Running Mesos/Marathon on ARM Servers

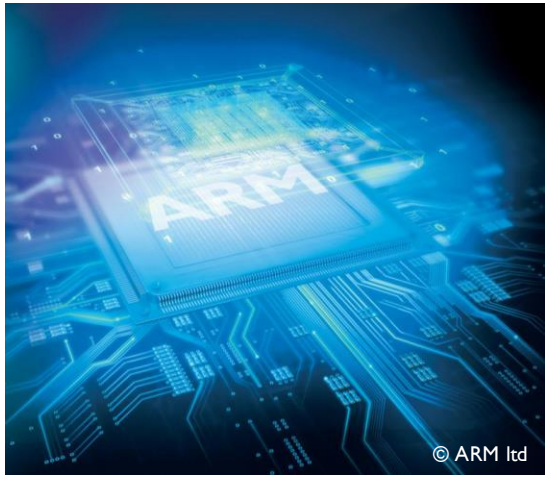
ARM

Yuqi Gu

Senior Software Engineer

MesosCon Asia, Beijing
Jun 22, 2017

Agenda



- Why run Mesos on AArch64?
- How to setup a Mesos/Marathon cluster on AArch64?
- Demo - Deploy and Scale the containerized Apps by Mesos/Marathon on ARM cluster

Why run Mesos on AArch64?

- What is “AArch64” ?

  AArch64 was introduced in ARMv8-A, which was announced in October 2011.

  ARMv8-A adds an optional 64-bit architecture named "**AArch64**“.

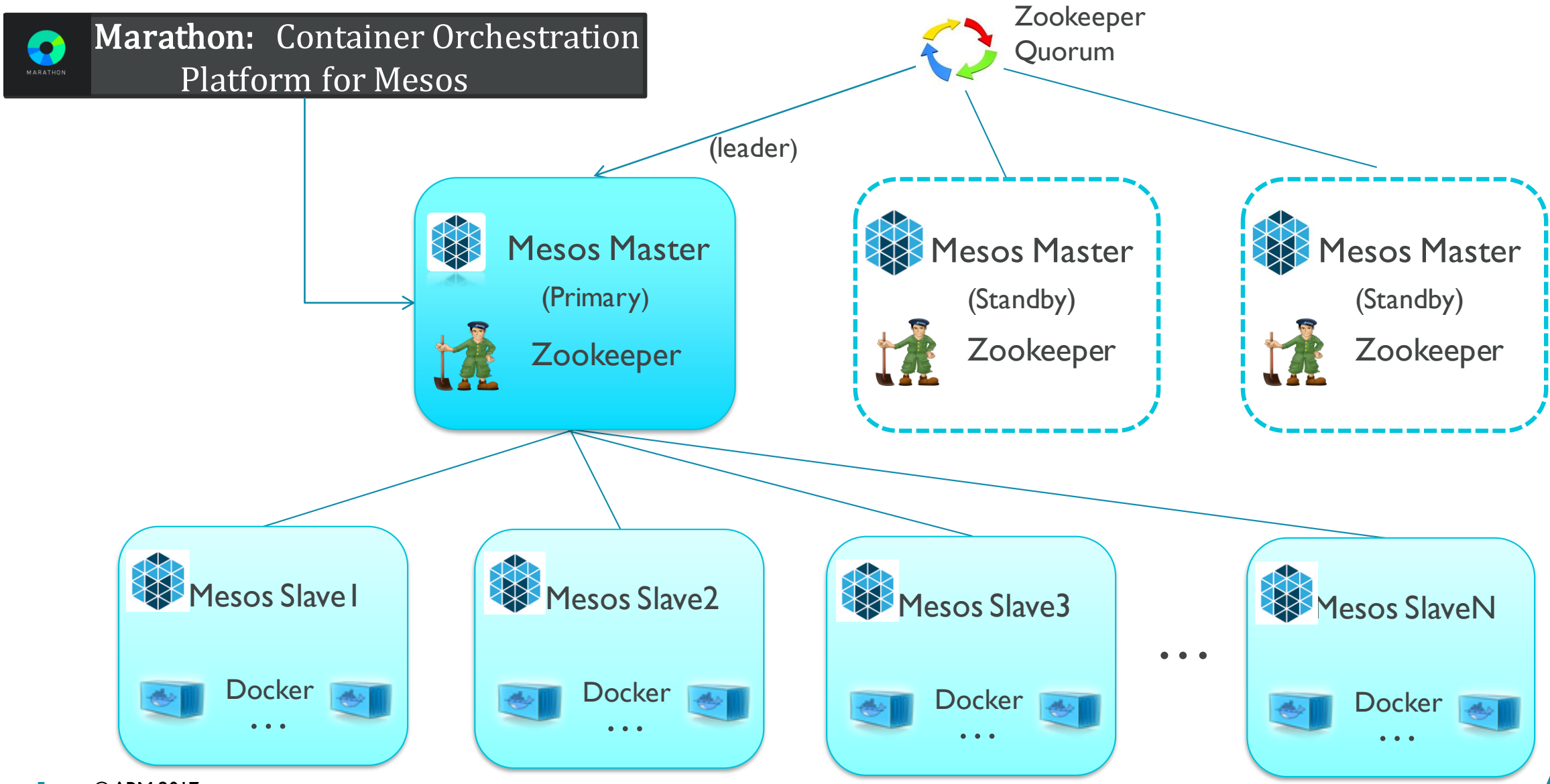
  AArch64 cores(Cortex-A53 & Cortex-A57) were announced on 30 October 2012.

- Distros supporting AArch64

Why run Mesos on AArch64?

- ARM Solution Benefits
  - Lower cost
  - Multi cores and instructions executed in parallel
  - Allow power headroom to be specialized
  - Power efficiency

A Mesos-Marathon cluster looks like …



ARM

How to Set up Mesos/Marathon on AArch64

- Build Mesos/Marathon packages for AArch64 on Ubuntu

  - The build steps on AArch64 are similar with those on other platforms. Except:

Upgrade Mesos 3rd party modules for 

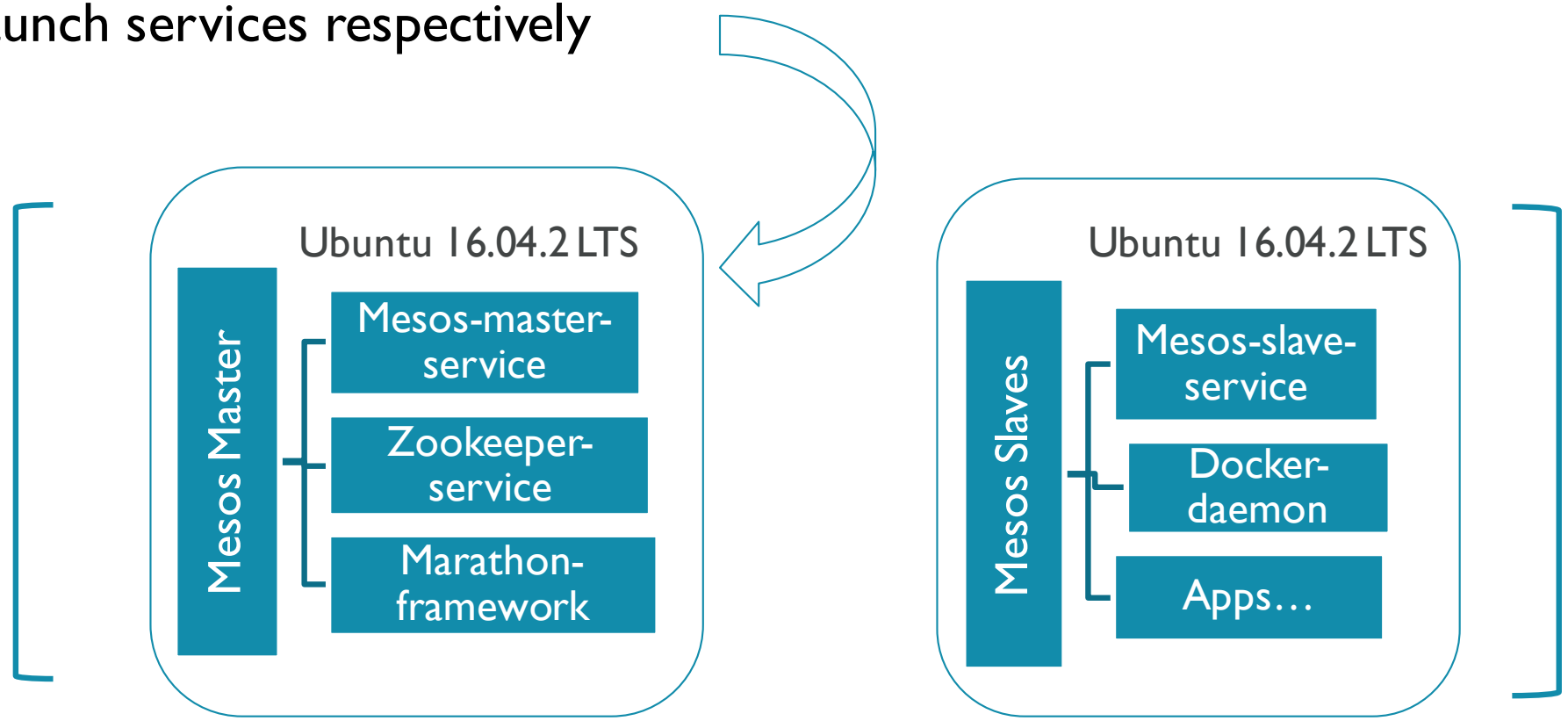
3rd party	No AArch64 Support	AArch64 Support
Protobuf	2.5.0	2.6.1 +
ZooKeeper	3.4.7	3.4.8 +
Libev	4.15	4.22 +
Leveldb	1.4	1.19 +

Fix build issues for 

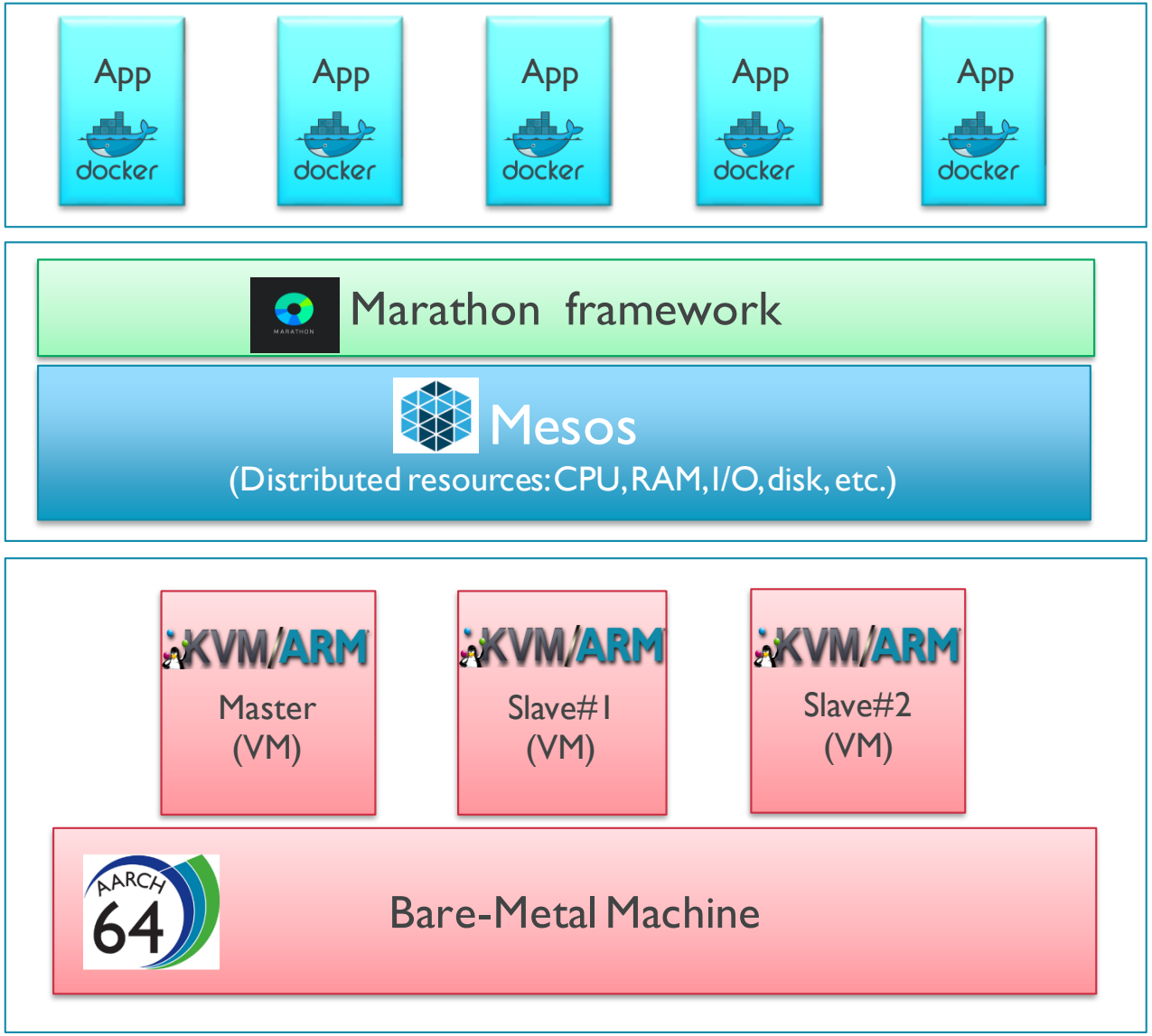
	Glog0.3.3
Error log:	Configure error: cannot guess build type
Solution:	➢ update 'config.guess' for AArch64

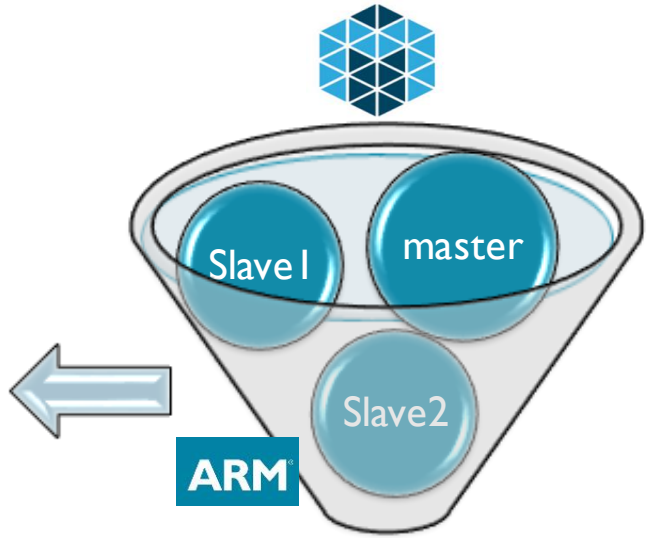
How to Set up Mesos/Marathon on AArch64

- Install/Configure Mesos agent/zookeeper/marathon on Master
- Install/Configure Mesos agent/docker on Slaves
- Launch services respectively



Mesos/Marathon Cluster in Demo

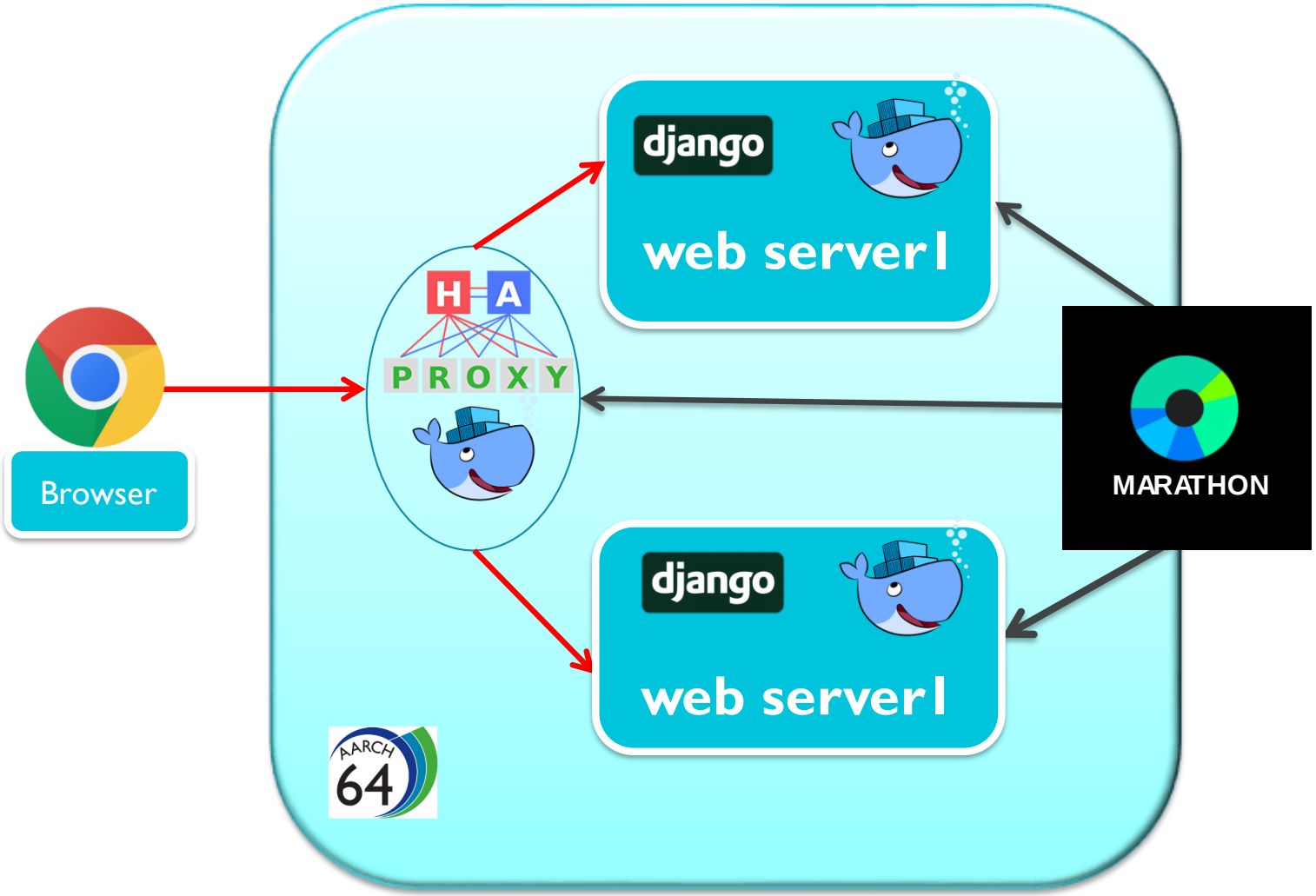




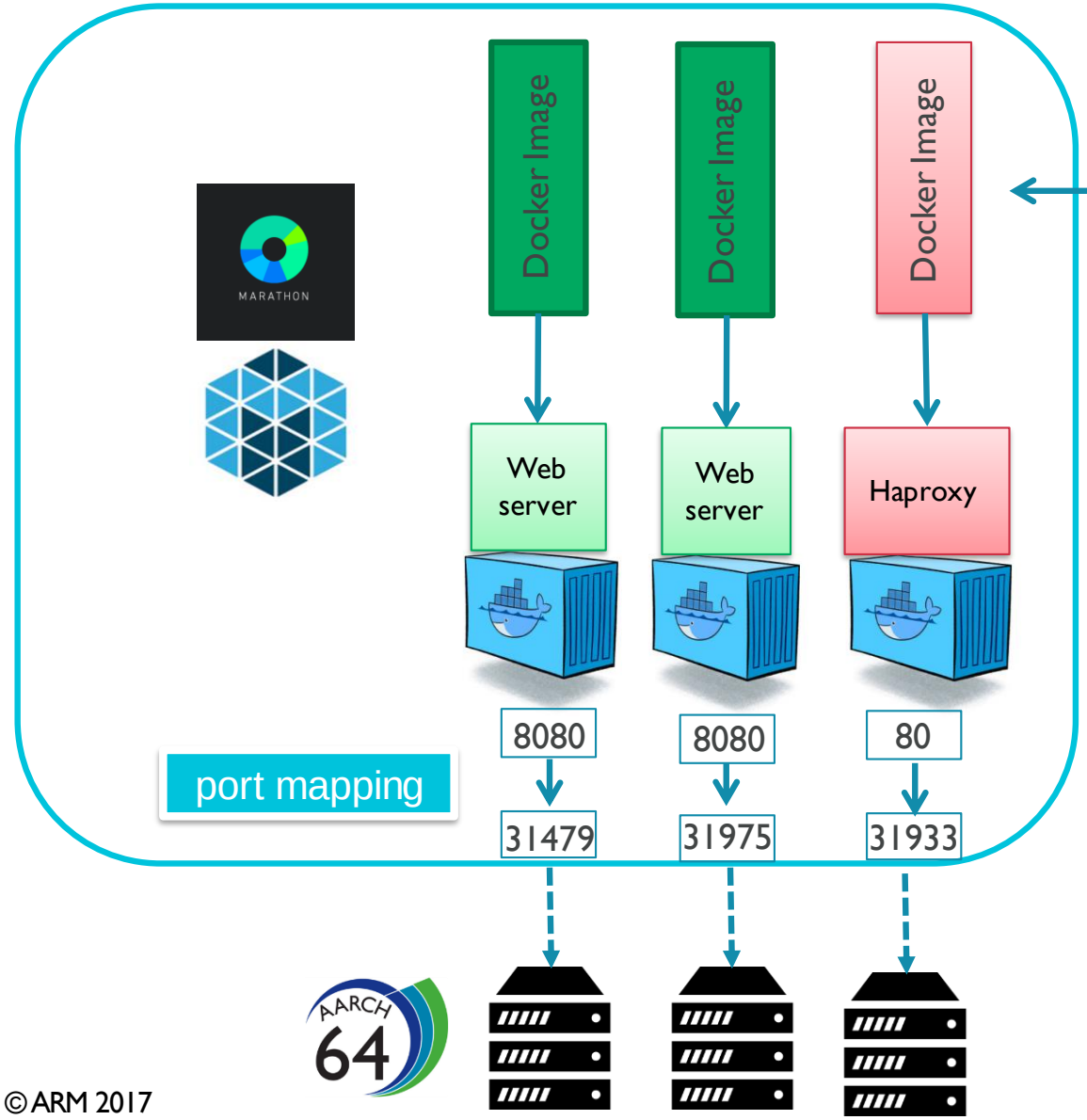
a single pool of resources

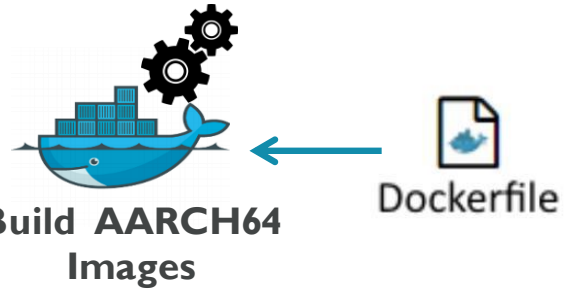
ARM

Containerized Applications on Mesos/Marathon Cluster



Containerized applications on Mesos/Marathon Cluster






```
linux@mesos-slave1:~$ docker images
REPOSITORY          TAG                 IMAGE ID
django/aarch64      latest              d67a9575f213
haproxy/aarch64     latest              d9650db06730
ubuntu/developer    latest              c100f160eb7b
ubuntu/base         latest              d4f0d3974bbb
ubuntu/core         latest              9b0da824c289
aarch64/redis       latest              d9cb46a450ac
```


ARM

Build AArch64 Docker Images and Customize Initialized Scripts

Build Django images on AArch64

Dockerfile

```
FROM ubuntu/developer
RUN apt-get update \
&& apt-get -y dist-upgrade \
&& apt-get install -y  python3-pip
RUN pip3 install --upgrade pip

#Install Django
RUN pip3 install Django
WORKDIR /home/ent-user
COPY django-show.sh ./
COPY prj_urls.py ./
COPY show_views.py ./
COPY demo-show.html ./
COPY prj_context_processor.py ./
RUN mkdir -p /home/ent-user/static-data
ADD static-data /home/ent-user/static-data

#Web server default port
ENV EX_PORT 12580
# Run Initialized Script
CMD ["/bin/bash", "/home/ent-user/django-
show.sh" ]
```

Customize Initialized Scripts

Scripts

```
# Start project
django-admin startproject $PRJ_NAME
cd $PRJ_DIRpython manage.py startapp $APP_NAME

#Config Django
sed -i '28a ALLOWED_HOSTS = ["*"]'
$PRJ_DIR/$PRJ_NAME/settings.py
sed -i '28d' $PRJ_DIR/$PRJ_NAME/settings.py
sed -i '64a  "demosite.context_processor.ip_address", '
$PRJ_DIR/$PRJ_NAME/settings.py
sed -i '39a "show",' $PRJ_DIR/$PRJ_NAME/settings.py

#Copy Web Server Configuration and static files
cp -af $TOP/show_views.py $APP_DIR/views.py
cp -af $TOP/prj_urls.py $PRJ_DIR/$PRJ_NAME/urls.py
cp -af $TOP/prj_context_processor.py
$PRJ_DIR/$PRJ_NAME/context_processor.py
cd $APP_DIR
mkdir static templates
cp -af $TOP/demo-show.html $APP_DIR/templates/demo-show.html
cp -af $TOP/static-data/* $APP_DIR/static/

# Start server
cd $PRJ_DIR
python manage.py runserver 0:$EX_PORT
```

Build AArch64 Docker Images and Customize Initialized Scripts

Build HAproxy images on AArch64

Dockerfile

```
FROM ubuntu/developer

#Install haproxy
RUN apt-get update \
    && apt-get install -y haproxy

# Default Ports for Haproxy
# backend servers
ENV SRV1_PORT 31000
ENV SRV2_PORT 32000

WORKDIR /home/ent-user

COPY haproxy.cfg ./

# Run Initialized Script
COPY haproxy-run.sh ./

CMD ["/bin/bash", "/home/ent-user/haproxy-
run.sh"]
```

Customize Initialized Scripts

Scripts

```
#!/bin/bash
TOP=`pwd`


#Config HAproxy
mkdir /run/haproxy
cp -af haproxy.cfg /etc/haproxy/haproxy.cfg
sed -i '/$/a ENABLED=1' /etc/default/haproxy

#haproxy backend servers Ports
sed -i "/#mytag/ a server server1  mesos-
slave1.shanghai.arm.com:${SRV1_PORT}\nserver server2
mesos-slave2.shanghai.arm.com:${SRV2_PORT}"
/etc/haproxy/haproxy.cfg

haproxy -f /etc/haproxy/haproxy.cfg
service haproxy start
```

ARM