

Rerverse Engineer: Lab 4 Report ProcDoc

Jesus Lopez, Alejandro Rodriguez, Eduardo Menendez
Dr. Saidur Rahman

November 3, 2024

Contents

1	ProcDot	2
1.1	Introduction	2
1.2	Malware	2
1.3	Exercise 1: Visualizing Process and Network Activity	2
1.3.1	Objective	2
1.3.2	Steps	2
1.3.3	Questions	2
1.4	Exercise 2: Correlating System and Network Activity	3
1.4.1	Objective	3
1.4.2	Steps	3
1.4.3	Questions	4

Chapter 1

ProcDot

1.1 Introduction

In this lab we are going to be exploring the tool of ProcDot. ProcDot turns thousands of monitored activities into a big behavioral picture, which can be interactively explored making behavioral malware analysis as efficient as it never was done. For the purpose of doing this lab, we used Windows 10 game edition.

1.2 Malware

For these exercises we are going to be using a malware we found on the internet, the name is Eternal Rocks, This is a network worm

we are going to put the link of the original repository for educational purposes Link for the virus

1.3 Exercise 1: Visualizing Process and Network Activity

1.3.1 Objective

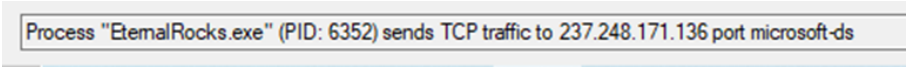
The Objective is to use ProcDot to visualize process creation and network activity of a malware sample.

1.3.2 Steps

1. Ensure ProcDot and Wireshark are installed.
2. Capture system activity with ProcMon while running a malware sample.
3. have ProcMon log as a PML file and Wireshark log as a PCAP file.
4. Load both logs into ProcDOT and generate the graph.

1.3.3 Questions

- Which processes were created or terminated by the malware?
 - So, the malware in this case it created two processes the first one being **Eternal-Rocks.exe** with **PID: 6352** this handles the TCP request. The second one.

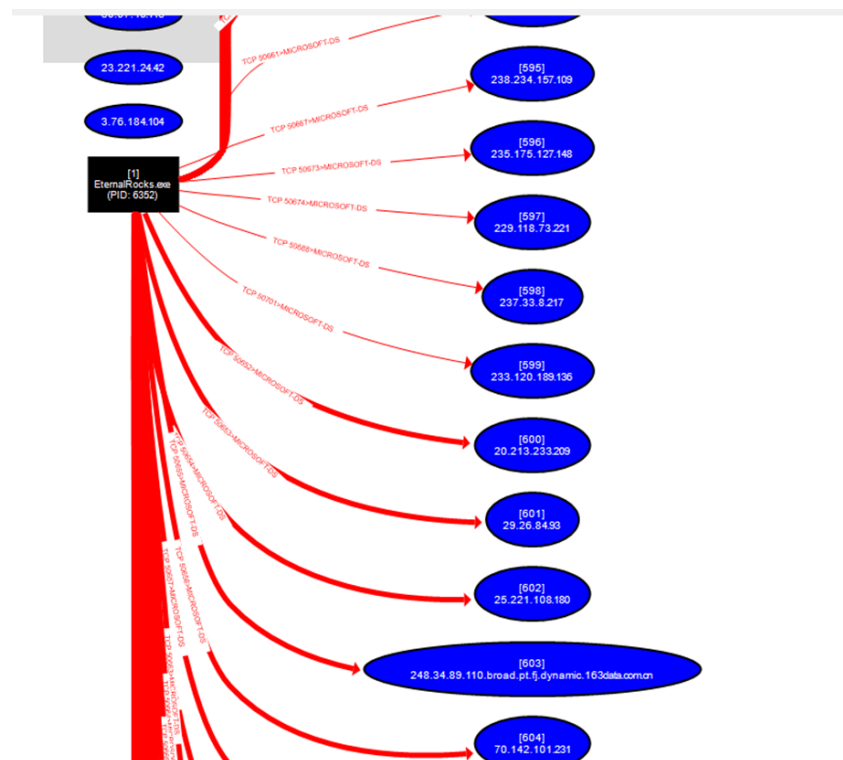


Process "EternalRocks.exe" (PID: 6352) sends TCP traffic to 237.248.171.136 port microsoft-ds

The second one is also **EternalRocks.exe** with the **PID: 7368**, This one creates log files and then it kills himself.



- Did the malware modify the file system or registry?
 - Well, it modifies the file system, one specific file that caught our eye was an XML that .NET commonly uses to keep track of the computers specification, this file is the *machine.config*, The worm EternalRockks modifies machine.config in order to keep useful data for the malware itself. Then it also modifies multiples registries such as **CLRLogDir**, **InstallRoot**, **SafeBoot**, **SessionManager** and **HKLM**
- Was any network communication initiated by the malware?
 - It initializes TCP connections to different IPs. In the picture below can be seen better.



1.4 Exercise 2: Correlating System and Network Activity

1.4.1 Objective

Use ProcDOT to correlate system activity and network behavior for a detailed analysis.

1.4.2 Steps

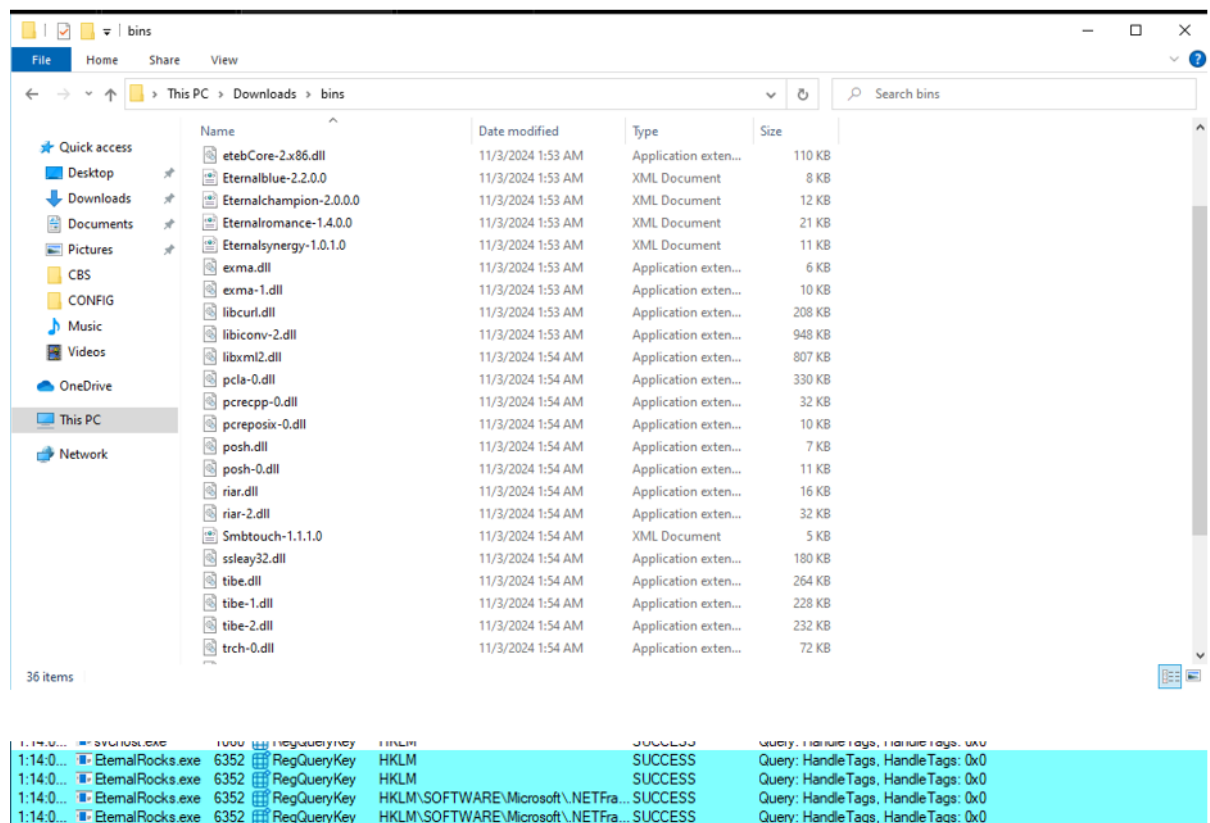
1. Capture system activity using ProcMon while running the malware sample.
2. Capture network traffic with Wireshark and save the PCAP file.

3. Load the Process Monitor Log (PML) and PCAP files into ProcDOT and generate the graph.

1.4.3 Questions

- How does the malware use the network? Is there any exfiltration or C2 communication?
 - We can see that there is C2 communication done by the malware, we can also find the persistence that the malware creates on the prefetch folder for windows, which is used to improve the performance of it by having pre-saved data of multiple applications.

EternalRocks takes advantages of this by creating its own pf file, so every time that the computer powers on, the malware will have a way into it. After this, it will also use the network to keep infecting the computer with even more malwares, that is what the process is trying to do by creating that many TCP connection.



- Are there any registry modifications that coincide with network activity?
 - Yes, mainly the modifications that were made to the registry is in HKLM where the network configuration details, like TCP/IP configuration, are located. We can see that the EternalRocks Process accesses and modifies this registry multiple times.

1:13:5...	EternalRocks.exe	6352	ReadFile	C:\Windows\Microsoft.NET\Framework...	SUCCESS	Offset: 2,565,120, ...	6836
1:13:5...	EternalRocks.exe	6352	RegQueryKey	HKLM	SUCCESS	Query: Handle Tag...	6836
1:13:5...	EternalRocks.exe	6352	RegOpenKey	HKLM\Software\Microsoft\NETFrame...	SUCCESS	Desired Access: R...	6836
1:13:5...	EternalRocks.exe	6352	CreateFile	C:\Windows\Microsoft.NET\Framework...	SUCCESS	Desired Access: R...	6836
1:13:5...	EternalRocks.exe	6352	QueryNetwork...	C:\Windows\Microsoft.NET\Framework...	SUCCESS	Creation Time: 12/7...	6836
1:13:5...	EternalRocks.exe	6352	CloseFile	C:\Windows\Microsoft.NET\Framework...	SUCCESS		6836
1:13:5...	EternalRocks.exe	6352	RegQueryKey	HKLM\SOFTWARE\Microsoft\NETFra...	SUCCESS	Query: Handle Tag...	6836
1:13:5...	EternalRocks.exe	6352	RegOpenKey	HKLM\SOFTWARE\Microsoft\NETFra...	SUCCESS	Desired Access: R...	6836
1:13:5...	EternalRocks.exe	6352	ReadFile	C:\Windows\System32\config\SOFTW...	SUCCESS	Offset: 28,774,400,...	6836
1:13:5...	EternalRocks.exe	6352	RegCloseKey	HKLM\SOFTWARE\Microsoft\NETFra...	SUCCESS		6836
1:13:5...	EternalRocks.exe	6352	RegQueryKey	HKLM\SOFTWARE\Microsoft\NETFra...	SUCCESS	Query: Handle Tag...	6836
1:13:5...	EternalRocks.exe	6352	RegOpenKey	HKLM\SOFTWARE\Microsoft\NETFra...	NAME NOT FOUND	Desired Access: R...	6836
1:13:5...	EternalRocks.exe	6352	RegCloseKey	HKLM\SOFTWARE\Microsoft\NETFra...	SUCCESS		6836
1:13:5...	EternalRocks.exe	6352	ReadFile	C:\Windows\Microsoft.NET\Framework...	SUCCESS	Offset: 9,955,840, ...	6836
1:13:5...	EternalRocks.exe	6352	ReadFile	C:\Windows\Microsoft.NET\Framework...	SUCCESS	Offset: 2,413,568, ...	6836
1:13:5...	EternalRocks.exe	6352	ReadFile	C:\Windows\Microsoft.NET\Framework...	SUCCESS	Offset: 9,434,112, ...	6836
1:13:5...	EternalRocks.exe	6352	ReadFile	C:\Windows\Microsoft.NET\Framework...	SUCCESS	Offset: 2,413,568, ...	6836
1:13:5...	svchost.exe	1060	ReadFile	C:\Windows\System32\wbem\Heposto...	SUCCESS	Offset: 23,044,036,...	5884
1:13:5...	EternalRocks.exe	6352	CreateFile	C:\Windows\Microsoft.NET\Framework...	SUCCESS	Desired Access: G...	6836
1:13:5...	TrustedInstaller...	4340	RegOpenKey	HKLM\SOFTWARE\Policies\Microsoft\...	SUCCESS	Desired Access: R...	1308
1:13:5...	TrustedInstaller...	4340	RegQueryValue	HKLM\SOFTWARE\Policies\Microsoft\...	NAME NOT FOUND	Length: 24	1308
1:13:5...	TrustedInstaller...	4340	RegCloseKey	HKLM\SOFTWARE\Policies\Microsoft\...	SUCCESS		1308
1:13:5...	TrustedInstaller...	4340	RegOpenKey	HKLM\SOFTWARE\Microsoft\Window...	SUCCESS	Desired Access: R...	1308
1:13:5...	TrustedInstaller...	4340	RegQueryValue	HKLM\SOFTWARE\Microsoft\Window...	NAME NOT FOUND	Length: 24	1308
1:13:5...	TrustedInstaller...	4340	RegCloseKey	HKLM\SOFTWARE\Microsoft\Window...	SUCCESS		1308
1:13:5...	EternalRocks.exe	6352	ReadFile	C:\Windows\Microsoft.NET\Framework...	SUCCESS	Offset: 2,954,240, ...	6836
1:13:5...	EternalRocks.exe	6352	ReadFile	C:\Windows\Microsoft.NET\Framework...	SUCCESS	Offset: 2,937,856, ...	6836
1:13:5...	EternalRocks.exe	6352	ReadFile	C:\Windows\Microsoft.NET\Framework...	SUCCESS	Offset: 2,597,888, ...	6836
1:13:5...	EternalRocks.exe	6352	RegQueryKey	C:\Windows\Microsoft.NET\Framework...	SUCCESS		6836
1:13:5...	EternalRocks.exe	6352	ReadFile	C:\Windows\Microsoft.NET\Framework...	SUCCESS	Offset: 4,095, Leng...	6836
1:13:5...	EternalRocks.exe	6352	ReadFile	C:\Windows\Microsoft.NET\Framework...	SUCCESS	Offset: 2,307,072, ...	6836

- Can you spot patterns in process creation and network usage?

– The main pattern that was noticed was that every time that a TCP connection was terminated, the thread was destroyed and then it was created a new one to initialize the TCP connection to a new IP. This process was repeated over and over again through the logs of the analysis.