Red Hat

Security Symposium

#SecuritySymposium

Security Symposium

The security implications of running software in containers

Scott McCarty
Principal Technical
Product Manager

2020

#SecuritySymposium

2020 Security Symposium Welcome

Thank you for joining us for two days of security technology conversations.

A few notes:

- We have three tracks with multiple talks:
    - Security and Compliance Automation (May 14)
    - Containers and Kubernetes Security (May 14)
    - DevSecOps (May 15)
- → You must register for each unique webinar and panel.
- All sessions will be available on-demand, kindly register and you'll be invited to view on-demand presentations.
- View attachments tab for links to presentations and/or collateral.
- Want more? Grab this ebook on Boosting Hybrid Cloud Security red.ht/security101
- The panels are live, send us your questions throughout each day to infrastructure@redhat.com.
- Keep an eye out for the 'Financial Services Security Automation Summit' on June 11 on BrightTalk .

Red Hat

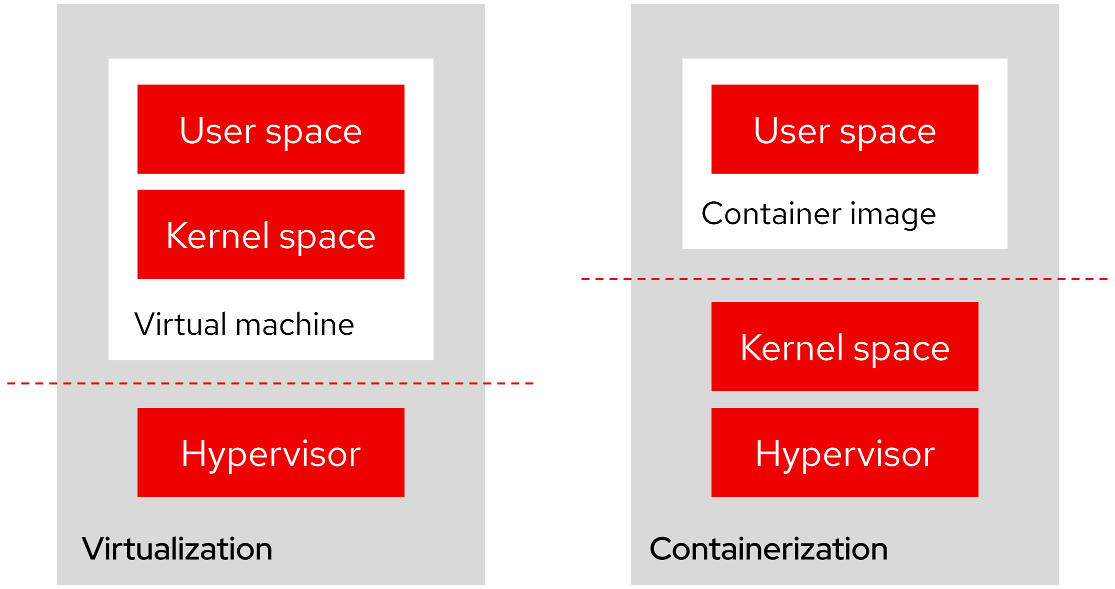
“Just because you’re paranoid doesn’t mean they aren’t after you.”

— Joseph Heller, *Catch-22*

The problems

Containers don’t contain

Dan Walsh (my shirt is dedicated to you)

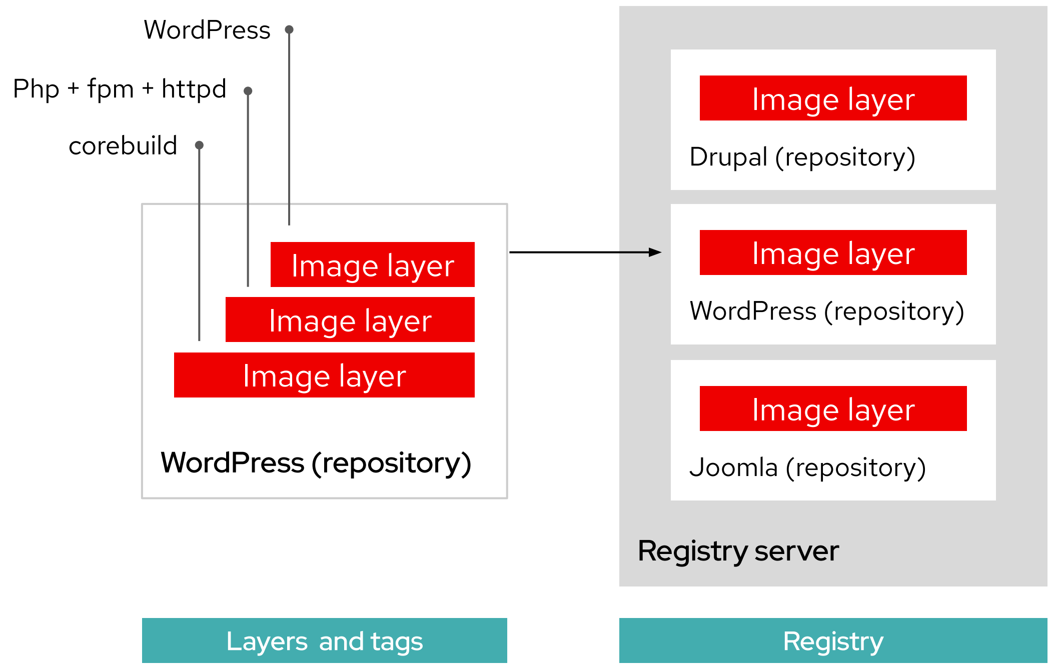


Move the kernel around or move the user space around:

- Fancy processes
- Breaking the OS into 2 pieces
- All containers share a kernel
- Root-only exploits can be bad

Container images

Currency for collaboration



Developers, operations, middleware, performance, and security specialists all have a role to play.

- Fancy files?
- Who controls what?
- Who is responsible for what?
- What about bad content?

Hard work

Code: mysqld

Configuration:

/etc/my.cnf

Data: /var/lib/mysql

Other stuff

Red Hat

New concepts

CIA

Not them, but yeah, they might be after you too ...

Confidentiality

Has data leaked from the container platform?

Integrity

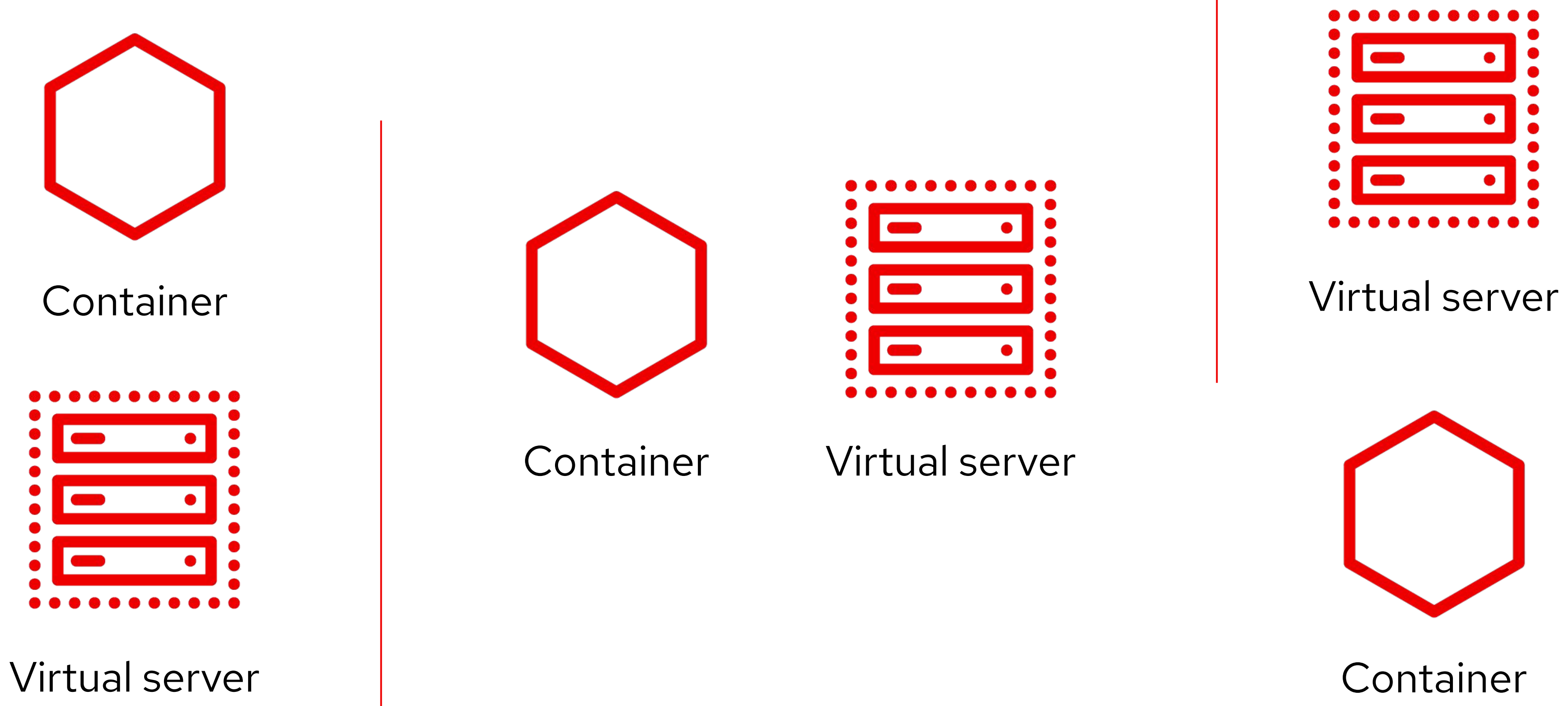
Has somebody tampered with the container?

Availability

Is the container up and running?

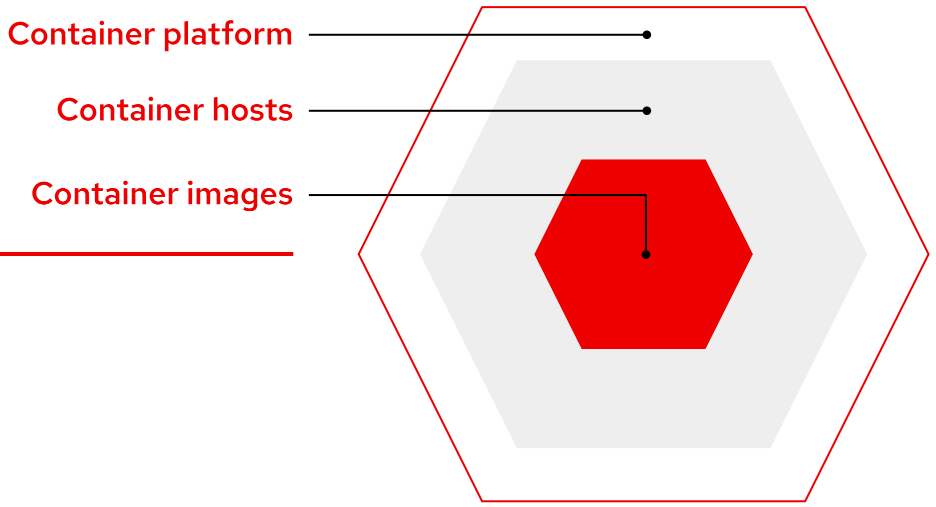
Red Hat

Integrity


Container
Virtual server
Container
Virtual server
Virtual server
Container

Defense in depth

The practice of arranging defensive lines or fortifications so that they can defend each other, especially in case of an enemy incursion

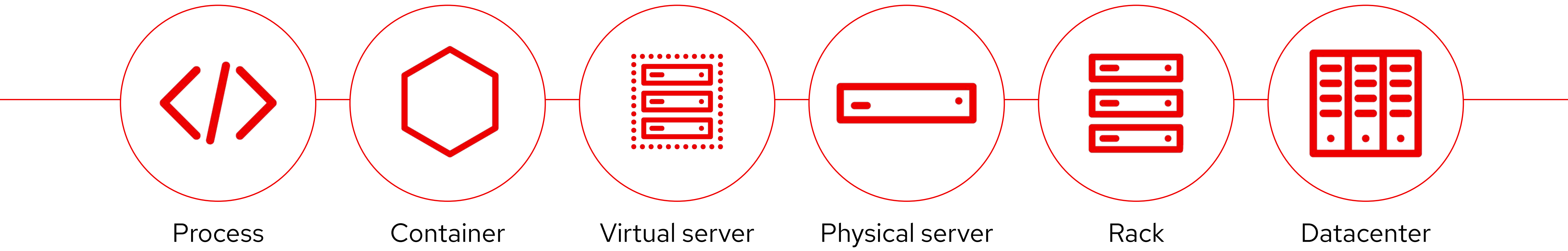


Can we harden each layer?

- ✓ Image scanning, signing, and blueprinting
- ✓ Container host hardening
- ✓ Platform delegation practices

Red Hat

The tenancy scale



Security controls

SELinux

- Who you can talk to

  Which objects in the kernel can communicate with other objects

SECCOMP

- What you can say

  Limiting system calls is like limiting what words can be said

New technical controls

Container images

Containers add the ability to easily apply techniques

- Bill of materials
- Signing
- Read-only containers
- Podman diff to see what changed in a container

Our current operating model controls

- ✓ Trusted content
- ✓ Content provenance
- ✓ Security scans
- ✓ Remediation/patching
- ✓ Bill of materials
- ✓ CVE databases
- ✓ Security response teams
- ✓ Limited root access
- ✓ Limited user access

Red Hat

Container host

We apply many of these techniques today:

- ✓ Kernel quality
- ✓ Capabilities
- ✓ Read-only images
- ✓ Limiting SSH access
- ✓ Well understood and controlled configuration
- ✓ Tenancy

Since containers are just fancy processes with a well-controlled user space, it's easier to apply techniques like …

SECCOMP + sVirt

Hardening:

```
NO_NEW_PRIVS, Read Only
Images, -cap-drop=ALL,
-user=user
```

Red Hat

Container platform

This layer exists in the world of physical and virtual servers but is typically an administrator-only tool, such as vCenter or HPSA.

In the world of containers, it's much more common to delegate some access to developers, architects, and application owners.

- Role-based authorization
- Authentication
- Environment isolation
- User demarcation
- Network separation
- Key management

Standard web application

Many security controls are inconvenient

Benefits

- Network firewall (possibly layer 7)
- Host-based firewall
- Kernel quality
- CVE database
- Well-understood tenancy
- Understood remediation and patching
- Security scanning

Limitations

- Tripwire, SELinux, SECCOMP usually disabled
- Mutable user space
- No temporal understanding
- No spatial understanding (code, configuration, data)
- No platform delegation granularity
- Patched infrequently

Containerized web application

Many security controls are essentially free

Benefits

✓

- All tools from standard web application
- Read-only containers
- Signing
- Platform delegation
- Spatial and temporal understanding of containers and application
- Updates practiced more

Limitations

✗

- Tenancy not well understood
- Shared kernel
- Applications hard to break up into code/configuration/data
- More infrastructure (platform and management)
- Need better understanding of applications

Red Hat

Questions?

Citations

GitHub: Supply chain demo: http://bit.ly/2aY1WEO

The New Stack: Container defense in depth: http://bit.ly/2buXflB

Red Hat: Architecting containers series: http://red.ht/2aXjVJF

Red Hat: A practical introduction to Docker terminology: http://red.ht/2beXHDD

WhatIs: Confidentiality, Integrity, and Availability: http://bit.ly/2bcStO9

Questions

redhat.
CONTAINERS FOR THE ENTERPRISE
DELIVER APPS FASTER
COMPREHENSIVE SECURITY
DEPLOY & MANAGE AT SCALE
UNIFIED ENVIRONMENT

Security Symposium

#SecuritySymposium

Red Hat Summit

Thank you

linkedin.com/company/Red-Hat

facebook.com/RedHatinc

youtube.com/user/RedHatVideos

twitter.com/RedHat

Red Hat

Red Hat Summit

The security implications of running software in containers

Taming container fears

Scott McCarty
Principal Product Manager
Containers: Red Hat Enterprise Linux & Red Hat OpenShift

Red Hat