Having fun with IoT:
Reverse Engineering and Hacking of Xiaomi IoT Devices
DEFCON 26 – Dennis Giese

Outline

- Motivation

- Xiaomi Cloud

- Overview of devices

- Reverse Engineering of devices

- Modification of devices

About me

- Researcher at Northeastern University, USA
  - Working with Prof. Guevara Noubir@CCIS
- Grad student at TU Darmstadt, Germany
  - Working with Prof. Matthias Hollick@SEEMOO
- Interests: Reverse engineering of interesting devices
  - IoT, Smart Locks
  - Physical Locks ;)
- [Insert more uninteresting information here]
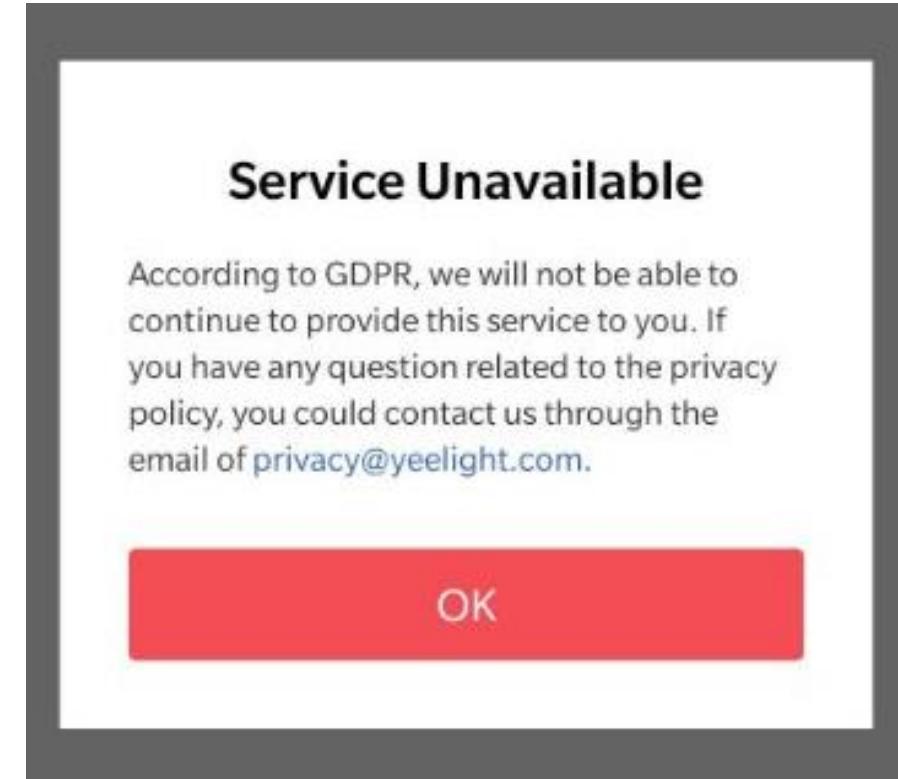
MOTIVATION

Why reverse IoT?

- (Find and exploit bugs to hack other people)

- De-attach devices from the vendor

- Enhance functionality
  - Add new features
  - Localization (e.g. Sound files)
  - Defeat Geo blocking

- Supporting other researchers

Service Unavailable

According to GDPR, we will not be able to continue to provide this service to you. If you have any question related to the privacy policy, you could contact us through the email of privacy@yeelight.com.

OK

Mon(IoT)or Lab@NEU

https://moniotrlab.ccis.neu.edu/

"Responsible disclosure"?

- Ethical question: "Responsible disclosure"?
  - Conflict:
    - Rootability vs. Device security
    - "Service for the Community" vs. Bug Bounty Program
 - Before DEFCON: contacted Xiaomi security team

How we started

May 2017
Mi Band 2
Vacuum Robot Gen 1

June 2017
Lumi Smart Home Gateway
+ Sensors

July 2017
Yeelink Lightbulbs (Color+White)
Yeelink LED Strip

How we continued

Yeelink Desk lamp
Philips Eyecare Desk lamp
Xiaomi Wi-Fi router

Yeelink/Philips Ceiling Lights
Philips Smart LED Bulb

Vacuum Robot Gen 2
Yeelink Bedside Lamp
Xiaomi (Ninebot) M365
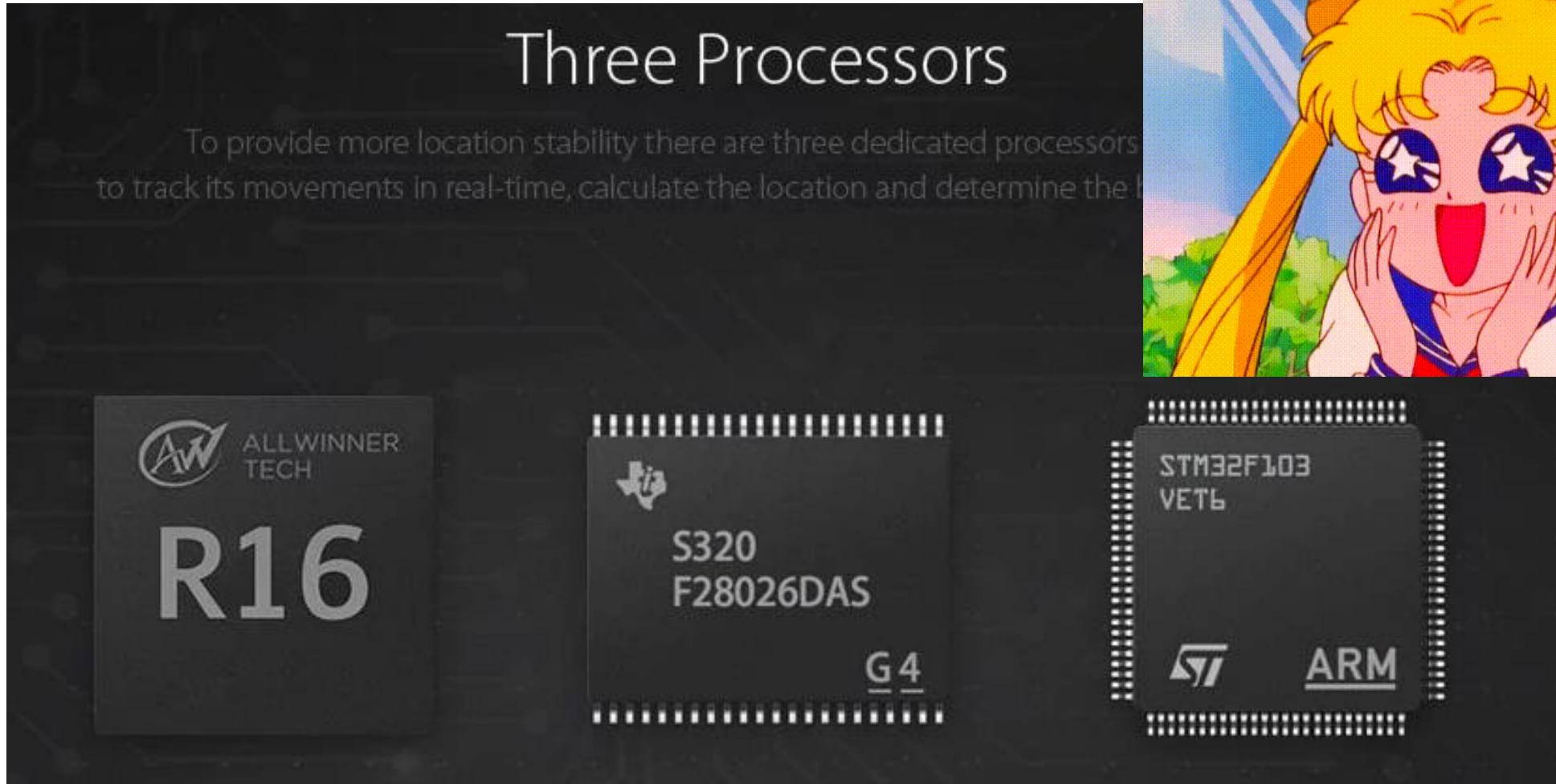
Lumi Aqara Camera
Yeelink Smart LED Bulb (v2)
Smart Power strip

Why Vacuum Robots?


Source: Xiaomi advertisment
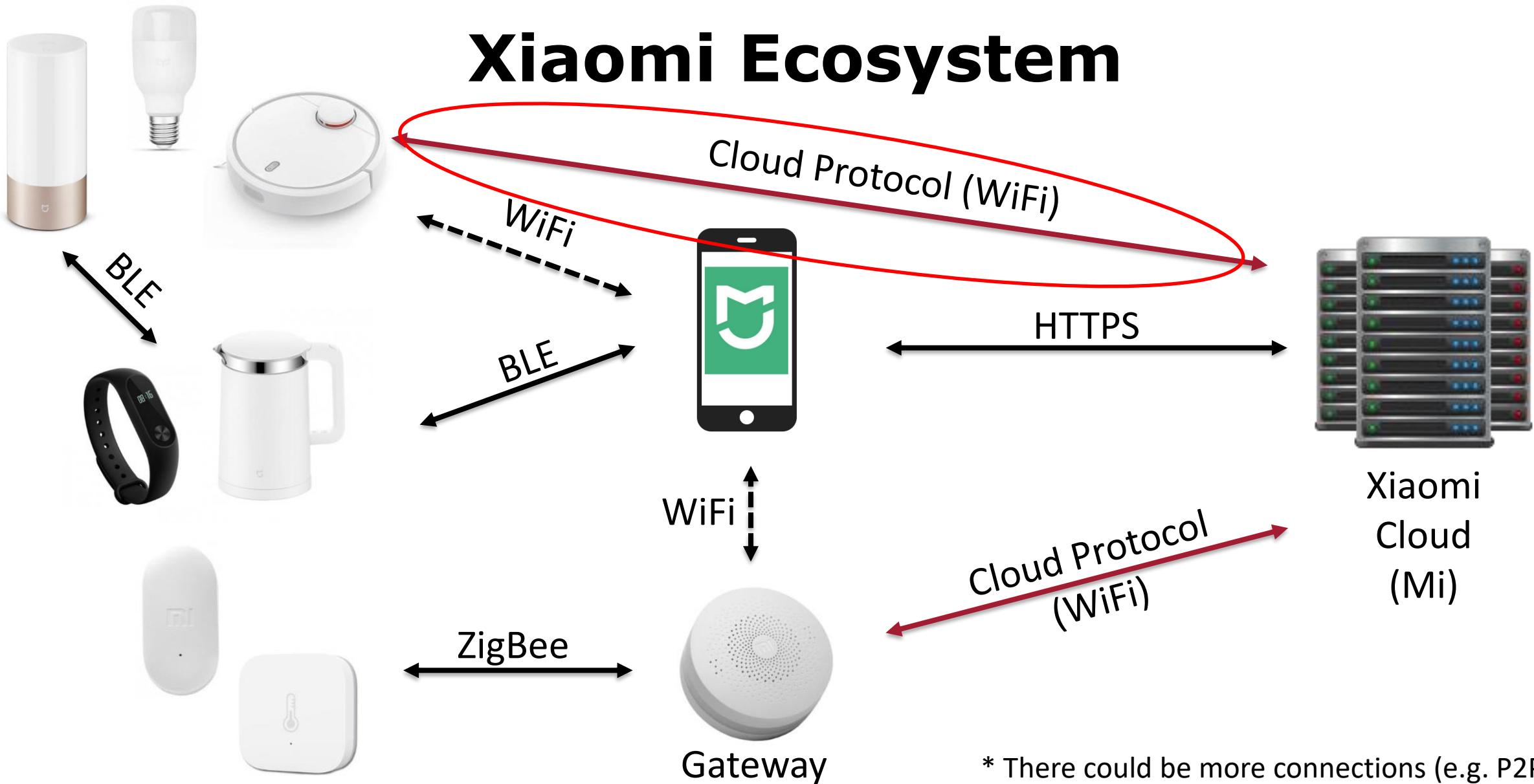
THE XIAOMI CLOUD

Xiaomi Cloud

- They claim to have the biggest IoT ecosystem worldwide
  - 85 Million Devices, 800 different models [1]
- Different Vendors, **one ecosystem**
  - Same communication protocol
  - Different technologies supported
  - Implementation differs from manufacturer
    - Software quality very different

[1]: https://www.espressif.com/en/media_overview/news/espressif-systems-integrated-xiaomis-plans-iot-development

https://github.com/MiEcosystem/miio_open
https://iot.mi.com/index.html

Xiaomi Ecosystem

Cloud Protocol (WiFi)

WiFi

BLE

BLE

HTTPS

WiFi

ZigBee

Cloud Protocol (WiFi)

Gateway

Xiaomi Cloud (Mi)

* There could be more connections (e.g. P2P, FDS)

Device to Cloud Communication

- DeviceID
  - Unique per device

- Keys
  - Cloud key (16 byte alpha-numeric)
    - Is used for cloud communication (AES encryption)
    - Static, is not changed by update or provisioning
  - Token (16 byte alpha-numeric)
    - Is used for app communication (AES encryption)
   - Dynamic, is generated at provisioning (connecting to new Wi-Fi)

Cloud protocol
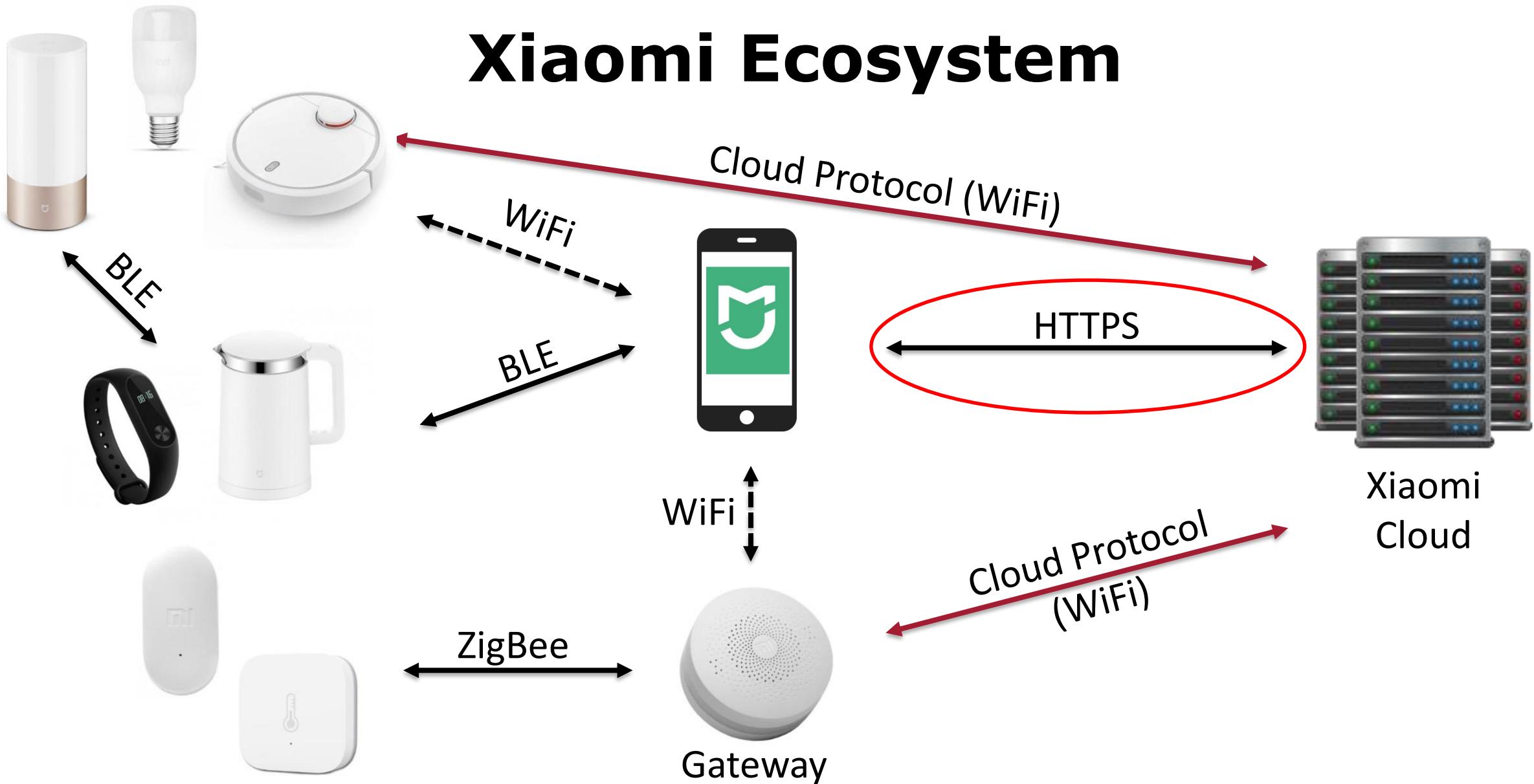
- Data
  - JSON-formatted messages
- Example of "Device registration"
  - {'id': 136163637, 'params': {'ap': {'ssid': 'myWifi', 'bssid': 'F8:1A:67:CC:BB:AA', 'rssi': -30}, 'hw_ver': 'Linux', 'life': 82614, 'model': 'rockrobo.vacuum.v1', 'netif': {'localIp': '192.168.1.205', 'gw': '192.168.1.1', 'mask': '255.255.255.0'}, 'fw_ver': '3.3.9_003077', 'mac': '34:CE:00:AA:BB:DD', 'token': 'xxx'}, 'partner_id': '', 'method': '_otc.info'}

Protocol for Firmware updates

- APP Updates
  - {"method":"miIO.ota","params":{"app_url":"http://cdn.cnbj0.fds.api.mi-img.com /miio_fw/upd_lumi.gateway.v3.bin?...","file_md5":"063df95bd5....cf11e","install":"1", "proc":"dnld install","mode":"normal"},"id":123}

- MCU/WiFi Updates
  - {"method":"miIO.ota","params":{"mcu_url":"http://cdn.cnbj0.fds.api.mi-img.com/ miio_fw/mcu_lumi.gateway.v3.bin? ...","install":"1","proc":"dnld install", "mode":"normal"},"id":123}

  No Integrity provided

- Subdevice Updates
  - {"crc32":"9460d9f0","image_type":"0101","manu_code":"115F","md5":"e9d62...a74d 8","model":"lumi.plug.v1","size":"186978","url":"http://cdn.cnbj2.fds.api.mi-img.com /lumi-ota/aiot-ota/LM15_SP_mi_V1.3.22_..._OTA_v22_withCRC.ota"}

Xiaomi Ecosystem

Cloud Protocol (WiFi)

WiFi

BLE

BLE

HTTPS

WiFi

ZigBee

Cloud Protocol (WiFi)

Xiaomi Cloud

Gateway

App to Cloud communication

- Authentication via OAuth

- Layered encryption

  - Outside: HTTPS

  - Inside: AES using a session key

- Message format: JSON RPC

- Device specific functions: provided by Plugins

App to Cloud communication
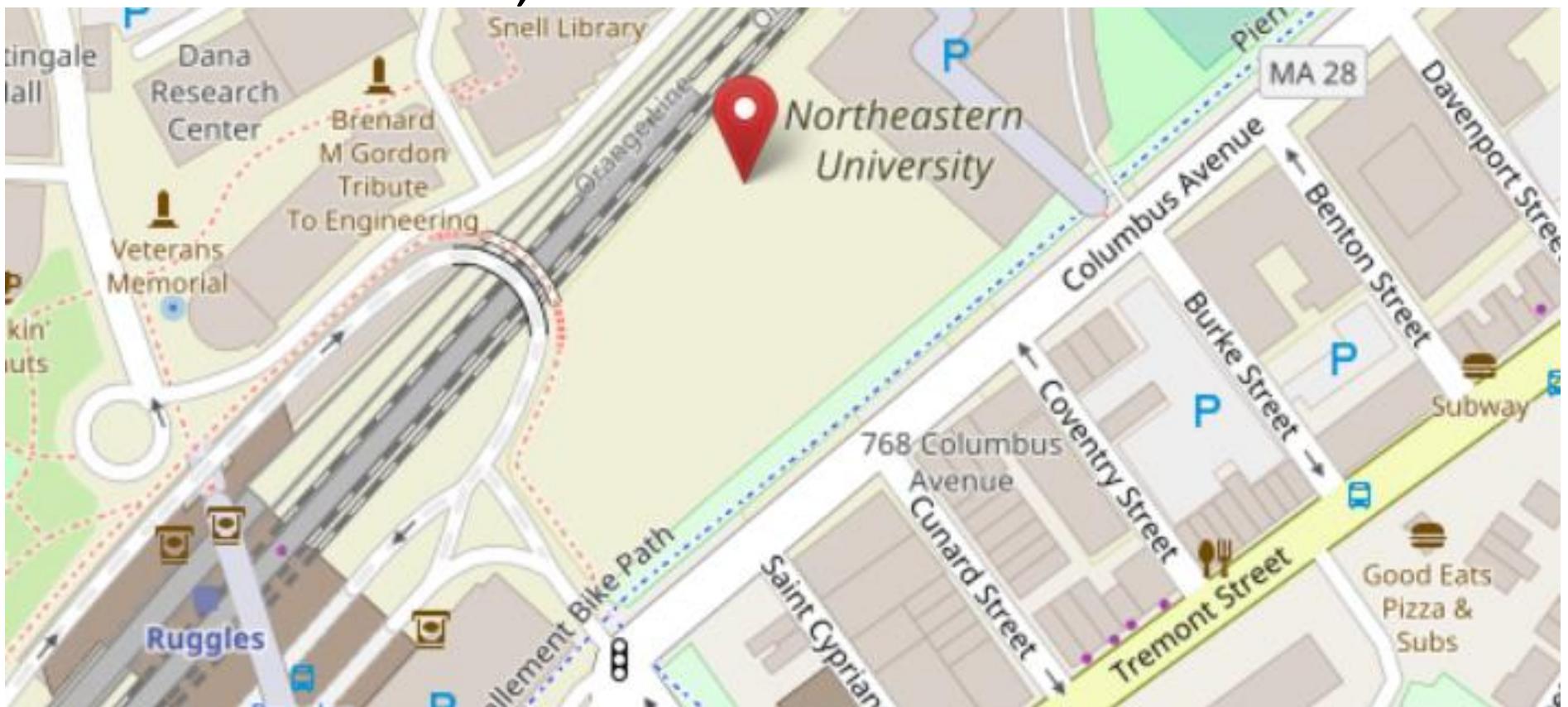
- REQ: api.io.mi.com/home/device_list method:POST params:[]
- RES:
{"message":"ok","result":{"list":[{"did":"659812 bc...zzz","name":"Mi PlugMini","localip":"192.1("mac":"34:CE:00:AA:BB:CC","ssid":"IoT","bssid" DD:EE","model":"chuangmi.plug.m1", "longitude":"-71.0872248","latitude":"42.33794500", "adminFlag":1,"shareFlag":0,"permitLevel":16,"isOnline":true, "desc":"Power plug on ","rssi":-47}
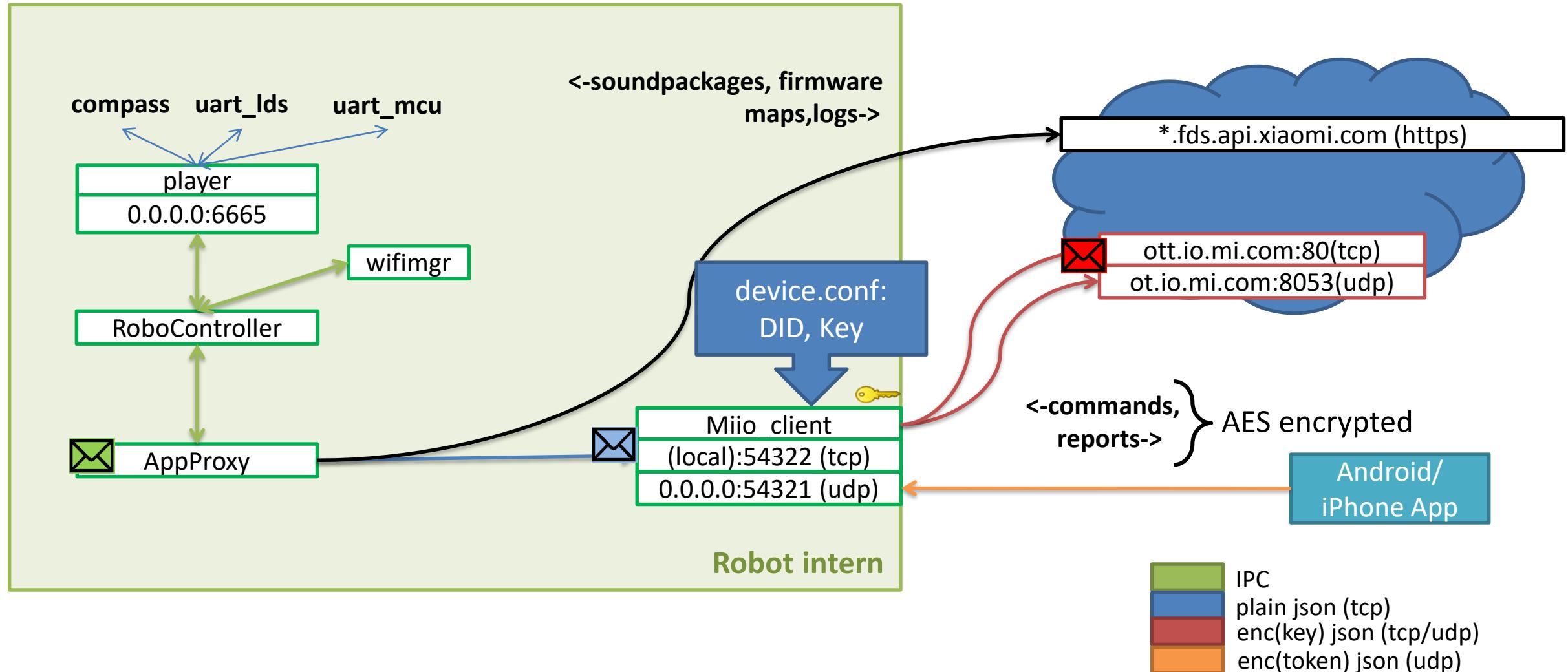
App to Cloud communication

- "longitude":"-71.0872248","latitude":"42.33794500"

Source: Openstreetmaps

Example of Communication relations
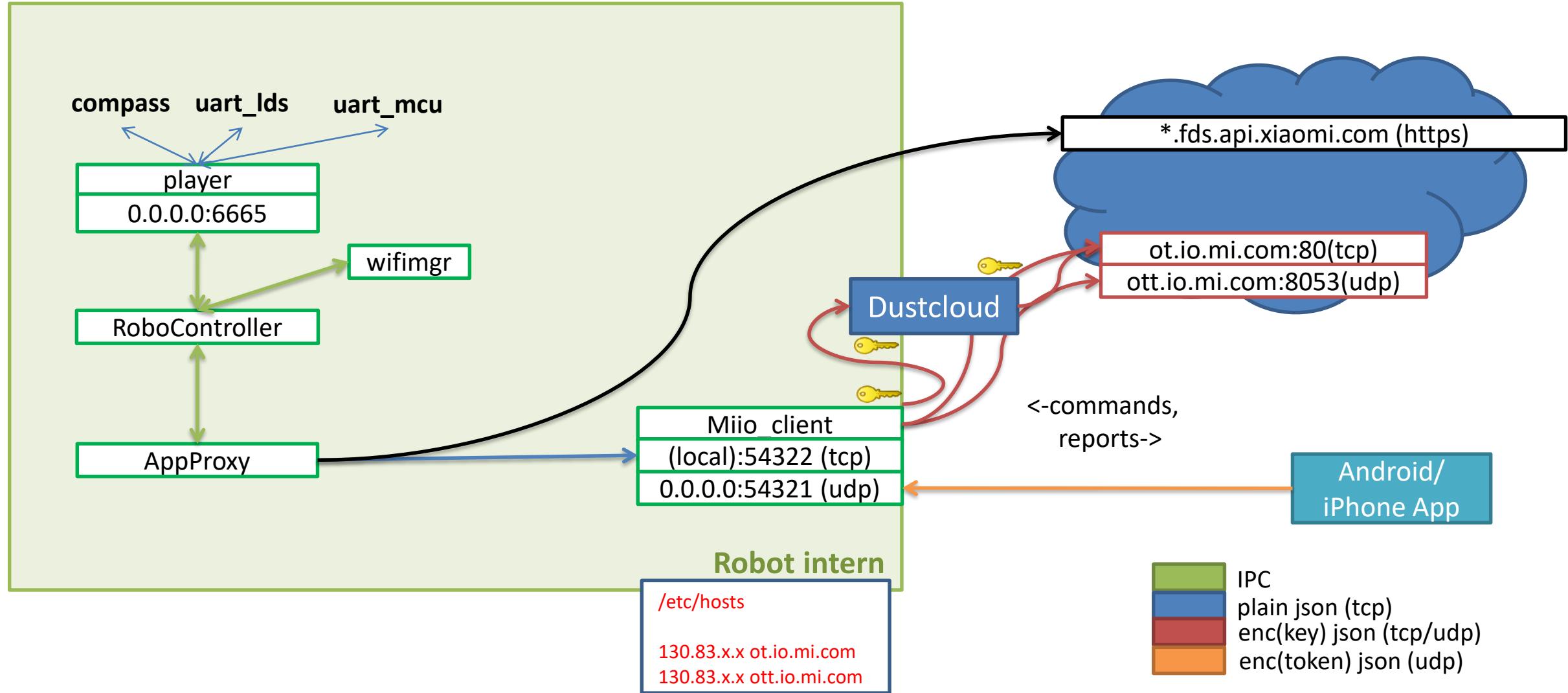
How to gain Independence

Xiaomi Cloud

Copyright: 20th Century Fox

Proxy cloud communication

What is Dustcloud?
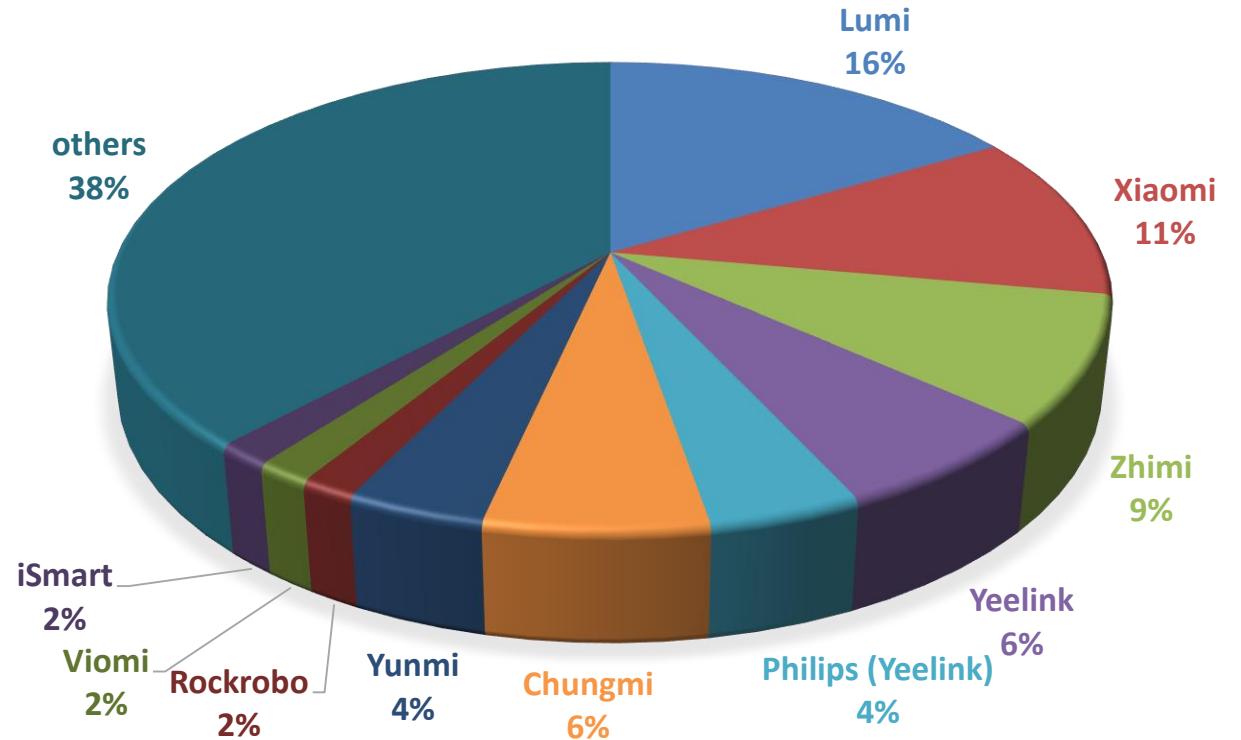
- Proxy or endpoint server for devices
  - Acts as Xiaomi Cloud emulation
  - Reads traffic in plaintext
  - May send commands to the device
    - Change or suppress commands (e.g. Updates)
- Requirements: Device ID, Cloud Key, DNS Redirection

LETS TAKE A LOOK AT THE PRODUCTS

Products

- ~260 different models supported (WiFi + Zigbee + BLE)
- Depending on selected server location
  - Mainland China
  - Taiwan
  - US
  - ...
  - models not always compatible
- My inventory: ~42 different models
  - 99 devices in total

Lumi 16%
Xiaomi 11%
Zhimi 9%
Yeelink 6%
Philips (Yeelink) 4%
Chungmi 6%
Yunmi 4%
Rockrobo 2%
Viomi 2%
iSmart 2%
others 38%

Values estimated, Mi Home 5.3.13, Mainland China Server

Products

Different architectures

- ARM Cortex-A

Focus of this talk

- ARM Cortex-M
  - Marvell 88MW30X (integrated WiFi)
  - Mediatek MT7687N (integrated WiFi + BLE)

Focus of my binary patching talk @IoT Village today

- MIPS

- Xtensa
  - ESP8266, ESP32 (integrated WiFi)

"Why I hate ESP8266" @IoT Village today

Operation Systems

- „Full Linux" e.g. Ubuntu 14.04
  - Vacuum cleaning robots
- OpenWRT
  - Xiaomi Wifi Speaker, Routers, Minij washing machine
- Embedded Linux
  - IP cameras
- RTOS
  - Lightbulbs, ceiling lights, light strips
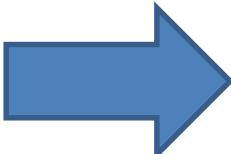
Implementations

	Vacuum Robot	Smart Home Gateway*	Philips Ceiling Light Yeelink Bedside Lamp
Manufacturer	Rockrobo	Lumi United	Yeelight
MCU	Allwinner + STM + TI	Marvell (Wi-Fi)	MediaTek (Wi-Fi + BLE)
Firmware Update	Encrypted + HTTPS	Not Encrypted (No SSL stack!)	Not Encrypted + HTTPS (No Cert check!)
Debug Interfaces	Protected	Available	Available

Bonus: Chinese device, but unknown communication to Server in Salt Lake City, USA (166.70.53.160)

roborock

LUMI

YEELIGHT

*Does not apply for DGNWG03LM (Gateway model for Taiwan)
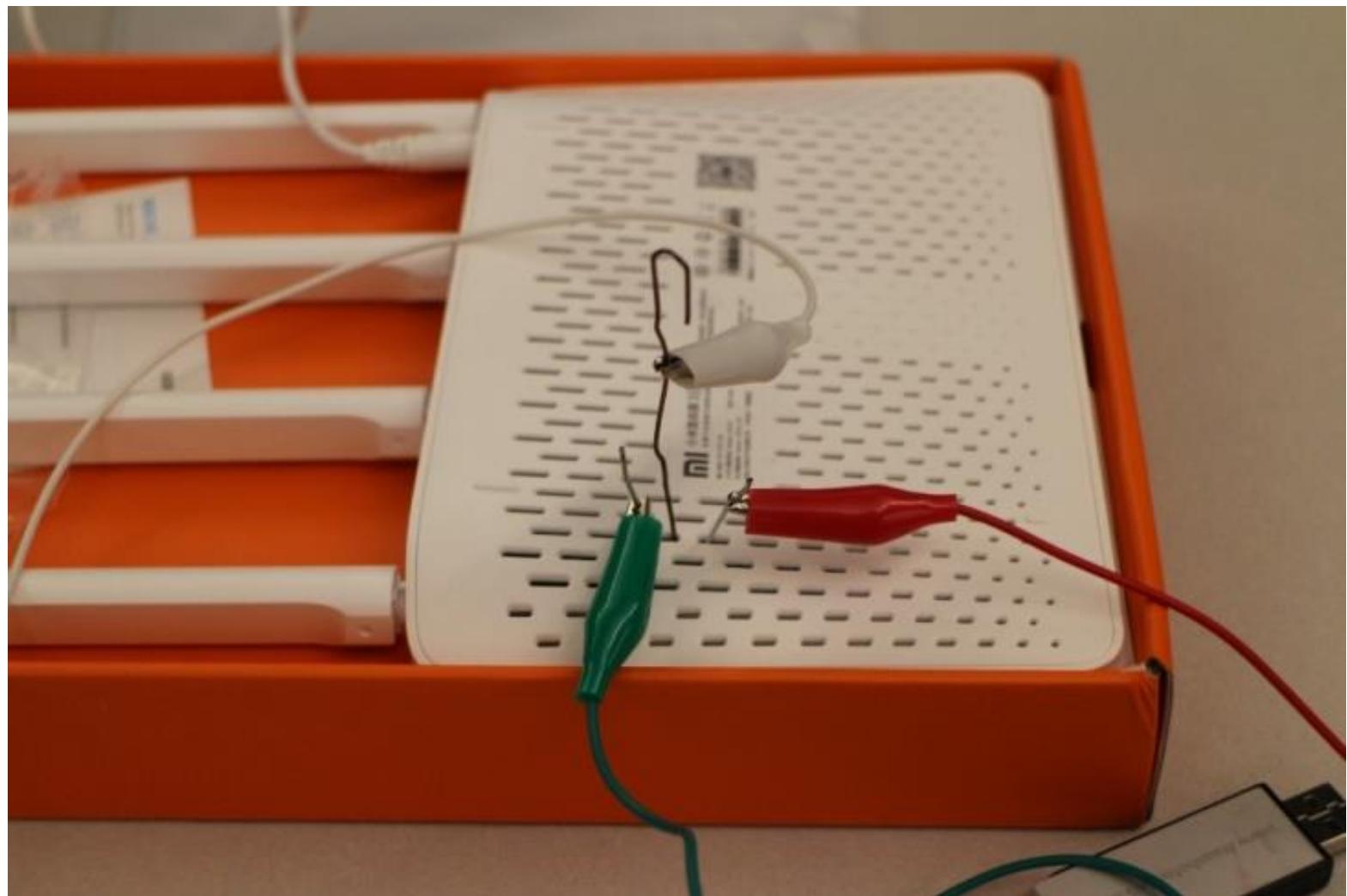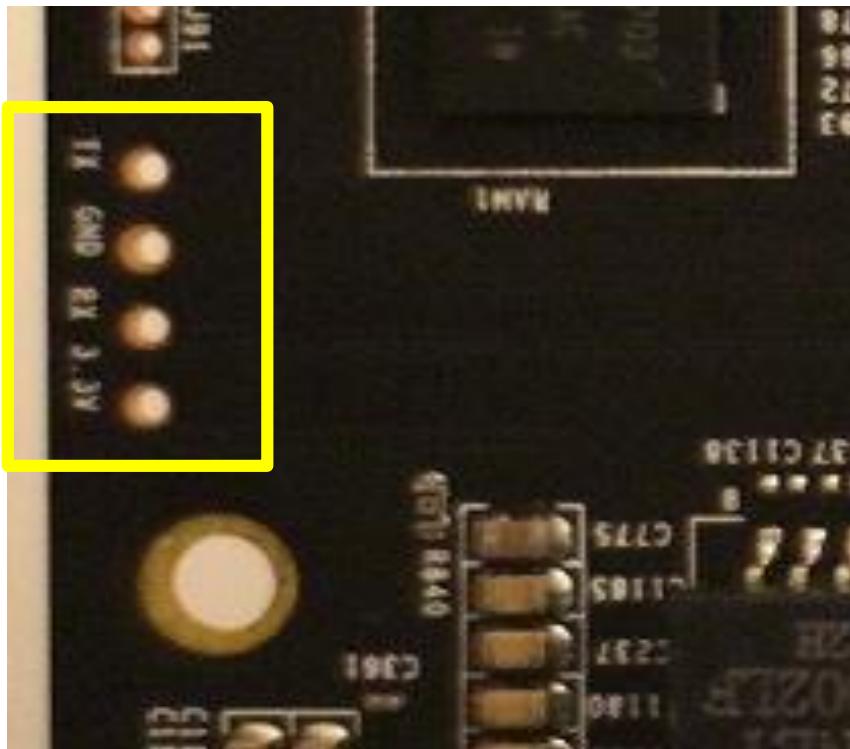
Good news

- Vendors/Developers are lazy

- Assumed development of firmware:

  – Take SDK/toolchain

  – Modify sample that the product runs

  – If it works: publish firmware

  ➡ All firmwares very similar (memory layout, functions, strings, etc)

LETS GET ACCESS TO THE DEVICES

Warranty seal?
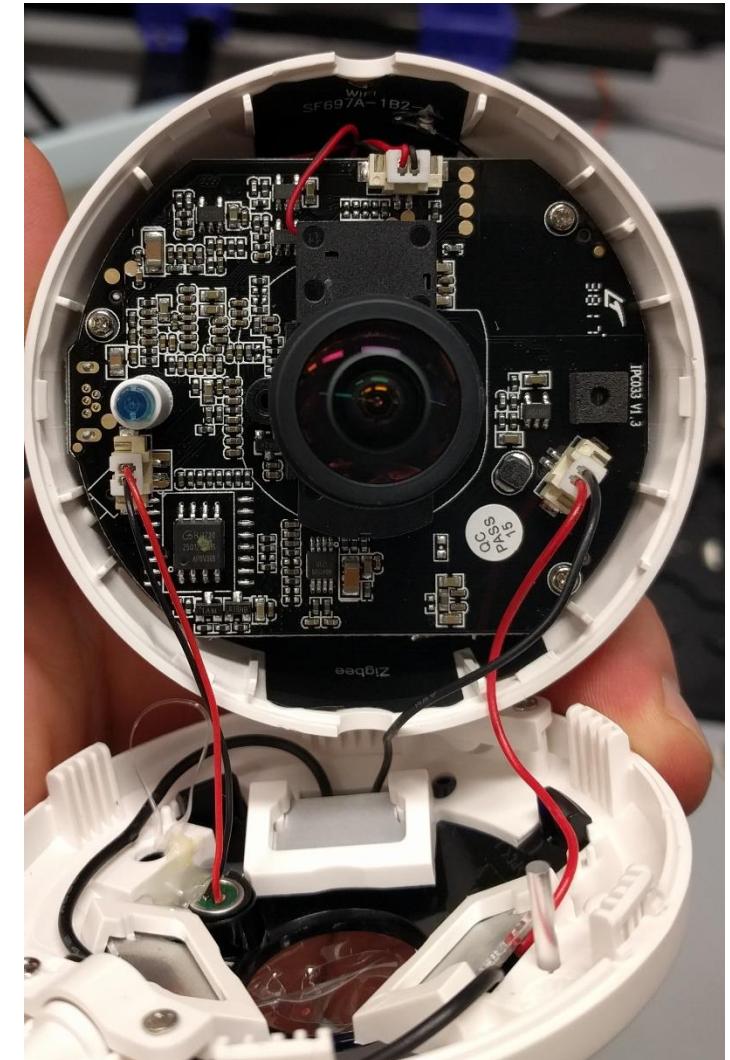
AQARA SMART IP CAMERA

Applies to: lumi.camera.aq1

Overview Hardware
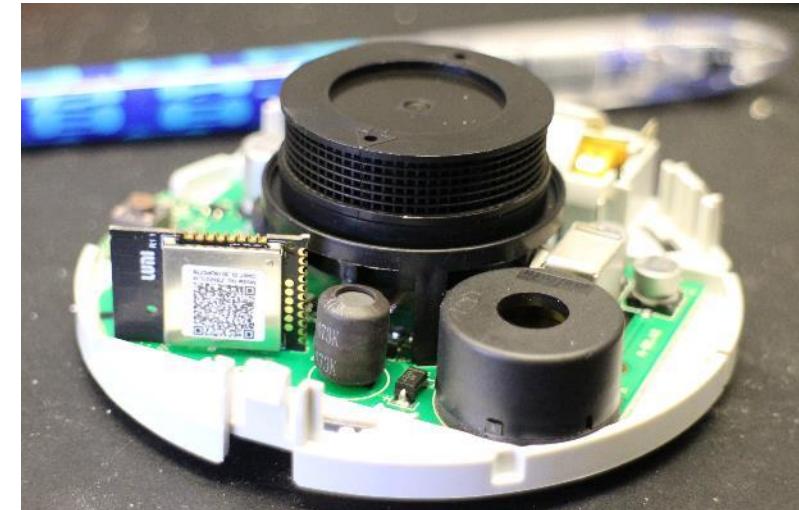
- CPU: Hi3518EV200
  - ARM Cortex-A
- RAM: 64MB
- Flash: 16MByte
- Wi-Fi: Mediatek MT7601UN via USB
- OS: Embedded Linux
- Zigbee-MCU: NXP JN5169
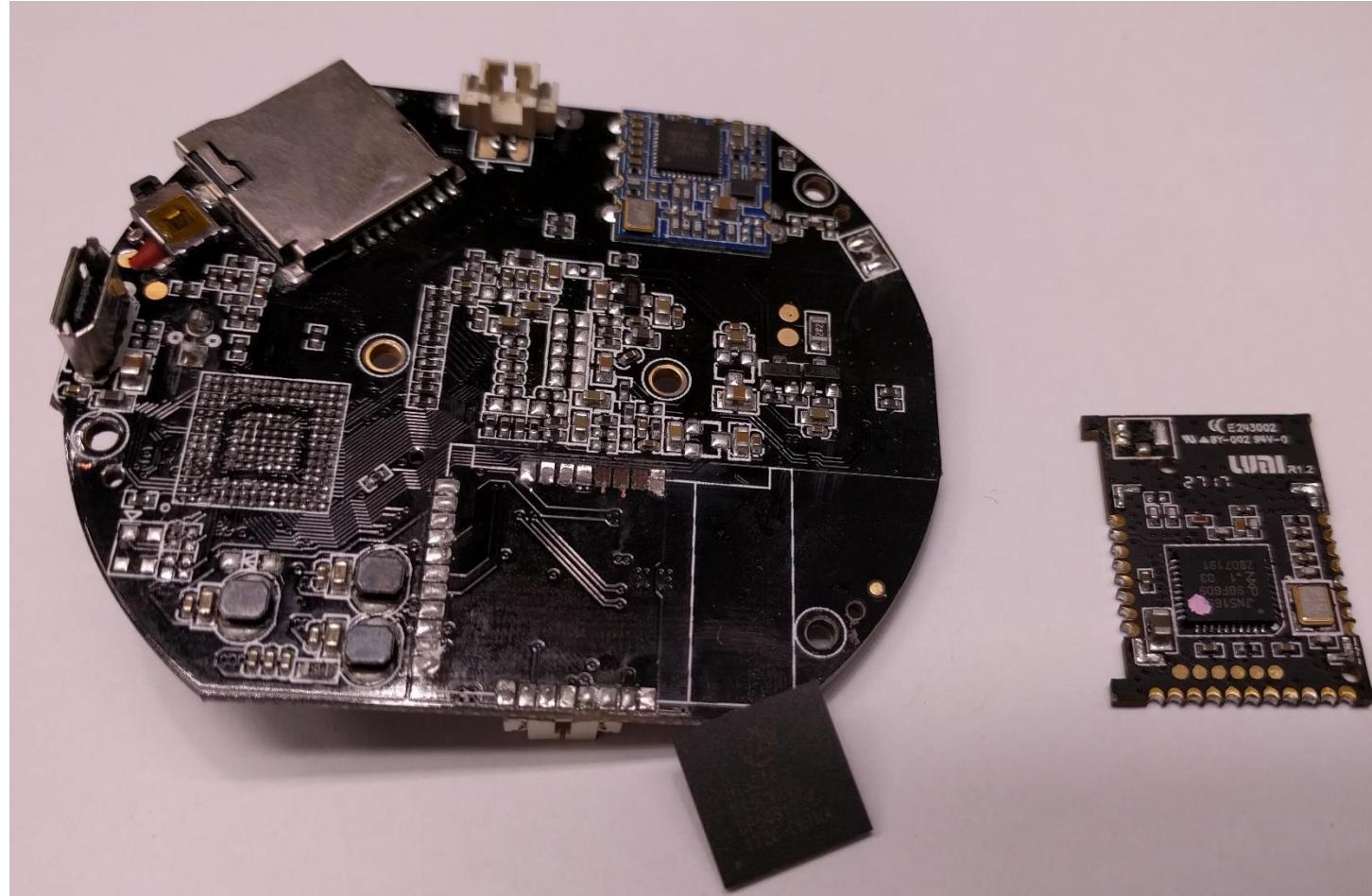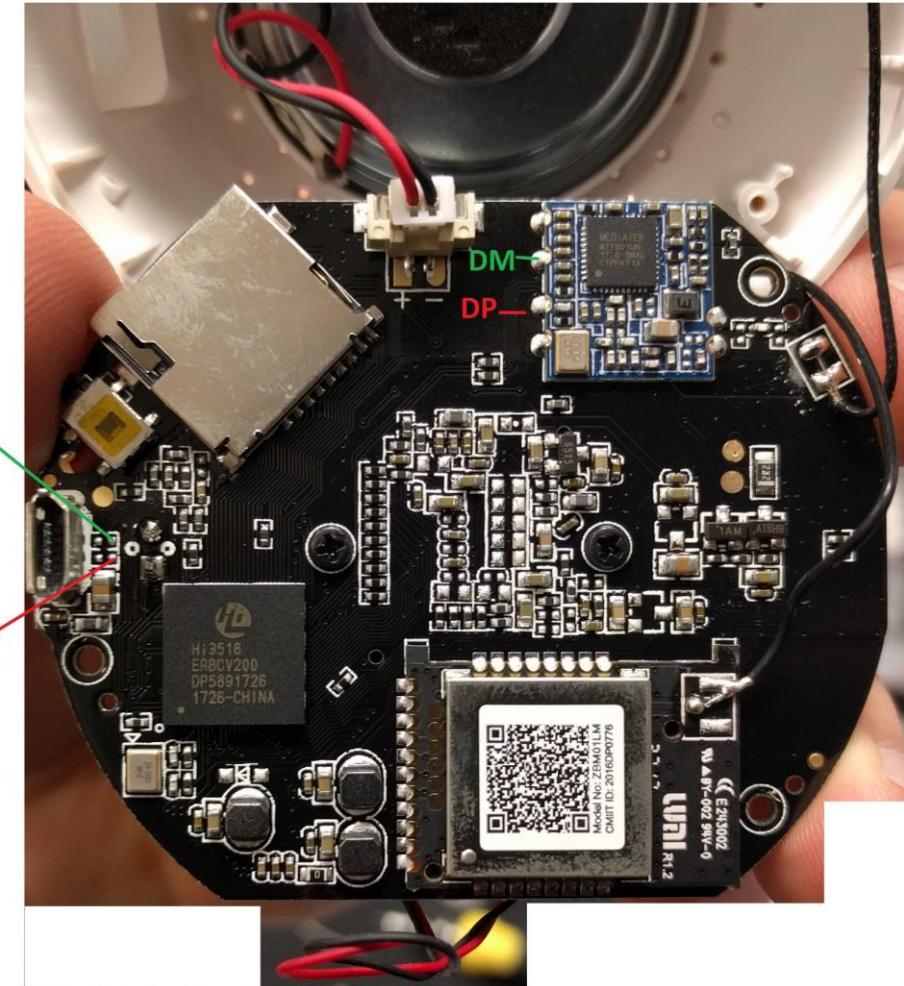
Devices connected via Zigbee

Zigbee (NXP JN5169) based

- Motion Sensor

- Temperature sensors

- Power Plug

- Smoke Detectors

- Smart Door Lock

- …

Serial port after bricking device

Leaked information

- JFFS2 filesystem not properly cleaned

- 3 different credentials from development devices leaked

```
0004cc10   e3 b5 3b e8 00 2c 23 20   63 61 74 20 2f 65 74 63   |..;..,# cat /etc|
0004cc20   2f 6d 69 69 6f 2f 64 65   76 69 63 65 2e 63 6f 6e   |/miio/device.con|
0004cc30   66 0a 23 20 64 69 64 20   6d 75 73 74 20 62 65 20   |f.# did must be |
0004cc40   61 20 75 6e 73 69 67 6e   65 64 20 69 6e 74 0a 23   |a unsigned int.#|
0004cc50   20 6b 65 79 29 70 00 00   4e 73 74 72 69 6e 67 0a   | key)p..Nstring.|
0004cc60   23 0a 64 69 64 3d 35 30   36 30 33 36 35 XX 0a 6b   |#.did=5060365X.k|
0004cc70   65 79 3d 4e 41 37 4e 69   6d 4b 6f XX XX XX XX XX   |ey=NA7NimKoXXXXX|
0004cc80   69 58 6e 0a 6d 61 63 3d   32 38 3a 36 43 3a 30 37   |iXn.mac=28:6C:07|
0004cc90   3a 32 45 3a XX XX 3a XX   XX 0a 76 65 6e 64 6f 72   |:2E:XX:XX.vendor|
0004cca0   3d 6c 75 6d 69 0a 23 20   6d 6f 64 65 6c 20 6d 61   |=lumi.# model ma|
0004ccb0   78 20 6c 65 6e 20 32 33   0a 80 02 94 03 00 02 2e   |x len 23........|
0004ccc0   63 61 6d 65 72 61 2e 61   71 31 0a 70 32 70 5f 69   |camera.aq1.p2p_i|
0004ccd0   64 3d 41 2c 00 00 03 30   31 31 31 41 0a 11 00 00   |d=A,...0111A....|
```

Rooting

- Serial was not necessary
  - open telnet server (port 23)
  - hardcoded root password in /etc/shadow
    - "root:IIfCcCAiKWPNs:17333:0:99999:7::"
    - DES-Crypt -> password truncated to 8 chars
    - Password: "lumi-201"
  - Same credentials for all cameras

Modifications
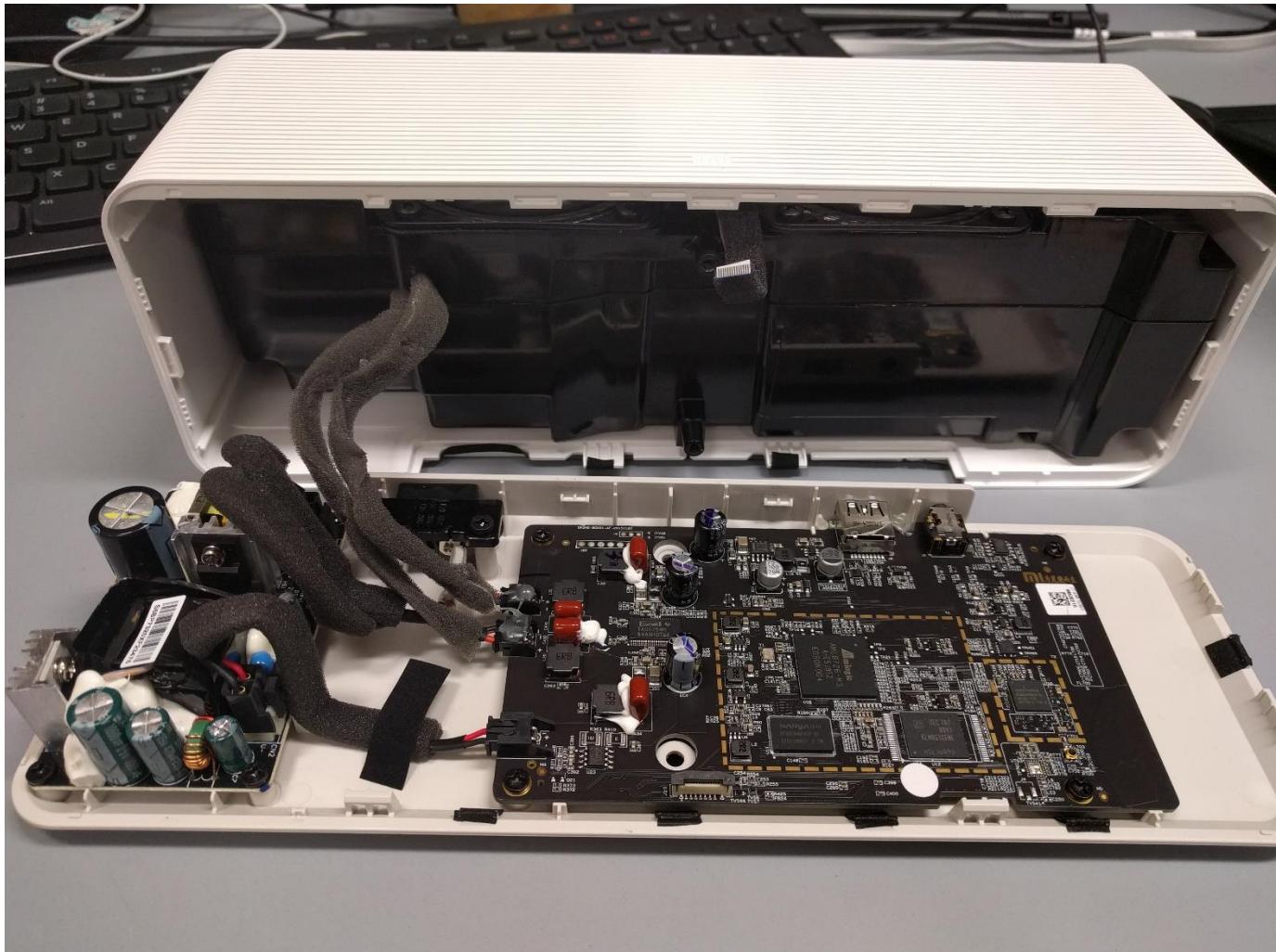
- Replace Chinese sound files
- Replace telnetd by dropbear (SSH)
- Change root password
- Replace Camera Software

WI-FI NETWORK SPEAKER
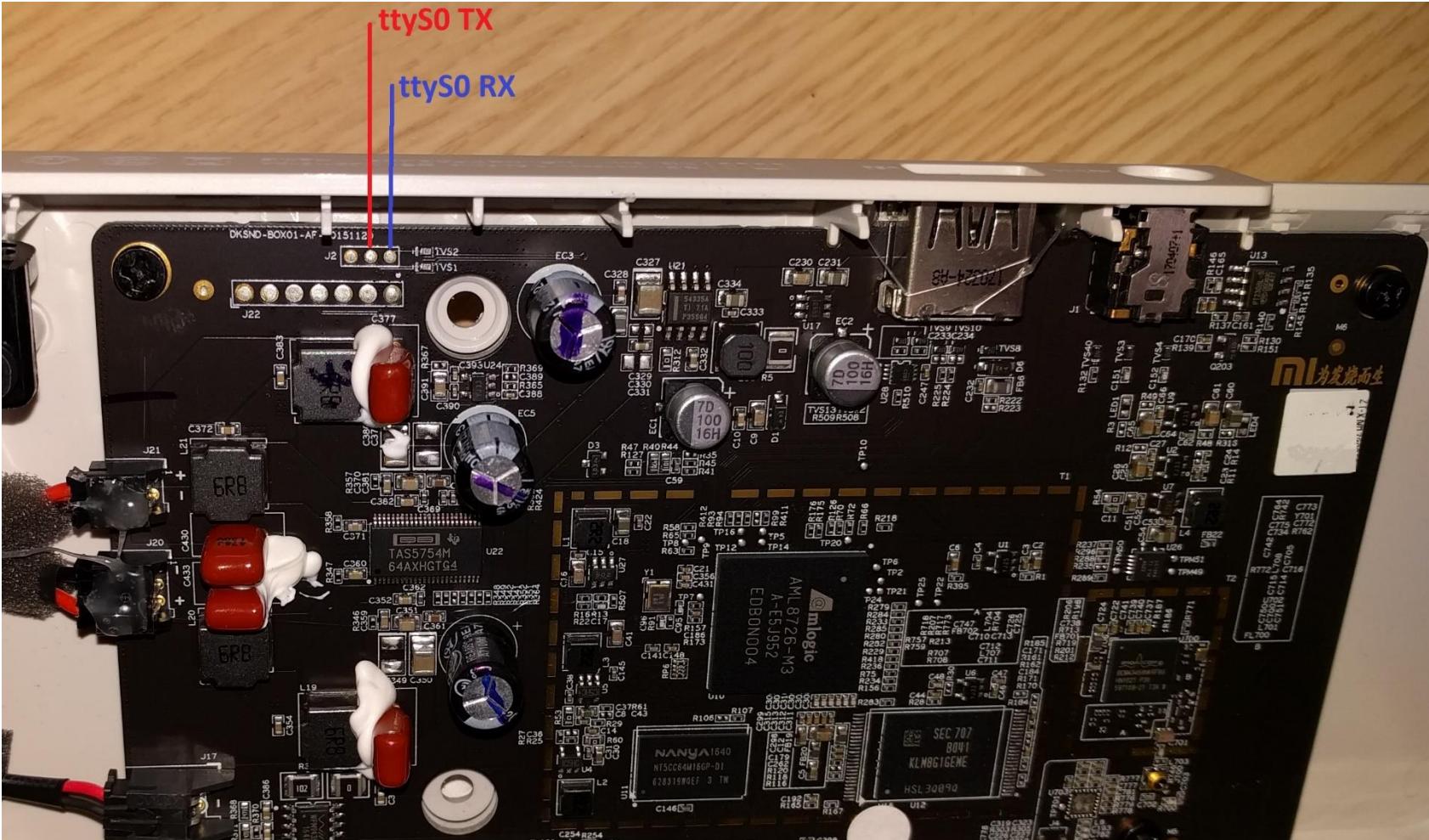
Applies to: xiaomi.wifispeaker.v1, basic idea also for xiaomi.router.*

Overview Hardware

- CPU: Amlogic Meson3
  - ARM Cortex-A
- RAM: 128MB
- Flash: 8GByte
- WI-Fi+BT: Broadcom BCM4345
- OS: OpenWRT
  - Samba 3.x
- Released: End 2016

Serial Port

Rooting

- Teardown of device not necessary

- Classic vulnerability: no input validation

> http://{ip}:9999/{ssdp id}/Upnp/resource/sys?command=nslookup&host=`echo 192.168.0.2`&dns_server=`/etc/init.d/ssh start`

Update (08.08.2018): Xiaomi claims this was fixed in an internal release in April 2018

Research in cooperation with "teprrr"

Firmware updates

- Query Update Information over HTTP
  - http://soundbar.pandora.xiaomi.com/XXXXXXX/XXXXXX
- Firmware updates over HTTP
  - packed LZMA in XML format
  - EXT2 images
  - No signatures

```
-<update>
    <md5sum>93b38d5dae7314893bfebe7f█████</md5sum>
    <total>40627992</total>
    <real_image_size>138801643</real_image_size>
    <image_offset>399</image_offset>
    <image_size>40627593</image_size>
    <online_update_flag>1</online_update_flag>
    -<package>
        http://package.box.xiaomi.com/mfsv2/download/fdsc3/p019ba████/aXEhZE8█████.zip
    </package>
    <image_md5sum>0568ad19c234405462378dfb██████</image_md5sum>
    -<image_package_backup>
        http://package.box.xiaomi.com/mfsv2/download/fdsc3/p01Od████/S73XpVW█████.zip
    </image_package_backup>
    -<image_package>
        http://package.box.xiaomi.com/mfsv2/download/fdsc3/p01Od████/S73XpVW█████.zip
    </image_package>
    -<package_backup>
        http://package.box.xiaomi.com/mfsv2/download/fdsc3/p019ba████/aXEhZE8█████.zip
    </package_backup>
    <package_version>1.4.0.20180403.194524</package_version>
    <image_version>1.4.0.20180403.194524</image_version>
</update>
```
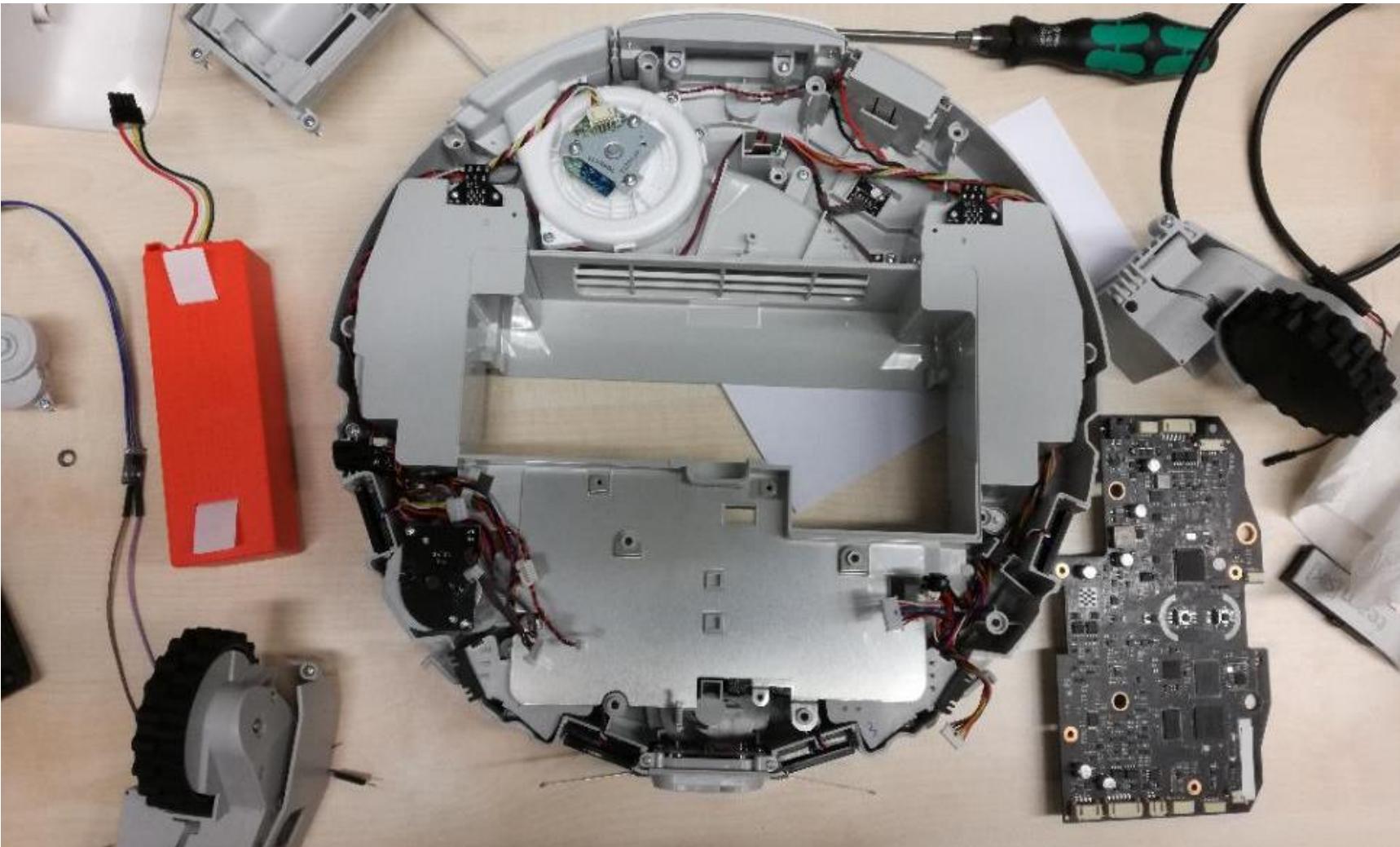
VACUUM CLEANING ROBOTS

Gen 1 Device Overview

Laser Distance Sensor (LDS)
Wall Sensor
Dustbin Sensor
Collision Sensor
Ultrasonic Radar Sensor

Electronic Compass
Cliff Sensor
Gyroscope / Accelerometer
Drop Sensor
Speedmeter
Fan Speed Sensor
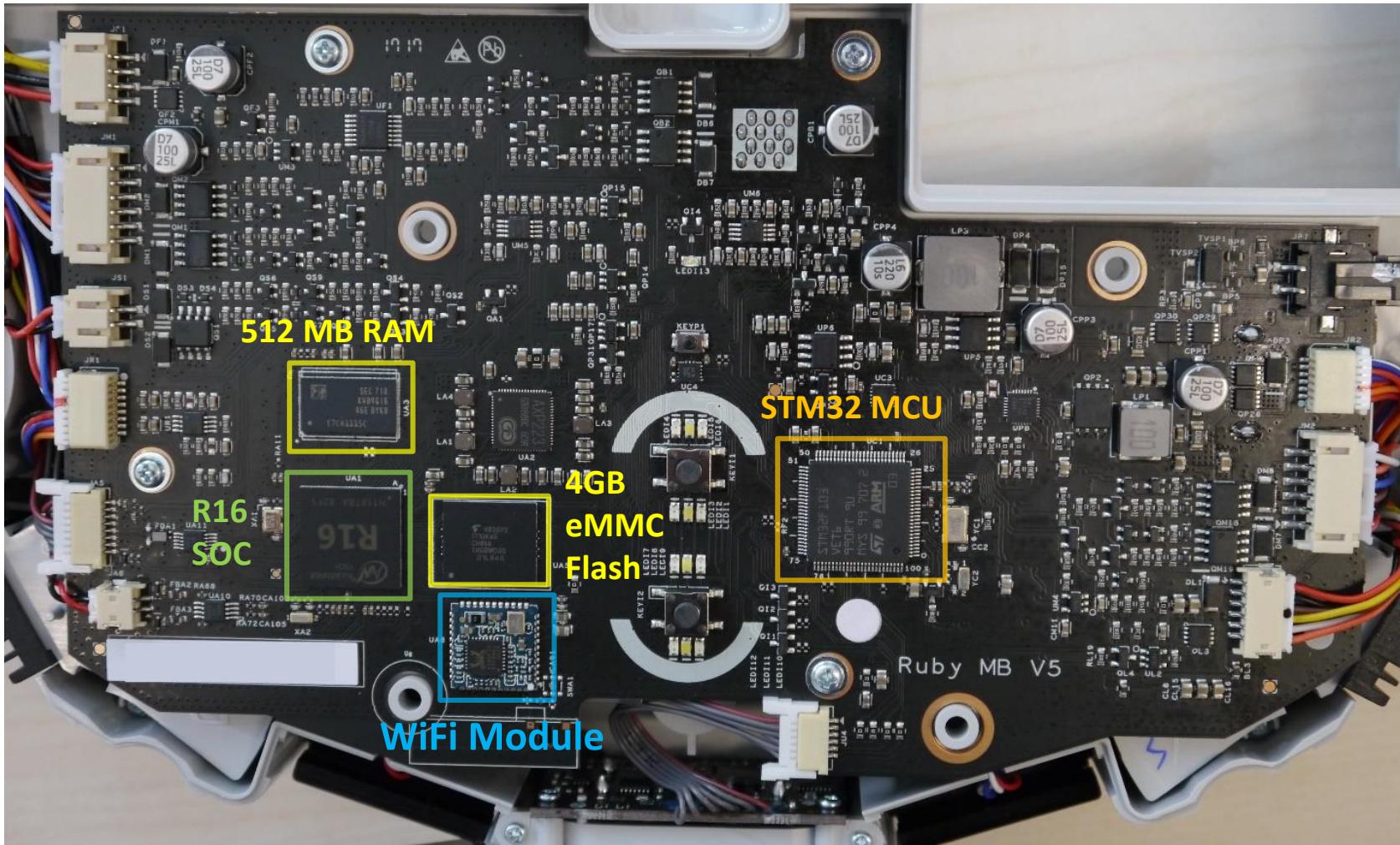
Source: Xiaomi advertisment
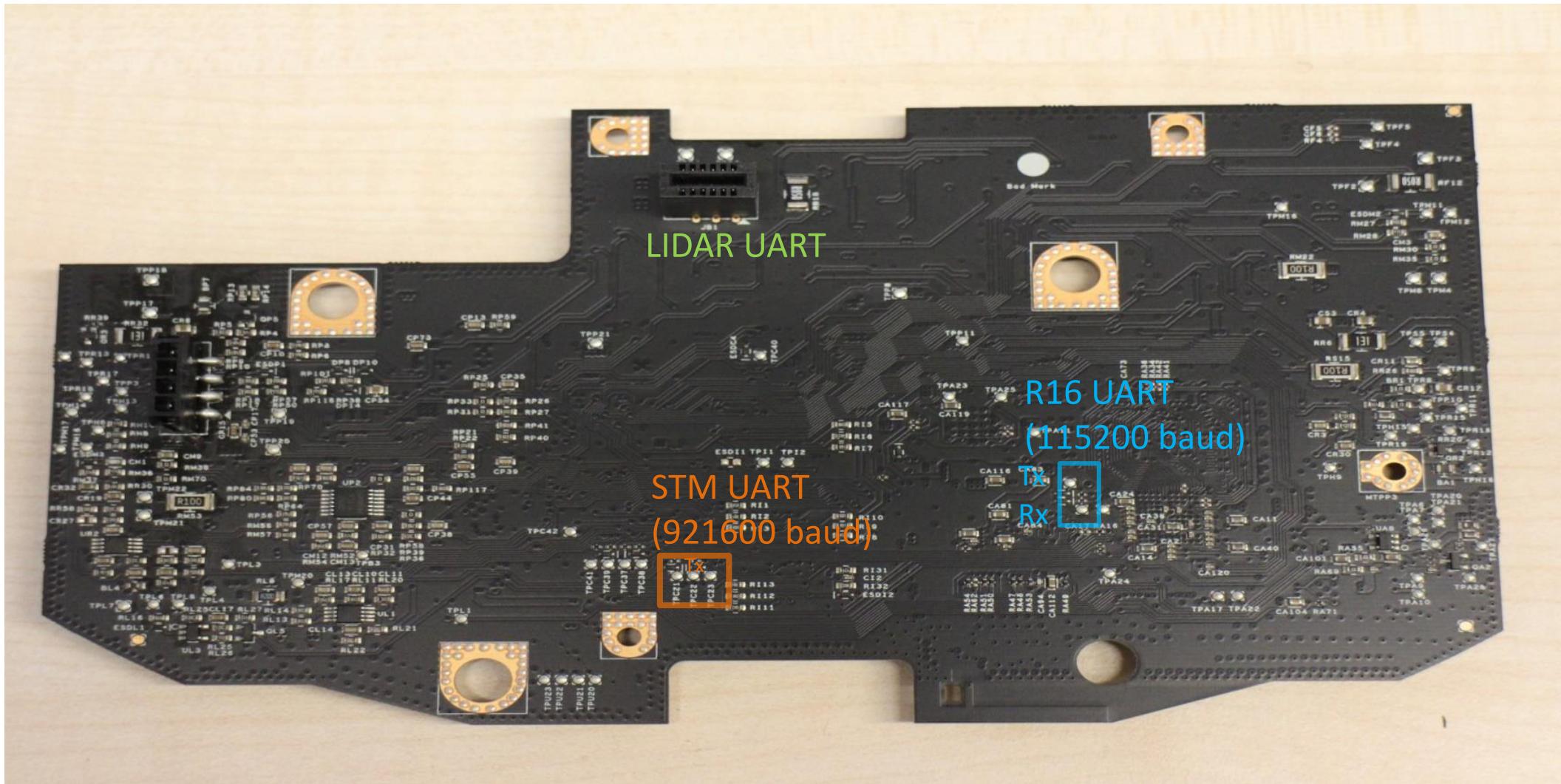
Teardown

Frontside layout mainboard

512 MB RAM

R16 SOC

4GB eMMC Flash

STM32 MCU

WiFi Module

Ruby MB V5

Backside layout mainboard

LIDAR UART

R16 UART
(115200 baud)

Tx

Rx

STM UART
(921600 baud)

Tx
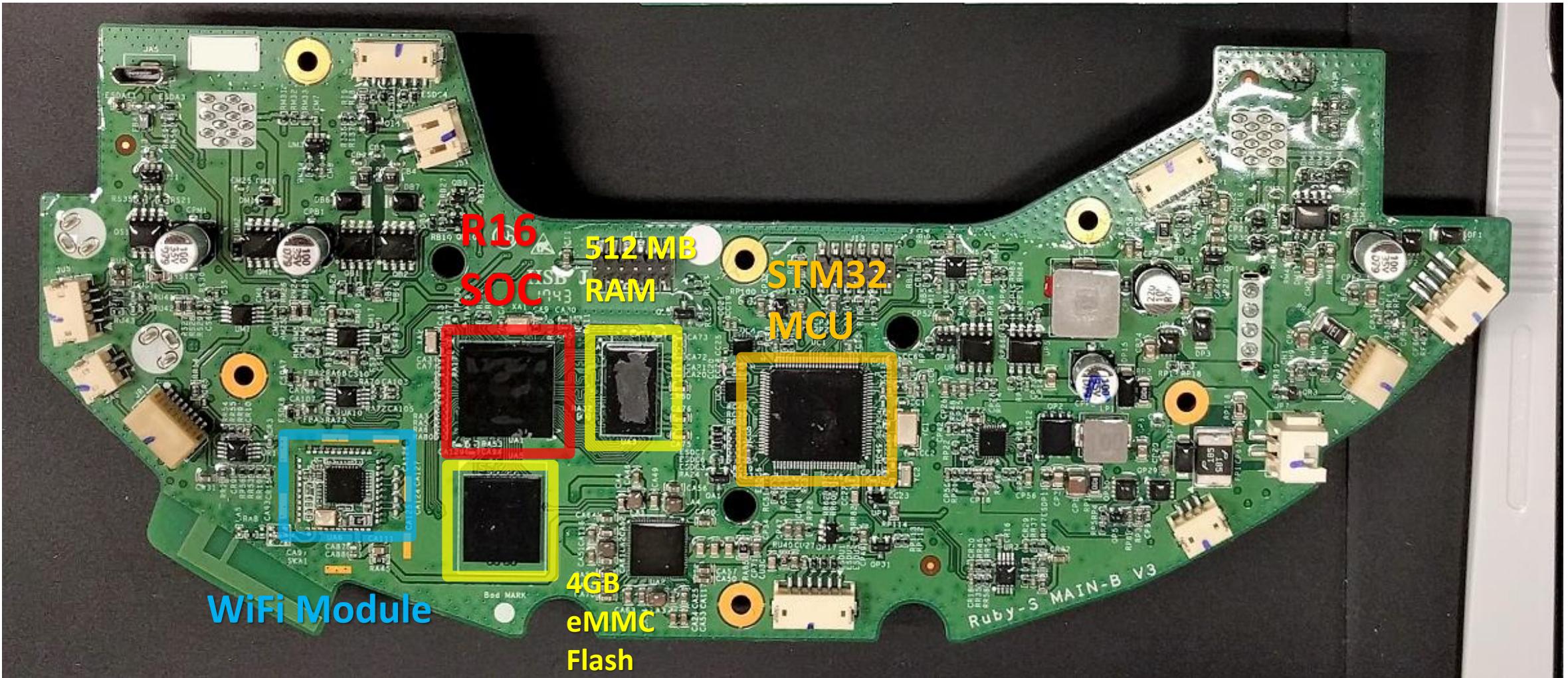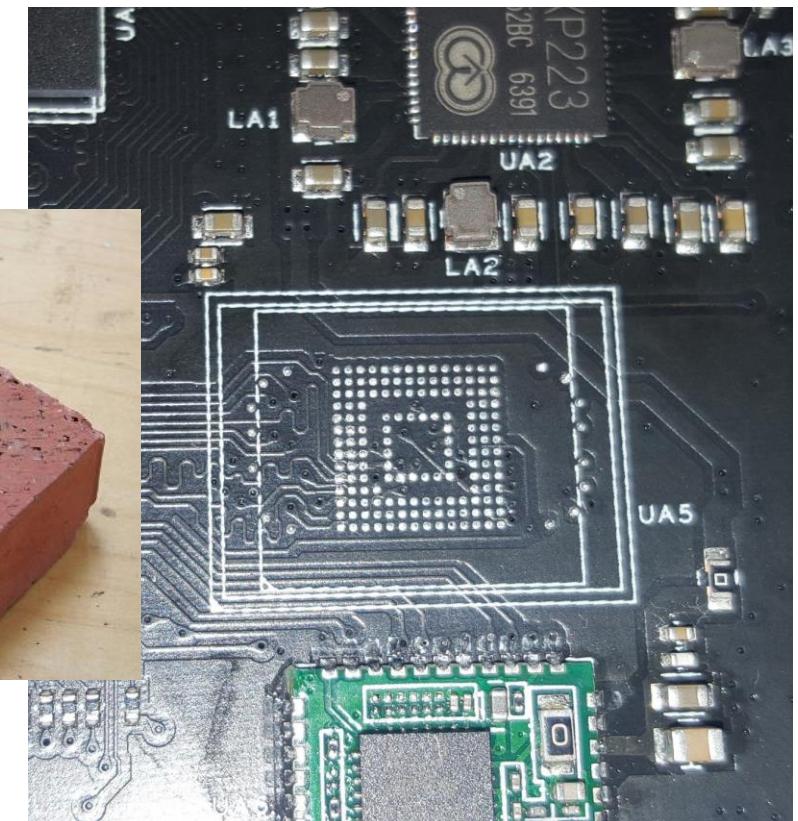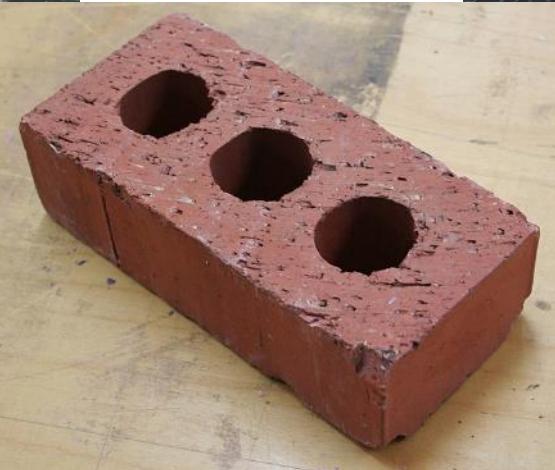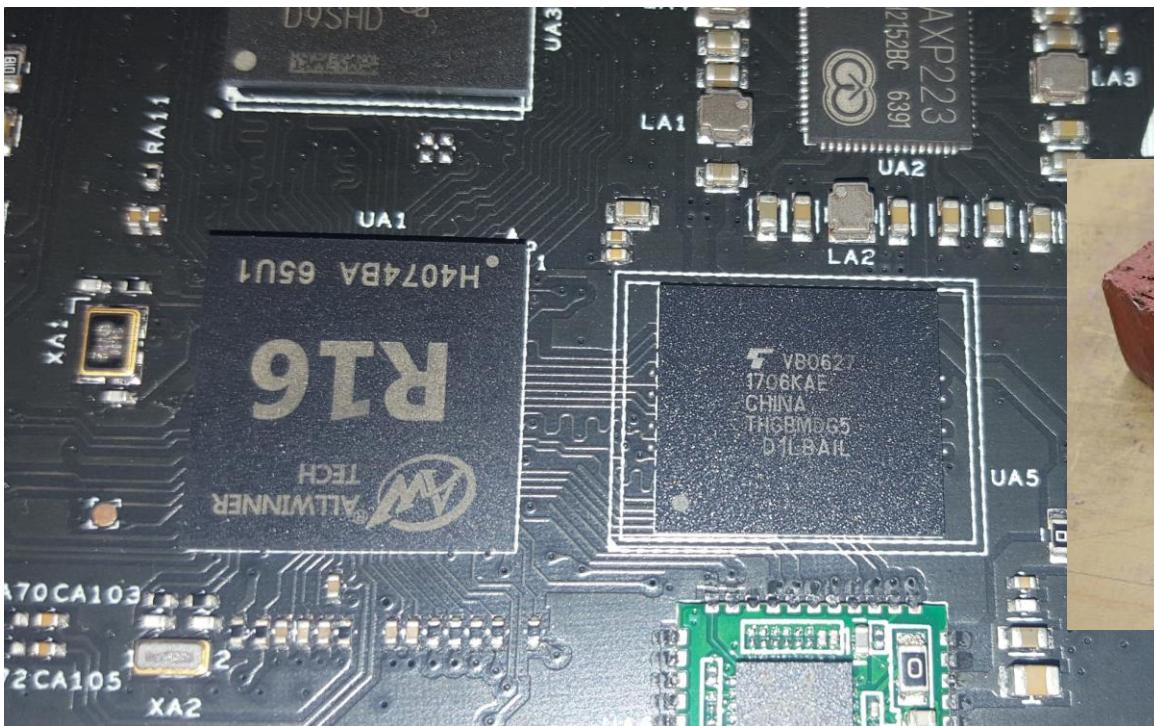
Frontside layout mainboard (Gen2)

Rooting

- Usual (possibly destructive) way to retrieve the firmware
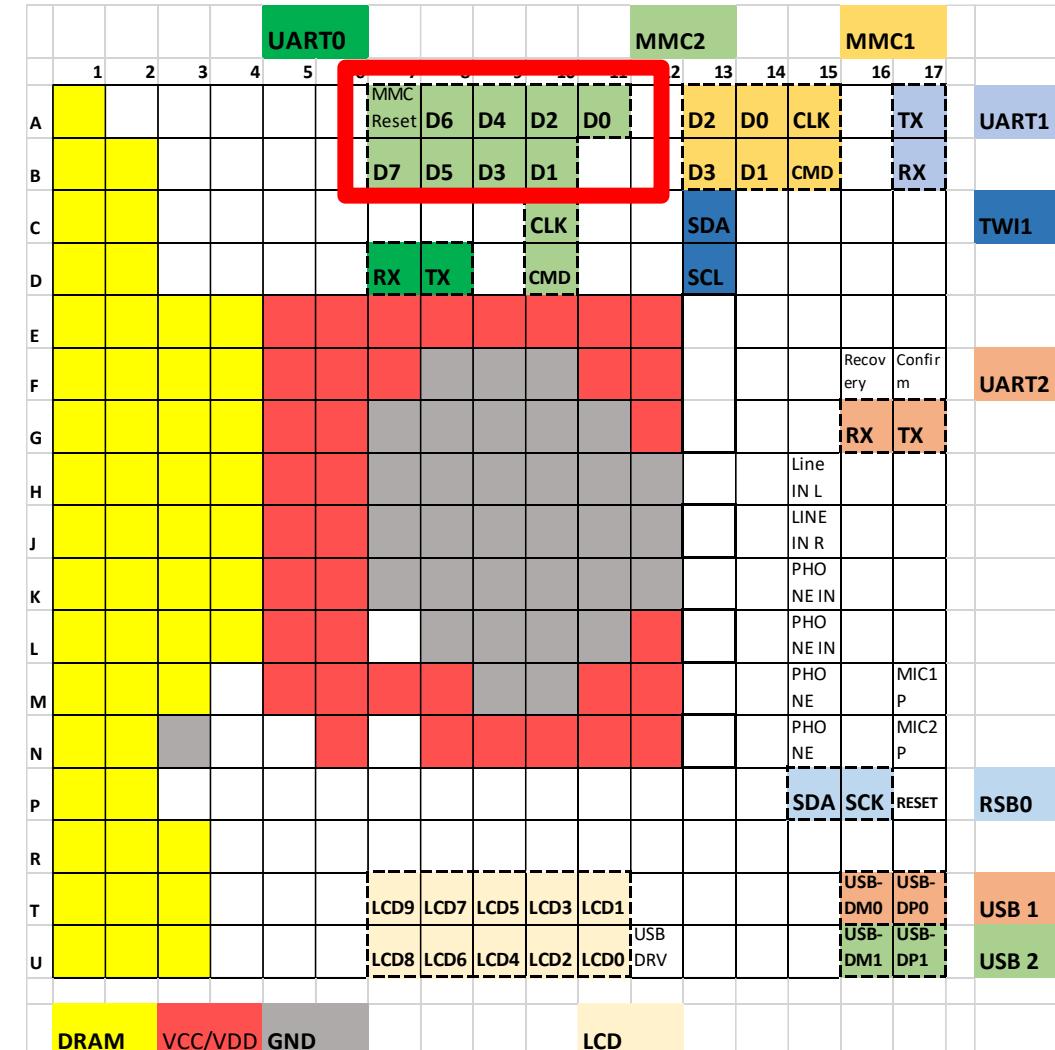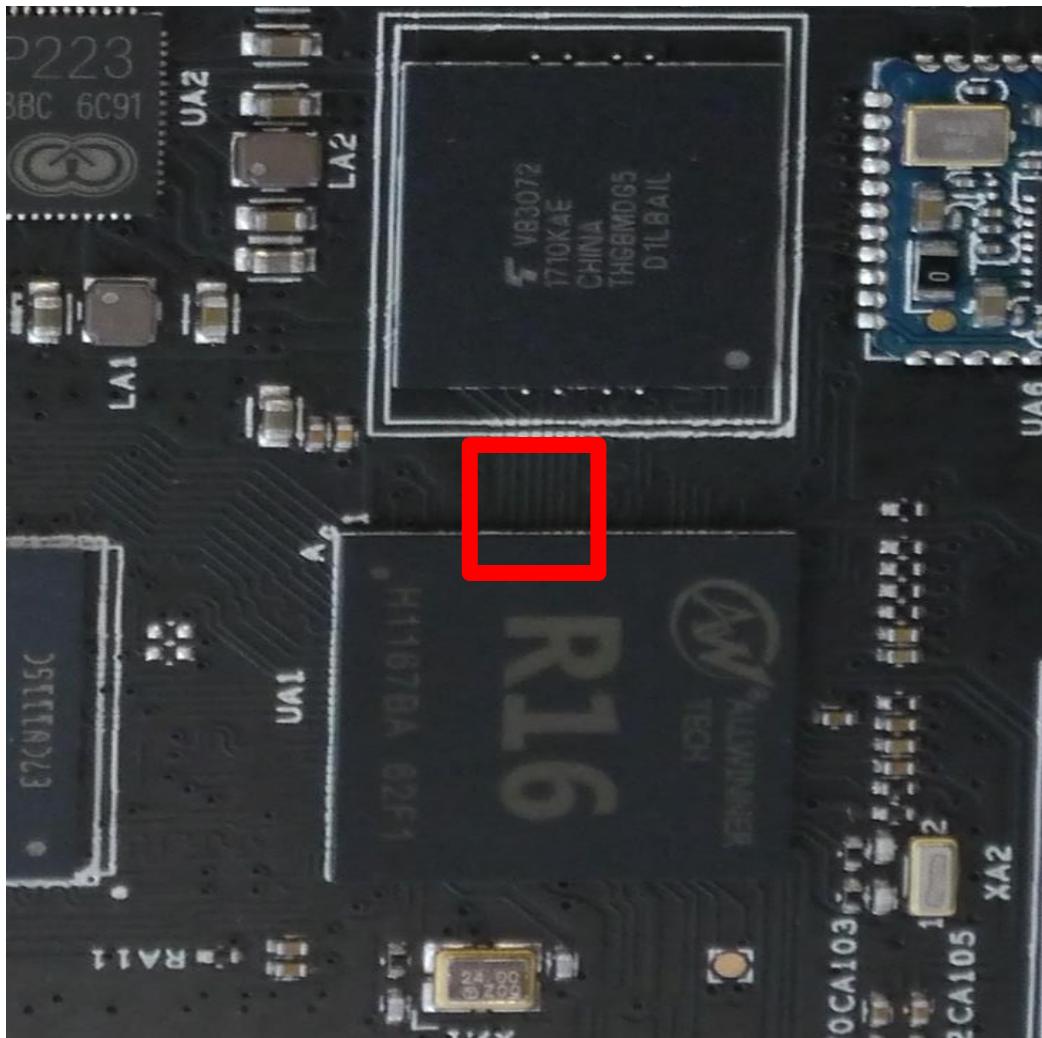
Rooting

Our weapon of choice:
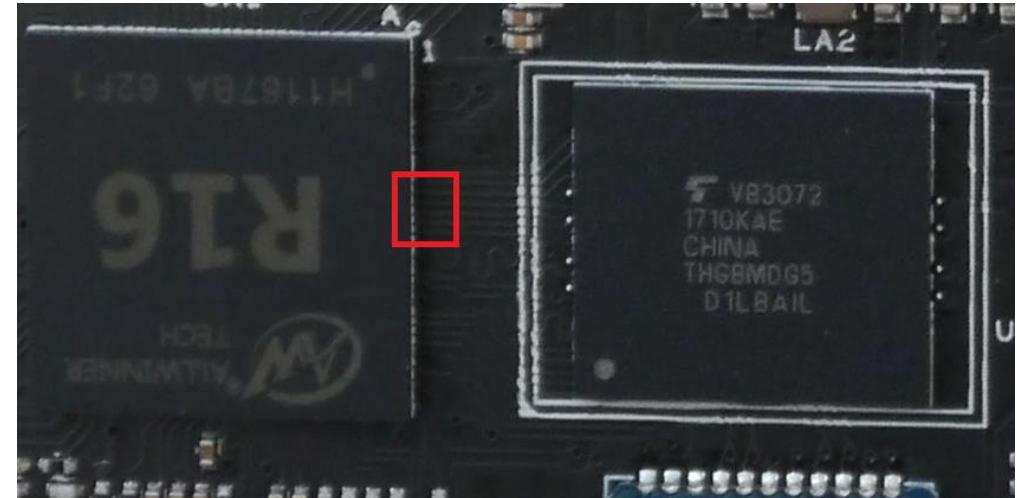
Pin Layout CPU

Rooting (Gen1 + Gen2)

- Shortcut the MMC data lines

- SoC falls back to FEL mode

- Load + Execute tool in RAM
  - Via USB connector
  - Dump MMC flash
  - Modify image
  - Rewrite image to flash

Software

- Ubuntu 14.04.3 LTS (Kernel 3.4.xxx)
    - Mostly untouched, patched on a regular base
- Player 3.10-svn
    - Open-Source Cross-platform robot device interface & server
- Proprietary software (/opt/rockrobo)
    - Custom adbd-version
- iptables firewall enabled (IPv4!)
    - Blocks Port 22 (SSHd) + Port 6665 (player)
    - Fail: IPv6 not blocked at all
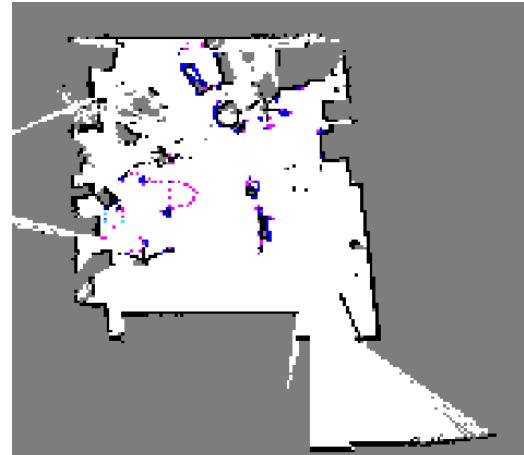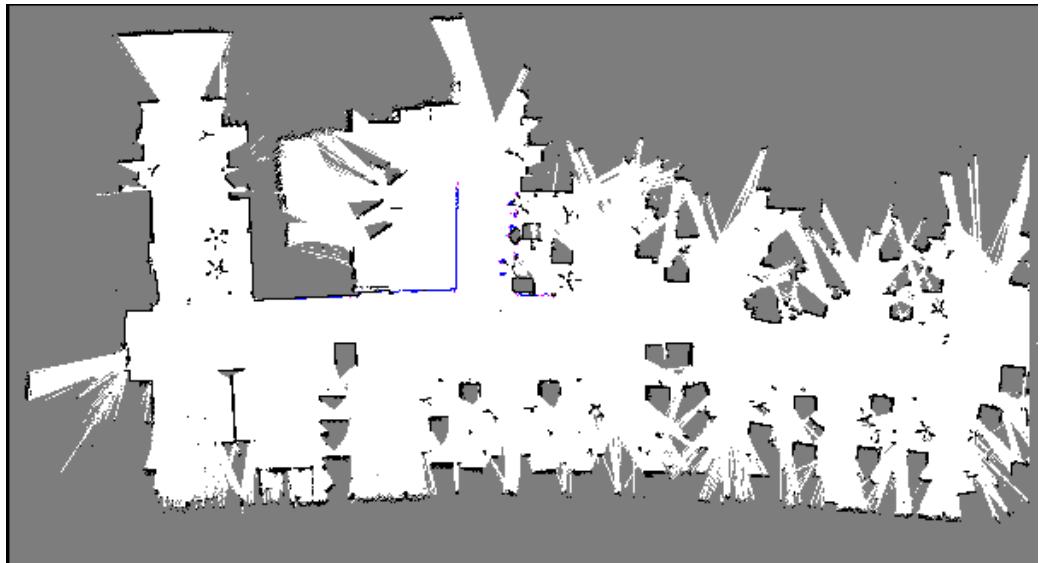
Available data on device

- Data
  - Logfiles (syslogs, stats, Wi-Fi credentials)
  - Maps
- Data is uploaded to cloud
- Factory reset
  - Does not delete data: Maps, Logs still exist

~100 Gbyte writes per Year

Available data on device
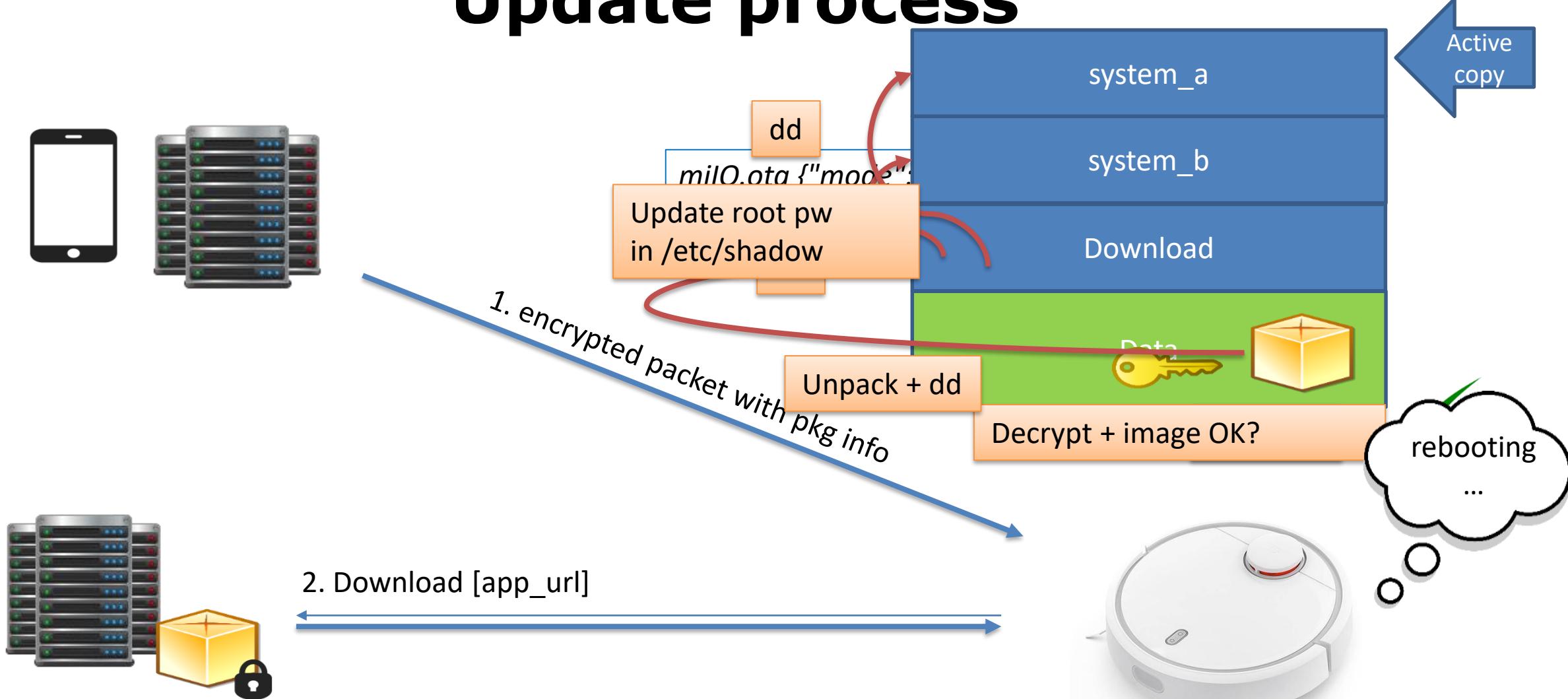
- Maps
  - Created by player
  - 1024px * 1024px
  - 1px = 5cm

eMMC Layout

Label	Content	Size in MByte
boot-res	bitmaps & some wav files	8
env	uboot cmd line	16
app	device.conf (DID, key, MAC), adb.conf, vinda	16
recovery	fallback copy of OS	512
system_a	copy of OS (active by default)	512
system_b	copy of OS (passive by default)	512
Download	temporary unpacked OS update	528
reserve	config + calibration files, blackbox.db	16
UDISK/Data	logs, maps, pcap files	~1900

Update process

system_a

system_b

Download

Active copy

dd

miIO.ota {"mode":

Update root pw in /etc/shadow

Unpack + dd

Data

Decrypt + image OK?

rebooting …

1. encrypted packet with pkg info

2. Download [app_url]

Firmware updates

- ## Integrity
  - MD5 provided by cloud

- ## Full images
  - Encrypted tar.gz archives
  - Contains disk.img with 512 Mbyte ext4-filesystem

- ## Encryption
  - Ccrypt [256-bit Rijndael encryption (AES)]
  - Static password: "<u>rockrobo</u>"

> Sound Packages
> Static password: "r0ckrobo#23456"

Lets root remotely

- Preparation: Rebuild Firmware

  – Include authorized_keys

  – Remove iptables rule for sshd

- Send „miIO.ota" command to vacuum

  – Encrypted with token

    - From app or unprovisioned state

  – Pointing to own http server

Lets root remotely

„Get Token"

„miIO.ota"

unprovisioned state

Webserver

SSH

Possible Countermeasures

- Changing the firmware key
  - Useless -> we will figure out ;)
- Encrypting/Obfuscating the log-files and maps
  - Recently introduced
  - Here is the AES128CBC-key: "RoCKR0B0@BEIJING"

Copyright: 20th Century Fox

How to get the log and map AES key?

- RRlogd uses AES encryption functions from OpenSSL library
  - Imported as dynamic library
  - Interesting function: EVP_EncryptInit_ex(…)
- Helpful tool: ltrace
  - Intercepts library calls
  - Shows contents arguments of function calls

Persistance

- Patch the recovery partition
  - Replace custom adbd with open source one
  - disable firewall
- Disable updates
  - Kill SysUpdate process
  - Disable Ccrypt
- Extract credentials
  - Content of "vinda.conf" = root password (XOR 0x37)
  - DID, cloud key

recovery	fallback copy of OS
system_a	copy of OS (active by default)
system_b	copy of OS (passive by default)
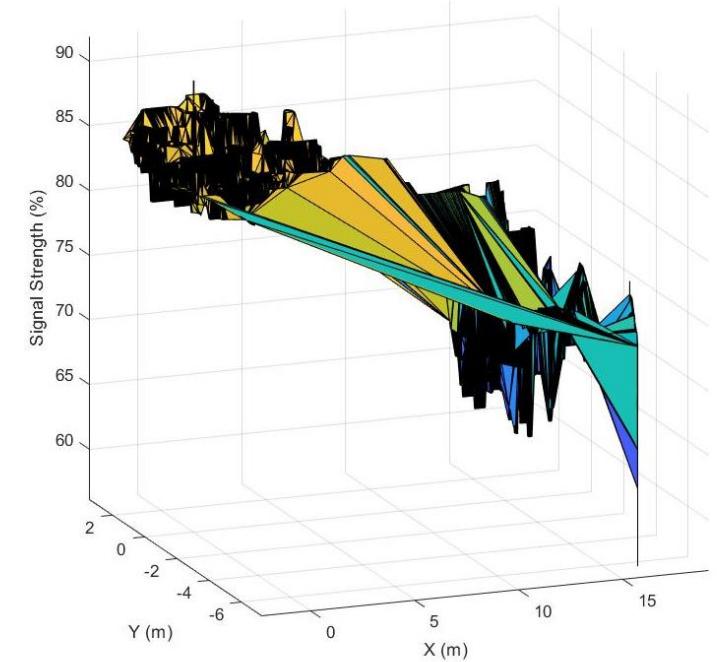
Side note about Entropy

- Recap: Token is AES256 key
  - Method used for Generation:
    - Initialization: srand(seed)
   - seed has 2^{31} states
    - 16 times rand()

Research in cooperation with Jan Ruge

Summary of the Vacuum

- Rooting
  - **Remote**! (No „foil attack" required anymore)
- Cloud Connection
  - Run **without** cloud
    - Support by third-party tools (e.g. FloleVac, FHEM, etc)
  - Run with your **own** cloud

https://xiaomi.flole.de/
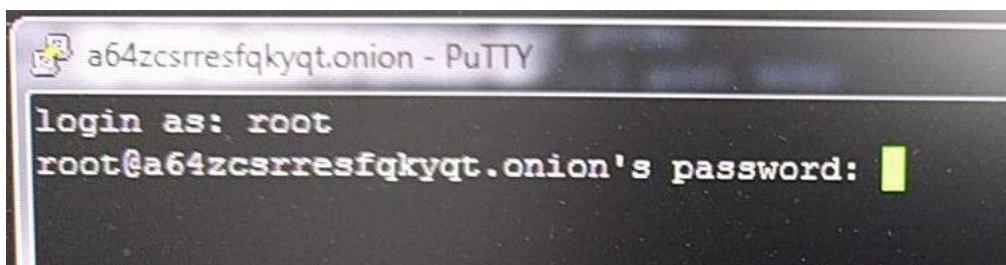
Signal Strength of Wifi

HAVING FUN IN HACKING

Connection to the Dark Side

- Idea by Prof. Noubir: Let's run Tor hidden services on IoT

  - Paper from 2015: OnionBots, a stealthy botnet with compromised IoT devices

- Easy to install in Ubuntu

  - Make SSH accessible via TOR

  - No need for NAT ;)

OnionBots: Subverting Privacy Infrastructure
for Cyber Attacks
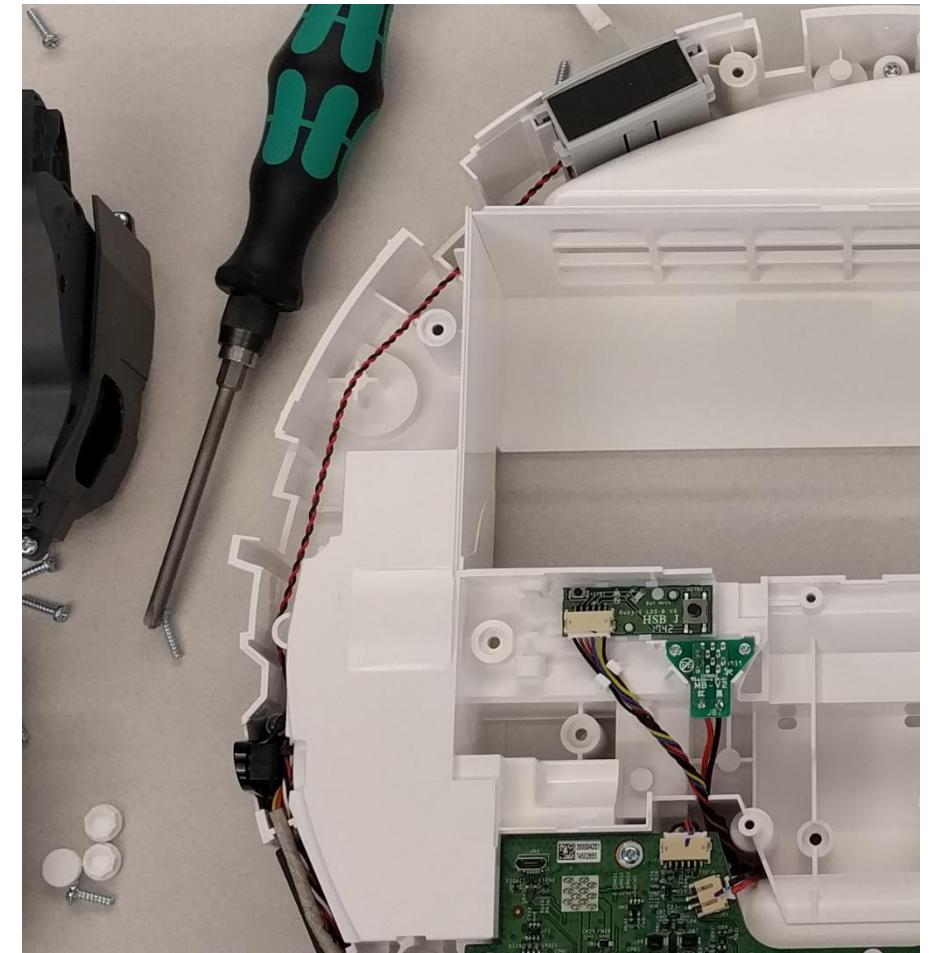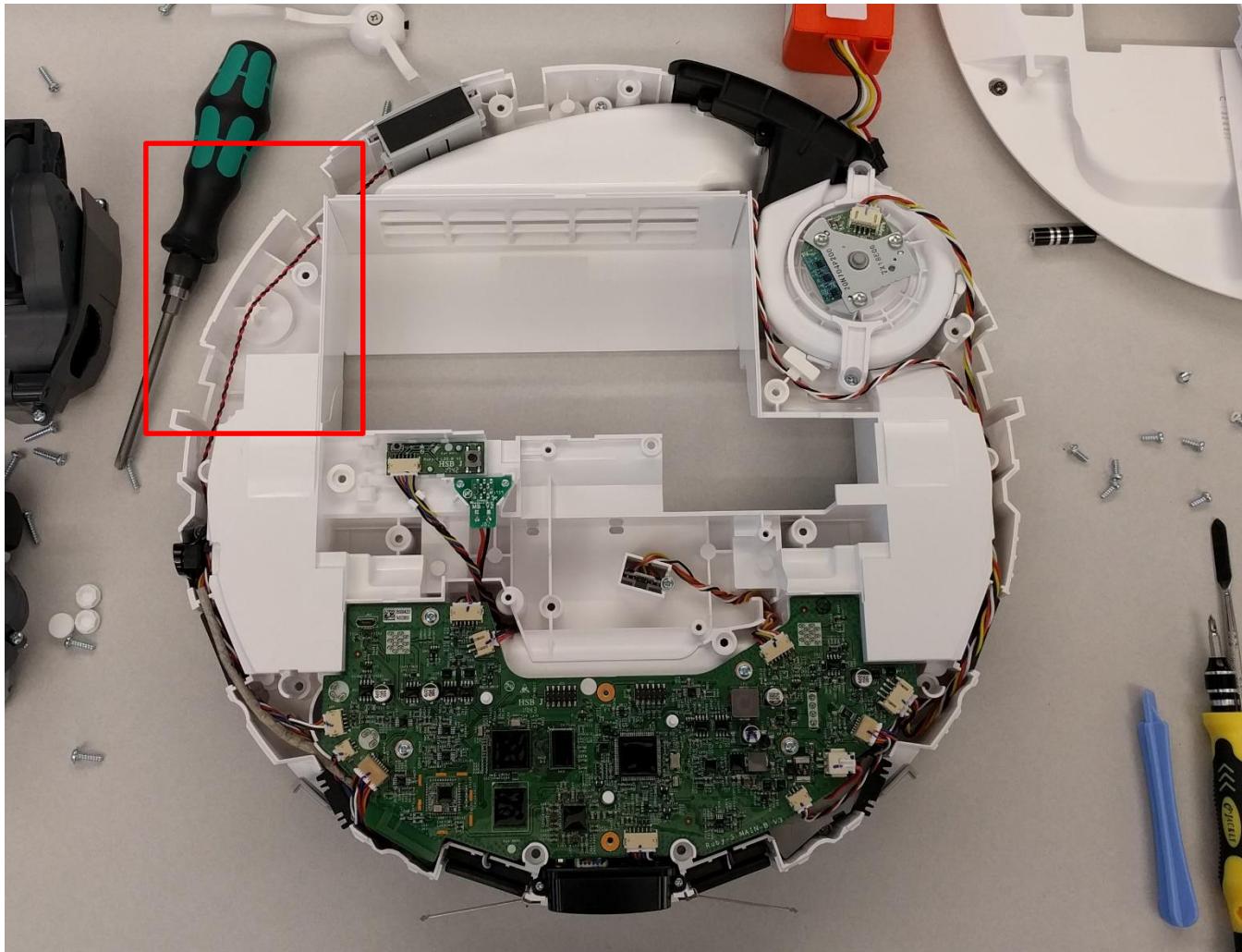
Amirali Sanatinia
Northeastern University
amirali@ccs.neu.edu

Guevara Noubir
Northeastern University
noubir@ccs.neu.edu

Abstract—Over the last decade botnets survived by adopting a sequence of increasingly sophisticated strategies to evade detection and take overs, and to monetize their infrastructure. At the same time, the success of privacy infrastructures such
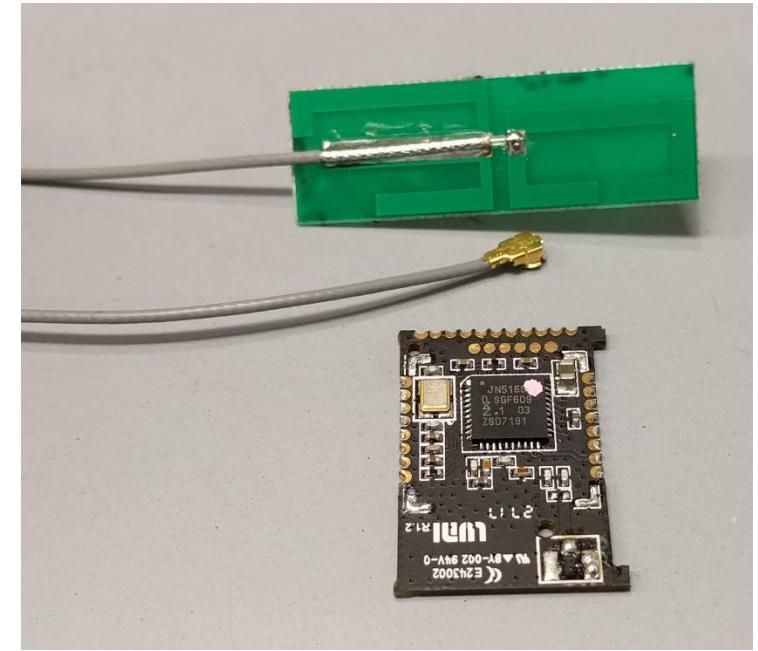
level of this arm-race. We contend that the next wave of botnets' sophistication will rely on subverting privacy infrastructure and a non-trivial use of cryptographic mechanisms. The

```
a64zcsrresfqkyqt.onion - PuTTY

login as: root
root@a64zcsrresfqkyqt.onion's password: 
```

https://arxiv.org/pdf/1501.03378.pdf

Using empty space

Using empty space
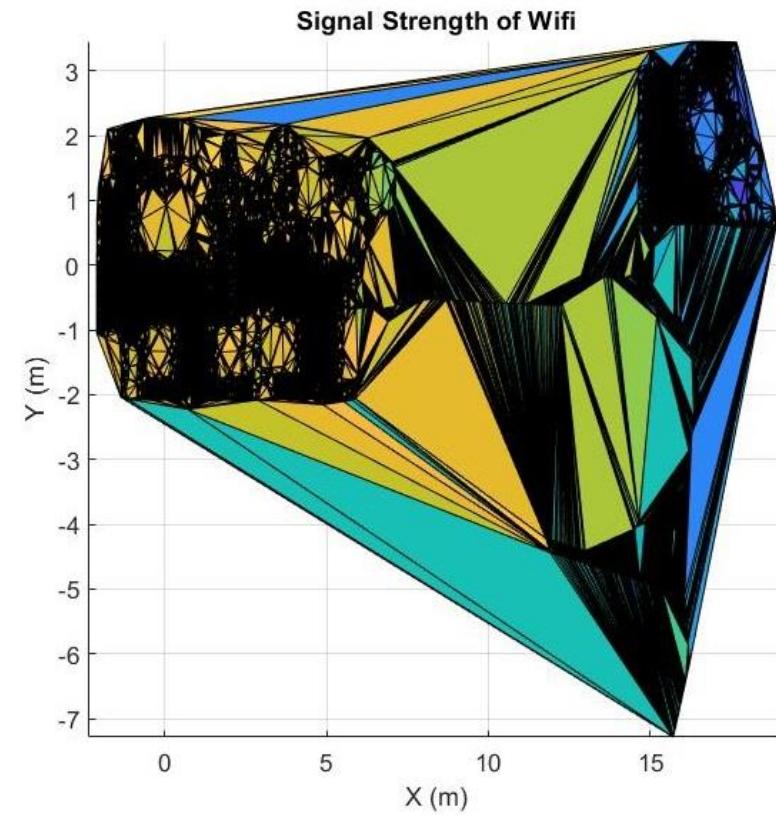
- Zigbee module fits in vacuum
  - Use serial connection
  - ARM binaries of Gateway run natively
  - Result: Zombie-Gateway-Vacuum
- USB stick
 - More space: mobile Data storage
 - Soldered to MicroUSB port

Mobile Wi-Fi mapper
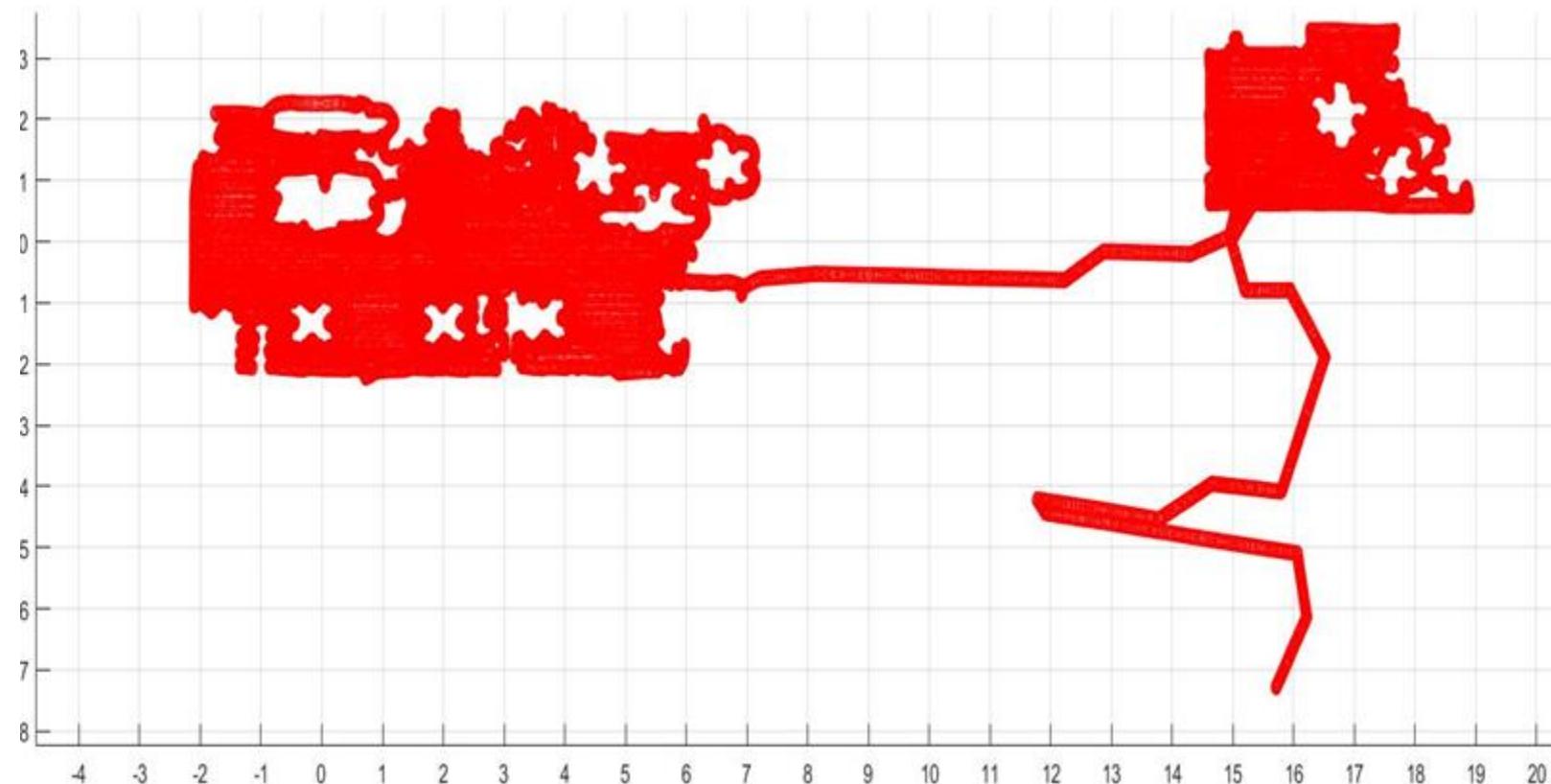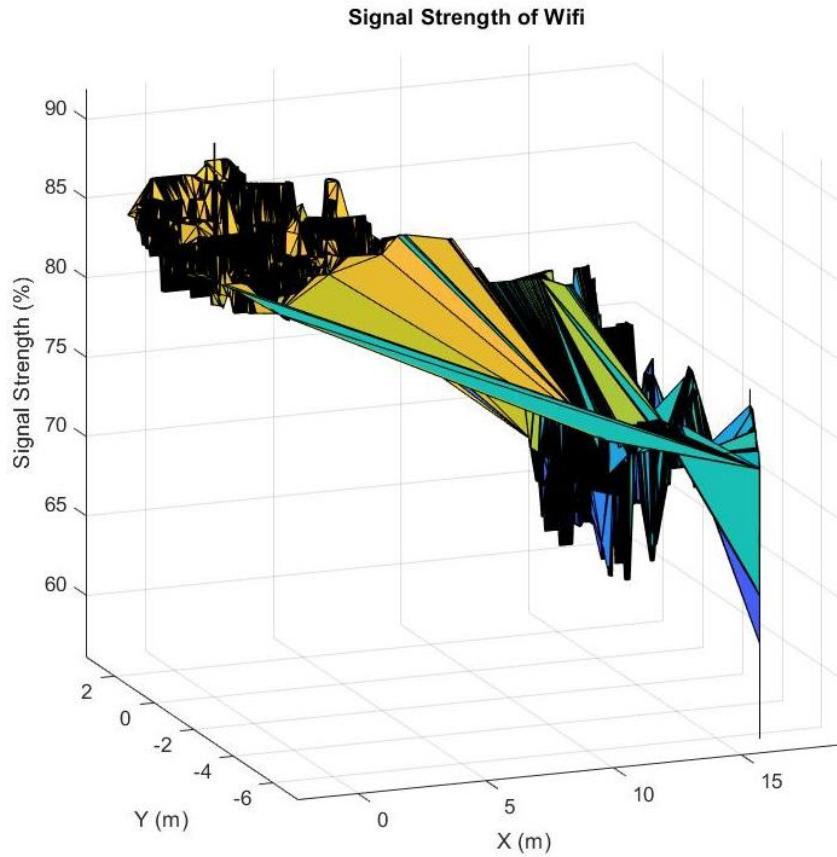
- Idea:
  - Parsing of position2d from player logfile

    {x_pos, y_pos, yaw_pos, x_vel, y_vel, yaw_vel}

  - Retrieving WiFi information from Linux kernel

    {link, level, noice, SSID, BSSID)

- Developed with Andrew Tu @HackBeanpot 2018, Boston

Mobile Wi-Fi mapper

Genuine + Jack Morton Office, NE Side, 5th floor

Mobile Wi-Fi mapper

Signal Strength of Wifi
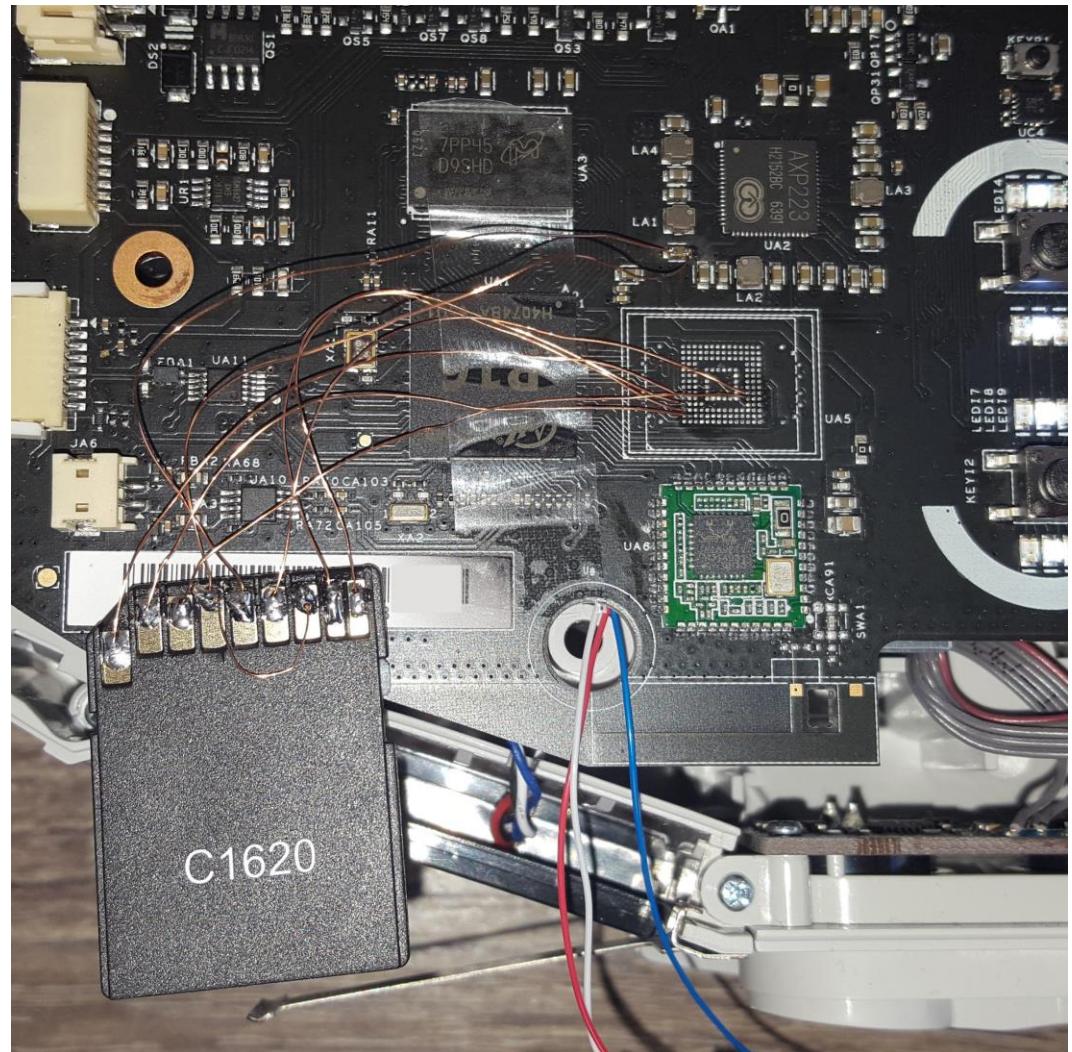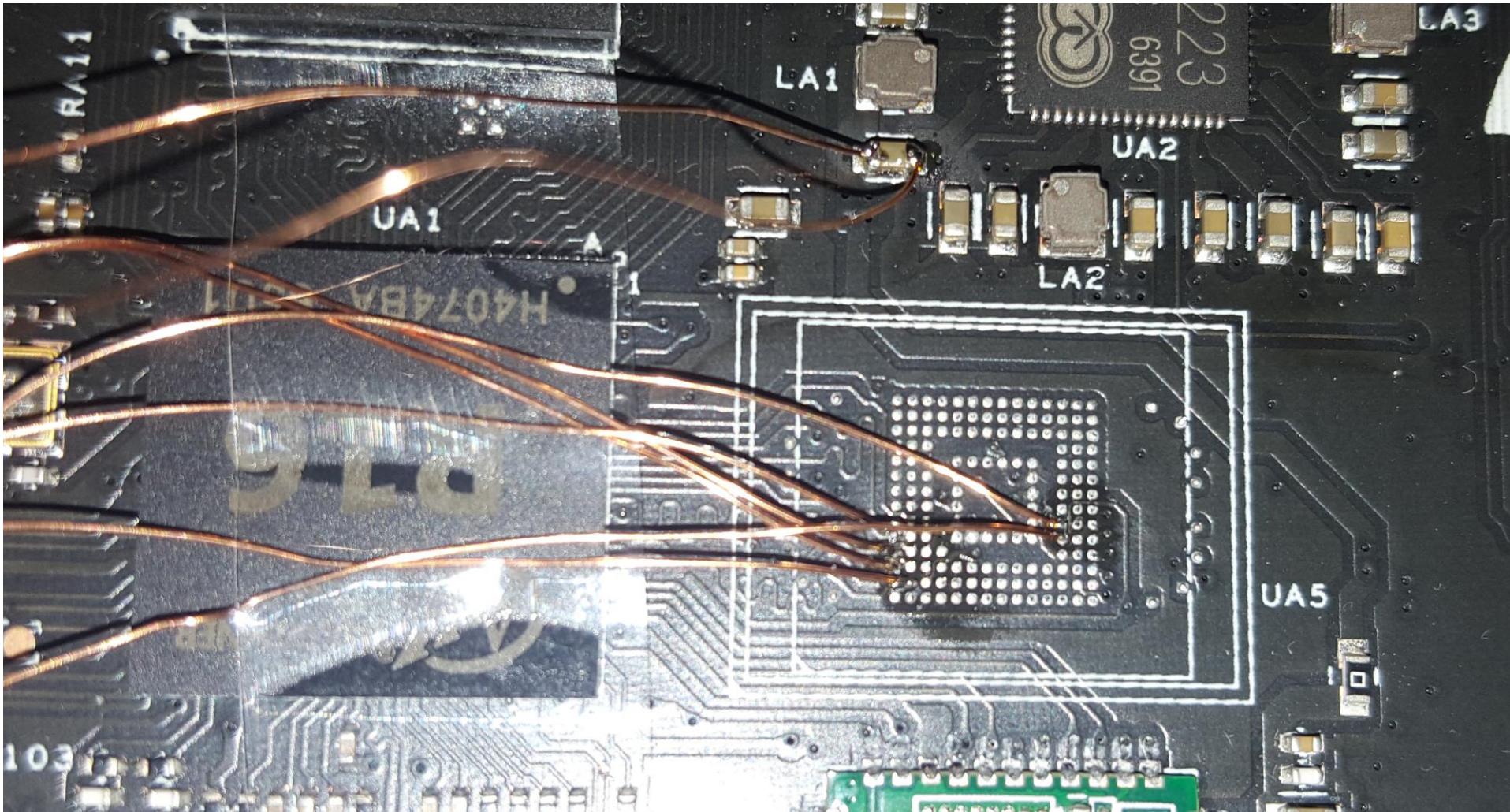
If in need of additional space

- Done by Dustcloud user

- Reason: broken MMC-Chip

- Not recommended for everyone ;)


```
[mmc]:  -------------------------mmc->clock 50000000--
[mmc]:  ------------------------mmc->bus width 4-----
[mmc]:  SD/MMC Card: 4bit, capacity: 7600MB
[mmc]:  boot0 capacity: 0KB,boot1 capacity:
[mmc]:  ***************SD/MMC 2 init OK!!!******
```

If in need of additional space

IoT chatting with IoT

One word of warning...

- Never leave your devices unprovisioned
  - Someone else can provision it for you
    - Install malicious firmware
- Be careful with used devices
  - e.g. Amazon Marketplace, Ebay, etc.
  - Some malicious software may be installed
- Never install rooted firmware from untrusted sources !!!!
  - Expecially not from russian forums!

Conclusion

- Basic best practices not used
  - firmware signatures ☹
  - HTTPS, certificate verification ☹
  - Hardware security features ☹

- Good
  - We can modify the devices

- Bad
  - Someone else can do too

Acknowledgements

- Daniel Wegemer (aka DanielAW)

- Prof. Guevara Noubir (CCIS, Northeastern University)
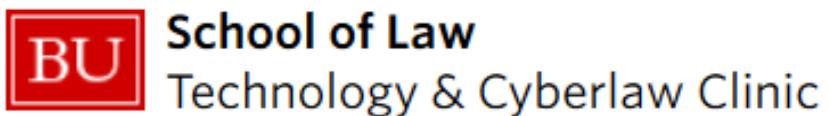
- Secure Mobile Networking (SEEMOO) Labs and CROSSING S1

- Andrew Sellars and Team (Boston University Technology & Cyberlaw Clinic)

Questions?

Meet me at the IoT Village here at Defcon

Contact:

See: http://dontvacuum.me
Telegram: https://t.me/kuchenmonster
Twitter: dgi_DE
Meet me in Boston/@DC617