The Magical Future

Immutable infrastructure, containers, & the future of microservices

PRESENTED BY:

Adam Miller

Senior Software Engineer, Red Hat, Fedora Engineering

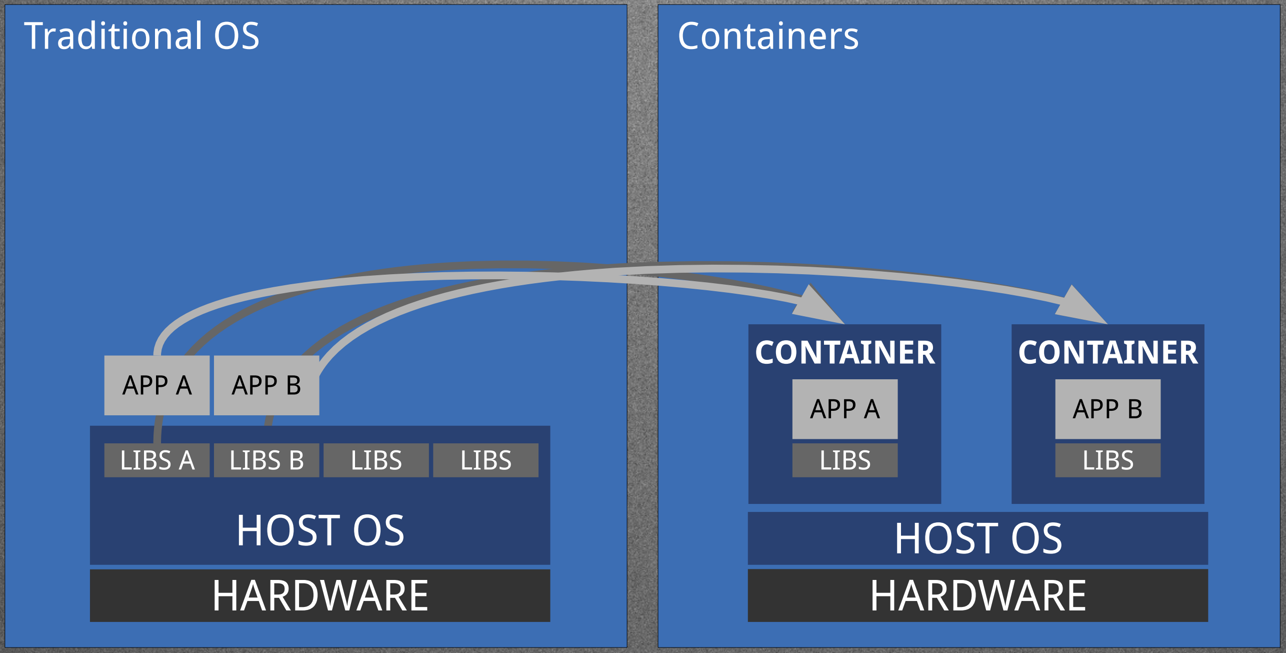
Today's Topics

- Define “containers” in the context of Linux systems
- Container Implementations in Linux
- Define “microservices”
- What Immutable Infrastructure is
  - Example of what Immutable Infrastructure deployment workflow looks like
- Fedora Cloud Atomic Host
  - How Fedora Atomic enables and enhances these concepts
- Kubernetes
  - Orchestrating the Immutable Infrastructure
- OpenShift Origin
  - Enabling the development and container building pipeline
- Q&A

Containers

What are containers?

- Operating-system-level Virtualization
  - We (the greater Linux community) like to call them “containers”
- OK, so what is Operating-system-level Virtualization?
  - The multitenant isolation of multiple user space instances or namespaces.



Containers are not new

- The concept of containers is not new
  - chroot was the original “container”, introduced in 1982
    - Unsophisticated in many ways, lacking the following:
      - COW
      - Quotas
      - I/O rate limiting
      - cpu/memory constraint
      - Network Isolation
  - Brief (not exhaustive) history of sophisticated UNIX-like container technology:
    - 2000 - FreeBSD jails
    - 2001 – Linux Vserver
    - 2004 – Solaris Zones
    - 2008 – LXC
      - This is where things start to get interesting

The Modern Linux Container is Born

- 2008 - IBM releases LinuX Containers (LXC)
  - Userspace tools to effectively wrap a chroot in kernel namespacing and cgroups
  - Provided sophisticated features the chroot lacked
- 2011 – systemd nspawn containers
  - run a command or OS in a light-weight namespace container. Like chroot, but virtualizes the file system hierarchy, process tree, various IPC subsystems, host and domain name.
- 2013 – DotCloud releases Docker (https://github.com/docker/docker)
  - Originally used LXC as the backend, introduces the Docker daemon, layered images, standard toolset for building images and a distribution method (docker registry). Later makes backend driver pluggable and replaces LXC with libcontainer as default.
- 2014 – CoreOS releases rkt (https://github.com/coreos/rkt) Rocket
  - rkt is an implementation of App Container(appc) specification and App Container Image(ACI) specification, built on top of systemd-nspawn.
  - ACI and appc aimed to be a cross-container specification to be a common ground between container implementations.

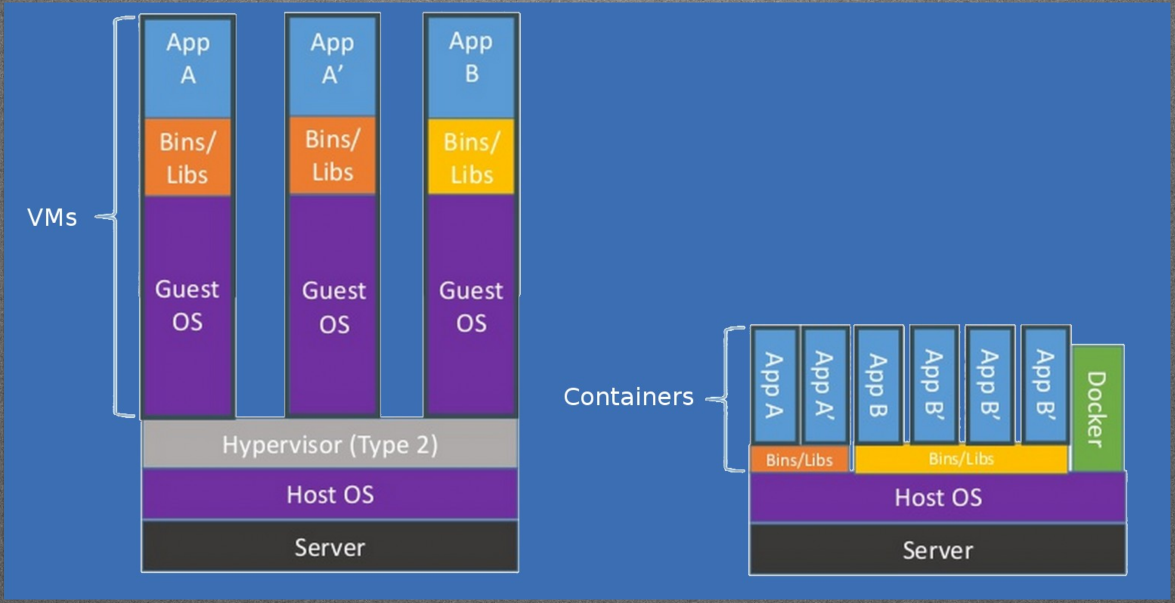
Modern Linux Container

- 2015 – Open Container Project (http://opencontainers.org/)
  - “The Open Container Initiative is a lightweight, open governance structure, to be formed under the auspices of the Linux Foundation, for the express purpose of creating open industry standards around container formats and runtime.” - http://opencontainers.org/
  - Initiative Sponsors: Apcera, AT&T, AWS, Cisco, ClusterHQ, CoreOS, Datera, Docker, EMC, Fujitsu, Google, Goldman Sachs, HP, Huawei, IBM, Intel, Joyent, Kismatic, Kyup, the Linux Foundation, Mesosphere, Microsoft, Midokura, Nutanix, Oracle, Pivotal, Polyverse, Rancher, Red Hat, Resin.io, Suse, Sysdig, Twitter, Verizon, VMWare

- 2015 – runC (http://runc.io/)
  - Stand-alone command line tool for spawning containers as per the OCP specification.
  - Containers are child processes of runC, no system daemon, can be embedded.
  - Shares technology lineage with Docker (libcontainer and others).
  - Compatible with Docker images.

Docker

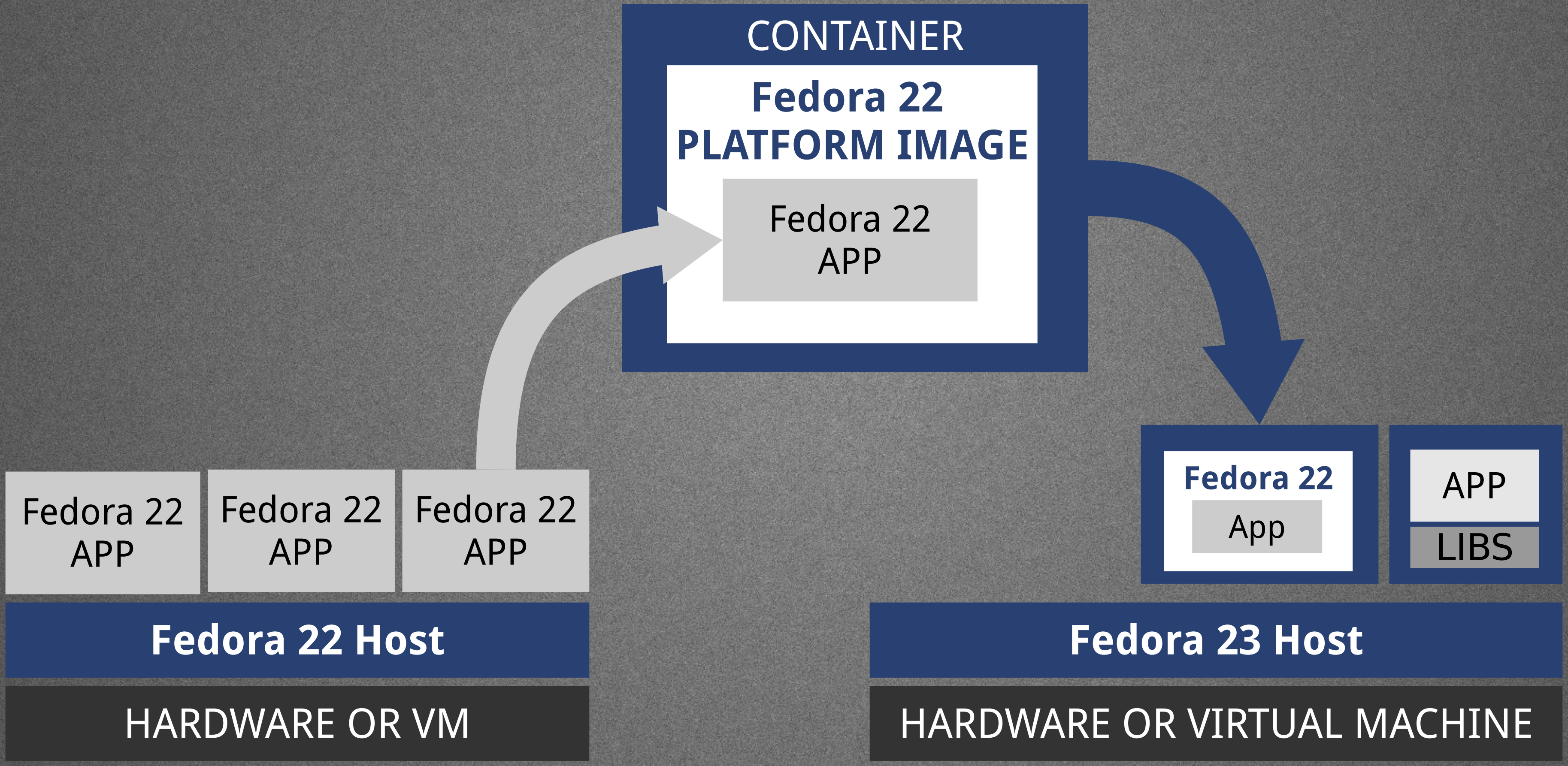
- Docker Daemon is the single point of entry, has language bindings for other clients and tooling. (Image verification)
- **Containers** are instances of **images**.
- Images are built in a standard way using Dockerfile
- Mr. SELinux (Dan Walsh) pushed SELinux support upstream to Docker.
- Pluggable backends for isolation mechanism, storage, networking, etc.



Dockerfile

```
FROM fedora
MAINTAINER http://fedoraproject.org/wiki/Cloud

RUN yum -y update && yum clean all
RUN yum -y install httpd && yum clean all
RUN echo "HTTPD" >> /var/www/html/index.html


EXPOSE 80


# Simple startup script
ADD run-httpd.sh /run-httpd.sh
RUN chmod -v +x /run-httpd.sh


CMD ["/run-httpd.sh"]
```

Container Platform Images



CONTAINER

Fedora 22 PLATFORM IMAGE

Fedora 22 APP

Fedora 22 APP

Fedora 22 APP

Fedora 22 APP

Fedora 22 Host

HARDWARE OR VM

Fedora 22

App

APP

LIBS

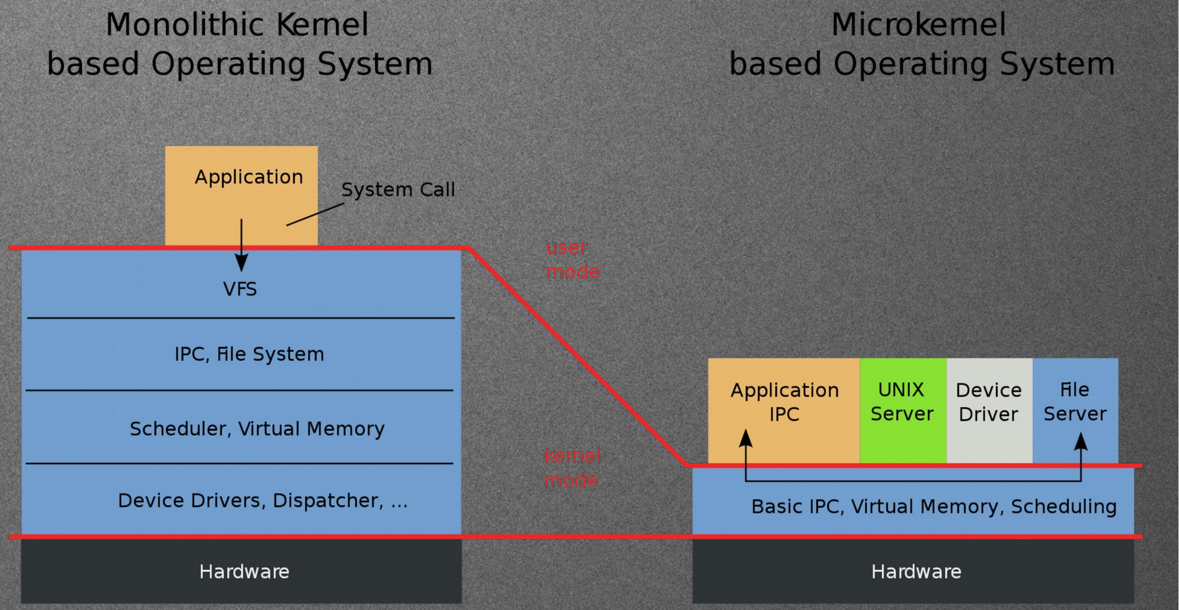
Fedora 23 Host

HARDWARE OR VIRTUAL MACHINE

Microservices

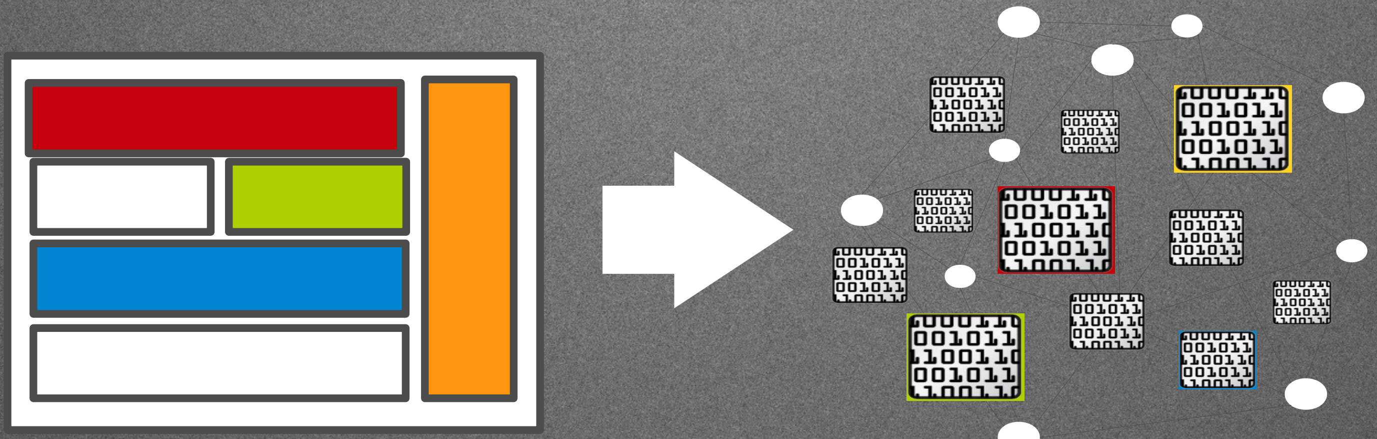
Microservices are not entirely new.

- The vocabulary term is “new-ish” (2012 – James Lewis and Martin Fowler)
- The idea is very old
  - Microkernels have existed since the 1980s
  - Could argue that system admins have been doing this with shell scripts and pipes for years
- Applying this concept to services higher in the stack is a newer trend
  - Heavily influenced by popular technologies such as web microframeworks and containers.



What are Microservices?

- Services, “the UNIX Way”
  - Do one thing, do it well.
  - Decouple tightly coupled services, make the architecture more modular.
- Loosely coupled services using programming language agnostic APIs for communication
  - Example: REST APIs



MONOLITHIC/LAYERED

MICROSERVICES

Immutable Infrastructure

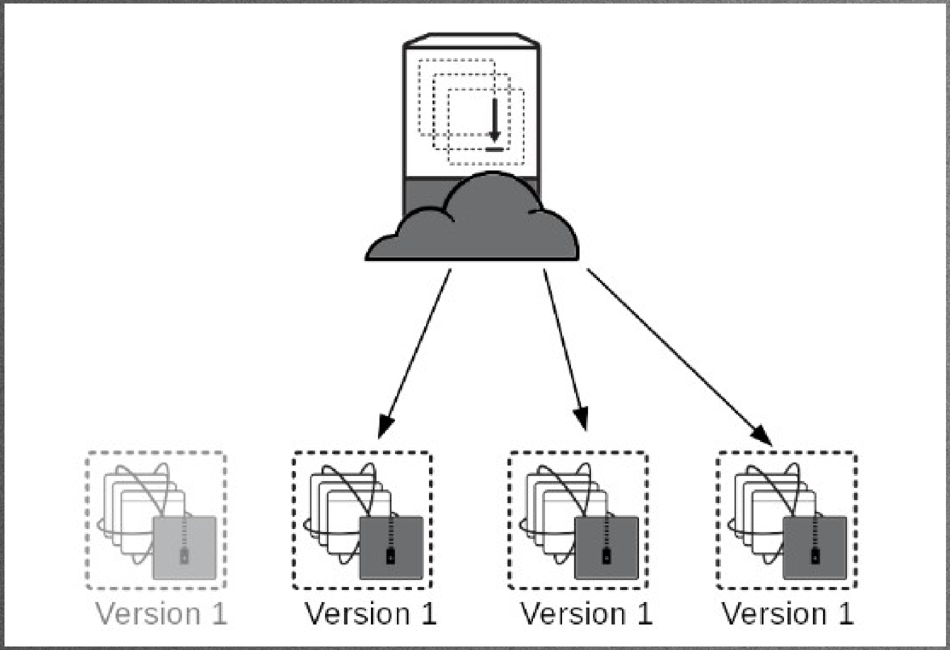
What is Immutable Infrastructure?

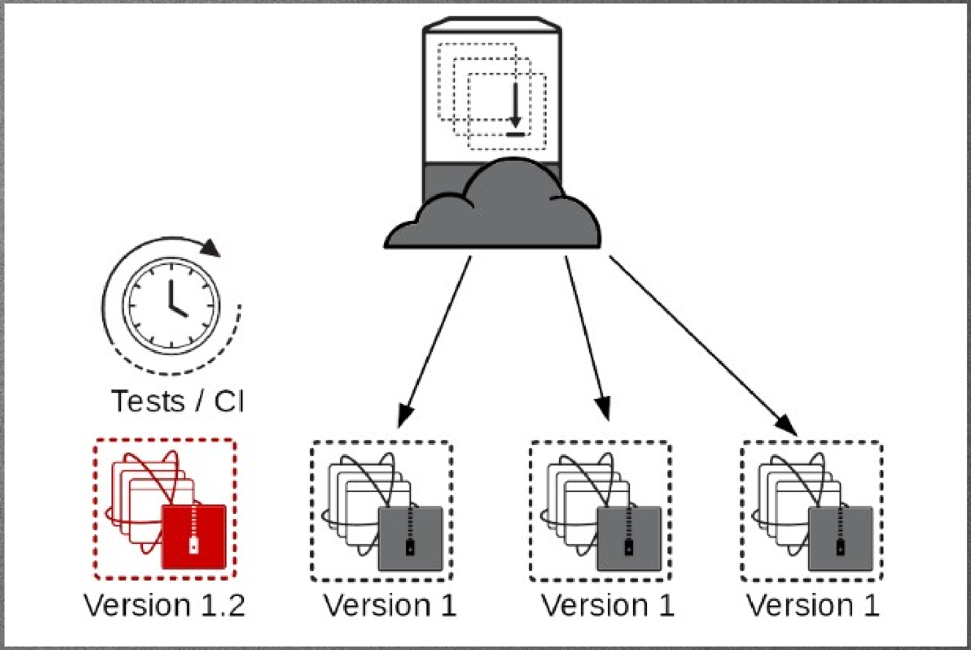
- Immutable Infrastructure is:
    - Fully automated
        - Can be deployed, destroyed, re-deployed without human intervention
        - Within reason, someone running the command or clicking the button is fine
    - Static
        - Once deployed, do not alter infrastructure components
        - If a change is needed, redeploy
- This is actually new!
    - Cloud technologies, Linux containers, and the tooling around them have allowed this new concept.

In Practice

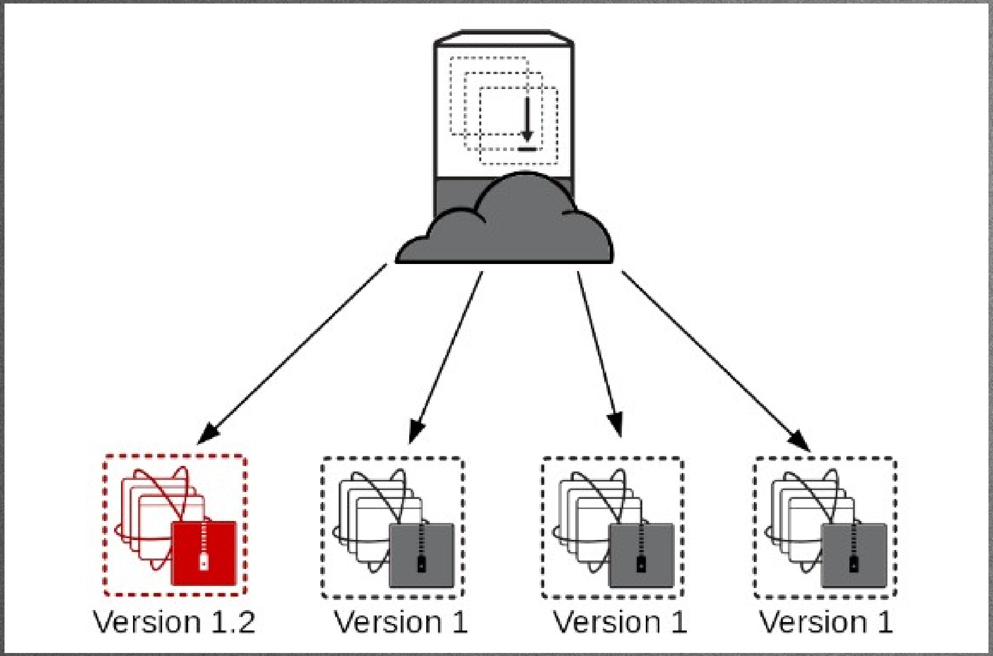
- What you deploy is now a “build artifact”
  - Example of a build artifact is a docker image
- Configuration Management is now part of the build
  - Run your build/shell script, ansible, saltstack, puppet, chef, etc. at build time
    - Example: in the Dockerfile
  - Possible exception is configuration files mounted into the container at runtime
    - Should be read-only, nothing should be mutable.
    - Provides flexibility in deploying between environments.
- Need a configuration change?
  - Build a new artifact
- Artifacts are then tested and “graduate” to production
  - Red/Black, Blue/Green, etc Deployment models

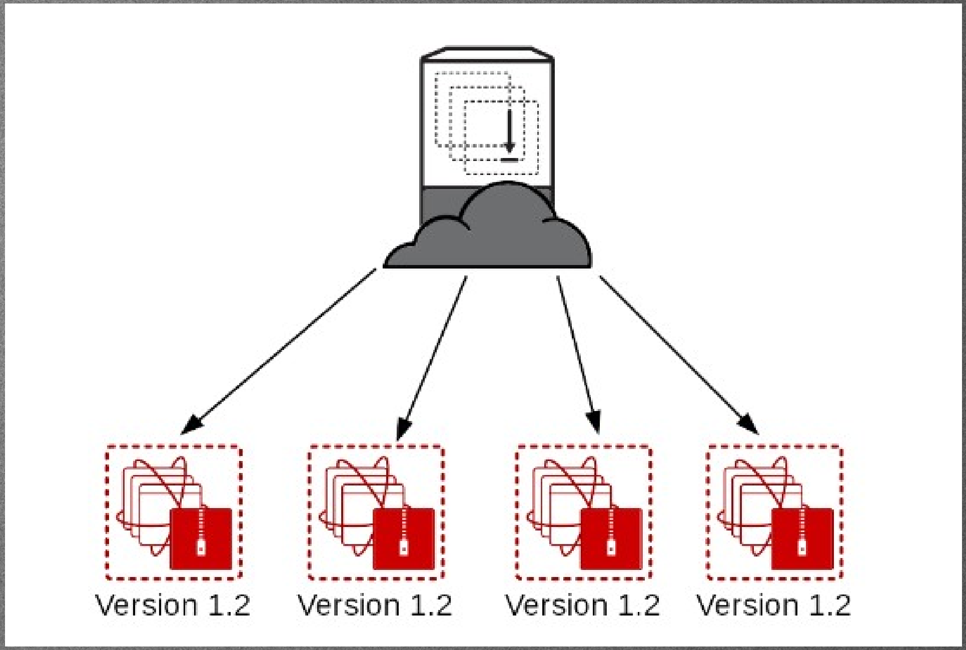
Deployment Example





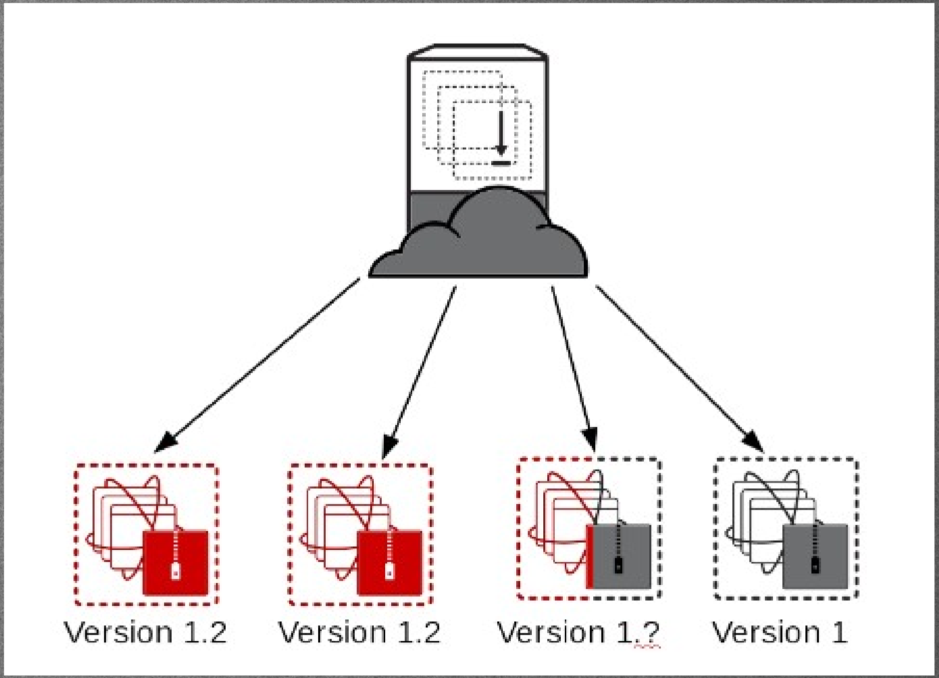
Deployment Example



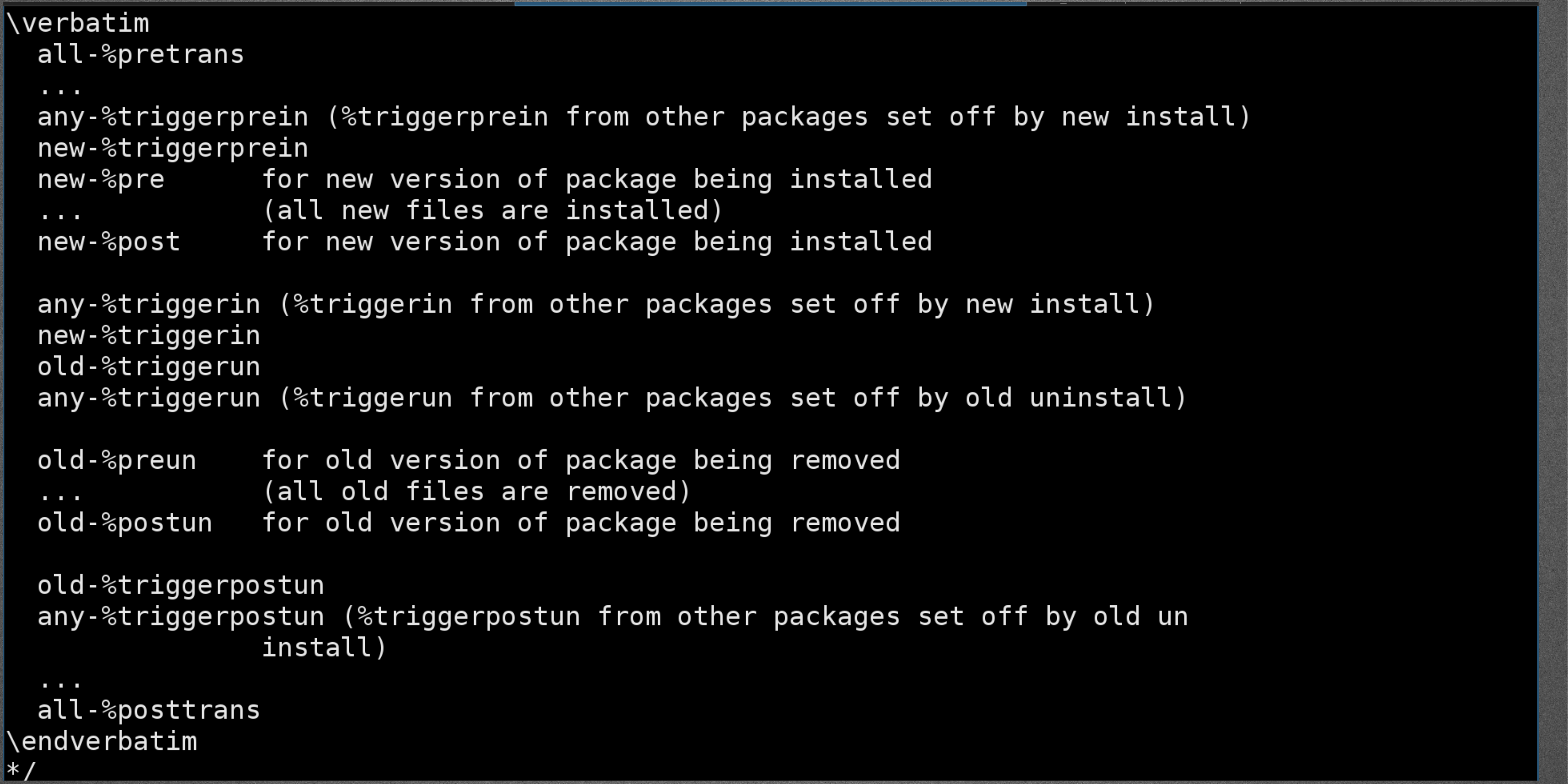


Potential Issues Avoided

- Start a traditional deployment or system upgrade
- Successful on part of the infrastructure
- Suddenly, a wild failure appears!
  - Use your imagination, anything that could interrupt a deploy.
- How clean is the rollback procedure?
- How do you verify the components?
  - Is your filesystem tree versioned?
  - Can you guarantee the order of upgrade trigger execution?
  - Do you know how far the package upgrade transaction made it before the failure?



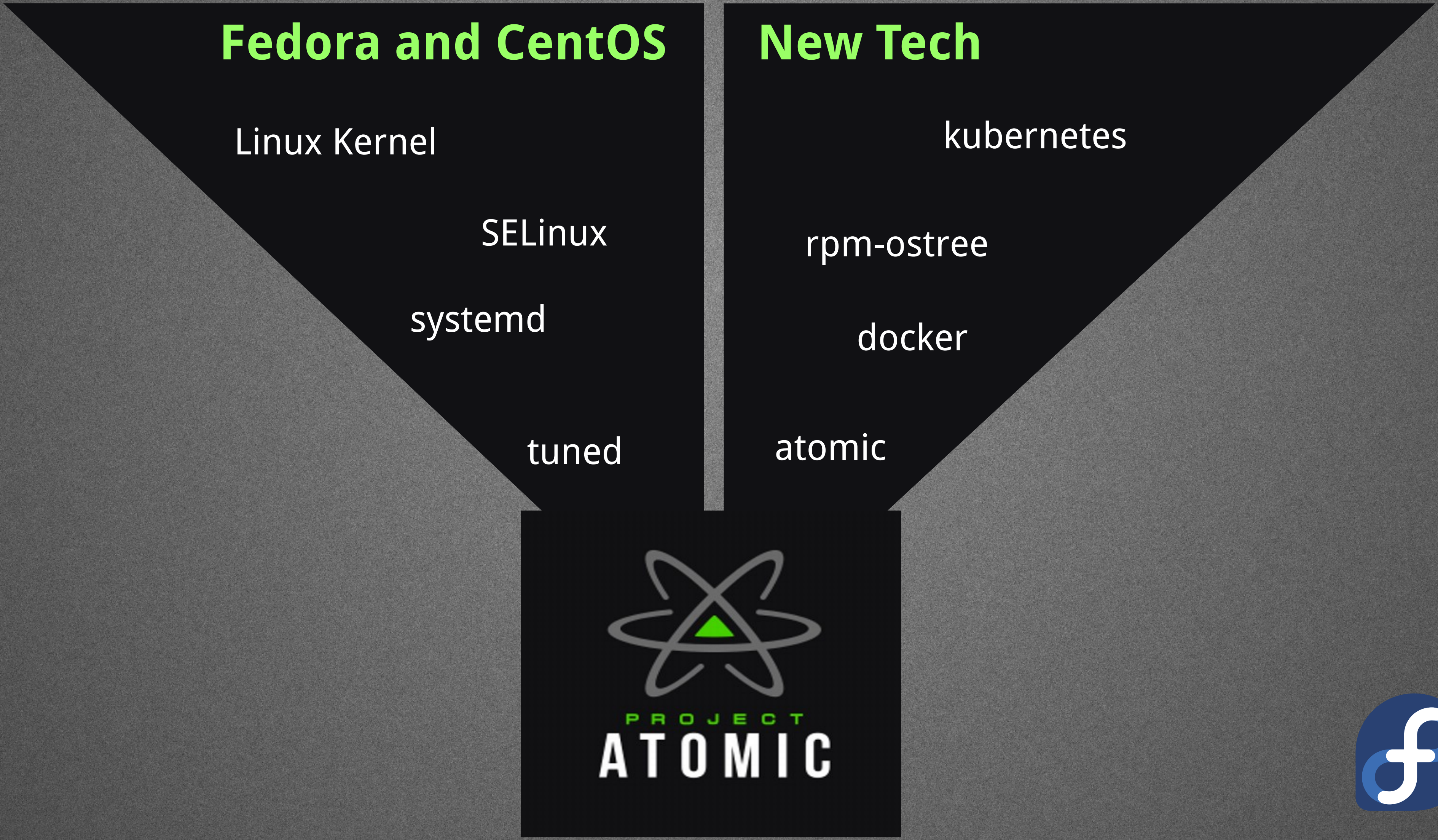
RPM Transaction Triggers



```
\verbatim
  all-%pretrans
  ...
  any-%triggerprein (%triggerprein from other packages set off by new install)
  new-%triggerprein
  new-%pre      for new version of package being installed
  ...           (all new files are installed)
  new-%post     for new version of package being installed

  any-%triggerin (%triggerin from other packages set off by new install)
  new-%triggerin
  old-%triggerun
  any-%triggerun (%triggerun from other packages set off by old uninstall)

  old-%preun    for old version of package being removed
  ...           (all old files are removed)
  old-%postun   for old version of package being removed

  old-%triggerpostun
  any-%triggerpostun (%triggerpostun from other packages set off by old un
                install)
  ...
  all-%posttrans
\endverbatim
*/
```

Immutable…
Operating Systems?

Project Atomic



Fedora and CentOS

Linux Kernel

SELinux

systemd

tuned

New Tech

kubernetes

rpm-ostree

docker

atomic

Fedora Atomic Host

Fedora Atomic Host	OPTIMIZED FOR CONTAINERS		
	MINIMIZED FOOTPRINT	**SIMPLIFIED MAINTENANCE**	**ORCHESTRATION AT SCALE**
Inherits everything from the “Parent” Distro. This is Fedora but with new delivery mechanism coupled with a new layer of abstraction on top of the package management.	Minimized host environment tuned for running Linux containers.	Atomic updating and rollback means it’s easy to deploy, update, and rollback using ostrees..	Build composite applications by orchestrating multiple containers as microservices on a single host instance.

Atomic Host

- Deployments and Upgrades are 'rpm-ostrees' and are not installed like traditional rpms
  - An 'ostree' is effectively an entire rootfs tree managed similar to git commits
  - 'rpm-ostree' is a utility built on top of ostree to allow trees to be built from collections of rpms
- Upgrades are atomic in nature
  - All or nothing (it either applied or it didn't)
  - Quick/easy rollback to previous tree
- Entire trees get tested as a cohesive unit
  - There's no questions about what versions of X, Y, or Z when troubleshooting

Atomic Host

- The 'atomic' command is (currently) a wrapper around 'rpm-ostree' and 'docker'

- Performing an upgrade

```
# atomic host upgrade
Updating from: fedora-atomic:fedora-atomic/f23/x86_64/docker-host
```

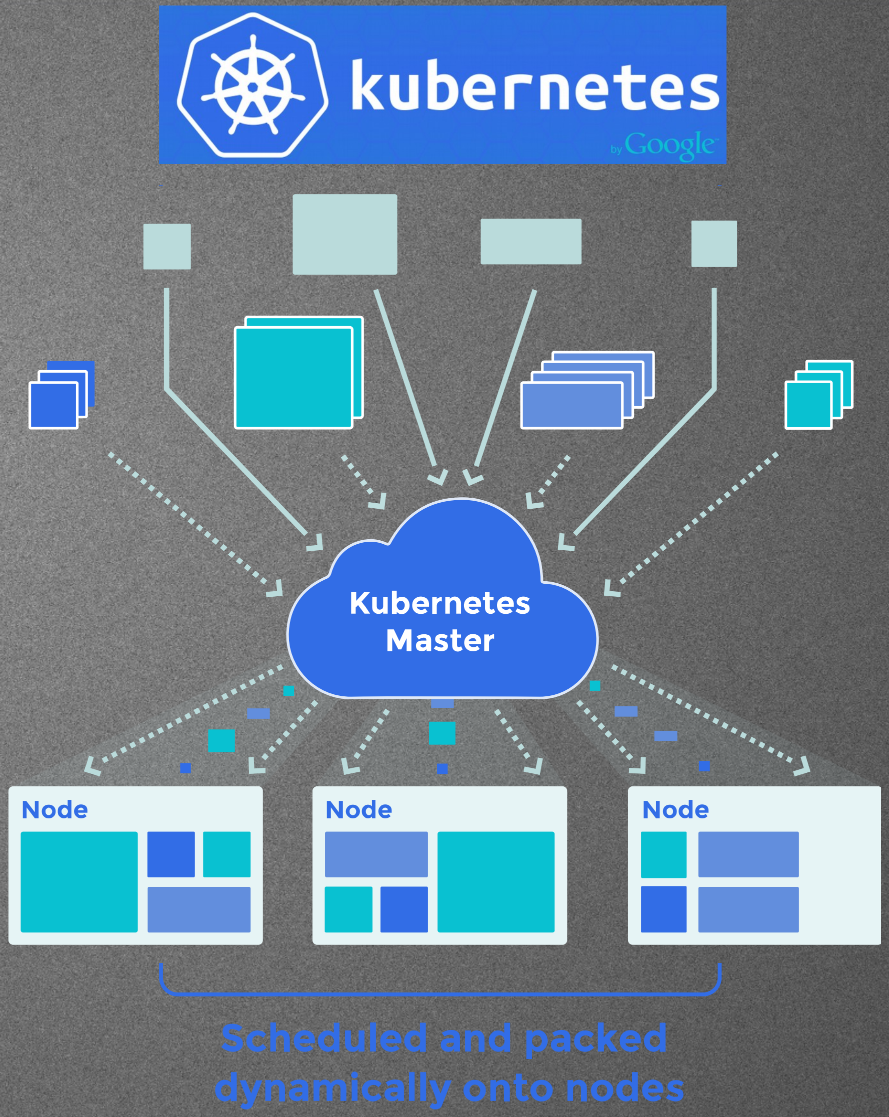
- Checking status

```
# atomic host status
  TIMESTAMP (UTC)         VERSION   ID             OSNAME           REFSPEC

* 2016-02-02 05:34:15     23.58     ae53656858     fedora-atomic    fedora-
atomic:fedora-atomic/f23/x86_64/docker-host
  2016-01-31 06:43:12     23.57     6adf2d354f     fedora-atomic    fedora-
atomic:fedora-atomic/f23/x86_64/docker-host
```

Orchestration

Kubernetes

- Distributed orchestration for containers
- “Pod” - Set of containers that share pid, network, IPC, and UTS namespace.
  - Are scheduled to nodes as an unit
- “Service” - Set of one or more Pods and a policy to access them
- Replication Controller manages pods
- Node level proxy load balances and proxies access to Services
- Pluggable overlay network provider
- Pluggable persistant storage provider

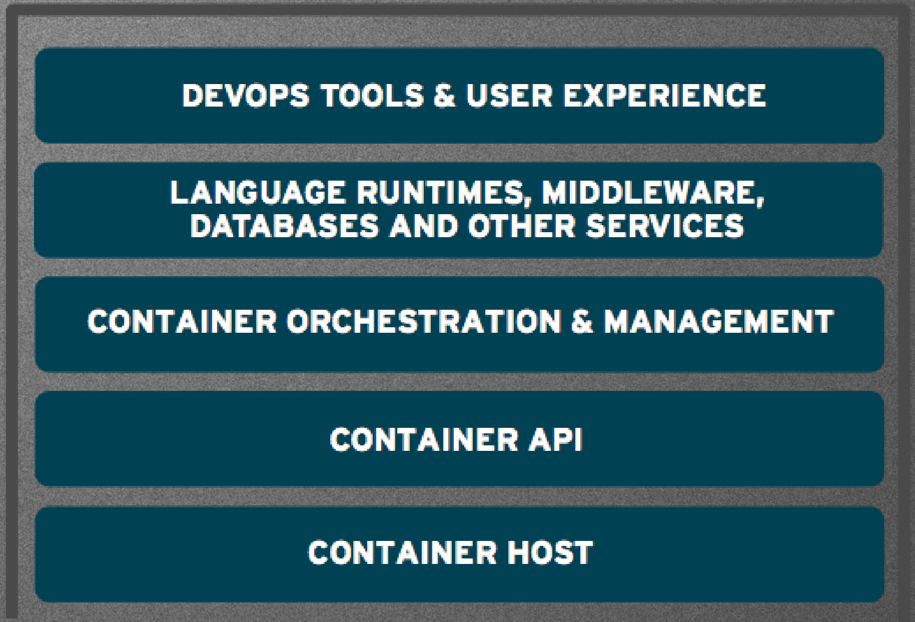


Developers!

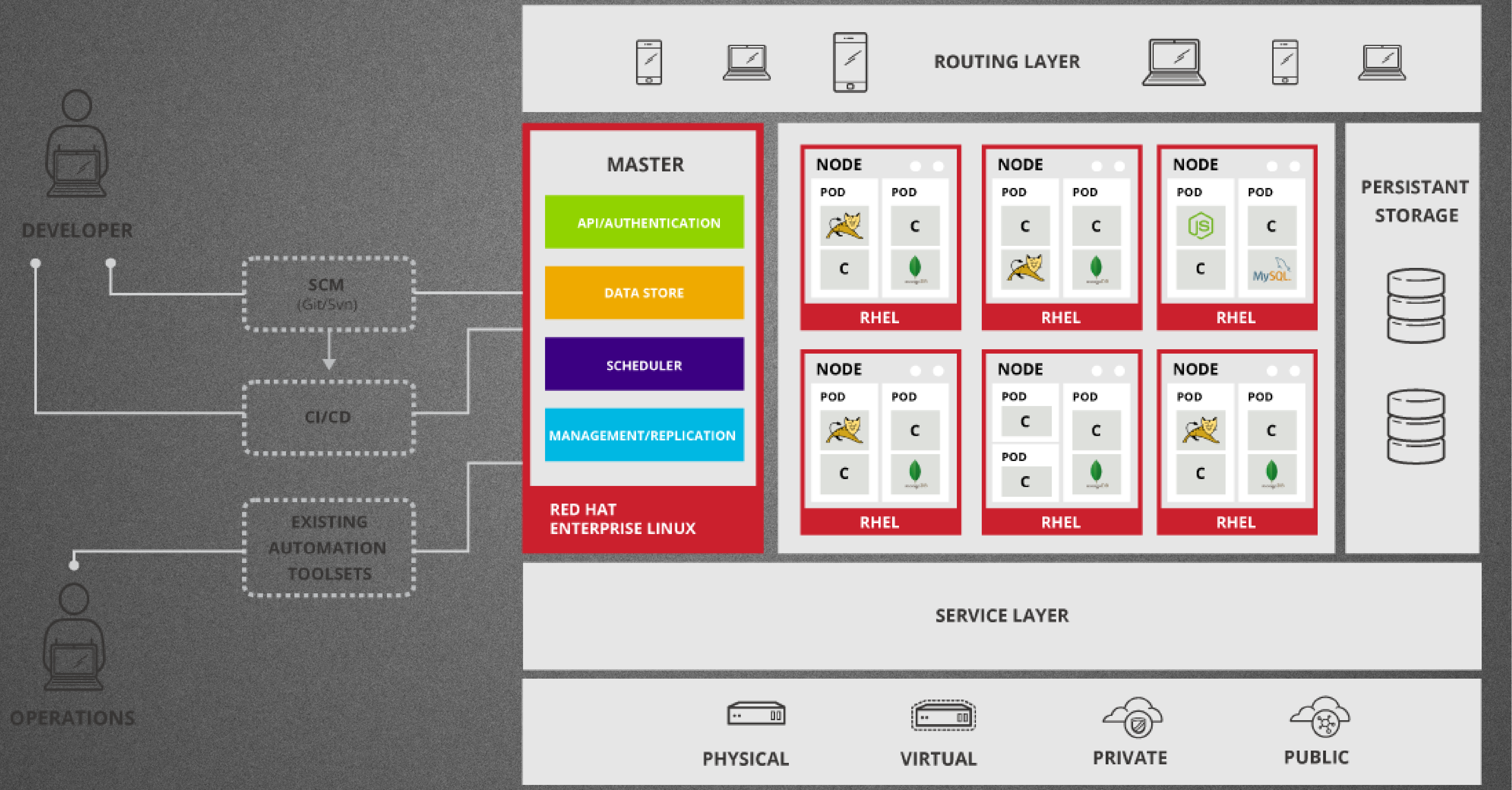
OpenShift Origin

- Standard containers API
- Web-scale container orchestration & management
- Container-optimized OS
- Large selection of supported application runtimes & services
- Robust tools and UX for Development & Operations
- Industry standard, web scale distributed application platform

OPENSHIFT origin


DEVOPS TOOLS & USER EXPERIENCE
LANGUAGE RUNTIMES, MIDDLEWARE, DATABASES AND OTHER SERVICES
CONTAINER ORCHESTRATION & MANAGEMENT
CONTAINER API
CONTAINER HOST

OpenShift Overview



Questions?

CONTACT:
maxamillion@fedoraproject.org
@TheMaxamillion

References

- http://chadfowler.com/blog/2013/06/23/immutable-deployments/
- http://blog.codeship.com/immutable-deployments/
- http://blog.codeship.com/immutable-infrastructure/
- http://martinfowler.com/articles/microservices.html
- http://microservicesbook.io/the-philosophy-of-microservice-architecture/
- http://nirmata.com/2015/02/microservices-five-architectural-constraints/
- http://2012.33degree.org/talk/show/67
- https://en.wikipedia.org/wiki/Operating-system-level_virtualization
- https://coreos.com/blog/rocket/
- https://coreos.com/blog/appc-gains-new-support/
- https://www.docker.com/
- https://github.com/docker/distribution
- http://www.redhat.com/en/insights/containers
- http://rhelblog.redhat.com/2015/05/07/stop-gambling-with-upgrades-murphys-law-always-wins/
- http://rhelblog.redhat.com/2015/05/05/rkt-appc-and-docker-a-take-on-the-linux-container-upstream/
- http://rhelblog.redhat.com/2015/04/01/red-hat-enterprise-linux-atomic-host-updates-made-easy/
- http://www.projectatomic.io/
- http://www.openshift.org//
- https://www.openshift.com
- http://www.redhat.com/en/about/blog/red-hat-and-google-collaborate-kubernetes-manage-docker-containers-scale
- http://rhelblog.redhat.com/2014/04/15/rhel-7-rc-and-atomic-host/
- http://opencontainers.org/
- http://runc.io/
- http://queue.acm.org/detail.cfm?id=2884038