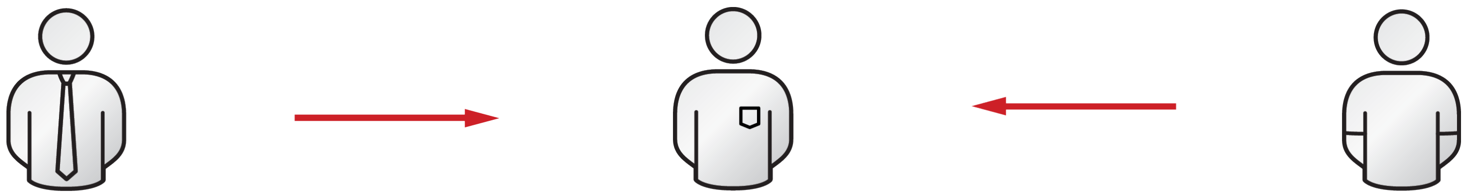
RED HAT® ENTERPRISE LINUX® ATOMIC HOST

+Docker + Cockpit

Alessandro Silva
Technical Account Manager | Plataform
26 Ago 2015

I.T. CHALLENGES



BUSINESS CHALLENGES

- Faster time to market
- Elastic, scalable, high performance
- Flexibility without lock-in, pay as you go

I.T. OPERATIONS CHALLENGES

- Increase operation efficiency
- Maximize resource utilization
- Reliable, secure, compliant

DEVELOPER CHALLENGES

- Reduce time to provision and develop, improve productivity
- Test new features and update applications faster
- Improve availability of platforms and resources

What is Red Hat Enterprise Linux Atomic Host?

- A variation of Red Hat Enterprise Linux optimized for Linux containers
- It includes the docker utility, the Docker daemon, Kubernetes, and rpm-ostree
- A Red Hat subscription permits RHEL Atomic Host software updates and yum updates in containers

RED HAT ENTERPRISE LINUX ATOMIC HOST

IT IS RED HAT ENTERPRISE LINUX

Inherits the complete hardware ecosystem, military-grade security, stability and reliability for which Red Hat Enterprise Linux is known.

OPTIMIZED FOR CONTAINERS

MINIMIZED FOOTPRINT

Minimized host environment tuned for running Linux containers while maintaining compatibility with Red Hat Enterprise Linux.

SIMPLIFIED MAINTENANCE

Atomic updating and rollback means it’s easy to deploy, update, and rollback using imaged-based technology.

ORCHESTRATION AT SCALE

Build composite applications by orchestrating multiple containers as microservices on a single host instance.

Container images and image registries

- Container images are read-only file system overlays used to create containers
- Image registries are centralized stores for container images
- RHEL Atomic Host is configured to use two public image registries:
  - registry.hub.docker.com (Docker Hub)
  - registry.access.redhat.com

How RHEL Atomic differs from stardard RHEL?

- yum is not used to install software or upgrade the system; RHEL Atomic includes new tooling that allows upgrades and rollback.
- There are only two writable directories: /etc and /var.
- The immutable bit has been set on the / directory. /usr is mounted read-only, but /usr/local is a symlink to /var/usrlocal.
- Other directories are symlinked to a writable location; for example, /home is a symlink to /var/home.
- No firewalld software is installed; only Netfilter and iptables.
- No client-side Kerberos tools are installed
- No iSCSI client nor initiator software is installed.

Managing container images

- docker search -s N name
- docker pull name
- docker load -i filename.tar
- docker images
- docker rmi name

Creating a container from an image

- # docker run -i -t name command

-i = interactive container

-t = allocate a pseudo-tty

name = name of the image to launch

command = program to launch inside the container

RHEL Atomic Host networking

- RHEL Atomic Host establishes a bridge called docker0
- A virtual interface is attached to docker0 when a container is launched
- The following command maps a RHEL Atomic Host port to a port inside the container when it is launched

```
# docker run -p HOST_PORT:CONT_PORT ...
```

Creating a simple web server container

- Start a container with a shell

```
# docker run -p 8080:80 -i -t rhel7 /bin/bash
```

- Install the necessary software in the container

```
# yum install -y httpd
```

- Create custom content

```
# echo 'Hello world!' > /var/www/html/index.html
```

Creating a simple web server container (continued)

- Confirm the web server publishes the correct content

```
# /usr/sbin/httpd -D FOREGROUND
# curl http://rhel-atomic-host.fqdn:8080
```

- For httpd, create a startup script

```
# vi /usr/sbin/my_httpd_startup.sh
# chmod 755 /usr/sbin/my_httpd_startup.sh
```

Startup script contents

```
#!/bin/bash
rm -rf /run/httpd
install -m 710 -o root -g apache -d /run/httpd
install -m 700 -o apache -g apache -d /run/httpd/htcacheclean
exec /usr/sbin/httpd -D FOREGROUND
```

Creating a container image

- Determine the container ID of the container to be saved as an image

```
# docker ps -a
```

- Create the image and assign it a tag

```
# docker commit container_id name:tag
```

- Use the new image to create a container and test it

```
# docker run name:tag …
```

- Optionally export the image to a file

```
# docker save name:tag > image-file.tar
```

THANK YOU