Guess who’s back

Emotet

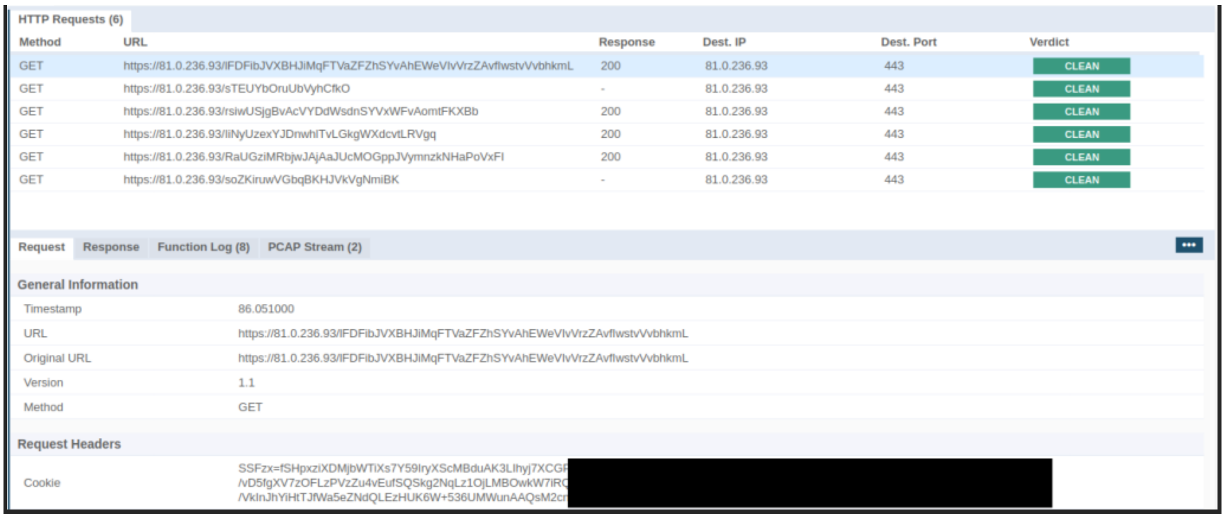
The (slighty) longer story:
On Sunday, November 14, at around 9:26pm UTC we observed on several of our Trickbot trackers that the bot tried to download a DLL to the system. According to internal processing, these DLLs have been identified as Emotet. However, since the botnet was taken down earlier this year, we were suspicious about the findings and conducted an initial manual verification. Please find first results and IOCs below. Currently, we have high confidence that the samples indeed seem to be a re-incarnation of the infamous Emotet.

We are still conducting more in-depth analyses to raise the confidence even further. New information will be provided as they become available.

Initial Analysis

Sunday, November 14, 9:26pm: first occurence of the URLs being dropped; the URL we received was `hxxp://141.94.176.124/Loader_90563_1.dll` (SHA256 of the drop: `c7574aac7583a5bdc446f813b8e347a768a9f4af858404371eae82ad2d136a01`). Internal processing detected Emotet when executing the sample in our sandbox systems. Notably, the sample seems to have been compiled just before the deployment via several Trickbot botnets was observed: `Timestamp : 6191769A (Sun Nov 14 20:50:34 2021)`

The network traffic originating from the sample closely resembles what has been observed previously (e.g. as described by Kaspersky): the URL contains a random resource path and the bot transfers the request payload in a cookie (see image below). However, the encryption used to hide the data seems different from what has been observed in the past. Additionally, the sample now uses HTTPS with a self-signed server certificate to secure the network traffic.



HTTP Requests (6)

Method	URL	Response	Dest. IP	Dest. Port	Verdict
GET	https://81.0.236.93/lFDFibJVXBHJiMqFTVaZFZhSYvAhEWeVlvVrzZAvflwstvVvbhkmL	200	81.0.236.93	443	CLEAN
GET	https://81.0.236.93/sTEUYbOruUbVyhCfkO	-	81.0.236.93	443	CLEAN
GET	https://81.0.236.93/rsiwUSjgBvAcVYDdWsdnSYVxWFvAomtFKXBb	200	81.0.236.93	443	CLEAN
GET	https://81.0.236.93/liNyUzexYJDnwhlTvLGkgWXdcvtLRVgq	200	81.0.236.93	443	CLEAN
GET	https://81.0.236.93/RaUGziMRbjwJAjAaJUcMOGppJVymnzkNHaPoVxFI	200	81.0.236.93	443	CLEAN
GET	https://81.0.236.93/soZKiruwVGbqBKHJVkVgNmiBK	-	81.0.236.93	443	CLEAN

Request | Response | Function Log (8) | PCAP Stream (2)

General Information

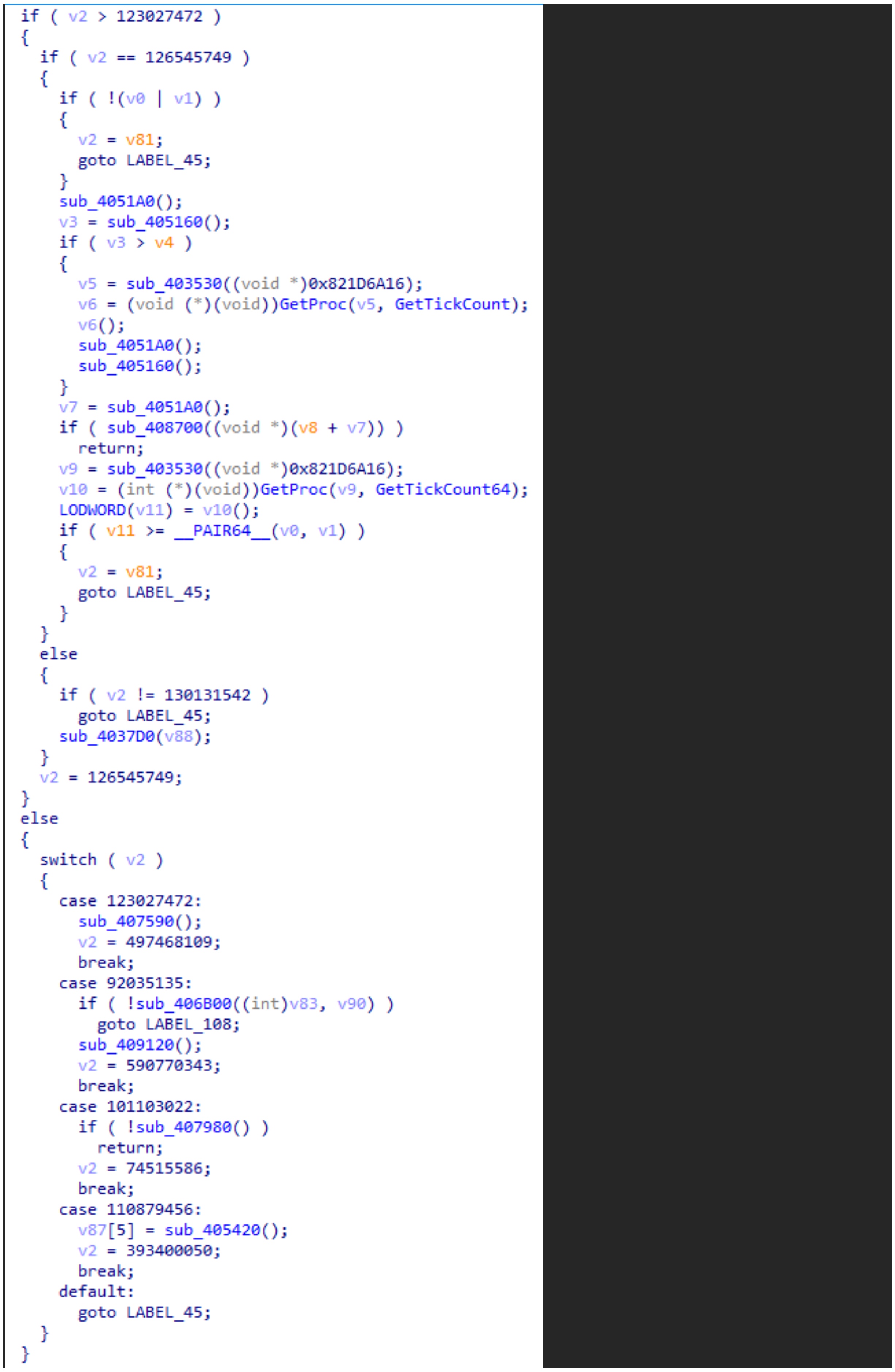
Timestamp	86.051000
URL	https://81.0.236.93/lFDFibJVXBHJiMqFTVaZFZhSYvAhEWeVlvVrzZAvflwstvVvbhkmL
Original URL	https://81.0.236.93/lFDFibJVXBHJiMqFTVaZFZhSYvAhEWeVlvVrzZAvflwstvVvbhkmL
Version	1.1
Method	GET

Request Headers

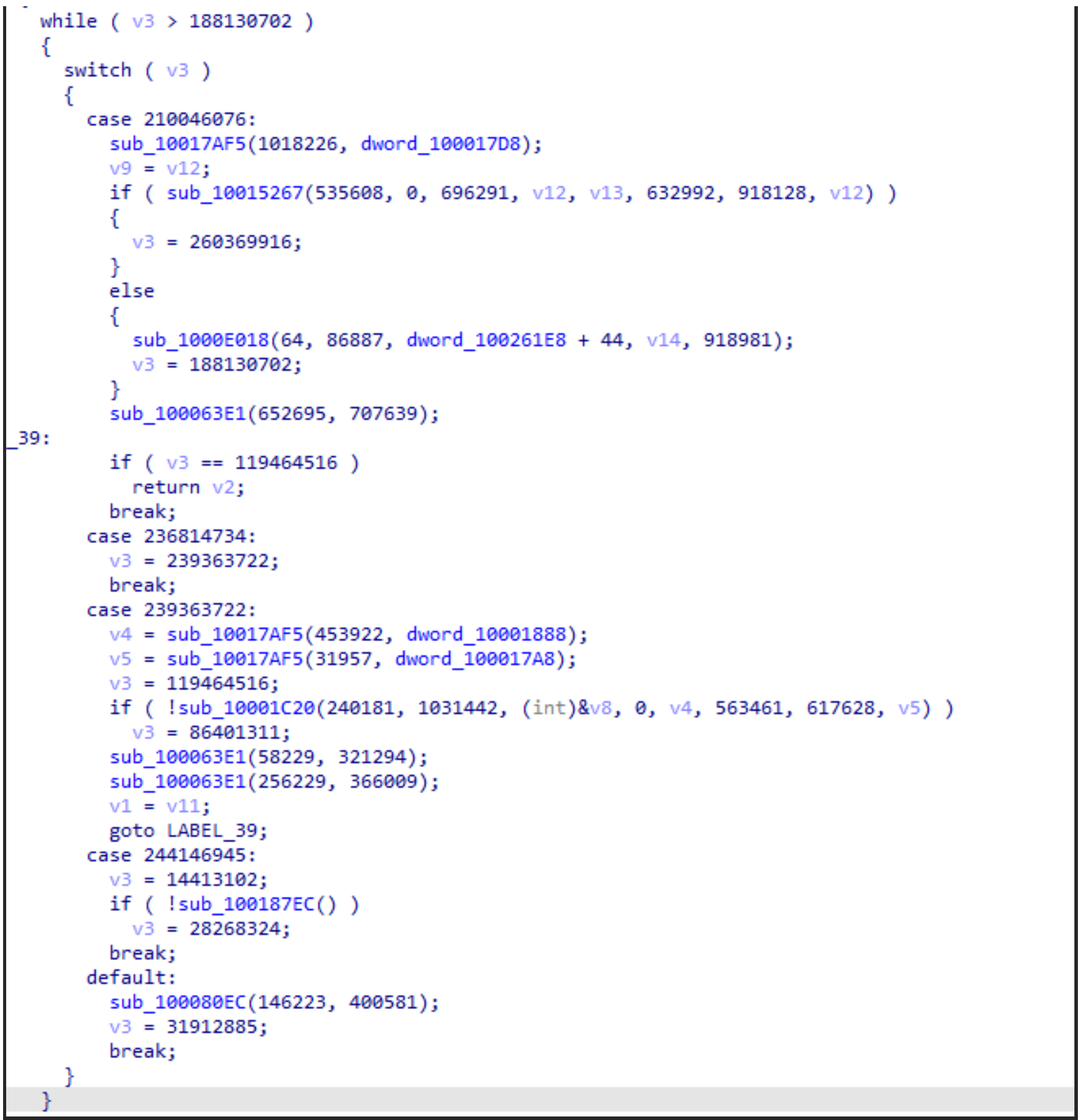
Cookie	SSFzx=fSHpxziXDMjbWTiXs7Y59IryXScMBduAK3LIhyj7XCGP /vD5fgXV7zOFLzPVzZu4vEufSQSkg2NqLz1OjLMBOwkW7iRC /VkInJhYiHtTJfWa5eZNdQLEzHUK6W+536UMWunAAQsM2cr [redacted]

Network Traffic originating from the DLL

A notable characteristic of the last Emotet samples was the heavy use of control-flow flattening to obfuscate the code. The current sample also contains flattened control flows. To illustrate the similarity in the style of the obfuscation, find two arbitrary code snippets below. Left side is a sample from 2020, on the right is a snippet from the current sample:



```
if ( v2 > 123027472 )
{
  if ( v2 == 126545749 )
  {
    if ( !(v0 | v1) )
    {
      v2 = v81;
      goto LABEL_45;
    }
    sub_4051A0();
    v3 = sub_405160();
    if ( v3 > v4 )
    {
      v5 = sub_403530((void *)0x821D6A16);
      v6 = (void (*)(void))GetProc(v5, GetTickCount);
      v6();
      sub_4051A0();
      sub_405160();
    }
    v7 = sub_4051A0();
    if ( sub_408700((void *)(v8 + v7)) )
      return;
    v9 = sub_403530((void *)0x821D6A16);
    v10 = (int (*)(void))GetProc(v9, GetTickCount64);
    LODWORD(v11) = v10();
    if ( v11 >= __PAIR64__(v0, v1) )
    {
      v2 = v81;
      goto LABEL_45;
    }
  }
  else
  {
    if ( v2 != 130131542 )
      goto LABEL_45;
    sub_4037D0(v88);
  }
  v2 = 126545749;
}
else
{
  switch ( v2 )
  {
    case 123027472:
      sub_407590();
      v2 = 497468109;
      break;
    case 92035135:
      if ( !sub_406B00((int)v83, v90) )
        goto LABEL_108;
      sub_409120();
      v2 = 590770343;
      break;
    case 101103022:
      if ( !sub_407980() )
        return;
      v2 = 74515586;
      break;
    case 110879456:
      v87[5] = sub_405420();
      v2 = 393400050;
      break;
    default:
      goto LABEL_45;
  }
}
```



```
  while ( v3 > 188130702 )
  {
    switch ( v3 )
    {
      case 210046076:
        sub_10017AF5(1018226, dword_100017D8);
        v9 = v12;
        if ( sub_10015267(535608, 0, 696291, v12, v13, 632992, 918128, v12) )
        {
          v3 = 260369916;
        }
        else
        {
          sub_1000E018(64, 86887, dword_100261E8 + 44, v14, 918981);
          v3 = 188130702;
        }
        sub_100063E1(652695, 707639);
_39:
        if ( v3 == 119464516 )
          return v2;
        break;
      case 236814734:
        v3 = 239363722;
        break;
      case 239363722:
        v4 = sub_10017AF5(453922, dword_10001888);
        v5 = sub_10017AF5(31957, dword_100017A8);
        v3 = 119464516;
        if ( !sub_10001C20(240181, 1031442, (int)&v8, 0, v4, 563461, 617628, v5) )
          v3 = 86401311;
        sub_100063E1(58229, 321294);
        sub_100063E1(256229, 366009);
        v1 = v11;
        goto LABEL_39;
      case 244146945:
        v3 = 14413102;
        if ( !sub_100187EC() )
          v3 = 28268324;
        break;
      default:
        sub_100080EC(146223, 400581);
        v3 = 31912885;
        break;
    }
  }
```

Conclusion (so far)

As per the famous duck-typing, we conclude so far: smells like Emotet, looks like Emotet, behaves like Emotet – seems to be Emotet.

We are currently updating our internal tooling for the new sample to provide more indicators to strengthen the claim that Emotet seems to be back.

IOCs

`URLs:`

```
hxxp://141.94.176.124/Loader_90563_1.dll



Hashes:

c7574aac7583a5bdc446f813b8e347a768a9f4af858404371eae82ad2d136a01 - Loader_90563_1.dll



Server List:

81.0.236.93:443

94.177.248.64:443

66.42.55.5:7080

103.8.26.103:8080

185.184.25.237:8080

45.76.176.10:8080

188.93.125.116:8080

103.8.26.102:8080

178.79.147.66:8080

58.227.42.236:80

45.118.135.203:7080

103.75.201.2:443

195.154.133.20:443

45.142.114.231:8080

212.237.5.209:443

207.38.84.195:8080
```

```
104.251.214.46:8080

138.185.72.26:8080

51.68.175.8:8080

210.57.217.132:8080



String List:

SOFTWARE\Microsoft\Windows\CurrentVersion\Run

POST

%s\rundll32.exe "%s",Control_RunDLL

Control_RunDLL

%s\%s

%s\%s

%s\%s%x

%s%s.exe

%s\%s

SHA256

HASH

AES

Microsoft Primitive Provider

ObjectLength

KeyDataBlob

%s\rundll32.exe "%s\%s",%s
```

```
Content-Type: multipart/form-data; boundary=%s



RNG

%s%s.dll

%s\rundll32.exe "%s",Control_RunDLL

%s%s.dll

%s\regsvr32.exe -s "%s"

%s\%s

%s%s.exe

SOFTWARE\Microsoft\Windows\CurrentVersion\Run

%s\rundll32.exe "%s\%s",%s

ECCPUBLICBLOB

ECDH_P256

Microsoft Primitive Provider

ECCPUBLICBLOB

Cookie: %s=%s



%s\rundll32.exe "%s\%s",%s

%s:Zone.Identifier

%u.%u.%u.%u

%s\%s

%s\*
```

```
%s\%s

WinSta0\Default

%s\rundll32.exe "%s",Control_RunDLL %s

%s%s.dll

ECCPUBLICBLOB

ECDSA_P256

Microsoft Primitive Provider

%s\%s

SHA256

Microsoft Primitive Provider

ObjectLength
```