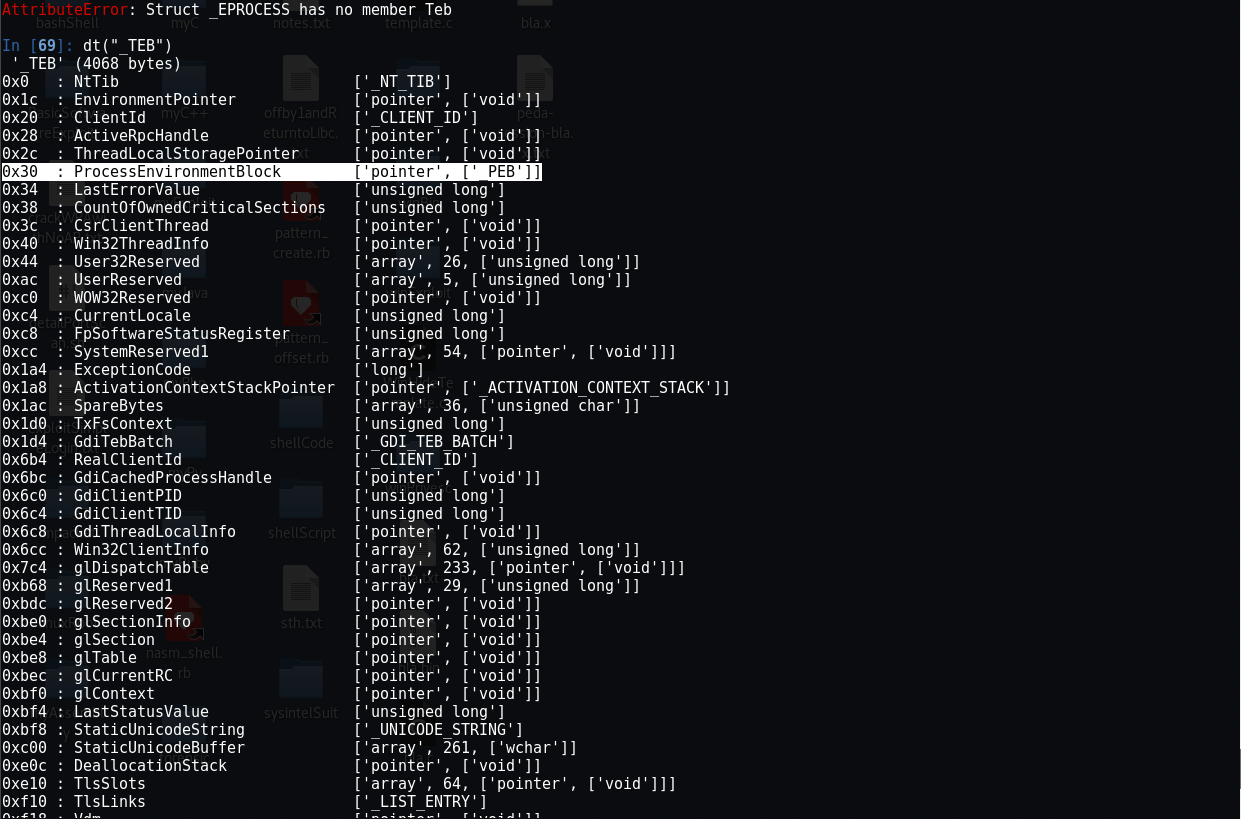
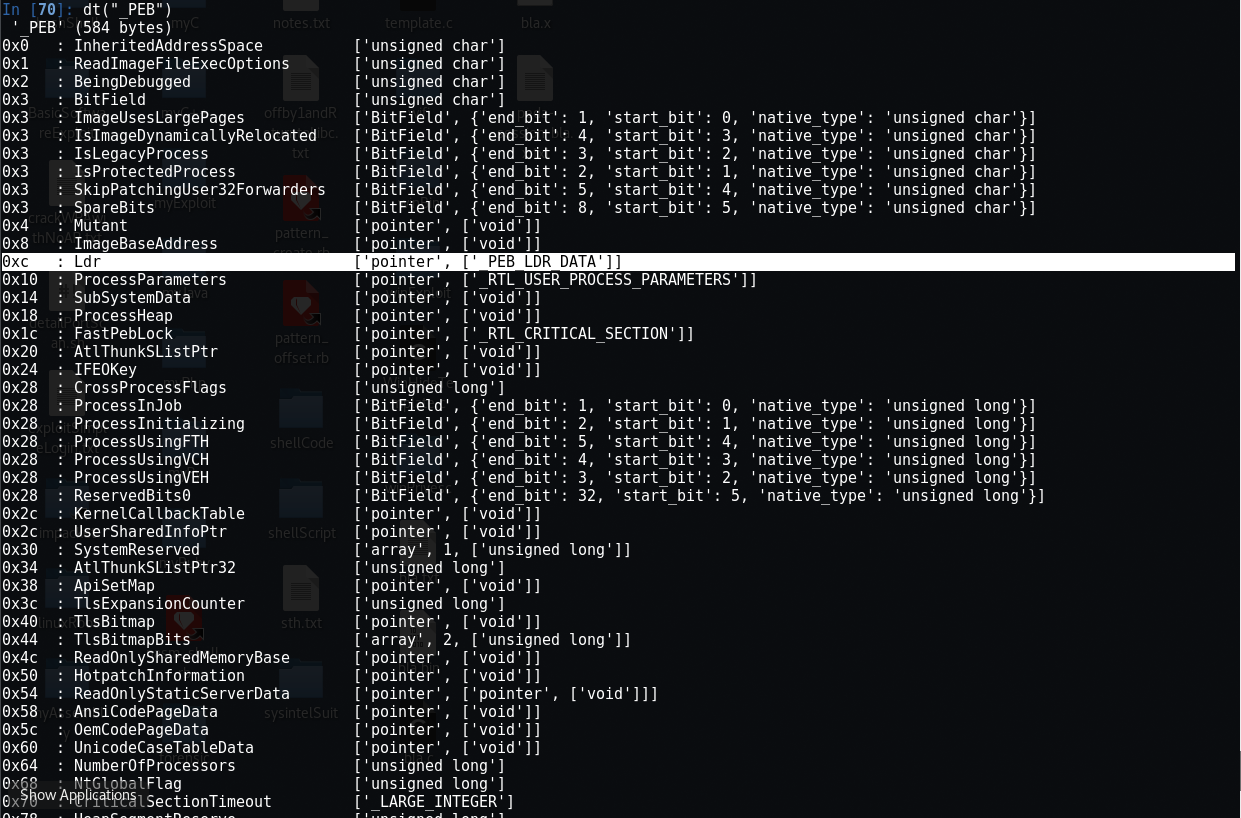
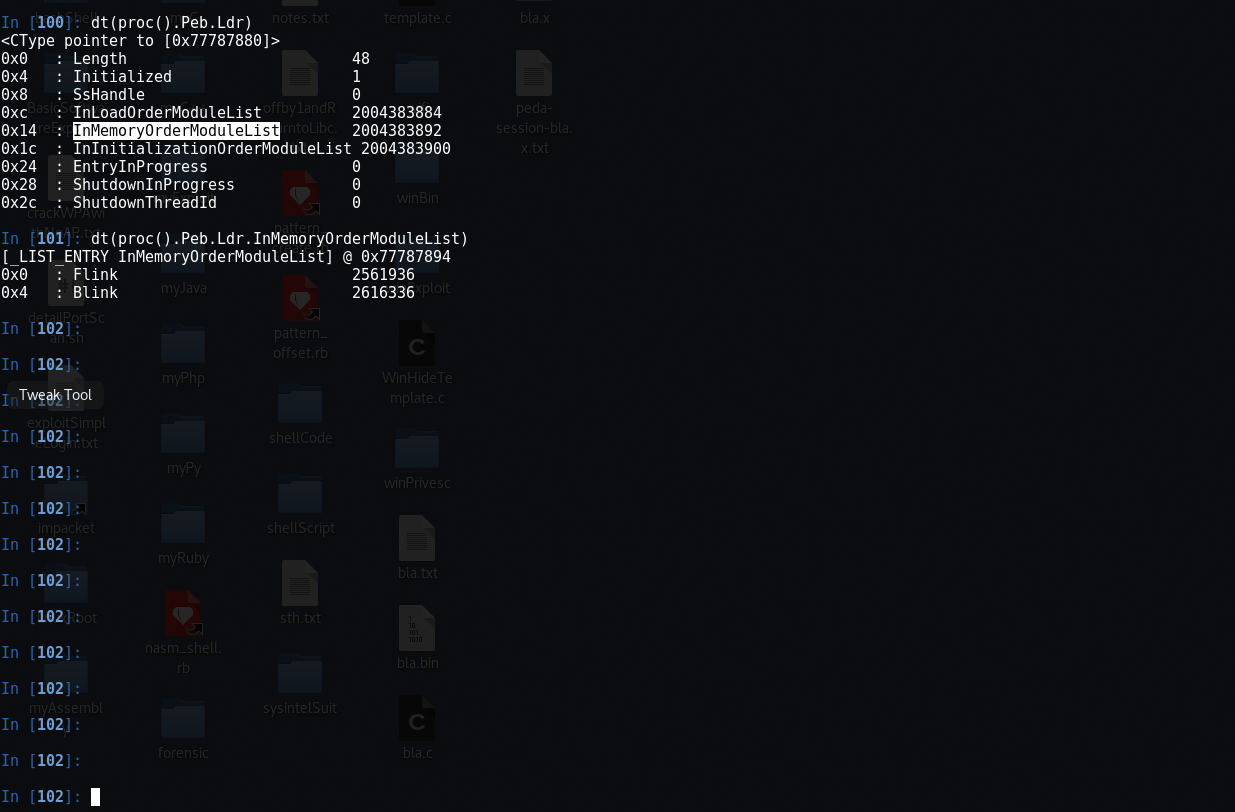
Finding kernel32.dll base



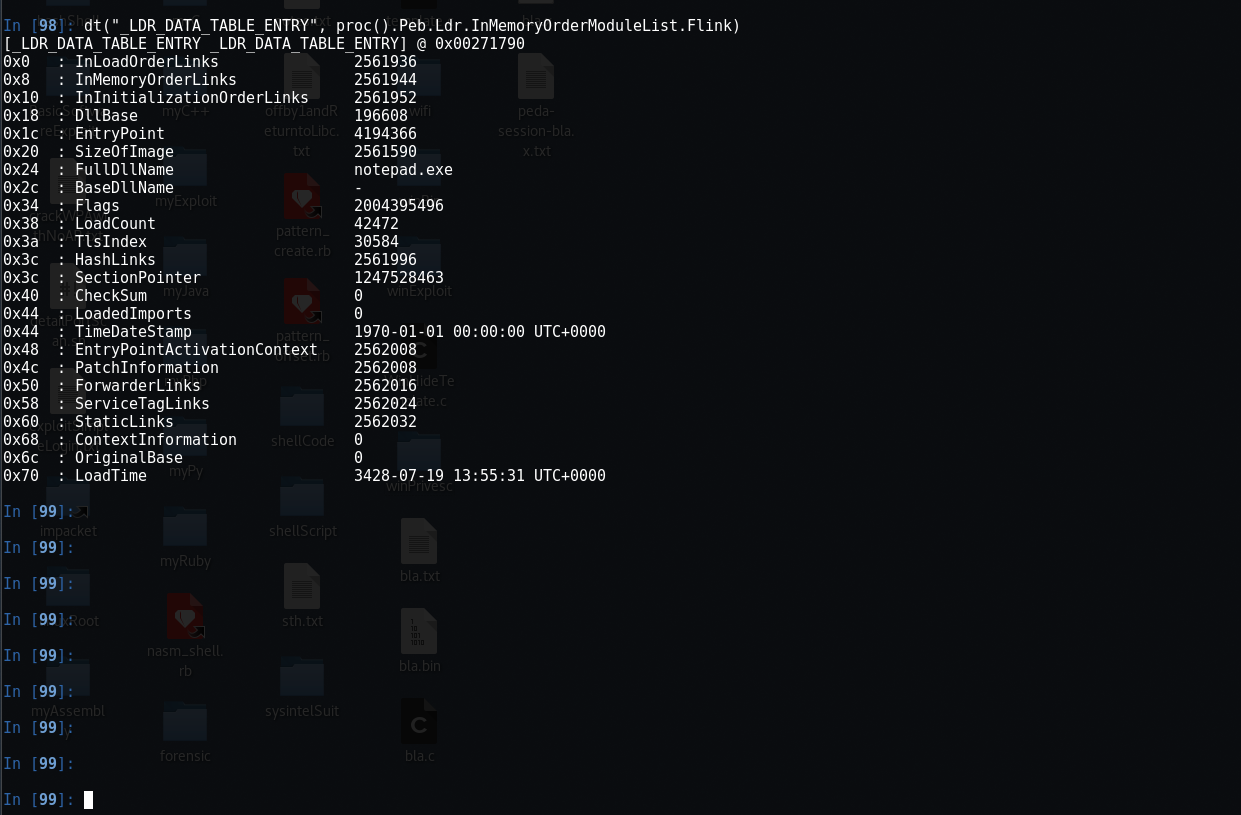
FS register stored the address of TEB, and at offset 0x30 is a pointer to Peb.



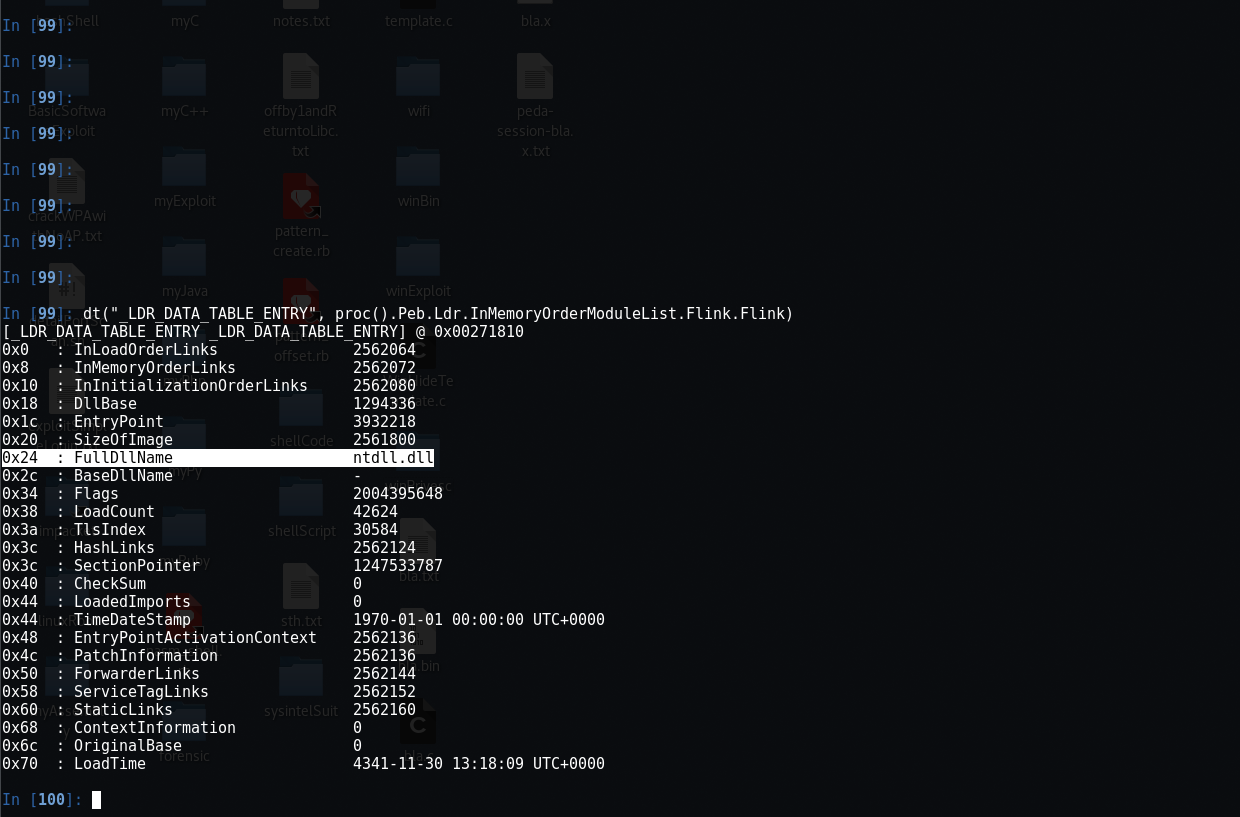
At offset 0xc of the Peb is Ldr, which is a pointer to \_PEB\_LDR\_DATA.



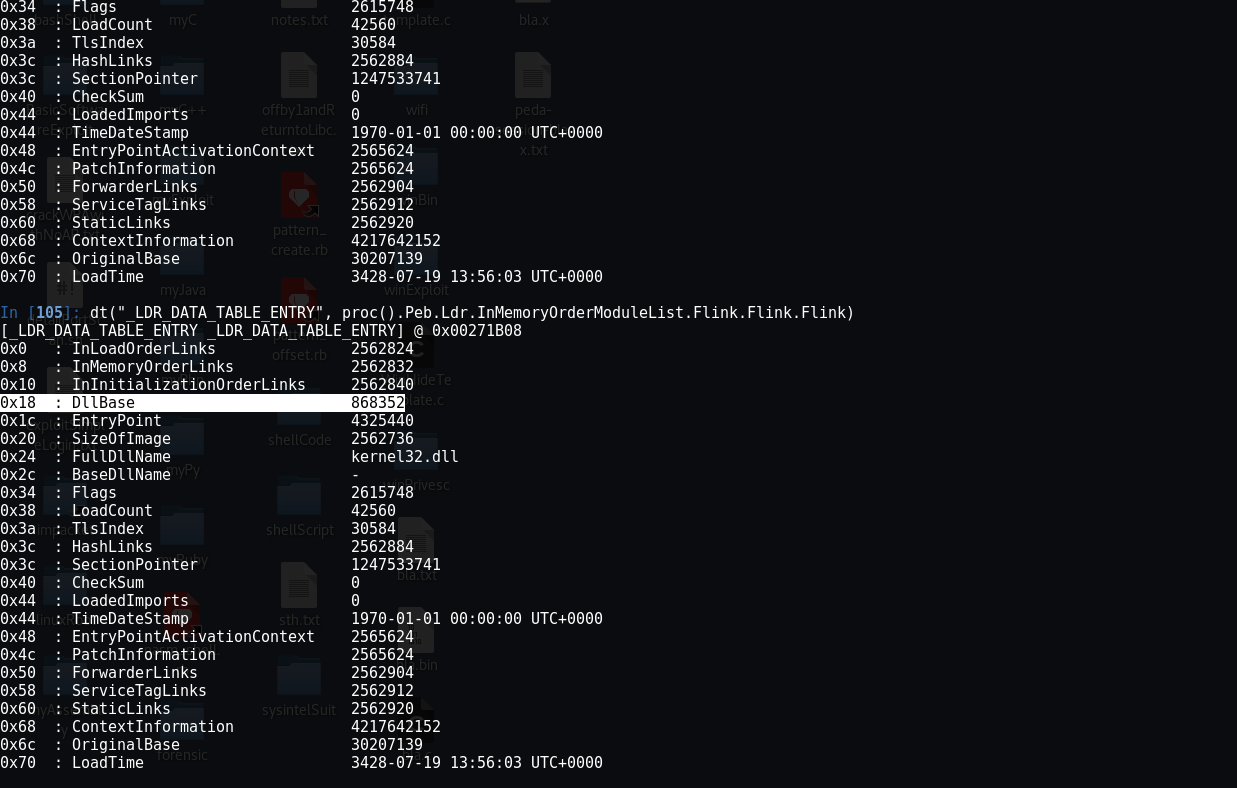
At offset 0x14 of \_PEB\_LDR\_DATA is InMemoryOrderModuleList, which is a pointer to one of the double link list that connects DLLs that are loaded into a process. Each DLL are stored in a \_LDR\_DATA\_TABLE\_ENTRY struct. The sequence of this DLL chain is based on the order of when DLLs are loaded into memory. So, the process image should be first, followed by ntdll.dll and kernel32.dll. Thanks to this reliable behavior, Windows shellcode writer can locate kernel32.dll, and later every proc and library that exist on the system.



Overlaying the \_LDR\_DATA\_TABLE\_ENTRY struct on this address of the first Flink, reveal the process image. In this case, its notepad.exe.



Ntdll.dll for the next Flink



And kernel32.dll for the next Flink. For the CPU, a DLL name doesn’t mean anything, but the base address (handle) does. This value is at offset 0x18. So if I was to translate the above explanation into shellcode, it would look like the following

xor ebx, ebx ;clear ebx

mov ebx, dword [fs:0x30] ;get TEB.Peb

mov ebx, dword [ebx + 0xc] ;get PEB.Ldr

mov ebx, dword [ebx + 14h] ;get LDR.InMemoryOrderModuleList

mov ebx, dword [ebx] ;get InMemoryOrderModuleList.Flink

mov ebx, dword [ebx] ;get InMemoryOrderModuleList.Flink.Flink

mov ebx, dword [ebx + 0x10] ;get ..Flink.Flink.Flink + 0x10 = DllBase

The reason why the last instruction is mov ebx, dword [ebx + 0x10] instead of mov ebx, dword [ebx + 0x18] is due to the position of the link list in the \_LDR\_DATA\_TABLE\_ENTRY struct. The InMemoryOrderLinks starts at offset 0x8 of the \_LDR\_DATA\_TABLE\_ENTRY, so it’ll need to add 0x10 in order to get to offset 0x18 of \_LDR\_DATA\_TABLE\_ENTRY struct.

Actually, according to The Art of Memory Forensic, using InLoadOrderModuleList may be a better candidate b/c

“InLoadOrderModuleList: A linked list that organizes modules in the order in which they are loaded into a process. Because the process executable is always first to load in the process address space, its entry is rst in this list. ” So the code can also be written as

xor ebx, ebx ;clear ebx

mov ebx, dword [fs:0x30] ;get TEB.Peb

mov ebx, dword [ebx + 0xc] ;get PEB.Ldr

mov ebx, dword [ebx + 0xc] ;get LDR.InLoadedOrderModuleList

mov ebx, dword [ebx] ;get InLoadedModuleLink.Flink

mov ebx, dword [ebx] ;get InLoadedModuleLink.Flink.Flink

mov ebx, dword [ebx + 0x18] ;get ..Flink.Flink.Flink + 0x10 = DllBase