"I will be very surprised if this comes to light"
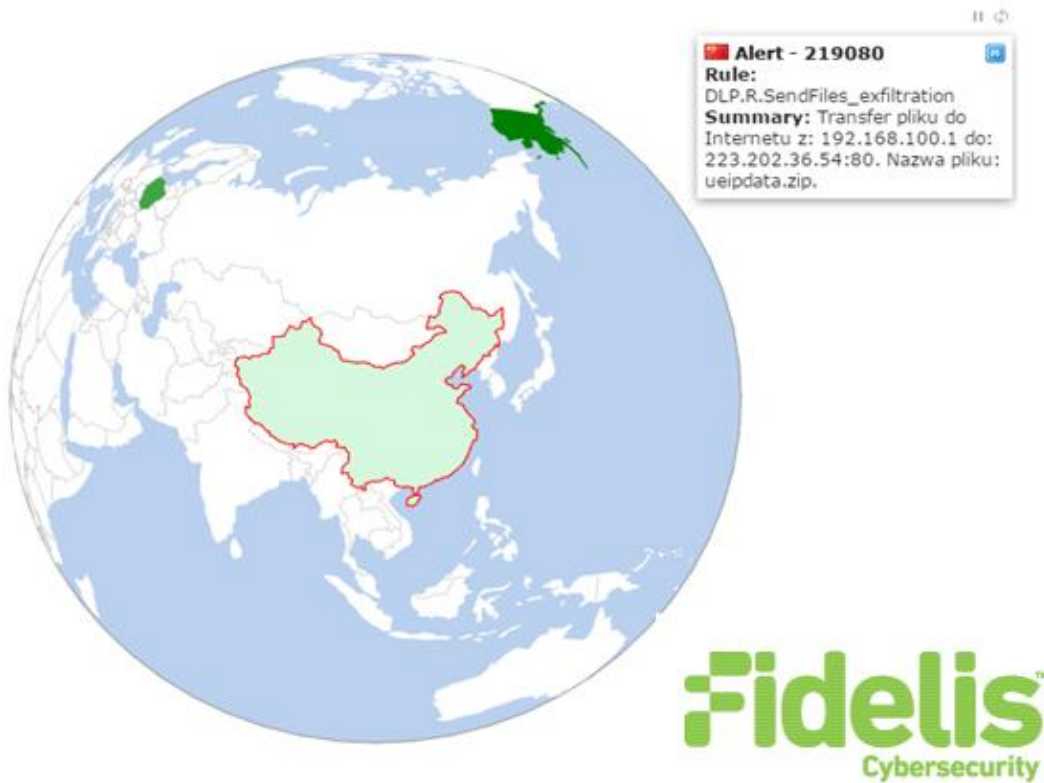
The following text presents the results of the report on the technical analysis conducted by third-line analysts from the Security Operations Center (SOC) run by Exatel S.A. The report was drawn up based on an incident which was identified by the incident response team operating as part of the Exatel's SOC at the end of March, while implementing the Fidelis Network advanced threat detection solution.

Owing to the information obtained during the analysis conducted using the code reverse engineering, the incident response team at the Exatel's SOC managed to reach the functionality which the authors of a fairly popular tool tried to embed in the software in order to send the specific type of content to their servers. What is important, the users of the browser were not aware of the fact of abuse, and even more, they were reassured by the manufacturer that this type of content will not be transferred anywhere without their explicit consent.

Soon after the internal LAN of the organisation was connected to the Fidelis Network platform for the purpose of monitoring, the incident response team from the Exatel's SOC started to register from several to several dozens of alarms per day regarding the violation of the `DLP.sendfiles.exfiltration` rule, which was implemented by the Exatel's SOC into that system for the purpose of monitoring whether documents - in general, broadly understood data - are not sent outside the web by means of the HTTP protocol and the POST method. This is actually how web browsers transmit various data to remote servers, including, for instance, files attached to messages sent using a webmail. It turned out that a small file bearing the name `ueipdata.zip` and weighing several hundred bytes is sent regularly via this protocol to a server in Beijing from 3 computers located in the internal corporate network.

Alert - 219080
Rule:
DLP.R.SendFiles_exfiltration
Summary: Transfer pliku do
Internetu z: 192.168.100.1 do:
223.202.36.54:80. Nazwa pliku:
ueipdata.zip.

The Fidelis Network platform, implemented at Exatel is provided with the web memory module which not only collects the metadata, but also the details of any transmission which violates the security policy in any way. It is able to remember and carry out an in-depth analysis – using the DPI (Deep Packet Inspection) – of both the communication protocols and diverse encoding methods of the files nested in protocol payloads. Owing to this, the security experts from Exatel have full knowledge about any possible incidents at their disposal.

The screenshot from the event analysis console of the Fidelis platform, which is shown above, presents the details regarding a single alarm generated as a consequence of violation of the previously mentioned rule, describing the potential exfiltration of data to the server in China.

The thing that attracted the attention of the specialists from the SOC was the fact that the sent `ueipdata.zip` file contains a single zipped `dat.txt` file, which, in reality, is not a text file, but rather it comprises data with large entropy, being either an output from the random generator, or a result of encryption. Furthermore, the type of the file sent – identified by the `content-type` field of the HTTP protocol – was labelled as `image/pjpeg`, that is,... an image:

However, the most surprising was the phrase which appeared several times in the content of the sent HTTP packet and contained the following text string:
"`IllBeVerySurprisedIfThisTurnsUp`".

```
POST /mx4/enc?keyid=default HTTP/1.1
Host: u.dcs.maxthon.com
Connection: keep-alive
Content-Length: 929
Content-Type: multipart/form-data; boundary=IllBeVerySurprisedIfThisTurnsUp
User-Agent: Mozilla/5.0 (Windows NT 6.1) AppleWebKit/537.36 (KHTML, like Gecko) Maxthon/4.4.5.10
Accept-Encoding: gzip,deflate
Accept-Language: pl-PL

IllBeVerySurprisedIfThisTurnsUp
Content-Disposition: form-data; name="recordid"

1
--IllBeVerySurprisedIfThisTurnsUp
Content-Disposition: form-data; name="trackdata"; filename="ueipdata.zip"
Content-Type: image/pjpeg
Content-Transfer-Encoding: binary

PK........eT.H.|..............dat.txt..._...eU..-..(c.3.iB...>.o...l..q.d.c;b.8v.s
..B7{..BnD.7n..Uq.fb,..T.JW.O2+...|.Jb.neJ.07s)y......A;..q../qt..7...>.7GN..P.......;....<...W.
.N...(a.U....[.5...Vj.8..@...r.6.#..UPK.........eT.H.|......................dat.txtPK....
--IllBeVerySurprisedIfThisTurnsUp
Content-Disposition: form-data; name="submitted"

hello
--IllBeVerySurprisedIfThisTurnsUp--
```

Under these circumstances, the first and probably the most obvious subconscious translation of the phrase was: "**I will be very surprised if this comes to light**." And taking into consideration the fact that April Fools' Day happened to be approaching, the experts from Exatel's SOC initially thought that perhaps one of their colleagues was testing if the newly installed Fidelis Network platform would be able to detect such an incident.

However, the translation of this phrase turned out to be wrong.

Further analysis of the coincidence of the name of the target server in China and the user-agent identifier recorded by Fidelis (the identifier which is usually used by the HTTP client for identification purposes) allowed the experts from Exatel to reach the true offender and learn the proper translation of the phrase.

```
POST /mx4/enc?keyid=default HTTP/1.1
Host: u.dcs.maxthon.com
Connection: keep-alive
Content-Length: 929
Content-Type: multipart/form-data; boundary=IllBeVerySurprisedIfThisTurnsUp
User-Agent: Mozilla/5.0 (Windows NT 6.1) AppleWebKit/537.36 (KHTML, like Gecko) Maxthon/4.4.5.10
Accept-Encoding: gzip,deflate
Accept-Language: pl-PL
```

The offender that stood behind the alarms in the Fidelis platform turned out to be the Maxthon web browser, created and developed by the Chinese.

maxthon
cloud browser

According to the data obtained from the **StatsMonkey** service in the year 2014, it occupies the sixth position with regards to popularity in both, Poland and in China.

Rank	China		Market Share
1	Chrome		49.51
2	IE		28
3	Sogou Explorer		8.67
4	QQ Browser		4.56
5	Firefox		4.34
6	Maxthon		2.59
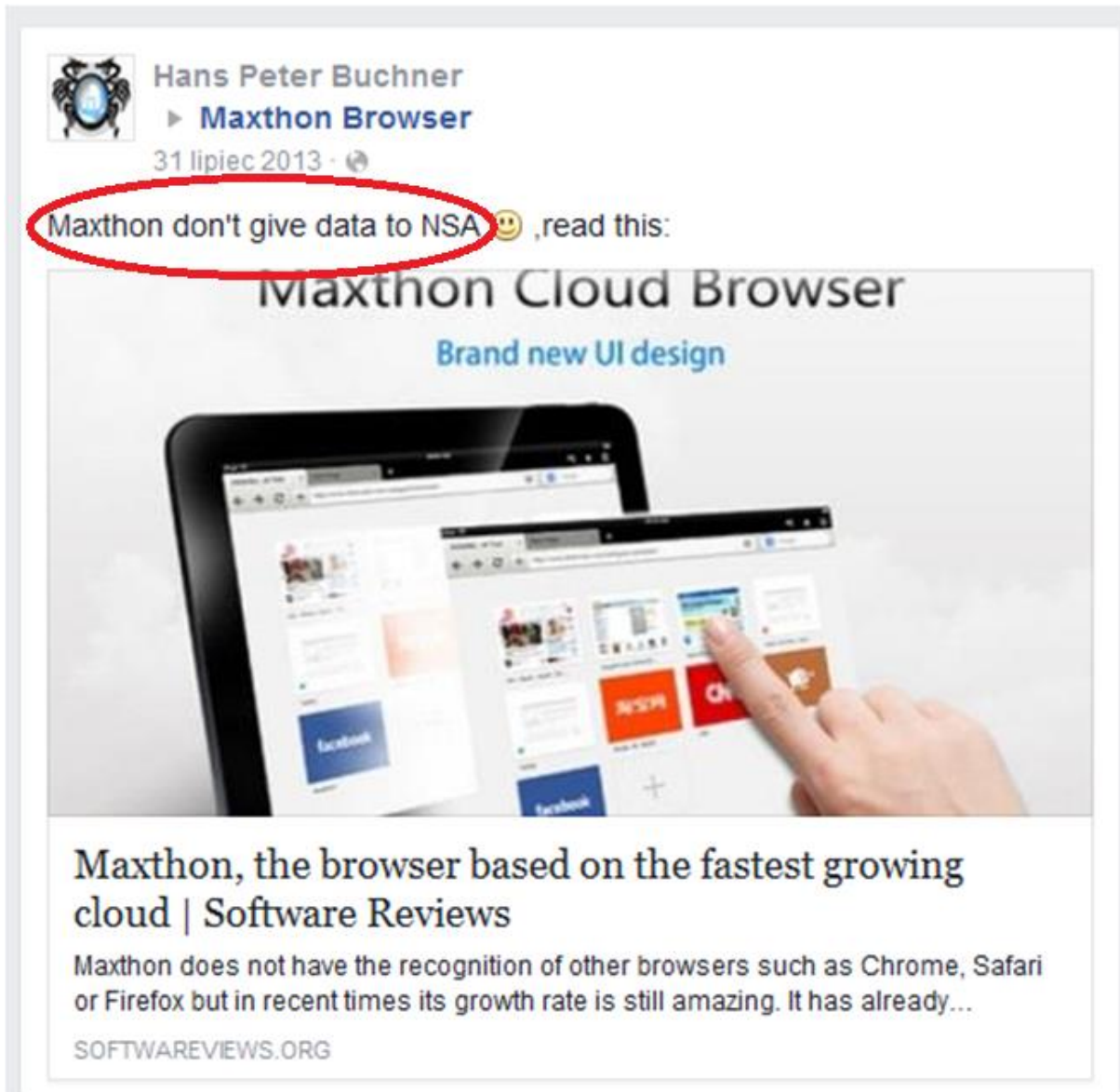
Rank	Poland		Market Share
1	Chrome		47.44
2	Firefox		35.83
3	Opera		7.8
4	IE		7.27
5	Safari		1.02
6	Maxthon		0.32

StatsMonkey, 2014

It was the Maxthon browser installed on computers of three company employees which sent the files that were noticed by the Fidelis platform. What adds irony to the whole matter, is that the creators of the browser inform on their website that it was created with the thought of ensuring security and privacy to the users in the light of scandals related to violation of the privacy by the American National Security Agency (NSA):

http://www.maxthon.com/blog/rightstarups-cloud-browser-with-muscle-security-startup-maxthon-caters-to-html5-users/

As can be read in the opinions on Maxthon, the users are also really fond of this browser because of the fact that… its creators do not share the data with the American National Security Agency (NSA):

Coming back to the previously mentioned text string: "**IllBeVerySurprisedIfThisTurnsUp",** which drew attention of the experts from the SOC, its appearance in the transmission was the result of both a coincidence and a sense of humour of one of the Chinese programmers. He used such a static text string in the code of the C++ library (based on the MFC framework) to separate the files nested in the HTTP transmission - in our case, by instructing the Maxthon server how to decode the ZIP file in the HTTP packet.

The library, which implemented the HTTP protocol client written by himself still in the year 2007:

```
261  void* pBuffer;
262  LPSTR szResponse;
263  CString strResponse;
264  BOOL bSuccess = TRUE;
265
266  CString strDebugMessage;
267
268  if (FALSE == fTrack.Open(_mFilePath, CFile::modeRead | CFile::shareDenyWrite))
269  {
270      AfxMessageBox(_T("Unable to open the file."));
271      return FALSE;
272  }
273
274  int iRecordID = 1;
275  strHTTPBoundary = _T("IllBeVerySurprisedIfThisTurnsUp");
276  strPreFileData = MakePreFileData(strHTTPBoundary, pcmname, iRecordID);
277  strPostFileData = MakePostFileData(strHTTPBoundary);
278
279  AfxMessageBox(strPreFileData);
280  AfxMessageBox(strPostFileData);
281
282  dwTotalRequestLength = strPreFileData.GetLength() + strPostFileData.GetLength() + fTrack.GetLength();
283
284  dwChunkLength = 64 * 1024;
285
286  pBuffer = malloc(dwChunkLength);
287
288  if (NULL == pBuffer)
```

was used by the creators of Maxthon to create a part of the browser functionality, and the true meaning of the aforementioned phrase was in fact: "I will be very surprised if this sequence of characters appears somewhere in the attached file sent by this program".

However we focused on the `ueipdata.zip` file, which leaves the computers on which the browser was installed in strange circumstances (and form). After a short investigation, the abbreviation – UEIP – was successfully deciphered as "User Experience Improvement Program". This is the name of the programme, which – as the creators of the browser claim – is voluntary and anonymous, and its aim is to help the creators in improving the browser by sharing the information about: the hardware on which the browser is installed, the data concerning the operating system, and possible error and crash data reported during the functioning of the browser.

User Experience Improvement Program

In order to understand our user's needs, and deliver better products and services to our user, we invite you to join our User Experience Improvement Program (UEIP).

Participate in this program will not affect your usage of our products and services.

While Participating in this program, you will not be disturbed in any way, such as popups, email, or phone surveys, etc.

Our products and services should work the same whether you participate in this program or not.

Users who choose to participate will send the following data to us:

System Information: Hardware and OS information, etc.

Product Usage: Which button is clicked most and what feature is used most, etc.

Product Settings: Provide information to improve default settings

Error and Crash Data: What error has happened and how many times this error has happened, etc.

The UEIP only collects information about Maxthon products and services. But since some other software might also affect the usage of our products and services (software conflict, security flaw, etc.), we might also collect information about them.

We respect your privacy. For more information please refer to our Privacy Policy

This program is totally anonymous.

No personally identifiable information will be collected. The data we collect is anonymous, and only useful to our product team.

This program is voluntary.

You can opt in/out this Program at any time by checking/unchecking Setup Center » Advanced » User Experience Improvement Program.

If you are uncomfortable with this program, or this program violates the policy of your company or organization, please choose not to participate.

If you have questions or concerns regarding this statement, please Contact Us.

According to its creators, it is possible to resign from the UEIP programme at any time and "the privacy of the user is respected".

The Exatel's security experts decided to check this. They installed the Maxthon browser on their test machine, making sure that they had unchecked the option of participation in the UEIP programme on the startup screen:
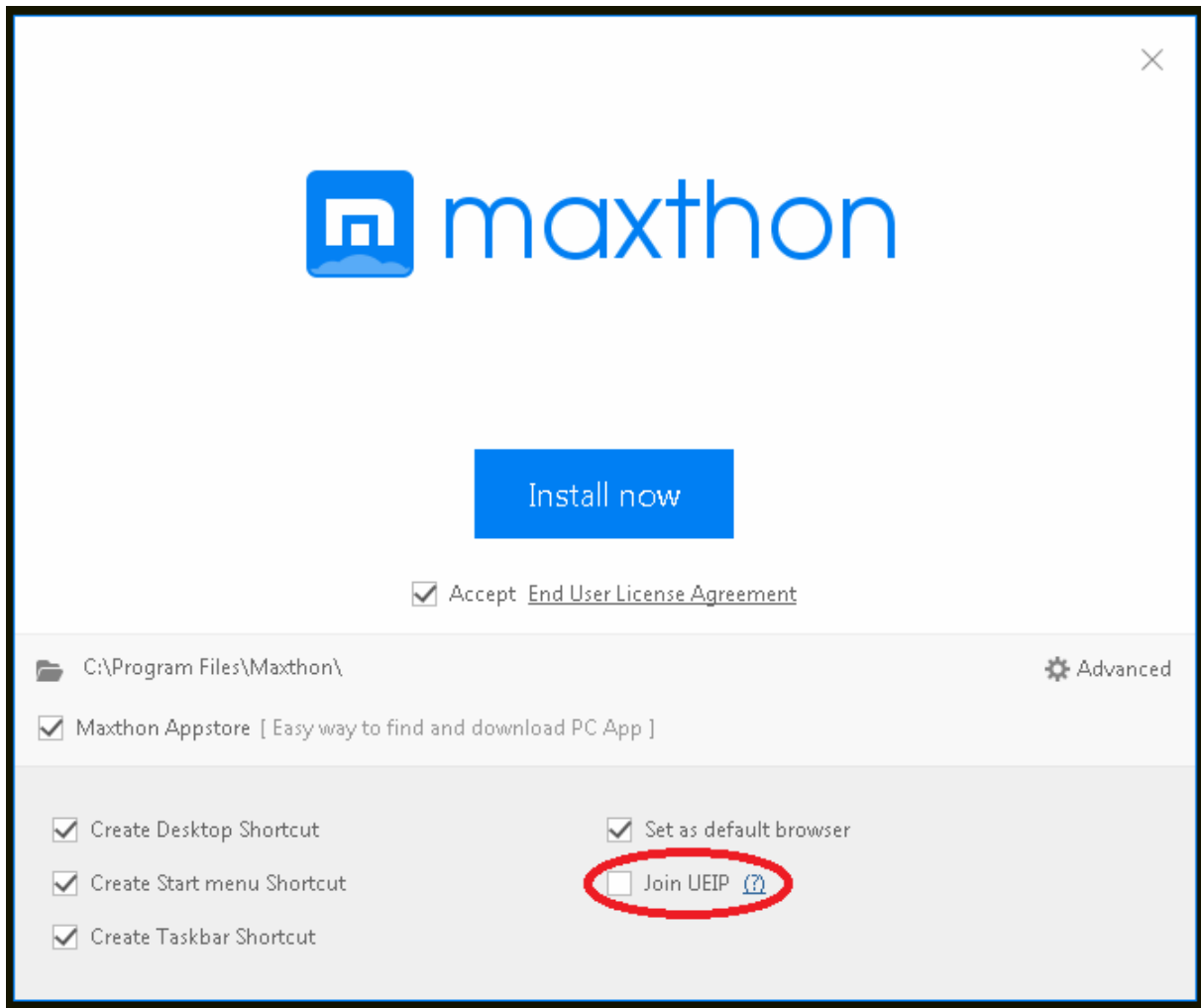
Result? Unfortunately none.

The TCP traffic monitoring on the network interface of the machine during the use of the browser showed regular communication with the same Maxthon server (`u.dcs.maxthon.com`), containing the `ueipdata.zip` file in its payload.

The specialists from the SOC were intrigued by several issues.

Firstly, why is the data of the UEIP programme transmitted to the Maxthon manufacturer despite the explicit lack of consent of the user?

Secondly, why is the `ueipdata.zip` file, which contains an apparently text file `dat.txt` that, in fact, is not a text file, sent further on pretending to be an image file?

And thirdly, what does the browser transfer to the Maxthon servers in the ZIP file?

The security experts from Exatel decided to investigate this matter in more details. For this purpose, they located the part of the code of the main process of the Maxthon browser that executes the data encryption command (the data that after encryption is saved in the `dat.txt` file, zipped into the `ueipdata.zip` file and transmitted to the Maxthon server). As they quickly noticed, the data is encrypted with a symmetric Rijndael (AES) algorithm, using a constant 16-byte key – "`eu3o4[r04cml4eir`", statically compiled in the browser code, without using any kind of obfuscation.

```
.rdata:48016520 ; const WCHAR ClassName
.rdata:48016520 ClassName:                              ; DATA XREF: sub_48010190+40↑o
.rdata:48016520                                         ; sub_48010190+6B↑o
.rdata:48016520                 unicode 0, <Maxthon3Cls_Ueip>,0
.rdata:48016542                 align 4
.rdata:48016544 aEu3o4R04cml4ei db 'eu3o4[r04cml4eir',0 ; DATA XREF: sub_4800FE20+A7↑o
.rdata:48016544                                         ; sub_480105D0+31↑o
.rdata:48016555                 align 4
.rdata:48016558 aMxencode_dll:                          ; DATA XREF: sub_4800FE20+CF↑o
.rdata:48016558                                         ; sub_480105D0+90↑o
.rdata:48016558                 unicode 0, <MxEncode.dll>,0
.rdata:48016572                 align 4
.rdata:48016574 aBrowser_svc_ue:                        ; DATA XREF: sub_480105D0+293↑o
.rdata:48016574                 unicode 0, <browser_svc_ueip_def>,0
.rdata:4801659E                 align 10h
```

The encryption key, along with the plain text data buffer to be encrypted and its size, just before building the new `ueipdata.zip` file and sending it to the Maxthon server, are provided as parameters in execution of `Encode` export function located in the Maxthon's dynamic library `MxEncode.dll` responsible for encryption of the UEIP data transmitted between the browser and the remote Maxthon server, and included in the ZIP files.

```
4800FEC5 loc_4800FEC5:                                  ; CODE XREF: sub_4800FE20+7F↑j
4800FEC5                 push    10h             ; Size
4800FEC7                 push    offset aEu3o4R04cml4ei ; "eu3o4[r04cml4eir"
4800FECC                 lea     ecx, [ebp+var_28] ; int
4800FECF                 mov     [ebp+var_14], 0Fh
4800FED6                 mov     [ebp+var_18], 0
4800FEDD                 mov     byte ptr [ebp+var_28], 0
4800FEE1                 call    sub_480022F0
4800FEE6                 sub     esp, 18h
4800FEE9                 mov     byte ptr [ebp+var_4], 1
4800FEED                 mov     ecx, esp
4800FEEF                 push    offset aMxencode_dll ; "MxEncode.dll"
4800FEF4                 call    sub_48002130
4800FEF9                 call    sub_48010210
4800FEFE                 mov     ecx, eax
```

Analysis has shown also that the MxEncode library was created using the `Crypto++` open source library which can be noticed in symbol table of the Maxthon's PE executable:

```
AVlogic_error@std@@
AVlength_error@std@@
AVout_of_range@std@@
AVtype_info@@
AVbad_exception@std@@
AV?$BlockCipherFinal@$0A@VEnc@Rijndael@CryptoPP
AV?$BlockCipherImpl@URijndael_Info@CryptoPP@@VBlockCipher@2
AVexception@std@@
AV?$FixedBlockSize@$0BA@@CryptoPP@@
AVEnc@Rijndael@CryptoPP@@
AV_Iostream_error_category@std@@
AV_Generic_error_category@std@@
AURijndael_Info@CryptoPP@@
AVNotImplemented@CryptoPP@@
AVAlgorithm@CryptoPP@@
AVDec@Rijndael@CryptoPP@@
AV?$TwoBases@VBlockCipher@CryptoPP@@URijndael_Info@2@@CryptoPP@@
AV?$BlockCipherFinal@$00VDec@Rijndael@CryptoPP@@@CryptoPP@@
AVNameValuePairs@CryptoPP@@
AVNullNameValuePairs@CryptoPP@@
AVInvalidKeyLength@CryptoPP@@
```

Further analysis demonstrated that the MxEncode library is also responsible for encryption and decryption of local Maxthon configuration files on the user's disk, which content is also protected by the manufacturer from the perspective of free viewing.

Taking the above-mentioned issues into consideration, the SOC experts from Exatel decided to monitor the communication between the Maxthon browser and its encryption module `MxEncode.dll`, and to conduct Man-In-The-Middle attack on the Maxthon encryption library.

They took advantage of the fact that in order to transmit the encrypted UEIP data to the server in China – Maxthon browser would first load the `MxEncode.dll` library located in its installation catalogue, transmit the data to be encrypted to the library (including the encryption code) triggering its export `Encode` function, and the library, after the data encryption, would return the encrypted output buffer to the Maxthon process, which would then transmit the already-encrypted data.

Thus, the experts from the SOC created their own DLL library which imitated the original `MxEncode` library, embedding their own two export functions – `Encode` and `Decode` – just like in the original DLL file.

```
#include <stdio.h>
#include <windows.h>
extern "C" {
char mxEncodeDLLFile[] = "MxEncodeOrig.dll";
char encFile[] = "enc.dat";
char decFile[] = "dec.dat";
typedefint (*MxDecodePtr)(char *outBuf, char *inBuf,
     intbufSize, unsigned char *key);
typedefint (*MxEncodePtr)(char *outBuf, char *inBuf,
     intbufSize, unsigned char *key);
__declspec(dllexport) intMxEncode(char *outBuf,
     char *inBuf, intbufSize,
     unsigned char *key)
{
     HMODULE lib = LoadLibrary(mxEncodeDLLFile);
     void *ptr = GetProcAddress(lib, "MxEncode");
     MxEncodePtrMxEncode = (MxEncodePtr) ptr;
     FILE *f=fopen(encFile, "ab");
     fprintf(f, "[ENC.KEY] %s\r\n", key);
     fprintf(f, "[ENC.SIZ] %d\r\n", bufSize);
     fprintf(f, "[ENC.BUF] ");
     fwrite(inBuf, 1, bufSize, f);
     fprintf(f, "\r\n");
     fclose(f);
     return MxEncode(outBuf, inBuf,
          bufSize, key);
}
__declspec(dllexport) intMxDecode(char *outBuf,
     char *inBuf, intbufSize, unsigned char *key)
{
     HMODULE lib = LoadLibrary(mxEncodeDLLFile);
     void *ptr = GetProcAddress(lib, "MxDecode");
     MxDecodePtrMxDecode = (MxDecodePtr) ptr;
     int ret = MxDecode(outBuf, inBuf,
          bufSize, key);
     FILE *f=fopen(decFile, "ab");
```

```
        fprintf(f, "[DEC.KEY] %s\r\n", key);
        fprintf(f, "[DEC.SIZ] %d\r\n", bufSize);
        fprintf(f, "[DEC.BUF] ");
        fwrite(outBuf, 1, bufSize, f);
        fprintf(f, "\r\n");
        fclose(f);
        return ret;
}
BOOL APIENTRY DllMain(HINSTANCE hModule,
        DWORD ul_reason_for_call,
        LPVOID lpReserved)
{
        return TRUE;
}
}
```

In both functions, they inserted the code that saved the data of every single Maxthon browser's encryption request on the disk, into a file indicated by them. After receiving the request for encryption and saving the data on the disk, the library provided by the Exatel experts should load the true Maxthon's encryption library (that was renamed to `MxEncodeOrig.dll`), triggering the relevant encryption function, and return the encrypted data to the Maxthon browser, which will thereafter transmit the data to the Maxthon server.
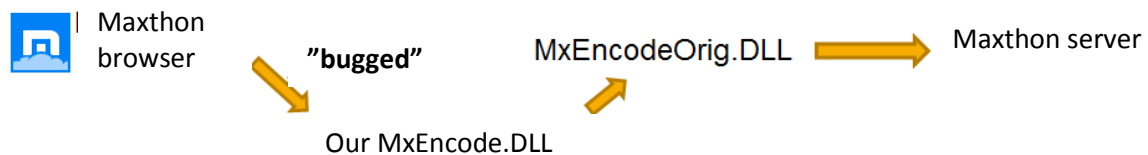
Thus, they allowed Maxthon to let all the data through their library which encryption would be required by the browser before its transmission to China. Using this method, aside of obtaining the entire already-decrypted UEIP transmission to the servers in Beijing, they also let Maxthon decrypt the configuration files, additionally capturing the decryption keys and the data returned by the `Decode` function of the original `MxEncode` library.

Then, the browser was launched to check the effect.

Just after Maxthon was launched, it loaded the MxEncode library and requested encryption of the first data before its transmission, providing the experts from Exatel with the encryption key, which had been obtained during the prior analysis using the reverse engineering.

```
[ENC.KEY] eu3o4[r04cml4eir
[ENC.SIZ] 384
[ENC.BUF] {"uid":"0","l":"en-us","sv":"5.2.3790.Service Pack
2","cv":"4.9.2.1000","pn":"max4web","d":"ADF17F6774C613DDFBB47FF40A96556B27740000
","ueip":0,"screen":"1020x608","net":"","hd":"{\"cpu\":[{\"name\":\"Intel(R)
Core(TM)2 Duo CPU     T7300  @ 2.00GHz\",\"maxclockspeed\":\"2000\"}],\"mem\":\"5
36248320\"}","db":"C:\\Program Files\\Maxthon\\Bin\\Maxthon.exe","a":1,"cmd":""}


[ENC.KEY] eu3o4[r04cml4eir
[ENC.SIZ] 5072
[ENC.BUF] {"uid":"0","l":"en-us","sv":"5.2.3790.Service Pack
2","cv":"4.9.2.1000","pn":"max4web","d":"ADF17F6774C613DDFBB47FF40A96556B27740000
"}
{"pt":"adblock","dt":"users","dr":"1460930014000:1460930014000","m":"blockednum",
"n":"0","o":"","p":"","data":""}
{"pt":"adblock","dt":"users","dr":"1460930014000:1460930014000","m":"adblockenabl
ed","n":"no","o":"","p":"","data":""}
{"pt":"settings","dt":"users","dr":"1460930016000:1460930016000","m":"startwith",
"n":"home","o":"","p":"","data":""}
{"pt":"settings","dt":"users","dr":"1460930016000:1460930016000","m":"homepage","
n":"","o":"","p":"","data":"{\"url\":\"[\\\"http:\\\/\\\/i.maxthon.com\\\/initial
_configuration.htm\\\"]\"}"}
```

As can be seen, the transmission to the server contained: Windows Service Pack version, version of the Maxthon browser, screen resolution (of the virtual machine), type and frequency of the processor and local path in which Maxthon was installed on the disk. The values of configuration variables were also sent, namely: information whether the `adblock` was on or not, the number of already blocked ads and the website address of the home page.

The aforementioned data can be considered consistent with the list of information which transmission is mentioned by the authors in the description of the UEIP programme (leaving aside the fact that the user did not give their consent to join this programme).

Then, the SOC specialists from Exatel focused on capturing the the `MxEncode` library encryption activity at the moment of opening a website. After visiting the first website (it was Onet for that matter), it turned out that the fact of visiting this website was also recorded and reported to the Maxthon server.
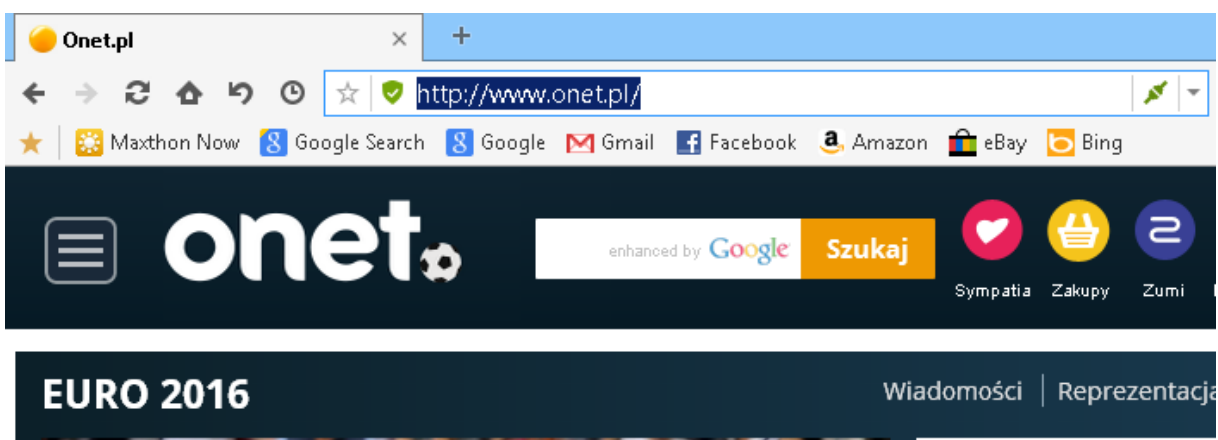
{"uid":"0","l":"en-us","sv":"5.2.3790.Service Pack
2","cv":"4.9.2.1000","pn":"max4web","d":"ADF17F6774C613DDFBB47FF40A96556B27740000
"}
{"pt":"addressField","dt":"ui","dr":"1465664729000:1465664729000","m":"input","n"
:"url","o":"www.onet.pl","p":"unselected","data":""}
{"pt":"addressField","dt":"ui","dr":"1465664729000:1465664729000","m":"input","n"
:"url","o":"http:\/\/www.onet.pl\/","p":"unselected","data":"{\"index\":0,\"lengt
h\":1}"}

The same referred to information about each visited website.

Logging to an e-mail account:

{"uid":"0","l":"en-us","sv":"5.2.3790.Service Pack
2","cv":"4.9.2.1000","pn":"max4web","d":"ADF17F6774C613DDFBB47FF40A96556B27740000
"}
{"pt":"addressField","dt":"ui","dr":"1460803397000:1460803397000","m":"input","n"
:"url","o":"http:\/\/poczta.onet.pl\/","p":"unselected","data":""}
{"pt":"addressField","dt":"ui","dr":"1460803397000:1460803397000","m":"input","n"
:"url","o":"http:\/\/poczta.onet.pl\/","p":"unselected","data":"{\"index\":0,\"le
ngth\":6}"}

Visit on the website of the Polish parliament:

{"uid":"0","l":"en-us","sv":"5.2.3790.Service Pack
2","cv":"4.9.2.1000","pn":"max4web","d":"ADF17F6774C613DDFBB47FF40A96556B27740000
"}
{"pt":"addressField","dt":"ui","dr":"1460803878000:1460803878000","m":"input","n"
:"url","o":"sejm.gov.pl","p":"unselected","data":""}
{"pt":"addressField","dt":"ui","dr":"1460803879000:1460803879000","m":"input","n"
:"url","o":"http:\/\/sejm.gov.pl\/","p":"unselected","data":"{\"index\":0,\"lengt
h\":6}"}

Visit on the Bank's website:

{"pt":"addressField","dt":"ui","dr":"1460803951000:1460803951000","m":"input","n"
:"url","o":"mbank.pl","p":"unselected","data":""}
{"pt":"addressField","dt":"ui","dr":"1460803951000:1460803951000","m":"input","n"
:"url","o":"http:\/\/mbank.pl\/","p":"unselected","data":"{\"index\":0,\"length\"
:6}"}

Thus, all queries by means of the GET method of the HTTP protocol were sent to the Maxthon server.

In short, what does this mean?

The entire user's website browsing history reaches the server of the Maxthon creators in Beijing, including contents of all the entered Google search queries.

While continuing the web surfing using Maxthon with "encryption MITM mode" built in by the Exatel, the experts noticed that also the complete list of software installed on the computer, including precise version numbers, is transferred from their test machine to China server, once in about five transmitted `ueipdata.zip` files.

{"pt":"basicInfo","dt":"content","dr":"1467801694000:1467801694000","m":"software
list","n":"","o":"","p":"","data":"{\"softwarelist\":\"7-Zip%2015.12:15.12,Maxtho
n%20Cloud%20Browser:4.9.3.1000,Microsoft%20Visual%20Studio%202010%20Tools%20for%2
00ffice%20Runtime%20(x86):10.0.50903,Mozilla%20Firefox%2043.0.4%20(x86%20pl):43.0
.4,Mozilla%20Maintenance%20Service:43.0.4,Notepad%2B%2B:6.9,Microsoft%20Office%20
Professional%20Plus%202013:15.0.4569.1506,WinISO:6.4.0.5170,WinRAR%205.31%20(32-b
it):5.31.0,VMware%20Tools:10.0.5.3228253,Python%202.7.11:2.7.11150,Microsoft%20Vi
sual%20C%2B%2B%202008%20Redistributable%20-%20x86%209.0.30729.4148:9.0.30729.4148
,Microsoft%20.NET%20Framework%204.6.1:4.6.01055,Microsoft%20Visual%20Studio%20201
0%20Tools%20for%20Office%20Runtime%20(x86):10.0.50908,Microsoft%20Office%20Profes
sional%20Plus%202013:15.0.4569.1506,Microsoft%20Access%20MUI%20(Polish)%202013:15
.0.4569.1506,Microsoft%20Excel%20MUI%20(Polish)%202013:15.0.4569.1506,Microsoft%2
0PowerPoint%20MUI%20(Polish)%202013:15.0.4569.1506,Microsoft%20Publisher%20MUI%20
(Polish)%202013:15.0.4569.1506,Microsoft%20Outlook%20MUI%20(Polish)%202013:15.0.4
569.1506,Microsoft%20Word%20MUI%20(Polish)%202013:15.0.4569.1506,Microsoft%20Offi
ce%20Korrekturhilfen%202013%20-%20Deutsch:15.0.4569.1506,Microsoft%20Office%20Pro
ofing%20Tools%202013%20-%20English:15.0.4569.1506,Narz%C4%99dzia%20sprawdzaj%C4%8
5ce%20pakietu%20Microsoft%20Office%202013%20%E2%80%94%20polski:15.0.4569.1506,Mic
rosoft%20Office%20Proofing%20(Polish)%202013:15.0.4569.1506,Microsoft%20InfoPath%
20MUI%20(Polish)%202013:15.0.4569.1506,Microsoft%20Office%20Shared%20MUI%20(Polis
h)%202013:15.0.4569.1506,Microsoft%20DCF%20MUI%20(Polish)%202013:15.0.4569.1506,M
icrosoft%20OneNote%20MUI%20(Polish)%202013:15.0.4569.1506,Microsoft%20Groove%20MU
I%20(Polish)%202013:15.0.4569.1506,Microsoft%20Office%20SM%20MUI%20(Polish)%2020
13:15.0.4569.1506,Microsoft%20Office%20SM%20UX%20MUI%20(Polish)%202013:15.0.4569
.1506,Microsoft%20Lync%20MUI%20(Polish)%202013:15.0.4569.1506,Microsoft%20.NET%20
Framework%204.6.1:4.6.01055,Microsoft%20Visual%20C%2B%2B%202008%20Redistributable
%20-%20x86%209.0.30729.6161:9.0.30729.6161,Adobe%20Refresh%20Manager:1.8.0,Adobe%
20Acrobat%20Reader%20DC:15.010.20060,Microsoft%20Visual%20C%2B%2B%202010%20%20x86

In reality, this piece of information transmitted without the user's knowledge between the Browser and the Maxthon server allows to conduct a very precise targeted attack. By gaining knowledge about the user's website browsing preferences, information about his Google searches as well as the complete list of software installed on the user's computer, the attacker only needs an e-mail address to which he will send a message (authenticated by its content), containing an attached armed remote code execution exploit.

Additionally, due to another mistake committed by the browsers' creators, this time an error in the cryptographic architecture, the data which is transmitted without the prior authorisation of the user may be intercepted and decrypted by any potential attacker. All the attacker's needs to accomplish that is to "stand" between the user browser and the Maxthon server to intercept the communication. The intercepted UEIP transmission may be decrypted using AES symmetric keys, obtained from the Maxthon's binary code, after reverse engineering the code.

One way to decrypt the intercepted files in the offline-mode is to use the following Python code, provided by the Fidelis Threat Research Team. The code employs AES128 ECB decription of the `dat.txt` file using the aforementioned encryption key acquired during reverse engineering:

```python
from Crypto.Cipher import AES
data = open('dat.txt','rb').read()
aes = AES.new('eu3o4[r04cml4eir', AES.MODE_ECB)
d = aes.decrypt(data)
print d
```

Thus, the experts from the SOC had good reasons to doubt the security of use of the Maxthon browser, just like its other users who noticed the `ueipdata.zip` files created on their disks.

Jayill

Freshman

●

Members

● 1

4 posts

Posted 14 January

I sincerely hope this is an honest mistake, otherwise it's dishonesty.

You say this:

User Experience Improvement Program

We respect your privacy.

This program is voluntary.

You can opt in/out this Program at any time by checking/unchecking Setup Center » Advanced » User Experience Improvement Program.

Yet, from the initial installation of Maxthon, I chose not to participate, and still I can see that the program is packaging the information to be exported. Currently I do not know if the file is sent before it is deleted (I check on that next), but it is evident that that information is still being gathered.

C:\...\AppData\Roaming\Maxthon3\Temp\ueip\dat.txt

C:\...\AppData\Roaming\Maxthon3\Temp\ueip\ueipdata.zip

Yes I doubled checked that the User Experience Improvement Program is unchecked.

Could an admin clear this up for me?

The Maxthon user asking question on the official Maxthon browser forum received an evasive answer from the authors, that there are... two different types of the UEIP programme.

BugSir007

Assistant Mage

●●●

Vice Admin

Vice Admin

● 375

1,333 posts

Posted 15 January

Hi Jayill,

Please hold on a moment, I'm confirming with my team now.

BugSir007

Assistant Mage

●●●

Vice Admin

Vice Admin

● 375

1,333 posts

Posted 15 January

Hi again,

There are two types of User Experience Improvement Program data:

Data collected when the user choose to participate in UEIP.

data collected regardless of whether users chose to participate in UEIP or not: Here, when users choose not to join UEIP, then we will not collect sensitive data. We will only collect some basic data such as browser start condition and not the data that involves the user's privacy.

Thanks.

On the other hand, the following request of the user, addressed both to the creators and to other users, for help in disclosing the accurate content of the `ueipdata.zip` file, whose creation was noticed by the user on their disk, resulted in sending a message to the user, that all answers to the questions can be found in the description of the privacy policy.

Jayill
Freshman
●

Members
⊕ 1
4 posts

Posted 15 January (edited)

This wasn't mentioned before, or did I miss it in the fine print. Perhaps it might be fair to users to clearly let them know what they are opting in to, especially since that gathered information is encrypted, and we have no way of knowing what's really being uploaded from our computers. Could you please specify what information exactly is being gathered from users who opt out of UEIP, because I noticed a lot of gathering taking place through my browser usage.

Here is a report of little less than 20 minutes in the attachment.

The zip file is created, and deleted, I count 7 times.

ueip recorded.txt

Edited 15 January by Jayill

To sum up the above considerations: the Maxthon browser is not secure.

It allows conducting the targeted attack on a selected user by revealing the browser authors the complete list of exact versions of programms, some of which may be vulnerable, also providing them with user's browsing history and Google searches.

The use of the symmetric cryptography and static encryption keys embedded in the code to obfuscate the transmission of the UEIP data, actually allows to conduct the Man-In-The-Middle attack by any attacker, resulting in decryption of the UEIP data intercepted between the user's browser and the Maxthon server in Beijing.

It is also worth emphasising that the Exatel's SOC got in touch with the creators of the Maxthon browser, sending a detailed technical report, with a request for Maxthon to respond, either in the form of a notice sent to the users about the type of data transmitted from their browsers to the Maxthon servers in Beijing, or in the form of a Maxthon browser software patch which would enable the alarmed users to deactivate effectively the transmission of the UEIP files to their servers. This request was ignored.

The latest version of the browser downloaded from the creators' website (version 4.9.3.1000) was tested by the Exatel's Security Operations Center team and still transmits the UEIP data, without respecting in any way the user's choice regarding the participation in the UEIP programme. Until the delivery of this text for publication, nothing has changed.

Security Operations Center, Exatel