

Further Updates in LODEINFO Malware - JPCERT/CC Eyes

By 喜野 孝太(Kota Kino)

Published: 2021-02-17 · Archived: 2026-04-05 23:06:55 UTC

February 18, 2021

- [LODEINFO](#)

The functions and evolution of malware LODEINFO have been described in our past articles in [February 2020](#) and [June 2020](#). Yet in 2021, JPCERT/CC continues to observe activities related to this malware. Its functions have been expanding with some new commands implemented or actually used in attacks. This article introduces the details of the updated functions and recent attack trends.

LODEINFO versions

At the time of the last blog update, the latest version of LODEINFO was v.0.3.6, and currently v0.4.8 is being used. Figure 1 shows the transition of LODEINFO versions based on JPCERT/CC's observation.

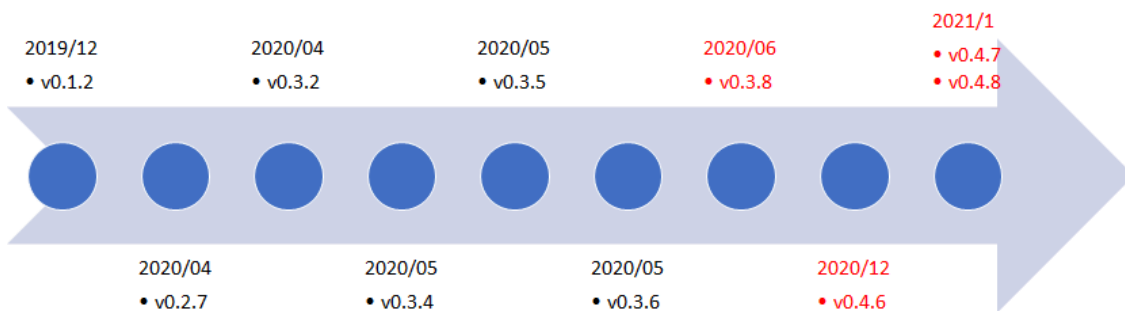


Figure 1 : LODEINFO versions

Decoy document

As we previously explained, LODEINFO infection spreads once a user enables the macro in a Word or Excel file attached to a spear phishing email. In some recent cases, these document files are protected with a password, which is specified in the email body. The Word document convinces the user to enable the macro as in Figure 2. (The statement in the yellow box is roughly translated as follows: In case Word application cannot open the

document properly, you may be able to open it with Word premium mode. To proceed, please click the button in the yellow message bar above.)

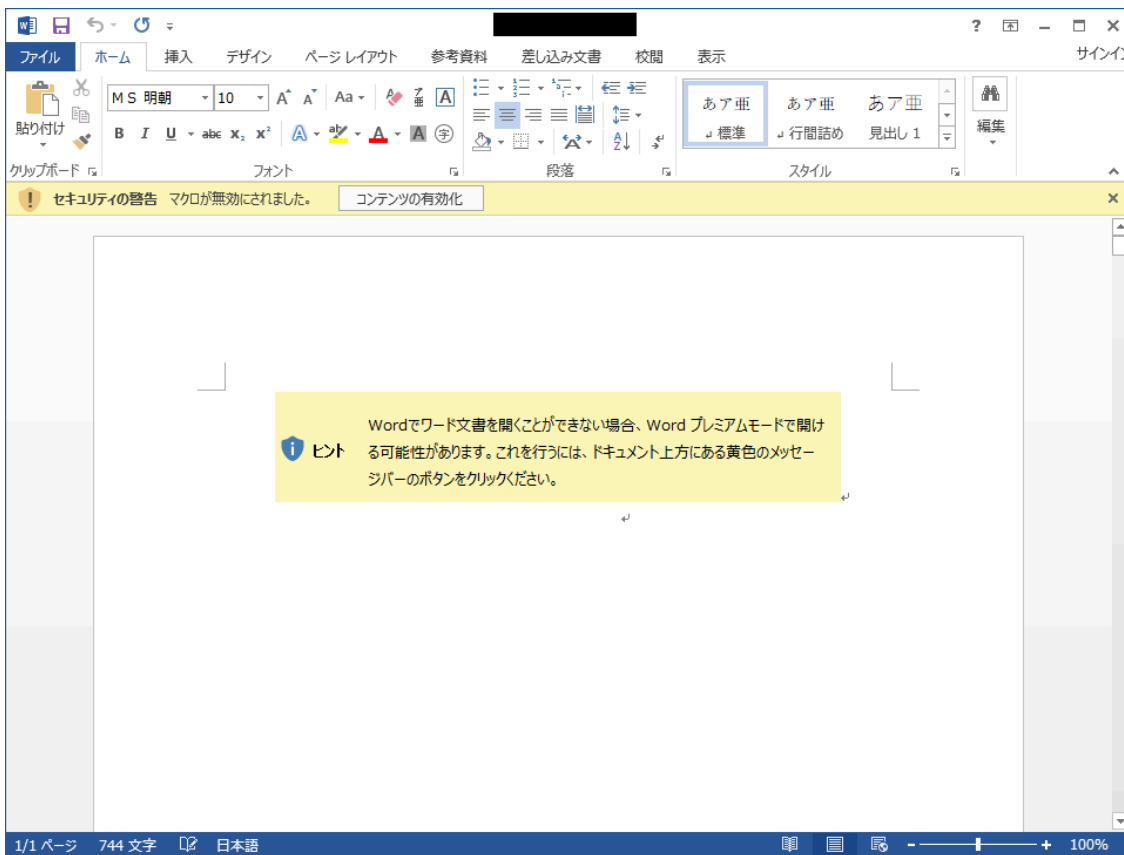


Figure 2 : Word document content sample

The document appears to be empty, however, there are hidden letters in small and white fonts, containing macro configuration values and BASE64-encoded strings of a zip file which stores LODEINFO.

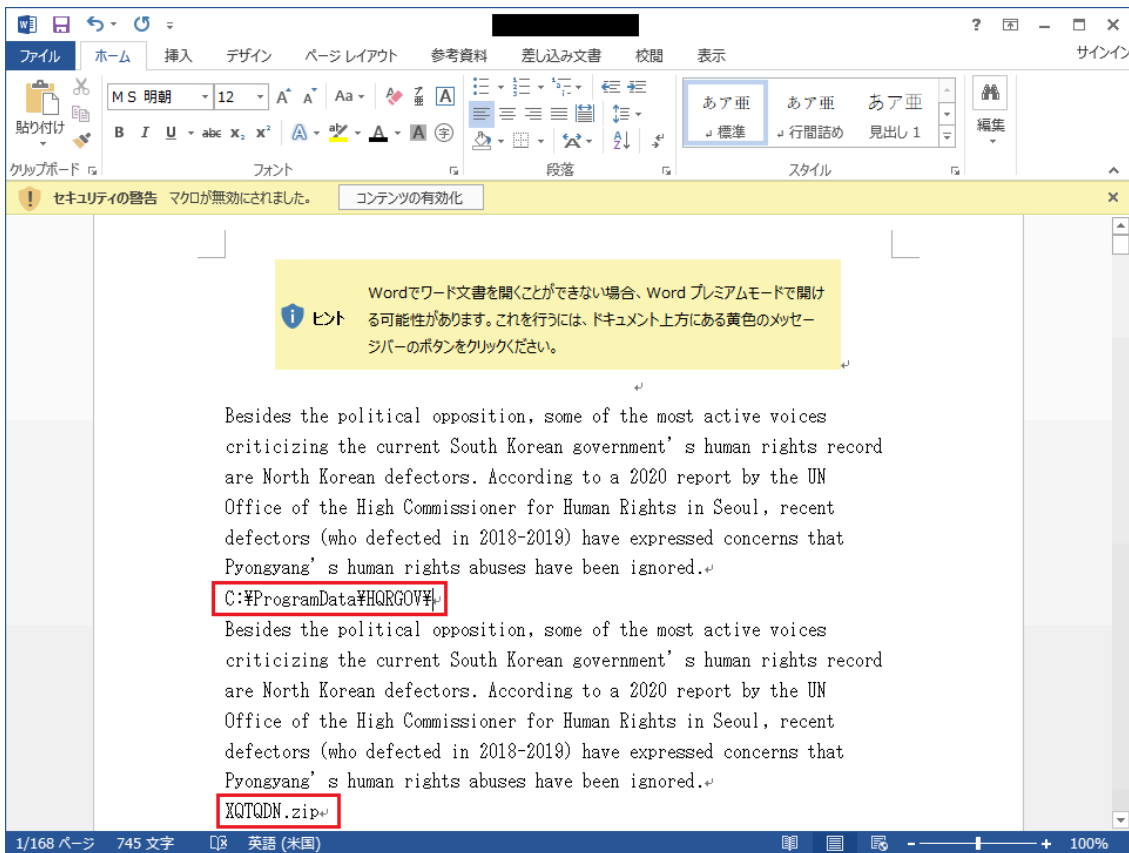


Figure 3 : Word document contents sample (after changing the font)

The macro uses a method called LOLBAS to execute LODEINFO. Below is the command for executing a file created.

```
rundll32.exe advpack.dll,RegisterOCX
```

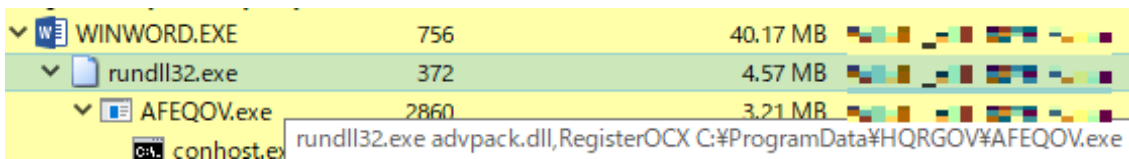


Figure 4 : Process after enabling macro

The code of the macro contained in the documents shows sentences in public articles related to the diplomatic relations between South Korea and Japan or North Korea in the comments.

```

14 Private Sub Form Close()
15 '原告は賠償金確保手段として南朝鮮内にある日本政府の資産の差し押さえ・売却を進めることが可能になった。
16 Dim QDN() As Byte
17 '22日に控訴期限を迎えたが、日本政府が訴訟参加を拒み控訴しなかったため、判決は23日0時に確定した。
18 '22日に控訴期限を迎えたが、日本政府が訴訟参加を拒み控訴しなかったため、判決は23日0時に確定した。
19 If Len(ActiveDocument.Content) < 100 Then GoTo QPO
20 '日本政府に原告1人当たり1億ウォン(約950万円)の損害賠償を命じる判決を出した。

```

Figure 5 : Comments in the macro

New commands

The latest LODEINFO v0.4.8 has the following additional commands compared to v.0.3.6. (See Appendix A for details.)

- ransom (implemented)
- keylog (implemented)
- mv
- cp
- mkdir
- ps
- pkill

The following sections describe some of the new features that are available in the newer versions.

Ransomware function

“ransom” command has been implemented in v.0.3.8 and after. The encryption algorithm is a combination of AES and RSA. The files are first encrypted with an AES key generated for each file. The key is then encrypted with the RSA public key embedded in the malware. After that, the message “WOW! THIS FILE HAS BEEN ENCRYPTED...” is inserted in the beginning of the file.

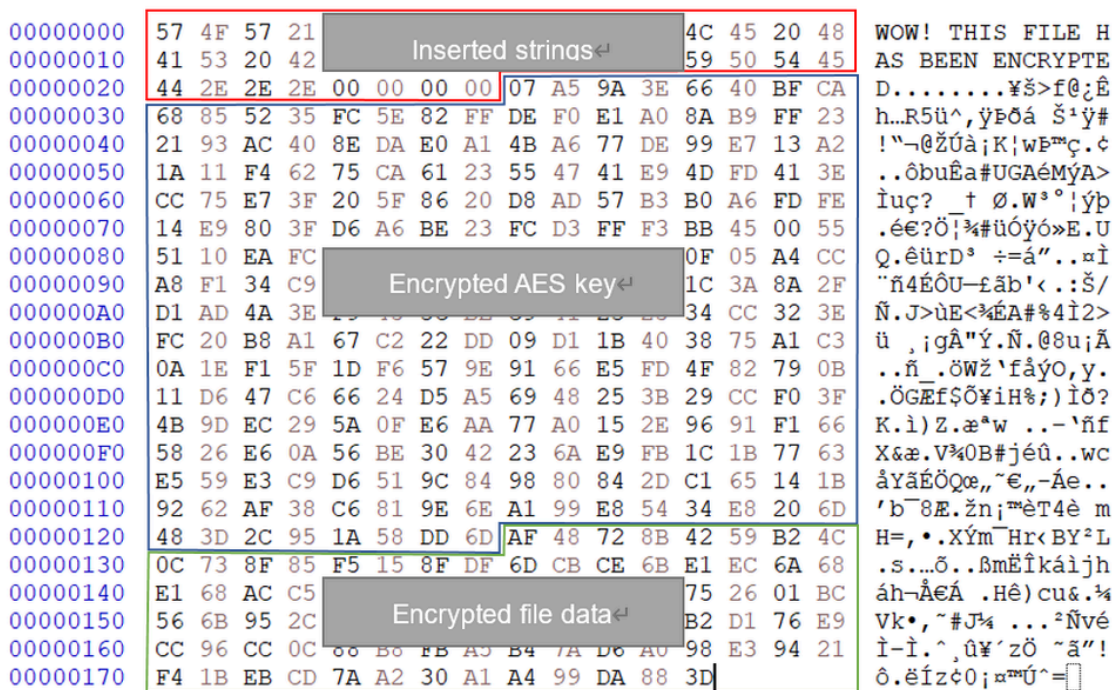


Figure 6 : Structure of the encrypted file

This process makes it difficult to decrypt the files. Files and folders to encrypt can be specified with the ransom command, however, those with file extensions and paths in Figure 7 are excluded.

```
65 strcpy(v2 + 272, "WINDOWS");
66 strcpy(v2 + 280, "Program Files");
67 strcpy(v2 + 296, "Program Files (x86)");
68 strcpy(v2 + 316, "ProgramData");
69 strcpy(v2 + 328, "Recovery");
70 strcpy(v2 + 340, "$Recycle.Bin");
71 strcpy(v2 + 356, ".iso");
72 strcpy(v2 + 364, ".exe");
73 strcpy(v2 + 372, ".dll");
74 strcpy(v2 + 380, "NTUSER.");
75 strcpy(v2 + 388, ".dat");
76 strcpy(v2 + 396, ".bin");
```

Figure 7 : Files excluded from encryption

In case a folder is selected, its path name is checked against the list, but not the individual files inside the folder. Therefore, files including these names listed above are encrypted in this case. Unlike other types of common ransomware, alteration of file extension, creation of ransom notes and/or change of background image do not occur. JPCERT/CC has not yet observed these features in actual attack cases, but they may be used for the purpose of deleting evidence or exfiltrating data.

Keylog function

“keylog” command has been implemented in v.0.4.6 and after. This command checks the following registry value to see if the option is enabled.

```
7  strcpy(reg, "SOFTWARE\\Microsoft\\Keyboard");
8  strcpy(&reg[28], "Enable");
9  enabled = 0;
10 v3 = 4;
11 aa_check_regvalue(this, HKEY_CURRENT_USER, (int)reg, (int)&reg[28], (int)&enabled, &v3);
12 return enabled;
```

Figure 8 : Keylog checks if it is enabled

If it is enabled, a file named “<NetBIOS name>.tmp” is created in %TEMP% folder, and stolen key strings are encoded and stored there. An XOR key is used for encoding, which contains the first 1 byte of the SHA512 value of the device’s NetBIOS name. The following is an example of code to decode the keylog file.

```
import os
import hashlib

name = os.getenv("COMPUTERNAME")
keylog_file = os.getenv("TEMP") + "/" + name + ".tmp"
hash_of_name = hashlib.sha512(name.encode("UTF-8")).hexdigest()
xor_key = int(hash_of_name[0:2], 16)

decode_data = bytes()
with open(keylog_file, "rb") as f:
    for ch in f.read():
        decode_data += (ch ^ xor_key).to_bytes(1, byteorder="big", signed=False)
```

```
print(decode_data.decode('shift_jis'))
```

One of the distinctive features of this function is that it checks if the device’s keyboard layout is set to Japanese according to the following criteria:

- “OverrideKeyboardIdentifier” value in HKLM\SYSTEM\CurrentControlSet\Service\i8042prt\Parameters is set to “PCAT_106KEY”
- “GetKeyboardLayout” function returns “1041”

If the device uses the Japanese keyboard layout, the key strings are converted accordingly. This fact implies that the attackers using LODEINFO malware target Japanese language users.

```
146 if ( v5[offsetof(c, PCAT_106KEY)] && ((_WORD)locale_id == 1041 || !(_WORD)locale_id) )
147 {
148     if ( key >= (unsigned int)'1' && key <= (unsigned int)'9' )
149     {
150         result = a3;
151         strcpy(&v20[4], "!\"#$%&'()");
152         *a3 = *(&v17 + v12);
153         return result;
154     }

```

Figure 9 : Checking keyboard layout

In closing

Attacks using LODEINFO has been continuously observed, and it is considered as a severe threat. We will keep an eye on this activity as it is yet likely to continue.

The hash value of the sample described in the article is listed in Appendix B, together with some newly confirmed C&C servers in Appendix C. Please make sure that none of your devices is communicating with such hosts.

- Kota Kino

(Translated by Yukako Uchida)

Reference

Appendix A New commands

Value	Contents
ransom	Encrypt files
keylog	Control keylogger
mv	Move files
cp	Copy files

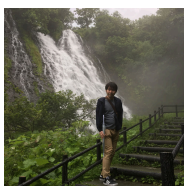
Value	Contents
mkdir	Create directory
ps	List process
pkill	Kill process

Appendix B SHA-256 has value of a sample

- 3fda6fd600b4892bda1d28c1835811a139615db41c99a37747954dccaebff6e (v0.4.6)

Appendix C C&C servers

- www.evonzae.com
- 45.76.216.40
- 103.140.45.71
- 139.180.192.19
- 167.179.84.162
- 167.179.65.11



[喜野 孝太\(Kota Kino\)](#)

Kota Kino is Malware/Forensic Analyst at Incident Response Group, JPCERT/CC since August 2019.

Related articles

```

*key = 0x4271480;
*key[4] = 0x015813C2;
*key[8] = 0x6d72834;
*key[12] = 0x00007069;
Dv[4] = 0x147421;
Zv[1] = 0x4080668;
Zv[2] = 0x30788529;
Zv[3] = 0x0000007;
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_7A00D60 = PMS405/T
v11 = w_ret_argOffset0x350(a1 + 1);
(C2->CryptSetKeyParam)(*v9, 1, 0v, 0); // DV = parameter
v12 = w_ret_argOffset0x350(a1 + 1);
(C2->CryptSetKeyParam)(*v9, 0, 0x0000, 0); // SP_7A00D60 = CBC
return *v4;
}
}

```

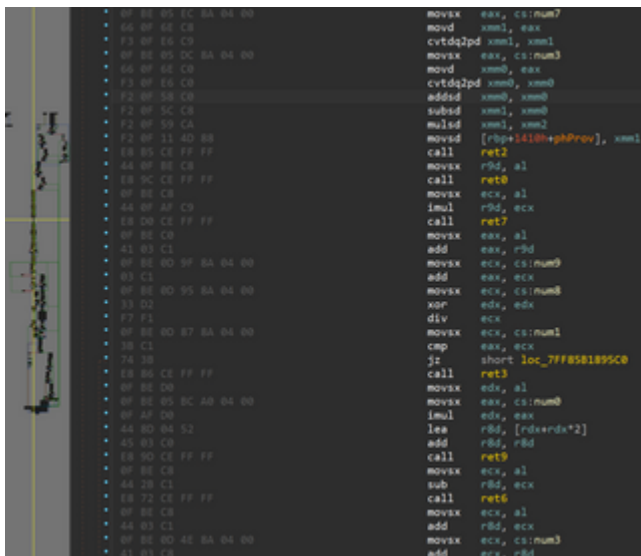
[Update on Attacks by Threat Group APT-C-60](#)

```

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 gQCN381HP2V3JD4
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+cRxoTLmhNoq2U
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+cRxoTLmhNoq2UTWk9o9RodcZtZXsk
bM7Tzk7UZjyapTIjfcq6BwMdsMx6gH40s1B/Swnc3wQxUb0aqEokKorZwmHU3wIDAQAB
-----END PUBLIC KEY-----

```

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

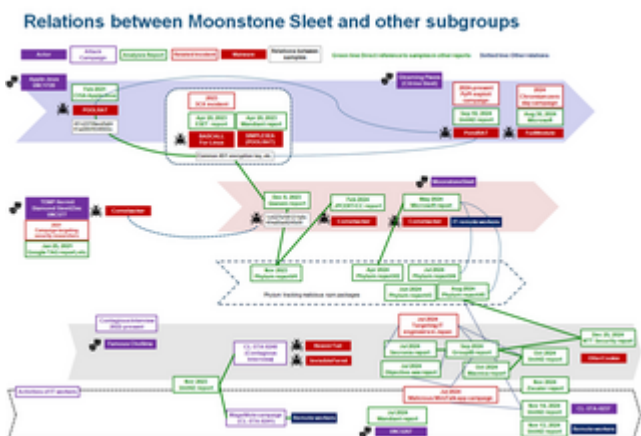


[Malware Identified in Attacks Exploiting Ivanti Connect Secure Vulnerabilities](#)

```
__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;
}
```

[DslodgRAT 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.jpccert.or.jp/en/2021/02/LODEINFO-3.html>