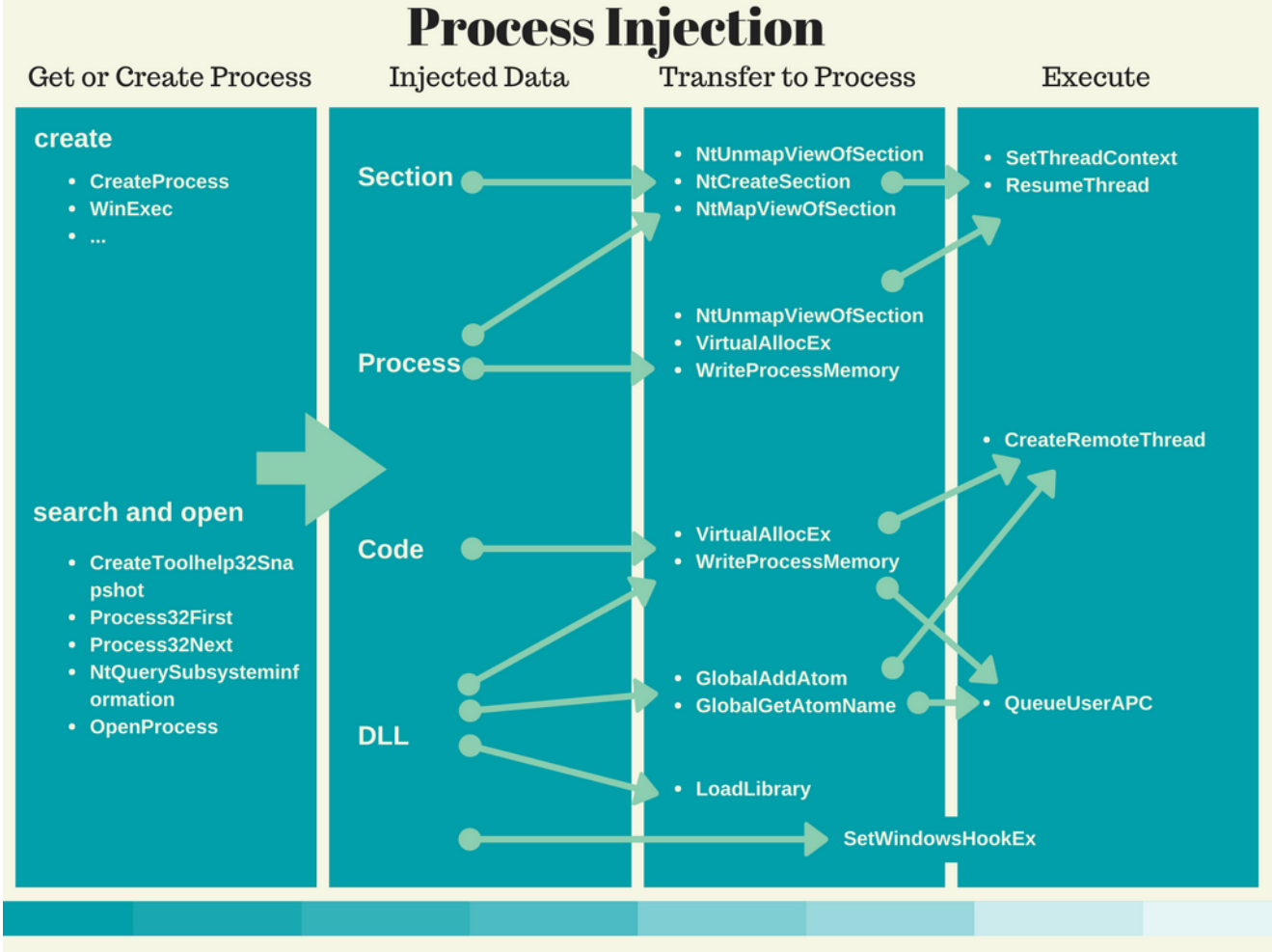
Process Injection Techniques

medium.com/@ozan.unal/process-injection-techniques-bc6396929740



This article contains an overview of what is DLL injection, Process Hollowing and Process Doppelgänging techniques.

Greetings, in this article we will look at what is **DLL injection** and what are the different techniques for this purpose. So let's start right away

First of all, let's find out what a DLL is before we enter the DLL injection. **DLL**(Dynamic Link Library)files are the common actions of the programs that are working in a single file, and if the program does not have the necessary functions during the program, it looks for it in the dynamic link library, namely in the DLL file. In this way, the programs will work on a single file, reducing memory usage to a minimum, increasing both the program speed and computer speed, since many DLLs use a single DLL instead of a file, it will not take up much space on the hard disk.

Now that we have learned what the DLL file does, there may be something in our minds about the DLL injection. Now we can start with the definition of DLL injection.

DLL injection is a method used by malware to hide, not attract attention or work with high rights. This method briefly aims to run the victim process with the rights of the victim by injecting harmful software into another process.

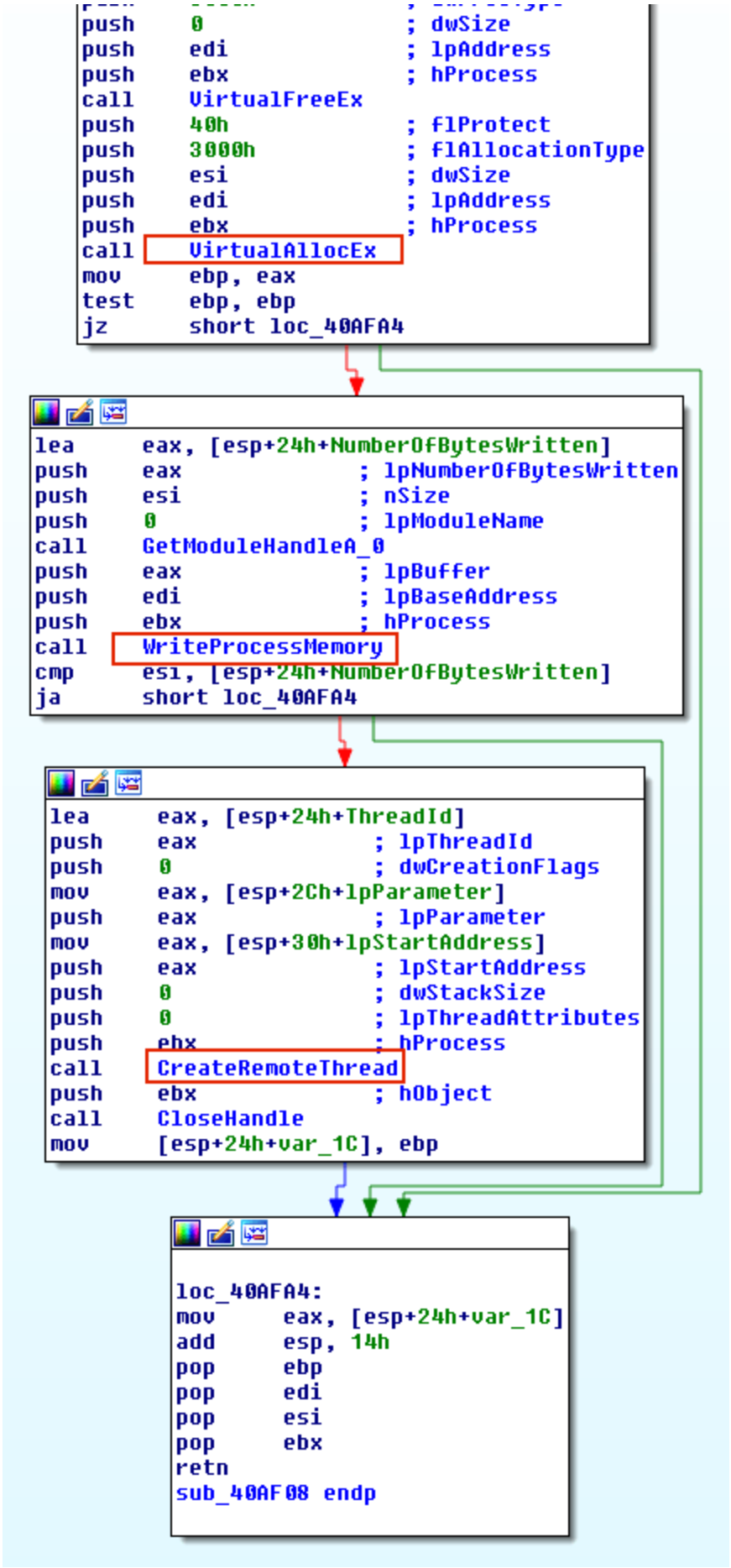
An example DLL injection steps include:

- First of all, a target must be determined for DLL injection. The most popular windows api that can be used for this process are **CreateToolhelp32Snapshot()**, **Process32First()** and **Process32Next ()** these functions take your process list and start all the processes from the beginning. Finally, the processes in this list are potentially exploitable processes.
- The process id (PID) of the specified process is passed to the **OpenProcess()** function to obtain a handle value to be used for that process access.
- A memory allocation is made within the memory area of the target process and the name of the malicious DLL that is desired to be loaded is written in this allocated area. The **VirtualAllocEx ()** and **WriteProcessMemory ()** functions are used for these operations, respectively. **VirtualAllocEx ()**, allocate memory in the memory area of the target process; **WriteProcessMemory ()** is also used to write the name of the malicious DLL in this allocated field.
- Finally, to have the code executed in another process, the malware calls APIs such as **CreateRemoteThread**, **NtCreateThreadEx**, or **RtlCreateUserThread**. The latter two are undocumented. However, the general idea is to pass the address of LoadLibrary to one of these APIs so that a remote process has to execute the DLL on behalf of the malware.

NOTE: In order to run the "**CreateRemoteThread()"** function, the target (victim) process and the harmful process must be the same process owner.

It is worth mentioning that the "**CreateRemoteThread"** function is monitored by many security products. Attackers therefore avoid using it.

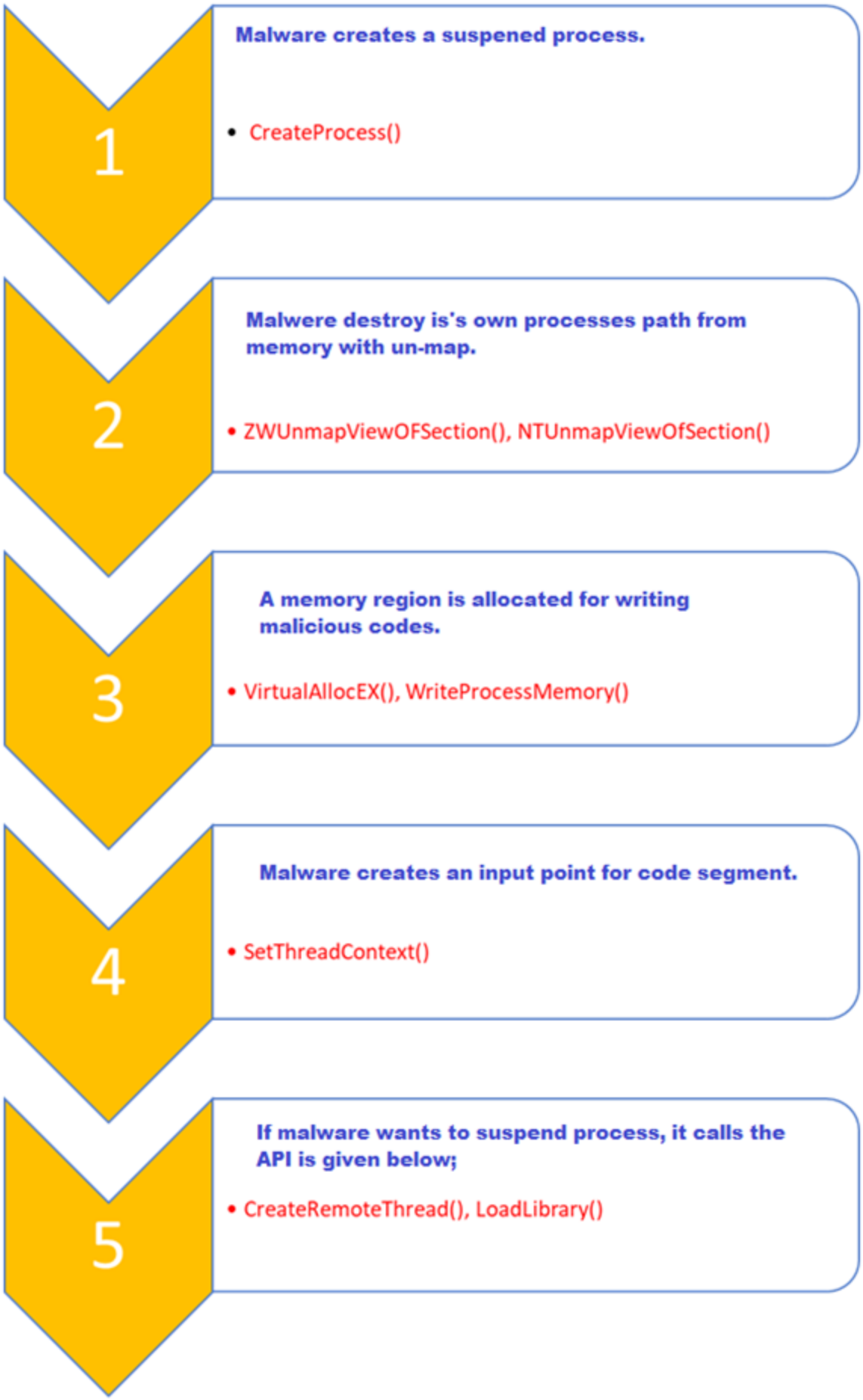
The screenshot below shows a malware called Rebhip that performs this technique.


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

Now that we have seen what DLL injection is and how it is done, we can look at different techniques and subfields.

Process Hollowing

The method called Process Hollowing is slightly different from the method we have described above. The malicious code that uses this method first runs a valid system process, then stops that process, and then replaces its codes with the original codes of that process, and finally runs the code it stopped and settled in the system.


1
Malware creates a suspened process.
• CreateProcess()
2
Malwere destroy is's own processes path from memory with un-map.
• ZWUnmapViewOFSection(), NTUnmapViewOfSection()
3
A memory region is allocated for writing malicious codes.
• VirtualAllocEX(), WriteProcessMemory()
4
Malware creates an input point for code segment.
• SetThreadContext()
5
If malware wants to suspend process, it calls the API is given below;
• CreateRemoteThread(), LoadLibrary()

Using the methods I described above, it is possible to easily detect harmful codes placed in the system by performing memory analysis. It will be enough for the memory analysis software that we will use to read the **VAD (Virtual Address Descriptors)** structure for each process and find the memory sections marked as executable **(Page_Execute_ReadWrite)** and then check if the codes in these memory sections have a corresponding response on the disk. You can easily identify Process Hollowing via the memory image. For this we can find out which processes are injected with volatility or Mandiant RedLine.

Let's look at how we can detect it from a sample memory image. The analysis will start from the injected process detection. We can detect them with **Volatility malfind.**

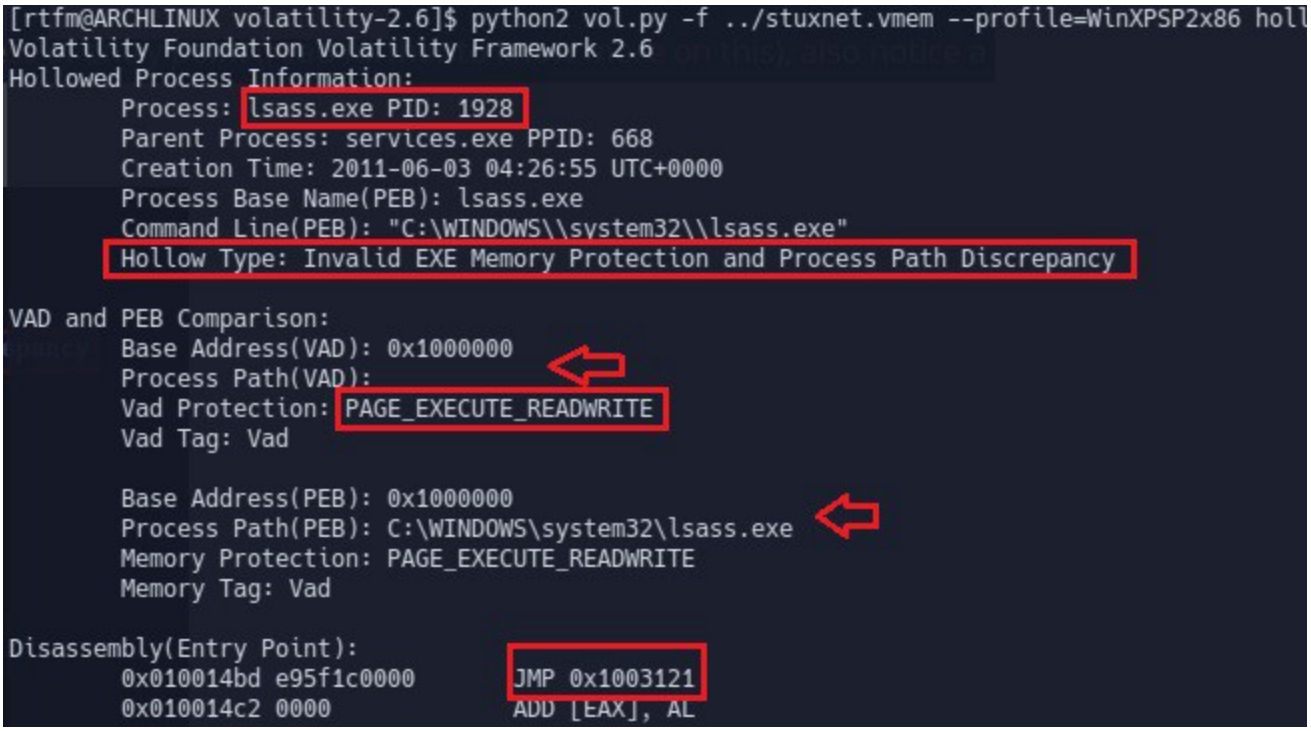
```
Process: lsass.exe Pid: 868 Address: 0x80000
Vad Tag: Vad  Protection: PAGE_EXECUTE_READWRITE
Flags: Protection: 6

0x00080000  4d 5a 90 00 03 00 00 00 04 00 00 00 ff ff 00 00   MZ..............
0x00080010  b8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00   ........@.......
0x00080020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0x00080030  00 00 00 00 00 00 00 00 00 00 00 00 08 01 00 00   ................
```

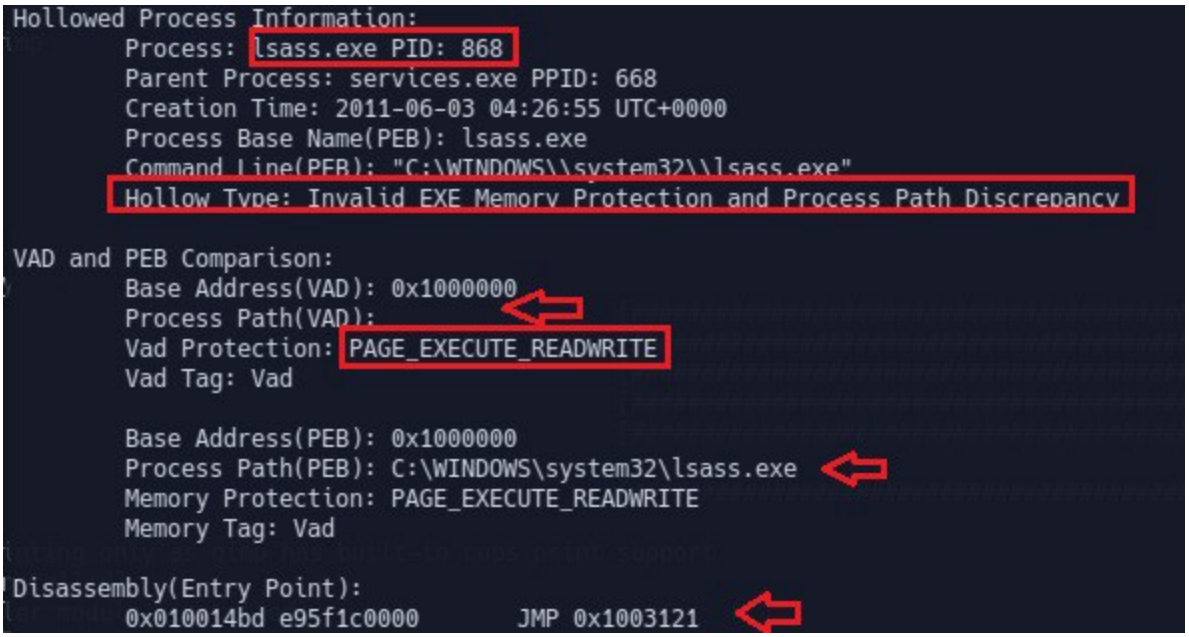
Hollow process injection can also be detected by looking for suspicious memory protection. Running the malfind plugin (which looks for suspicious memory protections) shows suspicious memory protection **(PAGE_EXECUTE_READWRITE)** at address 0x1000000 (which is base address of lsass.exe) indicating that lsass.exe was not loaded normally (but was injected). Any executable that is normally loaded will have a memory protection of **PAGE_EXECUTE_WRITECOPY.** This further confirms that lsass.exe (pid 868) loaded at 0x1000000 is not legitimate.

Automating Process Hollow Detection using HollowFind Plugin

> **Hollowfind** is a Volatility plugin to detect different types of process hollowing techniques used in the wild to bypass, confuse, deflect and divert the forensic analysis techniques. The plugin detects such attacks by finding discrepancy in the **VAD** and **PEB**, it also disassembles the address of entry point to detect any redirection attempts and also reports any suspicious memory regions which should help in detecting any injected code.

Below screenshot shows **hollowfind** plugin in action. Running the hollowfind plugin on the stuxnet infected memory image identified both **lsass.exe** processes (**pid 1928 and pid 868**) and it also reports the the invalid exe memory protection (**PAGE_EXECUTE_READWRITE**) and process path discrepancy between the **VAD and PEB** and also it disassembles the address of entry point (read further to know more on this), also notice a jump to the address 0x1003121 at the address of entry point.





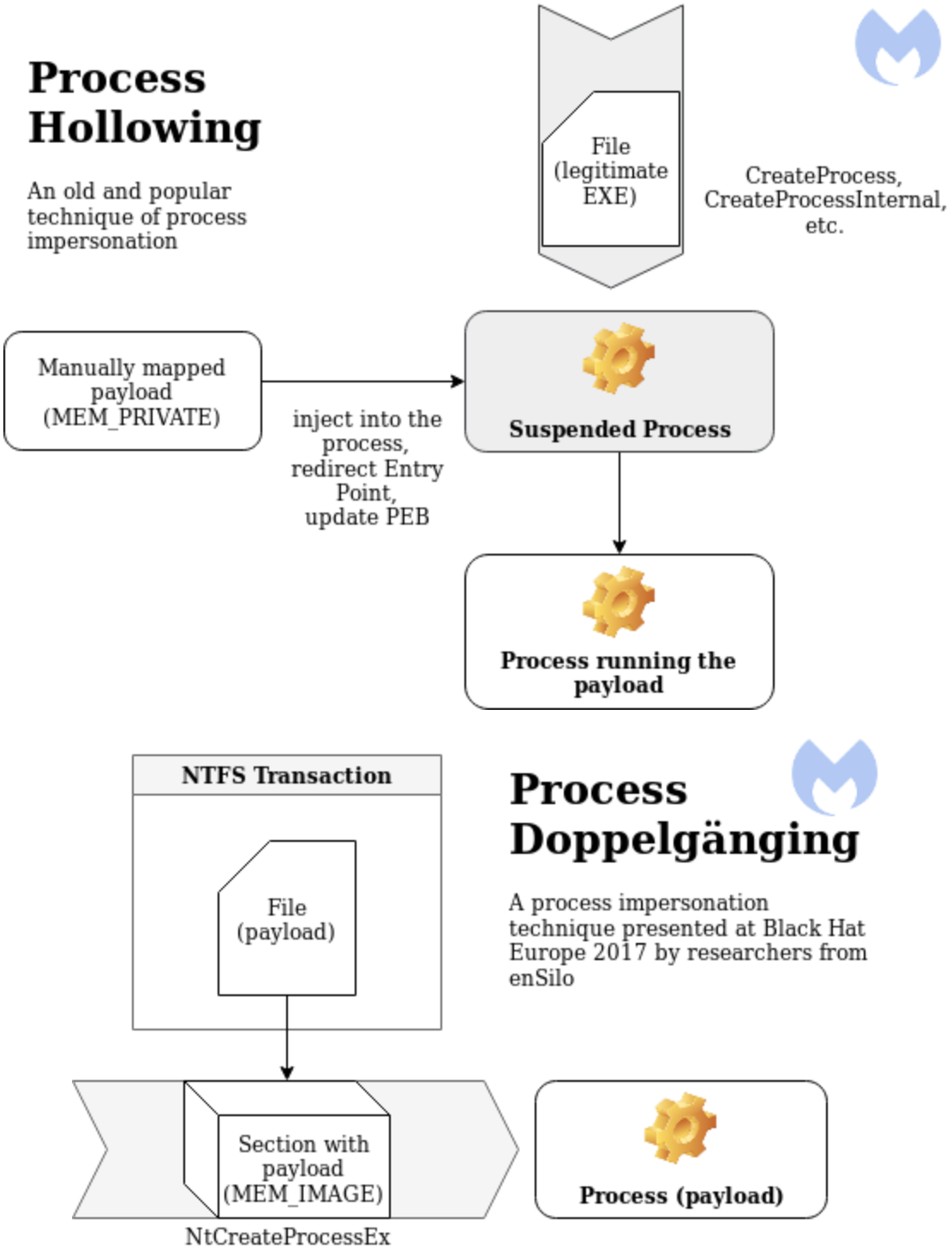
For a more detailed analysis, https://cysinfo.com/detecting-deceptive-hollowing-techniques/

Process Doppelgänging

Process Doppelgänging, one of the popular Code Injection techniques, was first announced by 2 security researchers working in enSilo company in BlackHat in 2017.

Process Doppelgänging is of great importance because it works successfully on all versions of Windows, including Windows 10. If it is similar to Process Hollowing, it has certain aspects that are strictly separated from Process Hollowing.

Process Doppelgänging was frequently used by malware as it was difficult to detect by many AV products when it first appeared.



Thanks to malwarebytes for the images :)

Process Hollowing first initiates the target process, then unmaps and injects the malicious code. Process Doppelgänging, on the other hand, writes the malicious code on the image before the process starts. This is actually the biggest difference between them.

Process Doppelgänging is implemented in 4 steps:

- **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.

As a result, the process can start in an injected state even after the contents of the file are undone. For this reason, it will appear that there are no problems by many AV products.

References

Ten process injection techniques: A technical survey of common and trending process injection…

Process injection is a widespread defense evasion technique employed often within malware and fileless adversary…

www.elastic.co

Process Doppelgänging

Windows Transactional NTFS (TxF) was introduced in Vista as a method to perform safe file operations. To ensure data…

attack.mitre.org

Detecting Deceptive Process Hollowing Techniques Using HollowFind Volatility Plugin

In this blog post we will look at different types of process hollowing techniques used in the wild to bypass, confuse…

cysinfo.com

Process Doppelgänging

Selamlar herkese. Bu yazıda PE injection tekniklerinden biri olan Process Doppelgänging'in teknik detaylarından…

kaganisildak.com

Process Doppelgänging meets Process Hollowing in Osiris dropper

Process doppleganging, a rare technique of impersonating a process, was discovered last year, but hasn't been seen much…

blog.malwarebytes.com

http://halilozturkci.com/adli-bilisim-zararli-kod-analizinde-dll-injection-ve-process-hollowing-tespiti/

Understanding And Detecting Dll 1nj3ct0n & Process Hollowing

Hello friends, in this blog post I will wirte about process injection, process hollowing and dll injection. I give some…

medium.com

https://www.blackhat.com/docs/eu-17/materials/eu-17-Liberman-Lost-In-Transaction-Process-Doppelganging.pdf

Ozan Unal

https://twitter.com/ozanunll

More From Medium