Para todos vosotros, niños y niñas, jóvenes y ancianos, civiles y militares sin graduación, ahora y en directo, el segundo tutorial en español sobre Palm Cracking por vuestro estimado amigo y compañero X-Grimator...

RECAUCHUTANDO CÓDIGOS

¡¡Hola a todos, gamberros!!... ¡damas y caballeros, bienvenidos al circo de variedades del domador X-Grimator, donde podrán contemplar mujeres barbudas, enanos deformes, códigos de Palm e instrucciones en Motorola ASM!.

1- Introducción:

Bueno, de nuevo vamos a meternos de narices en el apasionante mundo del Palm Reversing y las desastrosas consecuencias que ello puede tener para la salud de aquellos a los que no le gusta que pensemos por nosotros mismos.

Ni que decir tiene que la explicación de lo que es una Palm ya la expuse en el primer tuto y a él me remito en lo tocante a eso (RTFM, "read the fucking manual" o en lengua cristiana, "lee el jodido manual"). No voy a perder ni mi tiempo ni el vuestro repitiendo lo mismo.

La idea que vamos a desarrollar en este tuto es la de parchear un programita para que se registre sin problemas con cualquier serial que le metamos.

2- El objetivo:

Un juego de barcos llamado Battle Fleet. Que por cierto, hay que ser canalla para hacer un juego de guerra de barquitos y cobrar por ello. Esas cosas deberían ser freeware. Es como si yo ahora comercializo un juego de ordenador que vaya de canicas o peonzas. Eso está feo y no se hace, y sólo por ello debería calcinarse el programador en las llamas del más tórrido infierno (que por lo que sé, debe ser una oficina de Hacienda o una consulta de un dentista, je, je...).

Si queréis localizar el juego, no tenéis más que hacer uso y disfrute del Google, el mejor invento de la Humanidad después del papel higiénico. Imaginaos un mundo sin papel higiénico o sin Google... puafff, que grimaaaaaaaaaa...

3- Las herramientas:

En Palm tenemos muy pocos tutos sobre estos temas y menos herramientas todavía. Pensé meteros en este ensayo las herramientas como hice en el anterior pero al final sólo logro que el zip ocupe más, así que os recomiendo que busquéis en Internet que por algún sitio aparecerán. En este caso, nuestra carpa de circo se va a nutrir de...

a) El prestigioso acróbata **PalmdeMon 0.24**, que con su capacidad de desensamble nos permitirá ir dando saltos por el código.
b) El mundialmente conocido ilusionista, recién llegado de la misteriosa India, **Prc2bin**, único en convertir archivos .prc de Palm en binarios.
c) El hombre de acero, de músculos pétreos capaz de arrancarle la cabeza a un toro, **PilotDis**. Sólo él sabe coger un archivo y arrancarle los intestinos para que aparezca desensamblado.
d) El poder de la mente, la ciencia de la telepatía tiene su cúspide en **Prcedit 1.0 b2**. Con su concentración logrará saber todo lo que contiene la mente de un archivo .prc y cambiar muchas cosas de ella.
e) Y el domador de fieras y payaso, **X-Grimator**. Sin esta herramienta, no hay tutorial.

De todos modos no olvidéis que esto del crackeo es un arte en continua evolución e innovación, así que cualquier otra herramienta que conozcáis o aparezca en mercado, será bienvenida. No le hagáis ascos tampoco a **PrcExplorer 1.0.14**, es una pasada. Dios, lo que inventa la gente que piensa... bufff...

4- ¡¡Comienza el espectáculo!!:

Chim-pata-chim pata-chimmmmmmmmm... Lo primero que vemos, avezados crackers de Palm, es que sin el Palm Os Emulator o sin un dispositivo PDA difícilmente podemos crackear el programa, así que ni cortos ni perezosos, nos vamos a www.palm.com y nos downloadeamos el emulador, que es gratuito y una preciosidad. A mí me cae muy simpático...

Una vez hecho esto, Instalamos Battle Fleet (que nos habremos bajado de algún sitio) y tras ejecutarlo nos vamos directos a la pantallita de "register" donde metemos un número basura cualquiera (insisto en mi consejo: no hay nada mejor que meter vuestro DNI. A la hora de debuggear es un número que reconoceréis a la primera y que no da lugar a error con otras cadenas de números... ¡¡ayyyy la sabia experiencia del vetusto ingeniero inverso!!).

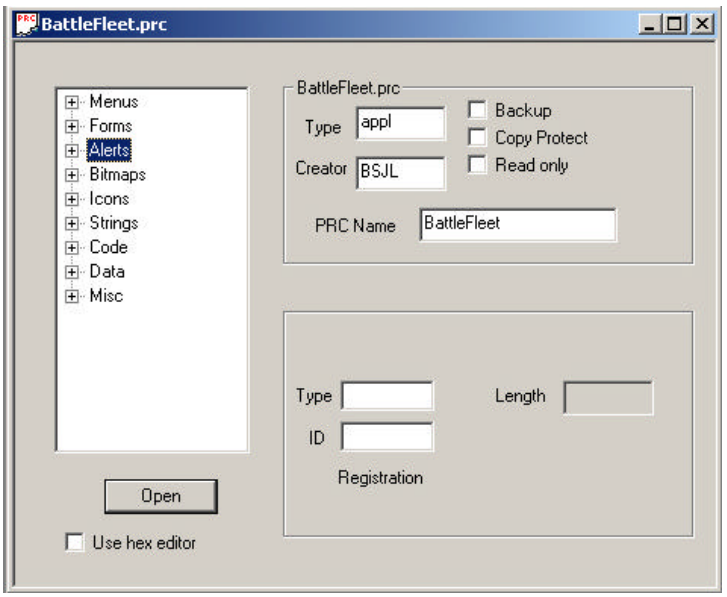
Maravilloso. Nos coge letras y números. Le damos a Ok y...

La madre que lo parió... no acertamos el serial válido... será desgraciado.

En fin, visto que la fiera se nos pone enrevesada, el domador X-Gri saca el látigo negro de seis colas y se dispone a azotar con toda su rabia. Para ello va a usar a su mejor telépata e hipnotizador: Prcedit.

Ejecutamos Prcedit y cargamos el jueguecito en él. Ummmm, parece que nos abre un montón de cosas en la ventana de la izquierda... ummmm. Como ya sabemos del primer tutorial, lo que nos interesa averiguar es qué mensaje de alerta nos ha aparecido.

El telépata nos dice, casi con sagaz certidumbre que la Talt 1900 es la que buscamos. Y hasta nos permite modificar el texto de la ventanita... esto me da una idea para un futuro keygen donde la propia ventana nos cante el serial correcto... quizá sea la segunda parte de esta lección, no sé...



Bueno, ya sabemos gracias a esto que la Talt1900 es la que nos chiva el error “Demo mode”... como si no lo supiésemos. Este Jim Love ni siquiera nos dice que el serial es incorrecto. Si hasta pretenderá despistarnos y todo el pobre... Lo mejor es eso que dice de que si te registras podrán crear grandes juegos... ¡será soberbio el capullo ese del Jim Amor!... con ese nombre igual es mariquita y todo...

Un Ok más y volvemos a la ventana principal del juego. Bueno, ya sabemos la táctica de nuestro enemigo. Veamos como se comporta ante la magia...

5- Magia potagia: abracadabra pata de cabra, quiero que Prc2Bin me facilite las cosas:

Pues nos toca tragarnos el código muerto para ver como funciona el tinglado éste. Y en este punto de las cosas, nadie mejor para ayudarnos que nuestro chamán del ilusionismo y la magia oscura de los hindis... Prc2bin.

Desde una ventana de consola DOS, abrimos Prc2bin y tecleamos

Prc2bin BattleFleet.prc

Y pummmm lo que era una chica del público sin complicidad alguna con nuestro mago, se convierte por arte de birlibirloque en una preciosa colección de archivos dónde solo nos interesarán dos de ellos: Talt1900 (sólo para editarlo con un editor cualquiera de texto y comprobar que efectivamente es lo que buscamos. El 1900 está en hexa, así que será Talt076c), y el code0001, que es el código propiamente, donde tenemos el tomate...

Muchas gracias Prc2bin, tu número ha sido fascinante. Aplausos por favor, y que no decaiga la expectación que ahora comenzamos con el número de fuerza.

6- ¡¡Quiero verte las tripas!!:

Haciendo balance de lo que hemos hecho hasta ahora, tenemos lo siguiente: un mensaje de error localizado, el nombre del archivo de tal mensaje y el código que tenemos que tragarnos. Hombre, se puede considerar ya un avance ¿no?, pero es el momento de la actuación de PilotDis. Temblad, pequeñines...

Ni cortos y mucho menos perezosos, nos vamos a una consola de DOS y de ahí al directorio donde tengamos PilotDis y en el que habremos metido el archivo code0001.bin. Una vez hecho esto, tecleamos

Pilotdis code0001.bin

y nos aparece un archivo llamado code0001.bin.s, que podemos editar con cualquier editor de texto, el Block de Notas, por ejemplo. Santo Cristo, que sangría se va a montar aquí... tenemos

ya el código desensamblado. El pseudo-animal de PilotDis le ha arrancado las vísceras al programita entre risas y alaridos sádicos de placer (tengo que dejar de ver películas de serie B que luego sueño). Es esto lo que nos interesa, así que vamos a analizar el código y olvidarnos de lo demás.

Comenzamos la fase zen-cracking, así que las instrucciones ahora son:

1- No pienses en más archivos que en el que estamos analizando.
2- Descuelga el teléfono y cancela tus citas (esta idea es de Látigo).
3- Prepárate un whiskila o un vodkila, como hacemos en el canal #crackers. Importante que el tequila sea de calidad. El whisky o el vodka... bueno, tal y como sabe esa historia no es tan importante...
4- Piensa mucho y ten más calma si cabe.

Estiro las piernas, pongo música para la ocasión y con el Block de Notas delante y code0001.bin.s, me dedico a buscar el siguiente string:

$76c

parece ser que esto significa que se está metiendo en la pila el mensaje 76c, que es el Talt1900 que habíamos localizado antes. El mensajito en cuestión parece que se carga muchas veces, pero sólo nos interesará ésta:

```
00007fca  4e4fa153                sysTrapFldGetTextHandle ✍ lee mi serial basura por 1ª vez
00007fce  2448                    MOVEA.L     A0,A2
00007fd0  200a                    MOVE.L      A2,D0
00007fd2  584f                    ADDQ.W      #4,A7
00007fd4  671c                    BEQ L870✍ de ser 0 me salta (debugger me dice que no salta)
00007fd6  2f0a                    MOVE.L      A2,-(A7)
00007fd8  4e4fa021                sysTrapMemHandleLock
00007fdc  584f                    ADDQ.W      #4,A7✍ esto es una corrección de pila
00007fde  2f08                    MOVE.L      A0,-(A7)
00007fe0  486dfbca                PEA         -1078(A5)
00007fe4  4e4fa0c5                sysTrapStrCopy
00007fe8  2f0a                    MOVE.L      A2,-(A7)
00007fea  4e4fa022                sysTrapMemHandleUnlock
00007fee  4fef000c                LEA 12(A7),A7
00007ff2  4eba8570      L870      JSR  L56✍ RUTINA "CHECKREGISTRATION" (vaya, vaya…)
00007ff6  2b40fd66                MOVE.L      D0,-666(A5)
00007ffa  0cad0000014afd66        CMPI.L      #330!$14a,-666(A5)
00008002  6410          BCC       L871✍ BRANCH IF CARRY CLEAR –C
```

---- A ESTE TROZO DE CÓDIGO NO LLEGAMOS SI NOS PASARON YA LOS 7 DIAS DE PRUEBA ----

```
00008004  3f3c07d0      MOVE.W #2000!$7d0,-(A7)✍ DEMO EXPIRED (si ya pasó el trial de 7 días)
00008008  4e4fa192                sysTrapFrmAlert
0000800c  38bc0016                MOVE.W      #22!$16,(A4)
00008010  544f                    ADDQ.W      #2,A7
00008012  6074                    BRA L876
```

```
00008014  0cad00000198fd66  L871      CMPI.L      #408!$198,-666(A5)
0000801c  6314              BLS       L872✍ SALTA SI ES LOWER O SAME THAN C/Z (no salta)
0000801e  3f3c076c          MOVE.W #1900!$76c,-(A7)✍ DEMO MODE FOR 7 DAYS (BAD BOY)
                                                     ¡¡¡¡APAREZCO AQUÍ!!!!
```

Una cosa que debemos apuntar aquí es el estudio de la línea

```
00007ff2  4eba8570      L870      JSR  L56✍ RUTINA "CHECKREGISTRATION"
```

aunque al desensamblar no la vemos exactamente así, si utilizamos el debugger que viene con el emulador de Palm y que se llama Palmdebugger (¡¡cielos, qué originales son los zombi-programadores!!) observamos que la rutina de la Label 56 tiene ese nombre.

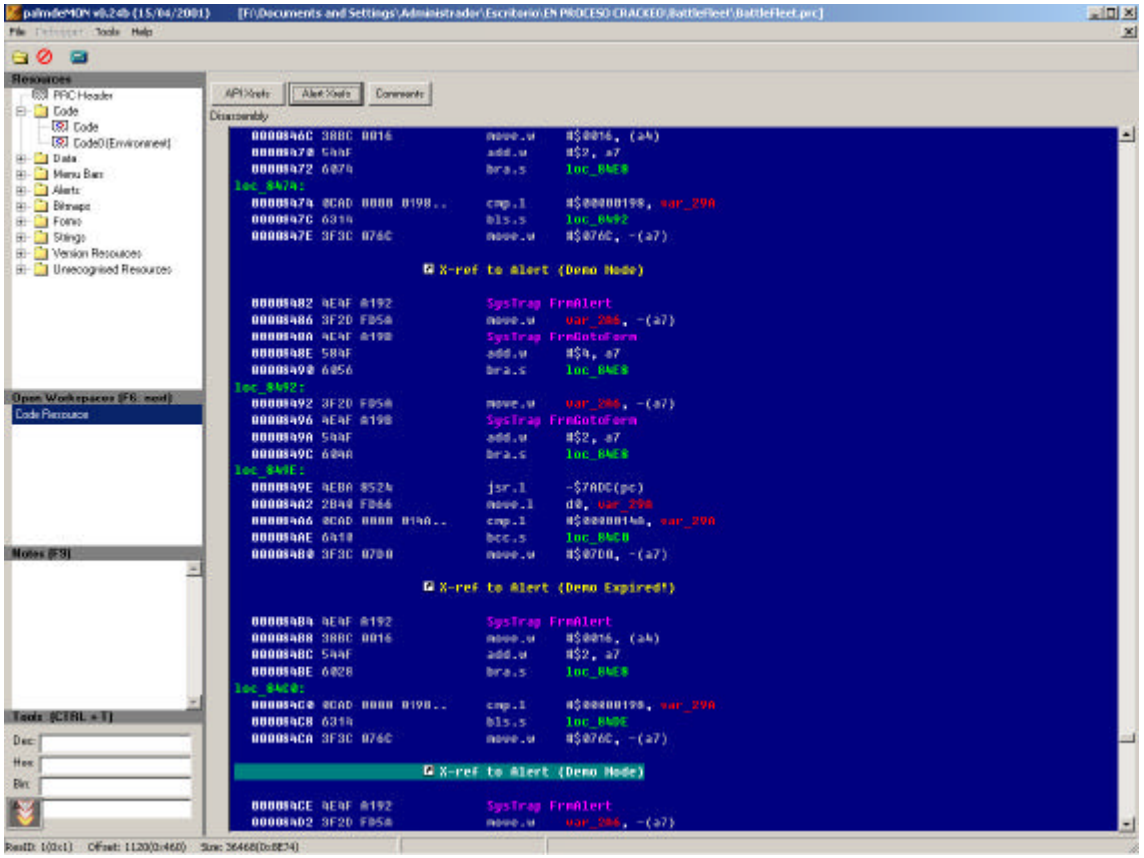
Para los que no lo sabéis, aunque esto sólo es un adelanto para próximas lecciones, el debugger se ejecuta lanzándolo con el comando

att

y los breakpoints en las apis se ponen como

atb “nombre systrap”

pero eso es otra historia que de momento no viene a cuento. Por cierto, una “label” es algo igual a un proceso, o una subrutina de la principal. Y JSR es un salto obligatorio hacia esa rutina. Bien, bien... podemos también usar a nuestro fabuloso malabarista PalmDeMon para dar saltitos de ranita por el código. Os pongo una pantalla para que os hagáis una idea de la potencia que puede llegar a tener este saltimbanqui:



pues aquí vemos un trocito del código que andamos revolviendo pero puesto bonito, para los que no os guste el encanto cutre del block de notas. El PalmDeMon es un pedazo desensamblador hecho por Carpathia que lamentablemente lleva parado un par de años. Con los botoncitos de arriba podemos ver las alertas, las llamadas a apis del la Palm (las famosas systrap) etc... una joya vamos, para que juguéis un poco...

7- Guay tío, mola, dabutén, fetén, pero no me entero de nada...:

Vale, vale, ya sé que me enrollo demasiado... vamos a meternos en harina.
Le damos un sorbito al whiskila y empezamos a mirar código y código:

00007ff2 4eba8570 L870 JSR L56✍ RUTINA ”CHECKREGISTRATION” (vaya, vaya…)
00007ff6 2b40fd66 MOVE.L D0,-666(A5)
00007ffa 0cad0000014afd66 CMPI.L #330!$14a,-666(A5)
00008002 6410 BCC L871✍ **BRANCH IF CARRY CLEAR –C**
……
00008014 0cad00000198fd66 L871 CMPI.L #408!$198,-666(A5)
0000801c 6314 BLS L872✍ **SALTA SI ES LOWER O SAME THAN C/Z (no salta)**

0000801e 3f3c076c MOVE.W #1900!$76c,-(A7)✍ DEMO MODE FOR 7 DAYS (BAD BOY)
¡¡¡¡APAREZCO AQUÍ!!!!

Yo aquí veo lo siguiente, es decir, un salto obligatorio a una label que encima se llama "checkregistration" o sea, de penita penita pena. Un par de líneas después me compara 330 decimal (14a hexa) con el registro a5-666. Aquí debo hacer una pequeña pausa para recordaros que en Motorola la ruta fuente-destino se lee al revés que en ASM x86, es decir

MOVE.L D0, -666(A5)

significa que movemos el contenido de D0 (tenemos como registros de datos D0 hasta D7, y como registros de direcciones A0 a A6, ya que A7 es la pila), digo que movemos D0 con tamaño Long (doble word) a la dirección A5 menos 666.

Bueno, después de esta breve explicación, seguimos diciendo que si la comparación no sale como debiera, me pega un salto a la label 871 que me hace otra comparación y que me manda directo a la orden de que salga el mensajito de bad boy. Para no aburriros demasiado, deciros que BLS significa branch lower same, que quiere decir, menor o igual al flag Z (cero) y BCC significa branch if carry clear –c. Esto es, dos saltos condicionales. Vaya vaya... por último, aunque no la puse, la siguiente línea sería

00008022 4e4fa192 sysTrapFrmAlert

que ya muestra el mensaje de alerta en la pantalla y que es la llamada a la api para esas cosas de mal vivir.

Respecto al primer salto, el de

00008002 6410 BCC L871✍ **BRANCH IF CARRY CLEAR –C**

no le damos más importancia, ya que es el que mira si pasaron o no los 7 días de prueba, y que nos llevaría a una ventanita de "Demo expired!" tan odiosa y que todos vimos alguna vez. Precisamente estaría una línea más abajo. En cuanto al segundo salto, el de

0000801c 6314 BLS L872✍ **SALTA SI ES LOWER O SAME THAN C/Z (no salta)**

el bueno de X-Grimator fue de campeón pensando que sabía mucho y ni corto ni perezoso le metió un pedazo BRA (salto obligatorio, salta siempre) que no valió para nada: no salía el mensajito malo pero no se registraba al volver a entrar. La típica trampa de elefantes, más grande que las mentiras que dice X-Gri para disimular que tiene resaca un domingo delante de sus padres. NOPear el salto BCC tampoco sirvió de nada. Sólo quedaba enfrentarse al Check Registration. La batalla sería dura.

8- Duelo en la carpa. Payasos asesinos:

Otro sorbito al vodkila (tengo dos vasos con distinta bebida, es que no me decidía) y nos vamos de cabeza a pelear con la rutina de chequeo. Cuerda floja ¡¡sin red!!.

Si observamos en L56 (la rutina de chequeo) encontramos al principio esto:

00000610 4e4fa2a9 sysTrapDlkGetSyncInfo ✍ **OBTIENE MI USER ID DEL HOTSYNC**
00000614 486dfba1 PEA -1119(A5)

hasta aquí podemos llegar buscando en el Block de notas "L56" en el código. La primera línea es la llamada a la api para leer nuestro nombre de registro en el Hotsync, nuestro ID de User, como en Windows cuando te pide usuario y empresa. En la segunda línea estamos cargando y haciendo apuntar la pila a a5-45f. Si en el debugger mirásemos esta dirección, veríamos el UserID que usamos para conectar datos de la Palm al PC. Hummmm, así que el serial no nos vale igual a todos ¿eh?. Buen dato. El Sr. Love ha hecho un keygen que calcula el numerito. No está mal, esto es más interesante que el primer tutorial...

Bueno, después de mucho código que os ahorro, la rutina de chequeo (tras haber tomado letra a letra del ID y haber calculado el número correspondiente) llega a esto:

```
0000079a  486dfbca                PEA    -1078(A5)      ✍ y a5-436 es mi serial basura
0000079e  486eff7e                PEA    -130(A6)   ✍ y a6-82 es "UBF11415-8014611200"
```

los PEA, como dijimos, cargan un valor y hacen que el puntero se fije en ellos. Son importantes. Y da la casualidad que tenemos dos seguidos. Y pardiez, el segundo es pero que muy extraño... ¿me entendéis verdad?... pero de momento, dejemos esto para ir a algo más sencillo. Seguimos bajando hasta

```
000007e8  4a2eff77          L63   TST.B   -137(A6) ✍ compara byte de a6-89 con cero
000007ec  672a                    BEQ     L64      ✍ si es igual, salta (no lo hace)
000007ee  0c840000014a            CMPI.L  #330!$14a,D4✍ compara 14a con D4
000007f4  6422                    BCC     L64      ✍ D4 está vacio y por ello no salta
000007f6  283c0000014a            MOVE.L          #330!$14a,D4✍ mueve 14a a D4
```

el salto BEQ y la comparación son lo que nos interesa recauchutar, y lo haremos así:

```
000007ec  4e71                    NOP
000007ee  283c0000014a            MOVE.L #330!$14a,D4
```

y con esto conseguimos que no realice nada cuando debería hacer el salto "igual a" hacia L64 y además desmontamos la comparación. Para poder hacer esto, cogéis vuestro editor hexadecimal favorito y cambiáis los opcodes (resaltados en negrita). Con esto habremos logrado que cualquier número sea aceptado por válido en nuestro querido y apreciado jueguecito.

9- The End y saludos:

Lo divertido se acaba muchachotes. Bueno, estoy seguro de que esto no le interesa a demasiada gente, pero bueno, así de duras son las cosas. Si lograsteis llegar hasta aquí, es que vuestro whiskila está ya vacío. Y si está vacío es que no os habéis enterado de nada y estáis beodos perdidos. Si vuestro whiskila/vodkila no está vacío es que no llegasteis hasta aquí, y por lo tanto tampoco pudisteis enteraros de cómo acababa esto. Ya sea por h o por b, no os habéis enterado. Total, nadie tiene una Palm...

Y para acabar, un saludo muy grande a Látigo por abrirme los ojos a este tipo de crackeo (suerte en tu búsqueda de empleo, chico. Europa no anda mejor que tu tierra) y a toda la gente del canal #cracker del IRC-Hispano por aguantar mis locuras. Un saludo en especial a eSn-mIn por haberme mencionado en su tutorial de Asprotec. Eso no quiere decir que me olvide de Nette, remains, jonas, mrsilver, TGILITO, Sh0tGan, juassss, khanete, suNeVil, y miles de ellos que ahora olvido pero que me ayudan constantemente con su experiencia y paciencia.

30/05/2002

Y por nada del mundo olvidéis el mayor lema cracker: "Pensad, pensad, pensad...". Desde que descubrí esto del crackeo, me aburren los juegos de estrategia... y a ver si los que el día de mañana llegáis a ser programadores os dedicáis a hacer protecciones un poco serias, y no las cochinadas que hace el Jim Love éste.

X-Grimator