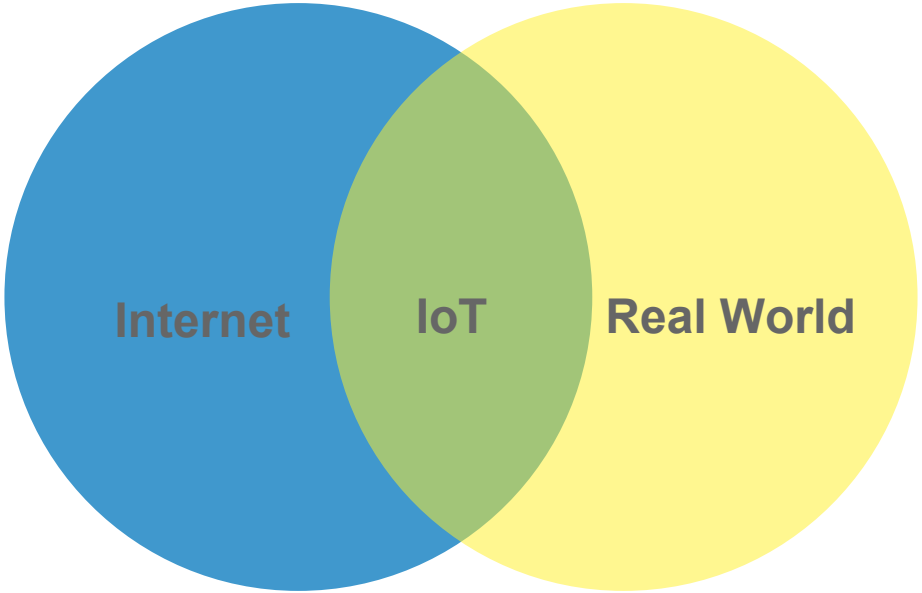
Assessing the Embedded Devices On Your Network

David Tomaschik, Google Security Assessments Team

About Me

- Security Engineer on Google’s Security Assessments Team
- CTF Player
- Maker
- Hold several certs, but actually proud of OSCP & OSCE

Insecurity of Things



What & Why?

- Black Box Assessment
  - Low Hanging Fruit
  - Component of Red Team/Pentest
  - Indication of Security Posture
- Target Audience
  - Software Security Background
  - ~No Electronics Knowledge Needed

Things will look a little different...

- Hardware, Software, and OS all come as part of a package.
- Almost always running a non-Windows OS
  - Most Common: Linux and VxWorks
  - Also Seen: FreeRTOS, eCos, etc.
- Most are non-x86
  - ARM, MIPS, PPC
- Often no visibility into internal workings
- Many devices made by hardware companies with no experience in:
  - Software
  - Security
  - Networking

Unique CIA Characteristics

Confidentiality

- Unexpected Data
  - Microphone
  - Camera
- Unusual Data
  - Biometrics

Integrity

- Physical Safety
  - Elevator Controllers
  - Machinery Controllers

Availability

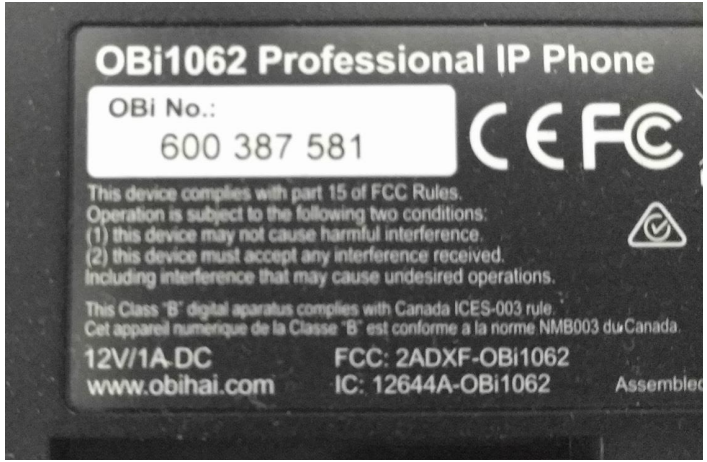
- Physical Safety
  - Elevator Controllers
  - Door Locks
- Chain Effects
  - Power Controllers

A Case Study


OBIHAI
12:22 1/19/15
Contacts
Current Calls
Call History
Product Info
Redial List
Call Forward
Do Not Disturb
Lines

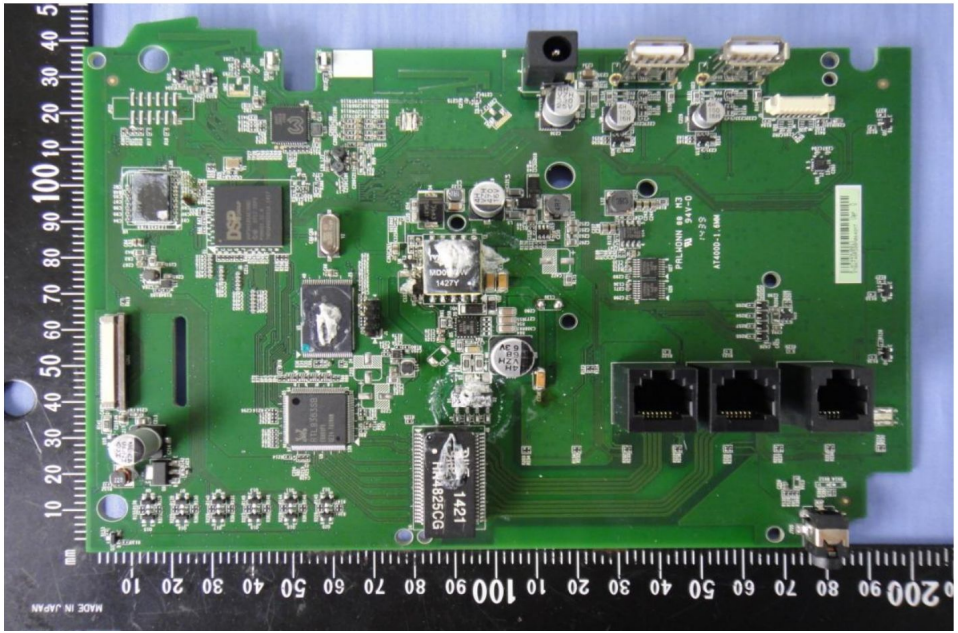
What do we know? (Recon)

- User manual
- FCC Database (fccid.io)
- Open it up and take a look!
- Firmware from website



What do we know? (Recon)

- User manual
- FCC Database (fccid.io)
- Open it up and take a look!
- Firmware from website

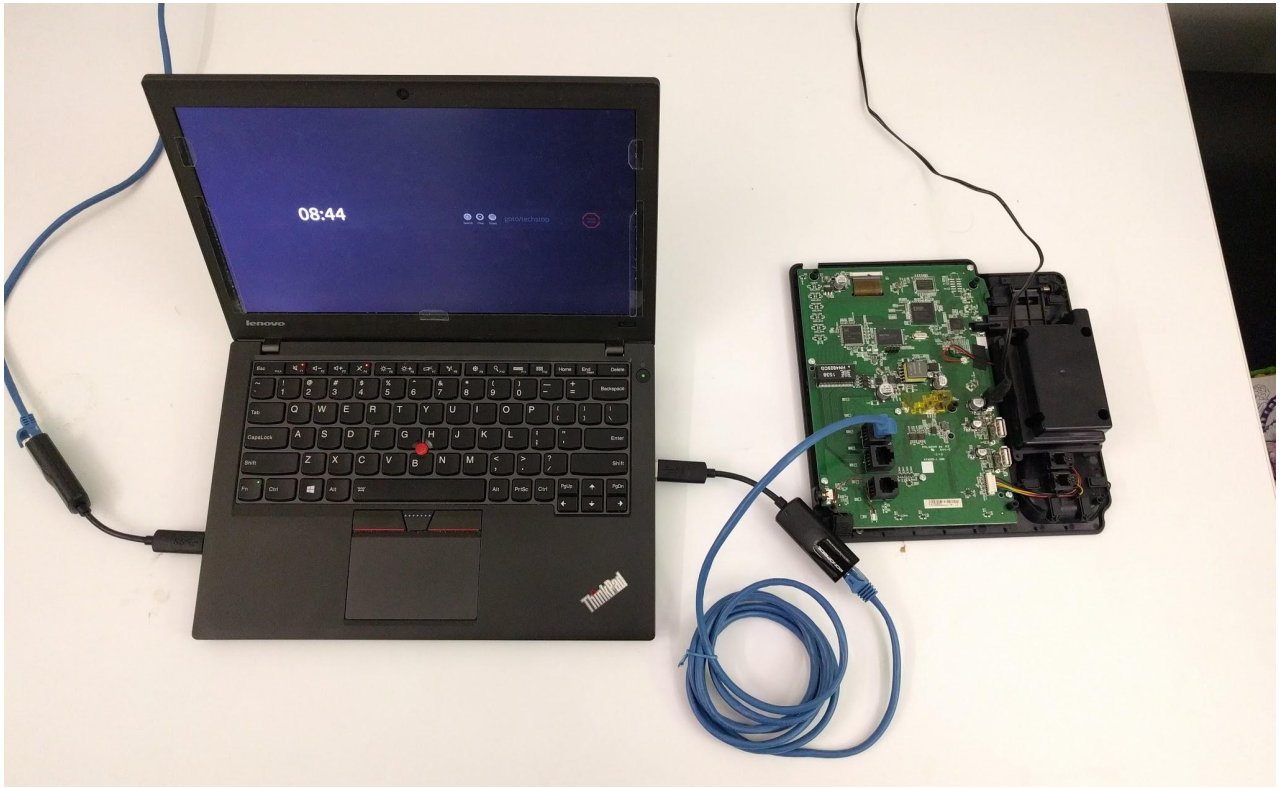


What do we know? (Recon)

- User manual
- FCC Database (fccid.io)
- Open it up and take a look!
- Firmware from website
  - binwalk
  - strings

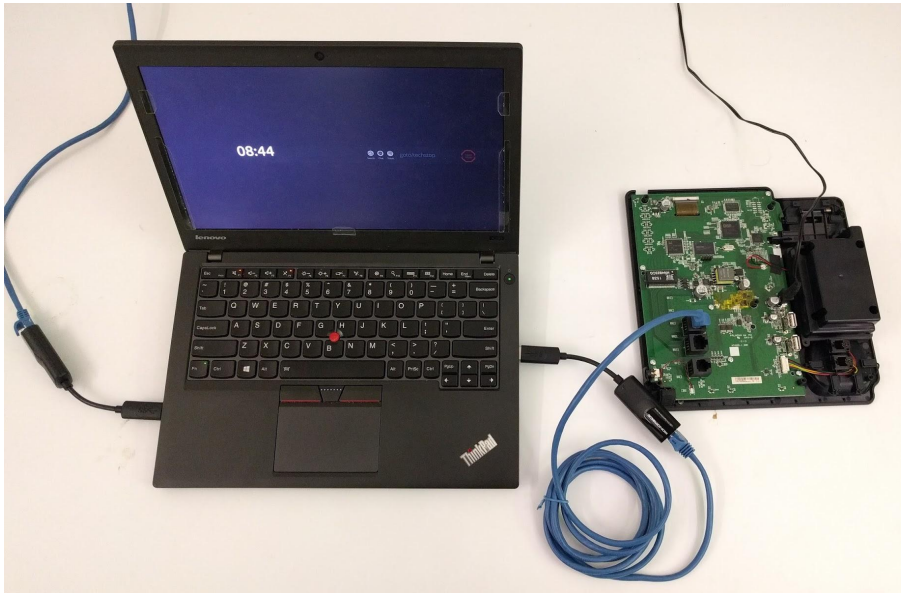
```
/obi/usbwifi.sh hidden "%s"
/obi/usbwifi.sh bss %s
/sbin/ifconfig wlan0 %s 2>/dev/null
/obi/usbwifi.sh aprefresh %s
```

Live Assessment

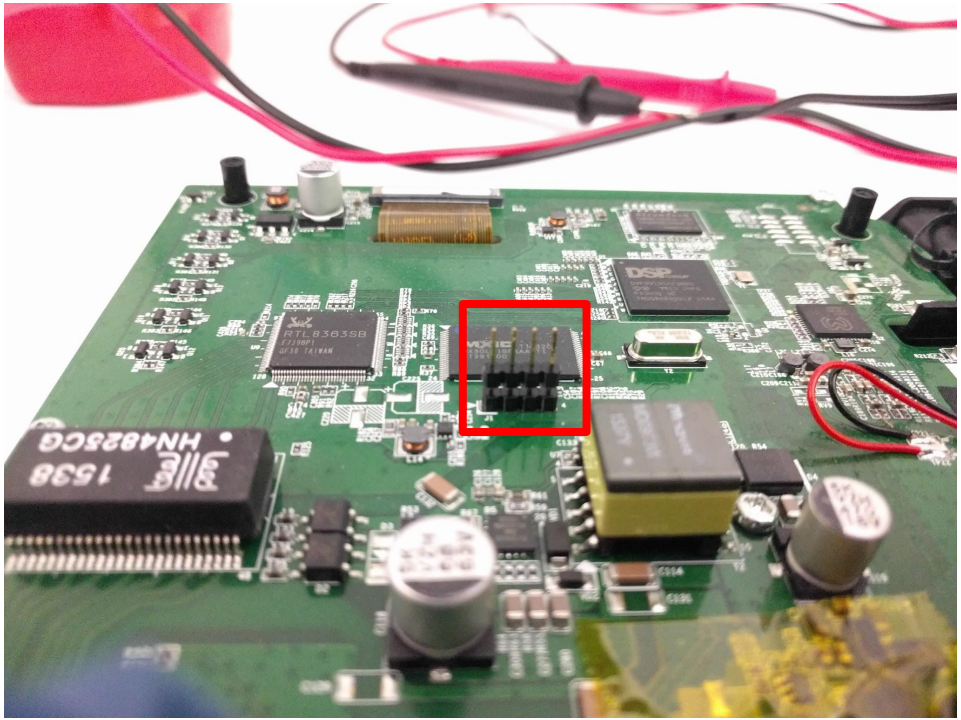


Live Assessment

- 2 USB NICs attached to VM
- Isolates traffic from testing host
- Capture all traffic
  - Bootup can be interesting!
  - Find IP from DHCP traffic, ping sweep
  - Maybe MITM HTTP(s)
- nmap
  - Go all out: it’s local.
  - `nmap -T4 -p- -sV -oA foo <IP>`
  - NMAP Scripts



Hardware Tricks

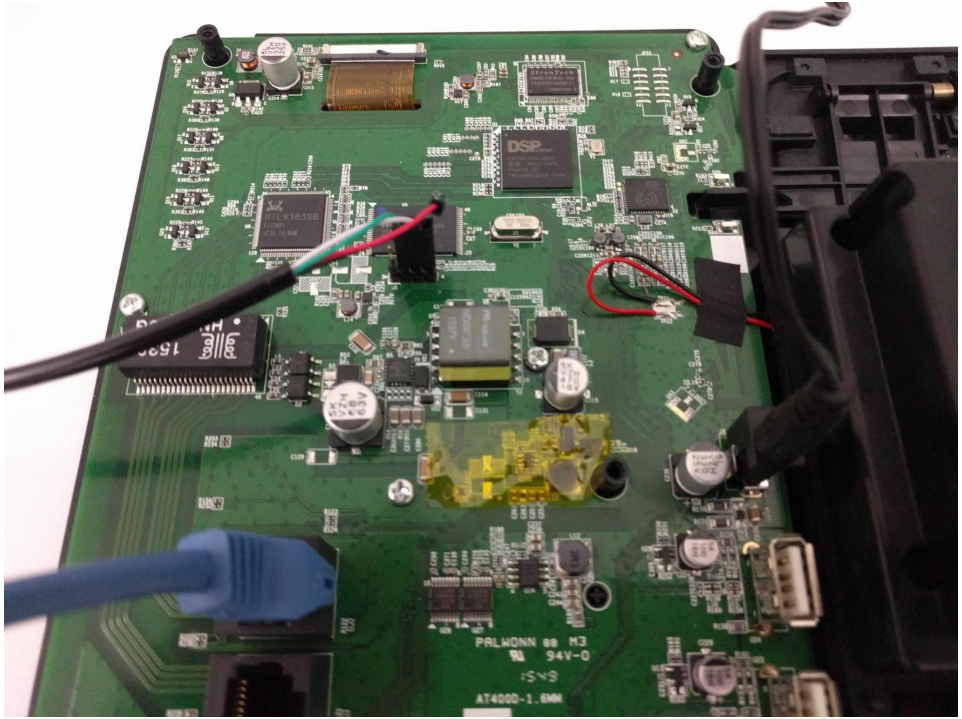


What's this?

4 pins, highlighted

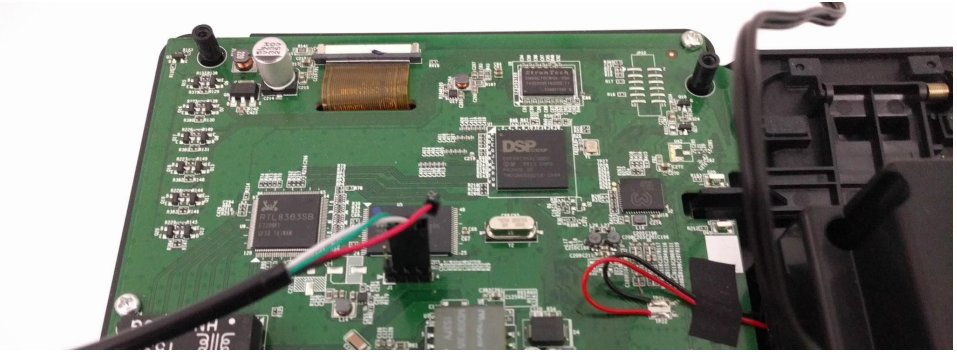
Hardware Tricks

- Serial Port
- UART (Universal Asynchronous Receiver Transmitter)
- Not RS-232!
- Cheap USB -> UART adapters
- Don’t connect the +V line!

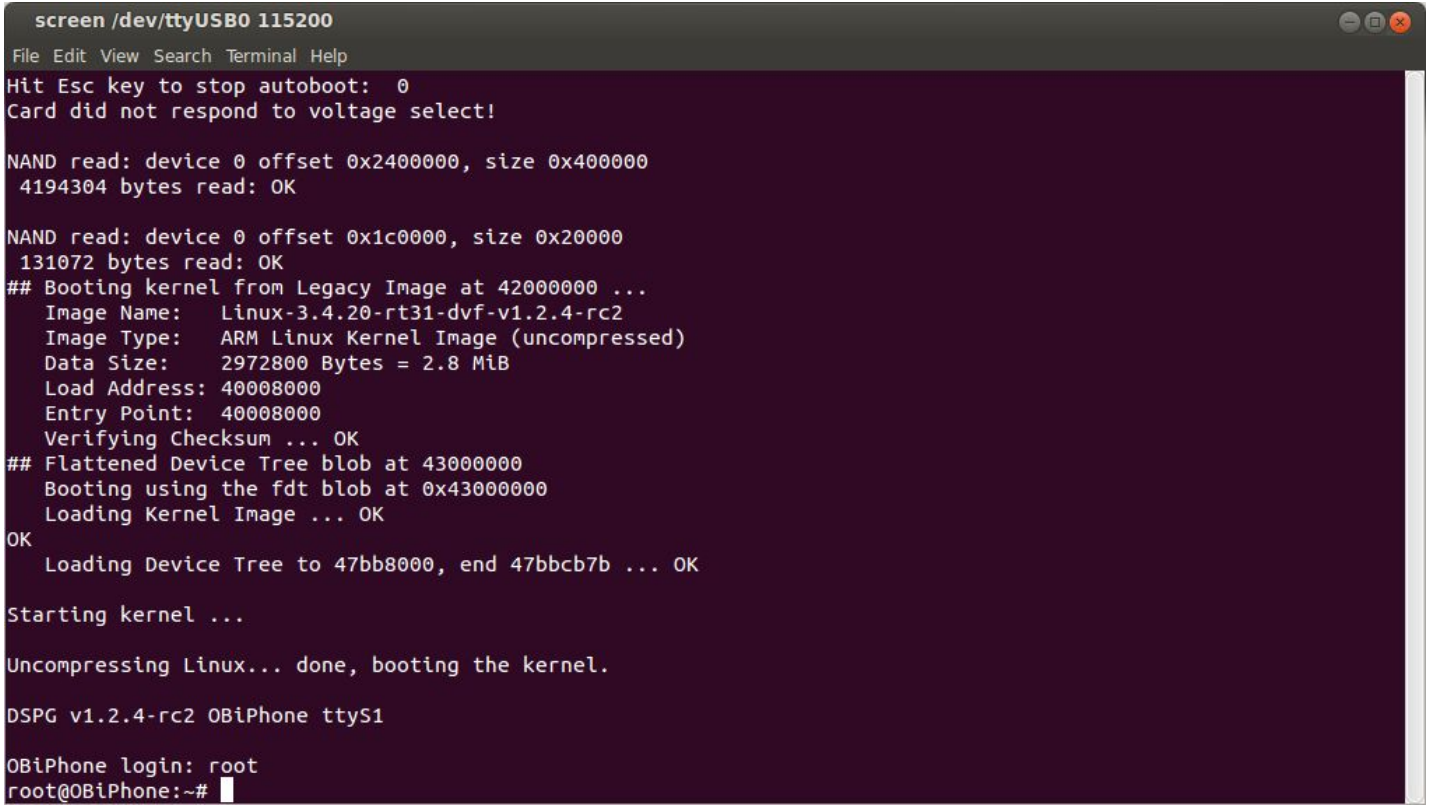


Hardware Tricks: Identifying Pins

- Multimeter in Resistance Mode
  - Find ground, 0 ohms to ground elsewhere
- Many ways to identify data pins
  - Logic Analyzer
  - Oscilloscope
  - Trial and Error
- Baud Rate
  - Logic Analyzer/Oscilloscope
  - Guess



Useful UART


screen /dev/ttyUSB0 115200
File Edit View Search Terminal Help
Hit Esc key to stop autoboot: 0
Card did not respond to voltage select!
NAND read: device 0 offset 0x2400000, size 0x400000
 4194304 bytes read: OK
NAND read: device 0 offset 0x1c0000, size 0x20000
 131072 bytes read: OK
Booting kernel from Legacy Image at 42000000 ...
   Image Name: Linux-3.4.20-rt31-dvf-v1.2.4-rc2
   Image Type: ARM Linux Kernel Image (uncompressed)
   Data Size: 2972800 Bytes = 2.8 MiB
   Load Address: 40008000
   Entry Point: 40008000
   Verifying Checksum ... OK
Flattened Device Tree blob at 43000000
   Booting using the fdt blob at 0x43000000
   Loading Kernel Image ... OK
OK
   Loading Device Tree to 47bb8000, end 47bbcb7b ... OK
Starting kernel ...
Uncompressing Linux... done, booting the kernel.
DSPG v1.2.4-rc2 OBiPhone ttyS1
OBiPhone login: root
root@OBiPhone:~#

Fuzzing/Debugging

- **Status Quo**
  - Emulation is hard(ware)
  - Watchdog triggers reboot
  - SIGSEGV handler
  - Capture state of crash?
    - No core files
    - No gdb
- **First Approach**
  - Cross-compiled gdbserver
  - Still rebooted each crash
  - Online approach, not suitable for fuzzing

Fuzzing/Debugging

- **Status Quo**
  - Emulation is hard(ware)
  - Watchdog triggers reboot
  - SIGSEGV handler
  - Capture state of crash?
    - No core files
    - No gdb
- **Second Approach**
  - Modify binary - nop out watchdog, SIGSEGV
  - Use core pattern across network
    `echo “|nc foo 9999” > /proc/sys/kernel/core_pattern`
  - Script analysis of core files
  - Can fuzz almost fast :)

Advanced Techniques

- JTAG
- Dumping Flash
  - In-place
  - Chip-off
- Extensive Firmware Modification & Replacement

Summary of Bugs

- **Memory Corruption**
  - Attacker-controlled free
  - Many null/invalid ptr dereferences
- **Command Injection**
  - WiFi config is hard, let’s shell out!
- **XSRF**
  - Everywhere
- **HTTP “Digest” Auth**
  - Ignore nonce, URI, etc.

Questions?

@Matir | https://systemoverlord.com