CS 373 Summer 2019

Week 5 Write Up

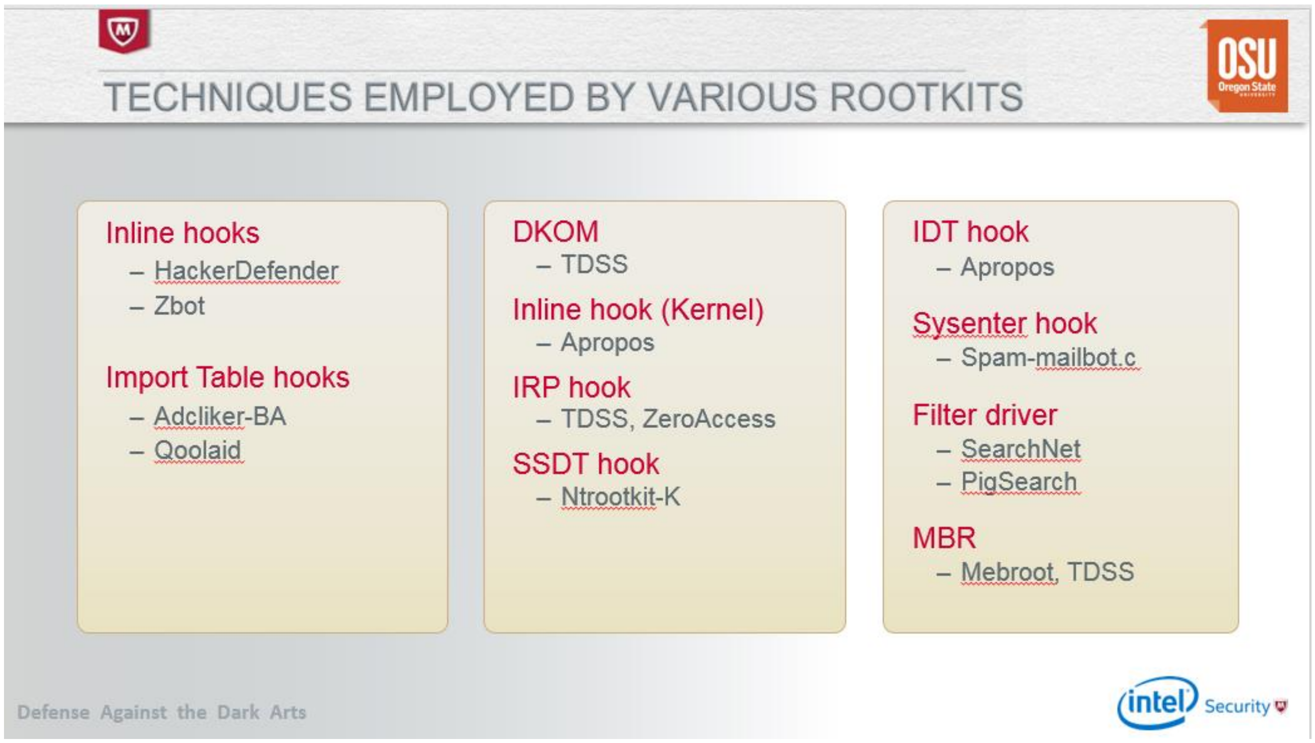
Kirsten Wollam

WINDOWS MEMORY MANIPULATION

In order to stick around in the attacked system and continue to function, malware creators have included ways for the malicious code to persist. One of the main ways to achieve this is to hide the code on the system so that it is not detected. This is often achieved using root kits, malware that actively conceals its existence from users and system processes. Root kits interrupt regular program flows and use them to hide undetected. This act is known as hooking. The root kit hijacks a normal process to execute malicious code then returns to the process, so it finishes as normal. Because the process finishes as normal, the user does not notice that there was anything unusual happening.

Root kits usually target the kernel files on the system. These are .sys files and they are the heart of the operating system. They provide basic services called by all the other parts of the OS and are the bridge between the OS and hardware. Process and memory management, file systems, networking, and device control all utilize the kernel. Because of the role they play, kernel files must have significant access on the system. This access makes them a great place to put malware.

With the release of 64-bit windows, it became much more difficult for rootkits to target the kernel, because there were many additional safety features. In particular they have to get around driver signing checks and secure boot. There are ways around these and evil forces are always out there continuing to make newer and sneakier rootkits.



TECHNIQUES EMPLOYED BY VARIOUS ROOTKITS

Inline hooks
- HackerDefender
- Zbot

Import Table hooks
- Adcliker-BA
- Qoolaid

DKOM
- TDSS

Inline hook (Kernel)
- Apropos

IRP hook
- TDSS, ZeroAccess

SSDT hook
- Ntrootkit-K

IDT hook
- Apropos

Sysenter hook
- Spam-mailbot.c

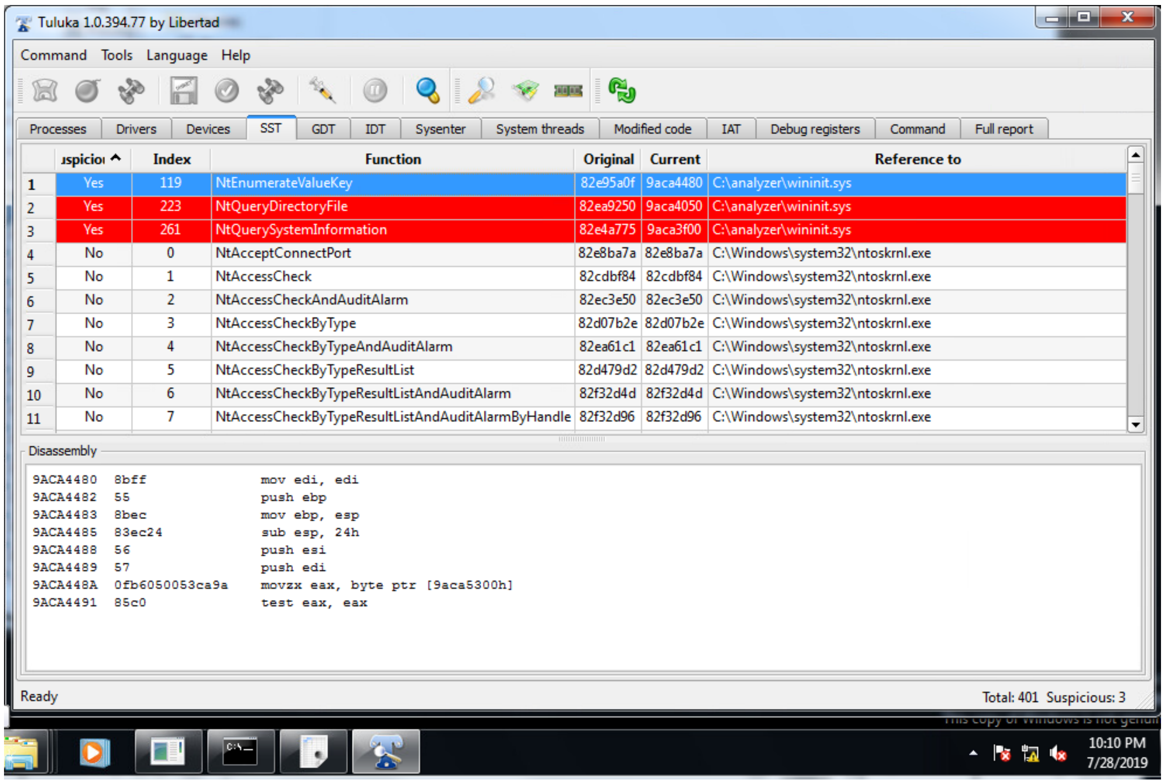
Filter driver
- SearchNet
- PigSearch

MBR
- Mebroot, TDSS

Defense Against the Dark Arts

Figure 1- windows memory manipulation slide 36

There are many different techniques that root kits use. One of those is to target the System service description table which lists all the system level api available to use in a table. What is actually stored in the table is a pointer to the api. To hook into that process the malware stores the pointer it is targeting, and then swaps it out for a pointer to the malware. once the malware is executed it calls the original pointer so that the api that was called runs. The user or systems are likely to not notice at all that there was any extra code run because the code they were trying to run did run properly. One way to discover this sort of malware is that the new pointer in the table will not point to a location in the windows kernel memory or by running a tool like Tuluka which will analyze these pointers are display ones that are potentially suspicious.


Tuluka 1.0.394.77 by Libertad
Command Tools Language Help
Processes Drivers Devices SST GDT IDT Sysenter System threads Modified code IAT Debug registers Command Full report
Disassembly
Ready
10:10 PM
7/28/2019

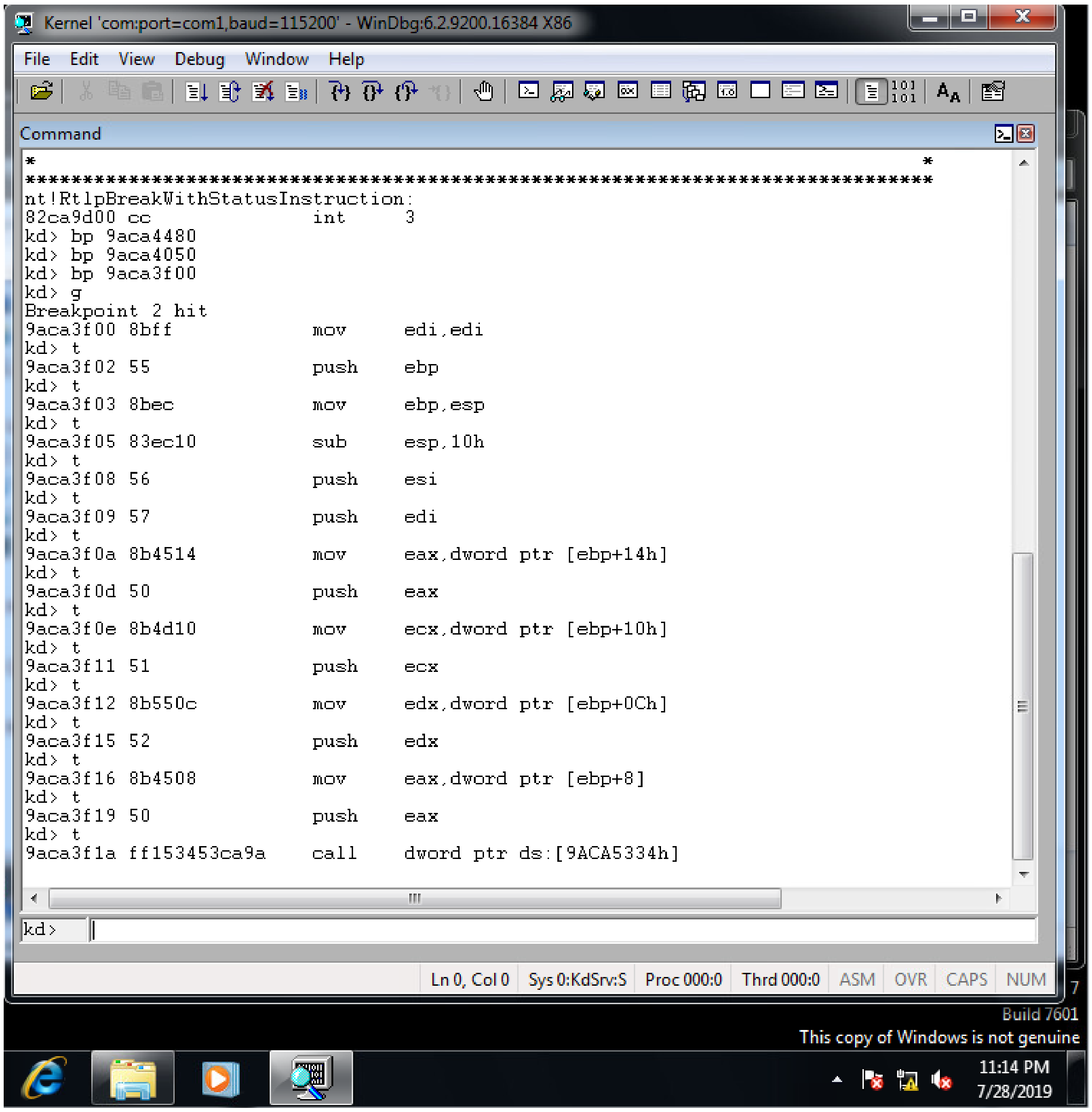
	uspicio	Index	Function	Original	Current	Reference to
1	Yes	119	NtEnumerateValueKey	82e95a0f	9aca4480	C:\analyzer\wininit.sys
2	Yes	223	NtQueryDirectoryFile	82ea9250	9aca4050	C:\analyzer\wininit.sys
3	Yes	261	NtQuerySystemInformation	82e4a775	9aca3f00	C:\analyzer\wininit.sys
4	No	0	NtAcceptConnectPort	82e8ba7a	82e8ba7a	C:\Windows\system32\ntoskrnl.exe
5	No	1	NtAccessCheck	82cdbf84	82cdbf84	C:\Windows\system32\ntoskrnl.exe
6	No	2	NtAccessCheckAndAuditAlarm	82ec3e50	82ec3e50	C:\Windows\system32\ntoskrnl.exe
7	No	3	NtAccessCheckByType	82d07b2e	82d07b2e	C:\Windows\system32\ntoskrnl.exe
8	No	4	NtAccessCheckByTypeAndAuditAlarm	82ea61c1	82ea61c1	C:\Windows\system32\ntoskrnl.exe
9	No	5	NtAccessCheckByTypeResultList	82d479d2	82d479d2	C:\Windows\system32\ntoskrnl.exe
10	No	6	NtAccessCheckByTypeResultListAndAuditAlarm	82f32d4d	82f32d4d	C:\Windows\system32\ntoskrnl.exe
11	No	7	NtAccessCheckByTypeResultListAndAuditAlarmByHandle	82f32d96	82f32d96	C:\Windows\system32\ntoskrnl.exe

```
9ACA4480  8bff              mov edi, edi
9ACA4482  55                push ebp
9ACA4483  8bec              mov ebp, esp
9ACA4485  83ec24            sub esp, 24h
9ACA4488  56                push esi
9ACA4489  57                push edi
9ACA448A  0fb6050053ca9a    movzx eax, byte ptr [9aca5300h]
9ACA4491  85c0              test eax, eax
```

Total: 401 Suspicious: 3

Figure 2 - Tuluka view of suspicious SST entries from agony.exe

WinDbg can also be used to analyze the kernel of an infected machine to locate the cuntion boundaries by setting breakpoints and stepping through the code.


Kernel 'com:port=com1,baud=115200' - WinDbg:6.2.9200.16384 X86
File Edit View Debug Window Help
Command
Ln 0, Col 0 Sys 0:KdSrv:S Proc 000:0 Thrd 000:0 ASM OVR CAPS NUM
Build 7601
This copy of Windows is not genuine
11:14 PM
7/28/2019

```
*                                                                              *
********************************************************************************
nt!RtlpBreakWithStatusInstruction:
82ca9d00 cc              int     3
kd> bp 9aca4480
kd> bp 9aca4050
kd> bp 9aca3f00
kd> g
Breakpoint 2 hit
9aca3f00 8bff            mov     edi,edi
kd> t
9aca3f02 55              push    ebp
kd> t
9aca3f03 8bec            mov     ebp,esp
kd> t
9aca3f05 83ec10          sub     esp,10h
kd> t
9aca3f08 56              push    esi
kd> t
9aca3f09 57              push    edi
kd> t
9aca3f0a 8b4514          mov     eax,dword ptr [ebp+14h]
kd> t
9aca3f0d 50              push    eax
kd> t
9aca3f0e 8b4d10          mov     ecx,dword ptr [ebp+10h]
kd> t
9aca3f11 51              push    ecx
kd> t
9aca3f12 8b550c          mov     edx,dword ptr [ebp+0Ch]
kd> t
9aca3f15 52              push    edx
kd> t
9aca3f16 8b4508          mov     eax,dword ptr [ebp+8]
kd> t
9aca3f19 50              push    eax
kd> t
9aca3f1a ff153453ca9a    call    dword ptr ds:[9ACA5334h]
kd>
```

Figure 3 - WinDbg step through on Kernel to find Agony

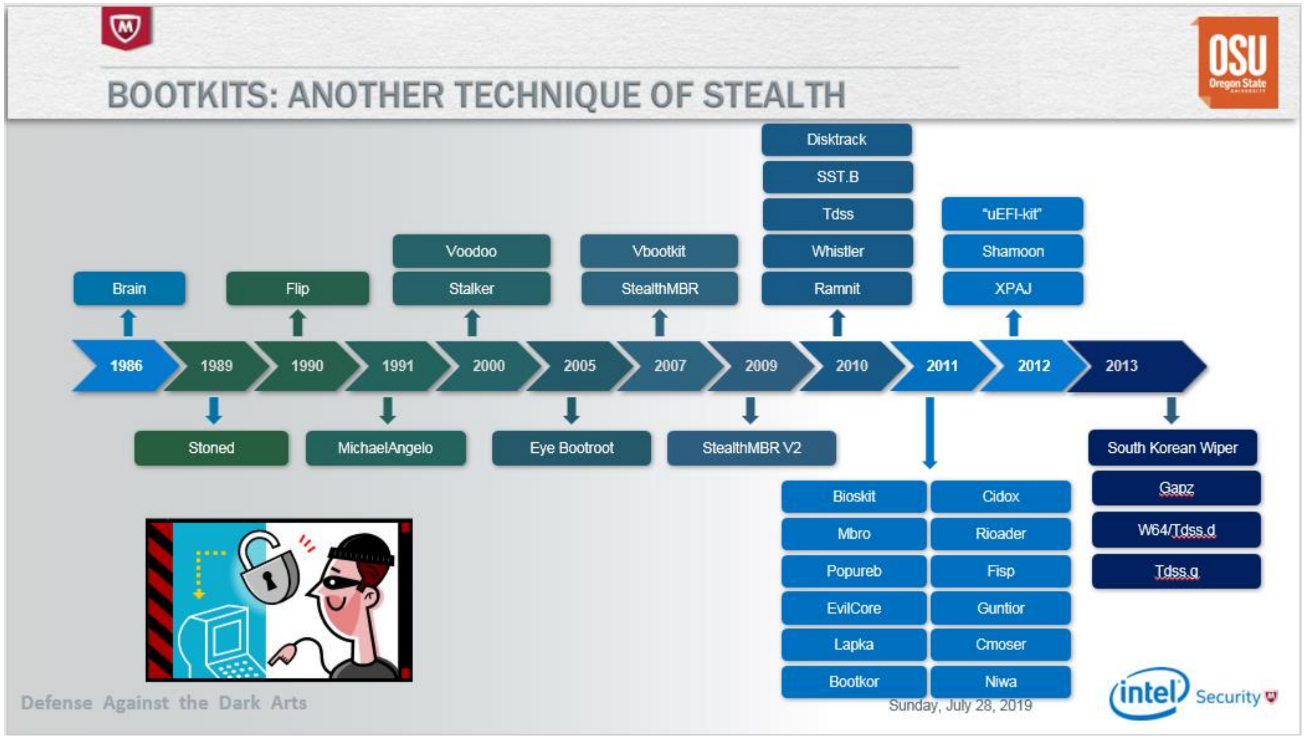
Another simple technique is to replace first 5 bytes of a normal dll with a new instruction to jump to bad code. Again, once the malware runs it will then call the initial code and goes right back to the normal flow. This is known as detouring. There are many locations that are able to have this sort of process injected.

Virus scan combats this by checking the pointer locations to verify they are still in the correct trusted areas. This is a cat and mouse game. Bad guys finding ways around new antivirus techniques and then antivirus getting more sophisticated to compensate.

Because infections are hard to find and remove once they have infected the kernel it is especially important to have antivirus that can catch them before they get installed.

Another stealth tactic is to forge file contents. In this strategy the malware overwrites existing files in such a way that they still look normal to antivirus. This is often better than hiding files because hidden files can be located using file system tools and parsers. That is much easier and consumes much less time than actually comparing files in a way sophisticated enough to find the forgery.

Another type of malware similar and to rootkits are bootkits. Where as rootkits are focused on stealth, bootkits take a different strategy and update the master boot record to execute their code. Because the malware is injected into the master boot record it runs before the operating system gets loaded and is able to bypass many security checks. Because it can do this it does not have to focus on hiding from the user in the operating system.


BOOTKITS: ANOTHER TECHNIQUE OF STEALTH

Defense Against the Dark Arts

Sunday, July 28, 2019

Figure 4- windows memory manipulation slide 37

One type of bootkit that is particularly popular currently and hard to deal with is a type of ransomware. In this technique the malware overwrites the master boot record with malware that blocks the machine from booting until the user enters a password. Unfortunately, the only way for the sure to get that password is to pay the malicious actor for it. Once the password is entered the malware typically will restore the master boot record leaving the system as it was before. Sometimes this type of malware will also encrypt the disk, making getting anything back very difficult without paying. In particular this is currently a huge trend against local and state governments, who often have the resources to pay much larger ransoms, and have data that is much more sensitive. (https://www.engadget.com/2019/07/26/local-governments-are-still-woefully-unprepared-to-fight-ransomw/)