A talk for BBLISA, 2014-10-08

Colin Walters, Platform Engineering, Red Hat, Inc.

Why is Colin here?

- Free Software
- Fun working in a global community
- Value of a subscription: Red Hat Enteprise Linux

Essential Atomic ingredients

- Host distribution
- Docker: Linux containers made easy
- SELinux and the Linux kernel: Container isolation, storage, namespaces, cgroups
- systemd: Making it easy to manage the base system
- rpm + OSTree: Compose and update the host system
- Kubernetes: Orchestrate containers

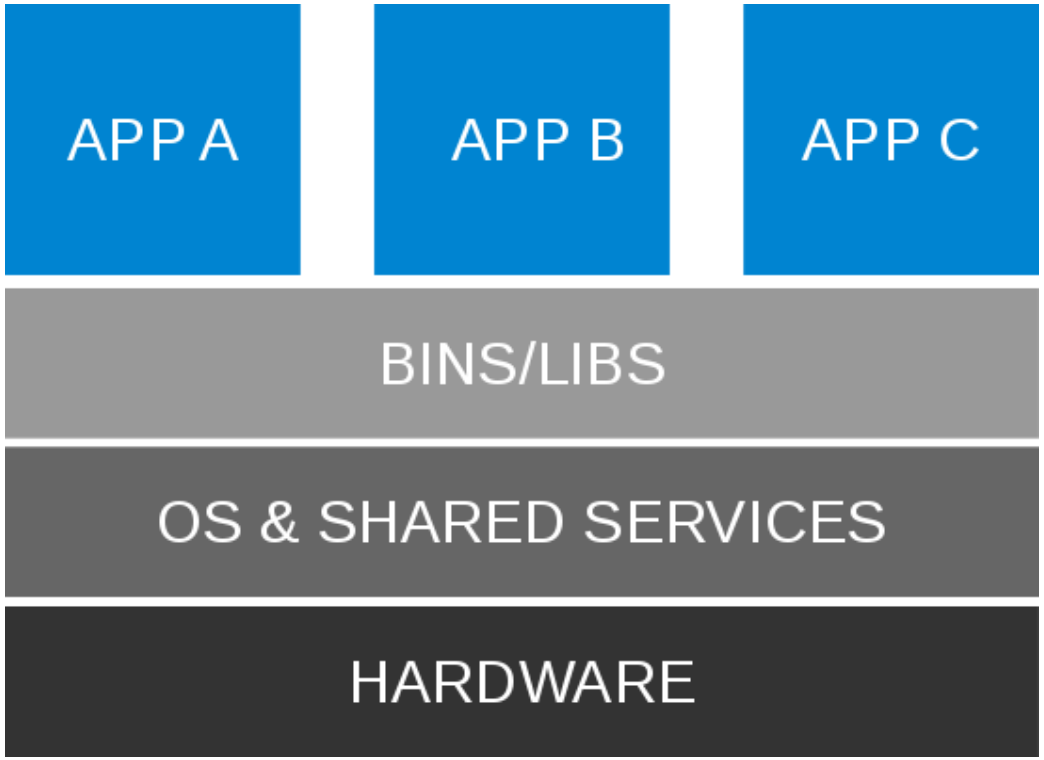
Project Atomic is not a distribution. It is a *pattern* around a set of upstream projects that can be applied to a distribution.

- CentOS
- Fedora
- Red Hat Enterprise Linux

Deliverables

- Atomic Host + regularly updated tree
- Docker Base Image (also updated, though only for critical errata)
- Additional packages to create layered images
- Docker Registry? Maybe.

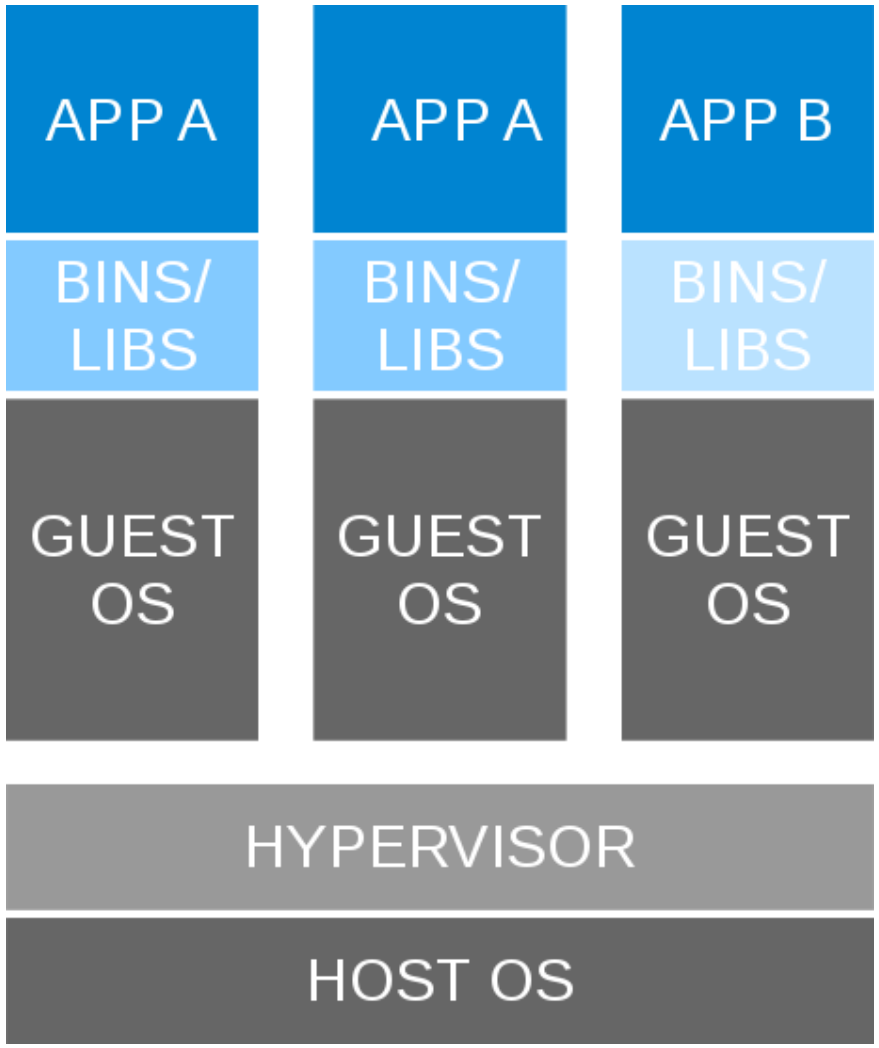
Background: Traditional



Background: Traditional

- Single userspace runtime (SCLs, per-user builds)
- Environment and life cycle defined by host OS
- Trend to isolate apps on hardware level
- Stable, long maintenance, few updates, hardware-centric
- Resources generally underutilized

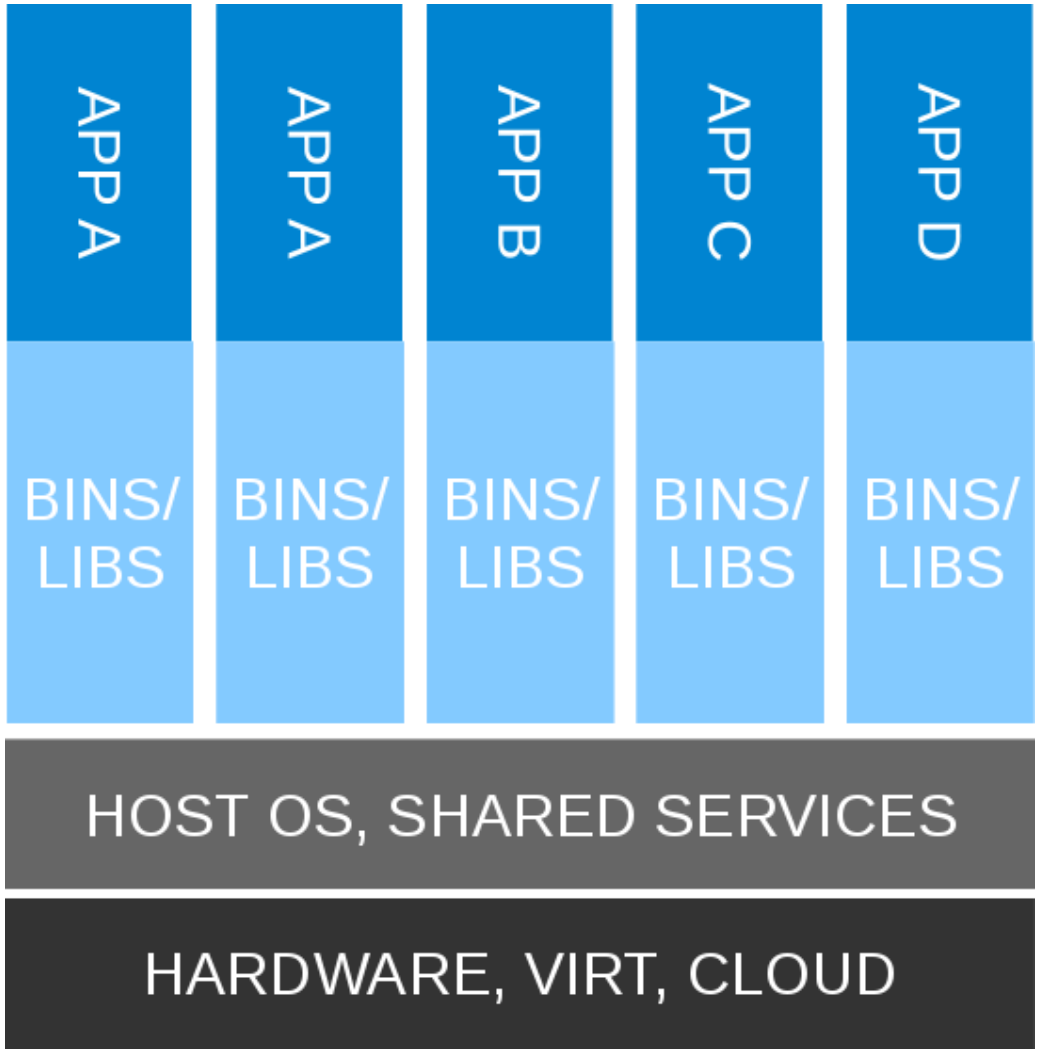
Background: Virt and IaaS



Background: Virt and IaaS

- App per VM
- Guest and lifecycle tied to VM
- Redundancy and overhead (multiple kernels, logging)
- Complex management / orchestration

Containerized



Containerized

- Applications are still isolated, but shared kernel
- Easily embed dependencies with apps
- Standard host services (SSH)
- Common logging, config mgmt, orchestration

Containerize all the things?

- Virt and containers are complementary
- Virt is a stronger security boundary
- No, you can't run Windows as a Linux container

Docker: What the Internet says

Docker: Linux containers made easy

- Union filesystem / snapshots
- Dockerfile: very easy to do something
- Port mapping
- Push/pull content (and the Hub)
- Most popular project on Github

Docker: The ugly

- Image security updates
- Running code as root

Docker: Demo time!

Classic problem: I have two web apps I want to run on the same host.

Assemble a Docker image with apache, from CentOS RPMs

```
$ cat > Dockerfile << EOF
FROM centos
RUN yum -y upgrade && yum -y install httpd && yum clean all
EXPOSE 80
CMD ["/usr/sbin/httpd", "-D", "FOREGROUND"]
EOF
$ sudo docker build -t cgwalters/apache .
```

Run it

```
$ sudo docker run -d -P cgwalters/apache
$ sudo docker ps -a
$ curl http://127.0.0.1:${port}
```

Container thinks it's port 80

Create derived images

```
$ cat > Dockerfile << EOF
FROM cgwalters/apache
echo "<body>Container one</body>" > /var/www/html/index.html
EOF
```

Run them both

```
# sudo docker run -d cgwalters/httpd1
# sudo docker run -d cgwalters/httpd2
# sudo docker ps
$ curl http://127.0.0.1:${firstport}
$ curl http://127.0.0.1:${secondport}
```

Isolation via namespaces

```
$ sudo docker run -t -i centos bash
# ps auxwf
# ip link
# mount
# hostname
```

Namespaces + cgroups in Linux kernel

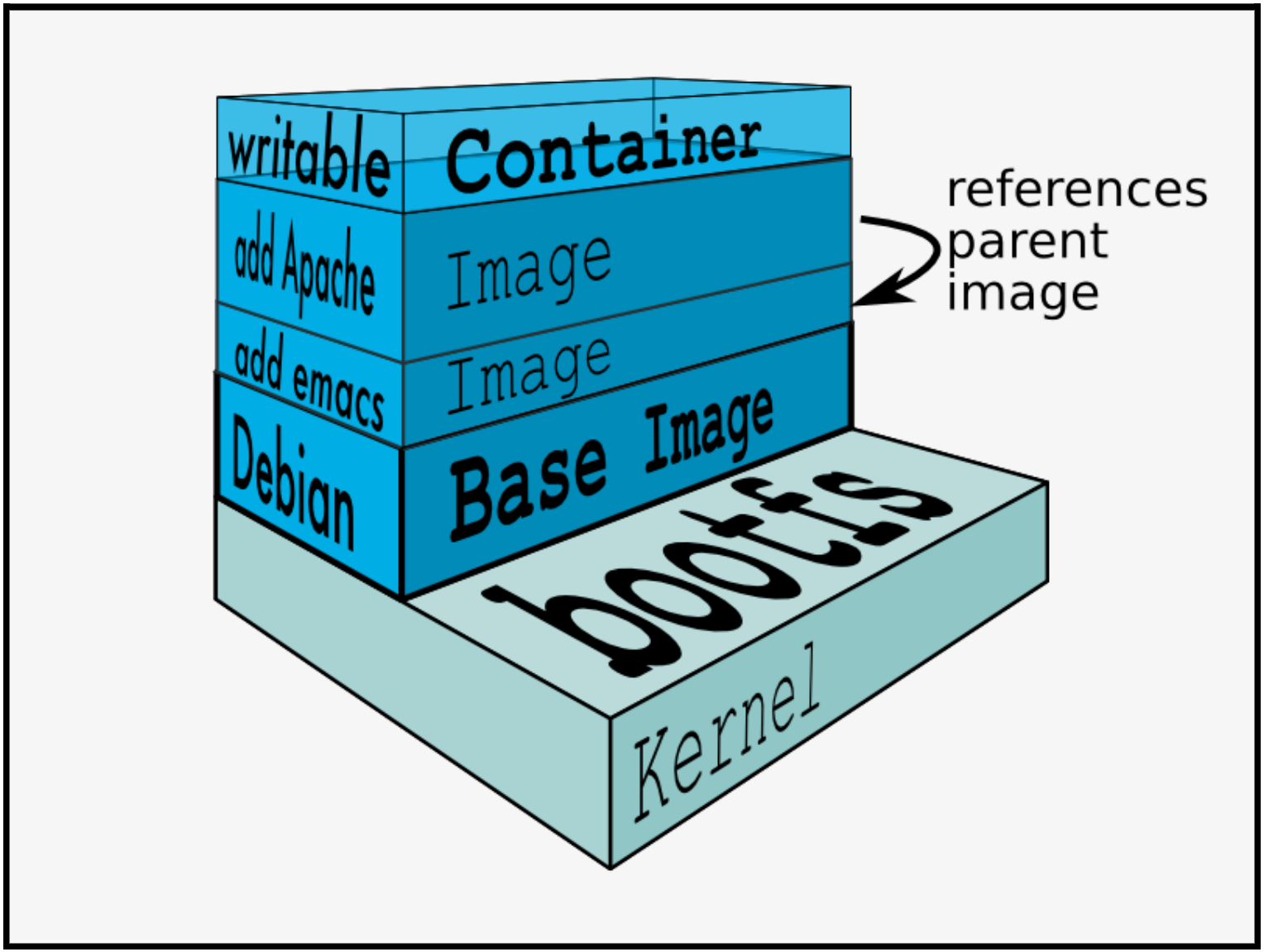
CLONE_NEWPID, CLONE_NEWNET, CLONE_...

”Containers don’t contain”

Blog entry

TL;DR: You are running code as root, kernel is large attack surface.

The Docker union filesystem



The Docker union filesystem

- Image: read-only, created locally or pulled and cached
- docker run creates a container from an image, *fast*
- Multiple backends: AUFS (out-of-tree kernel patch), BTRFS, Device Mapper (LVM)
- Future: overlayfs

Where's my data?

Written to the container by default (do not do this in production)

Do use docker run --rm

- Use Docker Volumes
- Bind mount to host storage: -v /hostpath:/containerpath
- Write to network data stores (Cassandra, Swift, MariaDB) and use remote logging

Atomic: default dedicated Docker storage

Docker storage scalability

```
# lvm lvs
LV          VG       Attr       LSize   Pool Origin Data%  Move Log Cpy%Sync Convert
docker-data atomicos -wi-ao----  11.54g
docker-meta atomicos -wi-ao----   1.00g
root        atomicos -wi-ao----   3.13g
swap        atomicos -wi-ao---- 128.00m
# docker info |grep Space
 Data Space Used: 594.6 Mb
 Data Space Total: 11820.0 Mb
 Metadata Space Used: 0.5 Mb
 Metadata Space Total: 1024.0 Mb
#
```

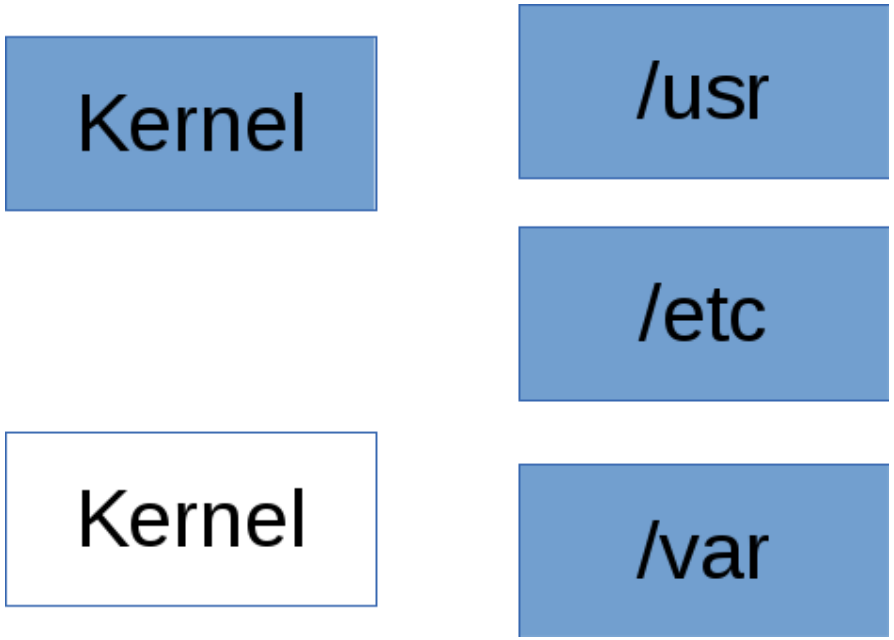
Host not just kernel+docker

- Kubernetes
- systemd
- OpenSSH
- Storage: NFS, Gluster, ...
- cloud-init
- SSSD
- Networking (NetworkManager, future: +Open vSwitch)
- Core dump collection (abrt/systemd-coredump)

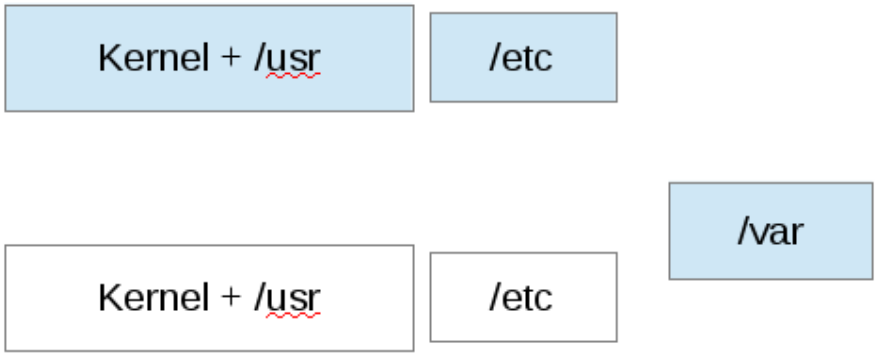
RPM+OSTree: Atomic host OS upgrades+rollback

- RPM: It's a UNIX system! I know this!
- OSTree: Atomically swap between bootable filesystem trees
- rpm-ostree (aka /usr/bin/atomic): Both client and server-side "compose" tooling

Traditional: yum/apt-get, etc.



OSTree (as used by rpm-ostree)



OSTree filesystem model

- /usr is a read-only bind mount. **Always.**
- /etc is "rebased" on upgrades - apply config diff to new /etc
- /var is untouched
- /home -> /var/home

Server: rpm-ostree compose tree

Client: rpm-ostree upgrade

- Input yum repositories + package set
- Kind of like: yum --installroot + git commit
- Unlike git, OSTree handles: uid+gid, xattrs (SELinux)
- Boot into ***new*** chroot
- OSTree also knows about bootloader, atomic swap

Demo time! Ctrl-C an OS upgrade

- Very satisfying

Multiple bootable roots, and starting an upgrade

```
# atomic status
  TIMESTAMP (UTC)         ID             OSNAME               REFSPEC
* 2014-10-07 19:29:55     9e8fd0f4bd     rhel-atomic-host     brew:rhel-atomic-host/7/
  2014-09-29 22:03:42     80986d2569     rhel-atomic-host     brew:rhel-atomic-host/7/
# atomic upgrade
Receiving objects: 71% (25/35) 2.4 MB
^C
```

Completing an upgrade

```
Copying /etc changes: 13 modified, 0 removed, 18 added
Transaction complete; bootconfig swap: yes deployment count change: 0)
Freed objects: 79.8 MB
Changed:
  NetworkManager-1:0.9.9.1-26.git20140326.4dba720.l7_0.x86_64
  NetworkManager-glib-1:0.9.9.1-26.git20140326.4dba720.l7_0.x86_64
  cloud-init-0.7.5-1.l7_0.x86_64
  dhclient-12:4.2.5-27.l7_0.1.x86_64
  dhcp-common-12:4.2.5-27.l7_0.1.x86_64
  dhcp-libs-12:4.2.5-27.l7_0.1.x86_64
  docker-1.2.0-17.l7.x86_64
  kernel-3.10.0-123.8.1.l7.x86_64
Removed:
  PackageKit-glib-0.8.9-11.l7.x86_64
  accountsservice-0.6.35-7.l7.x86_64
  accountsservice-libs-0.6.35-7.l7.x86_64
  make-1:3.82-21.l7.x86_64
```

Pre-reboot state

```
# atomic status
  TIMESTAMP (UTC)         ID             OSNAME               REFSPEC
  2014-10-08 12:23:35     ed78ff0b4d     rhel-atomic-host     brew:rhel-atomic-host/7/
* 2014-09-29 22:03:42     80986d2569     rhel-atomic-host     brew:rhel-atomic-host/7/
```

Reboot, something went wrong

```
# atomic rollback
```

Completely safe (and also atomic!) swap of bootloader entries.

Can I use Docker+Kubernetes via regular packages?

Yes.

For production, optimize storage like Atomic

Kubernetes

Container Farms

Even the Google Guys Say It's A Crappy Name

Where does Kubernetes Fit?

- Docker operates on single hosts
- Docker operates on individual containers (links excepted)
- Kubernetes spans hosts
- Kubernetes composes applications

Kubernetes Features

- Pods - Sets of Containers which share resources
- Services
  - Non-localized network access
  - Proxy/Load Balancing
- ReplicationController - HAish

Kubernetes Architecture

- Kubelet - container management agent
- App-Server - service portal
- Etcd - Clustering, State, Communications
- Kubecfg - Client

Contributors

- Google - Large Scale Cloud
  - Robustness by fault tolerance and scaling
- Red Hat - PaaS and Enterprise Apps
  - Robustness by point hardening
    - Individual containers
    - App level
    - Service level
    - Cluster level

TBD

- Smart Scheduling - Mesos?
- Secure Communications - Replace Etcd?
- Secure Container Environment
  - Tenant Isolation
  - Tenant secure access to containers
- Monitoring
  - Container presence, loading, thrashing
  - Container visibility
  - Network traffic
- Storage
  - Shared
  - Persistent

References and Resources

- Kubernetes: https://github.com/GoogleCloudPlatform/kubernetes
- Etcd: https://github.com/coreos/etcd

- Freenode IRC: #google-containers

Status of Project Atomic

- Much slower than expected for Fedora 21
- CentOS Atomic SIG also spinning up
- CentOS Docker Base Image is updated and widely used

Getting Involved

Project Atomic