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

Old
Innovation
Execution

New
Innovation
Execution

Static & Planned

Dynamic & Policy Driven

Ford

Tesla

IT’S NOT JUST SOFTWARE, THE DIGITAL LEADERS =

- Speed Up Innovation
  - Empowered organization
  - Move Fast, Break Things

- Shorten the Feedback Loop
  - Real-time data-driven intelligence & personalization
  - Culture of experimentation

20% vs. 25%
IT MUST EVOLVE & KEEP UP
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
DEVSECOPS
Reduce Risks, Lower Costs, Speed Delivery, Speed Reaction

Security Automation

Process Optimization

Continuous Security Improvement
CONTAINERS ENABLE DEVSECOPS

Build

FROM fedora:1.0
CMD echo “Hello”

Build file

Ship

docker.io
Private Registry

Container Image

Run

Physical, Virtual, Cloud

Container Instance
kubernetes

Security Platform

BARE METAL
VIRTUAL
PRIVATE CLOUD
PUBLIC CLOUD
ORCHESTRATION

Deployment, Declarative

Internet

Web
replicas=2,
role=web

Database
replicas=1,
role=db

Controller Manager & Data Store (etcd)

Services

Pods
role=web
role=db

Nodes
role=web

Nodes
HEALTH CHECK

Pods
- role=web
- role=db

Nodes
- role=db
- role=web

Services
- Web: replicas=2, role=web
- Database: replicas=1, role=db

Controller Manager & Data Store (etcd)

Internet
AUTO-SCALE

- **Internet**
- **Pods**
  - role=web
  - role=db
  - replicas=3
- **Nodes**
  - role=web
  - role=db
  - role=web
- **Services**
  - Web
    - replicas=3
    - role=web
  - Database
    - replicas=1, role=db
- **Controller Manager & Data Store (etcd)**

**50% CPU**
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

Container
Application
OS dependencies
LINUX
Guest VM
LAPTOP

Container
Application
OS dependencies
LINUX
BARE METAL

Container
Application
OS dependencies
LINUX
VIRTUALIZATION

Container
Application
OS dependencies
LINUX
PRIVATE CLOUD

Container
Application
OS dependencies
LINUX
PUBLIC CLOUD
CONTAINER IMAGE

JAR

Application

CONTAINER IMAGE

Application

Language runtimes

OS dependencies

1.2/latest

1.1
TREAT CONTAINERS AS IMMUTABLE

Container image

Application
- Language runtimes
- OS dependencies

Config
- Kubernetes configmaps
- secrets

Data
- Traditional data services, Kubernetes persistent volumes
CONTAINER IMAGE SIGNING
Validate what images and version are running

- Authenticating authorship
- Non-repudiation
- Ensuring image integrity
CONTAINER BUILDS
A CONVERGED SOFTWARE SUPPLY CHAIN
CUSTOM SUPPLY CHAIN
BUILD FILE BEST PRACTICES

- 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

Build file

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

CONTAINER HOST SECURITY
CONTAINERS ARE LINUX

Kernel

Hardware (Intel, AMD) or Virtual Machine

Container CLI

SYSTEMD

Containers

Docker Image

Unit File

Drivers

Cgroups

Namespaces

SELinux

Read Only mounts

seccomp
CGROUPS - RESOURCE ISOLATION

Container 1 slice

Container 2 slice

CPU  Memory  Network  Storage / IO
Namespaces - Process Isolation

Namespaces

- Mount
- UTC
- IPC
- PID
- Network
SELINUX - MANDATORY ACCESS CONTROLS

Discretionary Access Controls
(file permissions)

Mandatory Access Controls
(selinux)

Attacker
## SECCOMP AND LINUX CAPABILITIES

**FILTERING SYSTEM CALLS** and **DROPPING PRIVILEGES**

<table>
<thead>
<tr>
<th>Capability</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CAP_SETPCAP</td>
<td>Modify process capabilities</td>
</tr>
<tr>
<td>CAP_SYS_MODULE</td>
<td>Insert/Remove kernel modules</td>
</tr>
<tr>
<td>CAP_SYS_RAWIO</td>
<td>Modify Kernel Memory</td>
</tr>
<tr>
<td>CAP_SYS_PACCT</td>
<td>Configure process accounting</td>
</tr>
<tr>
<td>CAP_SYS_NICE</td>
<td>Modify Priority of processes</td>
</tr>
<tr>
<td>CAP_SYS_RESOURCE</td>
<td>Override Resource Limits</td>
</tr>
<tr>
<td>CAP_SYS_TIME</td>
<td>Modify the system clock</td>
</tr>
<tr>
<td>CAP_SYS_TTY_CONFIG</td>
<td>Configure tty devices</td>
</tr>
<tr>
<td>CAP_AUDIT_WRITE</td>
<td>Write the audit log</td>
</tr>
<tr>
<td>CAP_AUDIT_CONTROL</td>
<td>Configure Audit Subsystem</td>
</tr>
<tr>
<td>CAP_MAC_OVERRIDE</td>
<td>Ignore Kernel MAC Policy</td>
</tr>
<tr>
<td>CAP_MAC_ADMIN</td>
<td>Configure MAC Configuration</td>
</tr>
<tr>
<td>CAP_SYSLOG</td>
<td>Modify Kernel printk behaviour</td>
</tr>
<tr>
<td><strong>CAP_NET_ADMIN</strong></td>
<td>Configure the network:</td>
</tr>
<tr>
<td></td>
<td>- Setting the hostname/domainname</td>
</tr>
<tr>
<td></td>
<td>- mount(), unmount()</td>
</tr>
<tr>
<td></td>
<td>- nfsservctl</td>
</tr>
<tr>
<td><strong>CAP_SYS_ADMIN</strong></td>
<td></td>
</tr>
</tbody>
</table>
READ ONLY MOUNTS

/sys
/proc/sys
/proc/sysrg-trigger
/proc/irq
/proc/bus
CONTAINER HOST SECURITY

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
WHAT’S INSIDE MATTERS…

```c
#include<stdio.h>
main()
{
    printf("Hello World");
}
```

```java
public class HelloWorld {
    public static void main(String[] args) {
        System.out.println("Hello, World");
    }
}
```

```javascript
var http = require('http');
var server = http.createServer(function (request, response) {
    response.writeHead(200, {
        "Content-Type": "text/plain"
    });
    response.end("Hello World
");
});
server.listen(8000);
```

```php
<?php
    $s = "hello world";
    $s = $s/\n        "\n    
    print $s;
```

```bash
perl php
```

 PHYSICAL | VIRTUAL | PRIVATE CLOUD | PUBLIC CLOUD
---|---|---|---

# of critical, important and moderate vulnerabilities identified and fixed by Red Hat in RHEL 7 since GA
CONTINUOUS INTEGRATION WITH SECURITY SCAN
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

Scan
- OpenSCAP

Reports
- Compliance and Scoring
- CVE Database: CVE-2015-5477

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)
## 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
Set Password Strength Minimum Digit Characters

<table>
<thead>
<tr>
<th>Rule ID</th>
<th>accounts_password_pam_dcredit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Result</td>
<td>fail</td>
</tr>
<tr>
<td>Time</td>
<td>2015-07-31T14:57:17</td>
</tr>
<tr>
<td>Severity</td>
<td>low</td>
</tr>
</tbody>
</table>

Identifiers and References:
- CCE-27183-5
- 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$var_password_pam_dcredit/" /etc/pam.d/system-auth
else
  sed -i --follow-symlink "/etc/pam.d/system-auth so/ s/$/ 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
Recreate
RECREATE WITH DOWNTIME

Tests / CI

Version 1.2

Version 1

Version 1

Version 1
RECREATE WITH DOWNTIME

Version 1.2

Tests / CI

Version 1

Version 1

Version 1

Version 1
RECREATE WITH DOWNTIME

Use Case
• Non-mission critical services

Cons
• Downtime

Pros
• Simple, clean
• No Schema incompatibilities
• No API versioning
Rolling Updates
ROLLING UPDATES with ZERO DOWNTIME
Deploy new version and wait until it’s ready…

Health Check: readiness probe e.g. tcp, http, script
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
Blue / Green Deployment
BLUE / GREEN DEPLOYMENT

Route

Version 1

BLUE
BLUE / GREEN DEPLOYMENT

Version 1

Version 1.2

BLUE

GREEN
BLUE / GREEN DEPLOYMENT

Version 1

BLUE

Version 1.2

GREEN

Tests / CI
BLUE / GREEN DEPLOYMENT

Route

Version 1

BLUE

Version 1.2

GREEN
BLUE / GREEN DEPLOYMENT

Use Case
• Self-contained microservices (data)

Cons
• Resource overhead
• Data synchronization

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

Route

Rollback

Version 1

Version 1.2

BLUE

GREEN
RAPID INNOVATION & 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
CANYD DEPLOYMENTS

100%

Route

Version A

25% Conversion Rate

Version B

?! Conversion Rate

Tests / CI
CANTARY DEPLOYMENTS

50% 50%

Route

Version A  Version B

25% Conversion Rate  30% Conversion Rate
CANYON DEPLOYMENTS

Version A

25% Conversion Rate

Route

Version B

30% Conversion Rate

100%
CANTARY DEPLOYMENTS

100%

Version A

25% Conversion Rate

Route

Rollback

Version B

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 Namespace provides resource isolation

NETWORK ISOLATION

Multi-Environment

Multi-Tenant
NETWORK POLICY

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

Policy applied to namespace: project-a

```yaml
kind: NetworkPolicy
apiVersion: extensions/v1beta1
metadata:
  name: allow-from-same-namespace
spec:
podSelector:
  ingress:
  - from:
    - podSelector: {}
```
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

MONITORING & LOGGING
## Kubernetes Monitoring Considerations

<table>
<thead>
<tr>
<th>Stack</th>
<th>Metrics</th>
<th>Tool</th>
</tr>
</thead>
</table>
| 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 |
Aggregate platform and application log access via Kibana + Elasticsearch
STORAGE SECURITY
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.
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)
WHAT’S NEXT
Traffic Control
Service Resiliency
Chaos Testing
Observability
Security
OPERATORS

etcd Operator Logic

- Cluster “A” has 2 running pods:
  - name: A-000, version 3.0.9
  - name: A-001, version 3.1.0

- Differences from desired config:
  - should be version 3.1.0
  - should have 3 members

- How to get to desired config:
  - Recover 1 member
  - Back up cluster
  - Upgrade to 3.1.0
## DEVSECOPS METRICS

<table>
<thead>
<tr>
<th>Compliance Score</th>
<th>Deployment Frequency</th>
<th>Lead Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deployment Failure Rate</td>
<td>Mean Time to Recover</td>
<td>Service Availability</td>
</tr>
<tr>
<td>404 Page not found</td>
<td>99.999</td>
<td></td>
</tr>
<tr>
<td>99.999 Service Availability</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
THANK YOU

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