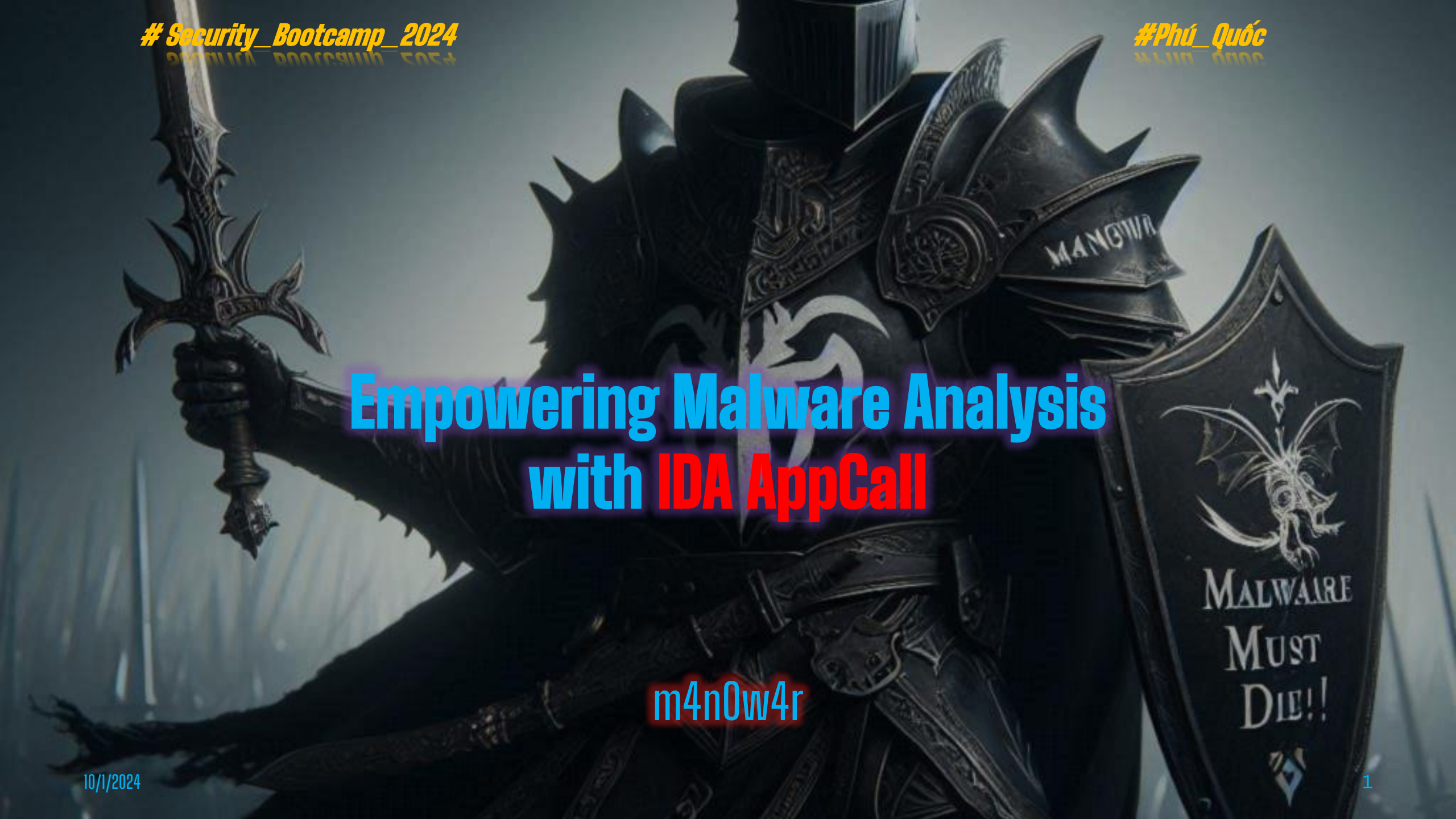


Empowering Malware Analysis with IDA AppCall

m4n0w4r

#Wh0_4m_I?

Virus Bulletin reposted

m4n0w4r @kienbigmummy · Jun 6

✍️My quick note about #DarkGate

🟥How to extract AutoIt script and deobf to recover original code.

🟥How to extract shellcode loader.

🟥How to decrypt and extract the final #DarkGate payload.

From kienmanowar.wordpress.com

3 31 100 7.9K

What we will cover

1. Summary of **AppCall** in IDA
2. Leveraging IDA AppCall in Malware Analysis
    1. **LokiBot**:
        - Recover API name by hash
    2. **Emotet**:
        - Decrypt strings
        - Extract C2s configuration

Summary of Appcall in IDA (1)

- A feature in **IDA Pro** that allow users to **call functions within the context of an active debugging session**. It facilitates **dynamic analysis** by enabling function calls as if they were made by the debugged application itself.
- **Identify the function** want to call: find the function and ensure it is properly defined with its prototype (**argument types and return type**).
- Make sure in the **proper context**: set a breakpoint in the appropriate location (e.g., main function) before calling the function.
- Issue the Appcall: **call function** with a specific prototype.

Summary of Appcall in IDA (2)

- Prototypes are crucial for Appcalls to work correctly: define the function's arguments and return type.
- Can use Appcall to call functions that are not originally exposed in IDA database, such as those loaded dynamically by the debugged process.
- Appcall can also be used to interact with complex data structures by using IDA's generic object handling capabilities.
- Utilize IDC/IDAPython for scripting complex operations.

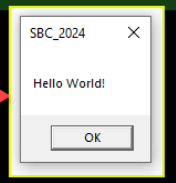
A simple example (1)

main.cpp

SBC_msgbox (Global Scope)

```
1  #include <Windows.h>
2
3  int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, LPSTR lpCmdLine, int nCmdShow) {
4      MessageBoxA(NULL, "Hello World!", "SBC_2024", MB_OK);
5      return 0;
6  }
```

```
WinMain(x,x,x,x)
WinMain(x,x,x,x)
WinMain(x,x,x,x)      ; Attributes: bp-based frame
WinMain(x,x,x,x)
WinMain(x,x,x,x)      ; int __stdcall WinMain(HINSTANCE__ *hInstance, HINSTANCE__ *hPrevInstance, char *lpCmdLine, int nCmdShow)
WinMain(x,x,x,x)      __stdcall WinMain(x, x, x, x) proc near
WinMain(x,x,x,x)                          ; CODE XREF: _WinMain@16_0↑j
WinMain(x,x,x,x)
WinMain(x,x,x,x)      var_C0= byte ptr -0C0h
WinMain(x,x,x,x)      hInstance= dword ptr  8
WinMain(x,x,x,x)      hPrevInstance= dword ptr  0Ch
WinMain(x,x,x,x)      lpCmdLine= dword ptr  10h
WinMain(x,x,x,x)      nCmdShow= dword ptr  14h
WinMain(x,x,x,x)
WinMain(x,x,x,x)          push    ebp
WinMain(x,x,x,x)+1        mov     ebp, esp
WinMain(x,x,x,x)+3        sub     esp, 0C0h
WinMain(x,x,x,x)+9        push    ebx
WinMain(x,x,x,x)+A        push    esi
WinMain(x,x,x,x)+B        push    edi
WinMain(x,x,x,x)+C        lea     edi, [ebp+var_C0]
WinMain(x,x,x,x)+12       mov     ecx, 30h
WinMain(x,x,x,x)+17       mov     eax, 0CCCCCCCCh
WinMain(x,x,x,x)+1C       rep stosd
WinMain(x,x,x,x)+1E       mov     esi, esp
WinMain(x,x,x,x)+20       push    0           ; uType
WinMain(x,x,x,x)+22       push    offset Caption ; "SBC_2024"
WinMain(x,x,x,x)+27       push    offset Text ; "Hello World!"
WinMain(x,x,x,x)+2C       push    0           ; hWnd
WinMain(x,x,x,x)+2E       call    ds:MessageBoxA(x,x,x,x)
WinMain(x,x,x,x)+34       cmp     esi, esp
WinMain(x,x,x,x)+36       call    j___RTC_CheckEsp
WinMain(x,x,x,x)+3B       xor     eax, eax
WinMain(x,x,x,x)+3D       pop     edi
WinMain(x,x,x,x)+3E       pop     esi
WinMain(x,x,x,x)+3F       pop     ebx
WinMain(x,x,x,x)+40       add     esp, 0C0h
WinMain(x,x,x,x)+46       cmp     ebp, esp
WinMain(x,x,x,x)+48       call    j___RTC_CheckEsp
WinMain(x,x,x,x)+4D       mov     esp, ebp
WinMain(x,x,x,x)+4F       pop     ebp
WinMain(x,x,x,x)+50       retn    10h
WinMain(x,x,x,x)+50   __stdcall WinMain(x, x, x, x) endp
WinMain(x,x,x,x)+50
```



A simple example (2)

Please enter script body

```
// Step 1: Get the address of MessageBoxA.
auto messagebox_addr = get_name_ea_simple("user32_MessageBoxA");

// Check if the address was resolved correctly
if (messagebox_addr == BADADDR) {
    Warning("Could not find MessageBoxA function address!");
} else {

    // Step 2: Permanently set the prototype for the function
    SetType(messagebox_addr, "int __stdcall MessageBoxA(HWND hWnd, LPCSTR lpText, LPCSTR lpCaption, UINT uType);");

    // Step 3: Prepare custom parameters.
    auto hWnd = 0;                      // No owner window
    auto lpText = "Welcome to SBC 2024!";  // Custom text
    auto lpCaption = "SBC_PQ_2024";     // Custom caption
    auto uType = 1;                     // MB_OKCANCEL (1)

    // Step 4: Invoke MessageBoxA.
    auto ret_val = messagebox_addr(hWnd, lpText, lpCaption, uType);

    // Step 5: Print the return value.
    Message("MessageBoxA returned: %d\n", ret_val);
}
```

Line:22 Column:52

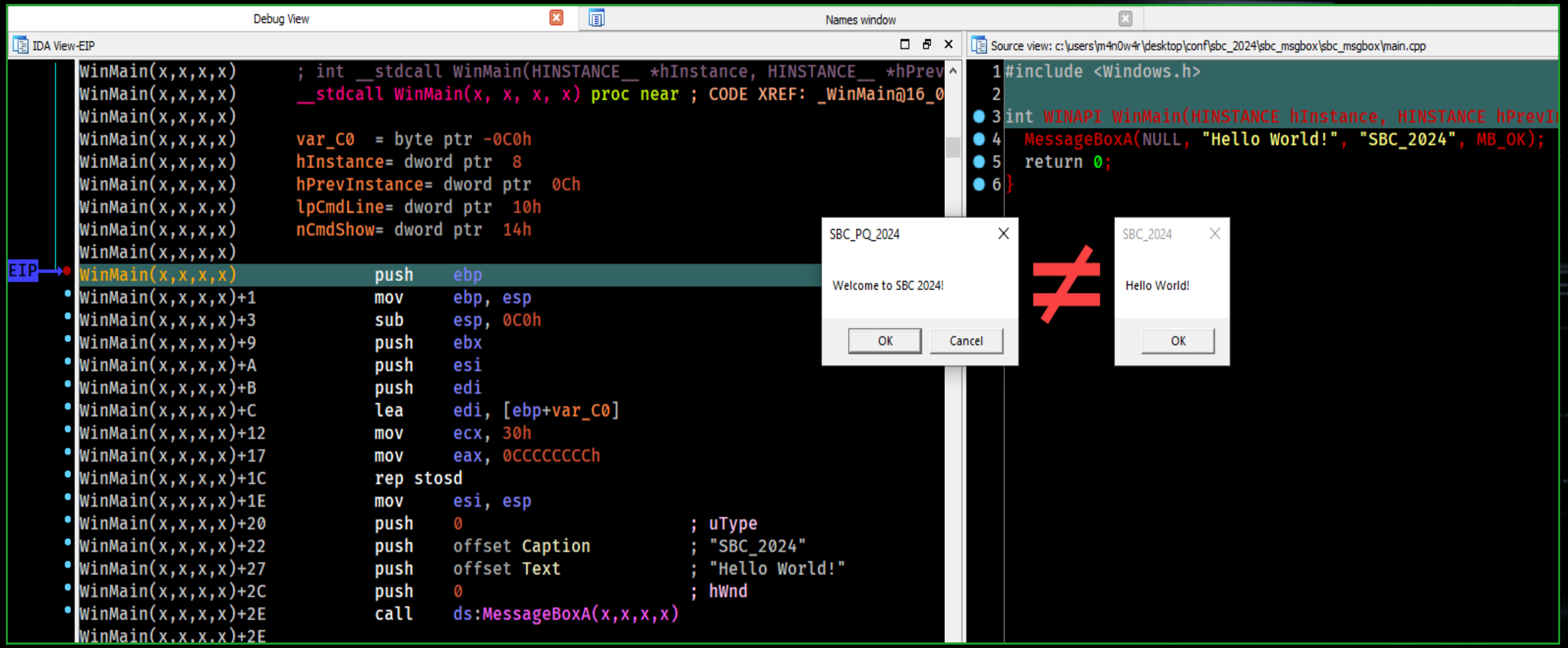
IDC Tab size 4

Syntax

C++

```
int MessageBoxA(
  [in, optional] HWND   hWnd,
  [in, optional] LPCSTR lpText,
  [in, optional] LPCSTR lpCaption,
  [in]           UINT   uType
);
```

A simple example (3)

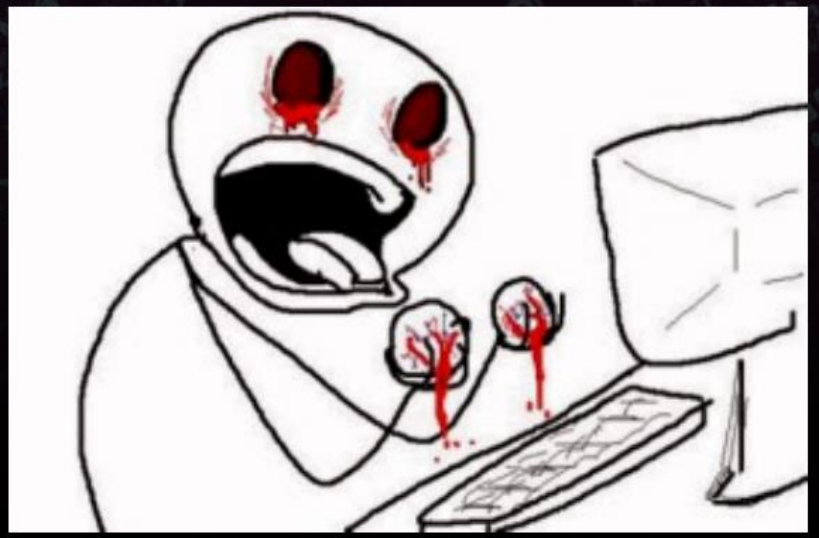


A Great Explanation Video

https://www.youtube.com/watch?v=U-oMpg1Mktg

Quote Of The Day

> Reverse engineering is like peeling an onion, each layer reveals a new surprise! But sometimes it can burn your eyes!



https://speakerdeck.com/fr0gger/binary-instrumentation-for-malware-analysis


LOKIBOT
MALWARE
MUST
DIE!

LokiBot (1)

LOKIBOT

lokibot loader trojan

8	21 ↓	15 ↑	2053
Global rank	Month rank	Week rank	IOCs

LokiBot was developed in 2015 to steal information from a variety of applications. Despite the age, this malware is still rather popular among cybercriminals.

Stealer
Type

ex-USSR territory
Origin

3 May, 2015
First seen

13 July, 2024
Last seen

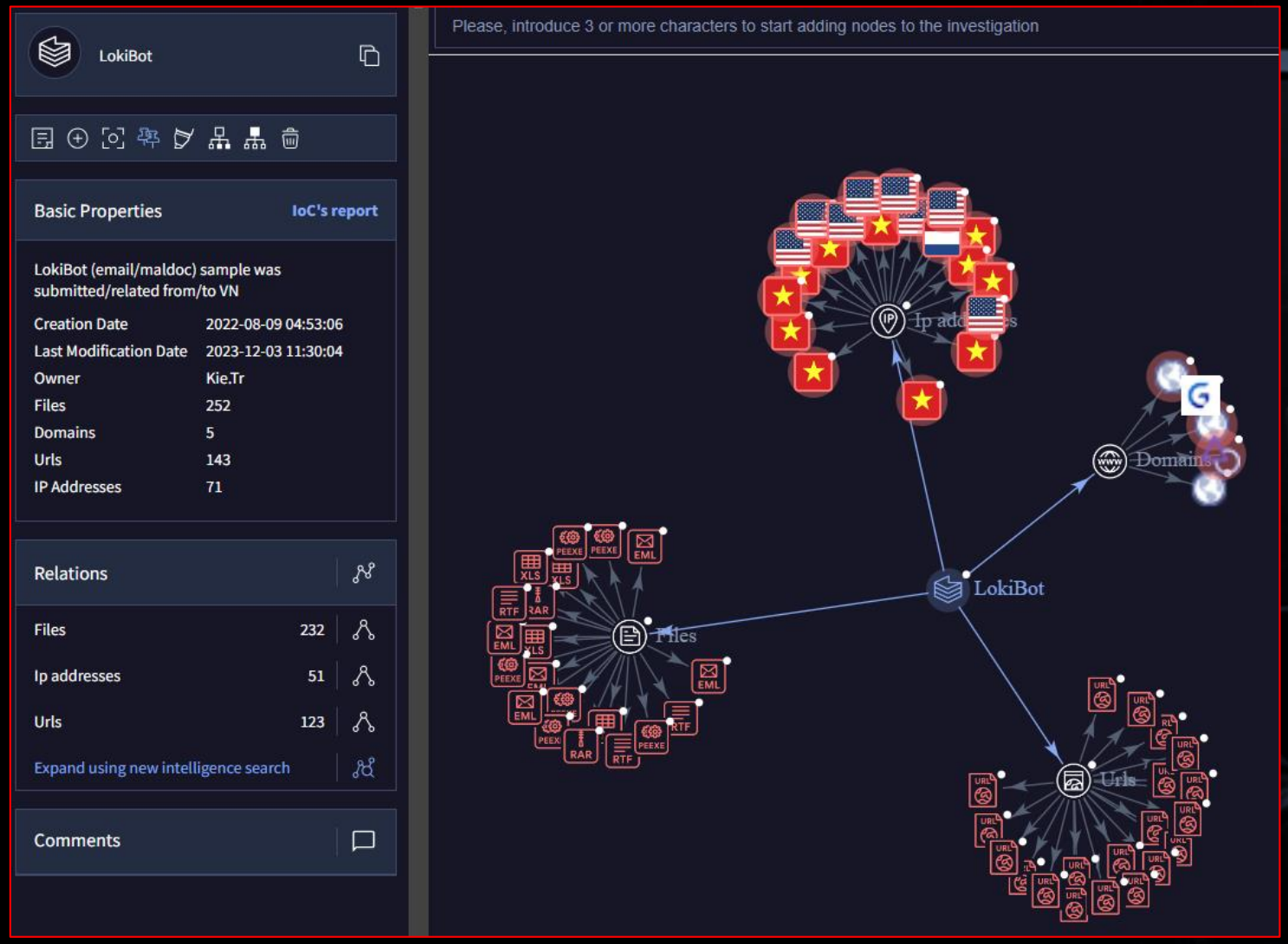
ALSO KNOWN AS

Loki
LokiPWS

IOCs

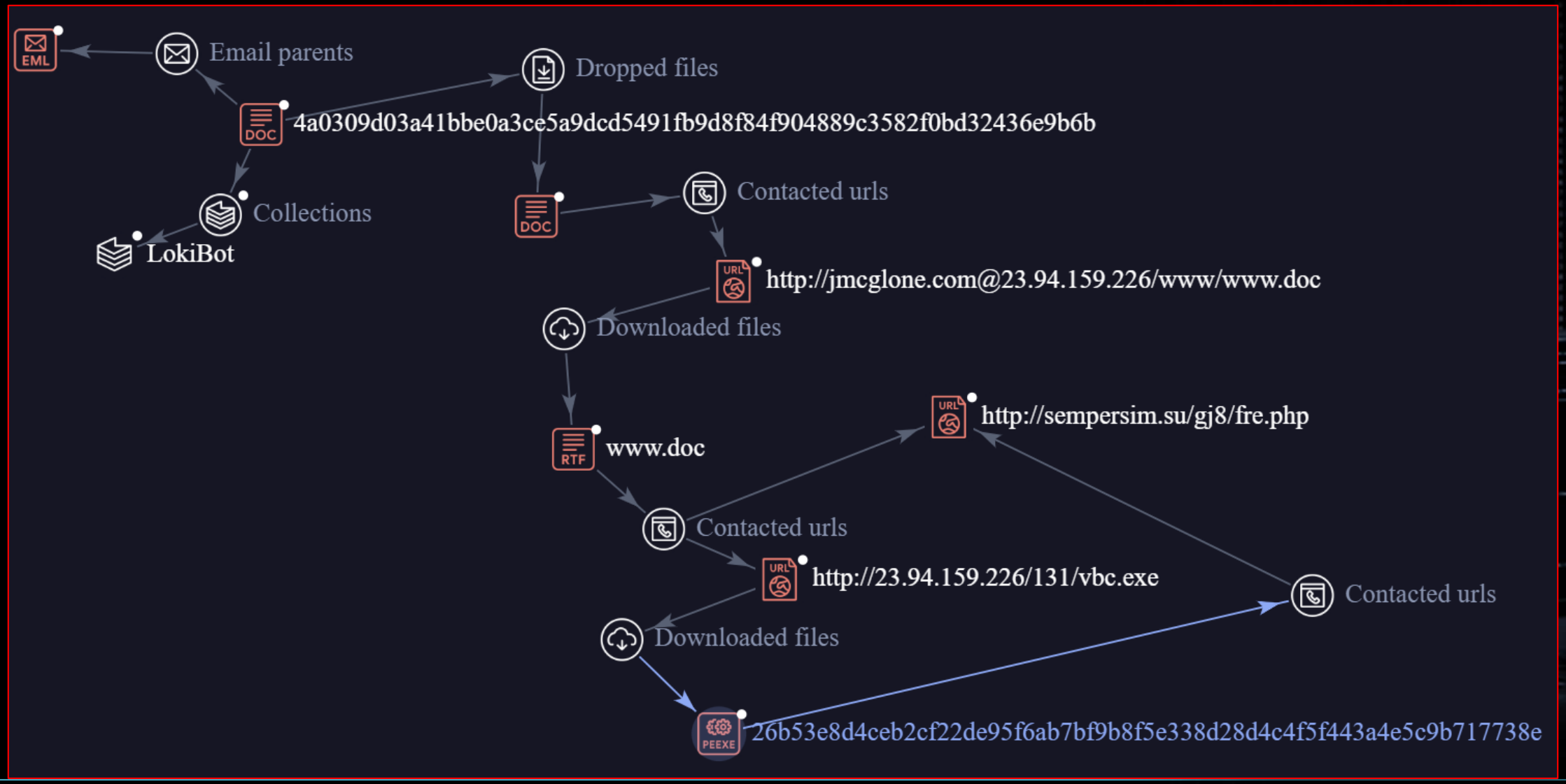
https://any.run/malware-trends/lokibot

LokiBot (2)


LokiBot
Please, introduce 3 or more characters to start adding nodes to the investigation
Basic Properties
IoC's report
LokiBot (email/maldoc) sample was submitted/related from/to VN
Creation Date 2022-08-09 04:53:06
Last Modification Date 2023-12-03 11:30:04
Owner Kie.Tr
Files 252
Domains 5
Urls 143
IP Addresses 71
Relations
Files 232
Ip addresses 51
Urls 123
Expand using new intelligence search
Comments
Ip addresses
Domains
Files
Urls
LokiBot

Reversing LokiBot

LokiBot Infection Chain



Dynamic Resolve API Functions

Address	Ordinal	Name	Library
WS2_32			
00415028		getaddrinfo	WS2_32
0041502C		freeaddrinfo	WS2_32
00415030	3	closesocket	WS2_32
00415044	4	connect	WS2_32
00415040	16	recv	WS2_32
0041503C	19	send	WS2_32
00415038	23	socket	WS2_32
00415034	115	WSAStartup	WS2_32
KERNEL32			
00415000		GetProcessHeap	KERNEL32
00415004		HeapFree	KERNEL32
00415008		HeapAlloc	KERNEL32
0041500C		SetLastError	KERNEL32
00415010		GetLastError	KERNEL32
ole32			
0041504C		CoCreateInstance	ole32
00415050		CoInitialize	ole32
00415054		CoUninitialize	ole32
OLEAUT32			
00415020	2	SysAllocString	OLEAUT32
0041501C	6	SysFreeString	OLEAUT32
00415018	8	VariantInit	OLEAUT32

```
ext:00402BF2 sub_402BF2 proc near                ; CODE XREF: sub_
ext:00402BF2         xor     eax, eax
ext:00402BF4         push    eax                 ; offset
ext:00402BF5         push    eax                 ; a3
ext:00402BF6         push    0DDE17395h          ; pre_api_hash
ext:00402BFB         push    eax                 ; dllIndex
ext:00402BFC         call    lkb_retrieve_api_addr
ext:00402BFC
ext:00402C01         jmp     eax
ext:00402C01
ext:00402C01 sub_402BF2 endp
```

```
1 int sub_402BF2()
2 {
3   // [COLLAPSED LOCAL DECLARATIONS. PRESS KEYPAD CTRL-"+" TO EXPAND]
4
5   api_addr = lkb_retrieve_api_addr(0, 0xDDE17395, 0, 0);
6   return api_addr();
7 }
```

Hardcore Reverser – try hard to understand logic

```
PVOID *__cdecl sub_4030A5(int dllIndex, int pre_api_hash)
{
  // [COLLAPSED LOCAL DECLARATIONS. PRESS KEYPAD CTRL-"+" TO EXPAND]

  dllBaseAddr = lkb_retrieve_dll_base(dllIndex);
  if ( dllBaseAddr )
  {
    return lkb_retrieve_api_addr_by_hash(dllBaseAddr, pre_api_hash);
  }
  return dllBaseAddr;
}
```

```
while ( TRUE )
{
  strAPIName = (arg_dllBaseAddr + *pFuncNamesTbl);
  apiNameLen = lkb_strlen(strAPIName);
  if ( lkb_calc_hash(strAPIName, apiNameLen) == pre_api_hash )
  {
    break;
  }
  ++pFuncNamesTbl;
  ++pNameOrdsTbl;
  ++pFuncNamesTbl;
  if ( ++count ≥ pExportDir->NumberOfNames )
  {
    return 0;
  }
}
```

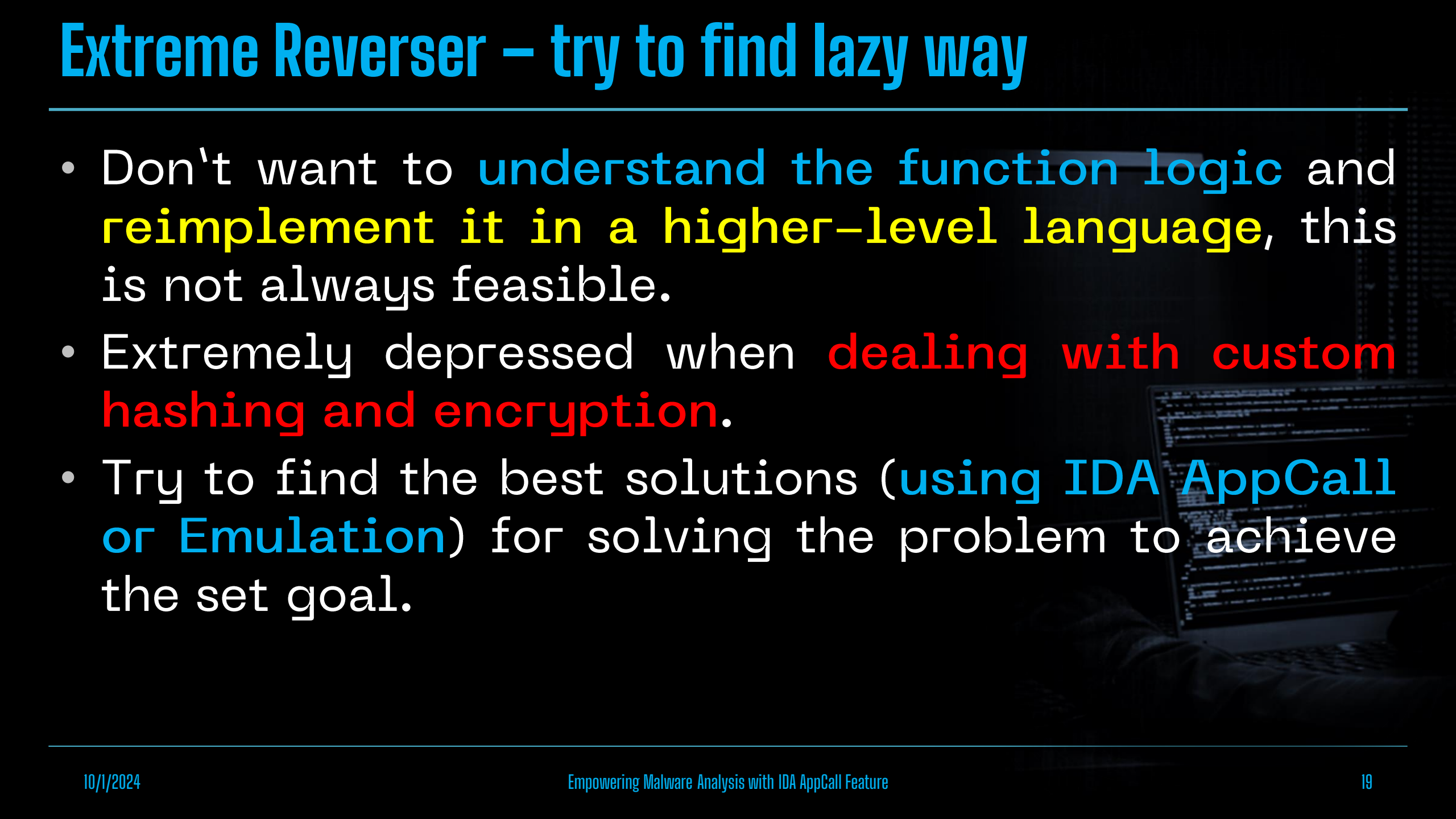
```
int __cdecl lkb_calc_hash(char *arg_strInput, size_t strLen)
{
  // [COLLAPSED LOCAL DECLARATIONS. PRESS KEYPAD CTRL-"+" TO EXPAND]

  hash = 0xFFFFFFFF;
  while ( strLen )
  {
    --strLen;
    hash ^= *arg_strInput++;
    count = 8;
    do
    {
      if ( (hash & 1) ≠ 0 )
      {
        hash ^= 0x4358AD54u;
      }
      hash >>= 1;
      --count;
    }
    while ( count );
  }
  return ~hash;
}
```

Hardcore Reverser – reimplement code

```python
def loki_calc_hash(func_name):
    """
        Calculate hash based on the function name
        """

    hash = 0xFFFFFFFF
    func_len = len(func_name)
    func_name = bytearray(func_name)
    i = 0
    while (func_len != 0):
        func_len -= 1
        hash = (hash ^ func_name[i])
        i += 1
        for j in range(8):
            if ((hash & 0x1) != 0):
                hash = (hash ^ 0x4358AD54)
            hash = (hash >> 1)

    return ~hash & 0xFFFFFFFF
```

Extreme Reverser – try to find lazy way

- Don't want to understand the function logic and reimplement it in a higher-level language, this is not always feasible.
- Extremely depressed when dealing with custom hashing and encryption.
- Try to find the best solutions (using IDA AppCall or Emulation) for solving the problem to achieve the set goal.

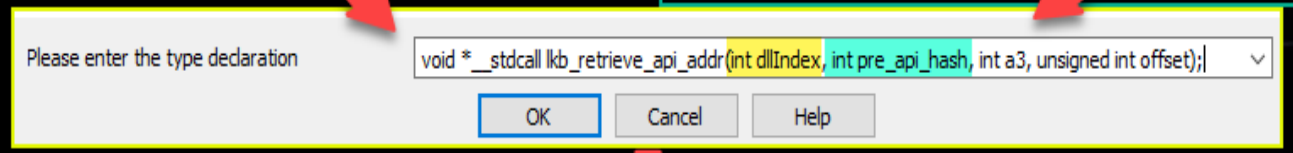


Recover API Name with IDA AppCall (1)

- **Initialization**: ensure function prototype are correctly defined in IDA Pro.

```
00413838        push    ebp
00413839        mov     ebp, esp
0041383B        push    0                       ; offset
0041383D        push    0                       ; a3
0041383F        push    0C5FA88F1h              ; pre_api_hash
00413844        push    0Ah                     ; dllIndex
00413846        call    lkb_retrieve_api_addr
00413846
```

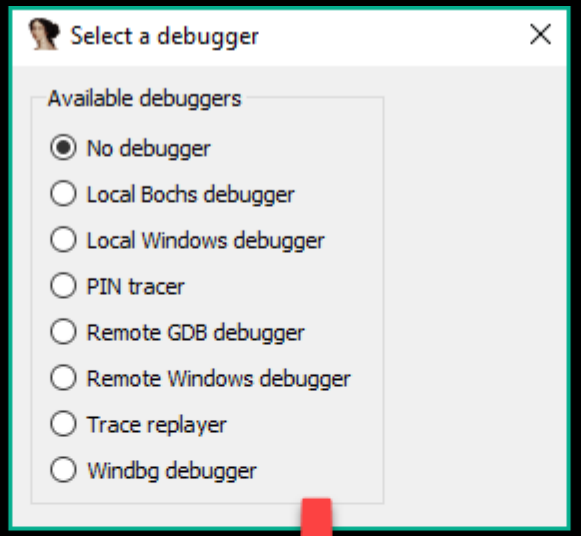
```
00403B98        push    ebp
00403B99        mov     ebp, esp
00403B9B        xor     eax, eax
00403B9D        push    eax                     ; offset
00403B9E        push    eax                     ; a3
00403B9F        push    0C33CBB31h              ; pre_api_hash
00403BA4        push    eax                     ; dllIndex
00403BA5        call    lkb_retrieve_api_addr
00403BA5
```




```
# Initialization ---------------------------------------
FUNC_NAME = "lkb_retrieve_api_addr"
PROTO = "void *__stdcall {:s}(int dllIndex, int pre_api_hash, int a3, unsigned int offset);".format(FUNC_NAME)
```

Recover API Name with IDA AppCall (2)

- **Autostart Debugger**: set breakpoints in the correct context to ensure the environment is ready for the function call




```
# Setting & Starting Debugger -----------------------------
idc.load_debugger("win32", 0)                   # select Local Windows Debugger
idc.set_debugger_options(idc.DOPT_ENTRY_BPT)    # break on program entry point
idc.start_process("","","")                     # start process with default options
idc.wait_for_next_event(idc.WFNE_SUSP, 3)       # wait until the process gets suspended
```

Recover API Name with IDA AppCall (3)

- **Using AppCall**: create a callable object and use it to execute target function with specific prototype.

```
# Execution ----------------------------------------------
retrieve_api_func = idaapi.Appcall.proto(FUNC_NAME, PROTO)  # create a callable object
resolve_n_comment(retrieve_api_func, FUNC_NAME)             # using callable object to exec function and retrieve result
```

Recover API Name with IDA AppCall (4)

```
for xref in idautils.XrefsTo(idc.get_name_ea_simple(func_name)):
    # retrieve required arguments
    call_addr = xref.frm
    print("[+] Processing function {:08X}".format(call_addr))
    arg1_addr = idaapi.get_arg_addrs(call_addr)[0]
    dll_index_val = retrieve_dll_indx(arg1_addr)

    arg2_addr = idaapi.get_arg_addrs(call_addr)[1]
    api_hash_val = retrieve_api_hash(arg2_addr)
    arg3_op = 0x0
    arg4_op = 0x0

    # Call function
    try:
        print ("    [-] dll_index value {:08X}".format(dll_index_val))
        print ("    [-] api_hash value {:08X}".format(api_hash_val))
        addr = func(dll_index_val, api_hash_val, arg3_op, arg4_op)
    except Exception as e:
        print("FAILED: appcall failed: {}".format(e))
        continue

    try:
        # Get exported names of all loaded modules
        names = idaapi.get_debug_names(idaapi.cvar.inf.minEA, idaapi.cvar.inf.maxEA)
        api_name = names[addr]
        # Add comments
        print ("    [-] Recovered 0x%x to %s" % (api_hash_val, api_name))
        idc.set_cmt(call_addr, "{:}".format(names[addr].replace("_", "!")), idc.SN_NOWARN)
    except:
        print("FAILED: to get exported name and add comment")
        continue
```

Result

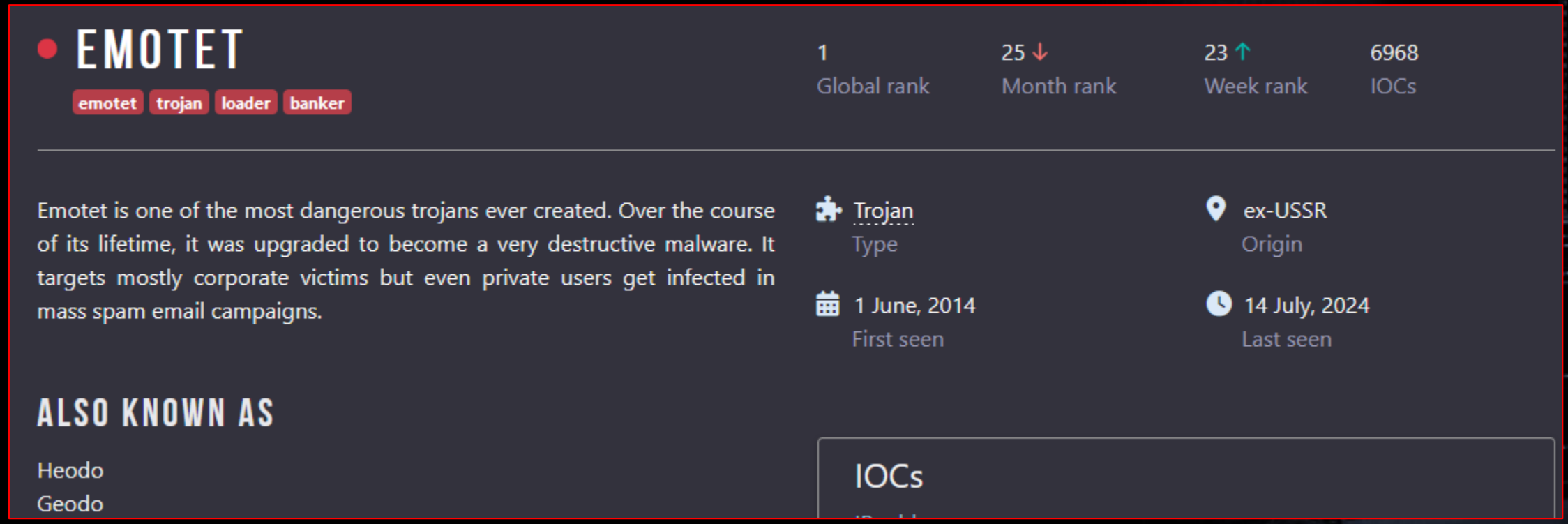
```
.text:00413973        lea     ecx, [ebp+var_14]
.text:00413976        push    ecx
.text:00413977        call    eax
.text:00413977
.text:00413979        call    sub_414059
.text:00413979
.text:0041397E        test    eax, eax
.text:00413980        jz      short loc_4139D3
.text:00413980
.text:00413982        call    sub_413D97
.text:00413982
.text:00413987        push    ebx                     ; offset
.text:00413988        push    ebx                     ; a3
.text:00413989        push    0CF167DF4h              ; pre_api_hash
.text:0041398E        push    ebx                     ; dllIndex
.text:0041398F        mov     esi, eax
.text:00413991        call    lkb_retrieve_api_addr   ; kernel32!CreateMutexW
.text:00413991
.text:00413996        push    esi
.text:00413997        xor     esi, esi
.text:00413999        inc     esi
.text:0041399A        push    esi
.text:0041399B        push    ebx
.text:0041399C        call    eax
.text:0041399C
.text:0041399E        call    ds:GetLastError
.text:0041399E
00012D91 00413991: sub_413866+12B (Synchronized with Hex View-1)
```

```
Output window
   [-] api_hash value E811E8D4
[INFO]    [-] Recovered 0xe811e8d4 to kernel32_LoadLibraryW    (log:write)
   [-] Recovered 0xe811e8d4 to kernel32_LoadLibraryW
[INFO] [+] Processing function 0041395F      (log:write)
[+] Processing function 0041395F
[INFO]    [-] dll_index value 00000000       (log:write)
   [-] dll_index value 00000000
[INFO]    [-] api_hash value E811E8D4        (log:write)
   [-] api_hash value E811E8D4
[INFO]    [-] Recovered 0xe811e8d4 to kernel32_LoadLibraryW    (log:write)
   [-] Recovered 0xe811e8d4 to kernel32_LoadLibraryW
[INFO] [+] Processing function 0041396E      (log:write)
[+] Processing function 0041396E
[INFO]    [-] dll_index value 00000000       (log:write)
   [-] dll_index value 00000000
[INFO]    [-] api_hash value E811E8D4        (log:write)
   [-] api_hash value E811E8D4
[INFO]    [-] Recovered 0xe811e8d4 to kernel32_LoadLibraryW    (log:write)
   [-] Recovered 0xe811e8d4 to kernel32_LoadLibraryW
[INFO] [+] Processing function 00413991      (log:write)
[+] Processing function 00413991
[INFO]    [-] dll_index value 00000000       (log:write)
   [-] dll_index value 00000000
[INFO]    [-] api_hash value CF167DF4        (log:write)
   [-] api_hash value CF167DF4
[INFO]    [-] Recovered 0xcf167df4 to kernel32_CreateMutexW    (log:write)
   [-] Recovered 0xcf167df4 to kernel32_CreateMutexW
[INFO] [+] Processing function 00413A4C      (log:write)
[+] Processing function 00413A4C
[INFO]    [-] dll_index value 00000000       (log:write)
   [-] dll_index value 00000000
[INFO]    [-] api_hash value E567384D        (log:write)
   [-] api_hash value E567384D
[INFO]    [-] Recovered 0xe567384d to kernel32_ExitProcess     (log:write)
   [-] Recovered 0xe567384d to kernel32_ExitProcess
Debugger: process has exited (exit code -1)
```


DEMO

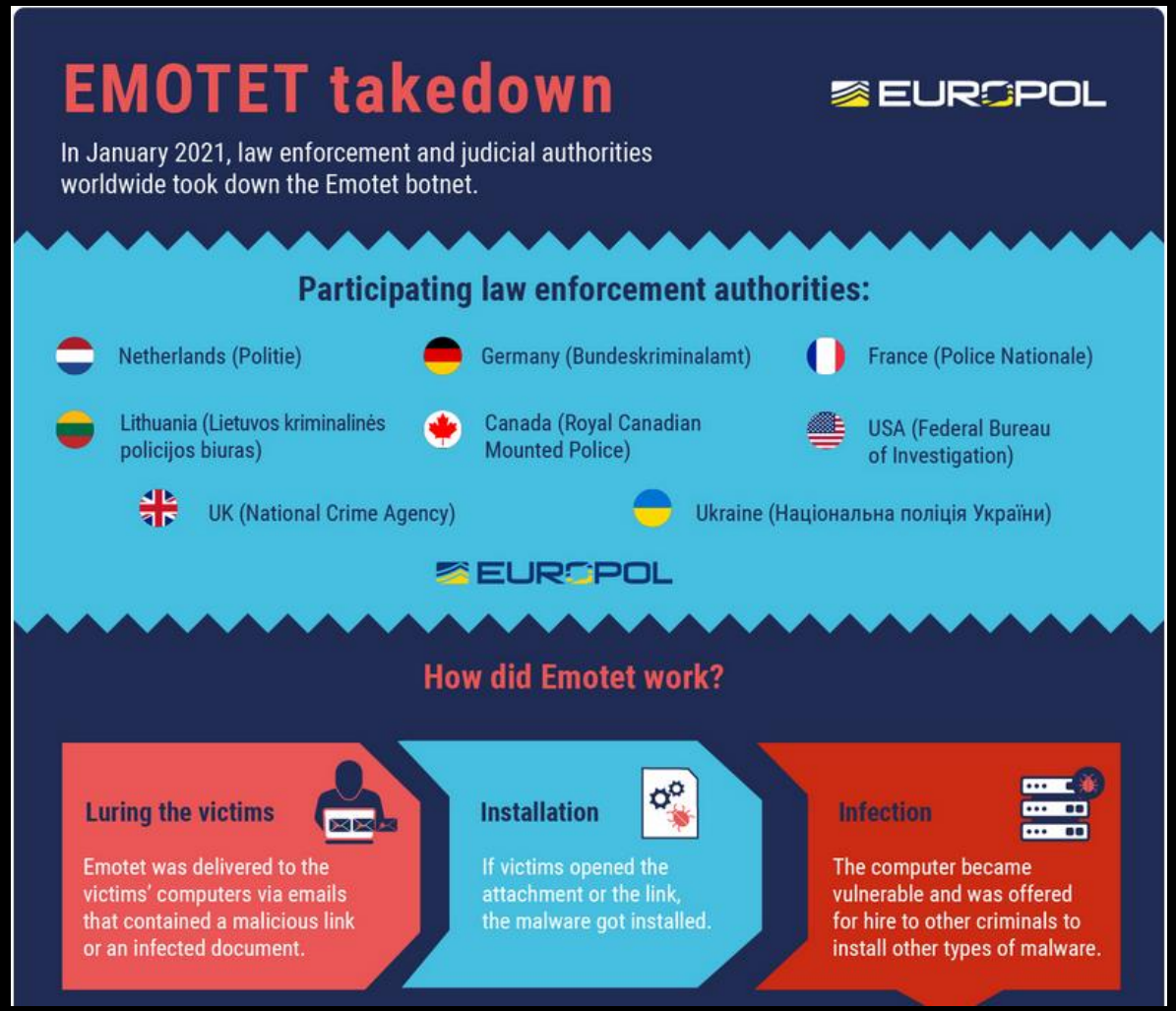
EMOTET

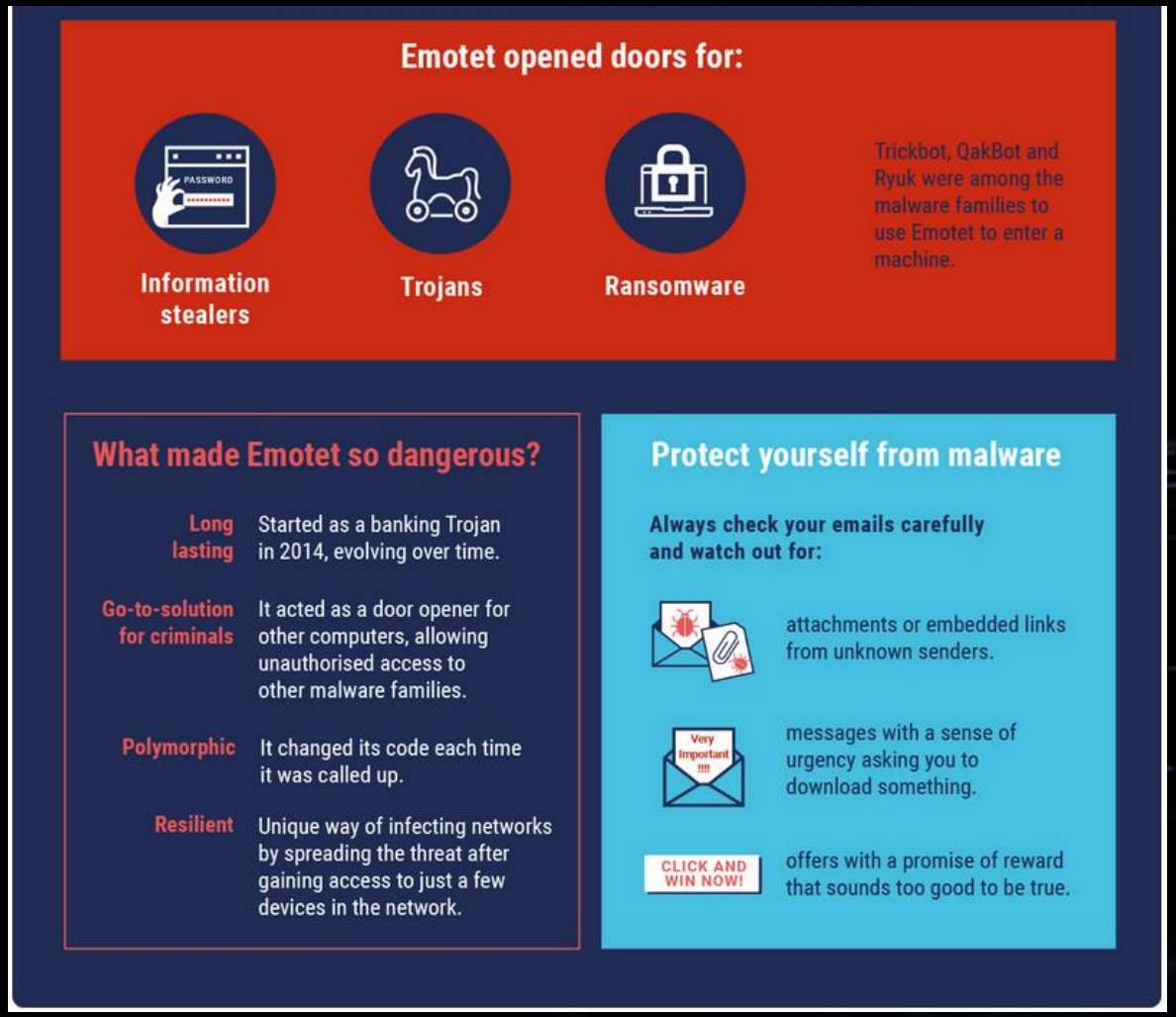
For those who don’t know (1)



https://any.run/malware-trends/emotet

For those who don’t know (2)





World‘s most dangerous malware EMOTET disrupted through global action

For those who don't know (3)

- Shortly after the Emotet takedown, a researcher observed a new payload pushed onto infected machines with a code contained a cleanup routine responsible for uninstalling Emotet after the April 25 2021 deadline.

All Emotet epochs now are delivering the payload (virustotal.com/gui/file/a9c68...) which has the code to remove Emotet on 25 March **2021** 12:00. I believe that **#Emotet** #Killed

```
Tm.tm_year = 121;
Tm.tm_mon = 3;
Tm.tm_mday = 25;
Tm.tm_hour = 12;
Tm.tm_min = 0;
_time64(&Time1);
v0 = _mktime64(&Tm);
*&Time1 = _difftime64(Time1, v0);
if ( *&Time1 > 0.0 )
{
  uninstall_emotet();
}
result = CreateThread(0, 0, uninstall_emotet_thread, 0, 0, 0);
if ( result != -1 )
{
  result = CloseHandle(result);
}
```

25/03/2021

This is exact date information of **#Emotet** uninstallation. It's 25 April **2021** 12:00 local time. Sorry for the mistake in my previous tweet 🙂.

Malwarebytes Threat Intelligen... @MBThreatInt... · Jan 29, 2021

We are checking on the #Emotet 'cleanup binary'.

It seems the actual date to trigger the uninstall routine is April 25.

More details to come.

/cc @campuscodi @LawrenceAbrams

virustotal.com/gui/file/a9c68...
docs.microsoft.com/en-us/cpp/c-ru...

```
1 HANDLE sub_10005F10()
2 {
3   __time64_t v0; // rax
4   HANDLE result; // eax
5   __time64_t Time1; // [esp+0h] [ebp-10h] BYREF
6
7   Tm.tm_year = 121;
8   Tm.tm_mon = 3;
9   Tm.tm_mday = 25;
10  Tm.tm_hour = 12;
11  Tm.tm_min = 0;
12  _time64(&Time1);
13  v0 = _mktime64(&Tm);
14  *&Time1 = _difftime64(Time1, v0);
15  if ( *&Time1 > 0.0 )
16    sub_10005CE0(Time1, HIDWORD(Time1));
17  result = CreateThread(0, 0, StartAddress, 0, 0, 0);
18  if ( result != -1 )
19    result = CloseHandle(result);
20  return result;
21 }
```

MSDN Time Structure Documentation:

tm_mon	Month (0 - 11; January = 0).

April 25 2021

For those who don’t know (4)

For those who don’t know (5)

Emotet Takedown: Time to Celebrate?

by Nikos Mantas on April 16, 2021

At the end of January 2021, Emotet, “the world’s most dangerous malware,” was taken down by law enforcement following an extensive effort by a global coalition of agencies across Europe and the U.S. The effort succeeded in taking down Emotet’s command-and-control infrastructure and at least two of the cybercriminals behind the malware were arrested.

The news of Emotet’s takedown was met with cheers from within the security community, but there were also concerns that the celebrations were premature and that the malware would be back up and running before long.

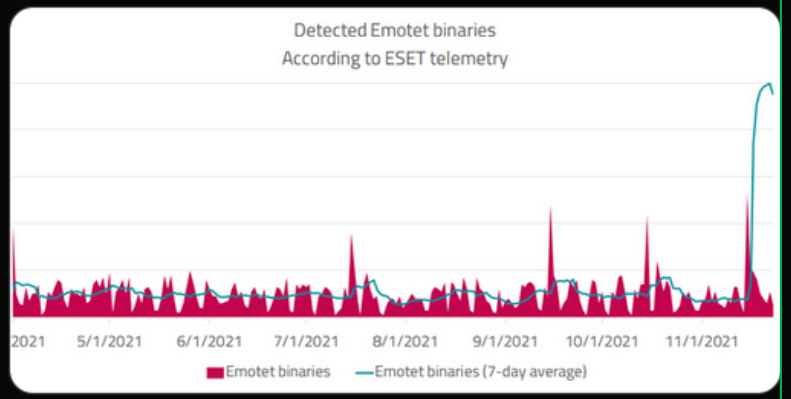
So, what impact will Emotet’s takedown *really* have on organizations today, and are we likely to see the malware rear its unwanted head again any time soon?

https://securityboulevard.com/2021/04/emotet-takedown-time-to-celebrate/

For those who don’t know (6)

ESET Research @ESETresearch · Nov 24, 2021

#BREAKING Bad news, as others already pointed out, **#Emotet** is **back**. #ESETresearch data shows that while Emotet almost completely disappeared after forced uninstallation in Apr 2021, the botnet’s activity has been picking up since the end of Aug. @ondrashmachula @jiriatvirlab 1/3

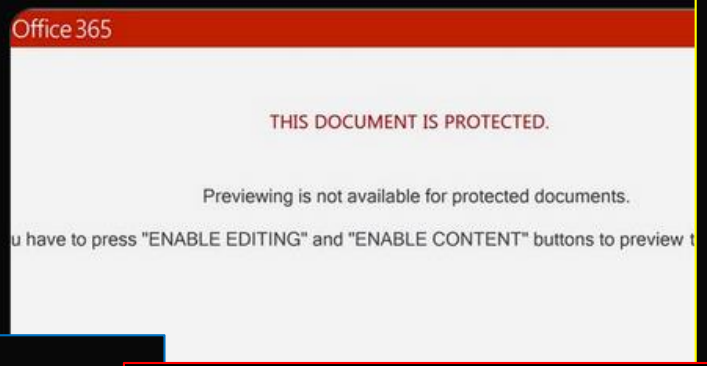


InQuest @InQuest · Nov 17, 2021

Guess who's **back**, **back** again
#Emotet's **back**, tell a friend.

InQuest Lab's Samples: labs.inquest.net/dfi/search/has...

#malware #threatintel



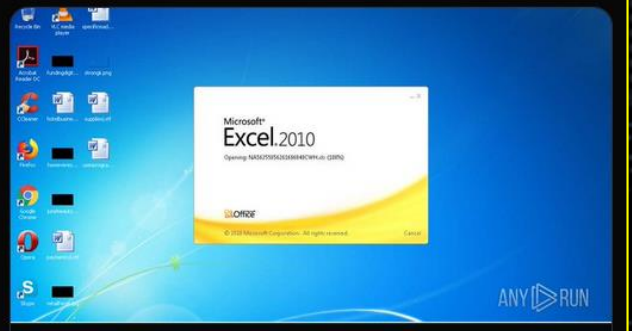
ANY.RUN @anyrun_app · Nov 3, 2022

#Emotet is **back**!

For 6 months, the infamous botnet has shown almost no activity, and now it’s distributing malspam.

Emotet uses a weaponized XLS file with a new lure to download payload and regsvr32 to run it.

✅Find samples in our Public Submissions!



Malwrologist @DissectMalware · Nov 16, 2021

#Emotet is **back** again

Need a script to #deobfuscate its #VBA?

See this:
gist.github.com/DissectMalware...

#oletools

Intel 471 @Intel471Inc · Nov 17, 2021

#Emotet is **back**. Here's what we've discovered so far.

intel471.com
Emotet is back. Here's what we know.
This marks the first time we observed Emotet malware activity after a takedown was announce...

For those who don’t know (7)

Emotet spamming 😮

Hứ

Tue 9:07 PM

Not on VT | [Payload]
MD5: 7EB3D94DBC35E317688F68C9B575176B

E4 botnet
514MB dll [Payload]
49 C&C

Cái gì thế em nhỏ, 514mb dll á

500MB file doc 😮

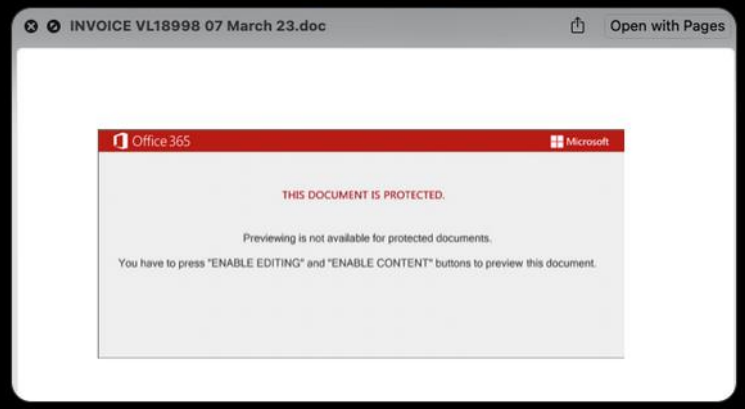
Brad
@malware_traffic

2023-03-07 (Tuesday) - Like others, I'm also seeing #Emotet #malspam, and finding the 500+ MB Word docs. Wait a minute... Emotet comes back after several months, and they're -still- using Microsoft Office docs? Bold move!

10:37 PM · Mar 7, 2023 · **4,158** Views

Cryptolaemus
@Cryptolaemus1

🚨Emotet Awakens🚨 As of 1200UTC Ivan finally got E4 to send spam. We are seeing Red Dawn templates that are very large coming in at over 500MB. Currently seeing a decent flow of spam. Septet of payload URLs and ugly macros. Sample: tria.ge/230307-pljqesh... 1/3



8:35 PM · Mar 7, 2023 · **20.2K** Views

TG Soft
@VirITeXplorer

2023-03-07 #Emotet restarted malspam operation on E4
The mail contains zip attachment without password. The zip file is abou
600 KB, but the doc extracted is about 500 MB.

@sugimu_sec @bomccss @58_158_177_102 @JAMESWT_MHT @zuinmichele @rbreabin

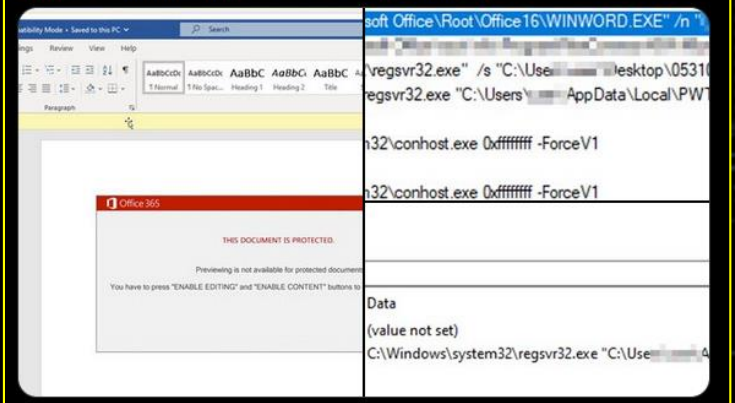
10:25 PM · Mar 7, 2023 · **1,127** Views

Max Malyutin
@Max_Mal_

Undead #Emotet is Back 🥷 500MB #MalDoc 🚨

Ivan is back with old-school #TTPs:
[+] AutoOpen VBA MalDoc macro - T1059.005
[+] Regsvr32 T1218.010
[+] System Network Discovery T1016
[+] Registry Run T1547.001

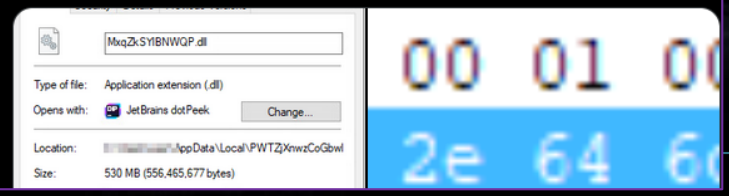
#DFIR Exec Flow: Winword.exe > Regsvr32.exe 💥



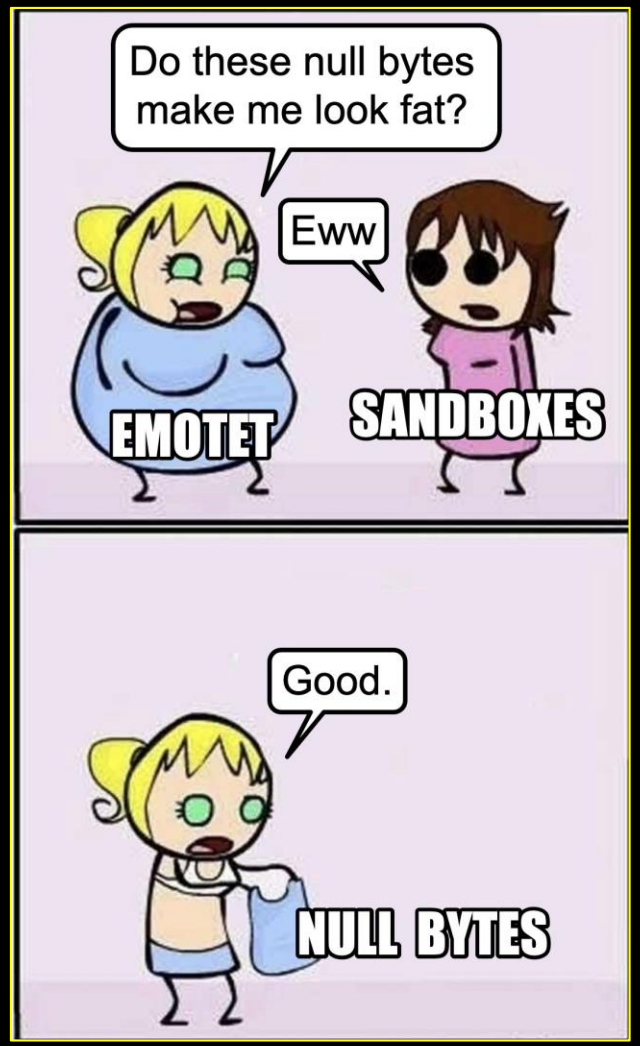
8:50 PM · Mar 7, 2023 · **53.7K** Views

Max Malyutin
@Max_Mal_

#Emotet DLL loader size: 530MB 🤯
Export func: DllRegisterServer
Internal name: E.dll



And Meme Everywhere...



For those who don’t know (8)

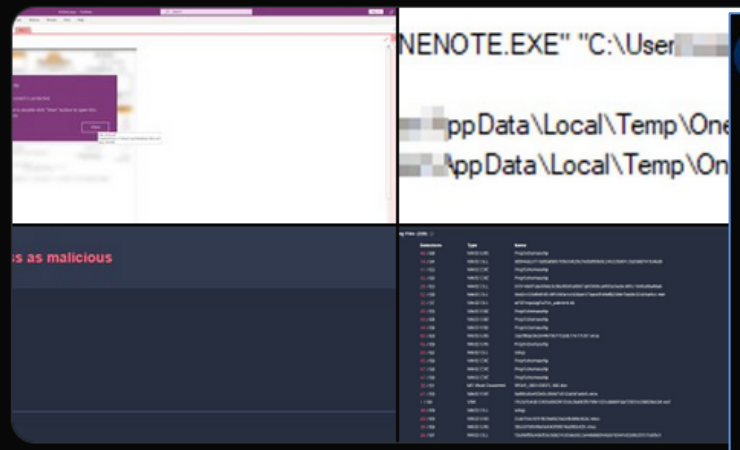
Max_Malyutin @Max_Mal_ · Mar 16

#Emotet Weaponized **OneNote** Infection #TTP 🚨

Exec Flow #DFIR: **OneNote**.exe > Wscript.exe > Regsvr32.exe

[+] VBScript T1059.005: .wsf (download and exec DLL loader)
[+] Regsvr32 T1218.010: DLL loader exec from **OneNote** Exported path

C2: 91.121.146[.]47 Port:8080

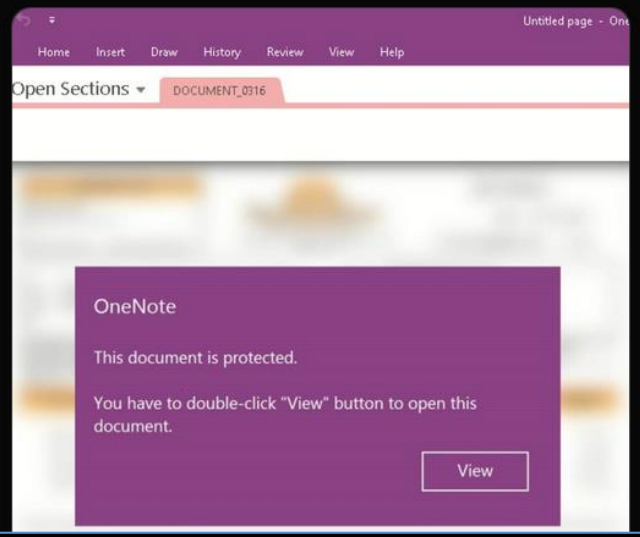


ANY.RUN @anyrun_app · Mar 16

📢 Crooks behind #Emotet keep changing delivery methods

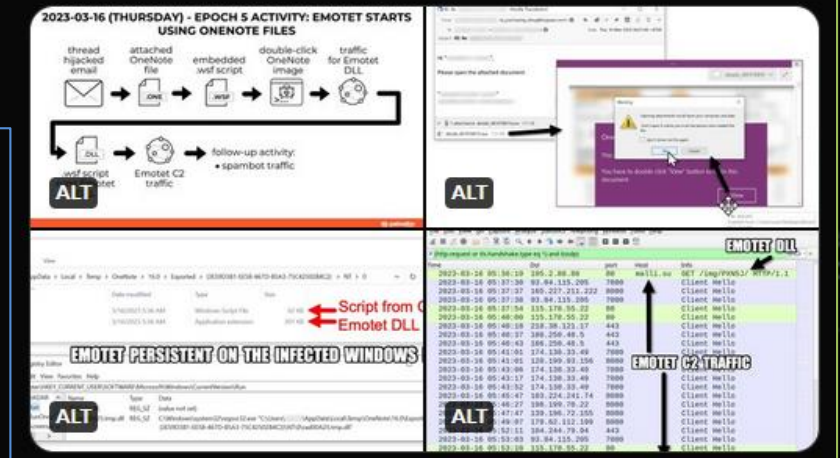
New updates in TTPs: now #Onenote maldoc starts Wscript to download payload and then launches regsvr32 to execute it. And #malware still requires manual execution to run.

Research it with ANYRUN:
app.any.run/tasks/fb5c5943...

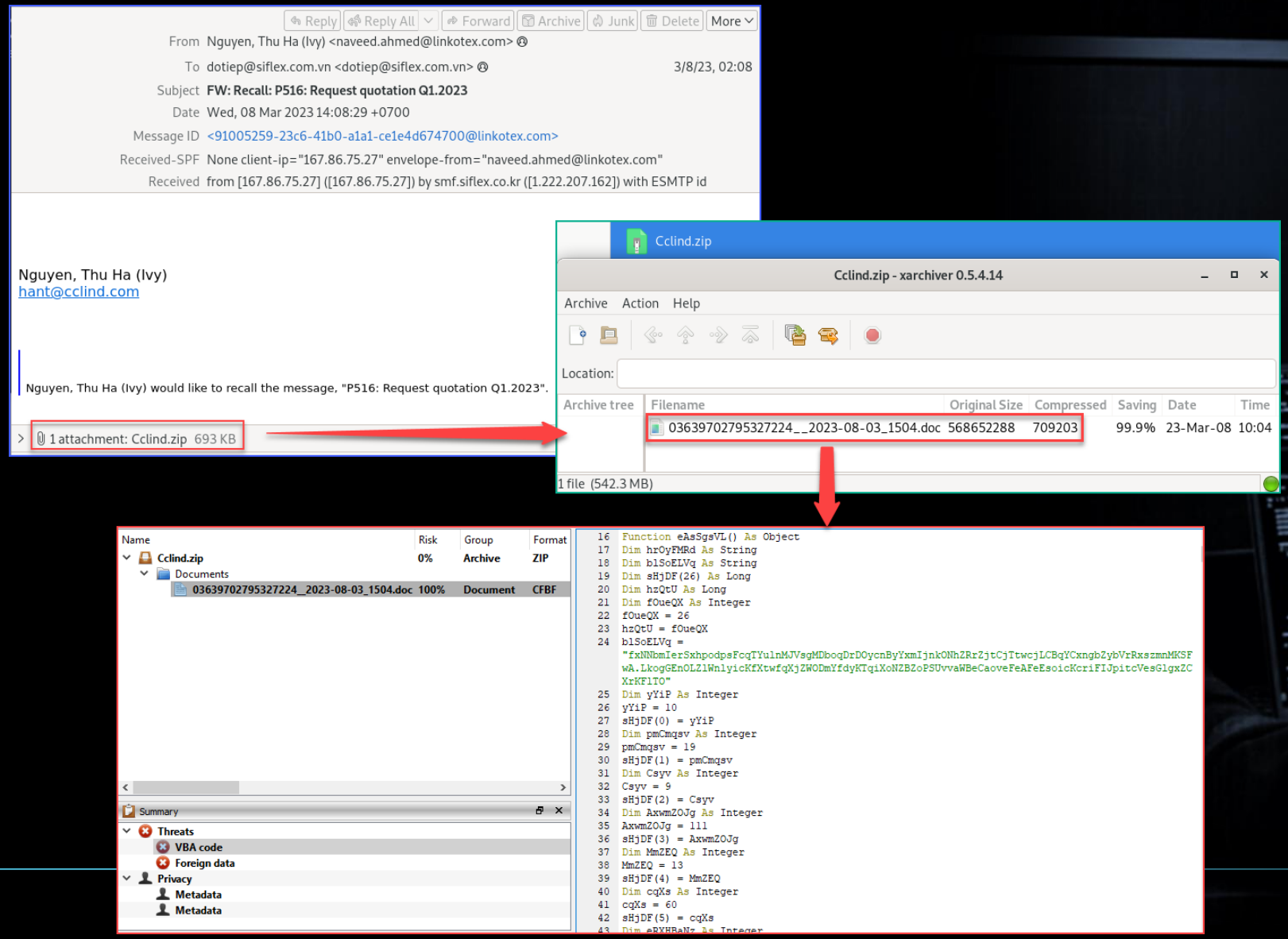


Unit 42 @Unit42_Intel · Mar 17

2023-03-16 (Thursday): #Emotet now also using #OneNote files, but we're still seeing zip attachments with inflated Word docs. IoCs from our latest infection and info on the malware (OneNote files, zip archives, inflated Word docs, etc.) available at bit.ly/3IhXWY2



From “Dong Lao” with Love



From “Dong Lao” with Love

FW: namphuong.nguyen@sutl.com.vn
From: Nam Phuong <namphuong.nguyen@sutl.com.vn>
Date: Mar 15, 2023, 11:49:11 PM
To: VNPT 18001260 N2 <1260mn.n2@vnpt.vn>
message_9761823.one

Hi VNPT,

Mình hay nhậõýợvẫ

Warm regards

(Ms.) Nam Phuong

Representative Office of SUTL Corporation Pte., Ltd.

8A Fl., SGR Building, 167-169 Dien Bien Phu Street, Dakao Ward

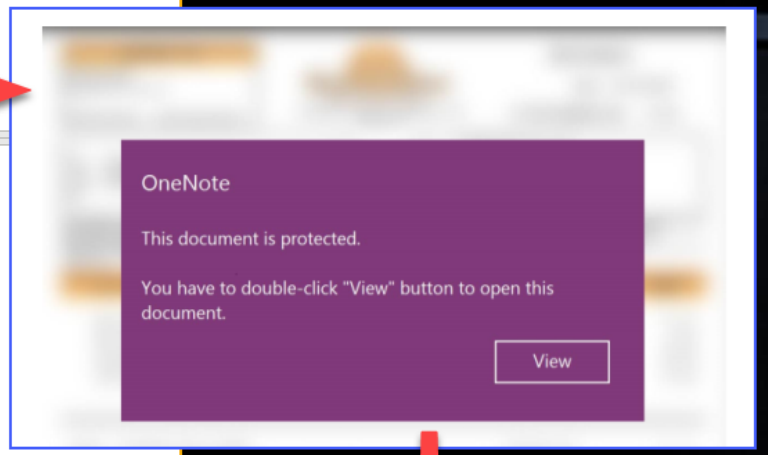
District 1, HCMC, Vietnam.

Cellphone : (84)-0937303638

From: <Trýõng Hỏả>thhai@vnpt.vn
Sent: Thursday, March 16, 2023 10:46 AM
To: namphuong.nguyen@sutl.com.vn
Subject: Re: namphuong.nguyen@sutl.com.vn

See attached

Yours sincerely,

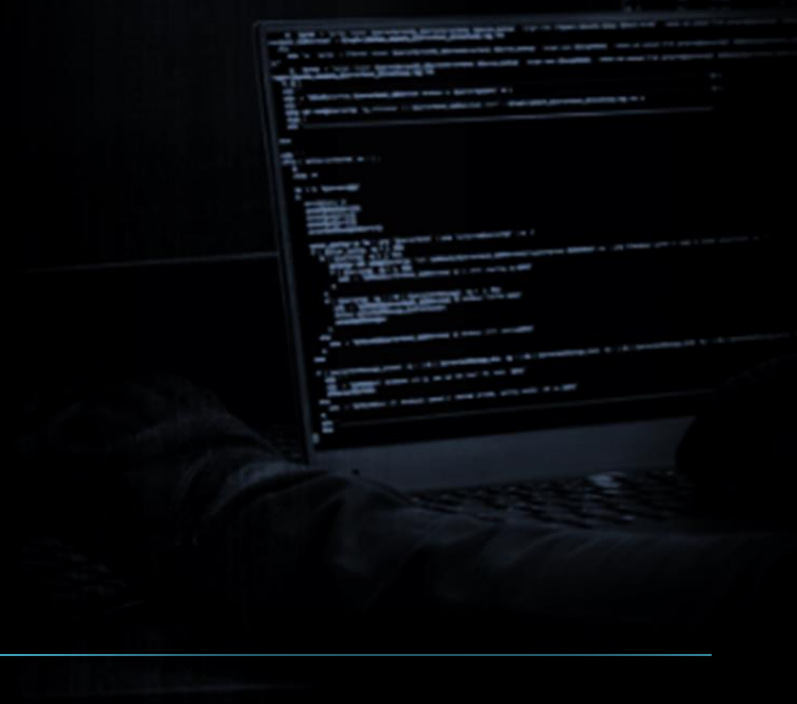



```
File: message_9761823.one
00000000: 3C 6A 6F 62 20 69 64 3D  22 63 6C 6F 63 6B 77 6F  <job id="clockwo
00000010: 72 6B 22 3E 0D 0A 3C 73  63 72 69 70 74 20 6C 61  rk">..<script la
00000020: 6E 67 75 61 67 65 3D 22  56 42 53 63 72 69 70 74  nguage="VBScript
00000030: 22 3E 0D 0A 74 61 6C 6B  65 64 79 20 3D 20 74 61  ">..talkedy = ta
00000040: 6C 6B 65 64 79 20 2B 20  28 22 5C 6F 63 77 31 31  lkedy + ("\ocw11
00000050: 39 33 34 5C 6F 63 77 31  31 36 32 38 5C 6F 63 77  934\ocw11628\ocw
00000060: 31 31 30 31 36 5C 6F 63  77 31 30 30 39 38 5C 6F  11016\ocw10098\o
00000070: 63 77 31 31 33 32 32 5C  6F 63 77 31 31 39 33 34  cw11322\ocw11934
00000080: 5C 6F 63 77 31 31 32 32  30 5C 6F 63 77 31 31 38  \ocw11220\ocw118
00000090: 33 32 5C 6F 63 77 36 32  32 32 5C 6F 63 77 34 39  32\ocw6222\ocw49
000000A0: 39 38 5C 6F 63 77 31 33  32 36 22 29 0D 0A 64 6F  98\ocw1326")..do
000000B0: 77 6E 73 74 61 69 72 73  79 20 3D 20 22 64 6F 77  wnstairsy = "dow
000000C0: 6E 73 74 61 69 72 73 79  22 0D 0A 77 69 74 74 79  nstairsy"..witty
000000D0: 79 20 3D 20 77 69 74 74  79 79 20 2B 20 28 22 62  y = wittyy + ("b
000000E0: 6A 73 7A 76 6D 5C 6F 63  77 66 61 6C 73 65 66 72  jszvm\ocwfalsefr
000000F0: 65 73 68 79 62 61 72 62  61 72 6F 75 73 6C 79 79  eshybarbarouslyy
00000100: 64 69 73 74 72 75 73 74  65 64 79 66 72 65 73 68  distrustedyfresh
00000110: 79 22 29 0D 0A 61 70 74  65 6E 74 79 20 3D 20 22  y")..aptenty = "
00000120: 61 70 74 65 6E 74 79 22  0D 0A 75 6E 66 6F 72 74  aptenty"..unfort
00000130: 75 6E 61 74 65 62 75 74  79 20 3D 20 6D 69 64 28  unatebuty = mid(
00000140: 77 69 74 74 79 79 2C 37  2C 34 29 0D 0A 27 70 61  wittyy,7,4)..'pa
00000150: 72 74 69 6E 67 79 70 61  72 74 69 6E 67 79 0D 0A  rtingypartingy..
00000160: 73 6F 6C 65 6C 79 79 20  3D 20 53 70 6C 69 74 28  solelyy = Split(
00000170: 74 61 6C 6B 65 64 79 2C  75 6E 66 6F 72 74 75 6E  talkedy,unfortun
00000180: 61 74 65 62 75 74 79 2C  2D 31 2C 30 29 0D 0A 62  atebuty,-1,0)..b
00000190: 61 72 62 65 6C 65 64 79  20 3D 20 22 62 61 72 62  arbeledy = "barb
000001A0: 65 6C 65 64 79 22 0D 0A  66 6F 72 20 76 6F 6C 75  eledy"..for volu
000001B0: 74 70 61 74 79 20 3D 20  31 20 74 6F 20 55 62 6F  tpaty = 1 to Ubo
000001C0: 75 6E 64 28 73 6F 6C 65  6C 79 79 29 0D 0A 09 63  und(solelyy)...c
```

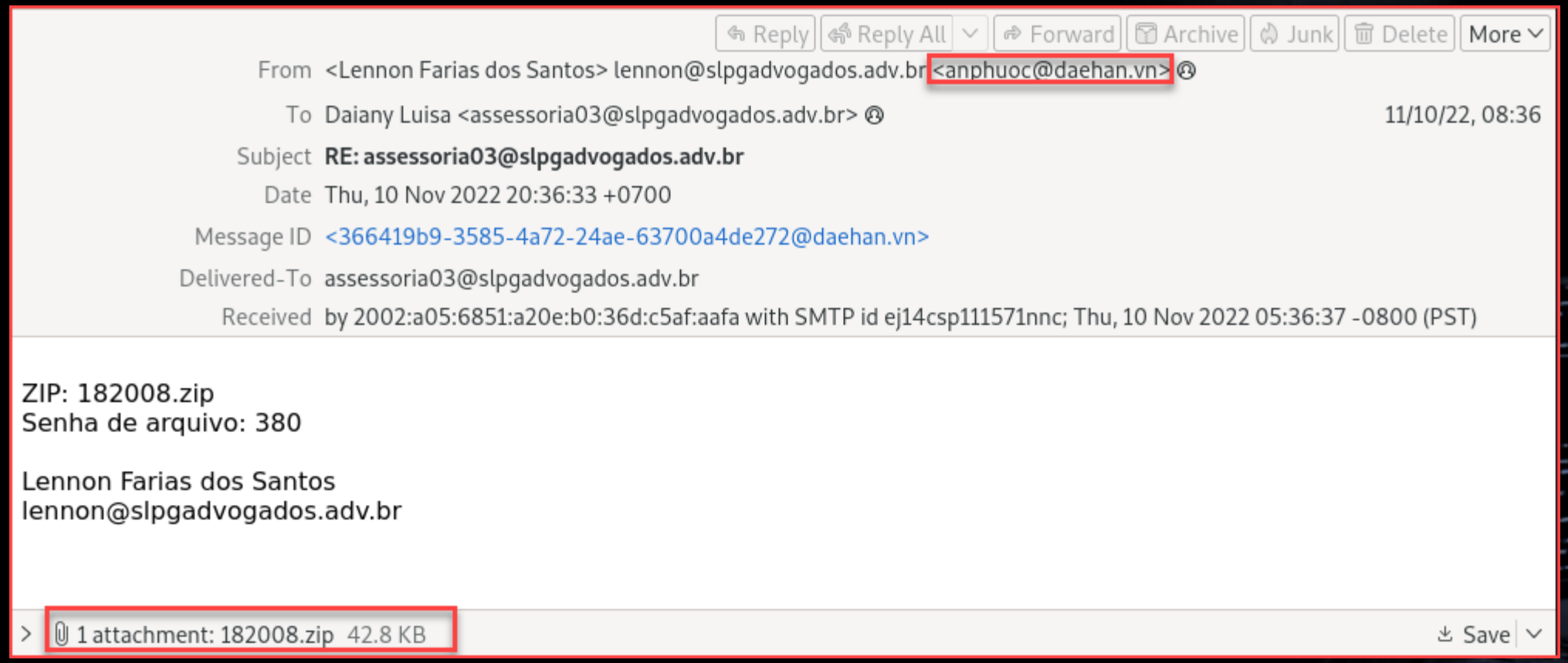
Reversing Emotet

Reversing Engineering Emotet

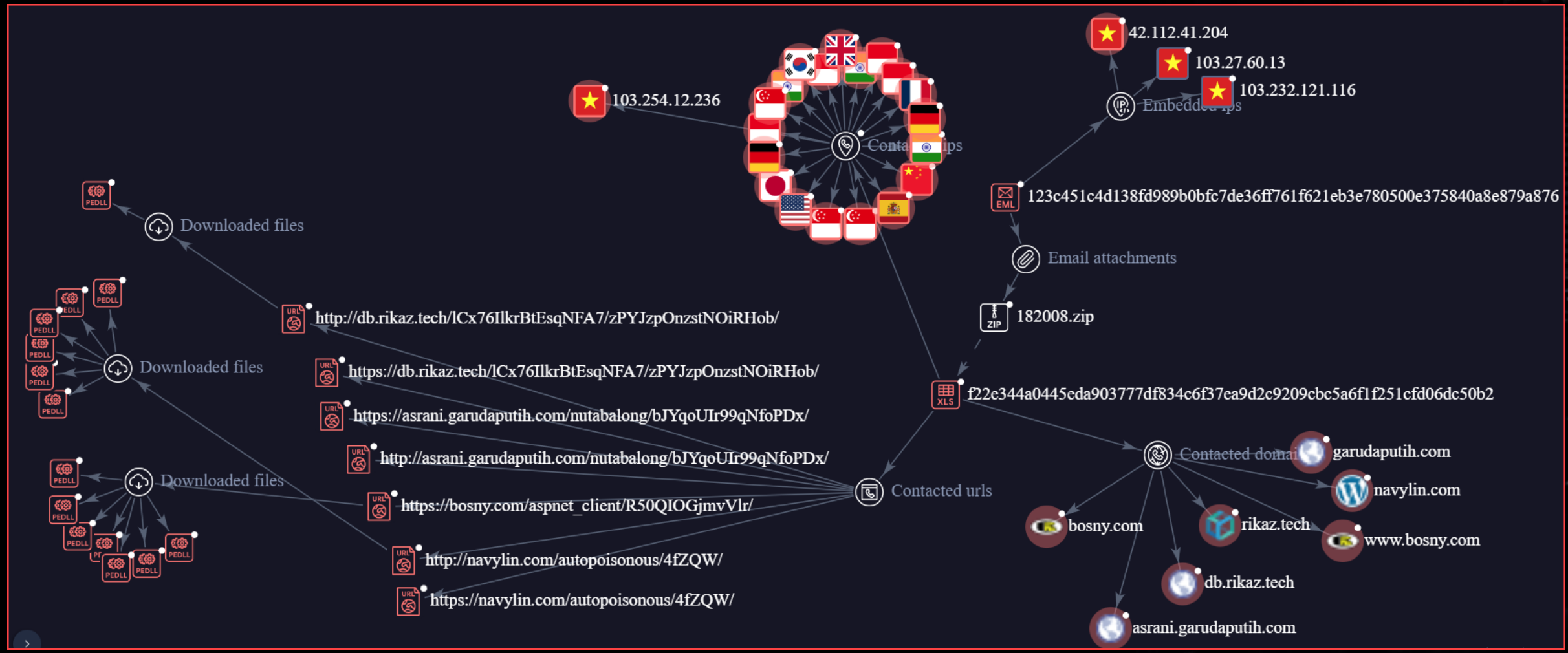
1. **Decrypt strings** by using IDA Appcall feature.
2. **Extract C2s configuration** also using IDA Appcall.



Context (1)

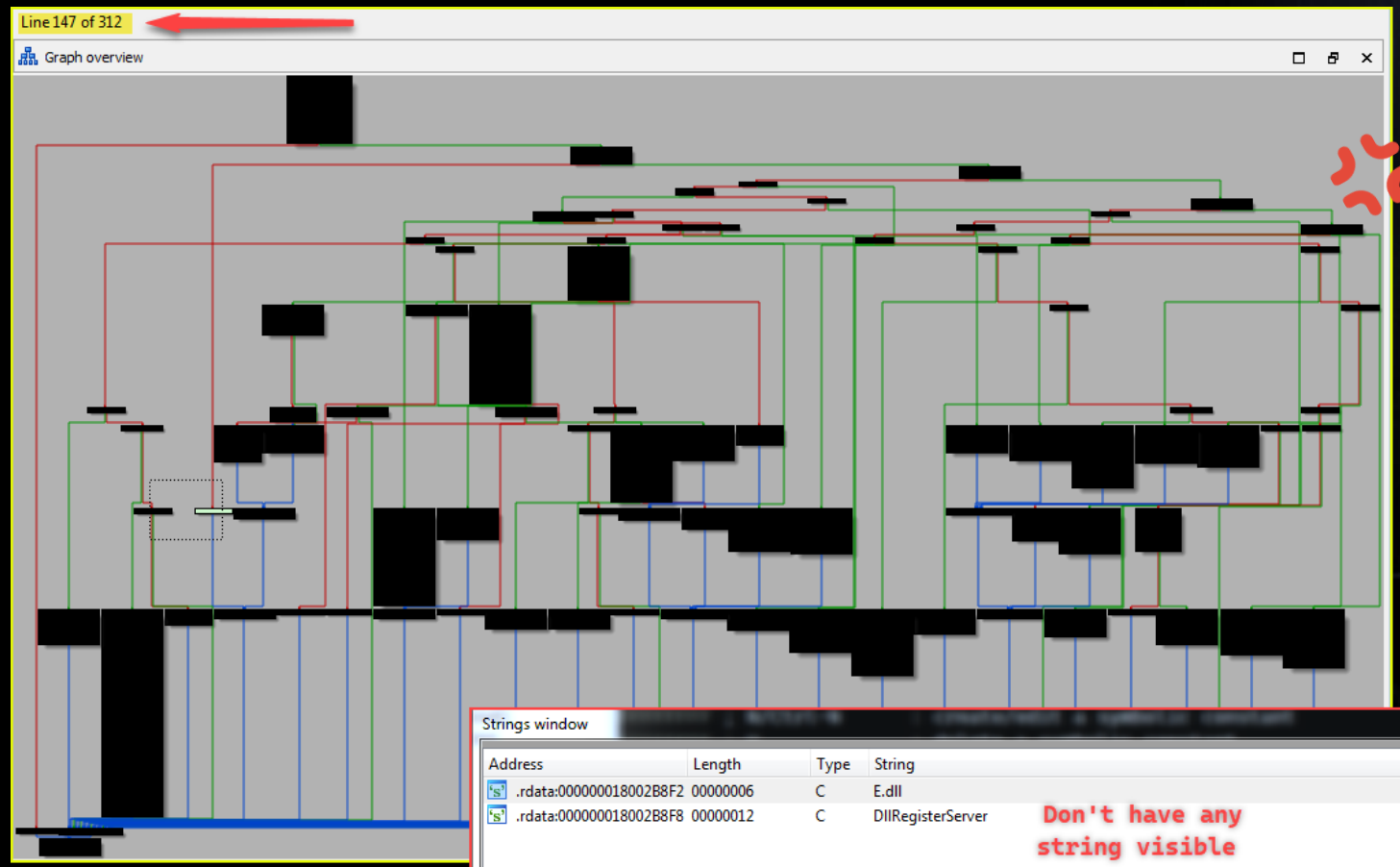

Reply
Reply All
Forward
Archive
Junk
Delete
More
From <Lennon Farias dos Santos> lennon@slpgadvogados.adv.br <anphuoc@daehan.vn>
To Daiany Luisa <assessoria03@slpgadvogados.adv.br>
11/10/22, 08:36
Subject RE: assessoria03@slpgadvogados.adv.br
Date Thu, 10 Nov 2022 20:36:33 +0700
Message ID <366419b9-3585-4a72-24ae-63700a4de272@daehan.vn>
Delivered-To assessoria03@slpgadvogados.adv.br
Received by 2002:a05:6851:a20e:b0:36d:c5af:aafa with SMTP id ej14csp111571nnc; Thu, 10 Nov 2022 05:36:37 -0800 (PST)
ZIP: 182008.zip
Senha de arquivo: 380
Lennon Farias dos Santos
lennon@slpgadvogados.adv.br
1 attachment: 182008.zip 42.8 KB
Save

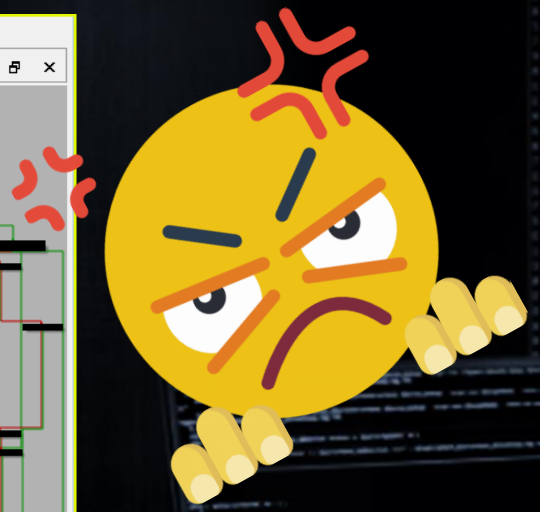
Context (2)


103.254.12.236
42.112.41.204
103.27.60.13
103.232.121.116
Embedded IPs
Contacted IPs
123c451c4d138fd989b0bfc7de36ff761f621eb3e780500e375840a8e879a876
Email attachments
182008.zip
Downloaded files
Downloaded files
Downloaded files
http://db.rikaz.tech/lCx76IlkrBtEsqNFA7/zPYJzpOnzstNOiRHob/
https://db.rikaz.tech/lCx76IlkrBtEsqNFA7/zPYJzpOnzstNOiRHob/
https://asrani.garudaputih.com/nutabalong/bJYqoUIr99qNfoPDx/
http://asrani.garudaputih.com/nutabalong/bJYqoUIr99qNfoPDx/
https://bosny.com/aspnet_client/R50QIOGjmvVlr/
http://navylin.com/autopoisonous/4fZQW/
https://navylin.com/autopoisonous/4fZQW/
f22e344a0445eda903777df834c6f37ea9d2c9209cbc5a6f1f251cfd06dc50b2
Contacted urls
Contacted domains
garudaputih.com
navylin.com
bosny.com
rikaz.tech
www.bosny.com
db.rikaz.tech
asrani.garudaputih.com

Decrypt Strings

- First look into the Core Emotet Dll...





Strings window

Address	Length	Type	String
.rdata:000000018002B8F2	00000006	C	E.dll
.rdata:000000018002B8F8	00000012	C	DllRegisterServer

Don't have any string visible

Imports

Address	Ordinal	Name	Library

...and no imports

Decrypt Strings (1)

- The fastest way is to find the function that calls the **LoadLibraryW** API because this function will take as an argument the name of the module to be loaded.

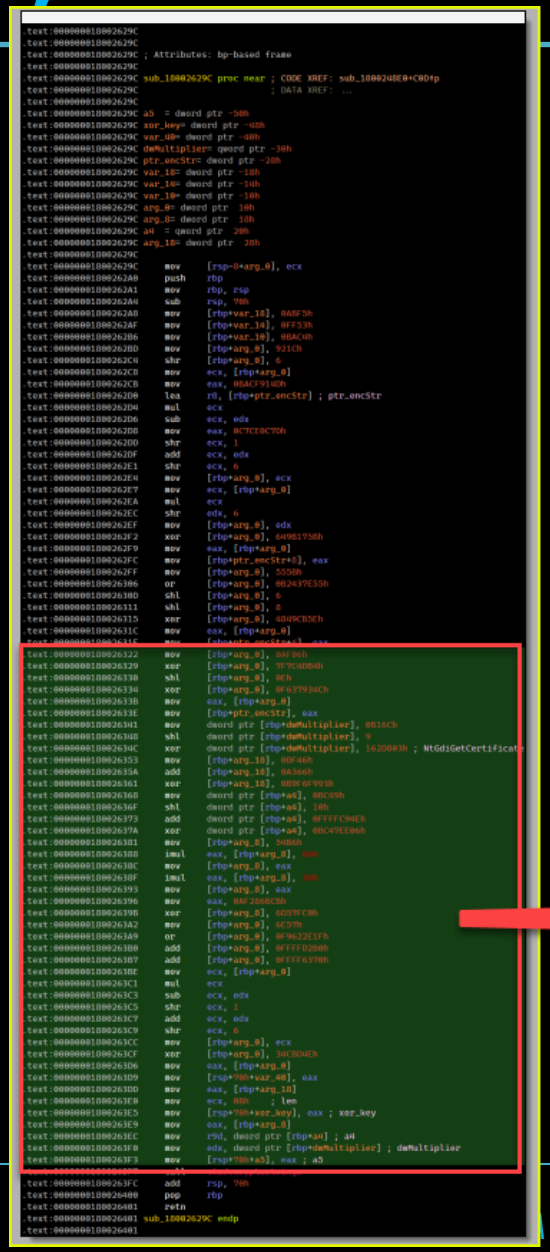
```
while ( TRUE )
{
  while ( TRUE )
  {
    while ( control_state_var > 0x9CC3 )
    {
      switch ( control_state_var )
      {
        case 0xD9D4:
          wstr_shlwapi_dll = et_return_wstr_shlwapi_dll();// 'shlwapi.dll'
          g_et_module_handle->shlwapi_dll_hdl = et_load_specified_module(wstr_shlwapi_dll, 0x28992i64, 0x25EFDi64, 0x43846i64, 0xFB2F7);
          et_free_heap_mem(0xB63FAi64, wstr_shlwapi_dll, 0xF503Di64, 0x2FC9Ci64);
          control_state_var = 0xF59F;
          break;
        case 0xDB5F:
          g_et_module_
          control_sta
          break;
```

return the specified Dll's name

```
HMODULE __fastcall et_load_specified_module(LPCWSTR lpLibFileName, __int64 a2, __int64 a3, __int64 a4, int a5)
{
  // [COLLAPSED LOCAL DECLARATIONS. PRESS KEYPAD CTRL-"+" TO EXPAND]

  et_process_args(lpLibFileName, a2, a3, a4);
  LoadLibraryW = *::LoadLibraryW;
  if ( *::LoadLibraryW )
    return LoadLibraryW(lpLibFileName);
  LoadLibraryW = et_retrieve_api_addr(func_kernel32_LoadLibraryW, kernel32_dll_hash);
  *::LoadLibraryW = LoadLibraryW;
  return LoadLibraryW(lpLibFileName);
}
```

Decrypt Strings (2)

- The pseudocode at **et_return_wstr_shlwapi_dll (sub_18002629C)** stores its encrypted string as stack string, then calls the **et_decrypt_string** function to decrypt.



```
WCHAR *__stdcall sub_18002629C()
{
  // [COLLAPSED LOCAL DECLARATIONS. PRESS KEYPAD CTRL-"+" TO EXPAND]

  v2 = 0xABF5;
  v3 = 0xFF53;
  v4 = 0xBAC4;
  encStr[2] = 0x649B175B;
  encStr[1] = 0x979E0B5E;
  encStr[0] = 0xCE9B134C;
  return et_decrypt_string(0xBi64, 3i64, encStr, 0xF2748i64, 0xE6510, 0xB9F77B3F);
}
```

Emotet stores its encrypted string as stack string

length of string

multiplier for allocate heap memory

encrypted string

key to perform xor

Decrypt Strings (3) (Pseudocode)

```
WCHAR *__fastcall et_decrypt_strings(__int64 len, __int64 dwMultiplier, _DWORD *ptr_encStr, __int64 a4, int a5, int xor_key)
{
  // [COLLAPSED LOCAL DECLARATIONS. PRESS KEYPAD CTRL-"+" TO EXPAND]

  dwMultiplier = dwMultiplier;
  len = len;
  et_process_args(len, dwMultiplier, ptr_encStr, a4);
  ptr_decStrBuf = et_allocate_heap_memory_wrap(8 * dwMultiplier);
  if ( !ptr_decStrBuf )
    return ptr_decStrBuf;
  cnt = 0i64;
  dwMaxCount = (4 * dwMultiplier + 3) >> 2;
  if ( ptr_encStr > &ptr_encStr[dwMultiplier] )
    dwMaxCount = 0i64;
  if ( dwMaxCount )
  {
    do
    {
      dwEncStr = *ptr_encStr;
      ++cnt;
      ++ptr_encStr;
      dwDecStr = xor_key ^ dwEncStr;
      *ptr_decStrBuf = dwDecStr;
      LOWORD(v17) = dwDecStr;
      dwDecStr >>= 0x10;
      ptr_decStrBuf += 4;
      ptr_decStrBuf[-3u] = v17 >> 8;
      ptr_decStrBuf[-2u] = dwDecStr;
      ptr_decStrBuf[-1u] = dwDecStr >> 8;
    }
    while ( cnt < dwMaxCount );
  }
  ptr_decStrBuf[len] = 0;
  return ptr_decStrBuf;
}
```

1

2

Decrypt Strings (4) (Verify)

- Trying to xor each value with **xor_key**

```
 8  encStr[2] = 0x649B175B;
 9  encStr[1] = 0x979E0B5E;
10  encStr[0] = 0xCE9B134C;
11  return et_decrypt_string(0xBi64, 3i64, encStr, 0xF2748i64, 0xE6510, 0xB9F77B3F);
12}
```

```
000257F7 sub_18002629C:11 (1800263F7) (Synchronized with IDA View-A, Hex View-1)
```

shlwapi.dll

Output window

```
IDC>0xCE9B134C ^ 0xB9F77B3F
 2003593331. 776C6873h 16733064163o 00000000000000000000000000000011101110110110001101000011100011b 'shlw....'
IDC>0x979E0B5E ^ 0xB9F77B3F
  778661985. 2E697061h  56322701410 00000000000000000000000000000010111001101001011100000110000 1b 'api.....'
IDC>0x649B175B ^ 0xB9F77B3F
 3714870372. DD6C6C64h 33533066144o 00000000000000000000000000000011011101011011000110110001100100b 'dll....'
```

- Or through debugging

Hex View-1

```
00000000002C6180  00 00 00 80 01 00 00 00  63 96 5D 68 E4 05 D9 01  ...€....c-]hä.Ù.
00000000002C6190  AB AB AB AB AB AB AB AB  AB AB AB AB AB AB AB AB  «««««««««««««««««
00000000002C61A0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00000000002C61B0  EE FE EE FE EE FE EE FE  1C AF 91 5B 8B F3 00 38  îþîþîþîþ.¯'[<ó.8
00000000002C61C0  73 00 68 00 6C 00 77 00  61 00 70 00 69 00 2E 00  s.h.l.w.a.p.i ...
00000000002C61D0  64 00 6C 00 6C 00 00 00  AB AB AB AB AB AB AB AB  d.l.l ... ««««««««
00000000002C61E0  AB AB AB AB AB AB AB AB  EE FE EE FE EE FE EE FE  ««««««««îþîþîþîþ
00000000002C61F0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
```

Hex View-2

```
UNKNOWN 00000000002C61C0: debug044:00000000002C61C0
```

Decrypt Strings (5) (Solution?)

- The encrypted string has a **variable length**.
- The values of the encrypted string are **dynamically calculated** by Emotet before being stored to the stack.
- → Therefore, it is difficult to get these values for writing script to perform decryption.
- → One of the most possible ways is to write a script that uses **IDA Appcall feature** to execute a call to the decryption function and receive the decrypted string as the return result.

Recover Original Strings with IDA AppCall (1)

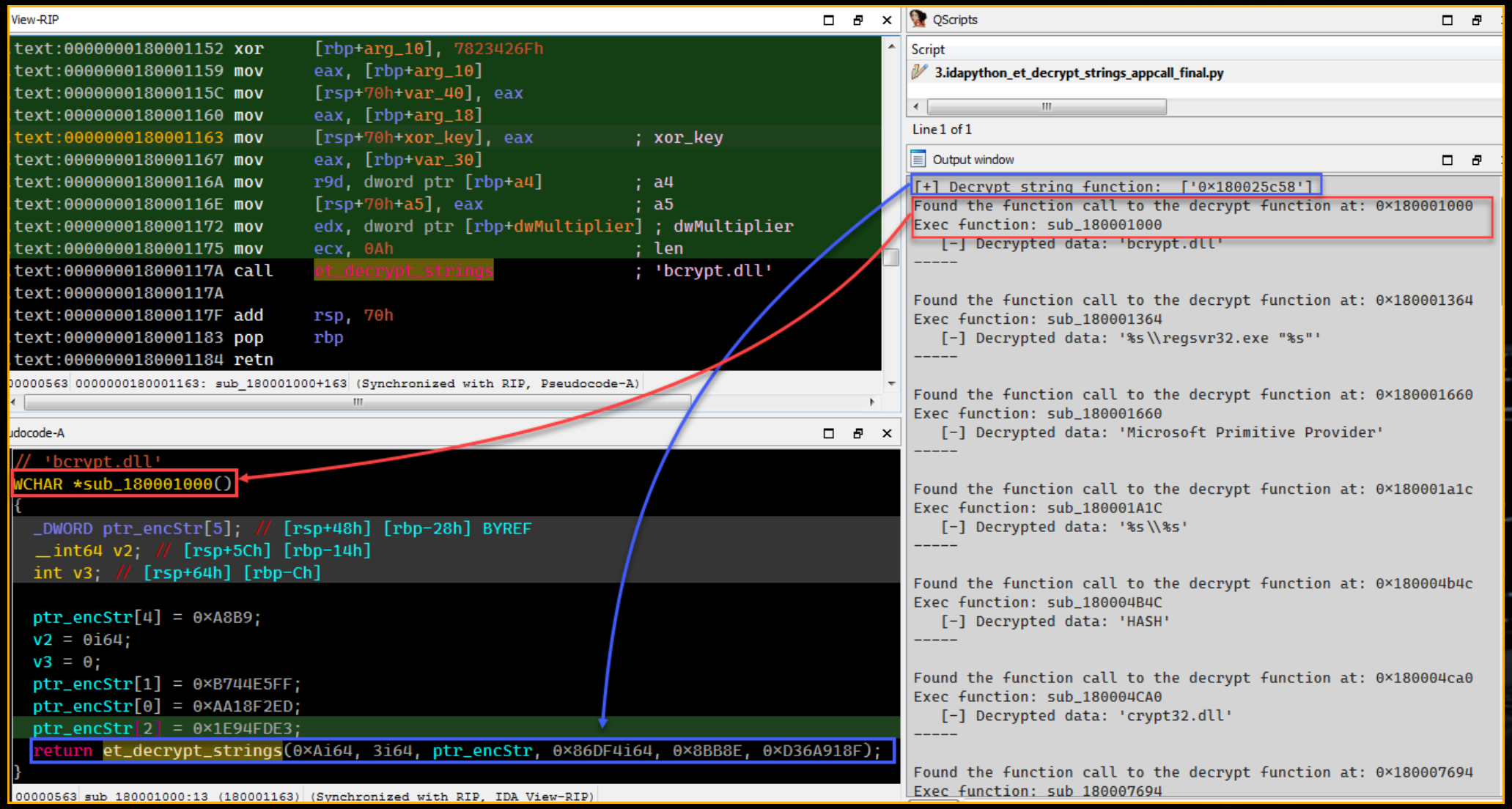
- Write IDAPython script to use AppCall feature.

```
def find_and_decrypt_data(func_addr):
    for call_addr in idautils.CodeRefsTo(func_addr, 1):
        func_call_addr = idaapi.get_func(call_addr).start_ea
        print ("Found the function call to the decrypt function at: 0x%x" % func_call_addr)

        dec_func_name = idc.get_func_name(func_call_addr)
        print ("Exec function: %s" % dec_func_name)
        dec_func_proto = "wchar_t * __fastcall {:s}();".format(dec_func_name)
        dec_func = idaapi.Appcall.proto(dec_func_name, dec_func_proto)

        #Call function to decrypt data and clean the decrypted data
        try:
            dec_data = dec_func()
            dec_data = clean_data(dec_data)
            if dec_data:
                print("   [-] Decrypted data: %s" % repr(dec_data))
                print('-----\n')
        except Exception as e:
            print("FAILED: appcall failed: {}".format(e))
            continue

        #Set comment
        try:
            idc.set_cmt(call_addr, repr(dec_data), idc.SN_NOWARN)
            idc.set_func_cmt(func_call_addr, repr(dec_data), 1)
        except:
            print("FAILED: to add comment")
            continue
```

1

2

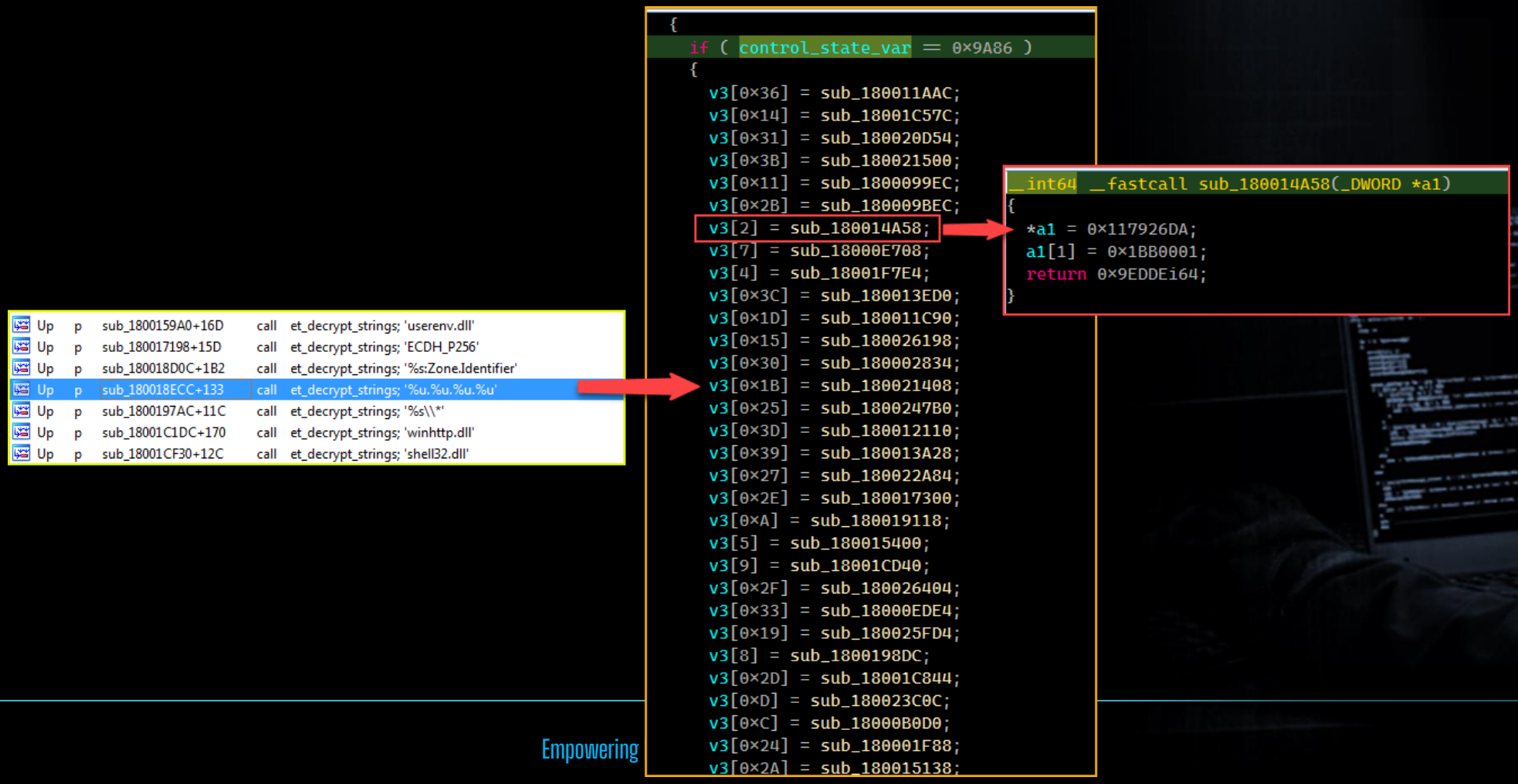
Recover Original Strings with IDA AppCall (2)




MALWARE
MUST
DIE
DEMO

Extract C2s Configuration (1)

- Based on the decrypted string '%u.%u.%u.%u' to determine the function responsible for obtaining the **C2 address (ip:port)**.



Extract C2s Configuration (2)

```
et_c2_func_arr.et_get_c2_ip_port_func[0x36] = et_get_c2_ip_port_54;
et_c2_func_arr.et_get_c2_ip_port_func[0x14] = et_get_c2_ip_port_20;
et_c2_func_arr.et_get_c2_ip_port_func[0x31] = et_get_c2_ip_port_49;
et_c2_func_arr.et_get_c2_ip_port_func[0x3B] = et_get_c2_ip_port_59;
et_c2_func_arr.et_get_c2_ip_port_func[0x11] = et_get_c2_ip_port_17;
et_c2_func_arr.et_get_c2_ip_port_func[0x2B] = et_get_c2_ip_port_43;
et_c2_func_arr.et_get_c2_ip_port_func[2] = et_get_c2_ip_port_2;
et_c2_func_arr.et_get_c2_ip_port_func[7] = et_get_c2_ip_port_7;
et_c2_func_arr.et_get_c2_ip_port_func[4] = et_get_c2_ip_port_4;
et_c2_func_arr.et_get_c2_ip_port_func[0x3C] = et_get_c2_ip_port_60;
et_c2_func_arr.et_get_c2_ip_port_func[0x1D] = et_get_c2_ip_port_29;
et_c2_func_arr.et_get_c2_ip_port_func[0x15] = et_get_c2_ip_port_21;
et_c2_func_arr.et_get_c2_ip_port_func[0x30] = et_get_c2_ip_port_48;
et_c2_func_arr.et_get_c2_ip_port_func[0x1B] = et_get_c2_ip_port_27;
et_c2_func_arr.et_get_c2_ip_port_func[0x25] = et_get_c2_ip_port_37;
et_c2_func_arr.et_get_c2_ip_port_func[0x3D] = et_get_c2_ip_port_61;
et_c2_func_arr.et_get_c2_ip_port_func[0x39] = et_get_c2_ip_port_57;
et_c2_func_arr.et_get_c2_ip_port_func[0x27] = et_get_c2_ip_port_39;
et_c2_func_arr.et_get_c2_ip_port_func[0x2E] = et_get_c2_ip_port_46;
et_c2_func_arr.et_get_c2_ip_port_func[0xA] = et_get_c2_ip_port_10;
et_c2_func_arr.et_get_c2_ip_port_func[5] = et_get_c2_ip_port_5;
et_c2_func_arr.et_get_c2_ip_port_func[9] = et_get_c2_ip_port_9;
et_c2_func_arr.et_get_c2_ip_port_func[0x2F] = et_get_c2_ip_port_47;
et_c2_func_arr.et_get_c2_ip_port_func[0x33] = et_get_c2_ip_port_51;
et_c2_func_arr.et_get_c2_ip_port_func[0x19] = et_get_c2_ip_port_25;
et_c2_func_arr.et_get_c2_ip_port_func[8] = et_get_c2_ip_port_8;
et_c2_func_arr.et_get_c2_ip_port_func[0x2D] = et_get_c2_ip_port_45;
et_c2_func_arr.et_get_c2_ip_port_func[0xD] = et_get_c2_ip_port_13;
et_c2_func_arr.et_get_c2_ip_port_func[0xC] = et_get_c2_ip_port_12;
et_c2_func_arr.et_get_c2_ip_port_func[0x24] = et_get_c2_ip_port_36;
et_c2_func_arr.et_get_c2_ip_port_func[0x2A] = et_get_c2_ip_port_42;
et_c2_func_arr.et_get_c2_ip_port_func[6] = et_get_c2_ip_port_6;
et_c2_func_arr.et_get_c2_ip_port_func[0x10] = et_get_c2_ip_port_16;
et_c2_func_arr.et_get_c2_ip_port_func[0x29] = et_get_c2_ip_port_41;
et_c2_func_arr.et_get_c2_ip_port_func[0x3F] = et_get_c2_ip_port_63;
et_c2_func_arr.et_get_c2_ip_port_func[0xF] = et_get_c2_ip_port_15;
et_c2_func_arr.et_get_c2_ip_port_func[0xE] = et_get_c2_ip_port_14;
et_c2_func_arr.et_get_c2_ip_port_func[0x32] = et_get_c2_ip_port_50;
et_c2_func_arr.et_get_c2_ip_port_func[0x17] = et_get_c2_ip_port_23;
et_c2_func_arr.et_get_c2_ip_port_func[0x37] = et_get_c2_ip_port_55;
et_c2_func_arr.et_get_c2_ip_port_func[0x20] = et_get_c2_ip_port_32;
et_c2_func_arr.et_get_c2_ip_port_func[0x38] = et_get_c2_ip_port_56;
et_c2_func_arr.et_get_c2_ip_port_func[0x35] = et_get_c2_ip_port_53;
et_c2_func_arr.et_get_c2_ip_port_func[0x22] = et_get_c2_ip_port_34;
et_c2_func_arr.et_get_c2_ip_port_func[0x18] = et_get_c2_ip_port_24;
et_c2_func_arr.et_get_c2_ip_port_func[0x3E] = et_get_c2_ip_port_62;
et_c2_func_arr.et_get_c2_ip_port_func[3] = et_get_c2_ip_port_3;
et_c2_func_arr.et_get_c2_ip_port_func[0x23] = et_get_c2_ip_port_35;
et_c2_func_arr.et_get_c2_ip_port_func[0x3A] = et_get_c2_ip_port_58;
```

64 functions

Offset	Size	struct et_c2_ip_port
		{
0000	0004	int ip_addr;
0004	0004	int port_num;
	0008	};

```
__int64 __fastcall et_get_c2_ip_port_45(EMOTET_C2_IP_PORT *et_c2)
{
  et_c2->ip_addr = 0x56D87E67;
  et_c2->port_num = 0x1BB0001;
  return 0x211F1i64;
}
```

Extract C2s Configuration (3)

```
__int64 __fastcall et_get_c2_ip_port_45(EMOTET_C2_IP_PORT *et_c2)
{
  et_c2->ip_addr = 0x56D87E67;
  et_c2->port_num = 0x1BB0001;
  return 0x211F1i64;
}
```

'%u.%u.%u.%u' ->
(0x67.0x7E.0xD8.0x56:0x1BB) =
103.126.216.86:443

Extract C2s Configuration (4)

- Finding the pattern to grab all addresses → "**48 8D 05 ?? ?? ?? ?? 48 89**"

```
                        loc_180019B20:                          ; CODE XREF: et_retrieve_c2s_ip_port+
48 8D 05 85 7F FF FF            lea     rax, et_get_c2_ip_port_54
48 89 85 D0 00 00 00            mov     [rbp+120h+et_c2_func_arr.et_get_c2_ip_port_func+1B0h], rax
48 8D 05 47 2A 00 00            lea     rax, et_get_c2_ip_port_20
48 89 45 C0                     mov     [rbp+120h+et_c2_func_arr.et_get_c2_ip_port_func+0A0h], rax
48 8D 05 14 72 00 00            lea     rax, et_get_c2_ip_port_49
48 89 85 A8 00 00 00            mov     [rbp+120h+et_c2_func_arr.et_get_c2_ip_port_func+188h], rax
48 8D 05 B2 79 00 00            lea     rax, et_get_c2_ip_port_59
48 89 85 F8 00 00 00            mov     [rbp+120h+et_c2_func_arr.et_get_c2_ip_port_func+1D8h], rax
48 8D 05 90 FE FE FF            lea     rax, et_get_c2_ip_port_17
48 89 45 A8                     mov     [rbp+120h+et_c2_func_arr.et_get_c2_ip_port_func+88h], rax
C7 85 30 01 00 00 A6 F6+        mov     [rbp+120h+arg_0], 0F6A6h
```

Occurrences of binary: 48 8D 05 ?? ?? ?? ?? 48 89

Address	Function	Instruction
.text:000000018000D9FF	et_main_proc	lea rax, et_main_routine_callback
.text:0000000180019B20	et_retrieve_c2s_ip_port	lea rax, et_get_c2_ip_port_54
.text:0000000180019B2E	et_retrieve_c2s_ip_port	lea rax, et_get_c2_ip_port_20
.text:0000000180019B39	et_retrieve_c2s_ip_port	lea rax, et_get_c2_ip_port_49
.text:0000000180019B47	et_retrieve_c2s_ip_port	lea rax, et_get_c2_ip_port_59
.text:0000000180019B55	et_retrieve_c2s_ip_port	lea rax, et_get_c2_ip_port_17
.text:0000000180019BB4	et_retrieve_c2s_ip_port	lea rax, et_get_c2_ip_port_43
.text:0000000180019BF0	et_retrieve_c2s_ip_port	lea rax, et_get_c2_ip_port_2
.text:0000000180019BFC	et_retrieve_c2s_ip_port	lea rax, et_get_c2_ip_port_7
.text:0000000180019C95	et_retrieve_c2s_ip_port	lea rax, et_get_c2_ip_port_4
.text:0000000180019CFF	et_retrieve_c2s_ip_port	lea rax, et_get_c2_ip_port_60
.text:0000000180019D4E	et_retrieve_c2s_ip_port	lea rax, et_get_c2_ip_port_29
.text:0000000180019D59	et_retrieve_c2s_ip_port	lea rax, et_get_c2_ip_port_21
.text:0000000180019DB4	et_retrieve_c2s_ip_port	lea rax, et_get_c2_ip_port_48
.text:0000000180019DC2	et_retrieve_c2s_ip_port	lea rax, et_get_c2_ip_port_27
.text:0000000180019E3E	et_retrieve_c2s_ip_port	lea rax, et_get_c2_ip_port_37
.text:0000000180019E80	et_retrieve_c2s_ip_port	lea rax, et_get_c2_ip_port_61
.text:0000000180019E8E	et_retrieve_c2s_ip_port	lea rax, et_get_c2_ip_port_57
.text:0000000180019ED7	et_retrieve_c2s_ip_port	lea rax, et_get_c2_ip_port_39
.text:0000000180019EE2	et_retrieve_c2s_ip_port	lea rax, et_get_c2_ip_port_46
.text:0000000180019FA6	et_retrieve_c2s_ip_port	lea rax, et_get_c2_ip_port_10
.text:0000000180019FF3	et_retrieve_c2s_ip_port	lea rax, et_get_c2_ip_port_5
.text:00000018001A06F	et_retrieve_c2s_ip_port	lea rax, et_get_c2_ip_port_9
.text:00000018001A0CB	et_retrieve_c2s_ip_port	lea rax, et_get_c2_ip_port_47
.text:00000018001A0D9	et_retrieve_c2s_ip_port	lea rax, et_get_c2_ip_port_51
.text:00000018001A111	et_retrieve_c2s_ip_port	lea rax, et_get_c2_ip_port_25
.text:00000018001A158	et_retrieve_c2s_ip_port	lea rax, et_get_c2_ip_port_8
.text:00000018001A164	et_retrieve_c2s_ip_port	lea rax, et_get_c2_ip_port_45
.text:00000018001A19C	et_retrieve_c2s_ip_port	lea rax, et_get_c2_ip_port_13
.text:00000018001A1A7	et_retrieve_c2s_ip_port	lea rax, et_get_c2_ip_port_12
.text:00000018001A238	et_retrieve_c2s_ip_port	lea rax, et_get_c2_ip_port_36
.text:00000018001A243	et_retrieve_c2s_ip_port	lea rax, et_get_c2_ip_port_42
.text:00000018001A24E	et_retrieve_c2s_ip_port	lea rax, et_get_c2_ip_port_6
.text:00000018001A2B1	et_retrieve_c2s_ip_port	lea rax, et_get_c2_ip_port_16
.text:00000018001A313	et_retrieve_c2s_ip_port	lea rax, et_get_c2_ip_port_41
.text:00000018001A31E	et_retrieve_c2s_ip_port	lea rax, et_get_c2_ip_port_63
.text:00000018001A32C	et_retrieve_c2s_ip_port	lea rax, et_get_c2_ip_port_15
.text:00000018001A4AB	et_retrieve_c2s_ip_port	lea rax, et_get_c2_ip_port_14
.text:00000018001A4B6	et_retrieve_c2s_ip_port	lea rax, et_get_c2_ip_port_50
.text:00000018001A4C4	et_retrieve_c2s_ip_port	lea rax, et_get_c2_ip_port_23
.text:00000018001A54E	et_retrieve_c2s_ip_port	lea rax, et_get_c2_ip_port_55

Automate Extract C2s using IDA AppCall (1)

- Write a Python function that performs the task of finding all the addresses of the functions responsible for setting the C2 address.

```
def search_c2_setup_func():
    c2_routines = list()
    seg_mapping = {idc.get_segm_name(x): (idc.get_segm_start(x), idc.get_segm_end(x)) for x in idautils.Segments()}

    patterns = ["48 8D 05 ?? ?? ?? ?? 48 89"]

    for pattern in patterns:
        start = seg_mapping['.text'][0]
        end = seg_mapping['.text'][1]
        while True:
            found_addr = ida_search.find_binary(start, end, pattern, 16, idc.SEARCH_NEXT|idc.SEARCH_DOWN)
            if found_addr != idc.BADADDR:
                start = found_addr + 1
            else:
                break
            print("0x%x %s" % (found_addr, idc.generate_disasm_line(found_addr, 0)))
            if idc.print_insn_mnem(found_addr) == "lea" and idaapi.get_func(found_addr).start_ea == ET_GET_C2s_FUNC_ADDR:
                c2_func_ea = idc.get_operand_value(found_addr, 1)
                if (c2_func_ea not in c2_routines) and (idc.get_func_flags(c2_func_ea)!= -1):
                    c2_routines.append(c2_func_ea)

    return c2_routines
```

Automate Extract C2s using IDA AppCall (2)

- Write IDAPython script to use AppCall feature.

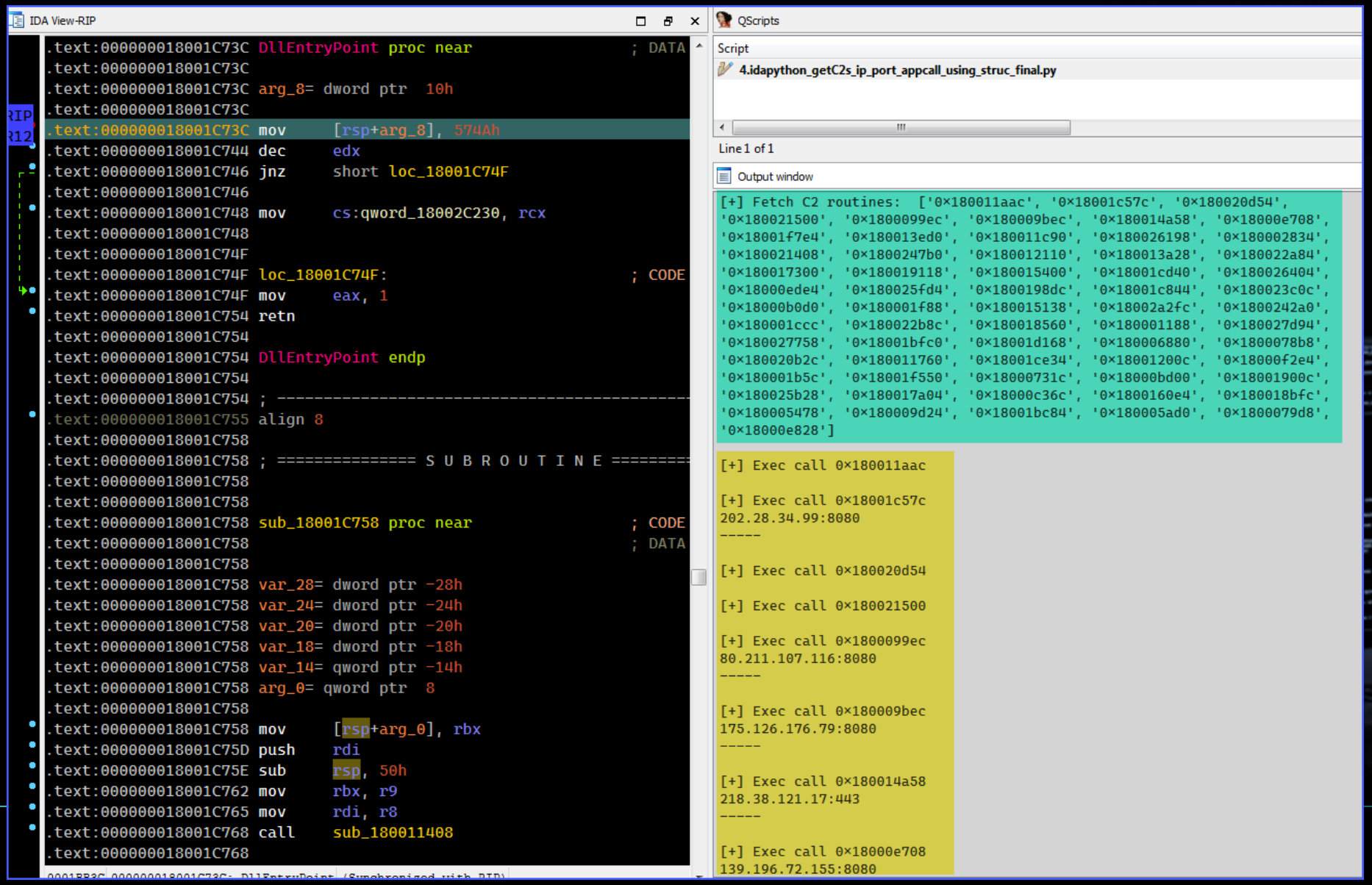
```python
def extract_c2_addr(c2_routine):
    c2_func_name = idc.get_func_name(c2_routine)
    c2_func_proto = "__int64 __fastcall {:s}({:s} *a1);".format(c2_func_name, STRUCT_NAME)

    c2_func = idaapi.Appcall.proto(c2_func_name, c2_func_proto)
    c2_ip_port = idaapi.Appcall.obj(ip_addr=int(), port_num=int())

    print("\n[+] Exec call 0x%x" % c2_routine)
    res = c2_func(c2_ip_port)
    ipnum = c2_ip_port.ip_addr
    portnum = (c2_ip_port.port_num >> 16) & 0xFFFF
    ip_addr = int2ip_little(ipnum & 0xFFFFFFFF)

    cc_addr =''

    if portnum in VALID_C2_PORTS:
        cc_addr = '%s:%d' % (ip_addr, portnum)
        print(cc_addr)
        print('-----')

    return cc_addr
```

2

```python
# ------------------Initialization ------------------
NUMBER_CC_SERVERS = 64
STRUCT_NAME = "et_c2_ip_port"
VALID_C2_PORTS = [443, 8080, 4143, 80, 7080]
ET_GET_C2s_FUNC_ADDR = 0x1800199E8

# ------------------Struct Creation ------------------
sid = idc.get_struc_id(STRUCT_NAME)
if (sid != -1):
    idc.del_struc(sid)
sid = idc.add_struc(-1, STRUCT_NAME, 0)
idc.add_struc_member(sid, "ip_addr", -1, idc.FF_DWORD, -1, 4)
idc.add_struc_member(sid, "port_num", -1, idc.FF_DWORD, -1, 4)
```

1

Automate Extract C2s using IDA AppCall (3)




MALWARE
MUST
DIE
DEMO

Resources

- Introducing the Appcall feature in IDA Pro 5.6
- Practical Appcall examples
- IDA Pro 5.6 Appcall user guide
- Decrypting malware strings with IDA's Appcall
- Native function and Assembly Code Invocation
- [RE025] TrickBot ... many tricks
- [RE026] A Deep Dive into Zloader – the Silent Night
- Defeating BazarLoader Anti-Analysis Techniques
- The DGA of Zloader

Resources

- [RE019] From A to X analyzing some real cases which used recent Emotet samples
- Emotet: Dangerous Malware Keeps on Evolving
- How the new Emotet differs from previous versions
- Emotet Config Extractor
- Emotet 64-bit
- [QuickNote] Emotet epoch4 & epoch5 tactics
- Emotet's Return in Fall 2022 - The Virus Is Back
- [Z2A]Bimonthly malware challege — Emotet (Back From the Dead)

End...