8.02 NumberGenericTableElements

We've taken a look at the initialization function and have a good idea of the base layout. Let's continue with what are probably the more simple functions to gather as much easy information as possible, then work on the more complicated functions. Let's start with `RtlNumberGenericTableElements`.

Think about how this function could work. One possible way is that the function will loop over the table. Another possibility is that the table contains an element that holds the number of elements in the table. Let's take a look.

Disassembly of `RtlNumberGenericTableElements`:

```
MOV EAX, DWORD PTR DS:[RCX + 0x24]
RET
```

This function is very simple but also very helpful. It takes one parameter which is a generic table. The function returns whatever is at offset +0x24 in the table. One slight issue though, +0x24 isn't an offset of 8 bytes. This means that we may be wrong about each member being 8 bytes. It may also mean that only a portion of a member is being accessed. Maybe some of the bytes contain information and the other bytes contain some other information or data.

It's usually safer to assume that this is a valid offset, and so there is a 4 byte element and not every element is 8 bytes. If we're wrong, I'm sure we'll find out.

Let's update our predicted table layout:

```
struct Table{
    UNKNOWN Member1;
    UNKNOWN_PTR Member2;
    UNKNOWN_PTR Member3;
    UNKNOWN_PTR Member4;
    UNKNOWN Member5;
    DWORD NumOfElements;
    UNKNOWN Member7;
    UNKNOWN Member8;
    UNKNOWN Member9;
};
```

<- Previous Lesson
Next Lesson ->

Chapter Home