Global Information Assurance Certification Paper

Automated Detection and Disinfection of Ransomware Attacks using Roadblock Software

GIAC (GREM) Gold Certification

Author: Hemant Kumar, hemantkumargrem@gmail.com
Advisor: Christopher Walker, CISSP
Accepted: November 15, 2019

Abstract

We often hear about ransomware locking data and demanding the ransom. Ransomware is a kind of malware that prohibits users from accessing their system or files and mostly requires a ransom payment to regain access. This results in data loss, downtime, lost productivity, including reputational harm. Financial losses from ransomware attacks are predicted to exceed 11.5 Billion Dollars in 2019 with ransomware attacks on businesses every 14 seconds.

The extension and complexity of ransomware are advancing at a high rate. Malware authors utilize several sophisticated techniques to evade current security defenses, and all the encryption happens in less than a minute. So, there is a need to develop an automated software that performs detection of various kind of ransomware without depending on the signature of malware, and that can also disinfect the live system against various kind of ransomware attacks under a minute and thus containing the infection from further spreading it to other systems. The software should also notify the incident response team of the detected ransomware attacks and its IOCs so that they can further protect the organization from a similar type of attack.

Roadblock software solves this problem by detecting various kinds of ransomware attacks and dis-infecting the system without any need for a reboot in less than a minute. It leads to no data loss, no downtime, no lost productivity, and no reputational harm. The dis-infection process is not dependent on malware signatures or malware coding, and it works by performing fast and deep forensics of the system that is pre-installed with Roadblock, so that it can detect new ransomware variant.

1. Introduction

Roadblock Software is an automated tool developed by me that detects various kinds of ransomware attacks on the live system and disinfects it without any need for a reboot. It performs several deep forensics of the system in very little time to find the ransomware and its instances that are running on the system and then stops all the ransomware instances and puts it in a quarantine folder for further analysis.

This research paper presents a case study of highly sophisticated ransomware named Petya from a cyber forensic point of view to analyze the activities Petya performs to encrypt the system. Petya is a family of encrypting ransomware that was first discovered in 2016. The malware targets Microsoft Windows-based systems, infecting the master boot record to execute a payload that encrypts a hard drive's file system table and prevents Windows from booting. It subsequently demands that the user make a payment in Bitcoin to regain access to the system ("Petya (malware)", n.d.).

This research paper also presents Roadblock inner functionalities and how it detects and disinfects the system from the Petya attack.

Video link to Roadblock Software execution: https://youtu.be/8VyugcD3zSA

Roadblock software download links:
Link 1: https://drive.google.com/drive/folders/1suf-PRywCLXpq6WLizrn111iRWWWJ-3w
Link 2: https://sourceforge.net/projects/Roadblock/files/Roadblock.exe/download

2. Analyzing artifacts on a system infected with Petya

2.1. Sample details of Petya

I obtained the sample of Petya ransomware from Virustotal.com. It is a window executable file.

MD5 Hash of Petya: c33ca4d8e0f2d473e943982618e322e6

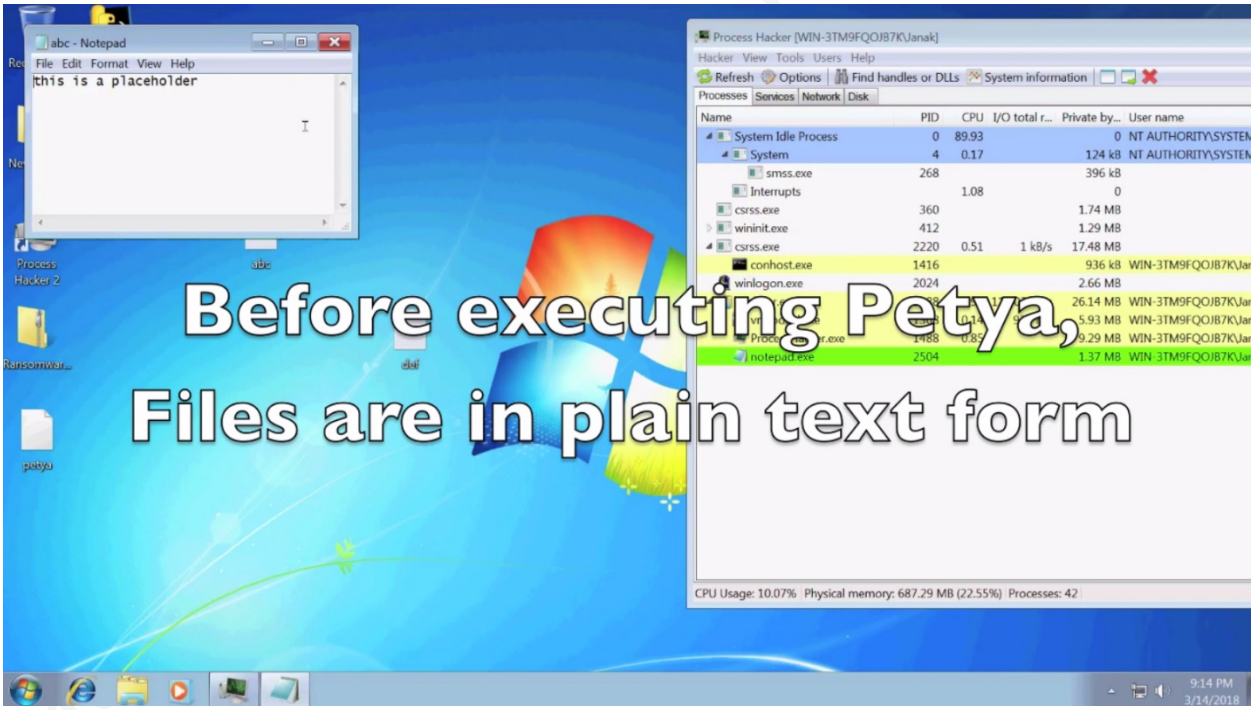
2.2. Details of lab setup

Operating System: Windows 7 Home Basic 64-bit (6.1, Build 7601)

The OS will run virtually using VMware Workstation software.

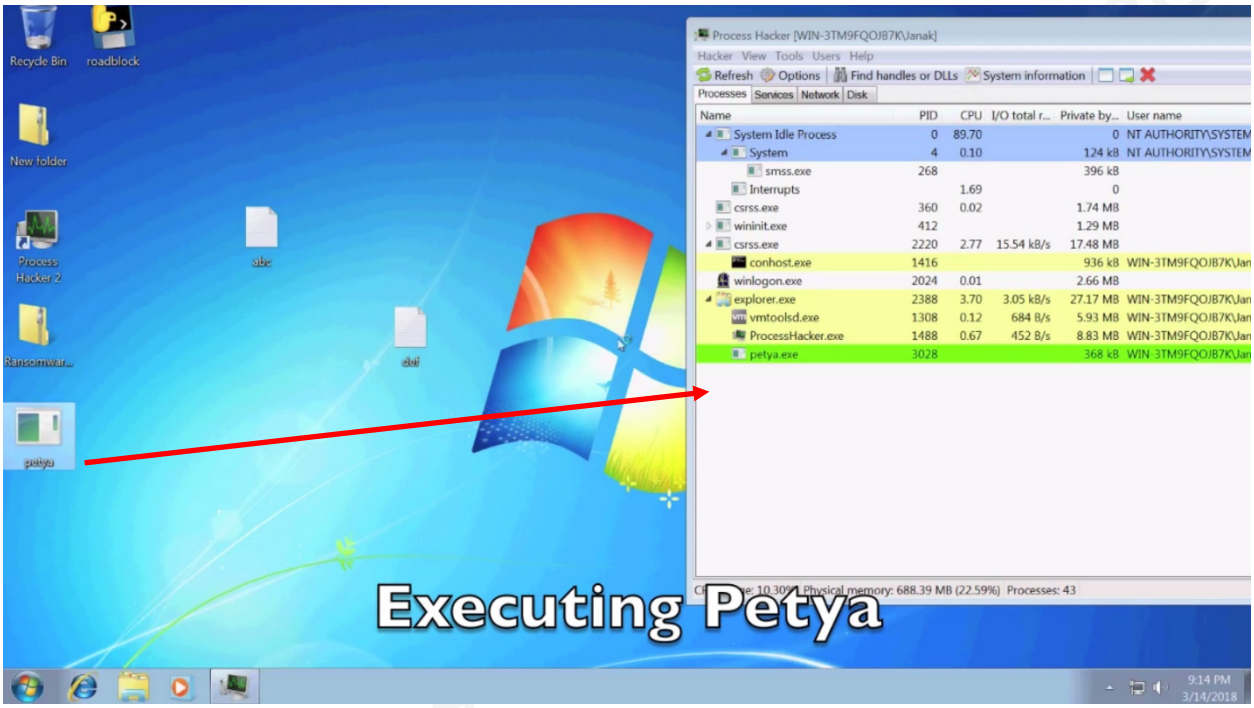
2.3. Executing Petya on the system

2.3.1. System state before execution

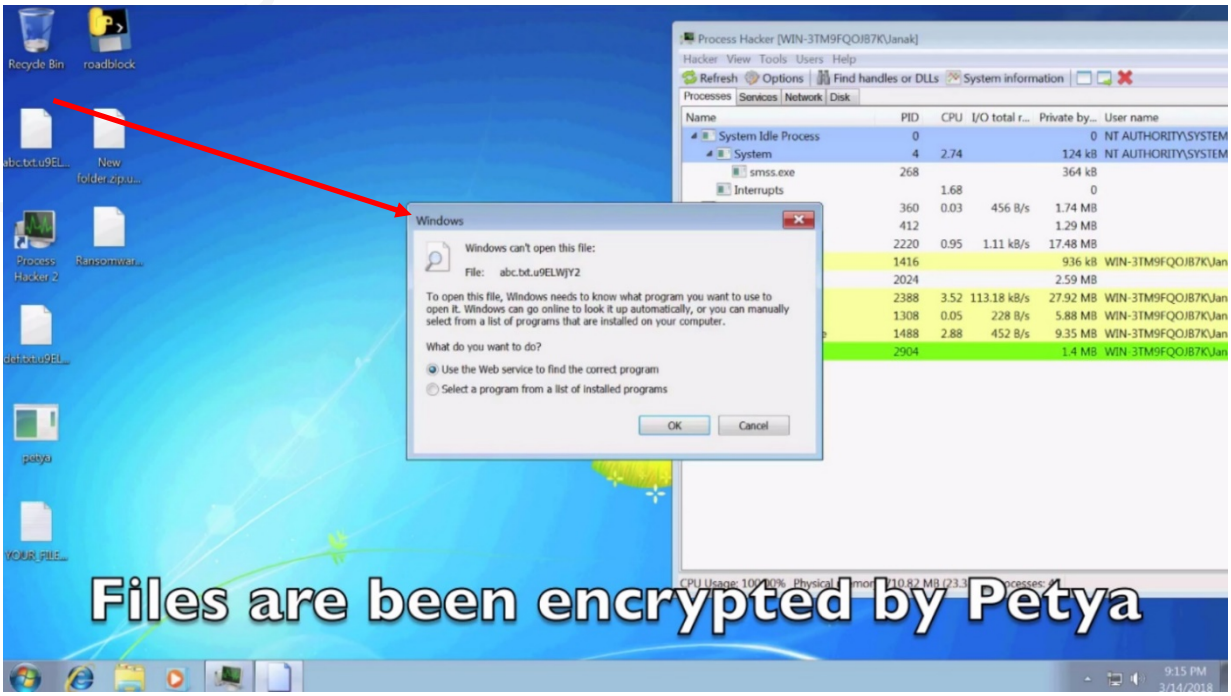


As per the snip, the files on the desktop are in plain text, the right side of the picture shows Process hacker software showing a list of normal windows process running on the system. The app shows detailed information about a process, including its icon, command-line, full image path, memory statistics, user account, security attributes.

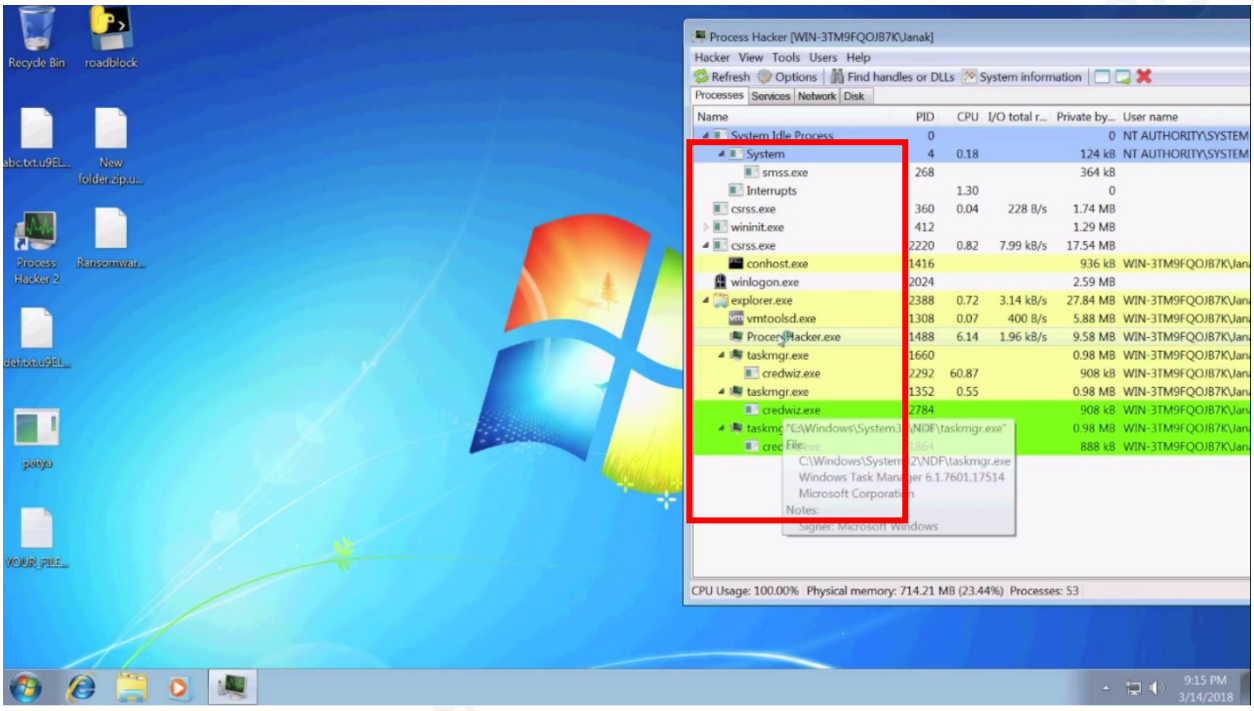
2.3.1. Executing Petya



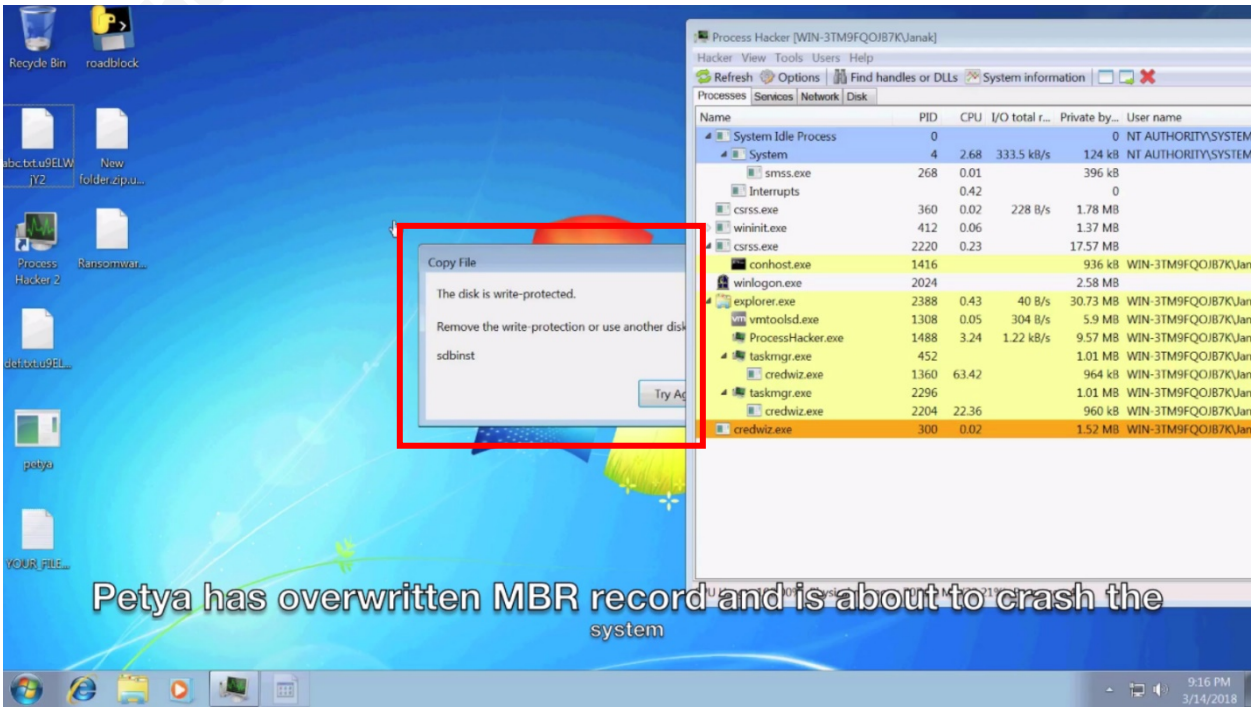
In the snip, we can see that Petya malware is currently present on the desktop. I have executed it by double-clicking the file. We can see petya.exe has started on the system as per Process hacker.



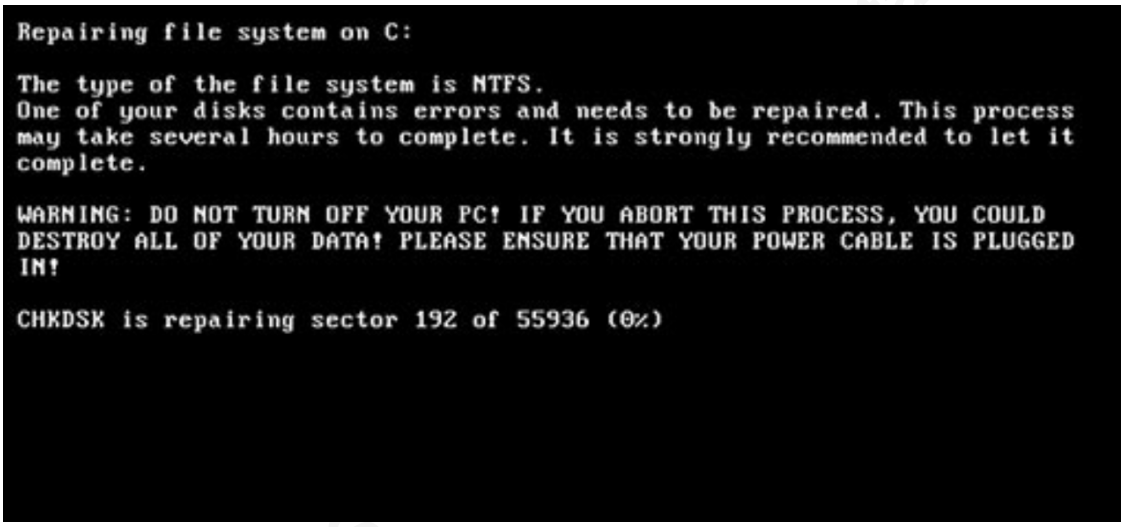
After some time, we see all the files have been encrypted on the system.



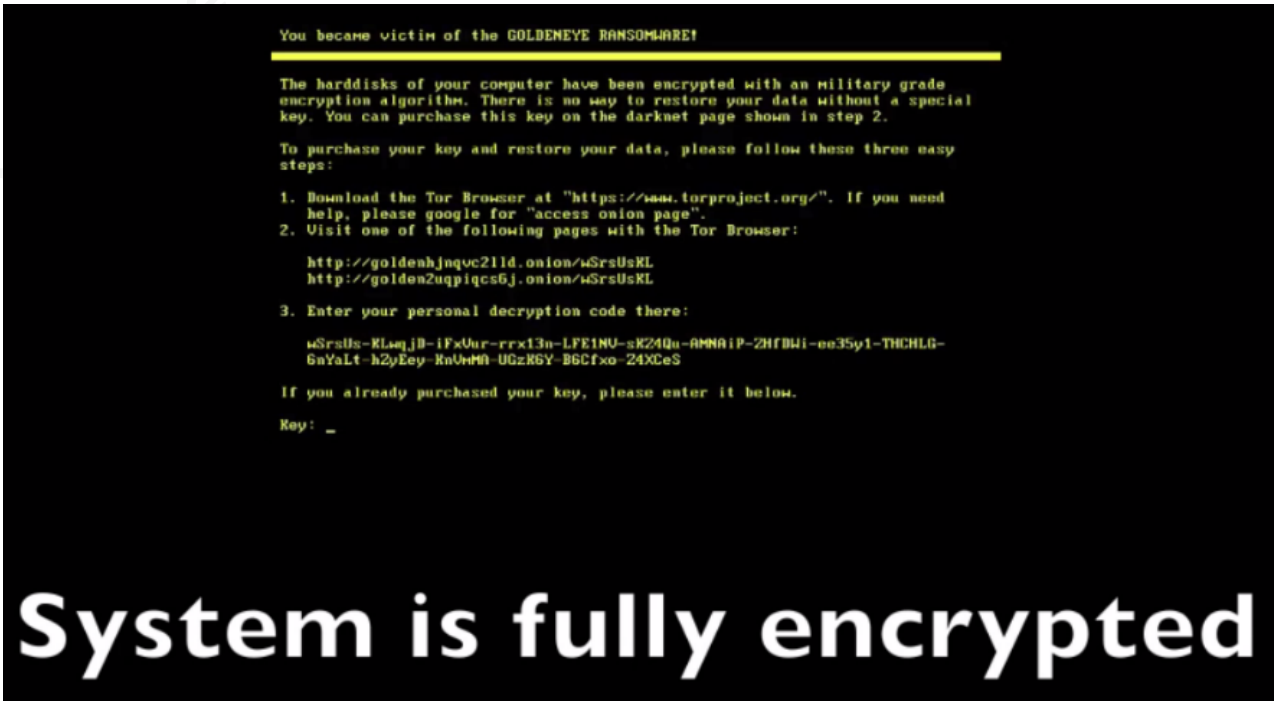
Now, we can see Petya is not running in the system but other processes with name credwiz.exe, taskmanager.exe are running on the system. These processes are normal windows processes but are running malicious code due to dll injection performed by Petya, and these processes encrypt all non-system files present on the system.



After some time, Petya has made changes to the MBR section of the HDD. The Master Boot Record (MBR) is the information in the first sector of any hard disk or diskette that identifies how and where an operating system is located so that it can be boot (loaded) into the computer's main storage or random-access memory (Margaret Rouse, 2005). Petya loads its malicious boot loader by making changes in the MBR and restarts the system.



Once the system restarts, the system does not boot to original Window 7 operating system but rather a fake disk repair screen is shown to the user as shown in the above snip.



After a while, Petya display message saying hard drive is encrypted, and it asks the user to get the decrypted file in return of ransom in the form of bitcoin payment.

2.4. Artifacts created by Petya while it executes on the system

2.4.1 Handles Information:

Handle (a utility developed by Sysinternals) can fetch information about open handles for processes running in the system. It provides a list of files, registry, directories accessed by any the process (Mark Russinovich, n.d.a.).

.

```
1759  ------------------------------------------------------------------------------
1760
1761  PkgMgr.exe pid: 1872 WIN-3TM9FQOJB7K\Testlab
1762
1763     10: File  (RW-)   C:\Windows
1764
1765     1C: File  (RW-)   C:\Users\Testlab\Desktop
1766
1767     20: File  (RW-)
      C:\Windows\winsxs\x86_microsoft.windows.common-controls_6595b64144ccf1df_5.82.7601.17514_none_ec83dffa859149af
1768
1769     24: File  (RW-)
      C:\Windows\winsxs\x86_microsoft.windows.gdiplus_6595b64144ccf1df_1.1.7601.17514_none_72d18a4386696c80
1770
1771     C0: File  (RWD)   C:\Windows\System32
1772
1773   49F4: File  (RWD)   C:\Windows\System32
1774
1775   49F8: File  (RWD)   C:\Windows\System32
1776
1777   4A10: File  (RWD)   C:\Windows\System32
1778
```

When Petya executes on the system, the files present on the system gets encrypted. Dummy files were placed in the Desktop, Documents, Download and User directory folder before infecting the system with Petya. One of the processes named PkMgr.exe with PID 1872 accessed the desktop path. The dummy files were present at this location during encryption.

2.4.2 Procmon logs artifacts:

Process Monitor (a utility developed by Sysinternals) monitors and displays in real-time all file system activity on a Microsoft Windows operating system (Mark Russinovich, n.d.b.).

It monitors and records all actions attempted against the Microsoft Windows Registry. It can be used to detect failed attempts to read and write registry keys. It also allows for filtering on specific keys, processes, process IDs, and values. Also, it shows how applications use files and DLLs, detects some critical errors in system files and more.

In this case, since Petya and its child processes are encrypting files on the drive, important events to look for are Writefile, SetDispositionInformationFile, and Process Create. These events are related to writing, deletion of file, and Process creation. Artifacts related to Petya that are captured by Procmon, particularly for these events are:

WriteFile entries: We can see below that PkgMgr.exe with PID 1872 is performing write operations on dummy files 1aaa_crypto.txt, 99hh_cryto.txt, eee_cryto.txt, zzz_crypto.txt.

```
11:07:56.0042832 PM PkgMgr.exe 1872 WriteFile C:\Users\Testlab\1aaa_crypto.txt SUCCESS Offset: 0, Length: 1,024,
Priority: Normal
11:07:56.0043948 PM PkgMgr.exe 1872 WriteFile C:\Users\Testlab\1aaa_crypto.txt SUCCESS Offset: 1,024, Length:
118, Priority: Normal
11:07:56.0056553 PM PkgMgr.exe 1872 WriteFile C:\Users\Testlab\99hh__crypto.txt SUCCESS Offset: 0, Length: 1,024,
Priority: Normal
11:07:56.0060803 PM PkgMgr.exe 1872 WriteFile C:\Users\Testlab\99hh__crypto.txt SUCCESS Offset: 1,024, Length:
118, Priority: Normal
11:07:56.0072240 PM PkgMgr.exe 1872 WriteFile C:\Users\Testlab\eee_crypto.txt SUCCESS Offset: 0, Length: 1,024,
Priority: Normal
11:07:56.0073233 PM PkgMgr.exe 1872 WriteFile C:\Users\Testlab\eee_crypto.txt SUCCESS Offset: 1,024, Length: 118,
Priority: Normal
11:07:56.0085603 PM PkgMgr.exe 1872 WriteFile C:\Users\Testlab\zzz_crypto.txt SUCCESS Offset: 0, Length: 1,024,
Priority: Normal
11:07:56.0086596 PM PkgMgr.exe 1872 WriteFile C:\Users\Testlab\zzz_crypto.txt SUCCESS Offset: 1,024, Length: 118,
Priority: Normal
11:07:56.0100978 PM PkgMgr.exe 1872 WriteFile C:\Users\Testlab\Downloads\1aaa_crypto.txt SUCCESS Offset: 0,
Length: 1,024, Priority: Normal
11:07:56.0101976 PM PkgMgr.exe 1872 WriteFile C:\Users\Testlab\Downloads\1aaa_crypto.txt SUCCESS Offset: 1,024,
Length: 118, Priority: Normal
11:07:56.0112748 PM PkgMgr.exe 1872 WriteFile C:\Users\Testlab\Downloads\99hh__crypto.txt SUCCESS Offset: 0,
Length: 1,024, Priority: Normal
11:07:56.0113753 PM PkgMgr.exe 1872 WriteFile C:\Users\Testlab\Downloads\99hh__crypto.txt SUCCESS Offset: 1,024,
Length: 118, Priority: Normal
11:07:56.0123717 PM PkgMgr.exe 1872 WriteFile C:\Users\Testlab\Downloads\eee_crypto.txt SUCCESS Offset: 0,
Length: 1,024, Priority: Normal
```

Process Create entries: Below highlighted entries display that petya.exe, PkgMgr.exe processes were running on the system. It also displays their PID and path where this malicious executable is present.

```
11:07:33.1309708 PM svchost.exe 624 Process Create C:\Windows\system32\DllHost.exe SUCCESS PID: 2308, Command
line: C:\Windows\system32\DllHost.exe /Processid:{AB8902B4-09CA-4BB6-B78D-A8F59079A8D5}
11:07:37.0244352 PM Explorer.EXE 2396 Process Create C:\Users\Testlab\Desktop\petya.exe SUCCESS PID: 2284,
Command line: "C:\Users\Testlab\Desktop\petya.exe"
11:07:46.0498603 PM petya.exe 2284 Process Create
C:\Users\Testlab\AppData\Roaming\{db69224a-79cf-499c-a8df-e17b9854625f}\PkgMgr.exe SUCCESS PID: 1872, Command
line: "C:\Users\Testlab\AppData\Roaming\{db69224a-79cf-499c-a8df-e17b9854625f}\PkgMgr.exe"
11:07:52.0430905 PM svchost.exe 624 Process Create C:\Windows\system32\DllHost.exe SUCCESS PID: 952, Command
line: C:\Windows\system32\DllHost.exe /Processid:{AB8902B4-09CA-4BB6-B78D-A8F59079A8D5}
```

2.4.3 Sysmon logs artifacts:

System Monitor (Sysmon, the utility developed by Sysinternals) is a Windows system service and device driver once installed on a system, remains resident across system reboots to monitor and log system activity to the Windows event log. It provides detailed information about process creations, network connections, and changes to file creation time (Mark Russinovich and Thomas Garnier, n.d.). The location of event file is C:\Windows\System32\winevt\Logs\Microsoft-Windows-Sysmon%4Operational.evtx.

When Petya executes, it initiates multiple child processes. If we parse the full Sysmon logs for Petya artifacts, we can get the parent-child relationship of different processes started by Petya.

Parent-Child relationship for Petya.exe captured from Sysmon logs in descending order:

['C:\\Users\\Testlab\\Desktop\\petya.exe', 'C:\\Windows\\explorer.exe', 'C:\\Windows\\System32\\userinit.exe', 'C:\\Windows\\System32\\winlogon.exe', 'C:\\Windows\\System32\\smss.exe', 'C:\\Windows\\System32\\smss.exe', 'System']

Parent-Child relationship in PID form for Petya.exe captured from Sysmon logs in descending order:

['2284', '2396', '2364', '436', '384', '248', '4']

Parent-Child relationship for PkgMgr.exe captured from Sysmon logs in descending order:

['C:\\Users\\Testlab\\AppData\\Roaming\\{db69224a-79cf-499c-a8dfe17b9854625f}\\PkgMgr.exe', 'C:\\Windows\\System32\\cs-CZ\\slui.exe', 'C:\\Windows\\explorer.exe', 'C:\\Windows\\System32\\userinit.exe', 'C:\\Windows\\System32\\winlogon.exe', 'C:\\Windows\\System32\\smss.exe', 'C:\\Windows\\System32\\smss.exe', 'System']

From this, we find PkgMgr.exe was executed by Slui.exe. Further automated mapping of Slui.exe needs to be performed to see if any other instances were created by slui.exe.

Parent-Child relationship in PID form for PkgMgr.exe captured from Sysmon logs in descending order:

['3660', '3620', '2396', '2364', '436', '384', '248', '4']

2.4.4 Autoruns Artifacts

Petya persists on system reboot by adding its instances to several AutoStart locations. Autorunsc tool developed by SysInternals displays what changes Petya has made in the system to persist after a reboot. Autorunsc is a command-line tool that has the most comprehensive knowledge of auto-starting locations of any startup monitor; it shows what programs are configured to run during system bootup or login and when you start various built-in Windows applications like Internet Explorer, Explorer, and media players. These programs and drivers include ones in your startup folder, Run, RunOnce, and other Registry keys (Mark Russinovich, n.d.c.).

From the autoruns entries captured by Autorunsc before and after executing Petya, we can see these are the entries created by Petya, once it executes.

In this case, since Petya was successful in overwriting MBR of the Hard drive with a malicious bootloader, so no suspicious autorun entries were found.

3. Roadblock detection and disinfection process

The Roadblock working methodology includes:

1. Detecting ransomware attacks on the system.

2. Performing an automated analysis of logs generated from utilities like Procmon, Handles, Autoruns, and Sysmon utilities.

3.1 Roadblock Installation Phase

Step1: User executes the Roadblock Executable

Step 2: Roadblock.exe creates Roadblock folder in C Drive

Step 3: Roadblock.exe extract embedded executables from its resource section and keep it inside the Roadblock folder.

The embedded executables are:

Sysmon, Procmon, Handles, Autorunsc,

Psexec: Developed by SysInternals, PsExec lets you execute processes on other systems, complete with full interactivity for console applications. It helps in running Roadblock software and its instances with NT AUTHORITY/SYSTEM privilege.

MBRFilter.Inf: MBR Filter is a disk filter driver designed by Cisco Talos to block write access to the Master Boot Record (MBR). MBR Filter prevents rootkits, boot kits, and ransomware, such as Petya Ransomware, from overriding the operating system's (OS) boot loader (Edmund Brumaghin and Yves Younan, 2016).

Roadblock_Conn: Connect with a centralized database and logs any instance of ransomware attacks on any system installed with Roadblock software.

Roadblock_Desktop: It runs in hidden mode in Session 0 and connects with Roadblock services that are running in Session 1. It changes the wallpaper, raises popup and notification when a ransomware attack gets detected on the system.

Roadblockreal.exe: It installs Sysmon, gets whitelisted autoruns entries, set Roadblock services, create dummy files, installs MBRFilter driver.

Roadblockreal2.exe: It monitors dummy files and activates the malware disinfection process in case of a ransomware attack.

Roadblock_starter.exe: It runs all services of Roadblock every time the system is turned on.

Roadblock_autoprocess.exe: Captures autorun entries when a ransomware attack is detected and compares the entries with whitelisted autorun entries.

Roadblock_procprocess.exe: It handles Procmon utility, captures the procmon logs, and parses it.

Step 4: Request for admin access incase Roadblock.exe is not running with admin privilege.

Step 5: Roadblockreal.exe runs with admin privilege.

Step 6: Roadblockreal.exe then execute a series of the process:

a) Set a service that runs Roadblockstarter.exe
b) Set Roadblock_desktop.exe to autostart with every reboot by making an entry in Software registry hive: Software\Microsoft\Windows\CurrentVersion\Run.

c) Install Sysmon.exe
d) Run Autorunsc.exe to get autoruns entries from the non-infected system and treat the entries as whitelisted autorun entries.
e) Create a hidden dummy file with random names (1aaa_crypto.txt, 99hh_cryto.txt, eee_cryto.txt, zzz_crypto.txt) on a specific location for all users at Desktop, Documents, Downloads, C drive. The random names are such that for a file listing of a folder, these random dummy files come at the top, middle and bottom.
f) Install driver MBRFILTER.INF.
g) Raise popup requesting the user to reboot the system for first time installation.

3.2 Roadblock first time running phase after installation:

All instances of processes created by Roadblock runs with NT AUTHORITY/SYSTEM privilege.

Step 1: Roadblock_starter that runs as a service checks if the Roadblockreal2.exe process is running in the system. If Yes go to Step 2 else go to Step 3

Step 2: Sleep and go to Step1 at every 120 seconds.

Step 3: Execute process Roadblock_conn.exe, Roadblock_procprocess.exe, Roadblockreal2.exe and then Roadblock_starter exits.

Step 4: Roadblock_conn connects with the server and updates the database with information on the system on which software got installed. Information is related to the hostname, IP Address of the system.

Step 5: Roadblockreal2.exe process has now started, and it checks if it's running for the first time in the system. If Yes, go to Step 6 else go to Step 8.

Step 6: Create hidden dummy files with random names on a specific location for all users at Desktop, Documents, Downloads, C drive. The random names are such that when ransomware does file listing of a folder, these random dummy files come at the top, middle and bottom.

Step 7: Get a hash of these dummy files.

Step 8: Monitor for any change in the entropy of the dummy files.

3.3 Roadblock Detecting Ransomware attack

Since Roadblockreal2.exe is monitoring dummy files, so if there is a ransomware attack on the system, these dummy files get encrypted. As soon as these dummy files get encrypted Roadblockreal2.exe performs a series of steps:

Step 1: Run Autorunsc.exe and get autorun entries post-infection.

Step 2: Get current Handles entries by running handles.exe.

Step 3: Raise a Popup on Screen, asking the user if the encryption of hidden dummy files has been intentional. If Yes. go to Step 4 else go to Step 5.

Step 4: Rebuilt dummy files and go to Step 1, if encryption happens again on dummy files else monitors dummy files.

Step 5: Get PID of core windows executable like lsass.exe, winen.exe.

Step 5a: Capture the Sysmon logs and parse it to get the parent-child relationship for all processes running on the system.

Step 6: As Handles entries got captured in Step 2, so now Roadblockreal2.exe checks if any process has accessed the desktop, documents, or download folder.

Based on this it gets suspicious process name with its PID that had a handle to one of the folders where hidden dummy files are located, and with this information it goes to step 10.

Step 7: Parsing the procmon logs and look for events Writefile, SetDispositionInformationFile, and Process Create since these events are related to writing and deletion of dummy files. Get suspicious process name and its PID from these entries, and with this information, it goes to Step 10.

Step 8: Compare autoruns entries captured from Step 1 and compares it with autorun entries captured during the installation of the Roadblock software. Gets new entries of the suspicious process added as autorun with its name and its PID and with this information it goes to Step 10.

Step 9: Does a final check of the process that was running during the time encryption occurred from the Sysmon logs and with suspicious process name and its PID information it goes to step 10.

Step 10: Now, it checks for signatures of the suspicious process using sigcheck.exe, and if the signature is non-valid, then it kills this process by name as well as by its PID. It also gets parent, child, and sub-child information of all processes linked with this suspicious process, and after checking the signature if it turns out to be non-valid, it kills all those processes. After killing the process, it moves the malicious process file from its original path to quarantine folder present inside c:\Roadblock folder. It also notes down the original path of the malicious process.

With the following automated analysis, Roadblock can completely detect and disinfect the system from Petya ransomware. I also tested Roadblock with other ransomware like Locky, Jigsaw malware and it was able to disinfect system under 30 seconds.

4. Codes of Roadblock Software

Roadblock software codes are available on GitHub for download and for further improvements. Link: https://github.com/hemantkumargrem/Roadblock

Steps to compile the executable from the Roadblock.py:

1. Use Pyinstaller to make executable of every python file. (Ex: "pyinstaller roadblock.py --onefile")
2. Embed all the executable (except Roadblock.exe) in Roadblock.exe using Resource hacker software following the sequence as mentioned in Roadblock.py at line no. 127.
3. The final Roadblock.exe is ready.

5. Conclusion

Roadblock software provides a promising way to protect any system against ransomware attacks by detecting and disinfecting the system with a reboot. It detects a ransomware attack on the system if there is any change of hash for any of the monitored dummy files. A popup is raised asking if the user performed the encryption intentionally. If not, then it performs automated analysis of handles logs, Sysmon logs, procmon logs, Autoruns logs. Based on the analysis, it can pinpoint ransomware and its instances running on the

system and then kills it and move the ransomware samples to a quarantine folder for further analysis. It also installs the MBRFilter driver that stops any ransomware to overwrite the MBR of the drive.

Future developments of Roadblock software are:

a) Taking code-injection into consideration when skipping a process analysis if it's signed.
b) Capturing RAM image, the moment ransomware attack is detected.
c) Performing Automated RAM Forensic to get the memory-based suspicious hidden process as well as getting decryption key of the files, if present. My Windows MemDiff Forensics Tool (WMDF) that uses volatility framework in the background can be integrated with Roadblock to perform automated RAM forensic. It is a remote agent that enables an analyst to collect the ram Image from the suspicious system, extract memory-related artifacts from it using Volatility as well as compare those collected artifacts with white-listed artifacts database. This tool got selected in the final review of Volatility Plugin Contest 2016.
d) Monitor entropy of any new file created on the drive to detect encryption. Entropy is the uncertainty metric of a random variable. Entropy above 6 signifies strong encryption. It enables Roadblock to detect ransomware attacks more accurately rather than just monitoring dummy files.
e) Performing automated code analysis of suspicious processes.
f) Adding the capability to kill hidden processes running in memory.
g) Tracking Inode number to track suspicious files created just after ransomware infection.
h) Use of pre-installed PowerShell, event logging features to remove the dependency of analyzing logs generated from third-party tools.

References

Petya (malware) (n.d.). Accessed on 20/04/2019 from https://en.wikipedia.org/wiki/Petya_(malware)

Margaret Rouse (2005). What is Master Boot Record (MBR)? - Definition from WhatIs.com. Accessed on 20/04/2019 from https://whatis.techtarget.com/definition/Master-Boot-Record-MBR

Mark Russinovich (n.d.a.). Handle - Windows Sysinternals Markruss. Accessed on 20/04/2019 from https://docs.microsoft.com/en-us/sysinternals/downloads/handle

Mark Russinovich (n.d.b.). Process Monitor - Windows Sysinternals Markruss. Accessed on 20/04/2019 from https://docs.microsoft.com/en-us/sysinternals/downloads/procmon

Mark Russinovich and Thomas Garnier (n.d.). Sysmon - Windows Sysinternals | Microsoft Docs. Accessed on 20/04/2019 from https://docs.microsoft.com/en-us/sysinternals/downloads/Sysmon

Mark Russinovich (n.d.c.). Autoruns for Windows - Windows Sysinternals | Microsoft Docs. Accessed on 20/04/2019 from https://docs.microsoft.com/en-us/sysinternals/downloads/autoruns

Edmund Brumaghin and Yves Younan (2016). Talos Blog || Cisco Talos Intelligence Group - Comprehensive Threat Intelligence – Atom. Accessed on 20/04/2019 from https://blog.talosintelligence.com/2016/10/mbrfilter.html