A DevOps State of Mind: Continuous Security with Kubernetes

Chris Van Tuin
Red Hat
Chief Technologist, NA West / Silicon Valley
cvantuin@redhat.com

“Only the paranoid survive”

- Andy Grove, 1996

THE WORLD IS AUTOMATING

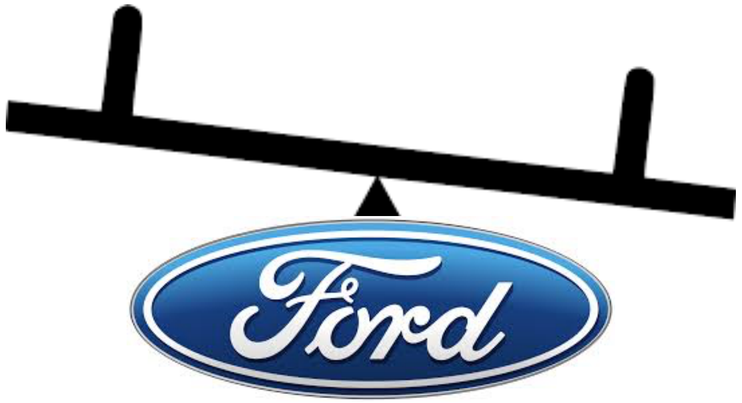
Those who succeed in automation will win


LEONI
MLN-060RB1
MLN-060RB5
STÄUBLI

THE CHALLENGE:
ENABLE INNOVATION AT SPEED, WHILE EXECUTING AT SCALE WITH EFFICIENCY

Old

Innovation Execution



Static &
Planned

New

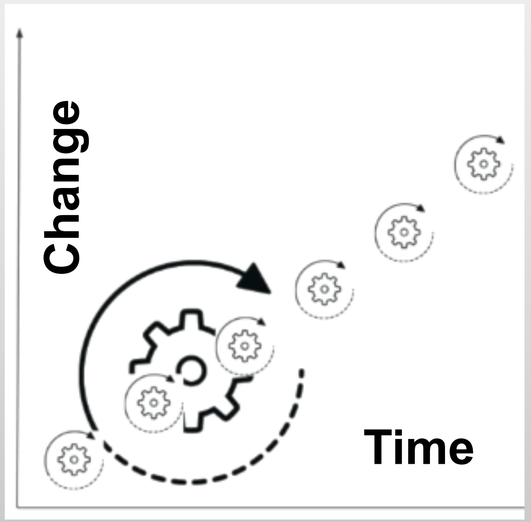
Innovation Execution

Dynamic &
Policy Driven

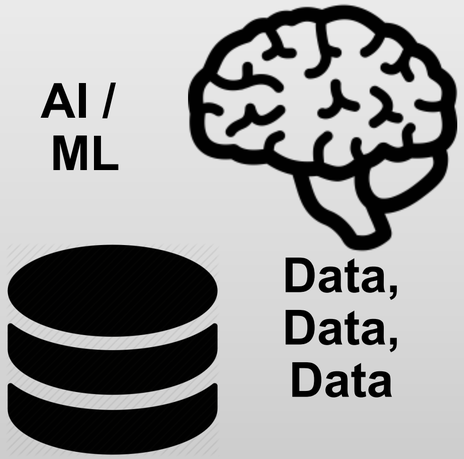
redhat.

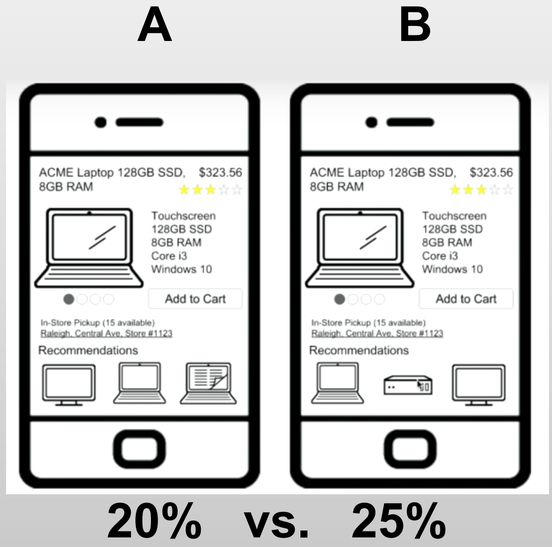
IT'S NOT JUST SOFTWARE, THE DIGITAL LEADERS =

Speed Up Innovation



Shorten the Feedback Loop





Empowered organization

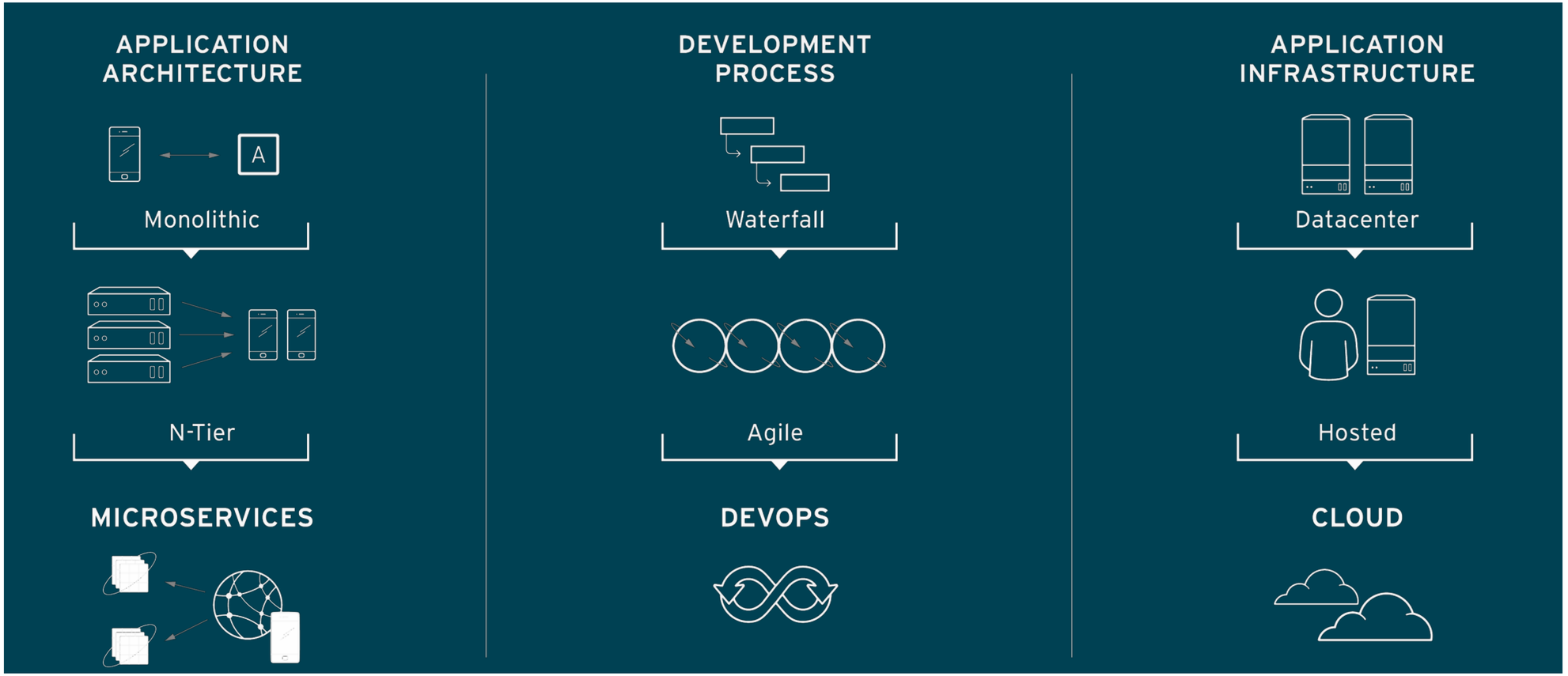
Move Fast, Break Things

Real-time data-driven intelligence & personalization

Culture of experimentation

redhat.

IT MUST EVOLVE & KEEP UP



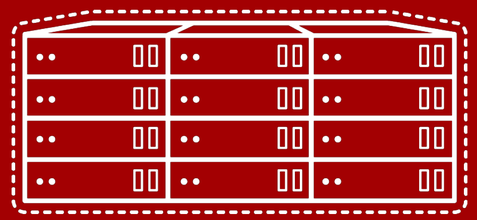
redhat.

SECURING THE ENTERPRISE IS HARDER THAN EVER

The way we develop, deploy and manage IT is changing dramatically led by DevOps, Cloud Native Applications, and Hybrid Cloud

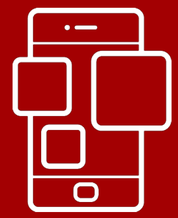
Menacing threat landscape

Dissolving security perimeter



Software-defined infrastructure

Cloud computing



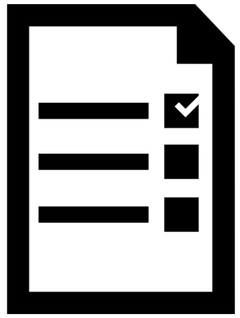
Applications & devices outside of IT control

TRADITIONAL NETWORK-BASED DEFENSES ARE NO LONGER ENOUGH

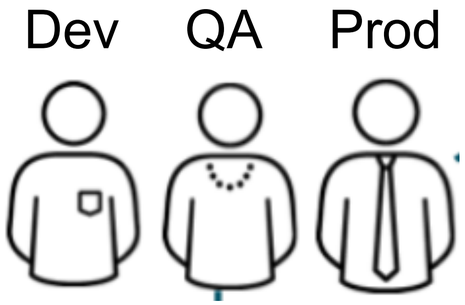
redhat.

DEVSECOPS

Reduce Risks, Lower Costs, Speed Delivery, Speed Reaction



Security Automation

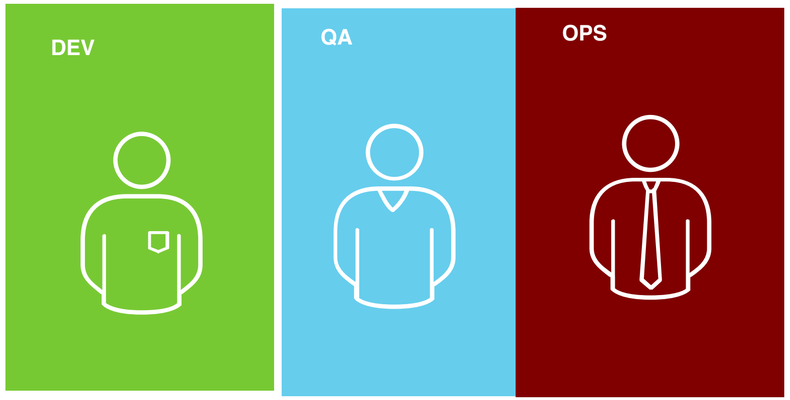


Process Optimization

Continuous Security Improvement

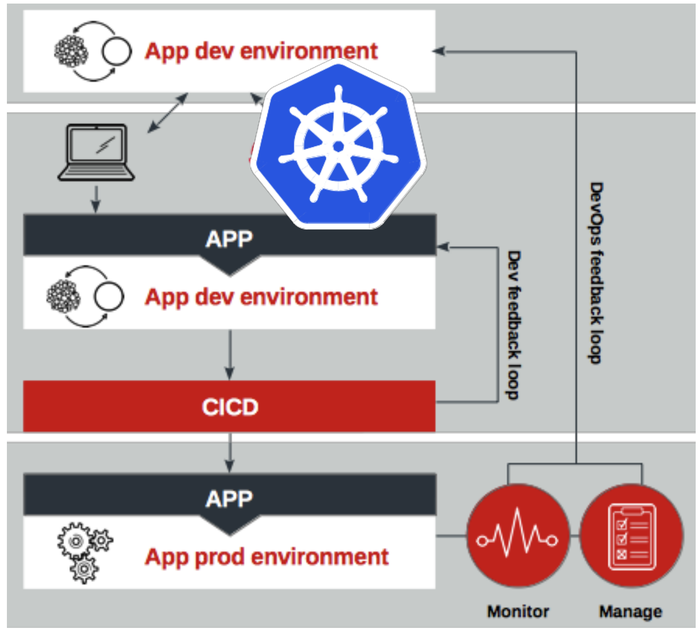
DEVSECOPS

Culture



+

Process



+

Technology

Cloud Native Applications

Linux + Containers
IaaS
Orchestration
CI/CD
Source Control Management
Collaboration
Build and Artifact Management
Testing
Frameworks

Open Source

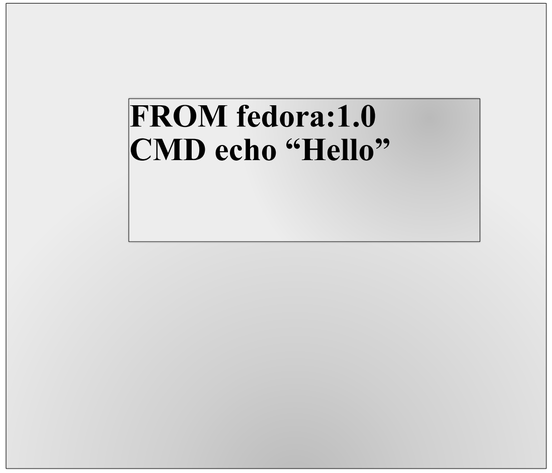
Hybrid Cloud

Security

redhat.

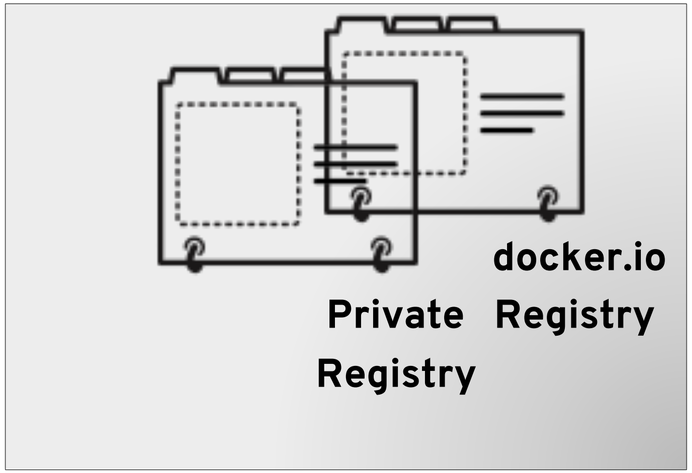
CONTAINERS ENABLE DEVSECOPS

Build



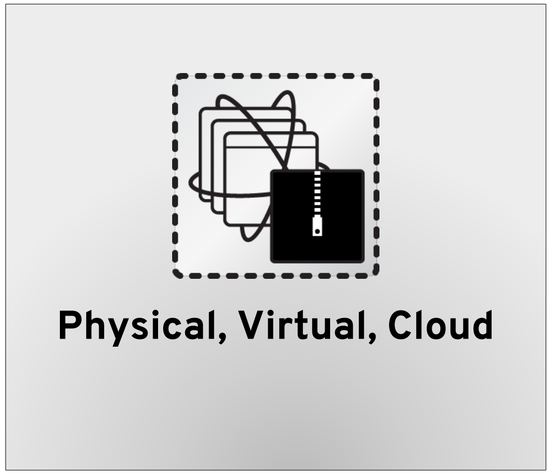
Build file

Ship



Container Image

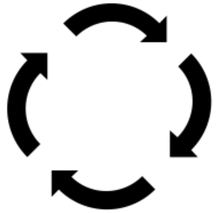
Run



Container Instance

CONTAINERS AT SCALE

Scheduling



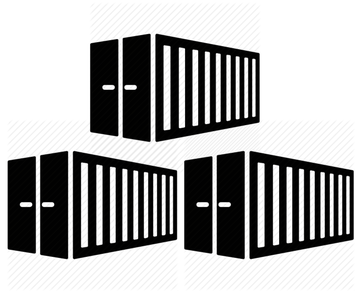
Lifecycle & health

Discovery

Monitoring

Scaling

Persistence



Aggregation

Security

BARE METAL

VIRTUAL

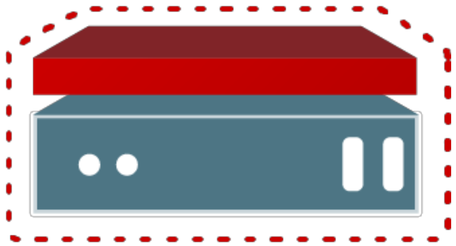
PRIVATE CLOUD

PUBLIC CLOUD

kubernetes

Security Platform

BARE METAL



VIRTUAL

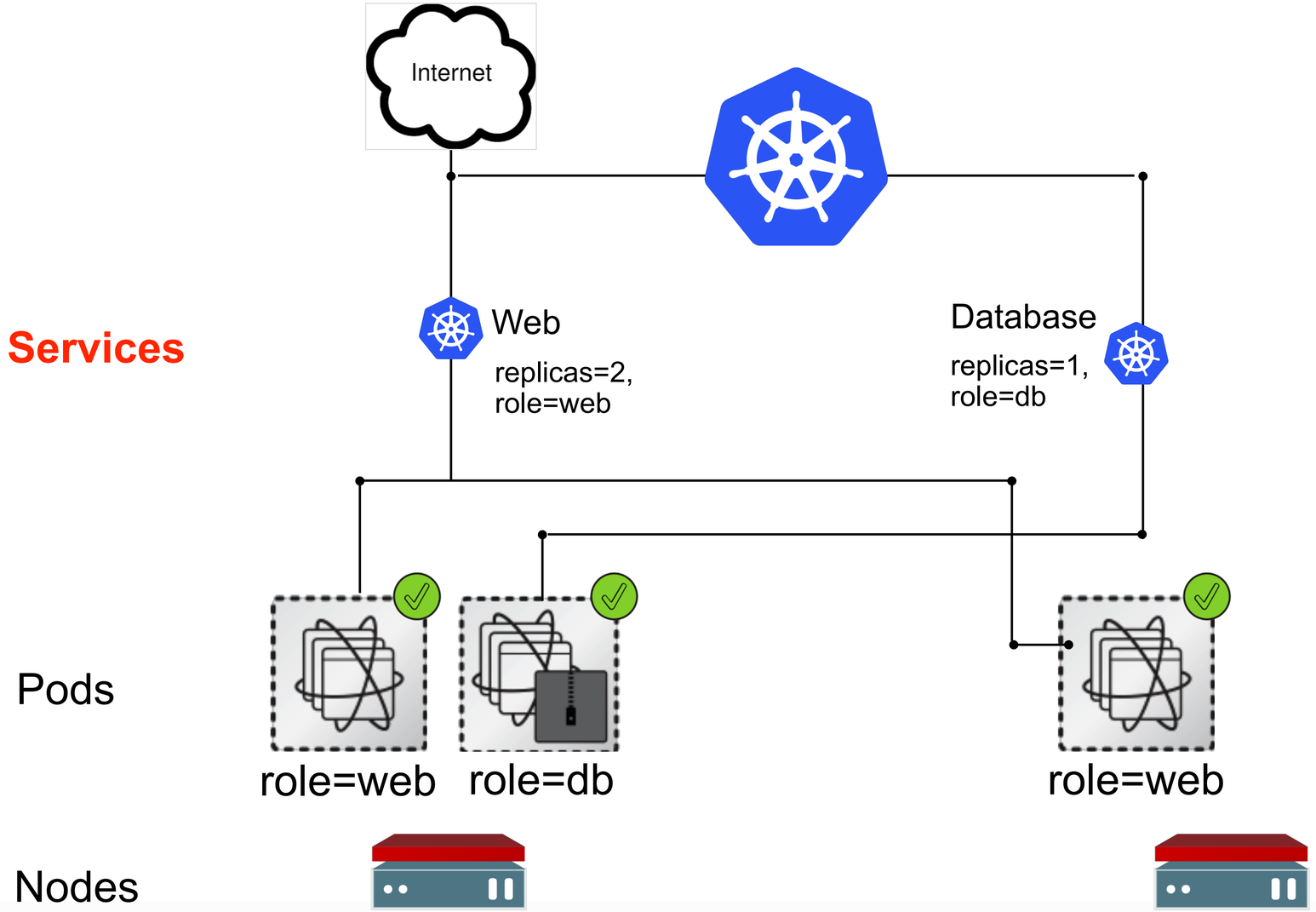
PRIVATE CLOUD

PUBLIC CLOUD

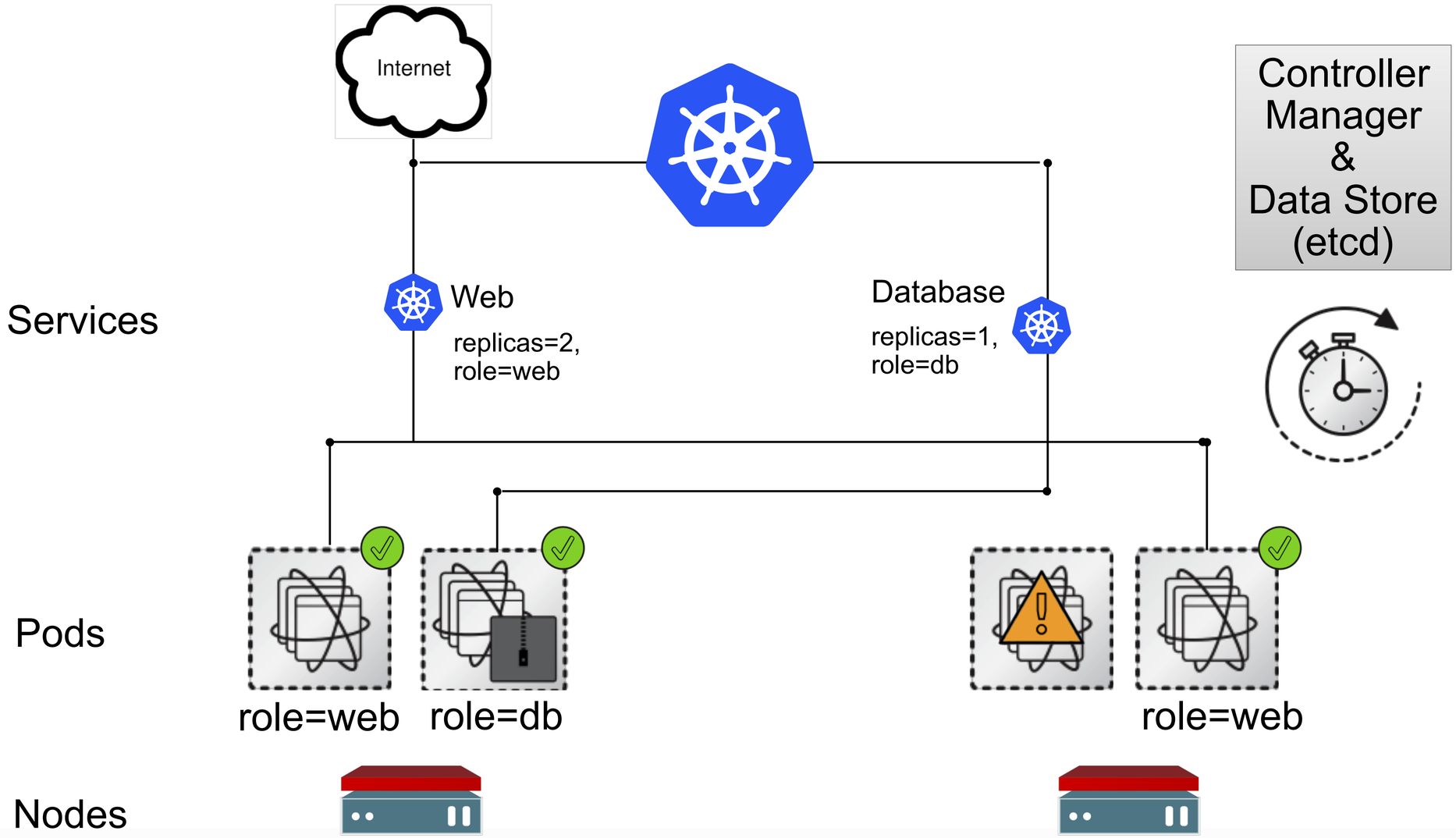
AUTOMATION

ORCHESTRATION

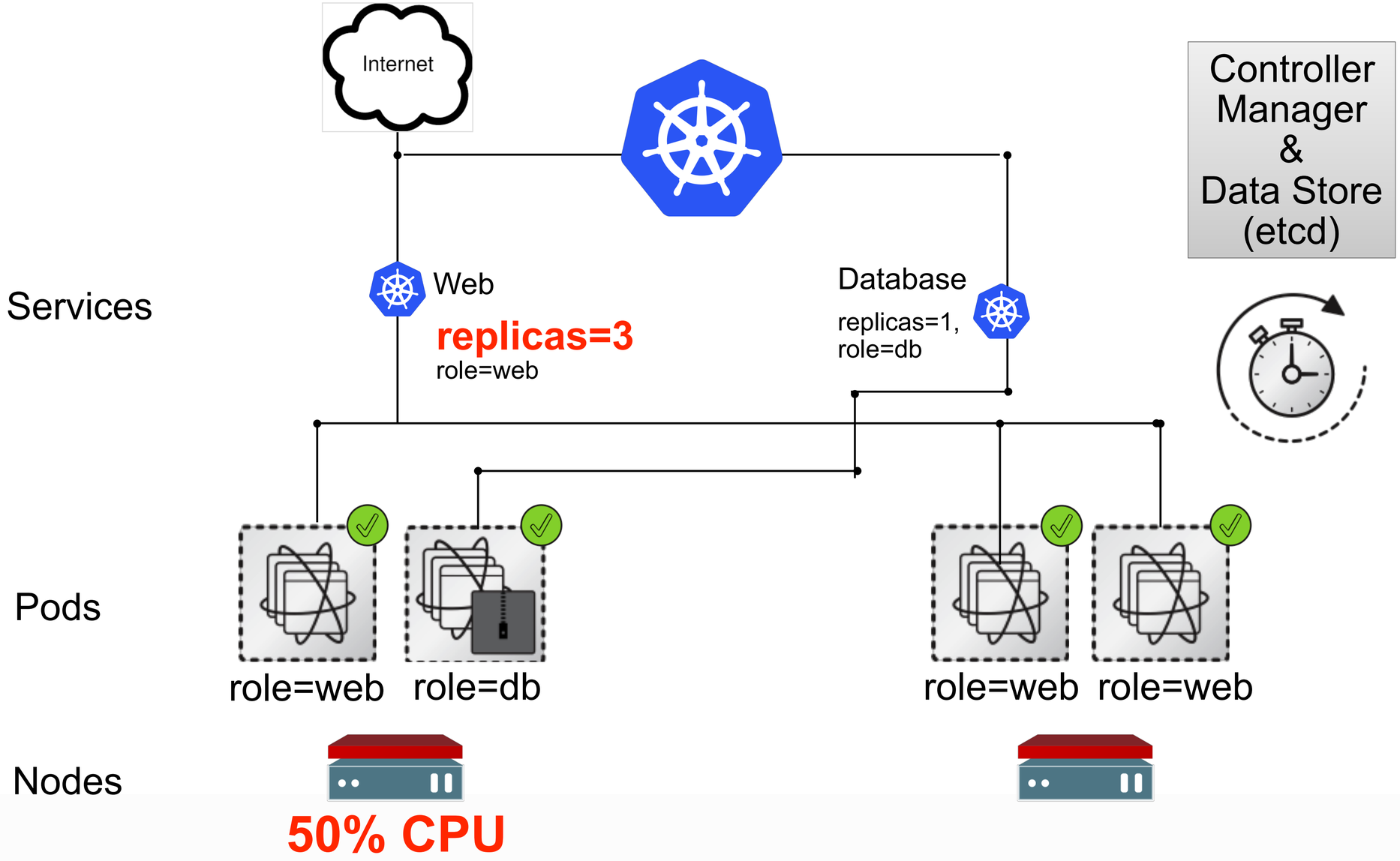
Deployment, Declarative



HEALTH CHECK

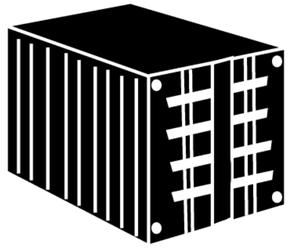


AUTO-SCALE

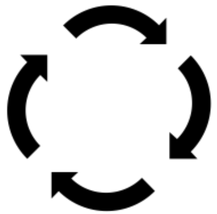


redhat.

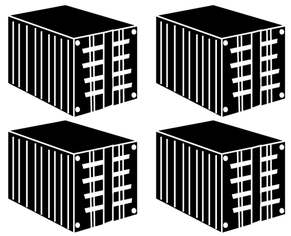
SECURING YOUR CONTAINER ENVIRONMENT



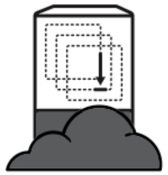
Images



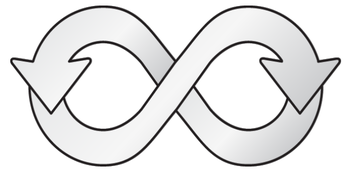
Builds



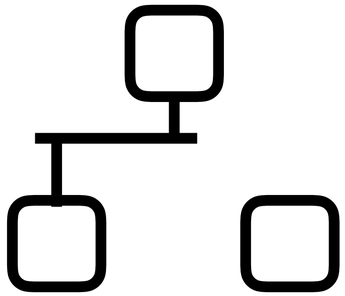
Registry



Container host



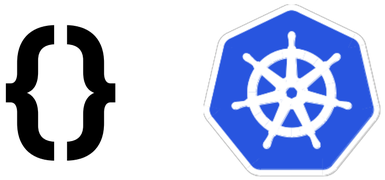
CI/CD



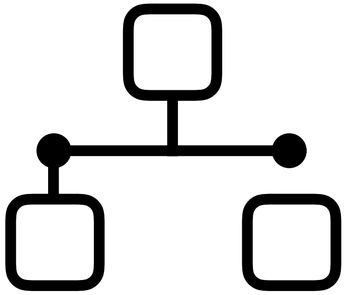
Network isolation

Monitoring & Logging

Storage



API & Platform access

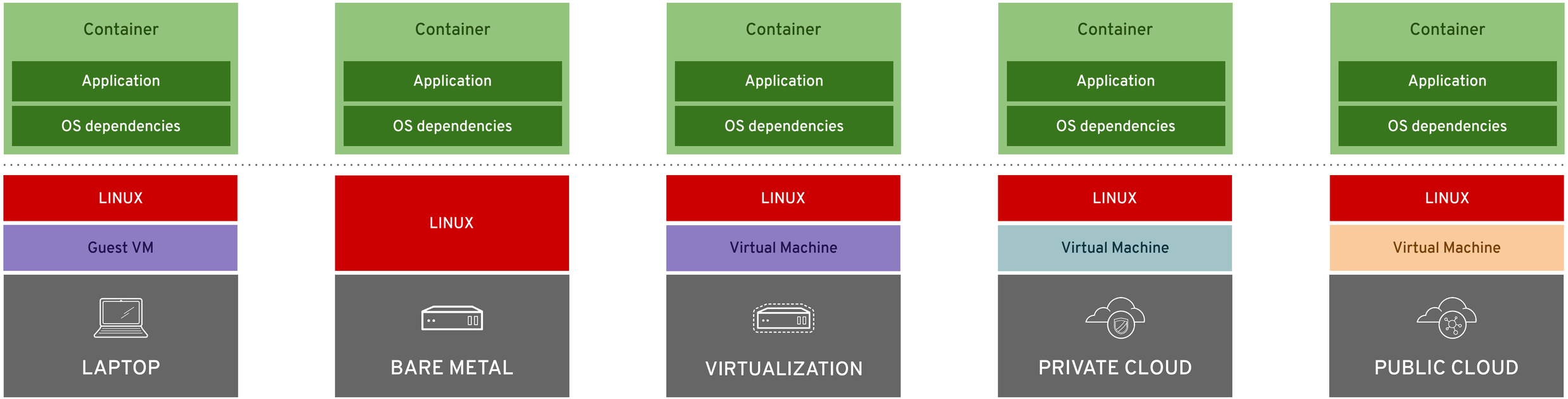


Federated clusters

CONTAINER IMAGES

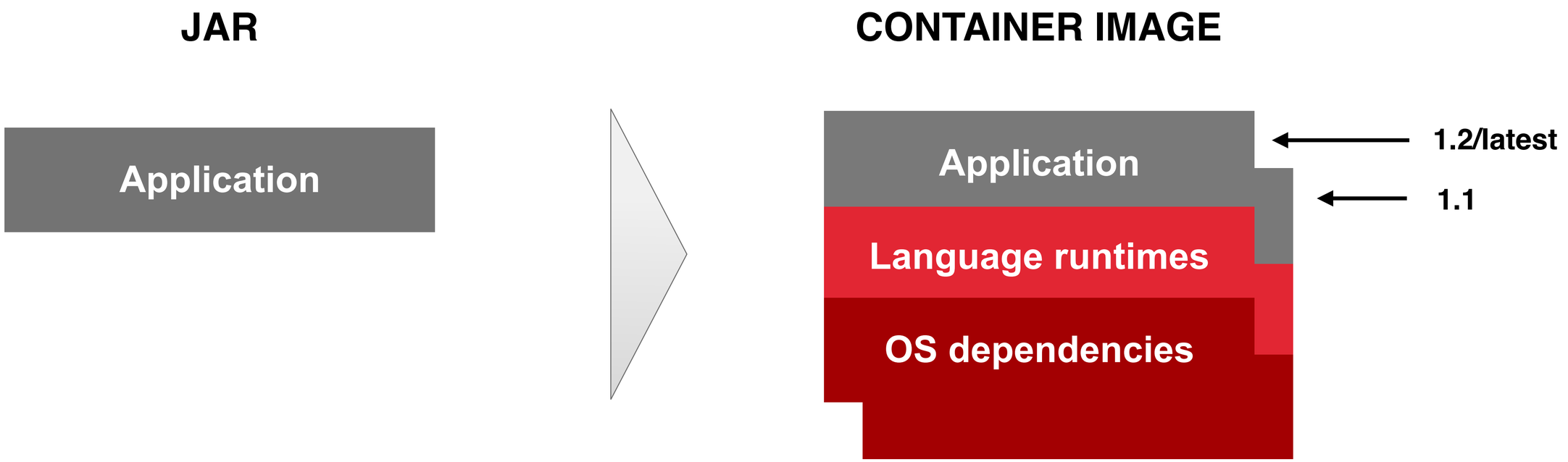
CONTAINERS - Build Once, Deploy Anywhere

Reducing Risk and Improving Security with Improved Consistency

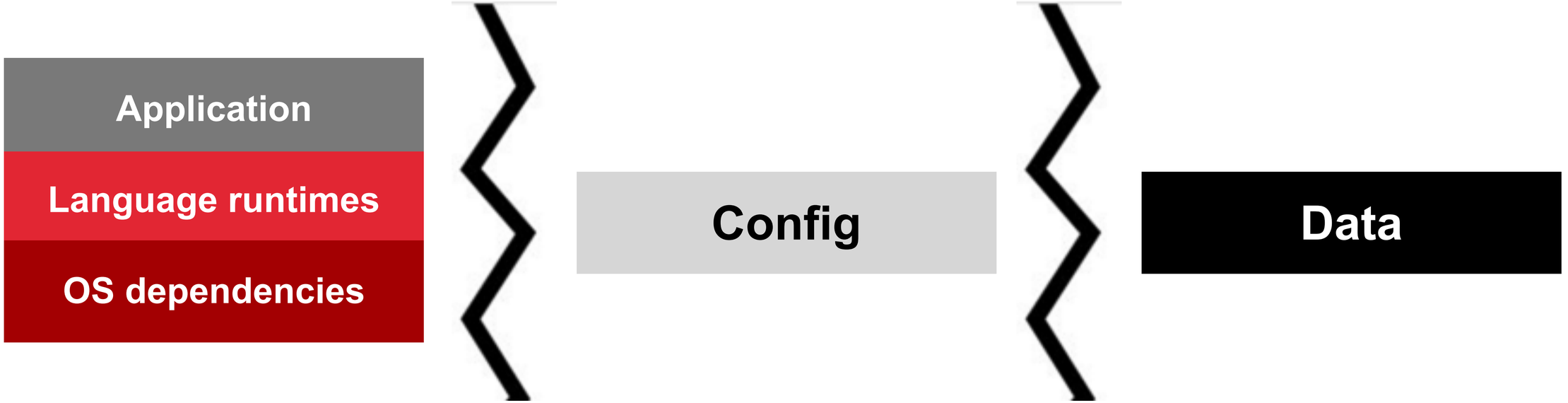


redhat.

CONTAINER IMAGE



TREAT CONTAINERS AS IMMUTABLE



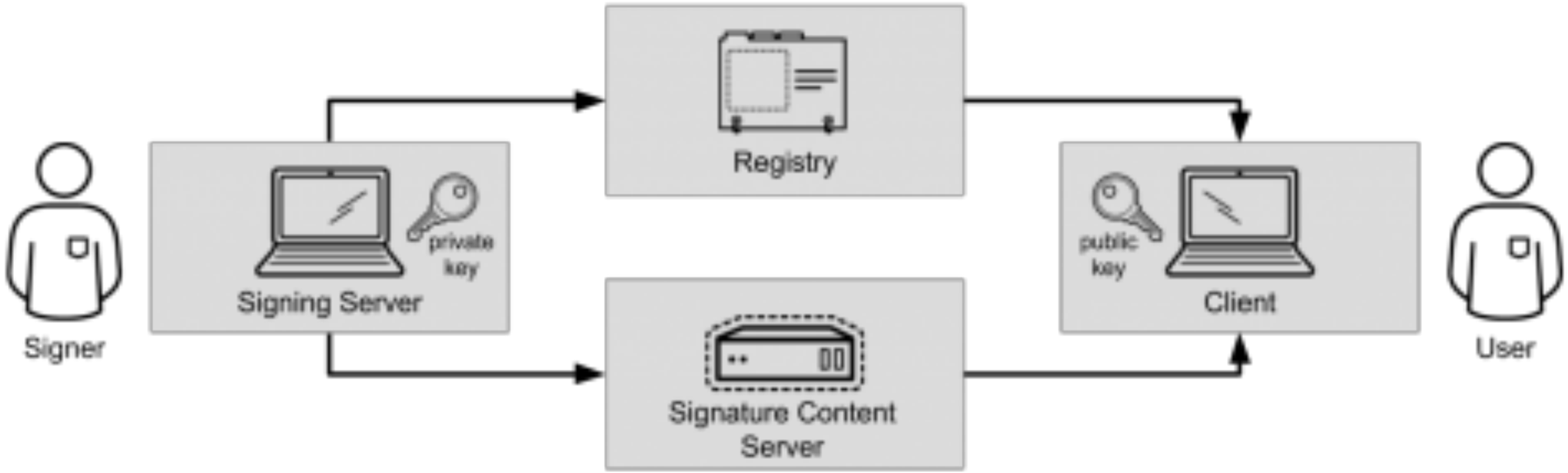
Container image

Kubernetes configmaps secrets

Traditional data services, Kubernetes persistent volumes

CONTAINER IMAGE SIGNING

Validate what images and version are running

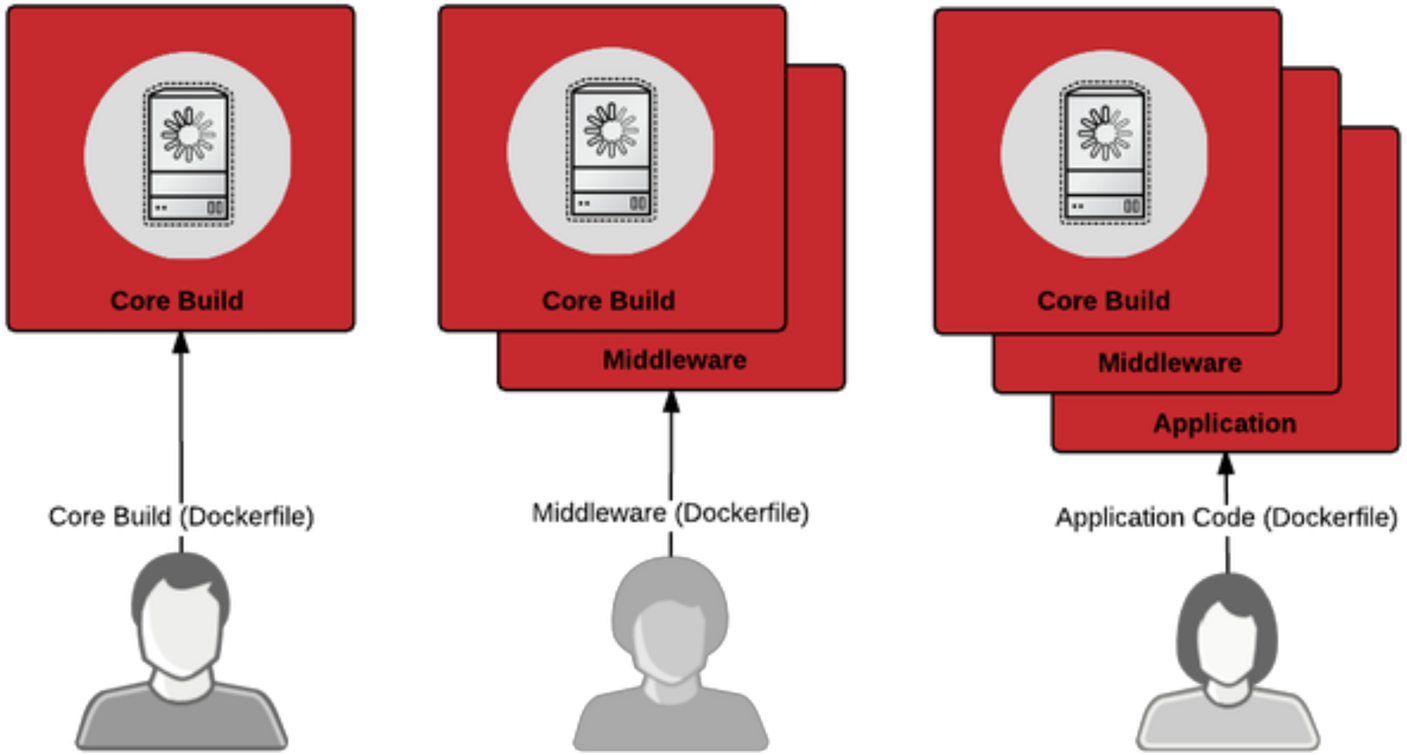


- Authenticating authorship
- Non-repudiation
- Ensuring image integrity

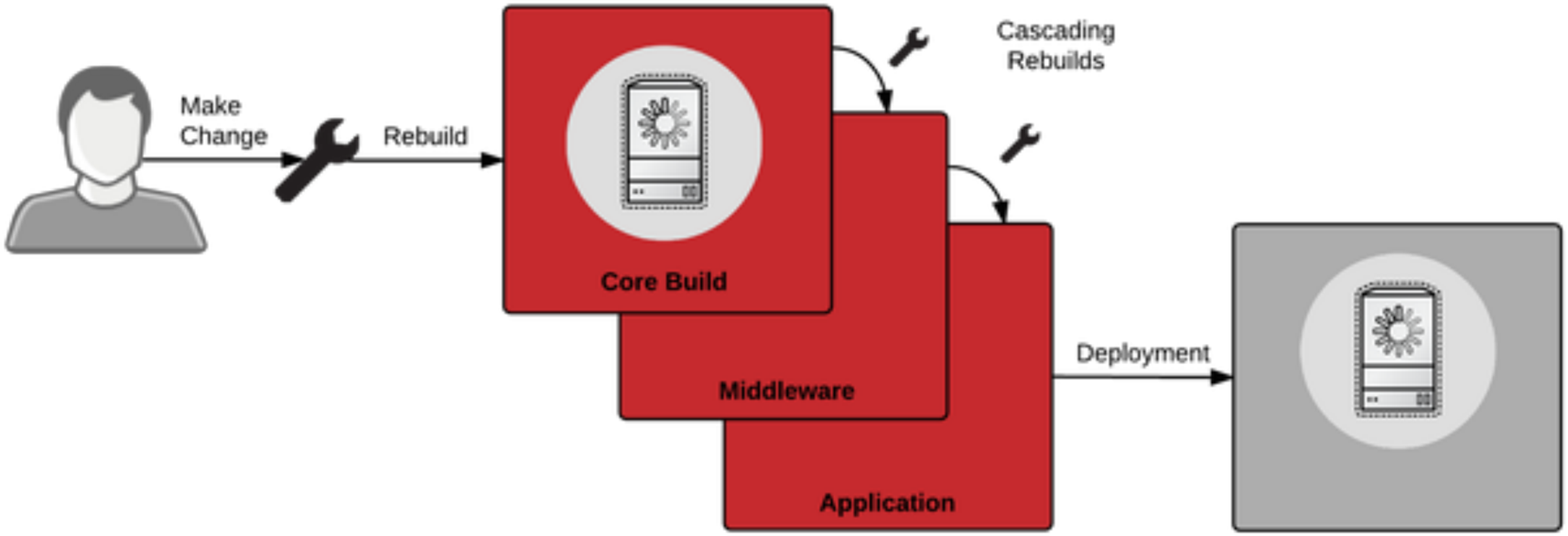
redhat.

CONTAINER BUILDS

A CONVERGED SOFTWARE SUPPLY CHAIN



CUSTOM SUPPLY CHAIN



BUILD FILE BEST PRACTICES

Build file

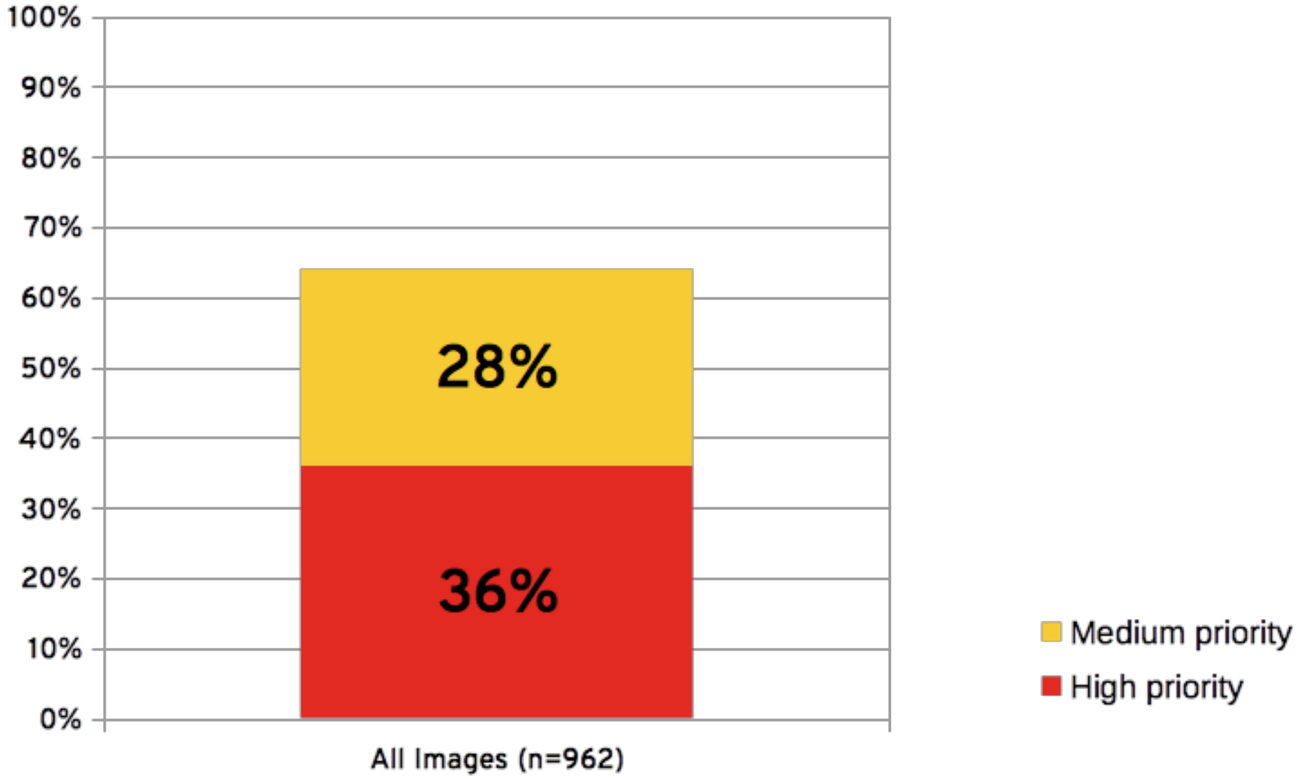
```
FROM registry.redhat.com/rhel7
RUN groupadd -g 999 appuser && \
    useradd -r -u 999 -g appuser appuser
USER appuser
CMD echo “Hello”
```

- Treat build file as a Blueprint
- Version control build file
- Don’t login to build/configure
- Be explicit with versions, not latest
- Always list registry pulling FROM
- Specify USER, default is root
- Each Run creates a new layer

CONTAINER REGISTRY SECURITY

WHAT'S INSIDE THE CONTAINER MATTERS

64% of official images in Docker Hub
contain high priority security vulnerabilities



examples:

ShellShock (bash)

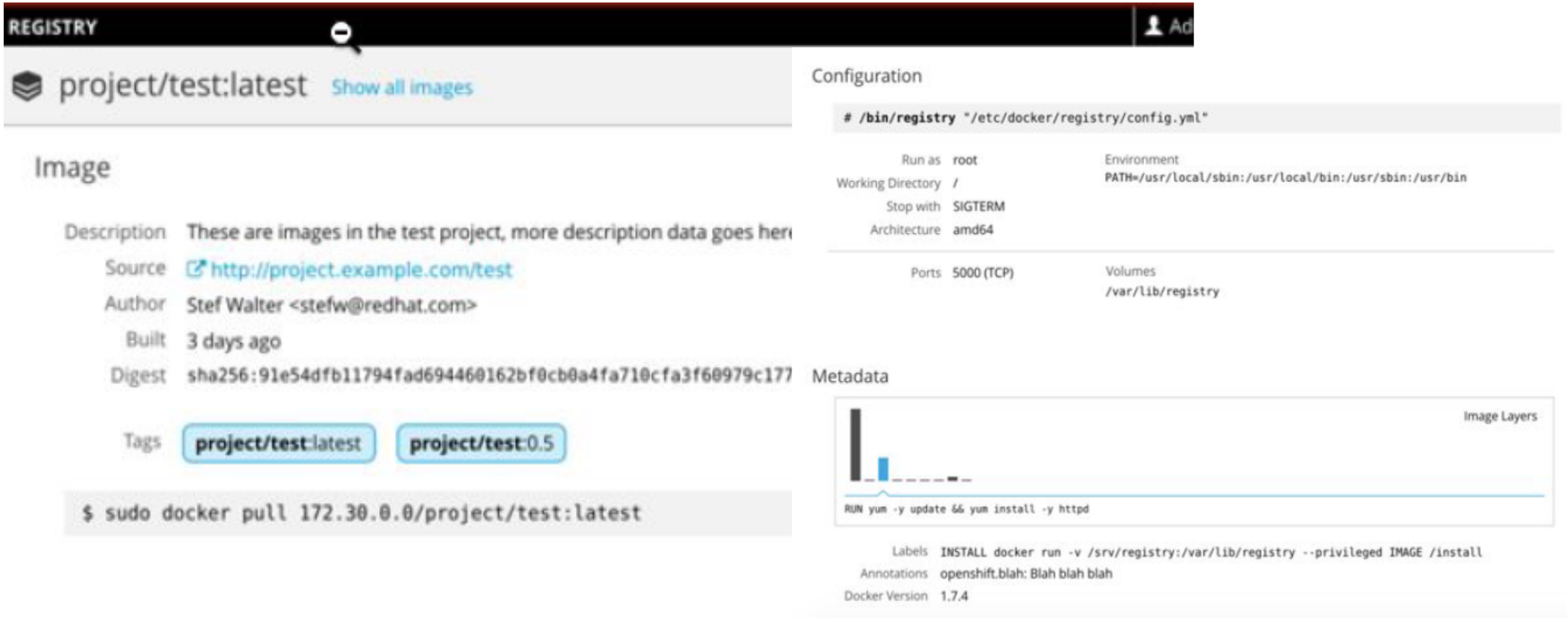
Heartbleed (OpenSSL)

Poodle (OpenSSL)

Source: *Over 30% of Official Images in Docker Hub Contain High Priority Security Vulnerabilities*, Jayanth Gummaraju, Tarun Desikan, and Yoshio Turner, BanyanOps, May 2015 (http://www.banyanops.com/pdf/BanyanOps-AnalyzingDockerHub-WhitePaper.pdf)

redhat.

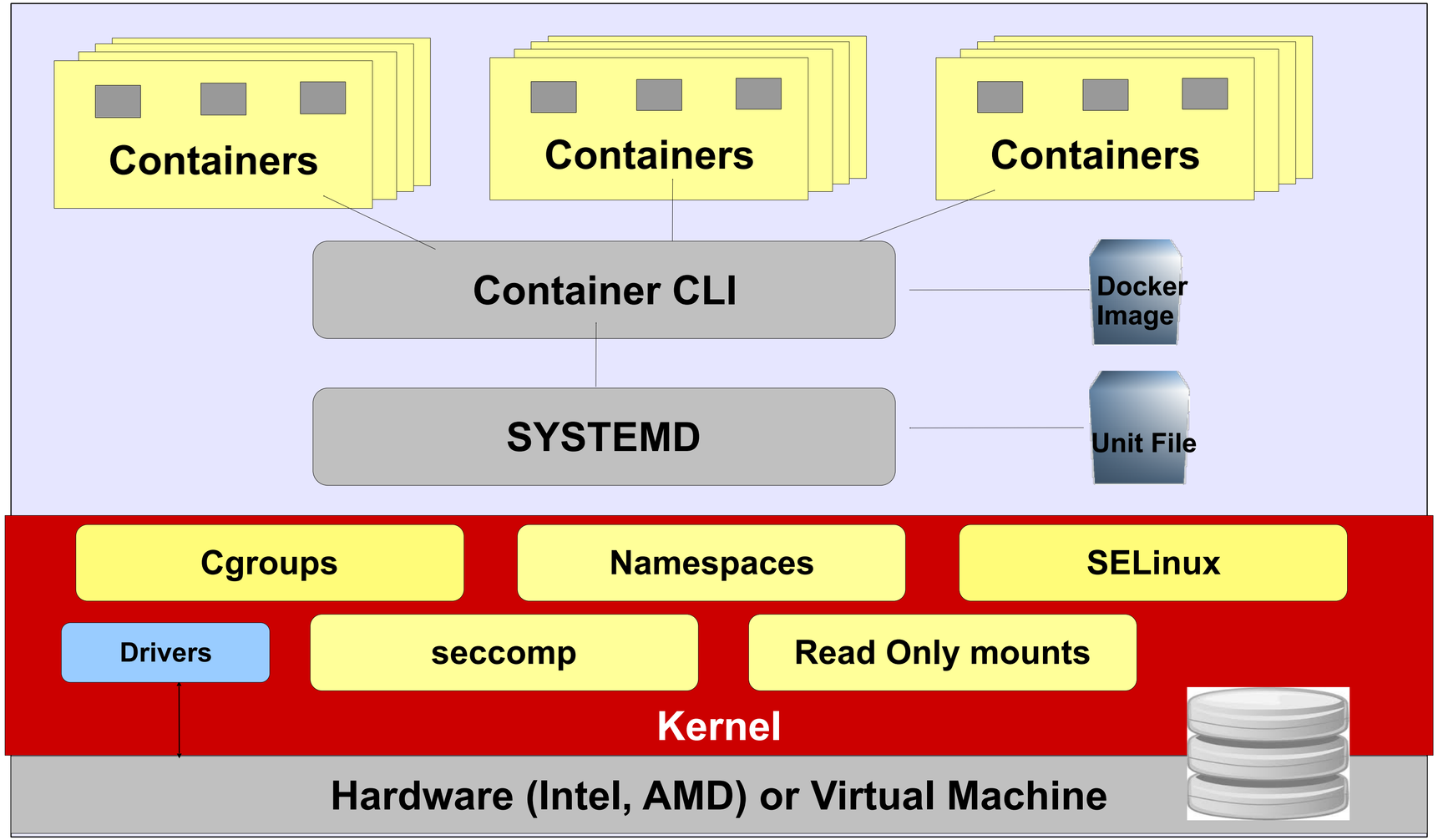
PRIVATE REGISTRY



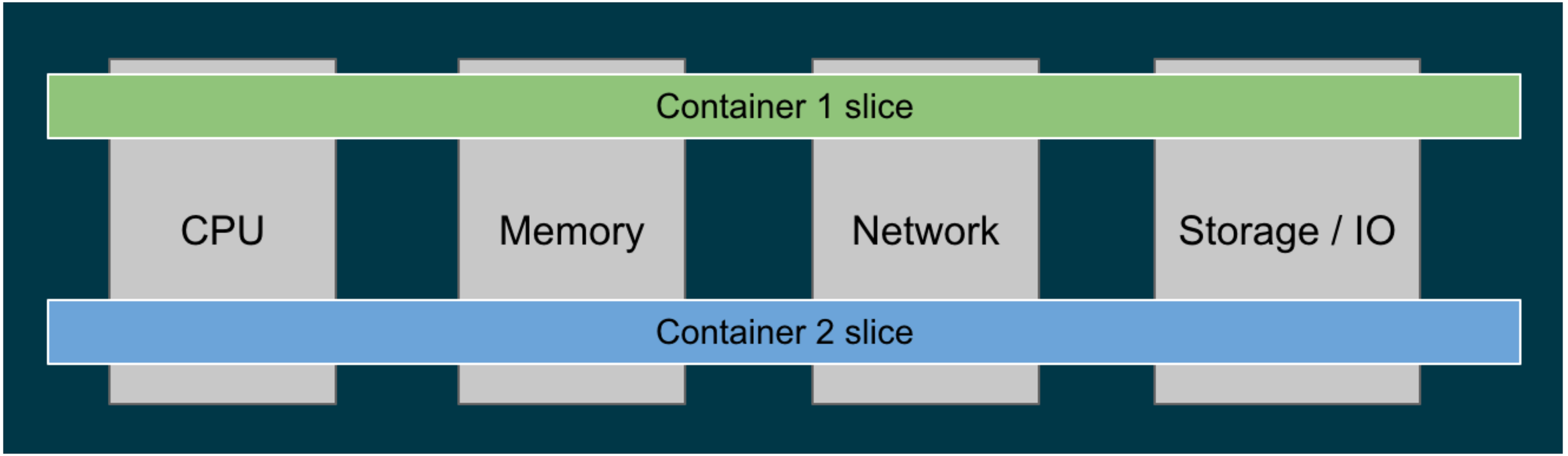
redhat.

CONTAINER HOST SECURITY

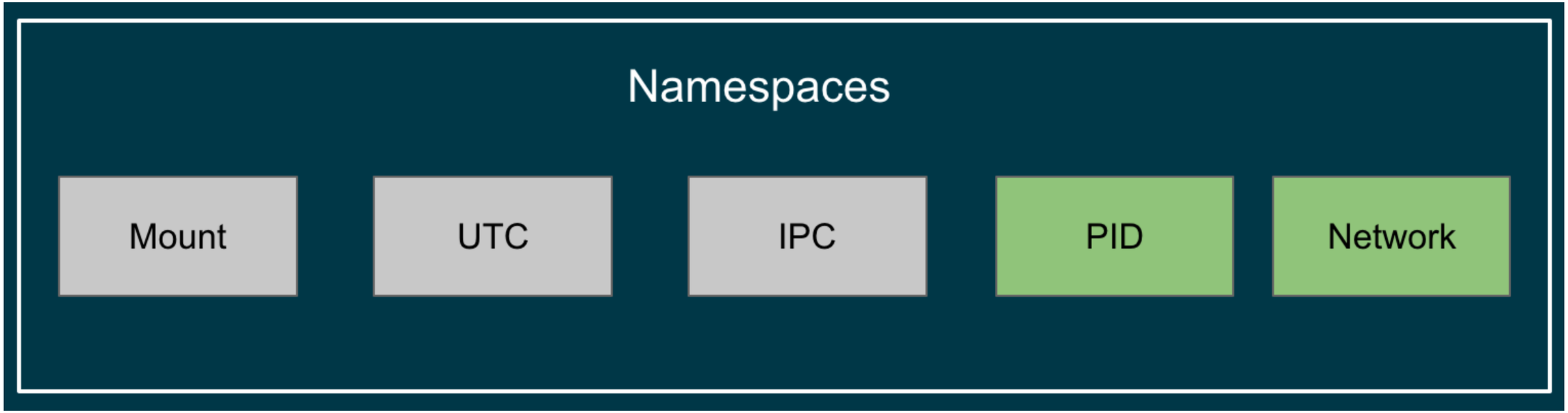
CONTAINERS ARE LINUX



CGROUPS - RESOURCE ISOLATION



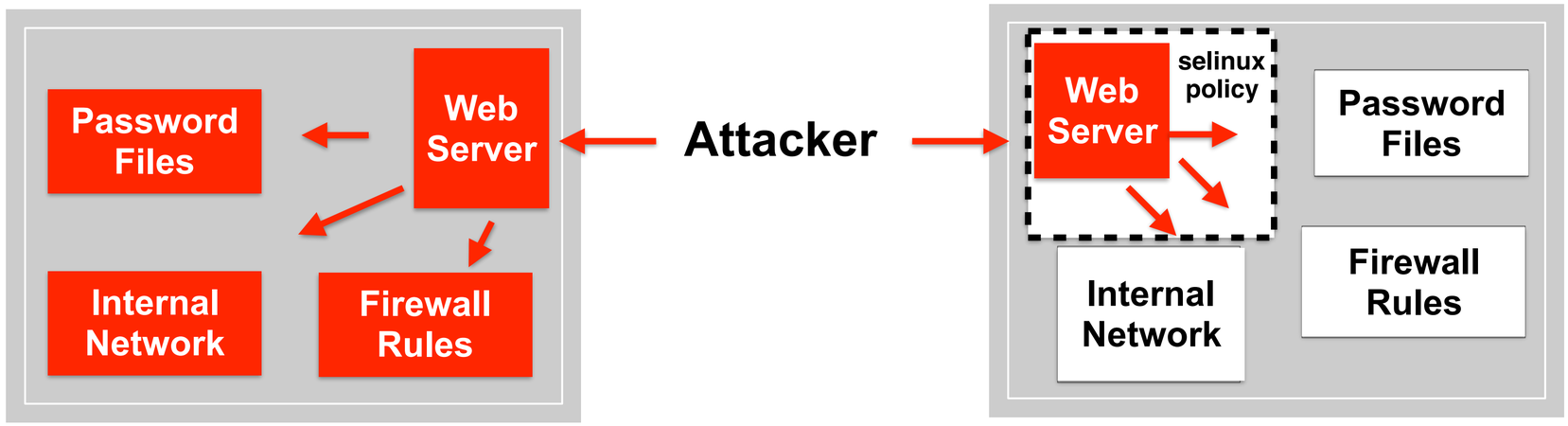
NAMESPACES - PROCESS ISOLATION



SELINUX - MANDATORY ACCESS CONTROLS

Discretionary Access Controls (file permissions)

Mandatory Access Controls (selinux)



SECCOMP AND LINUX CAPABILITIES

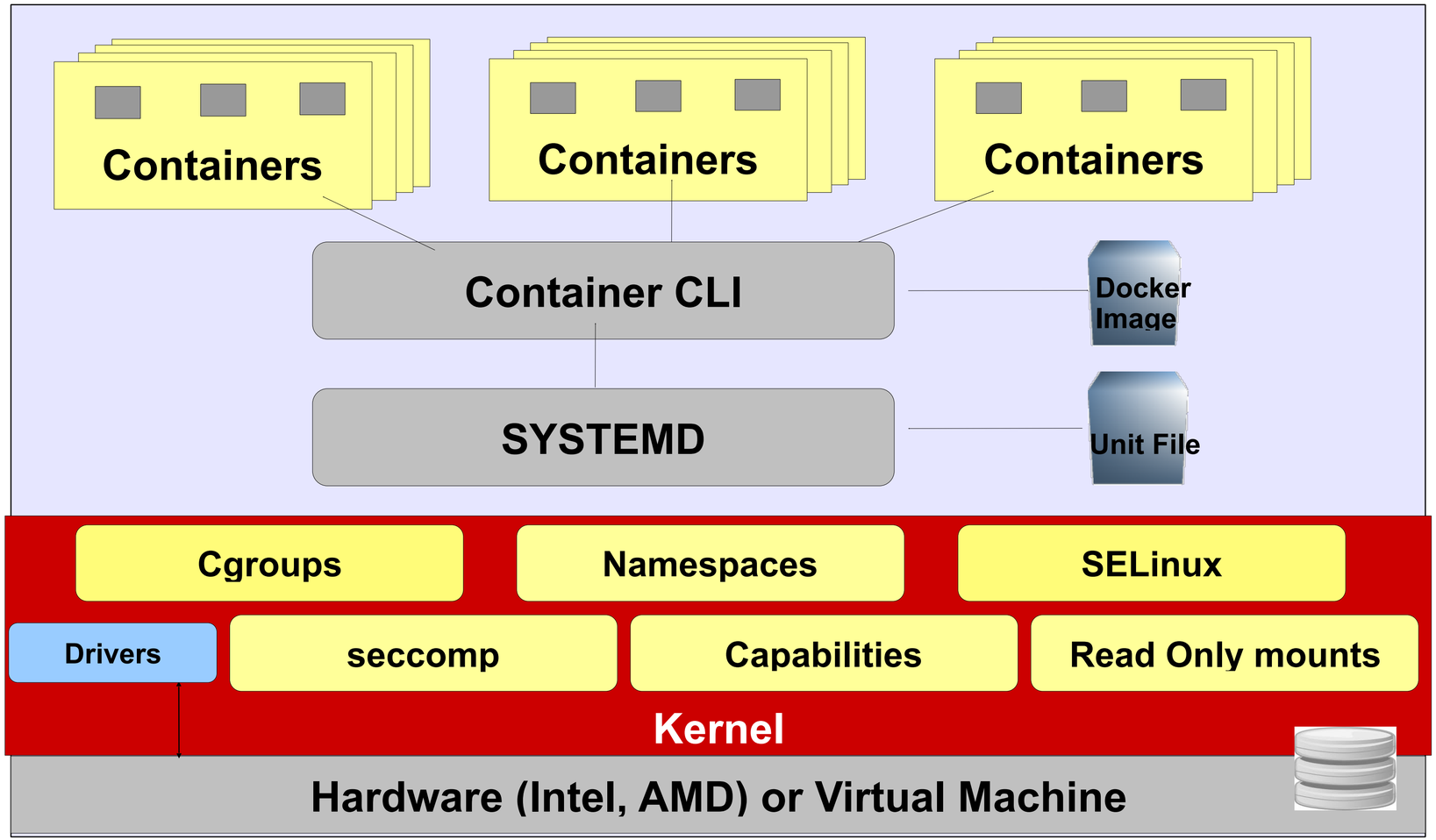
FILTERING SYSTEM CALLS and DROPPING PRIVILEGES

CAP_SETPCAP	Modify process capabilities
CAP_SYS_MODULE	Insert/Remove kernel modules
CAP_SYS_RAWIO	Modify Kernel Memory
CAP_SYS_PACCT	Configure process accounting
CAP_SYS_NICE	Modify Priority of processes
CAP_SYS_RESOURCE	Override Resource Limits
CAP_SYS_TIME	Modify the system clock
CAP_SYS_TTY_CONFIG	Configure tty devices
CAP_AUDIT_WRITE	Write the audit log
CAP_AUDIT_CONTROL	Configure Audit Subsystem
CAP_MAC_OVERRIDE	Ignore Kernel MAC Policy
CAP_MAC_ADMIN	Configure MAC Configuration
CAP_SYSLOG	Modify Kernel printk behaviour
CAP_NET_ADMIN	Configure the network: - Setting the hostname/domainname - mount(),unmount() - nfsservctl - ….
CAP_SYS_ADMIN	

redhat.

READ ONLY MOUNTS

CONTAINER HOST SECURITY



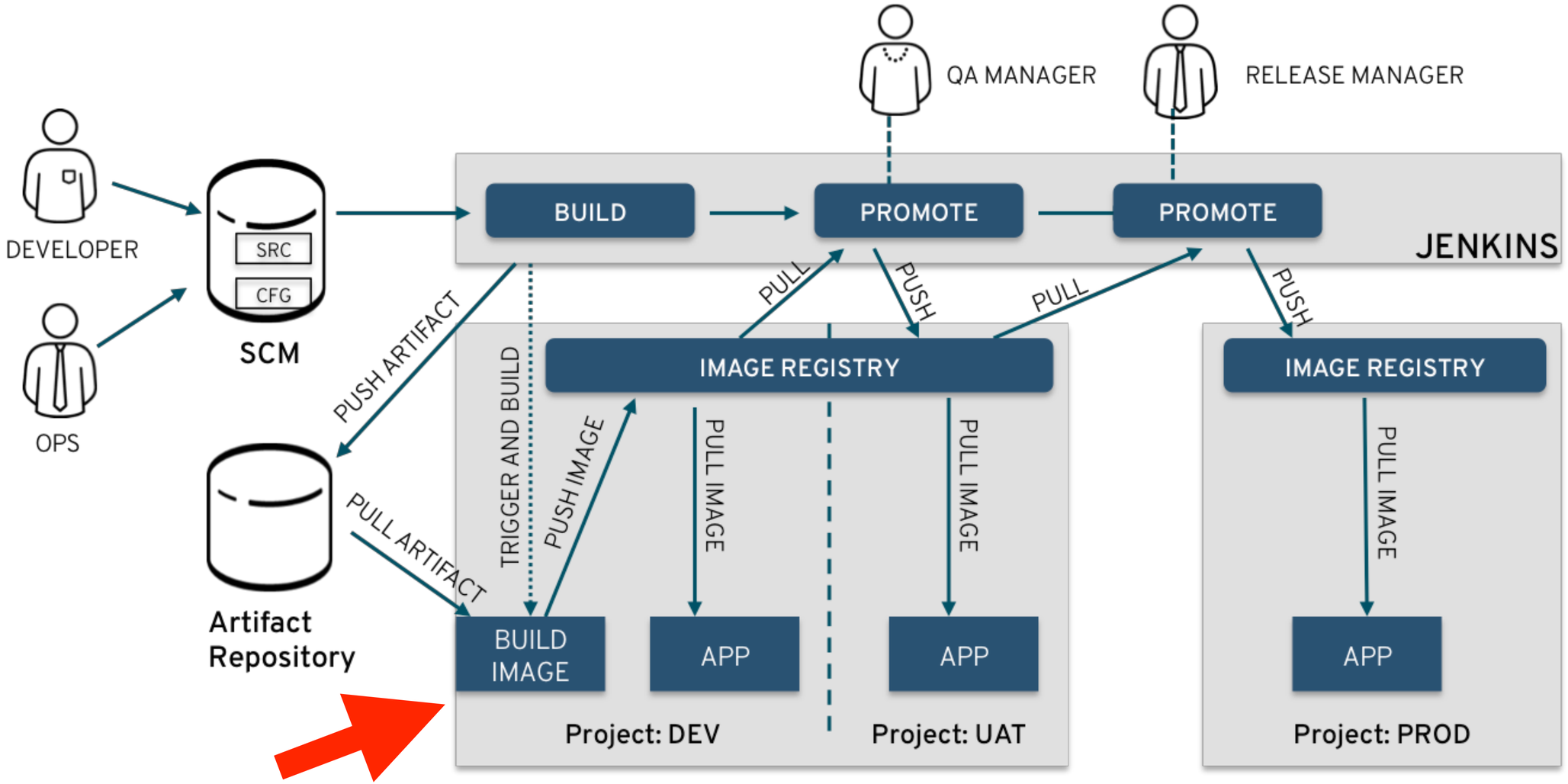
http://blog.kubernetes.io/2016/08/security-best-practices-kubernetes-deployment.html

Best Practices

- Don’t run as root
- If you must, limit Linux Capabilities
- Limit SSH Access
- Use namespaces
- Define resource quotas
- Enable logging
- Apply Security Errata
- Apply Security Context and seccomp filters
- Run production unprivileged containers as read-only

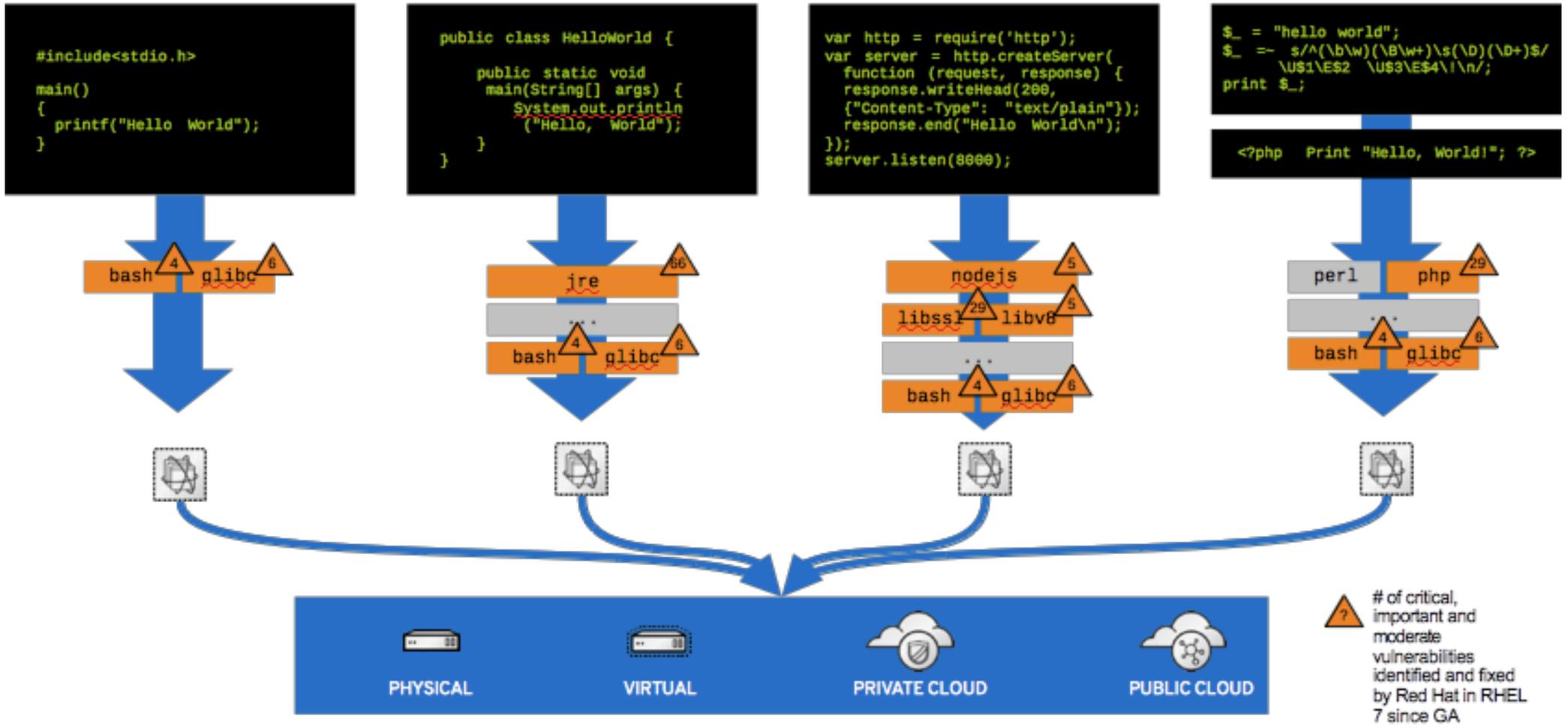
CONTINUOUS INTEGRATION WITH CONTAINERS

CONTINUOUS INTEGRATION + BUILDS



redhat.

WHAT'S INSIDE MATTERS…



redhat.

CONTINUOUS INTEGRATION WITH SECURITY SCAN

redhat.

AUTOMATED SECURITY SCANNING with OpenSCAP

Scan physical servers, virtual machines, docker images and containers for Security Policy Compliance (CCEs) and known Security Vulnerabilities (CVEs)

Content

SCAP Security Guide for RHEL

CCE-27002-5
Set Password Minimum Length

CVE DATABASE
CVE-2015-5477
Impact: Important
Public: 2015-07-28

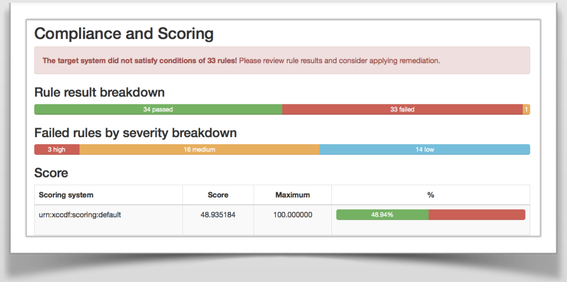
Scan

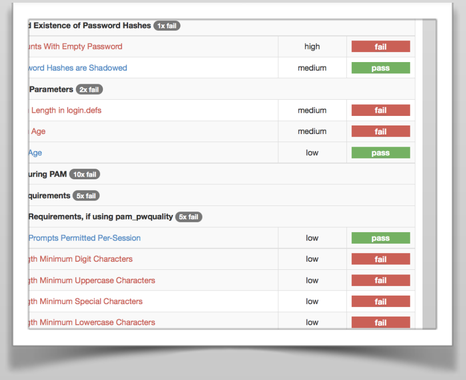
OpenSCAP

FOREMAN

docker

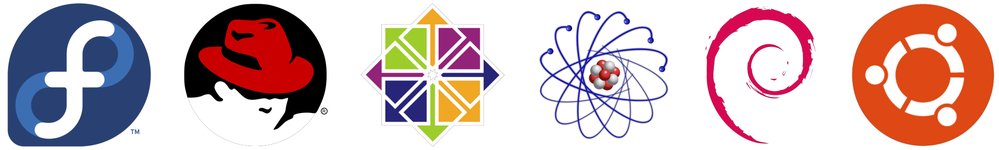
Reports





redhat.

Security Policies in SCAP Security Guide (partial)



Standard Docker Host Security Profile
Java Runtime Environment (JRE)
Upstream Firefox STIG
RHEL OSP STIG
Red Hat Corporate Profile for Certified Cloud Providers (RH CCP)
STIG for Red Hat Enterprise Linux 6, 7 Server
STIG for Red Hat Virtualization Hypervisor
Common Profile for General-Purpose Debian Systems
Common Profile for General-Purpose Fedora Systems
Common Profile for General-Purpose Ubuntu Systems

Payment Card Industry – Data Security Standard (PCI-DSS) v3

U.S. Government Commercial Cloud Services (C2S)
CNSSI 1253 Low/Low/Low Control Baseline for Red Hat Enterprise Linux 7
Criminal Justice Information Services (CJIS) Security Policy
Unclassified Information in Non-federal Information Systems and Organizations (NIST 800-171)
U.S. Government Configuration Baseline (NIAP OSPP v4.0, USGCB, STIG)

OpenSCAP BASE

SECURITY POLICY REPORT

Verify Proper Storage and Existence of Password Hashes 1x fail		
Prevent Log In to Accounts With Empty Password	high	fail
Verify All Account Password Hashes are Shadowed	medium	pass
Set Password Expiration Parameters 2x fail		
Set Password Minimum Length in login.defs	medium	fail
Set Password Minimum Age	medium	fail
Set Password Warning Age	low	pass
Protect Accounts by Configuring PAM 10x fail		
Set Password Quality Requirements 5x fail		
Set Password Quality Requirements, if using pam_pwquality 5x fail		
Set Password Retry Prompts Permitted Per-Session	low	pass
Set Password Strength Minimum Digit Characters	low	fail
Set Password Strength Minimum Uppercase Characters	low	fail
Set Password Strength Minimum Special Characters	low	fail
Set Password Strength Minimum Lowercase Characters	low	fail

redhat.

SECURITY POLICY REMEDIATION

Set Password Strength Minimum Digit Characters

Rule ID	accounts_password_pam_dcredit
Result	fail
Time	2015-07-31T14:57:17
Severity	low
Identifiers and References	identifiers: CCE-27163-5 references: IA-5(b), IA-5(c), 194, 194, 71,

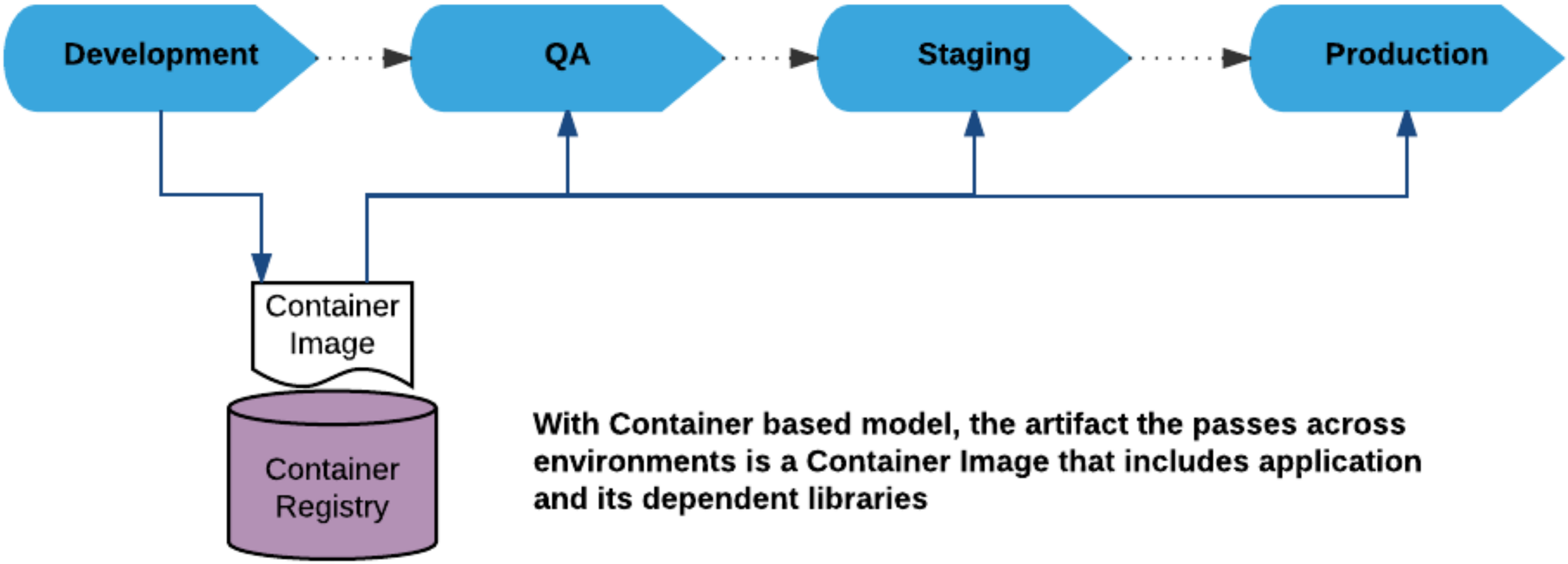
The pam_pwquality module's dcredit parameter controls requirements for usage of digits in a password. When set to a negative number, any password will be required to contain that many digits. When set to a positive number, pam_pwquality will grant +1 additional length credit for each digit. Add dcredit=-1 after pam_pwquality.so to require use of a digit in passwords.

Remediation script:

```
var_password_pam_dcredit="-1"
if grep -q "dcredit=" /etc/pam.d/system-auth; then
        sed -i --follow-symlink "s/\(dcredit *= *\).*/\1$var_password_pam_dcredit/" /etc/pam.d/system-auth
else
        sed -i --follow-symlink "/pam_pwquality.so/ s/$/ dcredit=$var_password_pam_dcredit/" /etc/pam.d/system-auth
fi
```

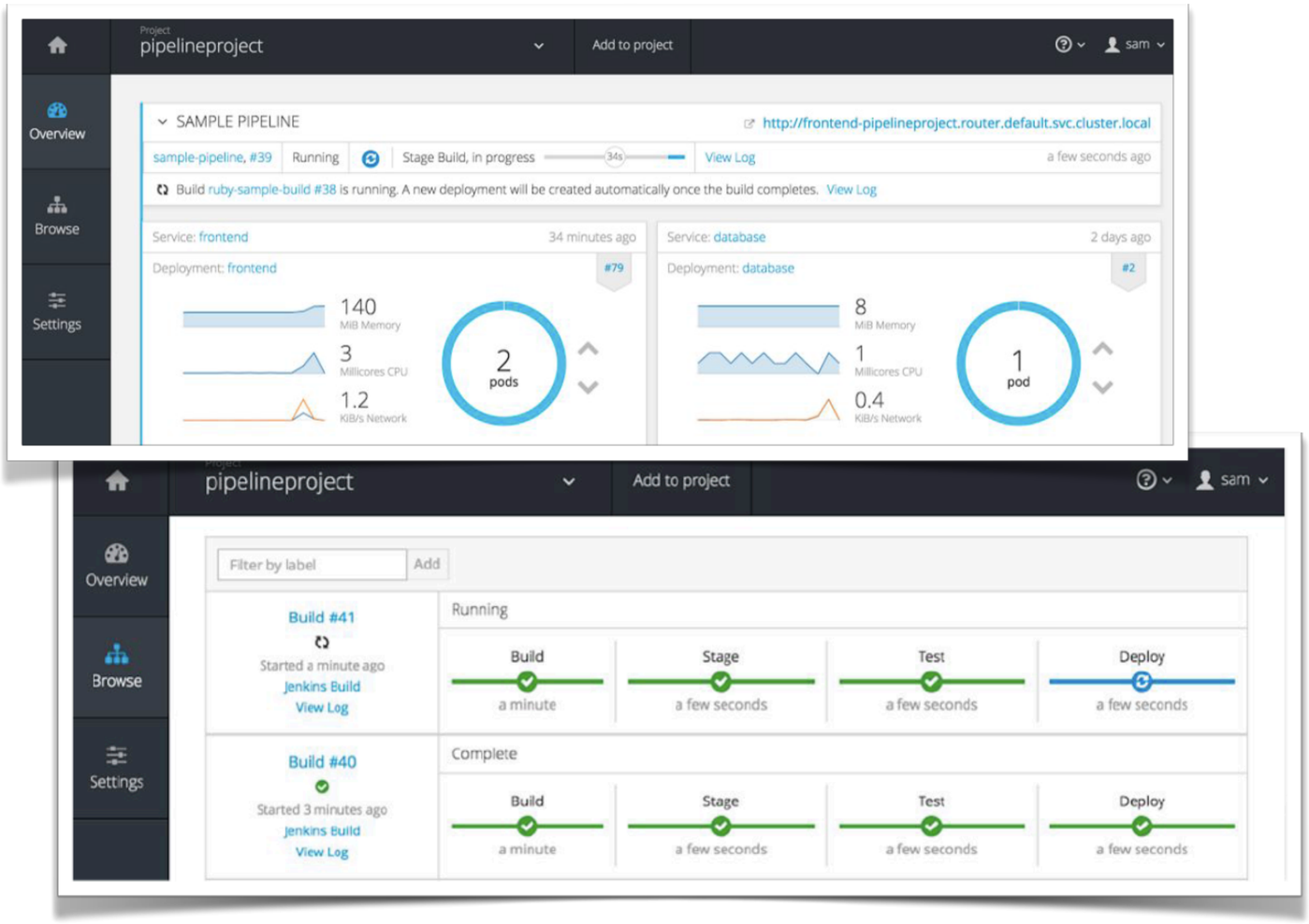
CONTINUOUS DELIVERY WITH CONTAINERS

CONTINUOUS DELIVERY WITH CONTAINERS



With Container based model, the artifact the passes across environments is a Container Image that includes application and its dependent libraries

CONTINUOUS DELIVERY DEPLOYMENT STRATEGIES



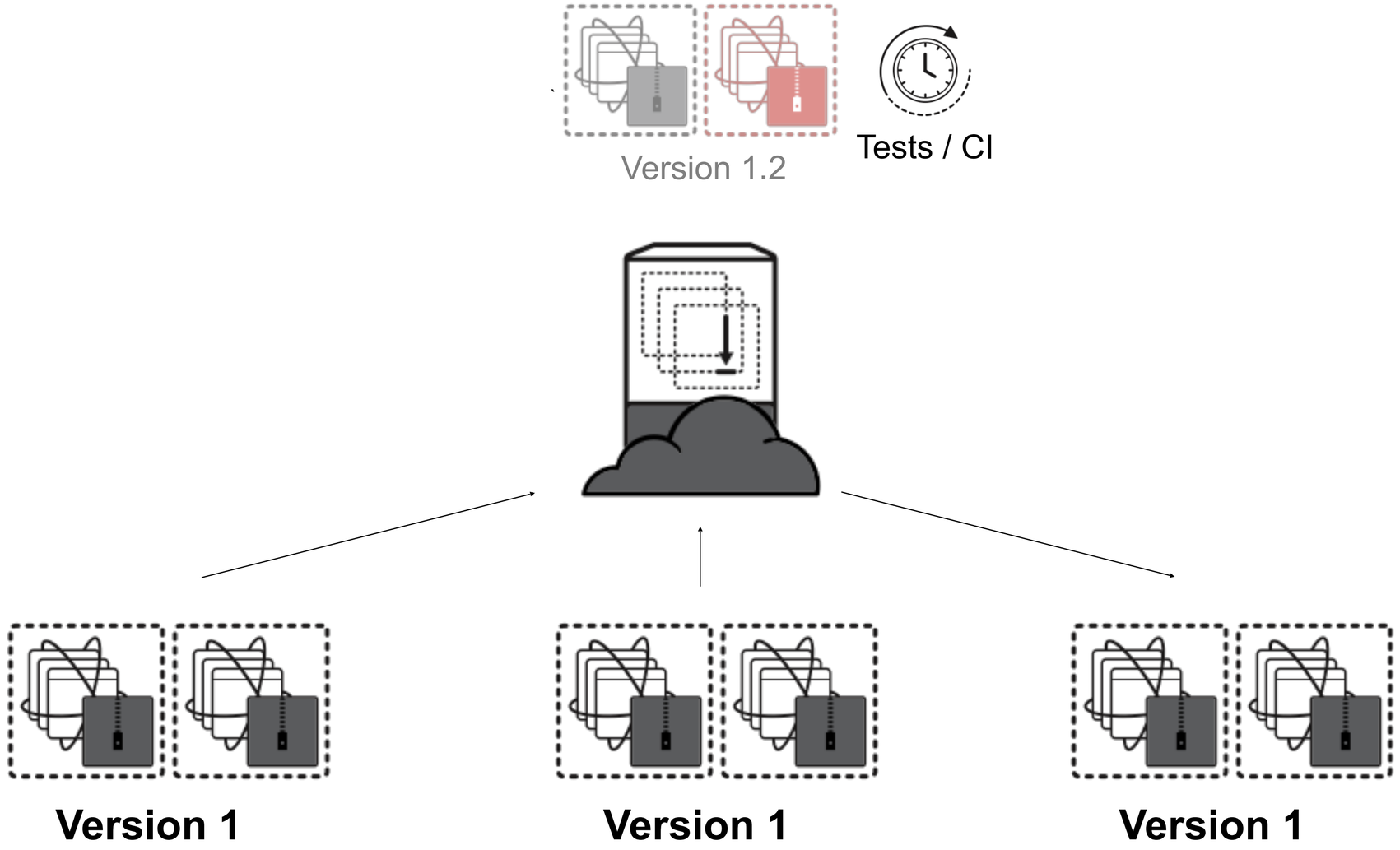
DEPLOYMENT STRATEGIES

- Recreate
- Rolling updates
- Blue / Green deployment
- Canary with A/B testing

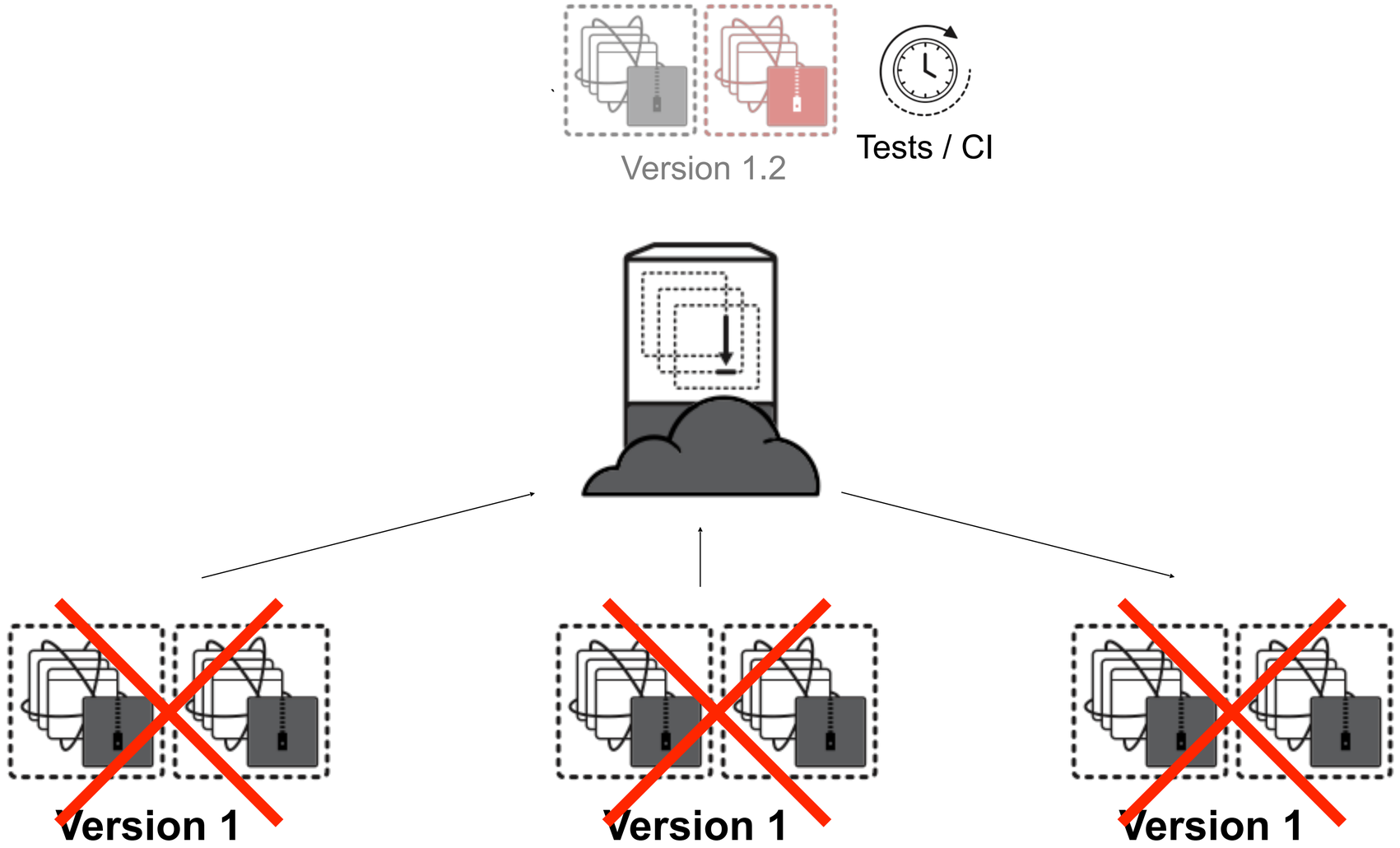
redhat.

Recreate

RECREATE WITH DOWNTIME



RECREATE WITH DOWNTIME

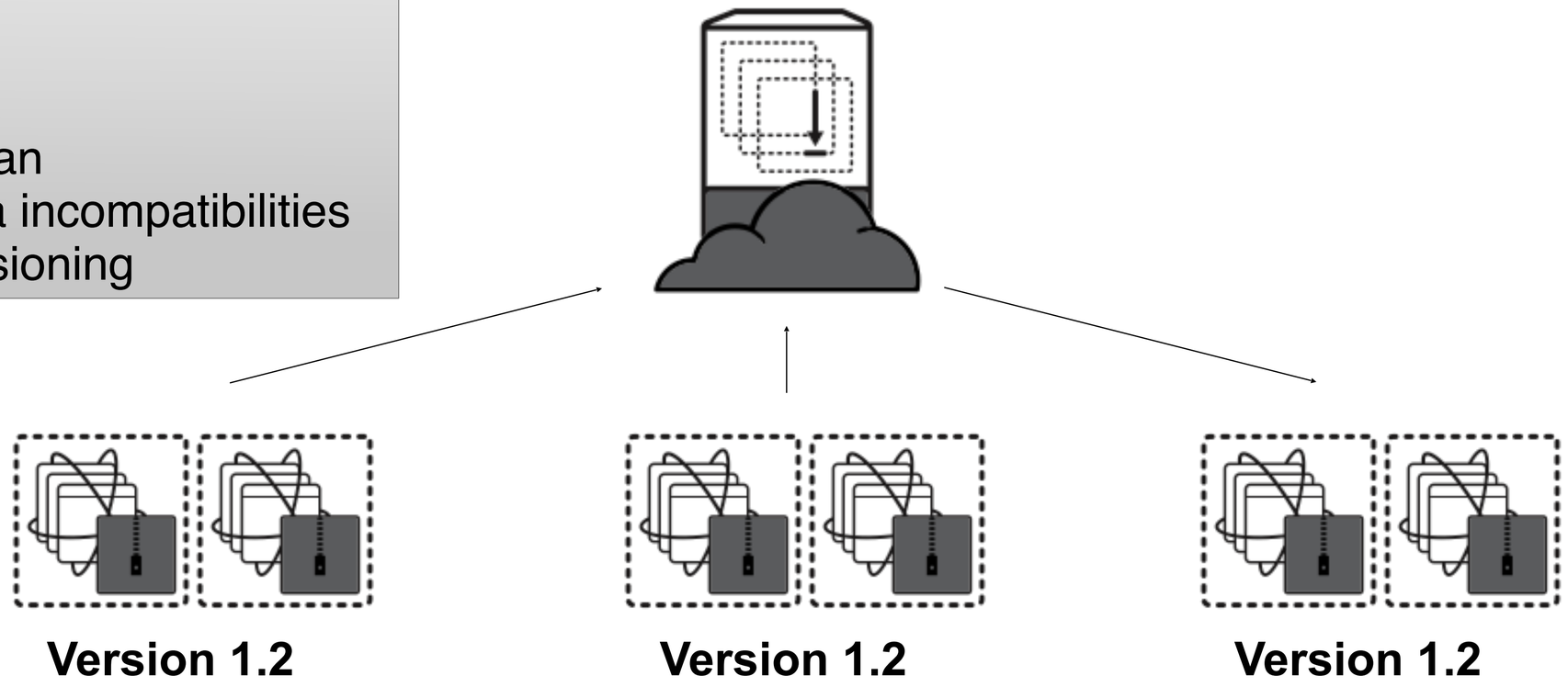


RECREATE WITH DOWNTIME

Use Case
- Non-mission critical services

Cons
- Downtime

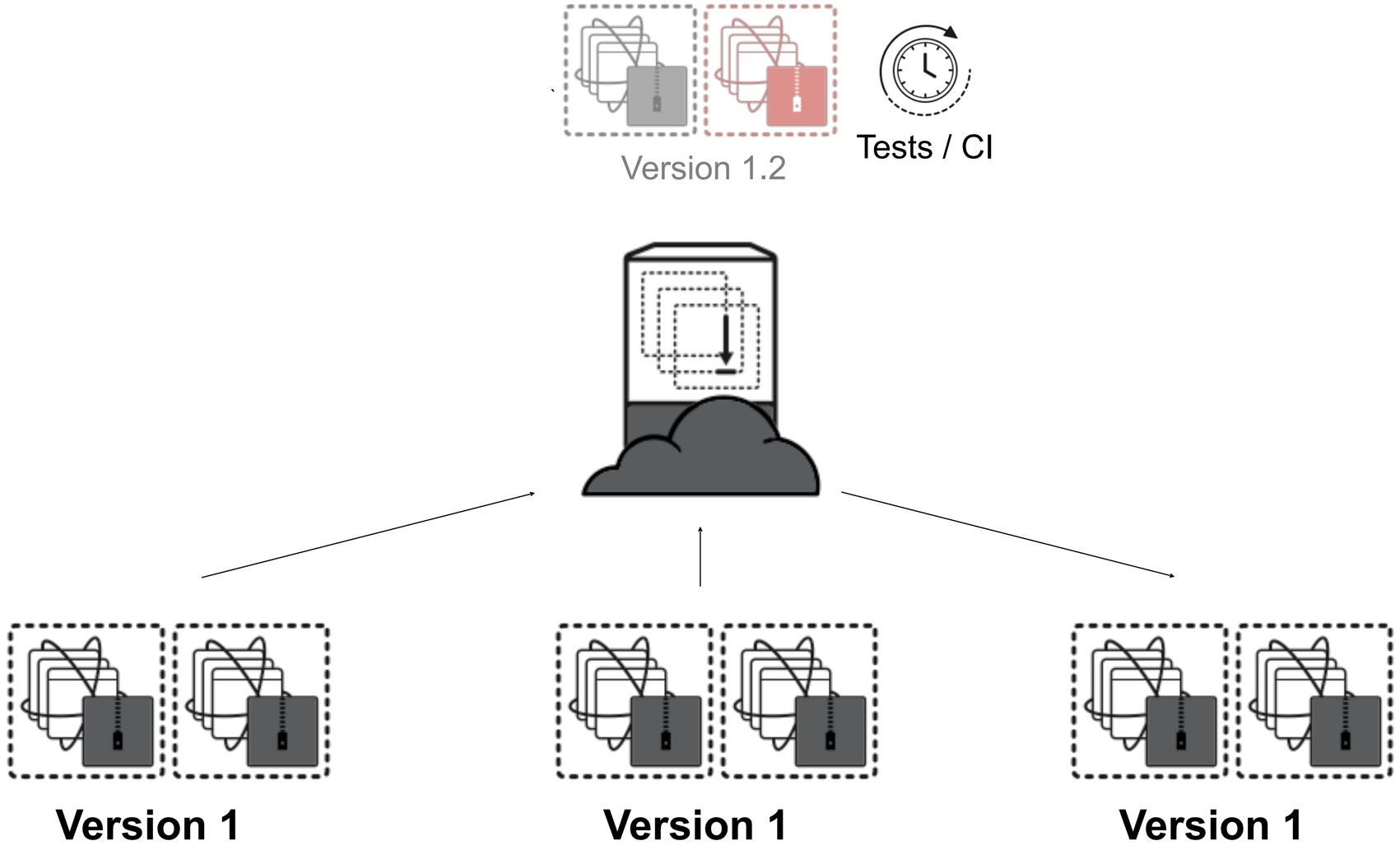
Pros
- Simple, clean
- No Schema incompatibilities
- No API versioning



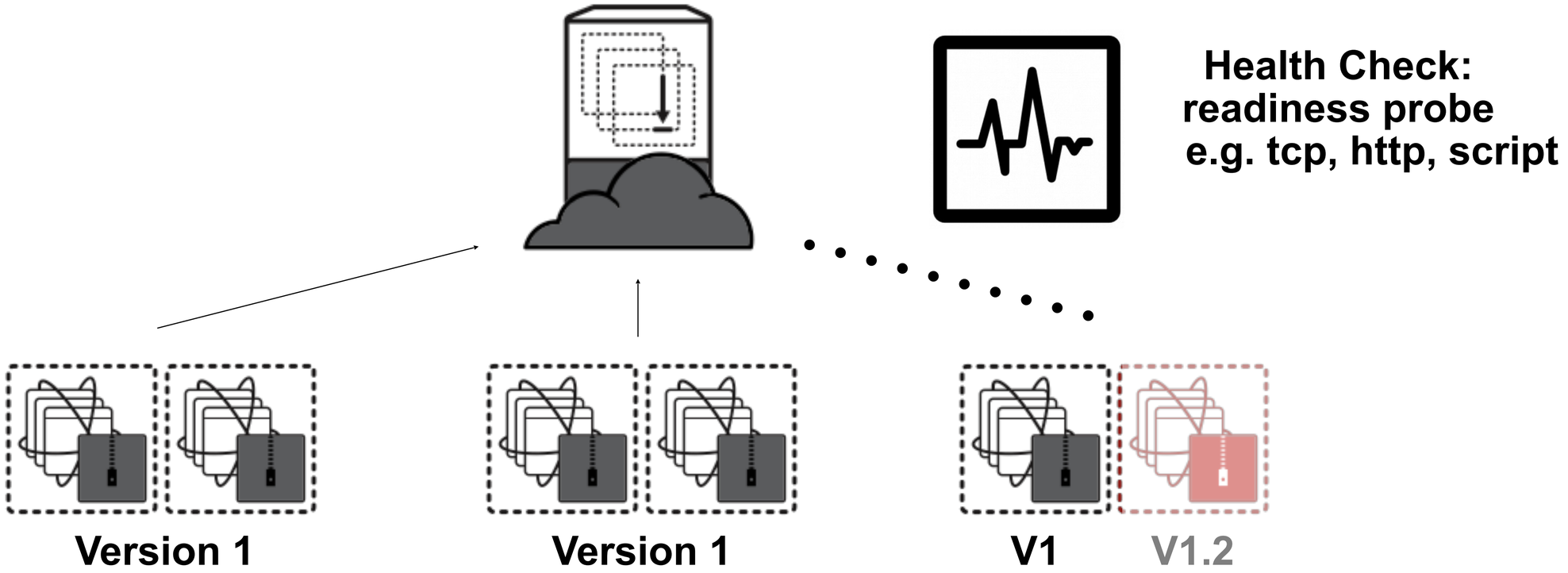
redhat.

Rolling Updates

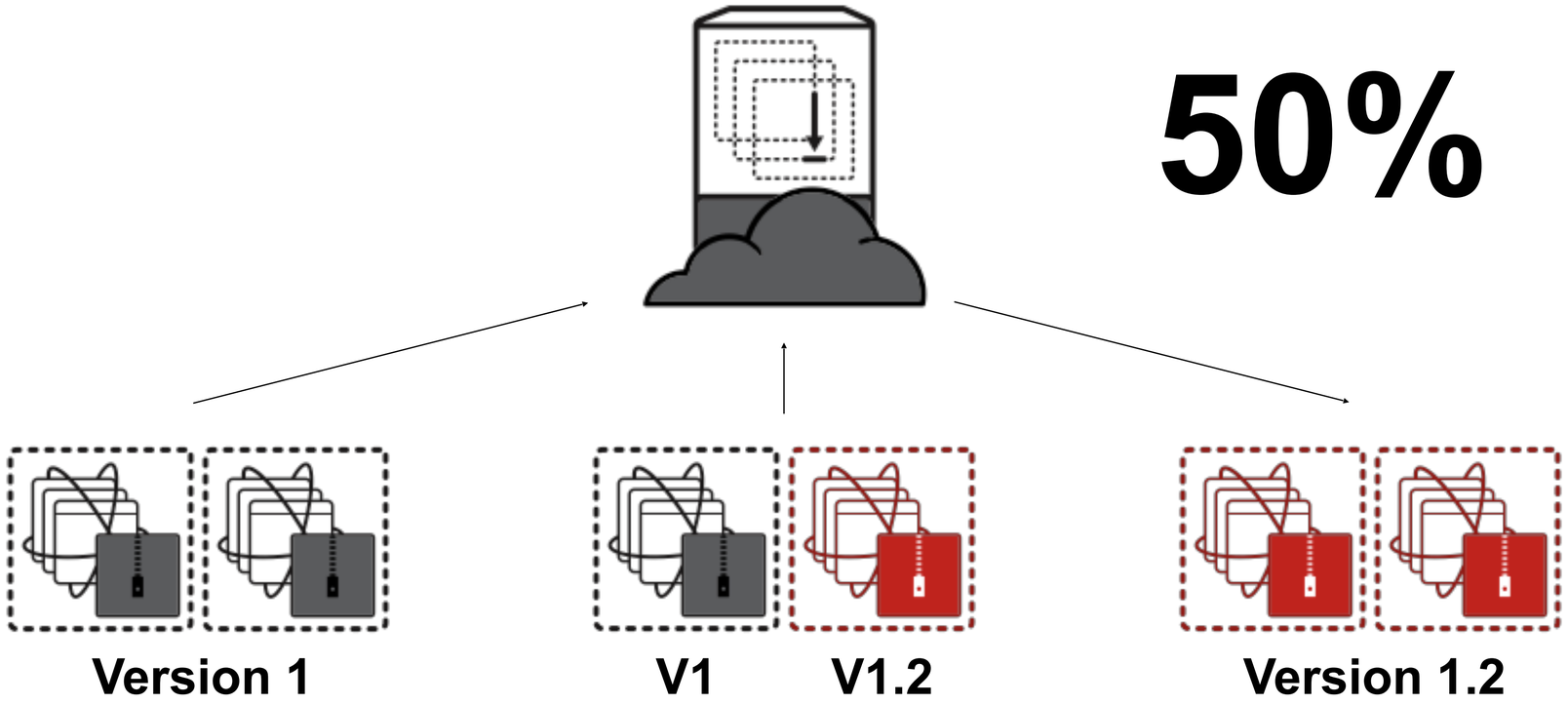
ROLLING UPDATES with ZERO DOWNTIME



Deploy new version and wait until it’s ready…



Each container/pod is updated one by one

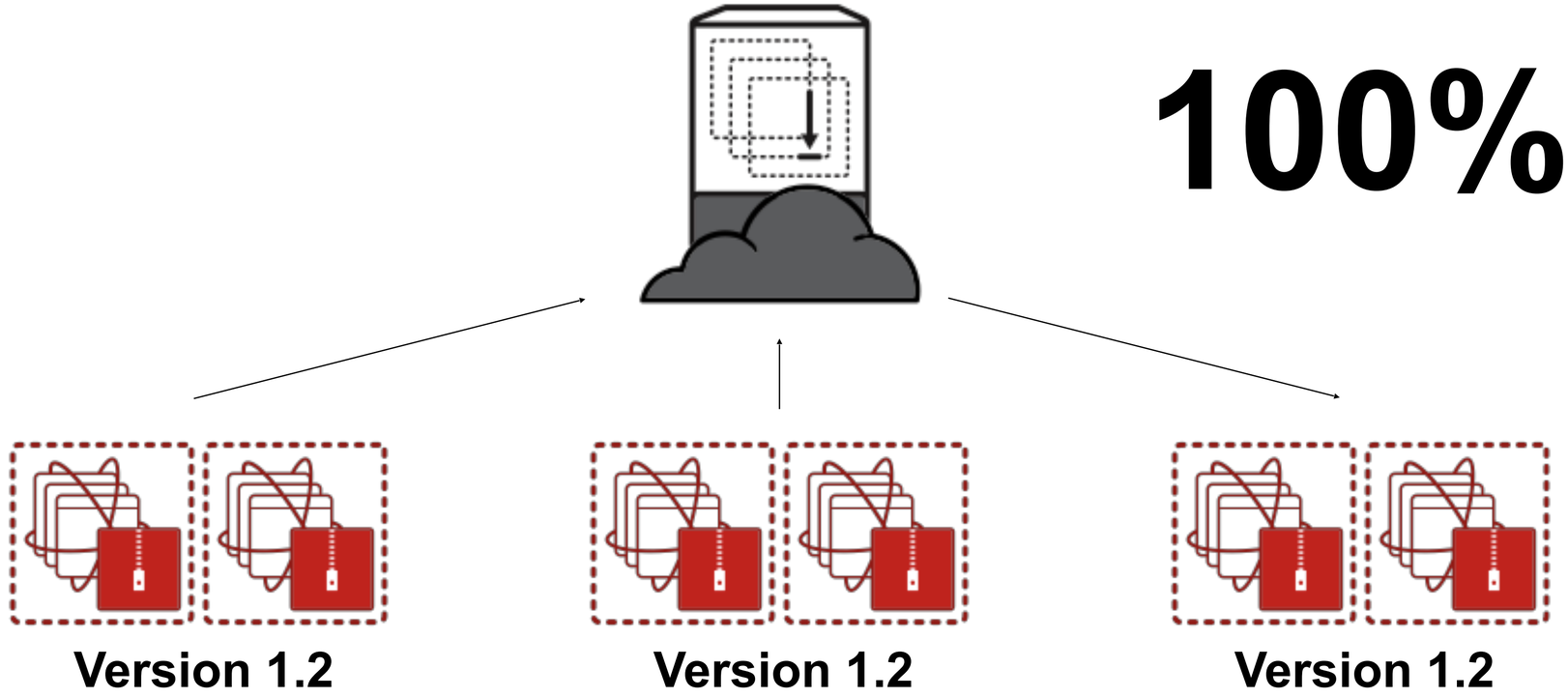


Each container/pod is updated one by one

Use Case
- Horizontally scaled
- Backward compatible API/data
- Microservices

Cons
- Require backward compatible APIs/data
- Resource overhead

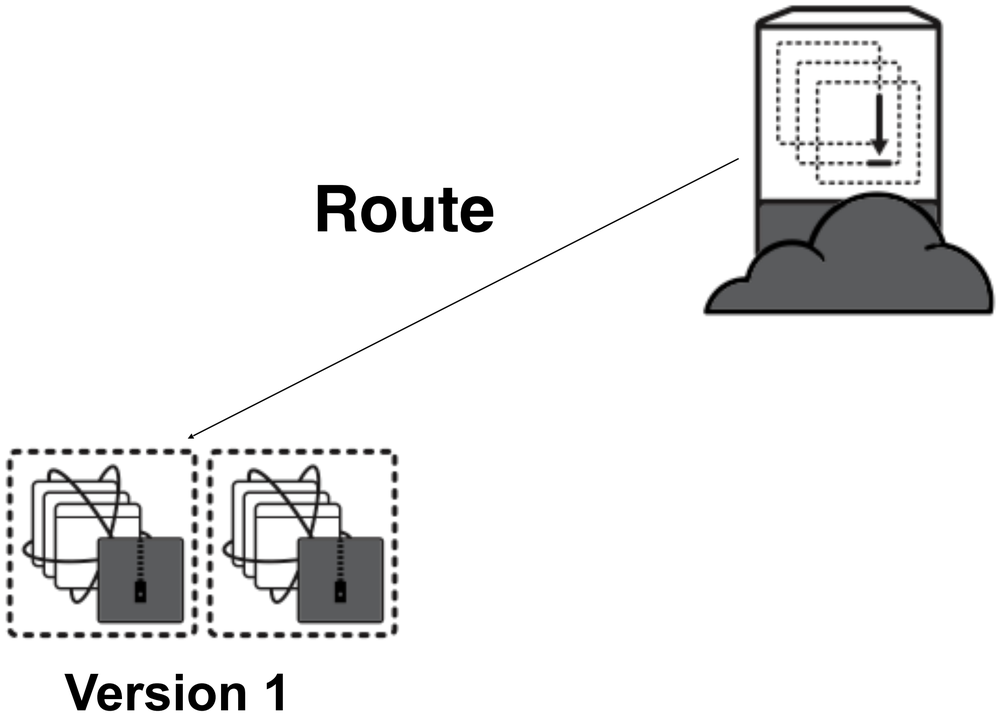
Pros
- Zero downtime
- Reduced risk, gradual rollout w/health checks
- Ready for rollback



redhat.

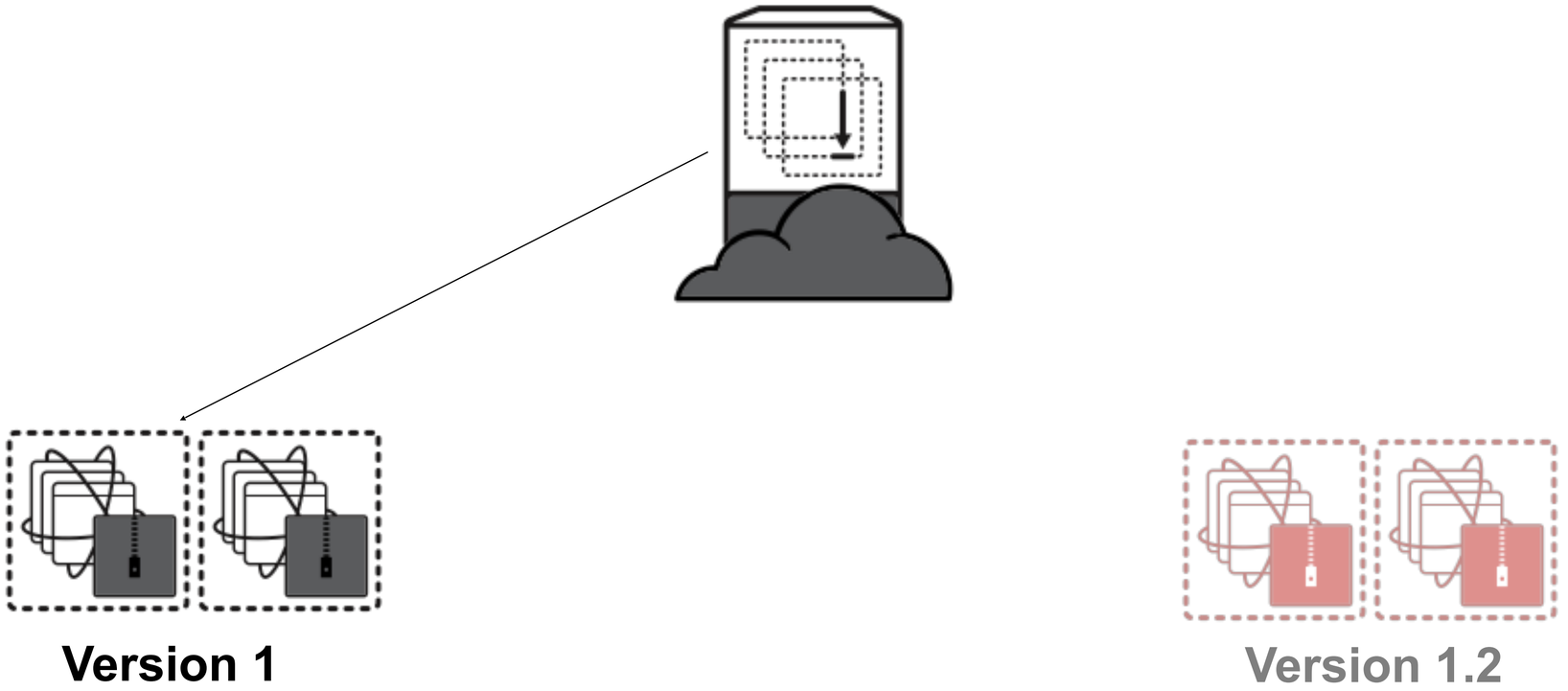
Blue / Green Deployment

BLUE / GREEN DEPLOYMENT

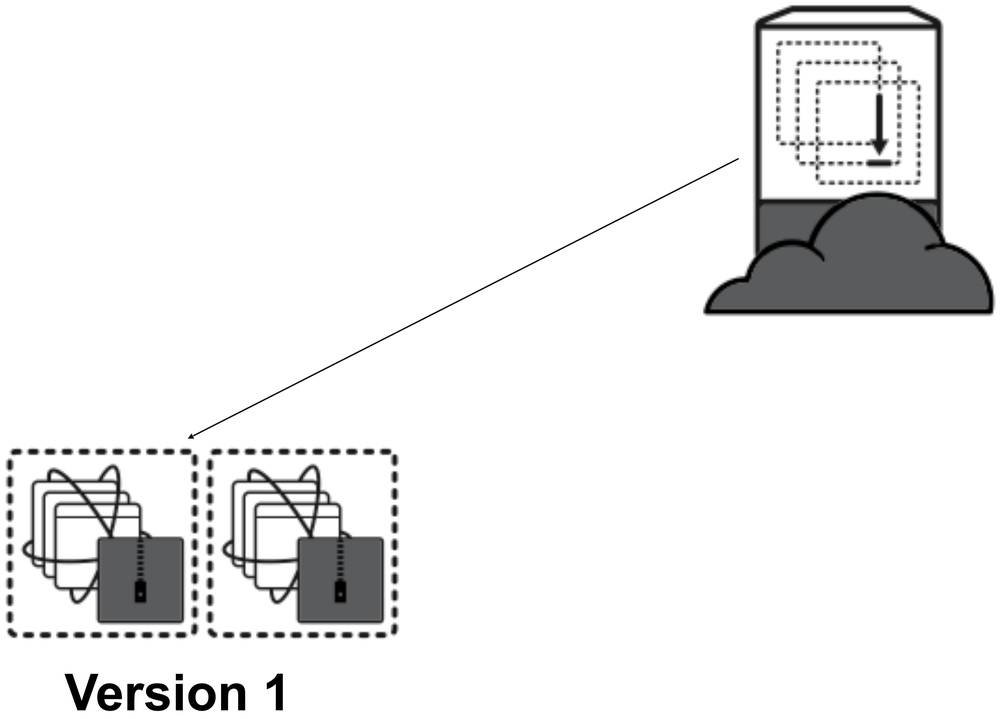


BLUE

BLUE / GREEN DEPLOYMENT



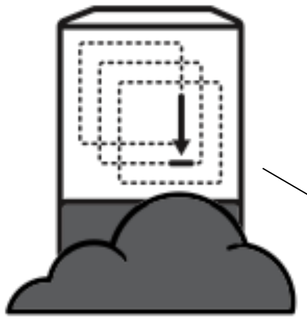
BLUE / GREEN DEPLOYMENT



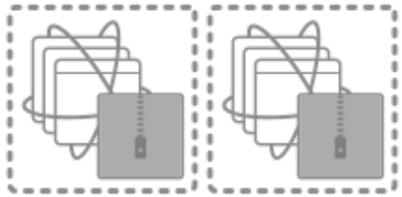
BLUE

GREEN

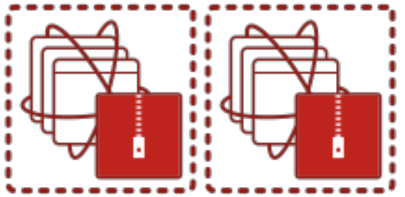
BLUE / GREEN DEPLOYMENT



Route



Version 1

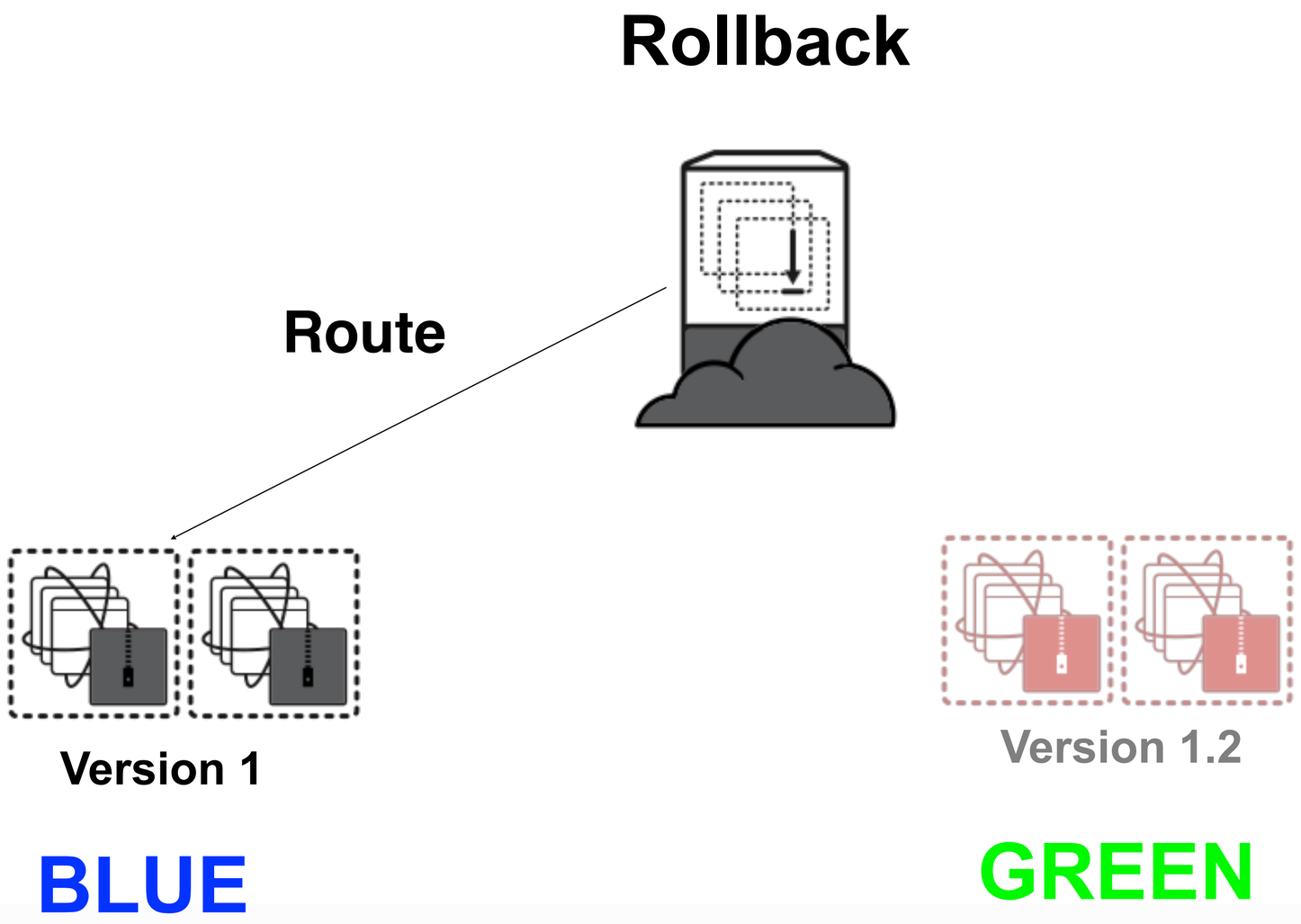


Version 1.2

BLUE

GREEN

BLUE / GREEN DEPLOYMENT



Use Case
- Self-contained micro services (data)

Cons
- Resource overhead
- Data synchronization

Pros
- Low risk, never change production
- No downtime
- Production like testing
- Rollback

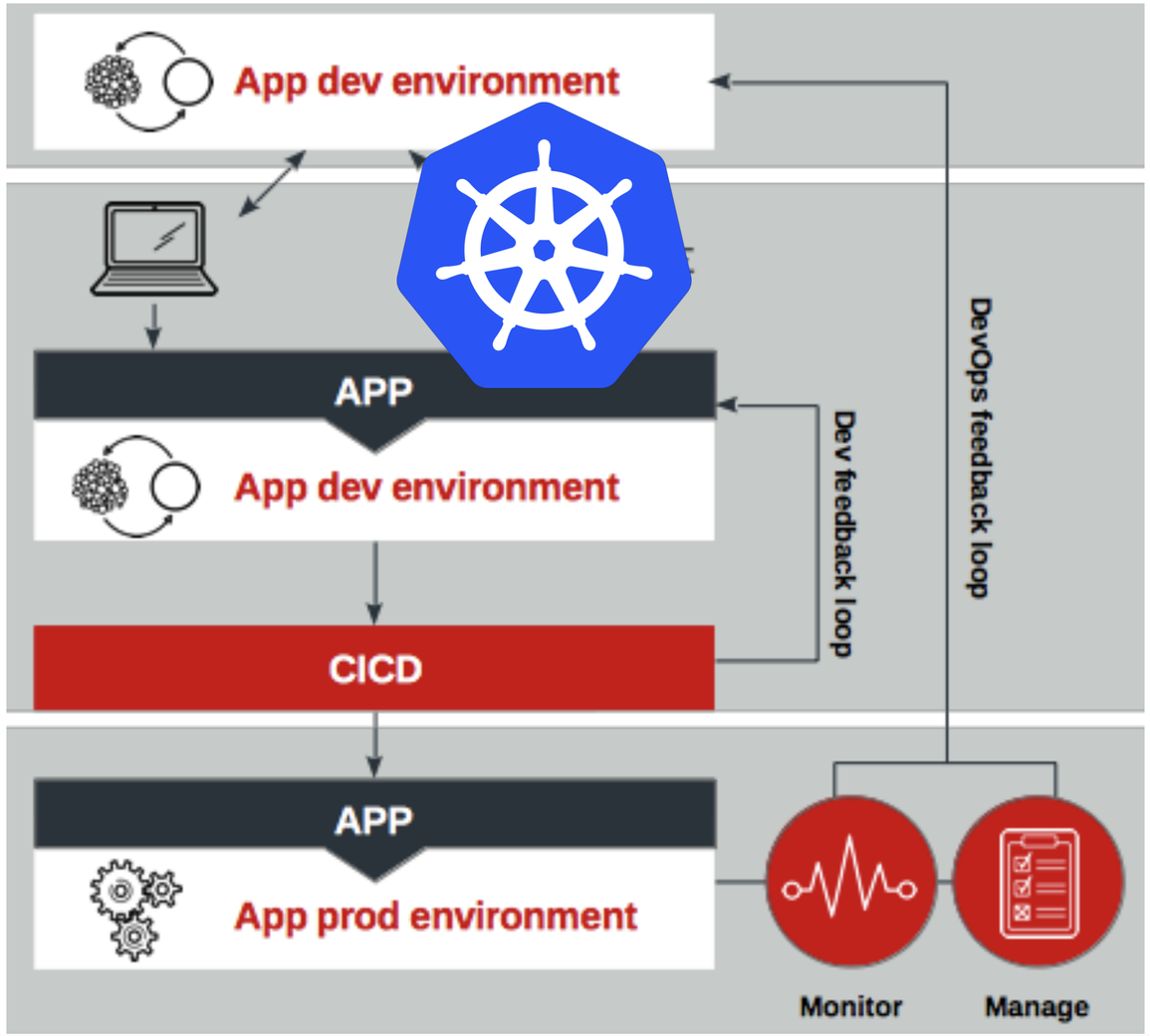
RAPID INNOVATION & EXPERIMENTATION

MICROSERVICES
RAPID INNNOVATION & EXPERIMENTATION

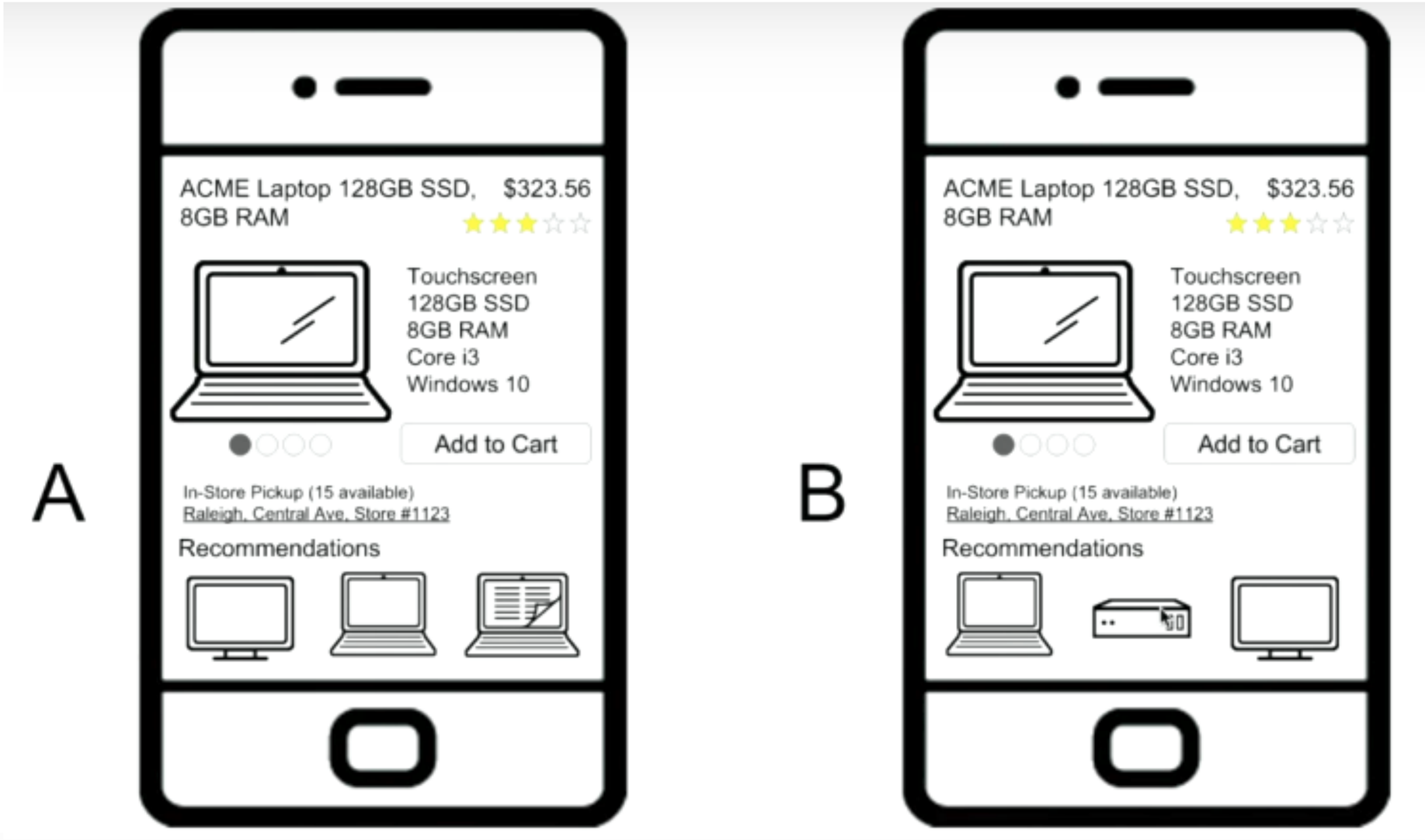
”only about 1/3 of ideas improve the metrics they were designed to improve.”

Ronny Kohavi, Microsoft (Amazon)

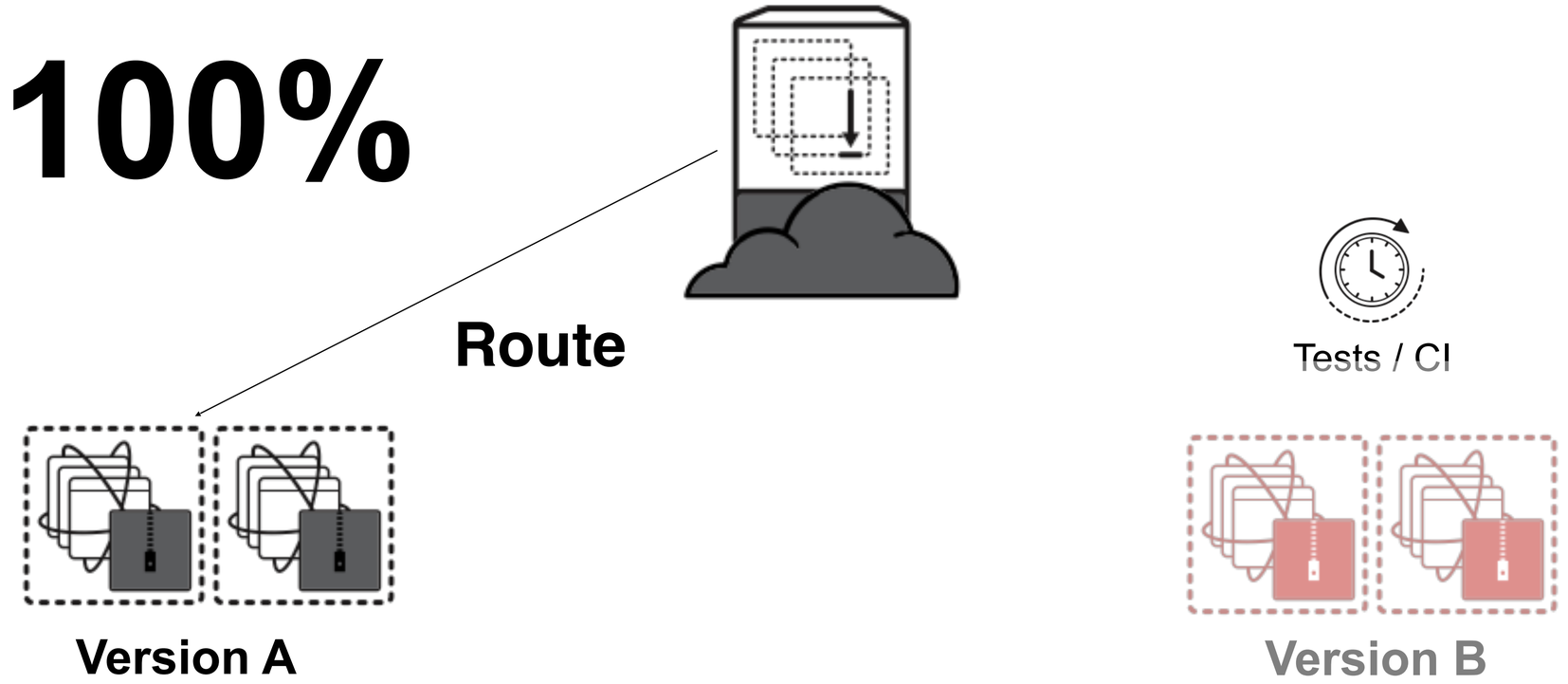
CONTINUOUS FEEDBACK LOOP



A/B TESTING USING CANARY DEPLOYMENTS



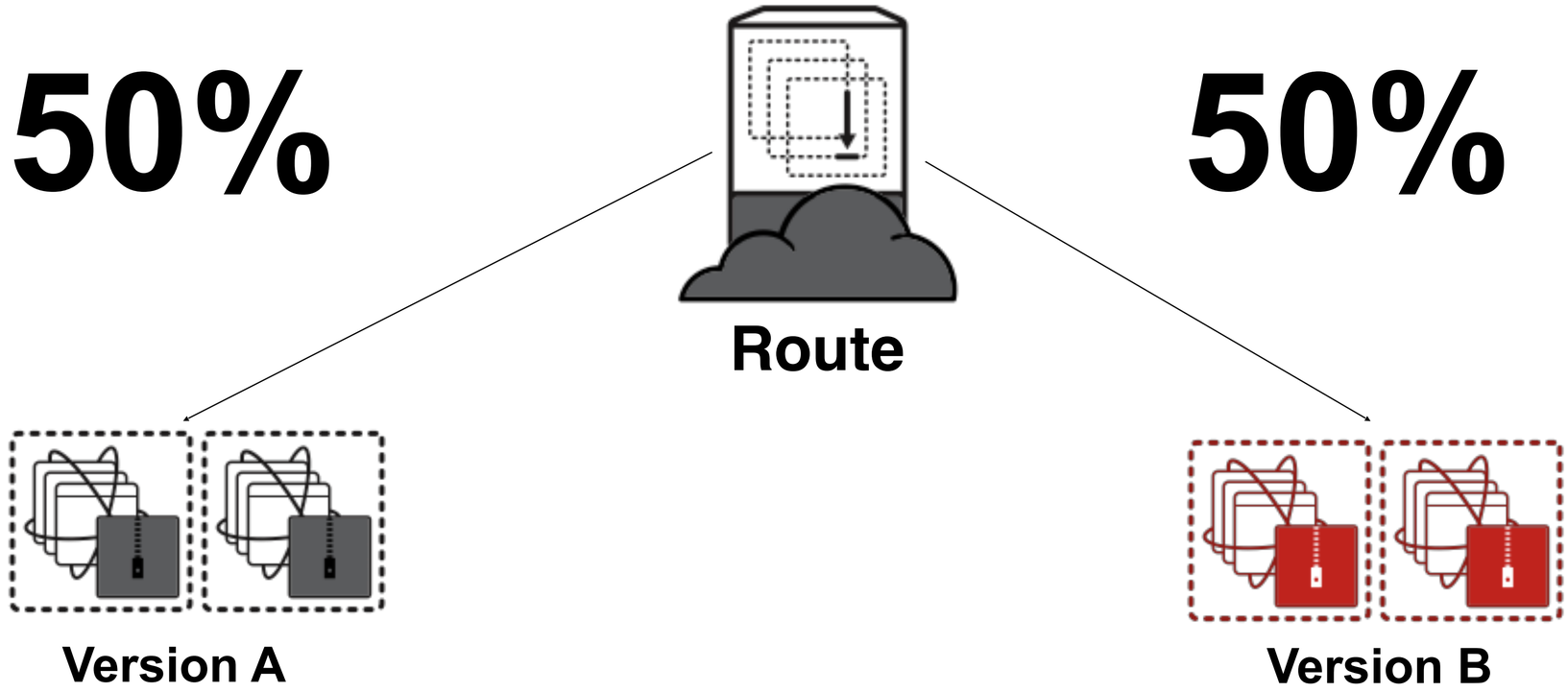
CANARY DEPLOYMENTS



25% Conversion Rate

?! Conversion Rate

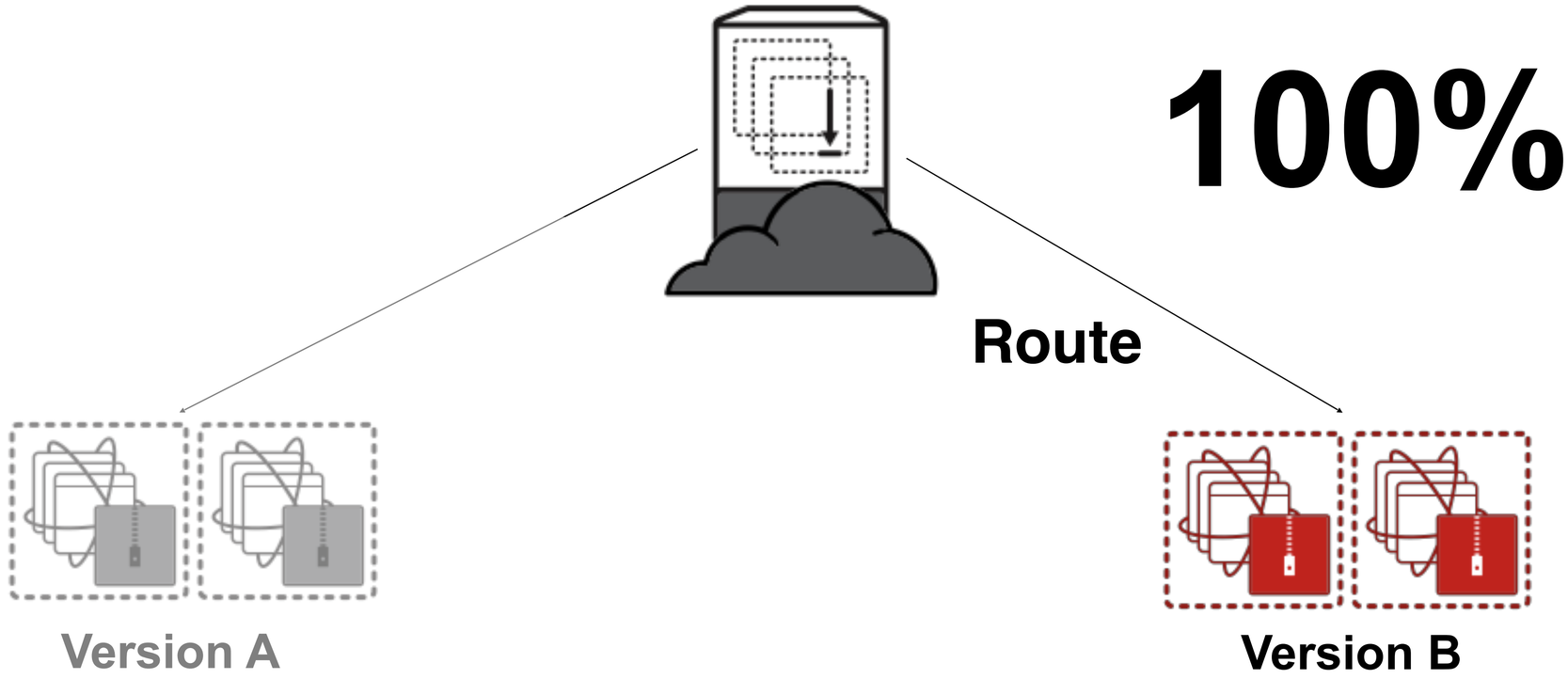
CANARY DEPLOYMENTS



25% Conversion Rate

30% Conversion Rate

CANARY DEPLOYMENTS



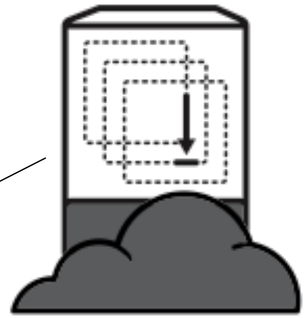
25% Conversion Rate

30% Conversion Rate

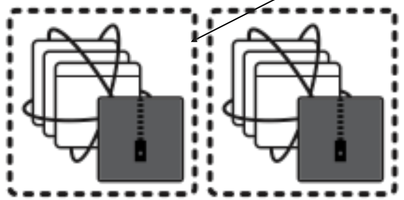
CANARY DEPLOYMENTS

Rollback

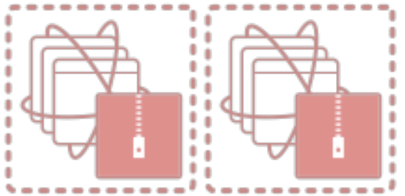
100%



Route



Version A

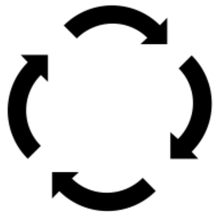


Version B

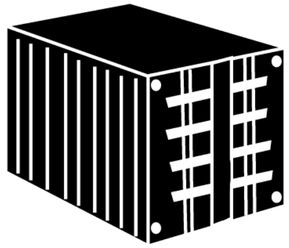
25% Conversion Rate

20% Conversion Rate

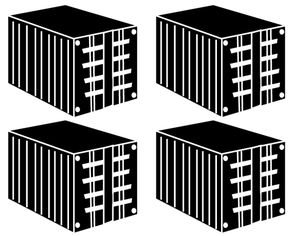
SECURING YOUR CONTAINER ENVIRONMENT



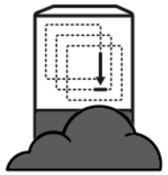
Builds



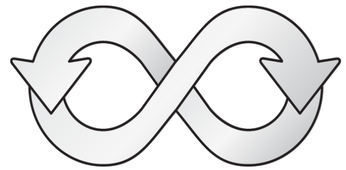
Images



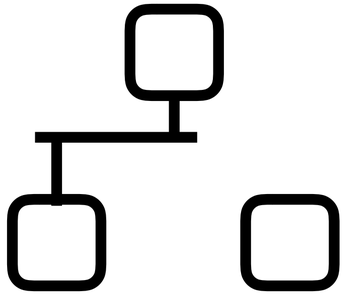
Registry



Container host



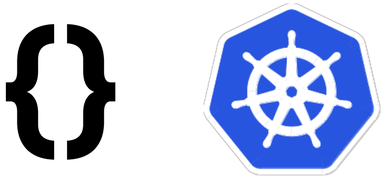
CI/CD



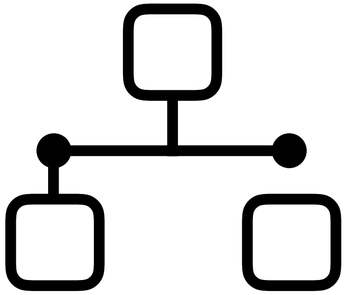
Network isolation

Monitoring & Logging

Storage



API & Platform access

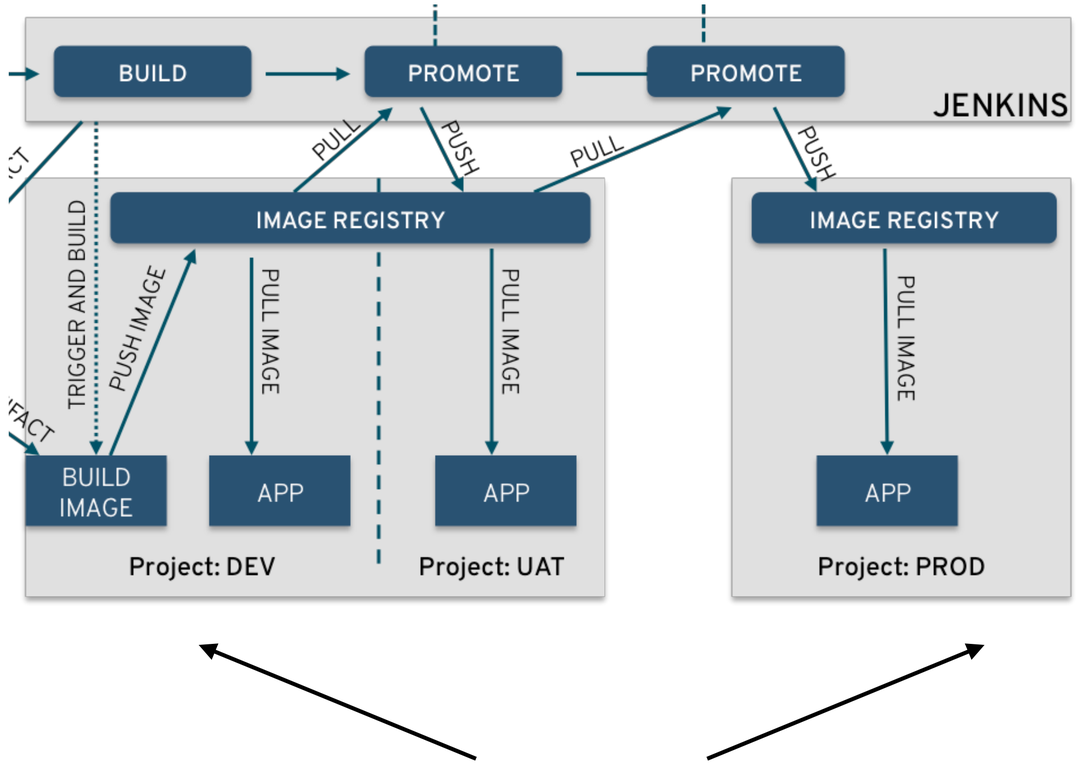


Federated clusters

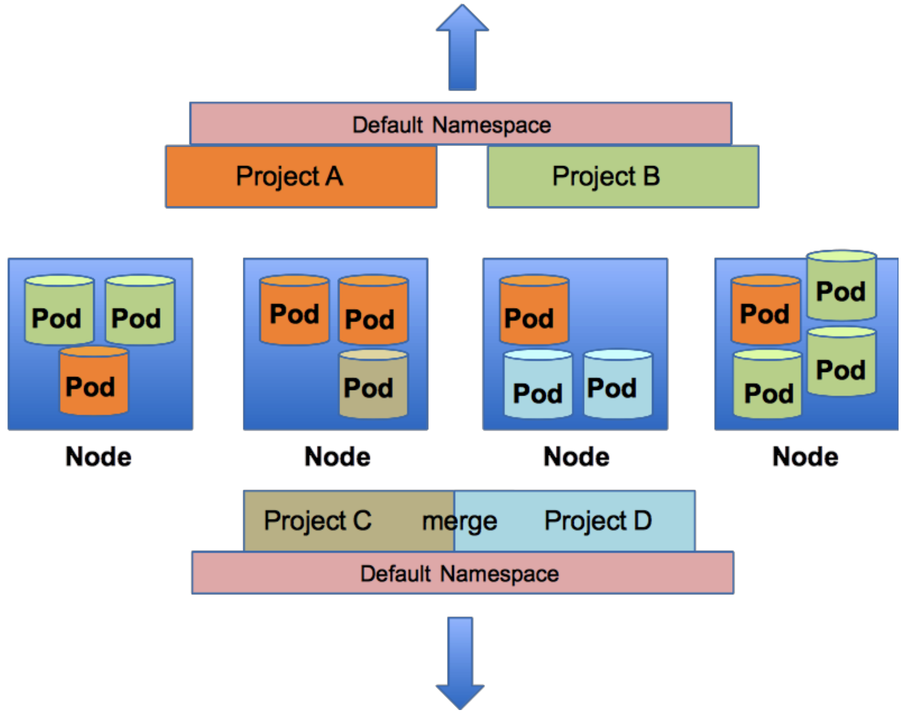
NETWORK SECURITY

NETWORK ISOLATION

Multi-Environment



Multi-Tenant

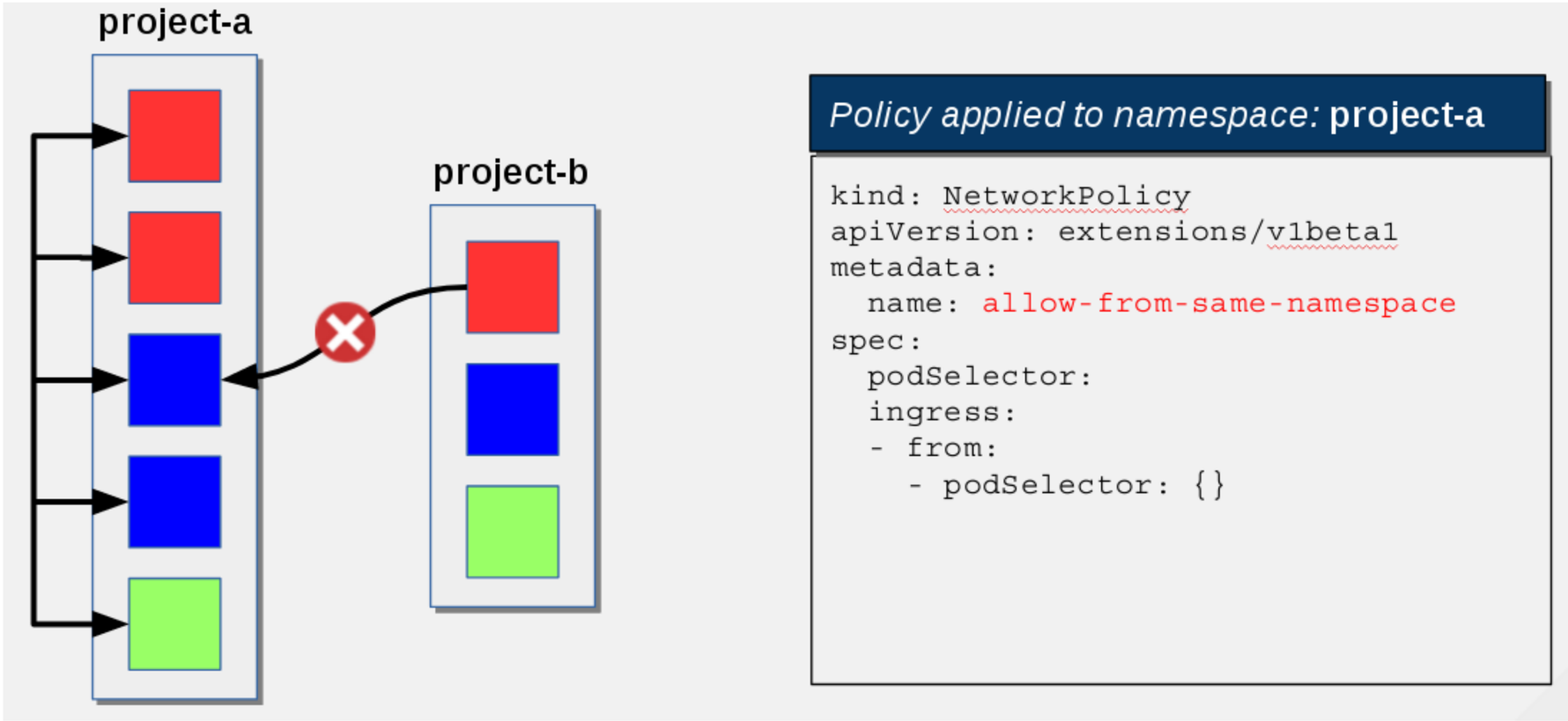


Network Namespace provides resource isolation

redhat.

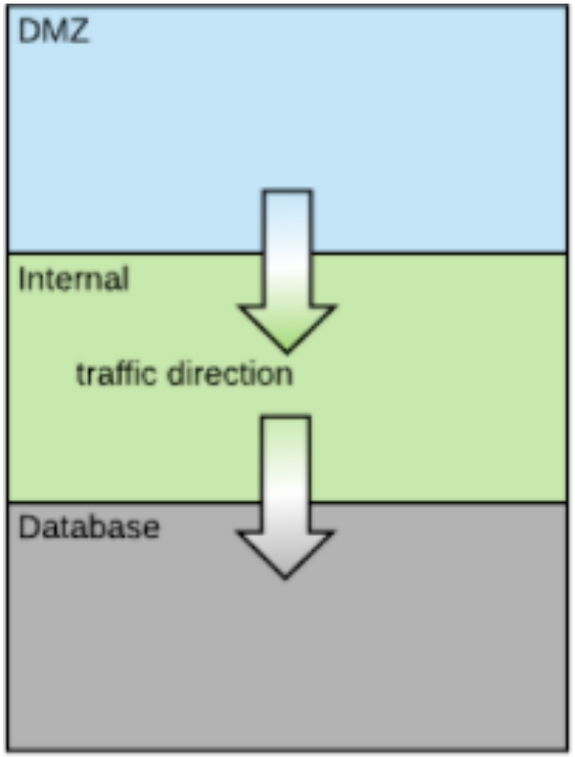
NETWORK POLICY

example:
all pods in namespace ‘project-a’ allow traffic
from any other pods in the same namespace.”



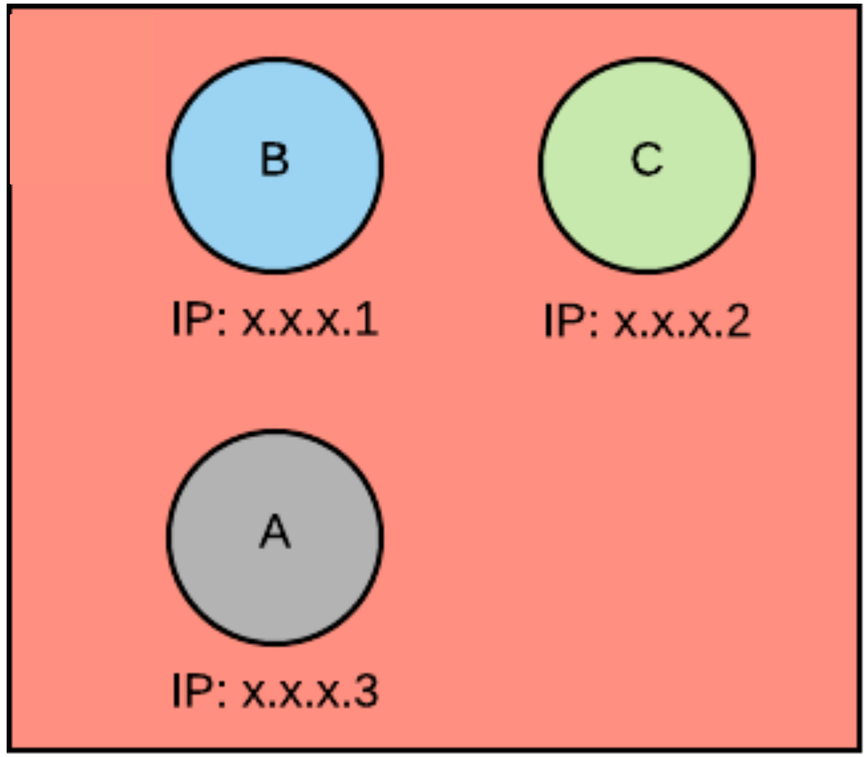
NETWORK SECURITY

Traditional Physical Network Model



- Each layer represents a Zone with increased trust - DMZ > App > DB, interzone flow generally one direction
- Intrazone traffic generally unrestricted

Kubernetes Logical Network Model

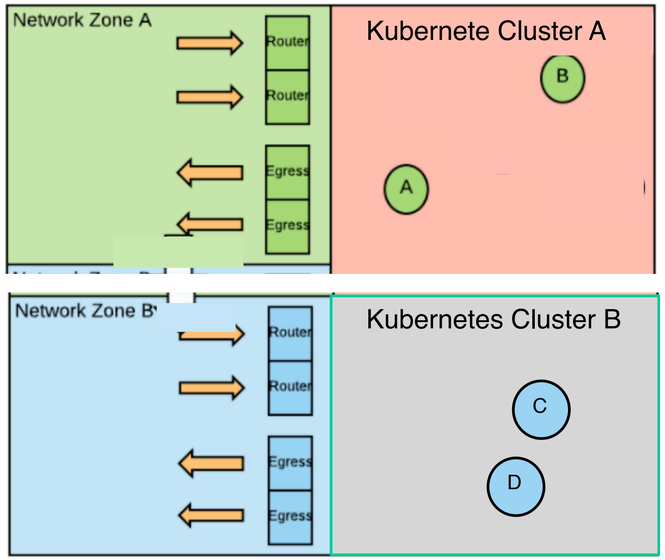


- Kubernetes uses a flat SDN model
  - All pods get IP from same CIDR
  - And live on same logical network
  - Assumes all nodes communicate

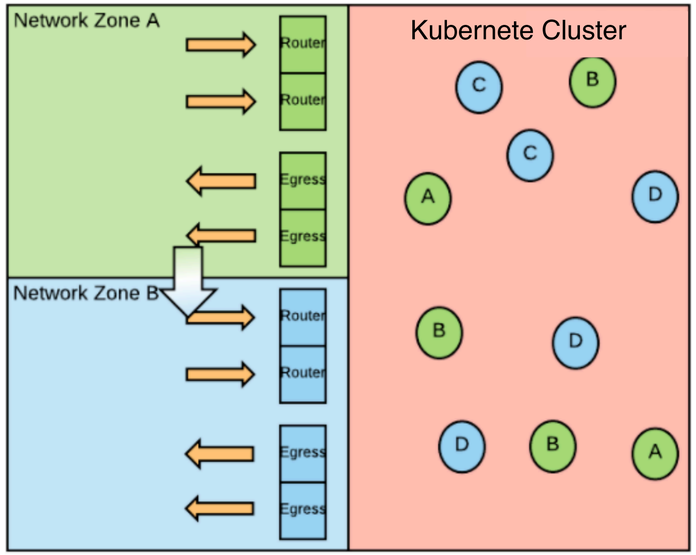
NETWORK SECURITY MODELS

Co-Existence Approaches

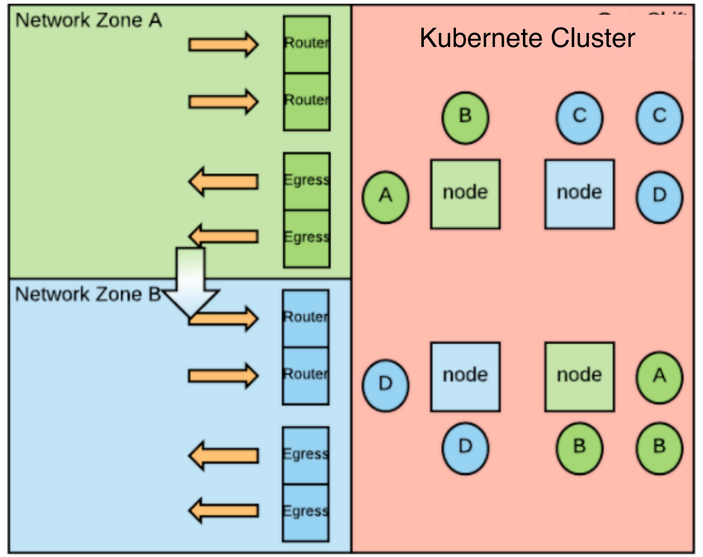
One Cluster Per Zone



One Cluster Multiple Zones



Physical Compute isolation based on Network Zones



https://blog.openshift.com/openshift-and-network-security-zones-coexistence-approaches/

redhat.

MONITORING & LOGGING

KUBERNETES MONITORING CONSIDERATIONS

Stack		Metrics	Tool
Application		Distributed applications - traditional app metrics - service discovery - distributed tracing	prometheus + grafana jaeger tracing istio
Kubernetes*		Cluster services, services, pods, deployments metrics	prometheus + grafana kubernetes-state-metrics probes
Container*		Container native metrics	kubelet:cAdvisor
Host		Traditional resource metrics - cpu, memory, network, storage	node-exporter

redhat.

LOGGING

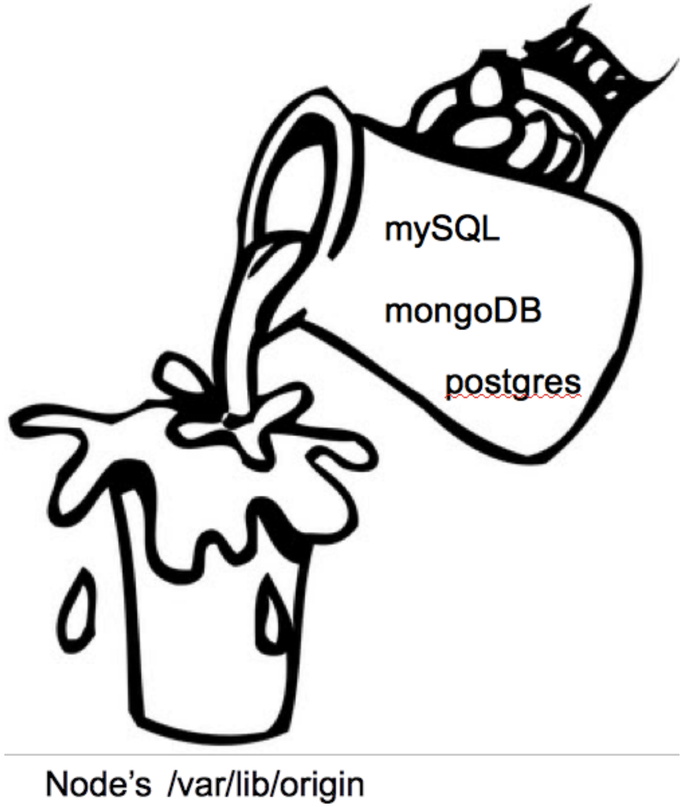
Aggregate platform and application log access via Kibana + Elasticsearch

redhat.

STORAGE SECURITY

STORAGE SECURITY

Local Storage Quota



Sometimes we can also have storage isolation requirements: pods in a network zone must use different storage endpoints than pods in other network zones.

We can create one storage class per storage endpoint and then control which storage class(es) a project can use

Security Context Constraints

redhat.

API & PLATFORM ACCESS

API & PLATFORM ACCESS

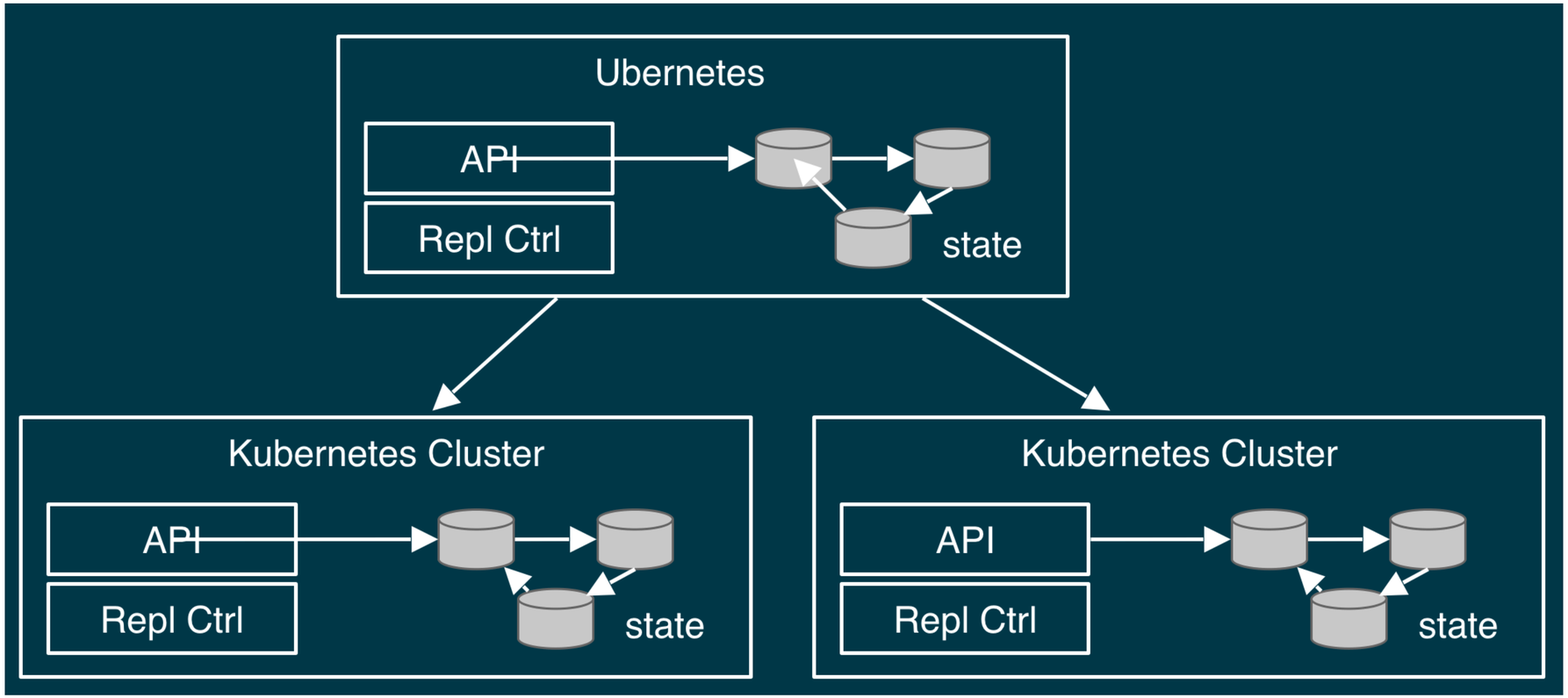
Authentication
via
OAuth tokens and
SSL certificate

Authorization
via
Policy Engine
checks
User/Group
Defined Roles

FEDERATION

FEDERATED CLUSTERS

Roles & access management (in-dev)

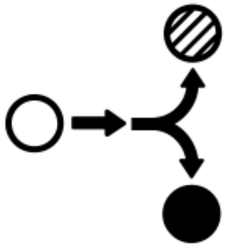


Amazon East

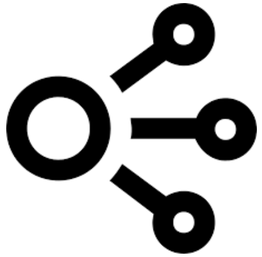
OpenStack

WHAT'S NEXT

Istio



Traffic Control



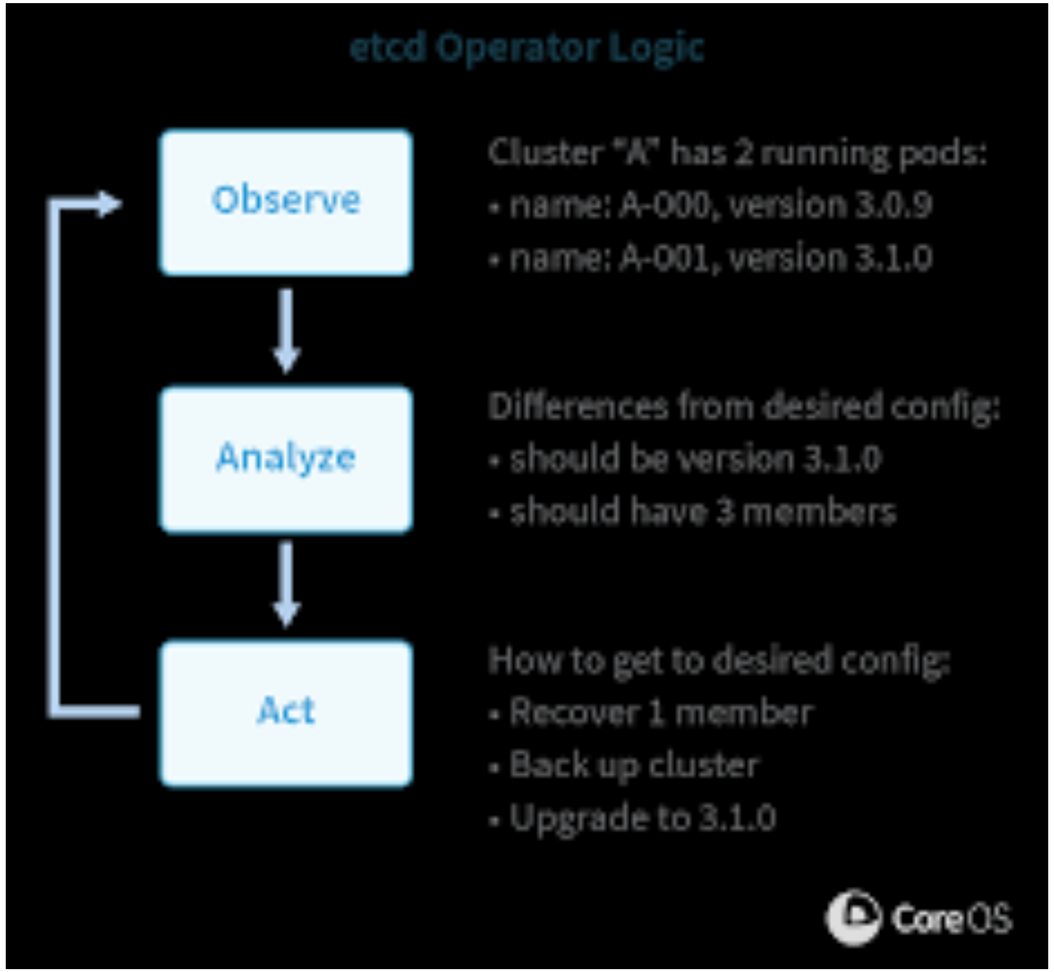
Service Resiliency

Chaos Testing

Observ-ability

Security

OPERATORS

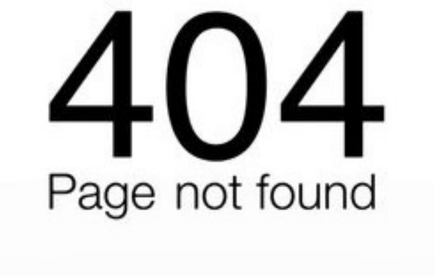


DEVSECOPS METRICS

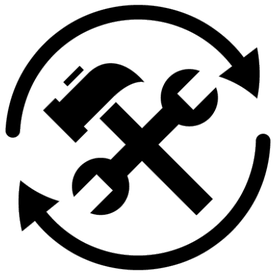
Compliance Score

Deployment Frequency

Lead Time



Deployment Failure Rate



Mean Time to Recover

99.999

Service Availability

THANK YOU

linkedin: Chris Van Tuin
email: cvantuin@redhat.com
twitter: @chrisvantuin