RSAConference2015

San Francisco | April 20-24 | Moscone Center

CHANGE

Challenge today's security thinking

SESSION ID: HT-W04

Don't Touch That Dial: How Smart Thermostats Have Made Us Vulnerable

Ray Potter

CEO
SafeLogic
@SafeLogic_Ray

Yier Jin

Assistant Professor
University of Central Florida
@jinyier

#RSAC

Flow

- The threat is real
- Connected convenience comes with risk
- Challenges
- What’s at Stake

What’s at Stake

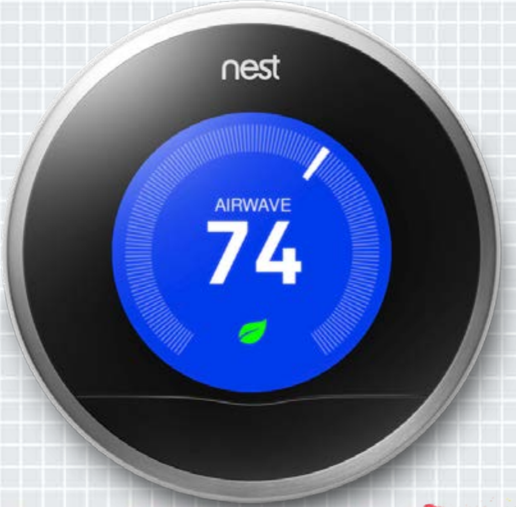
- Pattern recognition
- Identity theft
- Corporate espionage
- Life

Use Cases

Nest Thermostat

- Nest Labs founded by Tony Fadell
- Debuted in October 2011
- Acquired by Google in January 2014 ($3.2B)
- Over 40,000 sold each month

  Data from GigaOM as of January 2013
- Available in UK in April 2014
- Smart home API is released in June 2014



“Yes, hacking is in our thoughts. When you're talking about the home, these are very private things. We thought about what people could do if they got access to your data. We have bank-level security, we encrypt updates, and we have an internal hacker team testing the security. It's very, very private and it has to be, because it'll never take off if people don't trust it.”

- Tony Fadell

RSAConference2015

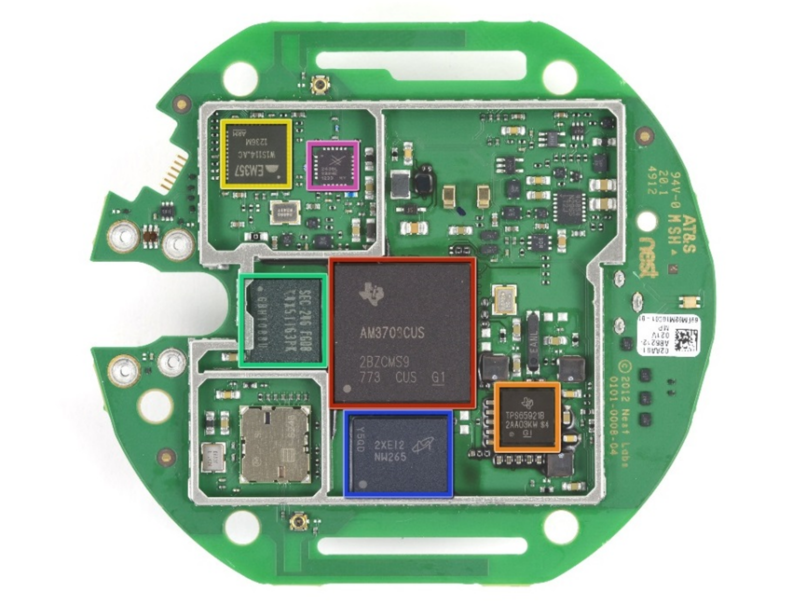
San Francisco | April 20-24 | Moscone Center

Nest Hardware

#RSAC

Front Plate

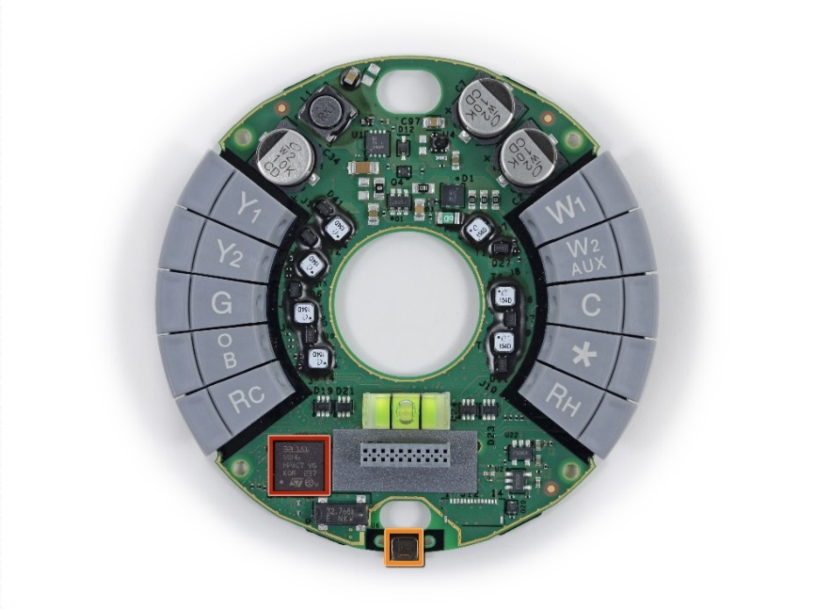
- “Display” board
- Graphics/UI, Networking
- Chips:
  - ARM Cortex A8 app processor
  - USB OTG
  - RAM/Flash (2Gb)
  - ZigBee/WiFi Radios
  - Proximity Sensors
- UART test points (silenced at bootloader)



Courtesy of iFixit

“Backplate” and Comms

- Hooks up to AC/Heating system. Charges battery via engineering wizardry
- Chips:
  - Independent ARM Cortex M3
  - Temp and Humidity Sensor
- Communications
  - Front to Back – UART
  - NEST Weave (802.15.4)
  - USB MSD (FW update)

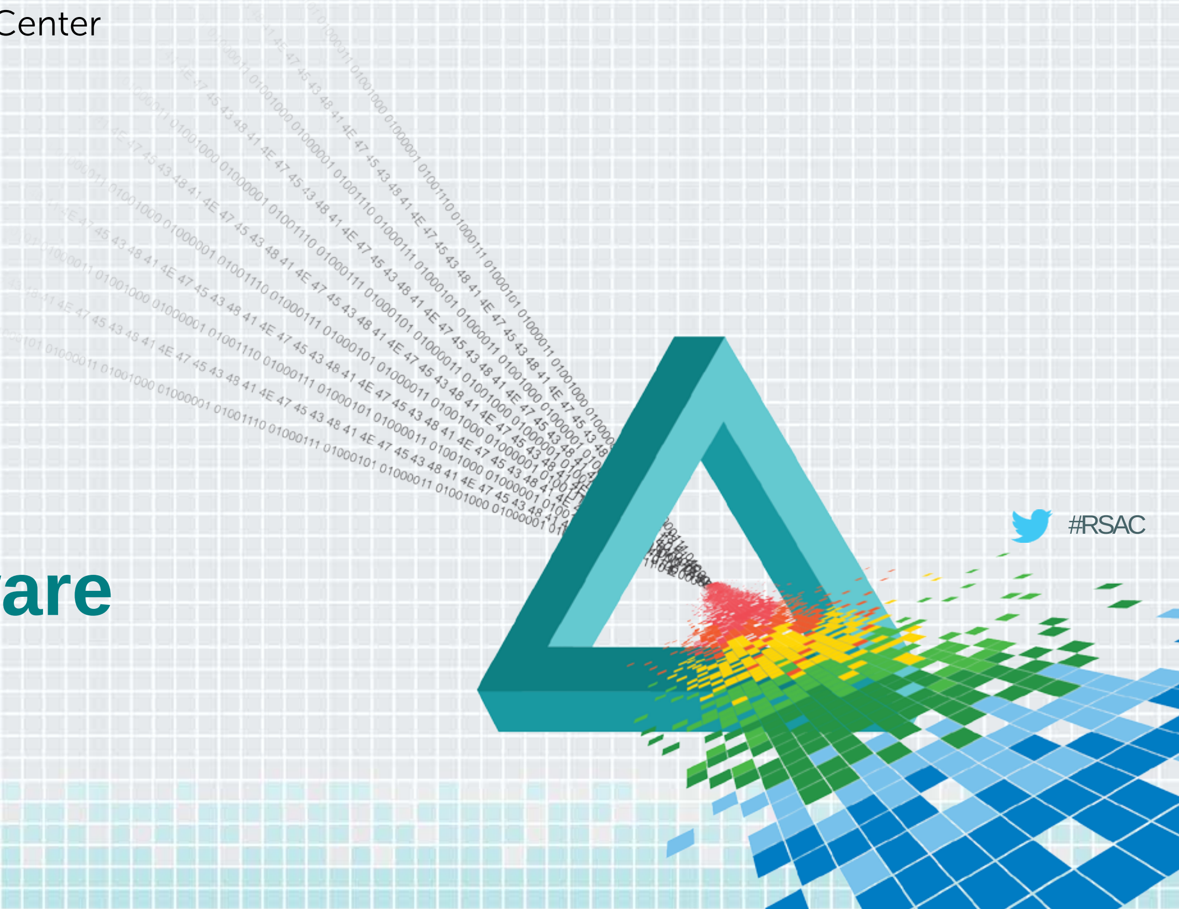


Courtesy of iFixit

RSAConference2015

San Francisco | April 20-24 | Moscone Center

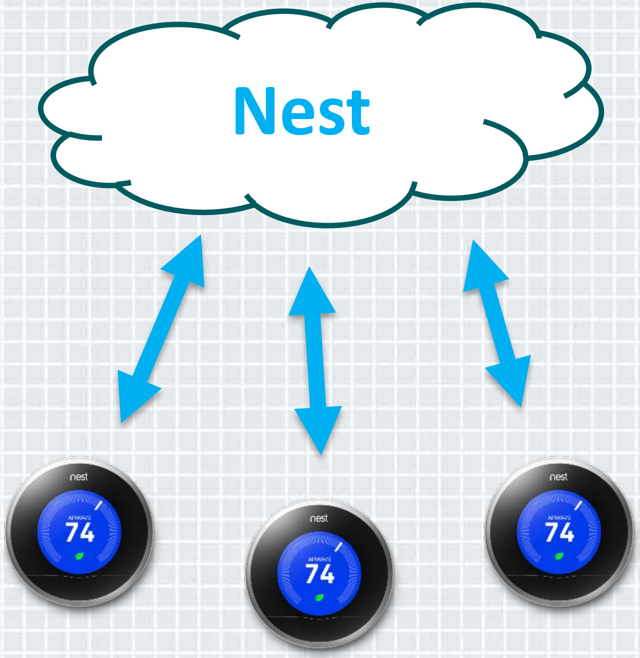
Nest Software



#RSAC

Nest Client

- Runs on a Linux based platform
- Handles interfacing between device and Nest Cloud services
- Automatically handles firmware updates
- Manual update available
  - Plug Nest into PC
  - Handled as a storage device
  - Copy firmware to drive
  - Reboot



Nest Firmware

- Signed firmware ☹
  - Manifest.plist
    - Hashes contents
  - Manifest.p7s
- Compressed but not encrypted or obfuscated
- Includes
  - U-boot image
  - Linux Kernel image
  - File system
  - nlbpfirmware.plist

Things Done the Right Way™

- Firmware signing using PKCS7
- Pinned Nest certificates for firmware verification
- All critical communications (any with secrets) over HTTPS
    - Other less secure ones over HTTP (firmware, weather)

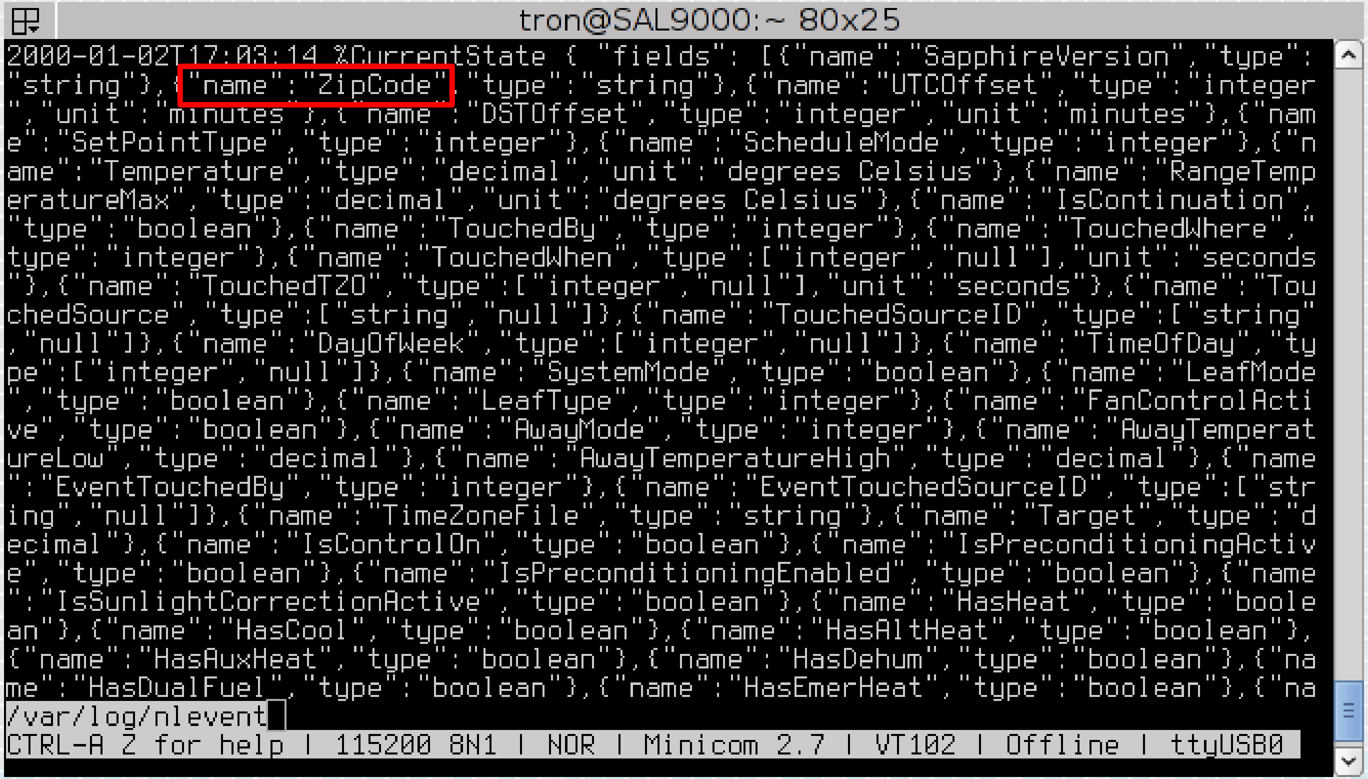
Things Done the Wrong Way™

- Firmware links downloaded using HTTP and download links do not expire
- Hardware backdoor left for anyone with a USB port to use
- Automatic updates

User Privacy

- Log Files
  - Internally stored and uploaded to Nest
  - Contents
- User Interface
  - Users are unaware of the contents of the log files
  - Users cannot turn off this option
- User network credentials are stored … in plain text!
- Users should be allowed to opt-out of the data collection?

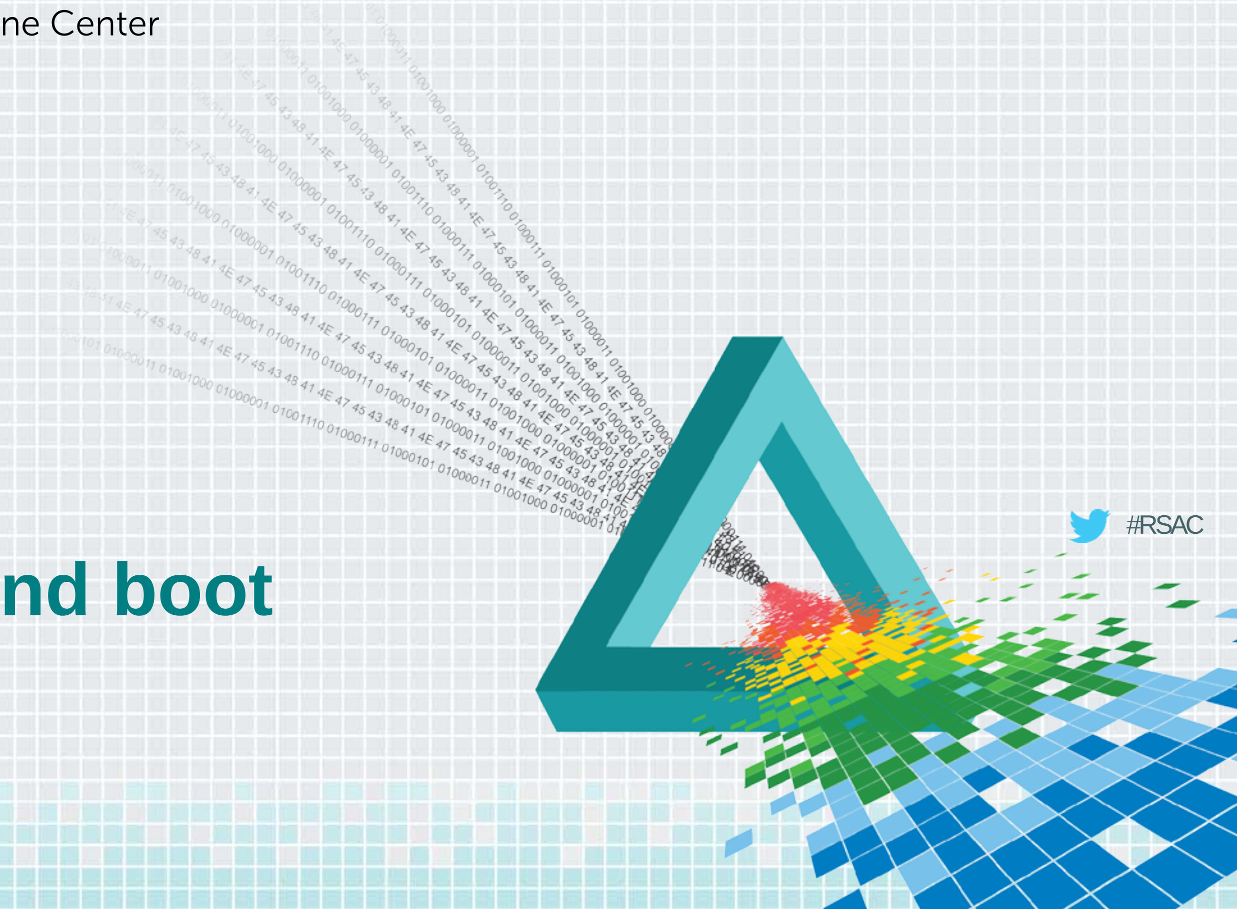
Log Files



```
2000-01-02T17:03:14 %CurrentState { "fields": [{"name":"SapphireVersion","type":
"string"},{"name":"ZipCode","type":"string"},{"name":"UTCOffset","type":"integer
","unit":"minutes"},{"name":"DSTOffset","type":"integer","unit":"minutes"},{"nam
e":"SetPointType","type":"integer"},{"name":"ScheduleMode","type":"integer"},{"n
ame":"Temperature","type":"decimal","unit":"degrees Celsius"},{"name":"RangeTemp
eratureMax","type":"decimal","unit":"degrees Celsius"},{"name":"IsContinuation",
"type":"boolean"},{"name":"TouchedBy","type":"integer"},{"name":"TouchedWhere","
type":"integer"},{"name":"TouchedWhen","type":["integer","null"],"unit":"seconds
"},{"name":"TouchedTZO","type":["integer","null"],"unit":"seconds"},{"name":"Tou
chedSource","type":["string","null"]},{"name":"TouchedSourceID","type":["string"
,"null"]},{"name":"DayOfWeek","type":["integer","null"]},{"name":"TimeOfDay","ty
pe":["integer","null"]},{"name":"SystemMode","type":"boolean"},{"name":"LeafMode
","type":"boolean"},{"name":"LeafType","type":"integer"},{"name":"FanControlActi
ve","type":"boolean"},{"name":"AwayMode","type":"integer"},{"name":"AwayTemperat
ureLow","type":"decimal"},{"name":"AwayTemperatureHigh","type":"decimal"},{"name
":"EventTouchedBy","type":"integer"},{"name":"EventTouchedSourceID","type":["str
ing","null"]},{"name":"TimeZoneFile","type":"string"},{"name":"Target","type":"d
ecimal"},{"name":"IsControlOn","type":"boolean"},{"name":"IsPreconditioningActiv
e","type":"boolean"},{"name":"IsPreconditioningEnabled","type":"boolean"},{"name
":"IsSunlightCorrectionActive","type":"boolean"},{"name":"HasHeat","type":"boole
an"},{"name":"HasCool","type":"boolean"},{"name":"HasAltHeat","type":"boolean"},
{"name":"HasAuxHeat","type":"boolean"},{"name":"HasDehum","type":"boolean"},{"na
me":"HasDualFuel","type":"boolean"},{"name":"HasEmerHeat","type":"boolean"},{"na
/var/log/nlevent
CTRL-A Z for help | 115200 8N1 | NOR | Minicom 2.7 | VT102 | Offline | ttyUSB0
```

RSAConference2015

San Francisco | April 20-24 | Moscone Center

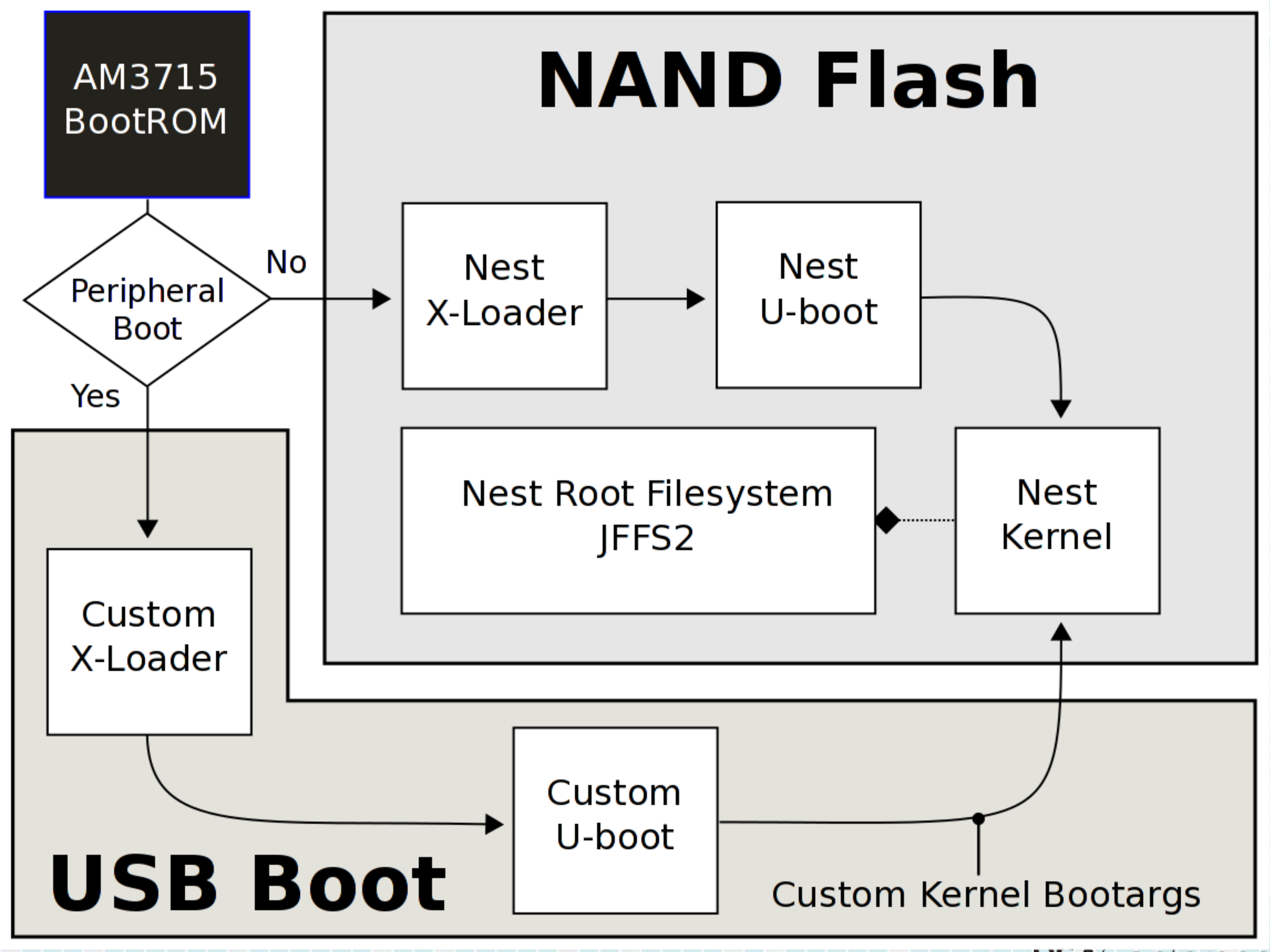


#RSAC

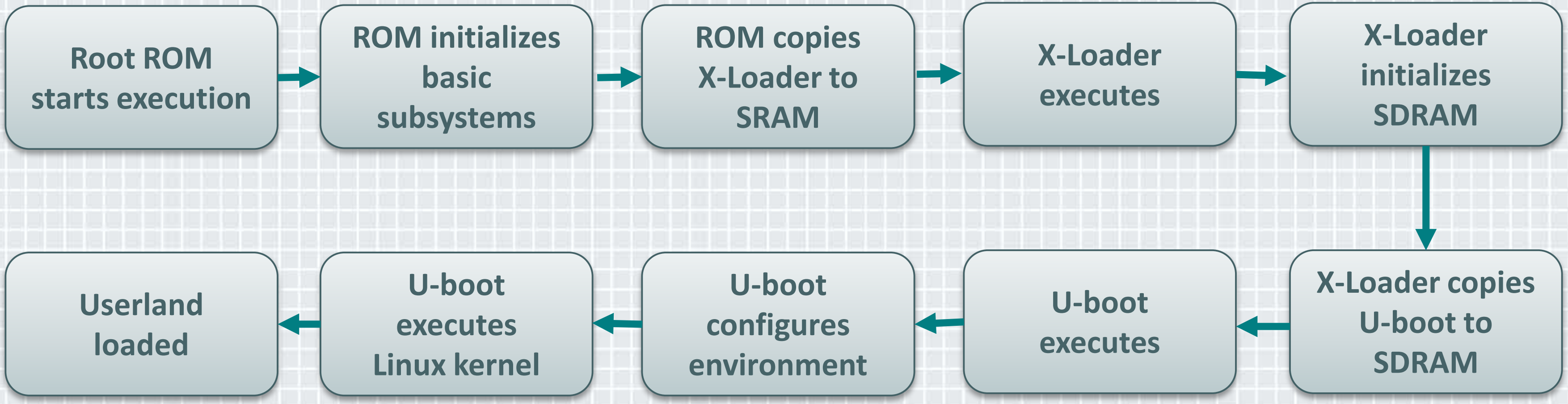
Processor and boot

Hardware Analysis

- TI Sitara AM3703
  - ARM Cortex-A8 core
    - Version 7 ISA
    - JazelleX Java accelerator and media extensions
    - ARM NEON core SIMD coprocessor
  - DMA controller
  - HS USB controller
  - General Purpose Memory Controller to handle flash
  - SDRAM memory scheduler and controller
  - 112KB on-chip ROM (boot code)
  - 64KB on-chip SRAM
  - Configurable boot options

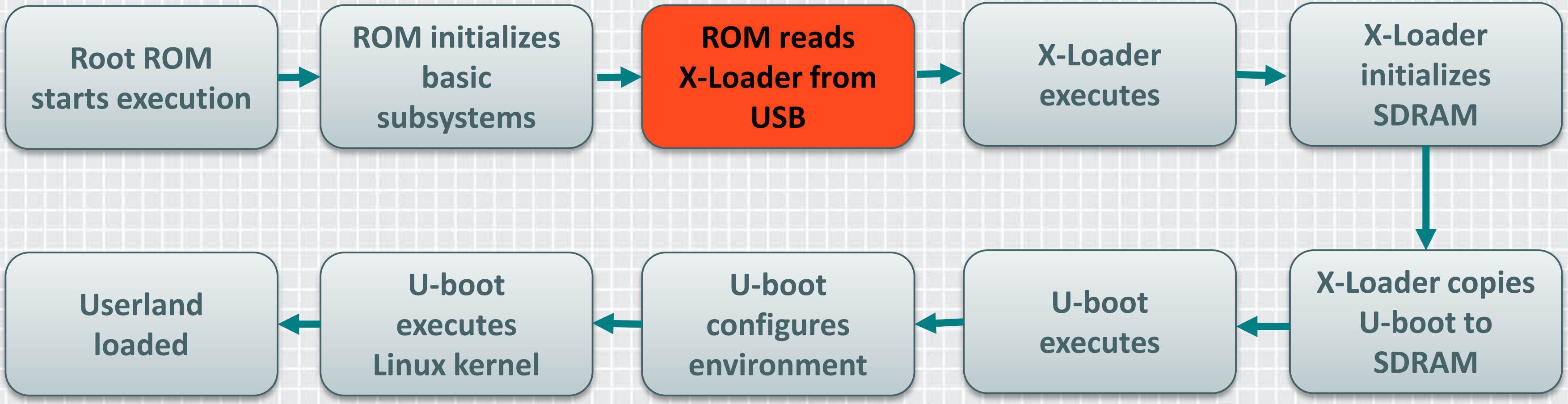

AM3715
BootROM
Peripheral Boot
No
Yes
NAND Flash
Nest X-Loader
Nest U-boot
Nest Root Filesystem JFFS2
Nest Kernel
Custom X-Loader
Custom U-boot
USB Boot
Custom Kernel Bootargs

Boot Process



Root ROM starts execution → ROM initializes basic subsystems → ROM copies X-Loader to SRAM → X-Loader executes → X-Loader initializes SDRAM → X-Loader copies U-boot to SDRAM → U-boot executes → U-boot configures environment → U-boot executes Linux kernel → Userland loaded

Boot Process



Root ROM starts execution

ROM initializes basic subsystems

ROM reads X-Loader from USB

X-Loader executes

X-Loader initializes SDRAM

X-Loader copies U-boot to SDRAM

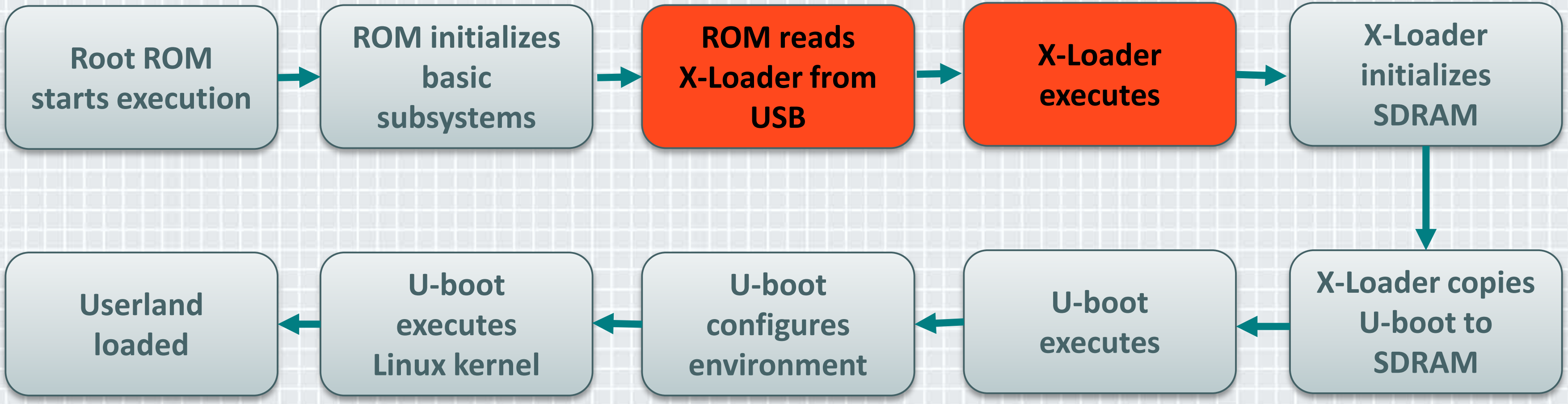
U-boot executes

U-boot configures environment

U-boot executes Linux kernel

Userland loaded

Boot Process



Root ROM starts execution

ROM initializes basic subsystems

ROM reads X-Loader from USB

X-Loader executes

X-Loader initializes SDRAM

X-Loader copies U-boot to SDRAM

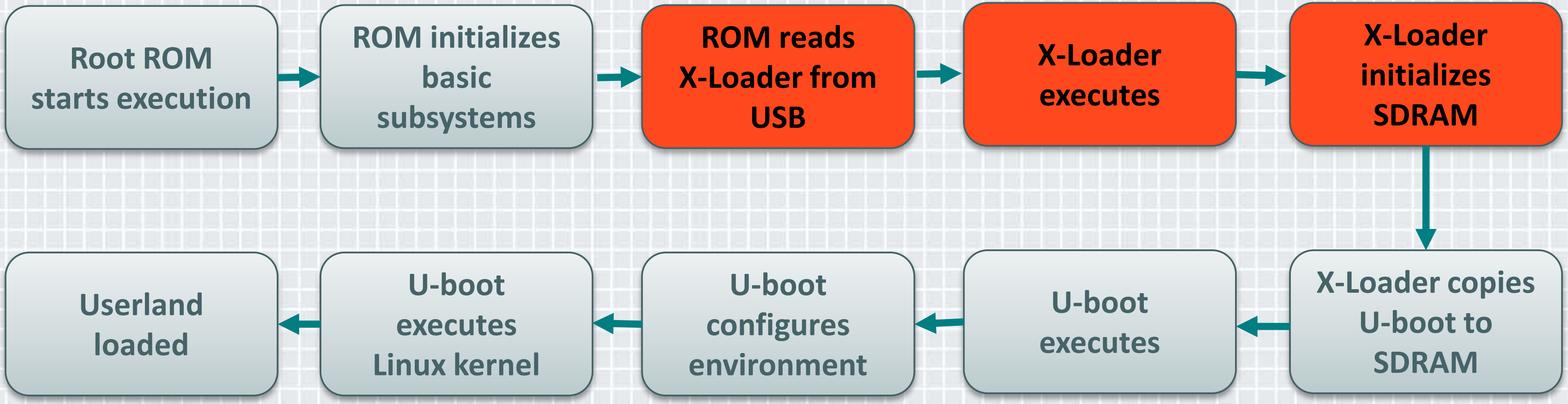
U-boot executes

U-boot configures environment

U-boot executes Linux kernel

Userland loaded

Boot Process



Root ROM starts execution

ROM initializes basic subsystems

ROM reads X-Loader from USB

X-Loader executes

X-Loader initializes SDRAM

X-Loader copies U-boot to SDRAM

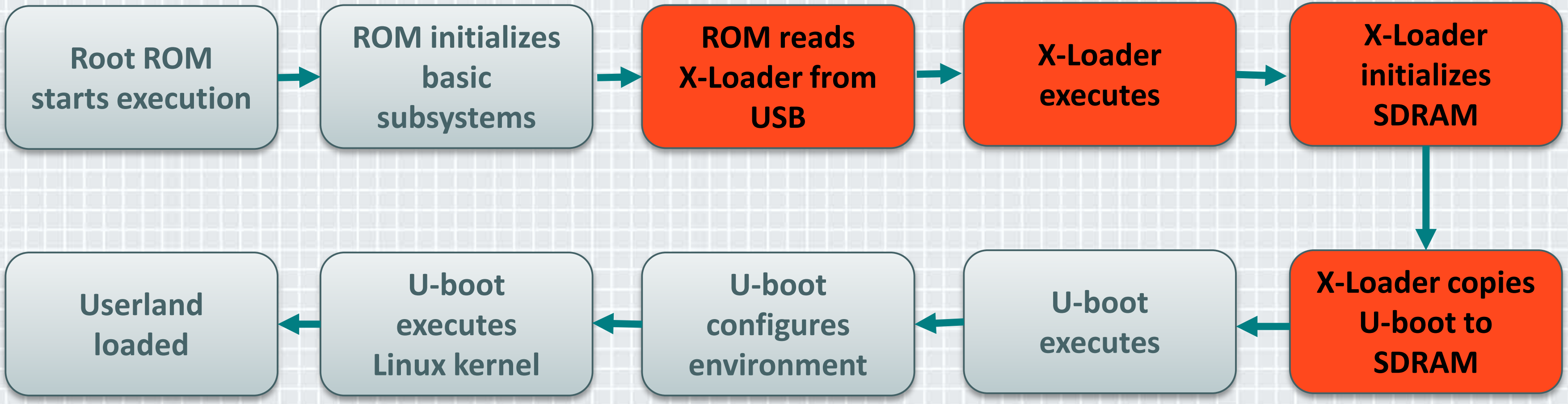
U-boot executes

U-boot configures environment

U-boot executes Linux kernel

Userland loaded

Boot Process



Root ROM starts execution

ROM initializes basic subsystems

ROM reads X-Loader from USB

X-Loader executes

X-Loader initializes SDRAM

X-Loader copies U-boot to SDRAM

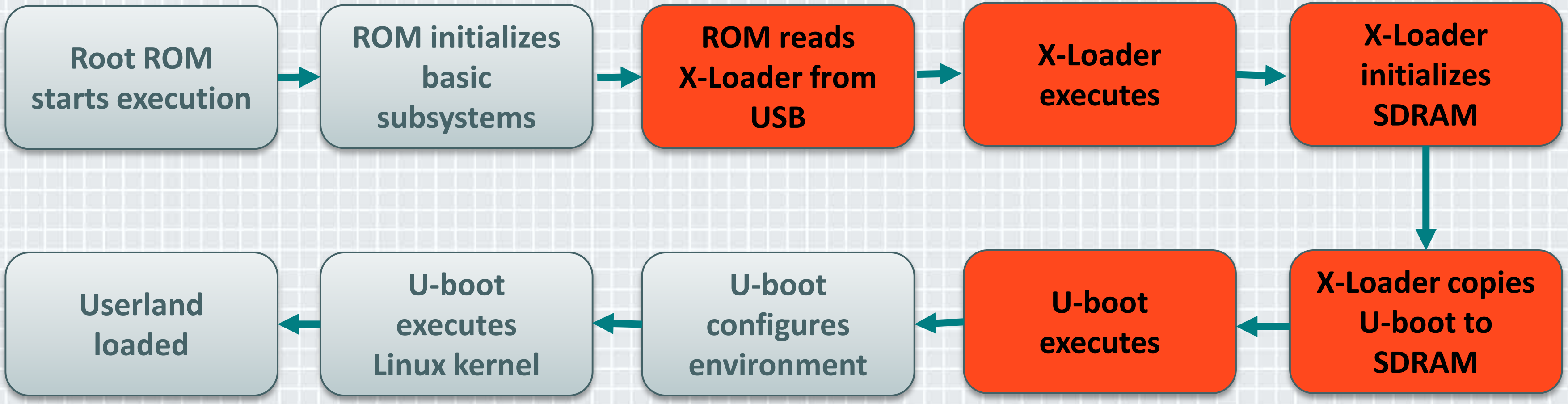
U-boot executes

U-boot configures environment

U-boot executes Linux kernel

Userland loaded

Boot Process



Root ROM starts execution

ROM initializes basic subsystems

ROM reads X-Loader from USB

X-Loader executes

X-Loader initializes SDRAM

X-Loader copies U-boot to SDRAM

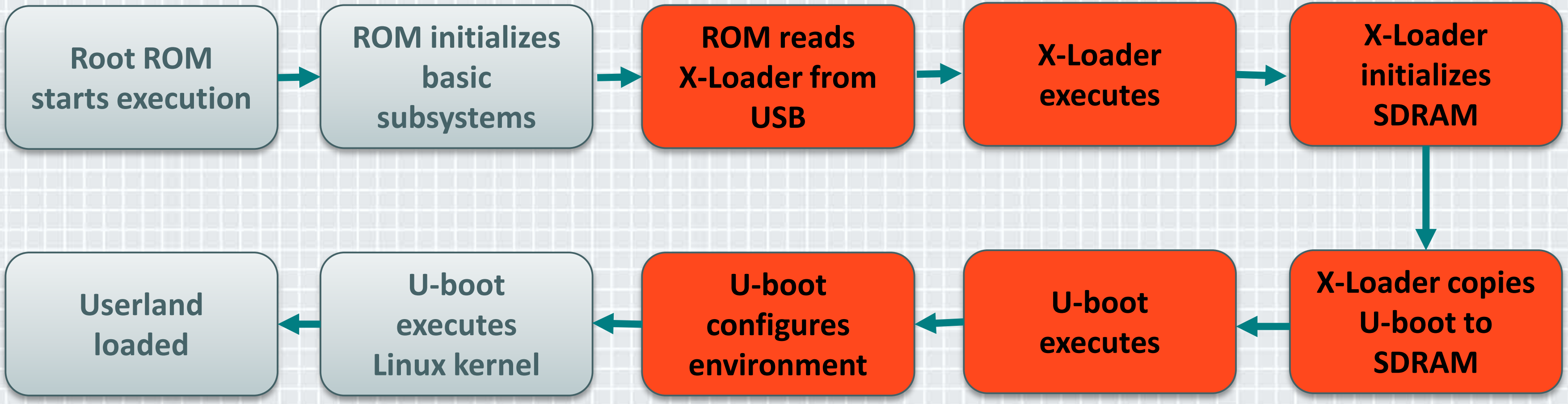
U-boot executes

U-boot configures environment

U-boot executes Linux kernel

Userland loaded

Boot Process



Root ROM starts execution

ROM initializes basic subsystems

ROM reads X-Loader from USB

X-Loader executes

X-Loader initializes SDRAM

X-Loader copies U-boot to SDRAM

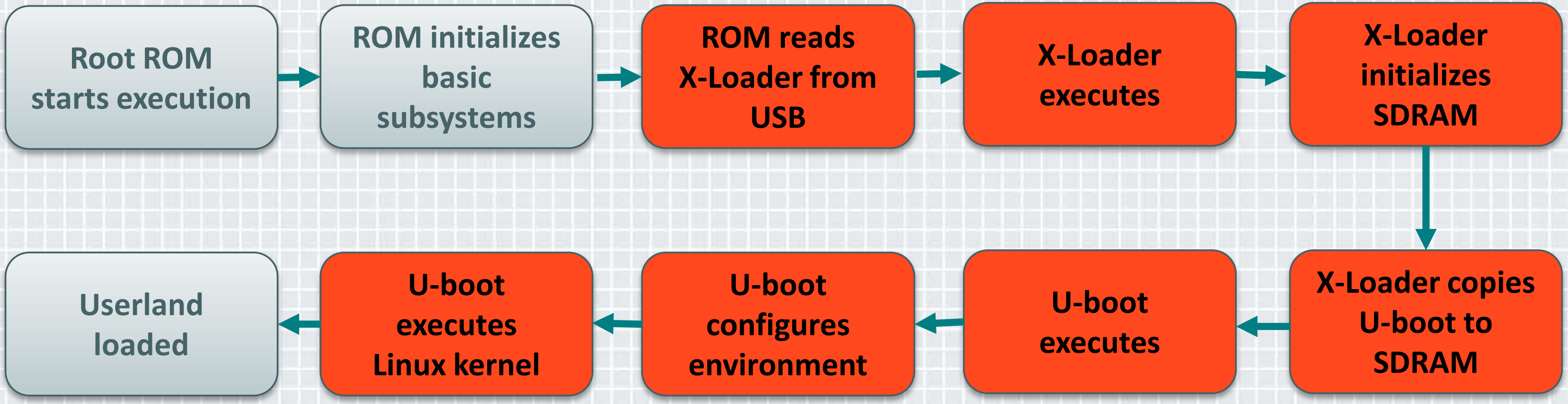
U-boot executes

U-boot configures environment

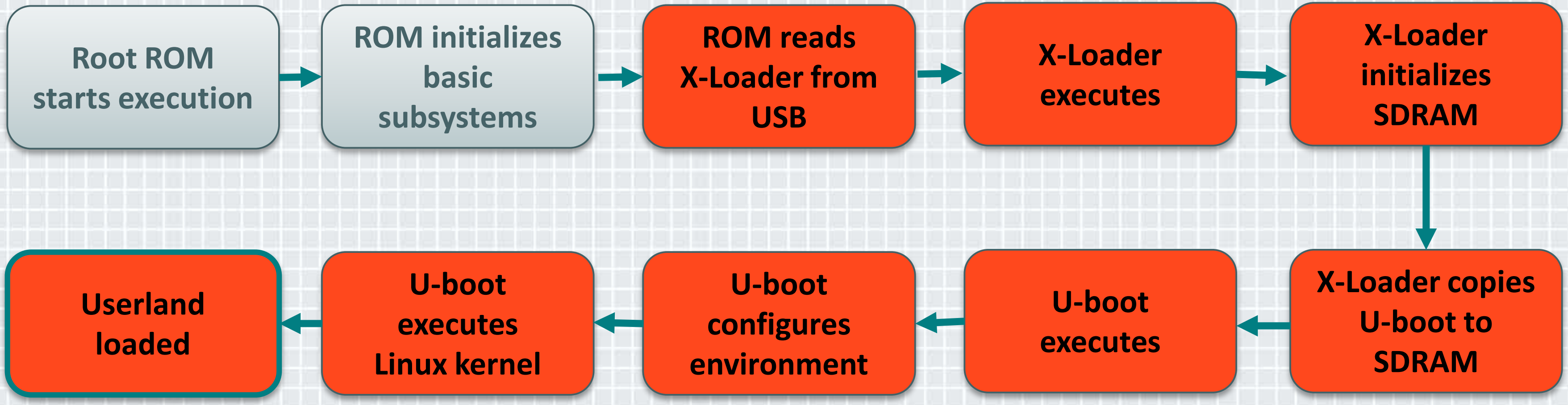
U-boot executes Linux kernel

Userland loaded

Boot Process


Root ROM starts execution
ROM initializes basic subsystems
ROM reads X-Loader from USB
X-Loader executes
X-Loader initializes SDRAM
X-Loader copies U-boot to SDRAM
U-boot executes
U-boot configures environment
U-boot executes Linux kernel
Userland loaded

Boot Process



Root ROM starts execution

ROM initializes basic subsystems

ROM reads X-Loader from USB

X-Loader executes

X-Loader initializes SDRAM

X-Loader copies U-boot to SDRAM

U-boot executes

U-boot configures environment

U-boot executes Linux kernel

Userland loaded

Device Initialization

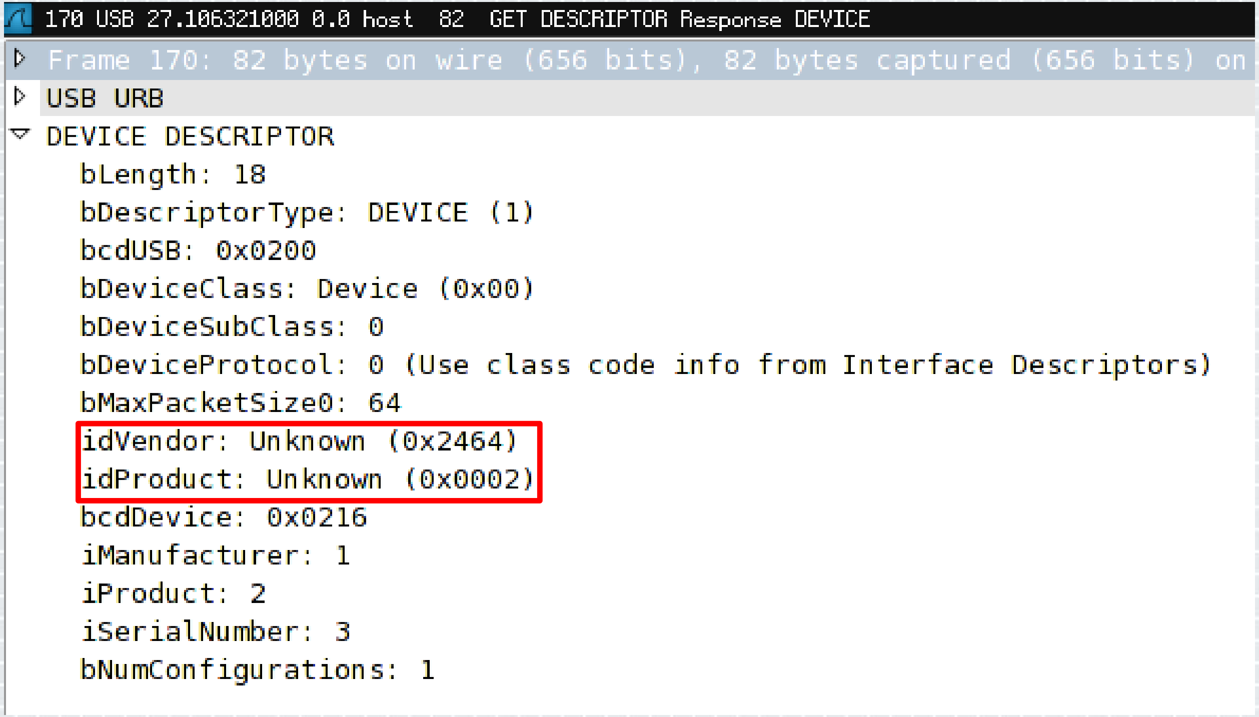
- Boot Configuration read from sys_boot[5:0]

Selected boot configurations					
sys_boot [5:0]	First	Second	Third	Fourth	Fifth
001101	XIP	USB	UART3	MMC1	MMC1
001110	XIPwait	DOC	USB	UART3	
001111	NAND	USB	UART3	MMC1	
101101	USB	UART3	MMC1	XIP	DOC
101110	USB	UART3	MMC1	XIPwait	
101111	USB	UART3	MMC1	NAND	

Device Programming

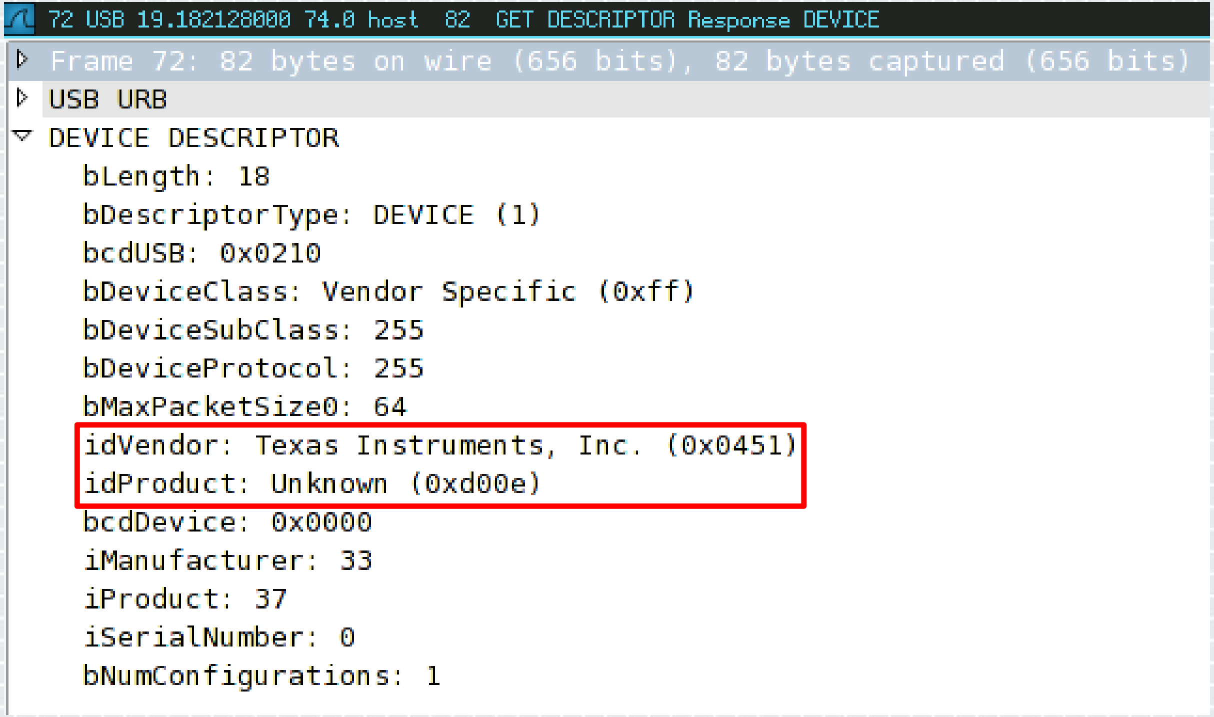
- Boot configuration pins 4..0 are fixed in Nest’s hardware
- sys_boot[5] is changes based on reset type
- Conveniently, circuit board exposes sys_boot[5] on an unpopulated header…

Nest USB Device Descriptor



```
170 USB 27.106321000 0.0 host  82  GET DESCRIPTOR Response DEVICE
▷ Frame 170: 82 bytes on wire (656 bits), 82 bytes captured (656 bits) on
▷ USB URB
▽ DEVICE DESCRIPTOR
    bLength: 18
    bDescriptorType: DEVICE (1)
    bcdUSB: 0x0200
    bDeviceClass: Device (0x00)
    bDeviceSubClass: 0
    bDeviceProtocol: 0 (Use class code info from Interface Descriptors)
    bMaxPacketSize0: 64
    idVendor: Unknown (0x2464)
    idProduct: Unknown (0x0002)
    bcdDevice: 0x0216
    iManufacturer: 1
    iProduct: 2
    iSerialNumber: 3
    bNumConfigurations: 1
```

TI USB Device Descriptor



```
72 USB 19.182128000 74.0 host   82  GET DESCRIPTOR Response DEVICE
▷ Frame 72: 82 bytes on wire (656 bits), 82 bytes captured (656 bits)
▷ USB URB
▽ DEVICE DESCRIPTOR
    bLength: 18
    bDescriptorType: DEVICE (1)
    bcdUSB: 0x0210
    bDeviceClass: Vendor Specific (0xff)
    bDeviceSubClass: 255
    bDeviceProtocol: 255
    bMaxPacketSize0: 64
    idVendor: Texas Instruments, Inc. (0x0451)
    idProduct: Unknown (0xd00e)
    bcdDevice: 0x0000
    iManufacturer: 33
    iProduct: 37
    iSerialNumber: 0
    bNumConfigurations: 1
```

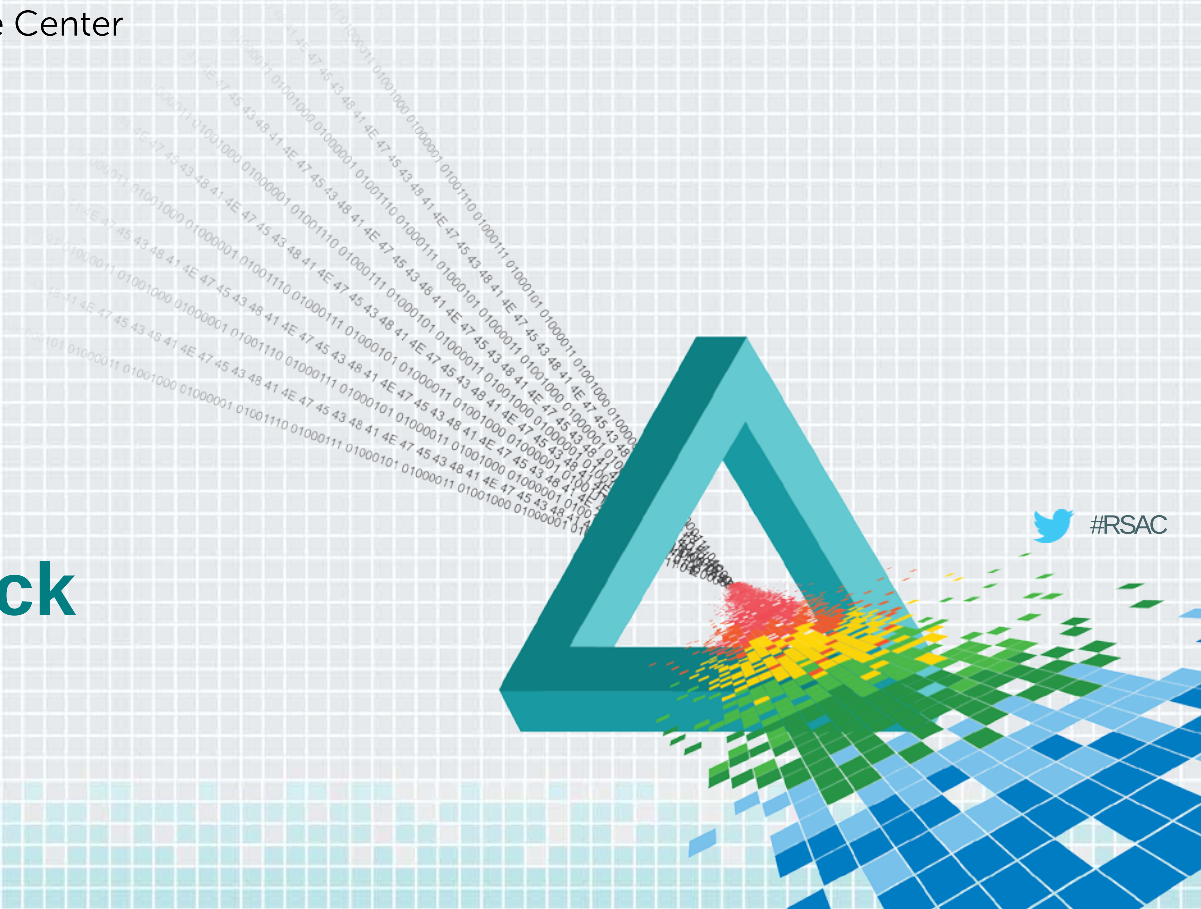
Implications

- Full control over the house
  - Away detection
  - Network credentials
  - Zip Code
  - Remote exfiltration
  - Pivoting to other devices

Control over all Nest devices

- Unauthorized ability to access Nest account
  - We now have the OAUTH secrets
- Ability to brick the device
  - We can modify the NAND
- Persistent malware in NAND
  - X-loader bootkit in NAND

The Attack



Attack

- Device Reset
  - Press the button for 10 seconds causing sys_boot[5] = 1’b1
- Inject code through the USB into memory and execute
  - Be quick!

Initial Attack

- Custom X-Loader to chainload U-Boot + initrd
- Custom U-Boot
  - Utilize existing kernel
  - Load our ramdisk (initrd)
- Ramdisk
  - Mount Nest’s filesystem and write at will
  - Arbitrary, scriptable, code execution
- Netcat already comes with the Nest

Refining a Backdoor

- Rebuild toolchain
- Cross-compile dropbear (SSH server)
- Add user accounts and groups
- Reset root password

Linux Kernel Modification

- A custom Linux kernel
- Custom logo
- Debugging capabilities (kgdb)
- Polling on OMAP serial ports

Demo

nest
Hello
Dave

Double-Edged Sword

- Positive View
  - The backdoor provide legitimate users to opt-out of uploading logs files
- Negative View
  - The backdoor may be maliciously exploited
- A Relief to Nest Labs
  - The backdoor needs physical access to the device (although remote attack is under investigation)

User Privacy Protection

- Privacy Patch Development
  - A patch is developed to protect user privacy
  - Users can select the data to be sent to Nest Cloud
  - Firmware upgrade will not cover the patch
- Patch Installation
  - Patch is installed through the hardware backdoor
  - One-button installation
  - Linux version is read for downloading

A Solution – Chain of Trust

- Code Authentication
  - Processor must authenticate the first stage bootloader before it is run
- Use public key cryptography
  - Userland protection
    - Only execute signed binaries
    - Filesystem encryption
  - Processor-DRAM channel protection

How to Apply This Knowledge

- Identify whether your product shares vulnerabilities with these examples.
- Build security strategy and implement NOW, don’t wait.
- Explore 3rd party validation and other ways to leverage proven security measures.
- Regardless of form factor, focus on the data.
- And of course, as a user, quarantine WiFi access for each of your IoT devices.