A Security State of Mind: Continuous Security with Kubernetes

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

“In short, software is eating the world.”

- Marc Andreessen, Wall Street Journal, August 2011
UBER, LYFT FALLOUT: TAXI RIDES PLUNGE 65% IN SAN FRANCISCO
“THROW IT OVER THE WALL”

Walled off people, walled off processes, walled off technologies
THE NEED FOR SPEED
THE ACCELERATION OF APPLICATION DELIVERY FOR THE BUSINESS

HOW
Waterfall
Agile
DevOps

WHAT
Monolithic Apps
N-tier Apps w/ Appservers
Microservices w/ APIs

WHERE
Physical Server
VMs
Containers

FASTER AND HIGHER QUALITY
LINUX CONTAINERS
BUILD, SHIP, RUN

Dockerfile
FROM fedora:latest
CMD echo "Hello"

Build
“docker build or commit”

Image

Ship
“docker push or pull <IMAGE_ID>”

Container
Physical, Virtual, Cloud

Run
“docker run <IMAGE_ID>”
CONTAINER SECURITY
WHAT IS THE GREATEST SECURITY RISK?

- 36% - Employees not taking proper security measures
- 32% - Outside breach
- 14% - Unpatched or unpatchable
- 11% - Internal attack by an employee
- 4% - Shadow IT
- 3% - Bring your own device/mobile

Source: Techvalidate/Red Hat
“Patch? The servers are behind the firewall.”

- Anonymous (far too many to name), 2005 - …
CONTAINER SECURITY RISKS

- Kernel exploits
- Denial of Service attacks
- Container breakouts
- Poisoned images
- Compromised secrets
BUILD, SHIP, RUN

Dockerfile

FROM fedora:latest
CMD echo “Hello”

Build

“docker build or commit”
CONSISTENT PACKAGING FORMAT
Docker provides a language agnostic packaging format and runtime API
# Packaged Dependencies

Package dependencies ensure consistency and portability

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

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

```bash
$_ = "hello world";
$_ =~ s/^\b\w(\B\w+)\s(\D)(\D+)$/
\U$1\E$2 \U$3\E$4\!
/;
print $_;
```

**Physical**
- bash
- glibc

**Virtual**
- java
- ...
- bash
- glibc

**Private Cloud**
- nodejs
- libssl
- ...
- bash
- glibc

**Public Cloud**
- perl
- ...
- bash
- glibc
TRADITIONAL SOFTWARE SUPPLY CHAIN

1. Core Build
2. Core Build
3. Core Build

- Core Build (Kickstart)
- Middleware (Manual TAR)
- Application Code (WAR)
A CONVERGED SOFTWARE SUPPLY CHAIN
BUILD, SHIP, RUN

Dockerfile

```
FROM fedora:latest
CMD echo “Hello”
```

Image

```
Red Hat Certified
Local Private Registry
```

Build

“docker build or commit”

Ship

“docker push or pull <IMAGE_ID>”
64% of official images in Docker Hub contain high priority security vulnerabilities.

Examples:
- ShellShock (bash)
- Heartbleed (OpenSSL)
- Poodle (OpenSSL)

SECURITY IMPLICATIONS

What's inside the container and where it comes from matters

```java
#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);
```

```javascript
$.send("hello world";
$.send("hello world");
```

```
<?php
    print "Hello, World!";
?>
```

# of critical important and moderate vulnerabilities identified and fixed by Red Hat in RHEL 7 since GA
OpenSCAP
Scan physical servers, virtual machines, docker images and containers for Compliance (CCEs) and known Vulnerabilities (CVEs)

Content

SCAP Security Guide for RHEL

CCE-27002-5
Set Password Minimum Length

Scan

OpenSCAP

FOREMAN

Reports

Compliance and Scoring

 CVE-2015-5477

Impact: important

Policy: not_applicable
BUILD, SHIP, RUN

Dockerfile

FROM fedora:latest
CMD echo "Hello"

Image

Red Hat Certified
Private Registry
docker.io

Container

Physical, Virtual, Cloud

Build

“docker build or commit”

Ship

“docker push or pull <IMAGE_ID>”

Run

“docker run <IMAGE_ID>”
VIRTUAL VS CONTAINERS

Virtual

- **APP A**
- **APP B**
- **LIBS A**
- **LIBS B**
- **LIBS**

Virtual Machine (Kernel)

HOST OS (Kernel)

HARDWARE

Containers

- **CONTAINER**
  - **APP A**
  - **LIBS**

HOST OS (Kernel)

HARDWARE
UNDERLYING TECHNOLOGY

- **Containers**
- **DOCKER CLI**
- **SYSTEMD**
- **Docker Image**
- **Unit File**
- **Cgroups**
- **Namespaces**
- **SELinux**
- **Drivers**
- **RHEL Kernel**
- **Hardware (Intel, AMD) or Virtual Machine**
WHAT ARE MY OPTIONS?

Security-by-Luck
- Unsupported distro
- End-of-lifed

Security-by-Firewall
- No patch management
- Untrusted containers
- selinux disabled

Trusted Platform
- Supported distro
- Patch management
- Trusted containers & host
- Container scanning
- selinux enforcing
CI/CD with Containers
More serious workloads require orchestration like Kubernetes to offload management overhead.
Automated Builds with Security Scan
Kubernetes: Deploy Application
Kubernetes: Deploy Application
Kubernetes: Deploy Application
New applications can be spun up and tested before old applications are removed, lowering risk for upgrades
Some call this method of deployment red-black deployment, admins won't get stuck in the middle of an upgrade.
Tests and certification can be done before customers access it.
Once ready, the new version is used and the old version can be removed
Rollbacks can be done using the same method if desired
CONTAINER BEST PRACTICES

• Only run container images from trusted parties
• Container apps should drop privileges
• Host operating system matters
• Apply kernel security fixes
• Do not disable selinux
• Examine container images for security flaws
• Automated build, automated deploy
• Incorporate security scans into your CI/CD pipeline
THANK YOU

Chris Van Tuin
cvantuin@redhat.com