Technical Trends in Recent Targeted Attacks

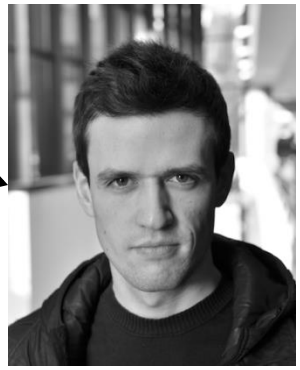
Gábor Pék
Laboratory of Cryptography and System Security (CrySyS)
Budapest University of Technology and Economics
www.crysys.hu

joint work with **Boldizsár Bencsáth**, **Levente Buttyán**, and **Márk Félegyházi**

HOW DID I GET HERE?

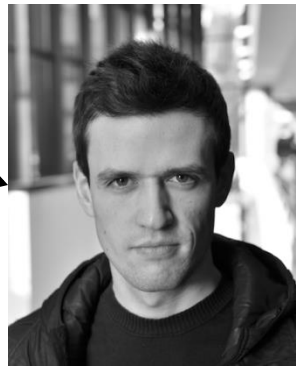
1

2

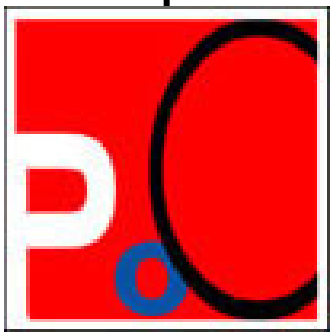


1

*Seems to be a correct invitation, but can be a perfect **spear phishing** attack.*



1



2

3

1

1

2

3

4

1

2

3

4

5

*”It is not a spam ☺. The email is correct. Vangelis is a great host as far as I am concerned. I've spoken at POC2012 and **confirm** that it is one of the **nicest conferences and hosts around**.”*

1

2

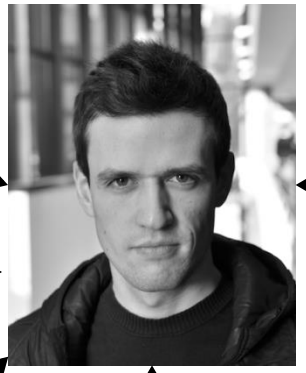
3

1

5

4

6



2

3

7

1

5

4

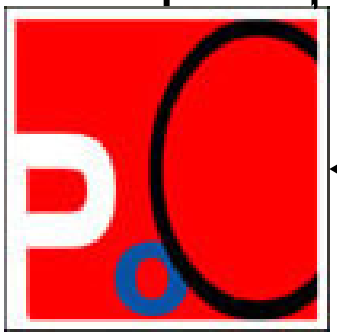
6

” I got a mail from Andrei. He was a good speaker of POC last year. ☺”

1

5

4



6

WHY CRYSYS LAB?

Stuxnet (June 2010)

"the Most Menacing Malware in History"

(Kim Zetter, Wired)

Targeted the Natanz ***nuclear enrichment plant*** in Iran

Modified PLCs

(Programmable Logic Controllers)

Destroyed hundreds of uranium centrifuges

Our contributions to Duqu case (Aug 2011 -

Discovery, naming, and first analysis of ***Duqu***

(large similarities with Stuxnet)

Identification of the dropper

(MS Word document with a 0-day Windows kernel exploit)

Development of the ***Duqu Detector Toolkit***

(opensource, focuses on heuristic anomaly detection)

The sKyWIper/Flame case (May 2012)

CrySyS Lab participated in an international collaboration to investigate

sKyWIper/Flame

(corresponding samples: Gauss, MiniFlame/SPE)

~ ***10000*** victims

Middle East (Iran, Sudan)

Dates of the sKyWIper/Flame case

National CERT of IRAN

analyses a ”malware” called Flamer

(May 27, 2012)

CrySyS Lab ***releases*** an initial report on a malware called sKyWIper

(May 28, 2012)

Kaspersky Lab ***releases*** details about their work on Flame

(May 28, 2012)

Miniduke

FireEye found a document with
0-day PDF exploit
(Feb 12, 2013)

PDFs with ***same 0-days***, but different malware module were also found

High-profile targets of MiniDuke

We ***expected*** the ***use*** of these ***against high-profile*** targets

We found ~***60 victim*** IP addresses

↓

Many ***high profile targets*** in governments and organizations

(including ***NATO***)

TeamSpy

Hungarian National Security Authority asks for our help on an already identified attack

(March 2013)

We analyzed ***new samples*** and ***investigated*** various C2 servers

↓

We obtained victim lists

TeamSpy

Multiple waves of attack campaign

(from 2004 or earlier)

Use of two main malware technologies

standard botnet client Teamviewer DLL

TeamSpy

Likely use of ***watering hole*** attack
e.g.,
based on Java exploit

Some tools ***were*** already ***known*** by
A/Vs

BUT, the whole ***story*** was ***never identified***

Motivation for the presentation

2006: Shady Rat

2007: Sinowal Trojan and variants

2008: Backshot Yankee

2009: GhostNet

2010: Stuxnet, Hydraq (Aurora), Quackbot

2011: ***Duqu***, Nitro attacks (Poison Ivy)

2012: ***Flame***, ***MiniFlame*** (SPE), ***Gauss***

2013: Red October, ***MiniDuke***, ***Teamspy***, Korplug, Kisuky, Hormesu, Janicab, NetTraveler, Icefog, Rarstone, Gh0st RAT, Korhigh, Opsiness, Kimsuky

Aspects of technical trends

Exploitation

Attacker techniques

Reconnaissance

Under the Radar

Encryption

Compression

Modularity and code reuse

C2 communication and infrastructure

Code signing

High profile targets

Stealthiness and persistency

Goal of Attackers

EXPLOITATION

Trend: Use of known and unknown exploits

Professional attackers use sophisticated ***zero-day*** exploits

BUT, other groups prefer ***known exploits in the first place***

Use of known exploits

NetTraveler (CVE-2012-0158, CVE-2013-2465)

Terminator RAT (CVE-2012-0158)

Rarstone (CVE-2012-0159)

EvilGrab (CVE-2012-0158)

BLYPT (CVE-2013-1493)

IceFog (CVE-2012-0158, CVE-2012-1856, CVE-2013-0422, CVE-2012-1723)

Janicab (CVE-2012-1723)

MiniDuke (CVE-2013-0640)

Appearance of zero-days

Stuxnet

(CVE-2010-2772)

Duqu

(CVE-2011-3402) + 9 corresponding CVEs

Zero-day platform: The ***Elderwood*** platform

Adobe Flash Player (CVE-2012-0779,CVE-2012-1535)

Microsoft IE (CVE-2012-1875)

Microsoft XML Core (CVE-2012-1889)

Example: Duqu dropper

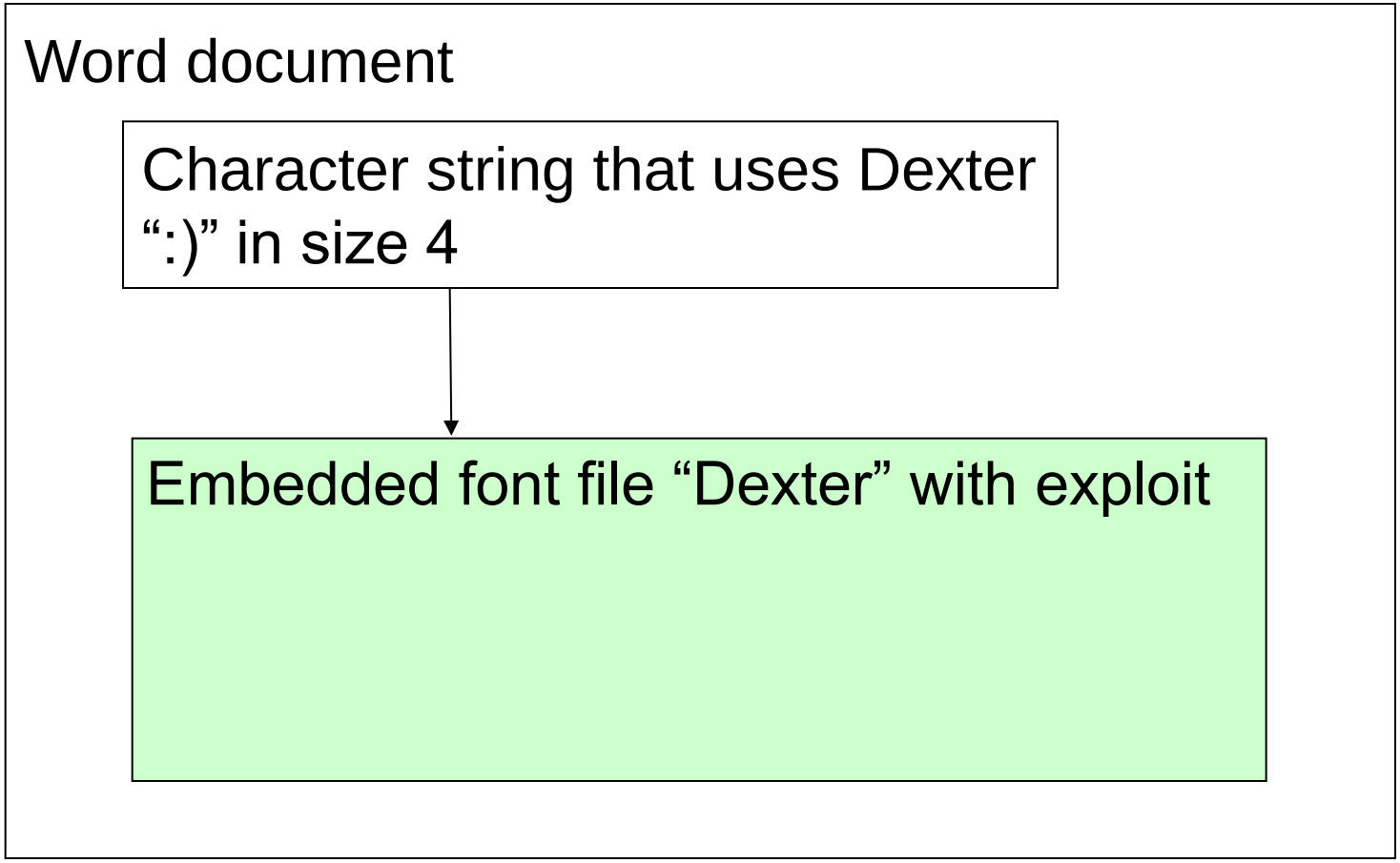
Duqu dropper was a ***.doc*** file with

embedded font (Dexter)

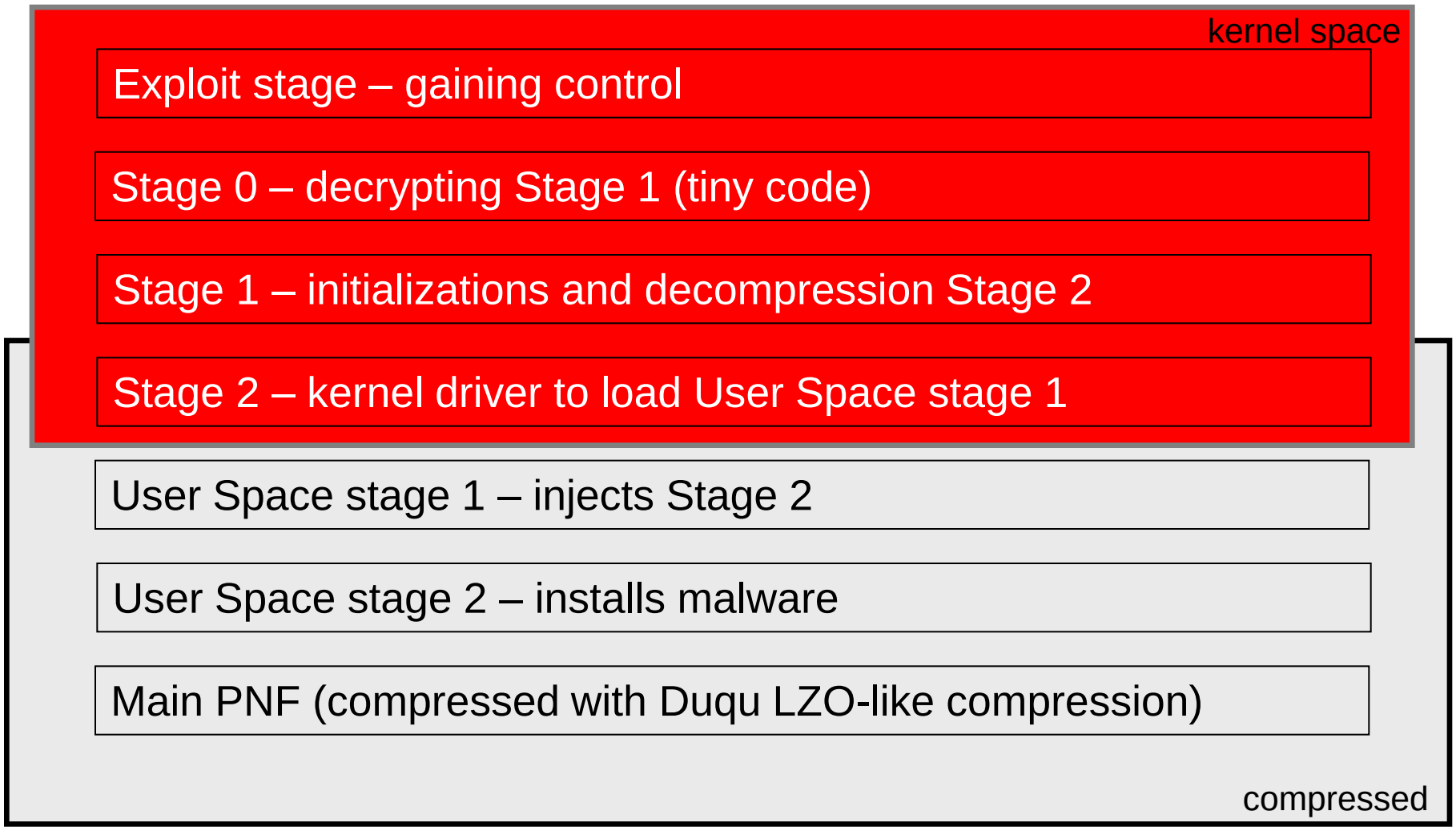
↓

Font exploited a ***Windows kernel vulnerability*** (CVE-2011-3402)

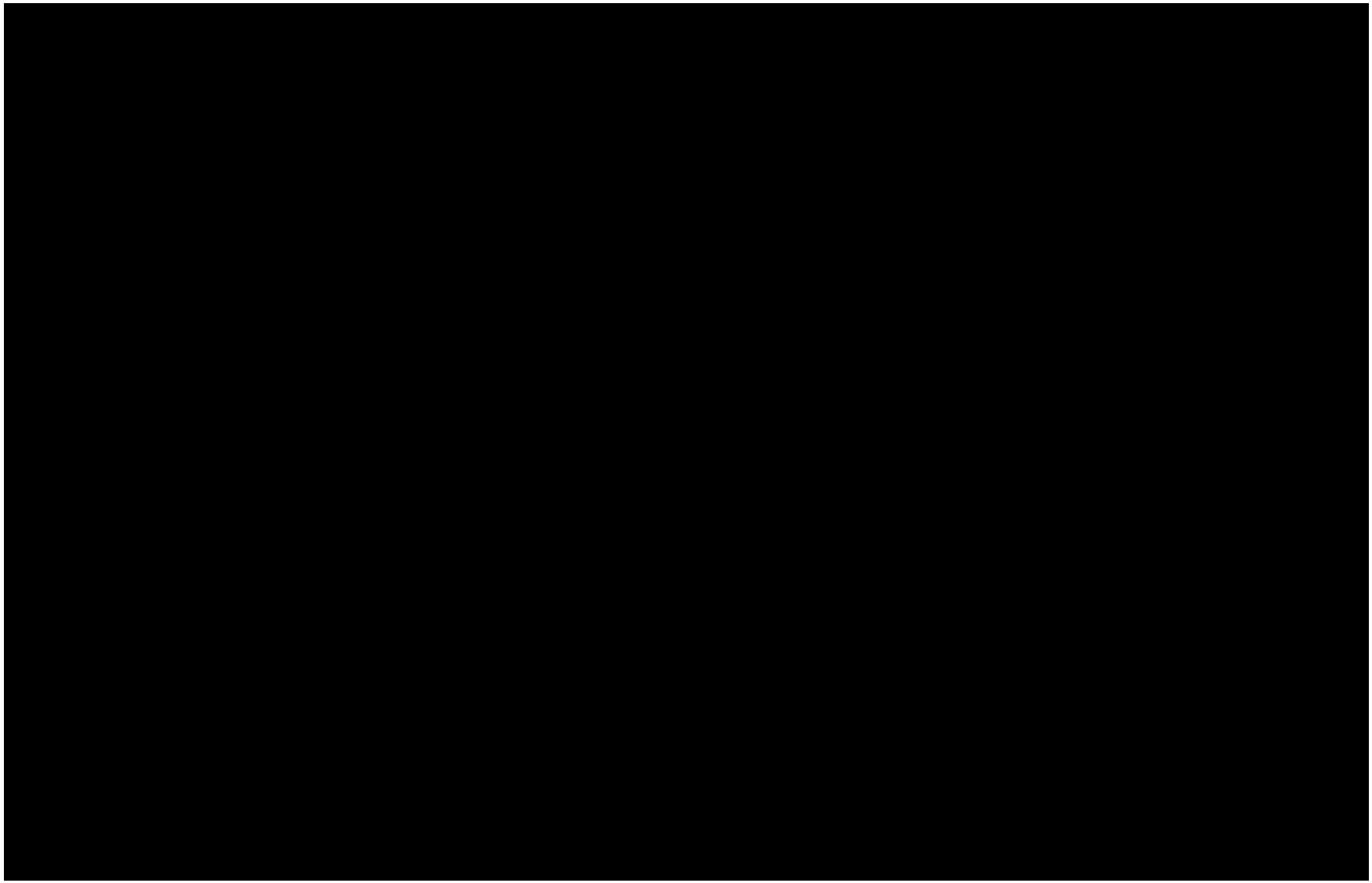
Dropper structure



Dropper font file logical structure



Duqu exploit demo



ATTACKER TECHNIQUES

Trend: Spear phishing and watering whole

Most of the time attackers use
spear phishing e-mails

However, ***watering hole*** attacks are getting more and more popular

Spear phising

Victims ***get an e-mail*** with attachments

Documents (DOC, PDF, RTF)

Compressed file formats (ZIP, RAR)

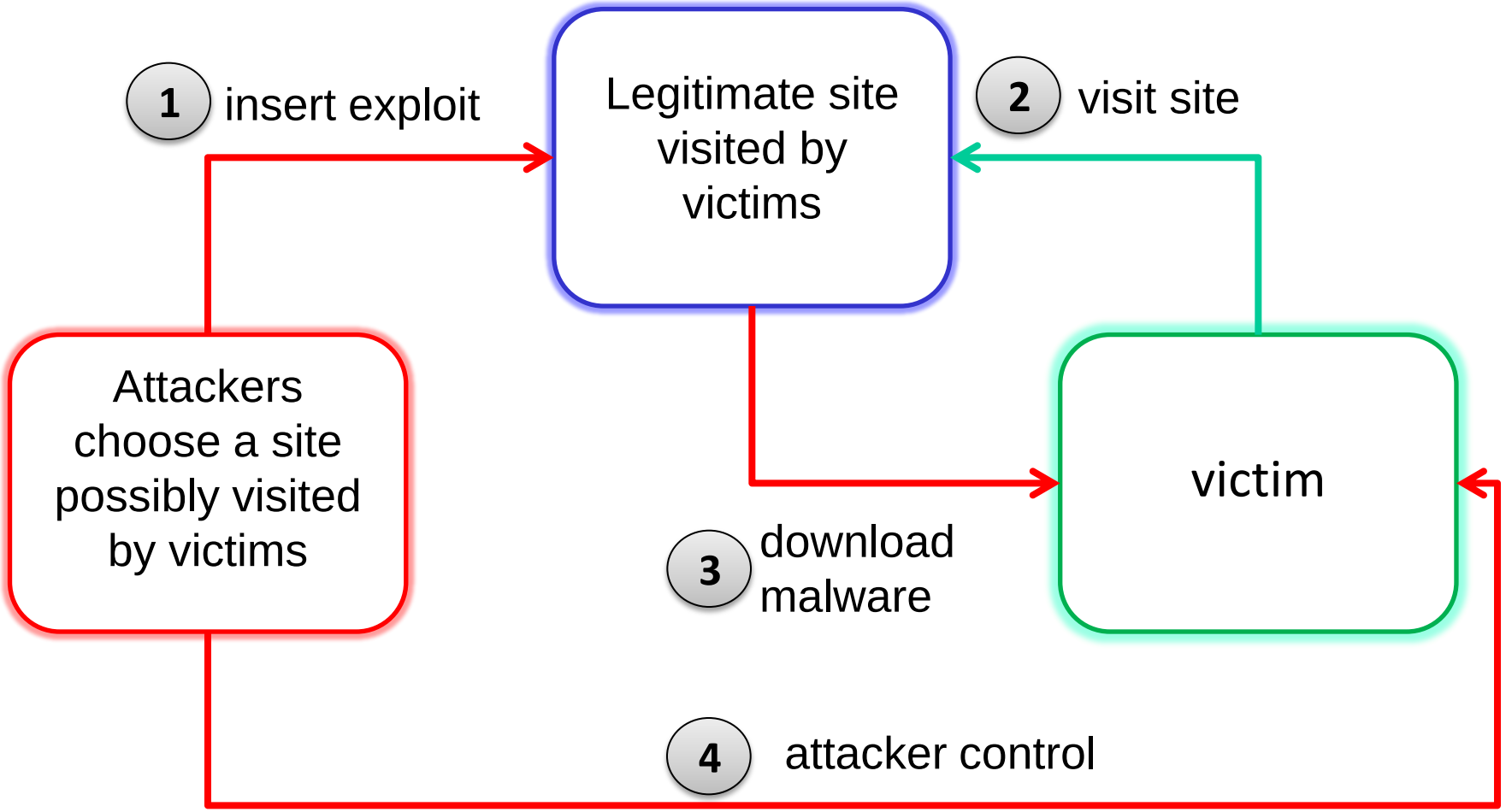
Attachments are typically

drive-by-downloaders

Examples

(Duqu, Miniduke, EvilGrab, NetTraveler, RARSTONE etc)

Watering hole attack



Example: VOHO attack campaign (June 2012)

Attacker: the China-based

Hidden Lynx

Target: ***US defence contractors***

However, files are protected by Bit9's

trusted file signing infrastructure

Example: VOHO attack campaign cont.

Use of ***watering hole*** attack against Bit9

Theft of the company's ***code signing certificates***

Goal: ***Sign malicious files*** with Bit9's protection

RECONNAISSANCE

Trend: Systematic reconnaissance

Use of well-known ***network scanners*** for vulnerability exploration

↓

Try to exploit ***known*** vulnerabilities with ***known exploits***

BUT, direct attacks via ***0-days*** are appearing

Example: Clever Kitten

Found by CrowdStrike

(April 2013)

Attackers: Indicators point to the Islamic Republic of ***Iran***

Example: Clever Kitten's reconnaissance

Use of ***Acunetix***

Web vulnerability scanner

PHP backdoor is uploaded via an exploitable site

Use of additional tools for ***lateral movement***

Packet sniffers to capture credentials

Additional vulnerability scanners

Contrast Example: Duqu's reconnaissance

Information gathering

about victims

Use of spear phishing email with

an attached ***0day*** DOC file

UNDER THE RADAR

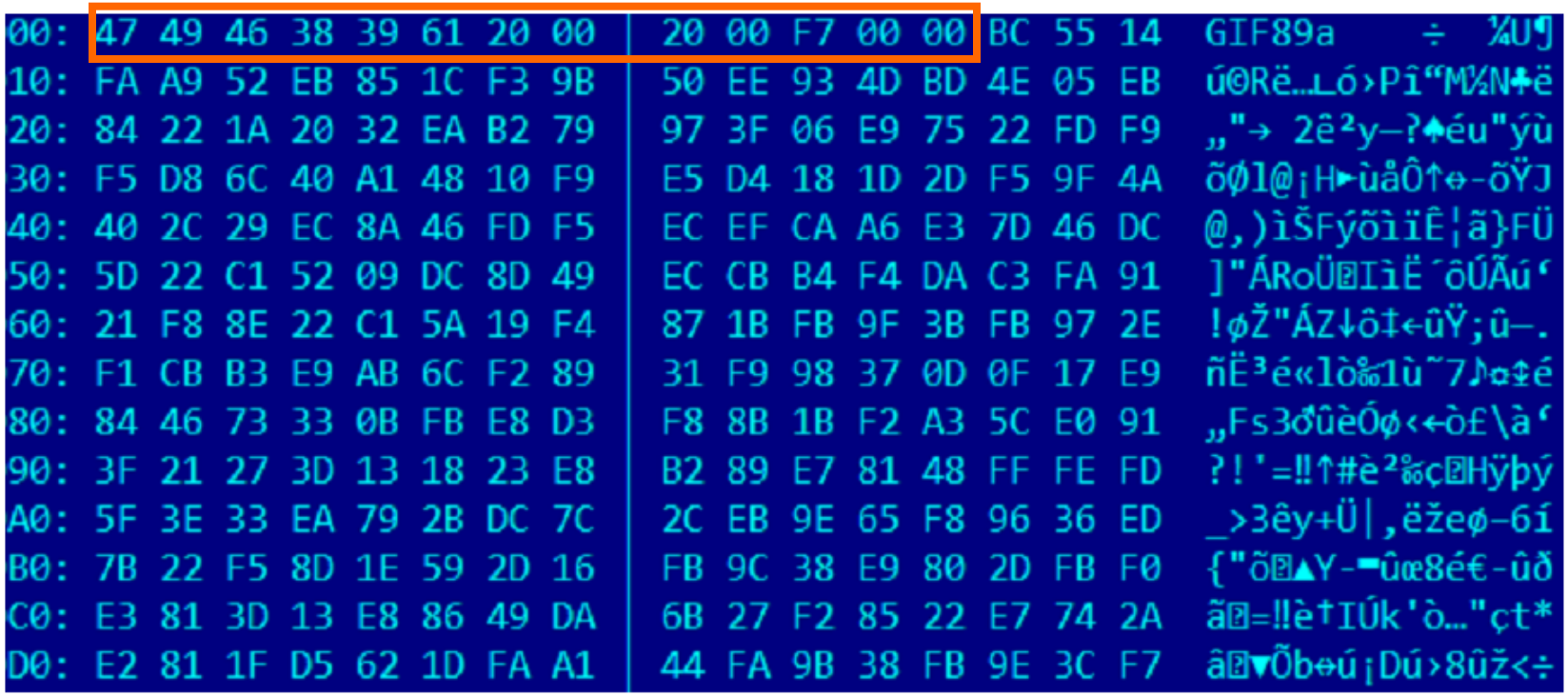
Trend: Stealthy C2 Communication

Use of ***image headers*** in
C2 communications recently

Goal: mislead perimeter defenses

Example: MiniDuke drive-by-download

- 13 bytes long GIF header + encrypted executable



Example: Duqu data exfiltration + C2 comm.

```
00000000  48 54 54 50 2f 31 2e 31  20 32 30 30 20 4f 4b 0d HTTP/1.1  200 OK.
00000010  0a 43 6f 6e 74 65 6e 74  2d 54 79 70 65 3a 20 69 .Content -Type: i
00000020  6d 61 67 65 2f 6a 70 65  67 0d 0a 54 72 61 6e 73 mage/jpe g..Trans
00000030  66 65 72 2d 45 6e 63 6f  64 69 6e 67 3a 20 63 68 fer-Enco ding: ch
00000040  75 6e 6b 65 64 0d 0a 43  6f 6e 6e 65 63 74 69 6f unked..C onnectio
00000050  6e 3a 20 43 6c 6f 73 65  0d 0a 0d 0a              n: Close ....
0000005C  32 45 30 0d 0a ff d8 ff  e0 00 10 4a 46 49 46 00 2E0..... ...JFIF
0000006C  01 01 01 00 60 00 60 00  00 ff db 00 43 00 02 01 ....`.`. ....C...
0000007C  01 02 01 01 02 02 02 02  02 02 02 02 03 05 03 03 ........ ........
0000008C  03 03 03 06 04 04 03 05  07 06 07 07 07 06 07 07 ........ ........
0000009C  08 09 0b 09 08 08 0a 08  07 07 0a 0d 0a 0a 0b 0c ........ ........
000000AC  0c 0c 0c 07 09 0e 0f 0d  0c 0e 0b 0c 0c 0c ff db ........ ........
000000BC  00 43 01 02 02 02 03 03  03 06 03 03 06 0c 08 07 .C...... ........
000000CC  08 0c 0c 0c 0c 0c 0c 0c  0c 0c 0c 0c 0c 0c 0c 0c ........ ........
000000DC  0c 0c 0c 0c 0c 0c 0c 0c  0c 0c 0c 0c 0c 0c 0c 0c ........ ........
000000EC  0c 0c 0c 0c 0c 0c 0c 0c  0c 0c 0c 0c 0c 0c 0c 0c ........ ........
```

Trend: Rare server-side polimorphism

Rare server-side polimorphism

However, the ***name of samples***, the ***name of modules***, ***the configuration*** and ***C&C servers*** are different

Example: Different sample names of MiniDuke

1109821546.gif 1118310968.gif 174239657.gif 2334309658.gif
2618653991.gif 2627081433.gif 3100425864.gif 3198217296.gif
3946889701.gif 3979106736.gif 4137794344.gif 626088424.gif
bg_aefk.gif bg_afvd.gif bg_dafd.gif bg_dasfs.gif bg_dfdsh.gif bg_dfell.gif
bg_dfesik.gif bg_dfeu.gif bg_dfew.gif bg_dfews.gif bg_dflj.gif bg_dfoiu.gif
bg_dfrio.gif bg_dfwe.gif bg_dsaf.gif bg_dsaffe.gif bg_dsef.gif
bg_dsert.gif bg_dwed.gif bg_edf.gif bg_edf_v2.gif bg_edfsa.gif
bg_edse.gif bg_eefds.gif bg_efd.gif bg_efdse.gif bg_efed.gif bg_efwe.gif
bg_ekjf.gif bg_ekks.gif bg_elfj.gif bg_elj.gif bg_esd.gif bg_ewfed.gif
bg_ewwe.gif bg_fdfe.gif bg_fed.gif bg_fefsf.gif bg_fked.gif bg_fwds.gif
bg_kefs.gif bg_kei.gif bg_keio.gif bg_kje.gif bg_kkf.gif bg_koe.gif
bg_ldfe.gif bg_leo.gif bg_lfe.gif bg_lkje.gif bg_lkjkef.gif bg_oef.gif
bg_ojlro.gif bg_qdf.gif bg_qrg.gif bg_rie.gif bg_ruie.gif bg_sasd.gif
bg_sdef.gif bg_sdefk.gif bg_sfef.gif bg_ureio.gif bg_wdf.gif

ENCRYPTION

Trend: Frequent use of simple ciphers

Frequent use of

Vigenere-like ciphers

or

simpler ***symmetric*** key algorithms

(e.g., XOR, RC4 etc)

Example: Understading Flame's "update" process

Flame ***abuses Windows Update*** to install malware components

Modified cabinet files (.cab) are download from update server

First stage: .cab files install a "loader"
(wusetupv.exe or similar)

Flame cabinet file decryption

.cab files are ***RC4*** encrypted inside

an

undocumented table format in

mscrypt.dat

mscrypt.dat table format

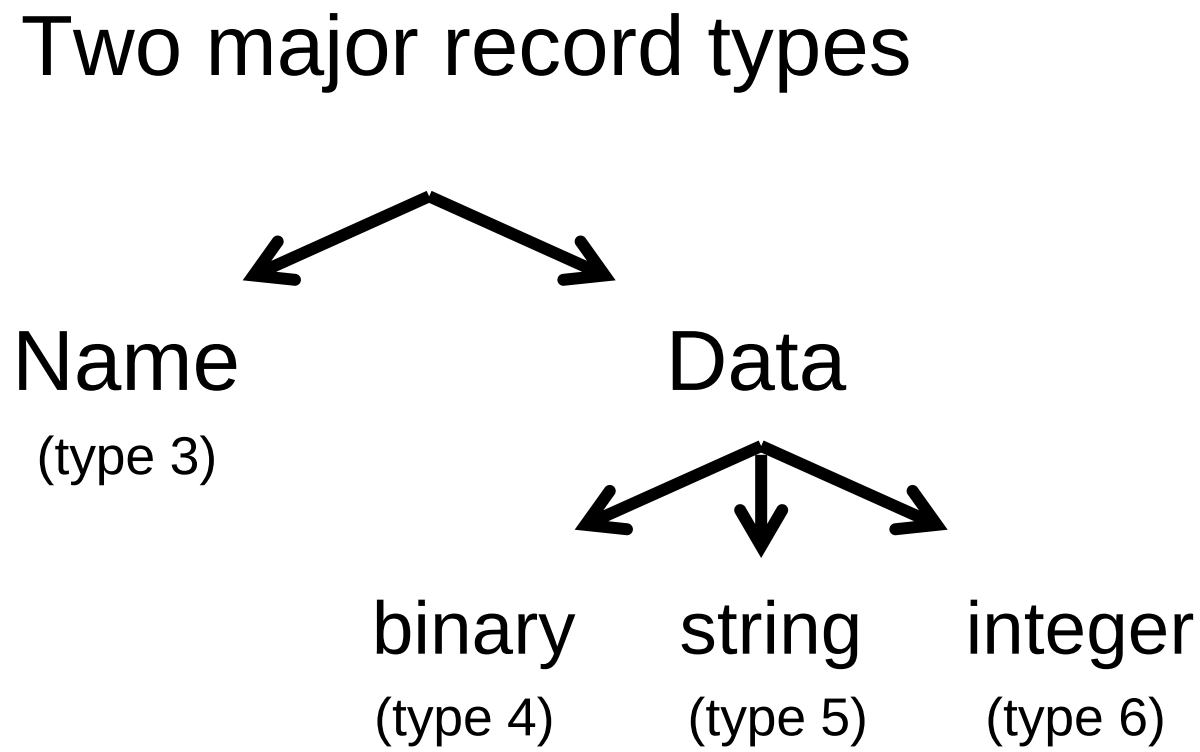
Record oriented table format

Records are ***padded to N*13*** bytes

and have an ***ID*** at the end

(very odd idea!)

mscrypt.dat table format



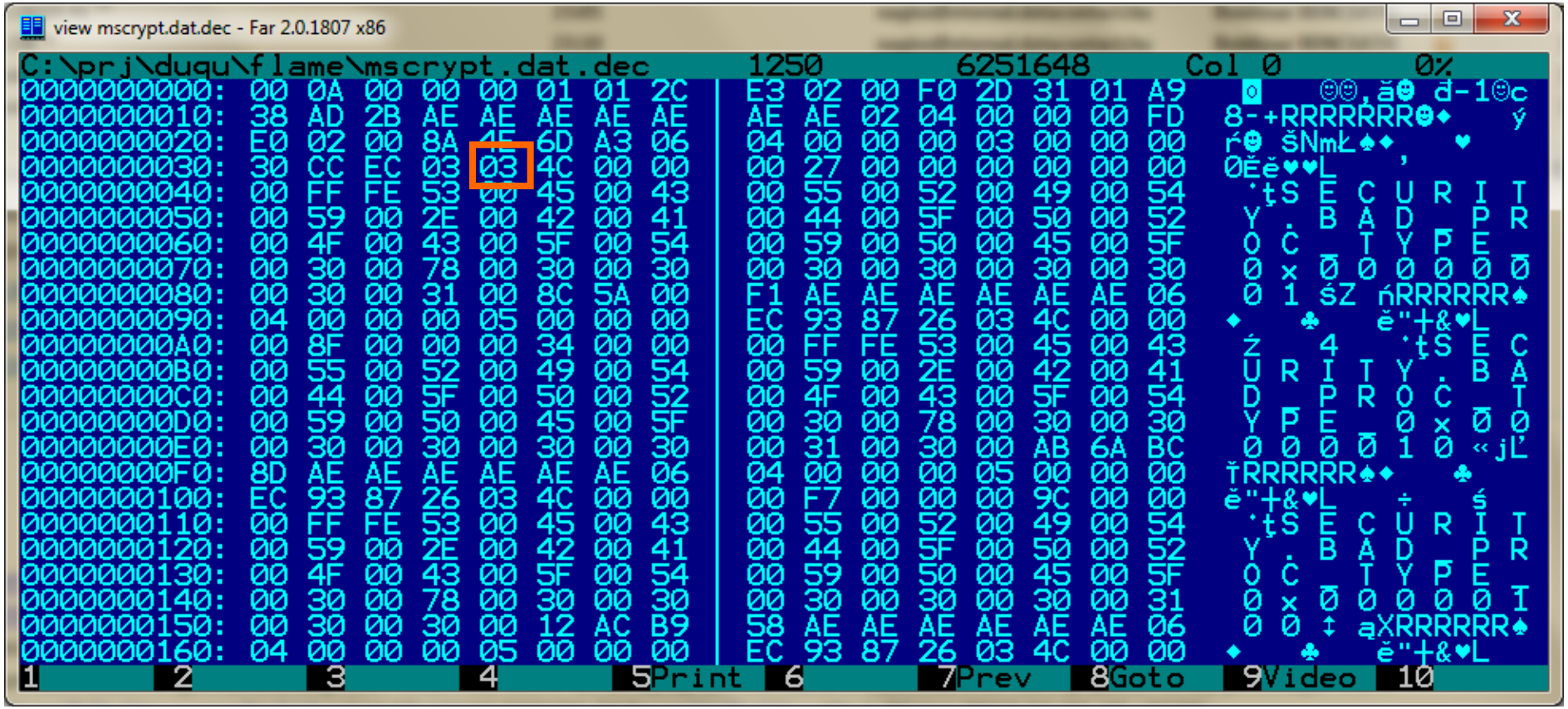
mscrypt.dat table linked list format

Each ***name*** record has a ***pointer to*** the corresponding ***data***

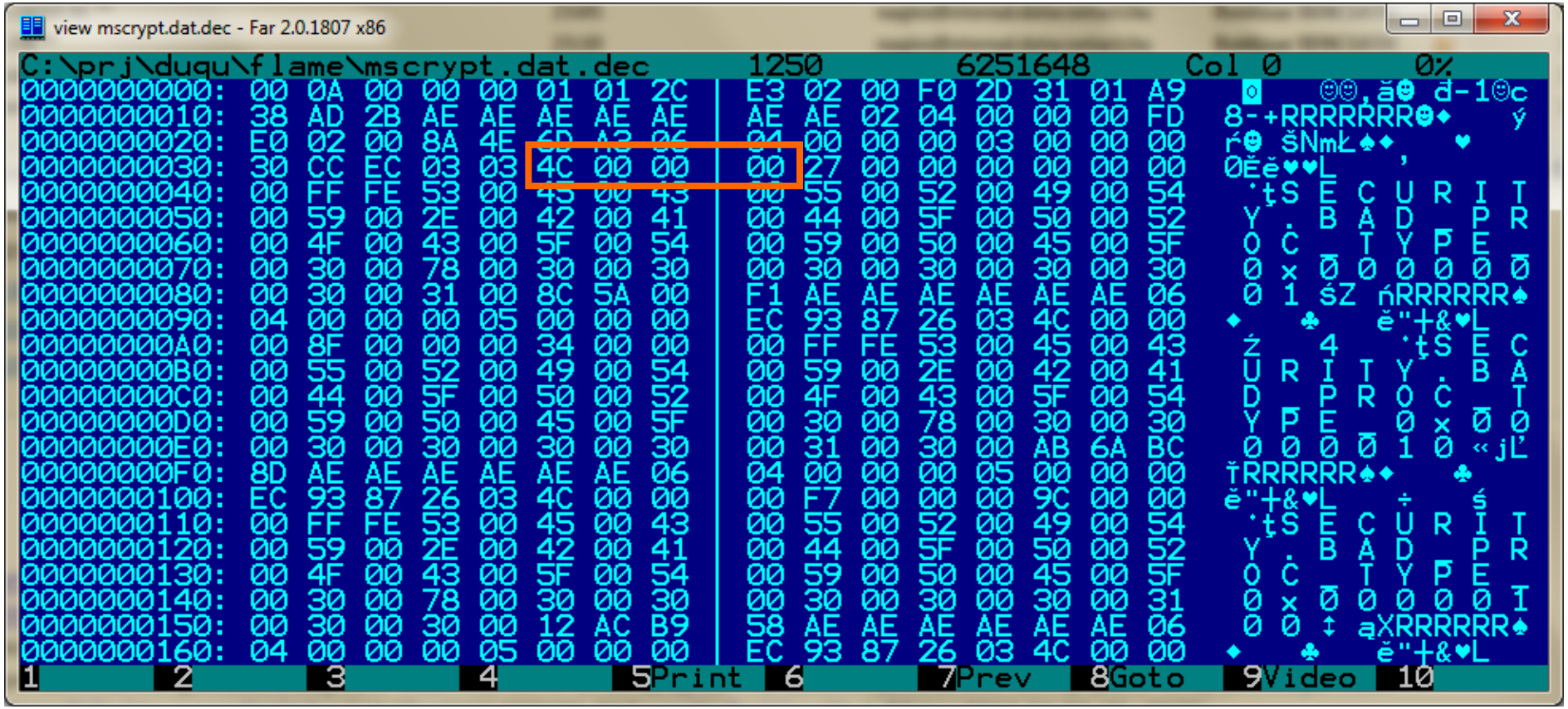
and

the ***previous name*** record

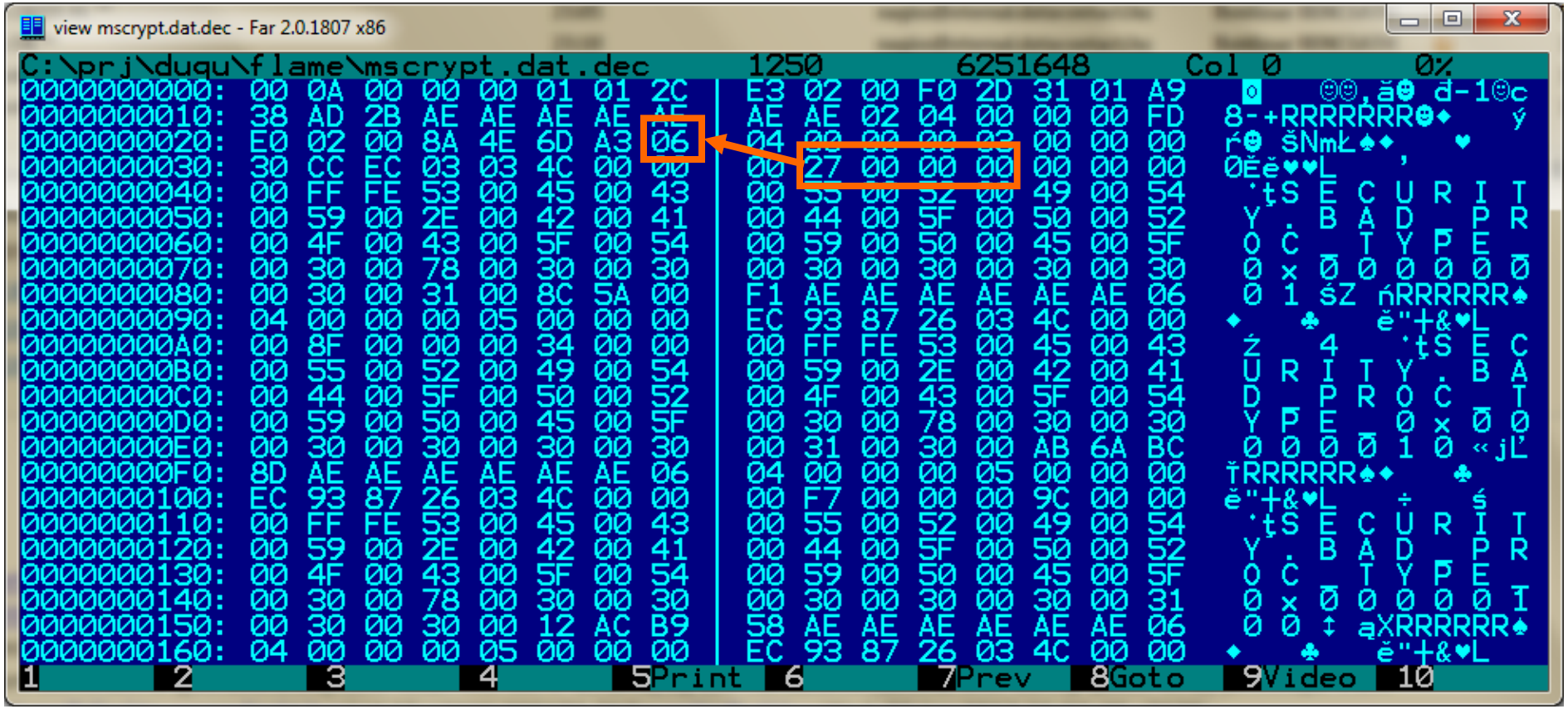
mscrypt.dat name record (type 3)



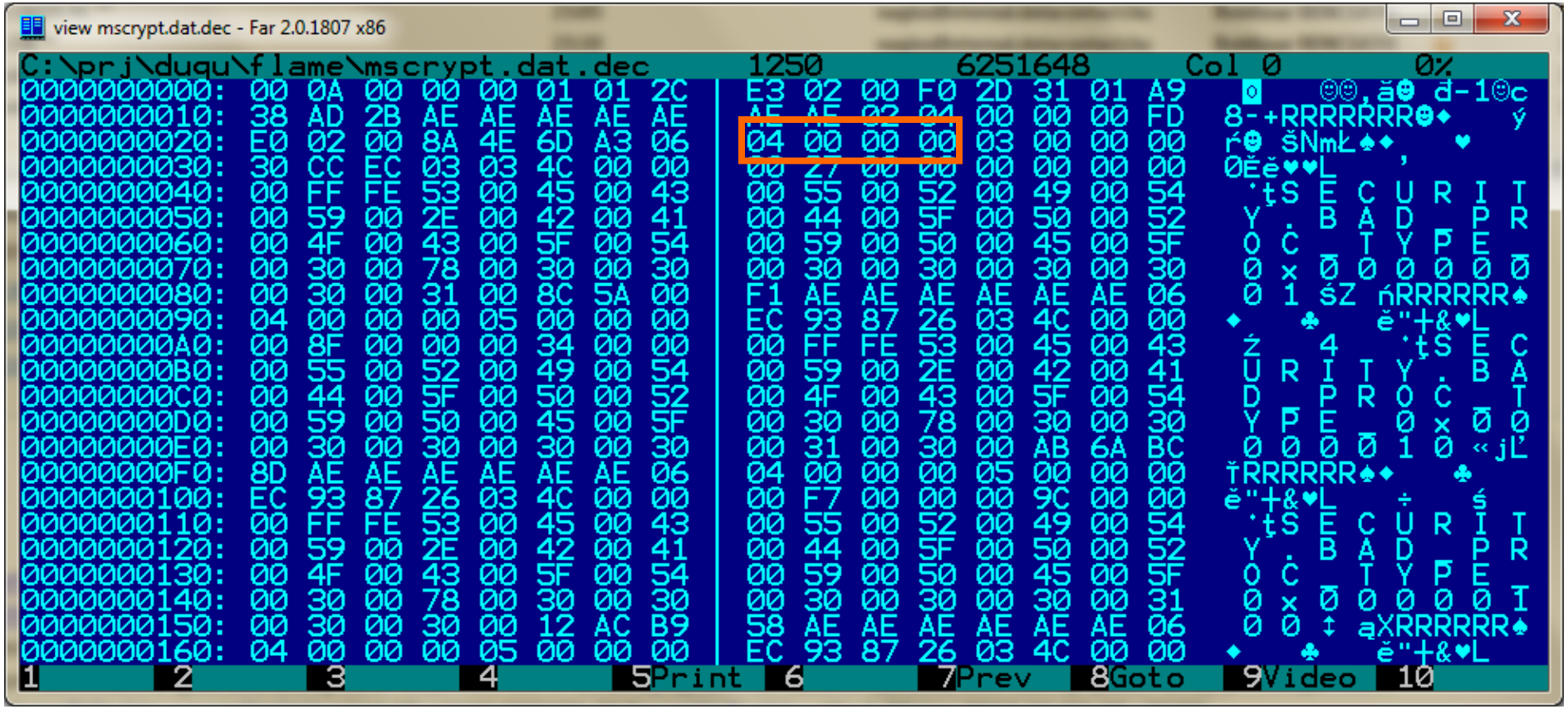
mscrypt.dat name record 4-byte length



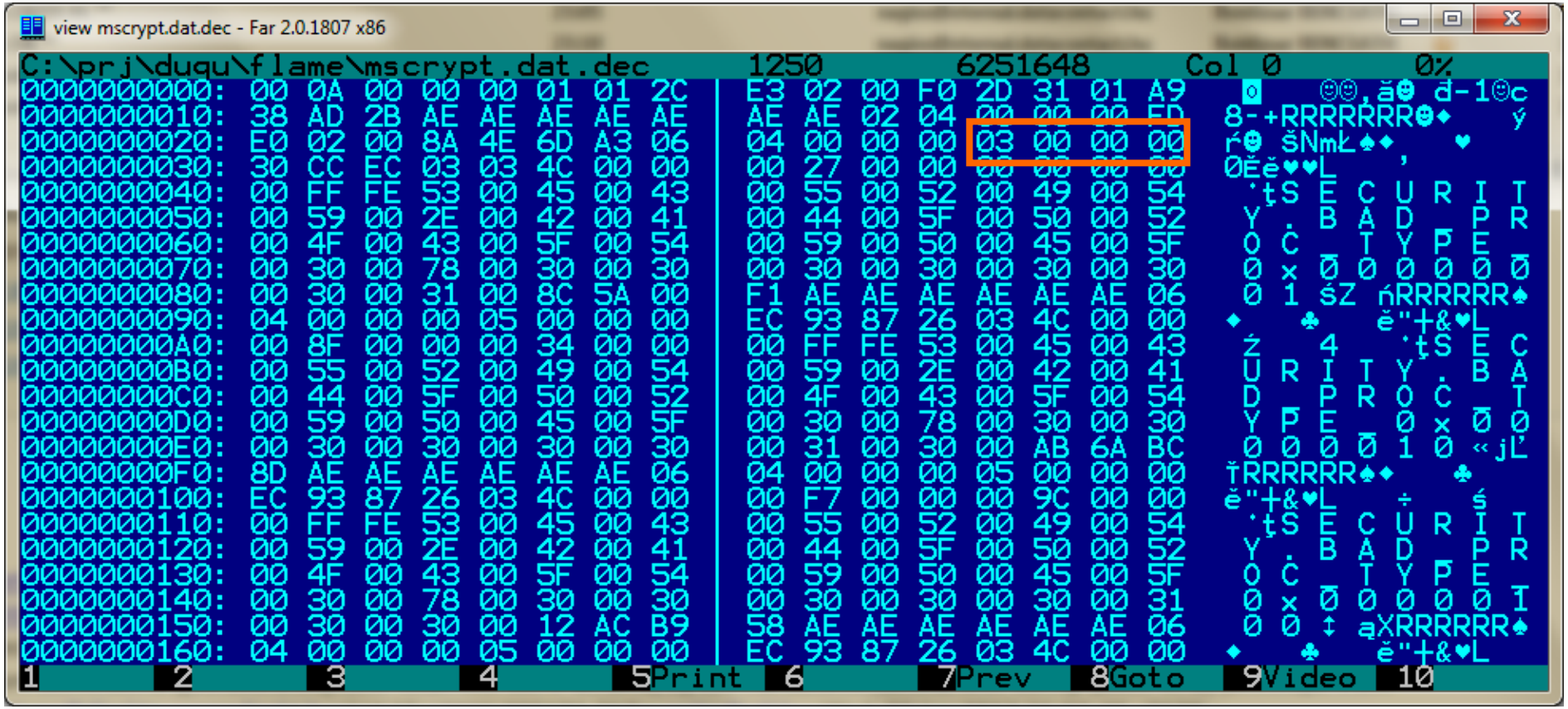
mscrypt.dat name record pointer to data record



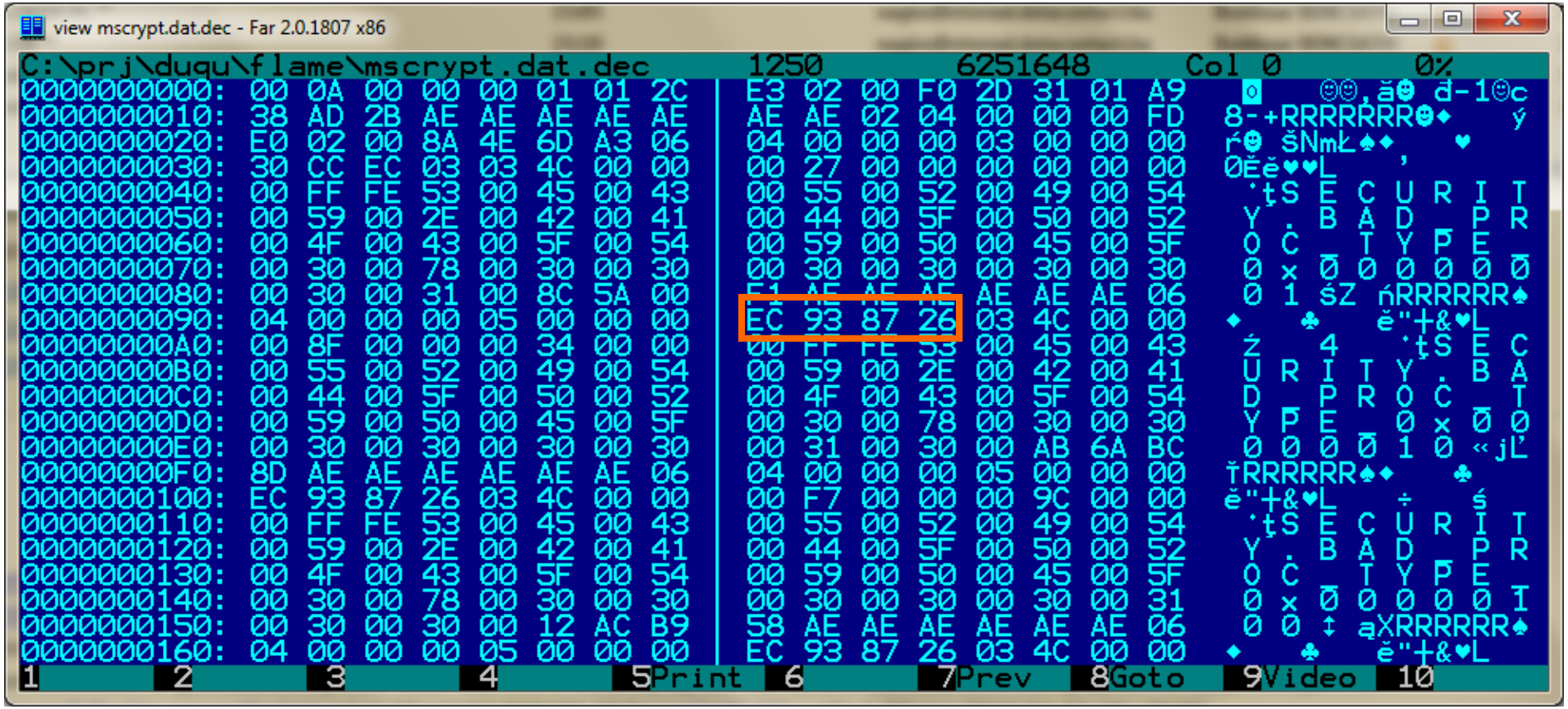
mscrypt.dat data record length



mscrypt.dat data record value

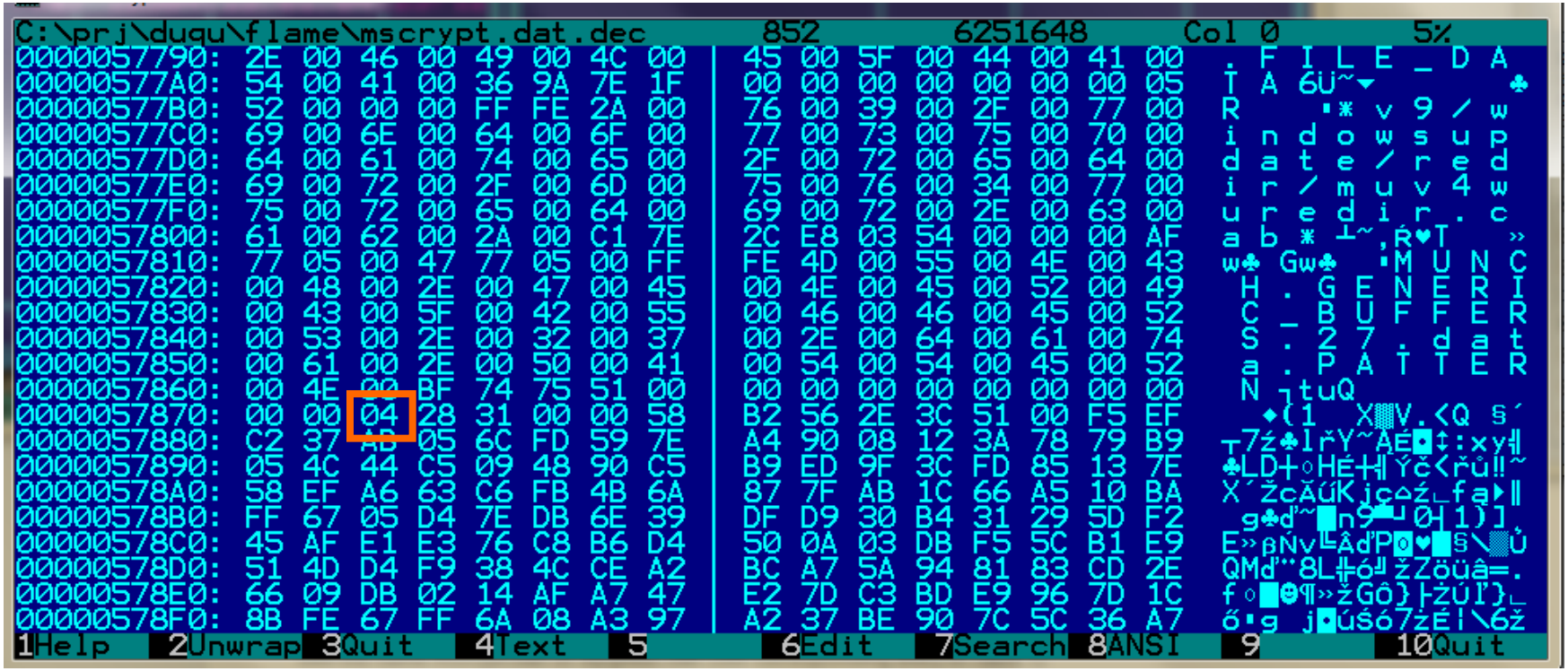


mscrypt.dat record ID

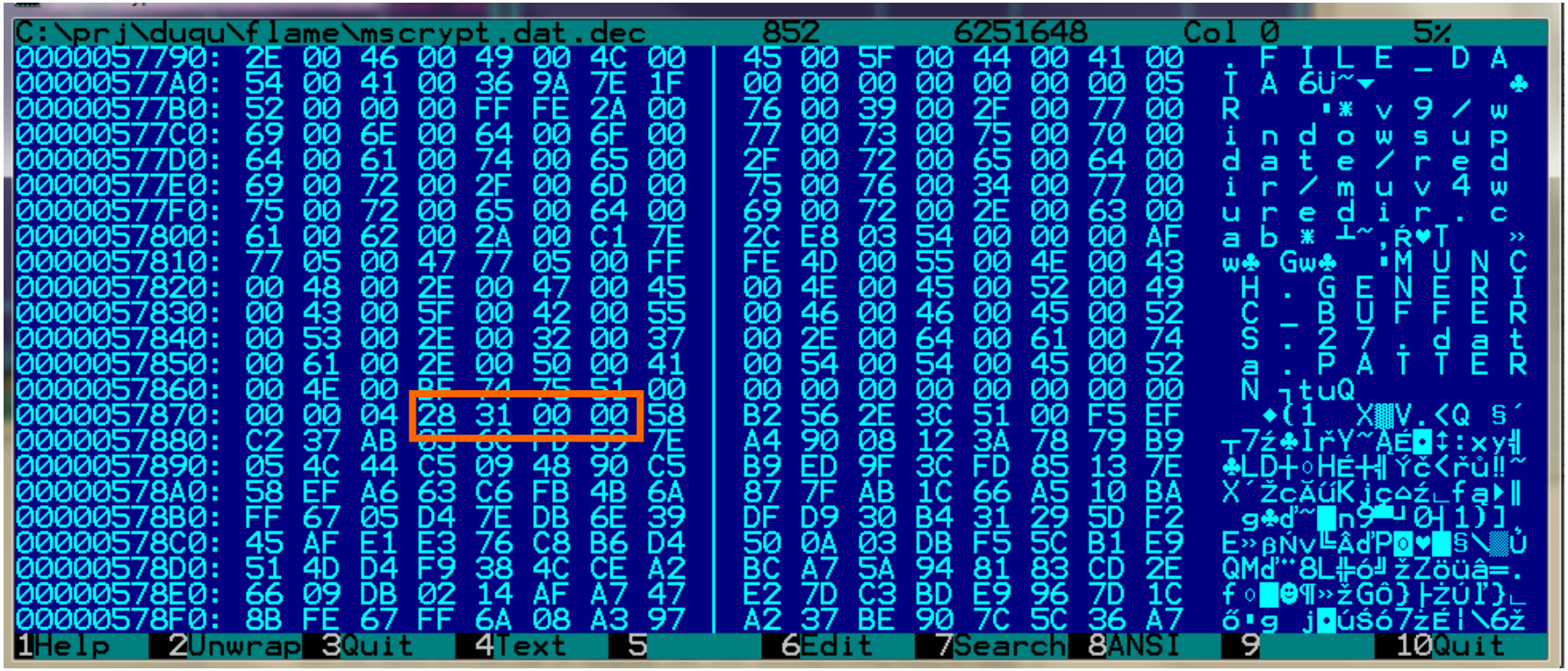


mscrypt.dat pointer to previous name record

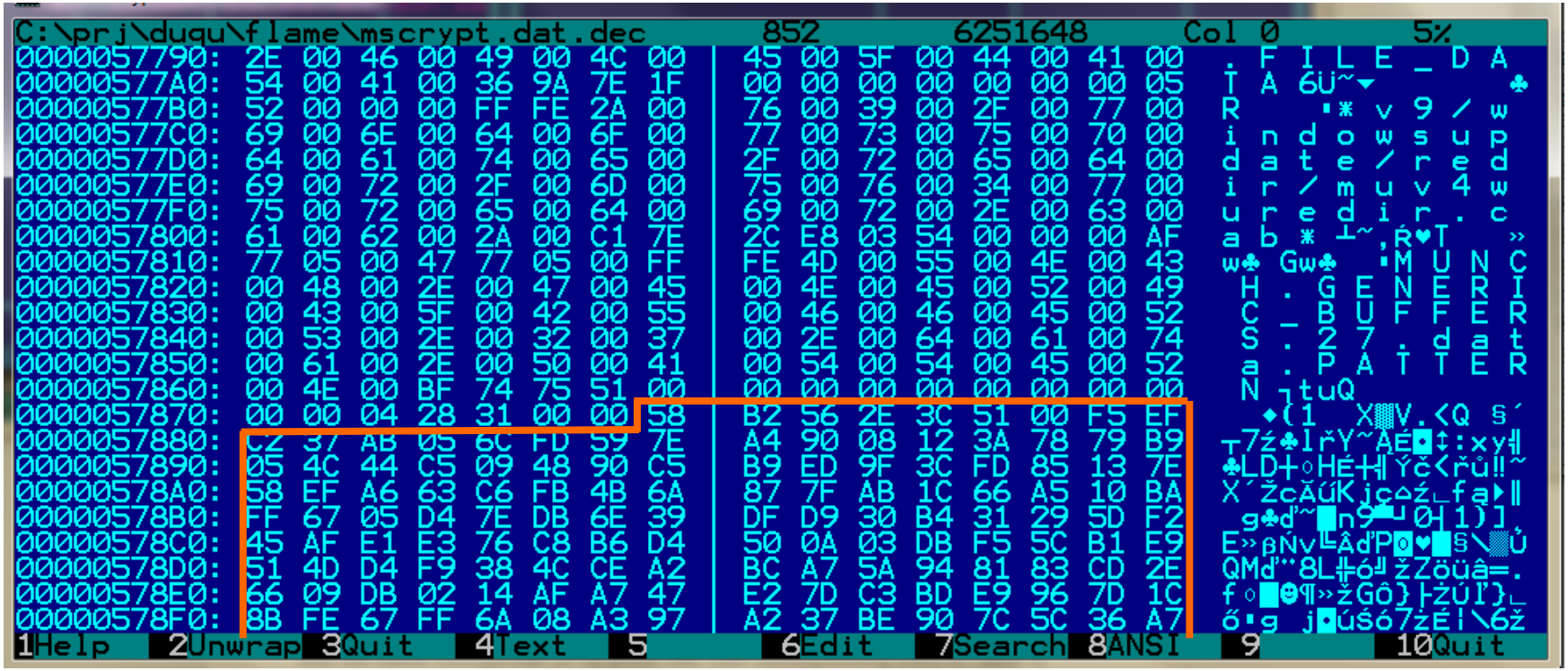
The cabinet files in mscrypt.dat - binary (type 4)



The cabinet files in mscrypt.dat - length



Wuident.cab RC4 encrypted + 4bytes ID



Wuident.cab encrypted + 4bytes ID

Cabinet files: Use of modified RC4

Flame's RC4 uses 104-byte keys

as described by Aleks Gostev

Strange key length!

Cabinet files: Use of modified RC4

However, the code contains a

100-byte long key string only

Extended by 4 pieces of 0x00 bytes

(can be used for key diversification)

Trend: Appearance of advanced encryptions

Appearance of ***asymmetric*** key encryption

(e.g., RSA)

and

advanced symmetric key algorihtms

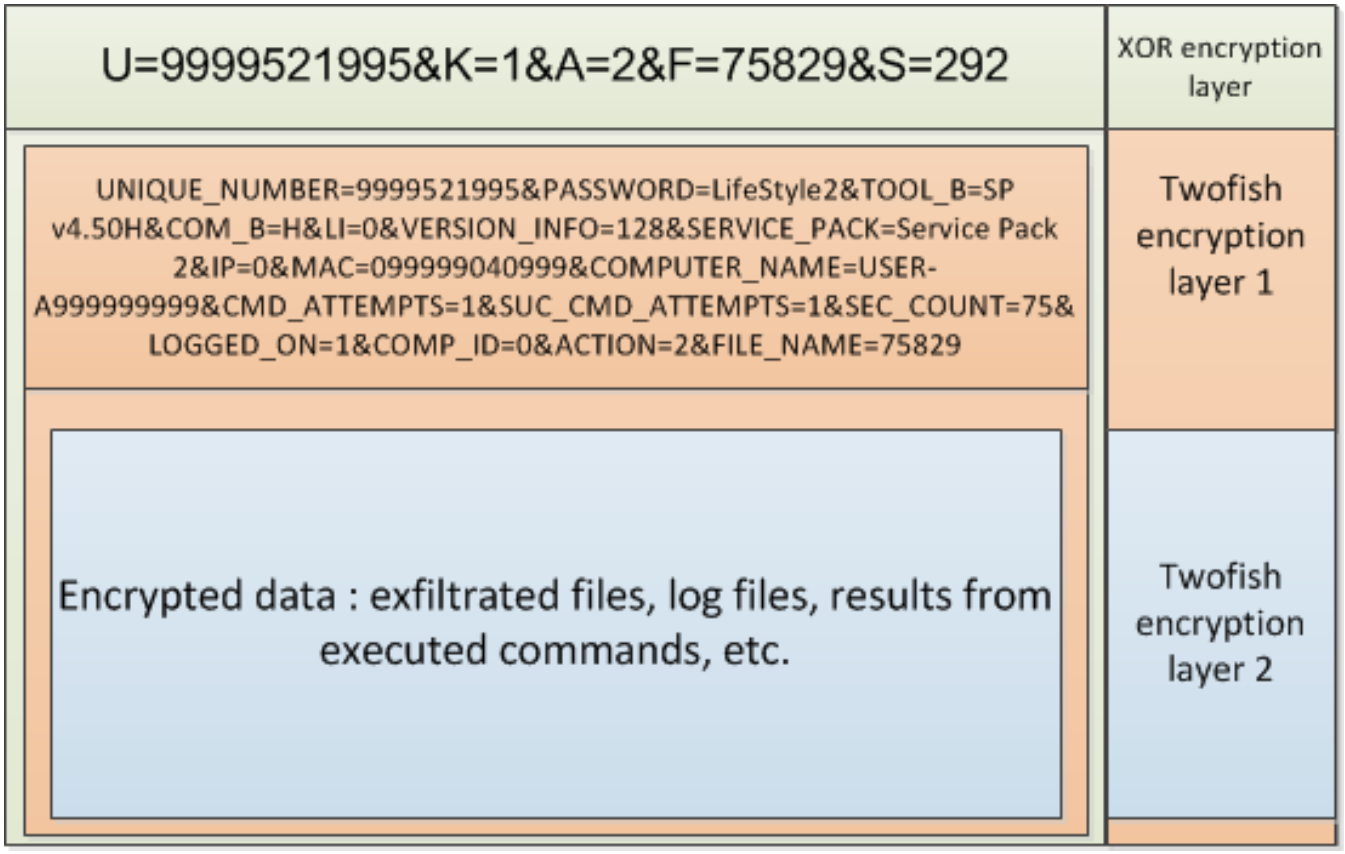
(e.g., Twofish, AES)

Example: SPE's C2 communication

The most detailed report contained some errors and missed some details

http://www.securelist.com/en/analysis/204792247/miniFlame_aka_SPE_Elvis_and_his_friends

SPE's C2 communication (original analysis)



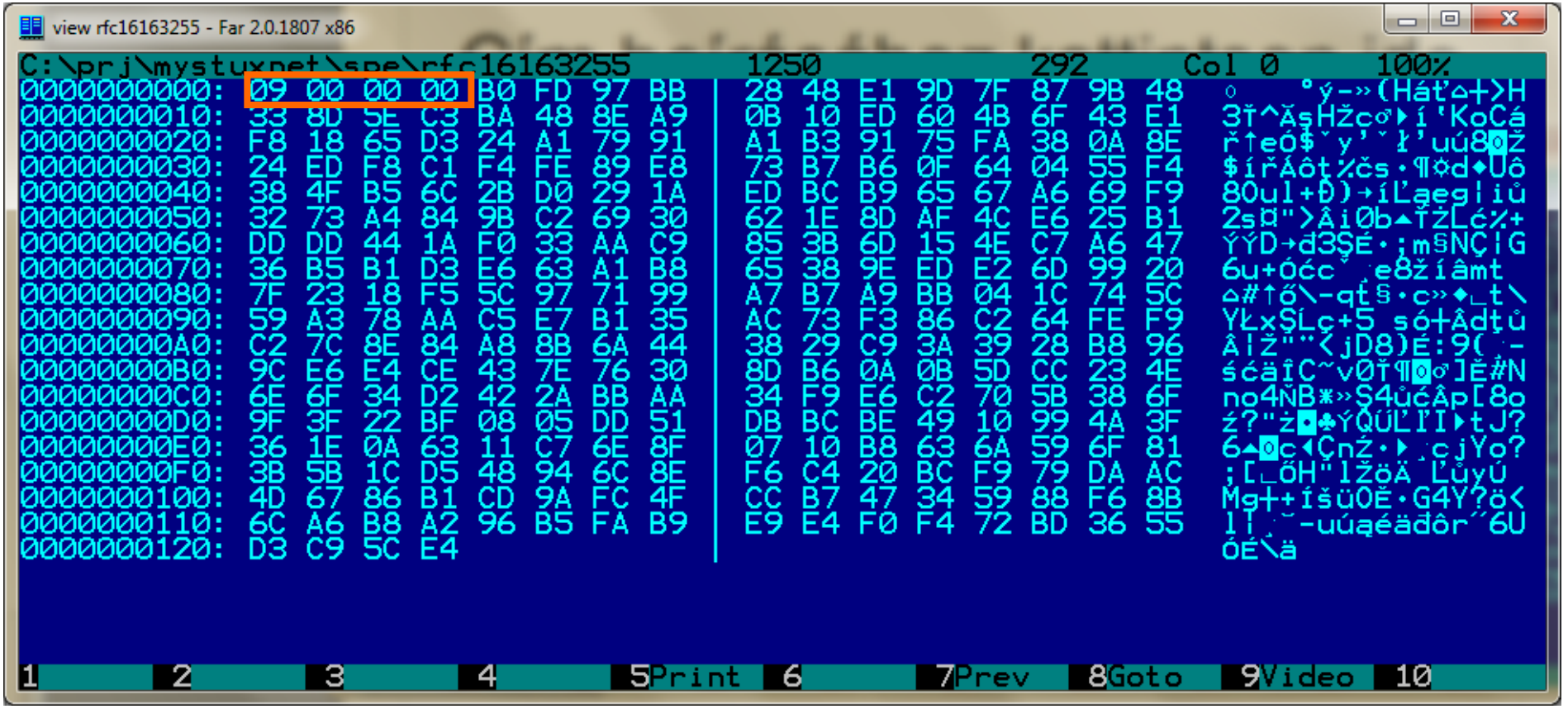
SPE's real C2 communication (no layers)

Example: SPE uses Twofish for C2 comm.

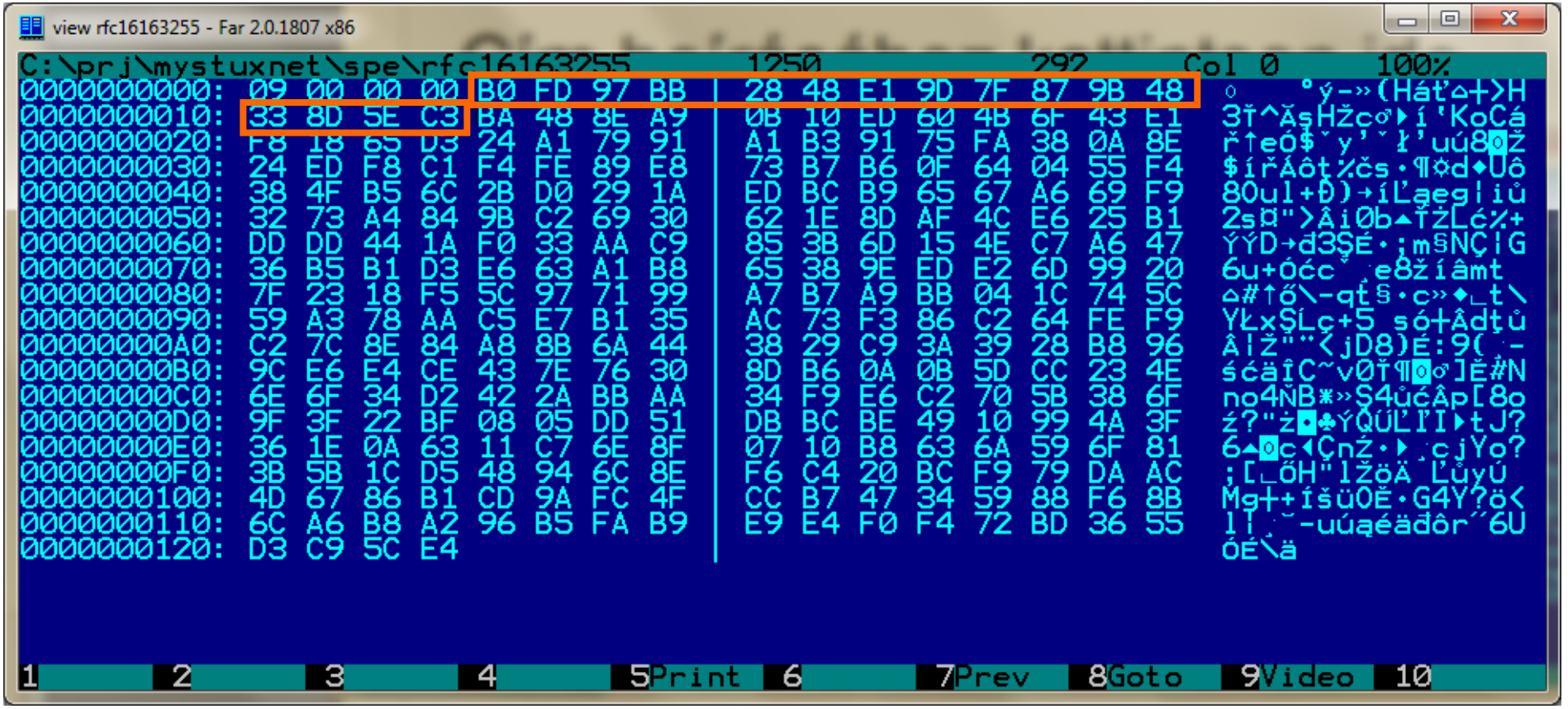
SPE's Twofish algorithm was not detailed

(e.g., uses CBC mode)

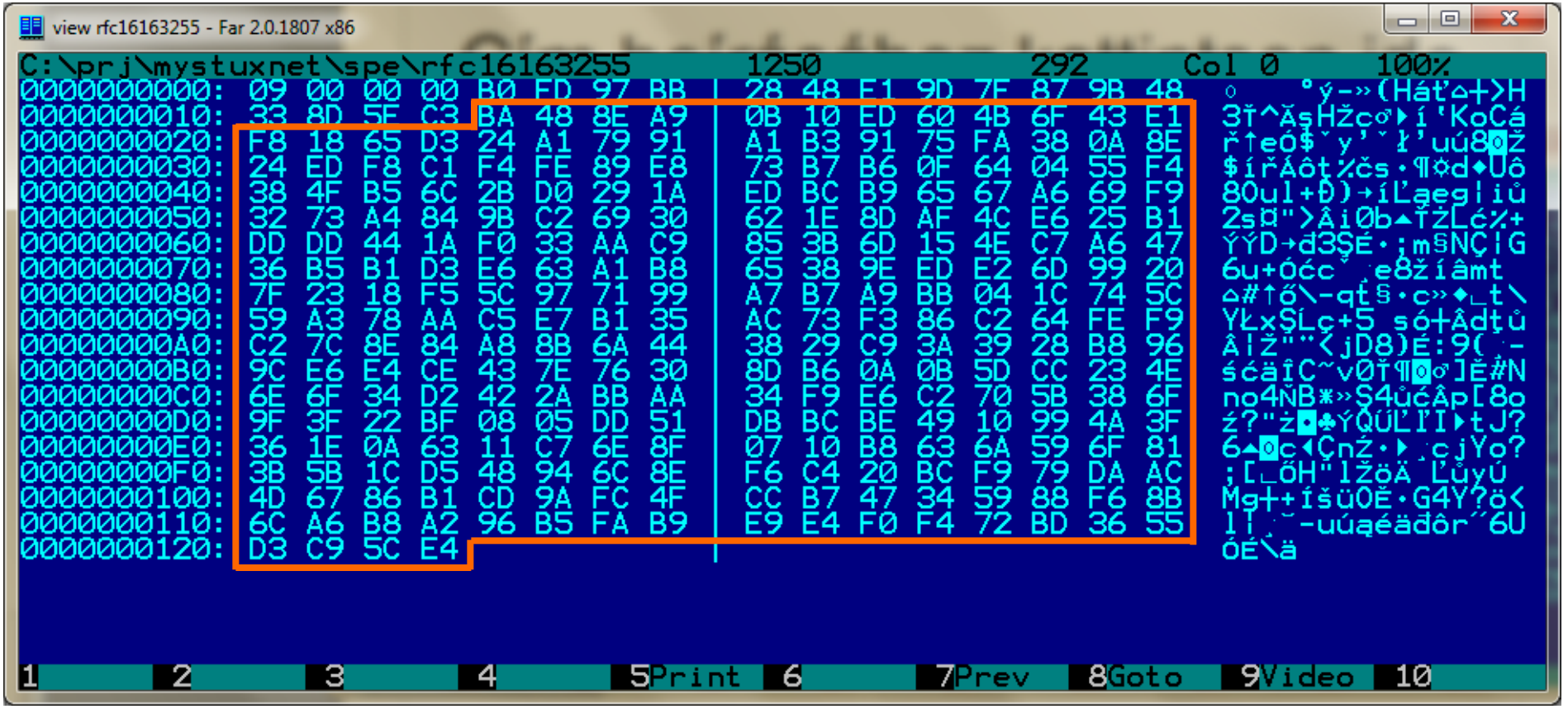
SPE’s 4-byte header



SPE’s 16-byte IV for Twofish



SPE’s encrypted part (padded CBC block)



SPE’s Twofish content unencrypted

COMPRESSION

Trend: Use of comprerssion algorithms

Use of either ***unmodified*** or ***modified*** versions of known compression algorithms

↓

E.g., LZMA, LZO, Bzip

Example: Decompressor in Duqu dropper

Duqu dropper decompressor			LZMA at read.pudn.com/downloads94/sourcecode/zip/372835/Source/lzma_depack.inc__.htm
seg000:000011C0 000	lea	eax, [ebx+eax*4]	@loc_401320:
seg000:000011C3 000	mov	ecx, eax	mov ecx,[edi]
seg000:000011C5 000	mov	eax, [ecx]	mov edx,eax
seg000:000011C7 000	mov	edx, [ebp-0Ch]	**shr edx,0Bh**
seg000:000011CA 000	**shr**	**edx, 0Bh** ;	**imul edx,ecx**
seg000:000011CD 000	**mul**	**edx**	cmp [ebp+0Ch],edx
seg000:000011CF 000	cmp	eax, [ebp-10h]	jnb @loc_40136C
seg000:000011D2 000	jbe	short loc_11FC	mov esi,[ebp-10h]
seg000:000011D4 000	mov	[ebp-0Ch], eax	mov eax,edx
seg000:000011D7 000	**mov**	**edx, 800h**	**mov edx,800h**
seg000:000011DC 000	sub	edx, [ecx] ;	sub edx,ecx
seg000:000011DE 000	**shr**	**edx, 5** ;	**shr edx,5**
seg000:000011E1 000	add	[ecx], edx	add edx,ecx
			xor ecx,ecx

Example: Duqu dropper compression

We found very similar code chunks
in ***LZMA***

However, we could ***no***t find an
identical implementation

Duqu dropper decompression/recompression

We ran ***Duqu decompressor*** to decompress the payload

And

Re-compressed with ***LZMA***

LZMA dictionary sizes

We got back the ***original byte stream***

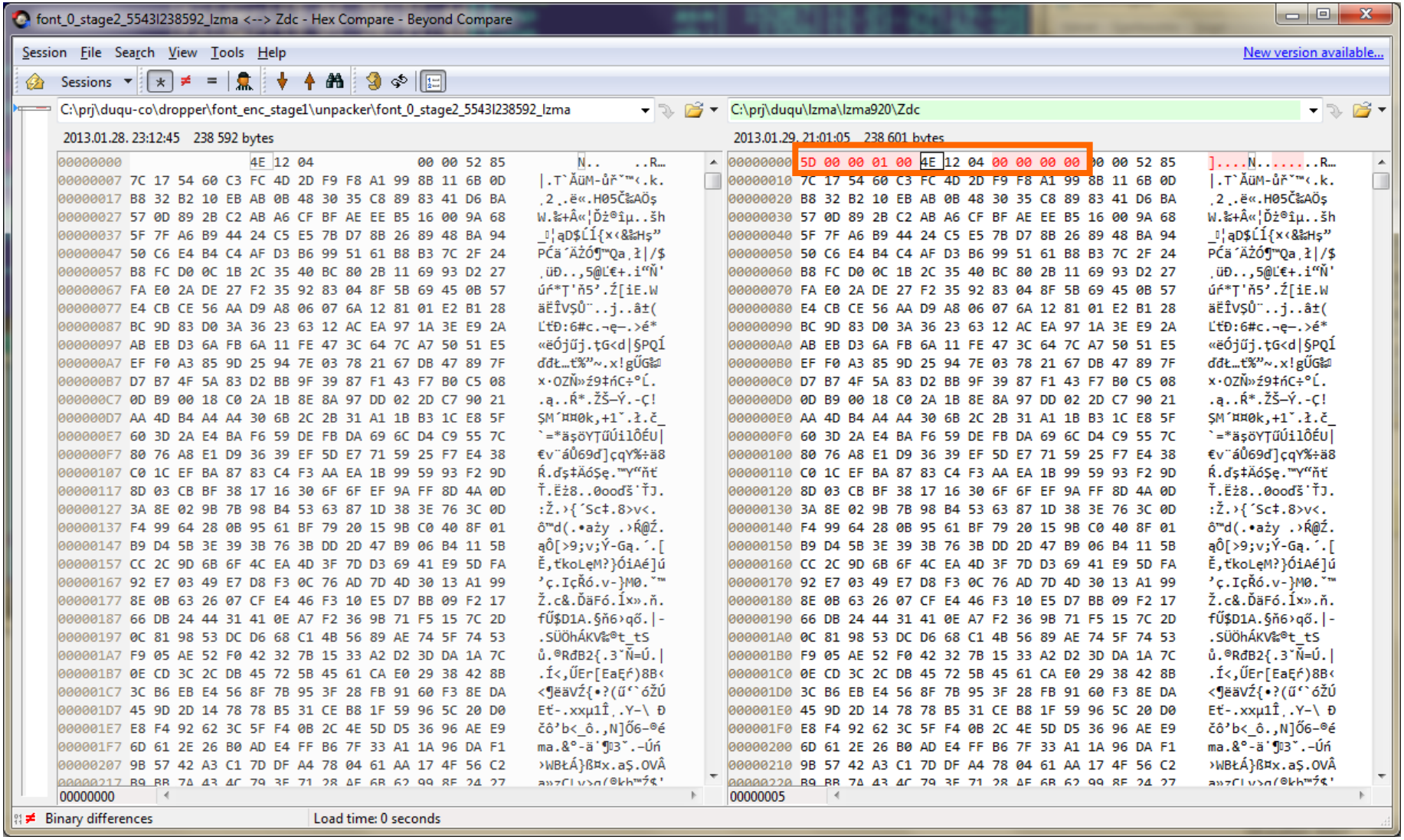
with ***d16*** dictionary size

(lzma.exe e Zd Zdc -a1 -d16)

The ***default*** lzma.exe dictionary size is

d22

Duqu dropper LZMA verified



MODULARITY AND CODE REUSE

Trend: Use of known and unknown modules

Attackers build code from either
self-made or ***external*** modules

E.g.,

Putty, Socks proxy, VNC, keyloggers

Example: Use of known modules

Stuxnet: LZO

Duqu: ~LZMA, LZO

Flame: putty, SQLite, libbz2, zlib, Lua

Teamspy: Teamviewer

Zeus (RCApp): VNC

What about licenses? ☺

LZO: GPL

LZMA: LGPL, Common Public License

Putty: MIT license

SQLite: public domain

libbz2: BSD-style license

zlib: very permissive zlib_license

Lua: MIT license

Teamviewer: commercial product

Example: Duqu's self-made screen recorder

First, a ***full screen*** is captured in 16 colors

(Saved as BMP with missing header)

and

only ***incremental*** parts are saved afterwards

(This was a joint work with one of our students, Roland Kamarás)

A sample for incremental screen capture data

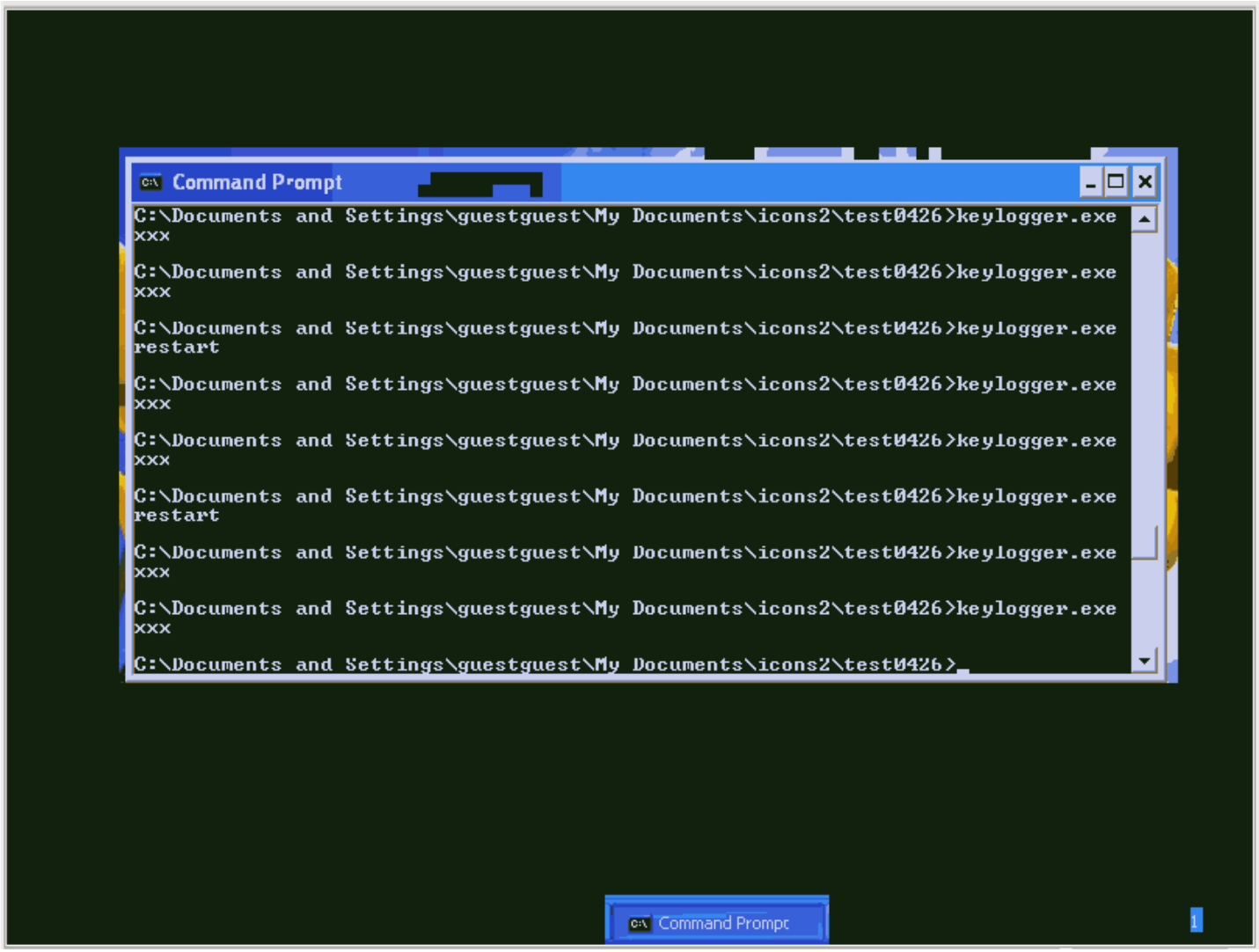
Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
00000000	F0	04	97	02	4A	00	03	00	08	08	00	01	00	00	00	80
00000010	00	33	33	33	33	22	22	22	22	11	11	11	11	44	44	44
00000020	44	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000030	00	00	33	33	33	33	22	22	22	22	11	11	11	11	44	44
00000040	44	44	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000050	00	00	00	33	33	33	33	22	22	22	22	11	11	11	11	44
00000060	44	44	44	00	00	00	00	00	00	00	00	00	00	00	00	00
00000070	00	00	00	02	10	80	00	33	33	33	33	22	22	22	22	11
00000080	11	11	11	44	44	44	44	00	00	00	00	00	00	00	00	00
00000090	00	00	00	00	00	00	00	00	33	33	33	33	22	22	22	22

Incremental screen capture (complicated format)

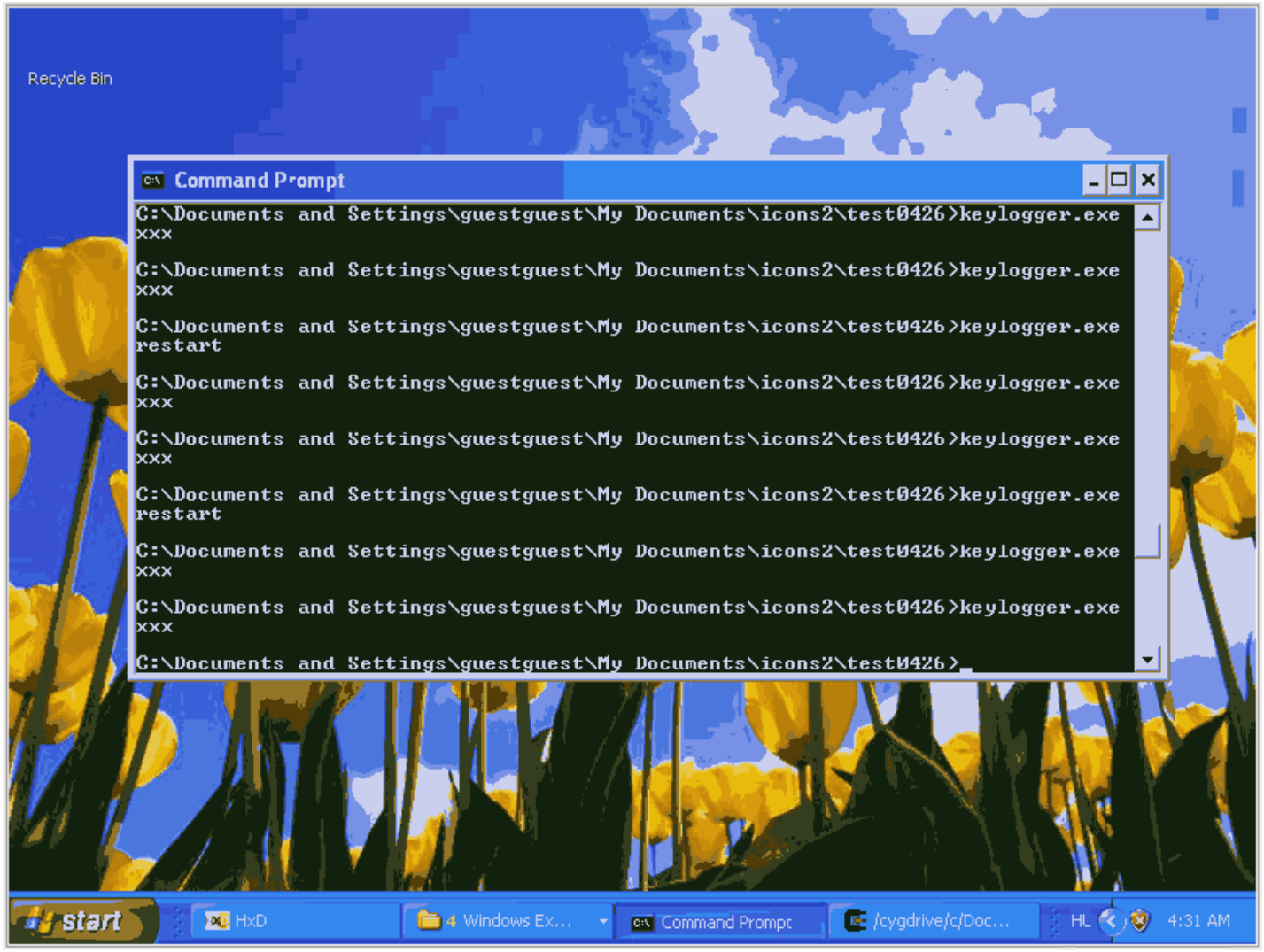
```
Incremental image file: dq_test/0010-03.dqp.out
==> File size with header: 17734 byte
==> File size without header: 17718 byte
==> Width: 1024 pix
==> Height: 768 pix
        ==> 0x00:00 records: 5
        ==> 0x00 records: 489
        ==> 0x01 records: 27
        ==> 0x02 records: 63
        ==> 0x04 records: 560
        ==> Other records: 53
                ==> 0x10 record: 28
                ==> 0x98 record: 1
                ==> 0x38 record: 1
                ==> 0x18 record: 11
                ==> 0x28 record: 4
                ==> 0x20 record: 2
                ==> 0x40 record: 1
                ==> 0xb0 record: 1
                ==> 0x48 record: 3
                ==> 0x60 record: 1
==> Sum: 1197 records.
```

Sample - Inremental image 1

Sample – Incremental image 2



Sample – merged image with incr. parts



C2 COMMUNICATION

Trend: Use of social media

No IRC and ***peer-to-peer*** C2 communication

Advanced tricks***:***

google queries, social media

Example: MiniDuke uses social media

Initial web page queries (DNS resolve?)

www.google.com - port TCP/80 - HTTP

twitter.com –port TCP/443 - SSL

www.geoiptool.com –port TCP/80 – HTTP

Google search strings

lUFEfiHKljfLKWPR

HkyeiIDKiroLaKYr

lUFEfiHKDroLaKYr

Example: MiniDuke C2 URLs on Twitter

The weather is good today. Sunny!
uri!wp07VkkxYt3Mne5uiDkz4Il/Iw48Ge/EWg==

Albert, my cousin. He is working hard.
uri!wp07VkkxYmfNkwN2nBmx4ch/Iu2c+GJow39HbphL

My native town was ruined by tornado.
uri!wp07VkkxYt3Md/JOnLhzRL2FJjY8l2It

Sample twitter message used by MiniDuke

C2 INFRASTRUCTURE

Trend: dedicated C2 infrastructure

Use of dedicated C2 servers

(mostly ***http)***

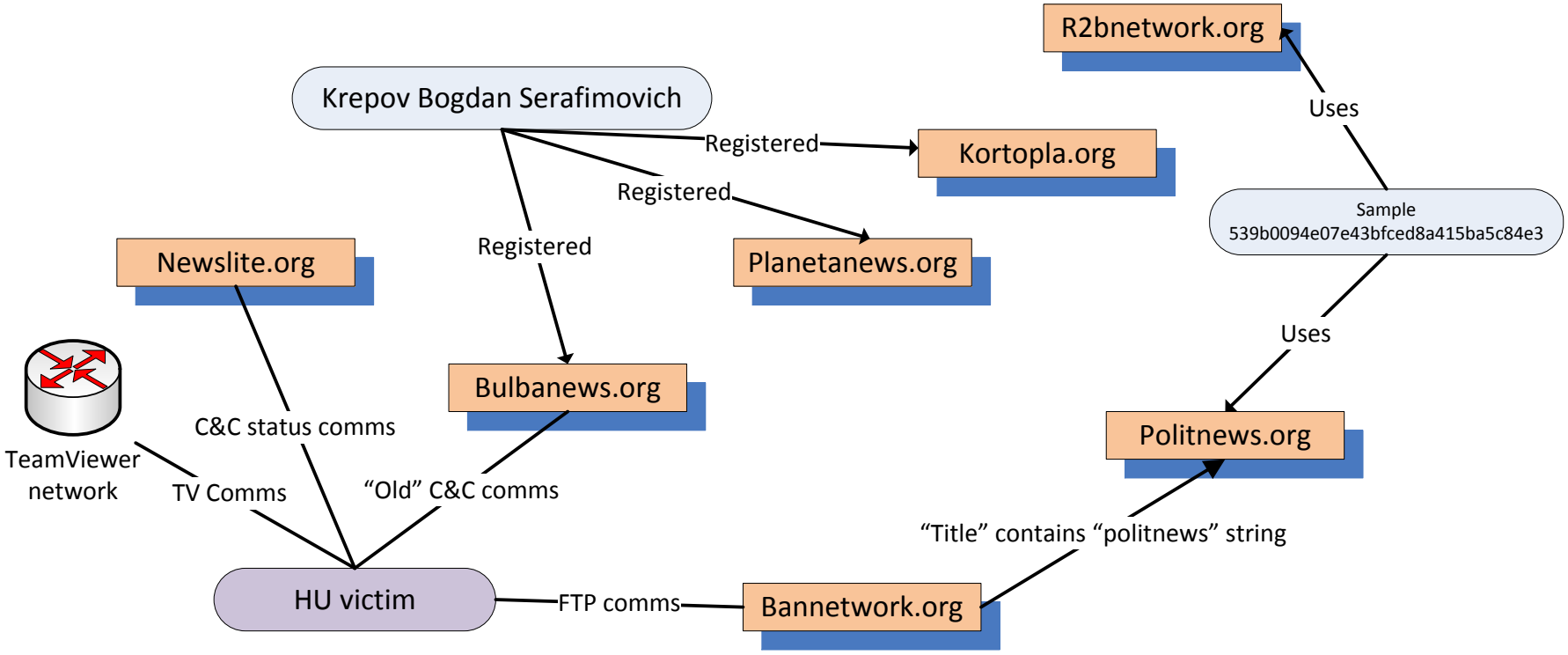
These are either

hacked sites or ***paid servers***

Example: Hacked sites for MiniDuke's C2

Attack location	C&C server	C&C IP / location
Hungary	arabooks.ch	194.38.160.153 / Switzerland
Luxembourg	artas.org	95.128.72.24 / France
Belgium	tsoftonline.com	72.34.47.186 / United States
(Multiple)	www.eamtm.com	188.40.99.143 / Germany

Example: Paid C2 servers of Teamspy



Trend: Stealthiness influences the C2 infrastructure

Stealthiness influences the number of victims per C2

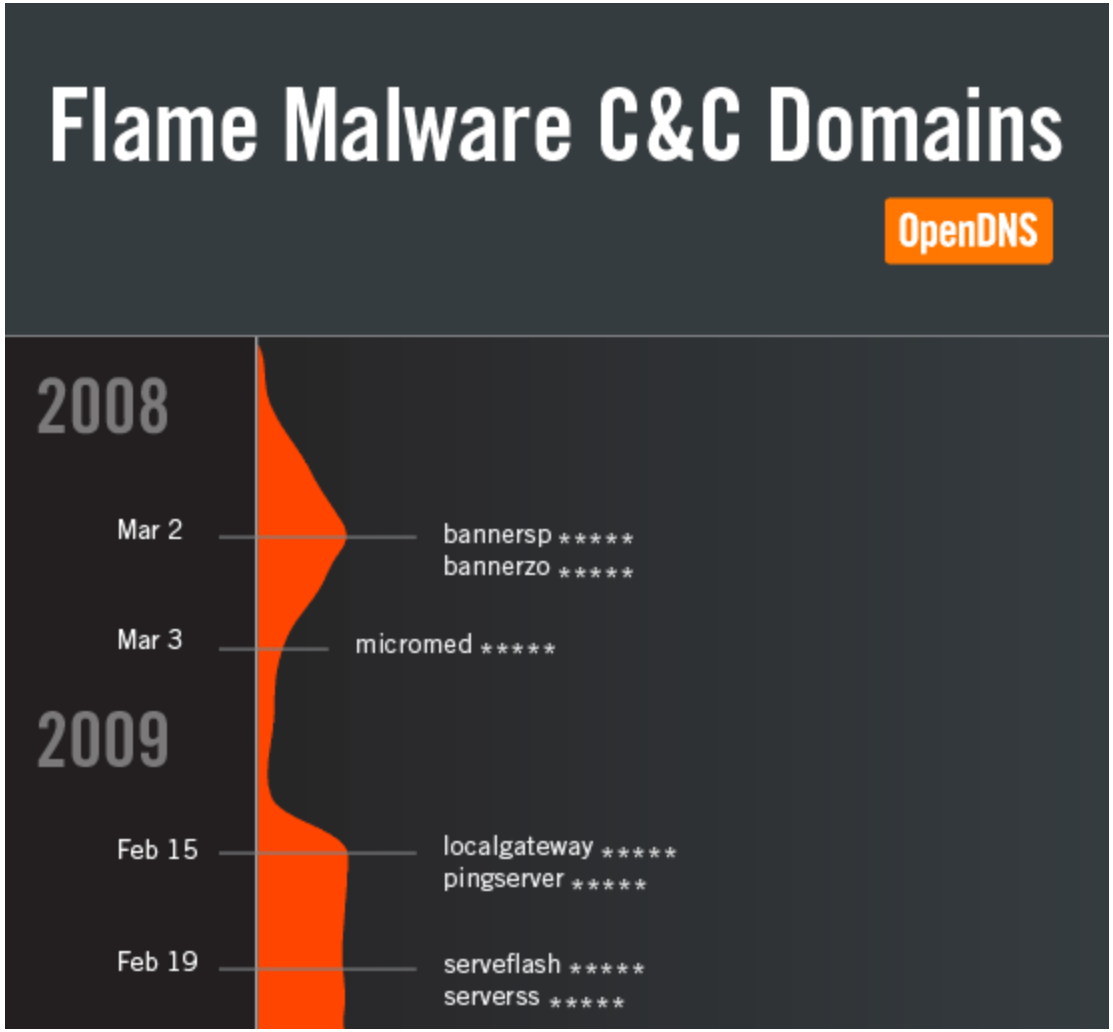
E.g.,

Duqu: ~20 / 5+ → 2-3

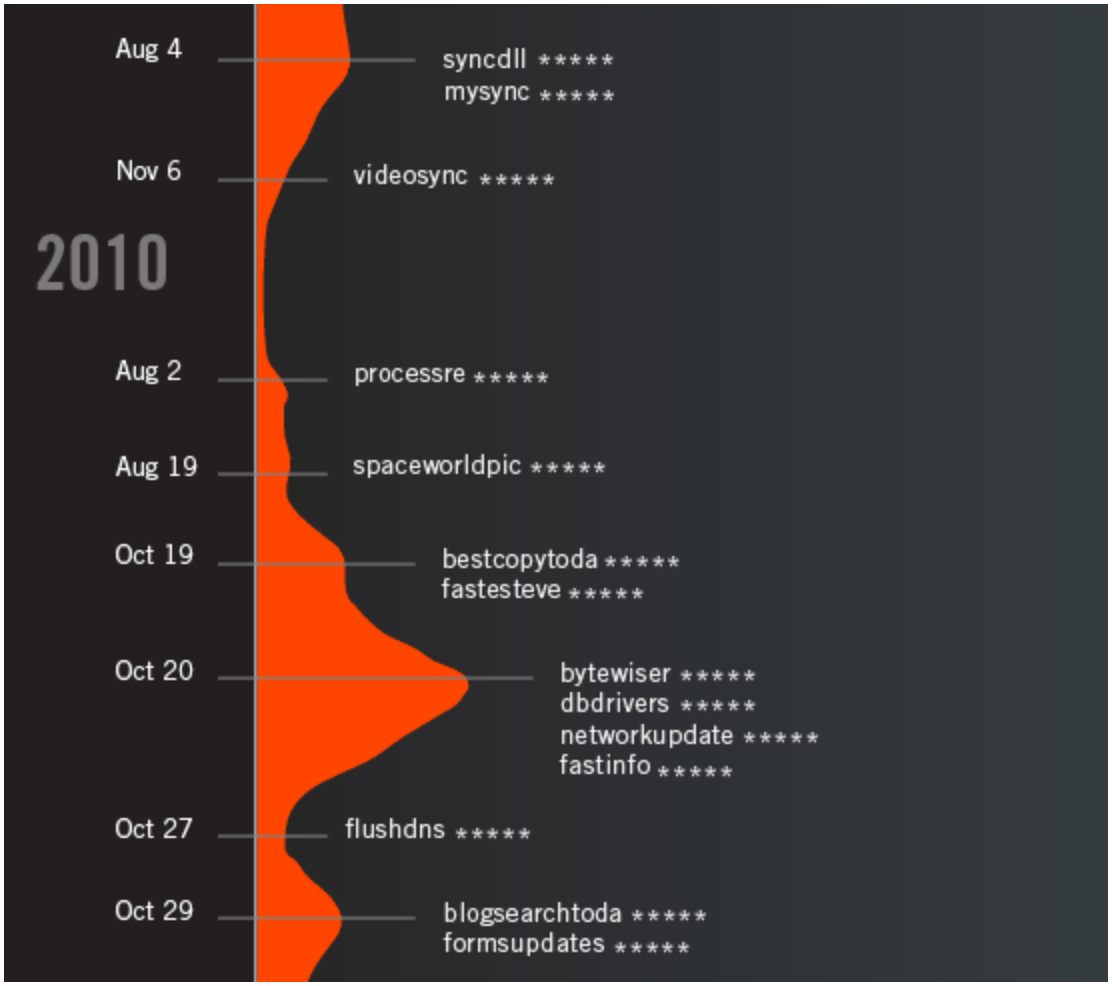
Teamspy: ~1000 / 10+ → 100

Flame: ~10000 / ~100 → ~100

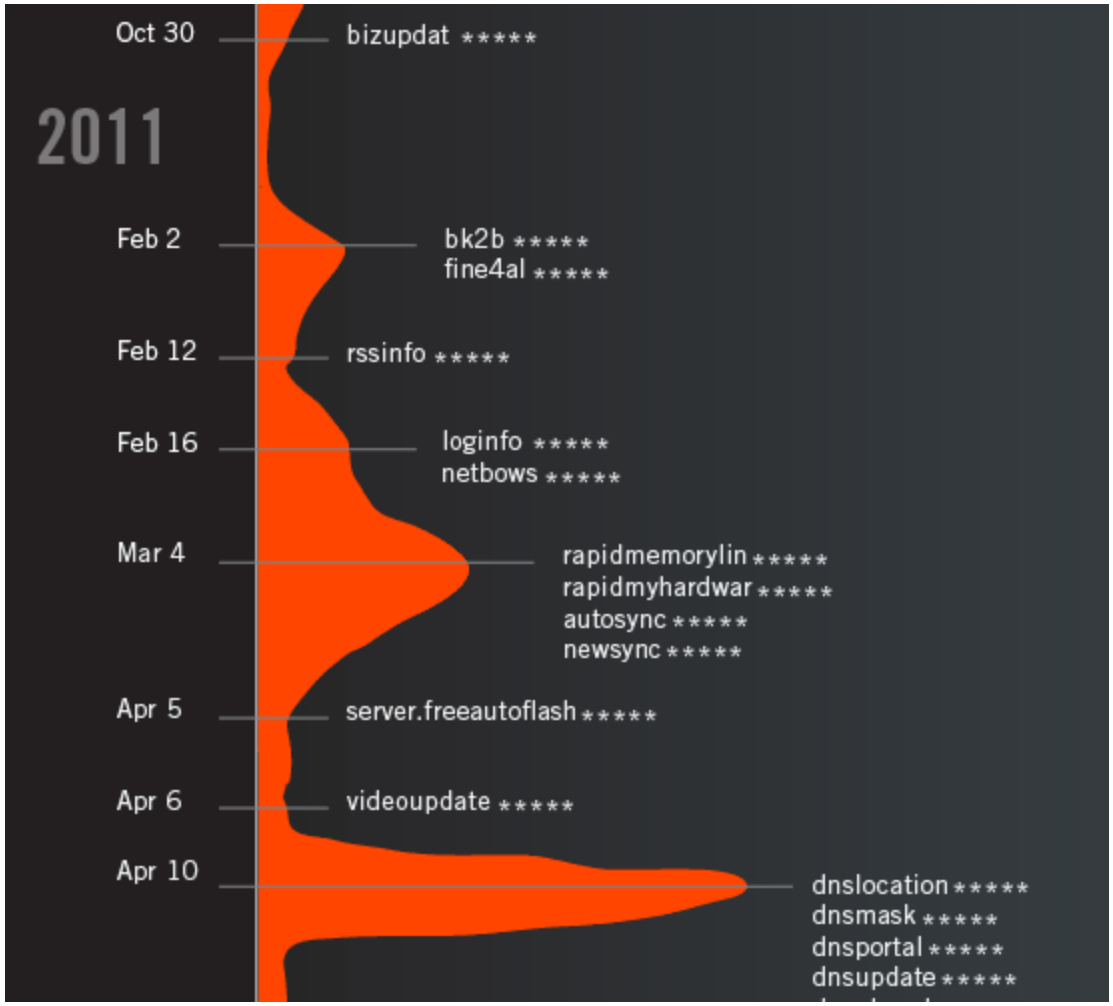
Example: Flame C&Cs – Kaspersky + OpenDNS



Example: Flame C&Cs – Kaspersky + OpenDNS



Example: Flame C&Cs – Kaspersky + OpenDNS



Example: Flame C&Cs – Kaspersky + OpenDNS

Example: Flame C&Cs – Kaspersky + OpenDNS

CODE SIGNING

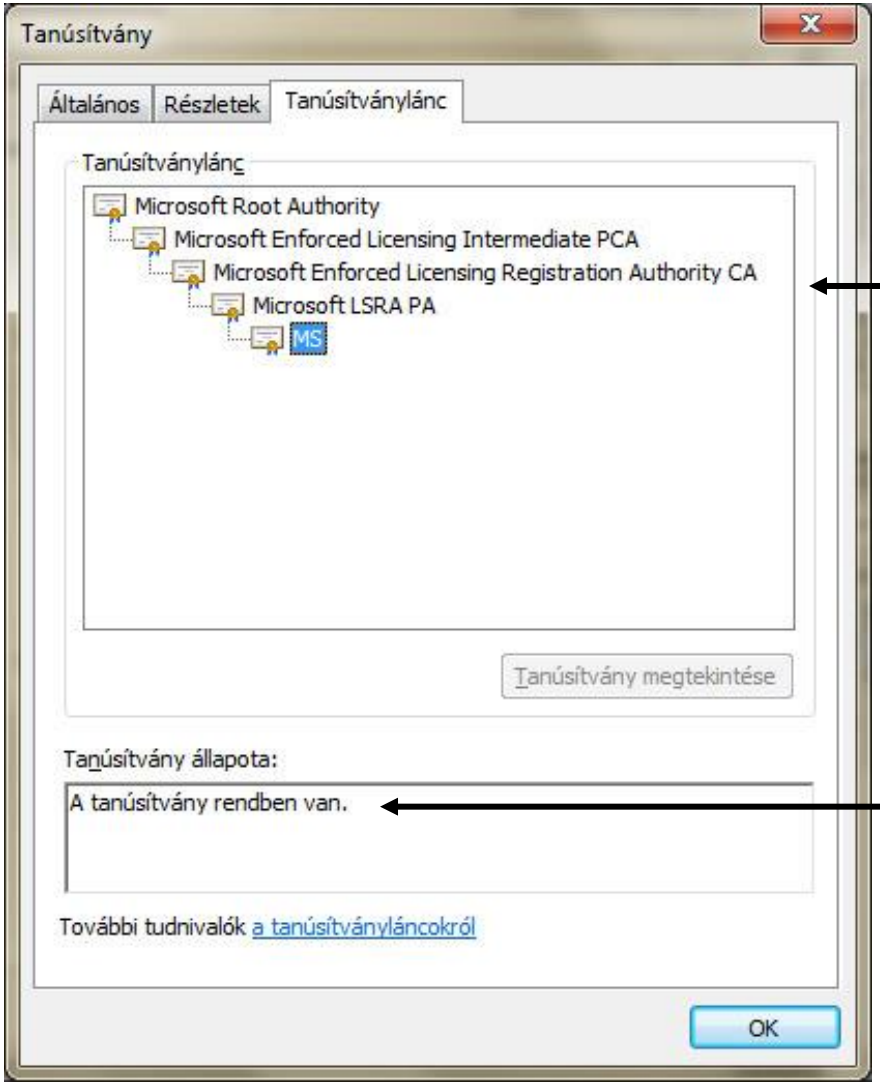
Trend: Evading digital signatures

Professional attackers enforce

valid code signing

Other groups simply ***evade*** verification

Example: The fake (but valid) cert. of Flame

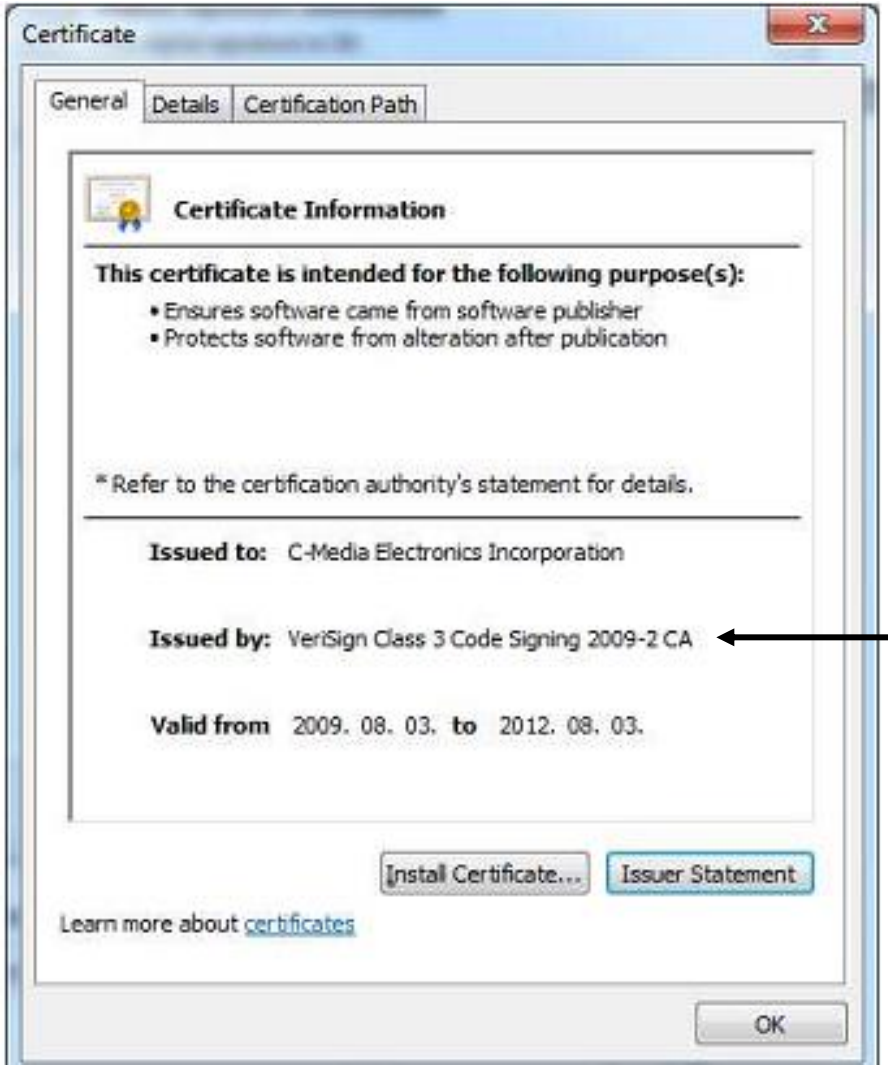


chains up to the MS root

can be used for code signing!

looks valid

Example: The valid certificate used by Duqu



Issued by: VeriSign

Example: Evading verification by DLL preloading

DLL preloading allows for

automatic DLL file load

Original DLL is replaced on the disk

Example: Evading verification by DLL preloading

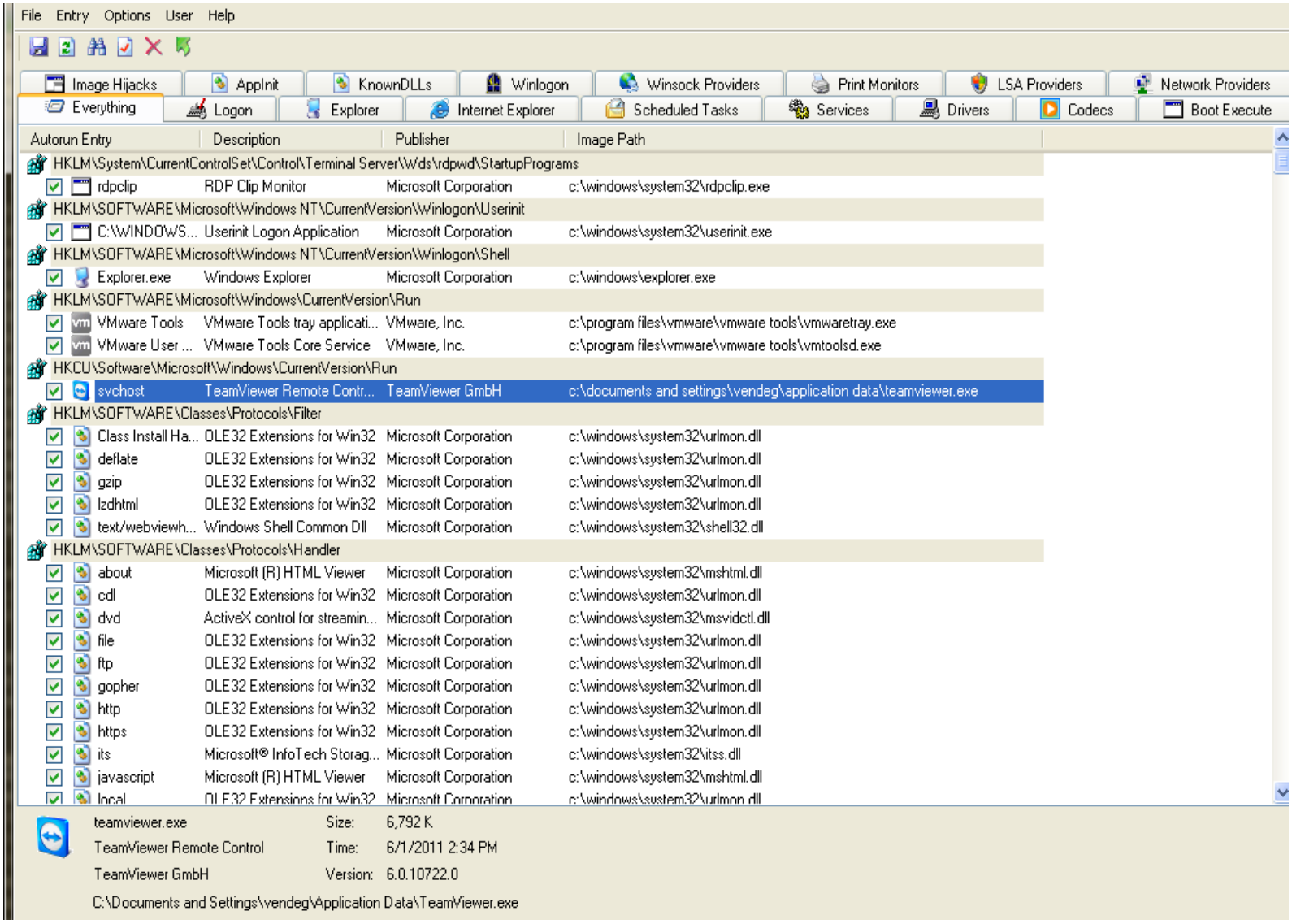
Digital signatures are ***verified*** on ***main executable*** but, not on loadable modules

E.g.,

EvilGrab

Teamspy (advapi32.dll)

Example: Teamspy DLL in signed Teamviewer

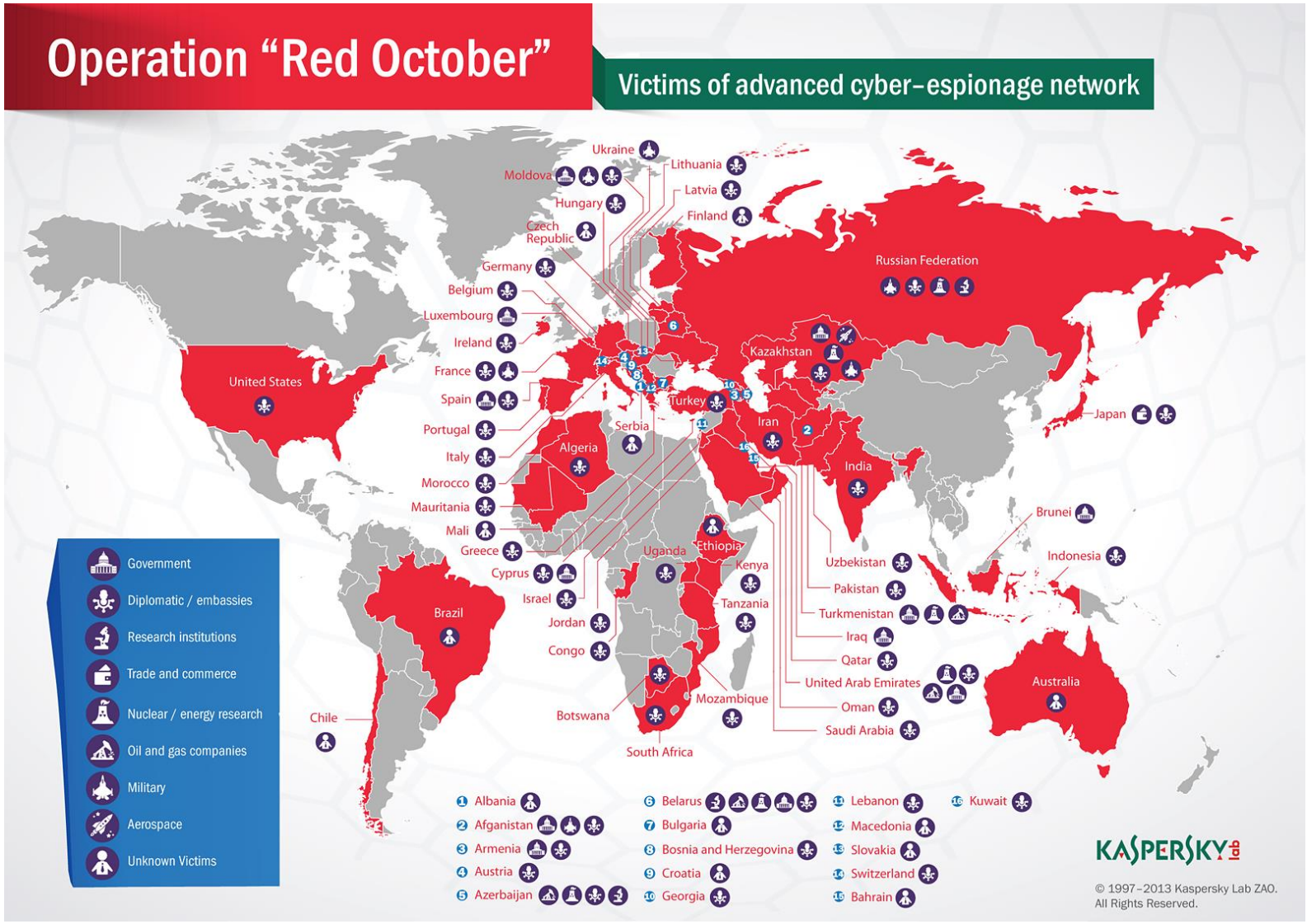


HIGH PROFILE TARGETS

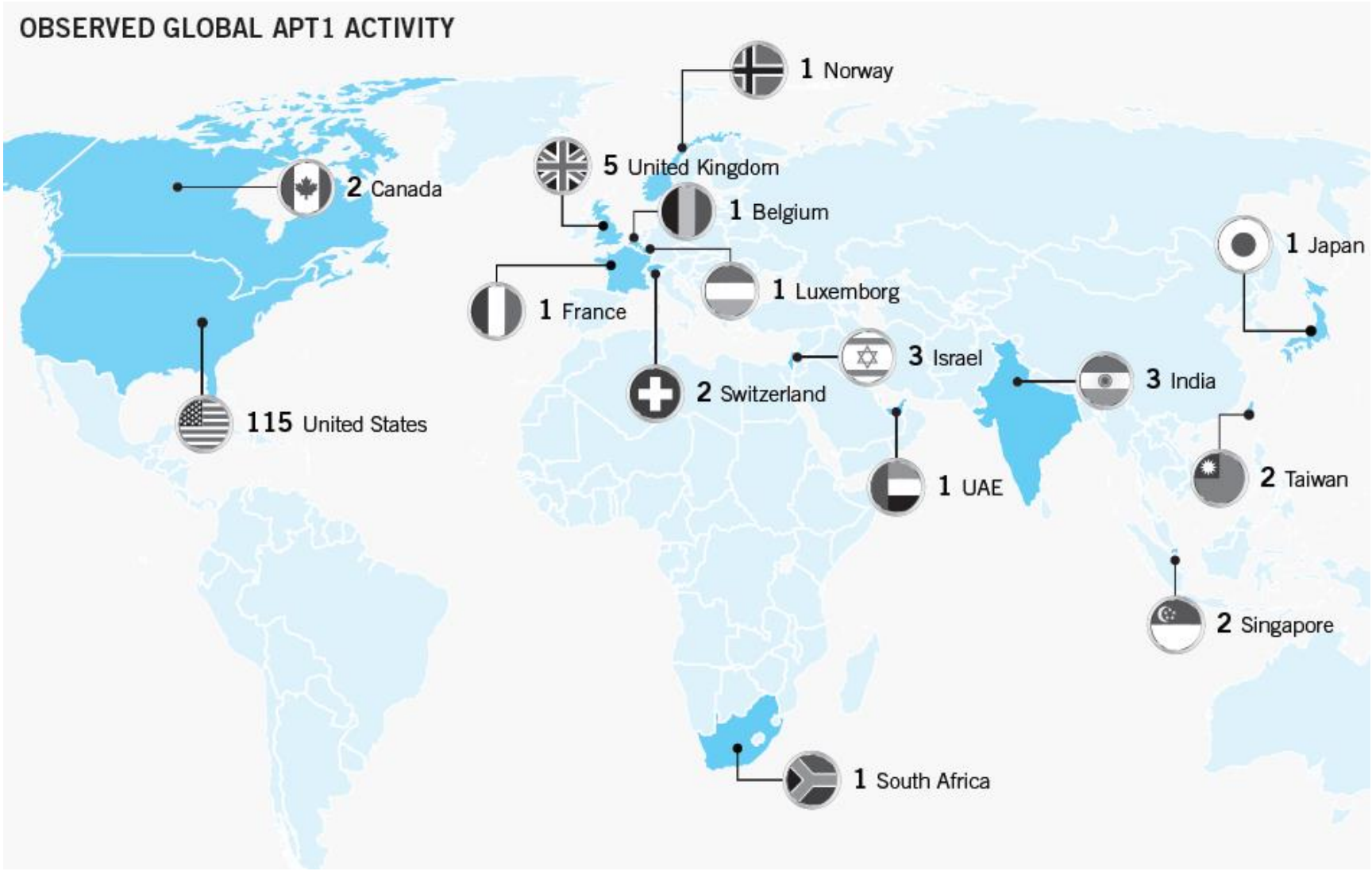
Trend: Everybody can be a target

All sectors are infected

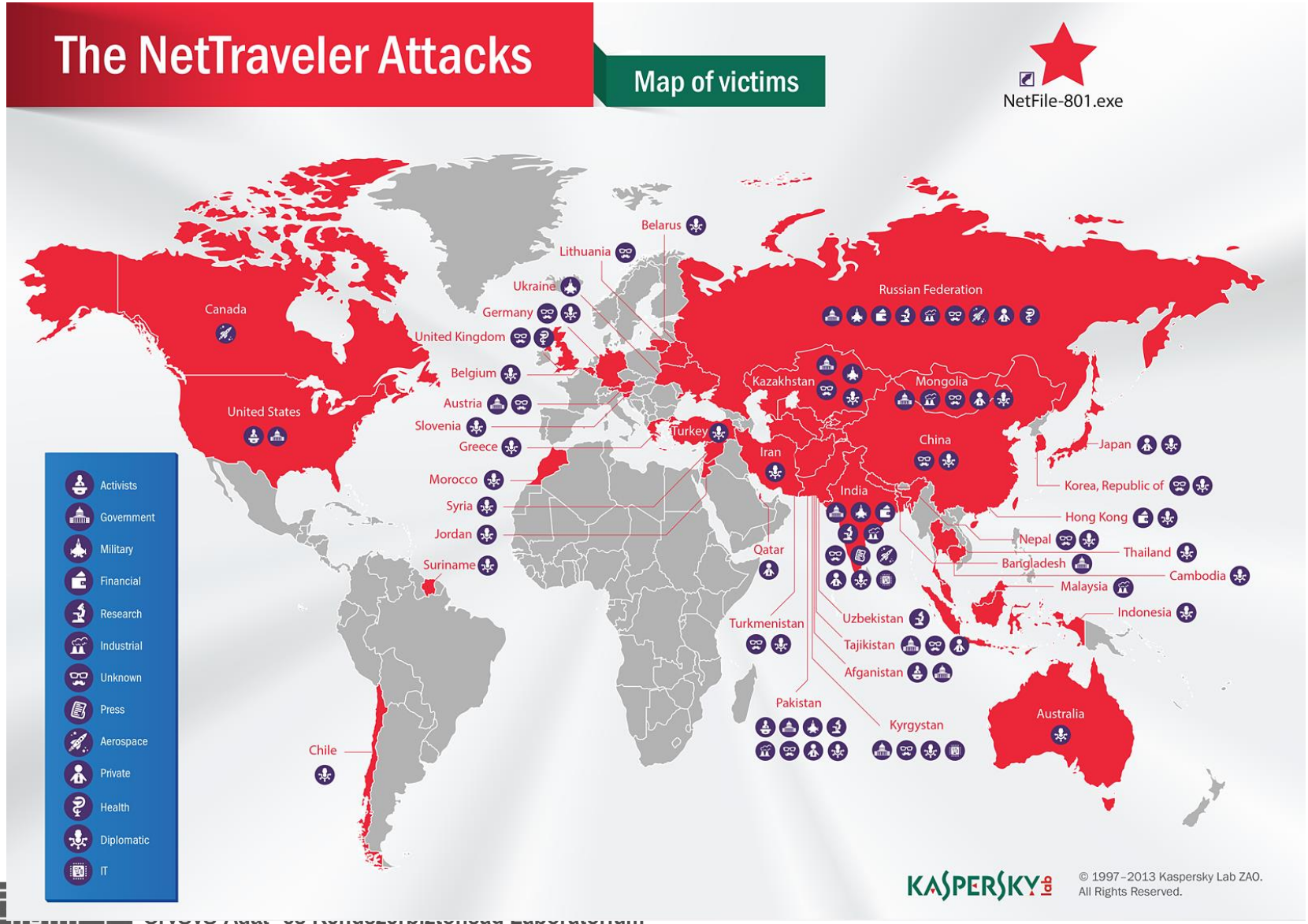
Red October: Jan 2013 (report from Kaspersky)



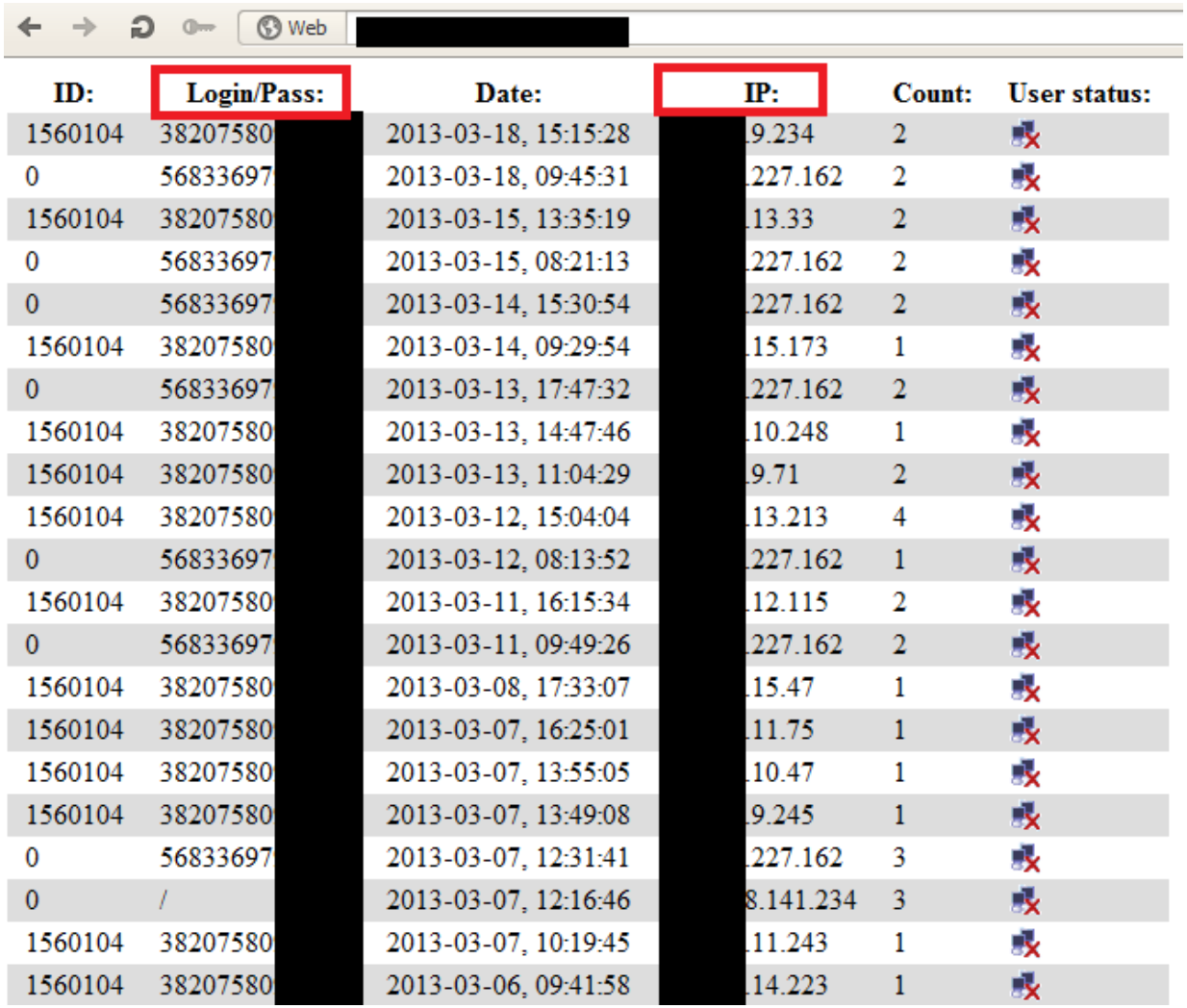
APT1 victims (report from Mandiant)



Nettraveler: June 2013 (report from Kaspersky)



TeamViewer dashboard at the attackers



Web [redacted]

ID:	Login/Pass:	Date:	IP:	Count:	User status:
1560104	38207580 [redacted]	2013-03-18, 15:15:28	[redacted] 9.234	2	
0	56833697 [redacted]	2013-03-18, 09:45:31	[redacted] 227.162	2	
1560104	38207580 [redacted]	2013-03-15, 13:35:19	[redacted] 13.33	2	
0	56833697 [redacted]	2013-03-15, 08:21:13	[redacted] 227.162	2	
0	56833697 [redacted]	2013-03-14, 15:30:54	[redacted] 227.162	2	
1560104	38207580 [redacted]	2013-03-14, 09:29:54	[redacted] 15.173	1	
0	56833697 [redacted]	2013-03-13, 17:47:32	[redacted] 227.162	2	
1560104	38207580 [redacted]	2013-03-13, 14:47:46	[redacted] 10.248	1	
1560104	38207580 [redacted]	2013-03-13, 11:04:29	[redacted] 9.71	2	
1560104	38207580 [redacted]	2013-03-12, 15:04:04	[redacted] 13.213	4	
0	56833697 [redacted]	2013-03-12, 08:13:52	[redacted] 227.162	1	
1560104	38207580 [redacted]	2013-03-11, 16:15:34	[redacted] 12.115	2	
0	56833697 [redacted]	2013-03-11, 09:49:26	[redacted] 227.162	2	
1560104	38207580 [redacted]	2013-03-08, 17:33:07	[redacted] 15.47	1	
1560104	38207580 [redacted]	2013-03-07, 16:25:01	[redacted] 11.75	1	
1560104	38207580 [redacted]	2013-03-07, 13:55:05	[redacted] 10.47	1	
1560104	38207580 [redacted]	2013-03-07, 13:49:08	[redacted] 9.245	1	
0	56833697 [redacted]	2013-03-07, 12:31:41	[redacted] 227.162	3	
0	/ [redacted]	2013-03-07, 12:16:46	[redacted] 8.141.234	3	
1560104	38207580 [redacted]	2013-03-07, 10:19:45	[redacted] 11.243	1	
1560104	38207580 [redacted]	2013-03-06, 09:41:58	[redacted] 14.223	1	

PERSISTENCY AND STEALTHINESS

Trend: Tradeoff between persistency and stealthiness

Tradeoff between ***persistency*** and ***stealthiness***

Examples: Duqu and Flame

Duqu ***erases*** itself after ***36 days***
in any circumstances

Flame ***erases*** itself
when the machine is under
heavy load or ***freezes***
(detection of AVs is default)

GOAL OF ATTACKS

Trend: Information stealing or data destruction

The goal of targetted attacks is either

information stealing

(E.g., Duqu, Flame, Red October etc)

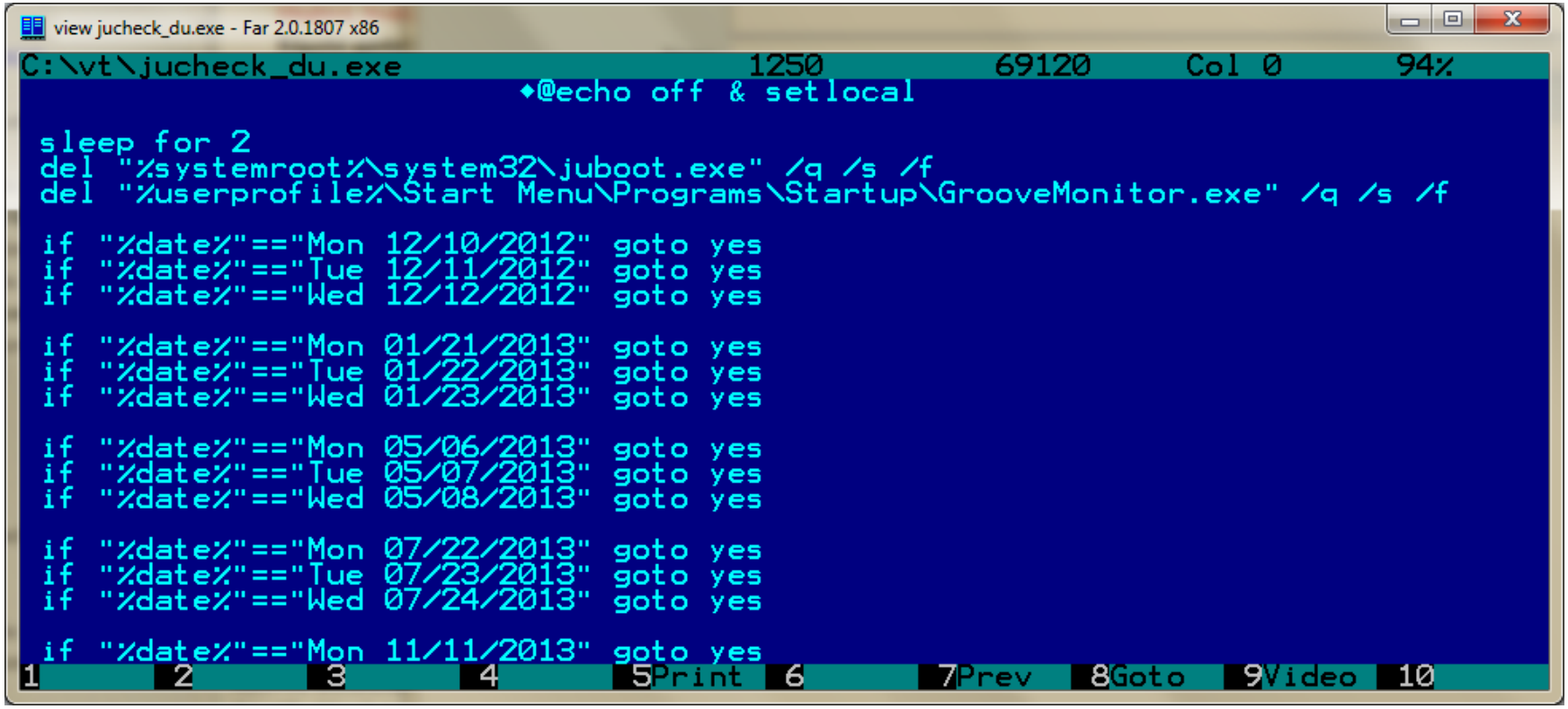
or

data destruction

(E.g., batchwiper, Trojan.Korhigh, Shamoon)

Example: Data destruction by Batchwiper

Simple bat file compiled with BAT2EXE



CONCLUSION AND FUTURE

Conclusions

Targeted attacks are ***difficult*** to handle

However, protecting against them is

not impossible

Future

What we need is

Learning

Better defense

and

Proper handling of incidents

Questions?

gabor.pek@ 

.hu