Coaching a Squad of
a.k.a. fuzzing on ARM
The speakers of SyScan
1000 cats
1e11 spiders
What do you want to do?
What do you want to do?

- RIP ARMS OFF
What do you want to do?

- RIP ARMS OFF
- HACK ARMS OFF
What do you want to do?

- RIP ARMS OFF
- HACK ARMS OFF
- meow meow meow meow meow meow meow meow meow meow meow meow meow meow meow meow meow meow meow meow meow
What do you want to do?

- RIP ARMS OFF
- HACK ARMS OFF
- meow meow meow meow meow meow meow meow meow meow meow meow meow meow meow meow meow meow meow meow meow meow meow meow meow meow meow meow
- weave wonderful webs
what will you do with vulns?

- I just like vulns
- We’ll exploit iPhone users working at XYZ
- We want to find vulns in ASDF
ARM is good for

- ARM operating systems
- Open source that builds on ARM
- ARM errata
- peripherals

Not so good for:

- processes that need huge amounts of RAM
- x86 stuff, Intel GPUs
Operating Systems

- Android
- Linux
- iOS (on Apple TV)
- Windows (probably hard)
- B2G (Servo)
- QNX and such
- Ubuntu Phone, Tizen, NetBSD, ...
What are people using now?
@jvanegue A few Dells with 64 cores and 256Gb of RAM each, expensive but effective. For some things, for others manual research is mandatory.
• 48 core VM hosts (4 x 12 core AMD Opteron)
• 20Gb Infiniband fabric
• $280 USD per core / $140 per VM
• Pulls ~10kW
• Total Hardware Cost ~ 30k USD
1U

- 2x95W cpus, 12 cores
- 32GB ram
- $5000
Gamer desktop

- 1x Intel Core i7
- 32 GB RAM
- $1000
ODROID-U2

- Exynos 4412, Cortex-A9, quad core @ 1.7ghz
- Mali-400 quad core GPU @ 440Mhz
- 2GB RAM
- 100mbit ethernet, 2x USB, USB host (fastboot & android), serial port (1.8v), power, mini-HDMI, sd-card, eMMC
- ~$100
ODROID-U2

- powered through DC jack (5V 2A)
- fwbl1 signed by Samsung
- bl2 signed by Hardkernel (they’ll sign one for you)
- mali gpu driver as binary blob
- supported by Hardkernel (drivers, kernel)
ODROID-U2

- iROM loads fwbl1 and bl2 from boot media
- no ethernet in u-boot (possible but hard)
- can upload kernel & initrd via serial cable
- fastboot accessible from u-boot
- flash partitions from usb host cable
MK802 II

- Allwinner A10, Cortex-A8, single core @ 1.5ghz
- Mali 400 GPU
- 1GB RAM
- HDMI, USB host, USB, power, 802.11
- 4GB onboard flash, sd-card slot
- ~$35
- Liberated by Allwinner Dev Team
considerations

- 1 box with 100 vms vs 100 boxen
- drivers
- kernels
- boot media
- cpu/ram/... distribution is fixed
drivers

- for linux, you need drivers for GPU, NIC, ...
- chances are good with big brand names
- chances not so good with unknown stuff (Allwinner A10 a notable exception)
- check forums about your exact board!
- or run old version of android
optimize throughput

- you want to get useful stuff done
- usage is irrelevant
  - don’t try to use every cpu cycle
  - don’t try to use every byte of ram
- this applies to life in general
fuzz targets

- not stuff you glued together
- actual software
  - browsers
  - open source libraries
  - peripherals
- mali gpu drivers
- kernels
Let’s make a browser!
Mishod Aubizov @miaubiz

@Glider is it possible to build chromium with asan for Ubuntu on arm?

Details  Reply  Delete  Favorite  More

Alexander Potapenko @Glider

@miaubiz I guess you'll be the pioneer. Dunno if security folks are interested in that though. LLVM ARM backend may also have subtle bugs.

Details
If one compiler could all platforms treat
Would men and angels then fall at its feet
And shun the hoary beast named GCC
Enthroning thus the king of binary

- Ben Nagy
cross compilation

- kernel builds in 18 minutes natively
- linking chromium takes 8 gb ram
- gcc is dead, use llvm
- build all tools from source: binutils, cmake, llvm
cross compilation

- use llvm revision known to work (e.g. from chromium)
- cmake -DLLVM_TARGETS_TO_BUILD="ARM;X86" -DLLVM_DEFAULT_TARGET_TRIPLE=arm-linux-gnueabihf /build/llvm
- make -j5 clang_rt.asan-arm
- sudo apt-get install binutils-arm-linux-gnueabihf
AddressSanitizer

# Architectures supported by ASan.
- x86_64 i386 powerpc64 powerpc
+ arm x86_64 i386 powerpc64 powerpc

(On Android asan is a shared library.)
+Configs += asan-arm
+Arch.asan-arm := arm
+CFLAGS.asan-arm := $(CFLAGS) -fPIC \ 
    -fno-built-in -mllvm -arm-enable-ehabi
+FUNCTIONS.asan-arm := $(AsanFunctions)\ $(InterceptionFunctions) \ $(SanitizerCommonFunctions)
+elseif("${LLVM_NATIVE_ARCH}" STREQUAL "ARM")
-  x86_64 i386 powerpc64 powerpc)
+  arm x86_64 i386 powerpc64 powerpc)
+  test_target_arch(arm "")
then I

GYP_CROSSCOMPILE=1 GYP_DEFINES='target_arch=arm linux_use_tcmalloc=0 armv7=1 arm_thumb=1 sysroot=/build/linaro arm_float_abi=hard enable_webRTC=0 disable_nacl=1 release_extra_cflags="-Wno-asn-operands-widths -Wno-return-type-c-linkage -Wno-parentheses-equality -Wno-enum-conversion -g" remoting=0 clang_use_chrome_plugins=0 enable_google_now=0 enable_language_detection=0 enable_automation=0 linux_breakpad=0 linux_use_gold_binary=1 linux_use_gold_flags=1 asan=1' GYP_GENERATORS=ninja gclient runhooks
- GYP_CROSSCOMPILE=1
- target_arch=arm armv7=1 arm_thumb=1 arm_float_abi=hard
- sysroot=/build/linaro
- linux_use_gold_binary=1
  linux_use_gold_flags=1
- ASAN_OPTIONS="malloc_context_size=0"
ARM errata
In certain rare sequences of code, the loop buffer may deliver incorrect instructions. (this erratum is for Cortex A15 only)
Exynos5: Fix errata 773022 and 774769 on Exynos5250

The exynos5250 chip is affected by the errata

773022 (disable loop buffer)
774769 (refrain streaming-write to ever allocate into the L2 cache)

This fixes random segmentation faults on Arndale for me, that mostly occurred because of the broken loop buffer.

Signed-off-by: Alexander Graf <agraf@suse.de>

“This fixes random segmentation faults on Arndale for me, that mostly occurred because of the broken loop buffer.”
What I got (~$3000)

- 20 x ODROID-U2
- 80 cores @ 1.7ghz
- 40 GB ram
- 20 operating systems
- 20 ethernet ports
- 20 serial ports
- 40 usb ports
- 17 sd cards, 3 eMMC
- 24-port ethernet switch
chef, puppet

dthis actually works:

```
rsync -ax --exclude hosts
        --exclude hostname --exclude
        ssh --exclude smsc95xx_mac_addr
root@li0.local:/ / 
```
parallel ssh linaro@{}.local
"hostname" :::: li0 li1 li2
li3 li4 li5 li6 li7 li8 li9
lia lib lic lid lie lif li10
and then:

- `/etc/init/fuzz.conf`:
  ```
  exec screen -D -m -c /home/fuzz/fuzz.screen
  ```

- `fuzz.screen`:
  ```
  screen -t fuzz0
  stuff "while true; do
  ~/fuzz/fuzz.sh 0 fuzz.html;
  sleep 1; done\012"
  ```
reuse everything from x86

- rsync
- bash
- node.js
- redis
- filesystem
I have 20 wall warts at 220V

that’s stupid. you should just get an ATX power supply and power them all from the 5 volt. then you could also power cycle them with a microcontroller.
I only have one serial cable

you know, you could just connect all those serial ports to an fpga and multiplex the signals.
Die Datenkrake

- arm m3 + actel fpga
- ~$100
- 48 gpio (can do 16x power + uart)
- custom pcb for going down to 1.8v for odroid uart
- custom pcb to distribute and control DC from ATX power supply
Die Datenkrake

- detect system crashes
- power cycle devices
- load kernel/initrd over serial
- diagnose corrupt bootloader
  (maybe emulate sdcard later :D)
could the fpga handle

- sd card
- usb devices
  (yes, especially if it wasn’t the cheapest fpga model)
- ethernet
- hdmi
so where is it?

- dc and serial connectors are in the mail
- custom pcb being designed as we speak
- I almost know VHDL
- dmitry is very good at this
Attribution

- gorilla: http://www.flickr.com/photos/mikejsolutions/74815604/lightbox/
- spiders: http://www.flickr.com/photos/photophilde/2518101974/sizes/l/in/photostream/
- cats: By Scott Granneman from St. Louis, MO, USA (Flickr)
- construction workers: http://commons.wikimedia.org/wiki/File:Construction_Workers_in_Maracaibo.JPG
- odroid-u2: Dmitry Nedospasov
- datenkrake: Thorsten Schroeder
ALL WINNERS SQUAD

CAPTAIN AMERICA

SUB-MARINER

HUMAN TORCH

Introducing John Bromfield

Dick Purcell

Van Johnson