Security Advisory 2019-021

Detecting and Preventing Emotet 2019 Campaign

September 30, 2019 — v1.0

TLP:WHITE

History:

- *30/09/2019 — v1.0 – Initial publication*

Summary

Since beginning of June 2019, the Emotet botnet stopped sending phishing emails to infect new victims. However, on August 22nd, 2019, the known Command-and-Control (CnC) servers started responding again [1]. Since September 16th, 2019, CERT-EU has been observing new phishing campaigns. To detect and prevent infection, CERT-EU analysed the behavior of those new versions of Emotet and hereby provides some recommendations for the SOC teams.

Technical Details

CERT-EU has observed several phishing campaigns delivering Emotet malware. All of them consisted of malicious MS Office document spawning **PowerShell** commands. Based on open-source information, there exists also a **Wscript** version [2], however, as we did not observe this strain, this advisory will focus on the **Powershell** version.

In the observed cases, the malicious MS Office document contains a macro. If the user clicks on **Enable Editing** and **Enable Content**, the macro will then spawn a Powershell command via a WMI Call. The Powershell command is base64-encoded and loops through different URLs (usually 5) to download and execute the Emotet malware.

Once installed, the Emotet malware will communicate via POST requests over HTTP with CnC servers [4].

Recommendations

Detection

There are several ways to detect Emotet infections. The most appropriate one depends on the visibility of the infrastructure and the stage of the infection. We present here three successful approaches to detect suspicious events generated by the 2019 Emotet campaign.

Detecting Initial Infection with Endpoint Monitoring

If the SOC has visibility on endpoint activity – especially process creation – it is possible to detect when a user has been tricked into opening a malicious document and activating content.

The initial infection happens when the user executes a macro embedded in a MS Office document (email attachment or downloaded via a link). The macro will then spawn a Powershell obfuscated command via WMI call. To detect suspicious event, SOC teams can trigger alerts when `powershell.exe` process is spawned by a `wmiprvse.exe` process as follows (Splunk syntax):

```
ParentImage="*\\wmiprvse.exe" AND Image="*\\powershell.exe"
```

However, depending on the environment, such query may generate false positives. If the SOC has visibility on the executed command line, it is possible to be more specific as both processes have specific options in the command line:

- `wmiprvse.exe` is called with the following options: `-secured -Embedding`
- `powershell.exe` is using base64-encoded command, which can be called with any reduction of the `encoded` option (`-e` , `-en` , `-enc` ...)

The following query should reduce the amount of false positive (Splunk syntax):

```
ParentImage="*\\wmiprvse.exe" AND Image="*\\powershell.exe" AND
  ParentCommandLine="*-secured*" AND CommandLine="*-e*"
```

Detecting Download of 2nd Stage

If the Powershell command is successfully executed, the malicious base64-encoded command will try to download the 2nd stage malware from different locations (usually 5). The script will stop when a download is successful.

It is possible to extract the URLs used to download the malicious binary from the base64-encoded string using a simple CyberChef [3] recipe as follows:

```
From_Base64('A-Za-z0-9+/=',true)
Remove_null_bytes()
Extract_URLs(false)
Find_/_Replace({'option':'Simple string','string':'\''},'',true,false,true,false)
Split('@','\\n')
```

By looking at DNS or Proxy logs, it is possible to roughly identify a successful infection:

- If all the extracted URLs/domains are contacted, the download was probably not successful (blacklist/website down/etc.).
- If only one or two of the URLs/domains were contacted and the command stopped before looping through all the possibilities, than the second stage was most probably downloaded.

Detecting CnC Communication with Proxy Logs

The 2nd stage malware use a very specific pattern for CnC communications [4]:

- POST request over HTTP,
- to an IP addresses (no DNS resolution),
- using a limited list of keyword in the URL path.

It is possible to detect such pattern with the following search in proxy logs (Splunk syntax):

```
http_method=POST AND ("/teapot/" OR "/pnp/" OR "/tpt/" OR "/splash/" OR "/site/" OR
  "/codec/" OR "/health/" OR "/balloon/" OR "/cab/" OR "/odbc/" OR "/badge/" OR "/dma/" OR
  "/psec/" OR "/cookies/" OR "/iplk/" OR "/devices/" OR "/enable/" OR "/mult/" OR "/prov/"
  OR "/vermont/" OR "/attrib/" OR "/schema/" OR "/iab/" OR "/chunk/" OR "/publish/" OR
  "/prep/" OR "/srvc/" OR "/sess/" OR "/ringin/" OR "/nsip/" OR "/stubs/" OR "/img/" OR
  "/add/" OR "/xian/" OR "/jit/" OR "/free/" OR "/pdf/" OR "/loadan/" OR "/arizona/" OR
  "/tlb/" OR "/forced/" OR "/results/" OR "/symbols/" OR "/rep|t/" OR "/guids/" OR
  "/taskbar/" OR "/child/" OR "/cone/" OR "/glitch/" OR "/entries/" OR "/between/" OR
  "/bml/" OR "/usbccid/" OR "/sym/" OR "/enabled/" OR "/merge/" OR "/window/" OR
  "/scripts/" OR "/raster/" OR "/acquire/" OR "/json/" OR "/rtm/" OR "/walk/" OR "/ban/") |
   regex "http://\d{1,3}.\d{1,3}.\d{1,3}.\d{1,3}"
```

Prevention

As explained before, Emotet infection relies on usual tricks to infect target like embedded macros and Powershell encoded commands. There are several strategies to prevent infections using these techniques, such as banning execution of MS Office macros or disabling Powershell script execution. Those strategies are not being always applicable because of IT of business needs.

However, it is possible to block the download of the second stage by forbidding Powershell process to download anything from Internet via local firewall rules [5].

There are two approaches possible to perform that:

- block all network connections for Powershell, or
- block communication with the proxy server for Powershell.

The firewall rules need to be applied for **all versions of PowerShell** installed on workstations.

More information on securing Powershell may also be found in [6] and [7].

References

[1] https://twitter.com/MalwareTechBlog/status/1164616966499254272

[2] https://www.bleepingcomputer.com/news/security/emotet-trojan-evolves-since-being-reawakend-here-is-what-we-know/

[3] https://gchq.github.io/CyberChef/

[4] https://blog.trendmicro.com/trendlabs-security-intelligence/emotet-adds-new-evasion-technique-and-uses-connected-devices-as-proxy-cc-servers/

[5] https://isc.sans.edu/forums/diary/Blocking+Powershell+Connection+via+Windows+Firewall/21829/

[6] https://www.cyber.gov.au/publications/securing-powershell-in-the-enterprise

[7] https://media.cert.europa.eu/static/WhitePapers/CERT-EU-SWP2019-001.pdf