

Quick analysis note about DealPly (Adware)

 kienmanowar.wordpress.com/2021/05/11/quick-analysis-note-about-dealply-adware/

May 11, 2021



Overview

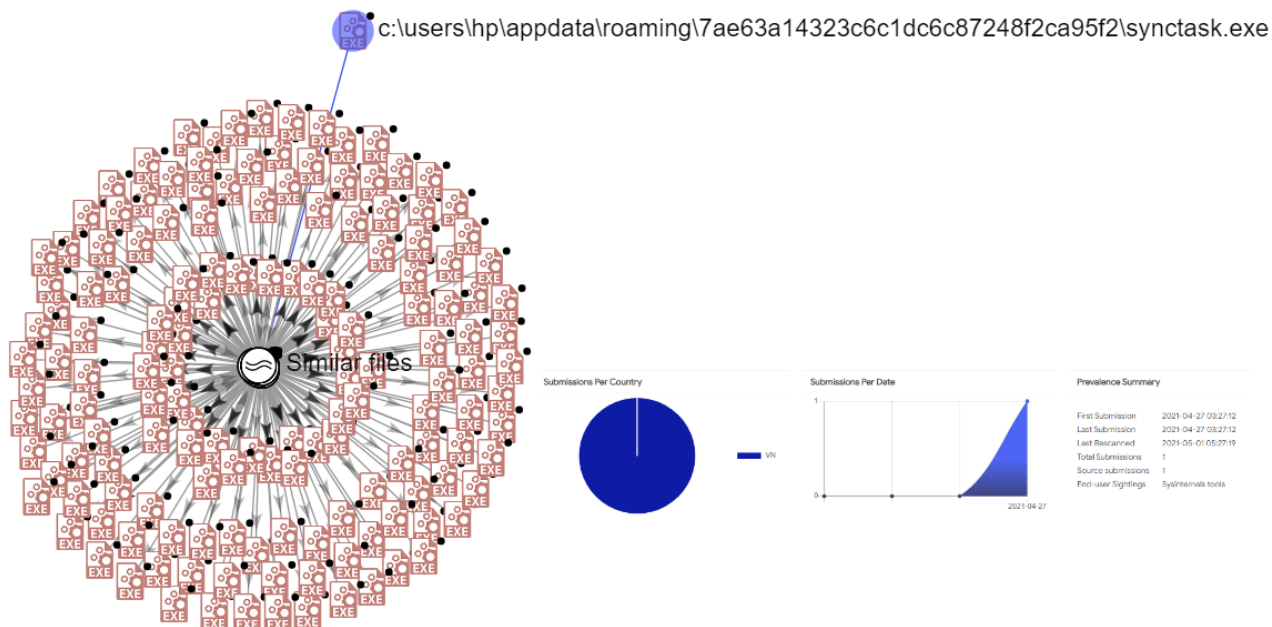
Some information about DealPly can be found here:

The post focuses on the following main sections:

- Unpack wrapper/loader to get main Dll payload.
- Decrypt C2url and strings are used in the malware code.

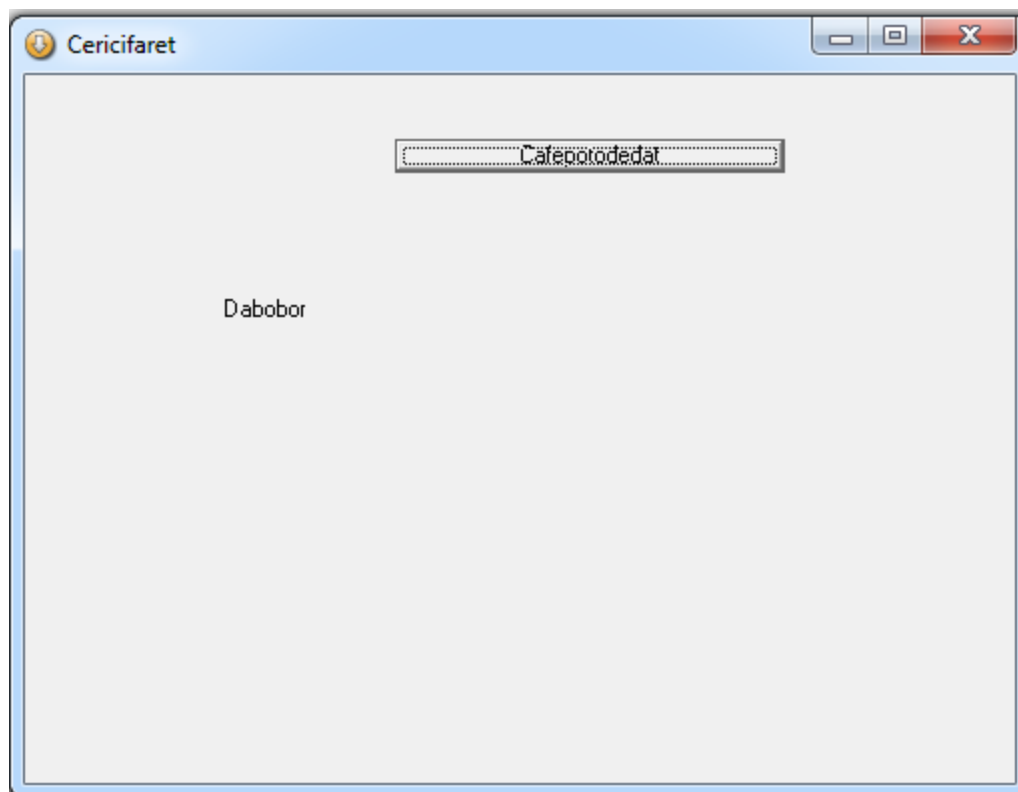
Sample:

SHA-256: [40584f79d109a18b1c4ea7e75a945324978652b6afcc9efbe62241717f0b4685](#)



Unpacking wrapper/loader to get main DLL payload

Most of the DealPly loaders are coded in Delphi. When it runs **without parameter**, it only shows the following form:



When executing **with parameters**, it unpacks a DLL to allocated memory. This DLL will be mainly responsible for connecting and interacting with C2. For unpacking, place breakpoint at **VirtualAlloc**, execute the program and follow the allocated memory region. Keep watching until the loader unpack a new PE (which is a dll), but it was destroyed all relevant information about **DOS_HEADER** and **NT_HEADERS**:

Address	Hex	ASCII
00790000	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00790010	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00790020	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00790030	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00790040	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00790050	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00790060	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00790070	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00790080	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00790090	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
007900A0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
007900B0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
007900C0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
007900D0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
007900E0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
007900F0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00790100	00 01 00 00 00 00 07 00 00 00 00 00 00 00 00 00
00790110	00 00 00 00 E0 00 60 20 00 00 00 19 00 00 00 00
00790120	00 2A 00 00 00 00 00 00 2C B3 01 00 00 10 00 00
00790130	00 C0 01 00 00 00 40 00 00 10 00 00 00 00 00 00
00790140	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00790150	00 30 02 00 00 04 00 00 00 00 00 00 00 00 00 00
00790160	00 00 00 00 2D 44 4F 53 00 00 00 00 00 00 00 00
00790170	00 00 00 00 10 00 00 00 F0 01 00 44 00 00 00 00
00790180	00 E0 01 00 9A 05 00 00 00 20 02 00 00 02 00 00
00790190	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
007901A0	00 00 02 00 28 15 00 00 00 00 00 00 00 00 00 00
007901B0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
007901C0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
007901D0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
007901E0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
007901F0	00 00 00 00 00 00 00 00 43 4F 44 45 00 00 00 00
00790200	44 A3 01 00 00 10 00 00 00 A4 01 00 00 04 00 00
00790210	00 00 00 00 00 00 00 00 00 00 00 00 20 00 00 00
00790220	44 41 54 41 00 00 00 00 FC 09 00 00 00 C0 01 00
00790230	00 00 00 00 00 A8 01 00 00 00 00 00 00 00 00 00
00790240	00 00 00 00 00 00 00 40 00 00 C0 42 53 53 00 00 00
00790250	75 0C 00 00 00 D0 01 00 00 00 00 00 B2 01 00
00790260	00 00 00 00 00 00 00 00 00 00 00 00 00 00 C0
00790270	2E 69 64 61 74 61 00 00 9A 05 00 00 00 E0 01 00
00790280	00 06 00 00 00 B2 01 00 00 00 00 00 00 00 00 00
00790290	00 00 00 00 40 00 00 C0 2E 65 64 61 74 61 00 00
007902A0	44 00 00 00 00 F0 01 00 00 02 00 00 00 B8 01 00
007902B0	00 00 00 00 00 00 00 00 00 00 00 00 40 00 00 50
007902C0	2E 72 65 6C 6F 63 00 00 28 15 00 00 00 00 02 00
007902D0	00 16 00 00 00 B0 01 00 00 00 00 00 00 00 00 00
007902E0	00 00 00 00 40 00 00 50 2E 72 73 72 63 00 00 00
007902F0	00 02 00 00 00 20 02 00 00 02 00 00 00 D0 01 00
00790300	00 00 00 00 00 00 00 00 00 00 00 00 40 00 00 50
00790310	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

Address	Hex	ASCII
007AF0B0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
007AF0C0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
007AF0D0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
007AF0E0	00 32 F0 01 00 01 00 00 00 01 00 00 00 01 00 00
007AF0F0	00 28 F0 01 00 2C F0 01 00 30 F0 01 00 44 B2 01
007AF100	00 40 F0 01 00 00 00 48 65 74 6F 62 6F 67 61 67
007AF110	2E 64 6C 6C 00 52 75 6E 00 00 00 00 00 00 00 00
007AF120	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
007AF130	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
007AF140	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
007AF150	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
007AF160	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
007AF170	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
007AF180	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
007AF190	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
007AF1A0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
007AF1B0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
007AF1C0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
007AF1D0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
007AF1E0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
007AF1F0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
007AF200	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
007AF210	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
007AF220	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
007AF230	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
007AF240	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
007AF250	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
007AF260	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
007AF270	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
007AF280	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
007AF290	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
007AF2A0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
007AF2B0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
007AF2C0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
007AF2D0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
007AF2E0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
007AF2F0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
007AF300	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
007AF310	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

Dump and fix **DOS_HEADERS** and **NT_HEADERS** will get the correct main DLL:

Offset	Name	Value	Meaning
1B800	Characteristics	0	
1B804	TimeDateStamp	0	Thursday, 01.01.1970 00:00:00 UTC
1B808	MajorVersion	0	
1B80A	MinorVersion	0	
1B80C	Name	1F032	Hetobogag.dll
1B810	Base	1	
1B814	NumberOfFunc...	1	
1B818	NumberOfNames	1	
1B81C	AddressOfFunc...	1F028	
1B820	AddressOfNames	1F02C	
1B824	AddressOfNam...	1F030	

Offset	Ordinal	Function RVA	Name RVA	Name	Forwarder
1B828	1	1B244	1F040	Run	

Analysis of DLL payload

Decrypt C2Url

Load the above dumped Dll file into IDA, go to the code of the **Run** function. Here, it calls **f_main_proc** function. The **f_main_proc** accepts the passed parameters, in which the third parameter is encoded C2.

```

CODE:0041B269 Run      endp
CODE:0041B269 ; -----
CODE:0041B26A align 4
CODE:0041B26C _Q_ofa_Q.w::str_nz(void) dd 0FFFFFFFh ; _top
CODE:0041B26C ; DATA XREF: Run+11to
CODE:0041B26C dd 16 ; Len
CODE:0041B26C db 'hz'-10+Q_ofa-Q.w',0 ; Text
CODE:0041B285 align 4
CODE:0041B288 _str_ka_aq___ dd 0FFFFFFFh ; _top
CODE:0041B288 ; DATA XREF: Run+16to
CODE:0041B288 ; Run+1Bto
CODE:0041B288 dd 8 ; Len
CODE:0041B288 db 'ka'aq_-'',0 ; Text

```

```

1 int __usercall Run@<eax>(int a1@<ebx>, int a2@<edi>, int a3@<esi>)
2 {
3     return f_main_proc(
4         &str_ka_aq___[1],
5         &str_ka_aq___[1],
6         &_Q_ofa_Q.w::str_nz[1]
7     ),
8     a1,
9     a3,
10    a2,
11    Webadapt::TBaseValuesListAdapter::ImplGetListNameOfValue,
12    Webadapt::TBaseValuesListAdapter::ImplGetListNameOfValue,
13    Webscriptas::TActiveScriptObjectFactory::CreateProducerObject,
14    0);

```

Diving into the code of `f_main_proc` will find the function responsible for performing the decoding of the malware's C2:

```

int __usercall Unit9_sub_00417670_00417670@<eax>(int szenC2Url@<eax>, int a2@<edx>, int a3@<ebx>, int a4@<esi>)
{
    int len; // eax MAPDST
    _EXCEPTION_REGISTRATION_RECORD *v8; // [esp-14h] [ebp-20h]
    void *v9; // [esp-10h] [ebp-1Ch]
    int *v10; // [esp-Ch] [ebp-18h]
    int v11; // [esp-8h] [ebp-14h]
    int v12; // [esp-4h] [ebp-10h]
    __int32 v13; // [esp+0h] [ebp-Ch]
    unsigned __int8 *tmp_tbl; // [esp+4h] [ebp-8h]
    int savedregs; // [esp+Ch] [ebp+0h]

    tmp_tbl = 0;
    v13 = 0;
    v12 = a3;
    v11 = a4;
    system__LStrAddRef_00403FF0(szenC2Url);
    v10 = &savedregs;
    v9 = &loc_41774D;
    v8 = KeGetPcr() -> NtTib.ExceptionList;
    __writefsdword(0, &v8);
    f_tranform_c2Url(szenC2Url, &v13);
    system__LStrLAsg_00403C48(&szenC2Url, v13);
    if ( system__DynArrayLength_00403E08(szenC2Url) >= 2 )
    {
        len = system__DynArrayLength_00403E08(szenC2Url);
        system__LStrCopy_00404058(szenC2Url, len - 1, 2, &tmp_tbl);
        len = system__DynArrayLength_00403E08(szenC2Url);
        system__LStrSetLength_004040E0(&szenC2Url, len - 2);
        f_decrypt_c2Url(
            szenC2Url,
            ((tmp_tbl[1] + (*tmp_tbl << 8)) & 0xF)
            + 0x10 * (tmp_tbl[1] & 0xF0)
            + (((tmp_tbl[1] + (*tmp_tbl << 8)) & 0xF00) << 8)
            + (((tmp_tbl[1] + (*tmp_tbl << 8)) & 0xF000) << 0xC),
            a2);
    }
    __writefsdword(0, v8);
    v10 = &loc_417754;
    return system__LStrArrayClr_00403BD4(&v13, 3);
}

```

The code at the `f_tranform_c2Url` function will recalculate the bytes of `enc2Url` :

```

encUrl_len = system__DynArrayLength_00403E08(encUrl);
system__LStrSetLength_004040E0(encUrl_transform, encUrl_len);
val_0x2C = 0x2C;
i = 1;
j = 1;
while ( j ≤ system__DynArrayLength_00403E08(encUrl) )
{
    c = encUrl[j++ - 1];
    if ( (c - 0x2B) ≥ 4u )
    {
        calced_val = f_calc_value(c, val_0x2C);
        *(sub_404050(encUrl_transform) + i++ - 1) = calced_val;
    }
    else if ( c = val_0x2C )
    {
        if ( j > system__DynArrayLength_00403E08(encUrl) )
        {
            goto exit_sub;
        }
        ptr_encUrl_transform = sub_404050(encUrl_transform);
        ptr_encUrl_transform[i++ - 1] = encUrl[j++ - 1];
    }
    else
    {
        val_0x2C = c;
    }
}

```

```

int __fastcall f_calc_value(int c, char val_0x2C)
{
    char tmp; // dl
    char tmp2; // dl

    tmp = val_0x2C - 0x2B;
    if ( tmp )
    {
        tmp2 = tmp - 2;
        if ( tmp2 )
        {
            if ( tmp2 = 1 )
            {
                c = c + 0x85;
            }
            else
            {
                c = c;
            }
        }
        else
        {
            c = c + 0x4B;
        }
    }
    else
    {
        c = c - 0x30;
    }
    if ( c < 0 || c > 0xFF )
    {
        c = 0x3F;
    }
    return c;
}

```

After completing the transform process, call to `f_decrypt_c2Url` function to perform decoding to C2. In essence, the function `f_decrypt_c2Url` function will perform `xor` to decrypt, `xor_key` is calculated from the *last 2 bytes* of the transformed `encC2Url` above:

```

f_decrypt_c2Url(
    szencC2Url,
    ((tmp_tbl[1] + (*tmp_tbl << 8)) & 0xF)
    + 0x10 * (tmp_tbl[1] & 0xF0)
    + (((tmp_tbl[1] + (*tmp_tbl << 8)) & 0xF00) << 8)
    + (((tmp_tbl[1] + (*tmp_tbl << 8)) & 0xF000) << 0xC),
    a2);

```

```

len_c2 = system__DynArrayLength_00403E08(encUrl);
i = 0;
if ( len_c2 ≥ 4 )
{
    tmp = tmp_val;
    ptr_encUrl = sub_404050(&encUrl);
    i = len_c2 / 4;
    if ( len_c2 / 4 - 1 ≥ 0 )
    {
        counter = len_c2 / 4;
        do
        {
            *ptr_encUrl ^= tmp;
            ++ptr_encUrl;
            --counter;
        }
        while ( counter );
    }
}
j = 0;
if ( len_c2 ≥ 4 * i + 1 )
{
    counter = len_c2 - 4 * i;
    idx = 4 * i + 1;
    do
    {
        ptr_encUrl_1 = sub_404050(&encUrl);
        ptr_encUrl_1[idx - 1] = *(&tmp_val + j) ^ encUrl[idx - 1];
        j = (j + 1) % 4;
        ++idx;
        --counter;
    }
    while ( counter );
}

```

With all the above information and pseudo-code, I rewrote the code that performs decoding C2 in Python as follows:

```

import numpy as np

#-----
def calc_value(c, val_0x2C):
    """
    tmp = val_0x2C - 0x2B
    if tmp:
        tmp2 = tmp - 2
        if tmp2:
            if (tmp2 == 1):
                c = c + 0x85
            else:
                c = c & 0xFF
        else:
            c = c + 0x4B
    else:
        c = c - 0x30

    if (c < 0 or c > 0xFF):
        c = 0x3F

    return c

#-----
def int_to_bytes(value, length):
    """
    result = []

    for i in range(0, length):
        result.append(value >> (i * 8) & 0xff)

    return result

#-----
def decrypt_c2url(encUrl, xor_tbl):
    """
    c2_url = ""
    dec_c2 = []
    j = 0
    len_c2 = len(encUrl)
    if len_c2 >= 4:
        i = len_c2 / 4
        if (len_c2/4 -1 >= 0):
            counter = len_c2/4
            while counter:
                for k in range(len(xor_tbl)):
                    dec_c2.append(encUrl[j] ^ xor_tbl[j%len(xor_tbl)])
                    j+=1
                counter-=1
    j = 0
    if (len_c2 >= 4 * i +1):
        counter = len_c2 - 4 * i
        idx = 4 * i + 1

```

```

        while counter:
            dec_c2.append(encUrl[idx-1] ^ xor_tbl[j])
            j = (j + 1) % 4
            idx +=1
            counter-=1

    for i in dec_c2:
        c2_url += chr(i)

    return c2_url

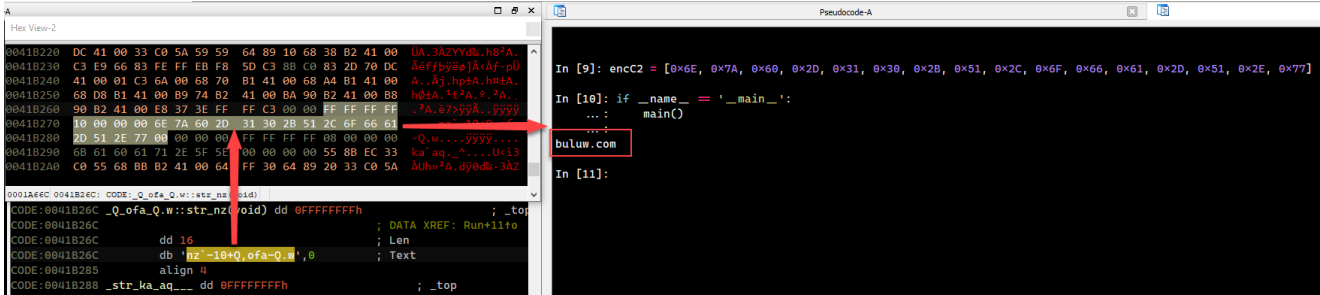
#-----
def main():
    """
    C2_transform = [0] * len(encC2)
    tmp_tbl = []
    val_0x2C = 0x2C
    i = 1
    j = 1

    while j <= len(encC2):
        c = encC2[j -1]
        j+=1
        if ((c - 0x2B) >= 4):
            calced_val = calc_value(c, val_0x2C)
            C2_transform[i-1] = calced_val
            i+=1
        elif (c == val_0x2C):
            if (j > len(encC2)):
                break
            C2_transform[i-1] = encC2[j-1]
            i+=1
            j+=1
        else:
            val_0x2C = c
    C2_transform = np.trim_zeros(C2_transform)
    tmp_tbl = C2_transform[len(C2_transform)-2:len(C2_transform)]
    C2_transform = C2_transform[:len(C2_transform)-2]
    tmp_val = ((tmp_tbl[1] + (tmp_tbl[0] << 8)) & 0xF) + 0x10 * (tmp_tbl[1] & 0xF0) +
    (((tmp_tbl[1] + (tmp_tbl[0] << 8)) & 0xF00) << 8) + (((tmp_tbl[1] + (tmp_tbl[0] <<
    8)) & 0xF000) << 0xC)
    xor_tbl = int_to_bytes(tmp_val, 4)

    print decrypt_c2url(C2_transform, xor_tbl)

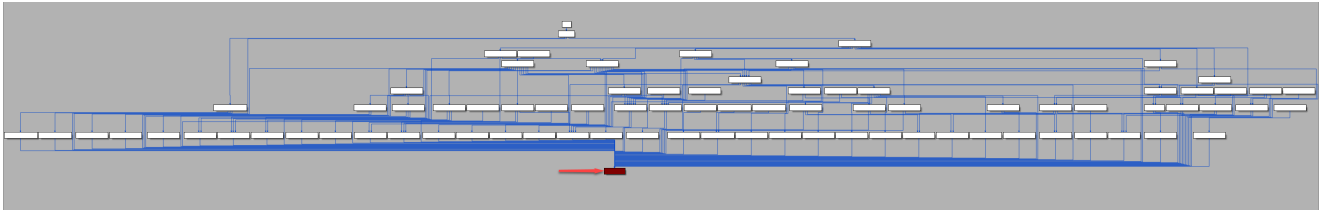
```

Execute the above script and check the results. As the result, this sample will connect to `buluw[.]com` :



Decrypt strings

All strings used by malware are encrypted and only decrypt when needed. Through the analysis of the code will find the function responsible for decoding:



The code at this function is as follows:

```

CODE:00405510 mov     ds:dword_41D724, eax
CODE:00405515 lea   eax, [ebp+pszEncStr]
CODE:00405518 mov     edx, ds:off_41C174
CODE:0040551E call   system_LStrFromPChar_00403DA0 ; Delphi
CODE:0040551E
CODE:00405523 mov     eax, [ebp+pszEncStr]
CODE:00405526 lea   edx, [ebp+pszDecStr]
CODE:00405529 call   f_decrypt_string
    
```

```

CODE:00414AA8
CODE:00414AA8 ;_DWORD *_fastcall f_decrypt_string(char *pszEncStr, char
CODE:00414AA8 f_decrypt_string proc near ; CODE XREF: _Unit5
CODE:00414AA8 ;_Unit5_sub_004054E
CODE:00414AA8 var_4 = dword ptr -4
CODE:00414AA8
CODE:00414AA8 push  ebp
CODE:00414AA9 mov   ebp, esp
CODE:00414AAB push  ecx
CODE:00414AAC push  ebx
CODE:00414AAD mov   ebx, edx
CODE:00414AAF mov   [ebp+var_4], eax
CODE:00414AB2 mov   eax, [ebp+var_4]
CODE:00414AB5 call  system_LStrAddRef_00403FF0
CODE:00414AB5
CODE:00414ABA xor   eax, eax ; eax = 0
CODE:00414ABC push  ebp
CODE:00414ABD push  offset loc_414AED
CODE:00414AC2 push  dword ptr fs:[eax]
CODE:00414AC5 mov   fs:[eax], esp
CODE:00414AC8 mov   ecx, ebx
CODE:00414ACA mov   edx, offset g_calc_tbl
CODE:00414ACF mov   eax, [ebp+var_4]
CODE:00414AD2 call  f_decrypt_str
CODE:00414AD2
CODE:00414AD7 xor   eax, eax
CODE:00414AD9 pop   edx
CODE:00414ADA pop   ecx
CODE:00414ADB pop   ecx
CODE:00414ADC mov   fs:[eax], edx
CODE:00414ADF push  offset loc_414AF4
CODE:00414ADF
CODE:00414AE4 loc_414AE4: ; CODE XREF: f_decrypt
CODE:00414AE4 lea  eax, [ebp+var_4]
CODE:00414AE7 call  system_LStrClr_00403BB0
CODE:00414AE7
CODE:00414AEC retn
    
```


As above picture, `f_decrypt_string` function will take as the argument of the address that contains the pointer to the encrypted string (ex: `off_41C174`). The function responsible for performing the decryption is `f_decrypt_str` , which takes an additional parameter `g_calc_tbl` – *this is table contains 256 elements, used for the calculation*. The code at `f_decrypt_str` function looks like this:

```

--writefsdword(0, &v4);
f_decrypt_str(pszEncStr, g_calc_tbl, pszDecStr);
--writefsdword(0, &v4);
v6 = &loc_414AF4;
return system__LStrClr_00403E08(&pszEncStr);
}

CODE:00414AF8 dd 0FFFFFFFh, 100h
CODE:00414B00 g_calc_tbl db 0, 0D3h, 0BAh, 30h, 007h, 008h, 0C1h, 0C9h, 0EBh, 84h, 0ADh; 0
; DATA XREF: f_decrypt_string+2210
CODE:00414B00 db 88h, 9Ch, 47h, 74h, 08Ch, 6Dh, 43h, 40h, 2Bh, 0AEh, 4Bh, 60h; 11
CODE:00414B00 db 5Dh, 5Ah, 0ECh, 31h, 0F3h, 9Fh, 56h, 1Fh, 1Eh, 93h, 57h, 5Bh; 23
CODE:00414B00 db 0F7h, 0C0h, 2Ch, 7Ch, 0C5h, 69h, 29h, 90h, 13h, 25h, 0E4h, 0A7h; 35
CODE:00414B00 db 99h, 3, 1Ah, 0CAh, 4Ch, 0DAh, 9Ah, 9Eh, 0F4h, 0DBh, 9Dh, 0A8h; 47
CODE:00414B00 db 65h, 0AFh, 0B3h, 0F5h, 51h, 12h, 58h, 7Dh, 11h, 6Ah, 5Ch, 0F0h; 59
CODE:00414B00 db 0Dh, 85h, 7Eh, 0BEh, 15h, 33h, 0B4h, 7Ah, 0FDh, 0ACh, 3Fh, 81h; 71
CODE:00414B00 db 0FFh, 0EDh, 5Eh, 1Dh, 21h, 41h, 0AAh, 18h, 22h, 45h, 17h, 55h; 83
CODE:00414B00 db 96h, 16h, 0A3h, 95h, 0A1h, 0D4h, 3Bh, 0EAh, 0E6h, 0BFh, 28h; 95
CODE:00414B00 db 44h, 89h, 0D9h, 10h, 0E7h, 0ABh, 0ABh, 8Ah, 0CEh, 0D2h, 0Eh; 106
CODE:00414B00 db 0CFh, 8Dh, 0A2h, 0DCh, 0C8h, 4Eh, 0DDh, 26h, 42h, 49h, 0E5h; 117
CODE:00414B00 db 88h, 52h, 0A4h, 0A5h, 9, 40h, 0F3h, 0A0h, 0Bh, 6Bh, 71h, 80h; 128
CODE:00414B00 db 92h, 76h, 0B5h, 0F5h, 2Ah, 0B6h, 8Ch, 20h, 0CDh, 62h, 5Fh, 0C6h; 140
CODE:00414B00 db 0EEh, 2Fh, 35h, 0BBh, 0Ch, 39h, 36h, 1Ch, 3Ah, 63h, 77h, 61h; 152
CODE:00414B00 db 82h, 83h, 87h, 2Eh, 6Fh, 0C2h, 59h, 70h, 50h, 0Ah, 14h, 3Ch; 164
CODE:00414B00 db 0CCh, 0C7h, 0EBh, 3Dh, 4Dh, 8Eh, 91h, 0C3h, 0D0h, 0E9h, 2, 90h; 176
CODE:00414B00 db 0Fh, 0CDh, 4Ah, 68h, 24h, 6, 0A9h, 0B7h, 0EFh, 27h, 97h, 0B1h; 188
CODE:00414B00 db 79h, 7, 32h, 94h, 0B0h, 0BDh, 72h, 75h, 0B8h, 0F1h, 73h, 1; 200
CODE:00414B00 db 64h, 0F6h, 0DEh, 4, 5, 6Ch, 34h, 38h, 78h, 7Bh, 0D6h, 0FAh; 212
CODE:00414B00 db 0F2h, 0FBh, 0FEh, 0FCh, 2Dh, 7Fh, 67h, 6Eh, 0B2h, 0B9h, 66h; 224
CODE:00414B00 db 8, 19h, 54h, 90h, 0C4h, 46h, 0D1h, 0E0h, 18h, 37h, 3Eh, 0D5h; 235
CODE:00414B00 db 23h, 86h, 8Fh, 0DFh, 0E1h, 0E3h, 4Fh, 0E2h, 53h; 247
CODE:00414C00 db 0
CODE:00414C01 db 0
CODE:00414C02 db 0
CODE:00414C03 db 0

```

```

system__LStrAsg_00403C04(pszDecStr, pszEncStr);
if ( *pszDecStr && system__DynArrayLength_00403E08(calc_tbl) == 0x100 )
{
    calc_tbl_val = calc_tbl[**pszDecStr]; // calc_tbl_val = calc_tbl[encStr[i]]
    tmp1 = calc_tbl_val % 7;
    tmp2 = calc_tbl_val % 9;
    strlen = system__DynArrayLength_00403E08(*pszDecStr);
    v6 = strlen < 2;
    strlen -= 2;
    if ( !v6 )
    {
        ++strlen;
        j = 2;
        do
        {
            // calc_tbl[encStr[j-1]] - tmp2 * (j - 1)
            for ( c = calc_tbl[*(*pszDecStr + j - 1)] - tmp2 * (j - 1); c < 0; c = 256 - -c % 0x100 )
            {
                *(sub_404050(pszDecStr) + j++ - 1) = c;
            }
            --strlen;
        } while ( strlen );
    }
    system__LStrCopy_00404050(*pszDecStr, tmp1 + 2, 0x7FFFFFFF, pszDecStr);
}

```

Based on the pseudo-code analyzed above, I rewrote the idapython script that decodes all the strings as follows:

```

import idc
import idutils

dec_routine = 0x00414AA8

calc_tbl = [0x00, 0xD3, 0xBA, 0x30, 0xD7, 0xD8, 0xC1, 0xC9, 0xEB, 0x84, 0xAD, 0x88,
0x9C, 0x47, 0x74, 0xBC, 0x6D, 0x43, 0x40, 0x2B, 0xAE, 0x4B, 0x60, 0x5D, 0x5A, 0xEC,
0x31, 0xF3, 0x9F, 0x56, 0x1F, 0x1E, 0x93, 0x57, 0x5B, 0xF7, 0xC0, 0x2C, 0x7C, 0xC5,
0x69, 0x29, 0x90, 0x13, 0x25, 0xE4, 0xA7, 0x99, 0x03, 0x1A, 0xCA, 0x4C, 0xDA, 0x9A,
0x9E, 0xF4, 0xDB, 0x9D, 0xA0, 0x65, 0xAF, 0xB3, 0xF5, 0x51, 0x12, 0x58, 0x7D, 0x11,
0x6A, 0x5C, 0xF0, 0x0D, 0x85, 0x7E, 0xBE, 0x15, 0x33, 0xB4, 0x7A, 0xFD, 0xAC, 0x3F,
0x81, 0xFF, 0xED, 0x5E, 0x1D, 0x21, 0x41, 0xAA, 0x18, 0x22, 0x45, 0x17, 0x55, 0x96,
0x16, 0xA3, 0x95, 0xA1, 0xD4, 0x3B, 0xEA, 0xE6, 0xBF, 0x28, 0x44, 0x89, 0xD9, 0x10,
0xE7, 0xA8, 0xAB, 0x8A, 0xCE, 0xD2, 0x0E, 0xCF, 0x8D, 0xA2, 0xDC, 0xC8, 0x4E, 0xDD,
0x26, 0x42, 0x49, 0xE5, 0x8B, 0x52, 0xA4, 0xA5, 0x09, 0x48, 0xF8, 0xA6, 0x0B, 0x6B,
0x71, 0x80, 0x92, 0x76, 0xB5, 0xF9, 0x2A, 0xB6, 0x8C, 0x20, 0xCB, 0x62, 0x5F, 0xC6,
0xEE, 0x2F, 0x35, 0xBB, 0x0C, 0x39, 0x36, 0x1C, 0x3A, 0x63, 0x77, 0x61, 0x82, 0x83,
0x87, 0x2E, 0x6F, 0xC2, 0x59, 0x70, 0x50, 0x0A, 0x14, 0x3C, 0xCC, 0xC7, 0xE8, 0x3D,
0x4D, 0x8E, 0x91, 0xC3, 0xD0, 0xE9, 0x02, 0x9B, 0x0F, 0xCD, 0x4A, 0x68, 0x24, 0x06,
0xA9, 0xB7, 0xEF, 0x27, 0x97, 0xB1, 0x79, 0x07, 0x32, 0x94, 0xB0, 0xBD, 0x72, 0x75,
0xB8, 0xF1, 0x73, 0x01, 0x64, 0xF6, 0xDE, 0x04, 0x05, 0x6C, 0x34, 0x38, 0x78, 0x7B,
0xD6, 0xFA, 0xF2, 0xFB, 0xFE, 0xFC, 0x2D, 0x7F, 0x67, 0x6E, 0xB2, 0xB9, 0x66, 0x08,
0x19, 0x54, 0x98, 0xC4, 0x46, 0xD1, 0xE0, 0x1B, 0x37, 0x3E, 0xD5, 0x23, 0x86, 0x8F,
0xDF, 0xE1, 0xE3, 0x4F, 0xE2, 0x53]

```

```

#-----
def get_encrypted_bytes(addr):
    """
    enc_bytes = []
    enc_bytes_addr = idc.get_wide_dword(idc.get_operand_value(addr,1))

    while idc.get_wide_byte(enc_bytes_addr) != 0x0:
        enc_bytes.append(idc.get_wide_byte(enc_bytes_addr))
        enc_bytes_addr += 1

    return enc_bytes

```

```

#-----
def decrypt(enc_str):
    """
    plaint_t = ""
    decStr = [0] * len(enc_str)
    calc_tbl_val = calc_tbl[enc_str[0]]
    tmp1 = calc_tbl_val % 7
    tmp2 = calc_tbl_val % 9
    strLen = len(enc_str) - 1
    j = 2
    for i in range(strLen):
        c = calc_tbl[enc_str[j-1]] - tmp2 * (j - 1)
        decStr[j-1] = c & 0xFF
        j+= 1

    for i in decStr[tmp1+1:]:
        plaint_t+= chr(i)

```

```

return plaint_t

#-----
def decrypt_strings(func_addr):
    """
    for x in idutils.XrefsTo(func_addr, 0):
        org_addr = x.frm
        curr_addr = x.frm
        addr_minus_20 = curr_addr - 20

        while curr_addr >= addr_minus_20:
            curr_addr = idc.prev_head(curr_addr)
            if 'edx' in idc.print_operand(curr_addr, 0) and
idc.get_operand_type(curr_addr, 1) == idc.o_mem:
                enc_bytes = get_encrypted_bytes(curr_addr)
                dec_str = decrypt(enc_bytes)
                print("org_addr: %s, decrypted string: %s" % (hex(org_addr),
dec_str))
                idc.set_cmt(org_addr, dec_str, 0)
            elif 'eax' in idc.print_operand(curr_addr, 0) and
idc.get_operand_type(curr_addr, 1) == idc.o_mem:
                enc_bytes = get_encrypted_bytes(curr_addr)
                dec_str = decrypt(enc_bytes)
                print("org_addr: %s, decrypted string: %s" % (hex(org_addr),
dec_str))
                idc.set_cmt(org_addr, dec_str, 0)

#-----
def main():
    """
    decrypt_strings(dec_routine)

if __name__ == '__main__':
    main()

```

Executing the above script:

Direction	Type	Address	Text
Up	p	sub_4054B8+71	call f_decrypt_string
Up	p	sub_4054B8+A7	call f_decrypt_string
Up	p	sub_4054B8+D6	call f_decrypt_string
Up	p	sub_4054B8+105	call f_decrypt_string
Up	p	sub_405944+5D	call f_decrypt_string
Up	p	sub_405944+83	call f_decrypt_string
Up	p	sub_405944+A9	call f_decrypt_string
Up	p	sub_405944+CF	call f_decrypt_string
Up	p	sub_405944+F4	call f_decrypt_string
Up	p	sub_405944+10D	call f_decrypt_string
Up	p	sub_405944+133	call f_decrypt_string
Up	p	sub_405944+158	call f_decrypt_string
Up	p	sub_405944+171	call f_decrypt_string
Up	p	sub_405944+197	call f_decrypt_string
Up	p	sub_405944+1BD	call f_decrypt_string
Up	p	sub_405944+1E3	call f_decrypt_string
Up	p	sub_405944+208	call f_decrypt_string
Up	p	sub_405944+221	call f_decrypt_string
Up	p	sub_405944+246	call f_decrypt_string
Up	p	sub_405944+25F	call f_decrypt_string
Up	p	sub_405944+284	call f_decrypt_string
Up	p	sub_405944+29D	call f_decrypt_string
Up	p	sub_405944+2C8	call f_decrypt_string
Up	p	sub_405C3C+81	call f_decrypt_string
Up	p	sub_405C3C+A6	call f_decrypt_string
Up	p	sub_405D2C+54	call f_decrypt_string
Up	p	sub_405D2C+78	call f_decrypt_string
Up	p	sub_405D2C+95	call f_decrypt_string
Up	p	sub_405D2C+BA	call f_decrypt_string
Up	p	sub_405D2C+DE	call f_decrypt_string
Up	p	sub_405D2C+102	call f_decrypt_string
Up	p	sub_405D2C+126	call f_decrypt_string
Up	p	sub_405D2C+14A	call f_decrypt_string
Up	p	sub_405D2C+16E	call f_decrypt_string
Up	p	sub_405D2C+192	call f_decrypt_string

Line 1 of 461

Direction	Type	Address	Text
Up	p	sub_4054B8+71	call f_decrypt_string; IPHLPAPI.dll
Up	p	sub_4054B8+A7	call f_decrypt_string; GetIfTable
Up	p	sub_4054B8+D6	call f_decrypt_string; GetAdaptersInfo
Up	p	sub_4054B8+105	call f_decrypt_string; GetNetworkParams
Up	p	sub_405944+5D	call f_decrypt_string; 00155D
Up	p	sub_405944+83	call f_decrypt_string; 0003FF
Up	p	sub_405944+A9	call f_decrypt_string; 0050F2
Up	p	sub_405944+CF	call f_decrypt_string; 000D3A
Up	p	sub_405944+F4	call f_decrypt_string; AZR
Up	p	sub_405944+10D	call f_decrypt_string; 123139
Up	p	sub_405944+133	call f_decrypt_string; 22000A
Up	p	sub_405944+158	call f_decrypt_string; AMZ
Up	p	sub_405944+171	call f_decrypt_string; 000C29
Up	p	sub_405944+197	call f_decrypt_string; 000569
Up	p	sub_405944+1BD	call f_decrypt_string; 001C14
Up	p	sub_405944+1E3	call f_decrypt_string; 005056
Up	p	sub_405944+208	call f_decrypt_string; VMW
Up	p	sub_405944+221	call f_decrypt_string; 001C42
Up	p	sub_405944+246	call f_decrypt_string; PRL
Up	p	sub_405944+25F	call f_decrypt_string; 00163E
Up	p	sub_405944+284	call f_decrypt_string; XEN
Up	p	sub_405944+29D	call f_decrypt_string; 080027
Up	p	sub_405944+2C8	call f_decrypt_string; VBX
Up	p	sub_405C3C+81	call f_decrypt_string; VMW
Up	p	sub_405C3C+A6	call f_decrypt_string; XEN
Up	p	sub_405D2C+54	call f_decrypt_string; 00059A3C7800
Up	p	sub_405D2C+78	call f_decrypt_string; 000000
Up	p	sub_405D2C+95	call f_decrypt_string; 000000
Up	p	sub_405D2C+BA	call f_decrypt_string; 005345000000
Up	p	sub_405D2C+DE	call f_decrypt_string; 00F1D000F1D0
Up	p	sub_405D2C+102	call f_decrypt_string; 00A0C6000000
Up	p	sub_405D2C+126	call f_decrypt_string; 000000000010
Up	p	sub_405D2C+14A	call f_decrypt_string; 000000000030
Up	p	sub_405D2C+16E	call f_decrypt_string; 028037EC0200
Up	p	sub_405D2C+192	call f_decrypt_string; FFFFFFFF

Line 1 of 461

All the strings are decrypted:

org_addr: 0x405529, decrypted string: IPHLPAPI.dll
org_addr: 0x40555f, decrypted string: GetIfTable
org_addr: 0x40558e, decrypted string: GetAdaptersInfo
org_addr: 0x4055bd, decrypted string: GetNetworkParams
org_addr: 0x4059a1, decrypted string: 00155D
org_addr: 0x4059c7, decrypted string: 0003FF
org_addr: 0x4059ed, decrypted string: 0050F2
org_addr: 0x405a13, decrypted string: 000D3A
org_addr: 0x405a38, decrypted string: AZR
org_addr: 0x405a51, decrypted string: 123139
org_addr: 0x405a77, decrypted string: 22000A
org_addr: 0x405a9c, decrypted string: AMZ
org_addr: 0x405ab5, decrypted string: 000C29
org_addr: 0x405adb, decrypted string: 000569
org_addr: 0x405b01, decrypted string: 001C14
org_addr: 0x405b27, decrypted string: 005056
org_addr: 0x405b4c, decrypted string: VMW
org_addr: 0x405b65, decrypted string: 001C42
org_addr: 0x405b8a, decrypted string: PRL
org_addr: 0x405ba3, decrypted string: 00163E
org_addr: 0x405bc8, decrypted string: XEN
org_addr: 0x405be1, decrypted string: 080027
org_addr: 0x405c0c, decrypted string: VBX
org_addr: 0x405cbd, decrypted string: VMW
org_addr: 0x405ce2, decrypted string: XEN
org_addr: 0x405d80, decrypted string: 00059A3C7800
org_addr: 0x405da4, decrypted string: 000000
org_addr: 0x405dc1, decrypted string: 000000
org_addr: 0x405de6, decrypted string: 005345000000
org_addr: 0x405e0a, decrypted string: 00F1D000F1D0
org_addr: 0x405e2e, decrypted string: 00A0C6000000
org_addr: 0x405e52, decrypted string: 000000000010
org_addr: 0x405e76, decrypted string: 000000000030
org_addr: 0x405e9a, decrypted string: 028037EC0200
org_addr: 0x405ebe, decrypted string: FFFFFFFF
org_addr: 0x405edb, decrypted string: FFFFFF
org_addr: 0x406355, decrypted string: ldr1
org_addr: 0x406398, decrypted string: ldr2
org_addr: 0x406444, decrypted string: ShellExecuteA
org_addr: 0x406461, decrypted string: shell32.dll
org_addr: 0x4064d4, decrypted string: ShellExecuteExA
org_addr: 0x4064f1, decrypted string: shell32.dll
org_addr: 0x4069c6, decrypted string: wininet.dll
org_addr: 0x4069fa, decrypted string: http://
org_addr: 0x406a20, decrypted string: https://
org_addr: 0x406a46, decrypted string: InternetOpenA
org_addr: 0x406a72, decrypted string: InternetConnectA
org_addr: 0x406a9e, decrypted string: HttpOpenRequestA
org_addr: 0x406aca, decrypted string: HttpAddRequestHeadersA
org_addr: 0x406af6, decrypted string: HttpSendRequestA
org_addr: 0x406b22, decrypted string: HttpQueryInfoA
org_addr: 0x406b4e, decrypted string: InternetReadFile
org_addr: 0x406b7a, decrypted string: InternetCloseHandle
org_addr: 0x407369, decrypted string: POST
org_addr: 0x4073a0, decrypted string: GET

org_addr: 0x4074bb, decrypted string: Host:
org_addr: 0x4074f8, decrypted string: Accept: /*/
org_addr: 0x407575, decrypted string: Host:
org_addr: 0x4075ca, decrypted string: Accept:
org_addr: 0x409145, decrypted string: kernel32.dll
org_addr: 0x40916e, decrypted string: VirtualAlloc
org_addr: 0x409235, decrypted string: \$SIG
org_addr: 0x409495, decrypted string: kernel32.dll
org_addr: 0x4094be, decrypted string: VirtualAlloc
org_addr: 0x409585, decrypted string: \$SIG
org_addr: 0x409700, decrypted string: Run
org_addr: 0x409947, decrypted string: RunEX
org_addr: 0x409982, decrypted string: https://
org_addr: 0x40999d, decrypted string: http://
org_addr: 0x4099fe, decrypted string: Run
org_addr: 0x409af4, decrypted string: \$UpdateSRV
org_addr: 0x409b21, decrypted string: \$UpdateLTR
org_addr: 0x409c53, decrypted string:
Software\Microsoft\Windows\CurrentVersion\Explorer\Advanced\
org_addr: 0x409c73, decrypted string: Hidden
org_addr: 0x409cd0, decrypted string: HideFileExt
org_addr: 0x409d2a, decrypted string: ShowSuperHidden
org_addr: 0x409e5c, decrypted string: CD_UIDC
org_addr: 0x409e80, decrypted string: CD_ins_guid
org_addr: 0x409ea4, decrypted string: CD_host_guid
org_addr: 0x409ec8, decrypted string: CD_iv
org_addr: 0x409eec, decrypted string: CD_aflt
org_addr: 0x409f16, decrypted string: a
org_addr: 0x409f40, decrypted string: aflt
org_addr: 0x409f64, decrypted string: uidp
org_addr: 0x409f88, decrypted string: IDT
org_addr: 0x409fbd, decrypted string: UID=
org_addr: 0x409ff3, decrypted string: &UID2=
org_addr: 0x40a02c, decrypted string: &UIDC=
org_addr: 0x40a067, decrypted string: &mguid=
org_addr: 0x40a0b0, decrypted string: &uidp=
org_addr: 0x40a0eb, decrypted string: &AppName=
org_addr: 0x40a116, decrypted string: &State=
org_addr: 0x40a151, decrypted string: &ins_guid=
org_addr: 0x40a17c, decrypted string: &host_guid=
org_addr: 0x40a1b7, decrypted string: &iv=
org_addr: 0x40a1e2, decrypted string: &aflt=
org_addr: 0x40a21d, decrypted string: &IDT=
org_addr: 0x40a248, decrypted string: &IRTYP=
org_addr: 0x40a286, decrypted string: &IRVER=
org_addr: 0x40a2c3, decrypted string: &OS=
org_addr: 0x40a308, decrypted string: &SV=
org_addr: 0x40a359, decrypted string: &lptp=1
org_addr: 0x40a38b, decrypted string: &lptp=0
org_addr: 0x40a3c1, decrypted string: &btry=1
org_addr: 0x40a3f3, decrypted string: &btry=0
org_addr: 0x40a426, decrypted string: &VMC=
org_addr: 0x40a46f, decrypted string: ®=
org_addr: 0x40a4a0, decrypted string: CDATA
org_addr: 0x40a4da, decrypted string: SOFTWARE\

org_addr: 0x40a564, decrypted string: &Src=
org_addr: 0x40a5b6, decrypted string: src.dat
org_addr: 0x40a642, decrypted string: &Lang=
org_addr: 0x40a690, decrypted string: &Lang=
org_addr: 0x40a6de, decrypted string: &ADVF=
org_addr: 0x40a734, decrypted string: &FS=
org_addr: 0x40a79f, decrypted string: &sha=
org_addr: 0x40a7f6, decrypted string: &st_dt=
org_addr: 0x40a867, decrypted string: &ParamALL=
org_addr: 0x40a9e9, decrypted string: UnNM
org_addr: 0x40accc, decrypted string: Date:
org_addr: 0x40af23, decrypted string: &Admin=1
org_addr: 0x40af49, decrypted string: &Admin=0
org_addr: 0x40af70, decrypted string: &Idle=
org_addr: 0x40afac, decrypted string: &TDY=
org_addr: 0x40afe3, decrypted string: <DY=
org_addr: 0x40b028, decrypted string: &TDYC=
org_addr: 0x40b0c3, decrypted string: https://
org_addr: 0x40b0e4, decrypted string: http://
org_addr: 0x40b158, decrypted string: Location:
org_addr: 0x40b377, decrypted string: script
org_addr: 0x40b3ac, decrypted string: Flags=
org_addr: 0x40b3ed, decrypted string: CHECK
org_addr: 0x40b589, decrypted string: DFN
org_addr: 0x40b81a, decrypted string: UpdTask.exe
org_addr: 0x40b840, decrypted string: SynHelper.exe
org_addr: 0x40b866, decrypted string: Updane.exe
org_addr: 0x40b892, decrypted string: Sync.exe
org_addr: 0x40b8be, decrypted string: ProductUpdt.exe
org_addr: 0x40b8ea, decrypted string: SyncTask.exe
org_addr: 0x40b913, decrypted string: SyncVersion.exe
org_addr: 0x40b956, decrypted string: .exe
org_addr: 0x40ba7e, decrypted string: https://
org_addr: 0x40ba99, decrypted string: http://
org_addr: 0x40bae0, decrypted string: CR
org_addr: 0x40bb1a, decrypted string: CD
org_addr: 0x40bb94, decrypted string: &uid=
org_addr: 0x40bbc3, decrypted string: &ins_guid=
org_addr: 0x40bbf2, decrypted string: &host_guid=
org_addr: 0x40bc21, decrypted string: &iv=
org_addr: 0x40bc59, decrypted string: &AL=
org_addr: 0x40bc8e, decrypted string: a
org_addr: 0x40bcc4, decrypted string: aflt
org_addr: 0x40bcf4, decrypted string: CD_UIDC
org_addr: 0x40bd27, decrypted string: CD_ins_guid
org_addr: 0x40bd5a, decrypted string: CD_host_guid
org_addr: 0x40bd8d, decrypted string: CD_iv
org_addr: 0x40bdc0, decrypted string: CD_aflt
org_addr: 0x40bdf3, decrypted string: CD_AL
org_addr: 0x40be3b, decrypted string: CD
org_addr: 0x40be8a, decrypted string: Local_Inst_DT
org_addr: 0x40bebb, decrypted string: SNR_FAIL
org_addr: 0x40bef0, decrypted string: URL
org_addr: 0x40bf23, decrypted string: AppName
org_addr: 0x40bf59, decrypted string: uidp

org_addr: 0x40bf9c, decrypted string: UDAT0
org_addr: 0x40bfee, decrypted string: UDAT0
org_addr: 0x40c021, decrypted string: RKL
org_addr: 0x40c06c, decrypted string: RVL
org_addr: 0x40c0ad, decrypted string: RKL
org_addr: 0x40c0e0, decrypted string: RVL
org_addr: 0x40c113, decrypted string: RLM
org_addr: 0x40c158, decrypted string: RLM
org_addr: 0x40c1eb, decrypted string: Inst_DT
org_addr: 0x40c221, decrypted string: IRVER
org_addr: 0x40c26d, decrypted string: IRBVER
org_addr: 0x40c2b5, decrypted string: IRTYP
org_addr: 0x40c2f4, decrypted string: TodayFN
org_addr: 0x40c327, decrypted string: TodayCntFN
org_addr: 0x40c906, decrypted string: SDT
org_addr: 0x40c9c2, decrypted string: Src
org_addr: 0x40c9fc, decrypted string: src.dat
org_addr: 0x40cad5, decrypted string: .del
org_addr: 0x40cbb2, decrypted string: nosct
org_addr: 0x40cd96, decrypted string: noun
org_addr: 0x40cde6, decrypted string: /Uninstall
org_addr: 0x40ce13, decrypted string: DelSelfDir
org_addr: 0x40ce40, decrypted string: /GID=
org_addr: 0x40ce9d, decrypted string: ProdName
org_addr: 0x40cefd, decrypted string: prod.dat
org_addr: 0x40d01e, decrypted string: BkScript
org_addr: 0x40d13f, decrypted string: Flags=
org_addr: 0x40d19c, decrypted string: Install
org_addr: 0x40d52b, decrypted string: UReg
org_addr: 0x40d610, decrypted string: UFile
org_addr: 0x40d6a8, decrypted string: UDAT
org_addr: 0x40d749, decrypted string: UExFile
org_addr: 0x40d8e2, decrypted string: Uninstall
org_addr: 0x40d918, decrypted string: Update Service
org_addr: 0x40d93a, decrypted string: Uninstall completed successfully. Please
restart your computer to clean up files.
org_addr: 0x40d9a3, decrypted string: &APN=
org_addr: 0x40d9cb, decrypted string: &S=Uninstall&IDT=
org_addr: 0x40d9f3, decrypted string: IDT
org_addr: 0x40da3c, decrypted string: &DT=
org_addr: 0x40da67, decrypted string: &IRTYP=
org_addr: 0x40daa5, decrypted string: &IRVER=
org_addr: 0x40dad2, decrypted string: &UID=
org_addr: 0x40db49, decrypted string: BkScript
org_addr: 0x40dba9, decrypted string: config.dat
org_addr: 0x40dd1c, decrypted string: DelSelfDir
org_addr: 0x40dea9, decrypted string: Update Service
org_addr: 0x40decb, decrypted string: Uninstall
org_addr: 0x40defd, decrypted string: update process?
org_addr: 0x40df56, decrypted string: Uninstall must Reboot your computer to delete
files.
org_addr: 0x40df75, decrypted string: Continue with uninstall?
org_addr: 0x40dfec, decrypted string: Uninstall necessario riavviare il computer per
eliminare i file.
org_addr: 0x40e00b, decrypted string: Continuare con la disinstallazione?

org_addr: 0x40e02f, decrypted string: Désinstaller devez redémarrer votre ordinateur pour supprimer les fichiers.
org_addr: 0x40e04e, decrypted string: Continuer la désinstallation?
org_addr: 0x40e072, decrypted string: Uninstall müssen Starten Sie Ihren Computer, um Dateien zu löschen.
org_addr: 0x40e091, decrypted string: Fahren Sie mit dem deinstallieren?
org_addr: 0x40e0b5, decrypted string: Uninstall deve reiniciar o computador para apagar arquivos.
org_addr: 0x40e0d4, decrypted string: Continue com a desinstalação?
org_addr: 0x40e0f5, decrypted string: Desinstalación debe reiniciar el equipo para eliminar archivos.
org_addr: 0x40e114, decrypted string: Continúe con la desinstalación?
org_addr: 0x40e135, decrypted string: Desinstalación debe reiniciar el equipo para eliminar archivos.
org_addr: 0x40e154, decrypted string: Continúe con la desinstalación?
org_addr: 0x40e17b, decrypted string: Update Service
org_addr: 0x40e1e0, decrypted string: noun
org_addr: 0x40e210, decrypted string: GID
org_addr: 0x40e2a9, decrypted string: /DoUninstall
org_addr: 0x40e2d2, decrypted string: /Uninstall
org_addr: 0x40e497, decrypted string: \$UpdateSRV
org_addr: 0x40e4e1, decrypted string: \$UpdateLTR
org_addr: 0x40e66d, decrypted string: script
org_addr: 0x40e6a2, decrypted string: Flags=
org_addr: 0x40e6e3, decrypted string: CHECK
org_addr: 0x40eac4, decrypted string: Update
org_addr: 0x40eadf, decrypted string: Install
org_addr: 0x40eaf8, decrypted string: Flags=
org_addr: 0x40eb50, decrypted string: IDT
org_addr: 0x40ec04, decrypted string: https://
org_addr: 0x40ec1f, decrypted string: http://
org_addr: 0x40ec41, decrypted string: /sanity/
org_addr: 0x40ef16, decrypted string: Ver
org_addr: 0x40ef38, decrypted string: DT:
org_addr: 0x40ef69, decrypted string: Ver:
org_addr: 0x40ef9f, decrypted string: BVer:
org_addr: 0x40efd5, decrypted string: Typ:
org_addr: 0x40f14e, decrypted string: nx
org_addr: 0x41045c, decrypted string: user32.dll
org_addr: 0x41048e, decrypted string: OemToCharA
org_addr: 0x41092a, decrypted string: kernel32.dll
org_addr: 0x410966, decrypted string: GetVersionExA
org_addr: 0x410a6b, decrypted string: ProductName
org_addr: 0x410a93, decrypted string: \Software\Microsoft\Windows NT\CurrentVersion\
org_addr: 0x410e2c, decrypted string: kernel32.dll
org_addr: 0x410e54, decrypted string: GetFileSize
org_addr: 0x411055, decrypted string: <?xml version="1.0" encoding="UTF-16"?>
org_addr: 0x411075, decrypted string: <Task version="1.1"
xmlns="http://schemas.microsoft.com/windows/2004/02/mit/task">
org_addr: 0x4110ae, decrypted string: <RegistrationInfo>
org_addr: 0x4110dd, decrypted string: <Description><#DESC#></Description>
org_addr: 0x41110c, decrypted string: </RegistrationInfo>
org_addr: 0x41113a, decrypted string: <#DESC#>
org_addr: 0x411171, decrypted string: <#TRIGGERS#>
org_addr: 0x4111a0, decrypted string: <Principals>

```

org_addr: 0x4111cf, decrypted string: <Principal id="Author">
org_addr: 0x411202, decrypted string: <UserId>SYSTEM</UserId>
org_addr: 0x411233, decrypted string: <LogonType>InteractiveToken</LogonType>
org_addr: 0x411260, decrypted string: <RunLevel>HighestAvailable</
org_addr: 0x411291, decrypted string: <RunLevel>LeastPrivilege</RunLevel>
org_addr: 0x4112c0, decrypted string: </Principal>
org_addr: 0x4112ef, decrypted string: </Principals>
org_addr: 0x411327, decrypted string: <Settings>
org_addr: 0x411362, decrypted string: <Enabled>>true</Enabled>
org_addr: 0x41139d, decrypted string: <Hidden>>false</Hidden>
org_addr: 0x4113d8, decrypted string: <RunOnlyIfIdle>>false</RunOnlyIfIdle>
org_addr: 0x411413, decrypted string: <WakeToRun>>false</WakeToRun>
org_addr: 0x41144e, decrypted string: <ExecutionTimeLimit>PT0S</ExecutionTimeLimit>
org_addr: 0x411489, decrypted string:
<DisallowStartIfOnBatteries>>false</DisallowStartIfOnBatteries>
org_addr: 0x4114c4, decrypted string:
<StopIfGoingOnBatteries>>false</StopIfGoingOnBatteries>
org_addr: 0x4114ff, decrypted string: <Priority>5</Priority>
org_addr: 0x41153a, decrypted string: </Settings>
org_addr: 0x411575, decrypted string: <Actions Context="Author">
org_addr: 0x4115b0, decrypted string: <Exec>
org_addr: 0x4115eb, decrypted string: <Command><#PROGRAM#></Command>
org_addr: 0x411626, decrypted string: <Arguments><#PARAMS#></Arguments>
org_addr: 0x411661, decrypted string: </Exec>
org_addr: 0x41169c, decrypted string: </Actions>
org_addr: 0x4116d7, decrypted string: </Task>
org_addr: 0x411729, decrypted string: <#PROGRAM#>
org_addr: 0x41176f, decrypted string: <#PARAMS#>
org_addr: 0x411833, decrypted string: <Triggers>
org_addr: 0x411854, decrypted string: <CalendarTrigger>
org_addr: 0x411899, decrypted string: <Repetition>
org_addr: 0x4118ca, decrypted string: <Interval>PT
org_addr: 0x4118f3, decrypted string: M</Interval>
org_addr: 0x411924, decrypted string: <Duration>P1D</Duration>
org_addr: 0x411955, decrypted string: <StopAtDurationEnd>>false</StopAtDurationEnd>
org_addr: 0x411986, decrypted string: </Repetition>
org_addr: 0x4119ab, decrypted string: <StartBoundary><#YEAR#>-<#MONTH#>-
<#DAY#>T<#HOUR#>:<#MI
org_addr: 0x4119dc, decrypted string: <Enabled>>true</Enabled>
org_addr: 0x411a17, decrypted string: <ScheduleByDay>
org_addr: 0x411a4e, decrypted string: <DaysInterval>1</DaysInterval>
org_addr: 0x411a88, decrypted string: </ScheduleByDay>
org_addr: 0x411ac5, decrypted string: </CalendarTrigger>
org_addr: 0x411b02, decrypted string: </Triggers>
org_addr: 0x411c13, decrypted string: <#HOUR#>
org_addr: 0x411c5d, decrypted string: <#MIN#>
org_addr: 0x411ca7, decrypted string: <#SEC#>
org_addr: 0x411cf1, decrypted string: <#YEAR#>
org_addr: 0x411d3b, decrypted string: <#MONTH#>
org_addr: 0x411d85, decrypted string: <#DAY#>
org_addr: 0x411de6, decrypted string: <#TRIGGERS#>
org_addr: 0x411e94, decrypted string: SYSTEM
org_addr: 0x411f7a, decrypted string: Tasks\
org_addr: 0x411fd5, decrypted string: *.job
org_addr: 0x412461, decrypted string: /interactive

```

org_addr: 0x4124af, decrypted string: at.exe
org_addr: 0x412551, decrypted string: Tasks\
org_addr: 0x4125a8, decrypted string: AT
org_addr: 0x4125d3, decrypted string: .job
org_addr: 0x41292f, decrypted string: .xml
org_addr: 0x41296e, decrypted string: /create /F /tn
org_addr: 0x412992, decrypted string: " /xml "
org_addr: 0x4129c7, decrypted string: schtasks.exe
org_addr: 0x412a25, decrypted string: /create /F /tn
org_addr: 0x412a52, decrypted string: " /tr "
org_addr: 0x412a97, decrypted string: /sc DAILY /ST
org_addr: 0x412aea, decrypted string: /RU SYSTEM
org_addr: 0x412b29, decrypted string: /IT
org_addr: 0x412b62, decrypted string: schtasks.exe
org_addr: 0x412d34, decrypted string: /query /xml
org_addr: 0x412d51, decrypted string: schtasks.exe
org_addr: 0x412d9c, decrypted string: </Task>
org_addr: 0x412e04, decrypted string: </Command>
org_addr: 0x412e21, decrypted string: <Command>
org_addr: 0x412ea5, decrypted string: /delete /F /TN "
org_addr: 0x412eda, decrypted string: schtasks.exe
org_addr: 0x412f28, decrypted string: </Task>
org_addr: 0x413074, decrypted string: SHGetSpecialFolderPathW
org_addr: 0x413091, decrypted string: shell32.dll
org_addr: 0x413c5a, decrypted string:
SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall\
org_addr: 0x413d72, decrypted string: SOFTWARE\Microsoft\Windows\CurrentVersion\
org_addr: 0x413d8e, decrypted string: Uninstall
org_addr: 0x413e1b, decrypted string: DisplayIcon
org_addr: 0x413e6a, decrypted string: DisplayName
org_addr: 0x413eb9, decrypted string: UninstallString
org_addr: 0x413f20, decrypted string: Publisher
org_addr: 0x414108, decrypted string: cmd.exe /Q /D /c del "
org_addr: 0x414178, decrypted string:
SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnce
org_addr: 0x4141ad, decrypted string: cmd.exe /Q /D /c del "
org_addr: 0x41421d, decrypted string:
SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnce
org_addr: 0x414636, decrypted string: GetPwrCapabilities
org_addr: 0x414824, decrypted string: open
org_addr: 0x4163d5, decrypted string: 000000000000
org_addr: 0x416449, decrypted string: 000000
org_addr: 0x4164cd, decrypted string: FFFFFFFF
org_addr: 0x4169f5, decrypted string: dir
org_addr: 0x416a16, decrypted string: /S "
org_addr: 0x416a35, decrypted string: *.*
org_addr: 0x416a6b, decrypted string: TIMEOUT
org_addr: 0x416bc2, decrypted string: cmd.exe
org_addr: 0x416bf3, decrypted string: /d /c
org_addr: 0x416c1c, decrypted string: & cmd /d /c del
org_addr: 0x416c71, decrypted string: CreateProcessA
org_addr: 0x416c9a, decrypted string: kernel32.dll
org_addr: 0x416dad, decrypted string: cmd.exe
org_addr: 0x416dde, decrypted string: /d /c
org_addr: 0x416e0d, decrypted string: & cmd /d /c rd /S /Q

org_addr: 0x416e6e, decrypted string: CreateProcessA
org_addr: 0x416e97, decrypted string: kernel32.dll
org_addr: 0x4170a7, decrypted string: MachineGuid
org_addr: 0x4170cf, decrypted string: Software\Microsoft\Cryptography
org_addr: 0x417116, decrypted string: MachineGuid
org_addr: 0x41713e, decrypted string: Software\Microsoft\Cryptography
org_addr: 0x417253, decrypted string: user32.dll
org_addr: 0x41727f, decrypted string: GetLastInputInfo
org_addr: 0x41734c, decrypted string: GetUserDefaultUILanguage
org_addr: 0x417369, decrypted string: kernel32.dll
org_addr: 0x417a58, decrypted string: .ini
org_addr: 0x417ab4, decrypted string: .txt
org_addr: 0x417ad7, decrypted string: .txt
org_addr: 0x4182d5, decrypted string: cmd.exe
org_addr: 0x418376, decrypted string: 1.dat
org_addr: 0x4183a9, decrypted string: 2.dat
org_addr: 0x418448, decrypted string: /d /c
org_addr: 0x4184a3, decrypted string: cmd /d /c copy /B /Y /V
org_addr: 0x4184fd, decrypted string: & cmd /d /c del "
org_addr: 0x418528, decrypted string: " & cmd /d /c del "
org_addr: 0x41857b, decrypted string: CreateProcessA
org_addr: 0x4185a4, decrypted string: kernel32.dll
org_addr: 0x41878f, decrypted string: psapi.dll
org_addr: 0x4187ce, decrypted string: GetModuleFileNameExA
org_addr: 0x4187f6, decrypted string: EnumProcessModules
org_addr: 0x418d30, decrypted string: mnprstghklbcdf
org_addr: 0x418d56, decrypted string: iuaaoeee
org_addr: 0x41a1e7, decrypted string: kernel32.dll
org_addr: 0x41a20d, decrypted string: FreeLibrary
org_addr: 0x41a237, decrypted string: EnterCriticalSection
org_addr: 0x41a261, decrypted string: LeaveCriticalSection
org_addr: 0x41a28b, decrypted string: WaitForSingleObject
org_addr: 0x41a2b5, decrypted string: CloseHandle
org_addr: 0x41a2df, decrypted string: GetExitCodeProcess
org_addr: 0x41a309, decrypted string: GetSystemDirectoryW
org_addr: 0x41a333, decrypted string: GetModuleFileNameW
org_addr: 0x41a35d, decrypted string: DeleteFileA
org_addr: 0x41a387, decrypted string: CreateFileW
org_addr: 0x41a3b1, decrypted string: CreateFileA
org_addr: 0x41a3db, decrypted string: ReadFile
org_addr: 0x41a405, decrypted string: WriteFile
org_addr: 0x41a42f, decrypted string: SetFilePointer
org_addr: 0x41a459, decrypted string: MoveFileW
org_addr: 0x41a48c, decrypted string: MoveFileA
org_addr: 0x41a4c2, decrypted string: FindFirstFileW
org_addr: 0x41a4f8, decrypted string: FindNextFileW
org_addr: 0x41a52e, decrypted string: FindClose
org_addr: 0x41a564, decrypted string: CreateProcessW
org_addr: 0x41a59a, decrypted string: CreateProcessA
org_addr: 0x41a5d0, decrypted string: GetStartupInfoA
org_addr: 0x41a606, decrypted string: CopyFileW
org_addr: 0x41a63c, decrypted string: GetTempPathA
org_addr: 0x41a6cf, decrypted string: PeekNamedPipe
org_addr: 0x41a705, decrypted string: CreatePipe
org_addr: 0x41a73b, decrypted string: GetFileAttributesW

org_addr: 0x41a771, decrypted string: GetShortPathNameA
org_addr: 0x41a7a7, decrypted string: GetShortPathNameW
org_addr: 0x41a7dd, decrypted string: GetComputerNameA
org_addr: 0x41a813, decrypted string: CreateDirectoryA
org_addr: 0x41a849, decrypted string: CreateDirectoryW
org_addr: 0x41a87f, decrypted string: RemoveDirectoryA
org_addr: 0x41a8b5, decrypted string: GetCurrentProcess
org_addr: 0x41a8eb, decrypted string: SetFileTime
org_addr: 0x41a921, decrypted string: GetVolumeInformationA
org_addr: 0x41a957, decrypted string: GetTickCount
org_addr: 0x41a98d, decrypted string: ExitThread
org_addr: 0x41a9c3, decrypted string: CreateThread
org_addr: 0x41a9f9, decrypted string: ResumeThread
org_addr: 0x41aa2f, decrypted string: GetLocalTime
org_addr: 0x41aa65, decrypted string: OpenProcess
org_addr: 0x41aa9b, decrypted string: GetSystemPowerStatus
org_addr: 0x41aad1, decrypted string: GetWindowsDirectoryW
org_addr: 0x41ab07, decrypted string: SetFileAttributesW
org_addr: 0x41ab3d, decrypted string: Sleep
org_addr: 0x41ab73, decrypted string: TerminateProcess
org_addr: 0x41ac02, decrypted string: advapi32.dll
org_addr: 0x41ac28, decrypted string: RegCloseKey
org_addr: 0x41ac52, decrypted string: RegOpenKeyExW
org_addr: 0x41ac7c, decrypted string: RegDeleteKeyW
org_addr: 0x41aca6, decrypted string: RegCreateKeyExW
org_addr: 0x41acd0, decrypted string: RegSetValueExW
org_addr: 0x41acfa, decrypted string: RegQueryValueExW
org_addr: 0x41ad24, decrypted string: RegDeleteValueW
org_addr: 0x41ad4e, decrypted string: RegEnumValueW
org_addr: 0x41ad78, decrypted string: RegEnumKeyW
org_addr: 0x41ae05, decrypted string: user32.dll
org_addr: 0x41ae2b, decrypted string: GetWindowThreadProcessId
org_addr: 0x41ae55, decrypted string: EnumWindows
org_addr: 0x41ae7f, decrypted string: WaitForInputIdle
org_addr: 0x41afaa, decrypted string: kernel32.dll
org_addr: 0x41afeb, decrypted string: LoadLibraryA
org_addr: 0x41b01c, decrypted string: GetProcAddress

End.

m4n0w4r