nccgroup
freedom from doubt

Exploring the Jolla Phone

Chris Weedon (@crweedon) // Intrepidus Group

Vitaly McLain (@send9) // Matasano

Drew Suarez (@utkan0s) // Matasano

nccgroup
freedom from doubt

Why this talk?

- Explore an interesting phone
- Show different attack surfaces a phone can have
    - Commonalities with mobile, Linux, ARM, etc
- Show how phone can be used for your purposes

nccgroup
freedom from doubt

Jolla: A History

- Nokia developed Maemo
- Then they merged it with Intel's Moblin
- This became MeeGo
- …and then they got rid of all Linux phones
- Engineers + Nokia "Bridge" fund == Jolla Oy

nccgroup
freedom from doubt

From MeeGo to Sailfish OS

- Funding but no intellectual property
- Mer == open-source MeeGo fork
- Combine open-source: Mer + Wayland + QT5/QML
- And proprietary: Silica (compliment to QtQuick), Lipstick (shell on top of Wayland)
- Change .deb -> openSuSE RPM, apt -> zypper, upstart -> systemd
- We get Sailfish OS!

The Other Half

- Really neat “smart covers” called Other Half
- Ambiance / theme based on cover
- Keyboard, other peripherals, etc

Unofficial Fan Art

THE OTHER HALF

jolla

nccgroup
freedom from doubt

Jolla's Boot/Recovery

- Structure
- Inspecting the Firmware
- Lock/Unlock
- Thoughts

nccgroup
freedom from doubt

Structure

- Android style images for recovery
  - Android boot header
  - zImage and rootfs.cpio
- Recovery consists of a few scripts
  - Menus/functions via script
  - Binary responsible for lock mechanism

nccgroup
freedom from doubt

Recovery / fastboot mode

- Access recovery with vol down + power at boot (no usb)
  - telnet based connection
  - menu system of shell scripts
- Access fastboot with vol down + power at boot (w usb)
  - needs identifier 0x2931 (fastboot -i 0x2931)
  - not all args supported, locked by default

nccgroup
freedom from doubt

What are we after?

- Understanding the image type
- Device topologies
- Ramdisk contents

```
utkanos@leviathan ~/jolla $ od -c mmcblk0p21.img | more
0000000  A  N  D  R  O  I  D  !  0  H  ]  \0 \0 200    200
0000020 257 022   6 \0 \0 \0    202 \0 \0 \0 \0 \0 \0 020 201
0000040 \0 001    200 \0 \b \0 \0 \0 \0 \0 \0 \0 \0 \0 \0
0000060 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0
0000100  i  n  i  t  =  /  s  b  i  n  /  p  r  e  i  n
0000120  i  t     r  o  o  t  =  /  d  e  v  /  m  m  c
0000140  b  l  k  0  p  2  8     r  o  o  t  f  s  t  y
0000160  p  e  =  b  t  r  f  s     r  o  o  t  f  l  a
0000200  g  s  =  r  e  c  o  v  e  r  y     n  o  i  n
0000220  i  t  r  d     a  n  d  r  o  i  d  b  o  o  t
0000240  .  h  a  r  d  w  a  r  e  =  q  c  o  m     u
0000260  s  e  r  _  d  e  b  u  g  =  3  1     e  h  c
0000300  i  -  h  c  d  .  p  a  r  k  =  3     m  a  x
0000320  c  p  u  s  =  2 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0
0000340 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0
```

```
Target File:   ../_p21.img.extracted/47F7
MD5 Checksum:  3f274fc9e520f8016050bae96f527ddf
Signatures:    285

DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
8355960       0x7F8078        Linux kernel version "3.4.0.20140403.1 (abuild@es-17-20) (gcc version 4.6.4 20130412 20) (gcc version 4.6.4 20130412 (Mer 4.6.4-1) (Linaro GCC 4.6-2"
8396948       0x802094        gzip compressed data, maximum compression, from Unix, NULL date: Wed Dec 31 18:00:00 1969
8617769       0x837F29        LZMA compressed data, properties: 0x6E, dictionary size: 1048576 bytes, uncompressed size: 262144 bytes
8617805       0x837F4D        LZMA compressed data, properties: 0x64, dictionary size: 1048576 bytes, uncompressed size: 262144 bytes
8805525       0x865C95        LZMA compressed data, properties: 0x64, dictionary size: 2097152 bytes, uncompressed size: 262144 bytes
11326327      0xACD377        Copyright string: " (c) 2006-2007 BalaBit IT Ltd."
13033628      0xC6E09C        ASCII cpio archive (SVR4 with no CRC), file name: "dev", file name length: "0x00000004", file size: "0x00000000"
13033744      0xC6E110        ASCII cpio archive (SVR4 with no CRC), file name: "dev/console", file name length: "0x0000000C", file size: "0x00000000"
13033868      0xC6E18C        ASCII cpio archive (SVR4 with no CRC), file name: "root", file name length: "0x00000005", file size: "0x00000000"
13033984      0xC6E200        ASCII cpio archive (SVR4 with no CRC), file name: "TRAILER!!!", file name length: "0x0000000B", file size: "0x00000000"
14095787      0xD715AB        LZMA compressed data, properties: 0xC0, dictionary size: 65536 bytes, uncompressed size: 131074 bytes
14173395      0xD844D3        LZMA compressed data, properties: 0xC0, dictionary size: 65536 bytes, uncompressed size: 131077 bytes
14872909      0xE2F14D        LZMA compressed data, properties: 0x5D, dictionary size: 131072 bytes, missing uncompressed size
14919971      0xE3A923        LZMA compressed data, properties: 0xC0, dictionary size: 524288 bytes, uncompressed size: 720896 bytes


Target File:   ../_p21.img.extracted/rootfs.cpio
MD5 Checksum:  99e5d158891b1712f859ffdcc3b14f7c
Signatures:    285

DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
0             0x0             ASCII cpio archive (SVR4 with no CRC), file name: "/init", file name length: "0x00000006", file size: "0x00000881"
2296          0x8F8           ASCII cpio archive (SVR4 with no CRC), file name: "/mnt", file name length: "0x00000005", file size: "0x00000000"
2412          0x96C           ASCII cpio archive (SVR4 with no CRC), file name: "/dev", file name length: "0x00000005", file size: "0x00000000"
2528          0x9E0           ASCII cpio archive (SVR4 with no CRC), file name: "/dev/pts", file name length: "0x00000009", file size: "0x00000000"
2648          0xA58           ASCII cpio archive (SVR4 with no CRC), file name: "/dev/shm", file name length: "0x00000009", file size: "0x00000000"
2768          0xAD0           ASCII cpio archive (SVR4 with no CRC), file name: "/var", file name length: "0x00000005", file size: "0x00000000"
2884          0xB44           ASCII cpio archive (SVR4 with no CRC), file name: "/var/cache", file name length: "0x0000000B", file size: "0x00000000"
```

binwalk

```
utkanos@leviathan ~/jolla $ ~/mkbootimg_tools/mkboot mmcblk0p21.img p21/
Unpack & decompress mmcblk0p21.img to p21/
  kernel         : zImage
  ramdisk        : ramdisk
  page size      : 2048
  kernel size    : 6113328
  ramdisk size   : 3543727
  base           : 0x80200000
  kernel offset  : 0x00008000
  ramdisk offset : 0x02000000
  second_offset  : 0x00f00000
  tags offset    : 0x00000100
  cmd line       : init=/sbin/preinit root=/dev/mmcblk0p28 rootfstype=btrfs rootflags=recovery noinitrd androidboot.hardware=qcom user_debug=31 ehci-hcd.park=3 maxcpus=2
ramdisk is gzip format.
Unpack completed.
utkanos@leviathan ~/jolla $ cd p21/
utkanos@leviathan ~/jolla/p21 $ ls
img_info  ramdisk  ramdisk.gz  zImage
utkanos@leviathan ~/jolla/p21 $ file zImage
zImage: Linux kernel ARM boot executable zImage (little-endian)
utkanos@leviathan ~/jolla/p21 $ ls ramdisk
bin  dev  etc  init  lib  mnt  proc  root  sbin  sys  tmp  usr  var
utkanos@leviathan ~/jolla/p21 $
```

mkboot

nccgroup
freedom from doubt

Topology

- ‘mount’ and /proc/partitions
- /dev/block/platform/[soc]/by-name
- ramdisk contents

```
lrwxrwxrwx 1 root root  22 2014-10-18 23:40 aboot -> ../../../../mmcblk0p17
lrwxrwxrwx 1 root root  22 2014-10-18 23:40 boot -> ../../../../mmcblk0p20
lrwxrwxrwx 1 root root  22 2014-10-18 23:40 drm -> ../../../../mmcblk0p19
lrwxrwxrwx 1 root root  21 2014-10-18 23:40 emgdload -> ../../../../mmcblk0p1
lrwxrwxrwx 1 root root  21 2014-10-18 23:40 fsg -> ../../../../mmcblk0p8
lrwxrwxrwx 1 root root  22 2014-10-18 23:40 misc -> ../../../../mmcblk0p23
lrwxrwxrwx 1 root root  22 2014-10-18 23:40 modem -> ../../../../mmcblk0p18
lrwxrwxrwx 1 root root  22 2014-10-18 23:40 modemst1 -> ../../../../mmcblk0p10
lrwxrwxrwx 1 root root  22 2014-10-18 23:40 modemst2 -> ../../../../mmcblk0p11
lrwxrwxrwx 1 root root  22 2014-10-18 23:40 pad1 -> ../../../../mmcblk0p22
lrwxrwxrwx 1 root root  22 2014-10-18 23:40 persist -> ../../../../mmcblk0p25
lrwxrwxrwx 1 root root  21 2014-10-18 23:45 Qcfg -> ../../../../mmcblk0p4
lrwxrwxrwx 1 root root  21 2014-10-18 23:40 Qdlog -> ../../../../mmcblk0p5
lrwxrwxrwx 1 root root  21 2014-10-18 23:40 Qfa -> ../../../../mmcblk0p3
lrwxrwxrwx 1 root root  21 2014-10-18 23:40 Qglog -> ../../../../mmcblk0p9
lrwxrwxrwx 1 root root  21 2014-10-18 23:40 Qlogfilter -> ../../../../mmcblk0p7
lrwxrwxrwx 1 root root  21 2014-10-18 23:40 QOTP -> ../../../../mmcblk0p2
lrwxrwxrwx 1 root root  21 2014-10-18 23:40 Qvariables -> ../../../../mmcblk0p6
lrwxrwxrwx 1 root root  22 2014-10-18 23:40 recovery -> ../../../../mmcblk0p21
lrwxrwxrwx 1 root root  22 2014-10-18 23:40 rpm -> ../../../../mmcblk0p16
lrwxrwxrwx 1 root root  22 2014-10-18 23:40 sailfish -> ../../../../mmcblk0p28
lrwxrwxrwx 1 root root  22 2014-10-18 23:40 sbl1 -> ../../../../mmcblk0p12
lrwxrwxrwx 1 root root  22 2014-10-18 23:40 sbl2 -> ../../../../mmcblk0p13
lrwxrwxrwx 1 root root  22 2014-10-18 23:40 sbl3 -> ../../../../mmcblk0p14
lrwxrwxrwx 1 root root  22 2014-10-18 23:54 security -> ../../../../mmcblk0p27
lrwxrwxrwx 1 root root  22 2014-10-18 23:40 ssd -> ../../../../mmcblk0p26
lrwxrwxrwx 1 root root  22 2014-10-18 23:40 swap -> ../../../../mmcblk0p24
lrwxrwxrwx 1 root root  22 2014-10-18 23:40 tz -> ../../../../mmcblk0p15
```

```
179       0   15267840 mmcblk0
179       1       4079 mmcblk0p1      emgdload
179       2      32768 mmcblk0p2       QOTP
179       3       4096 mmcblk0p3      Qfa
179       4       4096 mmcblk0p4      Qcfg
179       5       4096 mmcblk0p5      Qdlog
179       6       2048 mmcblk0p6      Qvariables
179       7       2048 mmcblk0p7      Qlogfilter
179       8       4096 mmcblk0p8      fsg
179       9      49152 mmcblk0p9       "SYSLOG"
179      10       4096 mmcblk0p10      modemst1
179      11       4096 mmcblk0p11      modemst2
179      12       2048 mmcblk0p12       SBL1
179      13       2048 mmcblk0p13       SBL2
179      14       2048 mmcblk0p14       SBL3
179      15       2048 mmcblk0p15       trustzone
179      16       2048 mmcblk0p16       rpm
179      17       2048 mmcblk0p17       aboot
179      18      65536 mmcblk0p18      "FIRMWARE"
179      19       8192 mmcblk0p19      "DRM"
179      20      12288 mmcblk0p20       12MB (GOOD TARGER FOR K/R) KERNEL
179      21      12288 mmcblk0p21       12MB (GOOD TARGET FOR K/R) RECOVERY
179      22       8192 mmcblk0p22      pad1
179      23       8192 mmcblk0p23      misc
179      24     520184 mmcblk0p24       "SWAP"
179      25       8192 mmcblk0p25      "PERSIST"
179      26          8 mmcblk0p26    ssd
179      27       8192 mmcblk0p27      "SECURITY"
179      28   14415855 mmcblk0p28       "HOME /"
```

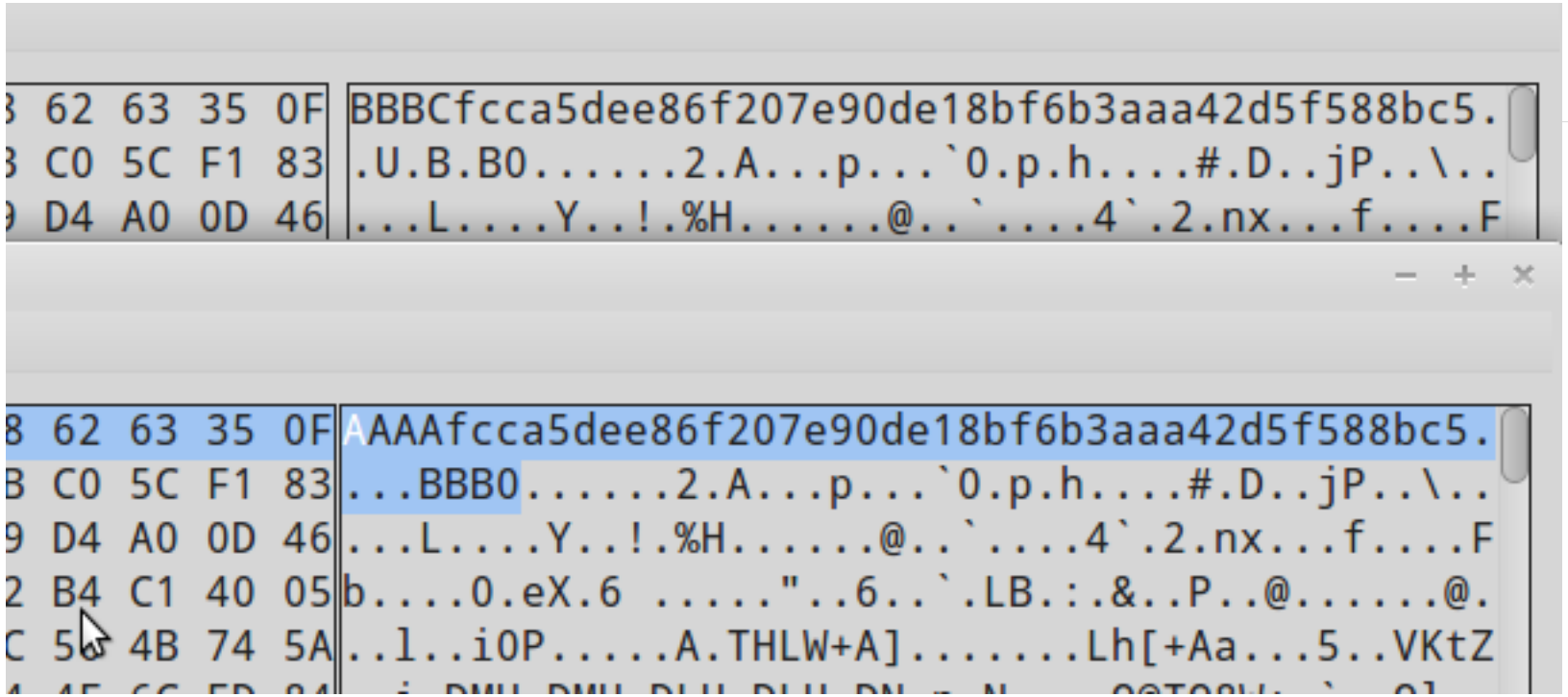
nccgroup
freedom from doubt

Device lock

- Set in userland via system settings
- Protects recovery shell and boot loader
- mmcblk0p27 (security partition)
  - header shows lock/unlock status
  - EVP_SHA1() and HMAC of pin code…

```
.text:00008B4C                ADD             R0, PC, R0 ; "/dev/mmcblk0p2"
.text:00008B50                ADD             R1, PC, R1 ; aRb ; "rb"
.text:00008B54                BL              fopen
.text:00008B58                SUBS            R4, R0, #0
.text:00008B5C                BEQ             loc_8B88
.text:00008B60                MOV             R1, #0x10
.text:00008B64                MOV             R2, #0
.text:00008B68                BL              fseek
.text:00008B6C                MOV             R0, R8
.text:00008B70                MOV             R1, #0xF
.text:00008B74                MOV             R2, #1
.text:00008B78                MOV             R3, R4
.text:00008B7C                BL              fread
.text:00008B80                MOV             R0, R4
.text:00008B84                BL              fclose
.text:00008B88
.text:00008B88 loc_8B88                       ; CODE XREF: .text:00008B5C↑j
.text:00008B88                BL              EVP_sha1
.text:00008B8C                MOV             R10, #0
.text:00008B90                LDR             R7, =(a02x - 0x8BA0)
.text:00008B94                MOV             R4, R10
.text:00008B98                ADD             R7, PC, R7 ; "%02x"
.text:00008B9C                MOV             R9, R0
.text:00008BA0                MOV             R0, R8
.text:00008BA4                BL              strlen
.text:00008BA8                LDR             R11, [R6,#8]
.text:00008BAC                MOV             R2, R0
.text:00008BB0                MOV             R0, R11
.text:00008BB4                STR             R2, [SP,#0x14]
.text:00008BB8                BL              strlen
.text:00008BBC                MOV             R1, R8
.text:00008BC0                LDR             R2, [SP,#0x14]
.text:00008BC4                MOV             R3, R11
.text:00008BC8                STR             R10, [SP,#4]
.text:00008BCC                STR             R10, [SP,#8]
.text:00008BD0                STR             R0, [SP]
.text:00008BD4                MOV             R0, R9
.text:00008BD8                BL              HMAC
.text:00008BDC                LDR             R3, =(word_9394 - 0x8BF0)
.text:00008BE0                MOV             R1, R10
.text:00008BE4                MOV             R2, #0x27
```

nccgroup
freedom from doubt



mmcblk0p27 (locked, unlocked)

nccgroup
freedom from doubt

Quirks

- 5 attempts at pin code, then throttled
  - After 5 wrong pins, a file is written to ramdisk
  - a reboot clears it (not surprising)

```
[root@Jolla nemo]# od -c p6_postblunlock.img
0000000  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0
*
0010240  \0  \0  \0  \0   d   f   s   c   k  \0  \0  \0  \0  \0  \0  \0
0010260  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0
*
0010660  \0  \0  \0  \0  \0  \0  \0 377 377 377 377 377  \0  \0  \0  \0
0010700  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0
*
0047500  \0  \0  \0  \0  \0   K   1   2   C   o   L   N   u   O   e   M
0047520  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0
*
10000000
```

mmcblk0p6 (after bootloader unlock)

nccgroup
freedom from doubt

Image signing?

- lk is patched in 1.1.0.38 (Uitukka)
- Fixes RSA cube root attack on signature
- Currently images are not signed…

> The kernel image is not signed. If you manage to replace the kernel image on disk with your own it should boot. There are long-term plans to offer a trusted boot chain, but not necessarily on this hardware.

nccgroup
freedom from doubt

Thoughts

- If you have a Jolla, enable device lock and developer mode!
  - not an ideal security model
  - at least some protection
- Interesting mix of different software may expose additional issues later on
- Init scripts? :)

nccgroup
freedom from doubt

Operating System

- Many interesting binaries on the device
- A lot of test binaries and applications left intact
  - Not sure if this is a result of enabling developer mode or if this is stock
    - Ex: qseecomd_security_test, oemwvtest, StoreKeybox
- Attack surface is potentially huge, but gets small quickly.
  - Virtually no listening services other than DHCP, so remote attack surface is small from a network perspective.
  - Leaves plenty of room for vulnerable applications

nccgroup
freedom from doubt

Operating System

- You say you want security? Sandboxing, ASLR, RELRO, PIE, NX, etc?
  - Nope… not here, Well, some of it is (see next page)
  - As of now, the system relies heavily on *nix USER/FS permissions
    - Which isn't bad… it's just not great
    - There are plans to implement these things in the future though…

>> 5. Information on kernel build and application build procedures. The
>> binaries seem to be all over the map in terms of executable protection
>> mechanisms(ASLR, NX, etc). Addtionally, there doesn't seem to be any
>> additional hardening of the underlying linux kernel such as grsec, PaX,
>> selinux.

We do not have any protection mechanisms in place at the moment. It is on the roadmap for future updates, though.

nccgroup
freedom from doubt

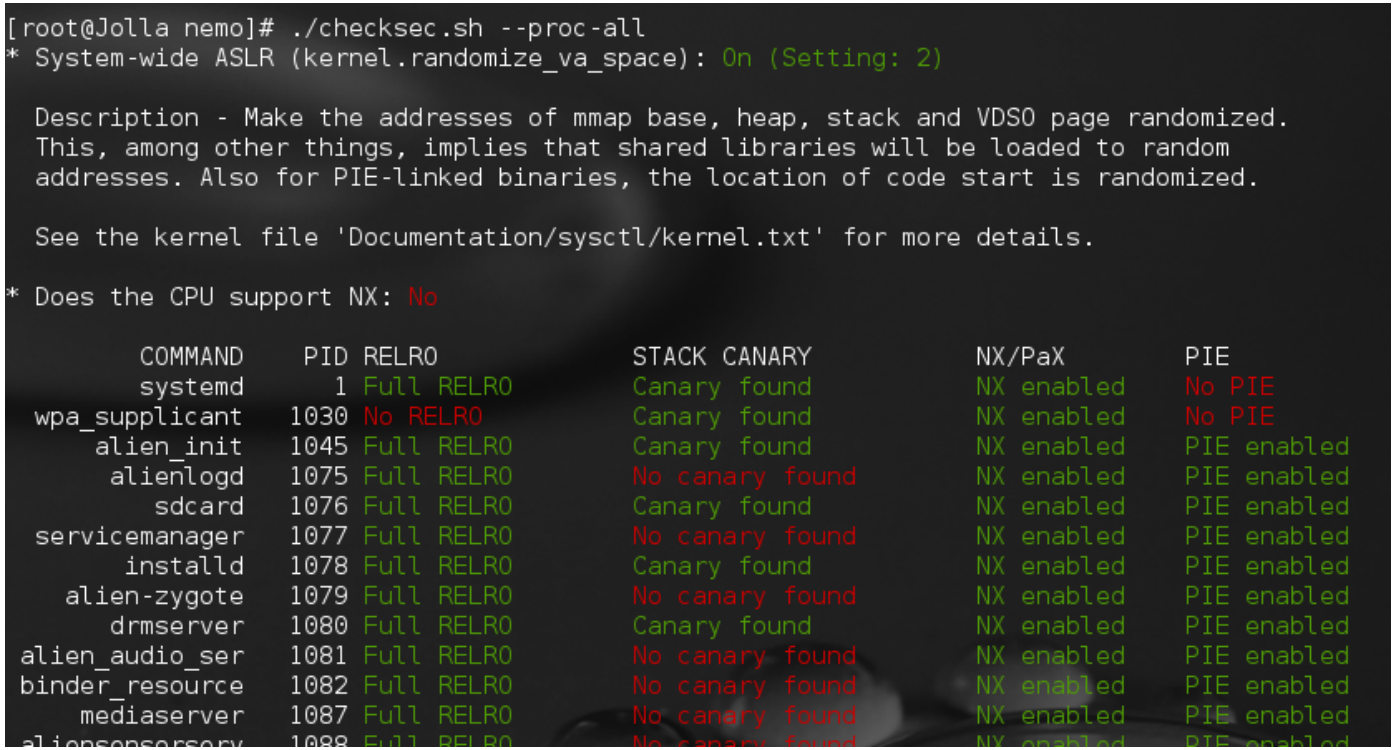
Operating System

- No Kern Heap Hardening
- No grsec/PaX
- No user copy checks
- No enforcement of read-only

nccgroup
freedom from doubt

Operating System

- CPU NX bit support? -- Nope



```
[root@Jolla nemo]# ./checksec.sh --proc-all
* System-wide ASLR (kernel.randomize_va_space): On (Setting: 2)

  Description - Make the addresses of mmap base, heap, stack and VDSO page randomized.
  This, among other things, implies that shared libraries will be loaded to random
  addresses. Also for PIE-linked binaries, the location of code start is randomized.

  See the kernel file 'Documentation/sysctl/kernel.txt' for more details.

* Does the CPU support NX: No

         COMMAND    PID RELRO           STACK CANARY          NX/PaX        PIE
         systemd      1 Full RELRO      Canary found          NX enabled    No PIE
  wpa_supplicant   1030 No RELRO       Canary found          NX enabled    No PIE
      alien_init   1045 Full RELRO      Canary found          NX enabled    PIE enabled
       alienlogd   1075 Full RELRO      No canary found       NX enabled    PIE enabled
          sdcard   1076 Full RELRO      Canary found          NX enabled    PIE enabled
  servicemanager   1077 Full RELRO      No canary found       NX enabled    PIE enabled
        installd   1078 Full RELRO      Canary found          NX enabled    PIE enabled
    alien-zygote   1079 Full RELRO      No canary found       NX enabled    PIE enabled
       drmserver   1080 Full RELRO      Canary found          NX enabled    PIE enabled
 alien_audio_ser   1081 Full RELRO      No canary found       NX enabled    PIE enabled
 binder_resource   1082 Full RELRO      No canary found       NX enabled    PIE enabled
     mediaserver   1087 Full RELRO      No canary found       NX enabled    PIE enabled
 aliensensorsery   1088 Full RELRO      No canary found       NX enabled    PIE enabled
```

nccgroup
freedom from doubt

Operating System

- Stack canaries? RELRO? PIE?
  - Some but not all

```
[root@Jolla nemo]# ./checksec.sh --dir /usr/bin
RELRO           STACK CANARY      NX            PIE             RPATH      RUNPATH      FILE
No RELRO        Canary found      NX enabled    No PIE          No RPATH   No RUNPATH   /usr/bin/a2p
No RELRO        No canary found   NX enabled    No PIE          No RPATH   No RUNPATH   /usr/bin/addftinfo
No RELRO        Canary found      NX enabled    No PIE          No RPATH   No RUNPATH   /usr/bin/addr2line
No RELRO        No canary found   NX enabled    No PIE          No RPATH   No RUNPATH   /usr/bin/ag-backup
No RELRO        No canary found   NX enabled    No PIE          No RPATH   No RUNPATH   /usr/bin/ag-tool
No RELRO        Canary found      NX enabled    No PIE          No RPATH   No RUNPATH   /usr/bin/ambienced
No RELRO        No canary found   NX enabled    No PIE          No RPATH   No RUNPATH   /usr/bin/apkd-launcher
No RELRO        Canary found      NX enabled    No PIE          No RPATH   No RUNPATH   /usr/bin/applydeltarpm
No RELRO        Canary found      NX enabled    No PIE          No RPATH   No RUNPATH   /usr/bin/ar
```

nccgroup
freedom from doubt

Application Layer

- Stock Applications
  - Most Applications are written in C/C++
    - Although there are lots of shell scripts on the device
  - Mix of ELF32 Arm7vh binaries and QML “applications”, I’m using application here very loosely
  - Often, the binaries have QT API calls embedded in them that leverage the QML “applications”.
    - Picture the binary as the service, and the QML as the GUI
  - What is QML?
    - QT Meta Language or QT Modeling Language
    - It’s like Javascript, Openscad, Python, and Latex all rolled into one
    - Used to describe what something will look like, and the action that thing will perform

nccgroup
freedom from doubt

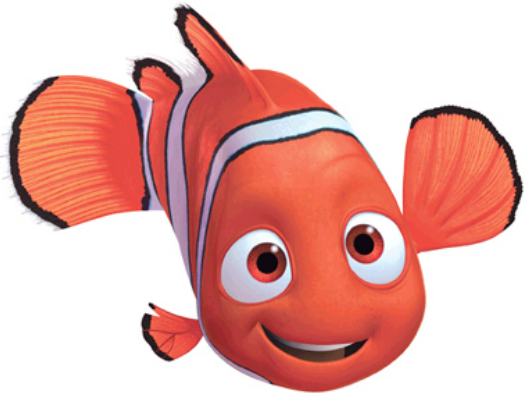
QML

- What is QML?(2)

```
import QtQuick 2.1
import Sailfish.Silica 1.0
import Sailfish.Lipstick 1.0
import QtQuick.Window 2.1

SystemDialog {
    id: dialog

    //% "Android Support Upgrade"
    title: qsTrId("aliendalvik-he-android_support_upgrade")
    contentHeight: descriptionLabel.implicitHeight + buttonOk.height + 6 * Theme.paddingLarge

    Label {
        id: descriptionLabel
        anchors {
            fill: parent
            margins: Theme.paddingLarge
        }
        width: parent.width - Theme.paddingLarge
        wrapMode: Text.Wrap
        //% "Android applications will be re-installed automatically due to new version of Android support."
        text: qsTrId("aliendalvik-la-start_reinstall_apks")
    }

    Button {
        id: buttonOk
        // TODO: Use the right Icon
        //% "Ok"
        text: qsTrId("aliendalvik-bt-ok")
        onClicked: {
```

The Userland

- All regular apps run as “nemo” (there’s one exception)
- That’s how you access phone, too
- Use SSH via USB or network in dev mode



nccgroup
freedom from doubt

Attack Surface: Userland

- Some binaries as root via invoker
- And there are some suids/sgids
- Interesting: owned by root or gid == privileged
- Interested in binaries NOT common to other Linux distros (Sailfish/Mer/Maemo binaries?)

nccgroup
freedom from doubt

D-Bus

- D-Bus used for IPC
- Common to other Linux environments
- …but everything runs as “nemo”
- dbus-monitor provided, acts as sniffer
- Regular user discovered Outlook passwords
- Interesting area to explore further

jolla

email security dbus-monitor

I was fiddling with the dbus-monitor and noticed the password for my exchange mail flicker by on the screen. It seems like this could be a huge security hole since any app monitoring the dbus could get access to my exchange mail. Here is a draft of what I saw.

asked **Apr 4 '14**
Xiph
671 ●10 ●16 ●20

updated **Aug 13 '14**
jit
116 ●4 ●8
jolla moderator

```
method call sender=:1.95 -> dest=org.freedesktop.DBus serial=31 path=/org/freede
sktop/DBus; interface=org.freedesktop.DBus; member=GetConnectionUnixProcessID
       string ":1.20"
    signal sender=:1.95 -> dest=(null destination) serial=32 path=/com/google/co
de/AccountsSSO/SingleSignOn/AuthSession_2; interface=com.google.code.AccountsSSO
.SingleSignOn.AuthSession; member=stateChanged
       int32 8
       string "The request is started successfully"
    method return sender=:1.95 -> dest=:1.20 reply_serial=233
       array [
          dict entry(
             string "Secret"
             variant             string "mypassword"
          )
          dict entry(
             string "UserName"
             variant             string "myemail@something.com"
```

nccgroup
freedom from doubt

A few interesting binaries…

- /usr/bin/simkit [sgid privileged] – New-er. Research ongoing ☺
- /usr/bin/csd [suid root/gid disk] – Diagnostic utility (can also be triggered via *#*#310#*#* on dialer). Neat by itself.
- /usr/libexec/mapplauncherd/booster-silica-qt5 [suid root] – Used to support Silica extensions, uses maplauncherd
- /usr/bin/devel-su [suid root] – Custom SU. Written in C! No stack canaries or PIE

nccgroup
freedom from doubt

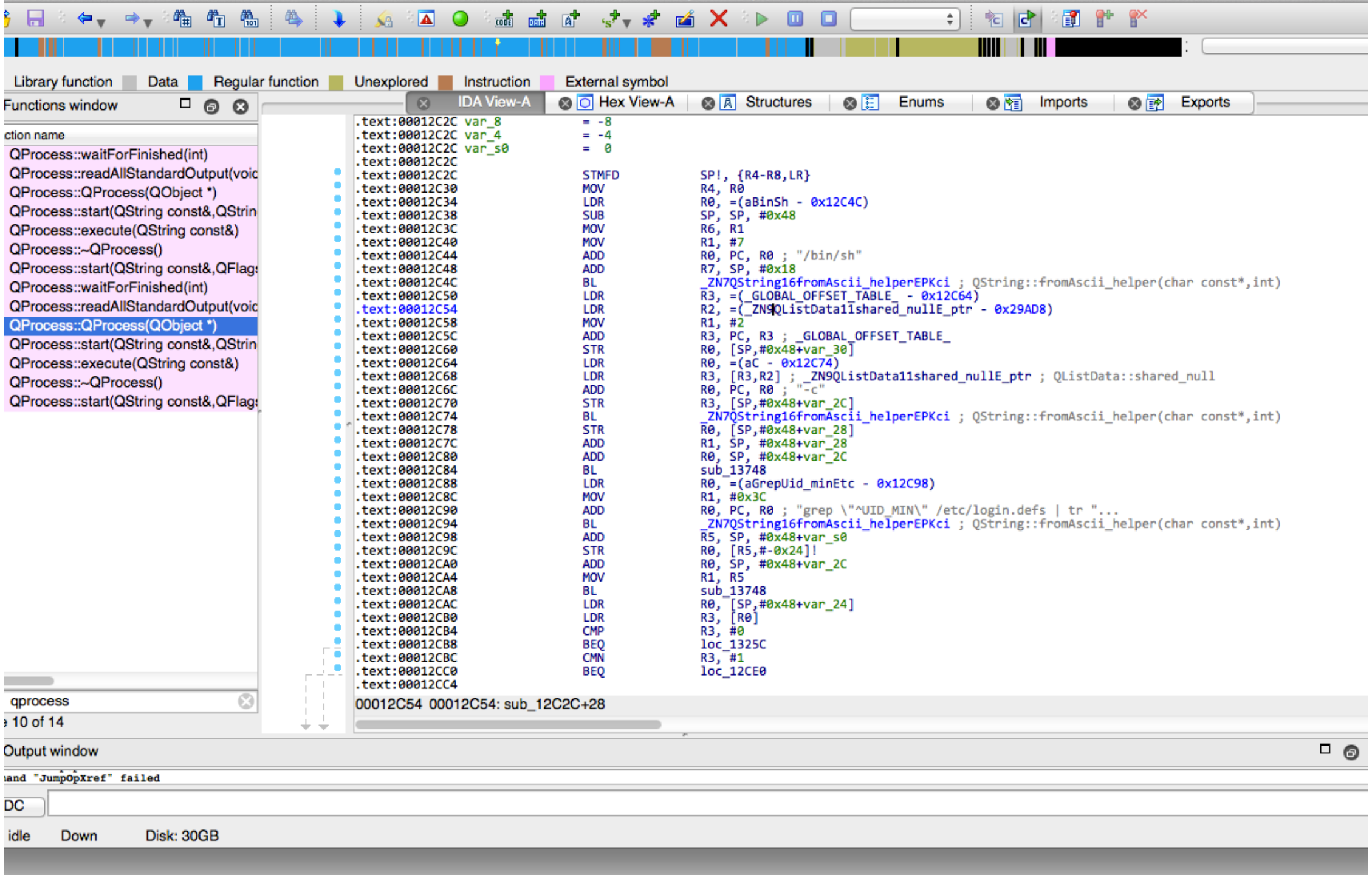
But what to do?

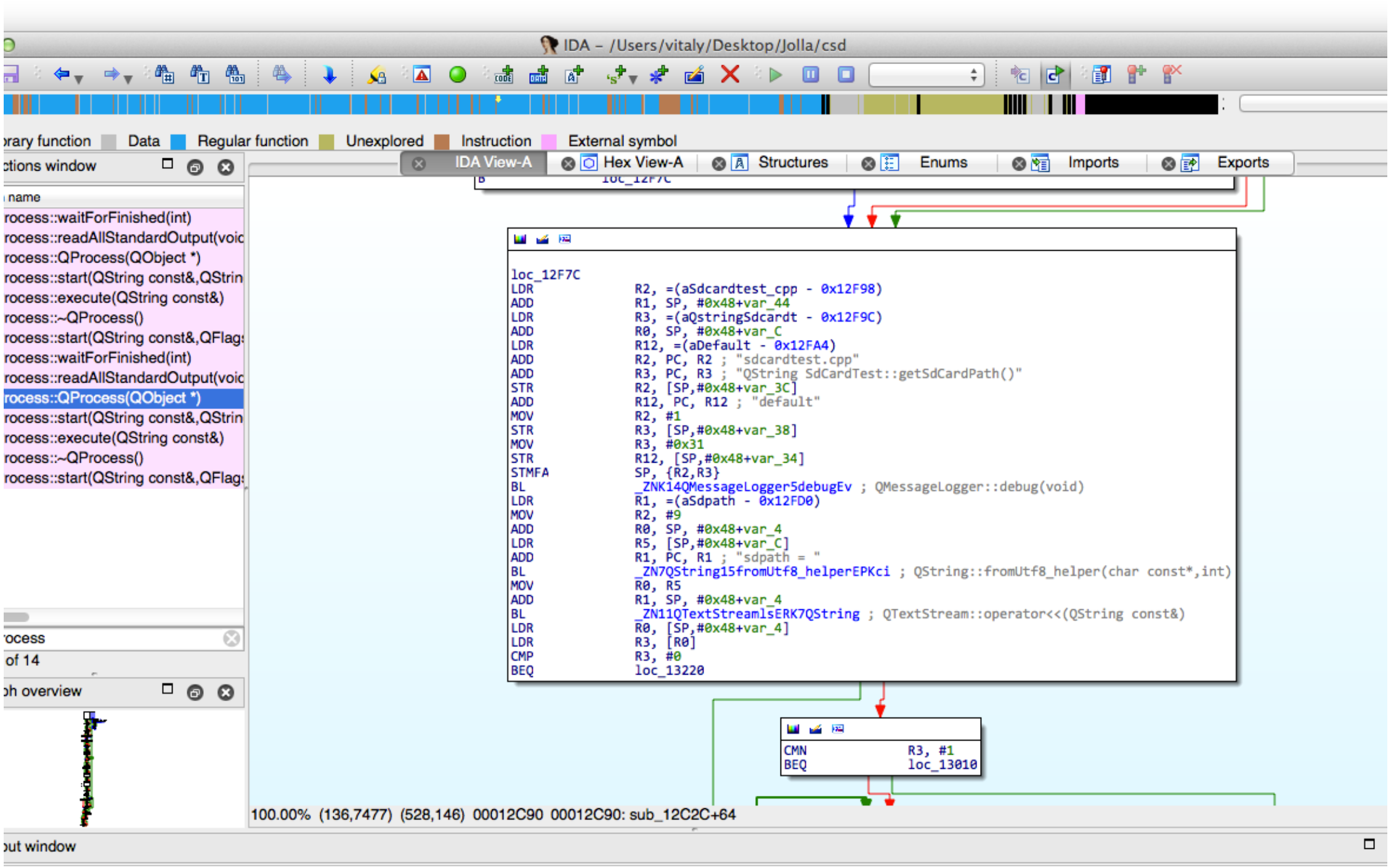
- Readelf, objdump, gdb, gdbserver available or install via pkcon (alternate repos available!)
- Memory corruption would be nice
    - Fuzz input
    - Fuzz environmental variables
    - Get more intelligent ☺
- But it's also very dangerous for suids to shell out
- We should look for system() and popen(), right?

nccgroup
freedom from doubt

Oh wait, C++ and QT

```
[nemo@Jolla ~]$ ls -al /usr/bin/csd
-rwsr-sr-x 1 root disk 140572 2014-05-21 13:52 /usr/bin/csd
[nemo@Jolla ~]$ readelf -a /usr/bin/csd | grep system
[nemo@Jolla ~]$ readelf -a /usr/bin/csd | grep popen
[nemo@Jolla ~]$ readelf -a /usr/bin/csd | grep QProcess
99: 00000000     0 FUNC    GLOBAL DEFAULT  UND _ZN8QProcess15waitForFini
   113: 00000000     0 FUNC    GLOBAL DEFAULT  UND _ZN8QProcess21readAllStan
   143: 00000000     0 FUNC    GLOBAL DEFAULT  UND _ZN8QProcessC1EP7QObject
   149: 00000000     0 FUNC    GLOBAL DEFAULT  UND _ZN8QProcess5startERK7QSt
   166: 00000000     0 FUNC    GLOBAL DEFAULT  UND _ZN8QProcess7executeERK7Q
   170: 00000000     0 FUNC    GLOBAL DEFAULT  UND _ZN8QProcessD1Ev
   235: 00000000     0 FUNC    GLOBAL DEFAULT  UND _ZN8QProcess5startERK7QSt
```


Library function
Data
Regular function
Unexplored
Instruction
External symbol
Functions window
ction name
QProcess::waitForFinished(int)
QProcess::readAllStandardOutput(voic
QProcess::QProcess(QObject *)
QProcess::start(QString const&,QStrin
QProcess::execute(QString const&)
QProcess::~QProcess()
QProcess::start(QString const&,QFlag
QProcess::waitForFinished(int)
QProcess::readAllStandardOutput(voic
QProcess::QProcess(QObject *)
QProcess::start(QString const&,QStrin
QProcess::execute(QString const&)
QProcess::~QProcess()
QProcess::start(QString const&,QFlag
qprocess
10 of 14
IDA View-A
Hex View-A
Structures
Enums
Imports
Exports
.text:00012C2C var_8 = -8
.text:00012C2C var_4 = -4
.text:00012C2C var_s0 = 0
.text:00012C2C
.text:00012C2C STMFD SP!, {R4-R8,LR}
.text:00012C30 MOV R4, R0
.text:00012C34 LDR R0, =(aBinSh - 0x12C4C)
.text:00012C38 SUB SP, SP, #0x48
.text:00012C3C MOV R6, R1
.text:00012C40 MOV R1, #7
.text:00012C44 ADD R0, PC, R0 ; "/bin/sh"
.text:00012C48 ADD R7, SP, #0x18
.text:00012C4C BL _ZN7QString16fromAscii_helperEPKci ; QString::fromAscii_helper(char const*,int)
.text:00012C50 LDR R3, =(_GLOBAL_OFFSET_TABLE_ - 0x12C64)
.text:00012C54 LDR R2, =(_ZN9QListData11shared_nullE_ptr - 0x29AD8)
.text:00012C58 MOV R1, #2
.text:00012C5C ADD R3, PC, R3 ; _GLOBAL_OFFSET_TABLE_
.text:00012C60 STR R0, [SP,#0x48+var_30]
.text:00012C64 LDR R0, =(aC - 0x12C74)
.text:00012C68 LDR R3, [R3,R2] ; _ZN9QListData11shared_nullE_ptr ; QListData::shared_null
.text:00012C6C ADD R0, PC, R0 ; "-c"
.text:00012C70 STR R3, [SP,#0x48+var_2C]
.text:00012C74 BL _ZN7QString16fromAscii_helperEPKci ; QString::fromAscii_helper(char const*,int)
.text:00012C78 STR R0, [SP,#0x48+var_28]
.text:00012C7C ADD R1, SP, #0x48+var_28
.text:00012C80 ADD R0, SP, #0x48+var_2C
.text:00012C84 BL sub_13748
.text:00012C88 LDR R0, =(aGrepUid_minEtc - 0x12C98)
.text:00012C8C MOV R1, #0x3C
.text:00012C90 ADD R0, PC, R0 ; "grep \"^UID_MIN\" /etc/login.defs | tr "...
.text:00012C94 BL _ZN7QString16fromAscii_helperEPKci ; QString::fromAscii_helper(char const*,int)
.text:00012C98 ADD R5, SP, #0x48+var_s0
.text:00012C9C STR R0, [R5,#-0x24]!
.text:00012CA0 ADD R0, SP, #0x48+var_2C
.text:00012CA4 MOV R1, R5
.text:00012CA8 BL sub_13748
.text:00012CAC LDR R0, [SP,#0x48+var_24]
.text:00012CB0 LDR R3, [R0]
.text:00012CB4 CMP R3, #0
.text:00012CB8 BEQ loc_1325C
.text:00012CBC CMN R3, #1
.text:00012CC0 BEQ loc_12CE0
.text:00012CC4
00012C54 00012C54: sub_12C2C+28
Output window
and "JumpOpXref" failed
DC
idle
Down
Disk: 30GB


IDA – /Users/vitaly/Desktop/Jolla/csd
rary function
Data
Regular function
Unexplored
Instruction
External symbol
ctions window
name
rocess::waitForFinished(int)
rocess::readAllStandardOutput(voic
rocess::QProcess(QObject *)
rocess::start(QString const&,QStrin
rocess::execute(QString const&)
rocess::~QProcess()
rocess::start(QString const&,QFlag
rocess::waitForFinished(int)
rocess::readAllStandardOutput(voic
rocess::QProcess(QObject *)
rocess::start(QString const&,QStrin
rocess::execute(QString const&)
rocess::~QProcess()
rocess::start(QString const&,QFlag
ocess
of 14
h overview
IDA View-A
Hex View-A
Structures
Enums
Imports
Exports
loc_12F7C
LDR R2, =(aSdcardtest_cpp - 0x12F98)
ADD R1, SP, #0x48+var_44
LDR R3, =(aQstringSdcardt - 0x12F9C)
ADD R0, SP, #0x48+var_C
LDR R12, =(aDefault - 0x12FA4)
ADD R2, PC, R2 ; "sdcardtest.cpp"
ADD R3, PC, R3 ; "QString SdCardTest::getSdCardPath()"
STR R2, [SP,#0x48+var_3C]
ADD R12, PC, R12 ; "default"
MOV R2, #1
STR R3, [SP,#0x48+var_38]
MOV R3, #0x31
STR R12, [SP,#0x48+var_34]
STMFA SP, {R2,R3}
BL _ZNK14QMessageLogger5debugEv ; QMessageLogger::debug(void)
LDR R1, =(aSdpath - 0x12FD0)
MOV R2, #9
ADD R0, SP, #0x48+var_4
LDR R5, [SP,#0x48+var_C]
ADD R1, PC, R1 ; "sdpath = "
BL _ZN7QString15fromUtf8_helperEPKci ; QString::fromUtf8_helper(char const*,int)
MOV R0, R5
ADD R1, SP, #0x48+var_4
BL _ZN11QTextStreamlsERK7QString ; QTextStream::operator<<(QString const&)
LDR R0, [SP,#0x48+var_4]
LDR R3, [R0]
CMP R3, #0
BEQ loc_13220
CMN R3, #1
BEQ loc_13010
100.00% (136,7477) (528,146) 00012C90 00012C90: sub_12C2C+64
ut window

nccgroup
freedom from doubt

Tried to have this executed…

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>

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

        setuid(0);
        setgid(6);

        FILE *f = fopen("flag", "w");

        fprintf(f, "UID, EUID: %d, %d\n", getuid(), geteuid());

        fclose(f);

        return(0);
}
```

nccgroup
freedom from doubt

Looked promising…

```
$ env PATH=.:$PATH /usr/bin/csd
[D] QWaylandEglIntegration::QWaylandEglIntegration:58 - Using Wayland-EGL
[W] QQmlImportDatabase::importPlugin:1697 - Module 'Sailfish.Silica' does not
contain a module identifier directive - it cannot be protected from external
registrations.
[D] FactoryUtils::getFlags:94 - FILE   said: "4436"
[D] FactoryUtils::isVerified:123 - Head =  "4436"
[D] FactoryUtils::writeCsdResults:55 - writeCsdResults:
"00000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000"
DeclarativeCoverWindow: I have a default alpha buffer
[D] FactoryUtils::writeCsdResults:55 - writeCsdResults:
"00000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000"
Clicked 28
[D] SdCardTest::getSdCardPath:49 - sdpath =  "/run/user//media/sdcard"
```

nccgroup
freedom from doubt

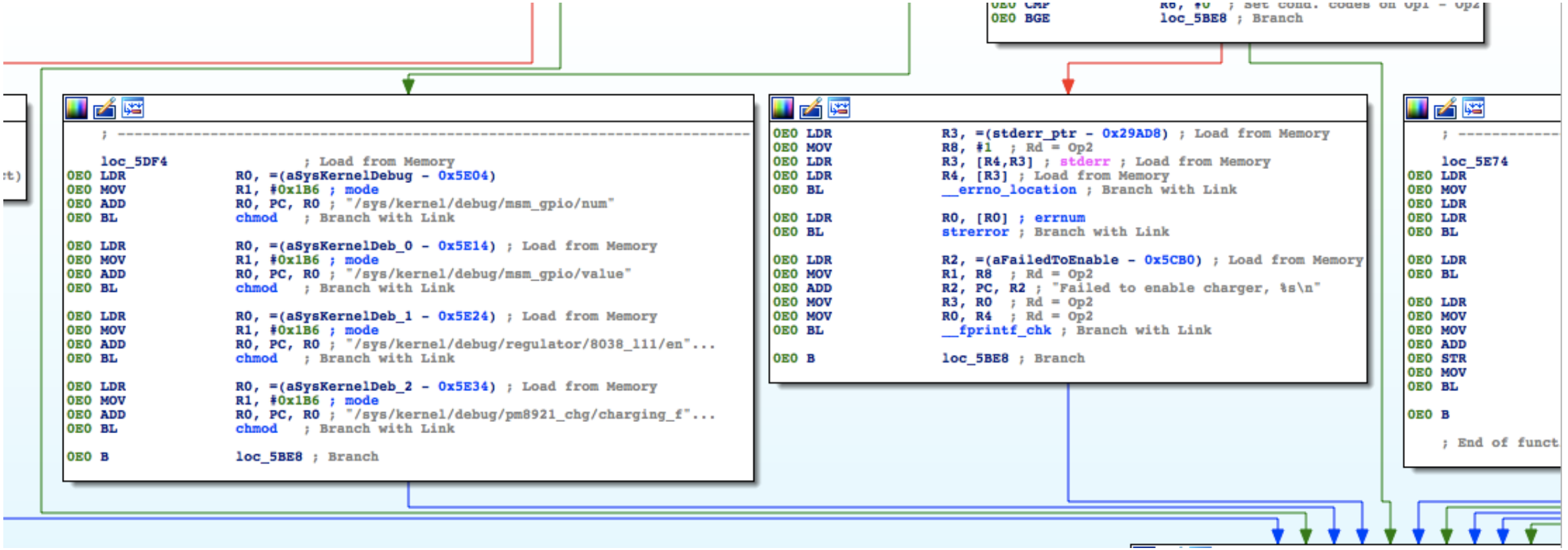
Nope ☹

- Content of “flag”: UID, EUID: 100000, 100000 (nemo)
- Drops privs
- Probably bash priv mode?

nccgroup
freedom from doubt

Turns out…

- Only doesn't drop privs for a few functions
- chmod()'s a few thing in /sys
- Not much you can do other than disable a charger…

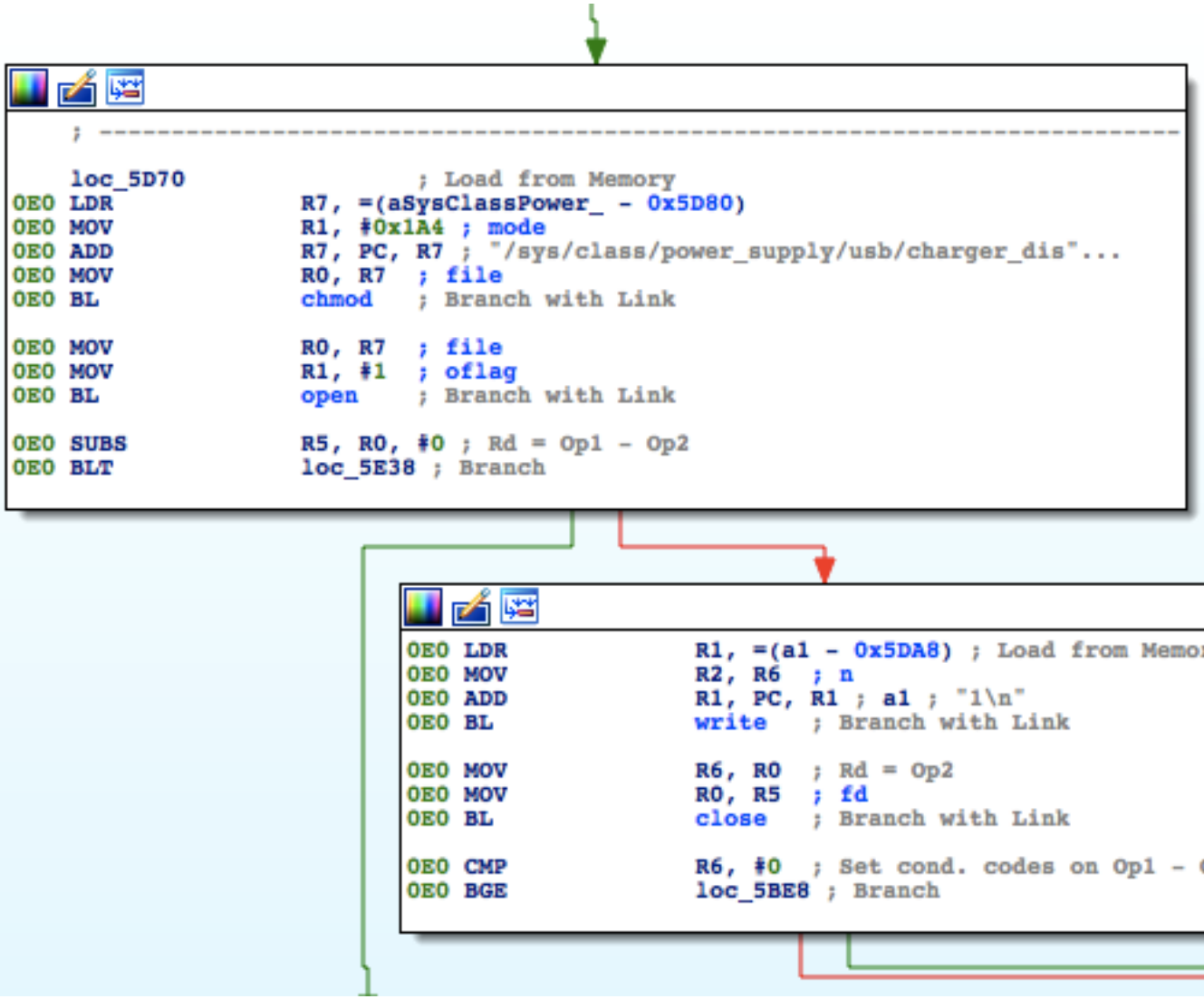


```
0E0 CMP             R6, #0  ; Set Cond. Codes on Op1 - Op2
0E0 BGE             loc_5BE8 ; Branch
```

```
    ; ---------------------------------------------------------------------------

    loc_5DF4                ; Load from Memory
0E0 LDR             R0, =(aSysKernelDebug - 0x5E04)
0E0 MOV             R1, #0x1B6 ; mode
0E0 ADD             R0, PC, R0 ; "/sys/kernel/debug/msm_gpio/num"
0E0 BL              chmod   ; Branch with Link

0E0 LDR             R0, =(aSysKernelDeb_0 - 0x5E14) ; Load from Memory
0E0 MOV             R1, #0x1B6 ; mode
0E0 ADD             R0, PC, R0 ; "/sys/kernel/debug/msm_gpio/value"
0E0 BL              chmod   ; Branch with Link

0E0 LDR             R0, =(aSysKernelDeb_1 - 0x5E24) ; Load from Memory
0E0 MOV             R1, #0x1B6 ; mode
0E0 ADD             R0, PC, R0 ; "/sys/kernel/debug/regulator/8038_l11/en"...
0E0 BL              chmod   ; Branch with Link

0E0 LDR             R0, =(aSysKernelDeb_2 - 0x5E34) ; Load from Memory
0E0 MOV             R1, #0x1B6 ; mode
0E0 ADD             R0, PC, R0 ; "/sys/kernel/debug/pm8921_chg/charging_f"...
0E0 BL              chmod   ; Branch with Link

0E0 B               loc_5BE8 ; Branch
```

```
0E0 LDR             R3, =(stderr_ptr - 0x29AD8) ; Load from Memory
0E0 MOV             R8, #1  ; Rd = Op2
0E0 LDR             R3, [R4,R3] ; stderr ; Load from Memory
0E0 LDR             R4, [R3] ; Load from Memory
0E0 BL              __errno_location ; Branch with Link

0E0 LDR             R0, [R0] ; errnum
0E0 BL              strerror ; Branch with Link

0E0 LDR             R2, =(aFailedToEnable - 0x5CB0) ; Load from Memory
0E0 MOV             R1, R8  ; Rd = Op2
0E0 ADD             R2, PC, R2 ; "Failed to enable charger, %s\n"
0E0 MOV             R3, R0  ; Rd = Op2
0E0 MOV             R0, R4  ; Rd = Op2
0E0 BL              __fprintf_chk ; Branch with Link

0E0 B               loc_5BE8 ; Branch
```

```
    ; ------------

    loc_5E74
0E0 LDR
0E0 MOV
0E0 LDR
0E0 LDR
0E0 BL

0E0 LDR
0E0 BL

0E0 LDR
0E0 MOV
0E0 MOV
0E0 ADD
0E0 STR
0E0 MOV
0E0 BL

0E0 B

    ; End of funct
```


; ---
loc_5D70 ; Load from Memory
0E0 LDR R7, =(aSysClassPower_ - 0x5D80)
0E0 MOV R1, #0x1A4 ; mode
0E0 ADD R7, PC, R7 ; "/sys/class/power_supply/usb/charger_dis"...
0E0 MOV R0, R7 ; file
0E0 BL chmod ; Branch with Link
0E0 MOV R0, R7 ; file
0E0 MOV R1, #1 ; oflag
0E0 BL open ; Branch with Link
0E0 SUBS R5, R0, #0 ; Rd = Op1 - Op2
0E0 BLT loc_5E38 ; Branch
0E0 LDR R1, =(a1 - 0x5DA8) ; Load from Memory
0E0 MOV R2, R6 ; n
0E0 ADD R1, PC, R1 ; a1 ; "1\n"
0E0 BL write ; Branch with Link
0E0 MOV R6, R0 ; Rd = Op2
0E0 MOV R0, R5 ; fd
0E0 BL close ; Branch with Link
0E0 CMP R6, #0 ; Set cond. codes on Op1 - Op2
0E0 BGE loc_5BE8 ; Branch

nccgroup
freedom from doubt

Shellshock

- It was vulnerable
- Couldn't find anything to use it on: nothing suid loaded env vars, dhclient not in use
- Maybe missed opportunity with some binaries that run from invoker. Or CSD.
- Patched in latest hotfix

nccgroup
freedom from doubt

What about the kernel?

```
# lsmod
Module                  Size  Used by
wlan                 2592646  0
cfg80211              144905  1 wlan

# uname -a
Linux Jolla 3.4.91.20140612.1 #1 SMP PREEMPT Mon Jun 16 17:24:16 UTC 2014 armv7l
armv7l armv7l GNU/Linux
```

nccgroup
freedom from doubt

Patched for most modern CVEs

```
$ for ((i = 39; i <= 150; i++)); do ./trigger_sock_diag
$i; done
Sending with family 39
Sending with family 40
Sending with family 41
Sending with family 42
Sending with family 43
Sending with family 44
Sending with family 45
Sending with family 46
.....
```

nccgroup
freedom from doubt

Attacking Sailfish users

- Mapping out attack surface
- One possibility: /usr/bin/jolla-settings parses VCF (vCard) files
- No crashes yet!

nccgroup
freedom from doubt

Pentesting Applications

- Traffic can be captured as easily as on any other Linux system
- Setup proxies for HTTP/HTTPS connections(we all know how to do that)
- Create your own IPTables rules and scripts to forward anything wherever you want
- Get Dynamic: Fashion Scripts, to load rules when certain applications run

More on proxying

- Browser traffic: .js file
- General traffic: long-hold WiFi SSID, click Edit
- Cert pinning (or client-side certs?) – Store, Updates
- Weirdness: if you check for updates, the actual updates are NOT cert pinned (snagged the RPMs this way)
- As an aside: it sends your Jolla creds with a hashed password
- Installing CA cert (like Burp's) is easy. Look online.
    - put in **/etc/pki/tls/certs/**
    - run **multi_c_rehash**

nccgroup
freedom from doubt

Third-Party Apps

- 3rd Party Apps
  - We reached out to Jolla to ask them what the lifecycle was like.
    - They seemed unsure of what we were asking…

nccgroup
freedom from doubt

Pentesting Applications

- Sailfish Quirks…
  - Everything is run through ‘invoker’
  - invoker was primarily designed to boost app startup times and save device memory
  - Also invoker handles Group and User Privs, such as access to the credentials store or contacts DB
- What is Invoker really?
  - Turns out invoker is basically just a wrapper to ‘mapplauncherd’
  - The invoker binary takes the app name and a default set of options in the invoker binary and passes them to mapplauncherd

nccgroup
freedom from doubt

Pentesting Applications

- It's similar to pentesting any linux system application:
  - Evaluate File Permissions
  - Use Old Friends like:
    - GDB
    - LDD
    - Strace
    - Strings
    - Etc…
    - Then find the location of the applications QML files and it's code review time

nccgroup
freedom from doubt

Pentesting Applications

- Android Hotness/Alien Dalvik
  - Alien Dalvik system is proprietary Myriad Group tech.
  - Binaries and libraries located in the '/system/' folder
  - Every Android app runs as a different user (user id > 10000)
  - Virtual to physical memory mapping controlled through '/usr/bin/vmtouch'
    - Vmtouch loads the system libraries and android libraries so your hardware works for both native and android apps seamlessly
    - Runs as root!
  - Communications for apps to the Sailfish OS and hardware done through the android-bridge('/opt/alien/system/genv/bin/alien_bridge_server')
  - A new bridge is stood up for each app that is launched

nccgroup
freedom from doubt

Pentesting Applications

- Android Hotness/Alien Dalvik(cont)
  - Also runs as root!
- Potential for malicious app disaster? You bet!
  - Possible attack vectors:
    - Remap huge chunks of memory
    - Request the android bridge to access system devices as root
    - Request the android bridge to access anything as root
  - Mitigations in place?
    - Only the linux user and file permissions stand in your way.
      - (cause those have never been bypassed…. Ever… right?) ;-)

nccgroup
freedom from doubt

Pentesting Applications

- Additional applications of the Alien Dalvik?
  - Obfuscated android app analysis!!
    - Ever mess with “kony” apps?
    - WTF is “kony”?

No, not this guy…

nccgroup
freedom from doubt

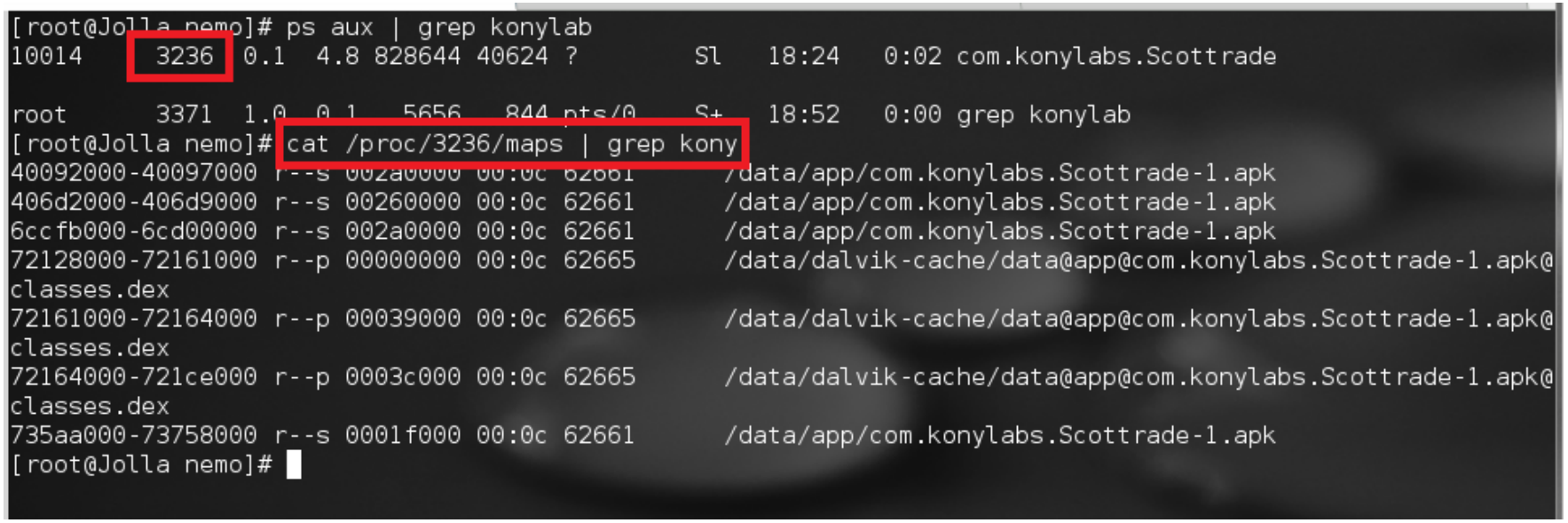
Pentesting Applications

- Lua bytecode VMs built into apps… in short, it sucks to analyze
- This is cool, like seriously cool, why you ask?
  - Traditional Android app analysis is mostly about fighting against the phone
  - Uses a top-down approach
- How Alien Dalvik helps?
  - Using a bottom-up approach, and working with the phone's built in developer tools we can easily access all the information about the application

nccgroup
freedom from doubt

Pentesting Applications

- Simplest way to use this bottom-up technique
  - Launch your app on the phone
  - Find the PID in the dev mode shell
  - Find the memory map for the PID



nccgroup
freedom from doubt

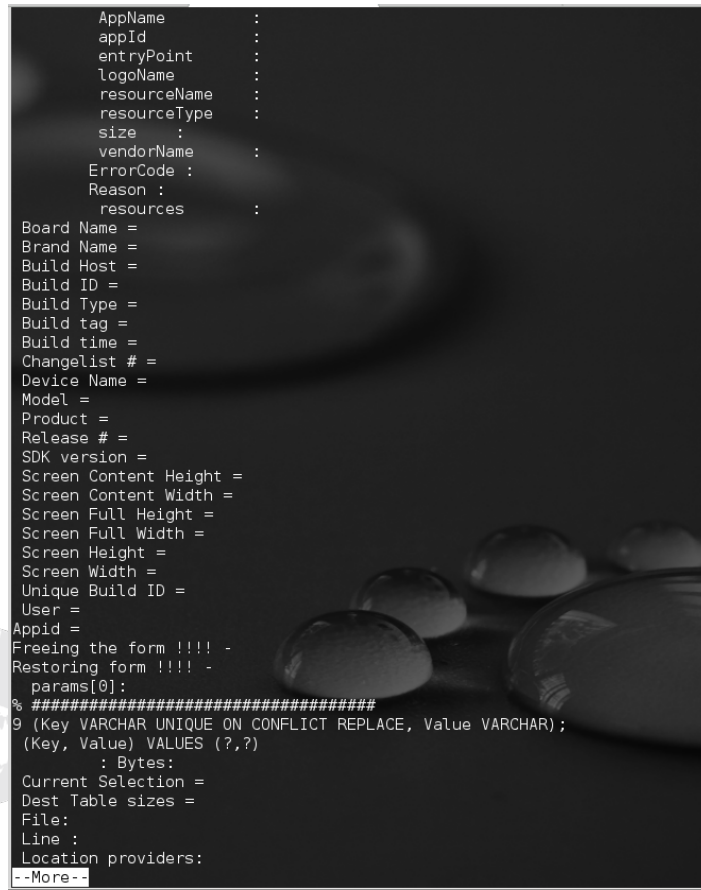
Pentesting Applications

- Use GDB to dump all the memory ranges

```
[root@Jolla nemo]# gdb --pid 3236
GNU gdb (GDB) Mer (7.5.1+git3)
Copyright (C) 2012 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show copying"
and "show warranty" for details.
This GDB was configured as "armv7hl-meego-linux-gnueabi".
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Attaching to process 3236
Reading symbols from /opt/alien/system/bin/alien-main...(no debugging symbols found)...done.

warning: .dynamic section for "/opt/alien/system/bin/linker" is not at the expected address (
 or version mismatch?)

warning: Could not load shared library symbols for 88 libraries, e.g. libc.so.
Use the "info sharedlibrary" command to see the complete listing.
Do you need "set solib-search-path" or "set sysroot"?
Reading symbols from /opt/alien/system/bin/linker...(no debugging symbols found)...done.
Loaded symbols for /opt/alien/system/bin/linker
0x401b0a60 in ?? ()
(gdb) dump memory /root/konytest1 0x6ccfb000 0x6cd00000
(gdb)
```

Pentesting Applications

- Transfer the memory dumps back to your host machine
- Reassemble the dumps, and analyze
  - Binwalk
  - Strings
  - Hexeditor
  - etc



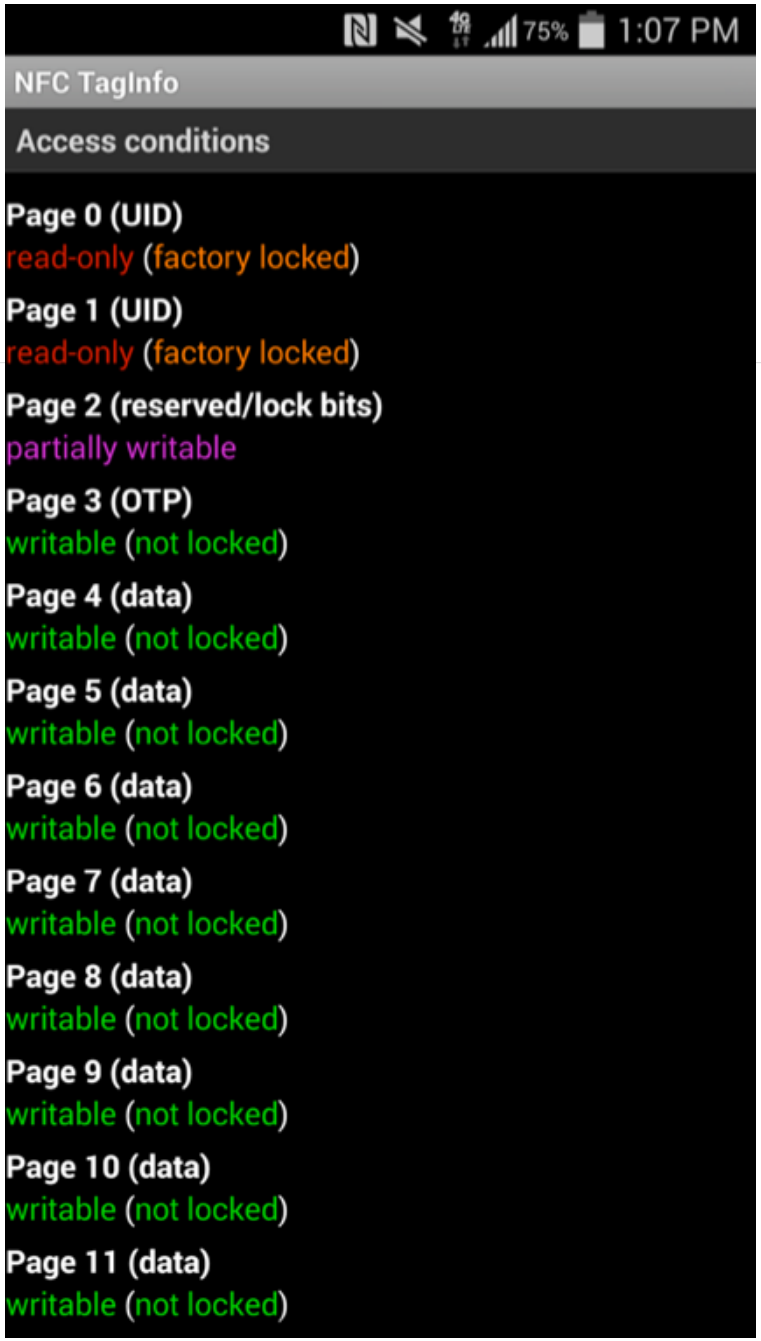
nccgroup
freedom from doubt

```
root@kali:~# binwalk konytest5

DECIMAL       HEX             DESCRIPTION
--------------------------------------------------------------------------------
11560         0x2D28          LZMA compressed data, properties: 0x65, dictionary size: 33554432 bytes, uncompressed
 size: 16777728 bytes
14325         0x37F5          LZMA compressed data, properties: 0xB5, dictionary size: 131072 bytes, uncompressed s
ize: 65537 bytes
113685        0x1BC15         LZMA compressed data, properties: 0xB5, dictionary size: 262144 bytes, uncompressed s
ize: 131074 bytes
123173        0x1E125         LZMA compressed data, properties: 0xB6, dictionary size: 262144 bytes, uncompressed s
ize: 131074 bytes
202480        0x316F0         LZMA compressed data, properties: 0x65, dictionary size: 33554432 bytes, uncompressed
 size: 16777472 bytes

root@kali:~#
```

nccgroup
freedom from doubt

Other Half: NFC

- NFC sticker tells phone what theme to download
- NFC radio only active when switched pressed
- Sticker is standard MiFARE Ultralight
- Handled by tohd daemon
- NFC stack in N9 fuzzed by Charlie Miller, no results. Different in Sailfish?



nccgroup
freedom from doubt

nccgroup
freedom from doubt

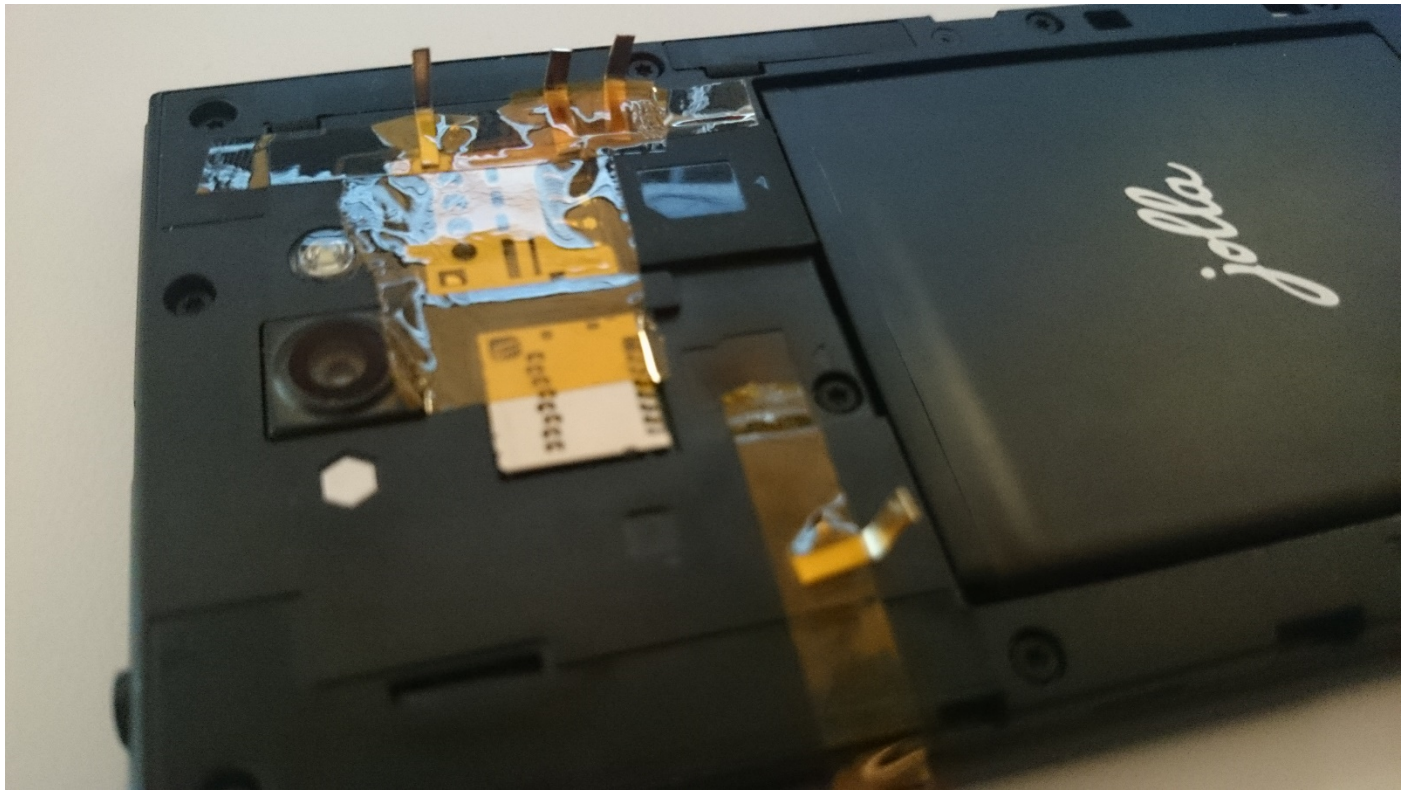
Other Halves

- I2C Port
  - Start by downloading the TOH Developer Kit:
    - Realize that is useless for I2C stuff
    - Develop your own methodology
  - Where my I2C fuzzers at google?
    - Seems like no one has ever bothered to fuzz I2C
    - Start by writing the dumbest I2C fuzzer ever
      - Materials:
      - Bus Pirate (Wanted to implement on an FPGA but my VHDL/ Verilog is garbage)
      - Logic Analyzer
      - Jolla in Developer Mode
      - GDB
      - Python(pyBusPirate)

Other Halves

nccgroup
freedom from doubt

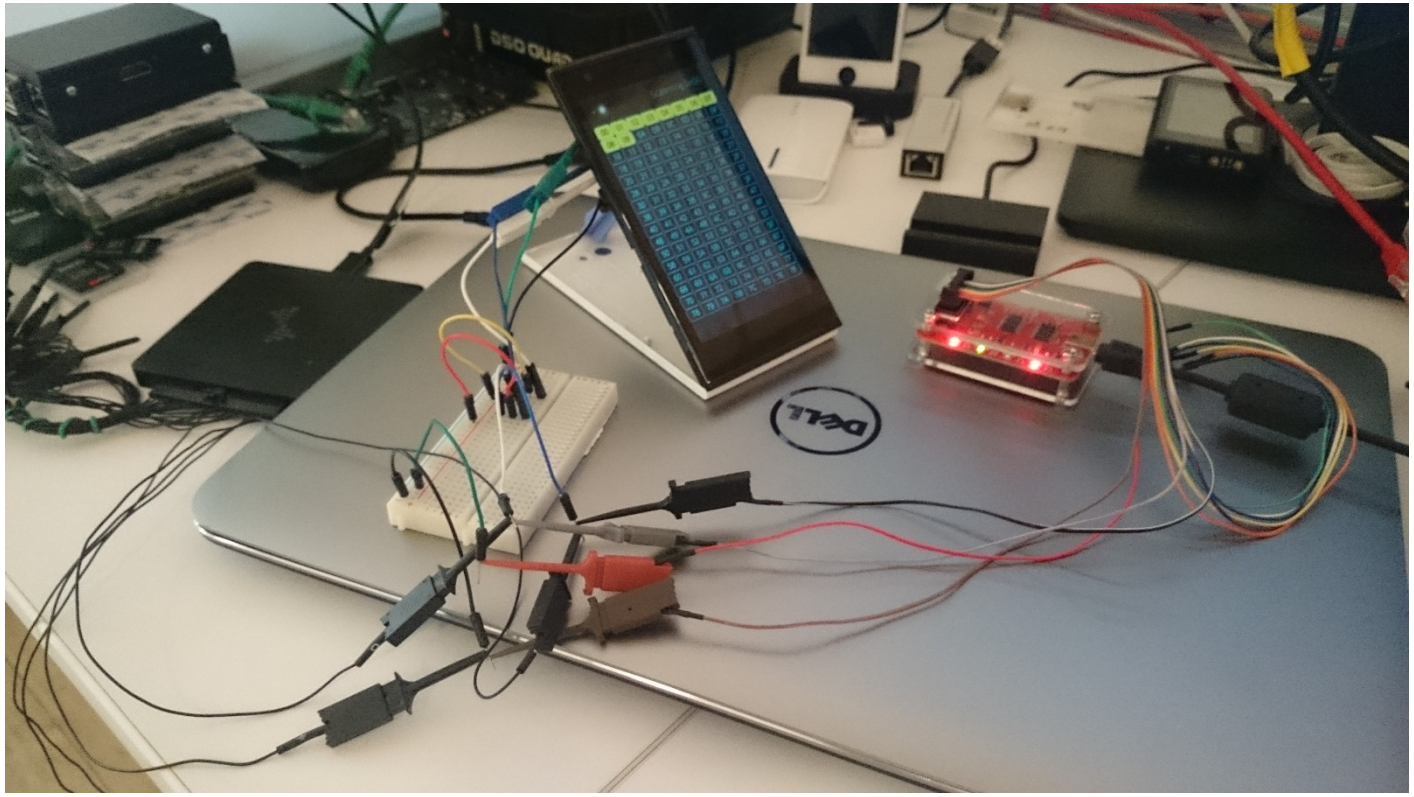
- I2C pins expanded



The Other Half

nccgroup
freedom from doubt

- Terrible test rig ☺



nccgroup
freedom from doubt

The Other Half

- Implement Dumb Fuzzer to send I2C data through Bus Pirate
- Hook up the logic analyzer to ensure the packets you think you're sending are being sent
- Monitor Phone and Processes to see what happens
    - Nothing happens
- Investigate

nccgroup
freedom from doubt

The Other Half: I2C

- Turns out, the phone has an Other Half Switch
- This switch *briefly* activates the I2C poll mechanism

The Other Half: I2C

- Figure out a way to bypass the Other Half Switch
  - Toggle switch by hand with wooden stick
  - Then repeat previous testing process
- Still… nothing happens…
  - The Jolla I2C port is designed to run in Master Mode, not Master/Slave.
  - This means our fuzzer needs to get a bit smarter, and wait for the initialization poll before firing of it's data
- Future Work. Rewrite the fuzzer to be smarter, iterate through single client and multi-client communications

nccgroup
freedom from doubt

Taking it Further

- Jolla PwnPhone?
  - This used to be easily done.
  - Neildk repo from open repos was nearly all that was needed
  - Everything else would need to be cross compile from the emulator or built on the device
    - Either way that's a pain in the A$$
  - However… Then update 1.0.8.19 came out…
    - Implementation of Polkit
    - Disallowed installation of apps from sources that didn't have a polkit config

nccgroup
freedom from doubt

Taking it Further

- WTF is polkit?
  - Also known as policy kit.
  - It's a component for controlling system-wide privileges in *NIX systems
- How does this effect your pwnphone plans?
  - Zypper(used for the open-repos packages) needs to implement proper polkit API calls in order for packages to properly install

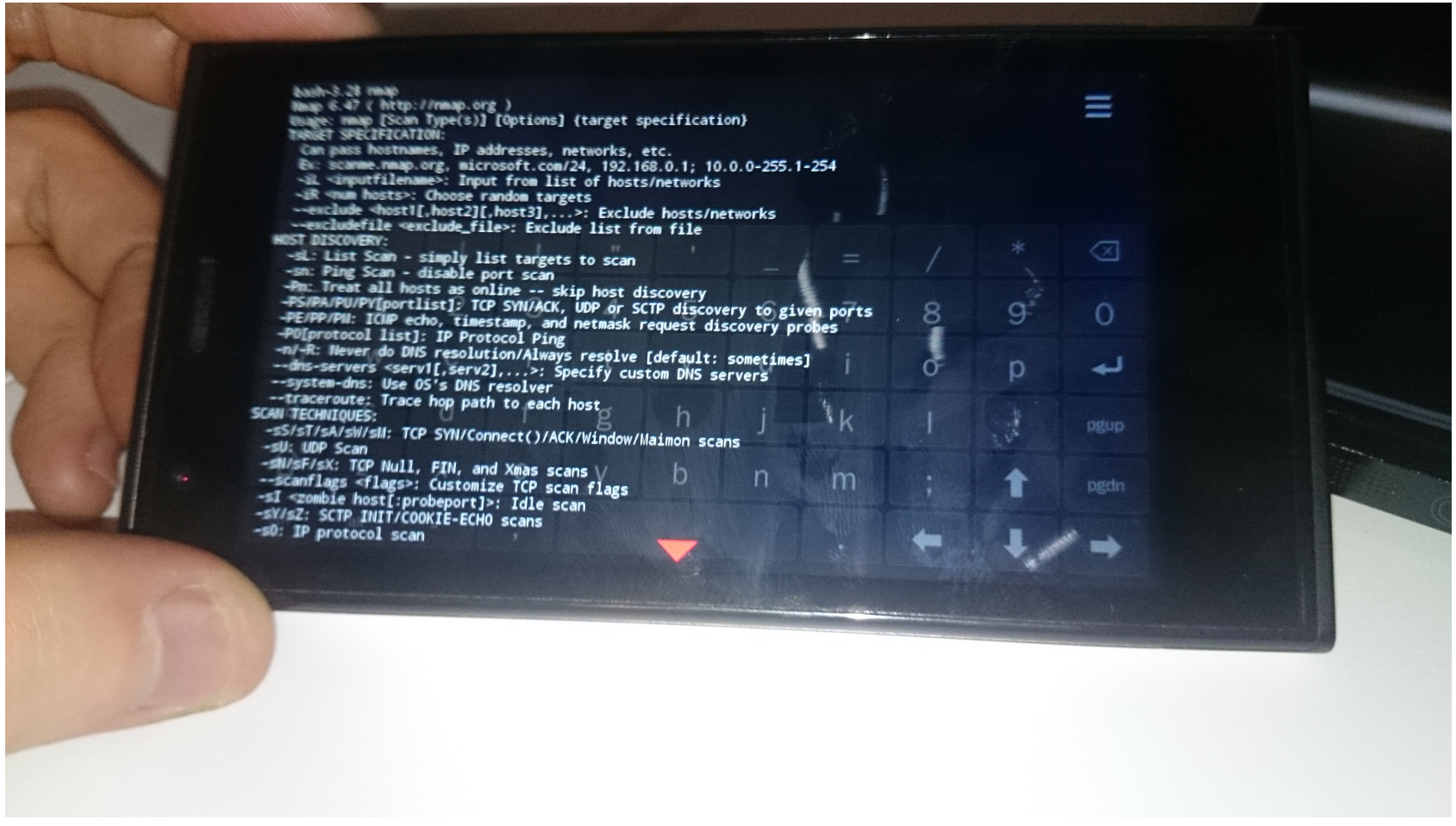
nccgroup
freedom from doubt

Jolla PwnPhone (quick and dirty)

- No need to actually bypass polkit
  - backup your data
  - revert back to factory image
  - install the open-repos
  - re-upgrade the firmware

nccgroup
freedom from doubt

Jolla PwnPhone



nccgroup
freedom from doubt

Taking it Further

- Hardening the Jolla/Sailfish
  - Building Hardened Kernels
    - Technically, Jolla is already working on this
      - I think if they crowd-sourced this, it'd move much faster
    - Difficult due to lack of sources available, QCOM, NDA stuff, etc…
    - However since the Bootloader is easily Unlocked…
      - Attempt at your own risk
      - Lots of information on the Maemo and Nemo Mobile sites

nccgroup
freedom from doubt

But.. where are the bugs?

- Why were there no bugs disclosed in your presentation? So Jolla has no bugs?
- Fairly well designed, but no claims about lack of bugs
- Will work with Jolla on anything we find — great company to work with so far!

nccgroup
freedom from doubt

Thank you!
Questions? Comments? Complaints?