R2Clutch

Patching iOS binaries

Alex Soler
(murphy)
@asOler

#Whoami

- Álex Soler (murphy)
  @as0ler
  - Red Team member in

akcent cybersecurity

- Barcelona Cybersecurity Meetups

Why i am here

- Let's see a real example of...
- - how easy is to learn and use r2.
  - how to create your own r2-based tool.
- This is an example about how to use r2 during iOS apps security assessments.

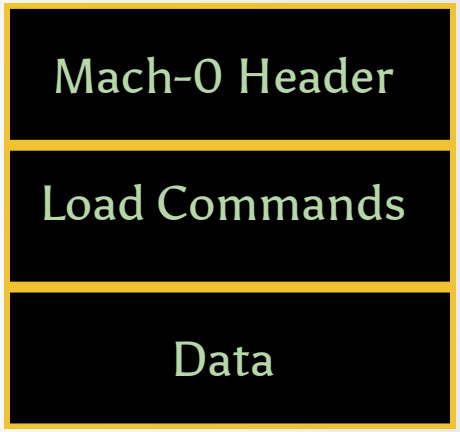
Index

- iOS Binary format
- iOS Security Architecture
- r2 on an iPhone
- Parsing and Patching
- Automating the process

iOS Binary Format

Mach-O File Layout

- **Header**: general information about the binary.
- **Load Commands**: kind of table of contents.
- **Data**: Organized in Segments and Sections.



iOS Binary Format

FatMach-O File Layout

```
struct fat_header {
        uint32_t magic;
        uint32_t nfat_arch; //Bin number
};


struct fat_arch {
  uint32_t cputype;     //Arch
  uint32_t cpusubtype;
  uint32_t offset;      //Bin offset
  uint32_t size;        //BIn size
  uint32_t alighn;
};
```

iOS Binary Format

pf Command

- print/define binary structures

$pf obj=xxdz prev next size name #Define an "obj" struct (hexflag hexflag hex string)

$pf.obj @ <addr> #Apply "obj" struct to addr

$pf. #List all formats

```
[0x00000100]> pf??
|pf: pf[.k[.f[=v]]|[ v]]|[n]|[0][ [sz] fmt] [a0 a1 ...]
| Format:
|  b       byte (unsigned)
|  B       resolve enum bitfield (see t?)
|  c       char (signed byte)
|  d       0x%%08x hexadecimal value (4 bytes)
|  D       disassemble one opcode
|  e       temporally swap endian
|  E       resolve enum name (see t?)
|  f       float value (4 bytes)
|  i       %%i integer value (4 bytes)
|  n       next char specifies size of signed value (1, 2, 4 or 8 byte(s))
|  N       next char specifies size of unsigned value (1, 2, 4 or 8 byte(s))
|  o       0x%%08o octal value (4 byte)
```

iOS Binary Format

FatMach-O Header

```
[0x00000000]> e cfg.bigendian = true
[0x00000000]> pf fatmach0_header=xi magic nfat_arch
[0x00000000]> pf.fatmach0_header @ 0x00
     magic : 0x00000000 = 0xcafebabe
 nfat_arch : 0x00000004 = 2
[0x00000000]> pf fatmach0_arch=xxxix cputype cpusubtype offset size align
[0x00000000]> pf.fatmach0_arch @ $$+`pfs.fatmach0_header~:[0]`
    cputype : 0x00000008 = 0x0000000c
 cpusubtype : 0x0000000c = 0x00000009
     offset : 0x00000010 = 0x00004000
       size : 0x00000014 = 1682640
      align : 0x00000018 = 0x0000000e
[0x00000000]>
```

FAT Struct is in big endian!

```
struct fat_header {
        uint32_t magic;
        uint32_t nfat_arch; //numBin
};
```

```
struct fat_arch {
  uint32_t cputype;     //Arch
  uint32_t cpusubtype;
  uint32_t offset;      //Bin offset
  uint32_t size;        //BIn size
  uint32_t align;
};
```

iOS Binary Format

Mach-O Header

```
[0x00000000]> pf.mach0_header @ mach0_header
      magic : 0x00000000 = 0xfeedface
    cputype : 0x00000004 = 0x0000000c
 cpusubtype : 0x00000008 = 0x00000009
   filetype : 0x0000000c = 0x00000002
      ncmds : 0x00000010 = 34
 sizeofcmds : 0x00000014 = 4160
      flags : 0x00000018 = 0x00200085
[0x00000000]>
```

```
struct mach_header {
   uint32_t magic;
   uint32_t cputype;
   uint32_t cpusubtype;
   uint32_t filetype;
   uint32_t ncmds;
   uint32_t sizeofcmds;
   uint32_t flags;
};
```

iOS Binary Format

Load Commands

```
Load command 0
      cmd LC_SEGMENT
  cmdsize 56
  segname __PAGEZERO
   vmaddr 0x00000000
   vmsize 0x00001000
  fileoff 0
 filesize 0
  maxprot 0x00000000
 initprot 0x00000000
   nsects 0
    flags 0x0
Load command 1
      cmd LC_SEGMENT
  cmdsize 736
  segname __TEXT
   vmaddr 0x00001000
   vmsize 0x0017e000
  fileoff 0
 filesize 1564672
  maxprot 0x00000005
 initprot 0x00000005
   nsects 10
    flags 0x0
```

```
[0x00000100]> pf.mach0_segment @ mach0_segment_0
      cmd : 0x0000001c = 0x00000001
  cmdsize : 0x00000020 = 56
  segname : 0x00000024 = __PAGEZERO
   vmaddr : 0x00000034 = 0x00000000
   vmsize : 0x00000038 = 0x00001000
  fileoff : 0x0000003c = 0x00000000
 filesize : 0x00000040 = 0x00000000
  maxprot : 0x00000044 = (octal)  000000000
 initprot : 0x00000048 = (octal)  000000000
   nsects : 0x0000004c = 0
    flags : 0x00000050 = 0x00000000
[0x00000100]> pf.mach0_segment @ mach0_segment_1
      cmd : 0x00000054 = 0x00000001
  cmdsize : 0x00000058 = 736
  segname : 0x0000005c = __TEXT
   vmaddr : 0x0000006c = 0x00001000
   vmsize : 0x00000070 = 0x0017e000
  fileoff : 0x00000074 = 0x00000000
 filesize : 0x00000078 = 0x0017e000
  maxprot : 0x0000007c = (octal)  000000005
 initprot : 0x00000080 = (octal)  000000005
   nsects : 0x00000084 = 10
    flags : 0x00000088 = 0x00000000
[0x00000100]>
```

```
struct lc {
        uint32_t cmd;
        uint32_t cmdsize;
        //Custom fields
};
```

iOS Binary Format

Interesting sections

- **__TEXT,__text**: Code
- **__TEXT,__cstring**: Constant C strings. A C string is a sequence of non-null bytes that ends with a null byte ('\0')
- **__TEXT,__objc_classname**: Class names
- **__TEXT,__objc_methname**: Class method names

Classdump

```
$rabin2 -c Instagram.arm_32.0
```

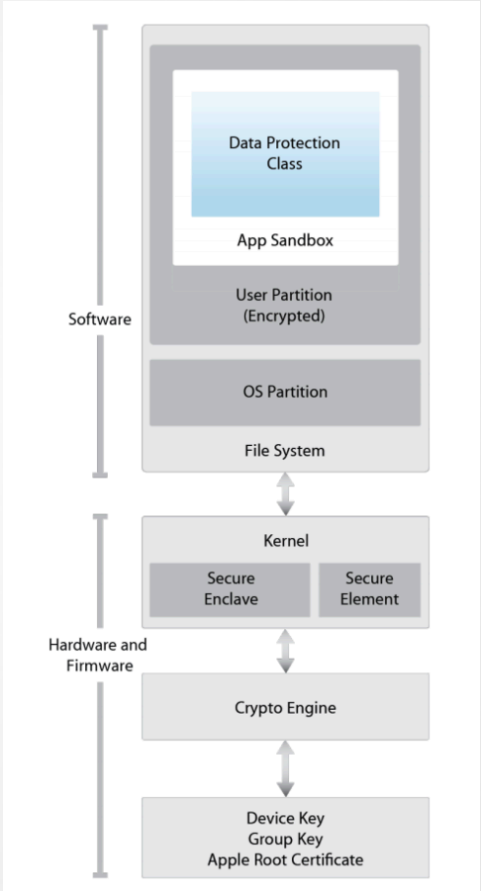
```
[0x100000de0]> iC
```

```
0x00c96248 class 0 IGAlbumCameraPermissionView
0x0011e3b0 method 0 initWithFrame:_8
0x0011ea2a method 1 layoutSubviews_15
0x0011ed48 method 2 refreshViewStateAnimated:
0x0011ef8c method 3 didTapCameraEnable
0x0011efca method 4 didTapMicrophoneEnable
0x0011f07a method 5 .cxx_destruct_42
0x0011f008 method 6 delegate_27
0x0011f026 method 7 setDelegate:_25
0x0011f03a method 8 cameraBlurOverlay
0x0011f04a method 9 textLabel_2
0x0011f05a method 10 cameraButton
0x0011f06a method 11 microphoneButton
```

iOS Security Architecture

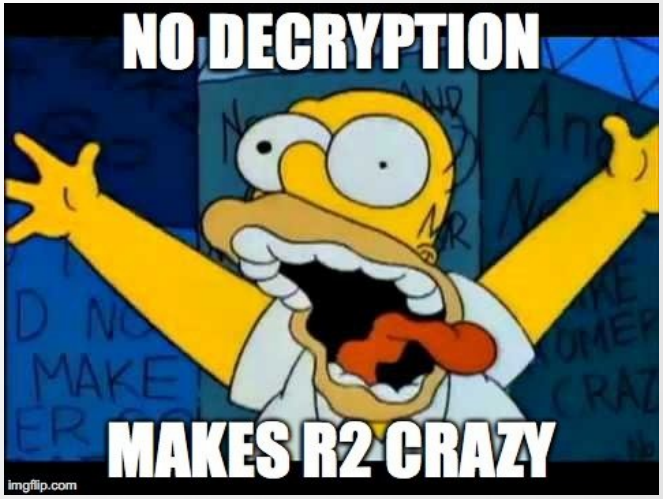
Application Security

- Sandboxing
- non-privileged user
- Encryption
- Address Space Layout Randomization
- Code Signing
- ....



Encryption

- Every application from App Store is encrypted:
- Not permit binary static analysis
- Not possible to see
  - Strings
  - Can't decompile it
  - No Class-dump



iOS Security Architecture

Encrypted decompilation

```
[0x00003860]> pd @ entry0
      |||   ;-- entry0:
      |||   ;-- section.0.__text:
      |||   ;-- func.00003860:
      |||   0x00003860      b361b74e       mrcmi p1, 5, r6, c7, c3, 5  ; [0] va=0x00003860 pa
      |||   0x00003864      b83dc4fe       mcr2 p13, 6, r3, c4, c8, 5
      |||   0x00003868      8c0ab53f       svclo 0xb50a8c
      |||   0x0000386c      245a017b       blvc 0x5a104
      |||   0x00003870      af7d755b       blpl 0x1d62f34
      |||   0x00003874      863a6972       rsbvc r3, sb, 0x86000
      |||   0x00003878      b5355c19       ldmdbne ip, {r0, r2, r4, r5, r7, r8, sl, ip, sp} ^
      |||   0x0000387c      a9341be1       tst fp, sb, lsr 9
     ||||   0x00003880      4b68149d       ldcls p8, c6, [r4, -0x12c]
    ,=====< 0x00003884      70b165c8       stmdagt r5!, {r4, r5, r6, r8, ip, sp, pc} ^
    |||||   0x00003888      efb4f0f9       invalid
    |||||   ;-- func.0000388d:
    |||||   0x0000388c  ~   d5ca8c1b       blne 0xfe3363e8
    |||||   0x00003890      1090           str r0, [sp, 0x40]
    |||||   0x00003892      dd52           strh r5, [r3, r3]
    |||||   0x00003894      3ade           udf 0x3a
    |||||   0x00003896      69ac           add r4, sp, 0x1a4
    |||||   0x00003898      785f           ldrsh r0, [r7, r5]
    ||`===< 0x0000389a      83d6           bvs 0x37a4
    || ||   0x0000389c      5fa0           adr r0, 0x17c
    || ||   0x0000389e      ca39           subs r1, 0xca
    || ||   0x000038a0      1747           bx r2
    || ||   0x000038a2      dd38           subs r0, 0xdd
    `-----> 0x000038a4      a5b5           push {r0, r2, r5, r7, lr}
     | ||   0x000038a6      43e99c38       strd r3, r8, [r3, -0x270]
     ||||   0x000038aa      1667           str r6, [r2, 0x70]
```

iOS Security Architecture

Encrypted Strings

```
[0x000ab81c]> iz
Do you want to print 33437 lines? (y/N)
vaddr=0x00a4bc49 paddr=0x00a47c49 ordinal=000 sz=5 len=4 section=3.__const type=ascii string=\en>d
vaddr=0x00a4bc63 paddr=0x00a47c63 ordinal=001 sz=5 len=4 section=3.__const type=ascii string=px=E
vaddr=0x00a4bc82 paddr=0x00a47c82 ordinal=002 sz=7 len=6 section=3.__const type=ascii string=a-\rXt,
vaddr=0x00a4bcc2 paddr=0x00a47cc2 ordinal=003 sz=8 len=6 section=3.__const type=ascii string=[80?3-1
vaddr=0x00a4bcf5 paddr=0x00a47cf5 ordinal=004 sz=6 len=5 section=3.__const type=ascii string=l]^yH
vaddr=0x00a4bd07 paddr=0x00a47d07 ordinal=005 sz=5 len=4 section=3.__const type=ascii string=LTST
vaddr=0x00a4bd31 paddr=0x00a47d31 ordinal=006 sz=6 len=4 section=3.__const type=ascii string=!;G
vaddr=0x00a4bd49 paddr=0x00a47d49 ordinal=007 sz=5 len=4 section=3.__const type=ascii string=;M\n,
vaddr=0x00a4bd7a paddr=0x00a47d7a ordinal=008 sz=6 len=4 section=3.__const type=ascii string=\rzkK
vaddr=0x00a4bdba paddr=0x00a47dba ordinal=009 sz=5 len=4 section=3.__const type=ascii string=]1JG
vaddr=0x00a4be0f paddr=0x00a47e0f ordinal=010 sz=5 len=4 section=3.__const type=ascii string=~-_V
vaddr=0x00a4be88 paddr=0x00a47e88 ordinal=011 sz=9 len=6 section=3.__const type=ascii string=ou\t l'
vaddr=0x00a4bebd paddr=0x00a47ebd ordinal=012 sz=5 len=4 section=3.__const type=ascii string=+KSw
vaddr=0x00a4bf4b paddr=0x00a47f4b ordinal=013 sz=8 len=6 section=3.__const type=ascii string=mM\RhF
vaddr=0x00a4bf9c paddr=0x00a47f9c ordinal=014 sz=7 len=5 section=3.__const type=ascii string=e\r^j
vaddr=0x00a4bfe8 paddr=0x00a47fe8 ordinal=015 sz=7 len=6 section=3.__const type=ascii string=2#m1f_
vaddr=0x00a4bfef paddr=0x00a47fef ordinal=016 sz=6 len=4 section=3.__const type=ascii string=Ÿ#\r
vaddr=0x00a4c00b paddr=0x00a4800b ordinal=017 sz=6 len=5 section=3.__const type=ascii string=\t"jK\v
vaddr=0x00a4c016 paddr=0x00a48016 ordinal=018 sz=5 len=4 section=3.__const type=ascii string=x\t\be
vaddr=0x00a4c01b paddr=0x00a4801b ordinal=019 sz=7 len=5 section=3.__const type=ascii string=Nv}C
vaddr=0x00a4c05a paddr=0x00a4805a ordinal=020 sz=7 len=4 section=3.__const type=ascii string=h:;
vaddr=0x00a4c0ae paddr=0x00a480ae ordinal=021 sz=7 len=5 section=3.__const type=ascii string=XR*
vaddr=0x00a4c0fb paddr=0x00a480fb ordinal=022 sz=6 len=4 section=3.__const type=ascii string=*uQ'
vaddr=0x00a4c13b paddr=0x00a4813b ordinal=023 sz=7 len=6 section=3.__const type=ascii string=C3Dj9\f
vaddr=0x00a4c1ae paddr=0x00a481ae ordinal=024 sz=6 len=4 section=3.__const type=ascii string=m/{
vaddr=0x00a4c1c0 paddr=0x00a481c0 ordinal=025 sz=6 len=4 section=3.__const type=ascii string=Ns
```

Encrypted Classnames

```
iPhone:~/PasswordKeeper root# rabin2 -c Password\ Keeper\ Lite_.orig
arw 0000000000000000000000000000000000000000000000000000000000000000
0x001a1b64 class 0 some_encrypted_data
0x0000397c method 0 some_encrypted_data_1
0x000039f4 method 1 some_encrypted_data_2
0x00003ddc method 2 some_encrypted_data_3
0x00004124 method 3 some_encrypted_data_4
0x00004130 method 4 some_encrypted_data_5
0x0000413c method 5 some_encrypted_data_6
0x000044a0 method 6 some_encrypted_data_7
0x000044d0 method 7 some_encrypted_data_8
0x00004548 method 8 some_encrypted_data_9
0x00004624 method 9 some_encrypted_data_10
0x00004734 method 10 some_encrypted_data_11
0x00004858 method 11 some_encrypted_data_12
0x000048cc method 12 some_encrypted_data_13
0x00004ab8 method 13 some_encrypted_data_14
0x00004b48 method 14 some_encrypted_data_15
0x000062f4 method 15 some_encrypted_data_16
0x00006490 method 16 some_encrypted_data_17
```

iOS Security Architecture

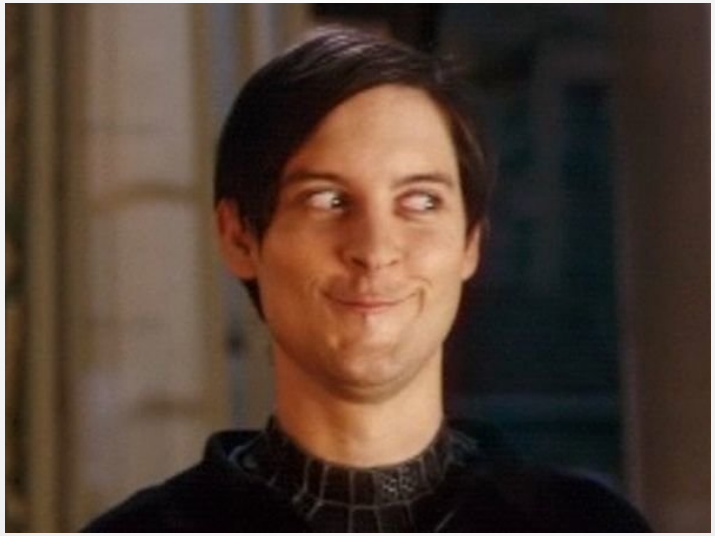
LC_ENCRYPTION_INFO

```
struct encryption_info_command {
   uint32_t cmd;    /* LC_ENCRYPTION_INFO */
   uint32_t cmdsize;  /* sizeof(struct encryption_info_command) */
   uint32_t cryptoff; /* file offset of encrypted range */
   uint32_t cryptsize;  /* file size of encrypted range */
   uint32_t cryptid;  /* which enryption system,
           0 means not-encrypted yet */
};
```

```
[0x00000000]> pf mach0_cmd_enc=didid cmd cmdsize cryptoff cryptsize cryptid
[0x00000000]> pf.mach0_cmd_enc @ mach0_cmd_11
      cmd : 0x00000a24 = 33
  cmdsize : 0x00000a28 = 20
 cryptoff : 0x00000a2c = 8192
cryptsize : 0x00000a30 = 1556480
  cryptid : 0x00000a34 = 1
[0x00000000]>
```

Solutions?

The binary is decrypted in memory



iOS Security Architecture

Clutch / Clutch2

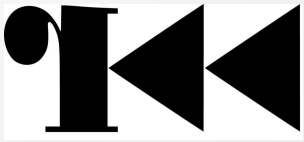
```
iPhone:~/Meetup root# Clutch2 -i
Installed apps:
1:   Pokémon GO <com.nianticlabs.pokemongo>
2:   Safe Password free for iPhone <org.e2uapp.passwordiphonelite>
3:   Instagram <com.burbn.instagram>
4:   LOLcat Translator: The TransLOLulator <.comsugarcodedapps.translolulator>
5:   Atriviate <com.panderetaestudio.Triviados>
6:   WhatsApp Messenger <net.whatsapp.WhatsApp>
7:   Meetup - Groups near you that make community real <com.meetup.iphone>
8:   Twitter <com.atebits.Tweetie2>
9:   LinkedIn <com.linkedin.LinkedIn>
10:  Password Manager - Secure Account Wallet Vault & Lock Apps Passcode Safe <com.GalaxyStudio.PasswordSafeFree>
11:  Spotify Music <com.spotify.client>
iPhone:~/Meetup root#
```

```
iPhone:~ root# Clutch2 -b 6
ASLR slide: 0x3c000
Dumping <ShareExtension> (armv7)
Patched cryptid (32bit segment)
ASLR slide: 0xf8000
Dumping <WhatsApp> (armv7)
Patched cryptid (32bit segment)
Writing new checksum
Writing new checksum
Finished dumping net.whatsapp.WhatsApp to /var/tmp/clutch/17A7EE1A-C27D-4246-9509-3EA2C8B9FA89
Finished dumping net.whatsapp.WhatsApp in 7.8 seconds
iPhone:~ root#
```

Clutch is not good...

- Can't decrypt some binaries
- Decrypt everything all the time (=slow)
- Lots of lines of code (difficult to fix)

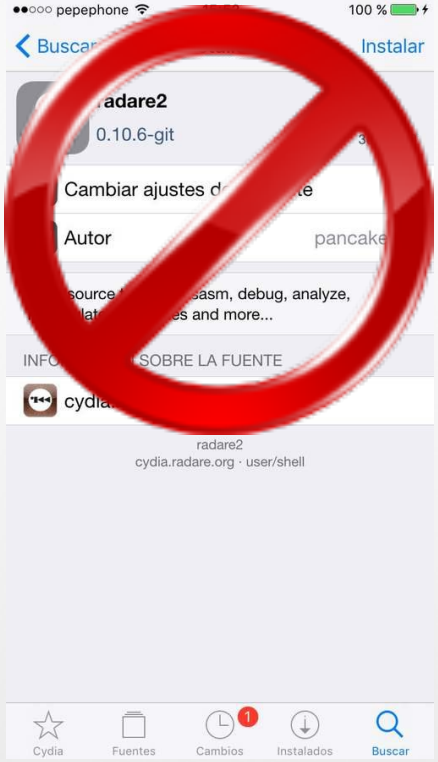
Any alternative?



R2 in your iPhone

From Cydia

(http://cydia.radare.org)



R2 in your iPhone

From GIT

- Compile Radare2 in local.

```
$git clone http://github.com/radare/radare2
$sys/ios-cydia.sh
```

- Copy .deb file into the iphone.

```
radare2/sys/cydia/radare2/radare2_0.10.6-git_iphoneos-arm.deb
```

- Installing deb

```
iphone$ dpkg -i radare2_0.10.6-git_iphoneos-arm.deb
```

- Check version

```
iPhone:~ root# r2 -v
radare2 0.10.6-git 12305 @ darwin-arm-64 git.0.10.1-1846-g61fd4f6
commit: 61fd4f6d0da6983b621264bff0036688cb3d0f17 build: 2016-09-09
iPhone:~ root#
```

R2 in your iPhone

Default

```
iPhone:~ root# r2 /bin/ls
arw 0000000000000000000000000000000000000000000000000000000000000000
 -- Of course r2 runs FreeBSD
[0x00008000]> px 4 @ 0x00
- offset -   0 1  2 3  4 5  6 7  8 9  A B  C D  E F  0123456789ABCDEF
0x00000000  cefa edfe                             ....
[0x00008000]>
```

Debug

```
iPhone:~ root# r2 -d /bin/ls
attach 1683 1683
bin.baddr 0x00004000
arw 0000000000000000000000000000000000000000000000000000000000000000
asm.bits 32
r_debug_reg: error reading registers
 -- The door controls time and space.
[0x00000000]> dm
sys 144K 0x00004000 - 0x00028000 s -r-x /bin/ls 00_copy/bin/ls
sys 016K 0x00028000 - 0x0002c000 s -rw- /bin/ls 01_copy/bin/ls
sys 016K 0x0002c000 - 0x00030000 s -rw- ? 02_copy
sys 020K 0x00030000 - 0x00035000 s -r-- /bin/ls 03_copy/bin/ls
sys 012K 0x00035000 - 0x00038000 s -r-- ? 04_copy
```

R2 in your iPhone

Remember to add the correct Entitlements

```
iPhone:~ root# cat radare.xml
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN"
"http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
 <dict>
  <key>app-identifier</key>
  <true/>
  <key>get-task-allow</key>
  <true/>
  <key>task_for_pid-allow</key>
  <true/>
 </dict>
</plist>
<!--
  This file must be used to 'sign' the 'radare' binary
  after compiling to permit debugging on the iphone-os
  $ ldid -Sradare.xml radare
-->
iPhone:~ root# ldid -Sradare.xml /usr/bin/radare2
```

R2 in your iPhone

And of course, you can check it with R2...

```
iPhone:~ root# r2 /usr/bin/r2
 -- I nodejs so hard my exams.What a nodejs!
[0x000089b8]> iC
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
 <dict>
  <key>app-identifier</key>
  <true/>
  <key>get-task-allow</key>
  <true/>
  <key>task_for_pid-allow</key>
  <true/>
 </dict>
</plist>
<!--
  This file must be used to 'sign' the 'radare' binary
  after compiling to permit debugging on the iphone-os
  $ ldid -Sradare.xml radare
-->


[0x000089b8]>
```

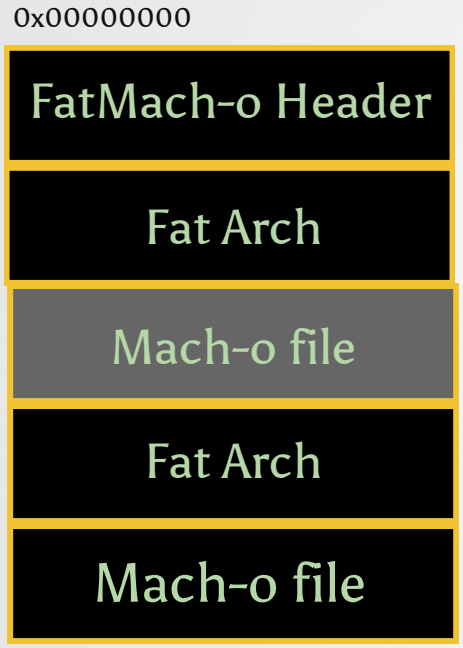
Parsing & Patching

To decrypt an app...

1. Extract mach0 bin from FatMach0 (if needed)
2. Identify whether PIE is enabled or not.
3. Identify encryption information.
4. Identify process address space.
5. Dump decrypted process into a file.
6. Overwrite original binary with the decrypted one.
7. Patch binary structure as a decrypted one.

Parsing & Patching

1. Extract mach0 bin from FatMach0 (if needed)



rabin2

```
iPhone:~/bin root# file Meetup
Meetup: Mach-O fat file with 2 architectures
iPhone:~/bin root# rabin2 -x Meetup
Meetup.fat/Meetup.arm_32.0 created (5298784)
Meetup.fat/Meetup.arm_64.1 created (6558352)
iPhone:~/bin root#
```

Custom

```
iPhone:~/bin root# r2 -nn -i fat.r2 Meetup
 -- Run .dmm* to load the flags of the symbols of all mod
[0x00000000]> e cfg.bigendian = true
[0x00000000]> pf.fatmach0_header.nfat_arch @ $$
nfat_arch : 0x00000004 = 2
[0x00000000]> pxW 4 @ `pfs.fatmach0_header~:[0]`+8~:[1]
0x00004000
[0x00000000]> f binoff=0x4000
[0x00000000]> pxW 4 @ `pfs.fatmach0_header~:[0]`+12~:[1]
0x0050da60
[0x00000000]> f binsize=0x0050da60
[0x00000000]> s binoff
[0x00004000]> wt /tmp/mach.bin binsize
dumped 0x50da60 bytes
Dumped 5298784 bytes from 0x00004000 into /tmp/mach.bin
[0x00004000]>
```

```
struct fat_arch {
  uint32_t cputype;
  uint32_t cpusubtype;
  uint32_t offset;   //fat_arch + 8
  uint32_t size;     //fat_arch + 12
  uint32_t alighn;
};
```

Parsing & Patching

2. Identify whether PIE is enabled or not.

```
[0x100005934]> k bin/cur/info/mach_flags.cparse
enum mach_flags{MH_NOUNDEFS=1,MH_INCRLINK=2,MH_DYLDLINK=4,MH_BINDATLOAD=8,MH_PREBOUND=0x10,MH_SPLIT_SEGS
,MH_TWOLEVEL=0x80,MH_FORCE_FLAT=0x100,MH_NOMULTIDEFS=0x200,MH_NOFIXPREBINDING=0x400,MH_PREBINDABLE=0x800
0,MH_SUBSECTIONS_VIA_SYMBOLS=0x2000,MH_CANONICAL=0x4000,MH_WEAK_DEFINES=0x8000,MH_BINDS_TO_WEAK=0x10000,
N=0x20000,MH_ROOT_SAFE=0x40000,MH_SETUID_SAFE=0x80000,MH_NO_REEXPORTED_DYLIBS=0x100000,MH_PIE=0x200000,M
=0x400000,MH_HAS_TLV_DESCRIPTORS=0x800000,MH_NO_HEAP_EXECUTION=0x1000000 }
```

```
[0x00000000]> pf.mach0_header @ mach0_header~flags
      flags : 0x00000018 = 0x00200085
[0x00000000]>
```

- Parse Mach-o Header
- Get flags field
- If contains 0x00200000 => PIE Enabled

```
enum {
  // Constant bits for the "flags" field in
  // llvm::MachO::mach_header_64
  MH_NOUNDEFS                = 0x00000001u,
  MH_INCRLINK                = 0x00000002u,
  MH_DYLDLINK                = 0x00000004u,
  MH_BINDATLOAD              = 0x00000008u,
  MH_PREBOUND                = 0x00000010u,
  MH_SPLIT_SEGS              = 0x00000020u,
  MH_LAZY_INIT               = 0x00000040u,
  MH_TWOLEVEL                = 0x00000080u,
  MH_FORCE_FLAT              = 0x00000100u,
  MH_NOMULTIDEFS             = 0x00000200u,
  MH_NOFIXPREBINDING         = 0x00000400u,
  MH_PREBINDABLE             = 0x00000800u,
  MH_ALLMODSBOUND            = 0x00001000u,
  MH_SUBSECTIONS_VIA_SYMBOLS = 0x00002000u,
  MH_CANONICAL               = 0x00004000u,
  MH_WEAK_DEFINES            = 0x00008000u,
  MH_BINDS_TO_WEAK           = 0x00010000u,
  MH_ALLOW_STACK_EXECUTION   = 0x00020000u,
  MH_ROOT_SAFE               = 0x00040000u,
  MH_SETUID_SAFE             = 0x00080000u,
  MH_NO_REEXPORTED_DYLIBS    = 0x00100000u,
  MH_PIE                     = 0x00200000u,
  MH_DEAD_STRIPPABLE_DYLIB   = 0x00400000u,
  MH_HAS_TLV_DESCRIPTORS     = 0x00800000u,
  MH_NO_HEAP_EXECUTION       = 0x01000000u,
  MH_APP_EXTENSION_SAFE      = 0x02000000u
};
```

Parsing & Patching

3. Identify encryption information

```
00121       LC_ENCRYPTION_INFO      = 0x00000021u,
00133       LC_ENCRYPTION_INFO_64   = 0x0000002Cu,
```

```
[0x00000000]> pf mach0_cmd_enc=xidid cmd cmdsize cryptoff cryptsize cryptid
[0x00000000]> pf.mach0_cmd_enc @ mach0_cmd_12
       cmd : 0x00000978 = 0x0000002c
   cmdsize : 0x0000097c = 24
  cryptoff : 0x00000980 = 16384
 cryptsize : 0x00000984 = 294912
   cryptid : 0x00000988 = 1
[0x00000000]>
```

```
[0x100005934]> k bin/cur/info/*~crypt
mach0_cmd_12.cmd=encryption_info
crypto=true
cryptid=0x1
cryptoff=0x4000
cryptsize=0x48000
cryptheader=0x978
objc_class_some_encrypted_data.offset=0x53d18
[0x100005934]>
```

Parsing & Patching

4. Identify process address space.

PIE

- Get binary memory map.

```
iPhone:~/Linkedin root# r2 -d LinkedIn
attach 1753 1753
bin.baddr 0x1000e0000
asm.bits 64
[p/native/xnu/xnu_threads.c:58 xnu_thread_get_drx] xnu_thread_get_drx: Invalid argument
[p/native/xnu/xnu_threads.c:58 xnu_thread_get_drx] xnu_thread_get_drx: Invalid argument
 -- r2 is a great OS, but a terrible hex editor
[0x120055000]> dm
sys 016K 0x00000001000e0000 - 0x00000001000e4000 s -r-x /private/var/containers/Bundle/A
69-8753-5046060DDF51/LinkedIn.app/LinkedIn 00_copy/private/var/containers/Bundle/Applica
3-5046060DDF51/LinkedIn.app/LinkedIn
```

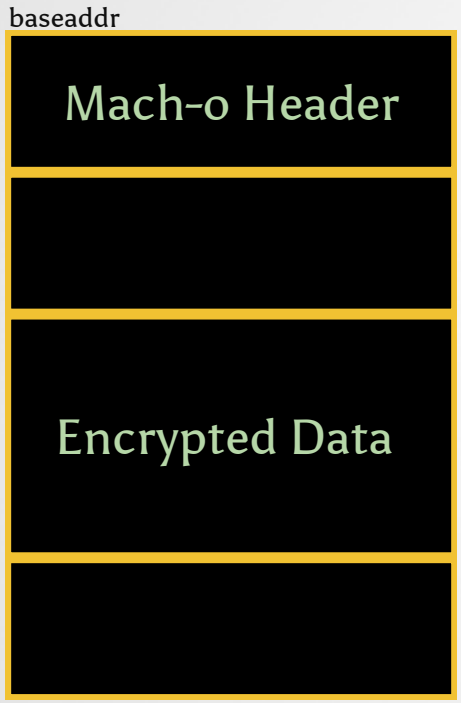
non-PIE

- Get vmaddr in TEXT segment

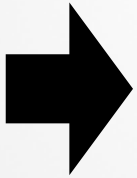
```
[0x00000000]> pf.mach0_segment @ mach0_segment_1
      cmd : 0x00000054 = 0x00000001
  cmdsize : 0x00000058 = 736
  segname : 0x0000005c = __TEXT
   vmaddr : 0x0000006c = 0x00004000
   vmsize : 0x00000070 = 0x0041c000
```

Parsing & Patching

5. Dump decrypted process into a file



cryptoff

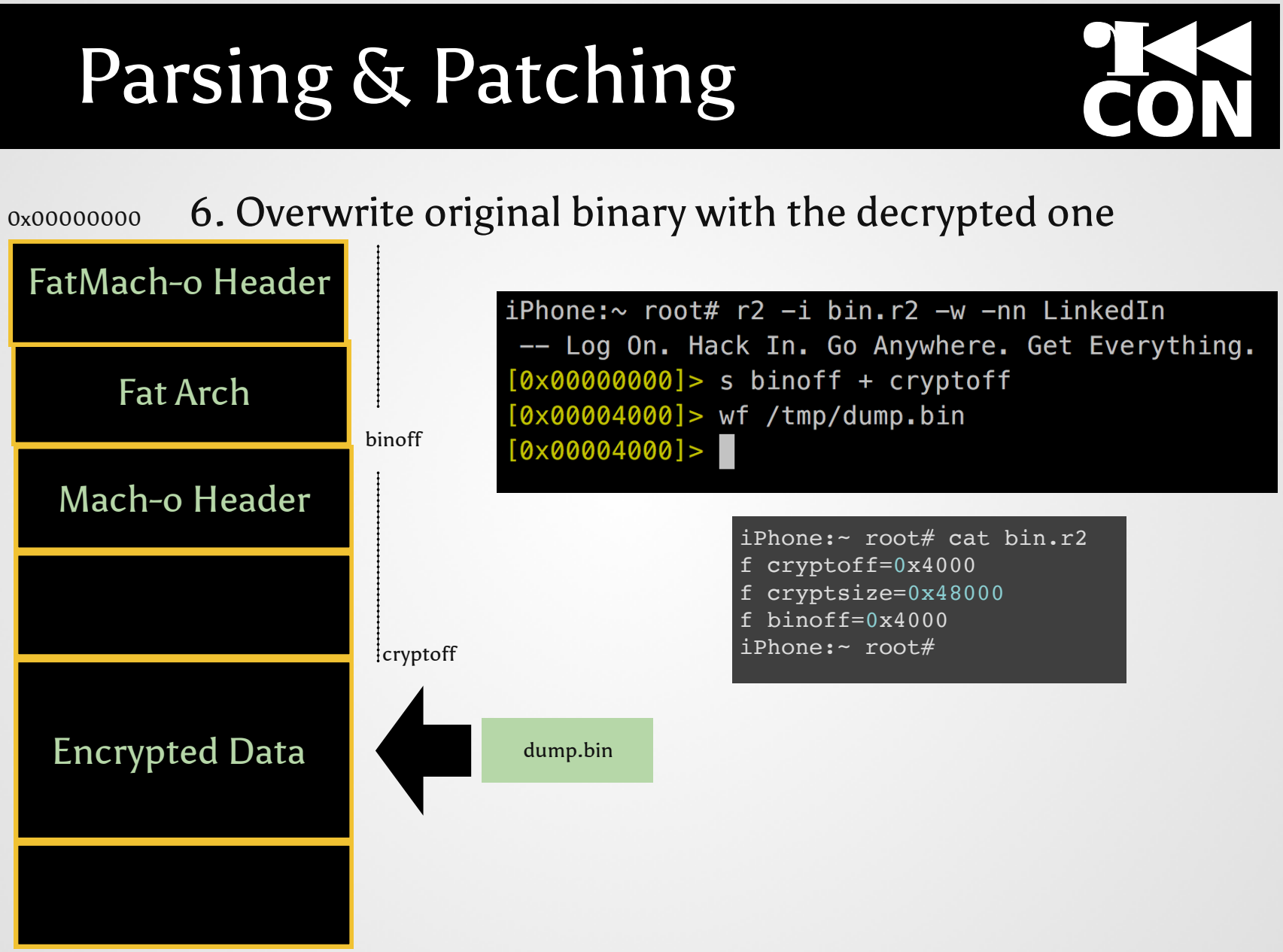


dump.bin

```
[0x1000e5934]> s baseaddr + cryptoff
[0x1000e4000]> wt /tmp/dump.bin cryptsize
dumped 0x48000 bytes
Dumped 294912 bytes from 0x1000e4000 into /tmp/dump.bin
[0x1000e4000]>
```

```
[0x120055000]> f baseaddr=`dm~r-x:0[2]`
[0x120055000]> f cryptoff=0x4000
[0x120055000]> f cryptsize=0x48000
[0x120055000]> f cryptheader=0x978
```

Parsing & Patching



Parsing & Patching

7. Patch binary structure as a decrypted one.

```
struct encryption_info_command {
   uint32_t cmd;
   uint32_t cmdsize;
   uint32_t cryptoff;
   uint32_t cryptsize;
   uint32_t cryptid;
};
```

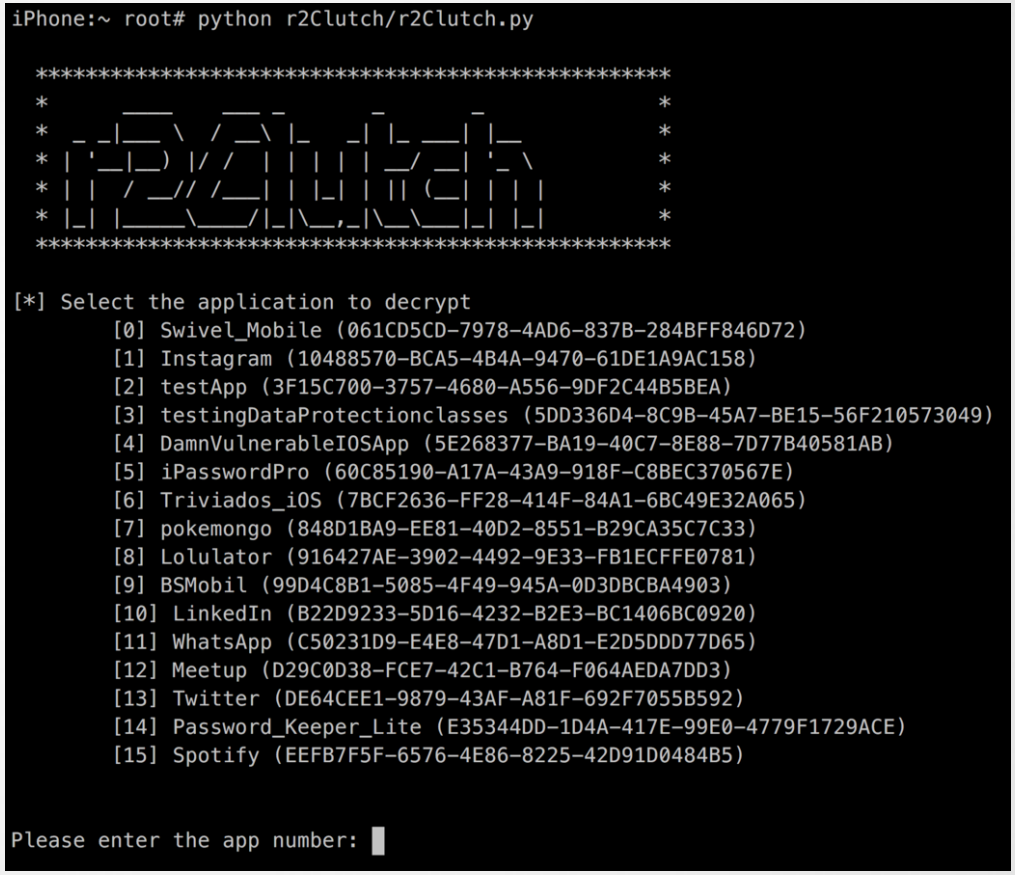
```
[0x00000a3c]> ?v mach0_cmd_12
0xa3c
[0x00000a3c]> k bin/cur/info/cryptheader
0xa3c
```

```
[0x00000000]> pf mach0_cmd_enc=xixxd cmd cmdsize cryptoff cryptsize cryptid
[0x00000000]> pf.mach0_cmd_enc @ mach0_cmd_12
       cmd : 0x00000c48 = 0x0000002c
   cmdsize : 0x00000c4c = 24
  cryptoff : 0x00000c50 = 0x00004000
 cryptsize : 0x00000c54 = 0x004b8000
   cryptid : 0x00000c58 = 1
[0x00000000]> wx 0x00 @ mach0_cmd_12+16
[0x00000000]> pf.mach0_cmd_enc @ mach0_cmd_12
       cmd : 0x00000c48 = 0x0000002c
   cmdsize : 0x00000c4c = 24
  cryptoff : 0x00000c50 = 0x00004000
 cryptsize : 0x00000c54 = 0x004b8000
   cryptid : 0x00000c58 = 0
[0x00000000]>
```

Automation

R2Clutch

- Python
- R2Pipe
- 32 / 64 bit Support



Automation

R2Clutch Demo

https://github.com/as0ler/r2clutch

Future work

- Improve speed
- Review some controls anti-cracking
- Integration with a iOS native App (RUST?)
- Distribution through Cydia

Thanks for Listening!

Questions?

 @as0ler