The Petya/NotPetya outbreak appears to have originated in Ukraine on June 27, 2017. It quickly spread globally within hours. The attack caught security researchers’ attention due to the rapid reports of global infections and because it used the same EternalBlue SMB exploit that was employed by WannaCry to spread to new machines. As it turned out, the malware used in this attack wasn’t WannaCry, but instead a variant of the Petya wiper that first emerged in March of 2016. It is also commonly referred to as Petya.A, Petwrap, NotPetya, exPetr, and GoldenEye. This particular Petya variant is different because it features a different encryption algorithm implementation than what was used before and looks to target different file types than previous variations. Originally mistaken as a form of ransomware due to the similarities with WannaCry, Petya is actually classified as wiper; even after victims pay the fee, all their data is wiped.

Petya wreaks havoc by locking the hard drive MFT and MBR sections and by preventing computers from booting. Victims are then blackmailed into paying a ransom, which is not recommended, or risk recovering their systems. It is said that the attackers are requesting $300 in Bitcoin per incident and that over 2000 users have been reported to be infected. Victims quickly discovered, however, that they were unable to get any of their data back - even after paying the ransom. The Ukraine government, banks, and electricity grids were hit the hardest, but companies in France; Denmark; and Pittsburgh, Pennsylvania have also been attacked.

The Petya wiper, unlike conventional ransomware, has more damaging consequence as it takes control of the infected host by modifying the Master Boot Record (MBR). Once the MBR is infected, it sets up a scheduled task to reboot after an hour of infection and upon reboot, the MBR code takes control of the boot sequence and prevents the operating system from coming up until the ransom key is provided.

As part of compromising the host system, it gets the handle to the physicalDrive and determines the disk parameters using Windows API DeviceIOControl(). Attempts are then made to overwrite the boot sector of the Disk.

Below are snapshots of the code, which reveal this functionality of the malware:

**Determine the Physical Disk**

```
 mov [local.9], ds:0
 mov word ptr ds:[esp-8], eax
 push ds:0
 mov [local.7], ds:0
 stos byte ptr ds:[esi]
 push [arg.1], ds:0
 push [arg.1], ds:0
 call diskconv.masm
 mov ds:0, 027cc450
 pop(ds)
 cmp [arg.1], ds:0
 pop(ds)
 dec [arg.1]
 xor eax, eax
 pop(ds)
 push ds:0
 call diskconv.masm
 push ds:0
 lea ds:0, [local.8]
 mov ds:0, 027cc450
 add ds:0, 0
 push ds:0
 lea ds:0, [local.15]
 mov word ptr ds:[esi], eax
 push ds:0
 lea ds:0, [local.15]
 mov ebx, ds:0
 push ds:0
 mov word ptr ds:[esi], ebx
 push ds:0
lea ds:0, [kernel32.GetSystemDirectoryA
 mov ds:0, 0006f730
 call ds:0
```

---

© 2017 Attivo Networks. All rights reserved.
www.attivonetworks.com
### Determine the Disk Parameters Using IO Control Codes

```assembly
; 75 7F  CALL DWord Ptr DS: [<KERNEL32.CreateFiles>] ; GetLastError
  MOV EAX, 0D0700C7
  JMP SHORT 027cc450.1000141D
  PUSH EDI
  PUSH EBI
  PUSH 3
  PUSH EBI
  PUSH [EBI + 4]
  CALL DWORD PTR DS: [<KERNEL32.CreateFiles>] ; CreateFileA
  MOV EAX, 0D0700C7
  JMP SHORT 027cc450.1000141D
  CALL DWORD PTR DS: [<KERNEL32.GetLastError>] ; GetLastError
  CMP EAX, EDI
  JNZ SHORT 027cc450.100013DZ
  MOV EDX, 0
  CALL DWORD PTR DS: [<KERNEL32.SetFilePointerEx>] ; kernel32.setFilePointerEx
  TEST EAX, EAX
  JNZ SHORT 027cc450.100013FR
  PUSH EDI
  MOV BL, [EAX + LOCAL.1]
  CALL DWORD PTR DS: [<KERNEL32.CreateFiles>] ; CreateFileA
  MOV EAX, 0D0700C7
  JMP SHORT 027cc450.1000141D
  CALL DWORD PTR DS: [<KERNEL32.SetFilePointerEx>] ; kernel32.setFilePointerEx
  CALL DWORD PTR DS: [<KERNEL32.WriteFile>] ; WriteFile
```

### Write to the Disk

```assembly
; 53  PUSH EBX
  53  PUSH EBX
  53  PUSH EBX
  53  PUSH EBX
  53  PUSH EBX
  53  MOV [EBI + 8:EBI + 11], AL
  53  PUSH EBX
  8045984  LEBL [LOCAL.1]
  50  CALL DWORD PTR DS: [<KERNEL32.CreateFiles>] ; CreateFileA
  804598E  MOV [EBI + 12], EBI
  8045990  CALL SHORT 027cc450.1000101D
  74  0C  PUSH EBX
  8045B4  LEBL [LOCAL.7]
  51  PUSH EAX
  6A 20  MOV EDX, 20
  8045BB  LEBL [LOCAL.23]
  51  PUSH EAX
  53  PUSH EBX
  53  PUSH EBX
  53  CALL DWORD PTR DS: [<KERNEL32.DeviceIoControl>] ; DeviceIoControl
  68 00001600  PUSH 1600000
  50  CALL DWORD PTR DS: [<KERNEL32.DeviceIoControl>] ; DeviceIoControl
```

© 2017 Attivo Networks. All rights reserved. www.attivonetworks.com
Once the MBR is overwritten, if the wiper runs under administrative privileges, it attempts to force shutdown of the system by calling the undocumented API “NtRaiseHardError” and then subsequently initiating the system shutdown.

In Attivo Labs, we observed the scheduled task being created on the infected system to reboot after 1 hr. To achieve this, it determines the local system time, appends the path of the windows utility “shutdown /r /f” to the system directory and then executes the schtasks.exe to create the schedule tasks to reboot the system.

Below is the snapshot of the scheduled task created on the system.
Once the system reboots at a scheduled time set by the malware, the overwritten boot loader with the malicious code kicks in and takes complete control of the system. It displays a screen faking the CHKDSK scan.

We believe the malware continues to encrypt the files at this stage. Once the disk is encrypted or a reboot is forced, the malware displays a ransom note with the name of README.TXT indicating the file encryption is complete.

At this stage, the machine is under complete control of the malware. The only way one could move forward or hope to move forward is by providing the decryption key as claimed by the ransom note. However, with this particular piece of malware, that is not possible.
Alerts raised by the BOTsink

The following alerts are raised by the Attivo BOTsink upon detection of malware’s activity:
Theft of deceptive credentials picked up by the wiper to move laterally to one of the engagement VM’s to compromise it.

In our research, Attivo Labs captured the following tactics, techniques and procedures of the compromise:
The Petya wiper used the built-in administrative share “ADMIN$” to connect to target machines. The PCAP file captured on the BOTsink show the credential and share that it connected to.

<table>
<thead>
<tr>
<th>Time</th>
<th>Source</th>
<th>Destination</th>
<th>Protocol</th>
<th>Len</th>
<th>Info</th>
</tr>
</thead>
<tbody>
<tr>
<td>13865.777718</td>
<td>1.1.1.51</td>
<td>1.1.1.1</td>
<td>SMB2</td>
<td>2</td>
<td>1. Negotiate Protocol Response</td>
</tr>
<tr>
<td>13865.789748</td>
<td>1.1.1.1</td>
<td>1.1.1.51</td>
<td>SMB2</td>
<td>3</td>
<td>2. Session Setup Request, NTLMSP_NEGOTIATE</td>
</tr>
<tr>
<td>13865.780159</td>
<td>1.1.1.51</td>
<td>1.1.1.1</td>
<td>SMB2</td>
<td>3</td>
<td>3. Session Setup Response, Error: STATUS_MORE_PROCESSING_REQUIRED</td>
</tr>
<tr>
<td>13865.781831</td>
<td>1.1.1.1</td>
<td>1.1.1.51</td>
<td>SMB2</td>
<td>6</td>
<td>4. Session Setup Request, NTLMSPAUTH_USER: User: 1.1.1.51!mysql_login</td>
</tr>
<tr>
<td>13865.782252</td>
<td>1.1.1.1</td>
<td>1.1.1.51</td>
<td>SMB2</td>
<td>1</td>
<td>5. Session Setup Response</td>
</tr>
<tr>
<td>13865.782746</td>
<td>1.1.1.1</td>
<td>1.1.1.51</td>
<td>SMB2</td>
<td>1</td>
<td>6. Create Request File:</td>
</tr>
<tr>
<td>13865.783256</td>
<td>1.1.1.1</td>
<td>1.1.1.51</td>
<td>SMB2</td>
<td>1</td>
<td>7. Find Request File: SMB2_FIND_NAME_INFO Pattern: pattya</td>
</tr>
<tr>
<td>13865.786488</td>
<td>1.1.1.1</td>
<td>1.1.1.51</td>
<td>SMB2</td>
<td>1</td>
<td>8. Find Request, Error: STATUS_NOT_SUCH_FILE</td>
</tr>
<tr>
<td>13865.785337</td>
<td>1.1.1.1</td>
<td>1.1.1.51</td>
<td>SMB2</td>
<td>1</td>
<td>9. Close Request File:</td>
</tr>
<tr>
<td>13865.786450</td>
<td>1.1.1.1</td>
<td>1.1.1.51</td>
<td>SMB2</td>
<td>1</td>
<td>10. Create Request File:</td>
</tr>
<tr>
<td>13865.786743</td>
<td>1.1.1.1</td>
<td>1.1.1.51</td>
<td>SMB2</td>
<td>1</td>
<td>11. Find Request File: SMB2_FIND_NAME_INFO Pattern: pattya</td>
</tr>
<tr>
<td>13865.787148</td>
<td>1.1.1.1</td>
<td>1.1.1.51</td>
<td>SMB2</td>
<td>1</td>
<td>12. Find Request, Error: STATUS_NOT_SUCH_FILE</td>
</tr>
<tr>
<td>13865.787488</td>
<td>1.1.1.1</td>
<td>1.1.1.51</td>
<td>SMB2</td>
<td>1</td>
<td>13. Close Request File:</td>
</tr>
<tr>
<td>13865.788151</td>
<td>1.1.1.1</td>
<td>1.1.1.51</td>
<td>SMB2</td>
<td>1</td>
<td>14. Create Request File:</td>
</tr>
<tr>
<td>13865.788689</td>
<td>1.1.1.1</td>
<td>1.1.1.51</td>
<td>SMB2</td>
<td>1</td>
<td>15. Find Request File: SMB2_FIND_NAME_INFO Pattern: pattya</td>
</tr>
<tr>
<td>13865.789389</td>
<td>1.1.1.1</td>
<td>1.1.1.51</td>
<td>SMB2</td>
<td>1</td>
<td>16. Find Request, Error: STATUS_NOT_SUCH_FILE</td>
</tr>
<tr>
<td>13865.790246</td>
<td>1.1.1.1</td>
<td>1.1.1.51</td>
<td>SMB2</td>
<td>2</td>
<td>17. Create Request File:</td>
</tr>
<tr>
<td>13865.790638</td>
<td>1.1.1.1</td>
<td>1.1.1.51</td>
<td>SMB2</td>
<td>2</td>
<td>18. Close Request File:</td>
</tr>
<tr>
<td>13865.790943</td>
<td>1.1.1.1</td>
<td>1.1.1.51</td>
<td>SMB2</td>
<td>2</td>
<td>19. Create Request File:</td>
</tr>
<tr>
<td>13865.791237</td>
<td>1.1.1.1</td>
<td>1.1.1.51</td>
<td>SMB2</td>
<td>2</td>
<td>20. Create Request File:</td>
</tr>
<tr>
<td>13866.792252</td>
<td>1.1.1.1</td>
<td>1.1.1.51</td>
<td>SMB2</td>
<td>2</td>
<td>21. Create Request File:</td>
</tr>
<tr>
<td>13866.792252</td>
<td>1.1.1.1</td>
<td>1.1.1.51</td>
<td>SMB2</td>
<td>2</td>
<td>22. Create Request File:</td>
</tr>
<tr>
<td>13866.792252</td>
<td>1.1.1.1</td>
<td>1.1.1.51</td>
<td>SMB2</td>
<td>2</td>
<td>23. Create Request File:</td>
</tr>
<tr>
<td>13866.792252</td>
<td>1.1.1.1</td>
<td>1.1.1.51</td>
<td>SMB2</td>
<td>2</td>
<td>24. Create Request File:</td>
</tr>
<tr>
<td>13866.792252</td>
<td>1.1.1.1</td>
<td>1.1.1.51</td>
<td>SMB2</td>
<td>2</td>
<td>25. Create Request File:</td>
</tr>
</tbody>
</table>
• Once connected, it copied the wiper binary to the C:\WINDOWS directory. Payload drop alerts are seen in the BOTsink dashboard capturing the files and processes used for compromise.

• File encryption alerts on the engagement decoys
- Petya wiper clearing out the events from Windows event log

The Attivo BOTsink analyzed every payload drop using a rich set of YARA rules. These rules were then used to detect behaviors that were deemed malicious. A screen capture of the report is seen below:
The malware moving laterally within the BOTsink deception network was captured and studied. Below is a capture of the malware attempting to spread laterally within the BOTsink engagement server. Additionally, we saw Petya attempting to scan for vulnerable machines on the same network. Note that this behavior differs from WannaCry, which reaches out to the Internet to scan for vulnerable machines.
How Attivo Prevents the Spread of Petya

The wiper attempts lateral movement by stealing credentials and trying to compromise systems in the network. To detect lateral movement, the Attivo BOTsink solution deploys engagement decoys across the network and the ThreatStrike Endpoint Deception Suite installs credential lures on all endpoints, collectively leading the wiper to the BOTsink engagement server. Using one of the many security vendor integrations, the BOTsink can prevent the wiper from spreading to other systems in the network.

The diagram below illustrates how the BOTsink prevents the spread of Petya:

The diagram below illustrates how Petya could spread throughout the network if the Attivo Networks solutions were not deployed.
Indicators of Compromise

Dropped EXEs:
- 2813d34f6197eb4df42c886ec7f234a1 (MD5)
- 7e37ab34ecdc3e77e24522ddfd4852d (MD5)

DLLs:
- 71b6a493388e7d0b40c83ce903bc6b04 (MD5)

Quarantine

Once the wiper engaged the decoy’s, the BOTsink analysis engine was set up to automatically send detailed attack forensics through integrations to 3rd party end-point containment providers. The BOTsink solution integrates with a comprehensive list of end-point security tools including Aruba, Carbon Black, McAfee, Cisco, and ForeScout. Security teams can then choose to either automatically or, through the BOTsink user interface, manually quarantine the threat to stop it from spreading to other systems in their network.

Conclusion

The Attivo BOTsink decoy system and ThreatSrike endpoint lures, working together, have been proven to efficiently detect Petya/NotPetya and other variants of malware. The solution provides accurate detection without reliance on signatures, pattern matching, or software updates.

Additional Information:

Possible Killswitch, Found for Petya (NotPetya) Wiper Outbreak
Cybereason security researcher Amit Serper has found a way to prevent the Petya (NotPetya/SortaPetya/Petna) wiper from infecting computers.


Cautionary note regarding spoofing activity

It is being reported that bad actors are attempting to gain credential access through phishing attempts claiming to be Government Agency entities helping to resolve vulnerabilities related to the ongoing campaign.

Organizations are reminded to verify the links and security certificates of any such email. Examples include [.com] vs [.gov] extensions on supposed agency websites. Note: Such phishing is not exclusive or limited to government agency spoofing.

About Attivo Networks

Attivo Networks® is the leader in dynamic deception technology for real-time detection, analysis, and accelerated response to advanced, credential, insider, and ransomware cyber-attacks. The Attivo ThreatDefend Deception and Response Platform accurately detects advanced in-network threats and provides scalable continuous threat management for user networks, data centers, cloud, IoT, ICS-SCADA, and POS environments. www.attivonetworks.com