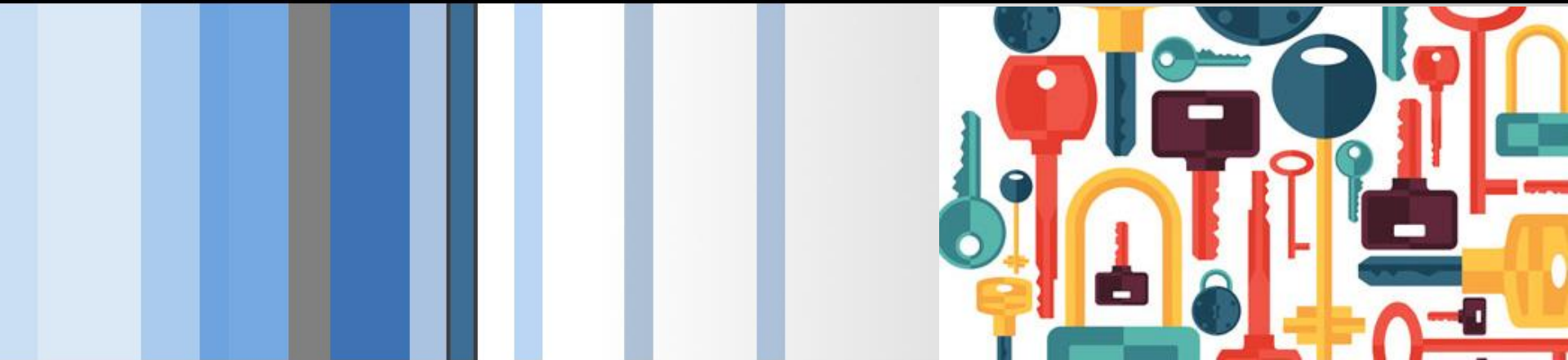


CSC 600 Advanced Seminar

STUXnet and Volatility

Si Chen (schen@wcupa.edu)



Analysis STUXnet Memory Dump Image



<https://github.com/volatilityfoundation/volatility>

Page ■ 3

STUXnet

Software Sabotage

How Stuxnet disrupted Iran's uranium enrichment program

1 The malicious computer worm probably entered the computer system – which is normally cut off from the outside world – at the uranium enrichment facility in Natanz via a removable USB memory stick.

2 The virus is controlled from servers in Denmark and Malaysia with the help of two Internet addresses, both registered to false names. The virus infects some 100,000 computers around the world.

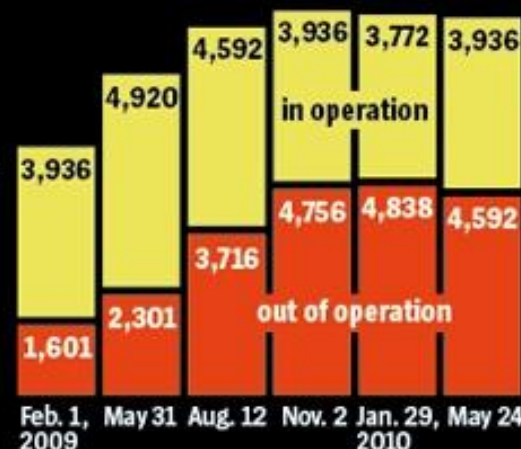
3 Stuxnet spreads through the system until it finds computers running the Siemens control software Step 7, which is responsible for regulating the rotational speed of the centrifuges.

4 The computer worm varies the rotational speed of the centrifuges. This can destroy the centrifuges and impair uranium enrichment.

Iranian centrifuges for uranium enrichment

UEN 671821

5 The Stuxnet attacks start in June 2009. From this point on, the number of inoperative centrifuges increases sharply.



Source: IAEA, ISIS, FAS, World Nuclear Association, FT research

Analysis STUXnet Memory Dump Image

```
vol.py -f stuxnet.vmem devicetree
```

```
DRV 0x022e54f8 \Driver\MRxNet
---| DEV 0x82125f10  FILE_DEVICE_DISK_FILE_SYSTEM
---| DEV 0x81dc49c0  FILE_DEVICE_DISK_FILE_SYSTEM
---| DEV 0x81fd59c0  FILE_DEVICE_CD_ROM_FILE_SYSTEM
---| DEV 0x81c8b500  FILE_DEVICE_CD_ROM_FILE_SYSTEM
---| DEV 0x821354b8  FILE_DEVICE_NETWORK_FILE_SYSTEM
---| DEV 0x81f0fc58  FILE_DEVICE_NETWORK_FILE_SYSTEM
---| DEV 0x81c0a910  FILE_DEVICE_NETWORK_FILE_SYSTEM
---| DEV 0x8226ef10  FILE_DEVICE_CD_ROM_FILE_SYSTEM
---| DEV 0x81f0ab90  FILE_DEVICE_DISK_FILE_SYSTEM
---| DEV 0x81fb9680  FILE_DEVICE_DISK_FILE_SYSTEM
---| DEV 0x82104700  FILE_DEVICE_DISK_FILE_SYSTEM
```

Analysis STUXnet Memory Dump Image

```
vol.py -f stuxnet.vmem devicetree
```

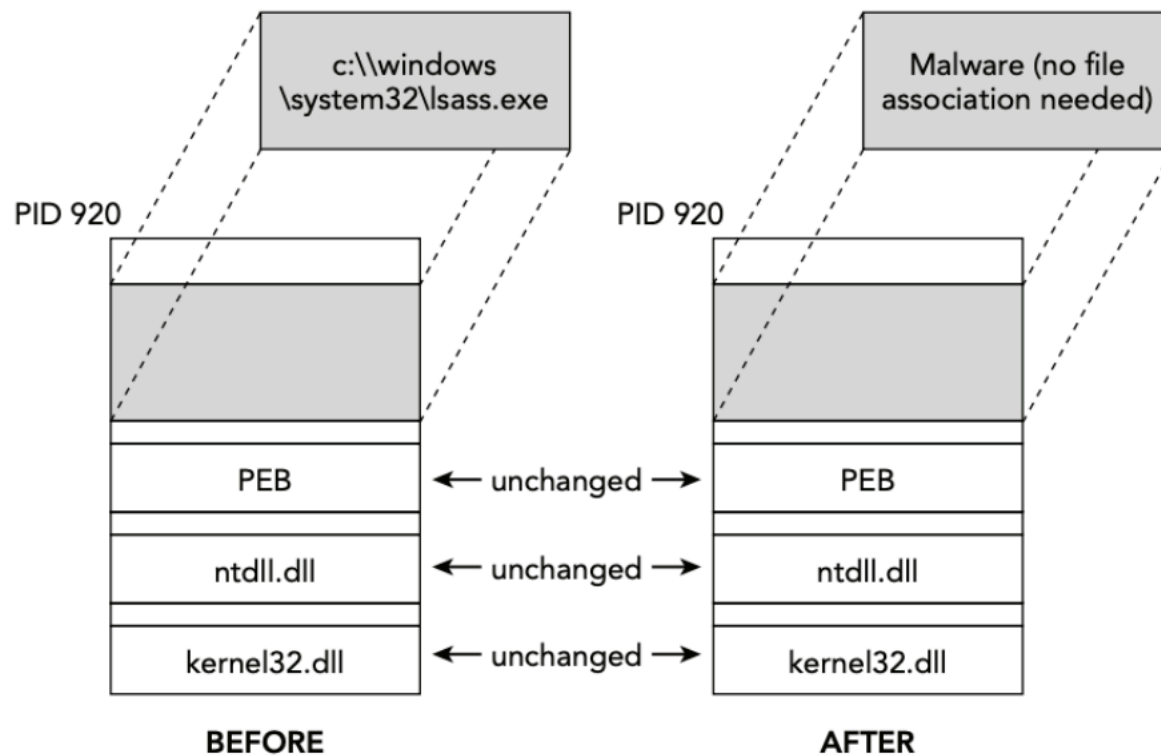
```
DRV 0x0205e5a8 \FileSystem\vmhgfs
---| DEV 0x820f0030 hgfsInternal UNKNOWN
---| DEV 0x821a1030 HGFS FILE_DEVICE_NETWORK_FILE_SYSTEM
-----| ATT 0x81f5d020 - \FileSystem\FltMgr FILE_DEVICE_NETWORK_FILE_SYSTEM
-----| ATT 0x821354b8 - \Driver\MRxNet FILE_DEVICE_NETWORK_FILE_SYSTEM
```

```
DRV 0x02476da0 \FileSystem\Cdfs
---| DEV 0x81e636c8 Cdfs FILE_DEVICE_CD_ROM_FILE_SYSTEM
-----| ATT 0x81fac548 - \FileSystem\FltMgr FILE_DEVICE_CD_ROM_FILE_SYSTEM
-----| ATT 0x8226ef10 - \Driver\MRxNet FILE_DEVICE_CD_ROM_FILE_SYSTEM
```

```
DRV 0x023ae880 \FileSystem\MRxSmb
---| DEV 0x81da95d0 LanmanDatagramReceiver FILE_DEVICE_NETWORK_BROWSER
---| DEV 0x81ee5030 LanmanRedirector FILE_DEVICE_NETWORK_FILE_SYSTEM
-----| ATT 0x81bf1020 - \FileSystem\FltMgr FILE_DEVICE_NETWORK_FILE_SYSTEM
-----| ATT 0x81f0fc58 - \Driver\MRxNet FILE_DEVICE_NETWORK_FILE_SYSTEM
```

Now Stuxnet can filter or hide specifically named files and directories on those file systems!

Hollow Process Injection



```
root@2f94317be09e:/workdir # vol.py -f stuxnet.vmem pstree
```

Virtual address descriptor (VAD)

- The Virtual Address Descriptor tree is **used by the Windows memory manager to describe memory ranges used by a process as they are allocated**. When a process allocates memory with VirtualAlloc, the memory manager creates an entry in the VAD tree.
- As a result of being hollowed, the virtual address descriptor (VAD) characteristics for the region are drastically different. Only the legitimate one still has a copy of the lsass.exe file mapped into the region.

- Based on Symantec report, Stuxnet has hooked Ntdll.dll to monitor for requests to load specially crafted file names.
- These specially crafted filenames are mapped to another location instead — a location specified by Stuxnet.
- The functions hooked for this purpose in Ntdll.dll are:
 - ZwMapViewOfSection
 - ZwCreateSection
 - ZwOpenFile
 - ZwCloseFile
 - ZwQueryAttributesFile
 - ZwQuerySection

API Hooking

Hook mode: Usermode
Hook type: NT Syscall
Process: 940 (svchost.exe)
Victim module: ntdll.dll (0x7c900000 - 0x7c9af000)
Function: ZwOpenFile
Hook address: 0x7c90004c
Hooking module: ntdll.dll

Disassembly(0):

0x7c90d580	b874000000	MOV EAX, 0x74
0x7c90d585	ba4c00907c	MOV EDX, 0x7c90004c
0x7c90d58a	ffd2	CALL EDX
0x7c90d58c	c21800	RET 0x18
0x7c90d58f	90	NOP
0x7c90d590	b875000000	MOV EAX, 0x75
0x7c90d595	ba	DB 0xba
0x7c90d596	0003	ADD [EBX], AL

Disassembly(1):

0x7c90004c	b202	MOV DL, 0x2
0x7c90004e	eb0c	JMP 0x7c90005c
0x7c900050	b203	MOV DL, 0x3
0x7c900052	eb08	JMP 0x7c90005c
0x7c900054	b204	MOV DL, 0x4
0x7c900056	eb04	JMP 0x7c90005c
0x7c900058	b205	MOV DL, 0x5
0x7c90005a	eb00	JMP 0x7c90005c
0x7c90005c	52	PUSH EDX
0x7c90005d	e804000000	CALL 0x7c900066
0x7c900062	f2	DB 0xf2
0x7c900063	00	DB 0x0

```
pinlog.py -f stuxnet.vmem apihooks
```

API Hooking

0x00bf00f2

```
0x7c900058 b205 MOV DL, 0x5
0x7c90005a eb00 JMP 0x7c90005c
0x7c90005c 52 PUSH EDX
0x7c90005d e804000000 CALL 0x7c900066
0x7c900062 f200bf005aff22 ADD [EDI+0x22ff5a00], BH
0x7c900069 696e20444f5320 IMUL EBP, [ESI+0x20], 0x20534f44
```

```
0x7c900066 5a POP EDX
>>> dis(0x7c900066)
0x7c900066 5a POP EDX
0x7c900067 ff22 JMP DWORD [EDX]
0x7c900069 696e20444f5320 IMUL EBP, [ESI+0x20], 0x20534f44
0x7c900070 6d INS DWORD [EDI], DX
0x7c900071 6f OUTS DX, DWORD [ESI]
0x7c900072 64652e0d0d0a2400 OR EAX, 0x240a0d
```

1. When the CALL at 0x7c90005d is executed, its return address (0x7c900062) is pushed onto the stack.
2. The POP EDX instruction at 0x7c900066 then removes that value from the stack and places it in EDX.
3. At 0x7c900067 , EDX is dereferenced and called. So the pointer being dereferenced is stored in 0x7c900062 .

Kernel Callback

- A **callback function** is one which is passed as an argument to another function and is invoked after the completion of the parent function.
 - In other words **callback** is a piece of executable code that is passed as an argument to other code, which is expected to *call back* (execute) the argument at some convenient time.
- Kernel Callback
 - Supported on 64 bit systems
 - Safe for multicore machines
 - Lists of events:
 - Process creation
 - Thread creation
 - System shutdown
 - File system registration
 - PnP(Plug and Play)
 - etc...

Analysis STUXnet Memory Dump Image

```
[quake0day@archlinux ~]$ python2 /usr/bin/vol.py -f stuxnet.vmem --profile=WinXPSP3x86 callbacks
```

```
Volatility Foundation Volatility Framework 2.6.1
```

Type	Callback	Module	Details
IoRegisterShutdownNotification	0xf88ddc74	Cdfs.SYS	\FileSystem\Cdfs
IoRegisterShutdownNotification	0xf8bb05be	Fs_Rec.SYS	\FileSystem\Fs_Rec
IoRegisterShutdownNotification	0xf8303c6a	VIDEOPT.SYS	\Driver\VgaSave
IoRegisterShutdownNotification	0xb2d108fa	vmhgfs.sys	\FileSystem\vmhgfs
IoRegisterShutdownNotification	0xf8bb05be	Fs_Rec.SYS	\FileSystem\Fs_Rec
IoRegisterShutdownNotification	0xf8303c6a	VIDEOPT.SYS	\Driver\mmnmd
IoRegisterShutdownNotification	0xf8303c6a	VIDEOPT.SYS	\Driver\vmx_svga
IoRegisterShutdownNotification	0xf8303c6a	VIDEOPT.SYS	\Driver\RDPCDD
IoRegisterShutdownNotification	0xf86aa73a	MountMgr.sys	\Driver\MountMgr
IoRegisterShutdownNotification	0xf8bb05be	Fs_Rec.SYS	\FileSystem\Fs_Rec
IoRegisterShutdownNotification	0xf8bb05be	Fs_Rec.SYS	\FileSystem\Fs_Rec
IoRegisterShutdownNotification	0xf8bb05be	Fs_Rec.SYS	\FileSystem\Fs_Rec
IoRegisterShutdownNotification	0xf853c2be	ftdisk.sys	\Driver\Ftdisk
IoRegisterShutdownNotification	0x805cdef4	ntoskrnl.exe	\FileSystem\RA
IoRegisterShutdownNotification	0xf83d98f1	Mup.sys	\FileSystem\
IoRegisterShutdownNotification	0x805f5d66	ntoskrnl.exe	\Driver\W
IoRegisterFsRegistrationChange	0xf84be876	sr.sys	-
GenericKernelCallback	0xf87ad194	vmci.sys	-
IoRegisterFsRegistrationChange	0xb21d89ec	mrxcnsl.sys	-
GenericKernelCallback	0xb240ce4c	PROCMON20.SYS	-
GenericKernelCallback	0x805f81a6	ntoskrnl.exe	-
GenericKernelCallback	0xb240cc9a	PROCMON20.SYS	-
GenericKernelCallback	0xf895ad06	mrxcnsl.sys	-
GenericKernelCallback	0xb240cb94	PROCMON20.SYS	-
GenericKernelCallback	0xb240cb94	PROCMON20.SYS	-
IoRegisterFsRegistrationChange	0xf84d54b8	fltMgr.sys	-
PsSetLoadImageNotifyRoutine	0xb240ce4c	PROCMON20.SYS	-
PsSetLoadImageNotifyRoutine	0x805f81a6	ntoskrnl.exe	-
PsSetLoadImageNotifyRoutine	0xf895ad06	mrxcnsl.sys	-
PsSetCreateThreadNotifyRoutine	0xb240cc9a	PROCMON20.SYS	-
PsSetCreateProcessNotifyRoutine	0xf87ad194	vmci.sys	-
PsSetCreateProcessNotifyRoutine	0xb240cb94	PROCMON20.SYS	-
KeBugCheckCallbackListHead	0xf83e65ef	NDIS.sys	Ndis miniport
KeBugCheckCallbackListHead	0x806d77cc	hal.dll	ACPI 1.0 - APIC platform UP
KeRegisterBugCheckReasonCallback	0xf8b7aab8	mssmbios.sys	SMBiosDa
KeRegisterBugCheckReasonCallback	0xf8b7aa70	mssmbios.sys	SMBiosRe
KeRegisterBugCheckReasonCallback	0xf8b7aa28	mssmbios.sys	SMBiosDa
KeRegisterBugCheckReasonCallback	0xf82e01be	USBPORT.SYS	USBPORT
KeRegisterBugCheckReasonCallback	0xf82e011e	USBPORT.SYS	USBPORT
KeRegisterBugCheckReasonCallback	0xf82f7522	VIDEOPT.SYS	Videoprt

Now Stuxnet can receive notification when new file system become available – So it can immediately spread or hide files

And is able to inject code into process when they try to load other DLLs.

Q & A

