JUNE 2017

DISSECTING NOTPETYA

SO YOU THOUGHT IT WAS A RANSOMWARE.

JOSEPH LANDRY, NIR IZRAELI, UDI SHAMIR, CALEB FENTON, ITAI LIBA

Executive Summary

NotPetya has been in the news a lately for being yet another ransomware attack that has spread like fire – affecting organizations in several verticals across 65+ countries, drawing comparisons with the WannaCry attack that recently hit over 200,000 machines globally.

While it shows characteristics similar to a ransomware, NotPetya is more akin to a wiper, which is generally regarded as a malware responsible for destroying data on the target's hard disk. The ransom collection as of this writing is just over $10,000. Additionally, the email address used in the ransom request have since been shut down.

NotPetya infects the master boot record (MBR) and prevents any system from booting. And even paying the ransom would not have recovered the machine! In that sense, it is also different from the 2016 Petya threat in that the damage from NotPetya is not reversible.

NotPetya leveraged the EternalBlue (well-known with WannaCry) as well as EternalRomance, both exploiting the MS17-010 vulnerability. However, the attackers also leverage other non-exploit, legal mechanisms to laterally spread – such as psexec and windows management interface, further expanding the reach to include machines patched for the MS17-010 vulnerability.

SentinelOne customers using SentinelOne Enterprise Protection Platform are proactively protected against this MBR attack. However, we also advise customers to ensure that all machines have installed the latest Windows updates to reduce the threat impact. Additionally, limiting or removing administrative permissions for regular users will further reduce the attack surface.

Table of Contents

- 027cc450ef5f8c5f653329641ec1fed91f694e0d229928963b30f6b0d7d3a745 Main DLL
    - 02ef73bd2458627ed7b397ec26ee2de2e92c71a0e7588f78734761d8edbdcd9f embeded 64-bit credential dumper
    - eae9771e2eeb7ea3c6059485da39e77b8c0c369232f01334954fbac1c186c998 embeded 32-bit credential dumper
    - f8dbabdfa03068130c277ce49c60e35c029ff29d9e3c74c362521f3fb02670d5 embeded psexec.exe (benign)

Synopsys

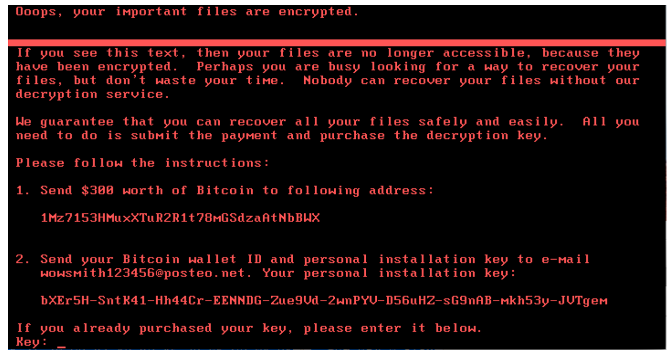
This ransomware sample implements worm functionality and has three methods of spreading:

1. Remote exploit for MS17-010 (EternalBlue, EternalRomance)
2. The psexec tool
3. Windows Management Instrumentation (WMI)

The exploit for the MS17-010 vulnerability will only infect unpatched systems, but the psexec and WMI methods will work on fully patched systems because they do not leverage an exploit. These two non-exploit methods use credentials extracted from the Local Security Authority (LSASS) in an attempt to authenticate to networked systems.

Like Petya, this sample will infect the Master Boot Record (MBR). The MBR normally contains 512 bytes of code that executes before Windows loads. By infecting this region of the hard-drive, this sample can lock a system and prevent Windows from booting. SentinelOne blocks the attempt to infect this critical region of the hard-drive.

If the MBR is infected, this screen will be seen after the infected machine reboots:



MBR ransom message

Similar to other MBR ransomware, this sample will encrypt the entire hard drive when booted into this mode but it also encrypts individual files before rebooting.

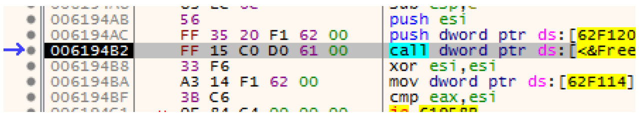
Static Features

The sample is a 32-bit DLL with one unnamed export. It isn't packed, and doesn't use string obfuscation. There are four obfuscated binaries in the resource section. One is the psexec utility, two are the 32 and 64-bit versions of the credential stealer, and the fourth binary is believed to be a component of the EternalBlue exploit.

Analysis

Installation

When the sample is first launched, it will ensure its main DLL is installed inside the "`C:\Windows`" directory. Malware typically copies itself as part of its installation routine, but this sample has a peculiar way of installing itself. Normal malware will have to create a new process after copying itself to it's final location. This sample will relocate itself in memory and free the original, removing the file lock on the disk.



```
006194AB  56                 push esi
006194AC  FF 35 20 F1 62 00  push dword ptr ds:[62F120
006194B2  FF 15 C0 D0 61 00  call dword ptr ds:[<&Free
006194B8  33 F6              xor esi,esi
006194BA  A3 14 F1 62 00     mov dword ptr ds:[62F114]
006194BF  3B C6              cmp eax,esi
```

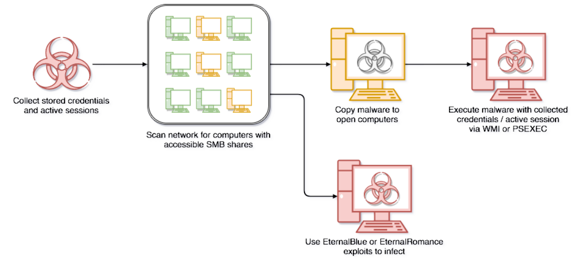
Freeing the original copy

Now the sample can delete the original copy of itself in a single process.

Lateral Movement

Uses two different methods to infect machines over the network:

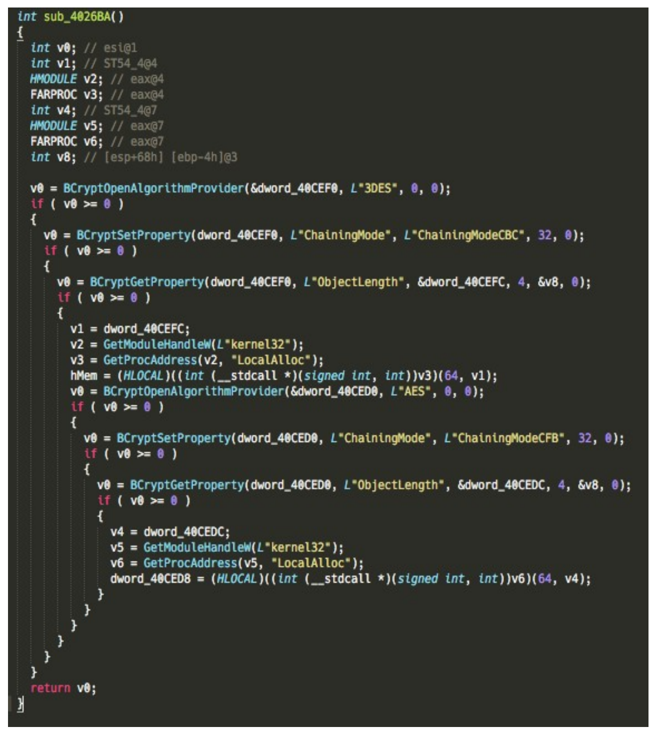
1. Stealing credentials using password scraping tool or re-using existing active sessions using file-shares to transfer the malicious file across machines on the same network
2. Using existing legitimate functionalities to execute the payload or abusing SMB vulnerabilities for unpatched machines.



Lateral infection flow

The malware brings [Mimikatz] modified code (32 and 64 bit, for each lsass version it encountered) in its resource section.

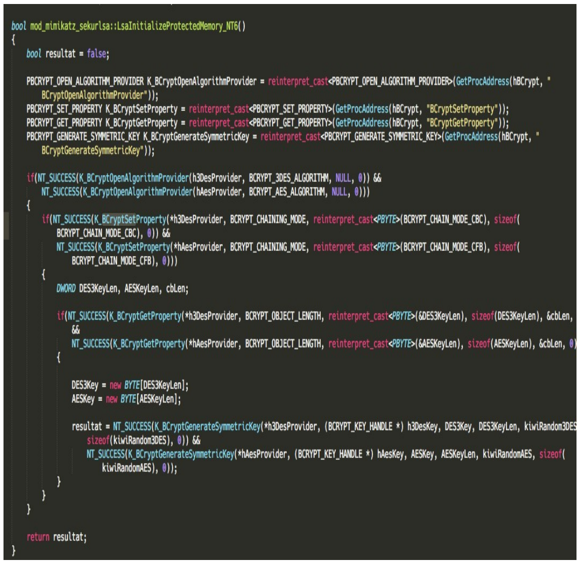
We can see code similarity to Mimikatz in the following example:



```
int sub_4026BA()
{
  int v0; // esi@1
  int v1; // ST54_4@4
  HMODULE v2; // eax@4
  FARPROC v3; // eax@4
  int v4; // ST54_4@7
  HMODULE v5; // eax@7
  FARPROC v6; // eax@7
  int v8; // [esp+68h] [ebp-4h]@3

  v0 = BCryptOpenAlgorithmProvider(&dword_40CEF0, L"3DES", 0, 0);
  if ( v0 >= 0 )
  {
    v0 = BCryptSetProperty(dword_40CEF0, L"ChainingMode", L"ChainingModeCBC", 32, 0);
    if ( v0 >= 0 )
    {
      v0 = BCryptGetProperty(dword_40CEF0, L"ObjectLength", &dword_40CEFC, 4, &v8, 0);
      if ( v0 >= 0 )
      {
        v1 = dword_40CEFC;
        v2 = GetModuleHandleW(L"kernel32");
        v3 = GetProcAddress(v2, "LocalAlloc");
        hMem = (HLOCAL)((int (__stdcall *)(signed int, int))v3)(64, v1);
        v0 = BCryptOpenAlgorithmProvider(&dword_40CED0, L"AES", 0, 0);
        if ( v0 >= 0 )
        {
          v0 = BCryptSetProperty(dword_40CED0, L"ChainingMode", L"ChainingModeCFB", 32, 0);
          if ( v0 >= 0 )
          {
            v0 = BCryptGetProperty(dword_40CED0, L"ObjectLength", &dword_40CEDC, 4, &v8, 0);
            if ( v0 >= 0 )
            {
              v4 = dword_40CEDC;
              v5 = GetModuleHandleW(L"kernel32");
              v6 = GetProcAddress(v5, "LocalAlloc");
              dword_40CED8 = (HLOCAL)((int (__stdcall *)(signed int, int))v6)(64, v4);
            }
          }
        }
      }
    }
  }
  return v0;
}
```

Pseudo code from eae9771e2eeb7ea3c6059485da39e77b8c0c369232f01334954fbac1c186c998

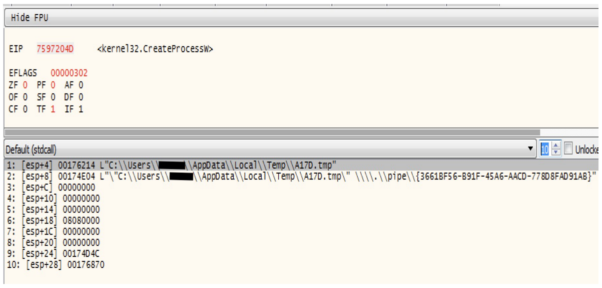


Mimikatz source code

This tool allows the attacker to scrape the credentials from lsass allowing it to further propagate over the network.

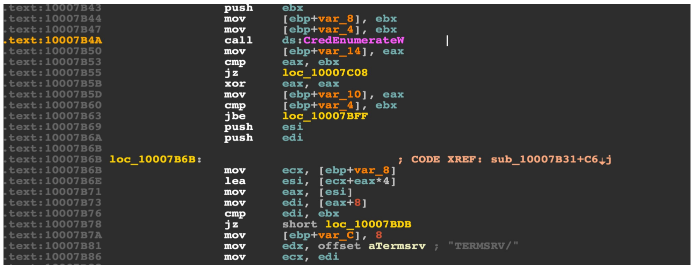
The attacker's assumption seems that most users will have Admin privileges hence the stolen credentials could allow it to spread with high privileges.

The Sample execute the Mimikatz clone using `CreateProcess()` and named pipe.



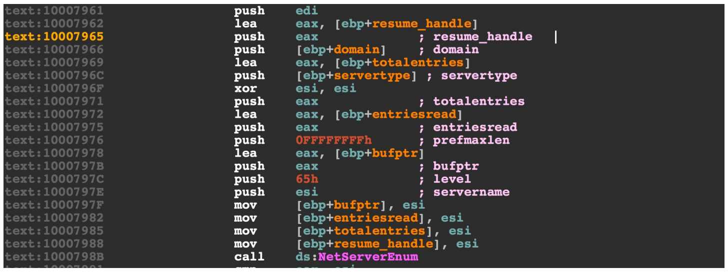
Mimikatz execution

In addition to Mimikatz credential scrapping, the malware also tries to steal credentials by using the `CredEnumerateW()` function to get other user credentials potentially stored on the windows credential store. If a credential name starts with "`TERMSRV/`" and the type is set as 1 (generic) it uses that credential to propagate through the network.



Scraping credentials from the credential store

When getting executed it will scan for microsoft network using the function `NetServerEnum()`. This function lists all servers of the specified type that are visible in a domain.



List all visible servers in a domain

The malware scans the local microsoft network on ports tcp/139 and tcp/445. The scan is probably to find candidates for the exploit in case there is no domain and it failed to scrape credentials.

In order to discover the network segments, the malware calls `DhcpEnumSubnets()` to enumerate dhcp subnets.

```
text:100090D1    push    eax             ; ElementsTotal
text:100090D2    lea     eax, [ebp+ElementsRead]
text:100090D5    push    eax             ; ElementsRead
text:100090D6    lea     eax, [ebp+EnumInfo]
text:100090D9    push    eax             ; EnumInfo
text:100090DA    push    400h            ; PreferredMaximum
text:100090DF    lea     eax, [ebp+ResumeHandle]
text:100090E2    push    eax             ; ResumeHandle
text:100090E3    lea     eax, [ebp+Buffer]
text:100090E9    push    eax             ; ServerIpAddress
text:100090EA    call    ds:DhcpEnumSubnets
text:100090F0    test    eax, eax
text:100090F2    jnz     loc_100091E8
text:100090F8    mov     eax, [ebp+EnumInfo]
text:100090FB    mov     eax, [eax]
text:100090FD    mov     [ebp+var_38], eax
```

Scanning the network and enumerating subnets

When the malware finds a valid remote machine, its connects authenticate using the scraped credentials, copies itself to the remote machine, and execute it using WMIC or PSEXEC.

The malware tries to copy the legitimate `psexec.exe` (typically renamed to `dllhost.dat`) from its resources section. It then copies itself over the network, executes its own copy remotely using PSEXEC.

```
text:100098D7    jz      loc_10009968
text:100098DD    push    offset aWbemWmic_exe ; "wbem\\wmic.exe"
text:100098E2    push    edi             ; pszPath
text:100098E3    call    ds:PathAppendW
text:100098E9    push    edi             ; pszPath
text:100098EA    call    ds:PathFileExistsW
text:100098F0    test    eax, eax
text:100098F2    jz      short loc_1000996E
text:100098F4    push    [ebp+arg_8]
text:100098F7    mov     ebx, ds:wsprintfW
text:100098FD    push    [ebp+arg_4]
text:10009900    push    [ebp+arg_0]
text:10009903    push    edi
text:10009904    push    offset aSNodeWsUserWsP ; "%s /node:\"%ws\" /user:\"%ws\" /passwor"...
text:10009909    push    esi             ; LPWSTR
text:1000990A    call    ebx ; wsprintfW
text:1000990C    mov     edi, eax
text:1000990E    lea     eax, [ebp+var_208]
text:10009914    push    eax
text:10009915    lea     eax, [esi+edi*2]
text:10009918    push    offset aProcessCallCre ; "process call create \"C:\\Windows\\Syst"...
text:1000991D    push    eax             ; LPWSTR
text:1000991E    call    ebx ; wsprintfW
text:10009920    add     edi, eax
text:10009922    add     esp, 24h
```

Executing the malware on the remote machine using WMI

```
text:10008859    push    ebx             ; dwDesiredAccess
text:1000885A    push    [esp+12B4h+TokenHandle] ; hExistingToken
text:1000885E    call    ds:DuplicateTokenEx
text:10008864    test    eax, eax
text:10008866    jz      loc_100088EE
text:1000886C    push    38h             ; Size
text:1000886E    lea     eax, [esp+12A4h+var_1268]
text:10008872    push    esi             ; Val
text:10008873    push    eax             ; Dst
text:10008874    call    memset
text:10008879    add     esp, 0Ch
text:1000887C    lea     eax, [esp+12A0h+ReturnLength]
text:10008880    push    eax             ; ReturnLength
text:10008881    push    38h             ; TokenInformationLength
text:10008883    lea     eax, [esp+12A8h+var_1268]
text:10008887    push    eax             ; TokenInformation
text:10008888    push    0Ah             ; TokenInformationClass
text:1000888A    push    [esp+12B0h+phNewToken] ; TokenHandle
text:1000888E    call    edi ; GetTokenInformation
text:10008890    test    eax, eax
```

Duplicate tokens of the existing connections

Encryption

Before the MBR encryption, the sample will encrypt individual files on the system. Unlike other ransomware, it does not rename the file with an extension to identify the encrypted files.

The encryption routine used is AES-128 in CBC mode. A random key is generated per drive.

```
v1 = (HCRYPTKEY *)&crypto_context->hAESKey;
v5 = CryptGenKey(crypto_context->hProvider, CALG_AES_128, 1u, (HCRYPTKEY *)&crypto_context->hAESKey);// CRYPT_EXPORTABLE
if ( v5 )
{
        v2 = *v1;
        *(_DWORD *)pbData = 1;          // CRYPT_MODE_CBC: 1
        CryptSetKeyParam(v2, KP_MODE, pbData, 0);
        v3 = *v1;
        *(_DWORD *)v6 = 1;              // PKCS5_PADDING: 1
        CryptSetKeyParam(v3, KP_PADDING, v6, 0);
}
return v5;
```

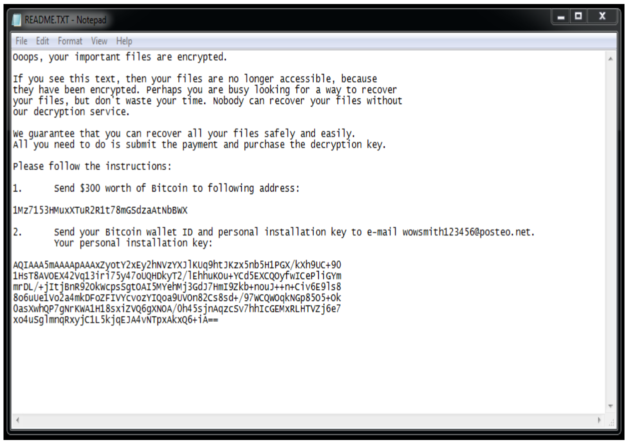
Random key generation

Encrypting an individual file is accomplished by mapping the file into memory, and running `CryptEncrypt()` over the first megabyte of the mapped file.

```
buff = MapViewOfFile(v5, 6u, 0, 0, (SIZE_T)file_size);
if ( buff )
{
        if ( CryptEncrypt(
                        crypto_context->hAESKey,
                        0,
                        Final,
                        0,
                        (BYTE *)buff,
                        (DWORD *)&file_size,
                        buff_size) )
        {
                FlushViewOfFile(buff, (SIZE_T)file_size);
        }
        UnmapViewOfFile(buff);
}
CloseHandle(hObject);
```

File encryption

After encrypting every file on the drive with the correct extension, the sample will then drop a ransom note to the root of the drive.



Ransom message

The base64 string at the end of the ransom note is the AES key that has been encrypted with the attacker's RSA public key.

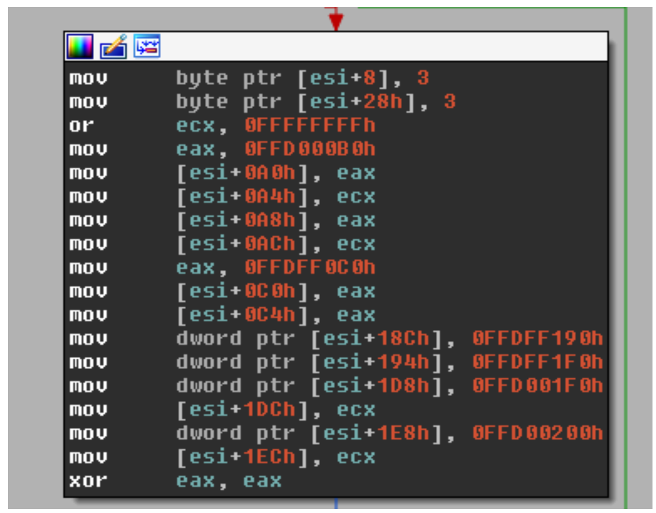
EternalBlue Exploit

As mentioned above, one lateral movement technique used by the malware if exploiting the recently discovered, believed to be developed by the NSA and known by the codename EternalBlue, given CVE ID CVE-2017-0144. The EternalBlue exploit was recently involved in another widespread worm dubbed WannaCry (AKA WannaCrypt), where Eternal-Blue was the main means of spreading.

The vulnerability exists in Window's file sharing protocol, called Server Message Block (or SMB for short). By sending a specially crafted packet to a remote computer running Windows on the SMB port (TCP/445) using version 1 of the protocol (SMBv1) allows the NotPetya malware to gain remote code execution abilities on victim computers. Microsoft issued a patch on March (MS17-010), but many users have failed to apply.

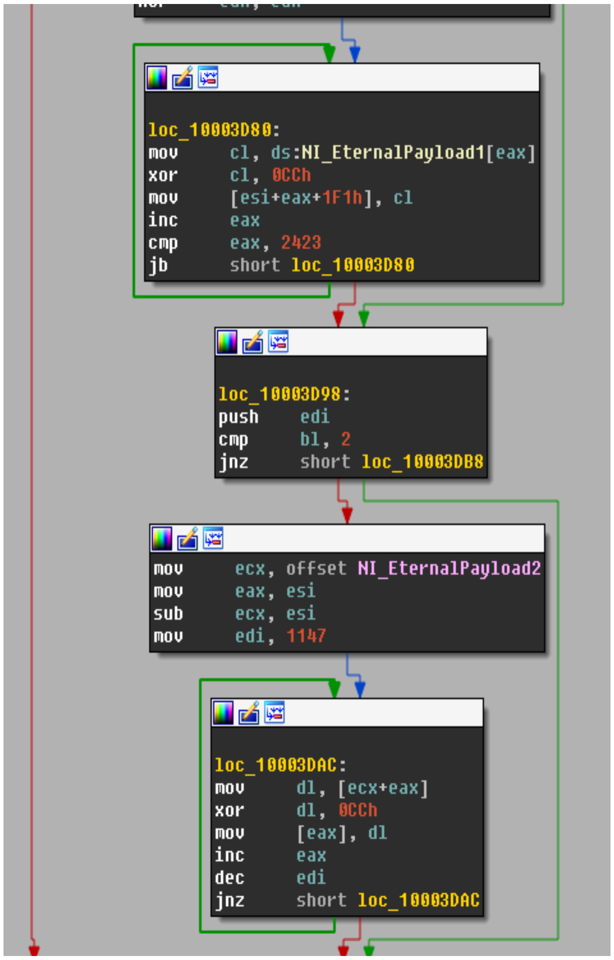
Usually as a last resort, after more "conventional" lateral movement techniques have been exhausted, NotPetya will resort to using the EternalBlue exploit it's packing in its resource sections.

The exploitation process starts at `sub_10005A7E`, which sets-up connections to potential victims after other infection approaches failed, and then goes on to calling `sub_10003CA0` which is in charge of decrypting and delivering the payloads to victims.



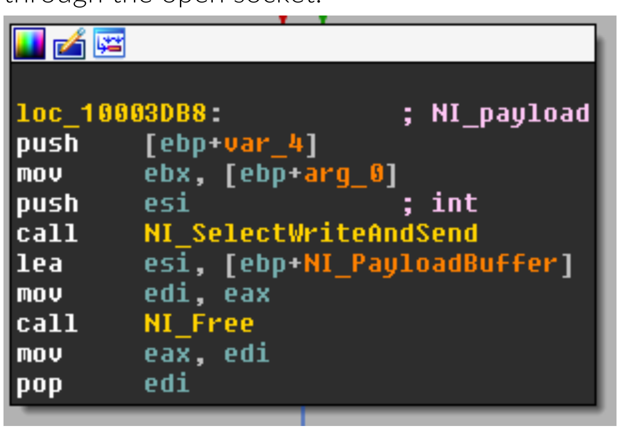
Construction of modified EternalBlue exploit

After manually constructing portions of the exploit, payload construction is finished by decrypting and adding two sections packed in the malware's resource section, as seen in the following figure:



Decryption of EternalBlue packets packed in malware's resources

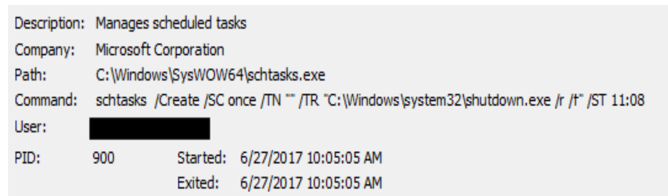
In the last figure, we can see how the previously constructed packet is delivered through the open socket:



Writing packet to open socket

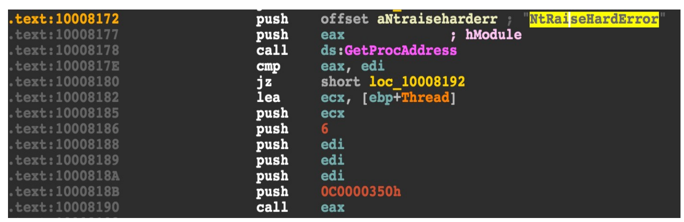
Reboot

Unlike typical MBR ransomware that reboots immediately after infecting the MBR, this sample needs time for its worm functionality to run. A task is created and scheduled for one hour after initial infection.



Scheduled "failsafe" task

The scheduled task seems to be a failsafe if the sample crashes or ends prematurely because it implements Petya's technique of calling `NtRaiseHardError()` to force a hard reboot.



Forcing reboot with NtRaiseHardError

Bitcoin Analysis

The Bitcoin ransom payment address is 1Mz7153HMuxXTuR2R1t78mGSdzaAtNbBWX. As of this writing, it has received 45 transactions totaling to 3.99 XBT (about $10400 @ $2600). Payments to this address have come from several exchanges including Coinbase and Poloniex.

Transactions were coming in quickly on the first day of the outbreak (June 27th) but have slowed considerably in the second day.

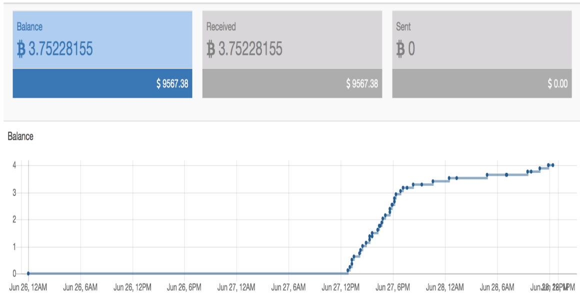
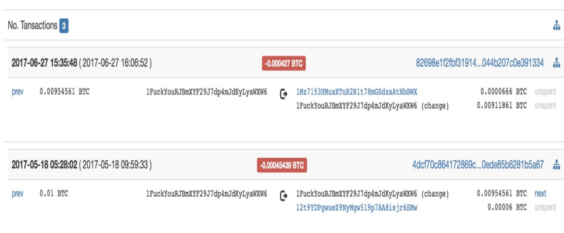


Image courtesy of [Neutrino](#)

It's possible that other variants exist in the wild with different ransom payment addresses, but they're not known to us.

As of now, none of the payments have moved from the original address so further analysis is not yet possible. However, one interesting observation is that the address 1FuckYouRJBmXYF29J7dp4mJdKyLyaWXW6 sent a small amount of BTC to both this ransom address and a ransom address associated with WannaCry.



“1FuckYouRJB…” transactions

Since the amounts sent were less than the ransoms, it's not clear what the payments were for. It seems unlikely that the address owner was hit by two ransomware families and improperly paid the ransom for both.

Bitcoin analysis was aided by Neutrino.

Conclusion

Worms haven't been prevalent in the past several years mainly because improvements to the Windows Update mechanisms. With the success of Miria and WannaCry, malicious actors are reassessing the effectiveness of worms in the "Automatic Update" era.

This sample seems to be developed for the purpose of damaging rather to extorting (ransomware). The code quality is good while the ransom part is sloppy. Seems like the actor wanted to sabotage the infected system rather gaining money out of it.

SentinelOne agent has detected and prevented this attack for all of our customers.

SentinelOne
Endpoint Protection Platform