Open Source Foundries

DIY Connected IoT Products Using Open Source Software

Alan Bennett
alan@opensourcefoundries.com

The abstract

For the past 2 years our team has built reference IoT products using Open Embedded / Yocto, the Linux kernel, Zephyr and some open source device management platforms. We have struggled through incomplete frameworks, proprietary radio bugs, multiple IP stacks, and an ever-growing number of CVEs. Now we want to share all of our knowledge in a hands-on workshop/tutorial. In the workshop, you will learn about all of the components involved in an open source end-to-end IoT system and be able to build, test, deploy and deliver software updates to fielded devices. From the cloud to the edge and into wireless sensor devices, we will show you how to take advantage of all that open source software has to offer to build safe, secure and updatable devices.

Background

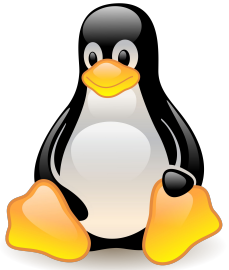
Open Source Foundries

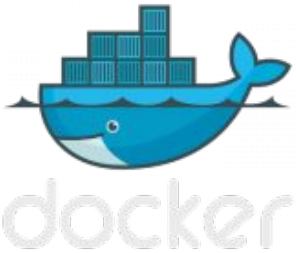
- Established October 1, 2018
  - Team was formerly known as Linaro Technologies Division
- Backgrounds in
  - Embedded Systems, (Linux, RTOS, PC BIOS, Windows, Android, etc...)
  - Linux Distributions
  - Consumer, Commercial, Military, Commercial Aviation Product development
  - Web frameworks
  - Advanced CI (LAVA, KernelCI.org)
- Other OSF Sessions at ELC / OpenIoT Summit
  - LWM2M & Zephyr - Mike Scott
  - Blockchain of things - Tyler Baker
  - Creating secure products using MCUBOOT and Zephyr - Marti Bolivar

Vocabulary: microPlatforms

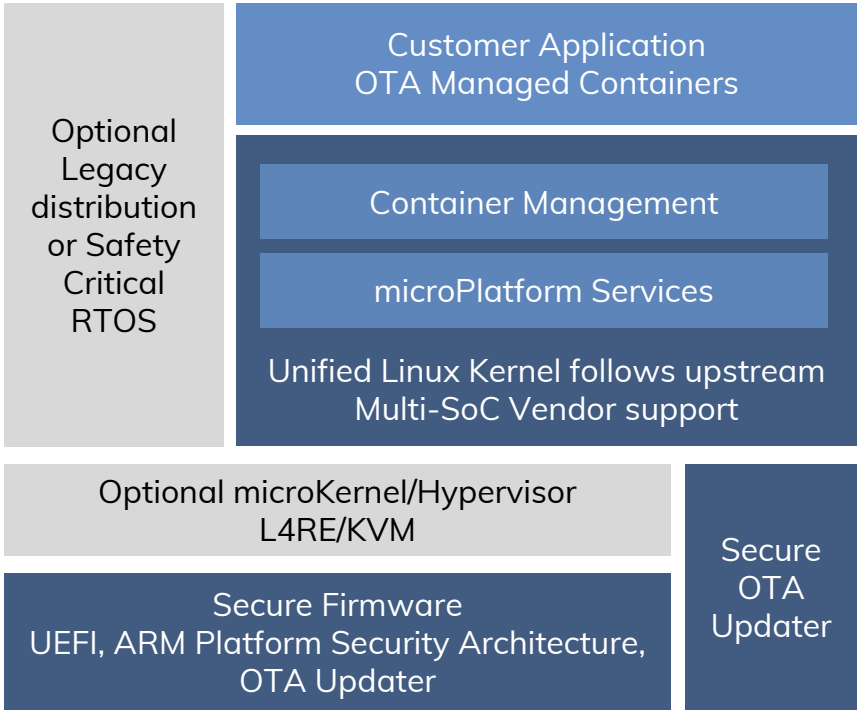
microPlatforms - OS / Distributions

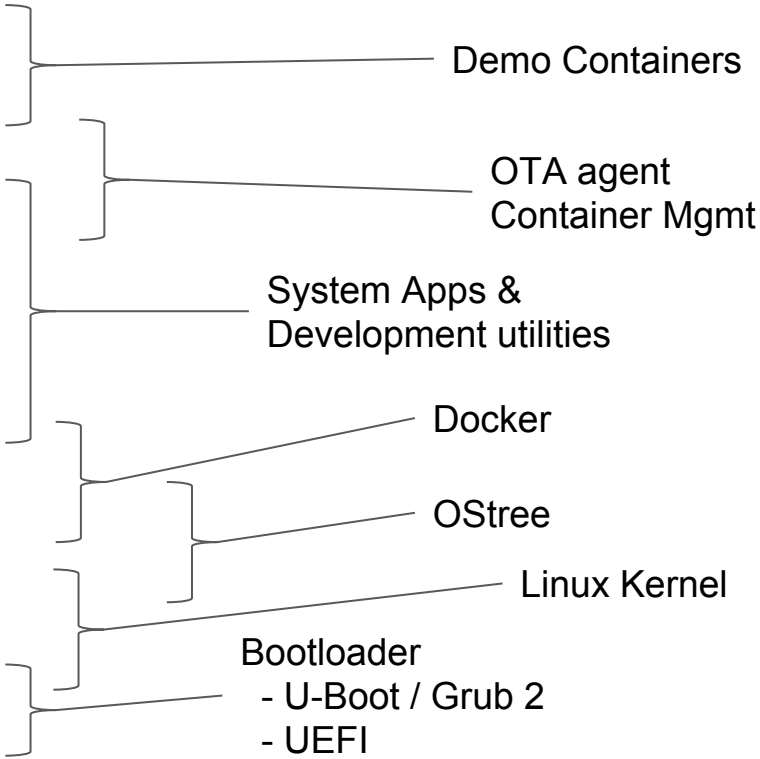
- Upstream, open source software
- microPlatforms are built directly from upstream open source projects
  - As close to tip as possible
  - Little or no non-upstream code
- Stabilized and tested for connected IoT use-case
- Continuous updates (integrated & fully tested)
  - Continuous merge-ups
- We publish open releases 1-2x / year
- We believe that the most secure and stable software is upstream software
- It's open source, there is No proprietary Lock-in





Linux microPlatform - the 'OS' for embedded systems





Demo: LMP

64-bit ARM
(armv8)

x86_64

32-bit ARM
(armv7)

Virtual
Machines

future,
tbd...

Note: the LMP Supports multiarch Containers

Single Dockerfile builds across all architectures; manifest points to arch-specific builds

Portainer

```
docker run -d -p 9000:9000 --restart always --name portainer  -v $PWD/data:/data -v
/var/run/docker.sock:/var/run/docker.sock portainer/portainer --logo
https://foundries.io/static/img/logo.png
```

Dump1090

```
docker run -d --restart always --privileged -v /dev/bus/usb:/dev/bus/usb -p 80:8080
--name dump1090 opensourcefoundries/dump1090:latest
```

Edge-X Foundry?

```
You can run complex and comprehensive edge software stacks on the LmP
```

Contents of today's Linux microPlatform**

Root /

```
bin -> usr/bin
boot
dev
etc
home -> var/rootdirs/home
lib -> usr/lib
media -> var/rootdirs/media
mnt -> var/rootdirs/mnt
ostree -> sysroot/ostree
proc
run
sbin -> usr/sbin
sys
sysroot
tmp
usr
var
```

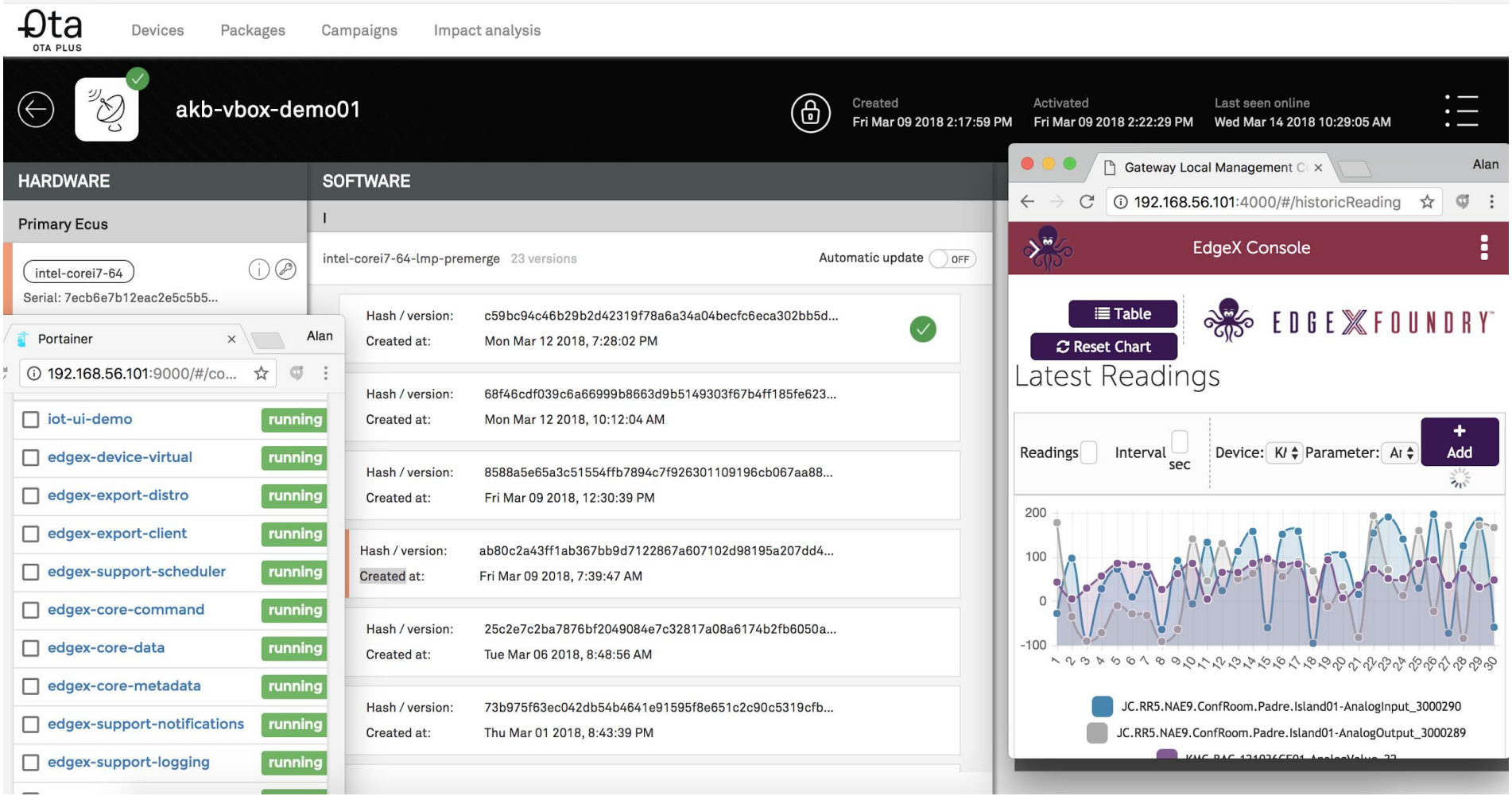
Filesystem size

```
637M    ./usr
145M    ./sysroot
0       ./dev
0       ./sys
14M     ./var
12M     ./etc
26M     ./boot
18M     ./run
0       ./tmp
0       ./proc
849M    .
```

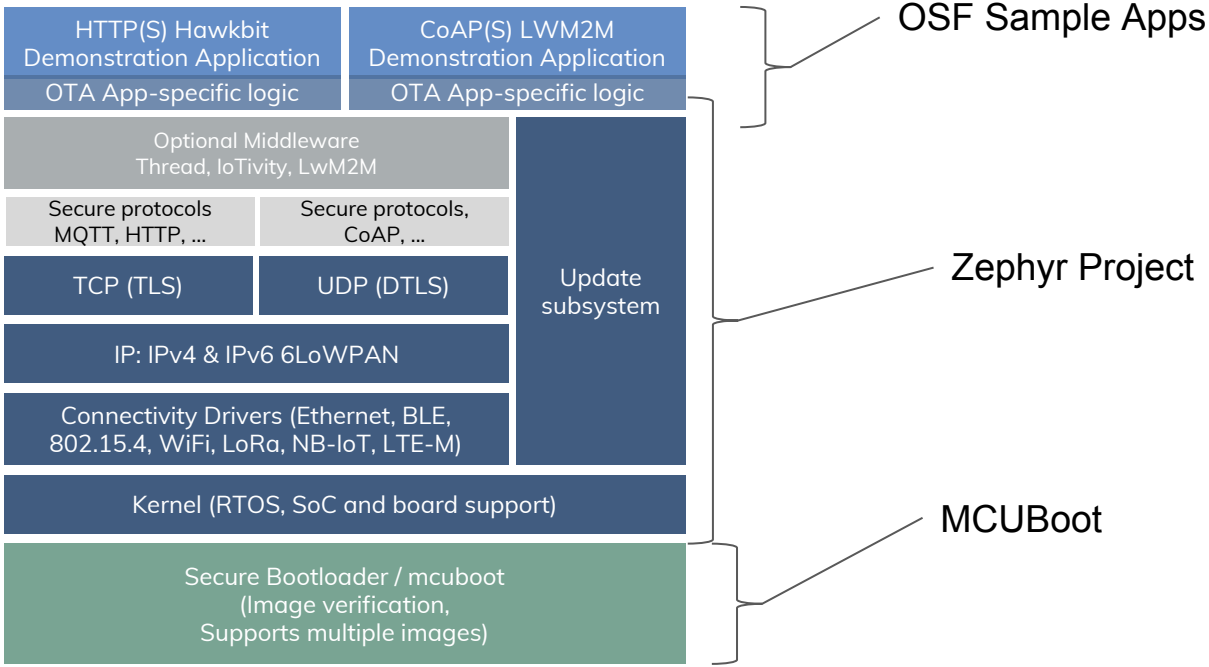
Running services

```
[] Kernel services
systemd-journal, udevd, timesyncd,
networkd, resolved
syslogd
dbus-daemon
Acpid
NetworkManager
Klogd
Avahi
Systemd-resolved
Dhclient
dockerd
polkit-d
```

** We are targeting a total size in the ~200 MB for the base LmP, Current builds are designed to enable maximum portability and functionality; It's not the time to optimize


Ota
OTA PLUS
Devices
Packages
Campaigns
Impact analysis
akb-vbox-demo01
Created
Fri Mar 09 2018 2:17:59 PM
Activated
Fri Mar 09 2018 2:22:29 PM
Last seen online
Wed Mar 14 2018 10:29:05 AM
HARDWARE
SOFTWARE
Primary Ecus
intel-corei7-64
Serial: 7ecb6e7b12eac2e5c5b5...
intel-corei7-64-lmp-premerge
23 versions
Automatic update
OFF
Hash / version: c59bc94c46b29b2d42319f78a6a34a04becfc6eca302bb5d...
Created at: Mon Mar 12 2018, 7:28:02 PM
Hash / version: 68f46cdf039c6a66999b8663d9b5149303f67b4ff185fe623...
Created at: Mon Mar 12 2018, 10:12:04 AM
Hash / version: 8588a5e65a3c51554ffb7894c7f9263011091 96cb067aa88...
Created at: Fri Mar 09 2018, 12:30:39 PM
Hash / version: ab80c2a43ff1ab367bb9d7122867a607102d98195a207dd4...
Created at: Fri Mar 09 2018, 7:39:47 AM
Hash / version: 25c2e7c2ba7876bf2049084e7c32817a08a6174b2fb6050a...
Created at: Tue Mar 06 2018, 8:48:56 AM
Hash / version: 73b975f63ec042db54b4641e91595f8e651c2c90c5319cfb...
Created at: Thu Mar 01 2018, 8:43:39 PM
Portainer
Alan
192.168.56.101:9000/#/co...
iot-ui-demo running
edgex-device-virtual running
edgex-export-distro running
edgex-export-client running
edgex-support-scheduler running
edgex-core-command running
edgex-core-data running
edgex-core-metadata running
edgex-support-notifications running
edgex-support-logging running
Gateway Local Management C
Alan
192.168.56.101:4000/#/historicReading
EdgeX Console
Table
Reset Chart
EDGE X FOUNDRY
Latest Readings
Readings
Interval
sec
Device:
Parameter:
Add
200
100
0
-100
JC.RR5.NAE9.ConfRoom.Padre.Island01-AnalogInput_3000290
JC.RR5.NAE9.ConfRoom.Padre.Island01-AnalogOutput_3000289

Zephyr microPlatform - OS for microcontrollers



Demo: ZmP - Zephyr dev - FASTLED light bulb

Zephyr UI's are likely coming, but why get tied into proprietary development methods

- Simple Dev environment - Github Atom & Terminal (CMAKE / GCC / …)

To bring several projects together we use Google's REPO

- After an init and a sync you have everything necessary to get started

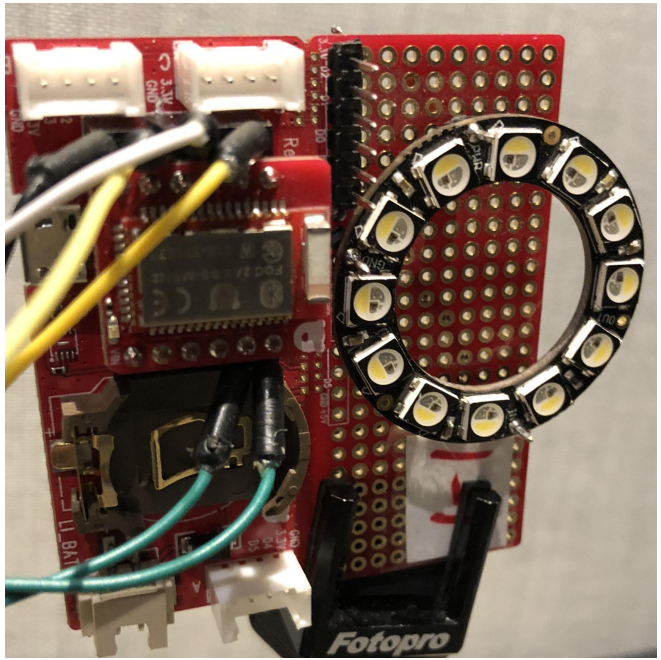
Demo: FASTLED circle

./zmp build --skip-signature -b nrf52_blenano2 zephyr/samples/drivers/led_ws2812/

cmake --build outdir/zephyr/samples/drivers/led_ws2812/nrf52_blenano2/app/ --target flash

: Change behavior

-re-build, re-flash



Build Zephyr microPlatform

Tooling

Currently we are using the git / repo-tool and the ‘zmp’ meta-tool

we are working with the Zephyr project to help define the best solution for the project

Repo Tool: https://source.android.com/setup/developing

The ZmP Manifest:

https://github.com/OpenSourceFoundries/zmp-manifest/blob/master/default.xml

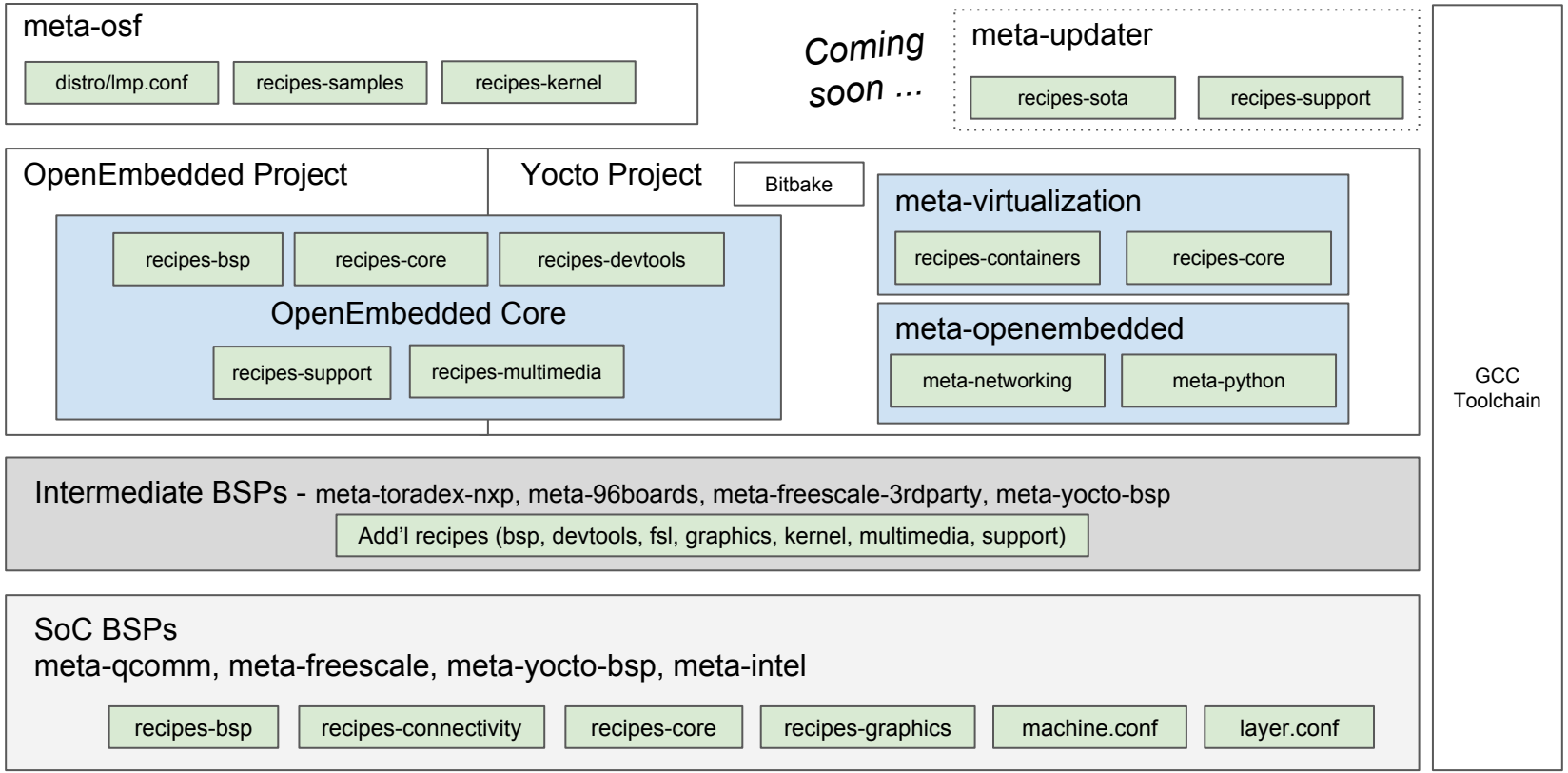
```
<manifest>
  <remote fetch="https://github.com/OpenSourceFoundries" name="OpenSourceFoundries" />

  <default remote="OpenSourceFoundries" revision="master" sync-j="4" />

  <project name="dm-hawkbit-mqtt" path="zephyr-fota-samples/dm-hawkbit-mqtt" remote="OpenSourceFoundries"
  <project name="dm-lwm2m" path="zephyr-fota-samples/dm-lwm2m" remote="OpenSourceFoundries" revision="9f
  <project name="mbedtls" path="mcuboot/sim/mcuboot-sys/mbedtls" remote="OpenSourceFoundries" revision="
  <project name="mcuboot" remote="OpenSourceFoundries" revision="94b2d79028049926303de587ab63fbc6cdfe4a58
  <project name="zephyr" remote="OpenSourceFoundries" revision="a02d2bd1c47c4a8ce58f0b570e20afcd1fffbf0d"
  <project name="zmp-build" path="build" remote="OpenSourceFoundries" revision="2c6fe0661e6dc236bbb46fcf6
    <linkfile dest="zmp" src="zmp.py" />
  </project>
  <project name="zmp-container" path="build/other/zmp-container" remote="OpenSourceFoundries" revision="c
  <project clone-depth="1" name="zmp-prebuilt" path="build/other/zmp-prebuilt" remote="OpenSourceFoundrie
</manifest>
```

Build Linux microPlatform

Linux microPlatform Architecture



Tooling

Currently we are using git / repo-tool to combine many repositories

https://source.android.com/setup/developing

Next: What does the Linux microPlatform look like?

https://github.com/OpenSourceFoundries/lmp-manifest/blob/master/default.xml

```
<manifest>
  <remote fetch="https://github.com" name="github" />
  <remote fetch="http://git.linaro.org" name="linaro" />
  <remote fetch="https://github.com/OpenSourceFoundries" name="OpenSourceFoundries" />
  <remote fetch="http://git.yoctoproject.org" name="yocto" />

  <default remote="github" revision="master" sync-j="4" />

  <project name="96boards/meta-96boards" path="layers/meta-96boards" revision="eee9e16eece8780ae44357f92664491174211a06" /
  <project name="Freescale/meta-freescale-3rdparty" path="layers/meta-freescale-3rdparty" revision="1a3fb4e0c726429dfa48f1
  <project name="core-containers" path="containers/core-containers" remote="OpenSourceFoundries" revision="62d1178666390e3
  <project name="cpriouzeau/meta-st-cannes2" path="layers/meta-st-cannes2" revision="8abc5197f3456590c493c59d1e6f696e25f88
  <project name="extra-containers" path="containers/extra-containers" remote="OpenSourceFoundries" revision="a80304f9f9ea6
  <project name="gateway-containers" path="containers/gateway-containers" remote="OpenSourceFoundries" revision="4861f0849
  <project name="git/meta-freescale" path="layers/meta-freescale" remote="yocto" revision="bf7fd9cfe0788fe2f819a4ae2cc7db8
  <project name="git/meta-intel" path="layers/meta-intel" remote="yocto" revision="7969d8e442bdefd8036a334ca9d9ce133272399
  <project name="git/meta-raspberrypi" path="layers/meta-raspberrypi" remote="yocto" revision="c47caaca325b8cd81ee5bcd7cb3
  <project name="git/meta-virtualization" path="layers/meta-virtualization" remote="yocto" revision="d1969606e3540d3771a5b
  <project name="git/meta-yocto" path="layers/meta-yocto" remote="yocto" revision="0d44e59bfaa95162cf2133df1d08f6419314bb8
    <linkfile dest="setup-environment" src="../../.repo/manifests/setup-environment" />
  </project>
  <project name="meta-osf" path="layers/meta-osf" remote="OpenSourceFoundries" revision="b46d57d863a98699a801a1b417c9988ef
  <project name="ndechesne/meta-qcom" path="layers/meta-qcom" revision="5fbd011c7852d46399018647d09e3b581b1b637d" />
  <project name="openembedded/bitbake" path="bitbake" revision="643eacb162b8710330ef292bfda21cfeab97f95c" />
  <project name="openembedded/meta-linaro" path="layers/meta-linaro" remote="linaro" revision="259b8bfbb5dc13a4173f9e90bdf
  <project name="openembedded/meta-openembedded" path="layers/meta-openembedded" revision="29a4983d5a4462d8e7b9abcd55bfb30
  <project name="openembedded/openembedded-core" path="layers/openembedded-core" revision="e0a4e78b879eeacff8ef6803c134505
</manifest>
```

Yocto Build system

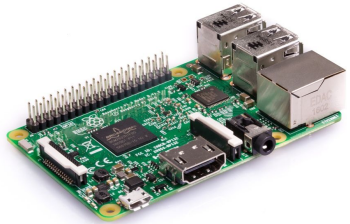
- Now you have the source, build it
- Building Yocto is ... it takes time but it can be easier
  - https://foundries.io/docs/latest/reference/linux-building.html
  - Shared cache ~ minutes depending on size of change
  - Comprehensive rebuild ~ 4-6 hours
  - Native
  - LMP build container, volume mount outputs, etc...
- Or you can just download some prebuilts for your target
  - https://foundries.io/mp/lmp/latest/artifacts/

LWM2M demo system

An end to end system

- Simulated Cloud
  - Run Leshan on a local Laptop
- Basic IoT Gateway
  - Start with bare metal
  - Add the Linux microPlatform
  - Add enablement Containers
    - BLE bridge
    - IP Proxies
- Endpoint Devices
  - Start from bare metal
  - Build and deploy software
  - Thermal Sensors in the device
  - Light Control
  - OTA -capable application

Device Management



Linux microPlatform

To build the IoT Edge Gateway we use the Linux microPlatform and Docker containers to provide data proxy and Bluetooth Low Energy / 6LoWPAN bridging

Zephyr microPlatform

IoT devices run the Zephyr microPlatform and communicate to the gateway using LWM2M across a wireless connection using Bluetooth Low Energy / 6LoWPAN.

Eclipse Foundation: Leshan LwM2M server

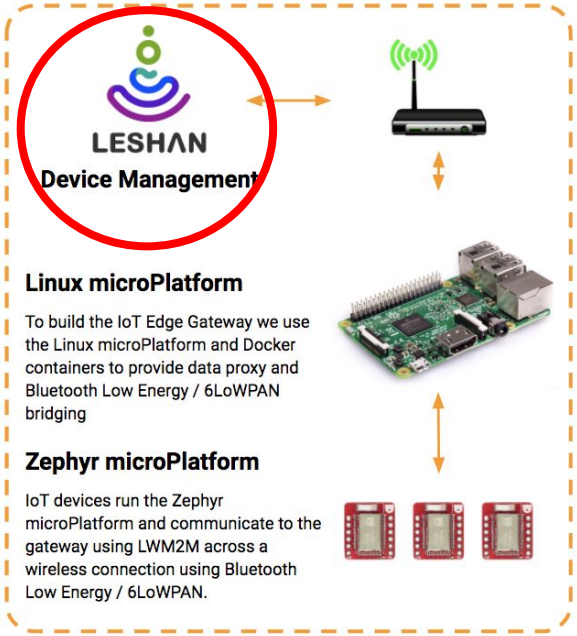
For simplicity, we are going to install Leshan onto the Gateway device

docker run opensourcefoundries/leshan -p 8081:8080

> Why aren't we using the upstream container? We rebuild to support multiarch

Open web browser

> http://192.168.1.125:8081/#/clients



Get LmP running - Add the gateway containers

- bt-joiner - Find devices and establish a BLE/IPv6/6LoWPAN bridge
- cf-proxy-coap-http - proxy CoAP to HTTP
- nginx-coap-proxy - proxy coap
- nginx-http-proxy - proxy IPv4 to the IPv6 6LoWPAN devices
- mosquitto - edge mqtt broker

- We use Ansible to remotely deploy the gateway containers
  - https://github.com/OpenSourceFoundries/gateway-ansible
  - GW_HOSTNAME=192.168.0.33 MGMT_SERVER=10.11.21.149 ./iot-gateway.sh

Build and flash the LWM2M Sample

```
repo init -u
https://github.com/opensourcefoundries/zmp-manifest
repo sync
./zmp build -b nrf52_blenano2 zephyr-fota-samples/dm-lwm2m
./zmp flash -b nrf52_blenano2 zephyr-fota-samples/dm-lwm2m
```

ZmP OTA

Upload binary to a route-able HTTP server

```
cd outdir/zephyr-fota-samples/dm-lwm2m/nrf52_blenano2/app/

python3 -m http.server
```

http://192.168.1.111:8000/zephyr/

Linux microPlatform OTA

Part 2: OTA for Linux Platforms

LmP OTA

- BoF at ELC-E in Prague, Oct. 2017
  - ostree
  - swupdate
  - Meta-mender
  - TUF and Uptane specifications for software updates
- Settled on a TUF/Uptane compliant ostree image
  - Developed initially for AGL (Automotive Grade Linux)
  - Provides a TUF and UPTANE compatible implementation

libostree / ostree

- OSTree only supports recording and deploying complete (bootable) filesystem trees (it’s not a package manager)
- OSTree updates are small, deduplicated
- The system is R/O and uses keep changes to /home, /etc and /var (i.e. Docker)
- Works on top of any filesystem or block storage layout
- OSTree repository stored in /ostree/repo, and a set of "deployments" stored in /ostree/deploy/$STATEROOT/$CHECKSUM.
- OSTree will perform a basic 3-way diff, and apply any local changes to the new copy, while leaving the old untouched.

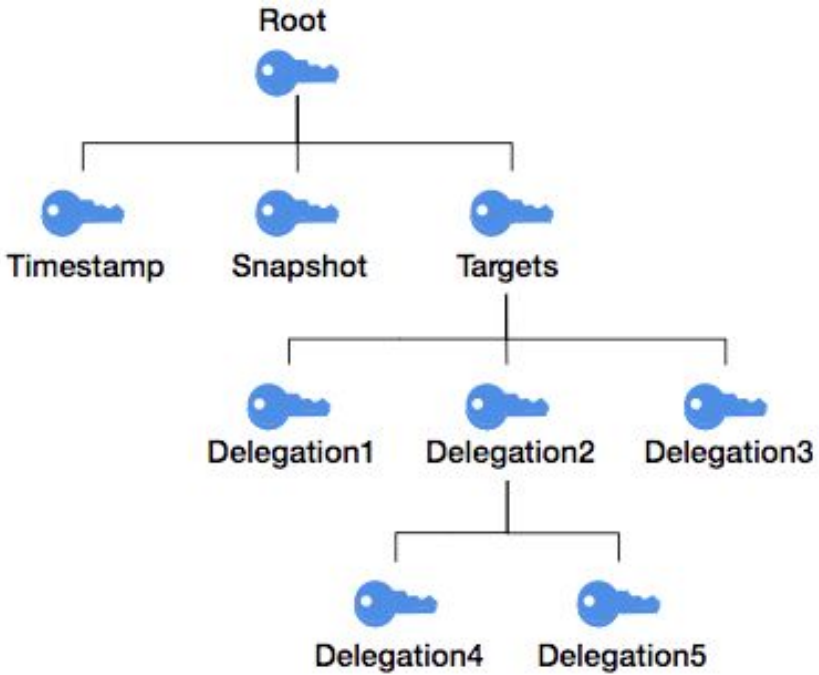
https://ostree.readthedocs.io/en/latest/manual/introduction/

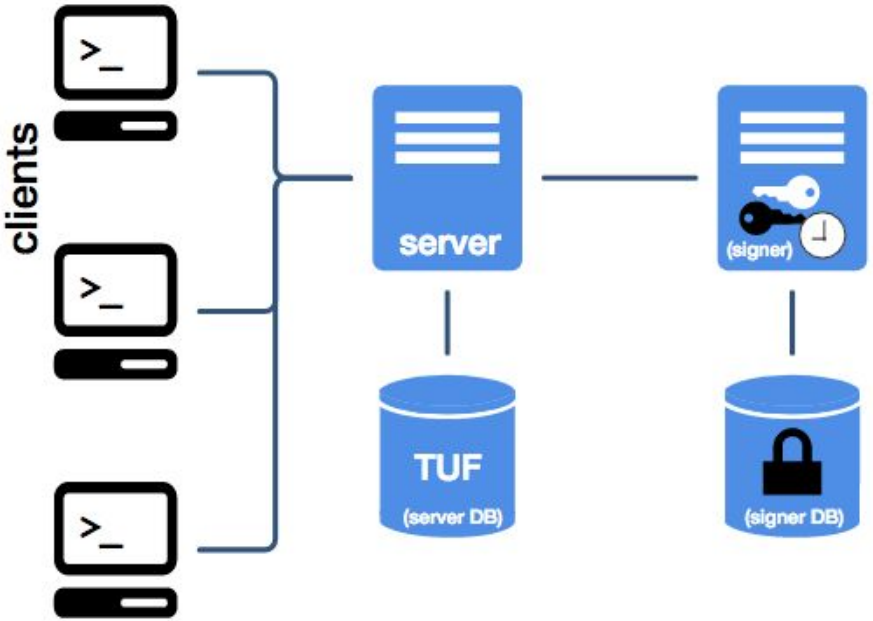
Threats covered by TUF - The update framework

Designed from the ground up to provide compromise resilience and protect against nation state attacks - circa 2009

- Arbitrary installation attacks.
- Endless data attacks.
- Extraneous dependencies attacks.
- Fast-forward attacks.
- Indefinite freeze attacks.
- Malicious mirrors preventing updates.
- Mix-and-match attacks.
- Rollback attacks.
- Slow retrieval attacks.
- Vulnerability to key compromises.
- Wrong software installation.

The TUF model





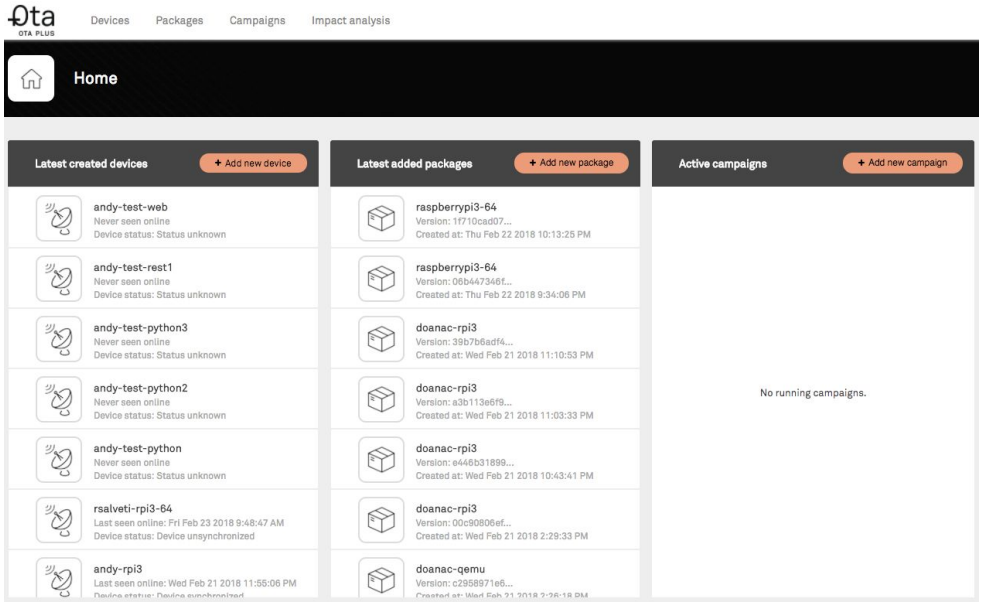
Use OTA CE to do an LMP Update

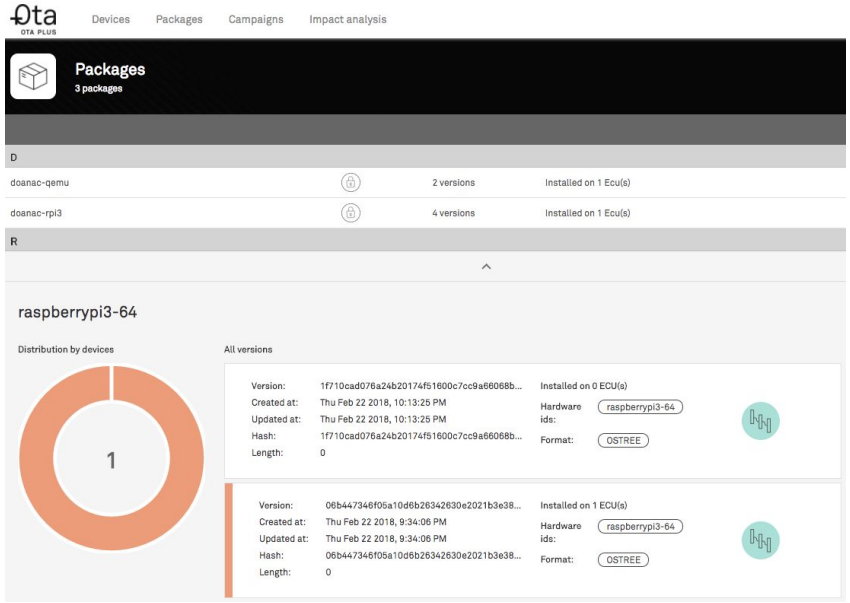
- Hosted Cloud
  - Run OTA Community Edition in the Cloud @ mgmt.foundries.io
- Basic IoT Gateway
  - Start with bare metal
  - Add the Linux microPlatform
  - Add enablement Containers
    - BLE bridge
    - IP Proxies
- Devices running LWM2M

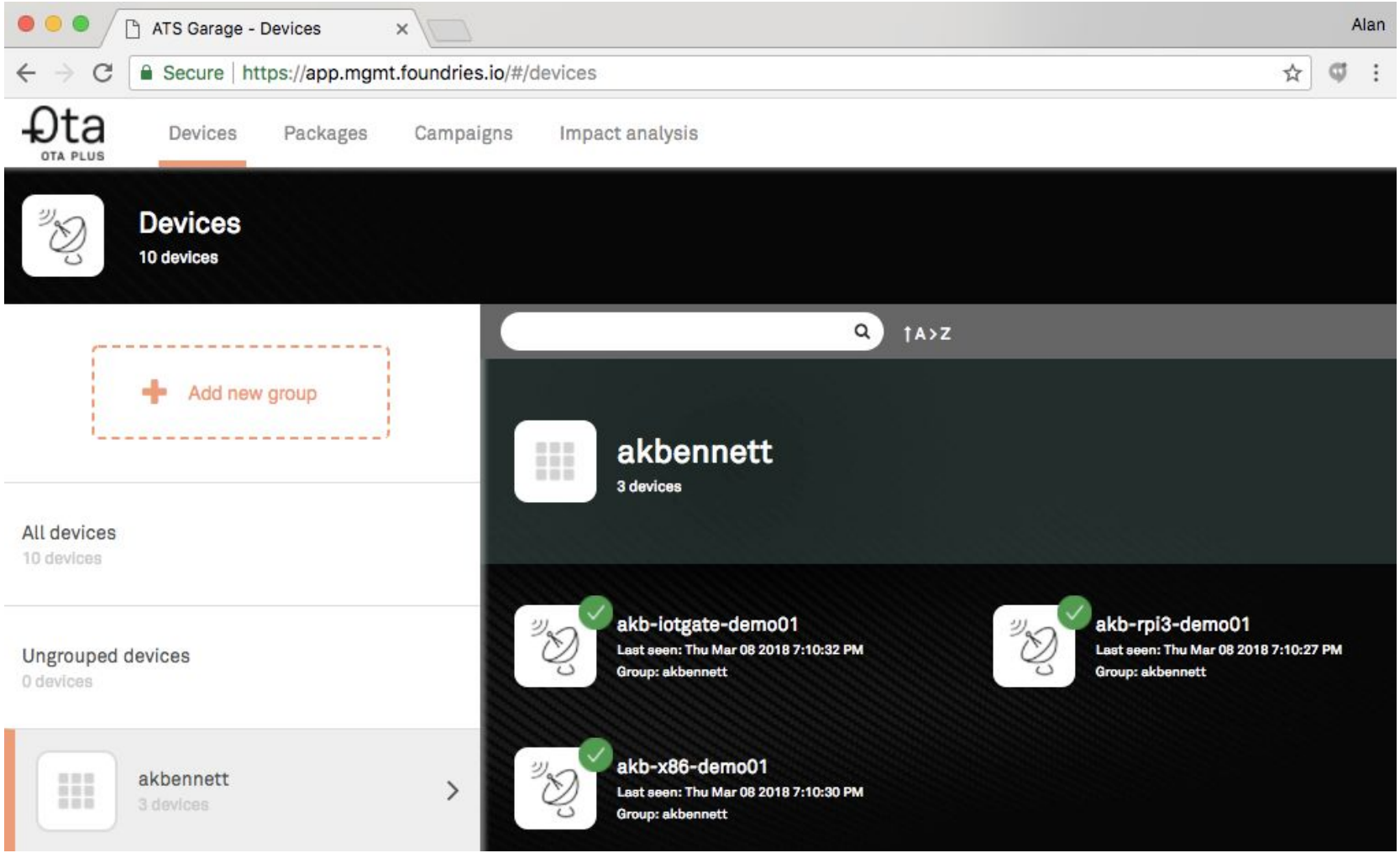
Linux microPlatform

To build the IoT Edge Gateway we use the Linux microPlatform and Docker containers to provide data proxy and Bluetooth Low Energy / 6LoWPAN bridging

OTA CE / Open-source / ATS Garage






ATS Garage - Devices
Alan
Secure
https://app.mgmt.foundries.io/#/devices
Ota
OTA PLUS
Devices
Packages
Campaigns
Impact analysis
Devices
10 devices
Add new group
All devices
10 devices
Ungrouped devices
0 devices
akbennett
3 devices
A>Z
akbennett
3 devices
akb-iotgate-demo01
Last seen: Thu Mar 08 2018 7:10:32 PM
Group: akbennett
akb-rpi3-demo01
Last seen: Thu Mar 08 2018 7:10:27 PM
Group: akbennett
akb-x86-demo01
Last seen: Thu Mar 08 2018 7:10:30 PM
Group: akbennett

What does the OTA CE look like

ATS GARAGE
A HERE Company

App
- This is the web interface. It uses the services below to support it.

Web-events
- This is a web-socket server that's used by the web interface

Treehub
- API to manage the OSTree blobs. The build process uploads to it and devices pull updates from it.

Device-registry
- API for registering and updating devices. eg - We create a device here as part of the implicit provisioning step.

Campaigner
- An API to manage rolling out updates to a fleet of devices

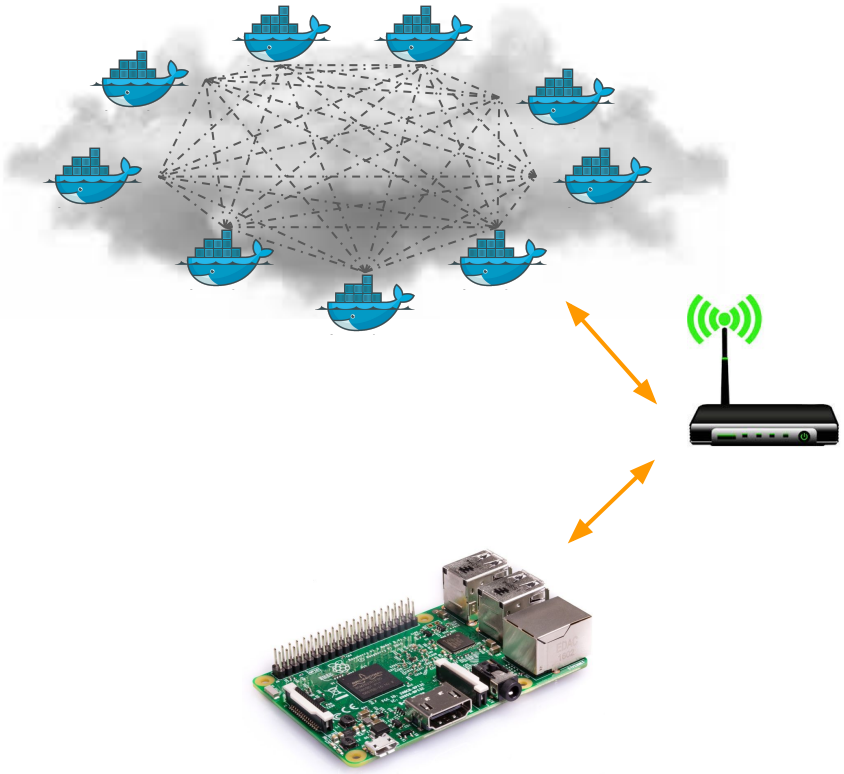
Director
- Orchestrates the installation of ECU-specific images. It uses online keys to sign metadata / which updates to install on which ECUs.

Gateway
- NGINX reverse-proxy to the treehub, director, and tuf-reposerver services.

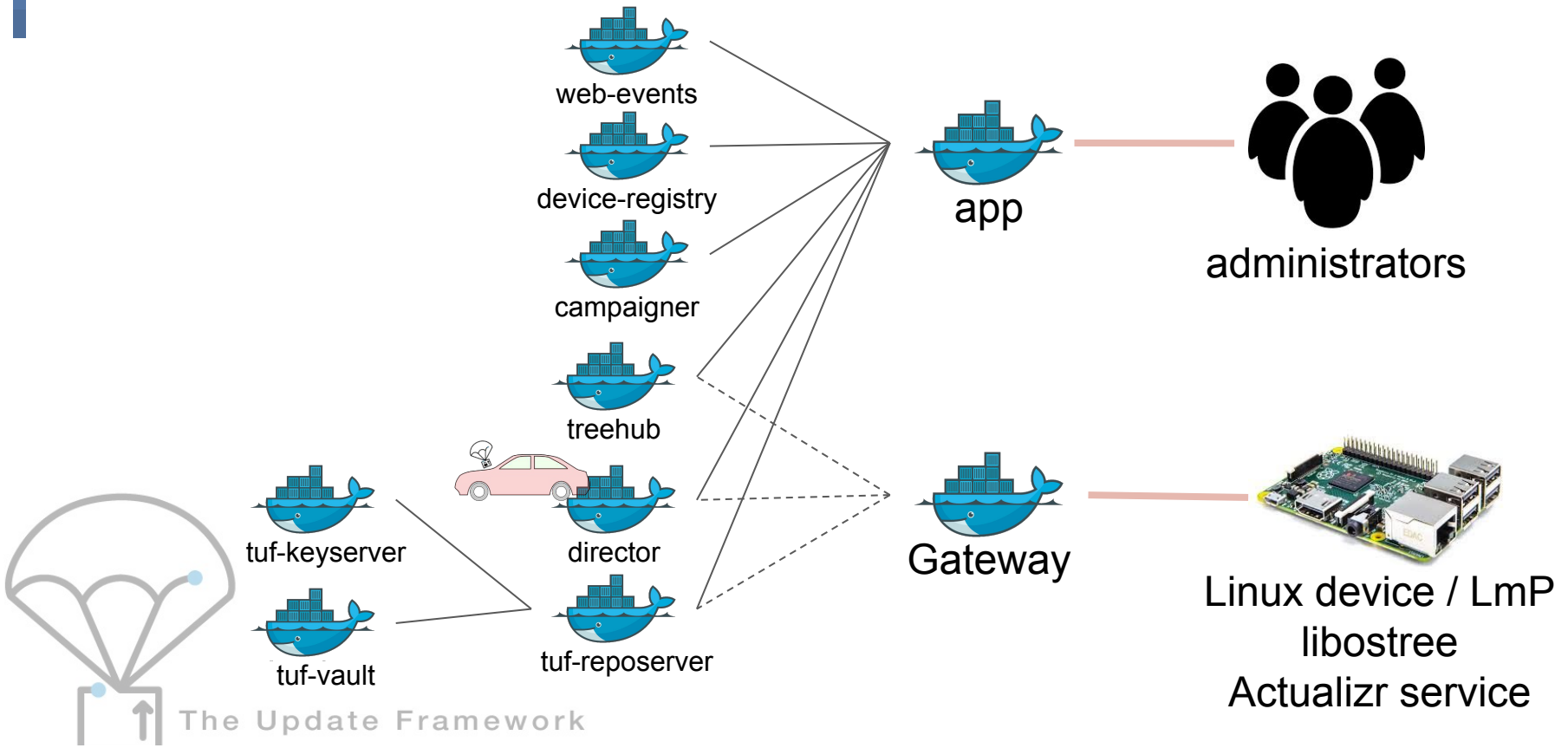
Tuf-keyserver
- Manages key generation and online role signing for tuf roles

Tuf-reposerver
- Manages tuf metadata for tuf Repositories

tuf-vault
- Credential storage



OTA CE microservice entry points



Thank you


OPEN SOURCE
FOUNDRIES


Embedded Linux
Conference


OpenIoTSummit