SHIP IT! CONTAINERIZING YOUR KDCS

2015 AFS & Kerberos
Best Practices Workshop

BIRCHBOX™

WHO AM I?

- Thomas Kula
- Systems Engineer with Birchbox
- @thomaskula
- @etc_kula
- github.com/{bbox-,}kula

WHAT IS BIRCHBOX?

A leading retailer **changing** the way consumers **shop** for beauty and grooming products, offering **try** through our monthly sample subscription boxes and **buy** through our online, physical and pop-up stores

WHAT IS BIRCHBOX?

Code Well Groomed

WHY AM I HERE?

- I like Kerberos
- Using since 2001
- Managing since 2005
- When hired at a company looking for a good IDM solution, I said “I know what you need…”

SO, CONTAINERS...

Are you talking about containers because your company sends hundreds of thousands of boxes each month?

TWO MINUTES ON CONTAINERS

- Really nothing more than namespaces
- Much like chroot turns `/some/path` into / for a process

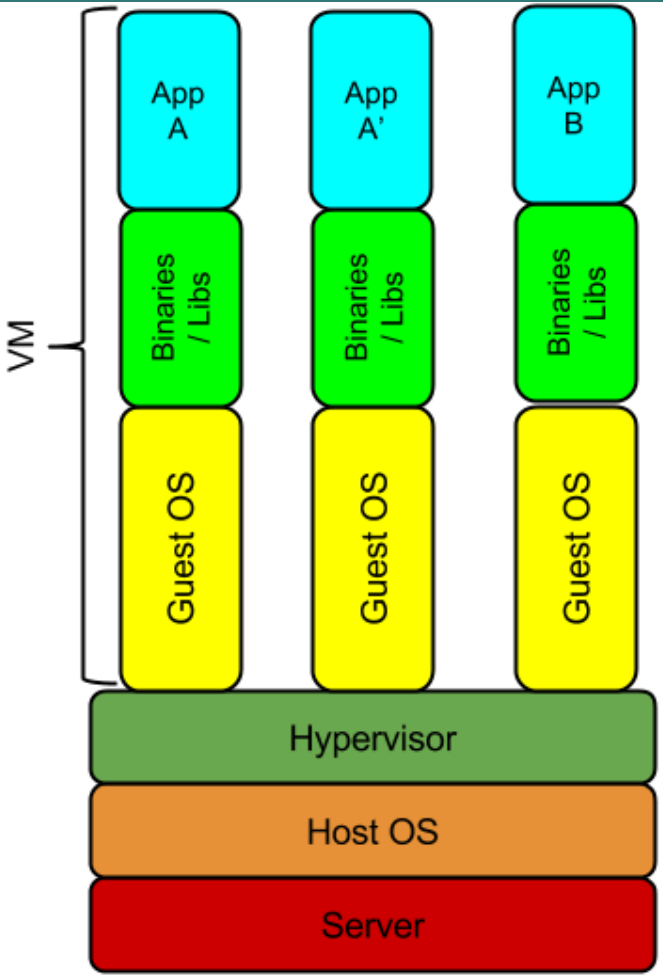
TWO MINUTES ON CONTAINERS

- Support for this has been in the Linux kernel for a long time
- Extend the idea to PIDs, {U,G} IDs, network namespaces, etc.
- Not really a new idea
- Other OSes have similar ideas

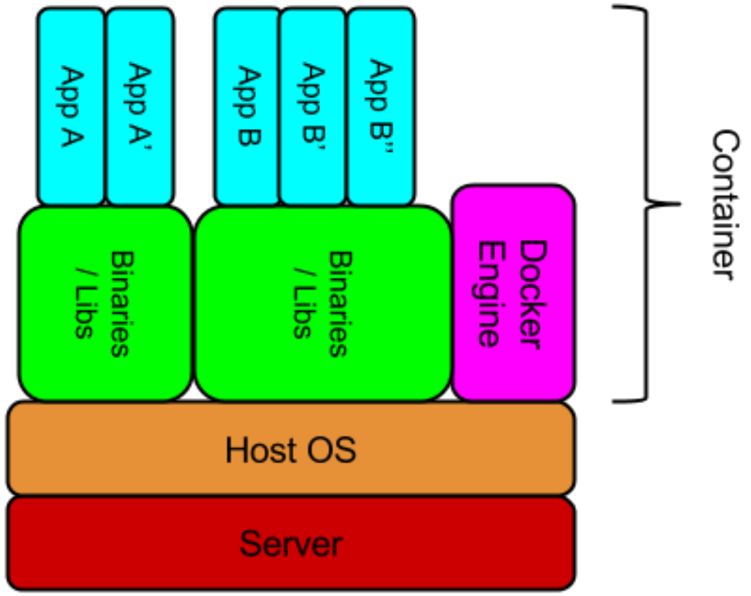
TWO MINUTES ON CONTAINERS

- Allows you to run a process
    - That thinks it is PID 1
    - Has its own network space
    - Distinguished / hierarchy
- The host OS translates that

TWO MINUTES ON CONTAINERS



- Thinking of it as "process virtualization" isn't too far off....



2015 AFS & Kerberos
Best Practices Workshop

DOCKER

- Namespaces are an old idea
- But in the past few years, Docker has become the dominant namespace solution for Linux

DOCKER

- Simply tooling, conventions and infrastructure built on top of existing namespace support
- Everything you do in Docker you can do natively
    - If you put enough effort into it

DOCKER

- Made it easy to
  - Build a container image
  - Distribute it
  - Run an application with mapped resources
  - Run it isolated from anything outside of the container

DOCKER AND KDCS

- Some of the conventions in the Docker world don’t mesh well with the Kerberos

THE DOCKER MINDSET

- Run an arbitrary number of instances
- They come and go at will
- Find other resources and announce yourself via some sort of service discovery

THE DOCKER MINDSET

- This works well for applications and systems designed around this mindset

THE KERBEROS MINDSET

- There are a small number of Kerberos servers
- They rarely change
- They rarely move
- They **are**

RECONCILIATION

- These are not insurmountable differences
- You probably won't be running your KDCs in a full Docker mindset
- But many of the tools have value

BUT WHY?

- Mostly because someone is bound to ask “Can you run your KDC in a container?”
- We’re shifting to containers pretty heavily
- Some of the tools/techniques are useful

A CONCRETE EXAMPLE

- Spinning up a test realm
- Using Docker Compose
  - Define a set of containers to run as a unit
  - Define links between them

MAKING A CONTAINER

```
FROM debian:jessie
MAINTAINER Thomas Kula <kula@birchbox.com>
LABEL Description="A base image with krb5-user
installed"
RUN apt-get update
RUN DEBIAN_FRONTEND='noninteractive' apt-get
install -y krb5-user
RUN useradd -m kula
```

MAKING A CONTAINER

```
$ time docker build -t afskbpw2015/krb-client:latest .
Sending build context to Docker daemon 2.048 kB
Sending build context to Docker daemon
Step 0 : FROM debian:jessie
 ---> bf84c1d84a8f
Step 1 : MAINTAINER Thomas Kula <kula@birchbox.com>
 ---> Running in 59c7c358fda3
 ---> 64cfdf2670b5
Removing intermediate container 59c7c358fda3
Step 2 : LABEL Description "A base image with krb5-user
installed"
 ---> Running in a8c38e9f48b9
 ---> a9141046fd82
Removing intermediate container a8c38e9f48b9
```

MAKING A CONTAINER

```
Step 3 : RUN apt-get update
 ---> Running in 6943e35706f6
Get:1 http://security.debian.org jessie/updates
InRelease [63.1 kB]
…
Removing intermediate container 6943e35706f6
Step 4 : RUN DEBIAN_FRONTEND='noninteractive' apt-get
install -y krb5-user
 ---> Running in 8e417bf1e66e
Reading package lists...
Building dependency tree...
The following extra packages will be installed:
  bind9-host geoip-database krb5-config krb5-locales
libalgorithm-c3-perl
…
```

MAKING A CONTAINER

```
…
 ---> 43f535d01c13
Removing intermediate container 8e417bf1e66e
Step 5 : RUN useradd -m kula
 ---> Running in 6d281a1ffc9f
 ---> 638606b4c5a2
Removing intermediate container 6d281a1ffc9f
Successfully built 638606b4c5a2

real    1m13.607s
user    0m0.032s
sys     0m0.040s
```

MAKING A CONTAINER

```
REPOSITORY                      TAG
afskbpw2015/krb-client          latest
IMAGE ID            CREATED                 VIRTUAL SIZE
638606b4c5a2        4 minutes ago           191.4 MB
```

- If you work at it, that image size can get much, much smaller
- I was just lazy, and based it off of a stock Debian Jessie container

MAKING A CONTAINER

```
FROM afskbpw2015/krb-client:latest
MAINTAINER Thomas Kula <kula@birchbox.com>
LABEL Description="A base kdc image, no kadmind"
RUN DEBIAN_FRONTEND='noninteractive' apt-get install -y
krb5-kdc
```

MAKING A CONTAINER

```
$ time docker build -t afskbpw2015/kdc:latest .
Sending build context to Docker daemon 2.048 kB
Sending build context to Docker daemon
Step 0 : FROM afskbpw2015/krb-client:latest
 ---> 638606b4c5a2
…
Package installation occurs here
…
 ---> 076b635c0ca8
Removing intermediate container adfcab757988
Successfully built 076b635c0ca8

real    0m20.269s
user    0m0.012s
sys     0m0.020s
```

WHAT DOES THIS GET ME?

- A series of containers
  - Immutable
  - I can shove around
  - Launch with a defined set of resources

RUNNING THE TEST CLUSTER

```
kdc0:
  image: afskbpw2015/kdc-admin
  hostname: kdc0.krb.example.com
  container_name: kdc0
  entrypoint:
    - '/usr/sbin/krb5kdc'
    - '-n'
  links:
    - dns0:dns0.example.com
  dns: 172.17.42.1
  environment:
    - DNSDOCK_NAME=kdc0
    - DNSDOCK_IMAGE=krb
  volumes:
    - /home/kula/afskbpw2015/kdc-cluster/state/common/krb5.conf:/etc/krb5.conf:ro
    - /home/kula/afskbpw2015/kdc-cluster/state/kdc0/etc/krb5.keytab:/etc/krb5.keytab
    - /home/kula/afskbpw2015/kdc-cluster/state/kdc0/etc/krb5kdc:/etc/krb5kdc
    - /home/kula/afskbpw2015/kdc-cluster/state/kdc0/var/lib/krb5kdc:/var/lib/krb5kdc
```

RUNNING THE TEST CLUSTER

```
kdc0-kadmin:
  image: afskbpw2015/kdc-admin
  container_name: kdc0-kadmin
  entrypoint:
    - '/usr/sbin/kadmind'
    - '-nofork'
  volumes_from:
    - kdc0
  net: 'container:kdc0'
```

RUNNING THE TEST CLUSTER

```
kdc1-kpropd:
  image: afskbpw2015/kdc-admin
  container_name: kdc1-kpropd
  entrypoint:
    - '/usr/sbin/kpropd'
    - '-d'
  volumes_from:
    - kdc1
  net: 'container:kdc1'
```

RUNNING THE TEST CLUSTER

```
$ docker-compose up
Creating dns0...
Creating krb-client...
Creating kdc1...
Creating kdc1-kpropd...
Creating kdc0...
Creating kdc0-kadmin...
Attaching to dns0, krb-client, kdc1, kdc1-kpropd, kdc0, kdc0-
kadmin
kdc1-kpropd_1 | Incremental propagation enabled
kdc1_1        | krb5kdc: starting...
kdc0-kadmin_1 | kadmind: create IPROP svc (PROG=100423, VERS=1)
kdc0-kadmin_1 | kadmind: starting...
kdc0_1        | krb5kdc: starting...
```

RUNNING THE TEST CLUSTER

```
kdc1-kpropd_1 | Initializing kadm5 as client kiprop/
kdc1.krb.example.com@EXAMPLE.COM
kdc1-kpropd_1 | kadm5 initialization succeeded
kdc1-kpropd_1 | Calling iprop_get_updates_1()
kdc0-kadmin_1 | iprop_get_updates_1: start, last_sno=14
kdc0-kadmin_1 | iprop_get_updates_1: clprinc=`kiprop/
kdc1.krb.example.com@EXAMPLE.COM'
kdc0-kadmin_1 |         svcprinc=`kiprop/
kdc0.krb.example.com@EXAMPLE.COM'
kdc0-kadmin_1 | iprop_get_updates_1: request UPDATE_NIL; Incoming
SerialNo=14; Outgoing SerialNo=N/A success
kdc0-kadmin_1 |         clprinc=`kiprop/
kdc1.krb.example.com@EXAMPLE.COM'
kdc0-kadmin_1 |         svcprinc=`kiprop/
kdc0.krb.example.com@EXAMPLE.COM'
kdc1-kpropd_1 | KDC is synchronized with master.
kdc1-kpropd_1 | Waiting for 15 seconds before checking for
updates again
```

RUNNING THE TEST CLUSTER

```
volumes:
    - /home/kula/afskbpw2015/kdc-cluster/state/common/krb5.conf:/etc/krb5.conf:ro
    - /home/kula/afskbpw2015/kdc-cluster/state/kdc0/etc/krb5.keytab:/etc/krb5.keytab
    - /home/kula/afskbpw2015/kdc-cluster/state/kdc0/etc/krb5kdc:/etc/krb5kdc
    - /home/kula/afskbpw2015/kdc-cluster/state/kdc0/var/lib/krb5kdc:/var/lib/krb5kdc
```

- With volumes, persistent data is saved
- Scribble all you want anywhere else
- Restart the container, it vanishes

WHAT ABOUT PRODUCTION?

- Powerful draws
  - Immutable containers
  - Restart and **know** what is in the process filesystem
  - Isolation
- How to best merge the two worlds remains to be seen

TANTALIZING FUTURE

- Docker is pulling out core plumbing bits into separate projects
  - libnetwork
  - Open Container Initiative
- As those pick up features, they get included in Docker

TANTALIZING FUTURE

- The most appealing
  - Running a root process in a container as a non-privileged user at the host level
  - runc (an Open Container project) **can** do this
  - But it's a very fast moving target and I haven't figured out all the knobs yet….

QUESTIONS?

`github.com/birchbox/afskbpw2015/`