Radare2 workshop

Zero Nights

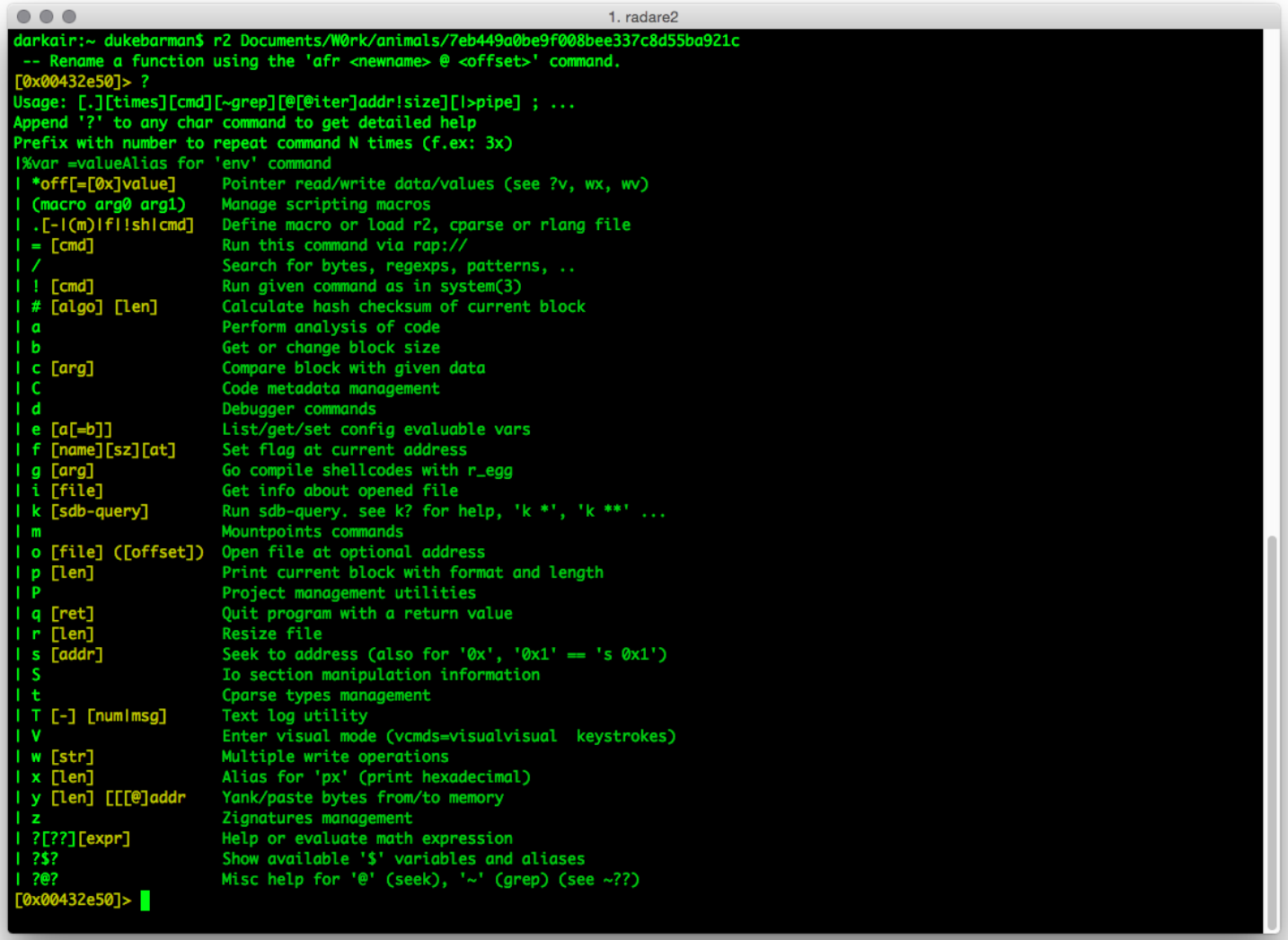
Спикеры

- Антон Кочков (xvilka)
  - Код безопасности
- Борис Рютин (dukebarman)
  - Цифровое оружие и защита (Esagelab)
- Отдельное спасибо Julien Voisin(jvoisin) и Maxim Morine (maijin)

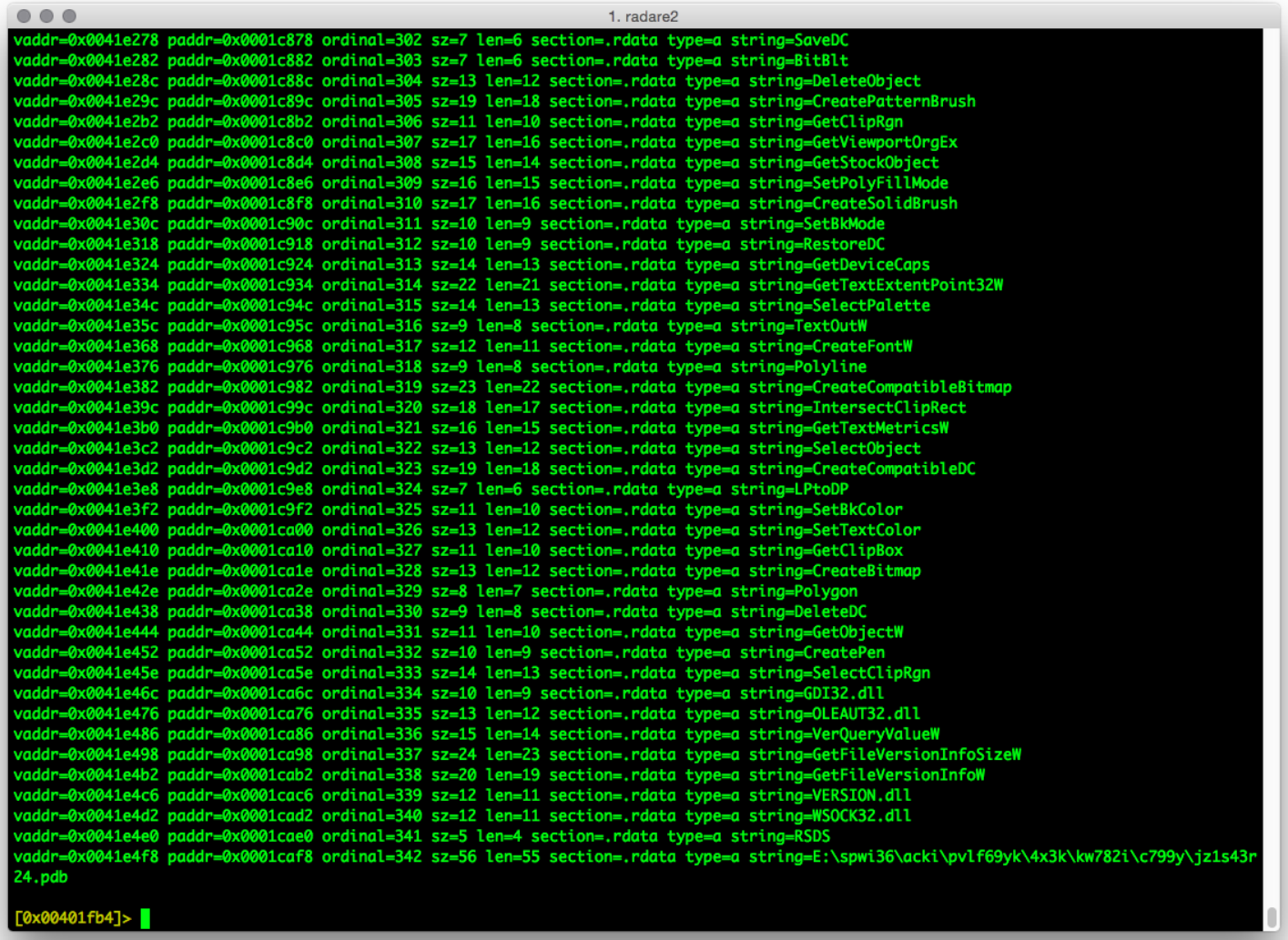
rasm2

```
darkair:~ dukebarman$ rasm2 -a x86 nop
90
darkair:~ dukebarman$ rasm2 -a x86 -d 'eb00'
jmp 2
darkair:~ dukebarman$ rasm2 -w cmpsb
cmp DS:[SI], ES:[edi] (esi++, edi++)
darkair:~ dukebarman$ rasm2 -w sqrtpd
compute square roots of packed double-fp values
darkair:~ dukebarman$
```

Команды в radare2



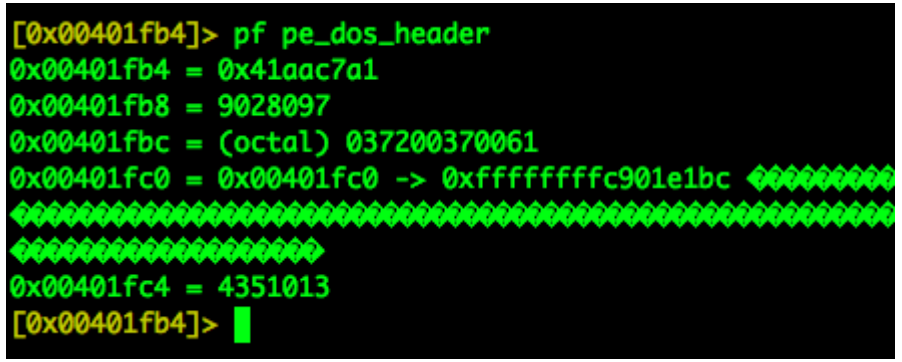
Быстрая обработка строк (iz, izz, rabin2 -h)



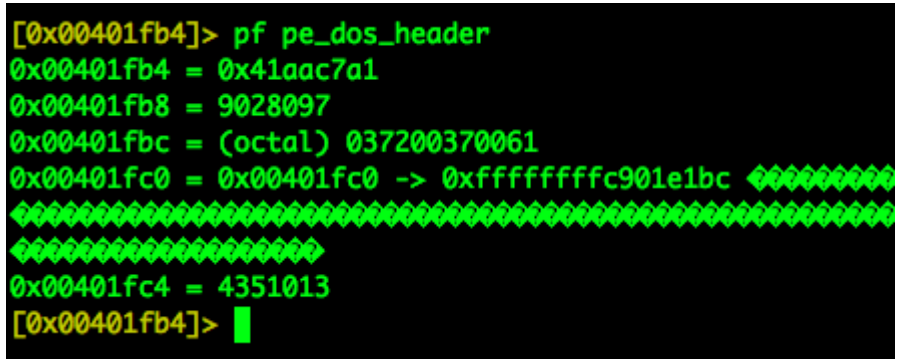
Информация о файле (i?, rabin2 -h)

```
[0x00401fb4]> iS
[Sections]
idx=00 vaddr=0x00401000 paddr=0x00000400 sz=107008 vsz=106592 perm=-r-x name=.text
idx=01 vaddr=0x0041c000 paddr=0x0001a600 sz=9728 vsz=9522 perm=-r-- name=.rdata
idx=02 vaddr=0x0041f000 paddr=0x0001cc00 sz=30208 vsz=49240 perm=-rw- name=.data
idx=03 vaddr=0x0042c000 paddr=0x00024200 sz=3584 vsz=3432 perm=-r-- name=.rsrc
idx=04 vaddr=0x0042d000 paddr=0x00025000 sz=10240 vsz=10044 perm=-r-- name=.reloc

5 sections

[0x00401fb4]>
```

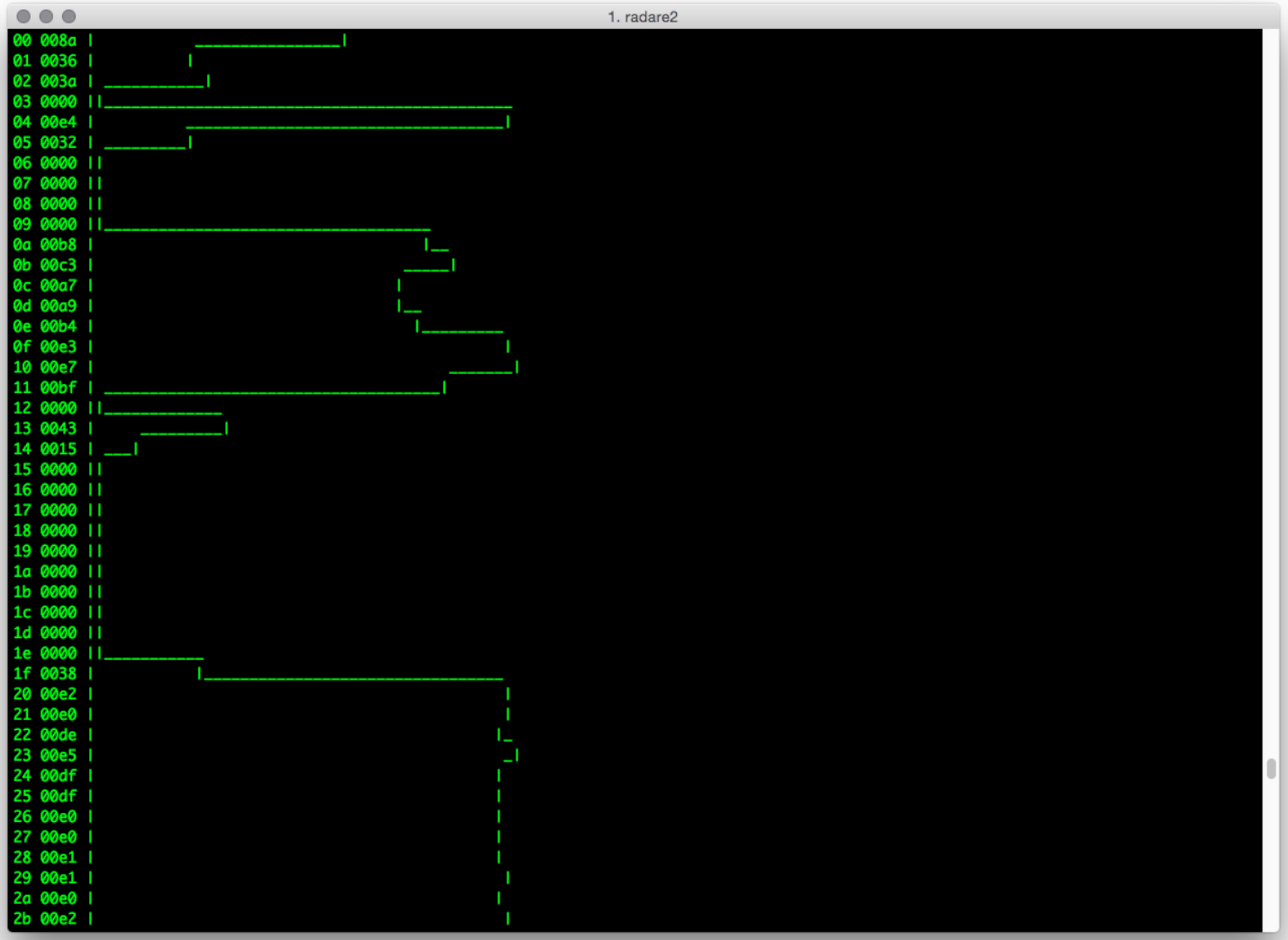
Формат файла (pf?)



Формат файла (pf?)



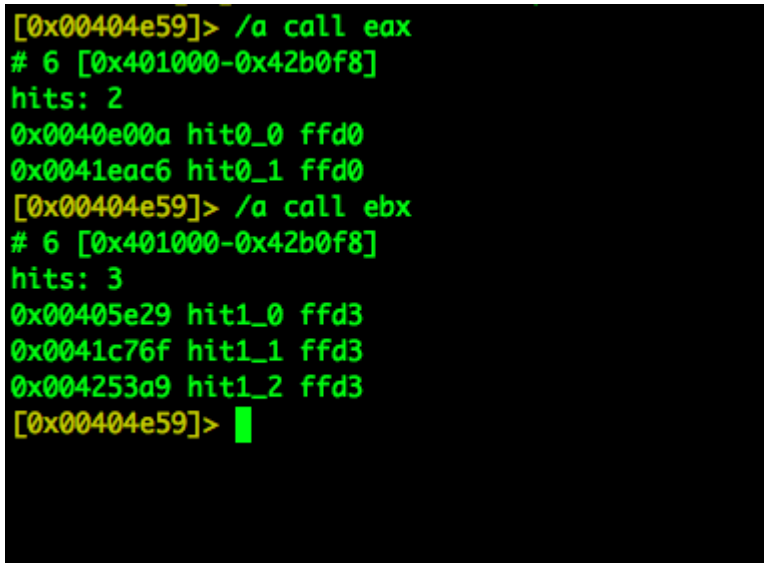
Обнаруживаем упаковщик и шифрование (#?, p=?, yara, /C?, /m?)



Первые шаги (a?, radare2 -A)

```
[0x00404e59]> aa
[0x00404e59]> b 20
[0x00404e59]> pd
/ (fcn) entry0 186
|          0x00404e59    55           push ebp
|          0x00404e5a    8bec         mov ebp, esp
|          0x00404e5c    81ec90080000 sub esp, 0x890
|          0x00404e62    83e000       and eax, 0
|          0x00404e65    89b500ffffff mov dword [ebp - 0x100], esi
|          0x00404e6b    89bdfcfeffff mov dword [ebp - 0x104], edi
|          0x00404e71    899df8feffff mov dword [ebp - 0x108], ebx
|          0x00404e77    8d742418     lea esi, dword [esp + 0x18]
|          0x00404e7b    c70613010000 mov dword [esi], 0x113
|          0x00404e81    896e08       mov dword [esi + 8], ebp ; [:4]=0
|          ; JMP XREF from 0x00404e94 (entry0)
|      .-> 0x00404e84    56           push esi
|      |   0x00404e85    ff1518704200 call dword [reloc.KERNEL32.dll_GetVersionExW_24]
|      |      unk(unk, unk, unk) ; section_end..reloc+-4370945
|      |   0x00404e8b    ff06         inc dword [esi]
|      |   0x00404e8d    36813e18010. cmp dword ss:[esi], 0x118
|      `=< 0x00404e94    72ee         jb 0x404e84
|          0x00404e96    fc           cld
|          0x00404e97    56           push esi
|          0x00404e98    ff1528704200 call dword [reloc.KERNEL32.dll_GetModuleHandleA_40]
|             unk(unk) ; section_end..reloc+-4370945
|          0x00404e9e    c7442404000. mov dword [esp + 4], 0 ; [:4]=0
|          0x00404ea6    6a00         push 0
[0x00404e59]>
```

Базовые команды (/?)



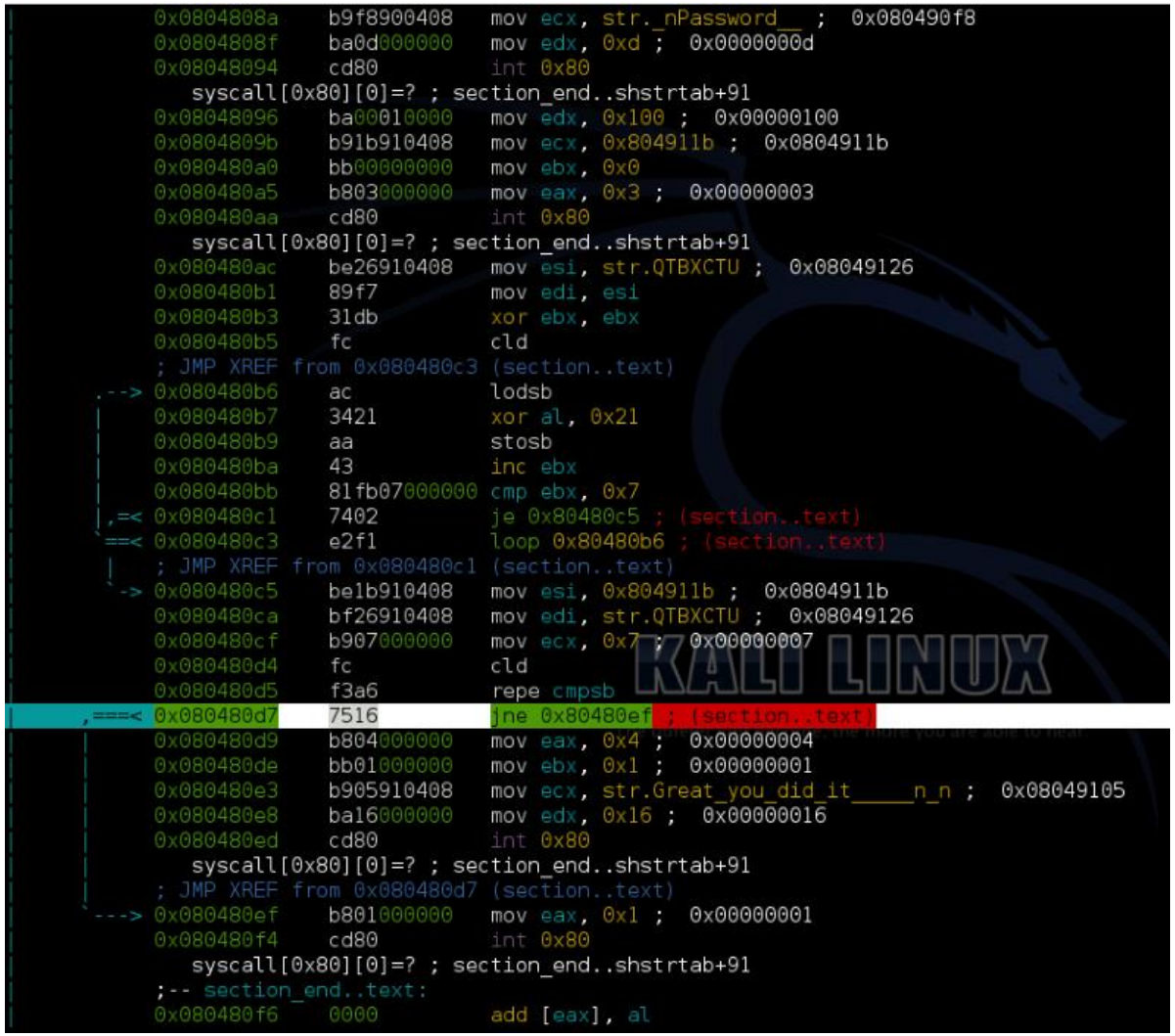
Базовые команды (p?)

```
[0x00404e59]> p?
|Usage: p[=68abcdDfiImrstuxz] [arg|len]
| p=[bep?] [blks]    show entropy/printable chars/chars bars
| p2 [len]           8x8 2bpp-tiles
| p6[de] [len]       base64 decode/encode
| p8 [len]           8bit hexpair list of bytes
| pa[ed] [hex|asm]   assemble (pa) disasm (pad) or esil (pae) from hexpairs
| p[bB] [len]        bitstream of N bytes
| pc[p] [len]        output C (or python) format
| p[dD][lf] [l]      disassemble N opcodes/bytes (see pd?)
| pf[?|.nam] [fmt]   print formatted data (pf.name, pf.name $<expr>)
| p[iI][df] [len]    print N instructions/bytes (f=func) (see pi? and pdi)
| pm [magic]         print libmagic data (pm? for more information)
| pr [len]           print N raw bytes
| p[kK] [len]        print key in randomart (K is for mosaic)
| ps[pwz] [len]      print pascal/wide/zero-terminated strings
| pt[dn?] [len]      print different timestamps
| pu[w] [len]        print N url encoded bytes (w=wide)
| pv[jh] [mode]      bar|json|histogram blocks (mode: e?search.in)
| p[xX][owq] [len]   hexdump of N bytes (o=octal, w=32bit, q=64bit)
| pz [len]           print zoom view (see pz? for help)
| pwd                display current working directory
[0x00404e59]>
```

Базовые команды (w?)

```
[0x00404e59]> w?
|Usage: w[x] [str] [<file] [<<EOF] [@addr]
| wc                list all write changes
| w[1248][+-][n]    increment/decrement byte,word..
| w foobar          write string 'foobar'
| wh r2             whereis/which shell command
| wr 10             write 10 random bytes
| ww foobar         write wide string 'f\x00o\x00o\x00b\x00a\x00r\x00'
| wa push ebp       write opcode, separated by ';' (use '""' around the command)
| waf file          assemble file and write bytes
| wA r 0            alter/modify opcode at current seek (see wA?)
| wb 010203         fill current block with cyclic hexpairs
| wc[ir*?]          write cache undo/commit/reset/list (io.cache)
| wd [off] [n]      duplicate N bytes from offset at current seek (memcpy) (see y?)
| wx 9090           write two intel nops
| wv eip+34         write 32-64 bit value
| wo? hex           write in block with operation. 'wo?' fmi
| wm f0ff           set binary mask hexpair to be used as cyclic write mask
| ws pstring        write 1 byte for length and then the string
| wf -|file         write contents of file at current offset
| wF -|file         write contents of hexpairs file here
| wp -|file         apply radare patch file. See wp? fmi
| wt file [sz]      write to file (from current seek, blocksize or sz bytes)
[0x00404e59]>
```

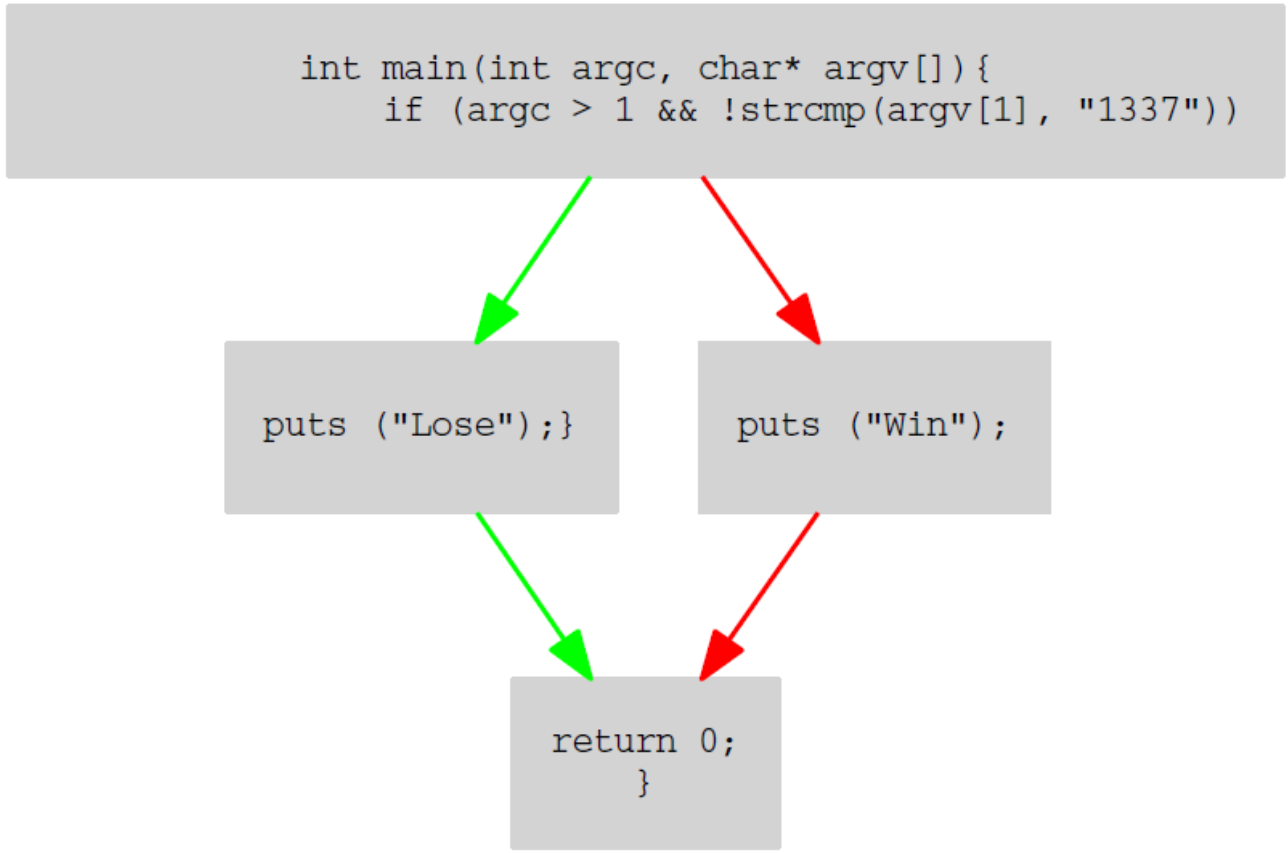
Crackme


0x0804808a b9f8900408 mov ecx, str._nPassword__ ; 0x080490f8
0x0804808f ba0d000000 mov edx, 0xd ; 0x0000000d
0x08048094 cd80 int 0x80
syscall[0x80][0]=? ; section_end..shstrtab+91
0x08048096 ba00010000 mov edx, 0x100 ; 0x00000100
0x0804809b b91b910408 mov ecx, 0x804911b ; 0x0804911b
0x080480a0 bb00000000 mov ebx, 0x0
0x080480a5 b803000000 mov eax, 0x3 ; 0x00000003
0x080480aa cd80 int 0x80
syscall[0x80][0]=? ; section_end..shstrtab+91
0x080480ac be26910408 mov esi, str.QTBXCTU ; 0x08049126
0x080480b1 89f7 mov edi, esi
0x080480b3 31db xor ebx, ebx
0x080480b5 fc cld
; JMP XREF from 0x080480c3 (section..text)
0x080480b6 ac lodsb
0x080480b7 3421 xor al, 0x21
0x080480b9 aa stosb
0x080480ba 43 inc ebx
0x080480bb 81fb07000000 cmp ebx, 0x7
0x080480c1 7402 je 0x80480c5 ; (section..text)
0x080480c3 e2f1 loop 0x80480b6 ; (section..text)
; JMP XREF from 0x080480c1 (section..text)
0x080480c5 be1b910408 mov esi, 0x804911b ; 0x0804911b
0x080480ca bf26910408 mov edi, str.QTBXCTU ; 0x08049126
0x080480cf b907000000 mov ecx, 0x7 ; 0x00000007
0x080480d4 fc cld
0x080480d5 f3a6 repe cmpsb
0x080480d7 7516 jne 0x80480ef ; (section..text)
0x080480d9 b804000000 mov eax, 0x4 ; 0x00000004
0x080480de bb01000000 mov ebx, 0x1 ; 0x00000001
0x080480e3 b905910408 mov ecx, str.Great_you_did_it_____n_n ; 0x08049105
0x080480e8 ba16000000 mov edx, 0x16 ; 0x00000016
0x080480ed cd80 int 0x80
syscall[0x80][0]=? ; section_end..shstrtab+91
; JMP XREF from 0x080480d7 (section..text)
0x080480ef b801000000 mov eax, 0x1 ; 0x00000001
0x080480f4 cd80 int 0x80
syscall[0x80][0]=? ; section_end..shstrtab+91
;-- section_end..text:
0x080480f6 0000 add [eax], al
KALI LINUX

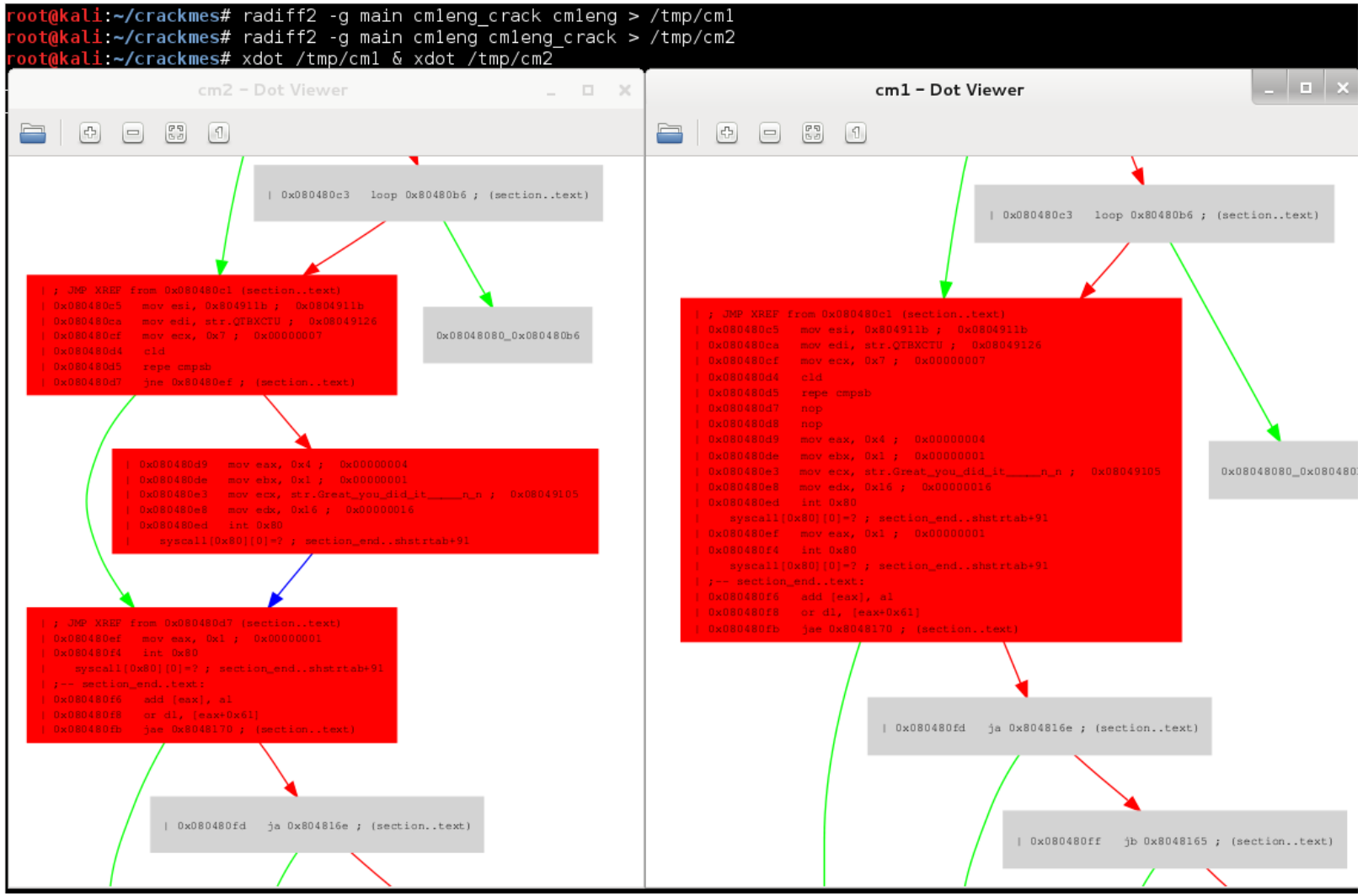
Crackme


[0x080480d7]> wx eb00
[0x080480d7]> pd 10
,=< 0x080480d7 eb00 jmp 0x80480d9
`-> 0x080480d9 b804000000 mov eax, 0x4 ; 0x00000004
0x080480de bb01000000 mov ebx, 0x1 ; 0x00000001
0x080480e3 b905910408 mov ecx, str.Great_you_did_it____n_n ; 0x08049105
0x080480e8 ba16000000 mov edx, 0x16 ; 0x00000016
0x080480ed cd80 int 0x80
syscall[0x80][0]=? ; section_end..shstrtab
0x080480ef b801000000 mov eax, 0x1 ; 0x00000001
0x080480f4 cd80 int 0x80
syscall[0x80][0]=? ; section_end..shstrtab
;-- section_end..text:
0x080480f6 0000 add [eax], al
0x080480f8 0a5061 or dl, [eax+0x61]
[0x080480d7]> q
root@kali:~/crackmes# ./cm1eng_crack
Password : dukebarman
Great you did it !:)

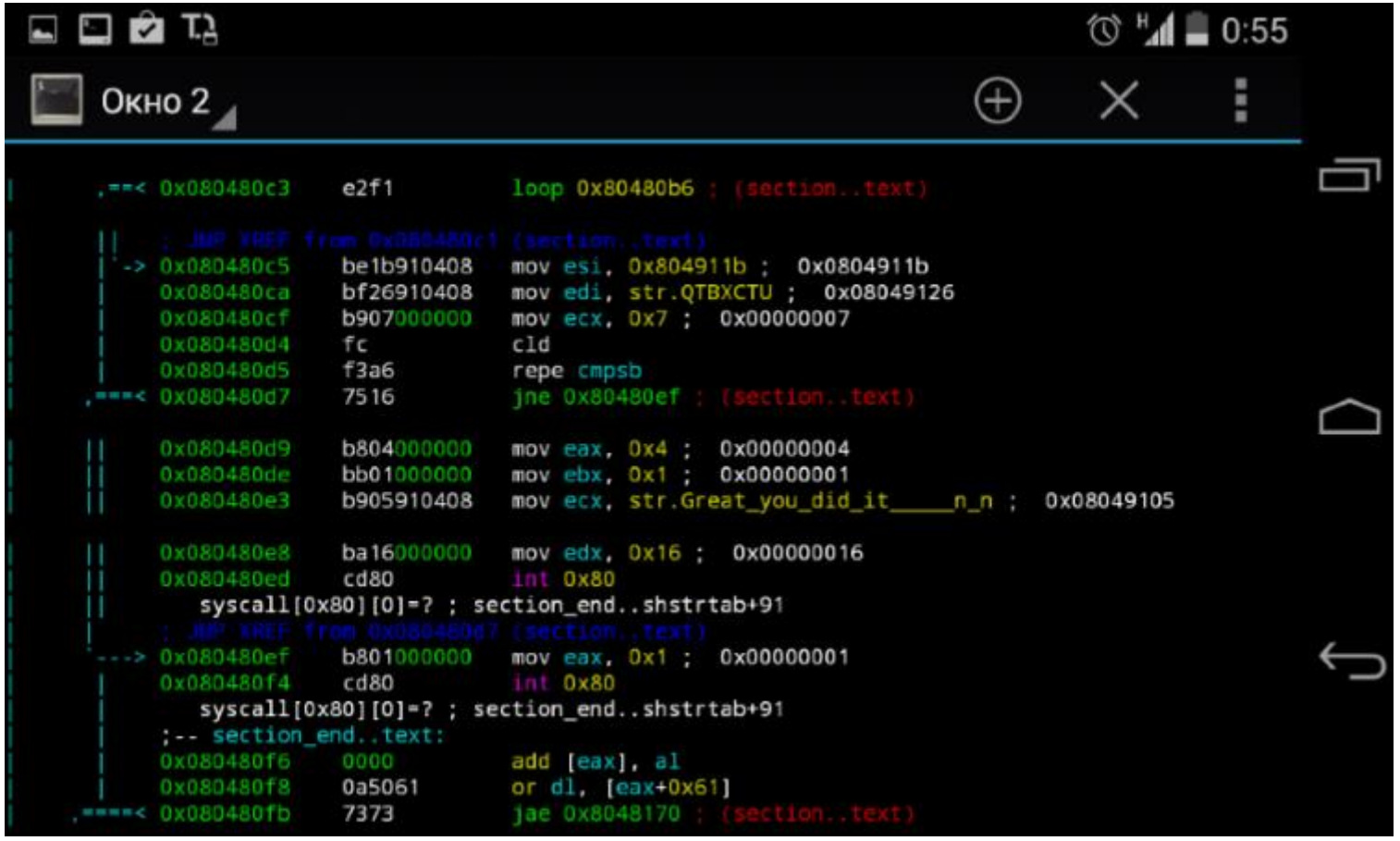
Graph (a?)


int main(int argc, char* argv[]){
if (argc > 1 && !strcmp(argv[1], "1337"))
puts ("Lose");}
puts ("Win");
return 0;
}

Graph (a?)


root@kali:~/crackmes# radiff2 -g main cm1eng_crack cm1eng > /tmp/cm1
root@kali:~/crackmes# radiff2 -g main cm1eng cm1eng_crack > /tmp/cm2
root@kali:~/crackmes# xdot /tmp/cm1 & xdot /tmp/cm2
cm2 – Dot Viewer
cm1 – Dot Viewer
| 0x080480c3 loop 0x80480b6 ; (section..text)
| ; JMP XREF from 0x080480c1 (section..text)
| 0x080480c5 mov esi, 0x804911b ; 0x0804911b
| 0x080480ca mov edi, str.QTBXCTU ; 0x08049126
| 0x080480cf mov ecx, 0x7 ; 0x00000007
| 0x080480d4 cld
| 0x080480d5 repe cmpsb
| 0x080480d7 jne 0x80480ef ; (section..text)
0x08048080_0x080480b6
| 0x080480d9 mov eax, 0x4 ; 0x00000004
| 0x080480de mov ebx, 0x1 ; 0x00000001
| 0x080480e3 mov ecx, str.Great_you_did_it_____n_n ; 0x08049105
| 0x080480e8 mov edx, 0x16 ; 0x00000016
| 0x080480ed int 0x80
| syscall[0x80][0]=? ; section_end..shstrtab+91
| ; JMP XREF from 0x080480d7 (section..text)
| 0x080480ef mov eax, 0x1 ; 0x00000001
| 0x080480f4 int 0x80
| syscall[0x80][0]=? ; section_end..shstrtab+91
| ;-- section_end..text:
| 0x080480f6 add [eax], al
| 0x080480f8 or dl, [eax+0x61]
| 0x080480fb jae 0x8048170 ; (section..text)
| 0x080480fd ja 0x804816e ; (section..text)
| 0x080480ff jb 0x8048165 ; (section..text)

Android


0:55
Окно 2
,==< 0x080480c3 e2f1 loop 0x80480b6 ; (section..text)
; JMP XREF from 0x080480c1 (section..text)
-> 0x080480c5 be1b910408 mov esi, 0x804911b ; 0x0804911b
0x080480ca bf26910408 mov edi, str.QTBXCTU ; 0x08049126
0x080480cf b907000000 mov ecx, 0x7 ; 0x00000007
0x080480d4 fc cld
0x080480d5 f3a6 repe cmpsb
,===< 0x080480d7 7516 jne 0x80480ef ; (section..text)
0x080480d9 b804000000 mov eax, 0x4 ; 0x00000004
0x080480de bb01000000 mov ebx, 0x1 ; 0x00000001
0x080480e3 b905910408 mov ecx, str.Great_you_did_it_____n_n ; 0x08049105
0x080480e8 ba16000000 mov edx, 0x16 ; 0x00000016
0x080480ed cd80 int 0x80
syscall[0x80][0]=? ; section_end..shstrtab+91
; JMP XREF from 0x080480d7 (section..text)
---> 0x080480ef b801000000 mov eax, 0x1 ; 0x00000001
0x080480f4 cd80 int 0x80
syscall[0x80][0]=? ; section_end..shstrtab+91
;-- section_end..text:
0x080480f6 0000 add [eax], al
0x080480f8 0a5061 or dl, [eax+0x61]
,====< 0x080480fb 7373 jae 0x8048170 ; (section..text)

Время для практики

r2 для разработки эксплойтов

r2 для анализа прошивок

Ресурсы

- Github repo
- Official website
- The r2 blog
- The r2 book
- Установка r2
- Twitter
- BM для воркшопа