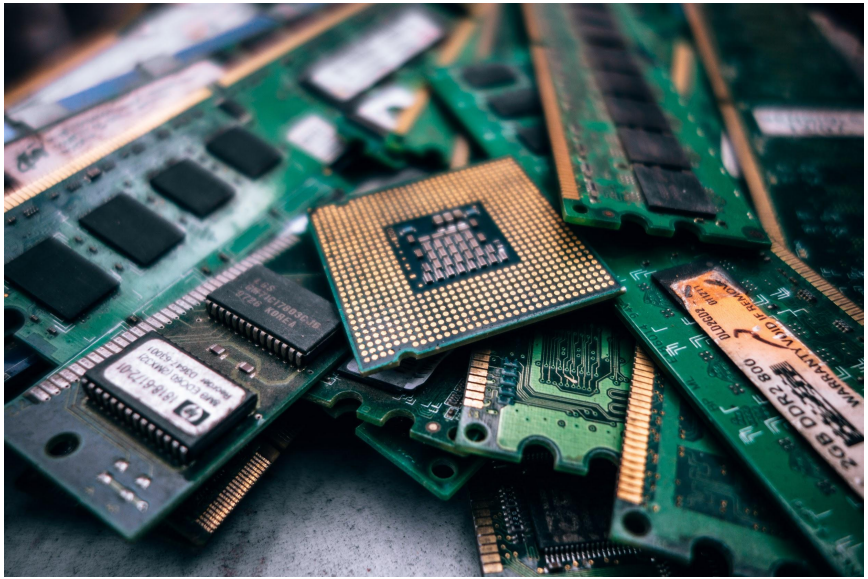
IOT Development from Prototype to Production

Drew Moseley
Technical Solutions Architect
Mender.io

Session overview

- Define IOT and markets
- Selecting hardware.
- Selecting system software.
- Design considerations for IOT development



About me

- Drew Moseley
  - 10 years in Embedded Linux/Yocto development.
  - More than that in general Embedded Software.
  - Project Lead and Solutions Architect.

  - drew.moseley@mender.io
  - https://twitter.com/drewmoseley
  - https://www.linkedin.com/in/drewmoseley/
  - https://twitter.com/mender_io

- Mender.io
  - Over-the-air updater for Embedded Linux
  - Open source (Apache License, v2)
  - Dual A/B rootfs layout (client)
  - Remote deployment management (server)
  - Under active development

IOT Definition

- “A network of internet-connected objects able to collect and exchange data using embedded sensors.”[1]
- IEEE (86 page PDF)[2]
- A “network of physical devices, vehicles, home appliances and other items embedded with electronics, software, sensors, actuators, and connectivity which enables these objects to connect and exchange data.”[3]
- Key characteristics:
  - Connected
  - Sensors
  - Actuators
  - Cloud Infrastructure

[1] http://www.businessinsider.com/what-is-the-internet-of-things-definition-2016-8

[2] https://iot.ieee.org/definition.html

[3] https://en.wikipedia.org/wiki/Internet_of_things

IOT Applications

- Consumer[1]
  - Nest thermostat
  - Smart lighting
  - Home security
  - Connected automobiles
- Industrial
  - Operations Centers
  - Factory/inventory management
- Enterprise
  - Supply chain management
  - Medical Device
  - Business Operations
- Municipal
  - Infrastructure monitoring/management
  - Traffic control
  - Public Transit

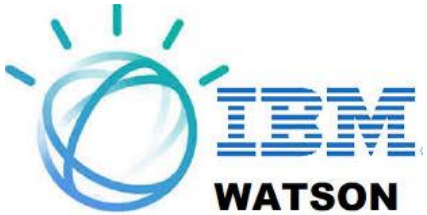
[1]Not an endorsement; I've not even used most of these examples

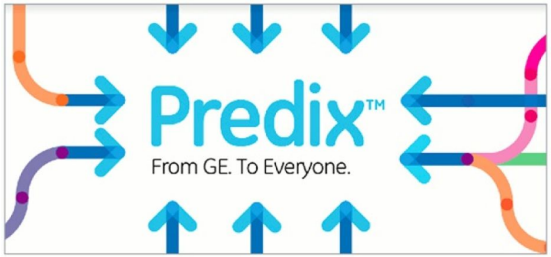
Cloud Infrastructure

Used for device control and data store.

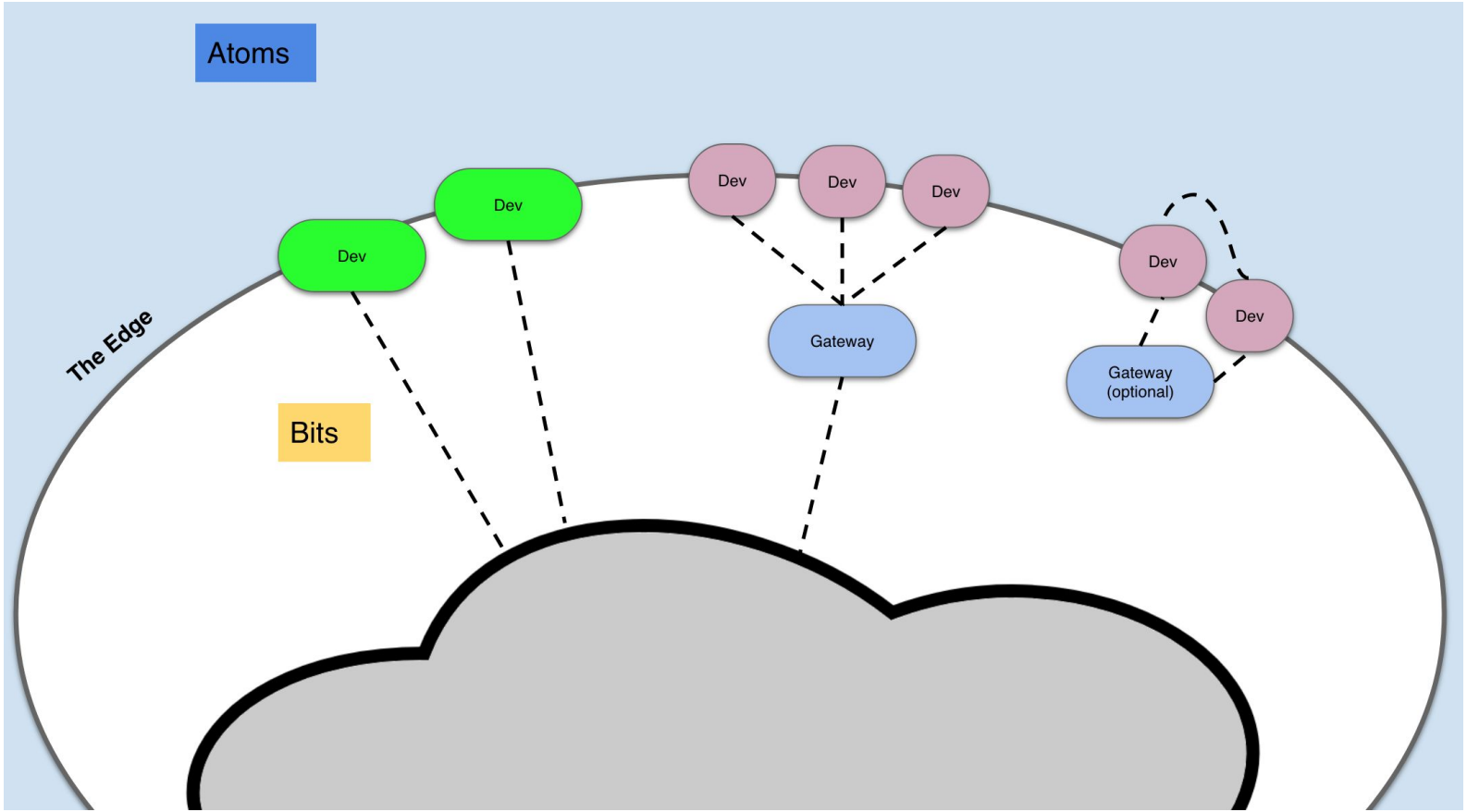
May provide AI and big data services.

May provide device fleet management/dashboard.



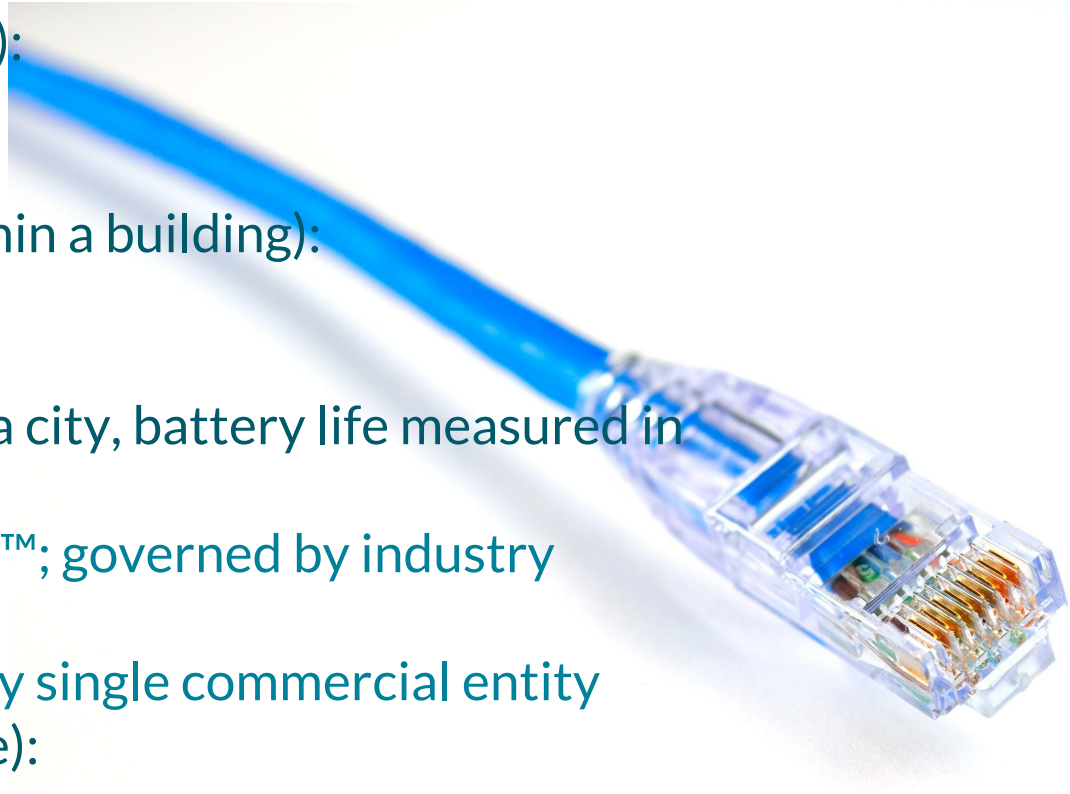


IOT Network Architecture



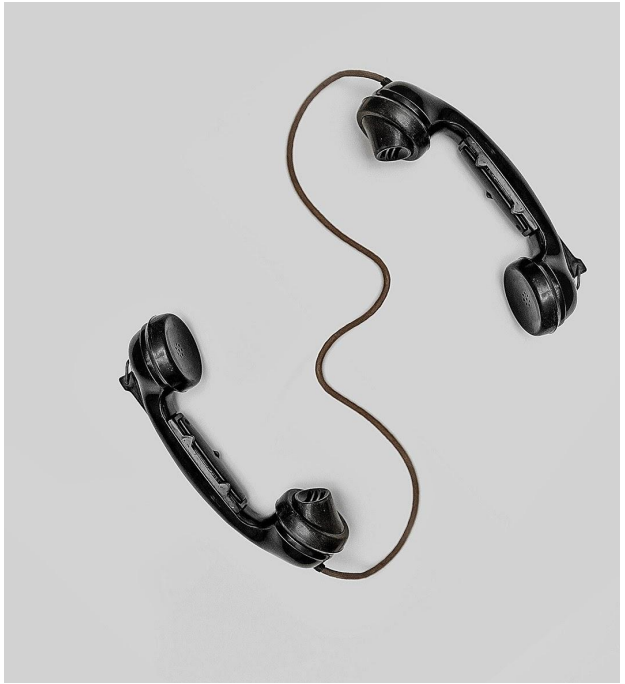
IOT Connectivity Options

- Short Distance (inches):
  - NFC
  - Bluetooth
- Medium Distance (within a building):
  - Wi-Fi
  - Ethernet
- Long Distance (within a city, battery life measured in years):
  - LoRa®/LoRaWAN™; governed by industry alliance
  - Sigfox; governed by single commercial entity
- Wide Area (nationwide):
  - Cellular/LTE



IOT Communication Protocols

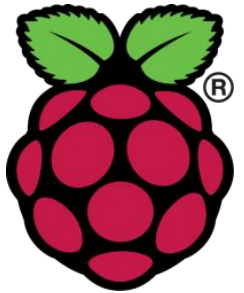
- HTTP/HTTPS REST APIs
- 6LoWPAN
  - IPv6 over LP-WAN protocols
- MQTT
  - Pub/Sub model
  - Lightweight in both code and bandwidth
  - OASIS Standard[1]
- ZeroMQ
  - Pub/Sub, Push/Pull, Router/Dealer
  - Open source (LGPL with a Static Linking Exception)
- Zigbee
  - Primarily for Home Automation
  - IEEE 802.15.4
- DDS (Data Distribution Service)
  - Global Data Space
  - Distributed with access controls



[1]https://www.oasis-open.org/news/announcements/mqtt-version-3-1-1-becomes-an-oasis-standard

Hardware Criteria

- MCU vs SOC
  - MCU generally not Linux
- On-board peripherals
- Hobbyist vs Commercial Vendor
  - Lead times
  - Inventories
- Battery vs Hard-wired
- Price
- Form factor:
  - Board (Beaglebone Black, Raspberry Pi 3)
  - Module (Toradex SOM, Raspberry Pi Compute Module)

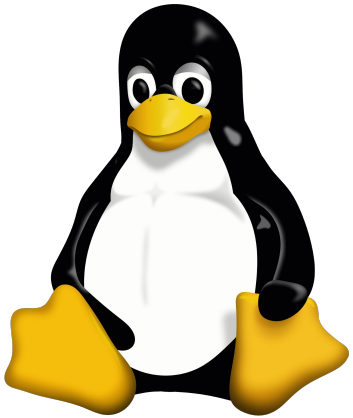


ARM mbed

Toradex

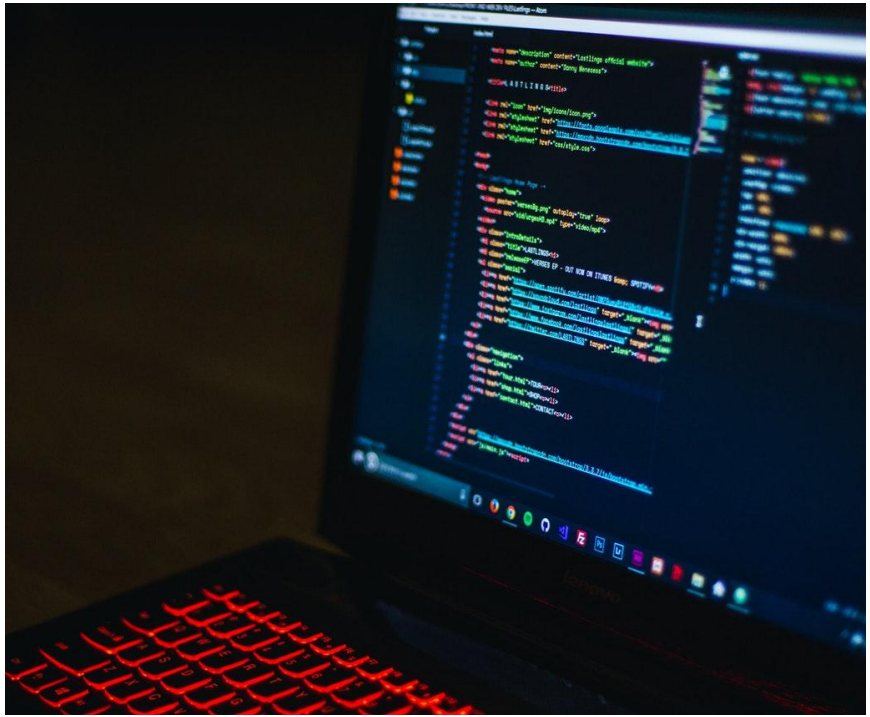
System Software Criteria

- OS vs RTOS vs Bare Metal
- System Development Tools
  - Yocto
  - Buildroot
  - OpenWRT
  - Debian
- Deployment Strategies
  - Hypervisors/Containers
  - AMP
- Security/Safety
  - ISO 26262
  - SELinux
  - AppArmor
  - SMACK: Simplified Mandatory Access Control Kernel



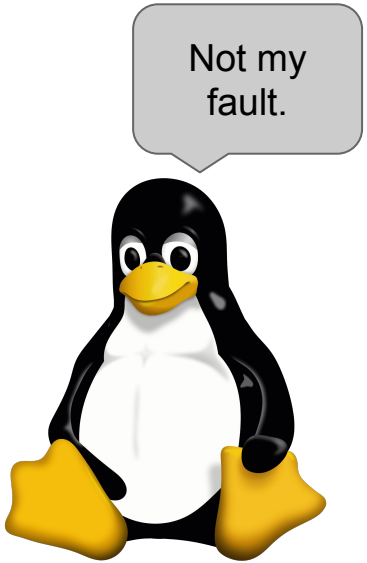
Application Software Criteria

- Application Development Frameworks
  - NodeRED
  - NodeJS
  - Eclipse Kura
  - Qt
- Application Development Environments
  - Eclipse
  - CLI
  - Commercial vs RYO/OSS
- Language Availability
  - C/C++/Python/Java/Javascript/ Golang
- 3rd party package availability



System Software Options - Non-Linux

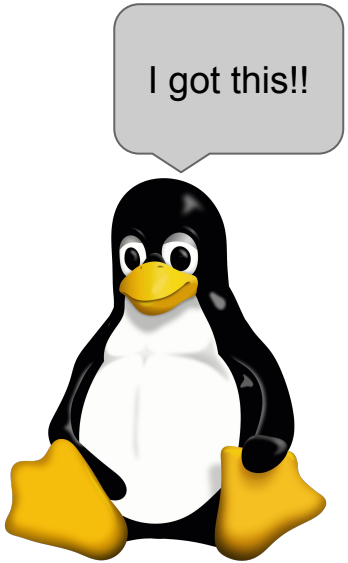
- Bare Metal/Embedded Control Loop
- Embedded RTOS[1]
  - OSS: FreeRTOS, IncludeOS, Apache Mynewt, Zephyr
  - Commercial: Nucleus, vxWorks, QNX
- “Desktop” class OS
  - Windows IOT Core



[1] https://en.wikipedia.org/wiki/Comparison_of_real-time_operating_systems

System Software Options - Linux

- Embedded Linux Options[1]
  - Desktop Class Distro
    - Direct Install
    - Packaging scripts
  - Embedded Distro Builder
    - Yocto
    - Buildroot
    - OpenWRT
  - Hybrid
    - ISAR
    - ELBE



[1] https://en.wikipedia.org/wiki/Linux_on_embedded_systems

Yocto Project - Overview

"It's not an embedded Linux distribution -- it creates a custom one for you"[1]

- Recipes, metadata, dependencies and configuration
- Primary output: package feed
- Secondary output: boot images
- Builds all components from source
- Mechanism, not policy

Products:

- Root filesystem image
- Kernel, Bootloader, Toolchain
- Package Feed

[1]See more at https://www.yoctoproject.org

Buildroot - Overview

"Buildroot is a simple, efficient and easy-to-use tool to generate embedded Linux systems through cross-compilation."[1]

- Primary output: boot images
- Does not support rpm-style package mgmt
- "Firmware Generator"
- Builds all components from source
- Focus on simplicity

Products:

- Root filesystem image
- Kernel, Bootloader, Toolchain

[1]See more at https://buildroot.org/

OpenWRT - Overview

"OpenWrt provides a fully writable filesystem with package management."[1]

Primary focus is networking

- Replacement firmware for consumer devices
- Primarily a binary distribution
- On-device package management

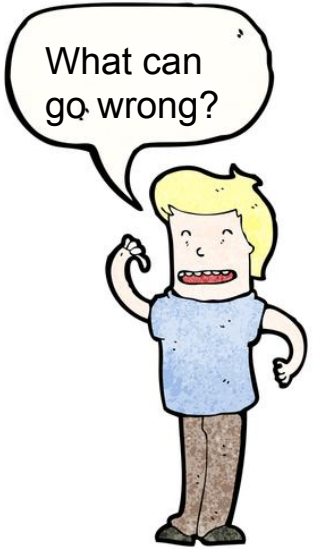
Products:

- Firmware image in device-specific format
- Network available package repositories

[1]See more at https://openwrt.org/

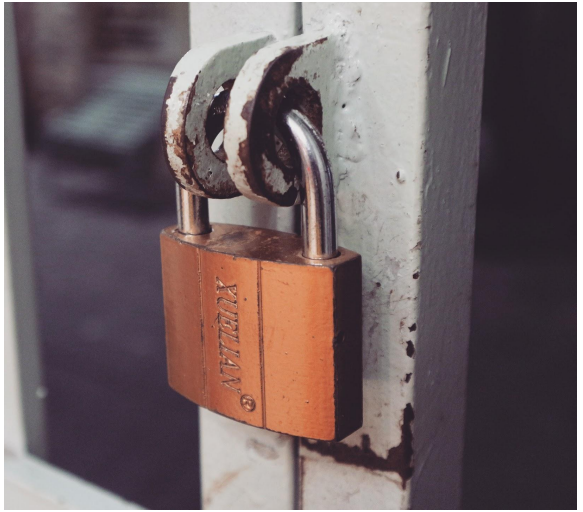
Deployment Considerations

- Device lifetimes.
- Managed vs unmanaged fleet:
  - Will you have direct control of deployed devices?
- Operating Environment:
  - How hostile is it?
  - How reliable is power and connectivity?
- Can the user modify the software?
- Is there some kind of end-user interface?
- Bandwidth:
  - Network
  - Cloud compute



Securing IOT Devices

- “The ‘s’ in IOT stands for security” - @tkadlec
- 1-25 bugs per 1000 lines of code*
  - Assume that all software components have vulnerabilities
- Use well-maintained software and keep it updated
- Review vendors for update policies
- General Security Practices
  - Principle of least privilege
  - Separation of privilege
  - Kerckhoff’s principle
    - “You can only design an encryption system that someone dumber than you cannot crack.”

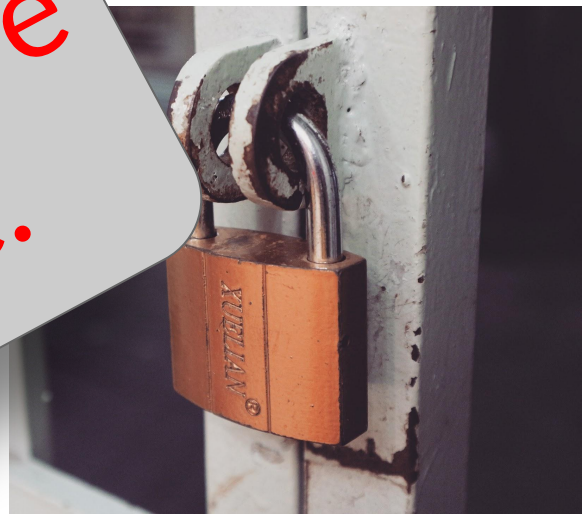


*Source: Steve McConnell, Code Complete

Securing IOT Devices

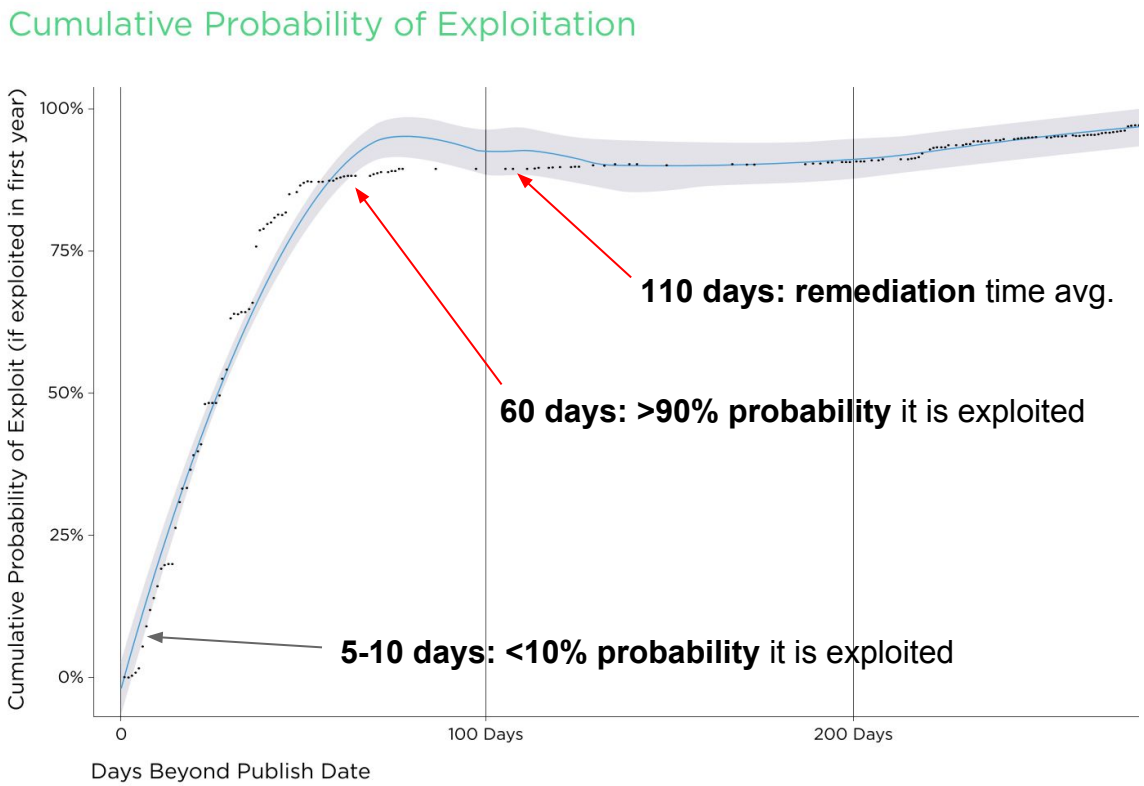
- "The 's' in IOT stands for security" - @tkadl
- 1-25 bugs per 1000 lines of code*
  - Assume that all software co… vulnerabilities
- Use well-maintained …
- Review vendors …
- General Sec…
  - Princi…
  - Separat…
  - Kerckhoi…
    - "You c… …n encryption system that so… …mber than you cannot crack."

OTA updates are a must have.



*Source: Steve McConnell, Code Complete

Security patching is done too late



Source: How the Rise in Non-Targeted Attacks Has Widened the Remediation Gap, Kenna Security

IOT Device Patching and Updates

- "**33% of current recalls** are for problems that could be fixed OTA" - ABI Research
- "OTA updates will **save carmakers $35B** in 2022" - IHS Automotive
- Considerations:
  - Long expected lifetime
  - No/expensive physical access
  - Unreliable power
  - Unreliable network connectivity
  - Public and insecure networks

OTA Update Design Criteria

- Robust - no bricked devices
- Secure - TLS and image signing
- Atomic - installed completely or not at all
- Consistent - test environment == production environment
- Automatic Rollback - safety
- Plugin architecture - expandability

Q&A - Thank you!

Resources:

- https://bit.ly/2GIKlUQ - Previous ELC Talk comparing Embedded Linux build systems
- https://ubm.io/2Iazdfn - Deeper dive into the Yocto project
- https://hub.mender.io/t/raspberry-pi-3-model-b-b/57 - Building Yocto for Raspberry Pi with Mender.

@drewmoseley

drew.moseley@mender.io