconectada, y el tiristor seguira manteniendose en la situacion adquirida.

Aplicaciones

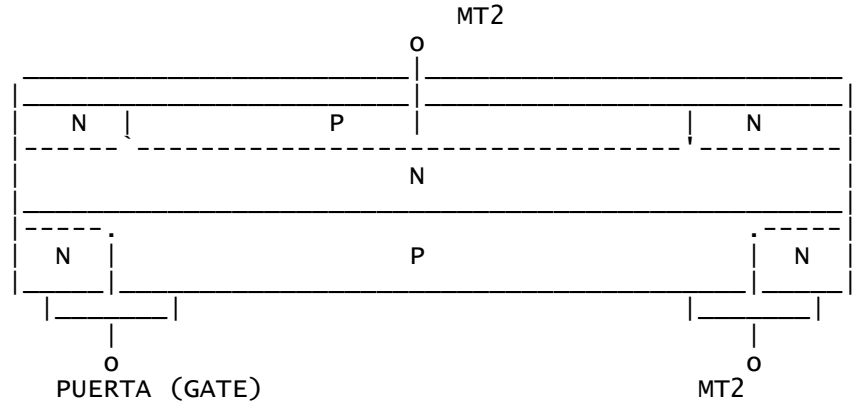
Las aplicaciones mas comunes del tiristor se extienden desde la rectificacion de corrientes alternas, en lugar de los diodos convencionales hasta la realizacion de determinadas conmutaciones de baja potencia en algunos circuitos, pasando por los los onduladores o inversores que convierten la corriente continua en alterna.

6 - El triac

El triac es un semiconductor que nacio de la necesidad de disponer de un interruptor controlado, similar al tiristor, pero que se utilizara en aplicaciones de corriente alterna.
La palabra triac es una abreviatura del nombre en ingles del componente, Trio de AC, o triodo de corriente alterna.

Estructura del triac

La estructura interna del triac esta compuesta por dos sistemas interruptores, uno pnpn, y otro npnp, que estan unidos en paralelo.
Los electrodos MT1 y MT2 son los principales, que en este caso no llevan nombre de anodo o catodo, porque trabajan con AC.



Disparo del triac

El disparo del triac se realiza de una manera similar al tiristor, o sea, aplicando una tension al electrodo llamado puerta.
Las posibilidades a la hora de realizar el disparo son muchas, pero se pueden resumir en 2 principales :

+ Disparo por CC

  Aqui la tension de disparo proviene de una fuente de CC que es aplicada por una resistencia limitadora de tension, para proteger al componente. Se necesita tambien un elemento que sirva de interruptor de CC, como un interruptor mecanico, un rele, un tiristor, etc

  Este sistema el el mas empleado en los circuitos electronicos que son alimentados por CC [la mayoria].

```
               `-------'              PUERTA   |
   VCA                              .-----------TR1
                                    |          |
    ~                          S1 \           |
                                  _|_          |
                        VCC -------            |  MT2
    o-------------------._______|____________|
```

```
                [ No dibujo el triac para no
                  hacer un ascii mas grande, pero
                  debes saber que el triac se
                  simboliza con TR en los
                  diagramas esquematicos ]
```

+   Disparo por AC

El sistema de disparo por CA se puede realizar con el terminal de puerta conectado directamente a la tension de red, con una resistencia limitadora, o a traves de un transformador. Tambien se necesita un elemento interruptor que excite el terminal en el momento preciso.

```
              .-------.
    o---------| CARGA |---------------------------------.  MT1
              `-------'                       Puerta  |
   VCA                  o--.___        __.--\/\/\-------TRIAC
                               `)||('        R           |
    ~           VDisparo        (||)                     |  MT2
                                )||(  T1                 |
                 ~              (||)                     |    [ T1 es un
                        o--.___.)||(_                    |   transformador]
                                     `-                  |
    o---------------------------------^----------------'
```

  + El diac

Este es un componente muy utilizado para realizar el disparo de un triac. El nombre diac viene del ingles Diode AC (diodo de AC). El diac tiene una estructura similar al triac, pero no cuenta con el electrodo de puerta. En su lugar, el diac conduce cuando la tension en sus terminales supera el llamado 'punto de disparo'.

Se emplea en circuitos que realizan un control de fase de la corriente del triac, ed forma que solo se aplica tension a la carga durante un ciclo de la corriente alterna.

Estos sistemas son reguladores de luminosidad reguladores de velocidad para motores de CA y regulacion de calefaccion electrica.

```
                   .-------.
    o--------------| CARGA |---o-----------.
                   `-------'   |   Puerta  |  MT1
   VCA                         \    .------TR1
                        RV -->/    |       |  MT2
    ~                          \  DIAC     |
                                |---'      |
                               _|_         |
                               -.-         |
    o-------.__________________|___________|
```

```
@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%
% 7 -  Circuitos integrados                                                @
@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%
```

7 - Circuitos integrados

El desarrollo y la fabricacion de circuitos integrados (ic's desde ahora) es el mayor avance de la electronica de los ultimos anyos, y al parecer este avance no se detiene.

Pero que c@r@jo es un circuito integrado?

Es un componente de estado solido, que integra muchos mas componentes en

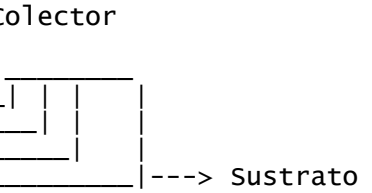
su interior, que son de un tamanio mas pequenio y que son mas estables y seguros.

[si nunca has visto un ic, abre el gabinete de tu pc y veras
muchas pastillas negras con pines a su alrededor. Esos son]

Sustrato semiconductor

Los ic's estan formados por un bloque de material semiconductor sobre el que se construyen las diferentes partes o componentes integrados.
En el sustrato se disponen impurezas de tipo N y tipo P, y forman los diversos componentes semiconductores. Si hace falta, tambien se pueden integrar zonas resistivas si son necesarias.

Ejemplo de un transistor NPN montado en un circuito integrado



Construccion de ic's

En la construccion de un ic se parte de un disenyo del esquema electrico, en el que se busca obtener un disenio compacto que ocupe la menor superficie posible.

Aparte de conseguir mayor portabilidad cuando se reduce el tamanio, se consigue mayor velocidad de procesamiento en el caso de los microprocesadores.

Como es esto ?
La electricidad es un movimiento de electrones, no?
Un electron es una particula que tiene masa, no?

[0.9107 * 10^ -27 g . Casi nada, pero tiene]

Los semiconductores tambien tienen protones, no?
Los electrones y los protones se repelen, no?
Llegaria mas rapido un electron que tiene que atravezar
2 metros de conductor lleno de protones que lo repelen,
o si tiene que recorrer 2 mm ?
Fin de la explicacion.

[Habia un mujer llamada Grace Hopper, que trabajaba en la computacion de la marina yanqui, que era conocida por regalar 'nanosegundos', un cable de unos 30 cm de largo, donde un electron tardaba aproximadamente un nanosegundo en ir de un extremo a otro]

Dejando de lado el tema de los electrones, continuemos con la integracion de resistencias. Como el sustrato ya es resistivo por si mismo [sirve de aislante], solo se necesita disponer de una pieza de sustrato del tamanio adecuado para realiar zonas resistivas.

En el caso de que se necesiten regiones aisladores que no esten cercanas al sustrato, se recurre a colocar zonas de polaridad inversa, o sea, se coloca una zona N si la que se desea aislar es P, o viceversa.

Tecnologias de fabricacion

En la actualidad existe varias tecnologias de fabricacion de ic's, pero podemos destacar dos principales .

+ Tecnologia Bipolar :

Se parte de una materia prima que es una barra de silicio de 2 a 3 cm de diametro, que se corta en obleas o rodajas de unas 30 micras de espesor (1 micra = 0.001 cm)

Sobre esta base se disponen capas de tipo P o N, segun el disenio del circuito. Para realizar esto, la oblea se coloca en un horno a unos 1200 §C, de manera que se forme una capa de dioxido de silicio (Si O2) sobre la oblea, que evita el el area cubiera se difuse.
A continuacion, esta capa se somete a un procedimiento fotografico donde seleccionan las capas de SiO2 que deben quedarm mientras el resto se elimina.
Este proceso se realiza cuantas veces sea necesario, y finalmente se realizan las conexiones hacia el exterior de la capsula con cobre u oro.

+ Tecnologia MOS

Se parte de los mismos principios de fabricacion de los transistores MOS. Una oblea de tipo N o P, donde se difunden 2 regiones de tipo N o P, segun corresponda.
Sobre esta capa se forma una de Si O2 igual que en la tecnologia bipolar y sobre ella se evapora una capa de alumino de 1 o 2 micras que corresponde a la puerta.
Los ic's MOS se fabrican en tres diferentes sistemas, en funcion del tipo de regiones semiconductoras que contengan :
PMOS, NMOS y CMOS.
Las dos primeras se usan mayormente en audio y amplificacion de seniales.
La ultima se usa en hardware para memorias y circuitos especificos del fabricante (ASIC)

Existe una clasificacion aparte de la tecnologia, que es en funcion del numero de componentes que contienen.

+ SSI (Small Scale Integration) Hasta 250 transistores
+ MSI (Medium Scale Integration) Hasta 2500 transistores
+ LSI (Large Scale Integration) Hasta 250.000 transistores
+ ELSI (Extra Large Scale Integration) Mas de 250.000

8 - El transistor como ampliicador

El transistor es un componente que puede realizar muchas tareas diferentes en un circuito electronico, pero la funcion mas comun y la mas importante es de actuar como amplificador.

Como funciona ?

Cuando un transistor amplifica una senial, en realidad lo que hace es entregar la misma senial, con corriente mas alta, o tension mas alta, o puede ser que las dos variables aumenten.

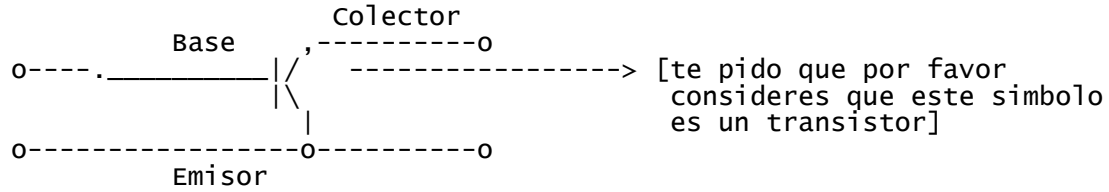
El transistor es capaz de amplificar corriente, es decir, cuando una intensidad que se aplica en uno de sus terminales (emisor o base casi siempre), responde con una corriente mayor en el de salida (colector)

Sin embargo, mediante esta forma de trabajo y cambiando solamente algunos parametros tambien podemos obtener otras amplificaciones como la de tension y la de potencia.

Amplificador en emisor comun

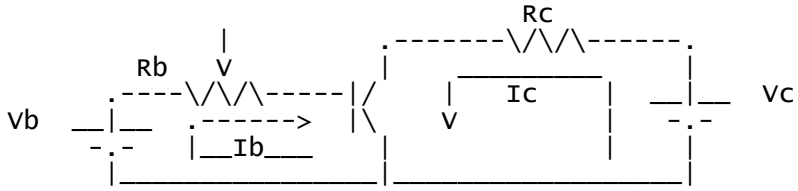
La disposicion mas comun cuando se necesita amplificar una senial es la de emisor comun.
Porque se llama asi ?

Porque el emisor del transistor sirve de punto comun para la entrada y la salida tambien.



La senial que se quiere amplificar entra por la base del transistor y la salida amplificada se tiene por el colector.

Vamos a estudiar esta etapa amplificadora con tensiones continuas, en una disposicion como la de la figura :



[las flechas senialan la circulacion de la corriente]

Bueno, que tenemos en esta falta de respeto de diagrama esquematico ?

Vb : Tension de la primera etapa
Rb : Resistencia que limita la corriente (variable solamente Rb)
Ib : Corriente de la primera etapa

Para la segunda etapa las indicaciones son equivalentes.
El transistor es un NPN.

Como funciona esta etapa ampificadora? (musica de suspenso)

En primer lugar, podemos ver dos fuentes de tension, Vb conectada a la base del transistor y Vc conectada al colector del transistor, a traves de unas resistencia Rb y Rc respectivamente.

Como el transistor es un NPN, la union emisor-base se polariza directamente, y por lo tanto, una corriente Ia circula por el circuito de base, que depende del valor de Rb.

(fin de musica de suspenso-comienza musica al estilo bonanza)

La union base-colector estara polzrizada inversamente, pero por el efecto de Vc, tambien se presenta una corriente Ic , que depende totalmente de Ib, pero esta es mucho mayor.

Si ahora variamos el valor de Rb, la corriente de base tambien variara junto con la de colector, pero esta varia con mayor magnitud, y tambien variara la tension que hay en Rc.

(fin musica bonanza)

Ganancia

[Mensaje para gmz : Ahi dice ganancia, no gancia]

Supongo que tu tienes muchas ganas de aprender, pero yo no me explico muy bien, asi que te dire que lo que ha pasado es que se produjo una amplificacion de corriente.
El valor de la amplificacion se conoce como Ganancia, que se indica con la letra griega Beta (B) que representa la ganancia de continua del transistor.

B = Ic / Ib

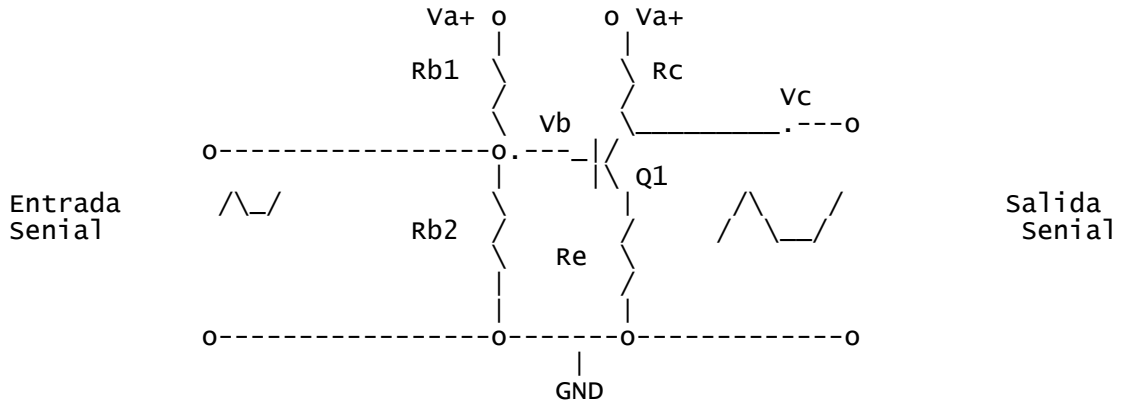
Tambien se produce una tension en Rc que depende de la corriente Ib (porque esta es la que se amplifica)

Si en lugar de variar Rb, se varia la tension Vb, se produce un efecto igual, que ademas, si el valor de Rb es el adecuado, tambien se produce una

ganancia de tension que se puede aplicar a la siguiente etapa, donde se obtienen en Rc.

[eh?]

Una etapa de este tipo puede ser algo asi :



El transistor es un NPN (para un PNP se usan tensiones opuestas)

Observas algo raro ?
Por supuesto, han desaparecido las baterias, y las he reemplazado con Va+ y GND (ground, tierrra).
Porque he hecho eso?
Porque las alimentaciones se comportan como cortocircuitos ante las seniales que son amplificadas, y dibujandolo asi se entiende mejor porque se sabe de antemano que todo el circuito trabaja con la misma fuente de alimentacion.

La base esta conectada a dos resistencias Rb1 y Rb2 que la polarizan en continua (le entregan continua), y el colector esta conectado a la CC tambien a traves de Rc.

En este caso, hay una circulacion de corriente entre base (Ib) y colector (Ic), con lo que en el punto en que se unen Rc y colector, que sera la salida, habra una tension Vc, menor que Va.

Si ahora aplicamos una tension alterna (la tension de red, senial de audio, de radiofonia, de video, etc) a la base con un nivel bajo, se obtiene a la salida una senial que tendra la misma forma pero de nivel mas alto.

Que significa esto ?
Que hemos conseguido una ganancia de tension en la senial.
La podemos calcular segun :

Ganancia Tension (Gv) = VSalida / VEntrada

[creo que hasta ahora vamos bien]

Punto de funcionamiento

Vamos a ver los dos aspectos funcionales que intervienen en una etapa amplificadora.

+ Punto de funcionamiento :

  Es la situacion que se crea sobre el transistor cuando la CC lo atraviesa. Entonces, deducimos que el punto de funcionamiento depende de los valores de Rb1, Rb2 y Rc, porque dependiendo de la CC que entre a la base, circulara mas o menos corriente, y lo mismo va a pasar en el colector, produciendo sobre Rc unas diferencias de potencial dependiendo de ella, y asi se fija la tension continua de Vc.

+ Ganancia de senial :

  Este punto se tiene en cuenta cuando el circuito trabaja con tensiones alternas (no me gusta decir AC porque todos se acuerdan de Ac/Dc)(tampoco digo CA porque todos se acuerdan de caca)

La ganancia de senial solo se produce si se ha elegido el punto de funcionamiento.

El circuito que estamos estudiando se puede completar una cuarta resistencia (Re) entre el emisor y el punto comun.
La cuarta resistencia sirve para estabilizar el circuito y para mejorar el disenio.

Efecto de la temperatura [si hace calor, se transpira. Si hace frio, no]

Cuando el transistor se coloca en esta disposicion, el efecto de la temperatura hace variar la corriente del colector.

La corriente Ic no es perfectamente constante para una Ib cualquiera, sino que si la temperatura aumenta, la Ic se hace mayor y la Vc se hace cada vez menor.

Este efecto hace variar el punto de funcionamiento y pueden haber recortes o distorsion en la senial. Esto se puede corregir con la cuarta resistencia Re que esta entre emisor y el punto comun.

Seguramente tu cabeza debe dar vueltas preguntandose como una miserable resistencia puede salvarnos el circuito..

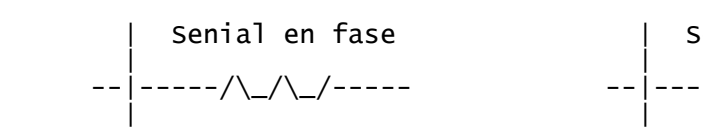
Si Ic aumenta, sobre Re se produce una caida de tension mayor y esta caida se resta de la tension emisor-base que habia, haciendo que la corriente de colector baje y problema solucionado.

Pero aun no estamos 100% seguros.
Si la etapa consume mas potencia de la que el transistor puede disipar, lo que tendremos sera un 'asado de transistor', porque se quemara y ya no habra nada que hacer, sino cambiarlo definitivamente.
Esto tambien tiene una POSIBLE solucion, que es montar el transistor sobre un disipador de calor, para que la superficie del transistor aumente y la disipacion de calor tambien.

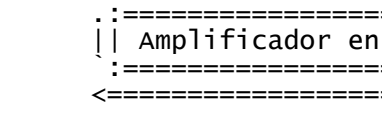
Digo posible solucion porque aunque agregues el disipador, si exiges al transistor mas de lo que deberias, igualmente se va a danyar.

Variacion de corriente en el colector

La variacion de corriente se puede aplacar un poco con realimentacion negativa. Esta tecnica consiste en introducir en la entrada del amplificador una parte de la senial que se obtiene a la salida, pero en fase contaria.



Te habras dado cuenta que de esta forma se pierde algo de ganancia, pero se obtiene mas respuesta del circuito y menor distorsion.



Si aun no te has aburrido de leer, te invito a estudiar la etapa de base comun.
En esta etapa, no hay que ser muy vivo pa darse cuenta que la ganancia de corriente sera muy baja, porque la corriente de emisor esta formada por dos partes : la de base y la de colector.
De esta manera la ganancia de corriente siempre es inferior a 1.
La ganancia de corriente en este tipo de amplificador se denomina alfa (a, la letra griega)

a = Ic / Ie [por si aun no te has dado cuenta, Ic es la corriente de colector, y Ie es la de emisor]

Ganancia de tension

Pero no todo es gris cuando miramos al cielo.
La ganancia de tension en esta etapa es bastante elevada.
Esto es porque la resistencia o impedancia de entrada del emisor es muy baja, y la de salida es muy alta.

O sea, que si aplicamos una senial de bajo nivel en la entrada, se producen unas pequenias variaciones de corriente de emisor, que llegan al colector y se aplican a la resistencia de carga Rc , que casi siempre es de alto valor.

A simple vista deducimos que las mismas variaciones de corriente aplicadas sobre dos resistencias de valores muy diferentes, traen dos niveles de tension con diferente magnitud, o sea, una amplificacion.

Que porque esto es asi ?
Es asi segun la Ley de Ohm

[como que no conoces la Ley de Ohm...aghh!..la pastilla!!!]
[mentira, como vas a conocer la Ley de Ohm si no he escrito nada sobre eso]

Aunque tambien la puedes haber estudiado en la escuela, si este es tu caso, vamos a desempolvar neuronas..

LEY DE OHM : I = V / R
V = I * R
R = V / I

Puedes haber visto un simbolo asi en algun lado..

```
   ___
  /__V__\  ---> esta linea simboliza division
 /  I | R  \
`-----^-----'
      `--------> esta simboliza multiplicacion
```

Entonces podemos obtener cualquiera de los valores con solo mirar el esquema.

Para acordarse mas facil : Ley de Ohm = (V)an los (I)ndios por el (R)io
[por eso era musica de bonanza mas arriba]

Bueno, dejemos el tema de la Ley de Ohm y sigamos con la senial del amplificador.

El nivel de la senial que tenemos sobre Rc en la salida sera (V=I*R) :
VSalida = Ic * Rc

Por otra parte, Ic = (VEntrada / Vsalida) * a [alfa]
Entonces VSalida= Ventrada * Rc * a / Rent

[es matematica pura]

Entonces la ganancia total es :

Gv = Vsalida / Ventrada = Rc * a /Rent

O sea, que la ganancia esta definida por la relacion entre las resistencias de colector y la resistencia de entrada del transistor.

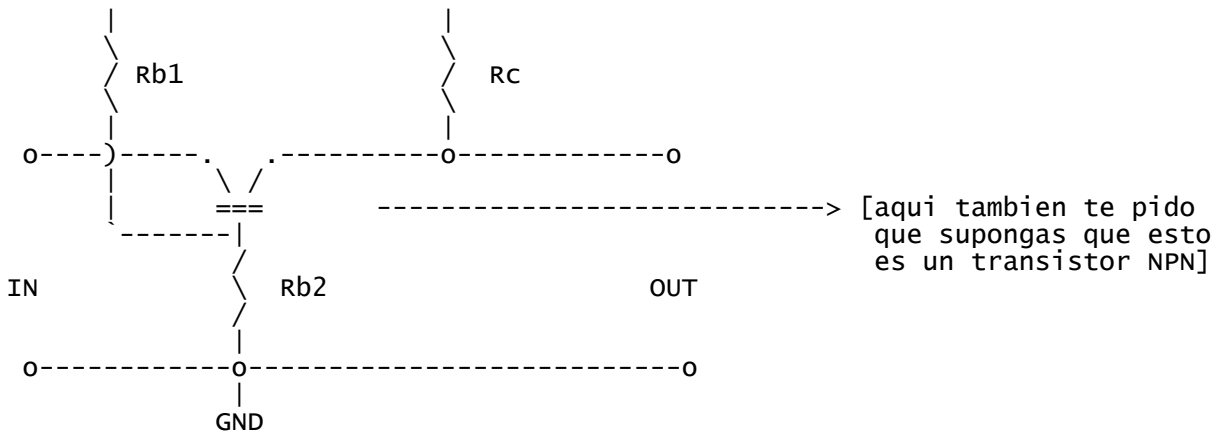
Vamos con un ejemplo para entender mejor.
Tenemos un transistor con una impedancia de entrada de 100 Ohm, una Rc de 10K (Ohm), con un factor a(alfa) de 0.9.

$$Gv = \frac{10000 * 0.9}{100} = 90$$

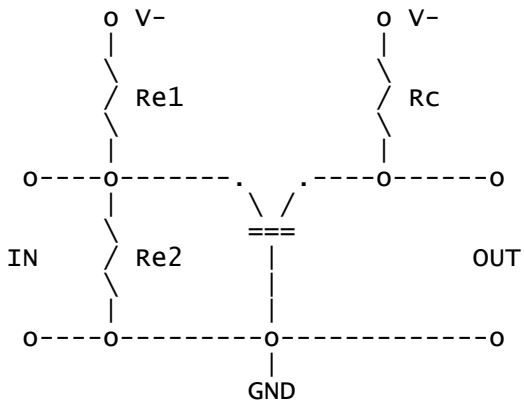
Lo que supone que si se aplica una senial de 0.01 V a la entrada, se obtendran 0.9 V a la salida [si la calculadora anda bien]

Aqui tienes el esquema electrico de la etapa de base comun.

o V+ o V+



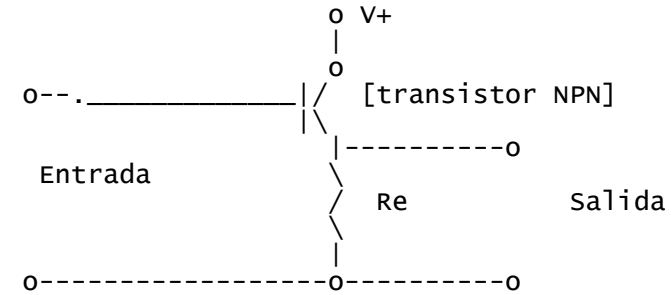
Y con un transistor PNP ..



Amplificador en base coumn

Ahora vamos con la etapa de colector comun, que tambien se suele llamar seguidor de emisor.

En esta etapa, la senial tiene entrada por la base y salida por el emisor. El colector sirve como punto comun.



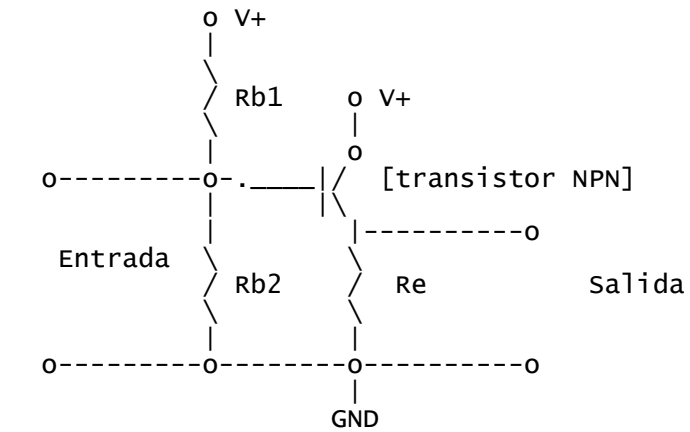
Si hemos entendido todo hasta ahora, a primera vista nos vamos a dar cuenta de que aca no hay ganancia de tension, es mas, la tension que saldra por el emisor sera algo inferior a la de entrada, porque es atenuada un poco por Re.

Pero ya dije que no todo es gris en el cielo.

Esta etapa tiene una impedancia de entrada muy alta y una de salida muy baja. Esto implica que la ganancia de corriente es muy elevada, porque la senial que tiene una tension baja, solo emplea una pequenia parte en excitar a la etapa por su alta resistencia.

En la salida, la situacion es diferente, porque el transistor va a entregar la corriente necesaria para a la resistencia de emisor para que sobre ella se encuentre la misma tension que en la entrada, lo que supone una corriente mas alta porque esta resistencia es de valor relativamente bajo.

En este ejemplo, se colocan las resistencias necesarias para situar la etapa en el punto de funcionamiento adecuado.



Si el transistor fuera un PNP, las tensiones serian opuestas.

Aun no capto

Vamos con un ejemplo de una etapa de colector comun, con una impedancia de entrada de 50K y una resistencia de emisor (Re) de 500 Ohm.
Si aplicamos una senial de entrada de 1V la corriente que circulara sera de ...

[sacando calculadora]

1V / 50000 Ohm = 0,02mA (0,00002 A)

Esta misma senial, ligeramente atenuada, aparece sobre Re, supongamos que ahora tiene 0,9 V.
La corriente sera entonces :

0.9V / 500 Ohm = 1.8 mA (0.00018 A)

El mismo valor se puede obtener dividiendo la resistencia de entrada por la de salida y multiplicando el resultado por 0.9, que es la atenuacion que se supone que se produce.

Tambien tenemos que tener en cuenta Rb1 y Rb2, que definen el punto de funcionamiento, junto con Re, que afectan a la resistencia de entrada de toda la etapa. Con estas resistencias, el transistor tendra buena estabilidad termica, como en la de emisor comun.

La etapa de colector comun casi siempre se usa como adaptadora de impedancias.

%@%@%@%@%@%@%@%@
% Resumiendo @
%@%@%@%@%@%@%@%@

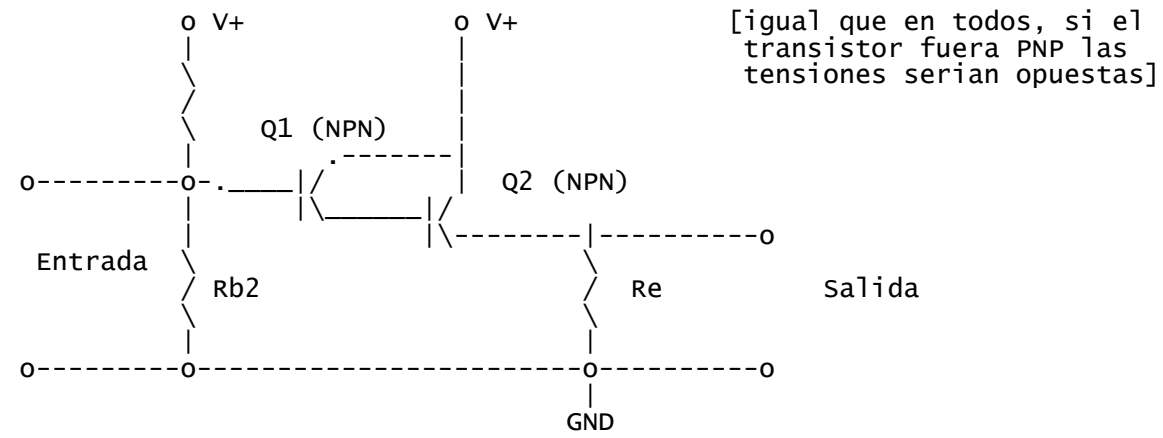
	Configuracion		
	Emisor Comun	Colector Comun	Base Comun
Impedancia de entrada	Media	Alta	Baja
Impedancia de salida	Media	Baja	Alta
Ganancia de tension	Media	Igual	Alta
Ganancia de corriente	Media	Alta	Igual
Inversion de fase entrada-salida	Si	No	No

Circuito Darlington

Hay una combinacion de transistores en colector comun denominada 'Darlington', que lleva ese nombre por un tal Darlington que realmente no tengo npi quien es [tampoco busque]

Esta combinacion acentua las propiedades del colector comun (mira el cuadro)

porque el segundo transistor multiplica la ganancia total porque recibe sobre su base la ganancia de emisor del primero.



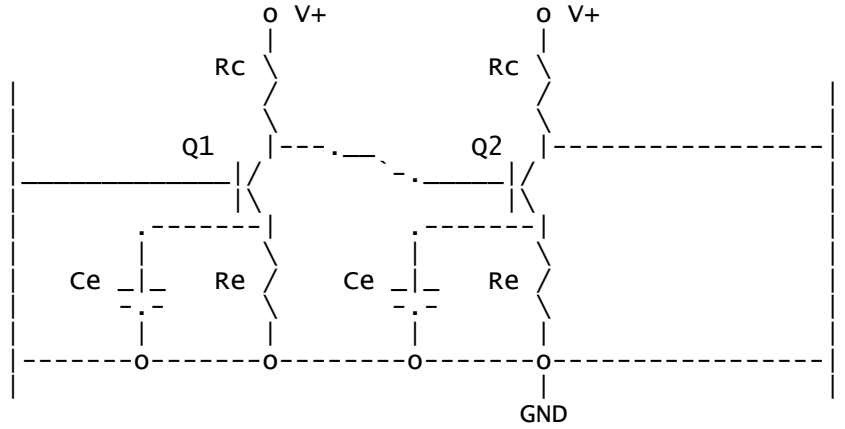
Existen montajes Darlington que vienen en una sola capsula como si se tratara de un unico transistor.

Amplificadores de varias etapas

Ya conocemos las etapas ampificadoras basicas a transistor. Ahora vamos a estudiar como enlazar varias etapas para completar un amplificardor de cualquier modelo.
Segun la forma de acoplamiento vamos a distingur tres grupos principales :

Amplificadores de CC [cc, no kk porque eso es desagradable]

Estos ampificadores se enlazan directamente sin necesidad de otro componente adicional. Son muy dificiles de diseniar porque los puntos de funcionamiento de los dos transistores estan relacionados entre si.

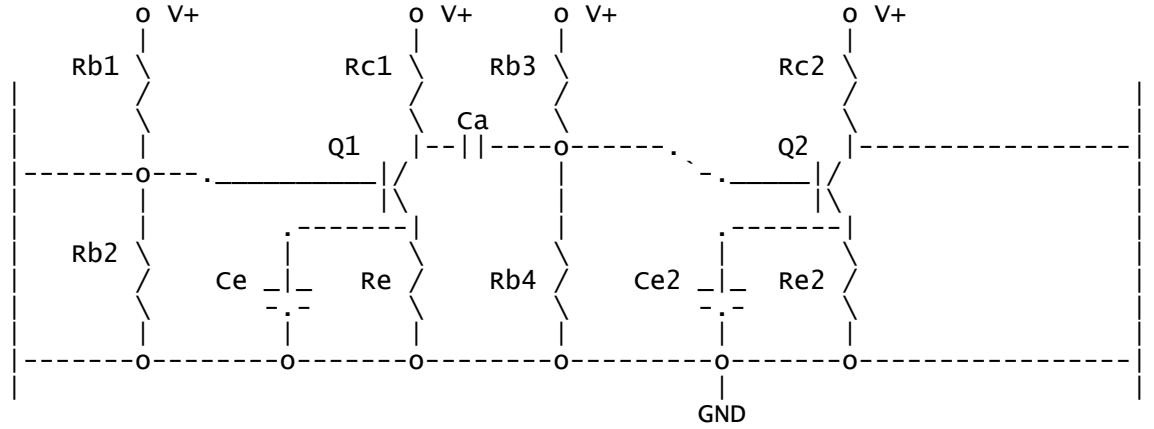


Estos amplificadores tienen el inconveniente que son mas sensibles a la temperatura.
La ventaja que tienen es que pueden trabajar con frecuencias desde 0Hz (corriente continua), hasta 8 o 10 MHz.

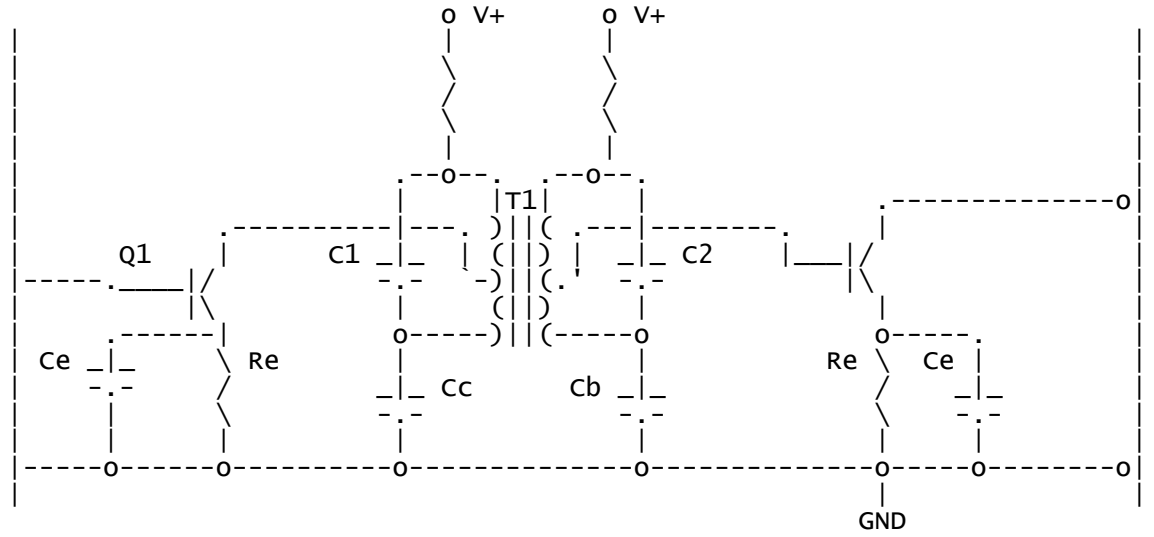
Amplificadores RC

En estos amplificadores, las etapas se acoplan con un capacitor que separa los niveles de continua entre ellas.
La salida de cada etapa se toma de la resistencia de carga de colector o emisor (Rc), para que sea llevada al capacitor que la lleva al paso siguiente.

El condensador que se elija tiene que tener una reactancia baja para que no atenue las seniales.
Los amplificadores RC son los que mas se usan para amplificar audio.



Hay otra clase de amplificadores RC, que se acoplan mediante transformadores. Entonces, la ganancia de estos solo es alta cuando la senial esta dentro del margen de frecuencia del transformador.



Clases de amplificacion

Segun la forma en que se amplifica la senial, se clasifican los amplificadores.

+ Clase A : En estos la corriente de salida sigue constantemente la forma de la senial, y no anula la senial en ningun momento.

+ Clase AB : En estos la senial se corta durante una fraccion por anulacion de la corriente. Se usa an algunos amplificadores de audio.

+ Clase B : La corriente se anula en un semiciclo de la senial. Se usa en muchos amplificadores de audio, cuando se emplean transistores en contrafase.

+ Clase C : La senial se anula durante un tiempo mayor de un semiciclo. La distorsion que producen es muy alta, pero se corrige con acoplamientos sintonizados y filtros.
  Se usa en alta potencia y amplificadores de radio.

Rendimiento

El rendimiento de un amplificador depende de su clase.
El rendimiento se calcula segun :

Rendimiento = PotenciaSenial / Potencia total

Clase	Rendimiento

A	25%
B	50%
AB	25 a 50 %
C	Mas de 50%

Resistencia de carga

Para saber el valor que va a tener la resistencia de carga, primero tenemos que saber la resistencia de entrada de la etapa, porque esta se comporta como carga del paso anterior.

Entonces, cuadno al circuito se le aplica una senial, Rc1 se pone en paralelo con la resistencia de entrada del paso siguiente, pasando por Ca, que hace de cortocircuito para la senial y a traves de los condensadores de la fuente de alimentacion [que no aparecen en el esquematico]

Teniendo en cuenta esto, la resistencia total de carga total es la que se obtiene en la asociacion en paralelo :

$$R\ Paralelo = \frac{R1 * R2}{R1 + R2}$$

[esto es asociacion de resistencias. Espera a la parte 3, que trata de los componentes electronicos mas frecuentes, asociacion de componentes y formulas, Leyes electricas y otras yerbas, que no son maria juana ni parecidas]

Que en este caso resulta en :

$$R\ Carga = \frac{Rc1 * R\ Ent}{Rc1 + R\ Ent}$$

Impedancia de entrada

La impedancia (o resistencia) de entrada de cada paso tambien esta relacionada con las dos resistencias Rb1 y Rb2, o Rb3 y Rb4.

Estas resistencias polarizan la senial y tambien se encuentran en paralelo, entonces, podemos calcular con :

$$Rb = \frac{Rb1 * Rb2}{Rb3 + Rb4}$$

Tomamos el resultado de esta ecuacion (Rb) y lo usamos en esta otra :

$$R\ carga = \frac{Rb * Rtransistor}{Rb + Rtransistor}$$

Que no sabes cual es la resistencia del transistor ?

Busca en inet, en la pagina del fabricante, en cualquiercosa.

Si el transistor es de potencia, tal vez puedas medirlo con el tester, colocandolo en R*1 y colocando la punta positiva en la base, y la negativa en el colector.

Ahora que sabemos el valor de todas las resistencias que afectan al funcionamiento, vamos con ..

Ganancia de tension

Y como calculamos esto ?

Multiplicamos la corriente de la senial que entra, por la ganancia de corriente (B, beta), y asi tendremos la intensidad que sale por el colector

$$Icolector = Ientrada * B$$

Luego multiplicamos Icolector por la resistencia de carga, y asi tenemos la tension de la senial (Vs). El calculo es mera Ley de Ohm.

$$Vs = Icolector * Rc$$

La corriente de base del segundo transistor se obtiene dividiendo Vs por

la resistencia de entrada del segundo transistor [ley de ohm].
  Si repetimos todo el calculo anterior a partir de esta corriente, tendremos la tension final de salida.

  Si aun no entiendes [no es que te trate de tonto, es que no se si me explico bien], veamos todas las formulas.

```
  Icolector = Ientrada * B
        |___
            |
            V
  Vs = Icolector * Rc
   \________
            |
            V
  Ientrada2 = Vs / Rentrada
   \_________________
                     |
                     V
  Icolector2 = Ientrada2 * Bsegundo_transistor
          |
          V
  Vs = Icolector2 / Rc
```

  Y entonces Vs es la ganancia total de tension del transistor.

  Decibel   [y el lector dijo 'bel'] [ya se que no es mio el chiste, pero es gracioso]
  """""""""

  El decilelio o decibel es una unidad de medida que se obtiene de comparar dos niveles de senial en un circuito, aunque tambien se usa para aplicaciones acusticas.

  Tomemos como ejemplo una etapa cualquiera de un amplificador, en la que tengamos una elecvacion del nivel de una senial, o sea la ganancia.

  La ganancia se calcula con :

```
       Vs   [ Voltaje salida ]
  G = ----
       Ve   [ Voltaje entrada ]
```

  Como hacemos para expresar la ganancia en decileles ?

  Ganancia(db) = 20 log G

  o sea

```
                        Vs
  Ganacia(Db) = 20 log ----
                        Ve
```

  Realimentacion del amplificador
  """""""""""""""""""""""""""""""

  Aja, muy bonito el titulo, pero que es la realimentacion ?
  Se dice que un circuito esta realimentado cuando la senial de entrada esta formada por otras 2 :

  + La senial que llega del exterior
  + Una fraccion de la senial de salida

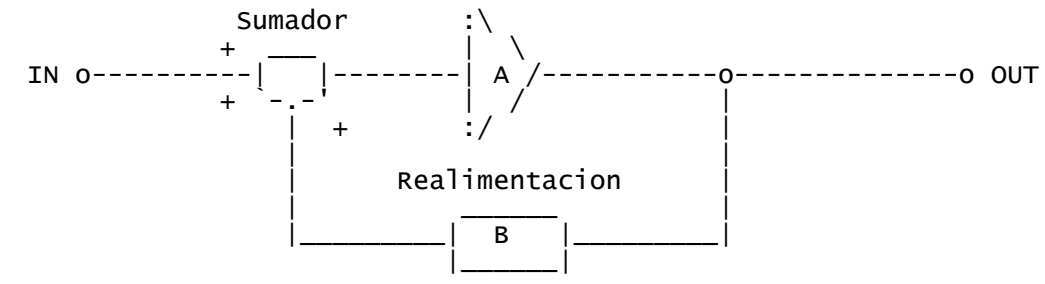
  La realimetacion se usa principalmente en 2 tipos de circuitos : Amplificadores y osciladores.

  Tenemos 2 tipos de realimentacion :

  + Realimentacion positiva : En este caso las dos seniales de la entrada se suman en fase, y la senial de salida es mas fuerte.

  + Realimentacion negativa : Las dos seniales se suman, pero una esta en contrafase con la senial que viene del exterior. El efecto que produce es disminuir la ganancia.

                                        Amplificador



$$\text{Salida} = \frac{A}{1(A*B)}$$

Efectos de la realimentacion negativa

	Realimentacion de tension	Realimentacion de corriente
Distorsion	Se reduce	Se reduce
Relacion Senial/Ruido	Mejora	Mejora
Impedancia entrada	Aumenta	Disminuye
Impedancia salida	Disminuye	Disminuye

9 - Osciladores a transistores

Otro de los circuitos que mas se utilizan en la practica, son los osciladores.

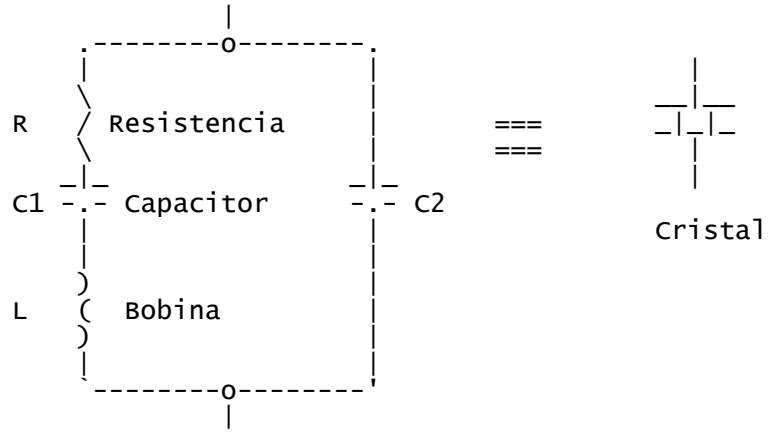
Que hace un oscilador ? La sola palabra lo dice :
Genera oscilaciones electricas permanentes, o sea una senial sinusoidial continua con unos valores de amplificaion y frecuencia determinados

[bueno, la palabra no decia todo eso]

Cuando el transistor trabaja como amplificador, es indispensable para armar osciladores, porque se encarga de entregar la energia que haga falta para que el conjunto L-C (bobina-capacitor) genere la senial.

Cristales de cuarzo

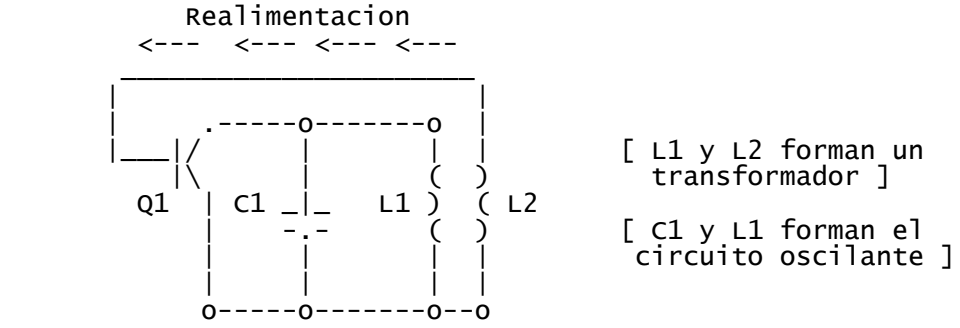
Seguramente habras visto o tengas un reloj a cuarzo. Porque se llama asi?
Porque la oscilacion del reloj no depende de un circuito L-C, sino que se reemplaza por un cristal de cuarzo, que cumple la misma funcion y genera una oscilacion mas estable. El inconveniente [no todo es bueno en el mundo] es que una vez que la frecuencia de oscilacion se fija, ya no se puede variar.



Funcion del transistor

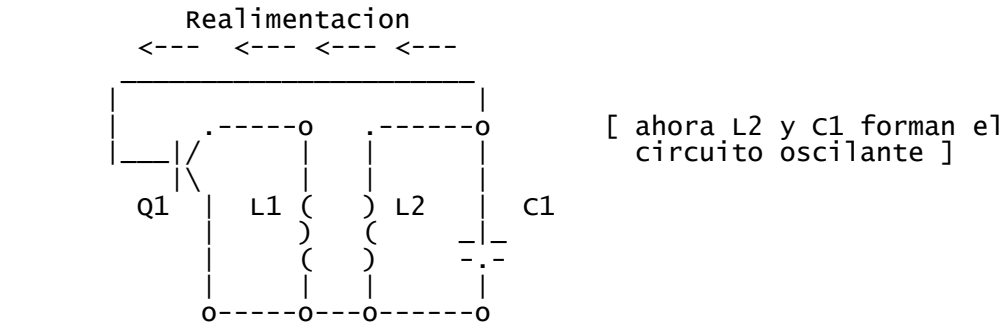
"""""""""""""""""""""""""""

Vamos a partir de un circuito l-c basico para analizar los diversos tipos de osciladores. Primero que nada, veamos este ascii :



El funcionamiento de este circuito es una pavada :
El secundario del transformador (L2) toma una cierta cantidad de la senial que sale del circuito oscilante y la inyecta en la base del transistor, que la inyecta otra vez al circuito L-C, que es tomada por L2, que la inyecta otra vez al circuito L-C, que es tomada por L2, que la inyecta otra vez al circuito L-C, que es tomada por L2, que la inyecta otra vez al circuito L-C, que es tomada por L2, que la inyecta otra vez al circuito l-c, que es tomada por L2, que la inyecta otra vez al circuito l-c ... (infinite loop)

Luego de leer la misma cosa una y otra vez, veamos otro circuito muy parecido :



[no voy a explicarlo porque si entendiste la recursividad de arriba, supongo que tambien entiendes este circuito]

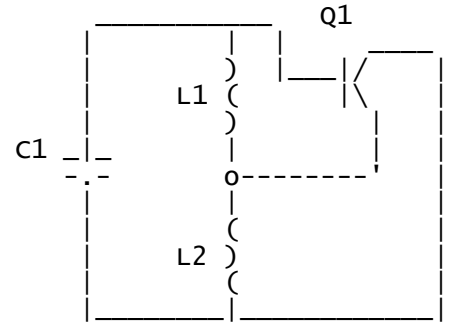
Osciladores Clasicos

"""""""""""""""""""""""""

Porque clasicos ?
Vienen en vinilo ?
Son en blanco y negro ?

No, porque son los 'padres' de todos los osciladores modernos.
Estos osciladores son los llamados Hartley y Colpits.

Oscilador Hartley :



En el esquematico vemos el oscilador, pero no he dibujado las alimentaciones ni los componentes necesarios para las polarizaciones.
Si miras bien al esquematico, podras ver que el circuito oscilante esta formado por C1 y la suma de las bobinas L1 y L2.

Una fraccion de la senial generada es tomada por la base y el emisor, y es amplificada por el colector, que completa el ciclo.

Si, ya se que no te interesa eso.
Quieres que te diga como calcular la frecuencia a la que oscila este circuito ...
Ok..

[sacando calculadora, pero la cientifica ahora]

```
                          1
Frecuencia = -----------------------------
              2 * PI  ________________
                    \| (L1 + L2 ) * C1
```

Ahora veamos el oscilador Colpits :

```
         ___________   Q1
        |         |  |      ____
        |         |  |___|/     |
        |    C1 _|_      |\     |
        |       -.-        |    |
        (         |        |    |
     L1 )         o--------'    |
        (         |             |
        |    C2 _|_             |
        |       -.-             |
        |         |             |
        |_________|_____________|
```

[no, espera ! Antes de empezar a insultarme mira bien el dibujo y fijate que ha cambiado]

Creo que no hace falta explicacion, si es que comprendiste el otro.

Y como calculamos la frecuencia de oscilacion de este ?
Es bastante parecido :

[si sacamos la calculadora cientifica]

```
                            1
Frecuencia = ----------------------------------
             2 * PI    _____________________
                   \  |     ( C1 * C2 )
                    \| L * ( ------- )
                            ( C1 + C2 )
```

Uso de cristales de cuarzo

Los cristales de cuarzo se pueden usar para reemplazar a las inductancias o bobinas.
Creo que se entiende mejor por dibujos.

```
         ___________   Q1
        |         |  |      ____
        |         |  |___|/     |
        |    C1 _|_      |\     |
        |       -.-        |    |
 XTAL __|__       |        |    |
     _|_|_        o--------'    |
        |         |             |
        |    C2 _|_             |
        |       -.-             |
        |         |             |
        |_________|_____________|
```

En este oscilador Colpits la inductancia se ha sustituido por un cristal.

```
@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%
% 10 - El transistor en el campo digital                                 @
@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%
```

Ah, como pasa el tiempo...
33 numeros de SET...
Supongo que no voy a tener que explicar a que me refiero
con 'el campo digital'...
[1s y 0s, ondas cuadradas y muuucho mas]

Caracteristicas del transistor

Cuando usamos transistores en aplicaciones digitales, entran en juego otras caracteristicas que no influyen en el trabajo analogico.

Son el estado de corte y de saturacion.

+ Corte : Un transistor esta en este estado, cuando la union base-emisor no esta suficientemente [que palabrota] polarizada para entrar en conduccion. (La tension tiene que ser menor que 0.2 V)

  Cuando estamos en esta situacion, se impide el paso de la corriente por los terminales de salida, produciendo que la tension en los terminales sea la misma que la de alimentacion.
  Si tenemos en cuanta las ensenianzas del Sr. George Boole,
  a la salida tenemos un 1. (Ver Set 26 : 0x09)

+ Saturacion : Es lo contrario del estado de corte, o sea, una corriente de base alta. A la salida, tendremos un 0.

Bueno, menos charla y mas accion
Veamos el primer circuito :

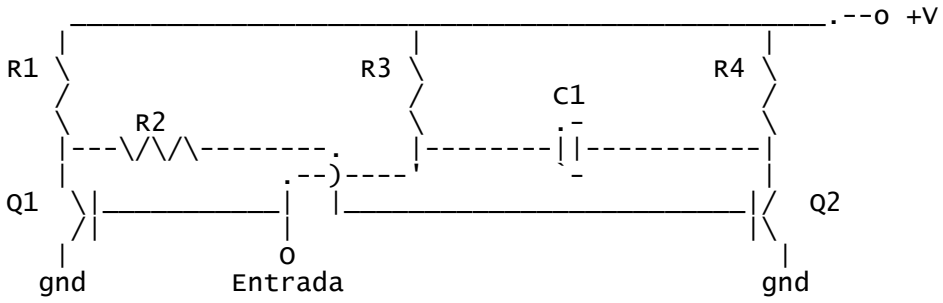
Monoestable a transistor

Que es un monoestable ?
Es un circuito que, cuando recibe una senial en su entrada, responde con otra en forma de impulso a su salida. La duracion de la senial depende de los componentes del circuito.
Para eso necesitamos un componente que controle la duracion del impulso.

Un pajaro ?
Un avion ?
Superman ?
Un chofisticado scannerwingateropasscrackerbackdoor multiplataforma y temporizador ademas ?

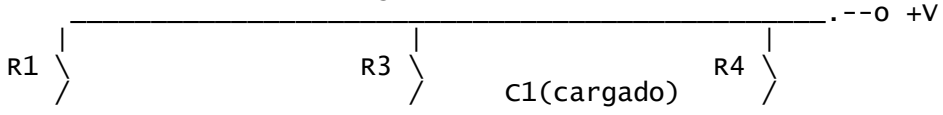
No.
Solo necesitamos un capacitor.

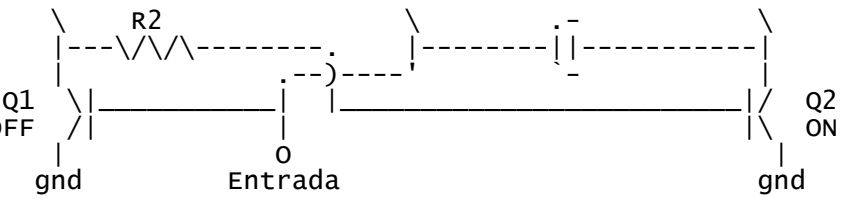
Este es el circuito mas elemental del monoestable :
[espero que se entienda]



[se supone que c1 es un capacitor electrolitico. Espera a la parte 3..]

Supongamos que Q1 esta saturado. La tension de colector sera practicamente la misma que la de emisor, que esta unido a masa, por lo que Q2 estara en corte [o cortado, como quieras].
C1 estara cargado con una tension cercana a la de alimentacion, por lo tanto el circuito puede permanecer en este estado hasta el fin de los tiempos, a where no man will ever go.





[sigue suponiendo que son transistores NPN]

Entonces, aplicamos un impulso negativo a la entrada [que casualmente se llama entrada], Q1 tendera a amplificarlo, ira al colector convertido en un impulso positivo [recordemos que el emisor comun cambia la fase de la senial] y luego pasara por R2, hacia la base de Q2, que lo amplificara otra vez y lo cambiara de fase, otra vez.

La condicion a la que tiende el circuito es que Q1 se ponga en corte, y Q2 se sature, como en la figura de arriba. Pero esta C1 en el circuito, que no se ha podido descargar porque el proceso descrito se produce con MUCHA rapidez.

El estado semiestable

Para entender esta parte nos haran falta algunas cosas :
PazYCiencia [www.noromperlascosascuandonofuncionen.com/intentardevuelta]
Cerebro [www.arribadelcuello.dentrocabeza.com]

Opcional :
Calculadora, que se consigue en partes separadas en :
[www.conseguirunrevolver.de/grancalibre]
[www.robarelnegociodearticuloselectronicos.de/tupueblo]

Miremos el circuito otra vez. Q2 esta saturado, y por lo tanto C1 esta virtualmente conectado a masa, y tendera a cargarse a traves de R3 hasta que tenga una tension igual a la de alimentacion (con la polaridad cambiada)

Este proceso supone que primero se descargue hasta que la tension en sus bornes sea nula [nula=0], para que pueda cargarse.
Pero como el terminal negativo de C1 esta conectado a la base de Q1, cuando la tension del capacitor sea 0 y comienze a hacerse de signo opuesto, Q1 empeazra a conducir (porque su base es mas positiva que su emisor).

Esto hara que sobre el colector aparezca una reduccion de tension, que sera llevada a Q2, iniciando un proceso como el de antes, pero en el sentido opuesto, que acabara con Q1 saturado y Q2 en corte.
Entonces, este estado no es estable hasta el fin de los tiempos como el otro, sino que el tiempo que permance este estado se calcula con :

[Bajar calculadora de www.estante.de/mueble]

Tiempo = 0.7 * C1 * R3

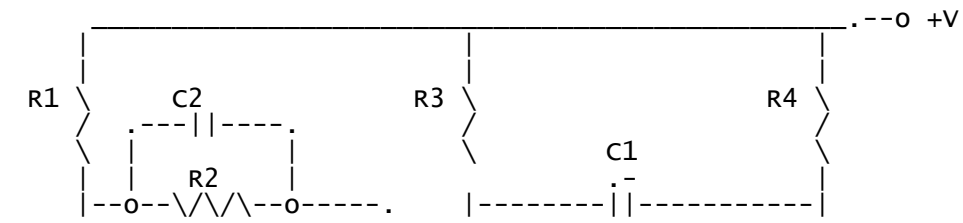
Tiempo esta dado en milisegundos(ms), C1 en microfaradios (uF) y R3 en Kilohmios.

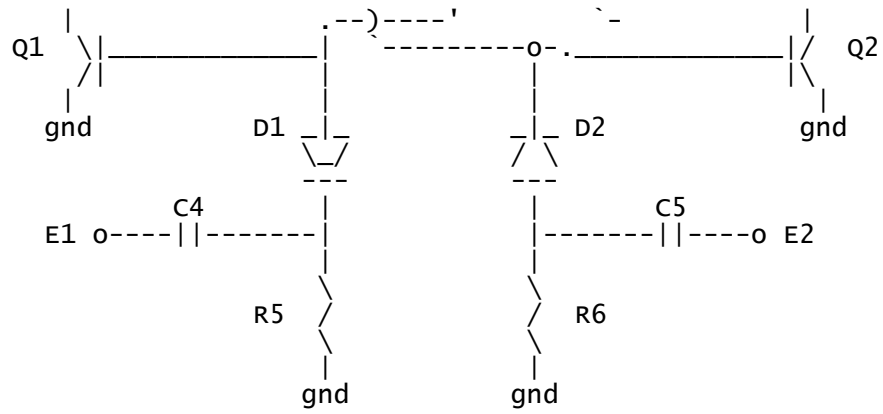
Si entendimos todo, sabemos porque se llama monoestable.

Un poco mejor

Como en SET hemos aprendido a no ser siempre newbies, sino que debemos ir mejorando, tambien vamos a aplicar eso al circuito.

Miremos este ascii :





Eh? Y que clase de garabato es eso?
Pido disculpas anticipadas, pero ya dije que no soy buen dibujante.

Bueno, ya se que quieres saber para que lleva todos esos componentes adicionales.

+ C2 : Sirve para acelerar el paso del estado estable al semiestable

Los diodos, capacitores y resistencia adicionales, sirven para disparar el circuito con tensiones negativas en E1, y tensiones positivas en E2.

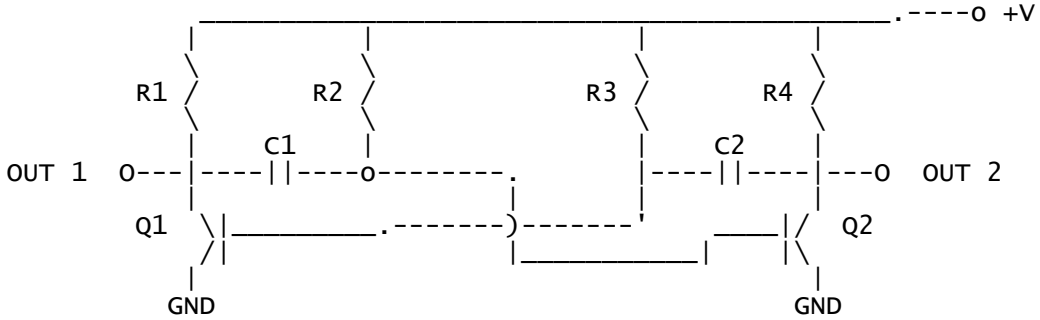
Multivibrador astable a transistor

Ooh! Que titulo!
Pero que es un multivibrador astable a transistor ?
Un juguete erotico con transistores?

Fundamento, funcionamiento, y si hablo miento

Con un multivibrador astable a transistor (m.vibrador desde ahora) tendremos un verdadero oscilador a transistor, pero con la particulariadad [que palabrota] de que la senial que se obtiene es cuadrada, como cabia esperar de un circuito digital.

Si tomamos [con fernet branca] un monoestable, y le agregamos un segundo condensador, como muestra la figura, de modo que el circuito tenga 2 estados semiestables, se consigue que el circuito no mantenga siempre el mismo estado, sino que cambie de uno a otro y repita el proceso.



El funcionamiento es identico al monoestable, y dejo como tarea pensar que sucede cuando Q1 queda saturado y Q2 en corte.
Si no entendiste [porque yo explico muy mal], preguntame a mi mail.

Si inmaginamos que los valores de C1 y C2 son iguales, y R2 y R3 tambien, entonces podremos calcular la frecuencia de oscilacion :

$$\text{Frecuencia} = \frac{0.7}{R * C}$$

[R es el valor ohmico de R2 o R3, y C es la capacidad de C1 o C2]

Si los valores de los componentes mencionados son diferentes, se puede calcular el tiempo que cada salida esta en estado alto, de la siguiente manera:

Tiempo en Out1 :   t1 = 0.7 * C1 * R2
Tiempo en Out2 :   t2 = 0.7 * C2 * R3

El tiempo es dado en ms, C es en microfaradios y R es en Kilohms.

Supongamos que C1 tiene un valor de 1 uF [microfaradio], C2 es de 3.3 uF, R2 es de 680 Ohm y R3 es de 1Kohm

t1 = 0.7 * 1 * 0.680.....0.476 milisegundos
t2 = 0.7 * 3.3 * 1.......2.310 milisegundos

O sea que Out1 esta 0.476 ms en un estado y 2.310 en otro estado.
Lo mismo pasa con Out 2 : 2.310 ms en un estado y 0.476 en otro.

Para que se entienda mejor, veamos [un ejemplo] las formas de onda que se obtienen de las dos salidas :

```
       T1    T2    T1    T2    T1    T2    T1    T2
       __ ________ __ ________ __ ________ __ ________
      |  |        |  |        |  |        |  |        |
          ________    ________    ________    ________
Out1 : __|        |__|        |__|        |__|        |__
       __          __          __          __          __
Out2 :   |________|  |________|  |________|  |________|
```

t1 = 82.91% del total   [esto se calcula con sencilla regla de tres]
t2 = 17.09% del total

La salida se puede usar para encender luces, gobernar reles o para lo que quieras.
Si lo usas para accionar un rele, te recomiendo que agregues esto al circuito :

```
  Desde la salida deseada              o V+
                                 Base  | Emisor
           o------------------.____|/
                                   |\
                      Colector       |----(
                                     V    )  Bobina del rele
                                     -    (
                                     |----)
                                     |
                                    gnd
```

Ahora si, por fin, el transistor es un pnp.

Si quieres montar CUALQUIERA de los circuitos con transistores (amplificadores, osciladores, astables, monoestables), te recomiendo que uses estos transistores [porque son baratos y seguros]

NPN : 2N3904 , BC548 , BC547
PNP : BC558 , BC 557

La disposicion de los pines en los 3 primeros es :

```
  ___
 /___\
| BC  | [o cualquiera de los otros npn]
| 548 |
|_____|
 | | |
 | | |
 C B E
```

En los PNP......
No me acuerdo!! [no es un chiste, es asi. Perdon]
Fijate en un manual de reemplazo, manual EGC, o donde sea.
En el caso de los diodos, recomiendo del tipo 1N4004, o mayor.

```
@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%
% 11 - Montajes practicos                                               @
@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%
```

Regulador de luminosidad con triac
""""""""""""""""""""""""""""""""""

Baliza electronica
""""""""""""""""""

```
    ___________________________________________.-------o--o +V
   |          |                |          |            |
   \          \                \          \            |
R1 /       R2 /             R3 /       R4 /            |
   \          \                \          \            |     Rele
   |    C1    |                |    C2    |        Q3  |
   |----||----o--------.       |----||----|-------.__|/
   |                   |       |          |          |\        P Comun
 Q1 \|_______.---------)-------'     ____|/ Q2         |---(     o------.
    /|                 |___________|     |\         D1 V   )      \     |
   |                                      |            -   (   o   o    |
  GND                                    GND           |---)   |   |    |
                                                      GND      |   |    |
                                                               |   |    |
                                                              (X) (X)   |
                                                               |   |    |
                                                               |___|    |
                                                ______           |      |
  (X) Es el simbolo de una lampara             |  AC  |----------'      |
                                               `------'-----------------'
```

R1, R4 : 1 Kohm (1k, 1000 ohms)
Q1, Q2 : BC548 o 2N3904
Q3   : BC558
D1   : 1N4004
C1, C2 : Ver abajo. Deben ser electroliticos y x 25V

Sienente libre para experimentar con los valores de R2 y R3, que deben ser iguales. Lo mismo con C1 y C2, deben ser iguales.
La fuente de AC es la tension de red, siempre que las lamparas sean las mismas que colocas en un portalampara de tu casa.
Si colocas lamparas de linterna, o algo parecido, es mejor si pones una fuente de CC [no importa la polaridad]

@%
% 12 - Despedida y agradecimientos   @
@%
[tambien desvario en forma de pelicula]

Espero que hayan disfrutado el articulo y que no se hayan aburrido.

Agradezco a todo el staff de SET
A madfran y grrl que son los que me soportan cuando consulto. =)

Nos vemos en otra historia, a la misma hora y por el mismo canal
y con el mismo ingles mal hablado
[si leen esto con el edit de dos, aprieten Ctrl + Flecha abajo a razon de una vez cada 1.5 segundos y parece como en la television]

starring
""""""""

compaq prolinea 3/25s .....as...... the maquinola (PC)
no hair man ....as...... elotro
SET ....as...... great ezine [very great]
madfran and grrl ....as...... contact with set
.... also as...... the guys who support me
gmz ....as...... the falled in love boy
.... also as...... the only who talks about hacking with me
.... also as...... the starwars adicct
.... also as...... the guy who will write with me (i hope)
cocaloca and branca ....as...... cool drink
esprait and gancia ....as...... another cool drink
lucky strike ....as...... very good tobacco
marlboro ....as...... good tobacco
next ....as...... bad tobacco
byte ....as...... old and good zine
.... also as...... eight bits or 1/1024 Kb

```
microsoft .................as..... billy's curro
alcatel331a ...............as..... the cell phone that rings in the
.................................. worse moment
laesquina.com .............as..... suyai's cibercafe (without cafe)
..................... also as..... free internet for hacking
i.nic.ito .................as..... the panqueque
v.cuello ..................as..... himself
..................... also as..... the explosives/war man
jony ......................as..... the guy who have to give me the HD
cevayos ...................as..... metal database
the guitar ................as..... distraction while im writing
metallica .................as..... cure for soul
menem .....................as..... the ladron
..................... also as..... lo hizo
caballeros de la quema ....as..... good old music


sonud track .......avaible in..... elotro's pirate cds

sound track contains :
'Fade to hack' ............by..... Metallica     (fade to black)
'El 386' ..................by..... Divididos     (el 38)
'The phreacking man' ......by..... Iron Maiden   (the wicker man)
'Dirty Windows' ...........by..... Metallica     (dirty window)
'Salir a hackear' .........by..... Divididos     (salir a comprar)
'A hack le monde' .........by..... Megadeth      (a tout le monde)
'Last reboot'..............by..... Papa roach    (last resort)
'The memory remains' ......by..... Metallica and herramientas de
                                   analisis forense, como lectores de
                                   la RAM

fuckz to E.R.Pache and M.btera [por desertores]
respect to gmz por la melena y por la adiccion al gancia
thanks to lucky strike, viceroy and marlboro
also thanks to branca, new style and carragal
Watch ftv(42), filmzone(49) and isat(47)
keep on hackin'

Cameras by Panascanic and Sansorongo
Filmed in chastricolor


Buena suerte,
elotro
elotro.ar@gmail.com

*EOF*
```

```
=)(=)(=)(=)(=)(=)(=)(=)(=)(=)(=)(=)(=)(=)(=)(=)(=)(=)(=)(=)(=
)   Curso de Electronica  -  Tercera entrega    (
=)(=)(=)(=)(=)(=)(=)(=)(=)(=)(=)(=)(=)(=)(=)(=)(=)(=)(=)(=)(=
```

Saludos gente de SET !!

Como podran ver, no he tenido otra cosa para hacer, y he seguido escribiendo articulos para su disfrute. [o no-disfrute]
Espero que se hayan divertido haciendo calculos con mi segundo articulo, y que no hayan quemado ninguna lampara con la baliza.

Cuando comence a escribir el articulo [exactamente en la linea de arriba], la letra 'y' de mi teclado dijo 'no va mas' y dejo de funcionar.
Asi que es un poco molesto tener que apretar Alt+121 o copy-paste de la letra 'y' para escribir, pero aun asi, seguire adelante hasta tener los 35$ que demanda un teclado.

Cuando veas una operacion matematica, es muy posible que no se entienda, que el resultado no sea correcto, o que simplemente sea inutil realizarla.
Pero, esto es ascii, no TeX.

Igualmente, estoy abierto a todas tus dudas, criticas, sugerencias, pedidos, donaciones, opiniones, saludos, amenazas.

Como haces para hacerme llegar esto ? -------> elotro.ar@gmail.com

[no es elorto, como algunos lo escriben. Aparte,
no tenemos porque ser maleducados]

Bueno, basta de charla y vayamos al grano.

Curso de electronica - Tercera Entrega - Componentes electronicos usuales

1. Resistencias
2. Resistencias variables y potenciometros
3. Resistencias especiales
4. Capacitores o condensadores
5. Bobinas o inductancias
6. Reles [o relais/relays]
7. Transformadores
8. Cristales de cuarzo
9. Montajes practicos <--[recibo sugerencias para este apartado]

%@%
@%@% 1 - Resistencias @%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%
%@%

Antes de comenzar, una pequenya aclaracion :
Yo he aprendido mal [no soy perfecto, ni lo quiero ser] , y uso el termino 'resistencia' para nombrar al componente electronico, cuando en realidad la palabra correcta es 'resistor'.

Creo que la mayoria de los lectores ya haran asimilado el concepto de resistencia, y sabran como es el aspecto del componente.
Si no es tu caso, no te preocupes, estamos aqui para que aprendas (y yo tambien)

Los metales son conductores de la electricidad, pero no son perfectos. Todos poseen una resistencia especifica, aunque sea muy baja, pero tienen. Para construir resistencias, no se usan metales, sino que se usan aleaciones y combinaciones de muchos elementos.

Las resistencias [aghh!, se dice resistores] se usan para distribuir correctamente las tensiones y corrientes en los circuitos electronicos [y electronicos].
Las resistencias se miden con una unidad llamada ohimo , que se simboliza

con la letra griega omega (ê) [es el caracter con codigo ascii 234].

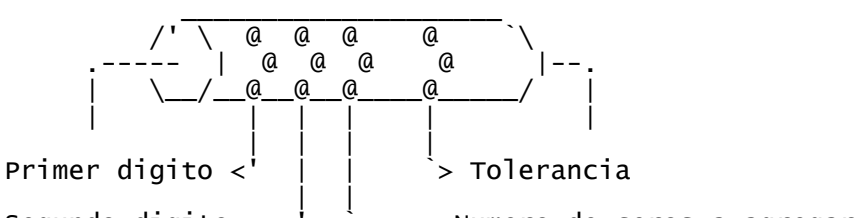
Codigo de colores

Si tienes una resistencia, tomala y mirala bien.
Diras, 'Muy bonitos estos colores, pero que son?'

Existe un codigo de colores para poder calcular el valor de las resistencias.

N§	Color
0	Negro
1	Marron
2	Rojo
3	Naranja
4	Amarillo
5	Verde
6	Azul
7	Violeta
8	Gris
9	Blanco
5%	Dorado
10%	Plateado

[se supone que esto es una resistencia]



No, ahora no lo he dibujado mal. El anillo indicador de la tolerancia se situa mas alejado del resto.

Como calculamos el valor de una resistencia ?
Inmaginemos una resistencia que tiene estos colores :

```
 Primer digito = Marron          Marron = 1
Segundo digito = Negro            Negro = 0
      N§ ceros = Rojo              Rojo = 2
    Tolerancia = Dorado          Dorado = 5%
```

El conjunto quedaria : 1000 5%
Que sucedio ?
Tomamos los 2 primeros digitos y colocamos al final la cantidad de ceros que indica.
Por lo tanto, la resistencia es de 1000 ohms, o 1Kohm, y tiene una tolerancia del 5%

Si aun no se entiende, vamos con otro ejemplo :

```
 Primer digito = Azul               Azul  = 6
Segundo digito = Gris               Gris  = 8
      N§ Ceros = Negro              Negro = 0
    Tolerancia = Plateado       Plateado = 10%
```

El conjunto quedaria : 68 10%
No, aqui tampoco me eqivoque.
Si el la banda que indica el numero de ceros tiene color negro, y el negro es igual a 0, entonces cuantos ceros agregamos.
Correcto, no agregamos nada.
Por lo tanto, la resistencia es de 68 Ohms, con una tolerancia del 10%.

Tolerancia

La tolerancia es uno de los valores mas importantes a la hora de elegir una resistencia.

Pero, que es ?
La tolerancia se usa porque es imposible definir un valor ohimico exacto cuando se fabrica un resistor [ahora si!]
Las margenes de tolerancia mas comunes, son del 5% y 10% .
Existen tambien resistencias con tolerancias del 20%, pero ya no se utilizan.

Como saber si las resistencias que tenemos cumplen con estos margenes de tolerancia ?

[sacar calculadora del mismo lugar de antes : http://www.estante.de/mueble]

Tomemos como ejemplo una resistencia de 1K, con una tolerancia del 10%.
Calculamos el 10% de 1000 (1k = 1000 ohms)

```
10 * 1000
--------- = 100     [no es dificil]
   100
```

Ahora restamos y sumamos ese valor al valor de la resistencia :

1000 - 100 = 900 .... 1000 + 100 = 1100

Ahora tomas tu tester y mides el valor de la resistencia, colocando la perilla selectora en R * 1000 (o 2000 segun la escala que tenga tu tester) y colocas cada punta en cada terminal de la resistencia.

[si muchacho! aprender electronica sin tester es
como aprender a nadar sin meterse en el agua]

Si la medida que obtienes se encuentra dentro de los 2 valores que calculamos arriba, entonces la resistencia se encuentra en buen estado.
Si la medida que obtienes es infinita (la aguja se va a fondo de escala en todas las escalas, o el display presenta un 1 solamente), es que la resistencia ya no conduce (no sirve).
Lo mismo pasa si la resistencia que mides es igual a 0.

Valores estandar

Los valores estandar para la comercializacion [que palabrota] de resistencias son :

```
1   1.5   2.2   3.3   4.7   6.8   10
1.2   1.8   2.7   3.9   5.6   8.2
1.1   1.3   1.6   2   2.4   3   4.3   5.1   6.2   7.5   9.1
```

La primera linea son los valores de las resistencias con tolerancia del 20%
La primera y segunda son para las tolerancias del 10%
Las 3 lineas son para las tolerancias del 5%

La tabla completa se obtiene multiplicando estos valores por 10, 100, 1000, 10000, 1000000 y 10000000.

Para no utilizar muchos numeros, se usa el prefijo K para designar un factor de multiplicacion de 1000, y M para 1.000.000

Potencia

Este es otro factor que tenemos que tener en cuenta cuando elegimos una resistencia.
La potencia es la cantidad de energia en forma de calor que es capaz de soportar.

Cuando una resistencia es atravezada por electricidad, esta pierde una cierta cantidad de la energia, que es ocupada en 'vencer' la dificultad que la resistencia le presenta. Esta energia se disipa en forma de calor.

Como calculamos la cantidad de calor que una resistencia disipa ?
Usando la ley de Joule :

W = I^2 * R , que expresa la potencia desarrollada en vatios.

Veamos un ejemplo :
Tenemos una resistencia de 3 ohm, que es atravezada por una corriente de 0.5 Amperios. Entonces podriamos calcular ...

```
W = (0.5)^2 * 3 ..[la calculadora!]
...W = 0.75 W
```

La resistencia disipa 0.75 W, que obviamente, si no cumple con esa capacidad de disipacion...fire!!!. Tendremos un asado de resistencia.

Tipos de resistencias

Las resistencias mas comunes se construyen con una mezcla de carbon, un material aislante, y un aglutinante [que palabrota].
Luego esta pasta se moldea en forma cilindrica y se le sujetan terminales en sus extremos. El cuerpo de la resistencia se cubre con cera o barniz [o acrilico en estos tiempos] y se marca el codigo de colores.

+ Resistencias piroliticas : Es un cilindro ceramico recubierto por una capa fina de carbon con dos casquillos metalicos en las puntas. El valor ohmico se fija mediante un 'gastado' de la capa de carbon.

+ Resistencias bobinadas : Son las que mas calor pueden disipar. Es un cilindro ceramico [si, otra vez] con hilo resistivo arrollado. Tambien tienen una capa de barniz o... [lo mismo de arriba]

Asociacion de resistencias

Supongamos que necesitamos una resistencia de 66 ohms. Este valor no se maneja comercialmente, entonces : ? Como hacemos para conseguir una ?

Simplemente colocaremos 2 resistencias de 33 ohms en serie.
Como es esto ?

Si colocamos resistencias en serie, como en el dibujo, la resistencia equivalente seria :

```
 33 Ohms  33 Ohms
--\/\/\---\/\/\---          R Total = R1 + R2
   R1       R2
```

Y asi haremos con todas las resistencias que queramos.
La regla para verificar el calculo, es que la resistencia total debe ser mayor que la resistencia de mayor valor.

Si las resistencias estan en paralelo, seria algo asi :

```
    R1                            1       1
.--\/\/\---.         R Total = ----- + -----
|          |                     R1      R2
`--\/\/\---'
    R2               Este calculo se puede convertir en :

                                R1 * R2
                     R Total = ---------
                                R1 + R2
```

En este caso, la resistencia total debe ser menor que la de menor valor. Si no es asi, hay que revisar el calculo.

```
@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@
@%@%  2  -  Resistencias variables y poteciometros  %
@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@
```

Resistencia variable : Dicese de la resistencia fisica de una persona, segun la cantidad de alcohol y nicotina que ingresa a su cuerpo.

[:D]

Las resistencias variables o potenciometros [se dice resistores, no lo olvides] se usan en lugares donde se necesita un control de la corriente que circula. Se usan en controles de volumen, de tono, de luminosidad, etc.

Estructura

La estructura de los potenciometros es casi siempre la misma :
Una resistencia fija que tiene dos terminales, una contacto movil que se mueve sobre la resistencia, y una capsula donde se aloja.
Los potenciometros casi siempre se usan como divisores de tension.

```
o--------.
         |                     Resistencia total = Ra + Rb
    R1   \ Ra
```

```
 VEntrada  / <-------o                                  Rb
           \ Rb      VSalida      VSalida = VEntrada * -------
  o--._____|_________o                               Rb + Ra
```

Tipos

+ Potenciometros de capa de carbon : Se construyen poniendo una capa de resina plastica, en forma de anillo. Arriba de esta capa se pone una capa de carbon, mezclado con resina liquida, que luego se hornea [no es comida :)] para que se endurezca.
  Luego se le coloca el cursor y todo lo demas.
  Los valores para estos potenciometros se encuentran entre los 50 ohms y los 10M Ohms.
  La potencia que pueden disipar casi siempre es de 2 W.

+ Potenciometros bobinados : No voy a decir lo mismo dos veces. Es como una resistencia bobinada con un cursor que se mueve sobre ella.
  Los valores mas comunes se encuentran entre 50 ohms y 10K ohms
  Pueden llegar a disipar hasta 1000W en los modelos mas grandes.

+ Multivueltas : Los potenciometros comunes solo giran en un angulo de 270§ pero los multivueltas giran de 2 a 10 vueltas [puede girar mas]
  La presicion que se alcanza con estos potenciometros es muy elevada.

Variacion

Existen 3 tipos principales de variaciones cuando se fabrican potenciometros.

+ Lineal : La resistencia aumenta en la misma proporcion en todo el recorrido del potenciometro.

+ Logaritmico : La variacion es lenta al principio, pero es elevada al final. Casi siempre se usa como potenciometro de volumen, porque da el aspecto de que el volumen es mas elevado.

+ Tambien existe la seno-coseno, donde la variacion es rapida-lenta-rapida.

```
@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@
@%@%  3  -  Resistencias especiales  @%@%@%@%@%@%
@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@
```

Resistencias NTC

En la entrega anterior del curso, dijimos que la resistencia de los materiales variaba segun la temperatura.
Las resistencias NTC se construyen segun esta propiedad, de modo que su valor ohmico es menor, mientras mayor sea la temperatura.
Su valor nominal se define casi siempre para 25§C.
Por cada grado que aumenta la temperatura, la resistencia disminuye su valor entre un 1% y 9% en los modelos mas comunes.

Resistencias PTC

Funcionan de manera inversa a las NTC.
Las resistencias PTC y NTC son usadas en termometros, controles de temperatura, circuitos de proteccion, etc. Tambien se les llama termistores.

Resistencias VDR

Tambien se les llama varistores. La funcion que tienen es la de disminuir su resistencia a medida que la tension aumenta. Se usan en circuitos de proteccion y regulacion de tension.

Bandas extensiometricas   [que palabrota]

Son bandas de material resistivo que se pegan a una pieza que se somete a estiramiento o presion. Cuando la longitud de la pieza varia, tambien

varia la resistencia de la banda.

```
@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@
@%@%   3  -  Capacitores o condensadores  @%@%@%@%
@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@
```

El condensador es uno de los componentes que no suelen faltar en ningun equipo electronico.

Como se forma un condensador ?
Basicamente [mente en basic :)], se forma con 2 placas metalicas separadas por un aislante que se llama dielectrico [que palabrota].
El dielectrico puede ser aire, papel, mica, ceramica, etc.

Casi siempre, el dielectrico se forma con laminas muy finas, para conseguir que las placas metalicas [se dice armaduras] se encuentren muy cerca una de otra.

```
                           _________|_________  _________
                        __|____________________|__        |
      Dielectrico <---- |_________________________|       |----> Armaduras
                          |____________________| _________|
                                    |
```

La capacidad del condensador se determina por la superficie que tienen las armaduras, y el espesor del dielectrico.
Mientras mas grandes sean las armaduras, y menor sea el espesor del dielectrico, mayor sera la capacidad del condensador.

Polarizacion
""""""""""""

Si tomamos un condensador, y le aplicamos una tension continua, no habra ninguna circulacion de corriente, porque el dielectrico es aislante.

Pero ya aprendimos en el capitulo anterior, que no todo es gris en el cielo.
La corriente continua produce una acumulacion de cargas en las armaduras. Esto quiere decir que en donde se conecte el polo negativo, se produce una acumulacion de electrones, y en el polo positivo, una disminucion de electrones.
Lo mismo pasa con el dielectrico, porque esta en contacto con las armaduras. Esto se llama polarizacion del dielectrico.

Si ahora desconectamos la tension, la acumulacion de cargas se mantiene, porque como las cargas son de diferente signo, tienden a atraerse.
Si juntamos las armaduras o las cortocircuitamos, habra una corriente entre ellas [con chispazo si las cargas son grandes], el condensador se descargara, y quedara en las condiciones iniciales.

Capacidad
"""""""""

Teniendo en cuenta lo de arriba, deducimos que la capacidad de un condensadores la posibilidad de acumular cargas electricas
El material que se utiliza para construir el dielectrico es la eleccion mas importante cuando se construye un condensador, porque determina la maxima tension que el condensador soportara, la capacidad y la corriente maxima.

La capacidad de un condensador se puede definir como la cantidad de carga que el condensador almacena cuando se le aplica una tension de 1V.

```
             Q          C = Capacidad en faradios   Q = carga en culombios
C = Er *   -----       Er = Permitividad relativa   V = Tension en voltios
             V
                       [ en realidad no es una letra 'E', sino que
                         es la letra griega epsilon, que se usa
                         para simboliazr la permitividad.
                         Es el codigo ascii 238 (î) ]
```

Tambien podemos usar nuestra 'querida' calculadora para calcular [obvio] la capacidad de un condensador segun sus dimensiones geometricas :

```
                 A       Ev = Permitividad del vacio [ es 1 ]
C = Er * Ev * -----       d = Espesor del dielectrico (en metros)
```

d A = Superficie de las armaduras (en m^2)

Si aun no se aburren de tantos calculos, tambien podemos calcular la energia contenida en un condensador :

$$W = \frac{1}{2} * C * V^2$$ W = Energia (en julios)

Permitividad [que palabrota]

Seguramente habran pensado como hice para saber la permitividad del vacio..
Pues, para satisfacer su hambre de conocimiento, aqui tienen la permitividad de otros materiales que se usan para construir condensadores.

Vacio	1
Aire seco	1.00059
Polietileno	2 a 3
Papel impregnado	4 a 6
Vidrio	4 a 7
Mica	4 a 7
Porcelana c/ dioxido de titanio	80 a 100
Porcelada fina	Mas de 1.500

Tension alterna

Supongamos que tu, lector, yo, escritor, y el correspondiente editor ; nos encontramos caminando por una calle cualquiera [con unas correspondientes copas de mas], y decidimos conectar un capacitor a corriente alterna.
Que sucedera .... !La hecatombe! !La debacle total! Una sucesion de hechos bochornosos en los que participan : el editor, elotro, el lector, el capacitor, menem, james hetfield, ivan noble y la AC!

Bueno, en realidad no sucede eso.
Lo que suceda con el capacitor despues de conectarlo a la AC, es una consecuencia de lo que sucede cuando se lo conecta a CC.

Como la AC varia cada cierto tiempo, la carga del condensador tambien varia a la misma frecuencia de la AC. El efecto es una circulacion de corriente, aunque esta no pase por el interior del condensador.

Aqui llegamos a una de las principales caracteristicas de un condensador :
Separar corrientes continuas de las alternas, cuando estas existen simultaneamente [que palabrota] en el mismo conductor.

Tipos de condensadores

Estas son las caracteristicas de los tipos de condensadores mas comunes, salvo el tipo 'plate' del que busque informacion, pero solo encontre que se usa en radiofrecuencia [cosa que ya sabia], y que se construyen con peliculas de plata [cosa que no sabia]

+ Condensadores de papel impregnado : Se fabrican arrolando las armaduras, que son hojas de Al, Sn o Cu, de unos 0.006mm de espesor, entre otras 2 hojas de papel de 0.007mm a 0.01mm de espesor.
  El papel se impregna con cera o algun aceite adecuado.
  La tension de trabajo de estos capacitores depende del espesor del papel, pudiendo alcanzar los 200000 voltios en algunos modelos.

+ Condensadores de papel metalizado : En una lamina de papel se coloca una fina [muy] lamina de metal. Se arrollan 2 de estas laminas, y se conectan los terminales.
  Son de gran capacidad, y pequenios. No se deben utilizar con tensiones menores que 10 V.

+ Condensadores de mica : La mica es un mineral muy estable, y se utilizaba como dielectrico para condensadores de radiofrecuencia. Como ya dije mas arriba, estos condensadores se sustituyeron por los de tipo 'plate'.

+ Condensadores ceramicos : Se construyen con materiales de este tipo, y se distribuyen en forma de 'lenteja' o en capsulas plasticas o metalicas.

+ Condesadores de plastico : Se utilizan mucho en estos dias. Estan formados por 2 tiras de poliester, cada una metalizada en un borde, y se colocan de manera que los bordes metalizados queden opuestos uno con otro.
Tienen la ventaja que si se supera la tension maxima, el dielectrico se perfora y el metal se vaporiza en la zona de la perforacion, con lo que siguen funcionando.

+ Condensadores electroliticos de aluminio : Son los de mayor capacidad junto con los de tantalo. Estan formados por una hoja de aluminio, cubierta con una capa de oxido que actua como dielectrico. La lamina esta impregnada de un acido denominada electrolito.
Se usan solo con tensiones continuas, y si la polaridad no se respeta, pueden llegar a estallar [no es C4, es un capacitor solamente]

+ Condensadores electroliticos de tantalo : Son similares a los de aluminio, pero en estos el electrolito suele ser de tantalo seco.
Se pueden usar con AC y CC.

Midiendo la capacidad

La capacidad de los condensadores se mide en una unidad llamada faradio, pero es excesivamente grande.
[un condensador de 1 faradio puede ser grande como mi casa. Es verdad]
Para solucionar esto, se toman otras unidades, que son fracciones del faradio :

+ Microfaradio o millonesima de faradio (0.000001 F)
Se representa con uF [en realidad no es una letra 'u', sino que es la letra griega mu]

+ Nanofaradio o milmillonesima de faradio (0.000000001 F)
1 nF = 0.0001 uF) . Se representa con nF

+ Picofaradio o billonesima de faradio (0.000000000001 F)
1 pF = 0.000001 uf , 1 pF = 0.001 nF
Se representa con pF

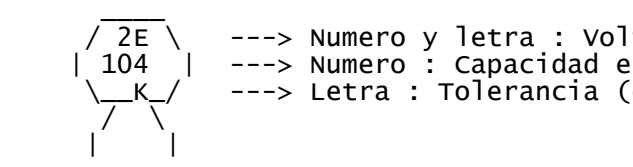
Para que el electronico no se confunda a la hora de medir la capacidad de los condensadores [o para que se confunda mas, como en mi caso, tambien se usa una designacion similar a la de las resistencias.
En este caso, se usa la letra 'K' para el picofaradio, es decir que 1 nF es igual a 1 KpF, o sea 1000 pF.
En general, cuando se vea en el cuerpo de un capacitor un valor, seguido por la letra 'K', se sabra de antemano que la capacidad esta expresada en picofaradios.

Esto nos lleva a un tema muy interesante... [y complicado a la vez]

Codigo JIS

El codigo JIS se usa mayormente en capacitores de poliester y ceramicos. Sinceramente, no tengo np-idea a porque lleva ese nombre [tampoco busque] Bueno, espero que alguien se acuerde de la persona que creo este codigo, y si aun vive, preguntarle porque lo hizo tan complicado.

Tomemos por ejemplo un capacitor ceramico :
[aclaracion : se supone que los capacitores ceramicos tienen forma redonda, bastante parecida a una lenteja, asi que no los comas]



Primero que nada, empecemos con el numero de la 2¦ fila : 104
Tomamos los dos primeros digitos : 10
Agregamos una cantidad de ceros que esta indicada por el ultimo numero : 4
La capacidad seria : 10 + (4 ceros) = 100000 pF

Para expresar este valor en uF, dividimos por 1000000
Para expresarlo en nF, dividimos por 1000

En general, para pasar de :

pF a uF = dividimos por 1000000 [no coloco separador de miles, porque
pf a nF = dividimos por 1000   yo uso '.' como punto decimal]

nF a uF = dividimos por 1000
nf a pF = multiplicamos por 1000

uF a nF = multiplicamos por 1000
uF a pF = multiplicamos por 1000000

Ahora vamos con la primera fila : Voltaje maximo, que en este ejemplo es 2E
[no es hexadecimal ni nada parecido]
Estas son las equivalencias :

1H -------> 50 V	2E -------> 250 V
2A -------> 100 V	2G -------> 400 V
2T -------> 150 V	2J -------> 630 V
2D -------> 200 V	

Por ultimo, la tolerancia :

F -------> 1 %	J -------> 5 %
G -------> 2 %	K -------> 10 %
H -------> 3 %	M -------> 20 %

O sea, que el capacitor que tomamos por ejemplo, era de 100000 pF, su voltaje maximo era 250 V, y su tolerancia era de 10%.

Valores estandar

Como en el caso de las resistencias, los capacitores tambien cumplen con una serie de valores estandar :

1 1.2 1.5 1.8 2.2 2.7 3.3 3.9 4.7 5.6 6.8 8.2

Si suponemos que estos valores estan en uF, entonces deducimos que la tabla completa se obtiene multiplicando por 10, 100 y 1000
Y los valores de las unidades menores (pf, nf), se obtienen dividiendo por 10, 100, 1000 y 10000.

Asociacion de capacitores

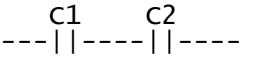
Supongamos que necesitamos un capacitor de 4400 uF. Este valor no se maneja comercialmente, entonces : ? Como hacemos para conseguir uno ?

Simplemente colocaremos 2 capacitores de 2200 uF en paralelo.
Como es esto ?
[es al contrario de las resistencias]

Si colocamos capacitores en paralelo, como en el dibujo, la capacidad equivalente seria :

Capacidad Total = C1 + C2

Si los capacitores estan en serie, seria algo asi :



$$\text{Capacidad Total} = \frac{1}{\frac{1}{C1} + \frac{1}{C2}}$$

Este calculo es equivalente a :

$$\text{Capacidad Total} = \frac{C1 * C2}{C1 + C2}$$

Y asi procederemos con todos los capacitores que queramos.

@%
@%@% 5 - Bobinas o inductancias @%@%@%@%@%@%@%@
@%

Las inductancias o bobinas son otros de los componetes mas utilizados a la hora de construir circuitos elcetronicos, sobre todo para el tratamiento de corrientes o seniales alternas.
Tambien son conocidas con el nombre de reactancias o choques, segun a la aplicacion que se destinen.

Las bobinas se componen de 2 partes basicas : El arrollamiento de hilo, y el nucleo. Segun como queramos que sea el disenio de la bobina, o la funcion que cumplira, el arrollamiento puede estar montado sobre un carrete.

El aspecto externo de muchas bobinas es muy parecido al de los transformadores, y se usan los mismos materiales para su construcion.

Ya se que en mis articulos digo que no todo es gris en el cielo, pero en el caso de las bobinas, el cielo no se presenta de color gris.
Cuando se trata de bobinas, el cielo es negro.
Casi todas las veces que utilizemos bobinas, tendremos que construirlas nosotros con nuestras propias manos y cerebro.

[busque por internet, pero el cerebro tendremos que conseguirlo nosotros, y lamentablemente yo tengo uno solo que funciona a medias]
[las manos pueden ser de cualquier persona que disponga de la suficiente cantidad de cerebro]

Cuando estamos disenyando una bobina, lo mas imprtante a tener en cuenta, es si la bobina va a ser atravezada por CC o AC.

Porque tenemos que saber esto ?
En la inmensa mayoria de las veces, los choques se usan como filtros de rectificadores, en los sistemas que convierten corriente continua en corriente alterna.

Para construir este tipo de bobinas, casi siempre se usan los mismos materiales que cuando se construyen transformadores. Tambien se suele usar un carrete E-I [ver seccion 7 - Transformadores].

Unidades de medida

La inmensa mayoria de las bobinas se miden en una unidad llamada Henrio (H) Tambien se puede utilizar el Gauss (G), o el oersted (o).

El henrio se calcula como la inductancia que posee una bobina que es atraveada por una tension de 1V, que varia a razon de 1A por segundo.
Como las otras unidades, tambien tiene submultiplos :

+ Milihenrio (mH) : 0.001 H
+ Microhenrio (uF) : 0.000001

Permeabilidad magnetica

El nucleo que se le coloca a las bobinas es uno de los factores mas improrantes a la hora de calcular la inductancia.
Existen materiales (como el hierro ferrita) que poseen una permeabilidad magnetica muy grande, y otros que no tienen ninguna, como los plasticos, o el aluminio.
La permeabilidad magnetica se representa con la letra griega mu (u)

[ya se que esa no es la letra mu, pero esto es ascii muchacho, no unicode]

Mientras mayor sea el valor de la permeabilidad magnetica, mayor sera la inductancia de la bobina sobre un numero de espiras fijo.

Factor de calidad

Hay otro valor muy importante, que relaciona la inducatancia de la bobina

con la resistencia del hilo. Este factor define la calidad de la bobina, que sera mayor mientras mayor sea la inductancia respecto a la resistencia.
  El factor de calidad se representa con la letra Q.

  Ahora... a sacar la calculadora !

```
      2 * PI * f * L        L = Largo de la bobina
  Q = ----------------      f = Frecuencia a la que se somete la bobina
             R              R = Resistencia del hilo
                            2 = No creo que haga falta explicar esto.
```

  PI = 3.14159265358979323846264 33 [o algo asi, pero si el procesador es pentium y de la primera version puede dar cualquier numero, menos este]

Coeficiente de autoinduccion [+ palabrotas]

  Que es esto con palabras tan raras?
  Es la relacion entre el flujo magnetico producido, y la intensidad que lo produce.

  Calculadora otra vez...

```
         N^2 * S            N = Numero de espiras
  L = u ---------           u = Es la letra griega mu!! [a veces odio el ascii]
            l               l = Longitud de la bobina en metros
```

  S = Seccion efectiva del nucleo.

    [ver seccion de los transformadores, porque no pienso copiar la misma cosa, para que lo tengas que leer 2 veces, y para que el archivo sea mas grande aun, porque en algun rincon del mundo debe haber lectores que usan un ordenador viejo como el mio, y no quiero ocupar memoria al dope]

Mas calculadora

  Vamos a suponer de que queremos armar una bobina, que no tenga nucleo.
  [bah, en realida si tiene un nucleo, que es de aire]
  Si tomamos la permeabilidad magnetica del aire y queremos calcular la inductancia de la bobina [que obviamente, tendra nucleo de aire] podemos usar esta ecuacion ..

```
                   n^2 * S         L = inductancia en microhenrios
  L = u * 1.257 * ----------       u = Permeabilidad magnetica (aire = 1)
                  10^8 * l         1.257 es una constante
                                   n = numero de espiras
                                   S = Seccion del nucleo(seccion = PI * Radio)
                                   l = largo de la bobina
```

  Pero no podemos ponernos a calcular todo esto, asi que lo que haremos es definir algunos valores para la bobina, y nos dedicaremos a calcular el numero de espiras que llevara :

```
          _______________
      \  |  L * l * 10^8
  n =  \ | -------------
        \|   1.257 * s
```

```
  @%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@
  @%@%  6  -  Reles  @%@%@%@%@%@%@%@%@%@%@%@%@%@%@%
  @%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%%@%@%@%
```

6 - Reles

 [o relevadores, o relays, o en 2 palabras : cualquiercosa queseleparezca]

  El nombre rele viene de la palabra francesa 'relais', que significa relevador. Si ponemos la materia gris a trabajar, podremos deducir que es una especie de interruptor, que se usa para abrir o cerrar circuitos electricos.
  Si buscamos, podremos encontrar 2 tipos de reles : electromagneticos y estaticos. Si no buscamos, no vamos a encontrar nada. :)

  Comencemos con los reles electromagneticos ..

  Estructura del rele

""""""""""""""""""""
Los reles se pueden dividir en 2 bloques fundamentales :

+ Circuito de excitacion , que es el encargado de recibir la senial de comando, que puede ser una corriente, pero lo mas comun es que sea una tension. Esta formado por una bobina y un conjunto magnetico. Este circuito se encarga de realizar las acciones que hagan falta para accionar el circuito de conmutacion.

+ El circuito de conmutacion es el conjunto de contactos que son accionados por el circuito de excitacion. Normalmente este circuito se encuentra en disposiciones normal abierto, mormal abierto y cerrado, y tambien se encuentra esta misma disposicion, pero en forma doble.

Enclavamiento del rele
""""""""""""""""""""""
Existen algunos reles que cuentan con un sistema de enclavamiento mecanico o magnetico.

Que es el enclavamiento ? [la palabrota]
El enclavamiento se usa para, cuando el rele fue excitado, una vez que se retire la senial que lo excito, permanezca en el estado en que estaba cuando fue excitado.

[leer eso 2 o 3 veces hasta entender]

El encalvamiento mecanico se constituye con un sistema de levas, que hace que los reles sean mas dificiles de diseniar y de construir.

[y eso hace que tengamos que gastar mas dinero cuando compramos un rele de estas caracteristicas]

Existen otros reles que actuan con enclavamiento magnetico [con un iman]

Reles estaticos
"""""""""""""""
Basicamtente son circuitos que cumplen la misma funcion que un rele, pero estos circuitos no cuentan con partes moviles.
Obviamente, tienen ventajas sobre los reles electromagneticos, porque pueden tener una mayor velocidad de conmutacion, y acciones practicamente ilimitadas, porque no hay partes moviles que se desgasten o rompan.
Me gustaria seguir explicando esta seccion aqui, pero en la proxima entrega [o en otra mas adelante] habra un circuito que es un rele estatico con triac, asi que lo podran aprender ahi.

@%@
@%%@ 7 - Transformadores @%@%@%@%@%@%@%@%@%@%
@%@

Si, el transformador...ese aparatito de forma rara que cuesta tanto dinero y que me costo tanto hacerlo funcionar..(si es que armaste la fuente de la primera entrega)

Bueno, pero no estamos aqui para que me insultes por haber puesto una fuente con transformador, tan clasica, en vez de una fuente conmutada..
Pero que es eso...? Mejor espera a las proximas entregas, aun no queremos adelantar conocimientos.

[para aprender a correr, primero hay que saber caminar, y antes saber gatear]

Bueno, ya me estoy yendo por las ramas..mejor volvamos con el transformador.

El transformador (trafo pa los amigos) es un interesantisimo elemento que es capaz de convertir corrientes alternas de determinada corriente e intensidad, en otra corriente alterna con intensidad y corriente diferentes a las originales.
El trafo se basa en un fenomeno fisico [de la fisica], que es el de induccion electromagnetica.

El funcionamiento del trafo se basa en este fenomeno, de forma que un campo magnetico variable que produce un conductor arrollado sobre un nucleo de

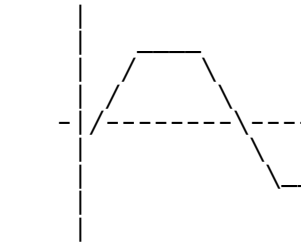
material magnetizable, cuando circula por el una corriente alterna, produce una determinada tension, que tambien sera alterna y de la misma frecuencia. Esta tension es independiente de la tension anterior, y se presenta en otro conductor independiente, arrollado sobre el mismo nucleo.

En resumen : Dos bobinas que estan arrolladas sobre el mismo nucleo, sin importar cual sea el material que lo forme, forman un transformador.

[fijate bien : sin importar el material del nucleo. Esto lo vamos a aplicar en proximas entregas, cuando veamos transmision de sonido por metodos inalambricos]

En efecto, cuando por una bobina de hilo conductor circula una corriente electrica, esta bobina se comporta como un iman. Este efecto se produce con corrientes continuas y corrientes alternas.
Para que se entienda mejor, diremos que alrededor de la bobina se extiende un campo magnetico que sera constante si la corriente es continua, pero si la corriente que circula por la bobina es alterna, el campo magnetico sera fluctuante.

Porque fluctuante ?
Recordemos como es la corriente alterna :
[por ejemplo la tension de red]



[si, si; ya lo se. La corriente alterna que circula por la red no tiene esa forma, sino que es senoidal. Pero yo pregunto algo : ?Como se dibuja un sinusoide con caracteres ascii?]

No vamos a meternos en el interior de la forma de onda, ni nada de eso.
[si lo haremos en proximas entregas. Recuerda que en el primer articulo dije que iba a ser una larga serie...]

Mejor veamos como es el recorrido que realiza la onda :
Muy bien, la onda sube y baja.
Es decir, que en un determinado momento, tiene una polaridad positiva, y en otro momento tiene una polaridad negativa.
Por lo tanto, el campo magnetico sigue el movimiento de la onda : En un momento las lineas de fueza apuntan hacia un sentido, y en otro, apuntan a un sentido opuesto.

Ok, ya sabemos porque el campo es fluctuante.
Este campo magnetico ejerce una fueza [magnetica], y la ejerce sobre todos los objetos que lo rodean. No hay que ser muy vivo para darse cuenta que la fuerza magnetica sera mayor en los objetos mas cercanos que en los que esta mas lejos del campo.

Si arrollamos otra bobina que llamaremos pirmario [fijate bien, ya arrollamos una] sobre el mismo nucleo, estara atravezada por el campo magnetico que teniamos.
Si este campo es constante, la otra bobina estara haciendo de estorbo, porque no tendremos ningun efecto util sobre la otra bobina, que llamaremos secundario.

Pero, si el campo es variable, tal como seria si aplicamos una corriente alterna en el primario, en el secundario tambien se inducira una corriente variable tambien, si es que se encuentra conectado a un circuito cerrado.
La corriente inducida es proporcional a la velociadad del cambio de la intensidad de la corriente del primario y sera muy similar a la del primario. [quiere decir que tendra la misma frecuencia]

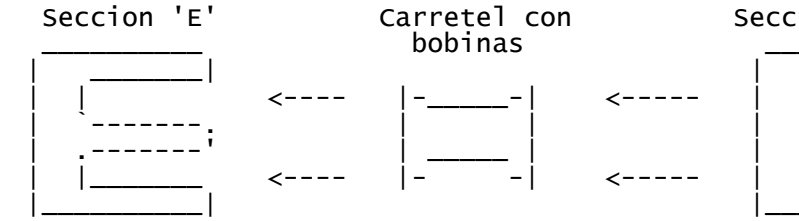
En cambio, la intensidad y/o la tension pueden ser mayores, menores o iguales a la del primario. Los valores pueden cambiar porque el numero de espiras de cada bobina puede ser diferente.

Ahora que ya comprendimos como funciona un transformador, vamos a divertirnos un poco con nuestra querida amiga, la calculadora ...

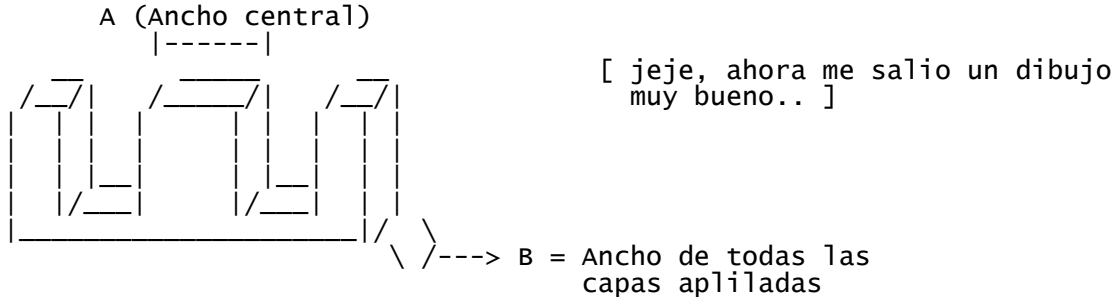
Construccion de un transformador

"""""""""""""""""""""""""""""""""""""

Cuando hablamos de las bobinas, explique que algunas se fabrican con un nucleo E-I, como el de los transformadores.
Pero como es este nucleo ?



Si, ya se que has visto un transformador. Pero lo que queremos hacer aqui, es tomar 2 medidas elementales a la hora de armar nuestro propio transformador, que podremos ocupar en una fuente de alimentacion o en lo que queramos [una fuente de alimentacion casi seguro]



Vamos a trabajar con las medidas reales de un transformador que tengo aqui mismo donde estoy escribiendo.
Este transformador [en realidad no es el transformador, es el nucleo] tiene un ancho(A) de 3.2 cm, y un aplilado de 3.9 cm

Una vez que sabemos estas medidas, podemos comenzar a trabajar.
Primero que nada, calculemos la seccion del nucleo :

Sec(nuc) = A * B .... S(nuc) = 3.2cm * 3.9cm = 12.5 cm^2

Porque centimetros cuadrados ?
Porque no solo multiplicamos los numeros, sino que las unidades tambien se multiplican. Entonces cm*cm = cm^2.

Dijimos que la seccion del nucleo era de 12.5 cm^2 [en adelante cm2]
Pero nada es perfecto. El nucleo, por mas bueno que sea, no tiene una eficacia del 100%. Segun el estado del nucleo, podemos tener una eficacia de:

Estado del nucleo	Eficiencia
Excelente	95 a 98 %
Muy bueno	90 %
Bueno	80 %
Regular	70 %
Malo	65 %

Si el nucleo es regular o malo, no conviene usarlo, porque va a generar muchas perdidas y mucho calor.

Ok, tomemos la seccion del nucleo y calculemos su eficiencia
[se dice asi, o eficacia, o efectividad ?]
En mi caso, el nucleo del transformador no es excelente, pero aun asi esta en buen estado, asi que vamos a calcular una eficiencia del 80 %

Sec(Efectiva) = 12.5 cm2 * 0.8
Sec(Efectiva) = 10 cm2

Entonces tenemos una seccion efectiva de 10 cm2.
Ahora calculemos cual es la potencia bruta del nucleo :

Potencia(Nucleo) = Sec(Efectiva) ^ 2
Potencia(Nucleo) = 10 * 10 [en este caso no colocamos las unidades, porque sabemos que la potencia se mide en watios]
Potencia(Nuclec) = 100 W

Dejemos la potencia de lado por un momento, y pasemos a calcular la relacion de transformacion, que se calcula en la cantidad de vueltas por volt. Volveremos a usar la potencia cuando calculemos la intensidad de los bobinados.

$$\frac{\text{Vueltas}}{\text{Volt}} = \frac{\text{Constante (45)}}{\text{Sec(Efectiva)}} = \frac{\text{45 vueltas * cm2}}{\text{10 cm2}}$$

Simplificamos cm2 (los tachamos), y nos queda lo siguente :

$$\frac{\text{45 vueltas}}{10} = 4.5 \text{ vueltas}$$

Una vez que obtenemos estos datos, podemos pasar a calcular el numero de vueltas que tendra el bobinado primario :

$$\text{Vueltas(Primario)} = \text{Voltaje deseado} * \frac{\text{Vueltas}}{\text{Volt}}$$

No se cual es la tension de red que se trabaja alla en la madre patria (porq yo soy de argentina), pero seguramente es de 220 V, o 110 V.
En el caso de argentina, la tension es de 220 V.
Vamos a tomar esta como ejemplo.

$$\text{Vueltas(Primario)} = 220 \text{ V} * \frac{4.5 \text{ vueltas}}{\text{Volt}}$$

Simplificando volt con volt, resultaria en esto :

Vueltas(Primario) = 220 * 4.5 vueltas
Vueltas(Primario) = 990 vueltas

Ahora vamos con el numero de vueltas del secundario :

$$\text{Vueltas(Secundario)} = \text{Voltaje deseado} * \frac{\text{Vueltas}}{\text{Volt}}$$

En este caso, yo opino que 40 V serian mas que suficientes.

$$\text{Vueltas(Secundario)} = 40 \text{ V} * \frac{4.5 \text{ Vueltas}}{\text{Volt}}$$

Simplificamos volt con volt otra vez..

Vueltas(Secundario) = 40 * 4.5 vueltas
Vueltas(Secundario) = 180 vueltas

Muy bien, ya sabemos cuantas vueltas de alambre llevara cada bobinado.
Ahora pasemos a calcular la intensidad de corriente que circulara por cada uno de los bobinados.
Este calculo es mera ley de ohm.

$$\text{Intensidad(Primario)} = \frac{\text{Potencia(Nucleo)}}{\text{Voltaje(Primario)}} \text{ ... } \frac{100 \text{ W}}{220 \text{ V}} = 0.45 \text{ A}$$

$$\text{Intensidad(Secundario)} = \frac{\text{Potencia(Nucleo)}}{\text{Voltaje(Secundario)}} \text{ ... } \frac{100 \text{ W}}{40 \text{ V}} = 2.5 \text{ A}$$

Muy bien, ya tenemos bastantes datos, pero aun no podemos armar el transformador, porque no podemos colocar un alambre cualquiera.
Tenemos que saber el diametro del alambre para poder comprarlo en una ferreteria o en una tienda de articulos electronicos.

Primero, vamos a calcular la seccion del alambre del bobinado primario. Vamos a trabajar con una constante, la letra griega delta(d), que representa la cantidad de corriente que el alambre de cobre comun puede soportar sin problemas. Este valor es de 3A/mm2.

```
                             Intensidad(Primario)      0.45 A
  SeccionAlambre(Primario) = -------------------- ... --------  = 0.15 mm2
                                      d                 3 A
                                                       -----
                                                        mm2


                               Intensidad(Secundario)      2.5 A
  SeccionAlambre(Secundario) = ---------------------- ... ------- = 0.83 mm2
                                        d                   3 A
                                                           -----
                                                            mm2
```

Ahora calculemos el diametro de cada alambre :

```
                                  ____________________________
                                \| SeccionAlambre(Primario) '
  DiametroAlambre(Primario) = 2 * ----------------------------- = 0.29 mm
                                             PI

                                   ______________________________
                                 \| SeccionAlambre(Secundario) '
  DiametroAlambre(Secundario = 2 * -------------------------------- = 0.58 mm
                                               PI
```

Obviamente, no conseguiremos alambre de .29 mm, o de .58mm, sino que compraremos uno que se le aproxime. Si el alambre es mas ancho aun, mejor aun.

Bueno, ya terminamos con el transformador. Lo que obtendriamos seria un transformador con estas caracteristicas :

Apilado = 3.9 cm
Ancho central = 3.2 cm
Seccion efectiva = 10 cm2
Potencia bruta del nucleo = 100 W
Relacion de transformacion = 4.5 vueltas / volt
Numero de vueltas del primario = 990 vueltas
Numero de vueltas del secundario = 180 vueltas
Voltaje de entrada = 220 V
Voltaje de salida = 40 V
Intensidad del primario = 0.45 A
Intensidad del secundario = 2.5 A
Diametro del alambre del primario = 0.29 mm (o sea, .30 mm o mayor)
Diametro del alambre del secundario = 0.58 mm (.60 mm o mayor)

Bueno, ahora descansa, ve a comprar colirio y tira bien lejos la calculadora.

No voy a tratar el tema de los transformadores de audiofrecuencia o radiofrecuencia, porque eso es adelantar conocimientos.

@%@
@%%@ 8 - Cristales de cuarzo @%@%@%@%@%@%@%@%@%
@%@

Seguramente habras visto un cristal de cuarzo, en alguan placa de ordenador o en algo parecido. Por fuera se ven como una caja plateada con unas letras escritas, pero son mucho mas que eso.
Si prestaste atencion a la seccion de osciladores, en la entrega anterior, recordaras que hable de los cristales de cuarzo, pero muy basicamente.
Ahora nos pondremos a trabajar en serio.

Estructura
""""""""""

El cristal de cuarzo que uno puede comprar en una tienda de electronica, generalmente viene en una capsula metalica. Dentro de la caja, hay una pieza

de cuarzo, que casi siempre es de forma circular o cuadrada, con unas metaliaciones [que palabrota] en cada extremo.
  Si, si, posiblemente sabias eso [posiblemente tambien no lo sabias] pero, ?Que es el cuarzo?

  El cuarzo es anhidrido de silicio (SiO2), que se encuentra en la naturaleza en muchas formas diferentes, pero la que a nosotros nos interesa es la que esta compuesta por cristales prismaticos hexagonales, acabados en piramides por sus extremos.

  [Ah no! Eso si que no! Esta bien que dibuje los circuitos esquematicos, tambien algunos componentes, e incluso, dibuje el nucleo del transformador en perspectiva. Pero un cristal prismatico hexagonal acabado en piramide por sus extremos, es demasiado para mi.]

Efecto piezoelectrico [que palabrota]
[Juro que es la ultima vez que digo eso]

  Una vez que se recortan las laminas de cuarzo desde el cristal, tienen un efecto piezoelectrico que es el que nosotros aprovechamos.

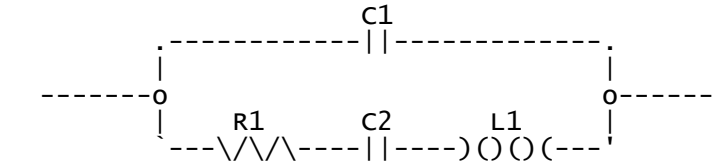
  Que es el efecto piezoelectrico ?
  Si tomamos la lamina de cuarzo, y le aplicamos una tension entre sus dos caras, se origina una deformacion magnetica, o sea, una oscilacion.
  Si en vez de aplicar la tension, colocamos un medidor en sus dos caras, y sometemos la lamina a vibracion, el efecto sera inverso, o sea que obtendremos una tension.

Resonancia del cristal

  Si en vez de aplicarle una tension continua, le aplicamos una alterna de la misma frecuencia a la que varia la lamina, de forma que se encuentren en resonancia (las 2 al mismo tiempo), la oscilacion sera reforzada, e incluso estabilizada.
  La resonancia desaparece si la tension alterna no sigue la misma frecuencia del cristal, y entonces la oscilacion se detiene.

Circuito equivalente del cristal

  Si, si. Ya se que en la segunda entrega tambien aparece, pero ya que estamos hablando de cristales, me parecio correcto colocarlo aqui tambien.



Formas de operacion

  Los cristales de cuarzo pueden trabajar de dos modos : Resonancia serie, y resonancia paralelo. Las frecuencias que obtenemos de estos modos son diferentes, y en el momento del disenio del cristal, se disenian especificamente [q pal...(esta es la ultima vez, en serio)] para que se usen en un solo tipo de resonancia.

  [tambien lo podemos usar en el otro modo, pero el cristal no trabajara correctamente]

  Si a un cristal lo hacemos trabajar en resonancia serie, solamente deberemos tener en cuenta la resistencia serie del circuito. Pero, si el cristal trabajara en resonancia paralelo, tambien tenemos que tener en cuenta la capacidad propia del cristal.

  En algunos modelos, sobre todo los de alta frecuencia, no se hace trabajar al cristal en su frecuencia natural de oscilacion, sino que se lo hace trabajar en multiplos de la frecuencia (armonicos), llegando hasta el 6§ o 7§ armonico en algunos casos.

  Algunos cristales (la mayoria) estan encapsulados al vacio, haciendo que el aire no frene la oscilacion del cristal.

[inmagina el efecto que tiene el aire cuando un
  cristal vibra a 600 o 700 MHz !]

Otra caracteristica que tenemos que tener en cuenta, es la variacion de frecuencia con la temperatura, que casi siempre se expresa en partes por millon (ppm/§C), y en forma de porcentaje.
Por ejemplo, si tenemos un cristal de 1 MHz con una variacion de 20ppm/§C, por cada grado que varie la temperatura, la frecuencia podra estar entre 999800 Hz, y 1000200 Hz.

Teniendo esto en cuenta, tambien existen modelos de cristales que vienen con una resistencia de calentamiento, de modo que no influya la temperatura externa.

```
%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%
@%@%  9  -  Montajes practicos  %@%@%@%@%@%@%@%@%@%
%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%
```

Se que he dicho muchas veces que no debemos adelantar conocimientos, pero la cuarta entrega se basara en algunos circuitos integrados muy comunes, y en este caso, 1 segundo de practica vale mas que 10 minutos de teoria.

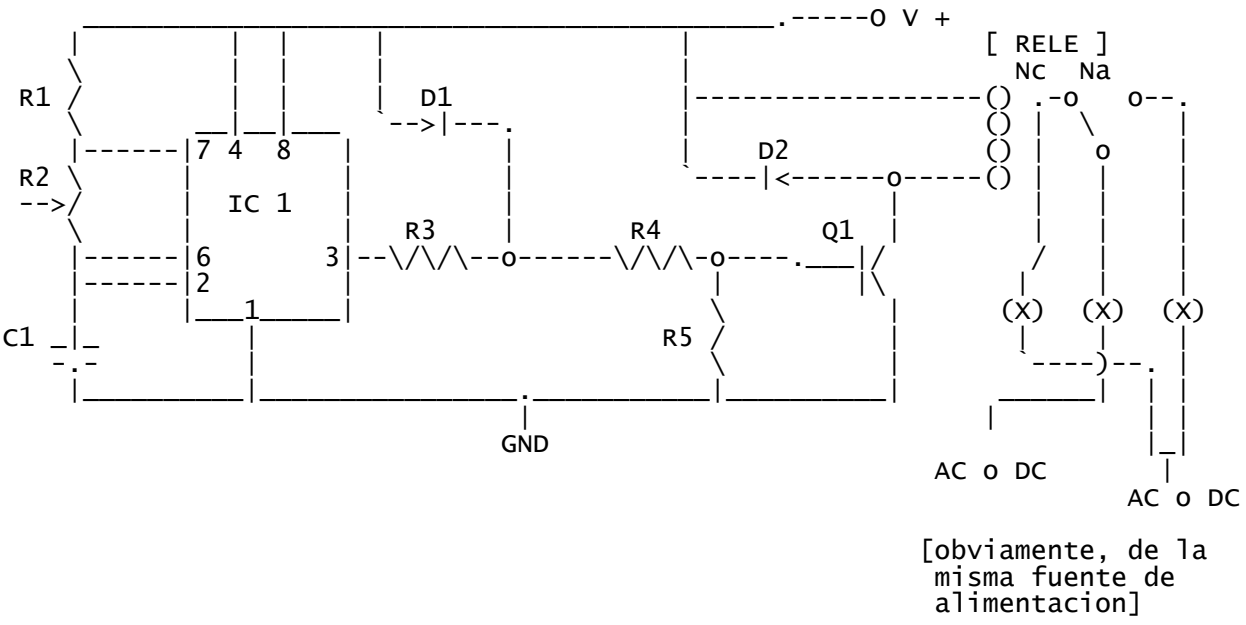
[igual, la teoria tambien es importante, porque si no la sabes,
  no entenderas como funcionan estos circuitos]

Conmutador electronico

Este circuito es similar al de la baliza electronica de la entrega anterior, pero este cuenta con el famoso 555.

Seguramente estaras diciendo 'Y que me importa eso ?'

Posiblemente no te interese, pero a mi y algun lector solitario por ahi, tambien le debe interesar.
Este circuito cuenta con muchas mejoras, como indicacion de estado alto o estado bajo, y regulacion de tiempo de conmutacion.



[obviamente, de la misma fuente de alimentacion]

R1 = 6.8 K
R2 = Potenciometro 10K
R3 = 220 Ohm
R4 = 2.7 K
R5 = 2.7 K

Q1 = 2N3904 / BC548
D1 = Diodo led del color que tu quieras
D2 = 1N4004 o mayor

C1 = 10 uF

IC1 = LM555 [tambien puede ser NE555]

(X) = Carga a conectar (lampara, motor, lo que quieras)

El rele es de 12 V.

En este caso, sientete libre para experimentar con el valor de C1, y jugando con el potenciometro.
Mientras mayor sea el valor de C1, mayor sera el tiempo entre conmutaciones.

El circuito funciona con +12 V.

```
%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%
@%@%  9  -  Despedida y agradecimientos  @%@%@%@%
%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%
```

Bueno, son las 4:03 AM [si el reloj de la bios esta bien], y me tengo que levantar como minimo, a las 9:00 AM porque tengo que ir a la escuela.
Bueh, la vida es asi, yo preferiria seguir escribiendo, pero este articulo llega hasta aqui.

Pero al menos me he divertido un rato, porque aproximadamente a las 3:00 siempre se juntan unos corredores de picadas [arrancones] en los semaforos que hay cerca de mi casa, asi que fui a verlos un rato.

Me gustaria un recibir un cierto feedback de parte de los lectores de SET, para conocer gente nueva con ganas de aprender cada dia mas, aparte de mejorar la calidad de los articulos cada dia.

Como siempre gracias a todo el staff de SET, a madfran, y a gmz.
Y por supuesto, a ti lector.

[gmz, cuando pensas escribir ?]

Nos veremos en proximas entregas.

Buena suerte,
elotro
elotro.ar@gmail.com

EOF

M A Q U I N A S D E T U R I N G

por elotro elotro.ar@gmail.com

Es mecanizable el proceso de freir un huevo? O de estacionar un auto?
Tal vez el de hacer una suma con lapiz y papel?
O buscar errores en el codigo de un programa de computadora?
Por mas que progresen las computadoras, esta probado que existen problemas que nunca lograran resolver...

Veremos que hay que tener mucho cuidado al hablar de cuando algo es mecanizable, o sea que se puede solucionar por un procedimiento mecanico efectivo. Para entender que significa esto se idearon unos dispositivos inmaginarios o abstractos que toman el rol de las computadoras modernas.

Antes que nada, conozcamos al genio detras del telon : Alan Turing

Alan Mathison Turing fue un destacado matematico y logico britanico. Trabajo en la Universidad de Princeton y formo parte del Foreing Office Britanico. Fue director adjunto del Manchester Automatic Digital Machine y estudio las capacidades de las maquinas pensantes y desarrollo metodos matematicos y fisicos para el estudio de la morfogenesis.

1. Comenzar por lo simple @

@%@%@%@%@%@%@%@%@%@%@%@

Que es una maquina de Turing? Se han ideado varios tipos de maquinas de Turing (MT), pero se ha probado que todos los tipos pueden resolver los distintos problemas, con lo cual basta que describa solo un tipo para que fijen la idea.
La idea de una MT es definir una computadora minima, que pueda hacer todo lo posible. Basicamente, tener una maquina de Turing significa tener:

a) Un vocabulario (conjunto finito de simbolos), de por lo menos 2 simbolos diferentes; dentro de este vocabulario tenemos un simbolo especial llamado 'blanco' y que represente la ausencia de otro posible simbolo.

b) Una cinta con infinitos casilleros, tantos como numeros naturales existen y numerados del 1 en adelante, junto con una cabeza lectora escritora.

c) Un conjunto finito de estados, la maquina estara en cada uno de ellos en cada instante

d) Un conjunto finito de instrucciones, o programa

e) Una configuracion inicial de la cinta, es decir, hay finitos casilleros con simbolos no blancos en la cinta y ademas una posicion inicial de la cabeza lectora-escritora.

La cabeza lectora-escritora apunta a un unico casillero de la cinta en cada instante; lee y escribe los posibles caracteres del vocabulario de la maquina y puede avanzar o retroceder una posicion cada vez que se le pida.
Es el programa (d) el que determina las acciones de la maquina.

2. De vuelta al principio @

@%@%@%@%@%@%@%@%@%@%@%@

Entonces: Que es un programa?. Un conjunto de instrucciones.
Esta concepcion de programa implica que estas instrucciones no tienen que estar ordenadas, vale decir, no forman una secuencia.
Cada instruccion dice algo asi como: Si el simbolo al que apunta la cabeza es X, y el estado actual es E, entonces escribir en ese lugar el simbolo Y, mover la cabeza una posicion (izquierda o derecha), o no moverla, y pasar al estado F.
Todos estos datos conforman la instruccion.

3. Vida, pasion y muerte de una MT @

@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@%@

Una MT 'vive' haciendo una serie de operaciones sobre una cinta infinita en ambos entidos, y pasando eventualmente de un estado a otro entre la ejecucion de estas operaciones. Parara cuando no encuentre la operacion necesaria, o bien, cuando se le diga que pare. [que pare, no que se le pareXD]

Componente a componente

""""""""""""""""""""""""

Una MT es basicamente un conjunto finito de instrucciones, cada una de ellas de la siguiente forma:

Si el estado actual es E, y el simbolo de la cabeza lectora es S, entonces haz lo siguiente:

- Imprimir en el lugar leido el simbolo T (que puede ser igual a S, o bien otro simbolo diferente)

- Moverse hacia el lado X (X puede ser izquierda o derecha, o nada, es decir que no se mueve)

- Pasar al estado U (puede ser cualquier estado, inclusive E)

Por ejemplo, esta es una posible instruccion de una MT:

Si el estado actual es 0, y el simbolo de la cabeza lectora es A, entonces:

- Imprime en el lugar leido el simbolo B
- Muevete a la derecha
- Pasa al estado 1

Haciendolo facil

""""""""""""""""

Para poder escribir las instrucciones de una forma comoda con pocos simbolos, se pueden escribir de esta manera:

0,a,b,D,1

Significa que si el estado es 0 y se lee a, escriba b, ir a la derecha y pasar al estado 1.

4. El limite de lo computable @

@%@%@%@%@%@%@%@%@%@%@%@%@%@%

Alan Turing ideo su maquina mucho tiempo antes de que existieran las primeras computadoras. En su epoca se trabajaba en forma teorica y se estaba consciente de la mayoria de los problemas de computabilidad [que palabrota] que se conocen hoy en dia.
Todo esto se sigue estudiando a nivel teorico, independientemente del progreso que se produzca con la aparicion de nuevas concepciones practicas.

[lease como los procesadores que intel saca al mercado y que los de latino america no podemos comprar por falta de $]

En realidad, el 99% de lo que se conoce como progreso en algun sentido, corresponde a mejoras y superacion de la calidad y la eficiencia, e incluso oferta de mejores facilidades, pero para realizar lo ya realizable anteriormente. Pero las computadoras siguen resolviendo los mismos problemas que a principio de siglo, solo que mas rapida y eficientemente.

[o sea que si corriendo Linux en un 386 sacaba 2+2 y me daba 4, corriendo Windows en un Pentium 4 generalmente se cascara el sistema, pero podria dar 4 tambien y en una forma visual y mas bonita [y cara]]

5. Nada nuevo bajo el sol @

@%@%@%@%@%@%@%@%@%@%@%@%

Todo lo que hacen las computadoras modernas, es decir computar o calcular [no creas que si tienes una SET en tu disco, dentro de el hay papeles con todos los articulos] , tambien lo puede hacer una MT apropiada, siempre que

se disponga del tiempo suficiente y del tamanio de cinta que se necesite.
  No quiero decir que sea conveniente calcular una raiz cuadrada en una MT [porque cuando termine ya habra salido SET 680 :)], pero es posible calcular potencias, logaritmos, ordenar, clasificar, buscar en listas. Todas las operaciones que tu, programador, estas acostumbrado a hacer.

  [no me digas que nunca has tocado un Qbasic, un gcc o un Logo por lo menos, porque no te voy a creer]

  Lo que muestra que esto es una evidencia a favor de que las maquinas de Turing representan las funciones computables, o sea que pueden calcular las cosas que deberians ser calculadas por medios mecanicos.
  Pero, Que es un procedimiento mecanico? Hasta hoy, no se esta en un acuerdo absoluto, pero se cree que se esta en el camino de la respuesta al estudiar estos procedimientos.

6. Falta de decision @
@%@%@%@%@%@%@%@%@%@%

  Se podria decir: 'Bueno, pero que problemas no son computables?' Los hay muchos, como el siguiente.

  Se sabe que en matematica no siempre es facil encontrar soluciones a las sumas algebraicas. Normalmente en la escuela se ensenia a encontrar soluciones de ecuaciones lineales, cuadraticas y otros tipos. Un matematico en su carresa aprende a resolver numerosos tipos de ecuaciones. Pero los metodos para resolver polinomios (son ecuaciones de grado arbitrario, como por ejemplo: Que valores de x satisfacen x^5 + 3 - x^3 - x^2 - 6x + 1 = 0) no son en general uniformes, en los que se deben hacer algunos pasos al tanteo junto con otros mas complejos; y en los que no se garantiza la posibilidad de encontrar la solucion entera.

  Es decir que no hay un algoritmo que resuelva el caso general. Claro que puede haber uno si la ecuacion es sencilla como la anterior.
[me parece qe no es tan sencilla porque aun estoy tratando de resolverla]

  La ecuacion puede ser resuelta usando un metodo de fueza bruta, probando todos los numeros reales hasta que se encuentre la solucion. Pero lo que estoy exponiendo aqui es que no existe un algoritmo que pueda resolver una ecuacion si en principio esta puede ser cualquiera.

  Si los algoritmos no son otra cosa que MT, esto es asi y ya debes saber que el gato no tiene 5 patas. Y como las computadoras no son otra cosa que MTs aunque mas rapidas y sofisticadas, entonces ninguna computadora puede resolver un problema de este tipo, hasta tanto no cambie la nocion de algoritmo y computable.

7. Datos insuficientes @
@%@%@%@%@%@%@%@%@%@%

    Existe lo que se denomina tesis de Church. Alonso Church fue un logico matematico de principios de siglo que se destaco, junto con otros, por sus investigaciones en la teoria de la computabilidad, es decir, que cosas son computables y cuales no lo son. En esta tesis se propone lo que se entiende formalmente como algoritmo.
    Sabemos que un algoritmo es todo lo que de alguna manera efectiva da un metodo para computar algo. Esta computacion puede hacerce de varias maneras posibles, pero basta con que pueda seguirse mecanicamente sin necesidad de esfuerzo adicional para que se llame metodo efectivo. Esto que resulta tan sencillo de expresar no es tan facil a la hora de precisar matematicamente los conceptos.
    Como puedes ir observando, al plantear un problema surgen naturalmente dudas cerca de la posibilidad de resolverlo. Por esta razon, Church propuso su tesis en la cual da formalmente distintas nociones de cuando algun procedimiento, merece ser llamado algoritmo. Ademas, prueba que todas esas nociones son equivalentes, en el sentido de que aceptando cualquiera de ellas como lo que se define a algoritmos posibles, se obtienen exactamente algoritmos identicos.
    Ni uno mas, ni uno menos. Esto sirve como evidencia de que la nocion de algoritmo como algo efectivamente computable es la que se afirma en esa tesis.

8. Volver al futuro @

@%@%@%@%@%@%@%@%@%@

La tesis de Church no puede ser demostrada. Esto no es debido a carecer de elementos matematicos o conocimiento suficiente. Se debe a que puede haber varias nociones distintas de cuando algo constituye un algoritmo, unas sin desmerecer las otras, pero no se puede saber con certeza si estas nociones dan o daran alguna vez procedimientos mecanizables. Es decir, si alguna computadora los puede llevar a cabo.

Lo que se hace en lugar de demostrar esta tesis, es dar mas evidencia a su favor. (Tambien pueden haber evidencias en contra, pero no es muy probable.

Cada vez que alguien descubre o inventa un mecanismo abstracto de computo que parezca representativo de lo que puede se un algoritmo de la manera mas general posible, se trata de probar que resulta equivalente o puede conseguir los mismo computos que aquellos realizados por Maquinas de Turing, por ejemplo.

Ademas, todas las variantes que se pueden presentar al esquema de maquinas de Turing, conducen al mismo concepto de algoritmo. Vale decir, si agregamos 2, 3, ... cintas, cabezas, mas simbolos, mas movimiento similares, etc.. no tendremos otra nocion de algoritmo. Podremos simplemente resolver los mismos problemas de antes, solo que de manera mas eficiente. [lo que pasa con linux y windows]

Con las computadoras de hoy en dia pasa exactamente esto. Todas en alguna manera son versiones sofisticadas de maquinas de Turing (paradojicamente, y en la practica, tienen memoria limitada, a diferencia de las MT.) No pueden resolver un problema que una MT no pueda.

Pero, Podra algun dia cambiar la nocion de algoritmo? Posiblemente, si las computadoras progresan lo suficiente como para permitir esto. Deberian no aumentar precisamente en velocidad, aunque esto sea tambien importante; sino tambien en poseer algun tipo de operacion primitiva que permita otros tipos de operaciones no conocidas o exploradas hasta el presente.

Quizas sea en la representacion de los datos. Quizas sea en la secuencia de los computos. Quizas nunca nadie lo sepa.

8. Variantes al esquema de MT @ %@%@%@%@%@%@%@%@%@%@%@%@%@%@%

Estas son las principales variantes a los esquemas de MT que suelen aparecer en la literatura de computacion teorica. Todas pueden usarse y combinarse, dando como consecuencia distintos estilos de MT, y por tanto, modelos diferentes de computadoras. Pero puede demostrarse formalmente que todos esos modelos son matematicamente equivalentes. Esto es, no hay ninguna funcion matematica que sea computable usando un posible modelo de maquina y que no lo sea por otro modelo.

Por ejemplo, si una funcion puede computarse usando 2 cintas, entonces tambien podra ser resuelta usando una sola cinta [mira mas abajo]. Si una funcion no puede computarse sin una instruccion especial de parada (entre otras), entonces tampoco podra computarse si se agrega esta instruccion de parada, no inporta en que instruccion se coloque.

Respecto de lo ilimitado de la cinta : puede ser ilimitada a izquierda o derecha, o en un solo sentido.

Numero de cintas : puede haber 1, 2 o mas cintas (incluso de distintos tipos), con la condicion de que una de ellas sea infinita.

Alfabeto: Debe ser un conjunto finito de simbolos, de uno o mas simbolos. Si hay uno solo, debe existir el 'blanco' o ausencia de simbolo. En caso contrario, todos los casilleros de la cinta tendran el mismo contenido y no habra computo alguno.

Conjunto de estados: Debe ser finito.

Conjunto de instrucciones : Debe tambien ser finito.

Instruccion de parada : Puede haber o no haber. Si no la hay, entonces se supone que cada vez que no hay una instruccion que contemple el estado de la maquina y el simbolo actual, esta debe parar.

Tipos de acciones : Las accines pueden ser del tipo escritura de un simbolo y movimiento hacia un lado, o bien contar separadamente con dos tipos de instrucciones, de escritura de un simbolo y otra de movimiento hacia un lado.

8. El huevo o la gallina @
%@%@%@%@%@%@%@%@%@%@%

Otro problema clasico probablemente indeducible [que palabrotas :)] es es de la parada de un MT. Que significa esto?
Es considerar la construccion de un algoritmo que reciba como entrada una MT incluyendo los datos de la cinta y la posicion inicial de la cabeza y tal que el decida si esa maquina parara o ciclara indefinidamente.
No es posible construir ese algoritmo con ningun lenguaje conocido, ni jamas podra hacerse, hasta tanto no cambie la nocion de algoritmo, tal como plantee antes.
Esto es equivalente a decir que no existe una MT que pueda recibir como entrada a otra MT arbitraria y decir SI o NO segun esta ultima pare o no pare.
Por supuesto, existen variantes a estos problemas, y tantos otros, que tambien son indecidibles.

[un ejemplo de problema indecidible es cual de todos los windows es el que peor funciona. Me contaron que el XP el que va ganando..]

8. Mira a donde fuimos a parar @
%@%@%@%@%@%@%@%@%@%@%@%@%

Se duda bastante que alguna vez exista una computadora con operaciones escencialmente tan diferentes que permita resolver problemas que hasta hoy se consideran no computables. [si leiste bien veras que no es imposible, sino que es demasiado complejo para la tecnologia de hoy en dia]
A veces se especula con su existencia puramente teorica. Por ejemplo, supongamos que podemos resolver el problema de la parada de una MT con una simple consulta a un cierto algoritmo. Esto se conoce como el uso de oraculos.
Entonces, tiene sentido estudiar que otros posibles problemas hasta ahora indecidibles podran ser decididos de esta manera.
No voy a tocar este tema ahora [porque me esta dando suenio], pero sabremos que aun asi habra otros problemas indecidibles. O sea, que ni agregando ciertos oraculos se podran resolver todos los problemas. [como el problema de que estoy soltero]

8. Y entonces que ? @
%@%@%@%@%@%@%@%@%@%

Luego de la decepcion que representa enterarse de algo asi como que no todo es algoritmizable [alguien aviseme si esa palabra existe], se abren nuevas puertas a la computacion teorica. A saber, interesa cada vez que alguien propone un problema bien planteado, el hecho de que sea solucionable computacionalmente [esta palabra tambien] o no.
Luego, si se descubre que es computable, entonces no solo queda pendiente encontrar algoritmos que lo resuelva, sino que hay que encontrar algoritmos que lo resuelvan de la manera mas eficiente posible. De esto se ocupa el area conocida como Complejidad Computacional.
Si en cambio, se descubre que no es computable, entonces aun quedan cosas por hacer. Los teoricos trabajan a veces viendo la forma de pasar a resolver por lo menos el problema de forma parcial.
Por ejemplo, asumir ciertas hipotesis, restringiendo el problema original y tratar de esa manera de encontrar algoritmos que resuelvan esos casos.
Tambien interesa ver un poco por donde se hallan los limites de lo computable a lo no compuable.

8. Moraleja @
%@%@%@%@%@%

La moraleja de este texto no es solo que descubras otros metodos de computacion, sino tambien que aprendas lo siguiente:
Ni el Pentium, ni tarjetas perforadas, ni Maquinas de Turing hacen la felicidad. Pero hay maneras y maneras de vivir feliz...y computar.

9. Ojo ! Todavia no termina @
%@%@%@%@%@%@%@%@%@%@%@%@%

Se que estas aburrido leyendo esto, pero robo otros minutos de tu tiempo para que veas este simulador de MT hecho en Quick Basic. [perdon por que sea en este lenguaje, pero hace poco que comence a aprender C y C++, y no tenia mucha confianza como para pasarlo de lenguaje]

```
<++> turing.bas
' Simulador de Maquinas de Turing
' Derechos reservados a Ariel Arbiser (c) 1994
' Extraido de la revista Byte Argentina, edicion febrero de 1994

' Si el autor original tiene algun problema con la reproduccion de este codigo
' en algun medio fisico o electronico, por favor comunicarse a los mails
' indicados arriba.

' Recomendado compilar con Quick Basic. Su compilacion en Fbasic u otro
' similar puede necesitar alguna modificacion de sintaxis en las declaraciones
' de subrutinas y matrices

' Comentarios son precedidos por comillas sencillas -> '


' COMIENZO DEL PROGRAMA
' declaraciones de subrutinas

DECLARE SUB Parar (m$)
DECLARE SUB Tecla ()
DECLARE SUB MuestraCinta (cinta$(), posic%)

DEFINT A-Z

' numero maximo de instrucciones y tamanio de la cinta

CONST true = 1, false = 0
CONST maxni = 50, maxcinta = 200

' declaracion de matrices

DIM insest(maxni), inscar$(maxni), insnuevo$(maxni), insmov$(maxni),
inspasa(maxni)   'esto va en la linea de arriba pero lo puse aca porque se
                 'pasa de las 78 columnas

DIM cinta$(maxcinta)


' inicializa la pantalla

SCREEN 0: WIDTH 80: CLS


' lee linea de comando

i = INSTR(COMMAND$, " ")
IF i = 0 THEN
LINE INPUT "Archivo de programa : "; f$
ELSE
f$ = MID$(COMMAND$, i)
END IF
f$ = LTRIM$(f$)


' abre archivos con datos de la maquina

OPEN f$ FOR INPUT AS 1


' lee cinta

INPUT #1, tamcinta

FOR c = 1 TO tamcinta
  INPUT #1, cinta$(c)
NEXT


' pone a cero el numero de instrucciones

ni = 0

' lee "programa"
```

```
' numero instrucciones

INPUT #1, ni


' instruciones

FOR r = 1 TO ni
 INPUT #1, insest(r), inscar$(r), insnuevo$(r), insmov$(r), inspasa(r)
NEXT


' ejecuta
' setea el numero de instrucciones a cero
niejec = 0

' setea la posicion de la cabeza a 1
posic = 1

' inicializa estado de la cinta
estado = 0

LOCATE 3: COLOR 11: PRINT STRING$(80, "=")


' ciclo principal
DO

' se paso ?
IF posic > maxcinta OR posic < 0 THEN Parar "Error: posicion fuera de la cinta"


' exhibe cinta
MuestraCinta cinta$(), posic


' ejecuta instruccion
car$ = cinta$(posic)

' busca simbolo en instrucciones
i = 0
enc = false
DO WHILE i < ni AND enc = false
    i = i + 1
    IF inscar$(i) = car$ AND insert(i) = estado THEN enc = true
LOOP


VIEW PRINT 4 TO 25
LOCATE 23
COLOR 13

PRINT "Estado: "; estado, "caracter: "; car$

' si no se encontro simbolo
IF enc = false THEN Parar "Maquina paro por ausencia de instruccion"

' si no
COLOR 12: PRINT "Ejecutando instruccion #"; i; "escribe "; insnuevo$(1); ,
" accion: "; insmov$(i); ", pasa al estado "; inspasa(i)

'     ^       ^       ^       ^       ^       ^
'     |       |       |       |       |       |
'eso va en la linea de arriba porque se pasa de 80 columnas


' actualiza estado
estado = inspasa(i)

' actualiza cinta
cinta$(posic) = insnuevo$(i)


' actualiza cabeza
```

```
SELECT CASE LCASE$(insmov$(i))
  CASE "i": posic = posic - 1
  CASE "d": posic = posic + 1
  CASE "s": Parar "Instruccion de parada"
  CASE ELSE
END SELECT

' incrementa el numero de ejecuciones

niejec = niejec + 1
COLOR 10: PRINT "Nro instrucciones ejecutadas : "; niejec
PRINT

' espera por una tecla
Tecla


' cicla indefinidamente o hasta detenerse
LOOP

SUB MuestraCinta (cinta$(), posic)

CONST anchocinta = 20
STATIC posicini

COLOR 12

VIEW PRINT 1 TO 3
LOCATE 1, 1: PRINT SPACE$(160)
LOCATE 1, 1

' Cinta visible
IF posic < posicini OR posic > posicini + anchocinta - 2 THEN posicni = posic
LOCATE 1: PRINT " ";

FOR i = posicini TO posicini + anchocinta
  PRINT "|";
  IF i > 0 AND i <= maxcinta THEN
    COLOR 11: PRINT cinta$(i)
     ELSE
      COLOR 7: PRINT ".";
   END IF
  COLOR 12: PRINT " ";
NEXT

PRINT "|";


' dibujar cabeza

LOCATE 2, 1

t = 4 * (posic - posicini) + 4

' dibujarla
COLOR 11: PRINT SPACE$(t); "^"; " ";

END SUB

SUB Parar (m$)

COLOR 15: PRINT m$
Tecla
SYSTEM

END SUB
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
SUB Tecla
 LOCATE 23, 21
 PRINT "Pulse una tecla"
 a$ = INPUT$(1)
 LOCATE 23, 21: PRINT SPACE$(15)

END SUB
```

<-->

Este es el modo en que opera el programa: Abre un archivo de texto ascii cuyo nombre lo ingresa el usuario, e interpreta cada una de las lineas de la manera siguiente.

La primera linea debe decir cuantos casilleros de la cinta se definen inicialmente segun su contenido. A continuacion se dan los caracteres escritos uno por uno, en orden, desde aquel en que la cabeza de la maquina comienza a leer. La siguiente linea dice el numero de instrucciones que posee la maquina.

Por ultimo, vienen las lineas de las instrucciones que posee la maquina. (fijate en Vida, Pasion y muerte de una MT para ver su formato), tantas como se indico en la linea anterior. El formato del archivo, es este:

```
[n: numero de caracteres iniciales en la cinta]
[caracter 1]
[caracter 2]
   ......
   ......
[caracter n]
[m: numero de instrucciones de la maquina]
[instruccion 1]
[instruccion 2]
   ......
   ......
[instruccion m]
```

El programa luego simula el funcionamiento de la maquina. Se detiene cuando no se encuentra contemplando entre las instruciones alguna combinacion de estado, simbolo leido, o bien cuando se ejecuta la instruccion parar.

Las acciones son :

```
i: mover a izquierda
d: mover a derecha
s: parar
cualquier otro simbolo : no moverse
```

Al correr el programa, se pregunta automaticamente el nombre del archivo que constituye el "programa" de la maquina. Supongamos que sea MT.TXT
Si se invoca desde DOS al programa ejecutable (despues que lo compilaste con Quick Basic), y asumiendo que se lo llamo TURING.EXE, entonces se puede pedir directamente el nombre del archivo segun la sintaxis :

```
TURING MT.TXT
```

Si no dispones de Quick Basic o no lo puedes conseguir en internet, (debe andar por algun lado..), visita www.qbasic.com, alli hay una seccion de compiladores Basic. Recomiendo el Fbasic, aunque tal vez tengas que cambiar algunas lineas del codigo.

10. Ojo ! Todavia no termina (parte dos) @

%@%

Incluyo unos ejemplos de MT para que crees el archivo [no creeras que voy a hacer todo por ti], y luego los pruebes con el simulador.
Todos los ejemplos estan bajo derechos reservados de Ariel Arbiser.

Gato x liebre

"""""""""""""""

Esta MT transforma una sucesion de letras "a" (aaaaa..), en una sucesion de letras "a" y letras "b" alternadas (abab..). Esta escrito segun la codificacion de "Vida, pasion y muerte de una MT"

```
0,a,b,D,1
0,b,a,D,1
1,a,a,D,0
1,b,b,D,0
```

Desde luego, se considera a 0 como el estado inicial de la maquina, en

este y todos los demas casos. Esta MT parara cuando encuentre un simbolo que no sea ni A ni B, puesto que no hay ninguna instruccion que contemple ese caso.
    Este problema de cambiar las letras que ocupan una posicion impar de la cinta, demanda 4 instrucciones. Puede haber problemas que demanden un numero menor o mayor de instrucciones.

Bloque mayoritario

Otro ejemplo puede ser este. Se desea buscar en la cinta la primera ocurrencia de un bloque de letras a (aaaa..) y cambiarlo a todo un bloque de letras b (bbbb..). Para resolverlo se pueden usar las siguientes instrucciones de MT. Se asume que los unicos simbolos que pueden aparecer en la cinta son a, b y c.

```
0,a,b,D,1
0,b,b,D,0
0,c,c,D,0
1,b,b,D,1
1,a,a,-,2
1,c,c,-,2
```

Esta maquina parara cuando encuentre un simbolo diferente de a,b; o bien cuando haya convertido al primer bloqe de letras a (aaaa..) en letras b (bbbb..).

Facil como 1+1 [cuanto sera 1+1 en windows xp ? :)]

Por ultimo, esta es una MT para incrementar en uno a un numero binario, esto es, en simbolos que representan a un numero en la cinta escrito en base 2. Los simbolos seran "a" representando al 0 y "b" representando al 1. Se puede usar cualquier otra representacion similar.
    Aclaro que el numero se escribe de izquierda a derecha como lo escribes normalmente. Finalmente, hay un simbolo especial, "x", que denota el blanco o delimitador. Hay simbolos "x" a ambos lados del numero, de modo que la MT sabra cuando empieza y termina el numero.

```
0,x,x,D,0   1)
0,a,a,D,0   2)
0,b,b,D,0   3)
0,x,x,I,1   4)
1,a,b,-,2   5)
1,b,a,I,1   6)
1,x,a,-,2   7)
```

La idea del funcionamiento de esta MT es el siguiente:
    Primero (1) la maquina saltea todos los "x" que pueda, moviendo la cabeza hacia la derecha, para encontrar el primer digito (el mas significativo) del numero.
    Luego (2 y 3), la maquina busca el ultimo digito del numero (el menos significativo), para comenzar alli la operacion de incrementar el numero.
    Con la instruccion 4, se coloca en el ultimo digito. Al llegar al estado 1 (5 y 6), la maquina comienza la operacion, de derecha a izquierda.
    La instruccion 6 corresponde a la idea intuitiva del acarreo [me llevo 1] que se va ejecutando hasta tanto este no ocurra mas. Se llega a la instruccion 7 si se produce un acarreo tal que el numero de digitos del resultado sera mayor que el numero original (sucede todas las cifras del numeros son unos), reemplazando de esta manera la "x" izquiera con un 1 adicional.
    Al llegar al estado 2, la maquina termina pues no hay ninguna instruccion que opere en ese estado.

11. Me voy a dormir @
@%@%@%@%@%@%@%@%@

Termino este largo [tal vez aburrido] articulo sobre las maquinas de Turing. Nos volveremos a ver ...
[parece sacado de una peli de terror :)]

Criticas, sugerencias, opiniones, dinero o lo que quieras enviar :
elotro.ar@gmail.com

EOF

Si, sabemos es 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.

-[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! ¿nadie quiere meterse/defender a las SGAE?

- 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?

- ORACLE, MySQL, MsSQL... ¿Alguien levanta el dedo?

- Mariconeos con LOTUS, nos encanta jugar con software para empresas, un gran olvidado del hacking "a lo bestia".

- 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. Os apercibis de las consecuencias ?

- 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 viendo 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, PORQUE CORRE SERIO RIESGO DE SER ELIMINADO.

- 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]----------

http://www.zine-store.com.ar - Argentina
http://qaldune.freeownhost.com - USA
http://www.hackemate.com.ar/ezines/set/ - Argentina (no muy acutalizado que digamos)

El resto que nos aviso de tener un mirror, o no lo encontramos o las paginas estaban desactivadas, ¡mala suerte!.

-[En nuestro proximo numero]----------

Antes de que colapseis el buzon de correo preguntando cuando saldra SET 34 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>

EOF

-[0x0B]--
-[Economia Fractal]--
-[by SET]---SET-33--

ECONOMIA FRACTAL

Aunque muchos se nieguen a reconocerlo deberiamos ser conscientes que el mundo se mueve gracias al dinero. Por mucho que nos duela no son las ONG las que conseguiran que la economia mundial se transforme de tal forma que nadie padezca hambre y miseria por falta de recursos materiales. Si tanta gente esta de acuerdo con esta afirmacion, se podria pensar que en un mundo saturado de capacidad de calculo y repleto de gente con ansias de hacer fortuna, se ha encontrado el metodo para conocer con precision el movimiento aparentemente caotico de cualquier grafico que intente seguir la evolucion de una variable economica. Sea el precio de una accion, de un metal, como de un bien agrario, podemos pensar que existen todopoderosos personajes que manejan las finanzas mundiales a su antojo, bien pertrechados con potentes ordenadores y con deleznables cientificos a sus ordenes. Estos personajes, segun los amantes de las teorias de las manos ocultas, consiguen comprar algo cuando esta barato y vender cuando su precio sube. Los bancos se han enganchado a esta ola de pensamiento y venden todo tipo de informacion basada en graficos de evolucion, plagado con cierto numero de terminologia pseudo estadistica. Como ya hemos dicho en otras ocasiones, ni este ni otros articulos escritos por nosotros van a haceros mas ricos, pero tal vez evite que os metais en inversiones de gran seguridad y totalmente adaptados a vuestro perfil de riesgo segun vuestros consejeros financieros, solo para verse convertidos en pobres como ratas tras perder todo vuestro dinero, que sea poco o mucho, es solo vuestro.

VISITA A UN PARAISO FINANCIERO

El edificio se alzaba gracil en el aire envuelto en una atmosfera asfixiante. Desafiando las leyes de la termodinamica, el color oscuro es el que destacaba en su fachada ondulada que absorbia su buen vatio por metro cuadrado procedente de un ardiente e implacable sol. Sin embargo la temperatura en el interior del edificio era muy agradable y permitia trabajar en cualquiera de sus salas de reunion con la chaqueta puesta y la corbata enfundada. Nuestro amigo lo habia comprobado el dia anterior. Mas dificil de comprobar era la temperatura en el espacio exterior. Todo se organizaba dentro de los suntuosos salones o en salas de decoracion fascinante y tampoco se promovian las visitas fuera de los muros acristalados. Nuestro espectador y viajero tuvo que casi escaparse despues del desayuno para poder comprobar que en los lujosos jardines que rodeaban el edificio de vidrio y cemento no habia casi nadie. Las hermosas piscinas, habian varias, estaban casi desiertas y entre la vegetacion tan solo se encontraban los obreros perfectamente uniformados y sin ninguna duda extranjeros en aquellas tierras y con visas muy bien controladas. Y dentro continuaba el despilfarro y lujo romano. Si hacia demasiado calor para utilizar las piscinas exteriores las interiores, convenientemente climatizadas, cumplian perfectamente su funcion. Si no apetecia pasear por los jardines exteriores bajo una atmosfera sofocante, todo una superficie ajardinada y perfectamente refrigerada, con pequenyos riachuelos incluidos, permitia relajarse bajo una boveda de vidrio y acero.

Uno se pregunta de donde viene semejante poderio economico para permitirse construir y mantener un tal edificio sobre las arenas del desierto. El asombro aumenta cuando mirando alrededor y se ve que toda una ciudad ha sido levantada en los alrededores bajo las mismas premisas de total desprecio al ahorro energetico y de toda logica economica. Si; todo esto es posible solo es gracias a la capacidad de generar dinero sobre una particular zona del planeta y lo mas asombroso es que esta generacion de dinero no proviene de ninguna actividad que produzca objetos tangibles. Son solos transacciones comerciales y servicios.
Nadie hubiera podido adivinar que esto ocurriera tan solo diez anyos atras. Los amantes de lo esoterico, diran que sin duda hay en el mundo fuerzas ocultas que dirigen los destinos del resto de los mortales y que todo habia sido planificado convenientemente hacia bastante tiempo. Nosotros no somos de esta opinion simplemente porque conocemos demasiado bien la voracidad de la naturaleza humana. Si hubiera alguien capaz de planificar la economia mundial no os quepa la menor duda que esta ya lo hubiera sido y que ahora seriamos verdaderos esclavos en manos de unos pocos y bien conocidos personajes que dominarian el planeta.

Serian bien conocidos ya que nadie es capaz de disfrutar de semejante poder y no hacer ostentacion del mismo. Los hechos nos muestran que los poderosos cambian con extraordinaria rapidez, de lo cual deducimos que no son mas que recien llegados, demasiado ocupados en mantener su status y sus privilegios

como para hacer gala de sus conocimientos. No, no son los resultados del conocimiento humano que han conseguido hacer crecer a velocidad de vertigo una ciudad en un desierto, en nuestra opinion ha sido simplemente la casualidad, aunque algunos la llaman fractabilidad. Hablaremos de esto.

LO QUE INTENTAN VENDERNOS LOS AGENTES FINANCIEROS

Cualquier publicacion financiera, cualquier web bancaria, incluso consejos y comentarios amistosos, dados en despachos de los directores de agencias bancarias. Todos os hablaran de riesgo. Las publicaciones financieras os daran preciosos graficos de la evolucion de los precios de las acciones y os daran lineas con previsiones de futuro. Las web financieras suministran un poco mas de informacion y colores, sobretodo colores. Dentro de la inundacion cromatica os mostraran graficos on-line y os ensenyaran lineas hablando de variabilidad y riesgo. En los despachos confortablemente acondicionados os diran confidencialmente que la ultima informacion reservada solo para clientes especiales consiste en una cartera especialmente disenyada para perfiles de bajo riesgo. El comentario puede variar en funcion de la percepcion que tenga el director de tu cara y puede que decida que eres un tipo que le gustan los consumos abundantes de adrenalina cuando lo cierto es que te asusta el hecho de comprar un billete de loteria.

Si somos un poco mas inquisitivos y se nos ocurre preguntar como se mide esto de la variabilidad y el riesgo, el director del banco saldra como pueda del atolladero, normalmente su nivel intelectual no pasa del resultado del ultimo partido de futbol. Si buscamos en Internet, encontramos muchas mas respuestas. Demasiadas respuestas. A veces la abundancia no es todo lo buena que parece. En vuestro honor vamos a intentar haceros un pequenyo resumen de lo que los economistas oficiales piensan y los fundamentos sobre los que se basan las acciones de los gerentes de nuestros dineros. No creais, tampoco es tan complicado y ¡cabe en este articulo !

Primera, y falsa, hipotesis.

La gente tiene un comportamiento racional. Como su unica meta es el enriquecimiento, buscaran toda la informacion relevante sin desestimar nada que pueda afectar a sus inversiones. No pagaran nunca mas de lo que pretendan obtener por los dividendos invertidos. Todo ello hara que el mercado se comporte de una forma lucida y razonable.

Como todo el mundo sabe, menos los economistas, de todo lo dicho lo unico cierto es que el principal objetivo de la gente es el ganar dinero. La forma en que se percibe el riesgo es muy subjetivo y las masas tienen un comportamiento gregario que a veces les incitan a tirarse en grupo al abismo o a invertir en un valor que carece totalmente de activos.

Segunda, y tambien falsa, hipotesis.

Todos tenemos un comportamiento parecido ya que al desear solo el dinero, tendemos a hacer nuestras inversiones bajo el mismo horizonte temporal. Miden sus beneficios bajo la misma optica. Nadie es capaz de manipular el precio de las cosas y tan solo sufrimos el movimiento de los valores.

Aunque no lo creais, todos los graficos que os pueden vender los agentes financieros se basan en esta hipotesis. Ya no hablemos de los casos en que es evidente que existe manipulacion de precios, pero es que en el caso de que no existieran los carteles, algunos estudios realizados por economistas que no son bienvenidos en los estamentos oficiales, muestran que la existencia de grupos que actuan por un lado de una "forma racional" y de otro formado por "impulsivos", implica una formacion de precios con todo tipo de anomalias.

Tercera, y nuevamente falsa, hipotesis.

La formacion de los precios es practicamente continua. Esta afirmacion parece de las mas evidentes, pero sin embargo basta para observar un grafico de precios de una accion, para darse cuenta que el ultimo precio de la tarde no es igual al primer precio de la manyana. Que ha podido ocurrir en la cabeza de las inversores durante la noche para decidir que tipo de diferencia debe ocurrir entre ambos precios ? Sin embargo, de nuevo toda la economia moderna se basa en las ideas de un tal Markowitz (http://www.nobel.se/economics/laureates/), que afirmaba que todas las decisiones en materia de inversion se podian reducir a dos numeros, la media y la variancia de los precios esperados.

Cuarta, tambien falsa y ultima, hipotesis.

Los cambios de precios sigue un rumbo aleatorio. Al movimiento aleatorio tambien se le llama browniano y viene prestado de la fisica. Sirve para describir el movimiento de las particulas de gas dentro de un recipiente a temperatura homogenea. A uno de los padres de la economia moderna, Bachelier (http://sjepg.univfcompte.fr/la_recherche/libre/bachelier/) se le ocurrio que el mismo principio podria aplicarse a la formacion de los precios. Las asunciones de esta hipotesis son fundamentales, y falsas. La primera de ellas es la independencia estadistica. O sea que un solo individuo no puede influir sobre la masa global y que el precio de hoy no tiene nada que ver con el de manyana. La segunda implica que el proceso es siempre el mismo. O sea que la formacion del precio del algodon en el siglo pasado sigue el mismo proceso que la formacion de precios de durante el 1 de octubre de 1987, dia que por si alguno no lo sabe, el Dow Jones cayo alrededor del 29%. La tercera asuncion falsa es que los precios siguen la evolucion de una campana de Gauss, lo que implica que los cambios pequenyos son la norma y los grandes solo ocurren cada millon de anyos.

LO QUE REALMENTE OCURRE EN NUESTRO ENTORNO

Si la verdad no se encuentra tan facilmente como intenta decirnos la franca sonrisa del empleado bancario en el confortable despacho, ¨Donde podemos encontrarla? La respuesta es que afortunadamente en ninguna parte, o mas bien, como afirma las antiguas corrientes filosoficas, se encuentra alrededor nuestro y la podemos admirar en cualquier detalle de la naturaleza. Antes de entrar en detalles, despidamonos amablemente del sonriente empleado, salgamos a la calle y reflexionemos sobre lo que realmente ocurre a nuestro alrededor.

Primera observacion. Los mercados son arriesgados. Parece que todo el mundo que tiene cuatro cuartos en el banco es consciente de ello, pero toda la informacion que recibimos cuando deseamos invertir en algo distinto a los bonos del estado, parece querer tranquilizarnos y afirmar que el riesgo esta bajo control. Nada mas lejos de la realidad. Las idas y venidas cual olas incontroladas son la norma en los mercados financieros y no la excepcion. Los movimientos de los precios no siguen una apacible y controlada campana de Gauss y las excepciones y movimientos erraticos es lo mas normal dentro de cualquier diagrama que pretenda seguir la evolucion del precio de un bien.

Segunda observacion. Las evoluciones de los mercados siguen pautas violentas. Podemos estar aburriendonos durante tres meses sin ver ninguna modificacion apreciable en el precio de la accion que desearíamos comprar y basta que descuidemos la labor durante una semana por vacaciones, para que el precio baje bruscamente por debajo del limite que nos habiamos fijado para retomar el vuelo y quedar de nuevo fuera de nuestro alcance. En este caso seria tan solo un caso de perdida de beneficio, pero lo peor puede ser si tras dos anyos de afanosos esfuerzos para constituir una cartera a prueba de cracks, en nuestra opinion y la de nuestra banquero, miramos el calendario y nos encontramos que estamos en el 1 de octubre del 1987. Como todos debieran recordar, el dia que muchas fortunas desaparecieron y grandes cantidades de ahorros se volatizaron.

Tercera observacion. No es lo mismo seguir el precio del oro, que el metro cuadrado edificado en Estepona. Los dos bienes tienen pautas diferentes pero lo mas curioso es que estas pautas son persistentes durante el transcurso de los anyos. Es justo lo contrario de lo que dicen muchos grandes economistas que intentar explicar el fallo de sus modelos en las injerencias externas a los mercados como guerras, catastrofes naturales, etc. Pero parece que no es asi. Los grandes cataclismos provocan alteraciones puntuales, pero si hay una razon para que la gente le guste vivir en el centro de Madrid, su metro cuadrado seguira subiendo aunque se haga estallar cientos de bombas en el. Solo si este "gusto" colectivo desaparece, empezara a bajar el precio, pero es poco probable que esto no ocurra en un ciclo corto.

Cuarta observacion. Los seguidores en las busquedas de tendencia de los graficos deberian dedicarse a cosas mas productivas. No conocemos a ninguno de ellos que se halla hecho millonario, salvo los que venden libros sobre el tema, que puede que no consigan grandes beneficios pero sin duda lograr algun dinero sin necesidad de grandes esfuerzos. Los mercados enganyan y no siguen pauta alguna a corto plazo, o incluso peor, en cuanto se detecta una cierta pauta esta puede cambiar totalmente.

Quinta y ultima observacion. El tiempo es relativo. Y esta enlaza con la segunda. Si nos muestran el grafico de no importa cual evolucion de precios y eliminan la escala de tiempos, es decir, no nos dicen si los datos se refieren a un dia, una semana o diez anyos, somos incapaces de distinguir dicha escala.

Los movimientos relativos se pueden producir a cualquier escala de tiempo. Y esta afirmacion es importante. Implica que las reglas de juego se encuentran escondidas en cualquier nivel temporal, pero tambien que este nivel temporal cuante muchisimo si queremos entrar y salir dentro de cualquier mercado importante. No se puede pretender realizar una buena inversion si resulta que pueden ocurrir vaivenes en sus precios inferiores a un dia y nosotros no tengamos capacidad o tiempo disponible para realizar un seguimiento diario.

LO QUE ALGUNOS ECONOMISTAS ICONOCLASTAS PIENSAN

De hecho nos estamos refiriendo sobre todo a gentes como Mandelbrot (http://www.math.yale.edu/mandelbrot/). Dicho personaje es mas conocido por el descubrimiento de las formulas matematicas capaces de generar imagenes que conservan su estructura por mucho que intentemos aumentar el detalle, o se a modificar la escala. Son las figuras que se han dado en llamar fractales. Puede que alguno no tenga noticia de este tipo de conjuntos pero se puede hacer una idea observando un arbol frondoso o el recorrido de una costa. O sea son estructuras que la naturaleza crea de forma natural. No debiera pues sorprendernos que sus formulas hayan sido utilizadas para generar imagenes virtuales que simulen paisajes inexistentes. Como muchos visionarios su aportacion es sencilla. Si hay una formula matematica capaz de generar fractales y dichas estructuras se encuentran de forma habitual en la naturaleza, puede que exista una relacion con el comportamiento economico que no es nada mas que el efecto de la accion de muchos animales juntos llamados seres humanos.

Pero es que hay un poco mas. Los seres humanos no estamos solos en la superficie de este planeta. Todo el se encuentra inmerso en una dinamica que sigue reglas que no peden explicarse de forma satisfactoria con las reglas estadisticas que usamos normalmente. En el siglo diecinueve se empezaron a construir las primeras grandes presas de agua que hoy conocemos. En el pasado el volumen de agua que una presa debia contener era elegido en funcion de la memoria historica de los viejos del lugar o simplemente no se conocian sobre que parametros se habian basado los arabes para calcular la altura de los pantanos que nos habian legado. Habia que empezar desde cero y construir un modelo matematico que ayudara en este calculo. El problema no es banal ya que si se disenya incorrectamente el volumen necesario, puede que en la proxima sequia el pantano se vacie y todos los que vivan de el se arruinen o mueran de hambre.

Teoricamente el problema es facil de abordar. Solo es necesario tener registros lo mas lejanos en el tiempo de los caudales y a partir de ellos calcular la media la desviacion standard y ya podemos saber cual es el volumen necesario.
Un tal Hurst, oscuro funcionario ingles, encargado de calcular la altura necesario para la primera presa en Egipto, descubrio que el resultado aplicado a la historia conocida, podia terminar con el pantano seco en algun momento veinte anyos en el futuro. Y ello era debido a dos factores. Las valores de sequedad anormales eran mas brutales de lo predicho por un modelo estadistico clasico y ello se juntaba con que los ciclos de sequedad tendian a unirse. Un anyo seco tendia a ser solo el principio de varios anyos secos.

Esta especie de malaventura en rachas la podemos encontrar en muchas facetas de la naturaleza que nos rodea. Las malas cosechas tienden a agruparse, los ataques de insectos conocidos no aparecen y desaparecen sino que perduran e incluso las poblaciones de algunas aves siguen pautas agrupadas en el tiempo. No estamos jugando en solitario en este planeta, sino que estamos inmersos en una inercia de ciclos y probablemente, dentro de nuestros cerebros se encuentran grabadas en nuestras celulas grises pautas que nos obligan actuar o a sobreactuar siguiendo los ciclos que nuestro cerebro ha generado.

Mandelbrot solo es un visionario capaz de hacernos ver que estamos jugando encima de una tarima podrida que se puede derrumbar en cualquier momento. De hecho esto ocurre con una frecuencia asombrosa, pero nadie parece apercibirse que las matematicas sobre las que se basan todos los analisis de riesgo, volatibilidad, valoracion de activos y demas historias financieras, fallan con demasiada frecuencia como para ser fiables y sin embargo continuan a escribirse manuales y a construir carteras financieras en base a modelos matematicos que no soportan la mas minima comparacion con la realidad. Es mas o menos como si construyeramos nuestras casas con formulas matematicas que se demostraran erroneos en un diez por ciento de los casos. Os aseguro que nadie habitaria en el interior de tales edificios pero continuamos a creernos la jerga financiera a pesar de ser incierta y ademas incomprensible.

Sobre la base de sus observaciones y formulaciones otros han seguido sus pasos siendo como los primeros del cuento del emperador desnudo, que se han atrevido a decir en voz alta lo que muchos pensaban y aplicaban en su estrategias privadas. "Aqui hay algo que va mal y lo mejor es cambiar de estrategia". No son muchos, son ampliamente criticados, pero se han hecho un cierto hueco en el mercado de las ideas nuevas. Empecemos por www.oanda.com, parece una de entre tantas otras webs que ofrecen informacion sobre el cambio de moneda. En el fondo es algo mas ya que no solo ofrece los tipicos servicios de cambio on-line sino que suministran datos sobre el comportamiento de los jugadores en el mundo del cambio de divisas. La informacion que dan permite saber en general y a la vez particular de que hacen los clientes con sus posiciones. Cuando abren, cuando cierren, como mantienen las posiciones y en la medida de lo posible porque. En la web complementaria de www.olsen.ch dan tambien una serie de explicaciones tecnicas de como se establecen los diversos algoritmos matematicos. Otros tambien se atreven con las inversiones no monetarias y en www.panagora.com se encuentra una companyia inversora donde ademas se explica como se utilizan las tecnicas fractales para analizar el futura de las inversiones.

En nuestra opinion hay todavia mucho que estudiar sobre las tecnologias fractales aplicadas a la economia y como en toda nueva tecnica hay ademas mucho oportunista, pero vale la pena seguir su evolucion y tener mas de una carta a jugar.

REFLEXIONANDO

Un tema nos ha llevado a otro, pero lo cierto es que nadie sabe porque se generan los ciclos economicas y ni siquiera si estos ciclos existen. Segun Mandelbrot los ciclos no existen, tan solo hay curvas que siguen unas leyes que no son en absoluto estadisticas aleatorias que puedan calcularse y permitirnos medir los extremos con riesgo. Siguen unas pautas dificiles de descubrir y que parecen tener memoria del pasado, no impidiendo esta memoria que de repente se produzcan espacios de inflacion que generen movimientos de la nada. No existe hoy en dia ordenador que nos permita ponernos a cubierto de estos vaivenes. Ni a nosotros ni al mas superpoderoso gobierno de la Tierra. Esto para nosotros es un consuelo ya que es garantia que ningun tirano sera capaz, por el momento, de encontrar las claves que le permitan dominar la economia mundial.

Si descendemos al pequenyo nivel de nuestras humildes economias, ¨Cual puede ser el mejor consejo para evitar quedar prendido en una brusca y profunda bajada de la bolsa del mercado inmobiliario, del precio del oro o del petroleo?. Para empezar lo mejor es recordar que las inversiones financieras son un asunto de riesgo y como tal hay que tratarlo. Solo se debe invertir el dinero del que facilmente podamos prescindir y no nos sea necesario para sobrevivir. Si se nos ocurre no seguir este consejo y alguien se endeuda para comprar acciones o un apartamento con animo especulador, lo mas probable es que durante la proxima crisis actue como la masa y siga sus dictados ya que sera incapaz de ir contra las reglas fractales que se encuentran grabadas a fuego en su ADN.

El siguiente consejo es recordar que nadie tiene las claves del futuro. Los "soplos", las informaciones confidenciales, solo se convierten en ciertas en contadisimos casos, porque uno de los razones que mueven las acciones de la gente es el ansia de poder y el dinero. Una buena informacion carece de valor en cuanto se sabe y por tanto nunca se da gratis. Las informaciones confidenciales gratuitas no tienen ningun valor y no deben ser tenidas en cuenta. Las informaciones que puedan suministrar los agentes financieros entran dentro de esta categoria. No conocemos a ningun director de sucursal que se haya enriquecido, salvo los casos en que se fugado a Brasil con el contenido de la caja fuerte. Dado su poca pericia demostrada para generar dinero dudamos que sean capaces de darnos consejos para ayudar a enriquecernos. Hay que saber mirar a nuestro alrededor.

El tercer consejo es que ante una conmocion de los mercados, antes de hacer algo, lo mejor es observar el pasado. No tanto para intentar ver tendencias, sino para intentar adivinar como cuanto fue en el pasado inestable el bien que ha empezado a tambalearse. Y sobretodo, aplicar la logica y dejar los delirios de faciles ganancias para los que pretendan ganar fortunas jugando a la loteria.

Cifrando Simbolos Estaticos en ELF

Por fkt <fkt algarrobo metawire.org>

20/08/2005

- Índice:

---

1. Introduccion

Hay muchos articulos sobre cifrar binarios, y algunos muy buenos, como por ejemplo el de scut y grugq [1], pero en ninguno habla sobre como cifrar un simbolo del binario y no el binario entero, ¨Para que cifrar un solo simbolo? Pues la verdad es que tiene poca utilidad, o por lo menos yo no le he encontrado mas de las que comento en el punto 3, pero la idea es original. Algo que quiero que quede claro es que no voy a explicar las especificaciones de los ELF, para eso ya hay un texto que lo hace y que lo podeis encontrar en [2] por ejemplo.

2. Cifrando simbolos

Bueno, usaremos el siguiente codigo de conejo de indias:
(prog.c):

```
#include <stdio.h>

void lala(char *s) {
  printf("%s", s);
}

void prueba_sym(void) {
  lala("hola\n");
}

int main(void) {
  lala("este si funciona\n");
  prueba_sym();

  return 0;
}
```

El codigo es muy simple, se limita a hacer un par de printf's, vamos a intentar cifrar el simbolo "prueba_sym", para ello tendremos que buscarlo en el binario:
(busca_sym.c):

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <sys/mman.h>
#include <unistd.h>
#include <fcntl.h>
#include <elf.h>
#include <errno.h>

static void
muestra_simbolos(Elf32_Ehdr *ehdr, Elf32_Shdr *shdr, unsigned
```

```
                  char *gstrs, char *fich, void *mmap_ptr, char *search_sym);

int main(int argc, char *argv[]) {
  int fd, i;
  void *fmap;
  struct stat st;
  Elf32_Ehdr *ehdr;
  Elf32_Shdr *shdr;
  unsigned char *strings;

  if (argc != 3) {
    fprintf(stderr, "Uso: %s <fichero> <simbolo>\n", argv[0]);
    exit(EXIT_FAILURE);
  }

  if ((fd = open(argv[1], O_RDWR)) < 0) {
    perror("open()");
    exit(EXIT_FAILURE);
  }

  if (fstat(fd, &st) < 0) {
    perror("fstat");
    close(fd);
    exit(EXIT_FAILURE);
  }

  fmap = mmap(NULL, st.st_size, PROT_READ | PROT_WRITE, MAP_SHARED, fd, 0);
  if (fmap == MAP_FAILED) {
    perror("mmap()");
    close(fd);
    exit(EXIT_FAILURE);
  }

  ehdr = (Elf32_Ehdr *) fmap;
  if (memcmp(ehdr, ELFMAG, 4)) {
    fprintf(stderr, "%s no es un ELF valido\n", argv[1]);
    munmap(fmap, st.st_size);
    close(fd);
    exit(EXIT_FAILURE);
  }

  /*
   * Guardamos un puntero a las strings
   */
  shdr = (fmap + ehdr->e_shoff);
  shdr += ehdr->e_shstrndx;
  strings = (fmap + shdr->sh_offset);

  /*
   * Recorremos las cabeceras de seccion, saltando la primera
   */
  shdr = (fmap + ehdr->e_shoff);
  for (i = 1, shdr++; i < ehdr->e_shnum; i++, shdr++) {
    if (shdr->sh_type == SHT_SYMTAB)
      muestra_simbolos(ehdr, shdr, strings, argv[1], fmap, argv[2]);
  }

  if (munmap(fmap, st.st_size) == -1) {
    perror("munmap()");
    exit(EXIT_FAILURE);
  }

  close(fd);
  exit(EXIT_SUCCESS);
}

static void
muestra_simbolos(Elf32_Ehdr *ehdr, Elf32_Shdr *shdr, unsigned char *gstrs,
                 char *fich, void *mmap_ptr, char *search_sym) {
  long off_fich;

  void *fmap = ehdr;
  unsigned char *strings;
  Elf32_Shdr *strshdr, *tmp;
  Elf32_Sym *sym, *end;
```

```
    strshdr = (fmap + ehdr->e_shoff);
    strshdr += shdr->sh_link;
    strings = (fmap + strshdr->sh_offset);

    /*
     * Recorro la lista de simbolos
     */
    sym = (fmap + shdr->sh_offset);
    end = (fmap + shdr->sh_offset + shdr->sh_size);
    for (; sym < end; sym++) {
      int es_codigo = 0;

      switch (ELF32_ST_TYPE(sym->st_info)) {
        case STT_FUNC:
          es_codigo = 1;
          break;
      }

      /*
       * Si es codigo, decimos en que seccion esta
       */
      if (es_codigo) {
        if (!strcmp(strings + sym->st_name, search_sym)) {
          if (sym->st_shndx) {
            tmp = (fmap + ehdr->e_shoff);
            tmp += sym->st_shndx;

            printf("Simbolo: %s\n", search_sym);
            printf("\tst_shndx: %5i", sym->st_shndx);
            printf("\tst_value:   0x%08x\n\tst_size:   %6i\t",   sym->st_value,   sym-
>st_size);
            printf("st_name: \"%s\"\n", strings + sym->st_name);
            printf("\tEste simbolo esta en la seccion %s\n", gstrs + tmp->sh_name);

            off_fich = (sym->st_value - tmp->sh_addr) + tmp->sh_offset;
            printf("Posicion Relativa del simbolo en el fichero: %d\n", off_fich);
          }
        }
      }
    }
  }
```

Vamos a explicar el programa un poco, primero mapea el binario en memoria, comprueba si es un ELF comprobando el numero magico ELFMAG. Guardamos un puntero a las strings del binario para luego comprobar el nombre del simbolo. Lo siguiente que hacemos es buscar la seccion symtab, ya que es ahi donde estan los simbolos, una vez la encontramos saltamos a la funcion muestra_simbolos con los punteros ajustados correspondientemente, en esta funcion vamos mirando que simbolos son codigo (STT_FUNC) y si lo son, los comparamos con el que estamos buscando. Vamos a probarlo:

```
# ./busca_sym prog prueba_sym
Simbolo: prueba_sym
        st_shndx:    12 st_value: 0x0804838b
        st_size:     24 st_name: "prueba_sym"
        Este simbolo esta en la seccion .text
Posicion Relativa del simbolo en el fichero: 907
```

Parece que funciona, pero aun no hemos cifrado nada, aunque teniendo todo lo anterior ya es realmente facil, como tenemos el offset del simbolo en el fichero basta con irnos a fichero_mapeado + offset y cifrarlo...

NOTA: Voy a poner el resultado del diff con el busca_sym.c en vez de poner el codigo entero.

(busca_ciph.diff):

```
--- busca_sym.c 2005-08-21 20:45:17.873299944 +0200
+++ ciph_sym.c  2005-08-21 20:46:01.907605712 +0200
@@ -11,7 +11,7 @@

    static void
    muestra_simbolos(Elf32_Ehdr *ehdr, Elf32_Shdr *shdr, unsigned
-                    char *gstrs, char *fich, void *mmap_ptr, char *search_sym);
+                    char *gstrs, char *fich, void *mmap_ptr, char *search_sym, int
n_xor);

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

```
      int fd, i;
  @@ -21,8 +21,8 @@
       Elf32_Shdr *shdr;
       unsigned char *strings;

  -    if (argc != 3) {
  -      fprintf(stderr, "Uso: %s <fichero> <simbolo>\n", argv[0]);
  +    if (argc != 4) {
  +      fprintf(stderr, "Uso: %s <fichero> <simbolo> <n_xor>\n", argv[0]);
         exit(EXIT_FAILURE);
       }

  @@ -65,7 +65,12 @@
       shdr = (fmap + ehdr->e_shoff);
       for (i = 1, shdr++; i < ehdr->e_shnum; i++, shdr++) {
         if (shdr->sh_type == SHT_SYMTAB)
  -        muestra_simbolos(ehdr, shdr, strings, argv[1], fmap, argv[2]);
  +                muestra_simbolos(ehdr,  shdr,  strings,  argv[1],  fmap,  argv[2],
atoi(argv[3]));
  +    }
  +
  +    if (msync(fmap, st.st_size, MS_SYNC) == -1) {
  +      perror("msync()");
  +      exit(EXIT_FAILURE);
       }

       if (munmap(fmap, st.st_size) == -1) {
  @@ -79,8 +84,10 @@

     static void
     muestra_simbolos(Elf32_Ehdr *ehdr, Elf32_Shdr *shdr, unsigned char *gstrs,
  -                  char *fich, void *mmap_ptr, char *search_sym) {
  +                  char *fich, void *mmap_ptr, char *search_sym, int n_xor) {
       long off_fich;
  +    size_t k;
  +    char *ptr;

       void *fmap = ehdr;
       unsigned char *strings;
  @@ -122,6 +129,11 @@

               off_fich = (sym->st_value - tmp->sh_addr) + tmp->sh_offset;
               printf("Posicion  Relativa  del  simbolo  en  el  fichero:  %d\n",
off_fich);
  +
  +            ptr = (char *)mmap_ptr + off_fich;
  +
  +            for (k = 0 ; k < sym->st_size ; k++)
  +              ptr[k] ^= n_xor;
             }
           }
         }
```

Para aplicar el parche hacer:

```
# patch -i busca_ciph.diff -o ciph_sym.c busca_sym.c
(Patch is indented 2 spaces.)
patching file busca_sym.c
```

Ya tenemos nuestro ciph_sym.c, vamos a probarlo y luego comentamos los cambios que hemos hecho respecto al busca_sym.c.

```
# gcc ciph_sym.c -o ciph_sym
# ./prog
este si funciona
hola
# ./ciph_sym prog prueba_sym 3
Simbolo: prueba_sym
        st_shndx:    12 st_value: 0x0804838b
        st_size:     24 st_name: "prueba_sym"
        Este simbolo esta en la seccion .text
Posicion Relativa del simbolo en el fichero: 907
# ./prog
este si funciona
Segmentation fault
```

Funciona!, cuando intentamos acceder a prueba_sym() muere el proceso.
Y lo podemos restaurar:

```
# ./ciph_sym prog prueba_sym 3
Simbolo: prueba_sym
        st_shndx:    12 st_value: 0x0804838b
        st_size:     24 st_name: "prueba_sym"
        Este simbolo esta en la seccion .text
Posicion Relativa del simbolo en el fichero: 907
# ./prog
este si funciona
hola
#
```

Todo esto esta muy bien pero quien dice que alomejor hemos cifrado partes que no deberiamos? Vamos a comprobarlo...

Sobre el prog sin cifrar...

```
# objdump -S --start-address=0x804838b --stop-address=0x80483a3 prog
0804838b <prueba_sym>:
 804838b:       55                      push   %ebp
 804838c:       89 e5                   mov    %esp,%ebp
 804838e:       83 ec 08                sub    $0x8,%esp
 8048391:       83 ec 0c                sub    $0xc,%esp
 8048394:       68 a7 84 04 08          push   $0x80484a7
 8048399:       e8 d2 ff ff ff          call   8048370 <lala>
 804839e:       83 c4 10                add    $0x10,%esp
 80483a1:       c9                      leave
 80483a2:       c3                      ret

080483a3 <main>:
 80483a3:       55                      push   %ebp
```

Sobre el prog cifrado...

```
0804838b <prueba_sym>:
 804838b:       56                      push   %esi
 804838c:       8a e6                   mov    %dh,%ah
 804838e:       80 ef 0b                sub    $0xb,%bh
 8048391:       80 ef 0f                sub    $0xf,%bh
 8048394:                       6b    a4    87    07    0b    eb    d1              imul
$0xfffffffc,0xd1eb0b07(%edi,%eax,4),%esp
 804839b:       fc
 804839c:       fc                      cld
 804839d:       fc                      cld
 804839e:       80 c7 13                add    $0x13,%bh
 80483a1:       ca c0 55                lret   $0x55c0

080483a3 <main>:
 80483a3:       55                      push   %ebp
```

Como se puede ver el codigo cambia totalmente y ciframos lo que queremos, pues lo que va despues de prueba_sym() (que es el main) no resulta afectado por el cifrado.

Vamos a comentar ahora los cambios que hicimos en busca_sym.c para obtener ciph_sym.c:
El cifrado que hacemos es un simple XOR con un numero que se obtiene de la linea de comandos, como ya comente antes para cifrar el simbolo lo unico que tenemos que hacer es sumarle el offset al archivo mapeado y cifrarlo. Por ultimo para que los cambios tengan efecto sobre el binario en disco hacemos un msync().

2.1 Binarios sin stripear

Todo lo que hemos comentado en el apartado 2 vale para binarios, tanto dinamicos como estaticos que no esten stripeados. Obviamente hemos usado como algoritmo de cifrado un simple XOR pero se podria usar cualquier otro como por ejemplo Blowfish, pero eso ya os lo dejo a vosotros :).

2.2 Binarios stripeados

Nos topamos con el primer problema, si han hecho un strip del binario no podremos buscar el simbolo como hemos hecho antes. Pero no pasa nada, porque como ya tenemos el offset relativo del simbolo en el fichero bastaria con hacer...
(ciph_sym_strip.c):

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <sys/mman.h>
#include <unistd.h>
#include <fcntl.h>
#include <elf.h>
#include <errno.h>

void ciph_strip(void *mmap_ptr, long offset, int size, int n_xor);

int main(int argc, char *argv[]) {
  int fd;
  void *fmap;
  struct stat st;
  Elf32_Ehdr *ehdr;

  if (argc != 5) {
    fprintf(stderr,     "Uso:     %s     <fichero>     <n_xor>     <offset_relativo>
<st_size_del_simbolo>\n", argv[0]);
    exit(EXIT_FAILURE);
  }

  if ((fd = open(argv[1], O_RDWR)) < 0) {
    perror("open()");
    exit(EXIT_FAILURE);
  }

  if (fstat(fd, &st) < 0) {
    perror("fstat");
    close(fd);
    exit(EXIT_FAILURE);
  }

  fmap = mmap(NULL, st.st_size, PROT_READ | PROT_WRITE, MAP_SHARED, fd, 0);
  if (fmap == MAP_FAILED) {
    perror("mmap()");
    close(fd);
    exit(EXIT_FAILURE);
  }

  ehdr = (Elf32_Ehdr *) fmap;
  if (memcmp(ehdr, ELFMAG, 4)) {
    fprintf(stderr, "%s no es un ELF valido\n", argv[1]);
    munmap(fmap, st.st_size);
    close(fd);
    exit(EXIT_FAILURE);
  }

  ciph_strip(fmap, atol(argv[3]), atoi(argv[4]), atoi(argv[2]));

  if (msync(fmap, st.st_size, MS_SYNC) == -1) {
    perror("msync()");
    exit(EXIT_FAILURE);
  }

  if (munmap(fmap, st.st_size) == -1) {
    perror("munmap()");
    exit(EXIT_FAILURE);
  }

  close(fd);
  exit(EXIT_SUCCESS);
}

void ciph_strip(void *mmap_ptr, long offset, int size, int n_xor) {
  size_t k;
  char *ptr;

  ptr = (char *)mmap_ptr + offset;
  for (k = 0 ; k < size ; k++)
    ptr[k] ^= n_xor;
}
```

El codigo es como el de antes pero quitando la parte que buscaba el simbolo
Vamos a ver si funciona:

```
# ./busca_sym prog prueba_sym
Simbolo: prueba_sym
        st_shndx:    12 st_value: 0x0804838b
        st_size:     24 st_name: "prueba_sym"
        Este simbolo esta en la seccion .text
Posicion Relativa del simbolo en el fichero: 907
# strip prog
# file prog
prog: ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), for GNU/Linux
2.2.5, dynamically linked (uses shared libs), stripped
# ./prog
este si funciona
hola
# ./ciph_sym_strip prog 3 907 24
# ./prog
este si funciona
Segmentation fault
# ./ciph_sym_strip prog 3 907 24
# ./prog
este si funciona
hola
```

Voila! Funciona!

Otra forma de hacerlo seria poniendo algun tipo de marca en el codigo del fichero a cifrar, y asi no tendremos que saber el offset ni nada. Pero por que hacerlo de esta manera y no de la anterior que es mas facil? Sencillo, porque por ejemplo podemos inyectar codigo en un binario ya stripeado. Usaremos el siguiente programa para mostrar esta forma:
(prog2.c):

```
#include <stdio.h>

void lala(char *s) {
  printf("%s", s);
}

void prueba_sym(void) {
  int pepe, juan = 1;

  pepe = 1;

  lala("Soy un codigo bueeeeno\n");

  if (pepe == juan) {

    __asm__("movl $0xdeadbeef,%eax");

    lala("Soy un codigo malo maloso!!\n");

    __asm__("movl $0xdeadbeef,%eax");
  } else {
    printf("No soy nadie\n");
  }
}

int main(void) {
  prueba_sym();

  return 0;
}
```

Nuestra marca sera el deadbeef :), eso sera lo que buscaremos en el binario para empzar a cifrar/descifrar. Los limites de cifrado los marcaran los 2 deadbeef. Al programa de cifrado le daremos para que tome como base el offset de la seccion .text en el fichero para asi no tener que buscar inutilmente.
El codigo es este:
(deadbeef.c):

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

```
#include <sys/stat.h>
#include <sys/mman.h>
#include <unistd.h>
#include <fcntl.h>
#include <elf.h>
#include <errno.h>

void ciph_deadbeef(void *mmap_ptr, long offset, int n_xor, size_t mmap_size);

int main(int argc, char *argv[]) {
  int fd;
  void *fmap;
  struct stat st;
  Elf32_Ehdr *ehdr;

  if (argc != 4) {
    fprintf(stderr, "Uso: %s <fichero> <n_xor> <.text_offset>\n", argv[0]);
    exit(EXIT_FAILURE);
  }

  if ((fd = open(argv[1], O_RDWR)) < 0) {
    perror("open()");
    exit(EXIT_FAILURE);
  }

  if (fstat(fd, &st) < 0) {
    perror("fstat");
    close(fd);
    exit(EXIT_FAILURE);
  }

  fmap = mmap(NULL, st.st_size, PROT_READ | PROT_WRITE, MAP_SHARED, fd, 0);
  if (fmap == MAP_FAILED) {
    perror("mmap()");
    close(fd);
    exit(EXIT_FAILURE);
  }

  ehdr = (Elf32_Ehdr *) fmap;
  if (memcmp(ehdr, ELFMAG, 4)) {
    fprintf(stderr, "%s no es un ELF valido\n", argv[1]);
    munmap(fmap, st.st_size);
    close(fd);
    exit(EXIT_FAILURE);
  }

  ciph_deadbeef(fmap, atol(argv[3]), atoi(argv[2]), st.st_size);

  if (msync(fmap, st.st_size, MS_SYNC) == -1) {
    perror("msync()");
    exit(EXIT_FAILURE);
  }

  if (munmap(fmap, st.st_size) == -1) {
    perror("munmap()");
    exit(EXIT_FAILURE);
  }

  close(fd);
  exit(EXIT_SUCCESS);
}

void ciph_deadbeef(void *mmap_ptr, long offset, int n_xor, size_t mmap_size) {
  size_t k, size_ciph = 1;
  char *ptr, *begin_ciph;

  ptr=(char *)mmap_ptr + offset;
  for (k = 0 ; k < (mmap_size-1) ; k++) {
    if (!memcmp(ptr+k, "\xef\xbe\xad\xde", 4)) {
      printf("Encontrado deadbeef de inicio en: <.text+%d>\n", k);
      break;
    }
  }

  begin_ciph = ptr + k + 4;
```

```
    k += 4;

    for ( ; k < (mmap_size-1) ; k++) {
      size_ciph++;

      if (!memcmp(ptr+k, "\xef\xbe\xad\xde", 4)) {
        printf("Encontrado deadbeef de final en: <.text+%d>\n", k);
        break;
      }
    }

    for (k=0 ; k < (size_ciph-3) ; k++)
    begin_ciph[k] ^= n_xor;
  }

  Probemoslo!...
  # gcc deadbeef.c -o deadbeef
  # gcc prog2.c -o prog2
  # strip prog2

  (Obtenemos el offset de .text)
  # objdump -h prog2

  prog2:     file format elf32-i386

  Sections:
  Idx Name          Size      VMA       LMA       File off  Algn
  ...
   11 .text         000001f4  080482c0  080482c0  000002c0  2**2
  ...

  (Pasamos de hexa a decimal)
  # perl -e 'print(0x2c0);'
  704
  # ./prog2
  Soy un codigo bueeeeno
  Soy un codigo malo maloso!!
  # ./deadbeef prog2 3 704
  Encontrado deadbeef de inicio en: <.text+248>
  Encontrado deadbeef de final en: <.text+269>
  # ./prog2
  Soy un codigo bueeeeno
  Segmentation fault
  # ./deadbeef prog2 3 704
  Encontrado deadbeef de inicio en: <.text+248>
  Encontrado deadbeef de final en: <.text+269>
  # ./prog2
  Soy un codigo bueeeeno
  Soy un codigo malo maloso!!

  Echemos un vistazo desde el punto de vista en ensamblador:
  Sobre el binario sin cifrar...
  #   objdump   -D   --section=.text   prog2   --start-address=0x804838b   --stop-
address=0x80483e7

  prog2:     file format elf32-i386

  Disassembly of section .text:

  0804838b <.text+0xcb>:
   (Aqui empieza prueba_sym)
   804838b:       55                      push   %ebp
   804838c:       89 e5                   mov    %esp,%ebp
   804838e:       83 ec 08                sub    $0x8,%esp
   8048391:       c7 45 f8 01 00 00 00    movl   $0x1,0xfffffff8(%ebp)
   8048398:       c7 45 fc 01 00 00 00    movl   $0x1,0xfffffffc(%ebp)
   804839f:       83 ec 0c                sub    $0xc,%esp
   80483a2:       68 db 84 04 08          push   $0x80484db
   80483a7:       e8 c4 ff ff ff          call   8048370 <printf@plt+0xc0>
   80483ac:       83 c4 10                add    $0x10,%esp
   80483af:       8b 45 fc                mov    0xfffffffc(%ebp),%eax
   80483b2:       3b 45 f8                cmp    0xfffffff8(%ebp),%eax
   80483b5:       75 1c                   jne    80483d3 <printf@plt+0x123>
   80483b7:       b8 ef be ad de          mov    $0xdeadbeef,%eax
   80483bc:       83 ec 0c                sub    $0xc,%esp
```

```
 80483bf:       68 f3 84 04 08          push   $0x80484f3
 80483c4:       e8 a7 ff ff ff          call   8048370 <printf@plt+0xc0>
 80483c9:       83 c4 10                add    $0x10,%esp
 80483cc:       b8 ef be ad de          mov    $0xdeadbeef,%eax
 80483d1:       eb 10                   jmp    80483e3 <printf@plt+0x133>
 80483d3:       83 ec 0c                sub    $0xc,%esp
 80483d6:       68 10 85 04 08          push   $0x8048510
 80483db:       e8 d0 fe ff ff          call   80482b0 <printf@plt>
 80483e0:       83 c4 10                add    $0x10,%esp
 80483e3:       c9                      leave
 80483e4:       c3                      ret
 (Aqui ya empieza el main)
 80483e5:       55                      push   %ebp
 80483e6:       89 e5                   mov    %esp,%ebp
```

Sobre el binario cifrado...

```
0804838b <.text+0xcb>:
 804838b:       55                      push   %ebp
 804838c:       89 e5                   mov    %esp,%ebp
 804838e:       83 ec 08                sub    $0x8,%esp
 8048391:       c7 45 f8 01 00 00 00    movl   $0x1,0xfffffff8(%ebp)
 8048398:       c7 45 fc 01 00 00 00    movl   $0x1,0xfffffffc(%ebp)
 804839f:       83 ec 0c                sub    $0xc,%esp
 80483a2:       68 db 84 04 08          push   $0x80484db
 80483a7:       e8 c4 ff ff ff          call   8048370 <printf@plt+0xc0>
 80483ac:       83 c4 10                add    $0x10,%esp
 80483af:       8b 45 fc                mov    0xfffffffc(%ebp),%eax
 80483b2:       3b 45 f8                cmp    0xfffffff8(%ebp),%eax
 80483b5:       75 1c                   jne    80483d3 <printf@plt+0x123>
 80483b7:       b8 ef be ad de          mov    $0xdeadbeef,%eax
 (Empieza la parte cifrada)
 80483bc:       80 ef 0f                sub    $0xf,%bh
 80483bf:       6b f0 87                imul   $0xffffff87,%eax,%esi
 80483c2:       07                      pop    %es
 80483c3:       0b eb                   or     %ebx,%ebp
 80483c5:       a4                      movsb  %ds:(%esi),%es:(%edi)
 80483c6:       fc                      cld
 80483c7:       fc                      cld
 80483c8:       fc                      cld
 80483c9:       80 c7 13                add    $0x13,%bh
 (Acaba la parte cifrada)
 80483cc:       b8 ef be ad de          mov    $0xdeadbeef,%eax
 80483d1:       eb 10                   jmp    80483e3 <printf@plt+0x133>
 80483d3:       83 ec 0c                sub    $0xc,%esp
 80483d6:       68 10 85 04 08          push   $0x8048510
 80483db:       e8 d0 fe ff ff          call   80482b0 <printf@plt>
 80483e0:       83 c4 10                add    $0x10,%esp
 80483e3:       c9                      leave
 80483e4:       c3                      ret
 80483e5:       55                      push   %ebp
 80483e6:       89 e5                   mov    %esp,%ebp
```

Como podemos comprobar funciona como queriamos. Quizas esta ultima sea la forma mas generica de hacerlo.

3. Aplicaciones

En este punto comentare las aplicaciones que yo le he contrado a todo esto, lo cual no quiere decir que no existan otras, y si encontrais alguna os agradeceria que me la dijeseis.

3.1 Shareware

Puede ser util para hacer software shareware, le damos al usuario el programa completo pero con la parte que queramos cifrada, cuando lo compre le damos el codigo para que pueda descifrarla y ya esta, asi evitaremos que se salten el chequeo de si esta o no registrado que tienen los tipicos programas shareware, porque no servira de nada saltarselo.
Obviamente a lo largo de este documento hemos usado un simple XOR, pero el algoritmo se puede complicar todo lo que queramos.

3.2 Anti-Forensics

Bueno, esto que ahora esta muy de moda, los analisis forenses, podemos evitar que en el analisis forense a la maquina comprometida sepan que codigo

malicioso intentabamos ejecutar, simplemente cifrando lo que queramos. Incluso podemos infectar un binario y hacer que el mismo autodescifre el codigo y lo ejecute en memoria sin tener que escribir el codigo malicioso en texto plano en disco, pero eso se sale de este articulo.

3.3 Otras

Pon aqui tu aplicacion favorita :))

4. Agradecimientos

Jamas me cansare darle las gracias a este hombre, Doing GRACIAS POR TODO!

5. Referencias

[1]: http://www.phrack.org/phrack/58/p58-0x05
[2]: http://www.muppetlabs.com/~breadbox/software/ELF.txt

EOF

-[0x0D]--
-[IDA para neofitos]--
-[by FCA00000]---SET-33--

IDA_info

En este articulo se cuenta informacion relevante al desensamblador
IDA = The Interactive Disassembler, y modos de mejorarlo.
Si has leido alguno de los textos escritos por mi, habras encontrado que
continuamente hago referencia a este desensamblador.
Tras muchas horas pasadas con este programa, creo que seria util compartir
con vosotros lo que yo he aprendido.

En la primera parte explico como hacerlo funcionar. Luego comento algunos
trucos que a mi me resultan utiles, y al final cuento varias maneras de
mejorarlo, completando la funcionalidad que se echa de menos.

Yo lo uso porque es el mejor para desensamblar programas de ARM, usado en
moviles Symbian. Tambien es capaz de desensamblar codigo de otros
procesadores, tales como Z80, 80x86, NET, Playstation, XBox, e incluso
Java y programas de Linux y Windows CE.

La ultima version que yo conozco es IDA Pro Advanced 5.0
Es una lastima que este disponible en tantos sitios de warez: eso solo hace
que los autores (www.Datarescue.com) pierdan interes en seguir desarrollando
su producto. Como siempre digo: si usas una aplicacion, comprala.

Recuerda que algunas leyes prohiben el desensamblado de programas ajenos.
En general se admite que desensambles programas hechos por ti.
Por tanto, desensamblar un virus que te ha infectado es ilegal :-%

Este parrafo explica por encima como manejarse con IDA. Obviamente la
mejor manera de adquirir practica es experimentando, asi que no te
extranye si vamos un poco rapido al principio.

Tras iniciar el programa lo primero que se hace es abrir el programa que
quieres desensamblar.
Si es capaz de identificar el procesador para el que esta escrito el
programa, lo sugiere. Si no, solicita el tipo de procesador.
La mayoria de las aplicaciones que yo he destripado han sido identificadas
correctamente. He tenido dificultades con los programas del ZX-Spectrum,
porque no entiende el formato, pero eso es normal.
Tambien da problemas con los programas extraidos de la memoria de los
moviles Symbian, pues la cabecera es incorrecta, y se lia al encontrar el
punto de entrada.

Este problema es facilmente solucionable eligiendo el tipo de fichero
como Binary en lugar de EPOC.
Esto me permite escribir el punto de entrada, y la direccion de carga, que
para Symbian resulta ser 0x64 bytes menos de lo que indica el programa a
desensamblar. Para los programas del Spectrum, hay que mirar el formato en
el que se guardan. El particular el formato SNA contiene una cabecera que
ocupa 48 bytes, seguido por el volcado de la memoria a partir de la
direccion 0x4000.

Entonces empieza automaticamente a desensamblar el programa, y a continuacion
presenta el listado; bien en modo de arbol de rutinas, bien en modo texto.

Lo importante de un desensamblado es poder seguir el flujo de informacion.
Si una rutina llama a otra, puedes hacer doble-click para verla.
Pulsando "escape" vuelves a la inicial.
Seleccionando una rutina, o una etiqueta, puedes pulsar "X" para ver desde
donde se llama; posiblemente desde varios sitios.
El menu View-OpenSubviews->FunctionCalls habilita una ventana flotante para
hacer mas facil esta tarea. Asi sabes de donde vienes, a donde vas, y
porque tan rapido.

Puedes abrir una ventana nueva (Alt+Enter) para investigar 2 rutinas a la vez.

Con las teclas F12 y CTRL+F12 puedes ver el flujo completo de las rutinas,
presentado en la aplicacion WinGraph (licencia GPL).
Eso si, el grafico completo de todas las rutinas puede que sea bestialmente
grande.

Mas eficiente es colocar el cursor en una rutina determinada y usar el boton derecho para sacar el grafico parcial de "Chart of xrefs to".

IDA permite escribir comentarios en el codigo pulsando ";".
Para renombrar las subrutinas y constantes hay que marcarlas con el cursor y pulsar "N".
Si quieres ver un dato en otra base numerica (Hex/Dec/Oct/Bin/ASCII) solo hay que marcarla y pulsar el boton derecho del raton.

Una parte fundamental del desensamblado es saber la representacion real de los datos. Para ello puedes abrir la vista HexDump para ver en hexadecimal los codigos de las instrucciones.
Posiblemente tambien quieras usar BotonDerecho->SynchronizeWith->IDA View-A para ver exactamente los datos de la instruccion que estas analizando.
Es una lastima que no permita modificar los datos en vivo, pero cualquier editor hexadecimal te sirve para esto.

Normalmente los programas llaman a otras rutinas del sistema operativo.
IDA sabe cuales son estas rutinas y sus parametros.
Un ejemplo de desensamblado de un programa en Windows tiene las instrucciones
```
.text:000129FB    push    offset SourceString ; SourceString
.text:00012A00    push    [ebp+DestinationString] ; DestinationString
.text:00012A03    mov     [ecx], eax
.text:00012A05    call    ds:RtlInitUnicodeString
```

y luego
```
.rdata:00013018 ; void __stdcall RtlInitUnicodeString(PUNICODE_STRING

                                DestinationString,PCWSTR SourceString)
.rdata:00013018 RtlInitUnicodeString dd ?    ; DATA XREF: sub_129A05
```

O sea, que sabe que 00012A05 llama a RtlInitUnicodeString y que los argumentos se definen en 000129FB y 00012A00

No solo esto, sino que es capaz de decir las librerias importadas por un programa. Puede extaer informacion de modulos compilados con informacion de debug, usados comunmente en Windows y Linux.

Tambien entiende constantes alfanumericas en varias implementaciones.
Una palabra, por ejemplo "Ayax" se puede representar con los bytes:
41 79 61 78 00 -> un byte por letra y el terminador 00 , tipico de lenguaje C
41 79 61 78 24 -> un byte por letra y el terminador 00 , usado en MS-DOS
41 00 79 00 61 00 78 00 00 00 -> dos bytes por letra y el terminador

doble-00 , tipico de unicode
04 00 00 00 41 00 79 00 61 00 78 00 00 00 -> cuatro bytes indicando la

longitud, seguidos por dos bytes por letra

y

el terminador doble-00 , tipico de unicode de

Symbian, llamado Unicode-Width-Pascal-4
Hay otros mas pero estos son los que IDA reconoce sin muchos problemas.
Bueno, el ultimo formato si da problemas, pero los veremos despues.

Para buscar un texto, instruccion, constante o cualquier otra cosa, usas las teclas Alt-T y lo escribes en el cuadro de dialogo. Para buscar datos en la ventana de representacion hexadecimal usas Alt-B

Otra opcion bastante potente es exportar los datos desensamblados. En el menu Options->General->Analysis->TargetAssembler puedes elegir un ensamblador . Luego en el menu File->ProduceFile->Create_ASM_File puedes extraer el listado, y ensamblarlo de nuevo.
En Symbian hay que hacerle algunos ajustes, pero acaba funcionando igual que el original. Todo depende del tiempo que quieras dedicarle.

Otra manera de extraer datos es usando el menu ProduceFile-Dump_database_to_IDC_file
El archivo generado contendra la informacion relevante para IDA: inicio de las rutinas, nombres dados a las variables, comentarios, referencias, constantes, puntos de entrada, segmentos, ...
Esto tiene un gran utilidad para hacer que IDA haga cosas que no estaban consideradas por los creadores de la herramienta, y sera usado en el tercer apartado de este articulo.

Otras ventanas utiles son las que muestran en una lista todas las funciones importadas y exportadas, o internas. Similarmente se puede ver el listado de todas las cadenas de texto: Shift-F12 . No te olvides de pulsar la etiqueta "String" para ordenarlas.

Yo suelo abrir la ventana de desensamblado, el FunctionCalls, y el volcado Hexadecimal. A veces abro 2 desensamblados. Con esto, me apanyo, aunque un monitor de 20 pulgadas no me vendria mal.

Es bastante util el menu Jump-MarkPosition (Alt-M) para marcar una posicion. Luego se puede volver a ella usando Ctrl-M . Obviamente lo mejor es que se pueden marcar varias posiciones y darles nombre que sean intuitivos.
Como no todo podia ser perfecto, esta informacion no se guarda en el IDC .

Bueno; hasta aqui es un minimo manual de uso.

Ahora comentare algunos trucos que yo encuentro utiles. Estan motivados por el uso que yo le doy a esta herramienta, habitualmente desensamblando programas para Symbian.

Lo primero es que el desensamblado automatico empieza por la rutina de inicio, y desensambla en forma de arbol: si A llama a B, entonces B tambien sera desensamblada. Esto se llama desensamblado de flujo lineal.
Pero si alguna rutina no esta llamada desde otra, quizas no la desensambla.
IDA tiene una metodologia de desensamblado llamada "Recursive traversal" en la que intenta everiguar si un trozo de bytes es codigo o datos.
Pero como todo artifacto fabricado por humanos, a veces falla.

Este problema no se percibe normalemente al desensamblar un programa, pero si en las librerias.
La solucion es marcar la primera linea del programa (Alt-L), hacer scroll hasta la ultima (Ctrl-PageDown), y pulsar "C" para convertirlo en codigo.
Lo mismo sirve para convertirlo en una cadena de caracteres, si notamos que en realidad son caracteres imprimibles.

El segundo se refiere a la busqueda. Normalmente pulsas Alt-T para decir el texto que quieres buscar, y luego Ctrl-T sigue la busqueda.
Pero es mucho mas eficiente poner el cursor en la palabra a buscar, y pulsar Alt-FlechaAbajo, o Alt-FlechaArriba . Por supuesto esto sirve para cualquier palabra: datos, nombres de rutinas, o incluso instrucciones.
Por poner un ejemplo, suponer que estas analizando una rutina que usa el registro R8, y quieres ver donde se ha inicializado. Asi que pones el cursor en la palabra "R8" y Alt-FlechaArriba te lleva a la instruccion anterior que usa tal registro. No va a resolver el hambre en el mundo, pero acelera el trabajo de analisis.

Otro truco tiene que ver con las constantes. Es posible asignarles nombres, o elegir de los nombres estandar de Windows, que quizas esten definidos en otro sitio. (Nota: ?Porque solo funcionara en Windows, y no en Symbian?)
Supongamos este codigo en windows:
```
.text:0001290F  call    ds:KfReleaseSpinLock
.text:00012915  mov     eax, 0C000009Ah
```

Ahora bien: ?que es el valor 0x0C000009Ah? Desde luego no parece elegido al azar.
Asi que pongo el cursor sobre este valor, pulso el boton derecho, y digo "Use standard symbolic constants".
Me sugiere utilizar el simbolo STATUS_INSUFFICIENT_RESOURCES , lo cual es consistente con la funcion KfReleaseSpinLock

Esto desde luego ayuda a saber lo que esta haciendo el programa.

Claro que a veces hay que romperse un poco la cabeza: el valor 0x39h en windows tiene estos posibles significados:
```
ATM_CAUSE_BEARER_CAPABILITY_UNAUTHORIZED
CR_INVALID_CONFLICT_LIST
DPFLTR_IDEP_ID
FILE_DEVICE_KSEC
LANG_HINDI
....
```
y por supuesto, mirando el codigo alrededor:
```
00012BC8     mov     di, [ecx]
00012BCB     cmp     di, 30h
```

```
00012BCF     jb      short loc_12BD7
00012BD1     cmp     di, 39h
00012BD5     jbe     short loc_12BDE
```

nos da la pista de que se compara si el valor de di esta entre "0" y "9",
es decir, verifica que es numerico.
O sea, que no es ninguna de estas constantes, sino simplemente el valor "9".

Si hubieramos elegido LANG_HINDI , al hacer doble-click el propio IDA abre
la ventana de Enumeraciones de tipo MACRO_LANG que nos indica que otros
posibles valores son:

```
LANG_AFRIKAANS   = 36h
LANG_GEORGIAN    = 37h
LANG_FAEROESE    = 38h
LANG_HINDI       = 39h
LANG_MALAY       = 3Eh
.....
```

y esto tambien ayuda a la hora de cambiar una variable por otra. Supongamos
que en otro programa se chequea que el lenguaje de instalacion es LANG_HINDI,
pero nuestro ordenador esta instalado en lenguaje LANG_MALAY . Entonces hay
que cambiar la comparacion

```
cmp     di, 39h
```

por

```
cmp     di, 3Eh
```

pero vamos, dudo que existan protecciones de este tipo.

A lo que iba, que en seguida me despisto.
A veces la variable no esta conocida en ninguna de las librerias por defecto.
Bueno, en esta ventana "Enums" es posible definir nuevos simbolos y usarlos
en cualquiera de las posteriores sesiones de desensamblado.

Otro truco: habras notado que si seleccionas una palabra, se marca en color
amarillo. Esto permite identificar rapidamente en pantalla todas las
ocurrencias de dicha palabra. Pero si pinchas en cualquier otro punto,
pierdes la seleccion. Para evitar esto puedes usar el icono
"Lock the current highlight".

Junto a este icono hay otro con muchos colorines que muestra un mapa del
programa. En color Azul oscuro corresponde al codigo desensamblado, en rosa
las funciones importadas, en gris los datos, burdeos para el codigo que no
tiene sentido, marron para las areas vacias, y azul claro para las funciones
de librerias.
Pues bien, es posible adecuar este area para que muestre otros tipos de
informaciones. Para ello se usa el dropdown llamado "Aditional display".
Por ejemplo, seleccionando "Marked positions" nos muestra un punto rojo en
el sitio donde hayamos puesto marcas.
Esto sirve para tener una vision de zonas de programa. Lastima que desde
esta ventana no se puede saltar a la direccion donde esta el codigo.

Cuando desensamblas un programa es bastante frecuente tener que mirar a la
vez en 4 sitios, asi que una navegacion eficiente es fundamental.

Ah, cuando buscas algo a lo largo del programa, esta ventanita muestra un
cursor que indica donde esta buscando. Asi sabes el porcentaje que lleva
analizado.

Otros de los puntos fuertes de IDA es que incluye un debugger. No lo he
usado apenas asi que no voy a comentar nada. Mi esperanza era que fuese
capaz de simular procesadores ARM, pero solo funciona para Windows y Linux,
los cuales, como digo, no he investigado en profundidad.

El mejor truco que puedo recomendar es usar un monitor grande. O dos. No se
como lo hago, pero nunca tengo suficiente espacio para trabajar comodamente.

Estos consejos ayudan a manejarse con el programa, pero la verdadera
potencia es que deja varias puertas abiertas para complementar la tarea.

Para comprender un programa, un debugger casi nunca es suficiente por si
mismo. A menudo hay que realizar tareas especificas que los disenyadores
de IDA no habian previsto.

Por ejemplo, suponer que quiero buscar todas aquellas variables que se inician con un valor entre 0x30 y 0x39 .
La unica manera es usar la opcion "Buscar" para localizar 0x3 (notar que me he dejado el ultimo digito) y anotar uno por uno los datos. Pero esto tambien encuentra datos tales como 0x3F , 0x399999 , ...

Gracias a que puedo extraer la informacion en un fichero ASM, puedo hacer un mini-programa en Perl (o usando grep , o AWK) para buscar esas ocurrencias:
grep "0x3?" salida.asm

Supongamos una tarea ligeramente mas compleja: sacar aquellas rutinas que:
-comparan un valor con otro
-siguen 4 o menos instrucciones
-saltan a otro sitio si el resultado no es cero

Un ejemplo seria:
01002E81 cmp dword_1008828, ebx
01002E87 mov eax, [ebp+wParam]
01002E8A jnz short loc_1002EF1

Esto se puede hacer tambien el Perl, algo asi:

-lee linea
-si es "cmp" entonces haz cnt=1
-si es "jnz" y cnt>0 entonces haz cnt=-1 , y muestra la direccion
-si cnt>0 entonces incrementa cnt
-si cnt>4 entonces cnt=-1

No es tan dificil, sobre todo haciendolo en Perl.

Otro caso: tomar una rutina, y ver desde donde es llamada, hasta 3 niveles de profundidad.
Para verlo mas claro suponer que la rutina se llama A. Entonces son validas:
- B, si llama a A
- C, si llama a B
- D, si llama a C

pero si E llama a D, entonces A esta mas alla de 3 niveles, y no vale.

Ahora las cosas se le complican a Perl, pues solo es capaz de leer lineas secuencialmente; no puede ir hacia atras, y mucho menos hacer arboles de llamadas.
Bueno, si que se puede hacer, pero yo no se como :-)

Lo que necesitamos es un sistema que pueda moverse por el codigo adelante y hacia atras, siguiendo flujos de subrutinas y multiples caminos. ?Quien puede hacer esto? Pues el propio IDA.

Sus desarrolladores tuvieron la misma idea, y decidieron permitir que se pudiera re-programar. Para ello inventaron un lenguaje llamado idc que es muy parecido al lenguaje C .

Para saber las cosas que se pueden hacer es bueno abrir el fichero de ayuda, y saltar al tema "Index of IDC functions". Presenta unas 400 funciones, pero tranquilo, que la mitad no se usan. El resto contiene muchas que son parecidas entre si: ej, GetStrucNextOff es similar a GetStrucPrevOff .

Las funciones tambien estan documentadas en el fichero idc.idc , por si quieres imprimirlas todas.

Vamos con un ejemplo sencillo: strstr
La declaracion es:
long strstr(string str,string substr); // find a substring, -1 = not found

O sea, que necesita un primer argumento con una frase, y un segundo argumento con la palabra a buscar.
Asi, este programa:

```
#include <idc.idc>
static main(void)
 {
```

```
 auto frase, palabra;
 frase = "Hay palabras dentro de esta frase.";
 palabra = "palabra";
 if( strstr(frase,palabra) >= 1)
      {
      Message("Encontrado!");
      }
 else
      {
      Message("No encontrado!");
      }
 return;
 }
```

Al cargarlo y ejecutarlo con F2 mostrara el mensaje "Encontrado!" en la ventana que esta en la parte inferior de IDA, llamada "Messages Window". Si no ves esta area, amplia la barra inferior de la aplicacion.

Tambien se puede usar la funcion
Warning
para mostarlo en una ventana de dialogo.

Otro programilla; esta vez le solicito al usuario una direccion de programa, y le digo a IDA que desensamble 8 bytes a partir de ahi:

```
#include <idc.idc>
static main(void)
 {
 auto mi_dir;
 mi_dir=0x0;
 mi_dir=AskAddr(mi_dir,"Introduce una direccion");
 AnalyzeArea(mi_dir,mi_dir+8);
 return;
 }
```

Un poco mas amigable es que el usuario ponga el cursor en una direccion, y desensamble desde ahi.
Para ello se usa

```
long ScreenEA(); // the current screen ea (linear address of cursor)
```

de este modo:
```
mi_dir=ScreenEA(); // en vez de mi_dir=0x0;
```

Para el caso que he propuesto de las busquedas en 3 niveles, debo usar
```
long    RfirstB (long To);  // Get first code xref to "To"
```
que busca referencias hacia esta direccion

y sus hermana
```
long    RnextB  (long To,long current); // Get next code xref to "To"
```
que busca posteriores referencias.

Nota: existen variaciones de estas funciones que buscan referencias desde codigo/datos, adelante/atras, estandar/extendidas.

El programa queda asi:

```
#include <idc.idc>
static main(void)
 {
 auto mi_dir;
 AnalyzeArea(INF_MIN_EA,INF_MAX_EA); // primero hay que analizar el codigo

// completo para que IDA marque las referencias
 mi_dir=ScreenEA();
 mi_dir=AskAddr(mi_dir,"Introduce una direccion");
 for ( x=RfirstB(ea); x != BADADDR; x=RnextB(ea,x) ) // por cada direccion
      {                                              // que apunta a mi_dir ...
      Message("Nivel 1: " + atoa(x) + "\n");          // imprimela
      for ( y=RfirstB(x); y != BADADDR; y=RnextB(ea,y) ) // y busca aquellas que
       {                                             //  la re-referencian
       Message("Nivel 2: " + atoa(y) + "\n");     // imprime las de segundo nivel
       for ( z=RfirstB(y); z != BADADDR; z=RnextB(ea,z) ) // y sigue buscando las
```

```
            {                                                       //  que la re-re-
referencian
            Message("Nivel 3: " + atoa(z) + "\n");
            }
       }
 return;
 }
```

Esto saca un listado, que es posible mandar a un fichero usando la variable
set IDALOG=c:\refs3Nivel.txt

Mas elegante es mandarlo a un fichero nombrado como la direccion misma:

```
nombre_archivo = "c:\\" + "refs_" + atoa(mi_dir) + ".txt"

my_archivo = fopen(nombre_archivo, "w");
.....
dato_hacia_fichero = "Nivel 3: " + atoa(z) + "\n";
writestr(my_archivo, dato_hacia_fichero);
....
fclose(my_archivo);
```

Como ves, bastante parecido al lenguaje C . Esto simplifica la curva de aprendizaje, suponiendo que ya sabes programar en C , claro.

Antes he dicho que no es posible guardar las marcas que pones cuando estas trabajando en el programa. Bueno, es posible hacerlo en lenguaje idc :

```
#include <idc.idc>
static main(void)
{
 for (slot = 1; slot < 1023; slot = slot + 1)
      {
      pos = GetMarkedPos(slot);
      if (pos>=0)
       {
       Message("Marca " + slot + " en direccion " + ltoa(pos) +
                              " con nombre " + GetMarkComment(slot) + "\n");
       }
      }
}
```

La mayoria de las cosas que se hacen a mano es posible tambien hacerlas desde un programa.
Hemos visto antes que se le pueden dar nombre a las constantes.
?Quieres saber como se hace desde idc ? Muy facil; usando

```
success SetConstName(long const_id,string name);
// ** set a comment of a symbolic constant
//         arguments:      const_id - id of const
//                         cmt     - new comment for the constant
//                         repeatable - 0:set regular comment
//                                      1:set repeatable comment
//         returns:        1-ok,0-failed
```

Por ejemplo:

```
SetConstName(0xFCA00000, "El_autor" );
```

hara que todas las ocurrencias de 0xFCA00000 pasen a llamarse "El_autor" .

Unas funciones muy utiles son las que convierten uno o mas bytes en otro tipo de datos.
Por ejemplo, en Symbian es comun que los bytes
10 00 ?? ??
se refieran a un UID - identificador de aplicacion.
Obviamente IDA no es consciente de ello, pero podemos forzarle:

```
dir_base = BeginEA();
dir_data = Dword (dir_base + 4 * 0x12 ); // el valor en la
                                                                        //
direccion (base+4*0x12) dice donde empieza el
                                                                       //
segmento de datos
dir_end = MaxEA(); // final del fichero analizado
for (dir=dir_data; dir<dir_end; dir=dir+4 ) // recorre el programa
```

```
 {
 dato = Dword(dir); // considera 4 bytes como un double-word, en memoria
 if (dato>=0x10000000 && dato<=0x1000FFFF  ) // si es 1000???? , entonces...
      {
      MakeWord(dir); // convierte los 4 bytes en una Double-Word, en el listado
      MakeName( dir, "UID_" + ltoa(dir,16) ) ; // le da nombre "UID_1000????"
      }
 }
```

Si no nos gustan los nombres de subrutinas que IDA ha generado se pueden cambiar con
`MakeName (long ea,string name);  // assign a name to a location`

Incluso a veces no me gustan los nombres de las instrucciones, asi que los renombro usando la funcion
`SetManualInsn (long ea, string insn);`

Este es un caso que yo necesito usar porque mi compilador no es capaz de entender la instruccion
`STMFD   SP!, ...`
sino que quiere que se llame
`STMFD!  SP, ...`

Por eso tengo una rutina que hace

```
for (my_dir=dir_base; my_dir<dir_end; my_dir=my_dir+4 )
 {
 ins=GetManualInsn(my_dir);
 if(strstr(ins,"STMFD   SP!")
       {
       ins_nueva="STMFD!  SP,"; // nueva instrucci;on
       ins_nueva=ins_nueva+substr(ins,11,-1); // concatena los caracteres

                                    // desde 11 hasta el final
       SetManualInsn(my_dir,ins_nueva);
       }
 }
```

Ah, y podemos generar el fichero ASM con todo el desensamblado:

```
GenerateFile( OFILE_ASM,
                                        fopen("salida.ASM","w"),
                                        dir_base,
                                        dir_end,
                                        GENFLG_MAPDMNG);
```

?Recordais que he dicho que no se puede modificar el programa binario cargado? Menti. Solo hay que usar la funcion
`void    PatchByte       (long ea,long value);`

Vamos con otro ejemplo: supongamos que quiero parchear un programa para que cambie unos bytes por otros.
Para ello me invento un formato:
`replace:3B284ADD3C202060CDA800682D18-FF284ADD3C202060CDA800682D18`
(en realidad no me lo he inventado: es el formato usado para parchear programas del Siemens-SX1. Notar que un fichero puede contener varias lineas con varios parches.)

La cadena inicial antes de ":" es lo que hay que buscar, y la cadena que sigue son los nuevos datos.
Cada 2 caracteres se combinan para generar un byte. Por ejemplo 3B significa el byte 0x3B .

Como veis, en este caso la cadena inicial y final solo se diferencian en el primer byte. Los restantes son para comprobar que solo queremos cambiar esta occurencia: si encuentra
`3B211111111111`
entonces no lo cambiara.

Para aplicar este parche a un fichero cargado en IDA hago un programilla:

```
#include <idc.idc>
static main(void)
```

```
{
parchea();
}

static parchea()
{
auto patchFileName, patchFile, string;
patchFileName = AskFileEx(0,"*.sxp","Elige el archivo con el parche");
patchFile = fopen(patchFileName, "rb"); // abre el archivo con los parches
while(string != -1) // sigue mientras no llegue al final del fichero
 {
 cad_leida = readstr(patchFile);  // lee una cadena completa del fichero
 if (strstr(cad_leida,"replace:") ==0) // si include la palabra "replace:" ...
      {
      pos_destino = strstr( cad_leida, "-" ); // busca el separador
      cadena_original=substr(  cad_leida,  8,  pos_destino);  //  parte  anterior  al
separador
      cadena_destino=substr(  cad_leida,  pos_destino,-8);  //  parte  posterior  al
separador

      dir_base = BeginEA();                   // empieza desde el inicio

      encontrado=FindBinary(dir_base, SEARCH_NEXT, cadena_original); // busca
      while(encontrado != BADADDR) // si lo encuentra...
       {
       for  (i=1;  i<strlen(cadena_destino);  i=i+2)  //  parte  la  cadena  destino  en
trozo de 2 letras
            {
            nuevo_byte_ascii=substr(cadena_destino,i,i+1); // obtiene este trozo
            nuevo_byte_hex = xtol(nuevo_byte_ascii);   // lo convierte a Hex
            PatchByte(encontrado+i,nuevo_byte_hex);   // lo modifica en IDA
            }
       }
      }
 }
}
```

Para ejecutar este script (y todos los demas) hay que ir al menu
File->IDC File
y cargarlo.

Un poco mas comodo es usar una tecla de acceso rapido. El fichero ida.idc se ejecuta al iniciar la aplicacion, asi que podemos incluir

```
AddHotkey("Shift-P","parchea");
```

Y en cualquier momento puedo pulsar Shift-P para invocarlo.

Si te fastidia ir al menu "File->IDC File" puedes redefinir todas las teclas de acceso directo en el fichero idagui.cfg
La mayoria de los menus no tienen teclas asignadas, pero puedes hacerlo.
Por ejemplo, este menu de cargar ficheros IDC tiene en idagui.cfg la linea
"Execute" = 0 // Execute IDC file
que puedes cambiar a
"Execute" = "Shift-F3" // Execute IDC file

Eso si, ten cuidado de no asignar varias acciones a la misma tecla.
Y recuerda que puedes cambiarlas sobre la marcha en un script usando AddHotkey .

Tambien puedes ejecutar un comando cualquiera con el menu Shift-F2 .
Esto es util si no quieres hacer un programa completo.

A veces queremos parchear el fichero binario directamente en el disco.
Para ello se usan las funciones fgetc/fputc como en cualquir programa en lenguaje C.

Si esto no es suficiente puedes usar la funcion Exec para ejecutar cualquier otro programa. Uno de estos casos esta provocado por el hecho de que IDA no es capaz de recompilar codigo ensamblador para procesadores ARM.
Supongamos que tengo un programa original en ARM con una rutina que hace
MOV R0, #0x28
lo cual se escribe con los bytes
```
```

```
28 00 A0 E3
```

Pretendo cambiarlo por
MOV R0, #0x29
pero IDA no sabe como ensamblarlo. Por eso defino una combinacion de teclas
SHift-K
que invoca a un script que hace:

```
nueva_instruccion=AskStr("","Nueva instruccion?"); // pide instruccion, ej. "MOV R0,
#0x29"
my_archivo = fopen("instruccion.asm", "w");      // abre fichero de salida
writestr(my_archivo, nueva_instruccion);       // escribe el codigo assembler
fclose(my_archivo);
Exec("compilaARM.bat instruccion.asm");          // compila la instruccion

                                          // con un compilador externo

my_archivo  =  fopen("instruccion.bin",  "r");        //  mira  si  se  ha  generado
correctamente
if(my_archivo==0)                        // si no, se queja
      Message("Archivo no existe\n");
else                           // todo correcto
 {
 ea=ScreenEA();              // mira donde tiene que parchear
 for(i=0;i<4;i=i+1)          // cada instruccion ARM ocupa 4 bytes
      {
      fgetc(my_archivo);       // que extrae del fichero
      PatchByte(ea+i);         // y mete en la memoria de IDA
      }
 fclose(my_archivo);
 }
```

Otro ejemplo en el que uso la funcion Exec es cuando necesito hacer muchas
cosas sobre el fichero desensamblado. Un caso tipico es un programa que
llama a otra rutina externa, y quiero saber los registros que modifica.
Veamoslo ma claro:
Tengo una rutina que hace

```
MOV     R1, R5
BL      TTime::MicroSecondsFrom(TTime)
```

y quiero saber si TTime::MicroSecondsFrom(TTime) altera el registro R4 .
En tal caso debere guardarlo antes de llamarla si no quiero perder su valor.

Esta rutina esta en otra libreria Euser.dll , lo que me obligaria a cargarlo,
desensamblarlo con IDA, e investigar un poco.

En lugar de eso, hago un fichero IDC que
-invoca otra sesion de IDA, usando Exec
-carga Euser.dll, usando la opcion input-file
-lo desensambla, si no lo esta previamente, usando AnalyzeArea
-salta a la rutina TTime::MicroSecondsFrom() , usando LocByName
-desde el principio hasta el fina de la rutina ...
-mira si aparece la palabra R4 , usando strstr
-en tal caso asume que se esta modificando.

una excepcion a esto es cuando la primera instruccion es del tipo

```
STMFD   SP!, {R4-R6,LR}
```

que indica que la propia rutina invocada se encarga de guardar los registros
que va a modificar.
Es facil de detectar: si la primera linea en assembler de la funcion
contiene la palabras "STM" y "R4" entonces se que se guarda.

Por supuesto, esto no es del todo correcto: podria suceder que
TTime::MicroSecondsFrom() llame a otra rutina
TInt64::operator_substract(TInt64 const &)
y que esta modificara R4.

Pero lo importante es que Exec permite automatizar este proceso.

No incluyo el codigo de este programa idc porque es demasiado largo.
Si quieres ver muchos ejemplos de scripts, mira en el directorio
IDA\idc*.idc

Yo tambien he encontrado algunos ejemplos en foros dedicados a reprogramacion de moviles.
Y el sitio adecuado para preguntar es www.openrce.org , que tiene una bonita coleccion de scripts. Mira tambien la seccion de links porque apunta a varios proyectos de open-source muy interesantes.

En otro articulo comente que IDA es capaz de desensamblar programas de Symbian pero para algunas de las rutinas externas no sabe su nombre correcto.
Por ejemplo cree que la rutina
CPeriodic::Start
se llama
CListBoxData::Reserved_2

En aquel momento yo no sabia donde estaba esta definicion. Bueno, pues la he encontrado.
Con la version original de IDA (ojo, no con la version pirata que circula por ahi) se incluyen en el directorio idsutil3 un par de programillas muy simples.
El primero en el que centrare mi atencion es ZIPIDS.EXE para descomprimir los ficheros de tipo IDS :

```
ZIPIDS -u C:\Ida\ids\epoc6\arm\euser.ids
File: euser.ids  ... {1875 entries [0/0/0]}  unpacked
```

y genera

euser.idt
con este contenido:

```
;DECLARATION
;ALIGNMENT 2

; Module Name and Description
0 Name=EUSER
;---------------------------------------
1 Name=memset
2 Name=memcpy
3 Name=newL__5CBaseUi
4 Name=ASin__4MathRdRCd
5 Name=ATan__4MathRdRCd
6 Name=ATan__4MathRdRCdT2
.....
302 Name=DaysInMonth__C5TTime
.....
1644 Name=__t13CArrayFixFlat1Zii
1645 Name=_._t13CArrayFixFlat1Z4TUid
1646 Name=__t13CArrayFixFlat1Z4TUidi
;------------------EOF------------------
```

Recordar que en Symbian (y tambien en las DLL de windows) cuando un progama quiere invocar a una rutina, no la llama por el nombre, sino por un indice relativo a la libreria.
Por ejemplo, si un programa quiere llamar a la funcion TTime::DaysInMonth entonces incluye una referencia a EUSER y llama a la funcion 302
Para solucionar el problema cuando la misma rutina existe varias veces con el mismo nombre pero con distintos argumentos, es necesario incluir el tipo de variables. Por eso el nombre es "decorado" con indicadores y obtiene el nombre
DaysInMonth__C5TTime

Este proceso se llama "mangling".

Como digo, esta es la lista que esta dentro del fichero euser.ids que es el que IDA usa para nombrar las rutinas.

Pero el telefono (y el emulador) usa los de la libreria real euser.lib
que puedo investigar con

```
AR2IDT.EXE C:\Symbian\6.1\Series60\Epoc32\Release\armi\urel\euser.lib
```

que produce

```
0 Name=EUSER
1 Name=memset
2 Name=memcpy
3 Name=newL__5CBaseUi
......
```

```
302 Name=DaysInMonth__C5TTime
.....
1644 Name=__t13CArrayFixFlat1Zii
1645 Name=_._t13CArrayFixFlat1Z4TUid
1646 Name=__t13CArrayFixFlat1Z4TUidi
1647 Name=__7RRegioniP5TRecti
1648 Name=ChunkHeap__8UserHeapR6RChunkii
......
1678 Name=Start__20CActiveSchedulerWait
1679 Name=_._20CActiveSchedulerWait
```

Por si no te has dado cuenta, en la referencia incluida por defecto en IDA existen solo 1646, pero en la libreria real existen las rutinas hasta 1679. Esto quiere decir que si un programa usa alguna de esas 33 rutinas IDA no es capaz de decir su nombre.

Para resolverlo se usa primero el programa
AR2IDT euser.dll
para producir euser.idt
y luego
ZIPIDS euser.idt
para generar correctamente euser.ids

De todos modos, las instrucciones estan en el fichero readme.txt

Obviamente el fichero de texto euser.idt se puede alterar si no te gustan los nombres de las funciones. En particular yo uso los nombres "de-mangleados" que obtengo de los ficheros del SDK.
Sin extenderme mucho: el programa
dumpbin /ALL cone.lib
produce

```
.........
      Symbol name  : DaysInMonth__C5TTime
      (public: static int TTime::DaysInMonth(void))
      Ordinal      : 302
.............
```

Un simple script en Perl permite incluir el nombre real en euser.ids para hacerlo mas entendible.

Esto se puede usar en cualquier programa. Por ejemplo si quieres desensamblar un programa en Linux; aunque IDA conoce los nombres de algunas librerias, hay demasiadas como para que la incluya todas, por lo que es posible generarlas a medida que las necesites.

Hay una tecnica bastante usada en programas de Windows que hace mas dificil la vida del "analista de programas". En vez de llamar a una rutina externa, la incluyen dentro del propio programa.
Este proceso se llama in-lining.

Por ejemplo la rutina strlen() es bastante simple. Para invocarla hay que incluir su libreria, usar una referencia al indice, definir un punto de entrada, e invocar a esta rutina.
Se ahorra algo de tiempo (e incluso de espacio) si incluyes su codigo directamente en tu programa.
Probablemente se puede hacer en menos de 10 bytes.
Pero claro, el que desensambla el programa se encuentra con una rutina que complica (aunque solo ligeramente) la interpretacion del desensamblado.

IDA es capaz de encontrar tales rutinas. Para ello usa algo llamado firma (signature), que no es mas que los primeros bytes de la rutina original.
Veamos el fichero flair/startup/pe_vc6.pat
que contiene las firmas generadas por el compilador VisualC 6.0
Una de las lineas dice:

```
558BEC6AFF68.....  __except_handler
```

lo cual significa que la rutina __except_handler comienza por estos bytes.
Cuando IDA encuentra estos bytes, sugiere renombrar la rutina encontrada como "__except_handler_XXX"

Para automatizar el proceso se usa
plb
que a partir de una libreria, genera un fichero de muestras PAT

con este, genera un fichero de firmas SIG

Un segundo uso es agrupar secuencias de bytes.
Suponer que te das cuenta de que a menudo aparecen las instrucciones
LDR R0, [R3]
ADD R0, #1
STR R0, [R3]

que en ARM se escribe como

00 00 93 E5
01 00 80 E2
00 00 83 E5

Esta instruccion es lo mismo que la pseudo-instruccion

INC [R3]

Asi que creas un PAT con la linea

000093E5010080E2000083E5.... INC [R3]

?ves por donde van los tiros? Se puede hacer el programa mas breve, con lo cual es mas facil de entender.
Esto es una tecnica usada en re-compiladores que generan codigo C a partir del listado en ensamblador.
Para ver un desarrollo de esta idea recomiendo el excelente programa
http://desquirr.sourceforge.net
que incluye una explicacion bastante detallada.

Por supuesto, el documento de obligada lectura es el decompilador DCC de Cristina Cifuentes y su tesis doctoral, en
http://www.it.uq.edu.au

En general esto de las muestras y las firmas es util cuando quieres sacarle mas partido a IDA y conoces el compilador usado para construir el programa que pretendes desensamblar.

Otra de las incomodidades de programas desensamblados es el uso de sentencias "switch"/"case".
En lenguaje C es comun usar algo asi:

```
switch(origen)
 {
 case 0: destino=8; break;
 case 1: destino=88; break;
 case 2: destino=888; break;
 case 3: destino=8888; break;
 }
```

El compilador traduce este codigo en:

```
LDR  R0, [origen]    // lee el valor   origen
LDR  R1, =tabla     // crea una tabla de 4 posibilidades
MOV  R0, R0*4      // cada valor de la tabla contiene un puntero a
ADD  R0, R1       // una subrutina.
BX   [R0]        // lee puntero[origen] , y salta a donde le dicen
tabla:
 DCD rutina0    // puntero a la subrutina de "case 0"
 DCD rutina1
 DCD rutina2
 DCD rutina3

rutina0:    // autentica rutina que ejecuta "case 0"
 MOV R0, 8
 B break
rutina1:
 MOV R0, 88
 B break
rutina2:
 MOV R0, 888
 B break
rutina3:
 MOV R0, 8888
 B break
```

```
break:
 STR R0, [destino]
```

Visto de otro modo: hace
`jmp rutina[origen*4]`

Desde el punto de vista del compilador esto tiene perfecto sentido. Lo malo es que IDA lo unico que ve es una tabla con valores, y cree que estamos en un bloque de datos.
Si es capaz de interpretar correctamente el codigo rutina0, ... rutina3 , pero desconoce desde donde se referencian.
Le podemos ayudar haciendo un script que a partir de una direccion busque si hay una tabla de saltos en las siguientes 8 instrucciones:

```
for (pos = inicio; pos < inicio+8*4; pos = pos + 4) // instruccion de 4 bytes
{
 ins=GetManualInsn(pos);
 if(strstr(ins,"DCD")) // hay un puntero?
      {                   // veamos si apunta a una rutina
                                                              //   la   instruccion   es
DCD xxxxxx
      valor_apuntado = Dword(substr(ins,5,-1)); // extraer  xxxxxx
      if( val>pos && val<(pos+80))  // si salta a alguna rutina cercana ...
       {
       result=MakeCode(valor_apuntado); // miro si es un trozo de codigo
       if(result>0)                     // si se ha podido convertir en codigo ...
            {
            AddCodeXref(pos,val,fl_F); // entonces marco la referencia
            MakeName(  pos,  "salto_a_"  +  ltoa(val,16)  )  ;  //  y  le  doy  un  nombre
bonito
            MakeComm ( pos, "salto " + ltoa(val,16) );   // y un comentario
            }
       }
      }
}
```

Por otro lado tambien podria usar la tecnica de las firmas. El bloque

```
LDR  R0, [origen]
LDR  R1, =tabla
MOV  R0, R0*4
ADD  R0, R1
BX   [R0]
```

es sintoma de que hay una aceso indexado, y aparece siempre relacionado con una tabla de saltos.
Asi que hago un script que en cada puntero a rutina, lo comenta con:
`DCD rutinaX ; case X`
que me sirve para saber cual valor saltara a esta rutina.

Bueno, esto ayuda a IDA a entender mejor el listado, pero quizas podria facilitarme la tarea A MI.
Para que esto sea posible, debo recuperar la estructura "switch"/"case" inicial. Durante este proceso puedo tengo que transformar el codigo desensamblado en algo parecido a lenguaje C, pero lo malo es que luego no podria ensamblarlo de nuevo con un ensamblador: necesitaria un compilador. El asunto es lo suficientemente complejo para explicarlo aqui, por lo que os emplazo a un proximo numero de SET.

Un tema que he investigado poco pero que promete mucho es el de los reconocedores de archivos.
En vez de investigar un programa a veces necesitas analizar los datos que ha producido.
Uno de los scripts incluidos en IDA sirve para desmenuzar ficheros de tipo AIF que contienen los recursos (iconos, strings, menus) que usa una aplicacion Symbian.
Cuando IDA reconoce que el fichero es de este tipo, procesa toda la estructura y la presenta de manera comprensible.
La manera de programar nuevos reconocedores es en lenguaje C leyendo byte por byte en fichero, y definiendo estructuras conteniendo esos bytes.
En general el tema depende del tipo de fichero y debo decir que a mi me

resulta mas facil hacerme un programa en C sin usar IDA en absoluto.

Quizas en un proximo articulo, aunque me parece que es un tema que
interesara a pocos de vosotros (asumiendo que lo que yo cuento interesa
a alguien, que a veces lo dudo)

Otros temas demasiados avanzados para contar en esta introduccion son:
-modulos, para definir otros procesadores y las instrucciones que usan.
 Aunque IDA soporta 35 familias de procesadores, quien sabe lo que los
 usuarios necesitaran en el futuro.
-plugins, para compilar funcionalidad que le falta a IDA, por ejemplo para
 relacionarlo con un debugger o un emulador y analizar el codigo sobre la
 marcha.
-wingraph, que es la aplicacion usada para mostrar las referencias entre
 rutinas en forma de arbol. Tiene algunos defectos que me gustaria
 modificar, por ejemplo poder colapsar algunas ramas inutiles. El fuente
 esta disponible y el formato usado (gdl-VCG) es facil de entender.
-sustitucion de grupos de instrucciones mediante macros
-regeneracion de codigo para posterior recompilacion

Lo que debe quedar claro es que el programa puede parecer caro, pero es
porque lo vale.

Para aprender tecnicas de desensamblado pueden ser utiles los libros de
la editorial "No Starch Press", en particular:
Hacking: The Art of Exploitation
Hacker Disassembling Uncovered, by Kris Kaspersky

Reversing: Secrets of Reverse Engineering, by Eldad Eilam
Reverse Compilation Techniques, by Cristina Cifuentes

Por ultimo me gustaria incluir un comentario que he leido en uno de estos
libros de desensamblado:
"It will become more understandable after studying the disassembled listing."

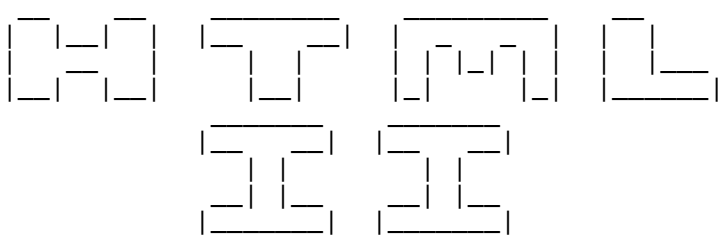
Y mi propio comentario: me extranyaria mucho. En general el codigo
desensamblado es el ultimo recurso yo que suelo usar.

EOF

-[0x0E]--
-[HTML II]---
-[by elotro]---SET-33---

Cocinando con elotro y SET ezine

Plato principal :



(o como volverse loco haciendo paginas web)

Segundo plato : PROGRAMACION DE SERVIDORES WEB (CGI)

Postre : - Formato MIME
- util.c
- Caracteres especiales y simbolos
- Webs recomendadas

Sobremesa : Aclaraciones y agradecimientos

El menu del plato principal incluye :

6.1.3 <DD>
6.2 Regular
6.2.1 <LI>
6.2.2 <UL>
6.2.3 Anidamiento de listas (no ordenadas)
6.2.4 <OL>
6.2.5 Anidamiento de listas (ordenadas)
6.3 Listas anidadas
7. Estilos
7.1 Estilo del caracter
7.1.1 <B>
7.1.2 <I>
7.1.3 <U>
7.1.4 <TT>
7.2 Estilo logico y de parrafo
7.2.1 <CITE>
7.2.2 <CODE>
7.2.3 <DFN>
7.2.4 <EM>
7.2.5 <KBD>
7.2.6 <SAMP>
7.2.7 <STRONG>
7.2.8 <VAR>
7.2.9 <BLOCKQUOTE>
7.3 Extensiones
7.3.1 <FONT>
7.3.2 Atributo Size
7.3.3 Atributo Color
8. Enlances
8.1 Marcadores o anclas
8.1.1 <A>
8.1.2 HREF, enlance externo
8.1.3 HREF, enlance interno
8.1.4 NAME, marcador de llegada
8.2 Recursos de un enlance
8.3 Tipos de URL
8.3.1 FTP
8.3.1 NEWS
8.3.1 TELNET
8.3.1 GOPHER
8.3.1 MAILTO
9. Incluyendo imagenes
9.1 Imagenes
9.1.1 <IMG>
9.1.2 Alineando imagenes
9.2 Imagenes en navegadores no graficos
9.3 Imagenes externas
9.4 Imagenes como hipervinculos
10. Fondo de la pagina, atributos extras al texto
10.1 Fondo
10.1.1 BGCOLOR
10.2.2 BACKGROUND
10.2 Color del texto
10.2.1 TEXT
10.2.2 LINK, VLINK, ALINK
11. Tablas
11.1 <TABLE>
11.2 <TR>
11.3 <TD>
11.4 <TH>
11.5 <CAPTION>
12. Paneles o marcos
12.1 Introduccion
12.2 Creacion de paneles
12.2.1 <FRAMESET>
12.2.1 <FRAME>
12.2.1 <NOFRAMES>
12.3 Utilizacion de paneles
12.3 Target

Y hasta aqui llegaremos por hoy, espero que no pasen hambre hasta Set 34

PLATO PRINCIPAL ----------.

Lenguaje HTML <----------'

1. Introduccion --.
<----------'

1.1 HTML 2 y 3

El nivel de HTML que se podria considerar usado en la actualidad, es el llamado HTML 3. Este curso se basa, ironicamente, en HTML 2.
Porque se basa en HTML 2: El nivel 2 del HTML es aceptado por practicamente todos los navegadores o browsers que existen en la actualidad, incluyendo aquellos que ya estan anticuados, tales como el NCSA Mosaic (que es muy buen navegador, pero muy antiguo) y otros navegadores basados en el modo grafico de DOS, que tuvieron muy poca difusion. Uno de ellos es el Web2Dos , que debe estar en algun hueco de internet.

1.2 Vale la pena aprender HTML ?

El HTML escrito a mano casi no es utilizado para la realizacion de paginas web, existiendo potentes editores WYSIWYG, como el Hotmetal Pro o el Microsoft Frontpage, que facilitan mucho la realizacion de documentos HTML.
Si tu eres de esos que andan cambiando paginas web, introduciendose en sistemas ajenos; entonces no vendra mal que aprendas algo de HTML, para poder realizar el cambio lo mas rapido y discreto posible.
Aparte, el saber no ocupa lugar (dependiendo del tamanio de tu HD) :)

Este manual basico de HTML abarca los mas basico del HTML, sin nada de zonas DIV, ni javascript, ni applets java.

2. Hipertexto --.
<----------'

Para entender rapidamente el hipertexto, se podria decir que es un sistema que vincula informacion mediante enlances.
Realmente, se deberia decir 'hipermedia' porque con la inclusion de imagenes, video, sonido, javascript; el mundo de la web ha cambiado totalmente.

2.1.1 Ejemplos de hipertexto : Libro

En un libro generalmente se accede por la primera pagina (home), siguiendo la lectura en el orden de las paginas. Las bifurcaciones en la lectura son introducidas por las notas al pie de pagina o por las aclaraciones del autor. El documento no se basta por si mismo, posiblemente habra que consultar otras obras para comprenderlo mejor o para ampliar los conocimientos.

2.1.2 Ejemplos de hipertexto : Documento tecnico

Tomemos por ejemplo el manual de un electrodomestico. Generalmente consta de un indice de contenidos, y el lector solo accede a los conceptos que desea leer, sin tener que pasar por los otros.
El acceso se realiza mediante el indice, la tabla de capitulos o el indice analitico. De esta forma se 'navega' en el texto.

2.2 Nocion de hipertexto

En el ejemplo del documento tecnico se observa la aparicion de 'enlances' hacia otras partes del texto, que no son contiguas. En un documento escrito, los enlances se ejecutan mediante la busqueda del tema en el indice y la busqueda de la pagina correspondiente.
En un docuemnto electronico, los enlances se ejecutan seleccionandolos y el enlance nos llevara a la pagina de destino.
El hipertexto no se encuentra solamente en los documentos HTML, pero este es el formato en el que mas se reconoce y el que mas evoluciona.
Con aplicaciones como Acrobat podemos realizar hipertexto en formato NO HTML, sino que en el caso del Acrobat, el formato es .pdf.
Otro ejemplo de hipertexto es el caso de los archivos de ayuda de Windows.
La rapidez del desplazamiento en el hipertexto (un clic del mouse), la facilidad de volver hacia atras, permiten una nueva concepcion del documento:

Paginas cortas, contenido esencial, atraccion hacia el lector; son los ingredientes basicos para realizar un buen hipertexto.

2.2.1 Escribiendo hipertexto

Para escribir hipertexto, podemos contar con Notepad en un entorno Windows; emacs o vi en un entorno *nix. Existe aplicaciones que permiten realizar un verdadero WYSIWYG, como las ya nombradas Hotmetal Pro y Microsoft Frontpage.
Existen conversores que traducen un texto en formato .txt, .rtf, .doc, .wri a documentos HTML.
[es muy probable que te vuelvas loco al escribir html]

2.2.2 Multimedia en la WWW

La WWW permite la inclusion de video, imagenes, sonido, y millones de posibilidades extras. Junto con estos elementos y la introduccion del javascript, la WWW es un sinfin de posibilidades.

RESUMEN : El hipertexto se basa en enlances:

Un enlance puede ser cualquier elemento en pantalla que este disponible con el mouse. Su objetivo puede ser cualquiera.

Tipos de enlance :
- Interno : Dentro del mismo documento y apunta a un marcador.
- Externo : A otro documento.
- Ejecutable : Se dirige a un programa o script que trata los datos introducidos, como una respuesta interactiva.

3. HTML --.
<----------'

3.1 Que es HTML ?

HTML (Hyper Text Markup Language, Lenguaje de marcas de hipertexto) es un lenguaje simple que se utiliza para crear documentos para acceder desde la WWW. No es un lenguje de descripcion de pagina como PostScript, sino que desde el HTML (junto con javascript) se pueden controlar practicamente todos los aspectos de la pagina. HTML es una implementacion simple de SGML (Standard Generalized Markup Language, Lenguaje generalizado estandar de marcas).

3.2 Donde encontrar HTML ?

El marco de ejecucion de la WWW y del lenguaje HTML es un modelo cliente--servidor, a traves de una red informatica tal como lo es Internet.

Cliente : Es una pc con un programa de consulta (navegador o browser), que se comunica con la red a traves de un protocolo llamado HTTP (HyperText Transfer Protocol) que proporciona la informacion estructurada en codigo HTML. El cliente es el que se encarga de la ejecucion de este codigo y el que produce la visualizacion del docuemnto.
Ejemplos de cliente son Netscape Navigator y Mozilla.

Servidor : Es un programa que atiende las peticiones del cliente. Verifica la validez del cliente y se asegura que la solicitud proviene de un cliente autorizado a acceder al docuemento. Si estas condiciones se cumplen, el servidor envia el documento al cliente. En caso de que la peticion del cliente haya sido un enlance ejecutable, el servidor enviara el documento HTML correspondiente o el archivo al que apunta el enlance. Aunque existen servidores para maquinas Macintosh, estos casi no se utilizan. La preferencia es la de servidores en maquinas RISC o PC con un SO Unix, y Windows NT en una proporcion menor. La multitarea y multiproceso del servidor es lo que permite un gran numero de consultas simultaneas.
Ejemplos de servidores son Apache o NCSA. En las maquinas Unix, el proceso se denomina 'HTTPD' (HyperText Transfer Protocol Daemon).

[un demonio, vamos!]

3.3 El URL

La interconexion de los documentos se realiza mediante un medio llamado URL (Uniform Resource Locator). El URL se compone de :

- El protocolo de intercambio entre el cliente y el servidor. Para acceder a docuemto HTML normalmente se utiliza el protocolo HTTP, pero existen otros protocolos, tales como FTP y otros menos utilizados.

- La direccion Internet del servidor. Esta direccion es unica en toda la red. Es la direccion TCP/IP de la maquina. Tiene la forma de una serie de numeros, tales como 134.158.69.110. Como el numero es dificil de memorizar, se utiliza el metodo DNS que resuelve la relacion entre la direccion IP del servidor y el nombre simbolico de la maquina.

[para dns no estoy yo, ver ezine eko magazine n§1]
[la 2 tiene info sobre como atacar estos servidores]

- El arbol de directorios que conduce al documento

- El nombre del documento

Existen otras opciones como el puerto de conexion, la informacion de autentificacion (user y pass), o los argumentos que se pasan a un programa de tratamiento de datos.

La sintaxis habitual es:

```
protocolo://nombre_servidor[:puerto]/directorio/subdirectorio/nombre_documento

Ejemplos: http://www.set-ezine.org/archivo.php
          ftp://ftp.microsoft.com/images/billysucks.jpg ;)
```

Si el servidor necesita autentificar el usuario, la sintaxis es:

```
protocolo://username;password@nombre_servidor[:puerto]/directorio/documento


Ejemplos:
     http://billy;gates@www.microsoft.com/images/yousuck.jpg
     http://john;smith@www.google.com:80/npi?query+billy+sucks+yes+sir
```

3.4 Donde se almacena el HTML ?

En el disco de la maquina que oficia de servidor, existe un directorio destinado a almacenar los ficheros HTML y todos los demas ficheros relacionados con la pagina web.
El directorio es (maquina unix): /usr/local/etc/htdocs

```
                                |_____________|
                                      ^ ^ ^
                                Invisible al cliente web
```

3.5 Proteccion de los documentos

El acceso a los documentos y la proteccion de los mismos se puede reducir mediante un sistema de proteccion desarrollado por el Web. No debe confundirse con la proteccion de Unix, que permite la lectura y eventualmente la ejecucion del fichero. En el sentido Web, el sistema nos permitira autorizar la lectura de un documento.

- A los lectores registrados que cuenten con un username y password. No se trata de tener una cuenta en la maquina servidor, sino de aquellos que esten autorizados segun el demonio HTTPD.

- A los lectores que acceden desde un ambito particular (sentido TCP/IP)

- A los lectores que accedan desde una maquina particular. Las protecciones aplicadas son muy similares a las del primer punto.

4. Escribiendo HTML --.
<----------------------'

4.1 Marcas

Como ya se ha dicho, el HTML describe la estructura de un documento para el WWW. Esto se obtiene encerrando cada una de las distintas estructuras de texto entre una serie de marcas, una al comienzo y otra al final. Las marcas son invisibles en la presentacion del navegador.

En HTML, las marcas se delimitan de la siguiente manera:

texto normal <MARCA>texto afectado por la marca</MARCA>
Observa el caracter '/' en la marca de fin.

Si se quieren ussar los caracteres <> en un texto normal, estos se reemplazan por : < = < > = >

Si escribimos : 'Los signos > y < delimitan las marcas'
en un navegador se vera de esta manera:
'Los signos < y > delimitan las marcas'

El texto en el interior de las marcas puede estar en mayusculas o minusculas, produciendo el mismo efecto. <a> y <A> son iguales.

4.2 Acentuacion

Para transmitir los documentos HTML, estos se transmiten como archivos en ASCII de 7 bits (caracteres 0/127, SET compatible). Los 7 bits de menor peso son los que indican el caracter. El octavo bit se utiliza como control de paridad para validar la exactitud de la transmision.
Como ya debes saber, estos caracteres no incluyen la enye(¤) y los acentos( ). En el postre tienes una tabla de los caracteres acentuados y los codigos que se deben utilizar para representarlos.

4.3 Estructura del HTML

Los docuemntos HTML aplican las siguientes marcas :

<HTML> : Indica el comienzo del codigo HTML
Ej:
```
<HTML>
  cuerpo del documento
</HTML>
```

<HEAD> : El titulo del documento se incluye aqui.
Ej:
```
<HTML>
 <HEAD>
  Articulo de HTML
 </HEAD>
  cuerpo del documento
</HTML>
```

<TITLE> : Es el texto que saldra en zona de titulo del navegador

<BODY> : Aqui reside el resto del documento

<ADDRESS> : Es un bloque destinado a dar informacion sobre el autor del documento. Los navegadores generalmente muestran este texto en cursiva.

<!--Comentarios> : Cualquie texto que comienze por <!-- y termine por --> sera tomado como un comentario y no sera mostrado por el navegador.
Es equivalente a /* */ en C, // en C++ y ' en Basic.

4.4 Atributos de una marca

En el entrono HTML las marcas pueden tener diferentes atributos. Si el atributo es numerico, se escribe tal cual. Si es alfanumerico, se escribe entre "comillas".

<MARCA atributo_1=numerico atributo_2="alfanumerico">
Ej: <A HREF="home/etc/documento.html">

4.5 Esqueleto de un documento HTML

Con los pocos elementos que hemos definido, podemos hacernos una idea de como se organiza un documento HTML. Puedes ver este trozo de texto con tu browser favorito, para visualizar el comportamiento del lenguaje HTML.

```
<HTML>
  <!-- Esqueleto HTML -->
  <HEAD>
```

```
    <TITLE> SET Ezine </TITLE>
  <HEAD>
  <BODY>
    SET Ezine presenta : HTML o como volverse loco haciendo paginas web.
    <ADDRESS>
      elotro - elotro.ar@gmail.com
    </ADDRESS>
  </BODY>
</HTML>
```

4.6 Probando documentos HTML

Una vez realizado el codigo HTML, debes guardar el archivo con extension HTML (.html)
Para probar los documento HTML, solamente tienes que abrirlos con tu browser preferido. Cuando realizes cambios, selecciona la opcion de acutalizar o reload, y podras ver el resultado de las modificaciones.

5. Divisiones --.
<----------------'

Generalmente, los textos informativos se dividen siguiendo una cierta jerarquia, como la del indice de este texto (1.1, 1.2, 1.2.1). Para que esta organizacion sea mas agradable y mas util para el lector, se puede aumentar o disminuir el tamanio de los caracteres presentados en la pantalla.

5.1 <Hn>

Esta marca es la que define el tamanio de los caracteres, donde 'n' varia de 1 a 6 (<h1> -> <h6>). Los caracteres mas grandes tienen el valor 1 y los mas pequenios tienen el valor 6. Generalmente, el texto entre estas marcas se trata en estilo negrita. La marca de principio y la marca de fin generan automaticamente un retorno de carro (ascii 13+10)

[o era ascii 10+13.., bah es un CRLF] [Carriage Return, line feed]

5.2

Cuando un browser ejecuta codigo HTML, ignora los saltos de linea que tiene el fichero fuente. Para insertar saltos de linea, se debe utilizar la marca
. Esta marca tiene la particularidad de ser una marca vacia, o sea que no necesita la inclusion de </BR>

5.3 <P>

La marca <P> provoca un salto al parrafo siguiente. Aunque se considera una marca vacia, es perfectamente valido terminar un parrafo con </P>

5.4 <HR>

Esta marca produce una linea horizontal en la ventana del browser. Es un recurso grafico muy atractivo.

[parece un cuaderno a rayas en el monitor]

5.5 <PRE>

Esta el nombre de esta marca proviene de preformated(preformateado). Respeta la paginacion precisa de un texto o de una porcion. Presenta el texto con una fuente monoespaciada como Courier.

Ej:
```
<html>
<head><title>Tabla Preformateada</title></head>
<body>
<h2> Ejemplo de tabla en texto preformateado (&lt;pre&gt;)</h2>
<pre>
+---------+----------+-----------+----------+---------+
|  Lunes  |  Martes  | Miercoles |  Jueves  | Viernes |
```

```
+---------+---------+----------+---------+---------+
|         |         |          |         |         |
|         |         |          |         |         |
|         |         |          |         |         |
+---------+---------+----------+---------+---------+
</pre>
</body>
</html>
```

El atributo width permite fijar en que numero de columna se fijara el texto preformateado. Por ejemplo, <pre width=40> situara el cursor en la columna 40.

6. Listas --.
<------------'

En HTML se pueden definir dos tipos de listas : Descirptivas y regulares.

6.1 Lista Descriptiva

* Idea del objeto
  Descripcion de objeto y bla bla bla bla
  bla bla bla bla bla bla bla bla bla bla
  bla bla bla bla bla bla bla bla bla bla

* Otro objeto
  Mas descripcion y mas bla bla bla bla
  bla bla bla bla bla bla bla bla bla bla
  bla bla bla bla bla bla bla bla bla bla

6.1.1 <DL>

La marca <dl> abre una lista descriptiva. Define el inicio de la lista y abre el espacio para incluir dos marcas mas : <dt> y <dd>
El atributo compact muestra en la misma linea el objeto y la descripcion. <dl compact>.
Esta marca debe cerrarse con </dl>

6.1.2 <DT>

Nombra un objeto de la lista. Su contenido no debe sobrepasar una linea. El estilo de la letra se puede ajustar a gusto del autor con todas las marcas que se consideren necesarias.

6.1.3 <DD>

Corresponde a la definicion del objeto. El tamanio de esta zona no esta limitado y cada linea se ubicara con una sangria en la izquierda.

Ejemplo de una lista descriptiva :

```
<html>
<head><title>Listas &lt;dl&gt;</title></head>
<body>
  <h2>Glosario(Ejemplo de lista de tipo definici&oacute;n(&lt;dl&gt;))</h2>
  <dl compact>
  <dt>ASCII<dd>American Standard Code for Information Interchange
  <dt>SET<dd>Saqueadores Edicion T&eacute;cnica
  <dt>elotro<dd>Referente a un idiota de sudamerica
  </dl>
</body>
</html>
```

6.2 Lista Regular

* Caballo
* Mula
* Burro
* Menem

1. Chimpance
2. Gorila
3. Mono
4. ViejaC

6.2.1 <LI>

Esta marca es vacia (no precisa). Precede a cada uno de los objetos de la lista. La sintaxis general de las listas regulares es :

```
<marca de apertura>
<li>Objeto
<li>...
<marca de cierre>
```

6.2.2 <UL>

Genera listas no numeradas. Cada uno de objetos va precedido por un simbolo [se dice vinieta segun Billy G] segun el nivel de anidacion que tenga.

Ejemplo de lista regular:

```
<html>
  <head><title>Listas regulares &lt;ul&gt;</title></head>
    <body>
     <h2>Ejemplo de lista no ordenada (&lt;ul&gt;)</h2>
      <ul>
        <li>Ficheros HTML
        <li>Ficheros de imagen
        <li>Ficheros de sonido
        <li>Ficheros de v&iacute;deo
        <li>Ficheros de prueba
        <li>Ficheros de ejemplo
        <li>Ficheros basura, como este.
      </ul>
    </body>
  </html>
```

6.2.3 Anidamiento de listas (no ordenadas)

Para generar listas no ordenadas anidadas, solamente hay que incluir otra marca <ul> dentro de una zona <ul> que no haya sido cerrada con . Por supuesto, esta marca tambien llevara su correspondiente marca de cierre

```
<html>
  <head><title>Listas no ordenadas</title></head>
  <body>
  <h2>Ejemplo de lista no ordenada anidada (&lt;ul&gt;)</h2>
    <ul>
    <li>Ficheros HTML
      <ul>
      <li>Ejemplos
        <ul>
        <li>Listas
        <li>Tablas
        <li>Estilos
          <ul>
          <li>Cita
          <li>Texto en enfasis
          </ul>
        </ul>
      </ul>
      <li>En curso    (como este)
      <li>Preescritos (como este)
    </ul>
  </body>
</html>
```

6.2.4 <OL>

Esta marca se usa para generar listas numeradas u ordenadas. Observa el ejemplo :

```
<html>
  <head><title>Listas regulares &lt;ul&gt;</title></head>
    <body>
     <h2>Ejemplo de lista ordenada (&lt;ol&gt;)</h2>
```

```
      <ol>
        <li>Ficheros HTML
        <li>Ficheros de imagen
        <li>Ficheros de sonido
        <li>Ficheros de v&iacute;deo
        <li>Ficheros de prueba
        <li>Ficheros de ejemplo
        <li>Ficheros basura, como este.
      </ol>
    </body>
  </html>
```

6.2.5 Anidamiento de listas (ordenadas)

```
<html>
  <head><title>Listas ordenadas</title></head>
  <body>
  <h2>Ejemplo de lista ordenada anidada (&lt;ol&gt;)</h2>
    <ol>
    <li>Ficheros HTML
      <ol>
      <li>Ejemplos
        <ol>
        <li>Listas
        <li>Tablas
        <li>Estilos
          <ol>
          <li>Cita
          <li>Texto en enfasis
          </ol>
        </ol>
      </ol>
      <li>En curso    (como este)
      <li>Preescritos (como este)
    </ol>
  </body>
</html>
```

6.3 Listas anidadas

Este es un buen ejemplo de la creacion de listas ordenadas y no ordenadas.

```
<html>
  <head><title>Ejemplo de anidamiento de listas</title></head>
  <body>
  <h2>Lista anidada</h2>
    <ul>
    <li>Ficheros HTML
      <ol>
      <li>Ficheros de prueba
        <ul>
        <li>prueba1.html
        <li>shit.html
        <li>index.html
        </ul>
      <li>Ficheros del servidor
      <li>Ficheros de ejemplos
      </ol>
    <li>Ficheros de sonido
    <li>Otros ficheros
    </ul>
  </body>
</html>
```

7. Estilos --.
<----------'

Se llama estilo al tipo de letra de los caracteres, al tamanio de la fuente, al color del caracter y a todas las demas caracteristicas que se puedan aplicar al caracter del documento HTML. [segun Billy G, esto se llama formato]

7.1 Estilo del caracter

Existen diversas marcas a la hora de elegir el estilo del caracter :

7.1.1 <B>

Produce un texto en negrita. Esta marca no es vacia, asi que se tiene que finalizar el texto con .

7.1.2 <I>

Produce un texto en cursiva. Tampoco es una marca vacia.

7.1.3 <U>

Produce un texto subrayado. Esta tampoco es vacia.

7.1.4 <TT>

Produce caracteres como los de una maquina de escribir (Courier), o sea, con fuente monoespaciada.

Ejemplo de uso de los estilos :

```
<html>
  <head><title>Estilos de texto</title></head>
  <body>
  El tratamiento de textos <b>WYSIWYG</b><i>(<b>W</b>hat <b>Y</b>ou <b>S</b>
  ee <b>I</b>s <b>W</>hat <b>Y</b>ou <b>G</b>et)</i> es un tratamiento de
  texto en lo vos ves <tt>es lo que obtenes.</tt>
  </body>
</html>
```

7.2 Estilo logico y de parrafo

Estas marcas de estilo son mas faciles de usar y mas intuitivas para el que escribe HTML. Como ya te debes haber dado cuenta, se llama estilo logico porque las marcas 'describen' el texto en su interior.

7.2.1 <CITE>

Se usa para una 'cita' de una obra o una referencia a otro docuemento. La mayoria de los navegadores presentan caracteres en cursiva para este texto.

```
  <html>
    <head><title>Estlo logico y de parrafo : &lt;CITE&gt;</title></head>
    <body>
    Texto fuera de la marca <cite>Texto dentro de la marca</cite>
    </body>
  </html>
```

7.2.2 <CODE>

Casi siempre se usa para representar codigo fuente de algun programa. Se usan caracteres monoespaciados, como fuente Courier.

```
  <html>
    <head><title>Estlo logico y de parrafo : &lt;CODE&gt;</title></head>
    <body>
    Assembler al dope <br>
    <code>
        MOV AX,BX
        INC BX
        MOV BX,AX
        DEC AX
        NOP
        NOP
        INT 20H
    </code>
    </body>
  </html>
```

7.2.3 <DFN>

Se usa para DeFiNir un texto. Segun el browser, se usan caracteres en cursiva o en negrita cursiva.

```
<html>
  <head><title>Estlo logico y de parrafo : &lt;DFN&gt;</title></head>
  <body>
  SET EZINE <br>
  <dfn>
  Publicacion electronica de origen espa&ntilde;ol, dedicada al 'under'
  de la escena informatica.
  </dfn>
  </body>
</html>
```

7.2.4 <EM>

Resalta un texto. Normalmente se usan caracteres en cursiva.

```
<html>
  <head><title>Estlo logico y de parrafo : &lt;EM&gt;</title></head>
  <body>
  Texto al dope <br>
  <em>
  Texto resaltado
  </em>
  </body>
</html>
```

7.2.5 <KBD>

Ilustra un ejemplo de entrada de texto por teclado. Usa caracteres monoespaciados. (KBD = KeyBoarD)

```
<html>
  <head><title>Estlo logico y de parrafo : &lt;KBD&gt;</title></head>
  <body>
  User : <kbd>elotro</kbd><br>
  Pass : <kbd>********</kbd>
  </body>
</html>
```

7.2.6 <SAMP>

Se usa para representar un ejemplo de un texto (ejemplo = SAMPle en ingles)

```
<html>
  <head><title>Estlo logico y de parrafo : &lt;SAMP&gt;</title></head>
  <body>
  Uso de &lt;SAMP&gt;<br>
  <samp>
  Este es un ejemplo del uso de la marca &lt;samp&gt;
  </samp>
  </body>
</html>
```

7.2.7 <STRONG>

Para destacar fuertemente un texto. Generalmente se usan caracteres en negrita.

```
<html>
  <head><title>Estlo logico y de parrafo : &lt;STRONG&gt;</title></head>
  <body>
  Texto sin destacar <br>
  <strong>
  Texto destacado
  </strong>
  </body>
</html>
```

7.2.8 <VAR>

Para representar una variable, por ejemplo de un codigo basic.

```
<html>
  <head><title>Estlo logico y de parrafo : &lt;VAR&gt;</title></head>
  <body>
  Variables enteras y de cadena :<br>
  <var>
  numero% = 1
  DIM a$(1 to 10)
  </var>
  </body>
</html>
```

7.2.9 <BLOCKQUOTE>

Es una combinacion de las marcas <pre> y <address>. Simboliza una direccion o una firma.

```
<html>
 <head><title>Estlo logico y de parrafo : &lt;BLOCKQUOTE&gt;</title></head>
    <body>
    Sr Lector :<br>
    Le comunicamos que este documento es un ejemplo del uso del lenguaje
    HTML, especificamente de la marca &lt;BLOCKQUOTE&gt;.<br>
    Saluda atte.
      <blockquote>
        elotro - elotro.ar@gmail.com
      </blockquote>
    </body>
</html>
```

7.3 Extensiones

Dentro del lenguaje HTML 2, Netscape propone extensiones al lenguaje, permitiendo ajustar el tamanio y el color de los caracteres.
[la mayoria de los navegadores actuales tambien permiten esto]

7.3.1 <FONT>

Esta marca es la que actua sobre los bloques de texto. Tiene 2 atributos :

7.3.2 Atributo Size

Permite regular la altura de los caracteres, mediante una escala de 1 a 7.

7.3.3 Atributo Color

Especifica el color de los caracteres mediante el modelo RGB :
Red Green Blue
Rojo Verde Azul Ej: Rojo = #FF0000
                    Blanco = #FFFFFF

```
<html>
 <head><title>Extensiones de Netscape</title></head>
 <body>
 La mayoria de los navegadores actuales tambien interpretan estas marcas<br>
 <font  size=1>1<font  size=2>2<font  size=3>3<font  size=4>4
 <font  size=5>5<font  size=6>6<font  size=7>7
 <font  size=2  color=#00FFFF>a
 <font  size=4  color=#FF0000>b
</body>
</html>
```

8. Enlances --;
<--------------'

Los capitulos de arriba se dedican a explicar la preparacion de un documento HTML en lo que se refiere a la estructura y la presentacion del texto.
Este capitulo se dedica exclusivamente a los enlances hipertextuales de

los documentos HTML.

8.1 Marcadores o anclas

Para comenzar a entender el hipertexto, primero vamos a aclarar unos conceptos: [hay que estudiar un poco tambien :)]

- Ancla de partida : Es la zona sobre la cual se hara clic para acceder al enlance. Puede ser texto, imagenes.

- Ancla de llegada : Es la zona a la que apunta el ancla de partida. Puede ser cualquier tipo de fichero.

8.1.1 <A>

Esta marca no se utilizara sola JAMAS.

8.1.2 HREF, enlance externo

Se trata de crear un enlace hacia otra pagina de la web, que puede estar en un servidor distinto.

```
<A  HREF="url de destino">zona_activable_con_atributos_visuales</A>

<A  HREF="http://www.set-ezine.org">SET Ezine</A>
```

8.1.3 HREF, enlance interno

El funcionamiento de esta marca es similar a la anterior, pero el destino del hipervinculo es una etiqueta o marcador dentro del propio documento. Sirve para navegar dentro de documentos extensos.

```
<A  HREF="#etiqueta">zona_activable_con_atributos_visuales</A>
```

Tambien se puede combinar con la marca hacia un documento externo :

```
<A  HREF="url_destino#etiqueta">zona_activable_con_atributos_visuales</A>
```

8.1.4 NAME, marcador de llegada

Se usa para definir un ancla de llegada que puede ser usado con la marca anterior.

```
<A NAME="etiqueta">zona_NO_activable_SIN_atributos_visuales</A>
```

8.2 Recursos de un enlance

Un enlance puede apuntar hacia cualquier tipo de documento :

- Ficheros HTML
- Imagenes
- Sonido
- Texto
- Cualquier otro tipo soportado por el navegador mediante la descarga de 'plug-ins'
- Si el navegador no soporta el tipo de documento, preguntara si lo deseamos descargar hacia nuestro PC.

8.3 Tipos de URL

Como ya debes saber, existen otros servicios disponibles en internet aparte del HTTP :

8.3.1 FTP

Se usa para trasnmitir ficheros de manera anonima. [bah, 'anonima']

El URL FTP se indica con :

FTP://nombre_servidor/directorio/subdirectorio

Para los admin (RTFM) : Muchos servidores web traen activada una opcion por defecto que muestra todo el contenido del directorio cuando la direccion contiene una barra (/) al final.

FTP://nombre_servidor/directorio/subdirectorio/

Tambien se puede agregar el nombre de un fichero en la direccion

FTP://nombre_servidor/directorio/subdirectorio/nombrefichero

Algunos accesos FTP necesitan una cuenta de usuario. Se accede de la siguiente manera :

FTP://cuenta:pass@nombre_servidor/directorio/subdirectorio/nombrefichero...

8.3.1 NEWS

Para acceder a un servidor de noticias

NEWS:nombreservidor_o_grupodenews

ej: news:alt.2600.new

8.3.1 TELNET

[Creo que no hace falta explicarlo.]
[Si crees que si hace falta, lee SET]

8.3.1 GOPHER

Para acceder a una sede gopher

gopher:nombre_sede

8.3.1 MAILTO

Para enviar correo electronico

mailto:nombredestinatario@servidor

9. Incluyendo imagenes --.
<----------------------'

Es dificil inmaginar una pagina web que no contenga imagenes, animaciones, o cualquier atributo visual que llame la atencion. Aqui nos ocuparemos de las imagenes [tal como dice el titulo :)]

9.1 Imagenes

Existen diversos formatos que pueden ser soportados por los navegadores a la hora de visualizar las paginas web :

- BMP
- JPG
- PCX
- XBM
- GIF Estatico (84a)
- TIFF
- RLE
- GIF Animado (87a)

Los mas recomendables son el GIF, JPG y RLE; porque son imagenes vectoriales que ocupan muy poco espacio. Ademas, el GIF y el JPG son practicamente los estandares a la hora de incluir imagenes.

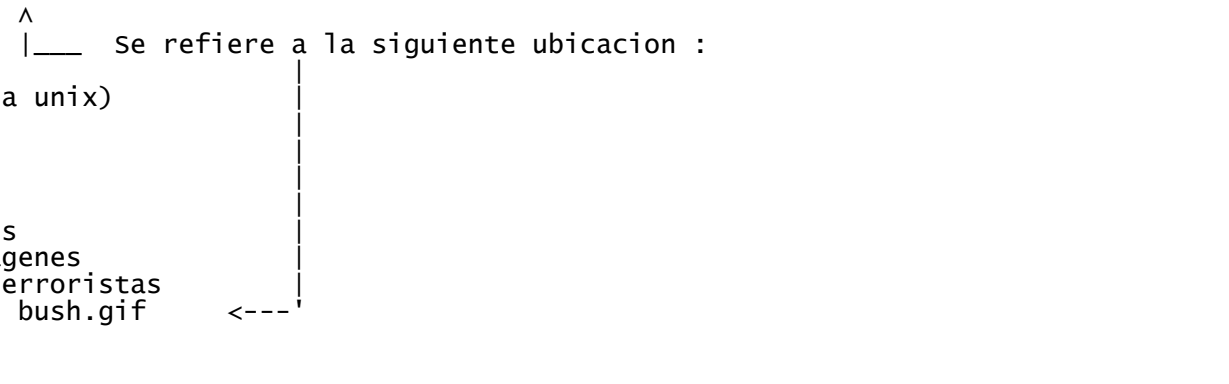
9.1.1 <IMG>

Es la marca que indica la inclusion de una imagen. Se utiliza con el atributo SRC (SouRCe - origen), de la siguiente manera :

```
<IMG  SRC=/directorio/subdirectorio/nombre_fichero_grafico>
```

Por ejemplo, si colocamos una marca como :

```
<IMG  SRC=/imagenes/terroristas/bush.gif>
```



El atributo SRC tambien permite especificar un URL, o sea que tambien se pueden incluir imagenes de otro servidor, tal como :

```
<IMG  SRC="http://www.microsoft/images/nudes/billyisbeingfucked.gif
```

9.1.2 Alineando imagenes

La marca IMG cuenta con otro atributo : ALIGN, que puede tener valores de :

- ALIGN=TOP coloca la imagen en la parte inferior de la linea actual (tiene el valor TOP porque se refiere al borde superior de la imagen, no de la linea, de modo que la parte de arriba de la imagen queda sobre la linea. Pasa igual con bottom)
- ALIGN=MIDDLE la coloca a la mitad de la linea actual
- ALIGN=BOTTOM la coloca en la parte superior de la linea

9.2 Imagenes en navegadores no graficos

Hay algunos navegadores que no trabajan en modo grafico, como Lynx, que no pueden mostrar las imagenes. Para que el documento no pierda el sentido y no despiste al lector, se utiliza un atributo que sustituye a la imagen por un texto en este tipo de navegadores.

```
<IMG  SRC="georgeb.gif"  ALT="Terrorista estadounidense">
```

9.3 Imagenes externas

Se puede ver imagenes mediante el uso de hipervinculos, de la siguiente manera:

```
Puede <a  href="billyg.gif">hacer clic aqui</a> para ver una foto de un
hombre inteligente y (honesto?) cuando joven, que se volvio ladron cuando
crecio.
```

9.4 Imagenes como hipervinculos

Se puede reemplazar el texto de un hipervinculo con una imagen:

```
<a  href="billyg.gif"> <img  src="microsoft.gif> </a>
```

10. Fondo de la pagina, atributos extras al texto --.

Las siguientes caracteristicas de los documentos HTML son del HTML 3, pero me parecio correcto incluirlas porque practicamente todos los navegadores las soportan.

Hay ciertos atributos de la marca BODY que cambian el color de la ventana del browser, del los caracteres y de los enlances.

La utilizacion de los atributos no cambia la estructura del documento HTML.

```
<html>
  <head>
    <title>Pagina de prueba</title>
  </head>
  <body  atributo1  atributo 2 ...>
  documento
  ...
  ...
  ...
  </body>
</html>
```

10.1 Fondo

10.1.1 BGCOLOR

Esta marca especifica un color para el fondo de pagina. Usa el modelo RGB
Por ejemplo :

```
<body  bgcolor="#ffffff">   Coloca fondo blanco
<body  bgcolor="#ff0000">   Coloca fondo rojo
```

10.2.2 BACKGROUND

Coloca una imagen de fondo de pagina, por lo tanto la carga de la pagina resulta mas lenta.

```
<body  background="fichero_grafico">
<body  background="nobush_globalpolice.gif">
```

10.2 Color del texto

10.2.1 TEXT

Controla el color de todo el texto estandar del documento.

```
<body  text="#rrggbb">  Modelo RGB
<body  text="#0000FF">  Texto azul
```

10.2.2 LINK, VLINK, ALINK

Controlan el color de los enlances del documento.

- LINK es color de los enlances no visitados (azul x default)
- ALINK es el color muy fugaz que aparece cuando se hace clic sobre un enlance (rojo x default). En estos dias casi no se aprecia debido a la velocidad de las conexiones.

  [claro que no todos tenemos una T1 en casa. Yo ni tengo modem]

- VLINK es el color del enlance que ha sido visitado (violeta x default)

```
<body  link="#ff0000"  alink="#0000ff"  vlink="#00gg00">
```

Este ejemplo presenta enlances rojos, que cambian a azul cuando se clickean y aparecen en verde cuando han sido visitados.

11. Tablas --.
<------------'

HTML 3 permite realizar tablas con regulacion de la alineacion, el tamanio y el espaciado de la celda.

[si, ya se que esto es de html 2 pero me parecio bien ponerlo aca]

11.1 <TABLE>

La marca <TABLE> se usa para la apertura de una tabla. El fin de la misma se indica con </TABLE>. Se pueden usar los atributos BORDER, CELLPADDING, CELLSPACING . El atributo WIDTH fuerza a la tabla a ocupar un cirto porcentaje de la ventana del browser.

- Border indica el tamanio del borde
- Cellpadding indica el tamanio de la celda
- Cellspacing indica el tamanio de la separacion de la celda

Ej:

```
<table  border=6  cellspacing=6  cellpadding=10>
```

11.2 <TR>

Inicia una fila en la tabla. Debe terminar con </tr>
Tiene 2 atributos : ALIGN para la alineacion horizontal y VALIGN para la vertical. Los valores de estos atributos pueden ser :

- TOP (superior) - MIDDLE (centro vertical)
- BOTTOM (inferior) - CENTER (centro hor.)
- RIGHT (derecha) - LEFT (izquierda)

ej : `<tr  valing=center  align=left><td>1<td>100<td>1000</tr>`

11.3 <TD>

Inicia una celda. Tiene los atribuos ALIGN y VALIGN.

11.4 <TH>

Es identica a <td> pero el texto de las celdas se coloca como texto de cabecera. Tiene los mismos atributos que <td>

11.5 <CAPTION>

Coloca un titulo encima (align=top) o debajo de la tabla (align=bottom)

Ejemplo de tabla :

```
<html>
  <head><title>Ejemplo de tabla</title></head>
  <body>
  <table  border=3  cellspacing=2  cellpadding=10>
  <caption  align="bottom">
  Tabla simple
  </caption>
  <tr  aling="center">
  <td><a  href="http://www.set-ezine.org">SET Ezine</a></td>
  <!-- ****************************** -->
  <!--  COLOCA UNA IMAGEN AQUI DEBAJO -->
  <!-- ****************************** -->
  <td><img  src="IMG_AQUI.GIF"></td>
  <td>Cambia el nombre de la imagen por la que quieras</td>
    </tr>
  <tr>
  <td>
  <ul>
  <li>Uno
  <li>Dos
  <li>Tres
  </ul>
  </td>
```

```
    <td>Puedes colocar cualquier elemento dentro de una tabla</td>
    <td>HTML</td>
      </tr>
    <tr>
    <td  aling="center"  valing="middle">
      Introduce tu nombre :
      </td>
    <td  aling=left>
    <form  method="post"  action="nada">
    <input  name="nombre">
    </td>
    <td  align="center">
    <input  type="submit"  value="Clic Me Pringao!">
    </form>
    </td>
    </tr>
    </table>
  </body>
  </html>
```

12. Paneles o marcos --.
<----------------------'

12.1 Introduccion

Cuenta la historia, que alla por el anyo 1996, Netscape propuso nuevos mandatos HTML que eran soportados por su nuevo navegador, el Netscape Navigator [Notescapes Navegando :)], en su version 2.0 (creo).

La idea era simple, pero muy novedosa y util : en vez de usar la ventana del browser para mostrar un documento, poruqe no mostrar 2, o 3, o mas aun.

Como se haria esto ? Muy simple, dividir la ventana del navegador en la cantidad de secciones deseadas, y colocar alli el documento que se deseara.

Cada una de las secciones recibe el nombre de panel (marco segun billy g)

12.2 Creacion de paneles

En realidad, Netscape solo agrego tres marcas al lenguaje HTML para aportar esta funcionalidad. Sin embargo, la estructura de un documento dividido en varios paneles difiere de la de un documento clasico. En un documento clasico, el cuerpo esta entre las marcas <body> y </body>. En el caso de los paneles, el cuerpo se inserta entre las marcas <frameset>..</frameset> y ademas solo puede componerse de las tres nuevas marcas <frameset>, <frame> y <noframes>.

Documento clasico

```
<html>
<head>
..
</head>
<body>
..
</body>
</html>
```

Documento de definicion de paneles

```
<html>
<head>
 ..
</head>
<frameset ...>
  <frameset>
   ..
  </frameset>
   ..
  <frameset>
    <frame ..>
      ...
  </frameset>
  </frame>
</frameset>
</html>
```

12.2.1 <FRAMESET>

Es la marca que permite dividir una zona en subzonas, verticales u horizontales. Si no hay ninguna zona definida aun, las dimensiones se aplican a toda el area del browser. La marca tiene 2 atributos :

Rows : Para dividir la zona en otras horizontales

ROWS="altura_zona1,altura_zona2,altura_zona..."

Los valores de la lista deben ser numeros enteros, con el siguiente formato :

n : indica la altura en pixeles
n% : indica la altura segun el porcentaje de la zona madre
n* : n es opcional aqui. El asterisco indica que se debe ocupar todo el espacio aun disponible

Cols : Usa la misma metodologia de rows, pero crea zonas verticales

Por ejemplo, para crear 3 zonas horizontales, 2 de igual tamanio y otra mas grande podriamos hacer lo siguiente :

<frameset rows="25%,25%,50%"> o <frameset rows="*,*,2*">

12.2.1 <FRAME>

Se utiliza para caracterizar las subzonas definidas mediante <frameset>
Se utilizan seis atributos

SRC="url" : Indica la url a mostrar en la zona

NAME="nombre" : Indica el nombre de la zona, a fin de que puede ser accedida desde un enlace. Los nombres predefinidos (_blank, _top, _parent, _self) no pueden ser usados ni tampoco cualquiera que comienze con un guion bajo (_)

MARGINWIDTH="n" : Indica el numero de pixeles de los margenes derecho e izquierdo de la zona.

MARGINHEIGHT="n" : Indica el numero de pixeles de los margenes superior e inferior de la zona.

SCROLLING="yes/no/auto" : Indica si la zona debe tener barra de desplazamiento

NORESIZE no tiene un valor. Indica si el tamanio de la zona puede ser ajustado desde el navegador.

12.2.1 <NOFRAMES>

Es una marca que se usa para presentar los documentos en los navegadores que no soporten paneles. Este texto es ignorado por los que si los soportan. El funcionamiento es simple : un browser solo tiene en cuenta las marcas que conoce. Uno que no maneje paneles ignorara lo comprendido entre <frameset> y </frameset> y tambien las marcas <noframes> y </noframes>.
Por lo tanto, si se retira todo este texto, solo queda lo comprendido entre <noframes> y </noframes>

12.3 Utilizacion de paneles

Ahora aprenderemos a crear documentos con paneles y como acceder a ellos mediante enlances.

12.3 Target

Ademas de las tres marcas que describi mas arriba, tambien se creo un nuevo atributo, que afecta a las maras que caracterizan a los enlances.
El atributo es TARGET, que precisa el nombre de la zona que debe recibir el documento correspondiente al enlance.
Por ejemplo :

Ver los usuarios

Target indica que se debe mostrar el documento 'users.html' en el panel

cuyo nombre es 'zonausuarios'. Es decir un panel definido con
<frame name="zonausuarios">.

  Si target apunta a un nombre de panel que no existe, se crea una
nueva ventana.

  El atributo TARGET posee unos valores predefinidos :

  * _blank, que crea una nueva ventana

  * _self, que carga el documento en la misma zona del enlance. Si no se
     indica el atributo target, _self se toma como valor predeterminado

  * _top, que indica que se eliminen todos los paneles de modo que el
    documento ocupe toda la ventana del browser

  * _parent, indica que el documento ocupe toda la superficie de la zona
    en la que se ha visualizado el documento que contiene el enlance.

 *

  Hasta aqui llega esta entrega, espero que les haya gustado.
  Nos veremos en otra apasionante entrega, a la misma hora y
  por el mismo canal.

  Buena suerte,
  elotro
  elotro.ar@gmail.com

    -
  | No hay que pelear hasta morir. Hay que pelear hasta vencer |
    -

                                                        Ernesto Che Guevara

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.

<++> keys/grrl.asc

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: PGP 6.0.2

mQDNAzcEBEcAAAEGANGH6CWGRbnJz2tFxdngmteie/OF6UyVQijIY0w4LN0n7RQQ
TydWEQy+sy3ry4cSsW51pS7no3YvpWnqb135QJ+M11uLCyfPoBJZCcIAIQaWu7rH
PeCHckiAGZuCdKr0yVhIog2vxxjDK7Z0kp1h+tK1sJg2DY2PrSEJbrCbn1PRqqka
CZsXITcAcJQei55GZpRX/afn5sPqMUslOID00cW2BGGSjtihplxySDYbLwerP2mH
uO1FBI/frDeskMiBjQAFEbQjR2FycnVsbyEgPGdhcnJ1bG9AZXh0ZXJtaW5hdG9y
Lm5ldD6JANUDBRA3BARH36w3rJDIgY0BAb5OBf91+aeDUkxauMoBTDVwpBivrrJ/
Y7tfiCXa7neZf9IUax64E+IaJCRbjoUH4XrPLNIkTapIapo/3JQngGQjgXK+n5pC
lKr1j6Ql+oQeIfBo5ISnNympJMm4gzjnKAX5vMOTSW5bQZHUSG+K8Yi5HcXPQkeS
YQfp2G1BK88LCmkSggeYklthABoYsN/ezzzPbZ7/JtC9qPK4O7Xmjpm//ni2E1OV
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-----
```

```
Tipo Bits/Clave    Fecha      Identificador
pub  1024/797CF983 2003/12/30 blackngel

-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: 2.6.3ia

mQCNAz/xegIAAAEEAKEQLvOk3XAXK44t/6WyIB8u5F7mkBQs5enmvfZY2HAF7mCG
c+7CSzS4+M13OcFjIUxm5b0zrjxVhN/NAvlq/OamiOQesuBO88Dd5a7n1cCXVqFu
A+5cnfHvT/9rYER2ewRRFCBZwYBuLf2HYgDzfn7NDKfkNI6uHw5FGtB5fPmDAAUR
tAlibGFja25nZWyJAHUDBRA/8guDmMtyBAFG0kUBAdjZAwC1qw1FAtEhash0+t/V
wUBmvCklJOO+SfJ7ZEs8Q4K+RMPfosgx8DcD9j/UxfYyHRZ9nKUTvD4O7Ca/Th1m
VXDX1iHVINCfbRkAPEQ6vgXwsBGI5wjhtLXfmUaqgTWUVMSJAJUDBRA/8XosDkUa
0Hl8+YMBAfPNA/9zpMUyWItQE2g0Jm81PKR6l8zovigAJVUaIZxoxEh7dN0c1A4+
RqXUZJs9NW+/goRT24mKurGjVMR/DOZ7hp4fRB23gKflaQ8YvmeOPJkAsiQD3AzN
fRqaWnfZZpEYTfgl+1JPXDT+1lfxutuWgNYAIQHRLORiAd/XGc5gcLqkgw==
=/8IH
-----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+ITEqIis89Uc2f8RiJqgDX5ZeL61OpoWLp4CqsdExxRixOasYdKRRr
978BA/9jdXL/mjheQUMV19IYB4nPEXJ7O6qjqopA15U5Lgn7Ndp+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+
4O6yrLiP+FBIEMm/WIzyiMWH6YxhUz9PN6HhcFJna50y4CRTQ5fFoksCaFd1hquQ
PZes1LI4MTYJ+cWaSpOWiEYEGBECAAYFAj926/IACgkQPlxjGDYynKZBzQCeJMjQ
1izVC11nVdN/Yz6nDs82CGwAn2V7opxfIynjKBxggv0/e88WJJSh
=FYUn
-----END PGP PUBLIC KEY BLOCK-----
```

jepkc

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: PGPfreeware 7.0.3 for non-commercial use <http://www.pgp.com>

mQGiBD6Q1qQRBADDe/cDUraPXLcQNi4K0LZVU3b9NeKooTaa7HRjpc7l3yLZBb8z
7Ewr3Qdhg15MTJCza6CQ5WsbmUdpU4dyh8QwgIB4hO93mwsOl5vzt3VfiZmPLL4Z
VJtBZgtTbqKk2LQtQlM4G6MCXr/TpZ2fJCxNgPH6WceeslRe+AC9Vj9epwCg/4yi
4i8bC79xrhBEc0c8on5bSkED/ieI6x/Nkcx/ZbsjILEyph8yvzdO7UFnRLfk2eB5
4cJa6W5KzkIlPxhjgkeaH4OwhXgufr/DwaiEavT4+Wpo1KqhEZhIumzOx/+Qd+P/
TtOxl2V482pahueVED3uqWJJjpdH3lL+J4koMenTxLN1uaoxF3itq3JOy9BG48G8
OWpIA/9SQmtYKHiJzPi/Yfz7npB+YYNhc0chpk7YijzQMvOl+ULQ42Tnl7BwospO
GkkhZmQjh2jklXpnM5pdlkQMcdWzN4wrwUb43tCUTsEoghyeLMCrE0Bnxc6z1Wgf
vqx6ZSuw69KygqFR8ItOj/2Rn6qRmSptLZnLnxAcKwqGAGu/UbQXamVwa2MgPGpl
cGtjQHlhaG9vLmNvbT6JAFgEEBECABgFAj6Q1qQICwMJCAcCAQoCGQEFGwMAAAAA
CgkQqxLUBMPf9AO56ACeJps2gFdLPdObMcdM6owaUURkoFYAoNH31fxE2vl1IYVL
7nOEWAHGHqrGuQINBD6Q1qUQCAD2Qle3CH8IF3KiutapQvMF6PlTETlPtvFuuUs4
INoBp1ajFOmPQFXz0AfGyOOplK33TGSGSfgMg71l6RfUodNQ+PVZX9x2Uk89PY3b
zpnhV5JZzf24rnRPxfx2vIPFRzBhznzJZv8V+bv9kV7HAarTW56NoKVyOtQa8L9G
AFgr5fSI/VhOSdvNILSd5JEHNmszbDgNRR0PfIizHHxbLY7288kjwEPwpVsYjY67
VYy4XTjTNP18F1dDox0YbN4zISy1Kv884bEpQBgRjXyEpwpy1obEAxnIByl6ypUM
2Zafq9AKUJsCRtMIPWakXUGfnHy9iUsiGSa6q6Jew1XpMgs7AAICB/9K3XPirZjx
j4N9s3Y1i+VJdrQR8lr9jKWK3w3ocIZWGNVlaPba7kWZrV42UIw2KTwfvjGmhIyW
yIwZ5xmfR71/7sNXdAZSerDLo+PAEGkVhU10Jq5/hDzphJcYz60IkQ7Pf7e+nOUb
PaFBeaiSc5zWlVvFNIT9OZ/cAAYW9EXrJXqGzaPjDgwL5uS2qpPVXnwQLjzH2Kcg
moyw8+BWomGCxaaKvo9+o+Obz4Om56+Gib6imKHUrvfJ4tdxRDf7s/MxIBBQDo/R
sLhfnKGX9kVN1JIUBHBJ4IrEih2RQA9RwVHmLy4svOtO7UaslX7JH1nV6F64qCO/
358NyqLPXiNHiQBMBBgRAgAMBQI+kNalBRsMAAAAAAoJEKsS1ATD3/QDWt0AnAr1
eyM2dCT5+97d4sicUGIpHrmwAJ9zO89I8B5lMJpy0tTiC9hwOBnyaQ=3D=3D
=3DbZRC
-----END PGP PUBLIC KEY BLOCK-----
```

Qaldune


```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v1.2.4 (GNU/Linux)

mQGiBEKaI6gRBACY68UW0MfpZyvJFLtn7NUZNDCpcrxAuv5QgmUDZU1ZJyZTKUc8
uzdsaMEH2+zno1gFrW51vgiBIpiGiA0KDBBQT7ryecuHeCcDOIsaP2IBE/ArbXnf
f942d/9oOUp5TxdVnAkMl46VA+9C+HEmU5JEBocdW1Sw5N2w2JUaJORTTwCgw2gv
RjiuGuL5eAX0HcFmLSrhbBUD/2w8EmV7L7mUZNQ2lmog6AHIWhoJW19wVgb4nccP
pl6nnRR6qO5uoZw+ir0zXyfAZvrFAJtAK0Ewrj/YY+f52nW7TcS/0lQ4tLztyxpJ
AdVTlKgTpiRMwobYxU0XIFYX+Ww0yKwhQ9d0m9u0o7XDR4otzcJ6Z8T1ejz1fp04
12/QA/4o1bLyarRnnfOL6XY2fSrqs0K1TGOL8Yu2VgSe1JN0iRcrjaottlyxYI4r
KalR83/LBRr3PUsAB+CJ82zN9XsOFEOPCLzf4OcJvNTT7i26HjisZLDwx7CxbaDH
aIRCQCouYBrkuOHQwsMj5oOIO7pBST12XFiISCDLStjOGb3YHLQbUWFsZHVuZSA8
cWFsZHVuZUBnbWFpbC5jb20+iGQEExECACQFAkKaI6gCGwMFCQWjmoAGCwkIBwMC
AxUCAwMWAgECHgECF4AACgkQSBVtx5NXcq1b9ACcDdoNZKXxZcP2hlJALppCZSg6
RkgAoIwl1cUBc8JI+l0d6HZfJIYjlCltuQENBEKaI6kQBACrkkEJzdgsOEz1Pi84
XysDOrR5qgujv8maoCKuhwXdy+0UYnaJEPB+z5GYA58Q1GNXIYFpgngivrFK/7o/
SazGdgq5coXTv+UJ7/zieUAdpoiKN9kpi2gF3NV4KkUh9C3p16nmU6n1Xtm9tptx
nu2uDm6JJoPJaXpGYZS7si3sKwADBgP/TNARMWtxYYJBbF8xyfBHDDQKMDCWY6gX
B+ba0kQvcc/9HtOpZHzGSBeJmwUwcDmiTCLrc4ITfITWzHGQpvlYz/LrIpIj/VqS
ocIGyiLG2OhTzL+ijd33DufrumotviRgUxVWowTSfO+lqRONxQ1nkloquxKnRSm+
z6evWln8gXiITwQYEQIADwUCQpojqQIbDAUJBaOagAAKCRBIFW3Hk1dyrTtjAKC0
FjAJJK2NuPiFg4nvdvnQENbmgQCfR1srTeM+YCQH/AdbNQtLV+pJzfU=
=up4N
-----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
AQEDwgQAtwrBv3To4QnN67jeNZSxjoZC2gAb7Yq4gueP2OyfARRlKOSompGgwypI
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 250'34 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.                                  |
 ------------------------------------------------------------------------
```

"No entiendes realmente algo a menos que seas capaz de explicarselo a tu abuela."
Albert Einstein.

SET, - Saqueadores Edicion Tecnica -. Numero #33

EOF