

Malware analysis

Carberp

Ralph Dolmans
Wouter Katz



UNIVERSITY OF AMSTERDAM

Research questions

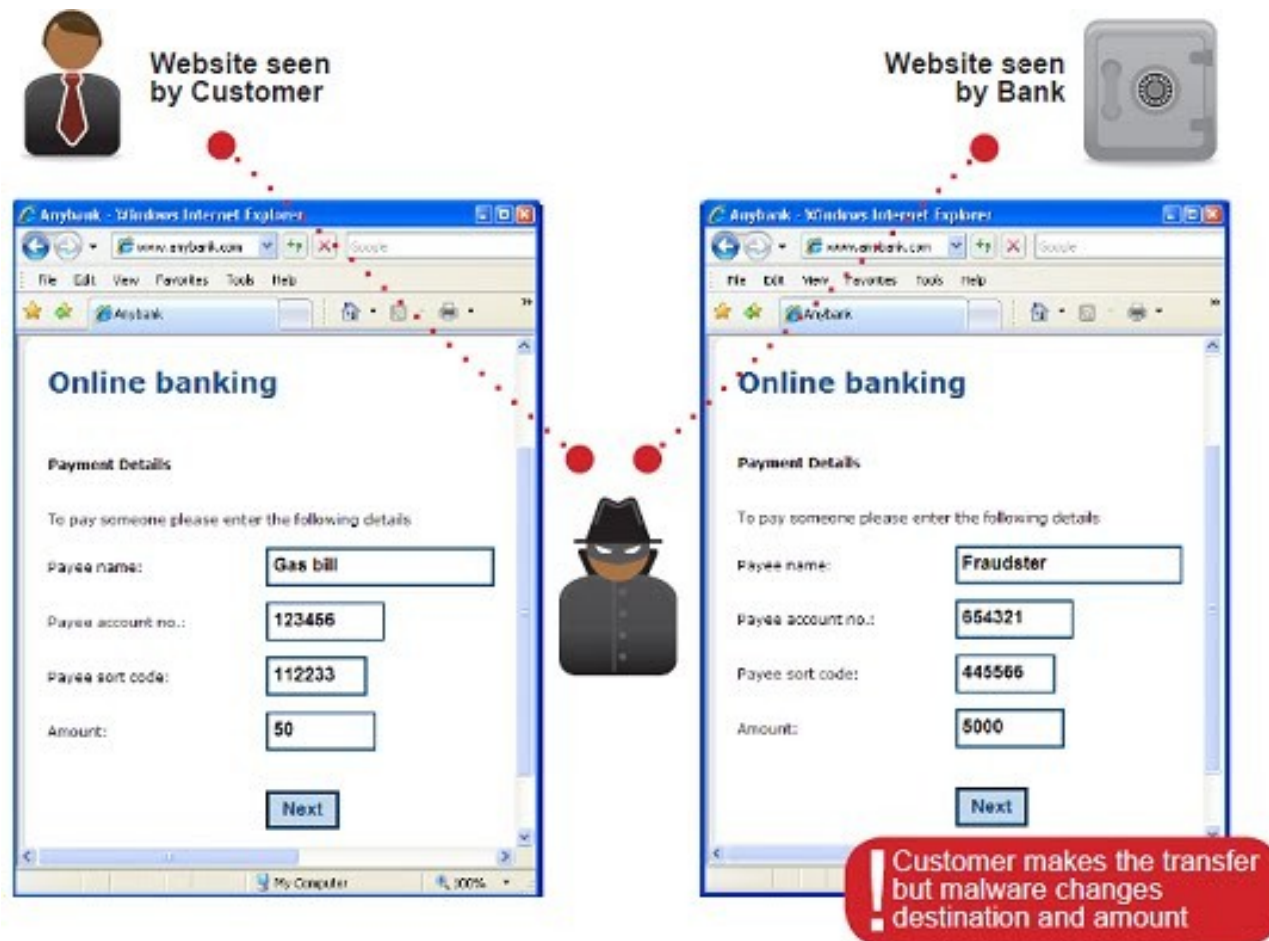
- What kind of anti-forensics techniques are being used by the latest version of Carberp?
- What behavior does the latest version of Carberp show?
 - Installation
 - Run-time
 - C&C

E-banking malware

- Steals your money
- Fake forms (HTML injection), Key logging, browser API hooks, ...
- Big players: Citadel, ZeuS, SpyEye, Carberp

Carberp - General behavior

- MITB for e-banking



Carberp - General behavior

- VNC
- Video recording
- Extra plugins (passw.plugin, stopav.plugin, miniav.plugin)

Installation

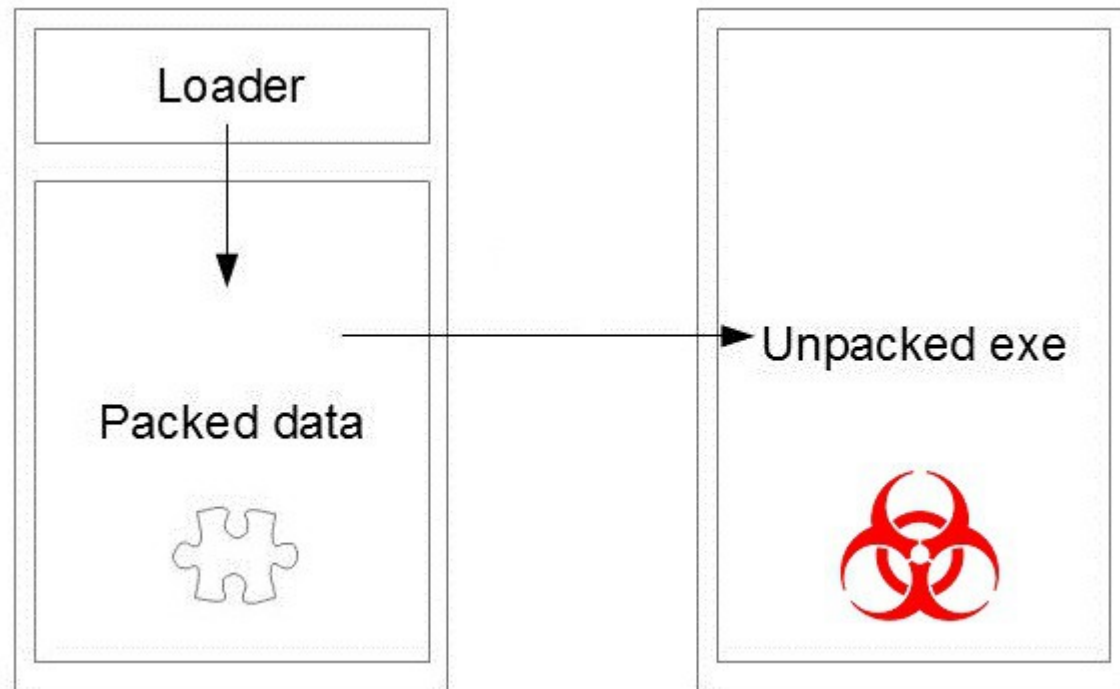
- Startup folder
- Windows service (svchost.exe)
- Contacts C&C server for updates/instructions

Anti-forensics

- Techniques used as countermeasures to forensic analysis
- In our malware sample, data hiding by means of:
 - Packing of the executable
 - Encryption of network traffic
 - Encryption of config files

Executable packing

- Uses small loader to unpack the 'real' executable



Executable packing

- How to obtain unpacked code?
- Run the executable, dump unpacked code from memory.
- Unpacked code contains references to Russian e-banking websites, VNC, password grabber, ...

API hooks

- GMER showed 4 hooks in ntdll.dll:
 - ntdll.dll!NtResumeThread
 - ntdll.dll!NtQueryDirectoryFile
 - ntdll.dll!NtClose
 - ntdll.dll!NtDeviceIoControlFile

API hook behavior

- How to determine what it does?

API hook behavior

- How to determine what it does?

The screenshot shows the OllyDbg interface for notepad.exe. The assembly window displays the following code:

```
7C90D7 . FF12 CALL DWORD PTR DS:[EDX]
7C90D7 . C2 0800 RETN 8
7C90D7 . 90 NOP
7C90D7 . B8 900000 MOV EAX,90
7C90D7 . BA 0003F MOV EDX,7FFE0300
7C90D7 . FF12 CALL DWORD PTR DS:[EDX]
7C90D7 . C2 0400 RETN 4
7C90D7 . 90 NOP
7C90D7 . B8 910000 MOV EAX,91
7C90D7 . BA 68FAC MOV EDX,0C8FA68
7C90D7 . FF12 CALL DWORD PTR DS:[EDX]
7C90D7 . C2 2C00 RETN 2C
7C90D7 . 90 NOP
7C90D7 . B8 920000 MOV EAX,92
7C90D7 . BA 0003F MOV EDX,7FFE0300
7C90D7 . FF12 CALL DWORD PTR DS:[EDX]
7C90D7 . C2 1C00 RETN 1C
7C90D7 . 90 NOP
7C90D7 . B8 930000 MOV EAX,93
7C90D7 . BA 0003F MOV EDX,7FFE0300
7C90D7 . FF12 CALL DWORD PTR DS:[EDX]
```

The registers window shows the following state:

Register	Value	Comment
EAX	0007D37C	
ECX	00000016	
EDX	00000016	
EBX	00000000	
ESP	0007D310	
EBP	0007D618	
ESI	7C90D580	ntdll.NtOpenFile
EDI	00100001	
EIP	7C90D750	ntdll.NtQueryDirectoryFile

The CPU window shows the following assembly code:

```
Imm=00000091 <decimal 145.>
EAX=0007D37C
ntdll.NtQueryDirectoryFile
```

Address	Hex dump	ASCII
010090	00 00 00 00 D4 70 00 01 00 00 00 00	lp @
010090	00 00 00 00 00 00 00 00 64 00 00 00	o t e p a
010090	4E 00 6F 00 74 00 65 00 70 00 61 00	é → \$
010090	FF FF FF FF 90 07 10 00 24 08 10 00	*o n o L o
010090	2A 09 10 00 6E 0A 10 00 4C 0A 10 00	â ð , ð
010090	84 0A 10 00 1C 0B 10 00 2C 0B 10 00	÷ ÷ L J J
010090	F6 0C 10 00 1C 0E 10 00 4A 0E 10 00	! * f * π
010090	8C 0F 10 00 92 0F 10 00 D6 16 10 00	R ð - * T *
010090	9E 17 10 00 AA 0F 10 00 D8 0F 10 00	* * > >
010090	F8 1F 10 00 08 10 10 00 3E 11 10 00	T ! U S ' S
010090	54 13 10 00 9A 15 10 00 60 15 10 00	H _ z _ ä _
010090	48 16 10 00 7A 16 10 00 84 16 10 00	_ f _ e _
010090	2A 16 10 00 C6 16 10 00 EC 16 10 00	. ð D ð U ð
010090	2C 17 10 00 44 17 10 00 56 17 10 00	C ð f ð 4 é @
010090	80 17 10 00 92 17 10 00 34 90 00 01	< é @ é @ L é @
010090	3C 90 00 01 40 90 00 01 4C 90 00 01	D é @ P é @ T é @
010091	44 90 00 01 50 90 00 01 54 90 00 01	\ é @ ' é @ d é @
010091	5C 90 00 01 60 90 00 01 64 90 00 01	l é @ p é @ t é @
010091	6C 90 00 01 70 90 00 01 74 90 00 01	é é @ i é @ é é @
010091	88 90 00 01 8C 90 00 01 90 90 00 01	

The registers window shows the return value and arguments:

Register	Value	Comment
EAX	00000016	RETURN from ntdll.NtQueryDirectoryFile
Arg1	19C	
Arg2	0	
Arg3	0	
Arg4	0	
Arg5	7D37C	
Arg6	7D3AC	
Arg7	268	
Arg8	3	
Arg9	1	
Arg10	7D394	
Arg11	0	

Memory injection

Explorer.exe

Notepad.exe

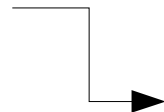
1. Explorer.exe spawns notepad.exe

Memory injection

Explorer.exe

Notepad.exe

1. Explorer.exe spawns notepad.exe



2. Loads ntdll.dll

3. Returns control to parent process

Memory injection

Explorer.exe

Notepad.exe

1. Explorer.exe spawns notepad.exe

2. Loads ntdll.dll

3. Returns control to parent process

4. Calls NtResumeThread:

Map memory region in notepad.exe
Copy malicious code to notepad.exe
Queue malicious code for execution
Call 'real' NtResumeThread

Memory injection

Explorer.exe

Notepad.exe

1. Explorer.exe spawns notepad.exe

2. Loads ntdll.dll

3. Returns control to parent process

4. Calls NtResumeThread:

Map memory region in notepad.exe
Copy malicious code to notepad.exe
Queue malicious code for execution
Call 'real' NtResumeThread

5. Run while being infected.

Hiding files

- ntdll.dll!NtQueryDirectoryFile
- Debugger made clear this hook is for hiding files
- Hidden directory in C:\Documents and Settings\All Users\Application Data

Config file encryption

- mnhs1st32.dat in hidden directory
- Assembly decryption routine found, implemented in python script
- Key found while debugging decryption routine:
HJGsdIk873d

Config file encryption

- XOR each plaintext byte with every key byte
- Before each XOR operation:
 - XOR input = Previous XOR output + (XOR round * plaintext byte position in line)
 - 1st byte: normal
 - 2nd byte input: +1 for round 2, +2 for round 3, ...
 - 3rd byte input: +2 for round 2, +4 for round 3, ...
 -

Config file encryption

- Strings in config file:
 - 696301E9F82608F7EC3CB37D2F44663C
 - 696301E9F82608F7EC3CB37D30046D2DA9
 - 696301E9F82608F7EC3CB37D33046D2DA9
- Plaintext:
 - defeatswirly.net
 - defeatswirly1.net
 - defeatswirly2.net

Network encryption

- Trojan sends HTTP requests to C&C
- All POST-data is encrypted
- Use debugging of the exe to find out how...

Network encryption

- Step through the code to find encryption algorithm
- Encrypted network traffic:
 - 8 byte IV, split into 2 x 4 bytes
 - 1st part IV+base64(RC2(plaintext))+2nd part IV
 - '=' or '==' in base64 always at the end
- RC2 encryption key = CD5ztnj3W1wgSH2M

Network encryption, example

- **HyIF**FI7RmWrgu4r40KdIP4t53IoM3AEGzKJiT
obwr4ex8WAfW59Oh6yNzIcn4RKSWCwT68Ih
PRPMJmEqm0NhqbGFA**IDcu**==
 - IV = HyIFIDcu
- Plaintext:
 - uid=a022A7D5C91DCED15F&av=&md5=a574fc3d
97149bcbf8bdccd5a8a73951

Data theft

- Several Russian banks targeted
- Browser API hooks to check if bank site is accessed
- Send CAB file with screenshot and keylog-file to C&C
 - Network traffic unencrypted

CAB file

The screenshot shows a Windows Internet Explorer browser window displaying the website 'Финам.ру' (Finam.ru). The browser's address bar shows 'http://www.finam.ru/'. The website content includes a red button labeled 'Получить' (Get) and a logo for 'ФИНАМ'. Overlaid on the browser are several windows:

- screenshot_post.cab [read only]**: A file explorer window showing a folder named 'ScreenShots' (105.2 kB) and two text files: 'LogData.txt' (667 bytes) and 'URL.txt' (16 bytes), both modified on 01 February 2013, 14:39.
- URL.txt**: A text editor window containing the URL 'http://finam.ru/'.
- LogData.txt**: A text editor window containing the text 'k<Click><Click><Click><Click><Click><Click>' and '<Click><Click><Click>online.payment.ru/'.

The background website also features a line chart for 'RTS Fut' with a price range from 1528,5 to 1555,8, and a section for 'Курсы валют' (Currency Rates) showing rates for USD, EUR, and RUB.

Conclusions

- Hiding files
- Memory injection
- Encryption
- Tries to steal information

Questions?