7.2.—Crear estructuras en IDA

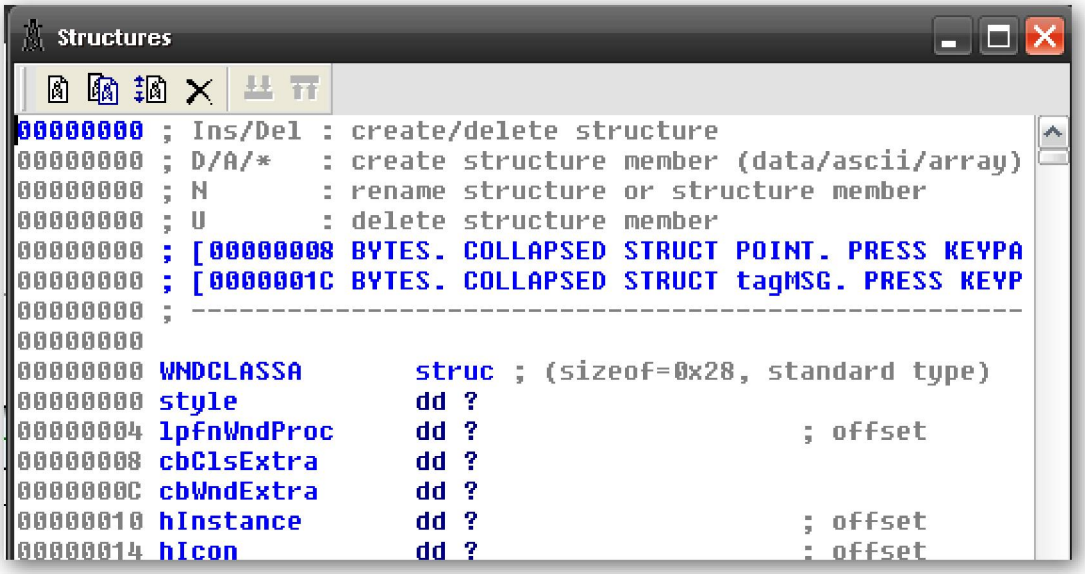
En la parte 6 estudiamos las características de IDA con respecto a la manipulación de conjuntos, las cuales nos permitían simplificar los listados de desensamblado reduciendo largas listas de declaraciones de datos en una sola línea de desensamblado. En estas siguientes secciones aprenderemos las facilidades que IDA nos proporciona para mejorar la comprensión del código que manipula en las estructuras. Nuestra meta es aprender a cambiar referencias de estructura como **[edx + 10]**, para que se nos muestre como por ejemplo **[edx + tora_struct.campo5]**.

7.2.1.—Realizar el esquema de una estructura de forma manual

Siempre que descubramos que un programa está manipulando una estructura de datos, tendremos que decidir si queremos incorporar los nombres de los campos de la estructura en el desensamblado, o saber a que corresponde cada Offset numérico que nos encontremos a lo largo del listado. En algunos casos, IDA podrá reconocer como se utiliza una estructura si está definida como parte de una **librery standar C** o del **Windows API**. En tales casos, IDA tendrá el conocimiento exacto del esquema de la estructura y será capaz de convertir los offset numéricos en nombres simbólicos de los campos. Este es el caso ideal, pero no siempre será así. De momento lo dejaremos aquí, ya volveremos a retomar esto una vez comprendamos un poco más sobre cómo IDA maneja las definiciones de estructura en general.

7.2.1.1.—Crear una nueva estructura (Create structure/Union)

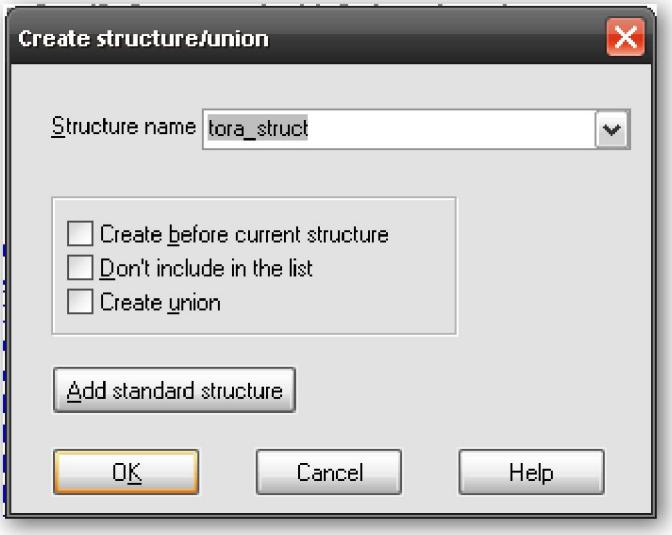
Cuando nos encontremos con un programa que utiliza una estructura de la cual IDA no tiene su esquema, IDA nos ofrece el poder saber la composición de la estructura y poderla especificar, y teniendola redefinida, incorporarla al desensamblado. La creación de una estructura se realiza en la ventana **Structures**, figura abajo. Ninguna estructura



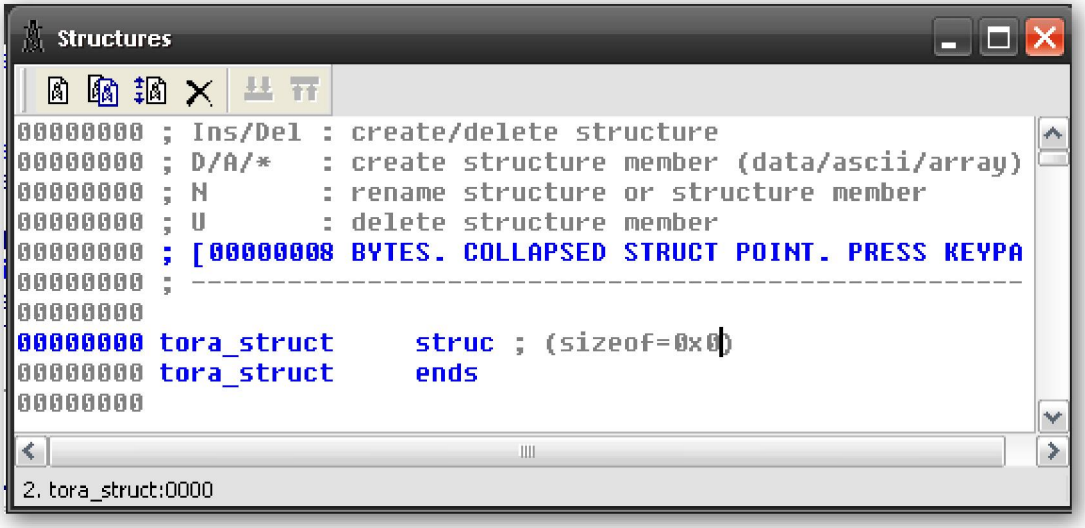
se podrá incorporar el desensamblado si nos está primero listada en la ventana **Structures**. Cualquier estructura tanto si es conocida o no por IDA, pero es reconocida para ser utilizada por un programa será automáticamente listada en la ventana **Structures**.

Existen dos razones por las que el uso de una estructura puede ser no reconocida durante la fase de análisis. En primer lugar, aunque IDA tenga conocimiento del esquema de una estructura en particular, puede ocurrir que no haya suficiente información para que IDA determine que el programa utiliza dicha estructura. En segundo lugar, la estructura puede ser fuera de lo "normal" con lo cual IDA no conoce nada de ella. En ambos casos el problema se puede superar y en ambos casos la solución empieza en la ventana **Structures**.

Como has podido observar las cuatro primeras líneas de texto de la ventana **Structures**, figura arriba, sirven como un constante recuerdo de las operaciones posibles en esta ventana. Las operaciones principales conciernen a añadir, quitar y editar estructuras. Para añadir una estructura se inicia utilizando la tecla **INSERT**, la cual nos abre un diálogo **Create Structure/Union** mostrado abajo.



A fin de crear una nueva estructura, deberemos primero especificar su nombre en el campo **Structure name**. La primera opción **Create before current structure** nos da la opción de, en dónde se mostrará la estructura, la segunda **Don't include in the list** si la seleccionamos no será mostrada la nueva estructura en la ventana **Structures**. La tercera opción, **Create union**, es para especificar si estamos definiendo una estructura **union** de estilo C o no. Una estructura **union** es similar a **struct**, en que esta puede estar constituida por muchos campos nombrados y cada uno puede diferir en su tipo. La diferencia entre las dos estriba en el hecho de que en una estructura **union** los campos se solapan unos con otros a fin de conseguir que sean todos del mismo tamaño, siendo el tamaño base el del campo más largo. Después del "kit-kat" teórico sigamos. En **structs** el tamaño es calculado con la suma del tamaño de cada campo, mientras que en **union**, el tamaño es calculado con el campo más grande. Finalmente el botón **Add standard structure** se utiliza para acceder a la lista de todas las estructuras de tipo de dato que IDA haya encontrado. El comportamiento particular de este botón lo estudiaremos, en una sección posterior. Una vez hayamos especificado un nombre de estructura y hagamos click en **OK**, una definición vacía de estructura se creará en la ventana **Structures**, ver figura abajo.

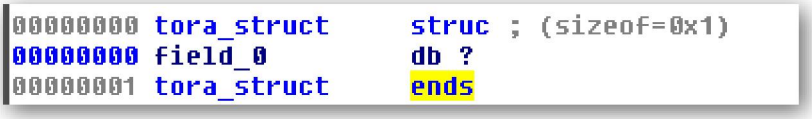


Ahora debemos editar esta definición de estructura para completar la definición del esquema de dicha estructura.

7.2.1.2.—Editar los elementos de una estructura

Vamos allá, para poder añadir campos en nuestra nueva estructura, debemos utilizar las órdenes **D**, **A** y **asterisco (*)** del teclado numérico. Inicialmente sólo podemos utilizar la órden **D**, el comportamiento de dicha órden dependerá de dónde esté ubicado el cursor. Por lo tanto, para añadir campos a una estructura es recomendable seguir estos pasos.

1. Para poder añadir un nuevo campo a una estructura, la posición del cursor tiene que estar en la última línea de la definición de estructura y seleccionarla, la que contiene **ends**, y pulsar **D.**



Esto nos produce un nuevo campo, que es añadido al final de la estructura. El tamaño del nuevo campo será habilitado de acuerdo con el primer tamaño seleccionado en el **data carousel** (Parte 6). El nombre de dicho campo será **field_N**, donde **N** es un offset numérico desde el inicio de la estructura al inicio del nuevo campo, por ejemplo **field_0**.

2. Si necesitamos modificar el tamaño del campo, primero nos tendremos que asegurar que el cursor esté posicionado en el nombre del nuevo campo y entonces seleccionar el tamaño correcto de dato pulsando repetidamente **D** de forma que realice el ciclo de tipos de dato del **data carousel**. Como alternativa, podemos realizar la acción **Options** > **Setup Data Types** para especificar un tamaño que no esté seleccionado en el **data carousel**. Si el campo es un **array**, haremos click derecho en el nombre y seleccionaremos **Array** para abrir el diálogo de especificaciones de array (Parte 6).

3. Para cambiar el nombre del campo de la estructura, hacemos click en el nombre del campo y utilizamos el atajo **N**, o click derecho en el nombre y seleccionamos **Name**; esto nos proporciona poder renombrar el campo.

Las siguientes ayudas pueden utilizarse para definir nuestras propias estructuras.

** El Offset de byte al campo se muestra como un valor hexa de ocho dígitos en el lado izquierdo de la ventana **Structures**.

** Cada vez que añadimos o quitamos un campo a la estructura o cambiamos el tamaño de un campo existente, el nuevo **sizeof** de la estructura se reflejará en la primera línea de la definición de la estructura.

** Podemos añadir comentarios al campo de una estructura de la misma forma que añadimos comentarios a cualquier línea de desensamblado. Lo haremos haciendo click derecho, o utilizando el atajo correspondiente, en el campo en el cual quieras añadir un comentario y seleccionando una de las opciones permitidas de comentarios.

** Contradiciendo las instrucciones de la parte superior de la ventana **Structures**, la instrucción **U** sólo borrará un campo de la estructura si éste es el último campo de la estructura. Para el resto de los campos, **U** solamente interpreta (undefine) el campo quitándole el nombre pero sin quitar los bytes distribuidos para el campo.

** Somos los responsables de la alineación apropiada de todos los campos dentro de la definición de la estructura. IDA no hace distinción entre estructuras comprimidas o no. Si necesitamos bytes de relleno para alinear correctamente los campos, somos los responsables de añadirlos. Los bytes de relleno es mejor añadirlos como **dummy fields** del tamaño apropiado, con lo cual podemos o no, elegir interpretarlos una vez los hemos añadido en los campos.

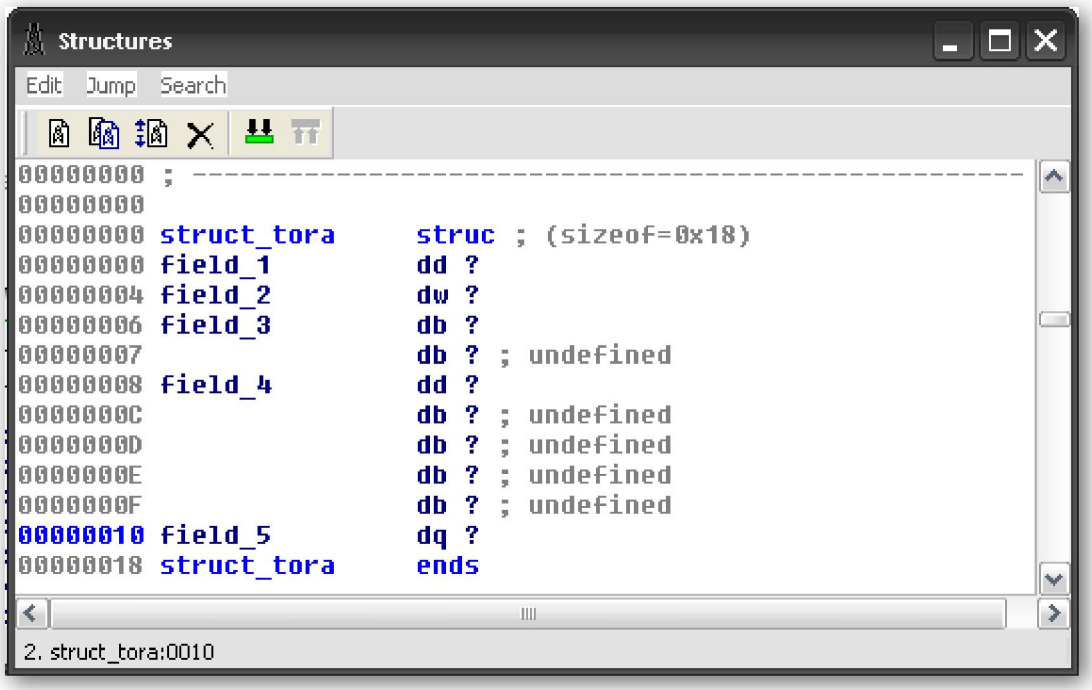
** Los bytes distribuidos dentro de la estructura se pueden quitar solamente si primero interpretamos (undefine) el campo y seguidamente realizamos las acción **Edit > Shrink Struct Type** lo cual quita los bytes interpretados.

** Los bytes pueden insertarse dentro de una estructura seleccionando el campo donde seguidamente se insertarán los bytes nuevos y realizando la acción **Edit > Expand Struct Type** con lo cual podremos insertar el número de bytes especificado antes de seleccionar el campo.

** Si conocemos el tamaño de la estructura pero no su esquema, necesitaremos crear dos campos. El primer campo será un array de **size-1** bytes. El segundo campo será un campo de **1-byte**. Después de haber creado el segundo campo, interpretamos el primer campo (array). El tamaño de la estructura se preservará, con lo cual podremos volver fácilmente más tarde a definir los campos y sus tamaños, más adelante aprenderemos más sobre esquemas de estructura.

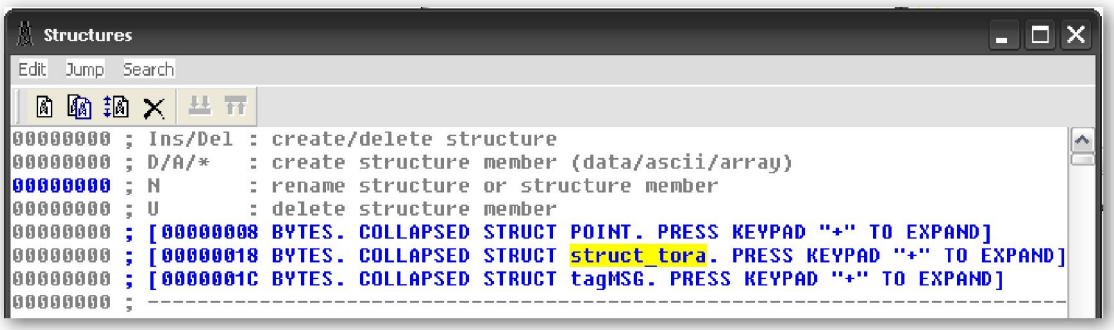
Para finalizar y comprobar las indicaciones anteriores, podemos ir aplicando todos los pasos explicados hasta aquí, añadir campo, establecer tamaño de campo, añadir relleno,

etc, con lo cual crearemos una estructura **struct_tora** (descomprimida) tal como vemos en la figura abajo.



En este ejemplo hemos incluido bytes de relleno para conseguir la alineación apropiada de campo, también hemos renombrado los nombres según los ejemplos utilizados anteriormente. Observa que los offset a cada campo y el tamaño total (24 bytes) de la estructura coincide con los mismos valores de los ejemplos anteriores.

Si en alguna ocasión ves que la definición de la estructura ocupa demasiado espacio en la ventana **Structures**, puedes encoger la definición de esta resumiéndola en una sola línea, seleccionando cada campo de la estructura y pulsando la tecla menos **(-)** del teclado numérico. Esto es útil realizarlo para que cuando una estructura que ya ha sido definida totalmente, se requiera una reedición de la misma, la versión encogida de **struct_tora** se muestra a continuación.



7.2.1.3.— Stack frame, una estructura especial

Puedes haberte dado cuenta, que las definiciones de estructuras tienen una vista parecida a la vista detallada del **stack frame** asociada a las funciones. Esto no es por casualidad, ya que internamente IDA maneja a ambos idénticamente. Ambos representan bloques de bytes contiguos que pueden ser subdivididos en componentes nombrados como campos, cada uno asociado con un offset numérico a la estructura. La diferencia, pequeña, entre ellos es que el stack frame utiliza offset de campos tanto negativos como positivos, centrados con el frame pointer o la dirección de retorno, mientras que las estructuras sólo utilizan offset positivos desde el principio de la estructura.