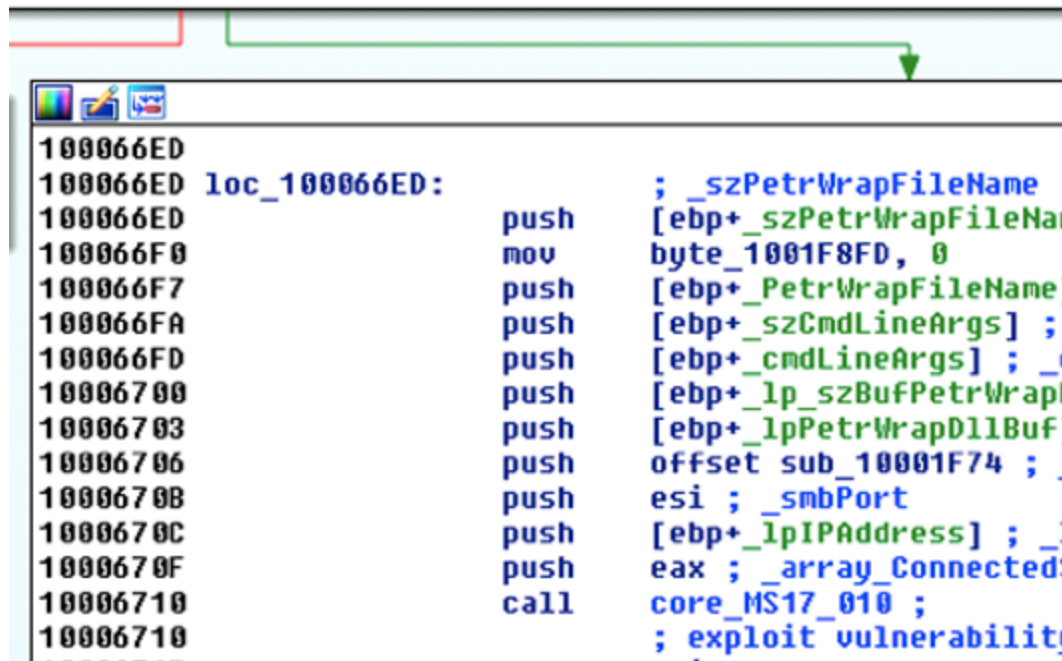


NotPetya Technical Analysis Part II: Further Findings

crowdstrike.com/blog/petrwrap-technical-analysis-part-2-further-findings-and-potential-for-mbr-recovery/

Shaun Hurley and Karan Sood

July 3, 2017



```
100066ED
100066ED loc_100066ED:          ; _szPetrWrapFileName
100066ED          push    [ebp+_szPetrWrapFileNam
100066F0          mov     byte_1001F8FD, 0
100066F7          push    [ebp+_PetrWrapFileName]
100066FA          push    [ebp+_szCmdLineArgs] ;
100066FD          push    [ebp+_cmdLineArgs] ; _c
10006700          push    [ebp+_lp_szBufPetrWrapC
10006703          push    [ebp+_lpPetrWrapDllBuf]
10006706          push    offset sub_10001F74 ; _
1000670B          push    esi ; _smbPort
1000670C          push    [ebp+_lpIPAddress] ; _]
1000670F          push    eax ; _array_ConnectedS
10006710          call   core_MS17_010 ;
10006710          ; exploit vulnerability
```

Update:

Due to naming convention consistency in the industry, CrowdStrike is now calling this variant of Petya – NotPetya.

Executive Summary

This technical analysis is a continuation of the previous technical blog ([NotPetya Technical Analysis – A Triple Threat: File Encryption, MFT Encryption, Credential Theft](#)) describing the threat of NotPetya, a destructive malware with self-propagation capabilities. After further analysis, CrowdStrike researchers discovered:

- How the NotPetya DLL loads functions to ensure proper cleanup from the victim machine
- The order in which the infection process takes place
- Potential for recovery of the victim machine’s MBR and boot manager when running one specific antivirus software

NotPetya DLL Loader

The NotPetya DLL, seen in the wild with the filename “perfc.dat”, takes the following steps during the DLL loading process to ensure that there is no trace of the NotPetya DLL (perfc.dll) having being on the system:

- Copies file contents from disk into a buffer residing in process memory
- Checks to determine if the DLL has already been loaded
- Copies the NotPetya DLL already mapped into process memory to another buffer
- Calculates and patches the relative offsets in the NotPetya copy with a new offset
- The new relative offset is calculated by subtracting the base address original NotPetya DLL from the RVAs identified in the relocation table of the NotPetya copy
- Calls the function at offset +0x94A5 located in the NotPetya copy
 - Calls FreeLibrary to unmap the original NotPetya DLL from process memory
 - Overwrites the original DLL file with null bytes
 - Deletes the original NotPetya DLL from the file system
 - Fixes the import table of the NotPetya DLL copy
 - Calls the perfc_1 (export ordinal 1) function to kick off the worm activity and file/MFT encryption

The file cleanup combined with the file and MFT encryption process make it difficult to recover the original NotPetya DLL.

SMB Exploitation and Infection

The EternalBlue and EternalRomance exploits are used for the MS17-010 vulnerability, and are subsequently used to propagate NotPetya to vulnerable hosts on the local subnet. The infection process occurs in the following order:

1. Test for vulnerable condition
2. Check Windows version
3. Trigger MS17-010 vulnerability. For more information, please click [here](#).
4. Deploy an SMB backdoor
5. NotPetya infection

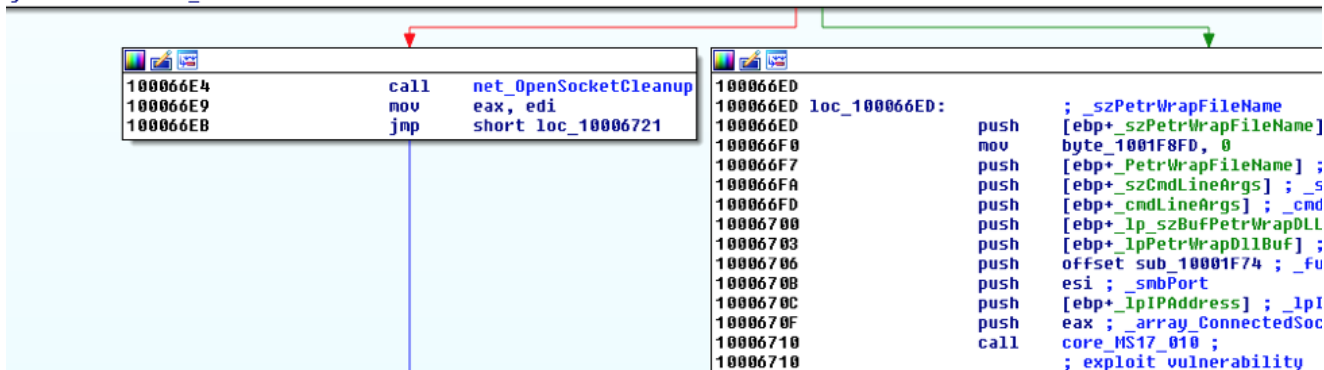
Test for Vulnerable Condition

As can be seen in the control flow graph, core_MS17_010 is initially called to determine if the exploit condition exists. If it does, core_MS17_010 is called again with slightly different arguments to exploit the vulnerability and send the shellcode.

```

push  0 ; _functionPtr
push  esi ; _smbPort
push  [ebp+_lpIPAddress] ; _lpIPAddress
push  eax ; _array_ConnectedSockets
call  core_MS17_010 ;
; Check for vulnerable condition
mov   edi, eax
lea   eax, [ebp+array_ConnectedSockets]
test  edi, edi
jz    short loc_100066ED

```

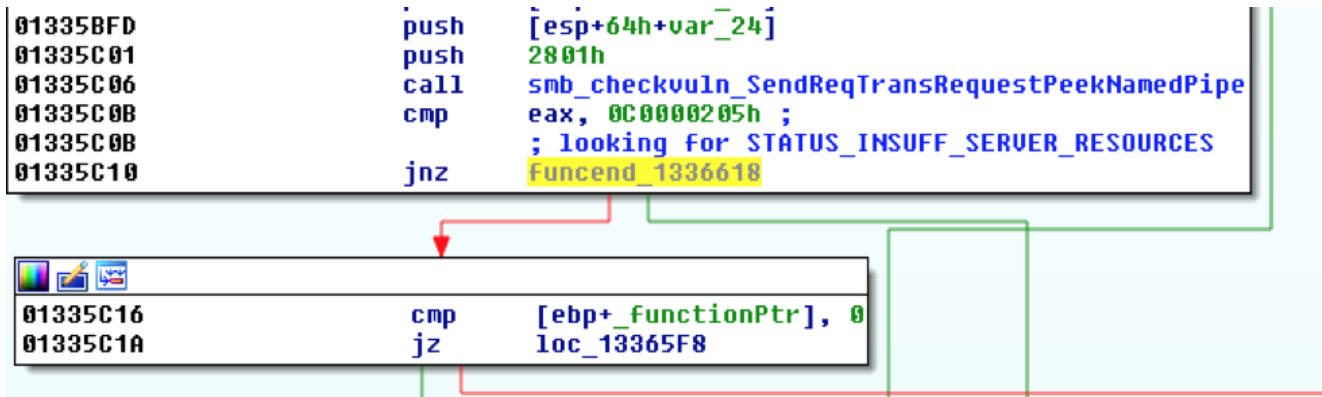


The following is an image and explanation of the SMB network traffic used to determine if the victim is vulnerable (Victim: 172.16.1.132):

9	1.284224	172.16.1.214	172.16.1.132	SMB	191 Negotiate Protocol Request
0	1.285002	172.16.1.132	172.16.1.214	SMB	187 Negotiate Protocol Response
1	1.285048	172.16.1.214	172.16.1.132	SMB	194 Session Setup AndX Request, User: anonymous
2	1.285390	172.16.1.132	172.16.1.214	SMB	259 Session Setup AndX Response
3	1.285431	172.16.1.214	172.16.1.132	SMB	146 Tree Connect AndX Request, Path: \\123.12.31.2\IPC\$
4	1.285804	172.16.1.132	172.16.1.214	SMB	114 Tree Connect AndX Response
5	1.285841	172.16.1.214	172.16.1.132	SMB	136 Trans2 Request, SESSION_SETUP
6	1.286175	172.16.1.132	172.16.1.214	SMB	93 Trans2 Response, SESSION_SETUP, Error: STATUS_NOT_IMPLEMENTED
7	1.286212	172.16.1.214	172.16.1.132	SMB Pi...	132 PeekNamedPipe Request, FID: 0x0000
8	1.286429	172.16.1.132	172.16.1.214	SMB	93 Trans Response, Error: STATUS_INSUFF_SERVER_RESOURCES
9	1.286463	172.16.1.214	172.16.1.132	SMB	93 Tree Disconnect Request
0	1.286662	172.16.1.132	172.16.1.214	SMB	93 Tree Disconnect Response
1	1.286694	172.16.1.214	172.16.1.132	SMB	97 Logoff AndX Request
2	1.286929	172.16.1.132	172.16.1.214	SMB	97 Logoff AndX Response

1. An initial SMB_COM_NEGOTIATE request and response
2. Session setup
3. Tree connect
4. PeekNamedPipe SMB transaction with FID: 0x0000
 1. Exploitable condition:
 1. Trans response, error: STATUS_INSUFF_SERVER_RESOURCES
 2. Error code: 0xC0000205
 3. Server is out of resources
5. Close session: Tree disconnect and AndX logoff

A check is then done to determine if the victim's response is STATUS_INSUFF_SERVER_RESOURCES (0xC0000205). An example of this is shown below:



If the error code does not exist, the session is cleaned up and NotPetya iterates to the next IP address. If the error code exists, the core_MS17_010 returns 0, and the vulnerability is exploited.

Check Windows Version

Prior to executing the exploit code, a test ensues to determine which version of Microsoft Windows is running on the system. This is done by parsing the “Session Setup AndX Response” packet. Once the version of the OS is identified it is mapped to a numeric value:

- 2 – Windows XP
- 3 – Windows XP Pro x64, Windows 2003, Windows 2003 R2
- 4 – Windows Vista
- 5 – Windows 7
- 6 – Windows 2008
- 7 – Windows 2008 R2

NotPetya will only exploit one of the Windows versions listed above. If the version does not match one of the identified Windows versions, then the function to execute the exploit code returns without executing.

The EternalBlue exploit is launched against only the following Windows versions:

- Windows 7
- Windows 2008
- Windows 2008 R2

The EternalRomance exploit is launched against only the following Windows versions:

- Windows XP
- XP Pro x64
- Windows Server 2003
- Windows Server 2003 R2
- Vista

The EternalRomance code path begins at 0x10005C4C.

Trigger MS17-010 (EternalBlue)

The following image and explanation refers to the SMB network traffic that starts the exploitation process:

172.16.1.214	172.16.1.132	SMB	191 Negotiate Protocol Request
172.16.1.132	172.16.1.214	SMB	187 Negotiate Protocol Response
172.16.1.214	172.16.1.132	SMB	194 Session Setup AndX Request, User: anonymous
172.16.1.132	172.16.1.214	SMB	259 Session Setup AndX Response
172.16.1.214	172.16.1.132	SMB	146 Tree Connect AndX Request, Path: \\123.12.31.2\IPC\$
172.16.1.132	172.16.1.214	SMB	114 Tree Connect AndX Response
172.16.1.214	172.16.1.132	SMB	136 Trans2 Request, SESSION_SETUP
172.16.1.132	172.16.1.214	SMB	93 Trans2 Response, SESSION_SETUP, Error: STATUS_NOT_IMPLEMENTED
172.16.1.214	172.16.1.132	SMB Pi...	132 PeekNamedPipe Request, FID: 0x0000
172.16.1.132	172.16.1.214	SMB	93 Trans Response, Error: STATUS_INSUFF_SERVER_RESOURCES
172.16.1.214	172.16.1.132	SMB	1138 NT Trans Request, <unknown>
172.16.1.132	172.16.1.214	SMB	93 NT Trans Response, <unknown (0)>
172.16.1.214	172.16.1.132	SMB	4207 Trans2 Secondary Request, FID: 0x0000
172.16.1.132	172.16.1.214	TCP	60 145 40200 54000 0x0000 0x0000 0x0000 0x0000

1. An initial SMB_COM_NEGOTIATE request and response
2. Session setup
3. Tree connect
4. PeekNamedPipe SMB transaction with FID: 0x0000
 1. Exploitable condition:
 1. Trans response, error: STATUS_INSUFF_SERVER_RESOURCES
 2. Error code: 0xC0000205
 3. Server is out of resources
5. NT trans request
 1. Followed by Trans2 secondary request, FID: 0x0000

This exploitation process is followed by a series of additional SMB packets. There are three types of packets that are identified based on size:

- TYPE 0:
 - TCP segment data size: 132 bytes
 - The first 10 bytes of the TCP segment data contain:
 - 0x0 – Null byte
 - 0x2 – 0xFFFF7
 - 0x4 – 2 byte value based on a timing check
 - 0x6 – 2 byte value based on another timing check
 - The rest of the packet contains null bytes
- TYPE 1:
 - TCP segment data size: 2920 bytes
 - This packet sends the kernel shellcode
 - This packet is sent with the same data 13 times

This first burst of requests are TYPE 0 packets.

To trigger the exploit, a 4207-byte “SMB Trans2 Secondary Request, FID: 0x0000” packet is sent to the vulnerable machine, as depicted below. This packet is built and sent in function sub_10003C0A.

Source IP	Destination IP	Protocol	Details
172.16.1.214	172.16.1.132	SMB	4207 Trans2 Secondary Request, FID: 0x0000
172.16.1.132	172.16.1.214	TCP	60 445 → 49266 [ACK] Seq=622 Ack=68168 Win=65536 Len=0
172.16.1.132	172.16.1.214	SMB	146 Trans2 Response<unknown>, Error: STATUS_INVALID_PARAMETER
172.16.1.214	172.16.1.132	TCP	2974 [TCP segment of a reassembled PDU]

The exploit code is contained within the 4096 byte “Extra byte parameters” section of the Trans2 packet. This 4096 buffer is filled with hex byte ‘\x54’ (‘T’). At offset 0xB within that buffer, the hex byte ‘\x51’ is used to specify the start of the data that will trigger the overflow on the vulnerable machine.

The specific code to trigger the overflow is the 175-byte chunk of binary data within the NotPetya binary, starting at address 0x10010B08. The code is not encoded.

Sending the Ring 0 Shellcode

Once the vulnerability has been exploited, the Ring 0 shellcode is sent using TYPE 1 packets.

The ring 0 shellcode, which contains the SMB backdoor code, is the 2423 byte chunk for binary data starting at address 0x1000123B0 within the NotPetya binary. The ring 0 shellcode is encoded with a simple byte xor. The xor key is 0xCC. The shellcode starts at offset 0x1F1 within the TCP segment data.

Deploy SMB Backdoor

The backdoor that gets deployed onto an exploited system appears to be a modified version (based on network traffic) of the DoublePulsar rootkit leaked by the Shadow Brokers actors.

Network Traffic

If there is a DoublePulsar knock feature, it is not used. Once the SMB backdoor is in place, the ‘Reserved’ field of the SMB Header for the response packet is set to 0x1100, as seen below. Normally this field is NULL.

SMB Header

Server Component: SMB

[\[Response to: 593\]](#)

[Time from request: 0.000

SMB Command: Trans2 (0x32

NT Status: STATUS_NOT_IMP

▷ Flags: 0x98, Request/Resp

▷ Flags2: 0xc007, Unicode S

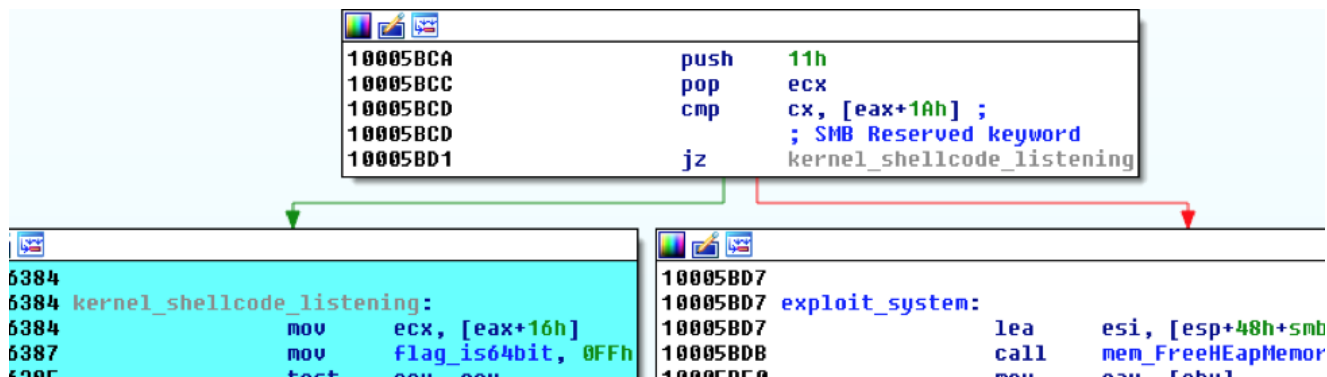
Process ID High: 7845

Signature: 0000000000000000

Reserved: 1100

▷ Tree ID: 2048 (\\123.12.

Prior to triggering the code to exploit the system, the “Reserved” field in the SMB header is checked to see if it is set to 0x11.



If the “Reserved” field in the SMB header is set to 0x11, the staging code is built and sent to the targeted machine.

172.16.1.214	172.16.1.132	SMB	4232 Trans2 Request, SESSION_SETUP
172.16.1.132	172.16.1.214	TCP	60 445 → 49287 [ACK] Seq=1452 Ack=113258 Win=65536 Len=0
172.16.1.132	172.16.1.214	SMB	93 Trans2 Response, SESSION_SETUP, Error: STATUS_NOT_IMPLEMENTED
172.16.1.214	172.16.1.132	SMB	4232 Trans2 Request, SESSION_SETUP
172.16.1.132	172.16.1.214	TCP	60 445 → 49287 [ACK] Seq=1491 Ack=117436 Win=65536 Len=0
172.16.1.132	172.16.1.214	SMB	93 Trans2 Response, SESSION_SETUP, Error: STATUS_NOT_IMPLEMENTED

“Trans2 Request, SESSION_SETUP” packets are used to send the staging DLL to the target.

NotPetya Infection

To deploy NotPetya onto the system, a staging DLL is injected into “lsass.exe.” Once the DLL is injected into lsass, NotPetya is written to the system in c:\windows:

lsass.exe	564	CreateFile	C:\Windows\olga.dll
lsass.exe	564	WriteFile	C:\Windows\olga.dll
lsass.exe	564	CloseFile	C:\Windows\olga.dll
lsass.exe	564	RegOpenKey	HKEY_M\System\CurrentC...

The NotPetya DLL keeps the same filename the file was called when executed on the attacking machine (in this case, olga.dll). Once the file is written, it can be launched.

The staging DLL executes the file in the same way as psexec and wmic: rundll32.exe. As can be seen below, the same arguments are passed to the NotPetya DLL.

Name	Description	Path
lsass.exe (564)	Local Security Aut...	C:\Windows\system...
rundll32.exe (3088)	Windows host proc...	C:\Windows\system...
lsmd.exe (572)	Local Session Man...	C:\Windows\system...
winlogon.exe (496)	Windows Logon A...	C:\Windows\system...
Explorer.EXE (1872)	Windows Explorer	C:\Windows\Explor...
vmtoolsd.exe (620)	VMware Tools Cor...	C:\Program Files\V...
cmd.exe (2268)	Windows Comman...	C:\Windows\system...
Procmon.exe (3240)	Process Monitor	C:\Users\malta\De...

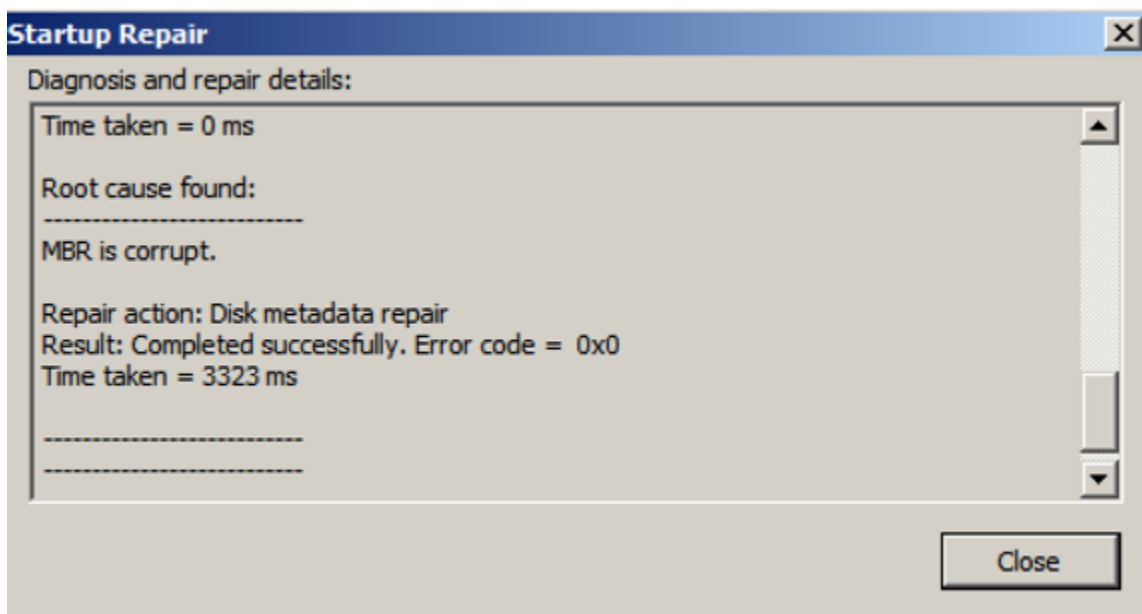
Description:	Windows host process (Rundll32)	
Company:	Microsoft Corporation	
Path:	C:\Windows\system32\rundll32.exe	
Command:	rundll32.exe c:\Windows\olga.dll,#1 10 "user:pass"	
User:	NT AUTHORITY\SYSTEM	
PID:	3088	Started: 7/2/2017 4:28:47 PM
		Exited: 7/2/2017 4:28:47 PM

MBR Restoration for Machines running Kaspersky Antivirus


Through analysis, it was discovered that if the victim machine has avp.exe (associated with Kaspersky antivirus) process running, NotPetya will NOT encrypt the MFT. Victim machines that have avp.exe running when impacted by NotPetya will simply have the the first 10 sectors of the physical disk overwritten with uninitialized data. For victim machines that do not have avp.exe running, NotPetya will overwrite the MBR with a custom boot loader, which will then load 16-bit code responsible for encrypting the MFT.



- Under System Recovery Options, click on "Use Recovery Tools"
- Click "Startup Repair"
- The MBR should now be repaired. Users can confirm this by clicking on "Click here for diagnostic and repair details" and scrolling down to the MBR section



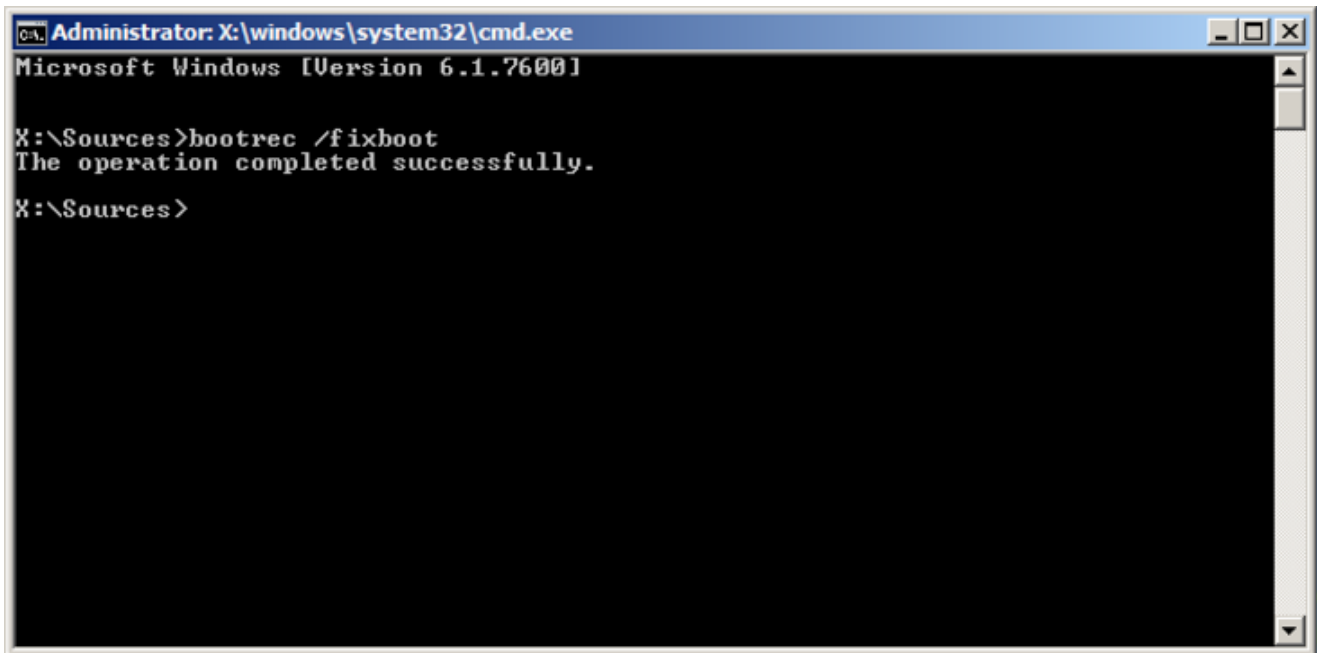
- Click Close → Finish. This will reboot the machine
- Some users might receive the following error message upon reboot. This error is due to the boot manager corruption



```
BOOTMGR is missing
Press Ctrl+Alt+Del to restart
```

- To rectify this issue, change the boot order to boot from the Windows disk image. This can be achieved by going into the BIOS settings
- Upon system reboot, follow the first 2 steps and the user should then see the OS under System Recovery Options
- Click on the OS, choose the “Use recovery tools” option, and click Next
- To repair the boot manager, click on “Command Prompt” and enter the following command:

bootrec /fixboot



```
C:\> Administrator: X:\windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7600]

X:\Sources> bootrec /fixboot
The operation completed successfully.

X:\Sources>
```

- Upon successful execution of the command, close the command prompt and click “Restart”
- Upon repairing the MBR and the boot manager, the user can boot the system without the Windows installation disk and access the system

The above steps will allow the user to access the system again, but certain files (that match the target extension list) on the system will still be encrypted and can only be decrypted with a private key possessed by the NotPety authors. Analysis also shows that the malware only encrypts the first 1MB of file contents, if the size of the file is greater than 1MB. The remaining contents are left untouched and intact.

```
20  if ( FileSize.QuadPart <= 0x100000 )
21  {
22      FileSize_1 = (LPCWSTR)FileSize.s.LowPart;
23      Final = 1;
24      FinalFileSize = 16 * ((FileSize.s.LowPart >> 4) + 1);
25  }
26  else
27  {
28      FileSize_1 = (LPCWSTR)0x100000;
29      FinalFileSize = 0x100000;
30  }
31  hobject_ = CreateFileMappingW(v3, 0, 4u, 0, FinalFileSize, 0);
32  hObject = hobject_;
33  if ( hobject_ )
34  {
35      v6 = MapViewOfFile(hobject_, 6u, 0, 0, (SIZE_T)FileSize_1);
36      if ( v6 )
37      {
38          if ( CryptEncrypt(*(_DWORD *) (a2 + 20), 0, Final, 0, (BYTE *)v6, (DWORD *)&FileSize_1
```

As the above image shows, line 20 compares the file size with 0x100000 (1,048,576 bytes or 1MB). If the file size is greater than 1MB, line 26 declares variables FileSize_1 and FinalFileSize to be 0x100000. These values are then used in CreateFileMappingW, MapViewOfFile and CryptEncrypt to encode the first 1MB of the file contents. The rest of the file remains untouched.

Additional Resources

For more information on CrowdStrike's proactive protection features see [the earlier CrowdStrike blog](#) on how [Falcon Endpoint Protection](#) prevents the NotPetya attack. In addition, watch a demo of [CrowdStrike Falcon® detecting and blocking NotPetya](#).