14.5 Ejemplos de scripts idc

Como siempre hemos dicho no hay mejor opción para aprender algo que ver ejemplos de lo estudiado. Por lo tanto vamos a ver distintos ejemplos de scripts IDC los cuales realizan distintas tareas específicas. Estos scripts presentados realizan distintas situaciones comunes de tareas sobre una base de datos.

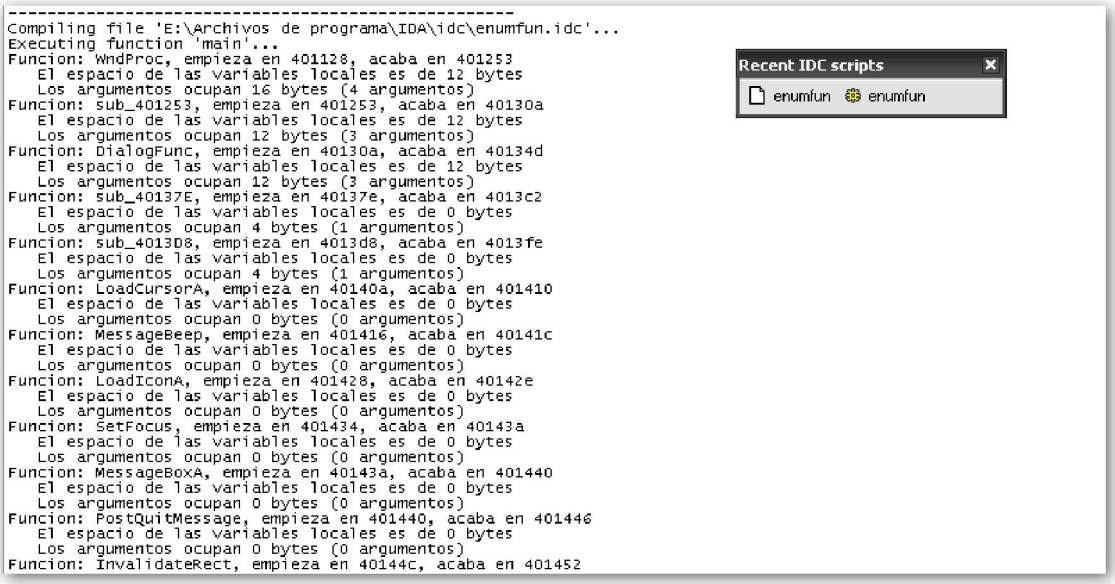
14.5.1.— Enumerar funciones

Muchos scripts operan con funciones individualmente. Ejemplos de ello son generar el árbol de llamadas a una función específica, generar el gráfico del control de flujo de una función o analizar el stack frame de cualquier función de una base de datos. El ejemplo siguiente itera a través de todas las funciones de una base de datos y nos imprime la información básica de cada función, como dirección de inicio y fin, el tamaño y cantidad de los argumentos y el tamaño de las variables locales de la función. Toda la información de salida se envía a la ventana de mensajes de IDA.

```
//El include obligado en cada script
#include <idc.idc>

//La funcion main obligada en cada script
static main() {

   //declaramos las variables
   auto direccion, final, argumentos, locales, estructura, primerArg, nombre, retorno;

   //inicializamos la variable dirección
   direccion = 0;

   //ejecutamos un bucle para que itere a todas las funciones de la base de datos
   //con la funcion NextFunction tomamos la direccion inicial de la siguiente funcion
   //si no es la última funcion itera a la siguiente función, si la es finaliza.
   for (direccion = NextFunction(direccion); direccion != BADADDR; direccion =
NextFunction(direccion)) {

      //Con la funcion Name nos proporciona el nombre de la funcion actual
      nombre = Name(direccion);

      //Con GetFunctionAttr nos da el atributo que queremos, en este caso con
      //FUNCATTR_END nos proporciona la dirección final de la funcion
      final = GetFunctionAttr(direccion, FUNCATTR_END);

      //la misma función anterior, pero con FUNCATTR_FRSIZE nos proporciona el
      //atributo del tamaño de las variables locales
      locales = GetFunctionAttr(direccion, FUNCATTR_FRSIZE);

      //Con GetFrame obtenemos el handle del stack frame de la funcion
      estructura = GetFrame(direccion);

      //Con GetMemberOffset determinamos el tamaño del stack frame
      retorno = GetMemberOffset(estructura, " r");

      //Desviacion de flujo, identifica si es función importada o no
      if (retorno == -1) continue;

      //El primer argumento de funcion se encuentra 4 bytes mas alla de la direccion
      //de retorno
      primerArg = retorno + 4;

      //Con GetStrucSize hallamos el tamano del stack frame, le restamos el primer
      //argumento para saber el espacio total que ocupan
      argumentos = GetStrucSize(estructura) - primerArg;

      //Nos imprime la información encontrada
      Message("Funcion: %s, empieza en %x, acaba en %x\n", nombre, direccion, final);
      Message("   El espacio de las variables locales es de %d bytes\n", locales);
      Message("   Los argumentos ocupan %d bytes (%d argumentos)\n", argumentos,
argumentos / 4);
   }
}
```

Este script utiliza algunas funciones IDC para manipular estructuras, con **GetFrame** obtenemos el **handle** del **stack frame** de cada función, con **GetStrucSize** determinamos el tamaño del **stack frame** pasándole la variable **estructura** donde tenemos el handle del stack frame de la función y restándole el primer argumento y con **GetMemberOffset** determinamos el **Offset** de la dirección, dentro de la estructura, donde está guardada la dirección de retorno de la función pasándole la variable **estructura** con el handle del stack frame de la función y la opción **r**. Hay que tener en cuenta que el primer argumento a la función se encuentra 4 byte más allá de la dirección donde está guardada la dirección de retorno, y que el tamaño del espacio de los argumentos de la función está calculado como el espacio entre el primer argumento y el final del stack frame. Debido a que IDA no puede generar el stack frame de funciones importadas, este script verifica si el stack frame de las funciones contiene la dirección de retorno guardada o no, como una forma para identificar las llamadas a funciones importadas. También hay que apuntar que el número de argumentos se halla dividiendo el tamaño total de estos por cuatro, en el script **argumentos / 4.**

Ahora lo guardamos como **enumfun.idc** en la carpeta de Ida, **idc**. Seguidamente cargamos nuestro **CRACKME.EXE** en Ida, y con la acción **File > IDC file** ejecutamos nuestro script **enumfun.idc** en la ventana de mensajes nos encontraremos con el siguiente listado de información:




```
------------------------------------------------------
Compiling file 'E:\Archivos de programa\IDA\idc\enumfun.idc'...
Executing function 'main'...
Funcion: WndProc, empieza en 401128, acaba en 401253
   El espacio de las variables locales es de 12 bytes
   Los argumentos ocupan 16 bytes (4 argumentos)
Funcion: sub_401253, empieza en 401253, acaba en 40130a
   El espacio de las variables locales es de 12 bytes
   Los argumentos ocupan 12 bytes (3 argumentos)
Funcion: DialogFunc, empieza en 40130a, acaba en 40134d
   El espacio de las variables locales es de 12 bytes
   Los argumentos ocupan 12 bytes (3 argumentos)
Funcion: sub_40137E, empieza en 40137e, acaba en 4013c2
   El espacio de las variables locales es de 0 bytes
   Los argumentos ocupan 4 bytes (1 argumentos)
Funcion: sub_4013D8, empieza en 4013d8, acaba en 4013fe
   El espacio de las variables locales es de 0 bytes
   Los argumentos ocupan 4 bytes (1 argumentos)
Funcion: LoadCursorA, empieza en 40140a, acaba en 401410
   El espacio de las variables locales es de 0 bytes
   Los argumentos ocupan 0 bytes (0 argumentos)
Funcion: MessageBeep, empieza en 401416, acaba en 40141c
   El espacio de las variables locales es de 0 bytes
   Los argumentos ocupan 0 bytes (0 argumentos)
Funcion: LoadIconA, empieza en 401428, acaba en 40142e
   El espacio de las variables locales es de 0 bytes
   Los argumentos ocupan 0 bytes (0 argumentos)
Funcion: SetFocus, empieza en 401434, acaba en 40143a
   El espacio de las variables locales es de 0 bytes
   Los argumentos ocupan 0 bytes (0 argumentos)
Funcion: MessageBoxA, empieza en 40143a, acaba en 401440
   El espacio de las variables locales es de 0 bytes
   Los argumentos ocupan 0 bytes (0 argumentos)
Funcion: PostQuitMessage, empieza en 401440, acaba en 401446
   El espacio de las variables locales es de 0 bytes
   Los argumentos ocupan 0 bytes (0 argumentos)
Funcion: InvalidateRect, empieza en 40144c, acaba en 401452
```

Performance Bigundill@