Blowing up the Celly

Building Your Own SMS/MMS Fuzzer

Brian Gorenc, Manager, Vulnerability Research
Matt Molinyawe, Security Researcher

Agenda

- Introduction
- Testing Environment
- Bug Hunting
- Live Demonstration
- Key Takeaways

Introduction

whois Brian Gorenc

Employer:	HP
Organization:	HP Security Research Zero Day Initiative
Responsibilities:	Manager, Vulnerability Research Organizing Pwn2Own Hacking Competition Verifying EIP == 0x41414141
Free Time:	Endlessly following code paths that don't lead to vulnerabilities
Twitter:	@MaliciousInput, @thezdi

whois Matt Molinyawe

Employer:	HP
Organization:	HP Security Research Zero Day Initiative
Responsibilities:	Security Researcher Enjoying funny and awesome proof of concepts Measuring my productivity in hours of YouTube watched Process Janitor – Make exploits shine and not crash Calc Connoisseur
Free Time:	DJ Manila Ice – Two time United States Finalist DJ Beat Contra using only the laser without death Beat QWOP Martial Arts
Twitter:	@djmanilaice

“Do-It-Yourself”

Fuzzing SMS/MMS is an interesting topic

Always-on technology

Limited in-line defenses

Every researcher will have a different take on the problem

Usually roll their own fuzzer along with mutation logic

Aim for this talk is to demonstrate approaches to get started in phone fuzzing

Using Android as the reference device for research/demonstration

Testing Environment

Virtual Lab and Configuration

Android Emulation

Easy to attain-> http://developer.android.com/sdk

Creating Virtual ARM devices is simple:

- android create avd –n MyDeviceName –t android-19 –b default/armeabi-v7a
- Use the UI with: android avd

Write scripts to generate the AVDs and to power them on

iOS Emulation

No default Messaging app on emulator

Windows Phone Emulation

Pull the SDK from here: http://dev.windowsphone.com/en-us/downloadsdk

Android Emulator Options

Cheaper than phones because they're free to create

Android SDK

Benefit of testing with several API versions

- ARM images
- x86 images

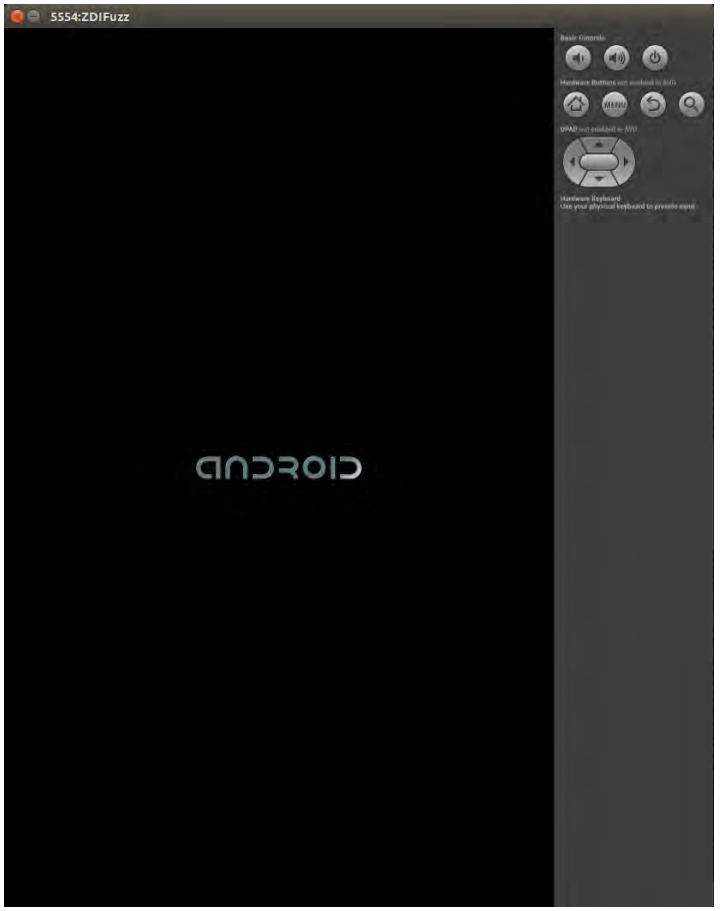
Emulations tend to be slow

Genymotion

Fast x86 Virtualbox Virtual Machines

User-friendly interface

Available at genymotion.com



Debugging

Attaching a debugger to the Virtual Device

On the Android Virtual Device:

- Shell into device
- Run gdbserver attached to the process “com.android.mms”
  - gdbserver :5039 –attach 1234
- Forward traffic to a tcp port
  - adb forward tcp:5039 tcp:5039

On your host machine:

- Download Android NDK: http://developer.android.com/tools/sdk/ndk/index.html
- Run a prebuilt gdb in there: arm-linux-androideabi-gdb for example
- Run the following command in the debug session:
  - target remote :5039

Attach, control and catch output of the debugger with Python.

Push debugger output to webapp/database.

Now you’re debugging!

Scripting/Automation for Emulators

SMS fuzzing on emulators:

Send PDU formatted messages with “send pdu” over the telnet channel

- Lots of prior research in this area.

Initial fails with MMS – Repetitive failures you learn from can lead to your success

Tried for weeks to get MMS networking working with emulators. It’s ok to give up sometimes.

Backing up your MMSs

Look at EasyBackup

- Installed this application to an emulator
- Was able to restore my MMS messages from my phone to an emulator
- Win!!! Yes it’s possible to create MMS messages on the emulator!

Looked at code and other things on the net

Was able to determine you can just manipulate mmssms.db (a sqlite database) without having to write Java (Hooray! Matt is a burnt out Sun Certified Enterprise Architect)

Scripting/Automation for Emulators

Save clean mmssms.db and compare with changed database

adb pull your clean database, make changes and then push the new database

- Interesting directories
  - /data/data/com.android.providers.telephony/databases – where mmssms.db is
  - /data/data/com.android.providers.telephony/app_parts – where attachments go

Send MMS to fake number

Alter tables: pdu, addr, part, canonical_addresses, and threads

- Easy to automate this with Python and sqlite3

Push the altered mmssms.db back to the phone

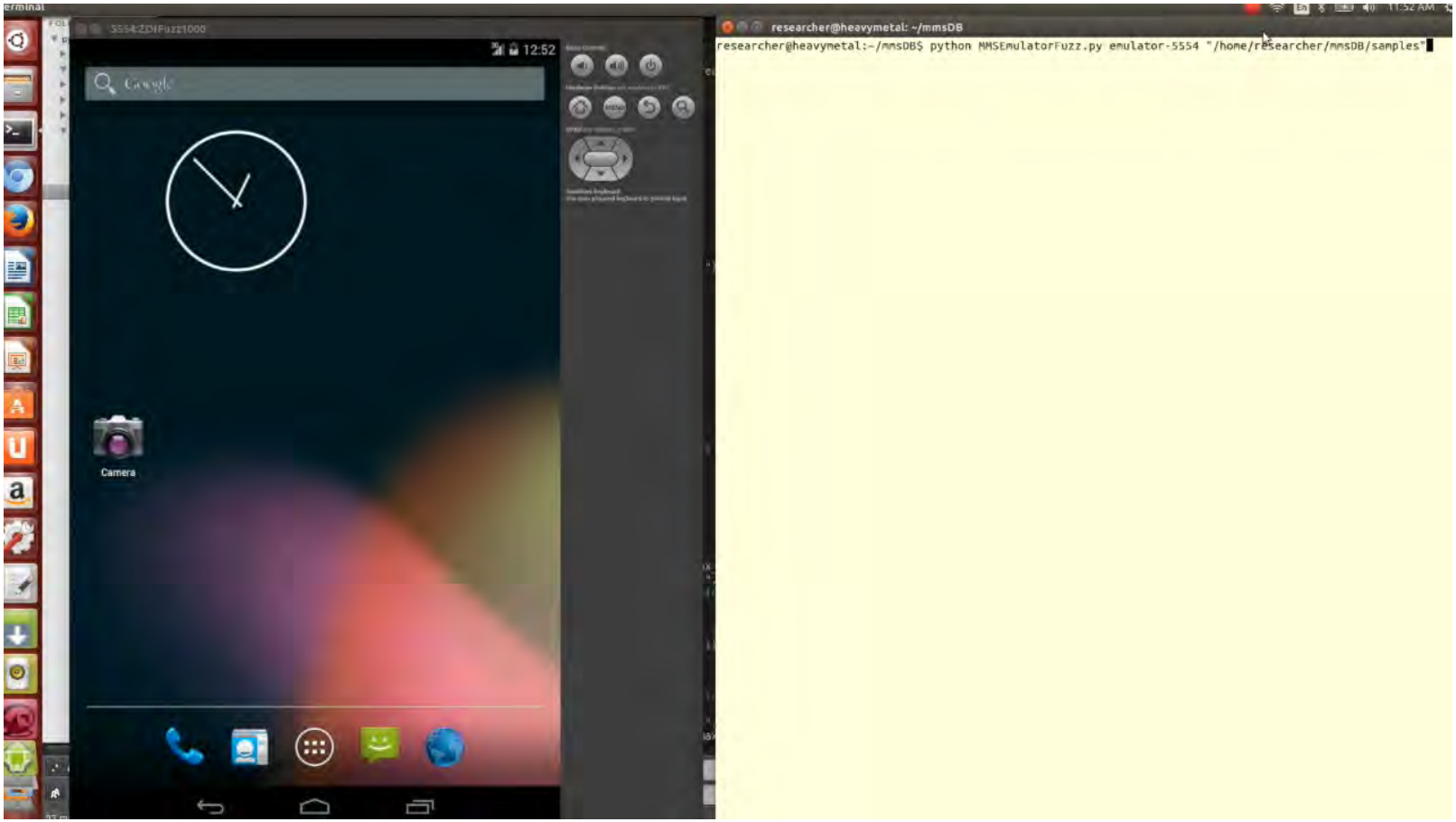
Make sure your set permissions back to radio:radio

Monkeyrunner

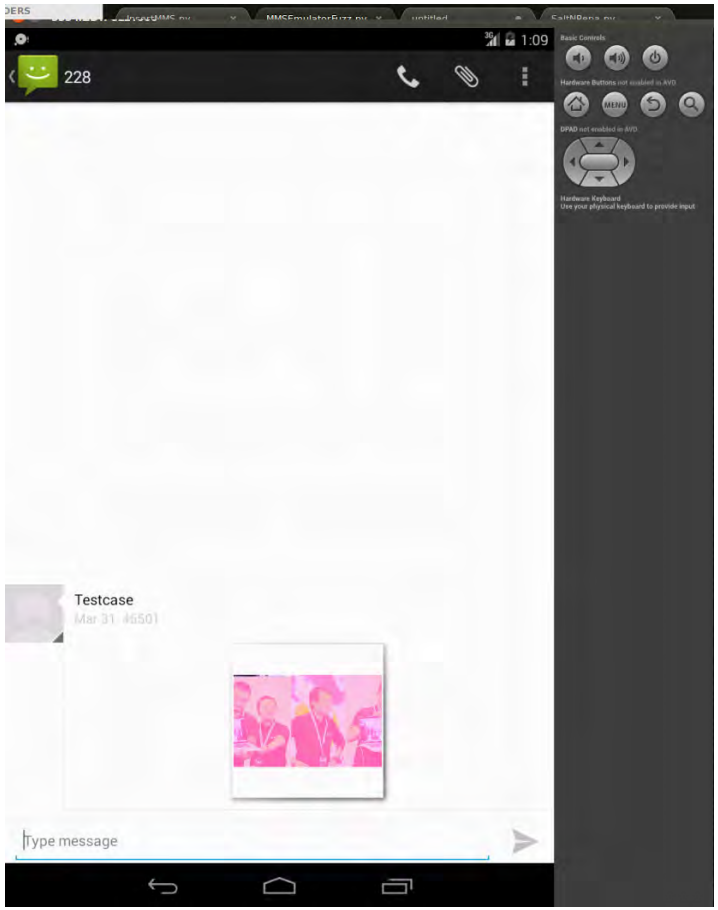
http://developer.android.com/tools/help/monkeyrunner_concepts.html

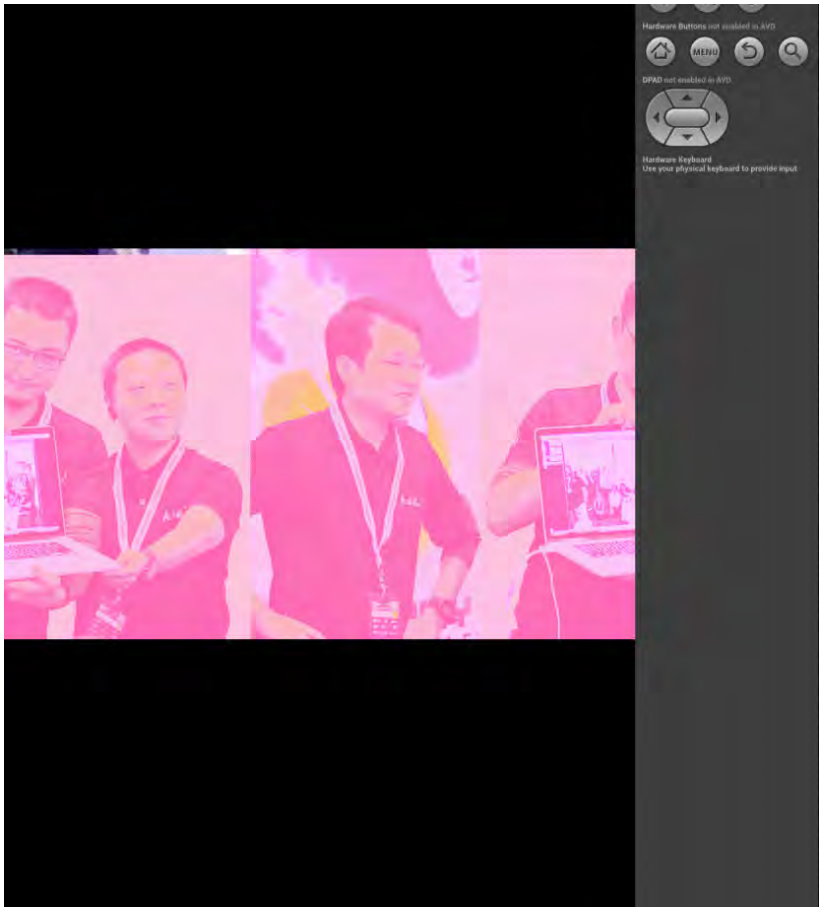
- Use this to click on the phone or to send text
- Effectively it is Jython scriptable automation in SDK tools

Multimedia Fuzz Case Generation and Deployment



Mangled Test Case





Real-World Lab and Configuration

Hardware

RX/TX

- Universal Software Radio Peripheral (USRP)
- BladeRF
- RangeNetworks Device

Emissions Control

- RF Enclosure

Photo: HP ZDI

Software

OpenBTS - http://www.openbts.org/

Base Station Information - http://openbsc.osmocom.org/trac

NanoBTS - http://openbsc.osmocom.org/trac/wiki/nanoBTS

Debugging Tools – usually come with the platform or you pay for one

Cell Phones and other materials

Your favorite cellphone target to fuzz

SIM cards

OpenBTS

Setting up OpenBTS

https://github.com/RangeNetworks/dev/wiki

Used Ubuntu 12.04 32-bit on a VM

Building and Finding Binaries for OpenBTS

These were heavily referenced

- https://wush.net/trac/rangepublic/wiki/BuildInstallRun
- svn co http://wush.net/svn/range/software/public

Built with --with-uhd (Ettus N210 USRP)

For ease, we built the transceiver from the svn checkout and installed the 4.0 binaries

UHD Drivers for Ettus N210 support

Available here: http://code.ettus.com/redmine/ettus/projects/uhd/wiki/UHD_Linux

Use the following commands to talk with the USRP once UHD drivers are built:

- uhd_find_device
- uhd_usrp_probe

USRP/Antennas/Cabling

Ettus N210 USRP

VERT900 Antennae

SMA Cable

RF Enclosures

Ramsey STE3000FAV: http://www.ramseytest.com/product.php?pid=10

Cells Phones/SIM Cards

Take your pick on Cell phones

Android

iPhone

Windows Phone

etc.

GSM

We set up a GSM network to look like an AT&T Network with the USRP in the enclosure

- Set GSM.Identity.MCC to 310
- Set GSM.Identity.MNC to 410

SIM Cards

Purchase these from “big box” stores

Our Bill of Materials

USRP and Accessories

USRP N210 Kit (782747-01) - $1,717.00

WBX-40 USRP Daughterboard - $480.00

USRP GPS-Disciplined Oscillator Kit - $758.00

SMA-to-SMA Cable Assembly - $30.00

VERT900 Vertical Antenna Dualband - $35.00

Total: $3,020.00

RF Enclosure and Accessories

STE3000FAV - $2,495.00

SMA Feedthrough Connectors

DB9 10 PF and DB9 100 PF Connectors

USB, RJ45 Adapter Kits

Total: $3,096.00

Cell Phones and SIMs

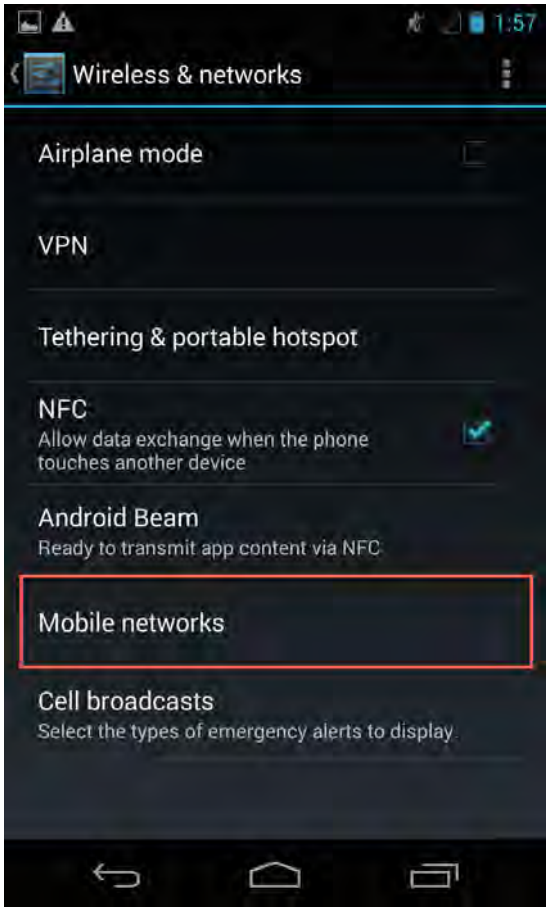
Unlocked Phones ~ $500

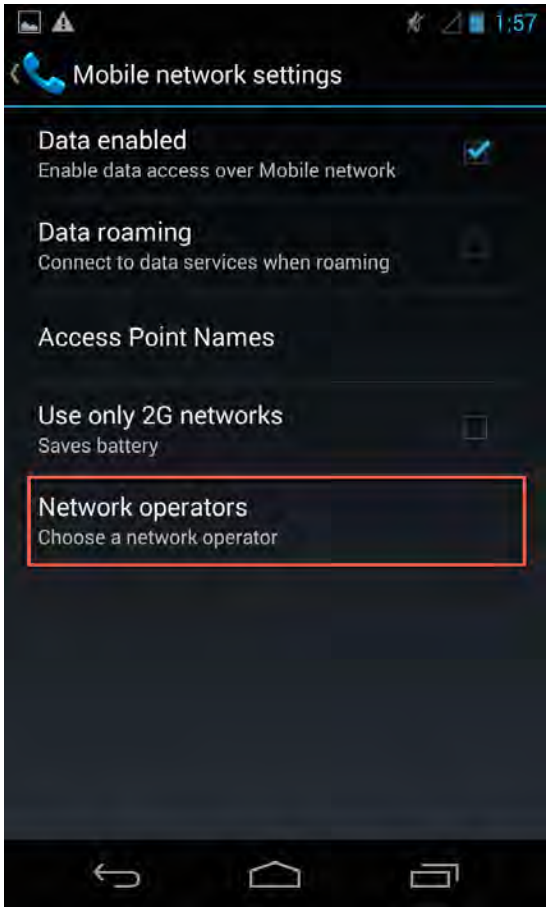
Pre-paid SIMs ~ $10-$20

Micro SIM Cutter Tool ~ $5

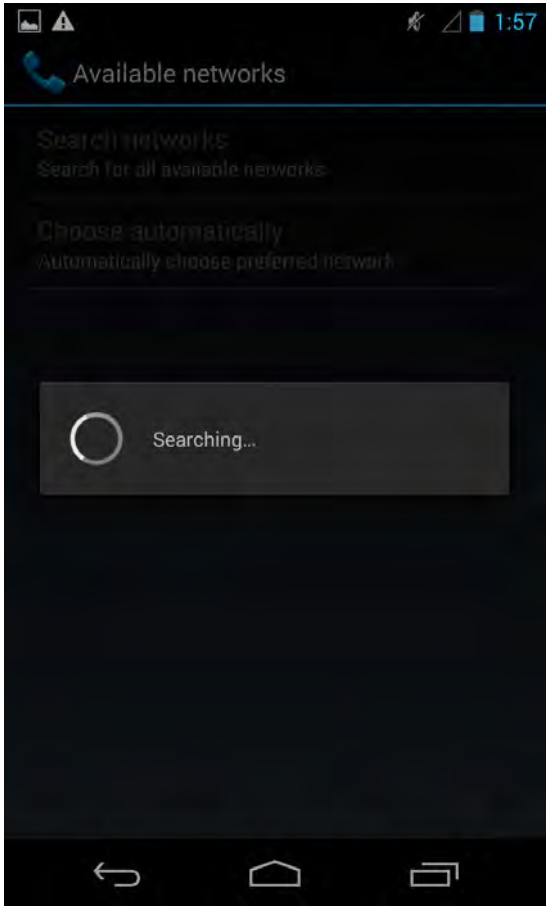
Total: ~$550

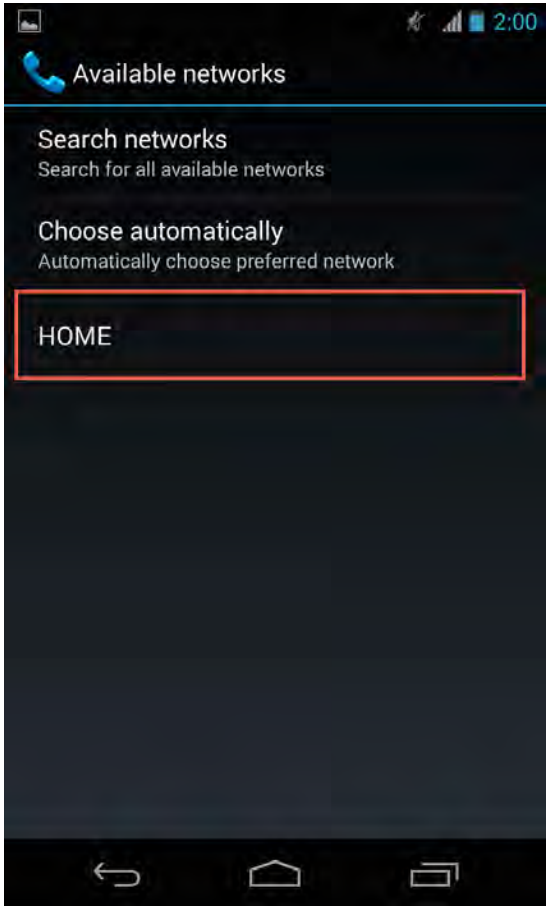
Connecting to the USRP on Android





Connecting to the USRP on Android





Time To Blow Up The Celly

Messaging From Within The RF Enclosure

Starting up OpenBTS

System Ready

```
1405019034.351185 3074184960:
system ready

1405019034.351746 3074184960:
use the OpenBTSCLI utility to access CLI
```

tmsis – Check Devices Connected

```
OpenBTS> tmsis
IMSI            TMSI  IMEI            AUTH CREATED ACCESSED TMSI_ASSIGNED
310410594683776 -     356489053684310 2    25h     25h      0
```

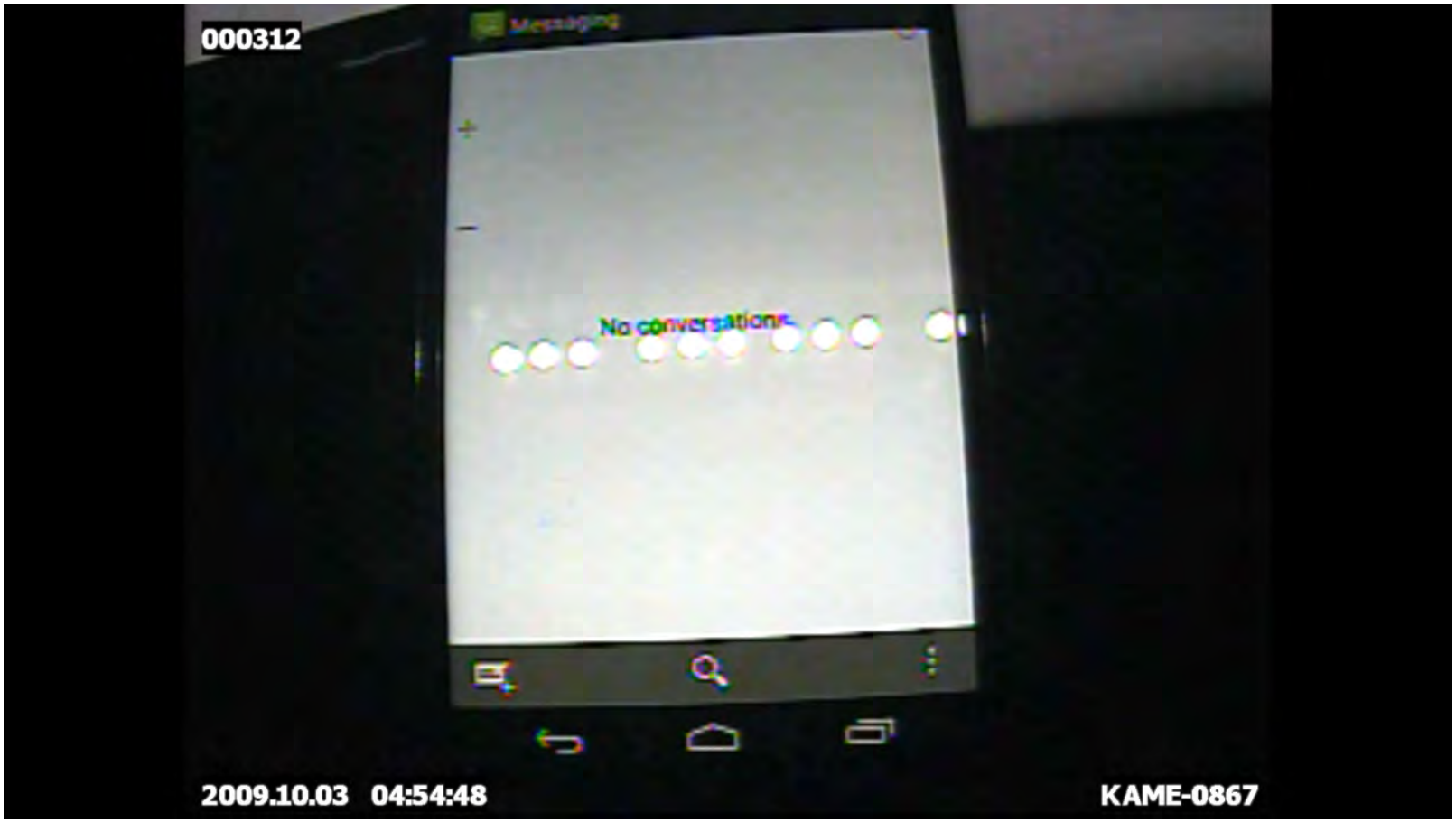
Sending Messages with OpenBTS

```
OpenBTS> sendsms 310410594683776 2813308004 MIKE JONES
message submitted for delivery

OpenBTS> sendsms 310410594683776 19006492568 Sir Mix A Lot
message submitted for delivery

OpenBTS> sendsms 310410594683776 8675309 Jenny
message submitted for delivery
```

Basic Text Messages



Bug Hunting

File Formats

Audio

"audio/aac”, "audio/amr”, "audio/imelody”, "audio/mid”, "audio/midi”, "audio/mp3”,"audio/mpeg3”, "audio/mpeg”, "audio/mpg”, "audio/mp4”, "audio/x-mid”, "audio/x-midi”, "audio/x-mp3”, "audio/x-mpeg3”, "audio/x-mpeg”, "audio/x-mpg”,"audio/3gpp”, "audio/x-wav”, "application/ogg"

Video

"video/3gpp”, "video/3gpp2”, "video/h263”, "video/mp4”

Pictures

"image/jpeg”, "image/jpg”, "image/gif”, "image/vnd.wap.wbmp” ,"image/png”,"image/x-ms-bmp”

Others

"text/x-vCalendar”, "text/x-vCard"

Easy File Format Candidates to find:

- https://github.com/klinker41/android-smsmms/blob/master/src/com/google/android/mms/ContentType.java
- Download AOSP (http://source.android.com)
- Source from Samsung (http://opensource.samsung.com/reception.do)
- rgrep for mime, image/, audio/, video/

Fuzzing Framework

Fuzzing Seeds

https://samples.libav.org/

http://samples.mplayerhq.hu/

Google out some file formats with filetype: operator

Mutation Libraries

Creating vcards and vcal

- http://vobject.skyhouseconsulting.com/
- https://pypi.python.org/pypi/vobject

Fuzzing pdu formats

- https://pypi.python.org/pypi/smspdu/

Fuzzing libraries

Hachoir

- https://bitbucket.org/haypo/hachoir/wiki/Home

Radamsa

- https://www.ee.oulu.fi/research/ouspg/Radamsa
- https://code.google.com/p/ouspg/wiki/Radamsa

Crash Triaging

Very easy to roll your own gdb wrapper and create a web app with database backend to distribute load

Live Demonstrations

Pray to the Demo Gods!

…but we have video backups

Key Takeaways

Blow Things Up!

Attractive targets

Filled with personal information and corporate secrets

Process information without user interaction

Handle large number of legacy formats

Decreasing barrier to entry

Leverage emulation provided by OS developers

Physical hardware becoming cheaper

Popularity of software defined radio increasing

Leverage previous lessons learned

Similar to fuzzing desktop apps to find bugs in MMS data handlers

Break through the mystique of cell phone research

Thank you