Immutable infrastructure, containers, & the future of microservices

Adam Miller
Senior Software Engineer, Red Hat
2015-08-22

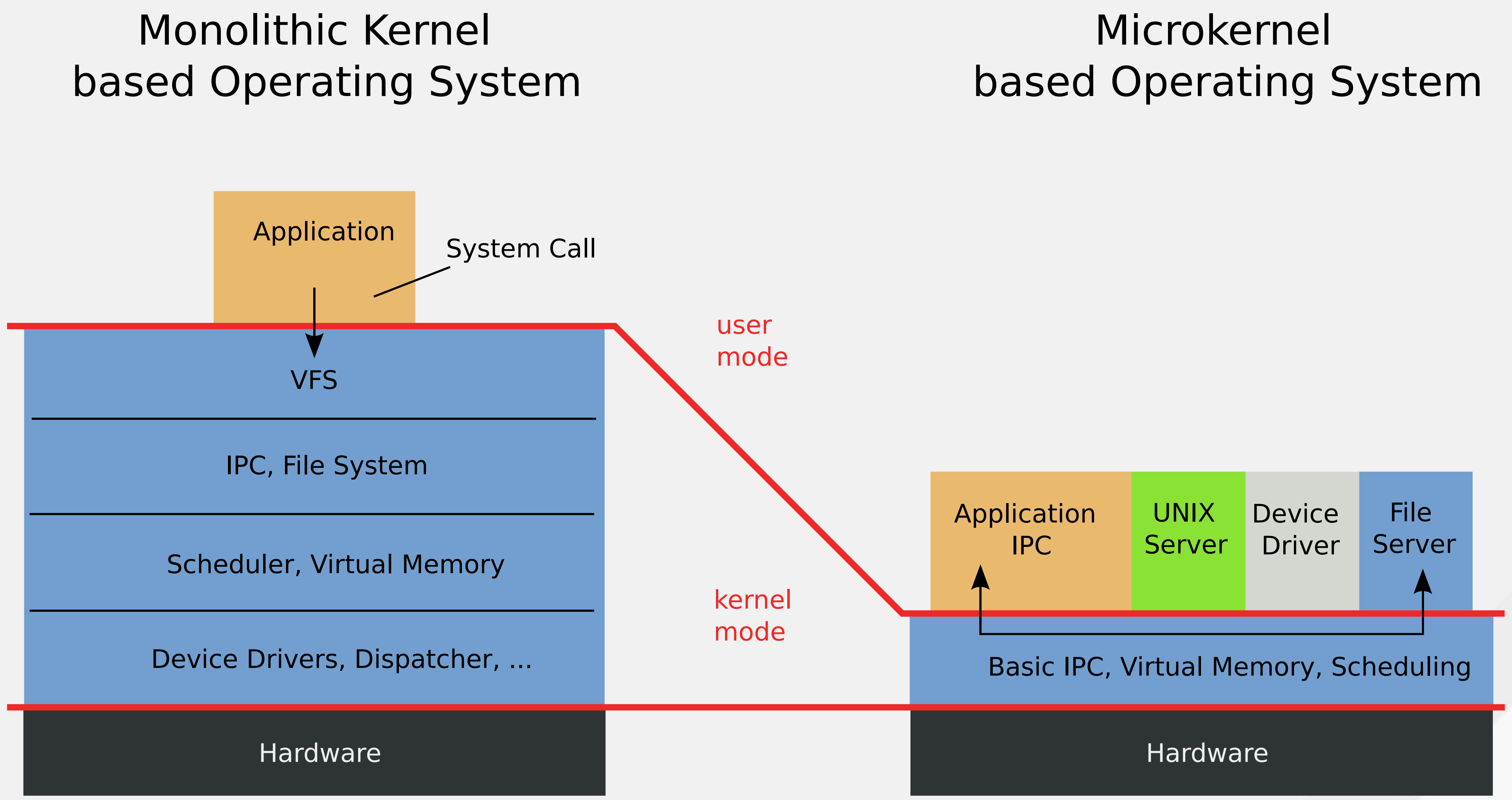
What we'll cover in this session

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

Microservices

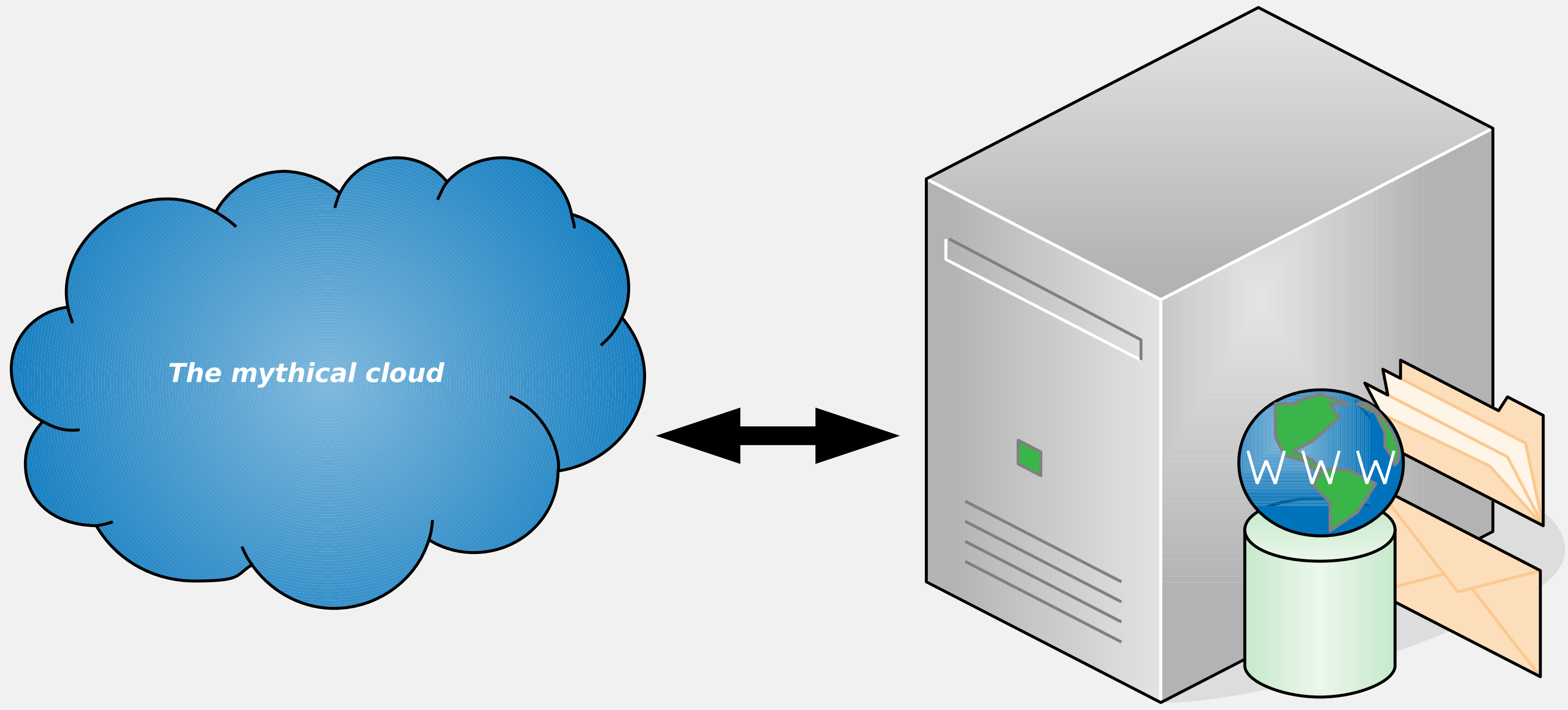
Microservices are not entirely new.

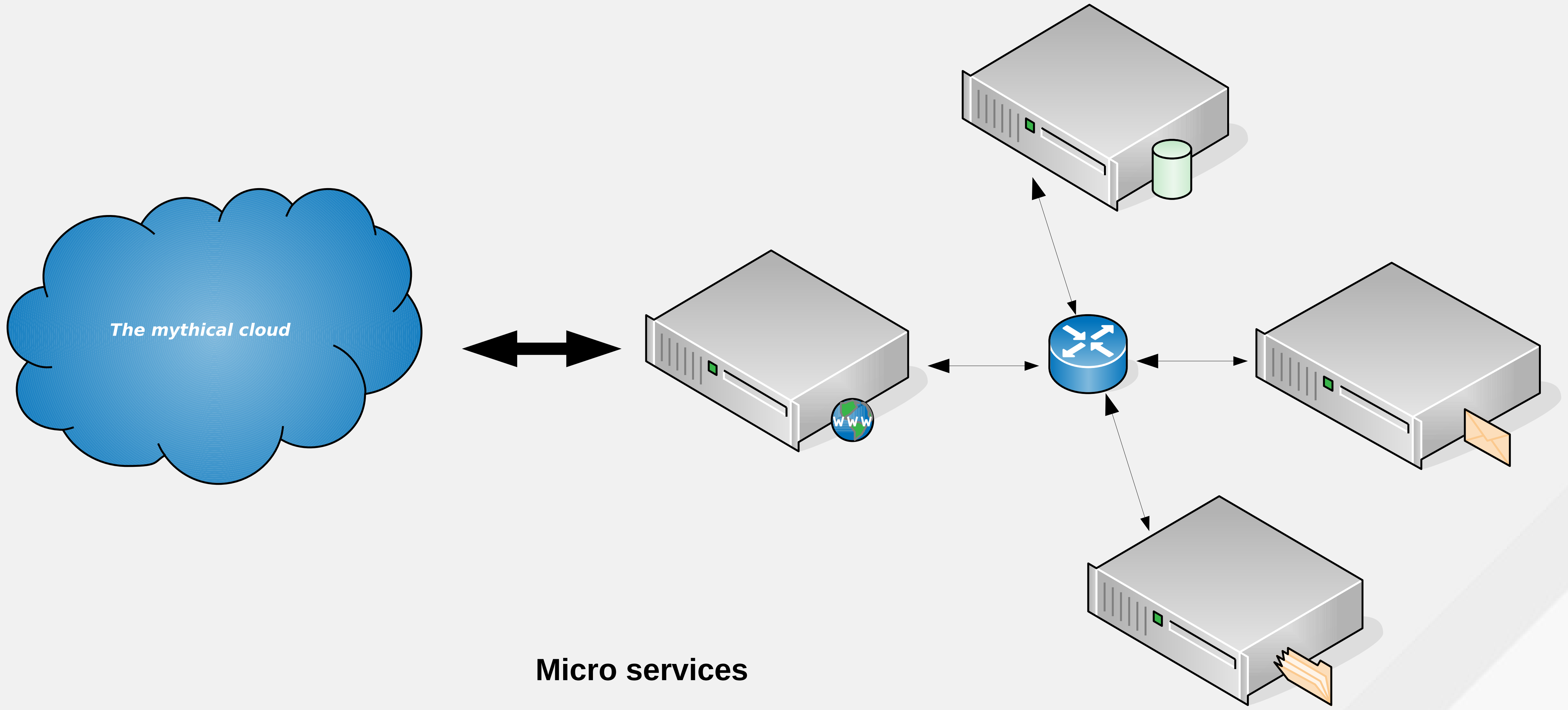
- The vocabulary term is “new-ish” (2012 – James Lewis and Martin Fowler)
- The idea is very old
  - Microkernels have existed since the 1980s
  - Could argue that system admins have been doing this with shell scripts and pipes for years
- Applying this concept to services higher in the stack is a newer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



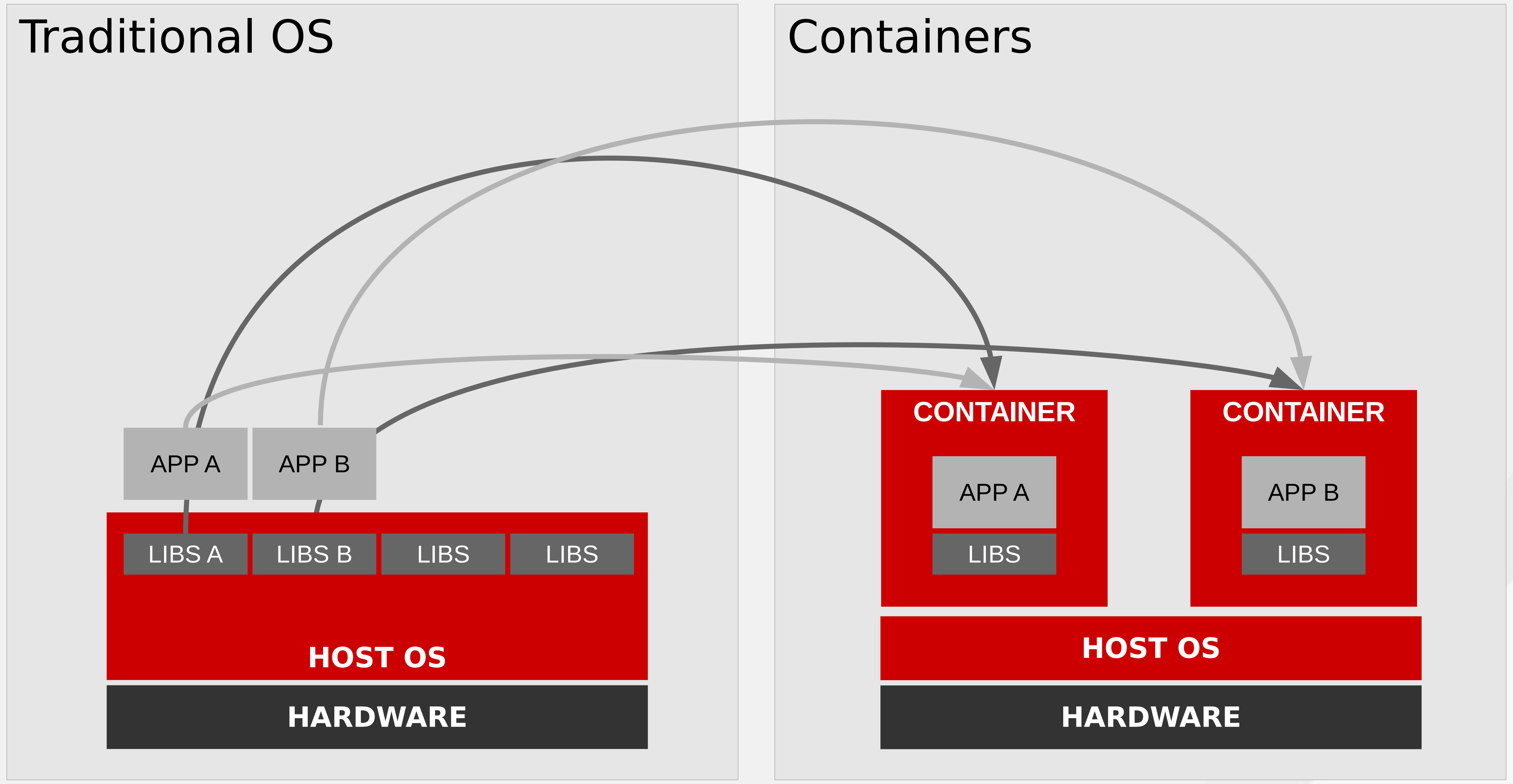


Containers

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

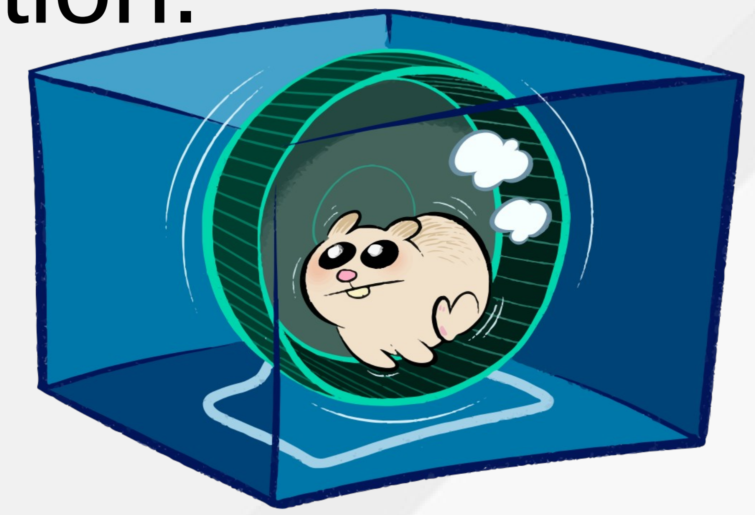
- 2013 – DotCloud releases Docker (https://github.com/docker/docker)
  - Originally used LXC as the backend, introduces the Docker daemon, layered images, standard toolset for building images and a distribution method (docker registry). Later makes backend driver pluggable and replaces LXC with libcontainer as default.

- 2014 – CoreOS releases rkt (https://github.com/coreos/rkt)
  - rkt is an implementation of App Container(appc) specification and App Container Image(ACI) specification.
    - ACI and appc aimed to be a cross-container specification to be a common ground between container implementations.

The modern Linux Container is born (Con't)

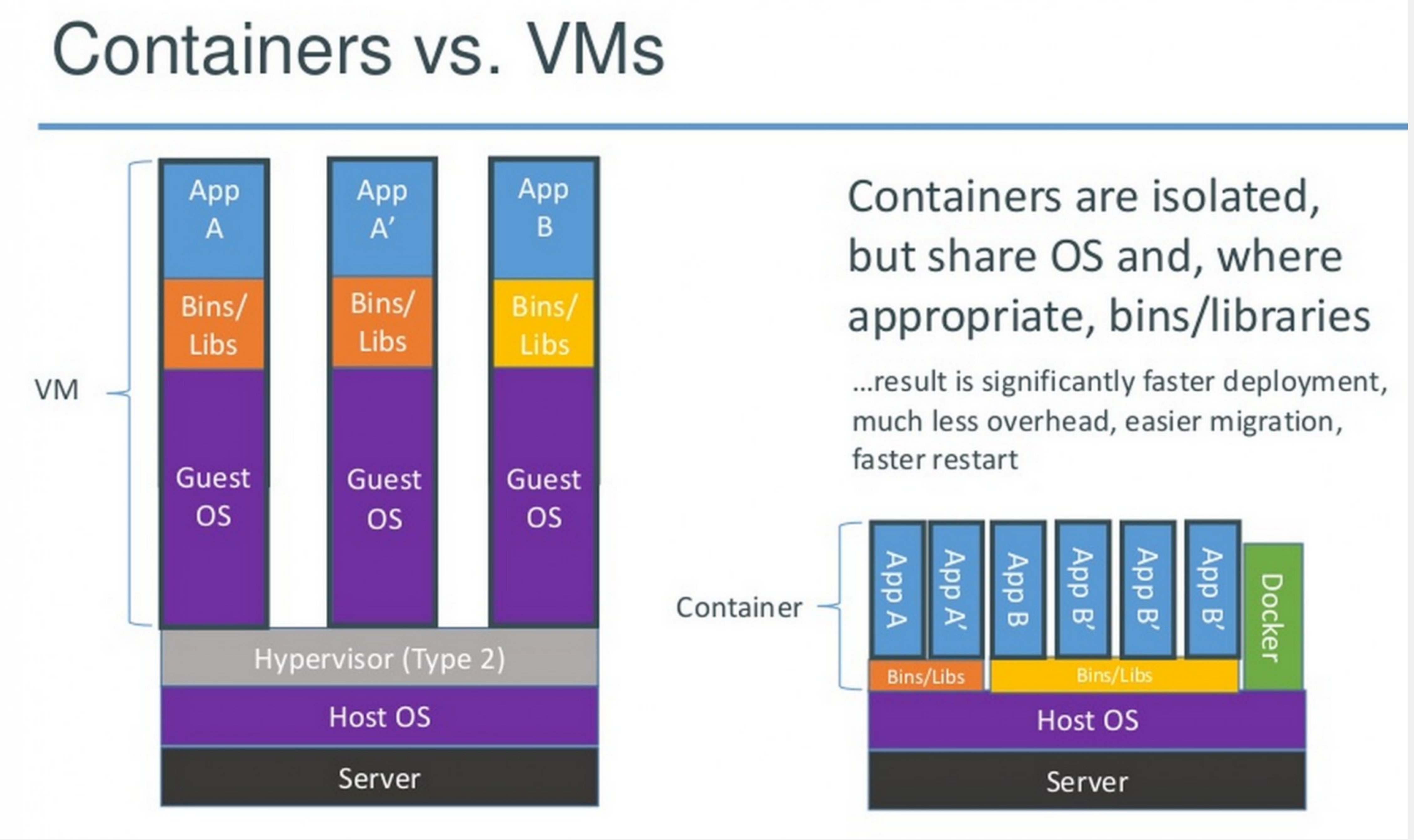
- 2015 – Open Container Project (http://opencontainers.org/)
  - “The Open Container Initiative is a lightweight, open governance structure, to be formed under the auspices of the Linux Foundation, for the express purpose of creating open industry standards around container formats and runtime.” - http://opencontainers.org/
  - 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
- Red Hat's own Mr. SELinux (Dan Walsh) pushed SELinux support upstream to Docker.
- Pluggable backends for isolation mechanism, storage, networking, etc.



Containers vs. VMs

Containers are isolated, but share OS and, where appropriate, bins/libraries

...result is significantly faster deployment, much less overhead, easier migration, faster restart

Docker

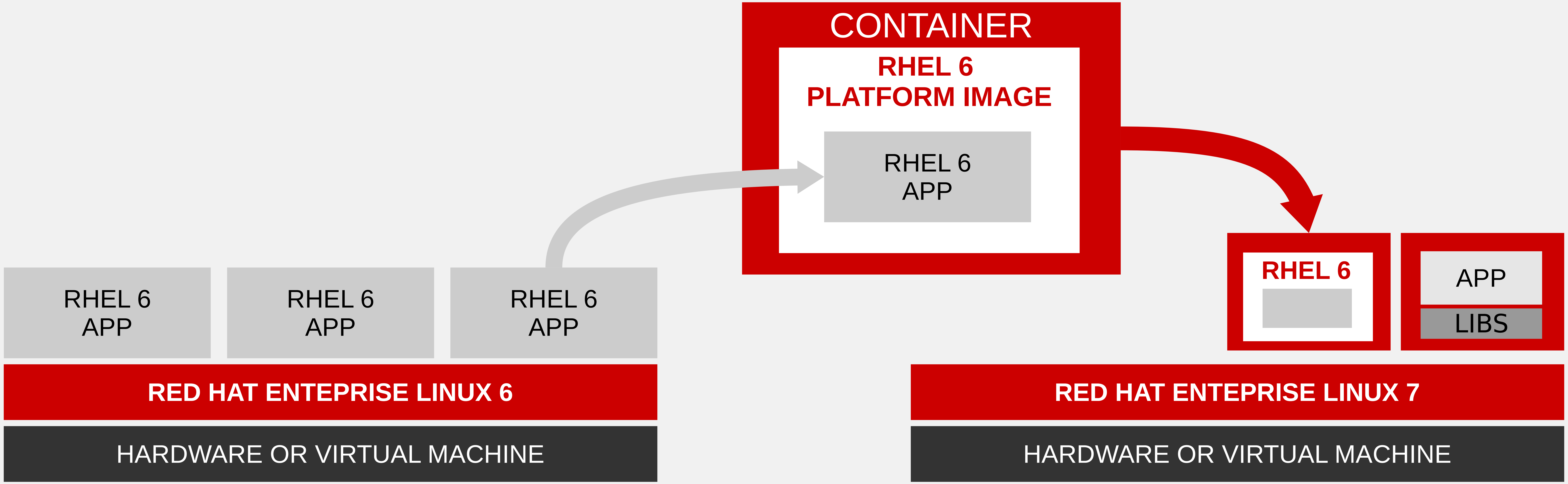
- Docker Daemon is the single point of entry, has language bindings for other clients and tooling. (Image verification)
- Containers are instances of images.
- Images are built in a standard way using Dockerfile
- Red Hat's own Mr. SELinux (Dan Walsh) pushed SELinux support upstream to Docker.
- Pluggable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

```
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"]
```

Brief History of Red Hat and Linux Containers

- Red Hat kernel developers involved in cgroups and namespaces pre-dating LXC
  - Kernel namespaces and cgroups are core kernel technologies that enabled LXC
- 2013-05-13: First public Open Source release of Docker from DotCloud
- 2013-09-19: Red Hat and Docker announce collaborative partnership
  - 2013-09-24: First upstream pull request merged into Docker from Red Hat developer
- 2014-07-10: Red Hat and Google announce partnership around Kubernetes for container orchestration
  - Red Hat is currently the #2 contributor to Kubernetes, second only to Google.
- 2014-08-14: Red Hat Announces OpenShift Architecture V3, based on Kubernetes
- 2015-05-04: Red Hat Developer joins the CoreOS App Container Spec community governance board

Red Hat Enterprise Linux and Containers


CONTAINER
RHEL 6
PLATFORM IMAGE
RHEL 6
APP
RHEL 6
APP
RHEL 6
APP
RHEL 6
APP
RED HAT ENTEPRISE LINUX 6
HARDWARE OR VIRTUAL MACHINE
RHEL 6
APP
LIBS
RED HAT ENTEPRISE LINUX 7
HARDWARE OR VIRTUAL MACHINE

- Deploy containerized RHEL 6 applications to RHEL 7 without porting or changing source code
- Make use of innovations in Red Hat Enterprise Linux 7 without compromising the reliability and security of existing Red Hat Enterprise Linux 6 apps
- Available as part of your Red Hat Enterprise Linux subscription

Immutable Infrastructure

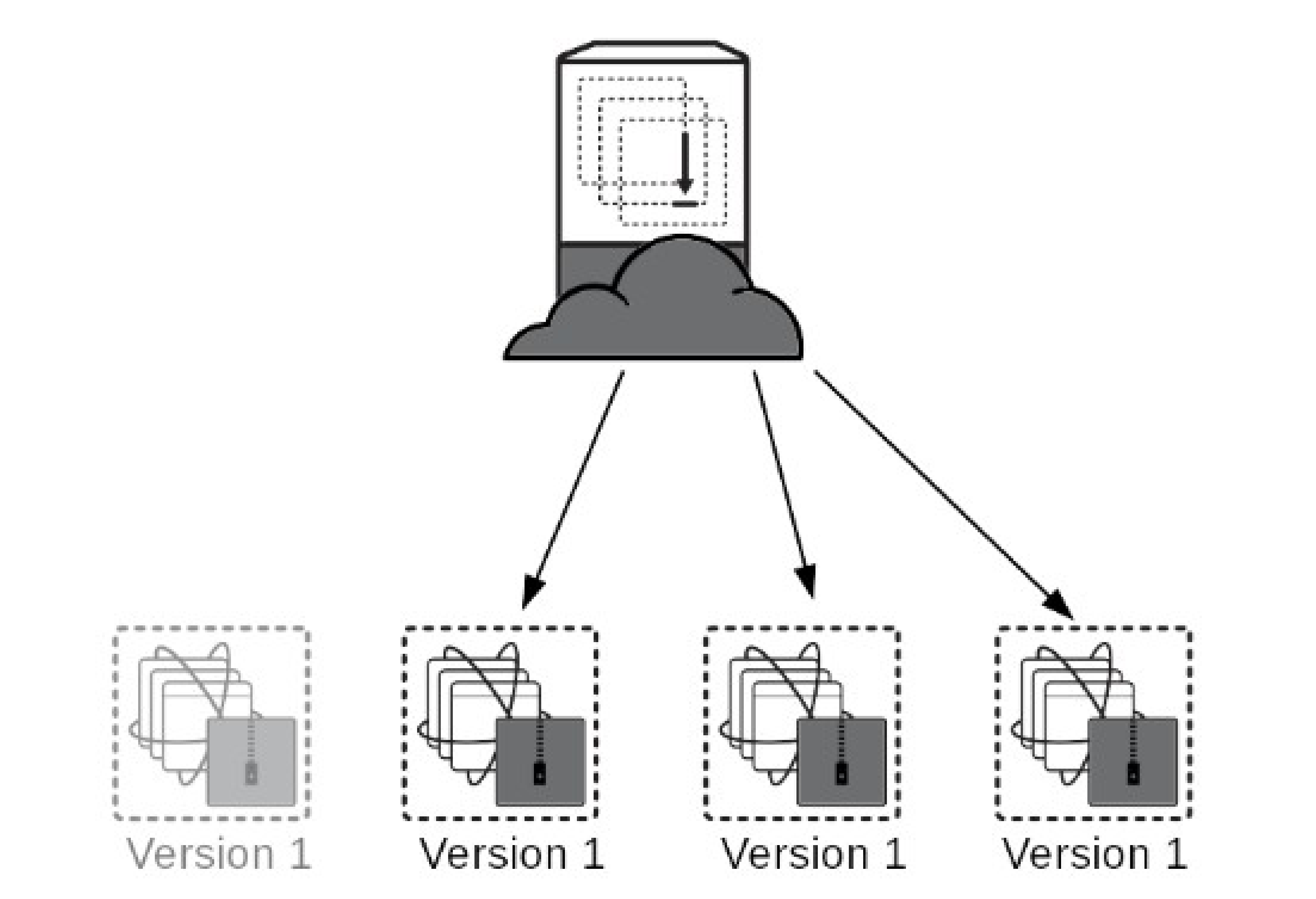
What is Immutable Infrastructure?

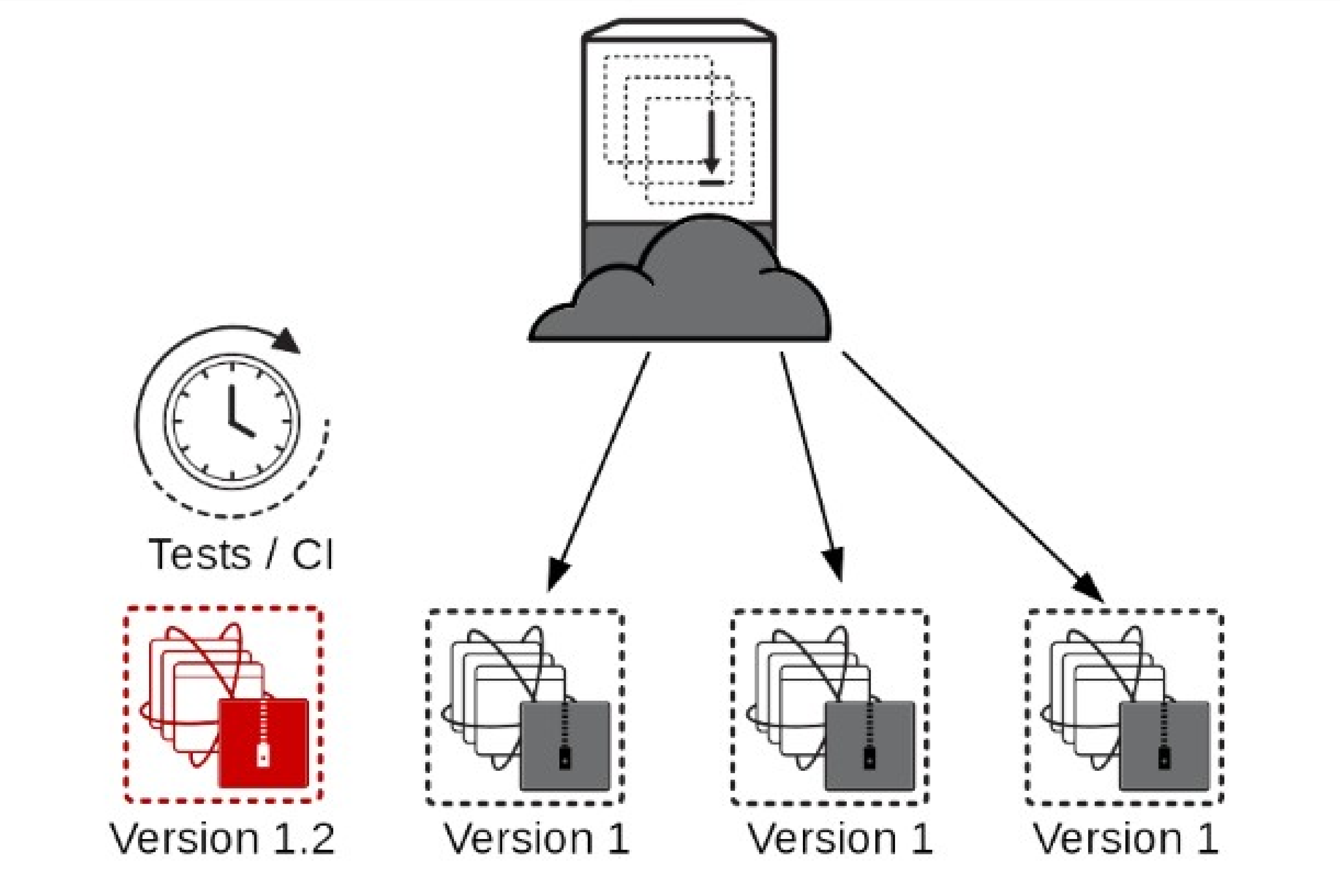
- Immutable Infrastructure is:
  - Fully automated
    - Can be deployed, destroyed, re-deployed without human intervention
      - Within reason, someone running the command or clicking the button is fine
  - Static
    - Once deployed, do not alter infrastructure components
    - If a change is needed,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

Immutable Infrastructure in Practice

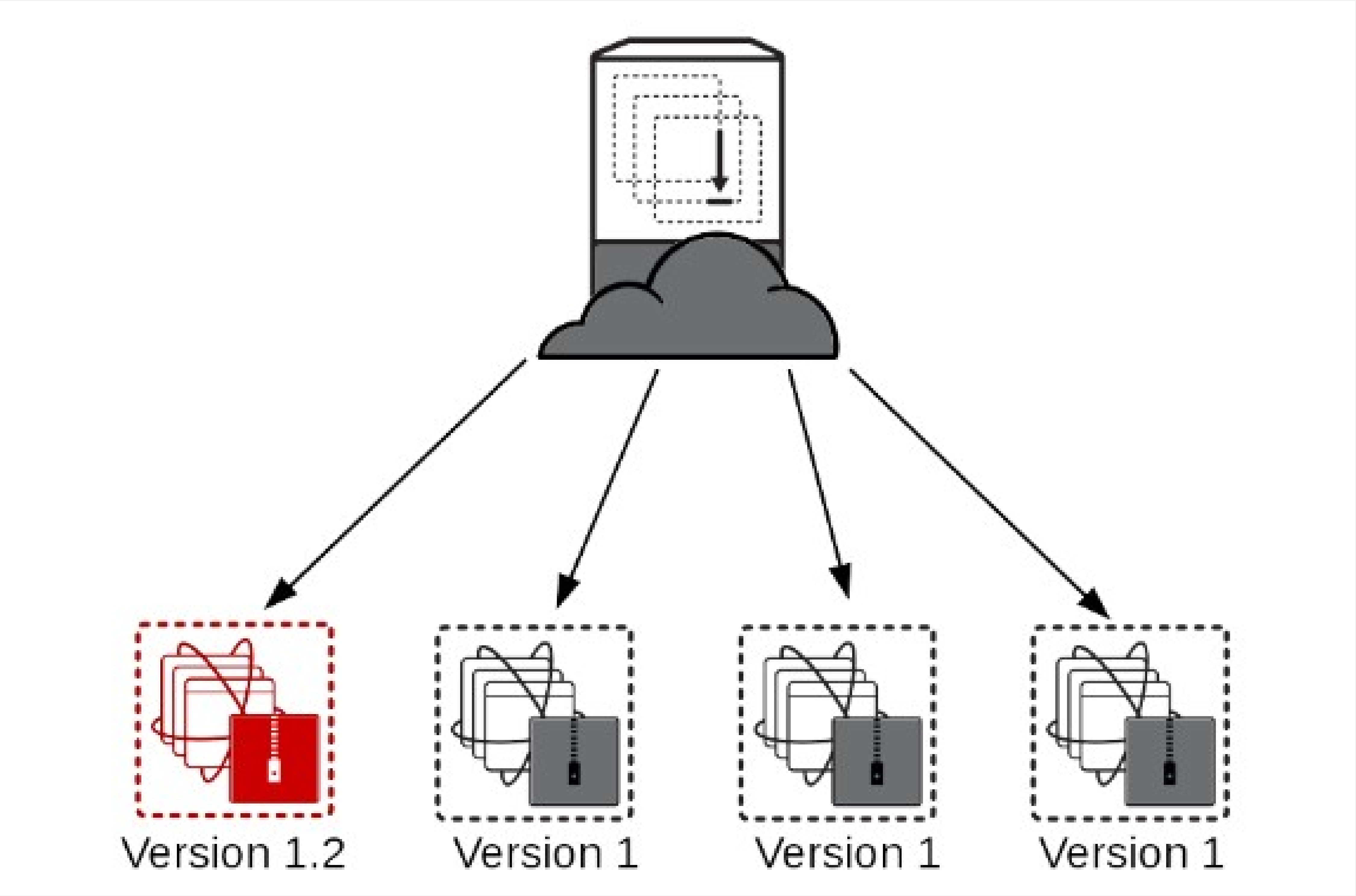
- What you deploy is now a “build artifact”
  - Example of a build artifact is a docker image
- Configuration Management is now part of the build
  - Run your build/shell script, ansible, saltstack, puppet, chef, etc. at build time
    - Example: in the Dockerfile
  - Possible exception is configuration 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

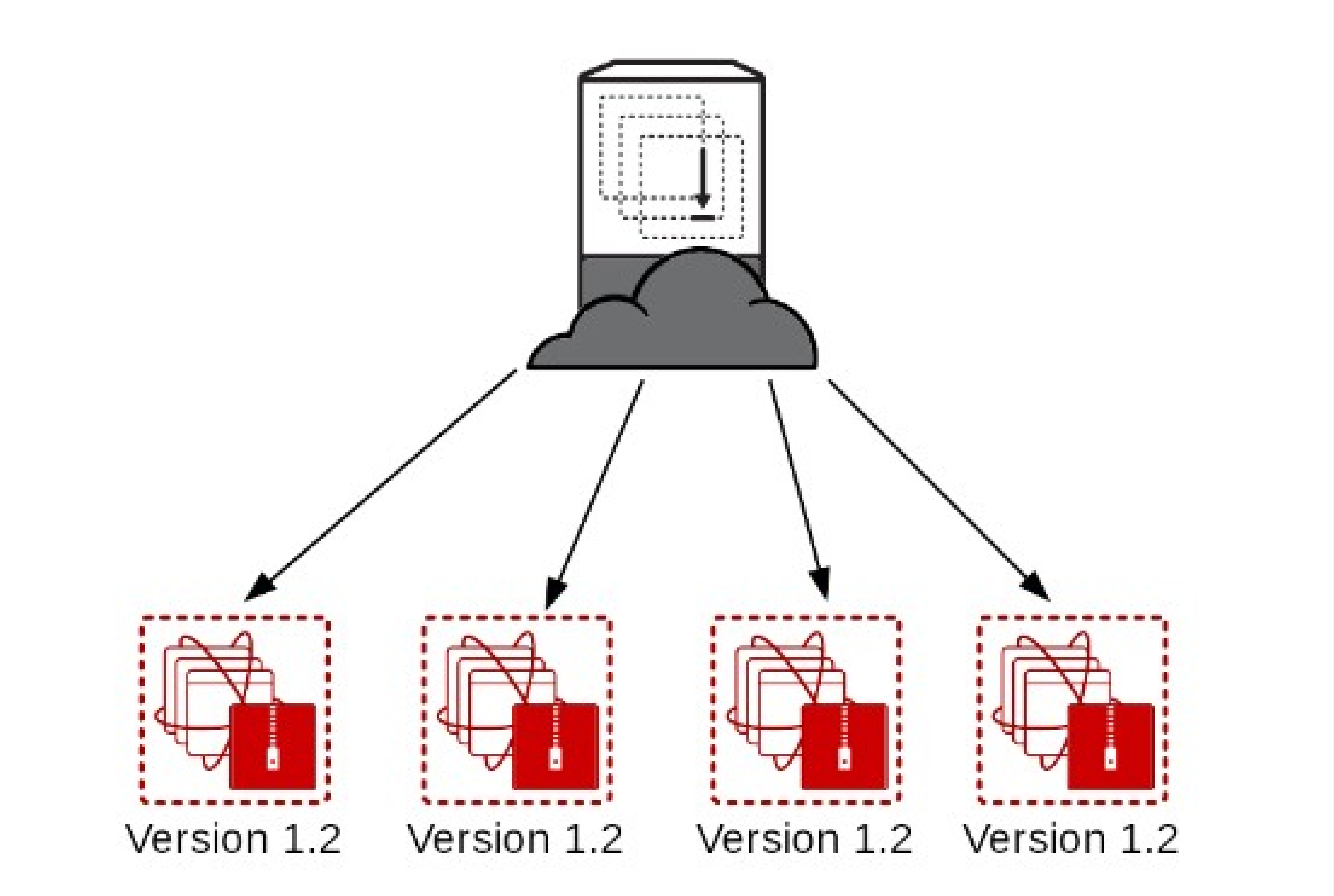
Immutable Infrastructure Deployment





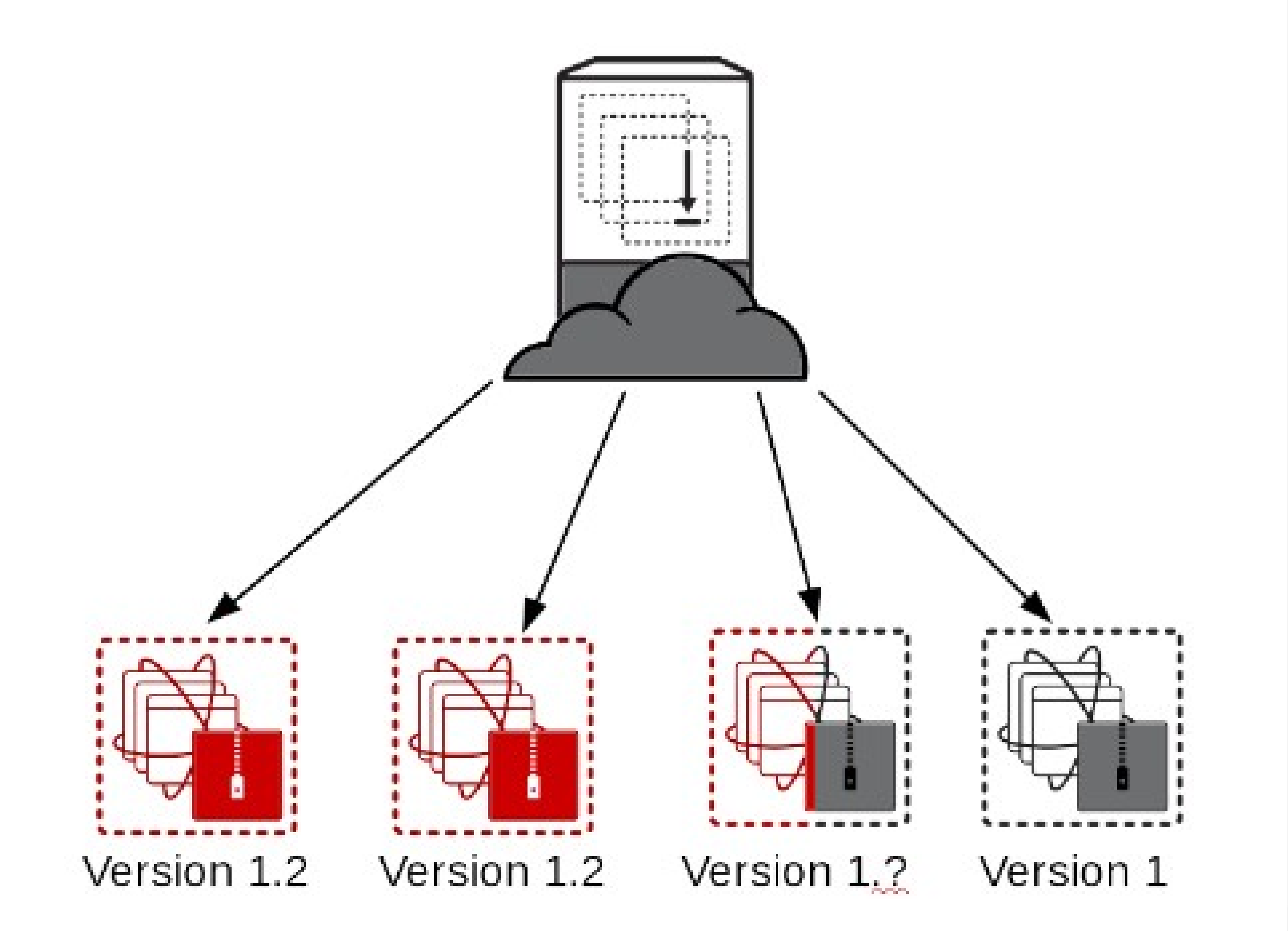
Immutable Infrastructure Deployment Continued



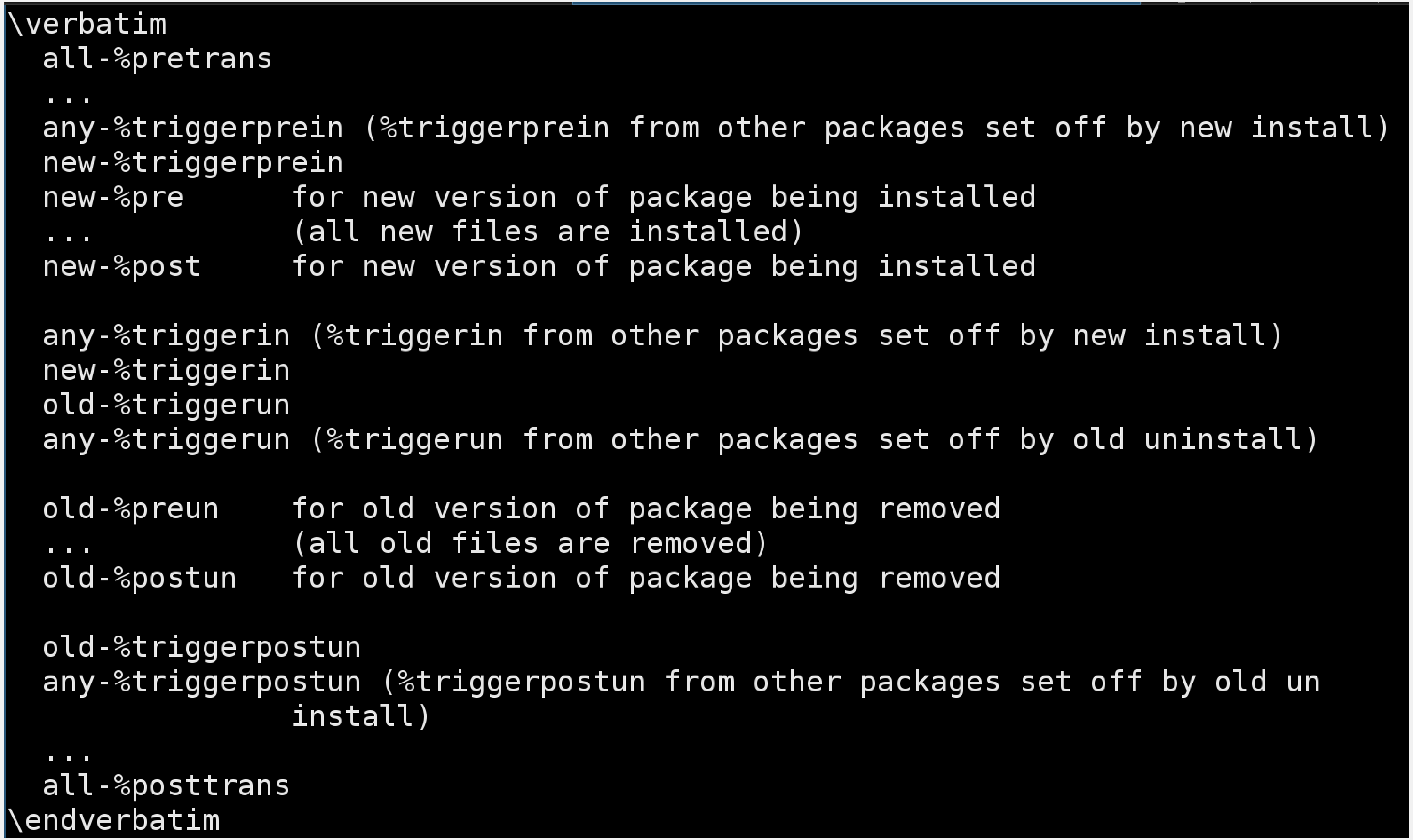


Example of Potential Issues Avoided

- Start a traditional deployment/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?



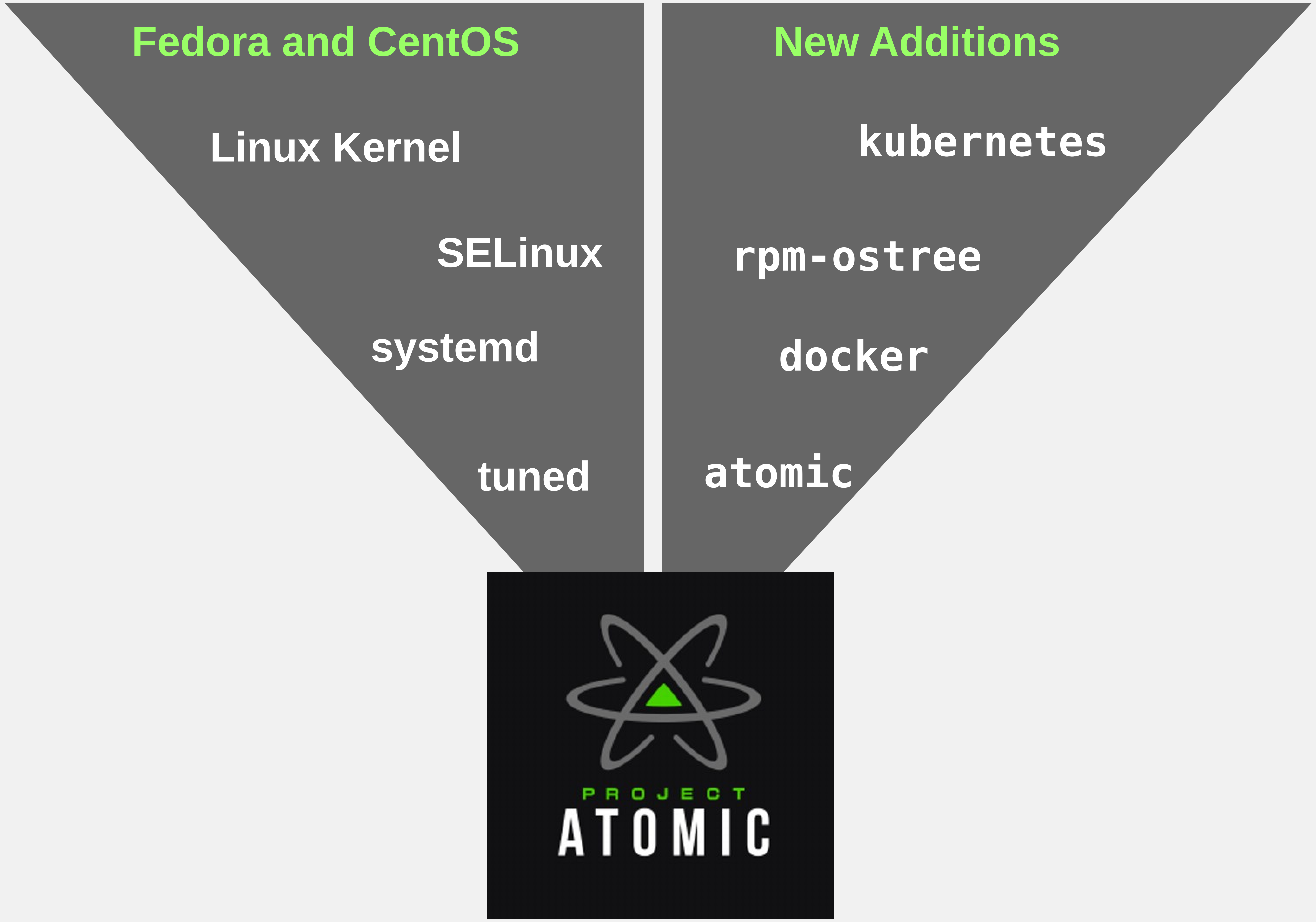
RPM Transaction Triggers



```
\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
```

What if we could do this with the entire Operating System?

Project Atomic: Atomic Host



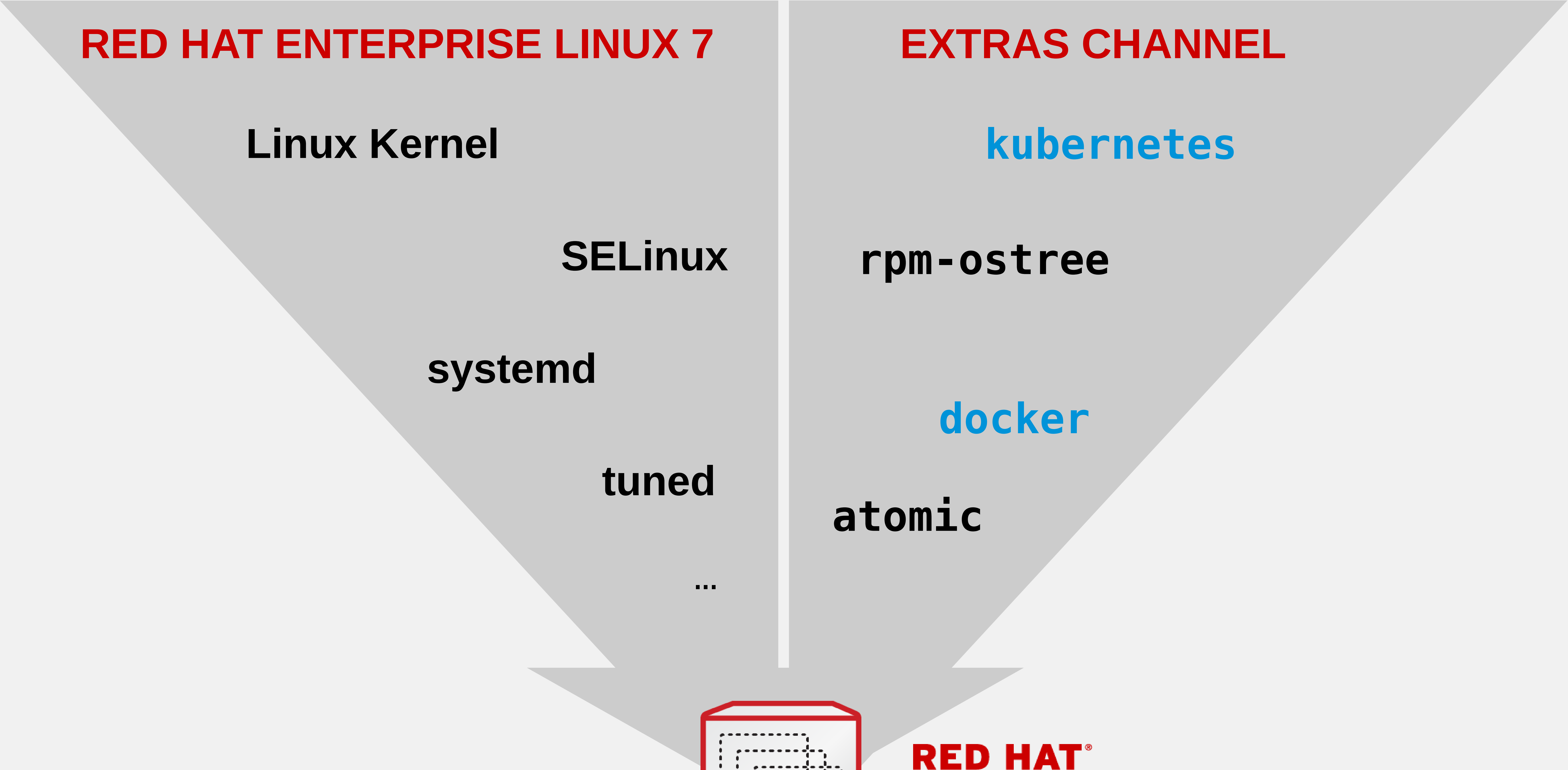
Fedora and CentOS

- Linux Kernel
- SELinux
- systemd
- tuned

New Additions

- kubernetes
- rpm-ostree
- docker
- atomic

Red Hat Enterprise Atomic Host



RED HAT ENTERPRISE LINUX 7

Linux Kernel

SELinux

systemd

tuned

...

EXTRAS CHANNEL

`kubernetes`

`rpm-ostree`

`docker`

`atomic`

RED HAT®
ENTERPRISE LINUX®
ATOMIC HOST

Red Hat Enterprise 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 for.

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.

Red Hat Enterprise 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

Red Hat Enterprise Atomic Host

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

- Performing an upgrade

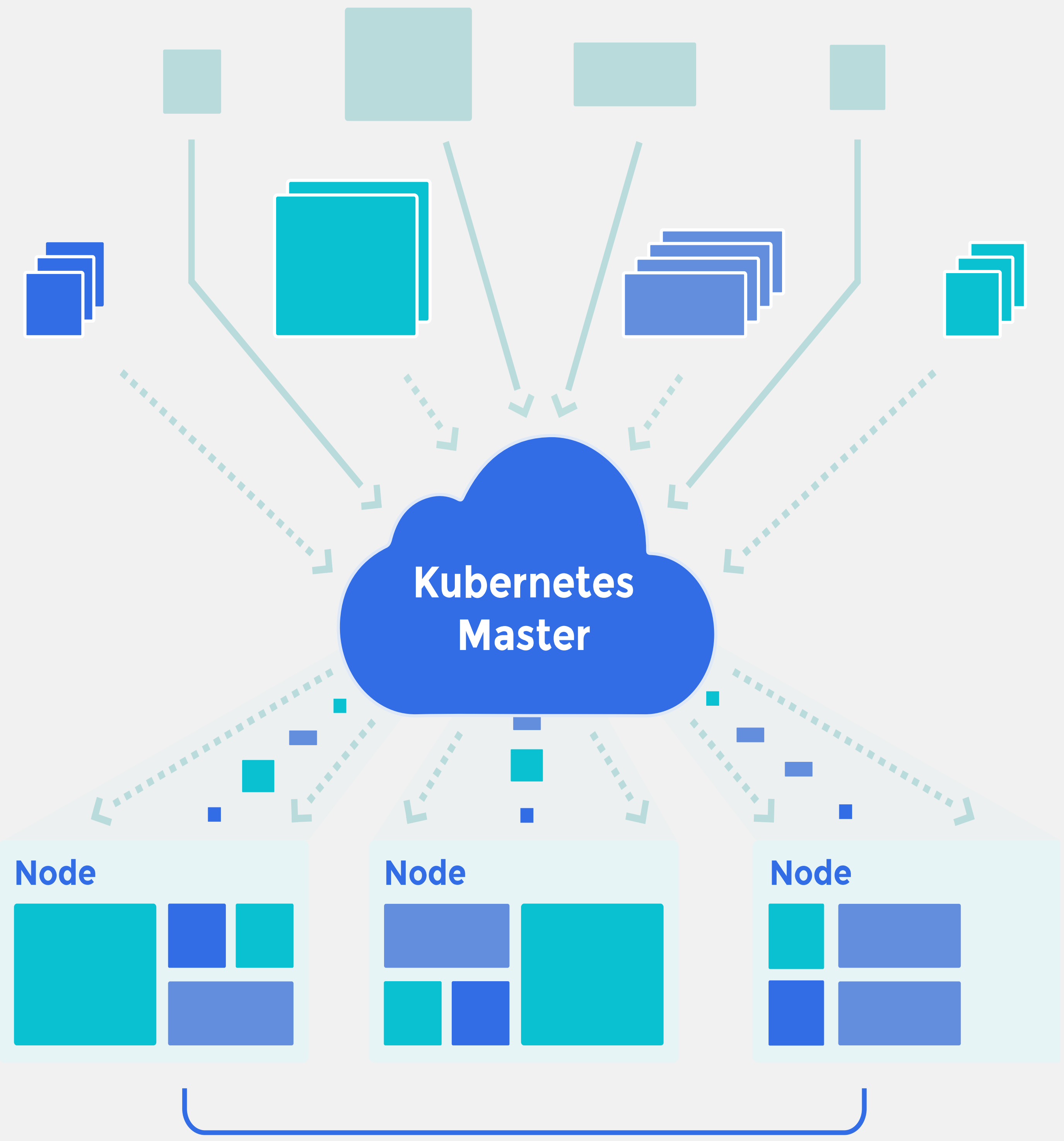
```
# atomic host upgrade
Updating from: rhel-atomic-host-ostree:rhel-atomic-host/7/x86_64/standard
```

- Checking status

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

* 2015-05-07 19:00:48     7.1.2       203dd666d3     rhel-atomic-host     rhel-atomic-host-
ostree:rhel-atomic-host/7/x86_64/standard
  2015-04-02 20:14:06     7.1.1-1     21bd99f9f3     rhel-atomic-host     rhel-atomic-host-
ostree:rhel-atomic-host/7/x86_64/standard
```

What about orchestration?

Kubernetes



Scheduled and packed dynamically onto nodes

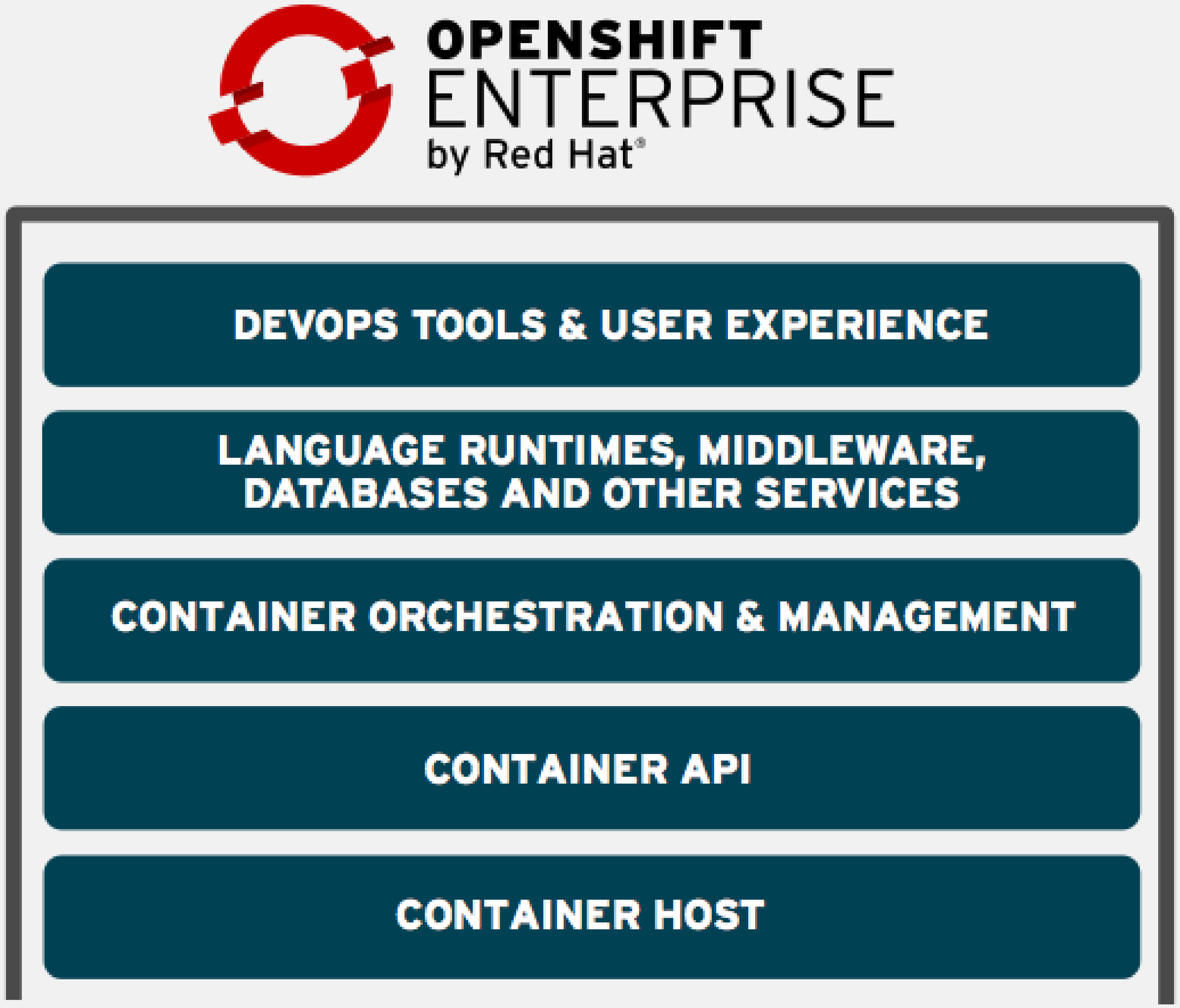
kubernetes
by Google

RED HAT ATOMIC

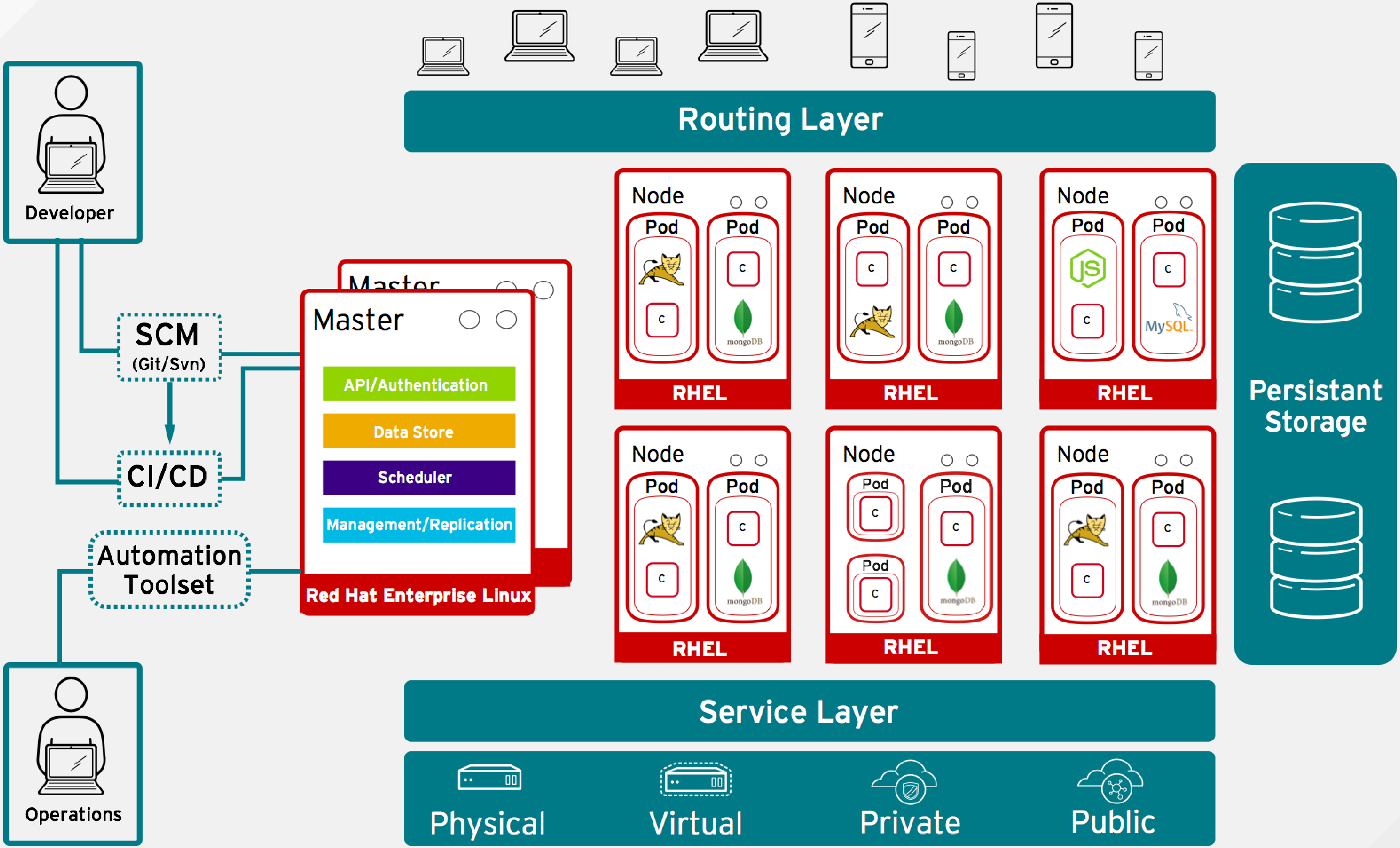
- Distributed orchestration for containers
- “Pod” - Set of containers that share pid, network, IPC, and UTS namespace.
  - Are scheduled to nodes as an unit
- “Service” - Set of one or more Pods and a policy to access them
- Replication Controller manages pods
- Node level proxy load balances and proxies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

Bringing it all together.

OpenShift 3

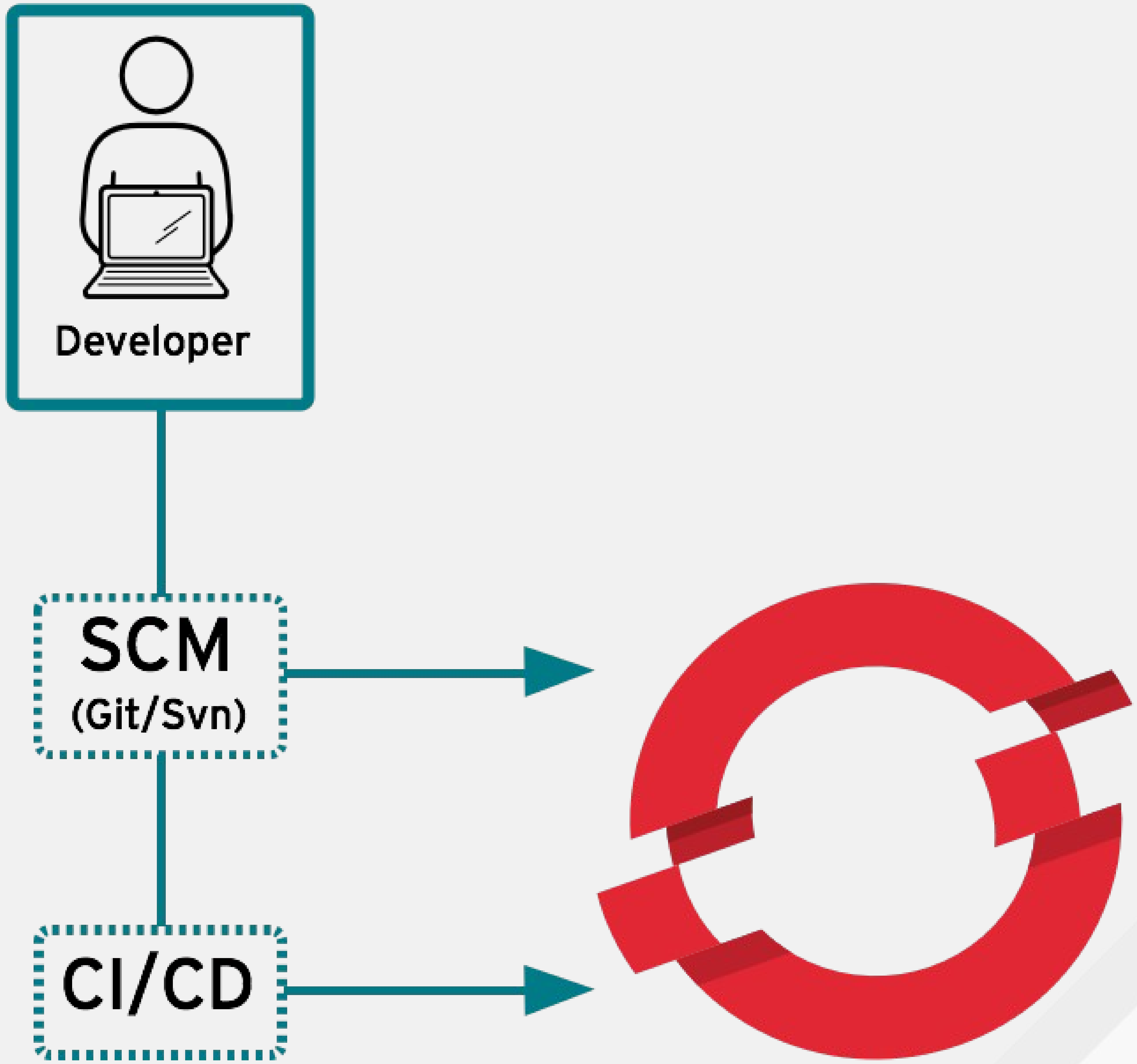

OPENSHIFT ENTERPRISE
by Red Hat®
DEVOPS TOOLS & USER EXPERIENCE
LANGUAGE RUNTIMES, MIDDLEWARE, DATABASES AND OTHER SERVICES
CONTAINER ORCHESTRATION & MANAGEMENT
CONTAINER API
CONTAINER HOST

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

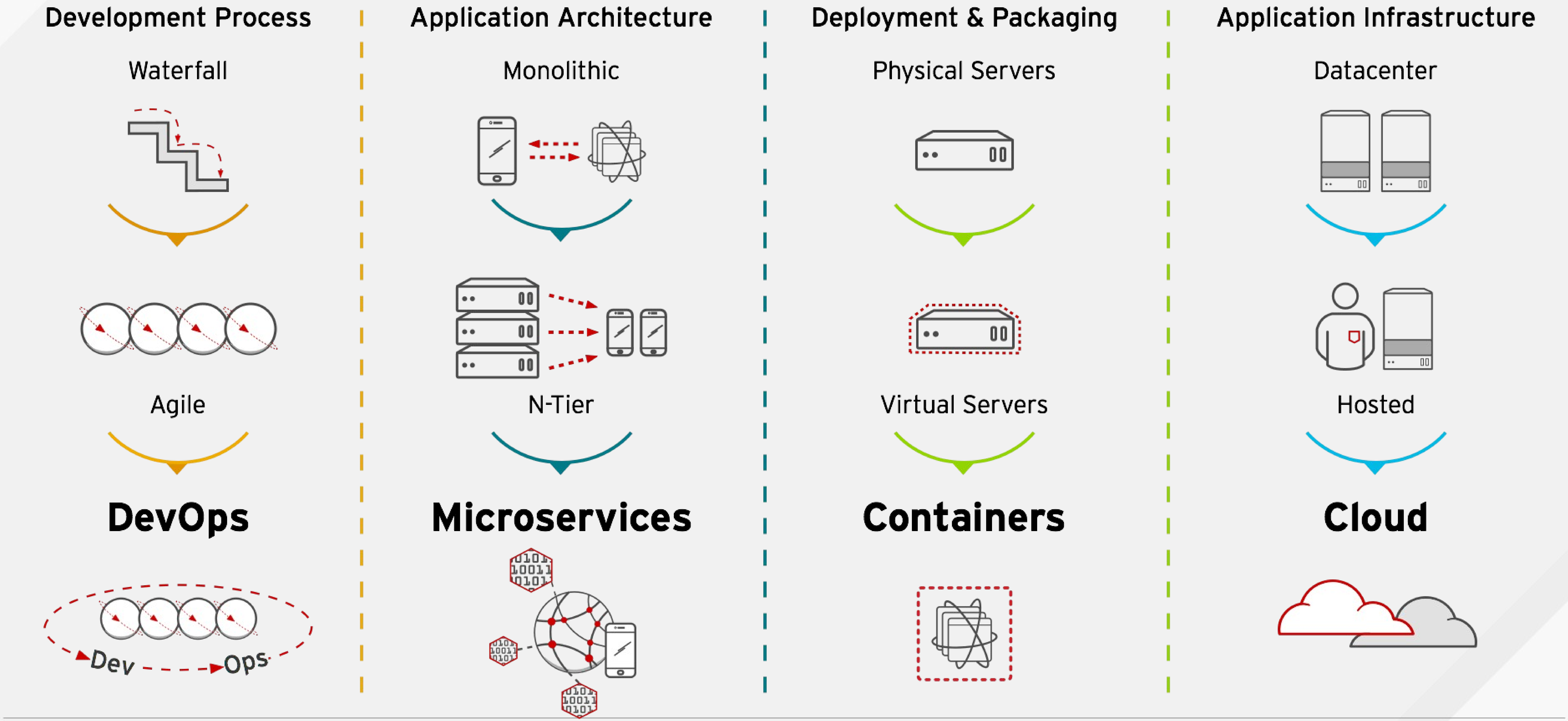

Developer
SCM
(Git/Svn)
CI/CD
Automation Toolset
Operations
Master
Master
API/Authentication
Data Store
Scheduler
Management/Replication
Red Hat Enterprise Linux
Routing Layer
Node
Pod
Pod
c
c
mongoDB
RHEL
Node
Pod
Pod
c
c
mongoDB
RHEL
Node
Pod
Pod
c
c
MySQL
RHEL
Node
Pod
Pod
c
c
mongoDB
RHEL
Node
Pod
Pod
Pod
c
c
c
mongoDB
RHEL
Node
Pod
Pod
c
c
mongoDB
RHEL
Persistant Storage
Service Layer
Physical
Virtual
Private
Public

Benefits for Developers

- Access a broad selection of application components
- Deploy application environments on-demand
- Leverage your choice of interface & integrate with existing tools
- Automate application deployments, builds and source-to-image
- Enable collaboration across users, teams & projects
- Full application lifecycle from Dev all the way to Production



IT Must Evolve to Stay Ahead of Demands



Development Process

Waterfall

Agile

DevOps

Application Architecture

Monolithic

N-Tier

Microservices

Deployment & Packaging

Physical Servers

Virtual Servers

Containers

Application Infrastructure

Datacenter

Hosted

Cloud

Questions?

THANK YOU!

Adam Miller
@TheMaxamillion
admiller@redhat.com

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/