The security implications of deploying software in containers

Hanover, MD

Scott McCarty
Principal Technology manager
@fatherlinux

Scott McCarty

Principal Product Manager,
Containers - OpenShift & RHEL

Red Hat

@fatherlinux

Red Hat | intel

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

— Joseph Heller, Catch-22

Twitter: @fatherlinux

Concept Refresher

Confidentiality, Integrity, Availability (CIA)

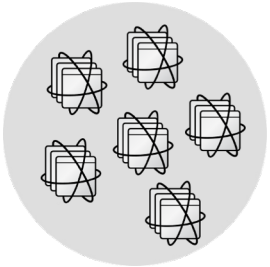
The basics still apply with containers…

CONFIDENTIALITY

Has data leaked from the container platform?

INTEGRITY

Has somebody tampered with the container?



AVAILABILITY

Is the container up and running?

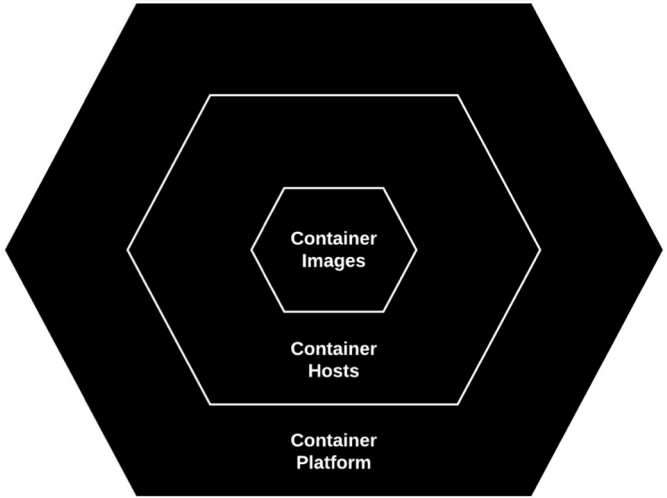
Red Hat | intel

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.

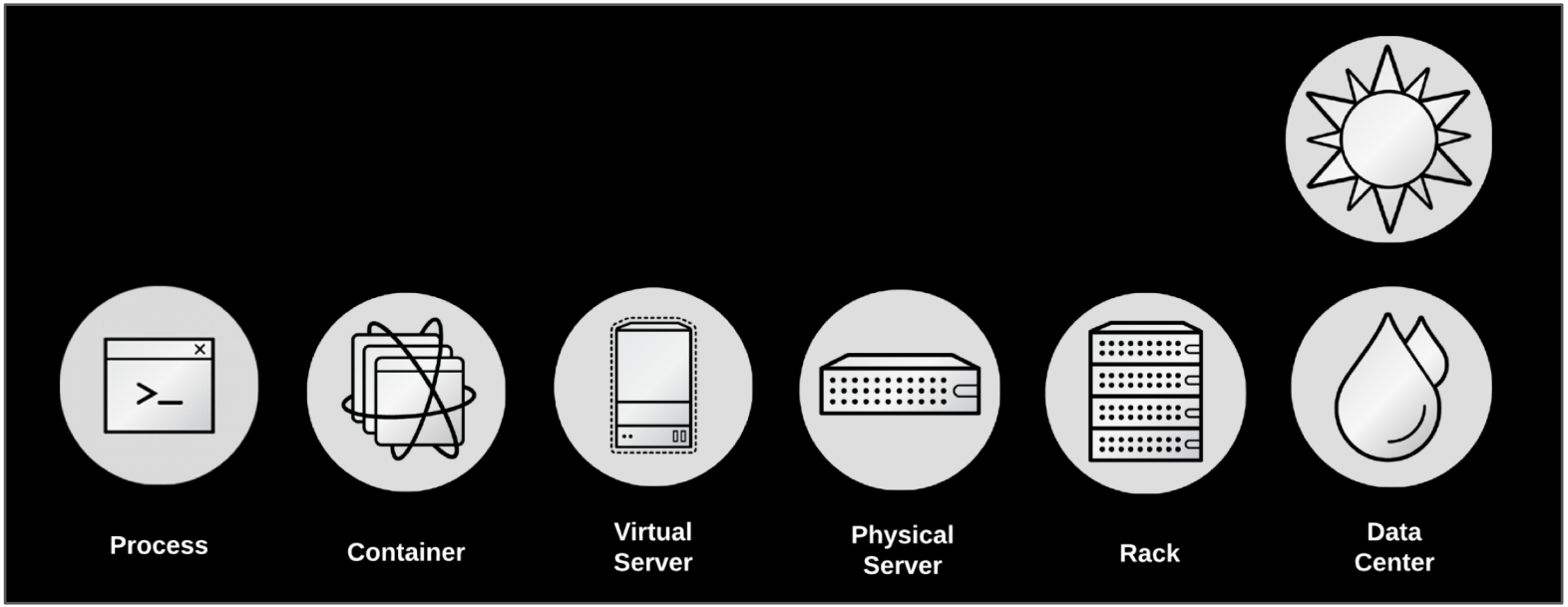
Can we harden each layer?

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



Red Hat | intel

The Tenancy Scale



Red Hat | intel

Important Questions

Question #1

Do I even need a base image?

Options:

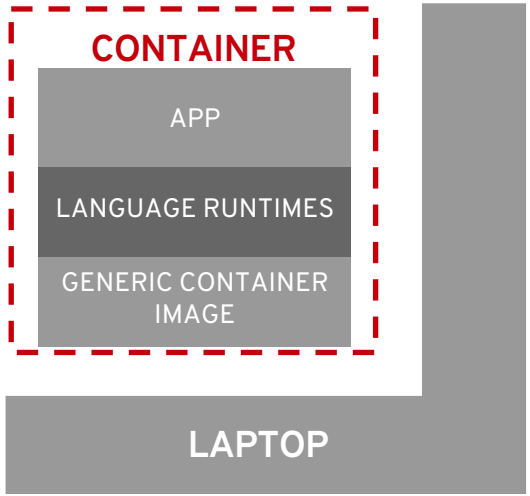
- YUM - just pull the packages you need in a multi-stage build
- Distroless - some programming languages compiled
- Scratch - literally nothing, just a scratch image

Reality:

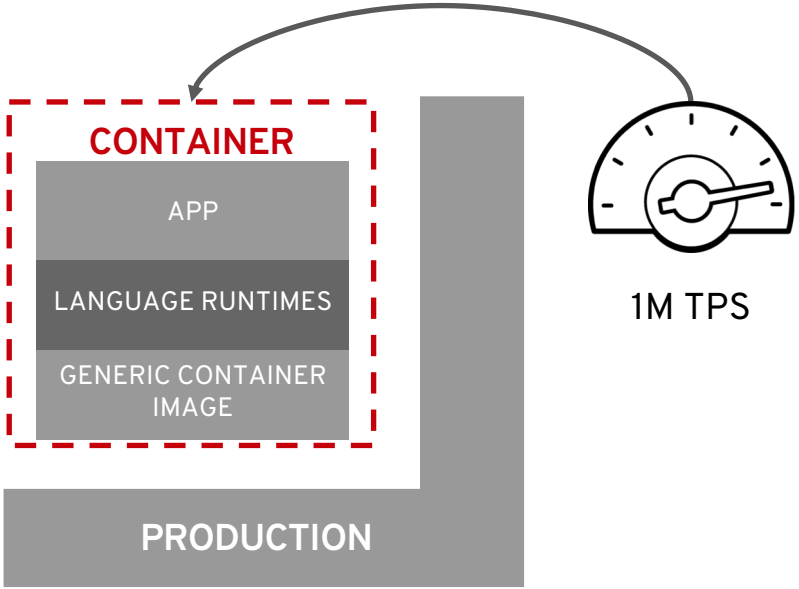
- You are likely pulling in pre-built packages
- You are compiling everything yourself
- When things break, it’s a developer action, not an operations action

Question #2

How do I guarantee performance in production?



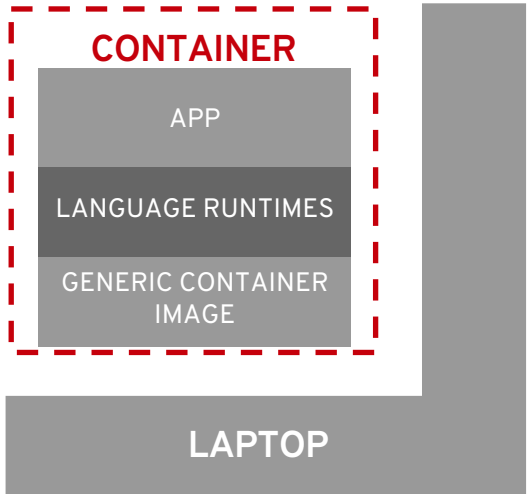
Works on my laptop



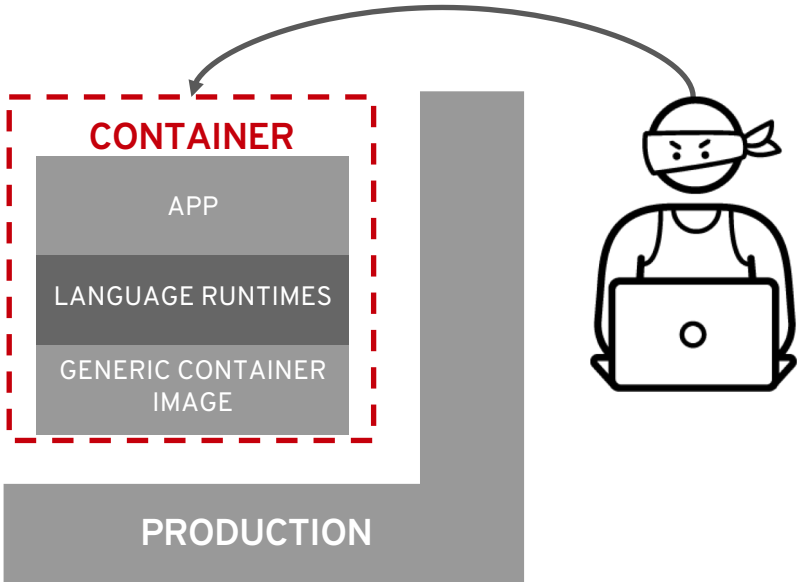
But, what about at 1M transactions per second

Question #3

How do I guarantee security in production?



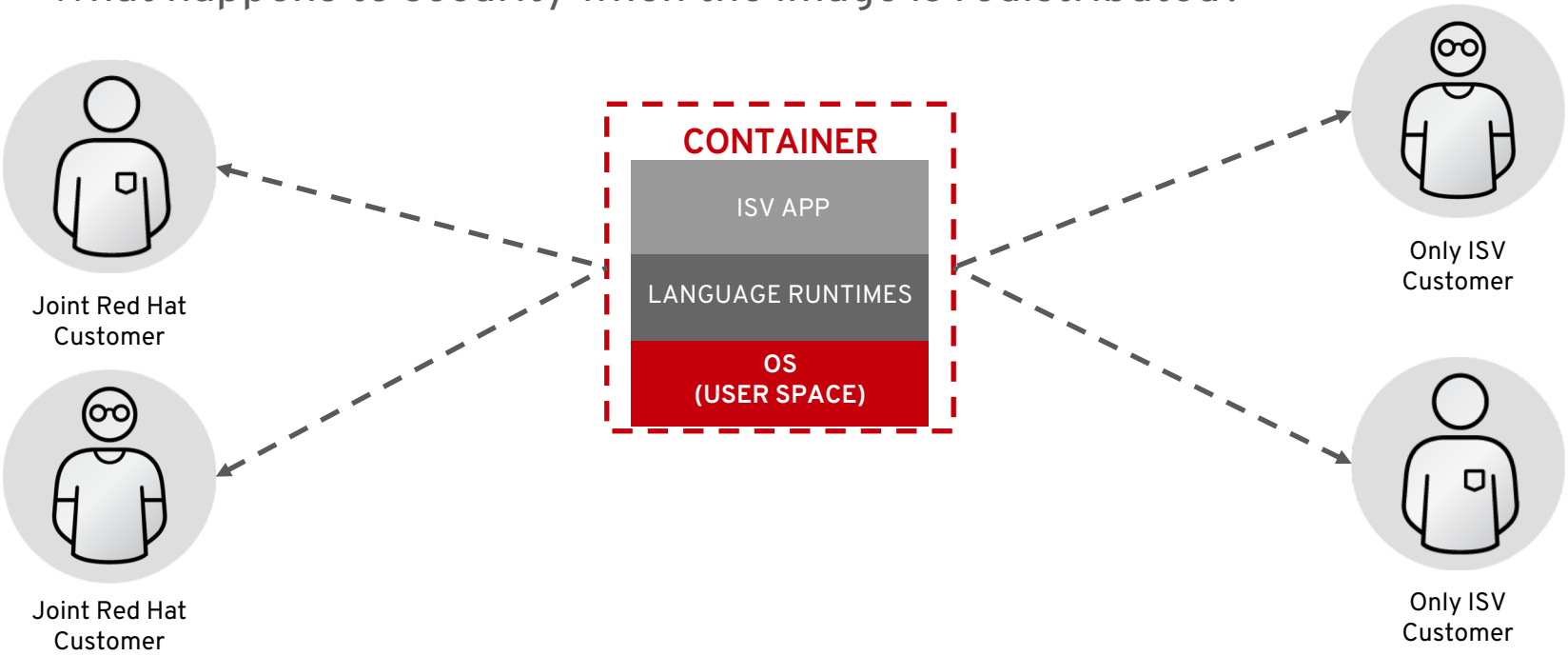
Works on my laptop



What about hackers?

Question #3

What happens to security when the image is redistributed?



Question #5

What else am I not thinking about?

Architecture

- C Library
- Core Utilities
- Size
- Life Cycle
- Compatibility
- Troubleshooting
- Technical Support
- ISV Support
- Distributability

Security

- Updates
- Tracking
- Security Response Team

Performance

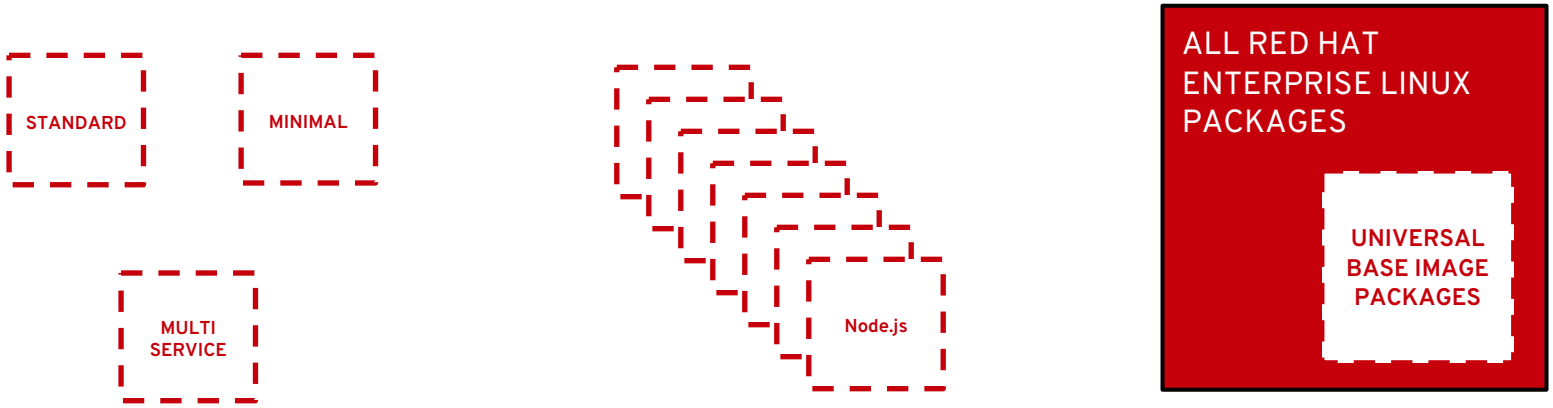
- Automated
- Performance Engineering

Basic Recommendations


Security starts at the factory, not the dock

Start With a Trusted Foundation

Trusted base images, language runtime images, and software packages



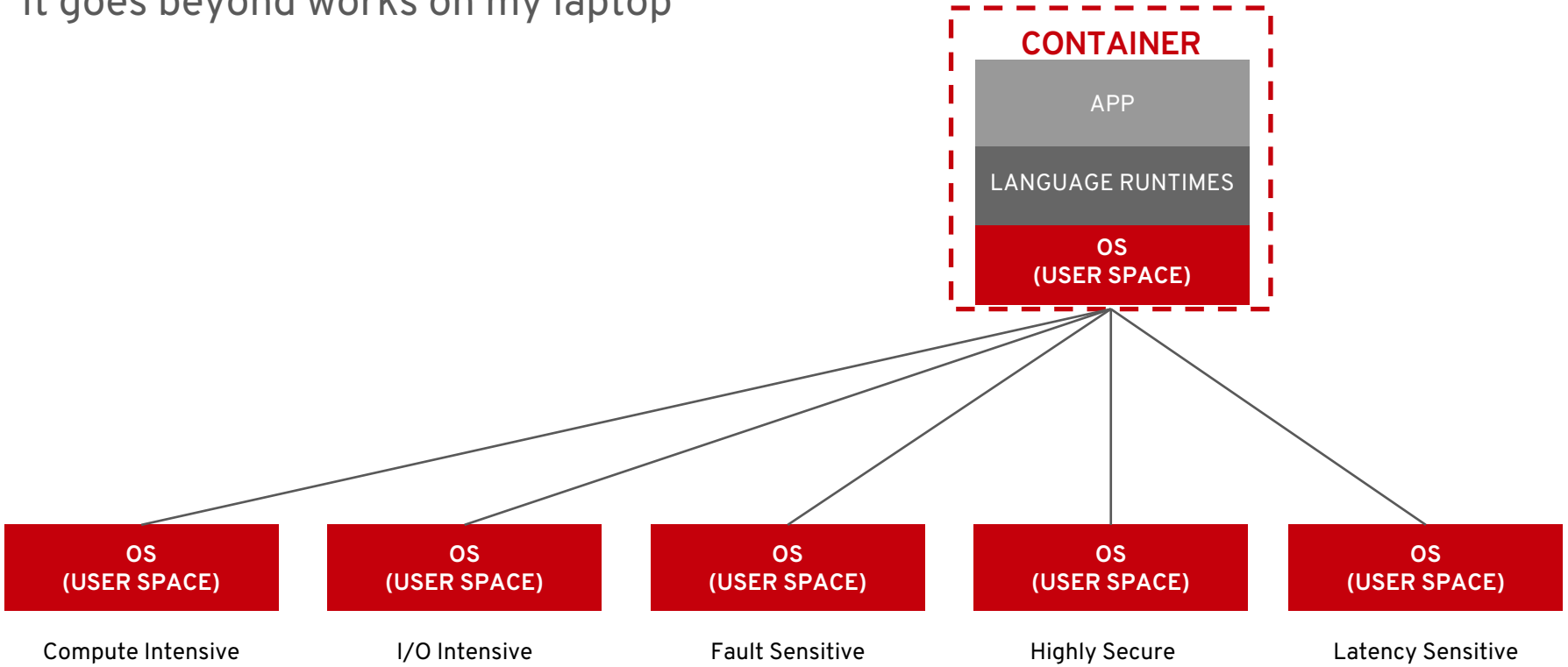
Base Images

Pre-Built Language Images

Package Subset

Same Bits Used in Mission Critical Workloads

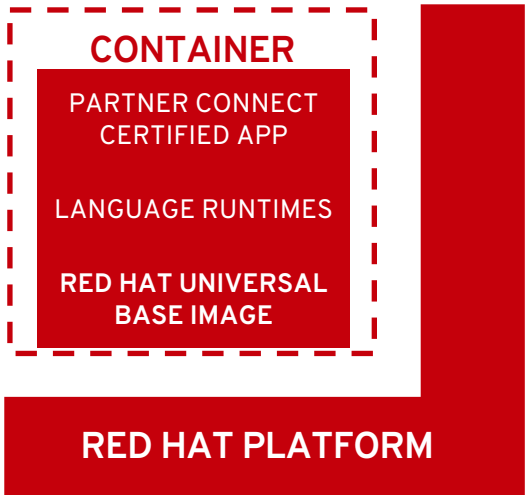
It goes beyond works on my laptop



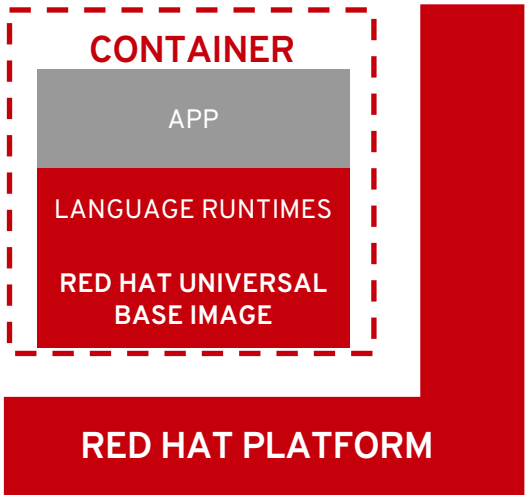

Security must be maintained at the dock

Red Hat Universal Base Image

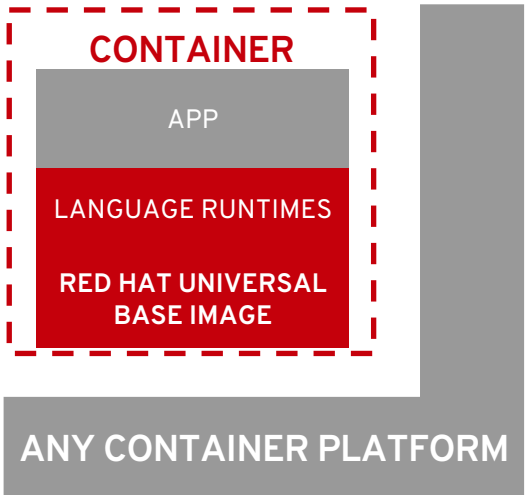
Provides a foundation, but not the only thing to think about



Think about application



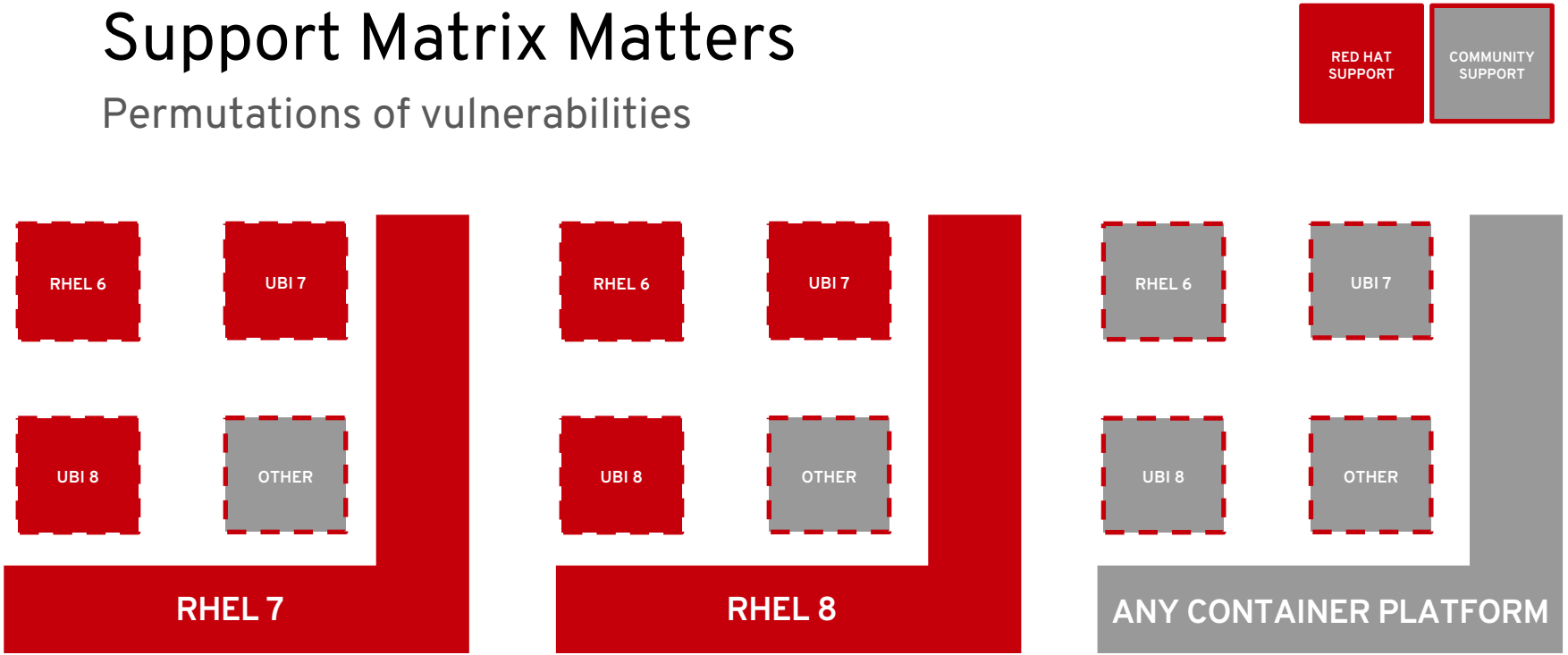
Think about platform



Think about the base image

Support Matrix Matters

Permutations of vulnerabilities



Red Hat Enterprise Linux 7

Red Hat Enterprise Linux 8

Like any upstream project

Twitter: @fatherlinux

Advanced Recommendations

Container Images

Our current operating model controls:

- Trusted Content (What's in the container matters. Don't install from hackme.com.)
- Content Provenance (Track who changed what.)
- Security Scans
- Remediation/Patching
- Bill of Materials
- CVE Databases
- Security Response Teams
- Limit Root Access (Don't oversell User Namespaces.)
- Limit User Access (Who controls content)

Containers add the ability to easily apply techniques such as:

- Bill of Materials
- Signing
- Read-only Containers (Read-only servers were popular in the late 90s.)
- Atomic diff/Docker diff to see what changed in a container

Container Hosts

Think about:

- Quality of Linux kernel
- Quality of container engine
- Quality of orchestration platform

Apply technologies and techniques:

- SECCOMP
- sVirt
- Hardening: NO_NEW_PRIVS, Read Only Images, –cap-drop=ALL, –user=user
- Capabilities
- Run read only
- Limiting ssh access (root access and users)
- Well understood/controlled configuration (OpenShift 4)
- Tenancy

Orchestration Platform

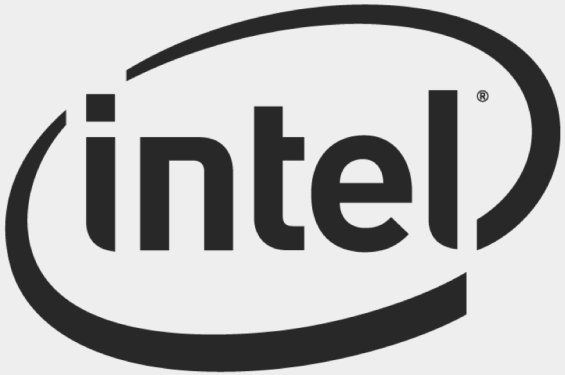
This layer exists in the world of physical and virtual servers but is typically an administrator only tool, such as vCenter or HPSA. In the world of containers, it's much more common to delegate some access to developers, architects, and application owners.

- Role-Based authorization
- Authentication (LDAP, network level access/restriction to the platform)
- Environment isolation (development, testing, production)
- User demarcation (kubectl exec)
- Network separation
- Key/secrets management
- Well understood/controlled configuration (OpenShift 4)

Other Good Talks

1:30 – 2:20 p.m.	**Automating security and compliance for hybrid environments** Lucy Kerner *senior principal security global technical evangelist and strategist, Red Hat*
2:30 – 3:20 p.m.	**Securing a multi-tenant Kubernetes cluster** Kirsten Newcomer *OpenShift senior principal product manager, Red Hat*
3:30 – 4:20 p.m.	**Container security and new container technologies** Dan Walsh *senior distinguished engineer, Red Hat*

Thank you to our partner



Further Reading

1:30 – 2:20 p.m.	**Automating security and compliance for hybrid environments** Lucy Kerner *senior principal security global technical evangelist and strategist, Red Hat*
2:30 – 3:20 p.m.	**Securing a multi-tenant Kubernetes cluster** Kirsten Newcomer *OpenShift senior principal product manager, Red Hat*
3:30 – 4:20 p.m.	**Container security and new container technologies** Dan Walsh *senior distinguished engineer, Red Hat*

Next Steps

- Speak with a Red Hat expert here at Security Symposium
- Look for the slides in a “Thank You” email from us in the next few days
- Stay up to date with Red Hat at redhat.com/security
- Visit redhat.com/events to find out about workshops and other events like this one coming to your area

Thank you for coming.

Feedback or questions?

infrastructure@redhat.com