BANDERAS DE META

La aventura de un pequeño gorrión

AUTOLOG 1.3m

Lección 4 de Palm Reversing by X-Grimator

Buenas, mis queridos amigos. Como cada año por estas fechas nos enfrentamos de nuevo al torneo mundial de carreras de pájaros, donde este año ha sido el competidor revelación nuestro gorrión Fernandito, tres veces campeón regional y uno nacional. Durante un año, Fernandito se ha entrenado duramente estilizando sus plumas, fortaleciendo sus alas y puliendo poco a poco su técnica de vuelo. Durante un año, Fernandito sólo se ha alimentado de lombrices bajas en grasas, frutos secos y agua del rocío en medio de una espartana preparación hasta lograr que su entrenador, el azor Bertoldo, afirmase que ya sabía más que él.
Ahora es el momento y Fernandito sólo aspira a las BANDERAS DE META.

1- El principio:

Como sabemos, todos los principios son duros, así que para no haceroslo más duro aún, me voy a remitir a las lecciones anteriores para extraer nuestro código del programita AutoLogMetric.prc. En el supuesto de que alguien no haya podido leer los otros tutoriales por lo que fuese, hago un muy breve resumen:

a) Herramientas:

Vamos a necesitar el emulador de Palm, llamado PalmOs Emulator, un editor de ejecutables .prc cuyo nombre es PrcEdit 1.0, PilotDis para desensamblarlo y prc2bin para ayudar a dividir en trocitos legibles todo esto. En esta lección en concreto nos va a ser imprescindible el PalmDebugger, que de forma similar al emulador nos lo bajamos de la web de PalmOs.

b) Primeros vuelos de Fernandito:

Analizamos el ejecutable con PrcEdit y en las alarm miramos la número 7200 (1c20 hexa), que es la de Bad Boy. Hacemos uso de una de las mejores herramientas crack que existen, el bolígrafo, y lo apuntamos en algún lado (¡la vieja escuela cracker!. Es que X-Gri lleva mucho ya en esto... je je).
Metemos el ejecutable en el mismo directorio que prc2bin y desde la consola DOS tecleamos

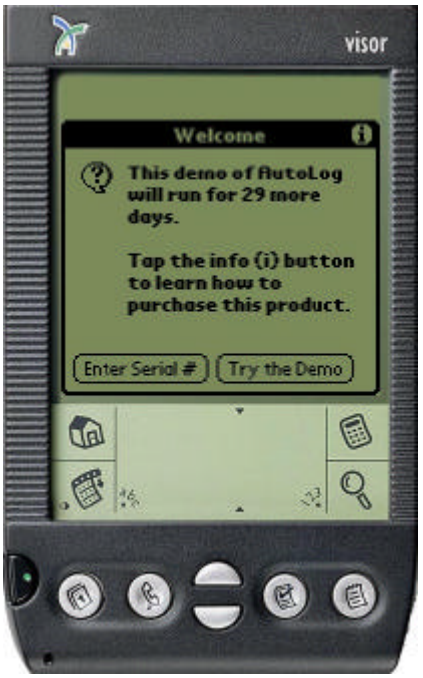
prc2bin autologmetric.prc

En el instante en que le demos a enter, una interminable colección de archivos aparecerá en ese directorio. Sólo nos interesa uno (bueno dos, pero el otro es sólo para confirmar algo). El interesante es code0001 (ummm los motivos son obvios) y Talt1c20 lo podemos abrir con un editor de texto para confirmar que es nuestro Bad Boy. Si aparece otro code, en seguida notaréis que no es el que nos interesa dado el tamaño tan pequeño que tiene (unos bytes apenas, que incluso en el diminuto mundo Palm es ya mucho). Tomamos ese code0001.bin y nos lo llevamos al directorio de PilotDis. Tecleamos en consola de MS-DOS pilotdis

pilotdis code0001.bin

y al darle a enter, ¡pummmm!, nos aparece un archivo llamado code0001.bin.s que puede leerse tranquilamente con el NotePad de Windows. Ojo, o cualquier editor de texto, no tiene que ser ese exactamente. Además, como somos crackers y odiamos a Microsoft, pues nos programamos uno en Delphi y así aprendemos un poco más.

Ya con eso, ejecutamos nuestro Emulador de Palm con nuestra ROM tal y como comenté en el último o penúltimo rollo que os metí y cargamos el programita. Tocamos con el ratón en el icono horrible del coche infantil, que da grima sólo verlo, y lo que nos encontramos en un programa tan inútil como éste y por el que pretenden cobrar tanto (como si fuésemos a crackearlo porque queremos usarlo, je, que ilusos, si esto sólo lo hacemos por aprender algo y no tragarnos la basura que echan en la televisión y que atrofia nuestros cerebros más que el vodkila) es lo siguiente:



Ayyyyyyy, esa avaricia de los programadores...¡¡todo por el dinero!!... apenas os diferenciáis de los políticos. Además si os fijáis, el primer botón es el de registrarse y luego el de probar. Eso es propaganda subliminal y está prohibida. Desastroso.

2- Levantando el vuelo. Primeros leñazos:

Fernandito ya está lo suficientemente preparado y nervioso. Con sus cicatrices de duro entrenar se pone en el nido de salida midiendo miradas con sus rivales. Sólo uno puede ganar. Los obstáculos son arduos, la carrera, extenuante, y la meta, sólo para uno. Eso ¡oh, sí!, hace arder la juventud en sus venas como un torrente de fuego (esto último suena a heroinómano XD).

Afila las garras de gorrioncito, pequeñitas ellas, y le da a Enter Serial # a ver que pasa. Dos cosas rondan su cabeza: el bad boy message y el código que tendrá que tragarse. Vamos allá. Lo que vemos es esto:

¿Ein?... maldición, ¿qué es eso que aparece debajo de nuestro nick?.
Fernandito se ha quedado de piedra al encontrarse en el nido, apenas a unos segundos del piar de salida, con un colibrí de grácil vuelo que hizo palidecer a la abubilla campeona del año pasado. Volar nervioso, ruido potente en el aleteo, pico aerodinámico... ¡santo Cristo, esto no me lo había advertido el entrenador?.

Mal empezamos cuando debajo de nuestro nick (apodo o pseudónimo, en español internacional) nos calcula un número para no se qué con nuestro serial. Uy, uy, uy que mareo me entraaaaaa... ya estoy pensando en llamadas a api que quizá nos hagan falta:

SysTrapDlkGetSyncInfo: llamada a api de Palm para leer nuestro user del HotSync (conexión con el Pc)

SysTrapFrmCustomAlert: aparición del mensaje bad boy.
SysTrapFldGetTextPtr: captura del string que acabamos de teclear.

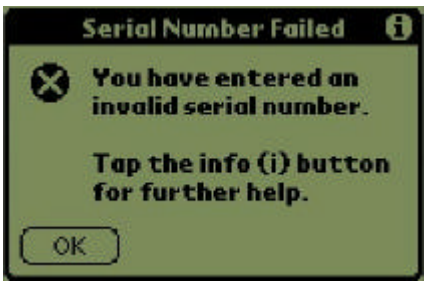
Ahí tenemos pues tres vías para empezar la carrera. Esto se avecina largo. Dos notas que os servirán de utilidad para futuros crackeos:

- FrmCustomAlert es muy similar a FrmAlert, sólo que de tres variables, muestra una en pantalla. A veces modificándolo un poco nos puede mostrar la segunda que puede ser ¡sorpresa! nuestro serial buscado.

- Otras llamadas a api interesantes en futuros crackeos pueden ser: StrCopy (copia string), StrCompare (compara strings, ésta es vital), FrmPopupForm (formularios popup). Poco a poco iremos descubriendo otras.
(NOTA PARA NEWBIES: "string" es una cadena de texto).

3- Piar de salida ¡comienza la carrera!... ¡pio!:

De acuerdo, metemos nuestro fake serial y nos encontramos con el mensajito de:



je, je, como si no lo supiéramos. Le damos a OK y volvemos a la ventanita del serial.
Fernandito sale volando en una explosión de potencia sacándole dos cuerpos a las demás aves, pero el colibrí no es malo y se le pone a una cabeza apenas. La grasa de sus plumas se pliega de tal forma que se asemeja a una bala de plata. Esos brillos del sol en sus ojos...

Vamos a abrir code0001.bin.s con el block de notas y buscaremos en la opción "Buscar" del NotePad lo siguiente:

$1c20

como veis, el procedimiento no varía nunca. Directos a ver dónde aparece el mensaje bad boy que vemos en la imagen de arriba. Para los newbies que se pierden ya a estas alturas, deciros que hemos buscado el valor hexadecimal del mensajito. Bien, pues lo vemos en

```
000060aa   3f3c1c20          MOVE.W      #7200!$1c20,-(A7)

000060fe   3f3c1c20          MOVE.W      #7200!$1c20,-(A7)

00006118   3f3c1c20          MOVE.W      #7200!$1c20,-(A7)
```

y, ya para nota, vemos el de good boy en

```
000060c8   3f3c1c84          MOVE.W      #7300!$1c84,-(A7)

000060e4   3f3c1c84          MOVE.W      #7300!$1c84,-(A7)
```

vaya, ya empezamos con varias llamadas. Aunque no me paré a mirarlo, no podemos olvidar que estamos con un trial de 30 días de prueba, así que no me extrañaría que el mensaje saliese en distintas líneas según la fecha en que nos registremos. Pero no deja de ser una suposición... Si nos fuéramos como locos al debugger y pusiésemos un breakpoint en SysTrapFrmCustomAlert, veríamos que no nos resolvería gran cosa, así que dejamos un momento de lado esta solución y nos vamos a ver qué ocurre con nuestro nick.

Mientras, Fernandito comienza a sudar. El colibrí intenta hacer juego sucio provocando turbulencias con el atronador batir de sus alas. Los demás rivales ya quedan varios metros atrás. Súbitamente una rama de abeto se interpone en su camino, pero curtido por entrenamientos castrenses en el gélido invierno, Fernandito inclina ligeremente sus plumas timoneras (o sea, las del culo XD) y en un suave pero seguro requiebro, logra zafarse de las agudas hojas del abeto esquivándolo por debajo. El colibrí, en un alarde de virtuosismo, inclina ligeramente su cuerpo y sobrepasa la rama por encima en un giro de noventa grados de su vertical, poniéndose de perfil. Un ensordecedor ruido estremece a los espectadores. El mirlo Mamberto, cae estrellado contra la rama rompiéndose un ala. Su caída en barrena hace gritar a un polluelo que contemplaba la carrera.

Pues a probar otro medio de ataque al programa. Vamos a ver qué ocurre si miramos el HotSync.
Volvemos a la pantalla de meter el serial y ejecutamos el PalmDebugger. Con el comando **att** conectamos el debugger (mirad que en connection esté seleccionado el emulador) y ponemos un breakpoint con **atb** de esta forma:

atb "DlkGetSyncInfo"

lo que ni corto ni perezoso nos hará llegar a

00005f78 4e4fa2a9 sysTrapDlkGetSyncInfo

Muy bien. Si escribimos en el debugger **il**, nos dará un listado de las próximas 10 instrucciones, lo que nos ayuda a localizar dónde andamos y apoyarnos con el listado muerto. Pasamos por encima de esa llamada a api con **t** para ejecutarla de golpe, y continuamos instrucción a instrucción con **s** a través del código.
Tras mucho mover, marear, girar, volver, revisar, saltos a subrutinas como SetFieldTextFromStr o MapHotSyncNameToPid, en las label 563 y 586 respectivamente y que el debugger se encarga de advertirnos, y tras muchos t y s según nos convenga, vemos en

00005fce 4fef000a LEA 10(A7),A7

que carga nuestro número de User Id, lo cual hemos averiguado con un

dm a7+a

ya que en hexadecimal, 10 se representa como a. Bueno, mucho tracear pero no hemos encontrado nada interesante que nos registre el programa. O si lo hay, al menos yo no lo he visto. Pero encontré otro camino más astuto para reventar este invento.

4- ¡Horror, enjambre de abejas!:

Luchando contra viento, ramas y el colibrí, Fernandito percibe algo extraño enfrente a él. En ese preciso instante, el cuervo Rave, antiguo campeón mundial, se adelanta a nuestro protagonista y a su rival en un picado acelerado y se sitúa a unos metros de ellos. El público pía atronador, salido de sus casillas y con furia. Bertoldo sonríe sin conocer su suerte. Apenas unos segundos más tarde, un enjambre de abejas, que se dibujaba como el extraño nubarrón que había visto Fernandito, se abalanza violento sobre el ex campeón, desestabilizando sus alas y llenándolo de dolorosos habones. Con una pata encogida, un ojo cerrado y vuelo resquebrado, el azor se retira al nido-boxes donde abandona la carrera. Pero este incidente dispersó a los insectos de forma caótica y Fernandito y el colibrí pueden pasar sin problemas ese obstáculo antes de que las abejas se reorganicen y eviten continuar la carrera a la abubilla que ostentaba el último título mundial, dos gaviotas, un petirrojo y un carrizo.

Pues nosotros también debemos meternos de lleno en ese nido de abejas. A ver, vamos a recapitular. Nos falla la búsqueda por bad boy message, nos falla el HotSync... ummm vamos a tener que usar el último recurso, la llamada a api que lee el texto que escribimos, FldGetTextPtr.

En el debugger tecleamos **atc** para borrar todos los breakpoints y lanzamos el programa de nuevo. En la ventana que pide el serial tecleamos en el debugger **att** para reconectarlo y

atb "FldGetTextPtr"

con esto lograremos parar la ejecución del programa justo en el momento de leer el número que tecleamos. Tras ello, tecleamos **g** para poder pasar del debugger a nuestro Emulador.

Metemos un fake serial, damos a OK y

00005e52 4e4fa139 sysTrapFldGetTextPtr

muy bien, un **t** en el debugger para ejecutar de golpe esa llamada a api y luego un **il** para poder seguir a la vez con el listado muerto. Nos interesa ese trozo de código, que es

```
00005e52  4e4fa139          sysTrapFldGetTextPtr
00005e56  2848              MOVEA.L     A0,A4
00005e58  3f3c1b61          MOVE.W      #7009!$1b61,-(A7)
00005e5c  2f03              MOVE.L      D3,-(A7)
00005e5e  4e4fa180          sysTrapFrmGetObjectIndex
00005e62  5c4f              ADDQ.W      #6,A7
00005e64  3f00              MOVE.W      D0,-(A7)
00005e66  2f03              MOVE.L      D3,-(A7)
00005e68  4e4fa183          sysTrapFrmGetObjectPtr
00005e6c  5c4f              ADDQ.W      #6,A7
00005e6e  2f08              MOVE.L      A0,-(A7)
00005e70  4e4fa139          sysTrapFldGetTextPtr
00005e74  2648              MOVEA.L     A0,A3
00005e76  2f0b              MOVE.L      A3,-(A7)
00005e78  2f0c              MOVE.L      A4,-(A7)
00005e7a  2f0a              MOVE.L      A2,-(A7)
00005e7c  4ebaff30          JSR         L594
```

os resalto en negrita las dos lecturas de texto que hace el programa y un salto a subrutina, que según el debugger se llama EnteredKeyValid, que ni que decir tiene que es un nombre evocador :o).

En 5e56 podemos hacer un **dm a0** que veremos nuestro fake serial como se copia de a0 a a4. Una línea más abajo carga la etiqueta de Field 1b61 que es la que PrcEdit nos dice que pone "User Id". No es importante, es un mero concepto de la parte de diseño del programa. Como no sé si os lo dije, recordad que tras escribir cualquier instrucción en el debugger, si queréis pasar al emulador, debéis teclear **g** ya que si no, no podréis pasar de uno a otro. De igual modo, esa orden os permite seguir hasta el siguiente breakpoint.
Bueno, llegamos al segundo FldGetTextPtr y tras pasarlo con un **t** seguimos mirando línea a línea de código, observando que en 5e76 y siguientes, tenemos en a4 el fake serial y en a3 el serial calculado para el nick, de manera que en 5e7c nos tira de cabeza a una subrutina que promete interesante.
O sea, mi fake y mi numero id en registros de dirección. Y en a2 un churro extraño que se repetirá más adelante. Cuando lleguemos a 5e7c, en vez saltarla con **t** vamos a entrar en ella. De continuar, en 5e86 (mirad vuestro listado muerto) saltaríamos a la L602 que nos conduce a

00005ee4 41fa006e LEA L607,A0

y que si miramos qué hay en a0 con **dm a0** en el debugger, nos encontramos con esto: "12180122" que es un número que si lo ponemos nos registra el programa para un solo uso, de manera que si salimos y volvemos a entrar ya no nos servirá más ese número comodín. Dios, los programadores están fatal de la cabeza... Cuando acabe el tutorial éste voy a estudiar mejor este tema y si se cuadra, hago un anexo. En 5eea tenemos una interesante llamada api que compara strings, en 5ef2 saltamos a 5f34, de ahí a 5f36 y 5f3a, que con el RTS de 5f40 nos dispara a 607a, que nos lleva al bad boy de 6118. Bueno, ese es el recorrido del programa para que podáis seguirlo en el listado muerto, pero de momento no nos interesa. Quede a título de referencia.
Por el momento nos vamos a meter de lleno en la subrutina esa de EnteredKeyValid, que parece interesante. Nos encontramos esto:

```
00005dae  4e560000          L594        LINK        A6,#0
00005db2  2f0a                          MOVE.L      A2,-(A7)
00005db4  2f03                          MOVE.L      D3,-(A7)
```

```
00005db6  246e0008                    MOVEA.L     8(A6),A2
00005dba  2f2e000c                    MOVE.L      12(A6),-(A7)
00005dbe  4e4fa0ce                    sysTrapStrAToI
00005dc2  2600                        MOVE.L      D0,D3
00005dc4  4ebafefe                    JSR         L583
00005dc8  4a00                        TST.B       D0
00005dca  584f                        ADDQ.W      #4,A7
00005dcc  6720                        BEQ         L596
00005dce  2f2e000c                    MOVE.L      12(A6),-(A7)
00005dd2  4e4fa0ce                    sysTrapStrAToI
00005dd6  2600                        MOVE.L      D0,D3
00005dd8  b6aa000a                    CMP.L       10(A2),D3
00005ddc  584f                        ADDQ.W      #4,A7
00005dde  650a                        BCS         L595
00005de0  b6aa000e                    CMP.L       14(A2),D3
00005de4  6204                        BHI         L595
00005de6  7001                        MOVEQ       #1,D0
00005de8  6016                        BRA         L598
00005dea  7000              L595      MOVEQ       #0,D0
00005dec  6012                        BRA         L598
00005dee  b6aa0012          L596      CMP.L       18(A2),D3
00005df2  650a                        BCS         L597
00005df4  b6aa0016                    CMP.L       22(A2),D3
00005df8  6204                        BHI         L597
00005dfa  7001                        MOVEQ       #1,D0
00005dfc  6002                        BRA         L598
00005dfe  7000              L597      MOVEQ       #0,D0
00005e00  261f              L598      MOVE.L      (A7)+,D3
00005e02  245f                        MOVEA.L     (A7)+,A2
00005e04  4e5e                        UNLK        A6
00005e06  4e75                        RTS
00005e08  8f                          DC.B        #143
00005e09  456e74657265644b6579        DC.B        'EnteredKeyValid'
00005e13  56616c6964
```

que nos hará interesantes llamadas a

```
0005cc4  4e560000           L583      LINK        A6,#0
00005cc8  2f0a                        MOVE.L      A2,-(A7)
00005cca  3f3c1f40                    MOVE.W      #8000!$1f40,-(A7)
00005cce  2f3c7445444b                MOVE.L #1950696523!$7445444b,-(A7)
00005cd4  4e4fa05f                    sysTrapDmGetResource
00005cd8  2448                        MOVEA.L     A0,A2
00005cda  200a                        MOVE.L      A2,D0
00005cdc  5c4f                        ADDQ.W      #6,A7
00005cde  670c                        BEQ         L584
00005ce0  2f0a                        MOVE.L      A2,-(A7)
00005ce2  4e4fa061                    sysTrapDmReleaseResource
00005ce6  7001                        MOVEQ       #1,D0
00005ce8  584f                        ADDQ.W      #4,A7
00005cea  6002                        BRA         L585
00005cec  7000              L584      MOVEQ       #0,D0
00005cee  245f              L585      MOVEA.L     (A7)+,A2
00005cf0  4e5e                        UNLK        A6
00005cf2  4e75                        RTS
00005cf4  95                          DC.B        #149
00005cf5  69734170706c69636174        DC.B        'isApplicationVirginal'
00005cff  696f6e56697267696e61
00005d09  6c
```

que es una subrutina dentro de ésta llamada IsApplicationVirginal, nombre que debería estar prohibido por sus connotaciones pornográficas... XD

Ciertamente, Fernandito consiguió salir de una buena maraña de abejas...

Entrando en la rutina EnteredKeyValid, llegamos a 5dbe, donde hay una llamada api que convierte nuestro string en número íntegro, cosa cierta si observamos que d0 toma como valor nuestro fake serial pasado a hexadecimal, y justo después, el salto a la aplicación virgen esa, que os señalé en 5dc4 en negrita. En esa rutina vemos cómo cargamos algo extrañísimo en 5cca y 5cce y como después nos marea mucho el registro d0 para que esté en cero y no en uno, incluso poniendo un salto en 5cea. Esto nos está haciendo pensar. Ummm, d0 no es igual si está a uno que si está a cero, ¿no?. A ver, seguimos con la ejecución de la rutina hasta volver a la de Entered etc, en 5dc8, que vuelve a testear d0. Vaya, vaya... salto si no es igual, comparación, salto, comparación y salto a L597. Bueno, esto se dice rápido, claro, pero en la práctica se traduce en muchas horas metiendo números en la Palm, traceando con el debugger, viendo qué caminos se toman... no penséis que esto es coser y cantar, ni de broma, pero en eso reside el encanto de este negocio. Además, sólo sudando se aprende (Dios, eso de la aplicación virgen ya me hace establecer comparaciones perniciosas...¿sudar, aprender?...¿qué es lo que aprendo sudando?... bufff....).
Ahora nos toca examinar el tramo final. Andiamo presto.

5- Recta final. En el horizonte, las banderas de meta:

La cosa se pone que arde. El público pía ensordecedor, una golondrina cae agotada, la cigüeña se retira con problemas de maniobrabilidad, dos palomas mensajeras son descalificadas por volar a cota no reglamentaria y tan sólo nuestro valiente Fernandito y su competidor el colibrí se aproximan a la meta. Ovaciones, tensión, esfuerzo. Se sacan mutuamente una cabeza, media cabeza, pugnando arrolladores por la bandera de meta ¿cómo acabaré éste lance?. Pasamos ahora mismo a verlo... el secreto está en LA BANDERA.

Situémonos. Habíamos repasado la ejecución del programa, nos habíamos metido en una rutina de nosequé de validkey y a su vez nos desviamos un poco a la rutina de una virgen, digooooooooo, a la rutina virgen XD. Si salimos de esta última (¿salir de una virgen?. En USA me meterían en la cárcel sólo por esta frase...) y volvemos a la rutina de EnteredkeyValid, nos topamos con este interesante pedazo de código:

```
00005dc8  4a00                    TST.B    D0
00005dca  584f                    ADDQ.W   #4,A7
00005dcc  6720                    BEQ      L596
00005dce  2f2e000c                MOVE.L   12(A6),-(A7)
00005dd2  4e4fa0ce                sysTrapStrAToI
00005dd6  2600                    MOVE.L   D0,D3
00005dd8  b6aa000a                CMP.L    10(A2),D3
00005ddc  584f                    ADDQ.W   #4,A7
00005dde  650a                    BCS      L595
00005de0  b6aa000e                CMP.L    14(A2),D3
00005de4  6204                    BHI      L595
00005de6  7001                    MOVEQ    #1,D0
00005de8  6016                    BRA      L598
00005dea  7000            L595    MOVEQ    #0,D0
00005dec  6012                    BRA      L598
00005dee  b6aa0012        L596    CMP.L    18(A2),D3
00005df2  650a                    BCS      L597
00005df4  b6aa0016                CMP.L    22(A2),D3
00005df8  6204                    BHI      L597
00005dfa  7001                    MOVEQ    #1,D0
00005dfc  6002                    BRA      L598
00005dfe  7000            L597    MOVEQ    #0,D0
00005e00  261f            L598    MOVE.L   (A7)+,D3
00005e02  245f                    MOVEA.L  (A7)+,A2
00005e04  4e5e                    UNLK     A6
00005e06  4e75                    RTS
```

El primer salto que he señalado en negrita lo hacemos queramos o no, ya que d0 lo compara con cero y así es. En la comparación de 5dee nos compara nuestro fake serial de d3 con algo que en apariencia no es nada y salta if carry set C (bcs), que no es el caso. Lo interesante viene ahora, en

```
00005df4  b6aa0016                CMP.L   22(A2),D3
00005df8  6204                    BHI     L597
```

ya que tras comparar mi fake serial con algo que seguimos sin saber qué es, me establece un salto branch if high, esto es, salta si es mayor que cero, salto que efectivamente se produce, y que nos lleva a

```
00005dfe  7000          L597      MOVEQ     #0,D0
00005e00  261f          L598      MOVE.L    (A7)+,D3
00005e02  245f                    MOVEA.L   (A7)+,A2
00005e04  4e5e                    UNLK    A6
00005e06  4e75                    RTS
```

que nos saca ya de la rutina NO SIN ANTES poner d0 con un valor de cero en L597. Pero ¿qué ocurriría si no hiciésemos el salto de 5df8?, pues que seguiríamos en

```
00005dfa  7001                    MOVEQ           #1,D0
00005dfc  6002                    BRA     L598
```

evitando el L597 y no sólo eso, si no moviendo el byte de d0 a uno. Y de ahí volvemos a la rutina principal que nos marea hasta el bad boy.
Es ahora cuando vuestra sagacidad cracker lo ha visto claro. Si d0 es cero el camino es uno, si es uno el camino es otro. El secreto, mis niños, está en la bandera: aprovechamos que el colibrí es daltónico y confunde la bandera con una mariposa. Fernandito tiene la carrera ganada.

A ver, hago un inciso para newbies. En toda esta milonga, hay una cosa a muy grandes rasgos que se llaman flag. No me quiero meter en el asunto porque excede este tutorial. Es igual a una señal de paso a nivel: si está subido puedes pasar y si no está, pues no puedes XD.
En este caso vemos que d0 se interesa especialmente por el valor que va a tener, si 1 o 0, e incluso al volver a la rutina principal veremos que en 5e80 nos vuelve a testear d0 y que de ello dependerá el salto de 5e86, y con ello el caminito a seguir hacia el bad o good boy.
Entonces, lo que se me ocurre, es cambiar la instrucción para que siempre tenga en d0 un uno. Para eso observamos la línea

```
00005dfe  7000          L597      MOVEQ     #0,D0
```

y la cambiamos por

```
00005dfe  7001          L597      MOVEQ     #1,D0
```

de forma que siempre nos dé d0 con valor uno. Para eso nos vamos a un editor hexadecimal, buscamos la secuencia de bytes (en este caso 7000) observando que los que hay antes y después son los de las instrucciones de antes y después de esta sentencia. Entonces es cuando ponemos **7001** donde pone **7000**.

Salimos del debugger, cargamos el nuevo programa parcheado en el Emulador, ponemos un fake serial cualquiera y al darle a OK

Pues de nada por purchasearte la basura de programa que no nos vale de nada.

6- Agradecimientos y notas:

Como siempre, llegamos a lo más divertido de los tutoriales. En éste debo hacer especial mención a tres personas, Crak3R Nuts, por su ayuda, Ziritione por haberse animado a empezar con crackeo de Palms, y SeRoCuLtO, por llevar más tiempo que yo aunque de forma más desapercibida y porque una vez se quejó de que no lo mencionaba en los tutos. Para este crack me ayudó mucho.

Ni que decir tiene, que siempre recuerdo a los chicos del irc-hispano en #crackers, como jonas_, seven, Horeb, Sunevil, Hadesh, uri2, eSn-mIn, daiamon, S-P-A-R-K, Eiji, DevilGun, ShotGan, cybdan, remains, y por supuesto y siempre Movsb. A él le debemos mucho los del canal ;o)

Y como notas, deciros que si tengo tiempo seguiré indagando en el programa. Me dejo en el tintero lo siguiente: comprobar el trial, sacar un número válido, ver qué es el número comodín. Si se terciase, os hago un anexo del tema.

X-Grimator, en algún lugar de Europa a 26/06/02

"Pensad, pensad, siempre pensad".