Balena as a Deployment Stack for SoCs? [Y/n/maybe?]
Overview

1. Our particular problem
2. Options considered
3. Balena
   a. System Architecture
   b. OS Architecture
   c. Deployment
   d. Open vs Commercial
   e. Building BalenaOS
   f. Balena mapped to our problem
   g. A hypothetical system deployment
4. Live Demo
5. Summaries & Discussion seeds
Our particular problem

1. Hundreds of SoCs
2. Heterogeneous:
   a. Hardware
   b. Development teams
   c. Purpose
3. ‘Modes’ requested
   a. Development
   b. Test stands
   c. Deployment & Debugging
   d. Operations & Monitoring
   e. Maintenance
Options considered

- Bash scripts
- Puppet, Ansible, Salt, Chef
  - Pros:
    - Scalable, good community, git-able, ssl/ssh, some GUIs
    - CERN uses puppet
    - Server-Client (Ansible client only)
    - Redundancy
- Docker and company
  - Compose
  - Swarm
  - Kubernetes

Requirements:

1. Scalable:
   - Boards, OSes, Applications, Servers, Crates & Mezzanines
2. Support multiple architectures
   - ARM, x86, 32/64b, Xilinx & Intel
3. Support for multiple applications per SoC OS
4. Easy to use and configure
5. Deployable:
   - Development, Test stands, Build servers, Operations, Local and Remote debugging
   - Automation, CI, version control?
6. Enhances existing workflows
7. Easy to monitor and debug (not a black box)
8. Long term maintainable and avoidance of technical debt
9. Open source ideally(?)
Balena System Architecture

Open source project, deploying yocto build linux supporting multiple docker containers, on embedded devices at scale

- Balena OS
- Balena Cloud
- Commercial and Open versions

- www.balena.io/
Balena System Architecture

- API
- ‘Docker’ Registry
- Git server
- Build server
- VPN manager
- Web GUI
- Balena OS on devices
BalenaOS Architecture
BalenaOS Architecture

- Yocto build base OS images available on dockerhub
- \(<\text{HW}> / <\text{Distro}> / <\text{Language}> / <\text{language version}> / <\text{distro version}>\)
  - Armv5e, armv6, armv7hf, aarch64, amd64, i386
  - Debian, Alpine, Ubuntu, Fedora (no Centos in commercial offering currently)
  - Node.js, Python, openJDK, Golan, Dotnet, C, ...
- Development vs production images
  - Passwordless ssh on 2222, docker on 2377
  - Getty on serial
  - Local mode
- Logs to RAM by default (save the SDcard)
- Runtime configuration to the host OS and supervisor: config.json
  - UUID or broadcast configured name
BalenaOS Architecture

- Development vs production images
  - Passwordless ssh on 2222, docker on 2377
  - Getty on serial
  - Local mode
- Logs to RAM by default (save the SDcard)
- Runtime configuration to the host OS and supervisor: config.json
  - Eg:
    - UUID or broadcast configured name
    - HW config settings (enable uart etc)
    - Update mode
    - Custom ENV
Balena: Deployment

- **Mechanisms**
  - `$ balena push: Git server > Build machine > update registry > update devices`
  - `$ balena deploy //Skip git & builder`
  - `$ balena build // Doesn’t update registry, useful for self tests`
  - **Update strategies:**
    - Download-then-kill (Default)
    - kill-then-download (RAM limited devices)
    - Delete-then-download (storage limited devices/large images)
    - Hand-over (zero downtime)

- **Fault tolerant update**
- **Secrets supported**
- **Full host and container ssh access**
- **Automation and continuous Integration**
Balena: Deployment

```
docker-compose.yml

version: '2'
services:
  frontend:
    build: ./frontend
    expose:
      - "80"
  proxy:
    build: ./haproxy
    depends_on:
      - frontend
      - data
    ports:
      - "80:80"
  data:
    build: ./data
    expose:
      - "8080"
  hello:
    build: ./hello
    ports:
      - "8000:8000"

Dockerfile.template

FROM resin/%%RESIN_MACHINE_NAME%%-golang:1.9-stretch AS build

WORKDIR /go/src/github.com/resin-io-projects/app

COPY /app ./

RUN go build

FROM resin/%%RESIN_MACHINE_NAME%%-debian:stretch


CMD ./app
```
Balena: Open vs Commercial

- Distinctions between OpenBalena and Balena Cloud
  - No web GUI
  - Single admin user
  - No binary container deltas
  - No support
  - Self hosted
Building BalenaOS-<Centos?> for SoCs

- **Tools**
  - Yocto
  - Qemu
  - Balena build
  - Configurable Continuous Integration
    - `git commit` → `git push` → Gitlab CI hook to build → Registry updated → Devices notified → Devices update according to update config
    - **Unit testing of applications?**
Building BalenaOS-<Centos?> for SoCs

- **Yocto layers**
  - poky
  - meta-openembedded
  - meta-balena
    - Meta-balena-thud (or other yocto v)
    - meta-resin-common
  - meta-xilinx
  - meta-balena-CERN_boardX
    - recipes-containers
    - recipes-kernel
    - recipes-bsp
    - recipes-core
Balena mapped to our problem

1. ShepHERD’s Crook
   (Beta & concept only design!)

   - Containerisation
   - APIs (C++)
   - Puppet & git
   - Configuration files
   - Standards
   - Central DAQ and RCMS integration
A hypothetical system deployment
Live Demo
Live Demo

BalenaCloud

CERN_Demo_Pi
(Application)

Raspberry Pi
Raspberry Pi

OpenBalena

CERN_Demo_BB
(Application)
BeagleBone Black

CERN_Demo_Pi
(Application)
Raspberry Pi
BalenaCloud - Application
BalenaCloud - Device
<table>
<thead>
<tr>
<th>Date</th>
<th>Time</th>
<th>Offset</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.06.19</td>
<td>07:51:44</td>
<td>-0400</td>
<td>data a user connected</td>
</tr>
<tr>
<td>10.06.19</td>
<td>08:00:53</td>
<td>-0400</td>
<td>data a user disconnected</td>
</tr>
<tr>
<td>10.06.19</td>
<td>08:00:56</td>
<td>-0400</td>
<td>data a user connected</td>
</tr>
<tr>
<td>10.06.19</td>
<td>08:14:25</td>
<td>-0400</td>
<td>data a user disconnected</td>
</tr>
<tr>
<td>10.06.19</td>
<td>08:17:33</td>
<td>-0400</td>
<td>data a user connected</td>
</tr>
<tr>
<td>10.06.19</td>
<td>08:45:19</td>
<td>-0400</td>
<td>data a user disconnected</td>
</tr>
<tr>
<td>10.06.19</td>
<td>08:53:03</td>
<td>-0400</td>
<td>data a user connected</td>
</tr>
<tr>
<td>10.06.19</td>
<td>09:02:57</td>
<td>-0400</td>
<td>data a user connected</td>
</tr>
<tr>
<td>10.06.19</td>
<td>09:03:05</td>
<td>-0400</td>
<td>data a user disconnected</td>
</tr>
<tr>
<td>10.06.19</td>
<td>09:03:08</td>
<td>-0400</td>
<td>data a user disconnected</td>
</tr>
</tbody>
</table>
BalenaCloud - Terminal Session

Terminal

Select a target

Host OS

- hello
- data
- frontend
- proxy
BalenaCloud - Deploy Changes

- BalenaCloud is a git remote for source
  - git add
  - git commit
  - git push balena

- When new code is pushed to BalenaCloud, it is built and deployed to all devices associated with the application
### OpenBalena - Application

<table>
<thead>
<tr>
<th>ID</th>
<th>APP NAME</th>
<th>DEVICE TYPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>CERN_Demo_BB</td>
<td>beaglebone-black</td>
</tr>
<tr>
<td>2</td>
<td>GitlabRunnerTestAppJetson</td>
<td>jetson-tx1</td>
</tr>
<tr>
<td>3</td>
<td>GitlabRunnerTestAppBeagle</td>
<td>beaglebone-black</td>
</tr>
<tr>
<td>6</td>
<td>CERN_Demo_Pi</td>
<td>raspberry-pi3</td>
</tr>
<tr>
<td>1</td>
<td>GitlabRunnerTestApp</td>
<td>raspberry-pi3</td>
</tr>
<tr>
<td>5</td>
<td>BaleanBeagleGost</td>
<td>beaglebone-black</td>
</tr>
</tbody>
</table>

```bash
[0] % balena app CERN_Demo_Pi
== CERN DEMO PI
ID: 6
DEVICE TYPE: raspberry-pi3
COMMIT: 2cf4c0693c8f8d8f3375204107912a34
```

```bash
[0] % balena app CERN_Demo_BB
== CERN DEMO BB
ID: 7
DEVICE TYPE: beaglebone-black
COMMIT: a895d349c386f4d8b0cd0c81ed446eb8
```
OpenBalena - Device

```
[0] % balena devices --application CERN_Demo_Pi
ID UUID DEVICE NAME DEVICE TYPE APPLICATION NAME STATUS IS ONLINE SUPERVISOR VERSION OS VERSION
7 2a0e240 dawn-moon raspberryPi3 CERN_Demo_Pi Idle true 9.15.0 balenaOS 2.36.0+rev2
6 12e308d empty-grass raspberryPi3 CERN_Demo_Pi Idle false 9.15.0 balenaOS 2.36.0+rev2

[0] % balena device 7
== SMALL DUST
ID: 4
DEVICE TYPE: beaglebone-black
STATUS: offline
IS ONLINE: false
IP ADDRESS: 192.168.0.100
APPLICATION NAME: GitlabRunnerTestAppBeagle
UUID: 7586b201b9d9d804a3c6fbc5edc01f6d
COMMIT: 44663c904bf2402aaec2ca449b20c36b
SUPERVISOR VERSION: 9.0.1
OS VERSION: balenaOS 2.29.2+rev3
DASHBOARD URL: https://dashboard.iot.crc.nd.edu/devices/7586b201b9d9d804a3c6fbc5edc01f6d/summary
```
OpenBalena - Logs

[0] % balena logs 2a0e240
[Logs] [6/5/2019, 2:54:39 PM] Downloading image 'registry.iot.crc.nd.edu/v2/3b23b8d96b2cfc77dc8b80742ad0@sha256:a93cd6d800e6e072d3f70000e2fbb2aa4d1b56c4cef395235b1657d36026a75d'
[Logs] [6/5/2019, 2:55:27 PM] Downloaded image 'registry.iot.crc.nd.edu/v2/3b23b8d96b2cfc77dc8b80742ad0@sha256:a93cd6d800e6e072d3f70000e2fbb2aa4d1b56c4cef395235b1657d36026a75d'
[Logs] [6/5/2019, 2:55:27 PM] Installing service 'main sha256:92e326fba0abde4ed16a58458c00df9d0dbe2a0333ec59295933d6451804'
[Logs] [6/5/2019, 2:55:29 PM] Installed service 'main sha256:92e326fba0abde4ed16a58458c00df9d0dbe2a0333ec59295933d6451804'
[Logs] [6/5/2019, 2:55:29 PM] Starting service 'main sha256:92e326fba0abde4ed16a58458c00df9dc8be2a0333ec59295933d6451804'
[Logs] [6/5/2019, 2:55:30 PM] Started service 'main sha256:92e326fba0abde4ed16a58458c00df9dc8be2a0333ec59295933d6451804'
[Logs] [6/5/2019, 2:55:31 PM] [main] Example app listening on port :80
Summaries 1

- Demo recap:
  - BalenaCloud - web interface, git integration
  - OpenBalena - CLI only
  - All devices in a Balena application are updated at once
  - Can operate on one device or entire fleet
Summaries 2

Requirements:

1. Scalable ✓
2. Support multiple architectures ✓
   a. ARMs, x86, 32/64b, Xilinx & Intel
3. Support for multiple applications per SoC OS ✓
4. Easy to use and configure ½ ✓
5. Deployable for: Development, Test stands, Operations, Local and Remote debugging ✓
6. Support automation, CI, version control ✓
7. Enhances existing workflows ½ ✓
8. Easy to monitor and debug (not a black box) ✓
9. Long term maintainable ✓
10. Open source ideally(?) ✓
Summaries 3

Requirements:

1. Scalable ✓
2. Support multiple architectures ✓
   a. ARMs, x86, 32/64b, Xilinx & Intel
3. Support for multiple applications per SoC OS ✓
4. Easy to use and configure ½ ✓
5. Deployable for: Development, Test stands, Operations, Local and Remote debugging ✓
6. Support automation, CI, version control ✓
7. Enhances existing workflows ½ ✓
8. Easy to monitor and debug (not a black box) ✓
9. Long term maintainable and avoiding technical debt ½ ✓
10. Open source ideally(?) ✓

Additionally
(heard at CERN this week):

1. Yocto based build advantageous given more flexibility
2. Mainline kernel supportable
Summaries 4

Considerations:

- Missing pieces:
  - Web GUI
  - Redundancy tolerance of cloud services
  - Single admin user on open cloud
- Requires we make balena yocto layers for each board and Centos (iff we want Centos)
- Selecting any external tool adds an external dependency
- PL management (easy to include but not standard)
$ Balena as a Deployment Stack for SoCs? [Y/n/maybe?]  
$ |
Balena: Features summary

- Docker volumes (default /data))
- Multiple applications per device
- Easy device provisioning
- Network configuration
- Unique IDs
- Local development
- Remote configuration
- Easy updates
- Device logs
- Group actions