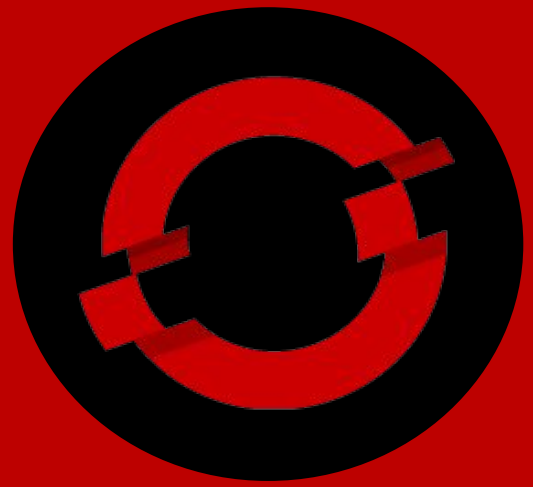
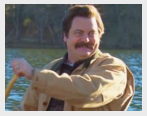


Joshua Smith
Application Platform Solutions Architect

Laine Vyvyan
Channel Solutions Architect



Red Hat

Laine Vyvyan

Josh Smith

Red Hat

AGENDA

The Analogy: Your Neighborhood in a Bad City

Applying the Analogy

Security as a Continuous Process

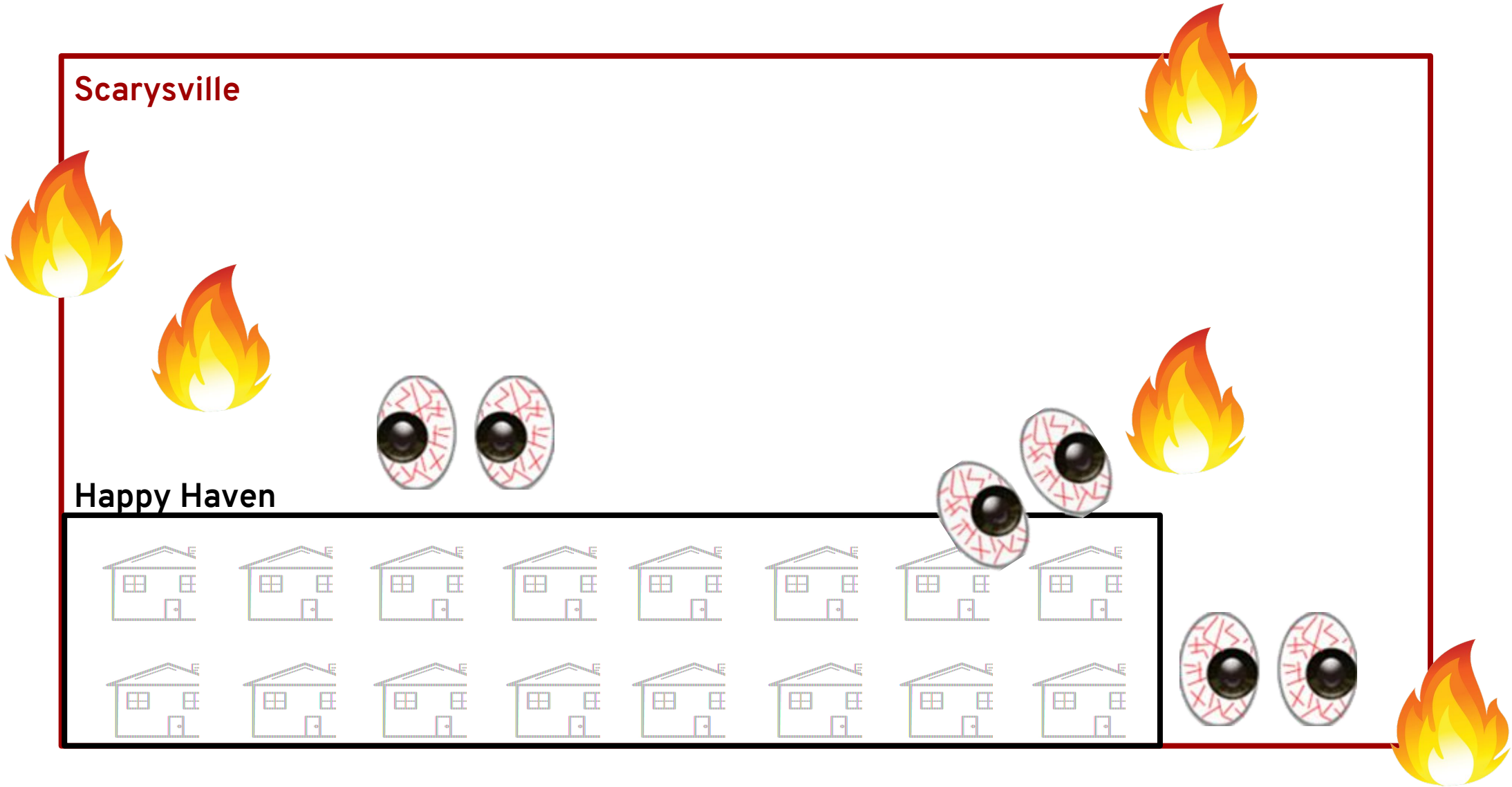
Building Alliances - or Why Security is Awesome

OUR GOALS FOR YOU

1. Understand the security challenges and **benefits** of running applications in containers via Kubernetes and OpenShift.
2. Understand that **security isn't the enemy of speed**.

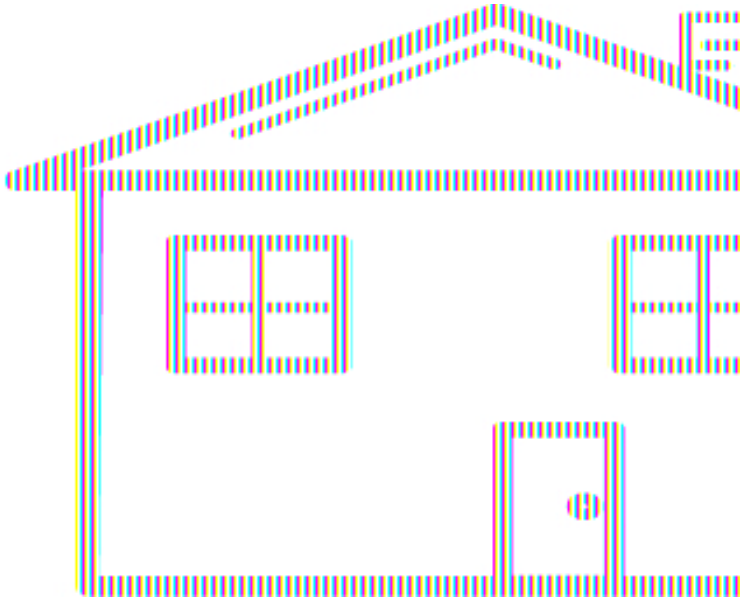
The Analogy:
Your Neighborhood in a Bad City

Red Hat

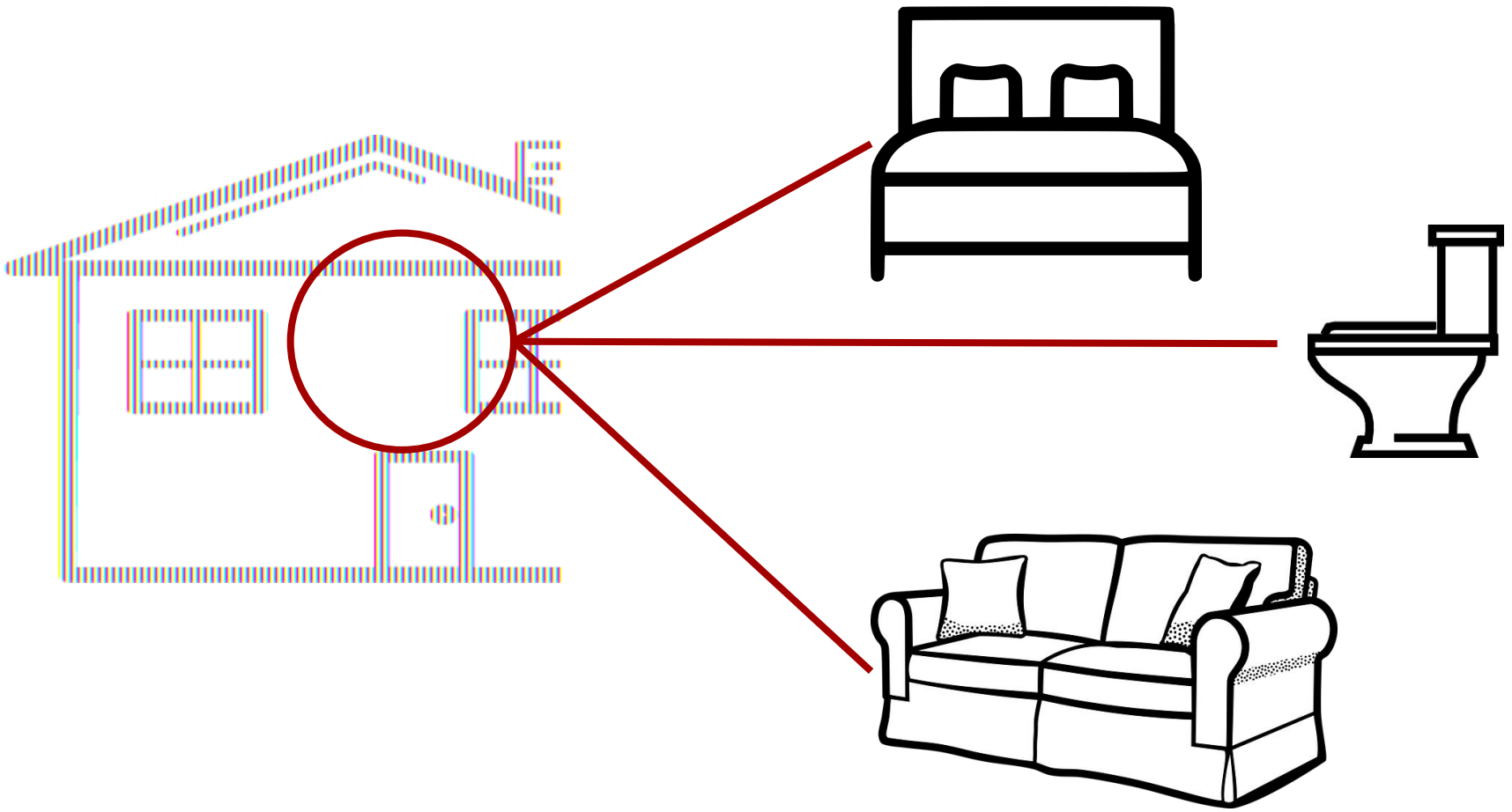


YOUR NEIGHBORHOOD

YOUR HOUSE



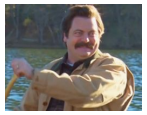
ROOMS IN YOUR HOUSE



IN GENERAL, YOU WANT...

As a resident of Happy Haven, you want a nice place to live - to *flourish* - where you feel like your wants and needs are addressed, and where you have the freedom to live as you see fit.

But you *also* want Happy Haven to be ***safe from Scarysville***.



TWO KINDS OF SECURITY RISKS

Necessary vulnerabilities

You need to have doors and windows in your houses, and gates into your neighborhood. But this allows entry by anyone with a key - *or* anyone with some determination or sneakiness.

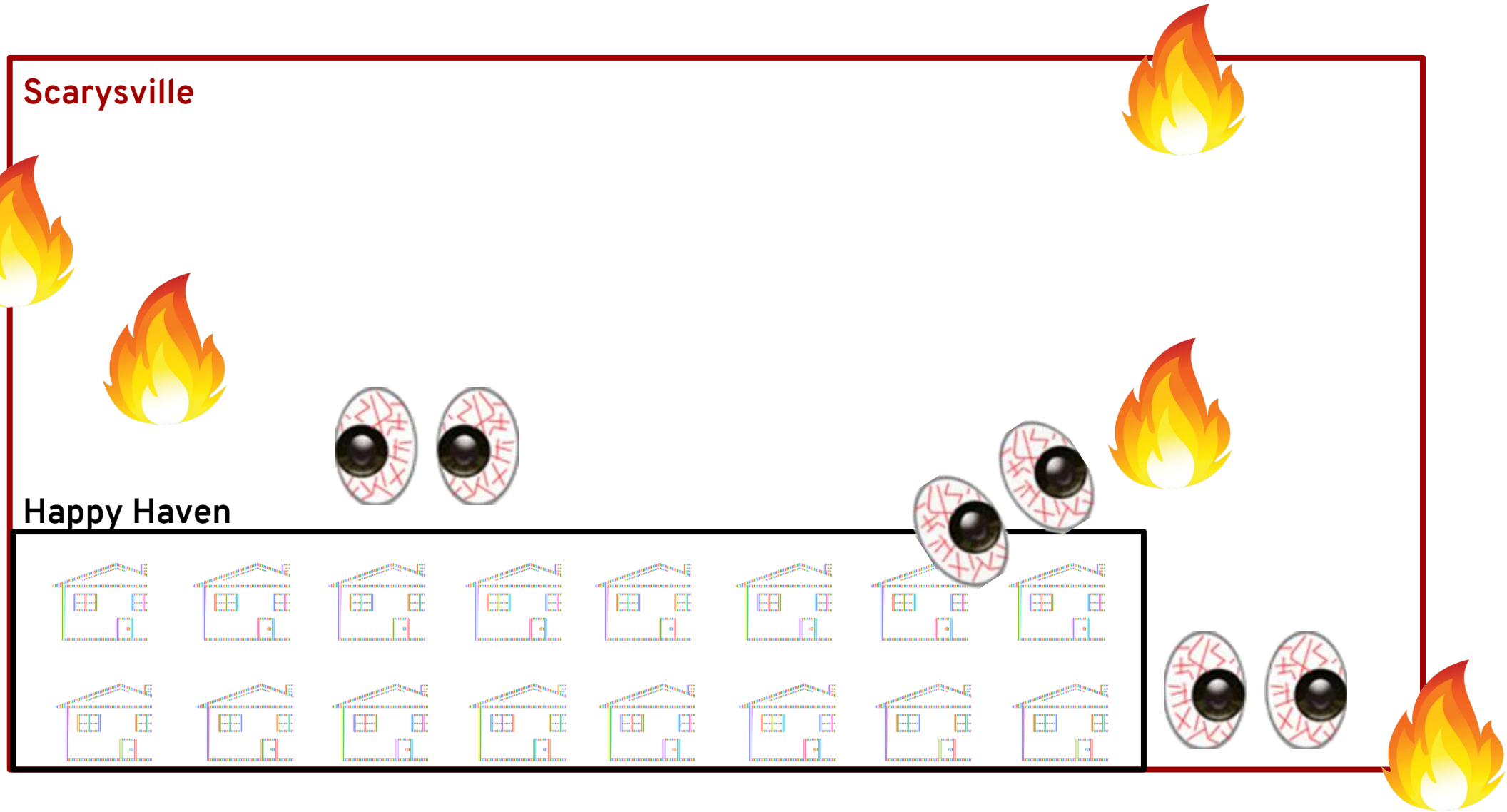
TWO KINDS OF SECURITY RISKS

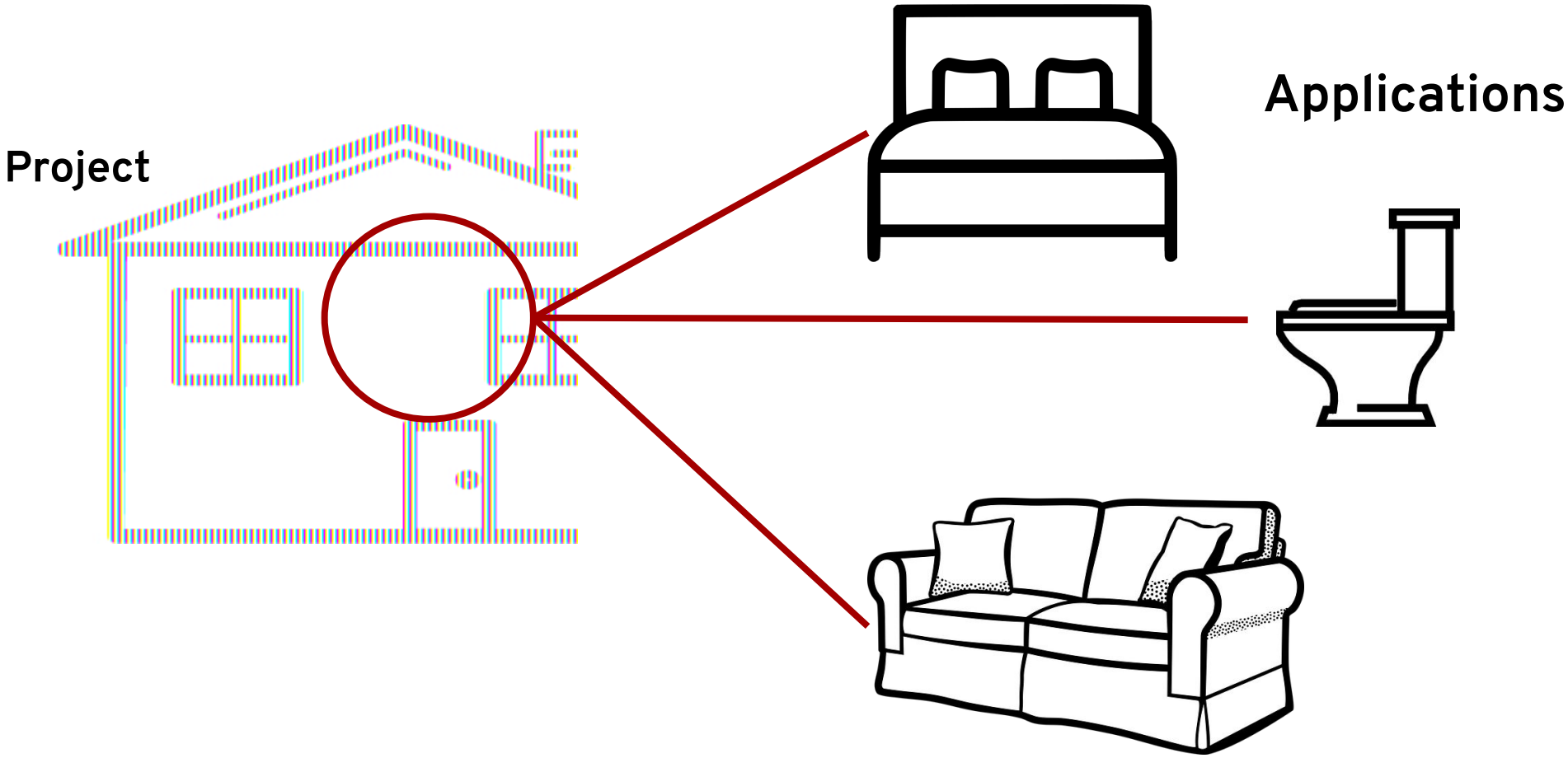
Wear over time

a. The house: stuff wears out and breaks - appliances, decks, sump pumps...
b. The neighborhood: ...stuff also wears out and breaks, but on a larger scale - sewers, electricity, etc

Applying the Analogy

Red Hat

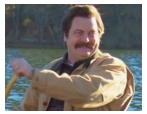




IN GENERAL, YOU WANT...

For your company, using the application platform you choose, you want a nice place for your applications to live - to *flourish* - where they can function per business requirements, delivered quickly and as close to the “right” way as possible.

But you *also* want that platform to be ***safe from the rest of the internet***. 



TWO KINDS OF SECURITY RISKS

Necessary vulnerabilities

You need to allow users into your platform, and into your applications. But this allows entry by anyone with a key *or* anyone with some determination or sneakiness.

TWO KINDS OF SECURITY RISKS

Wear over time

a. The house: stuff wears out and breaks - appliances, decks, sump pumps...

b. The neighborhood: ...stuff also wears out and breaks, but on a larger scale - sewers, electricity, etc

TWO KINDS OF SECURITY RISKS

Wear over time

a. The application: patches and updates have to be done - included libraries, etc. Also tech debt and introduced complexity.
b. The neighborhood: ...stuff also wears out and breaks, but on a larger scale - sewers, electricity, etc

TWO KINDS OF SECURITY RISKS

Wear over time

a. The application: patches and updates have to be done - included libraries, etc. Also tech debt and introduced complexity.

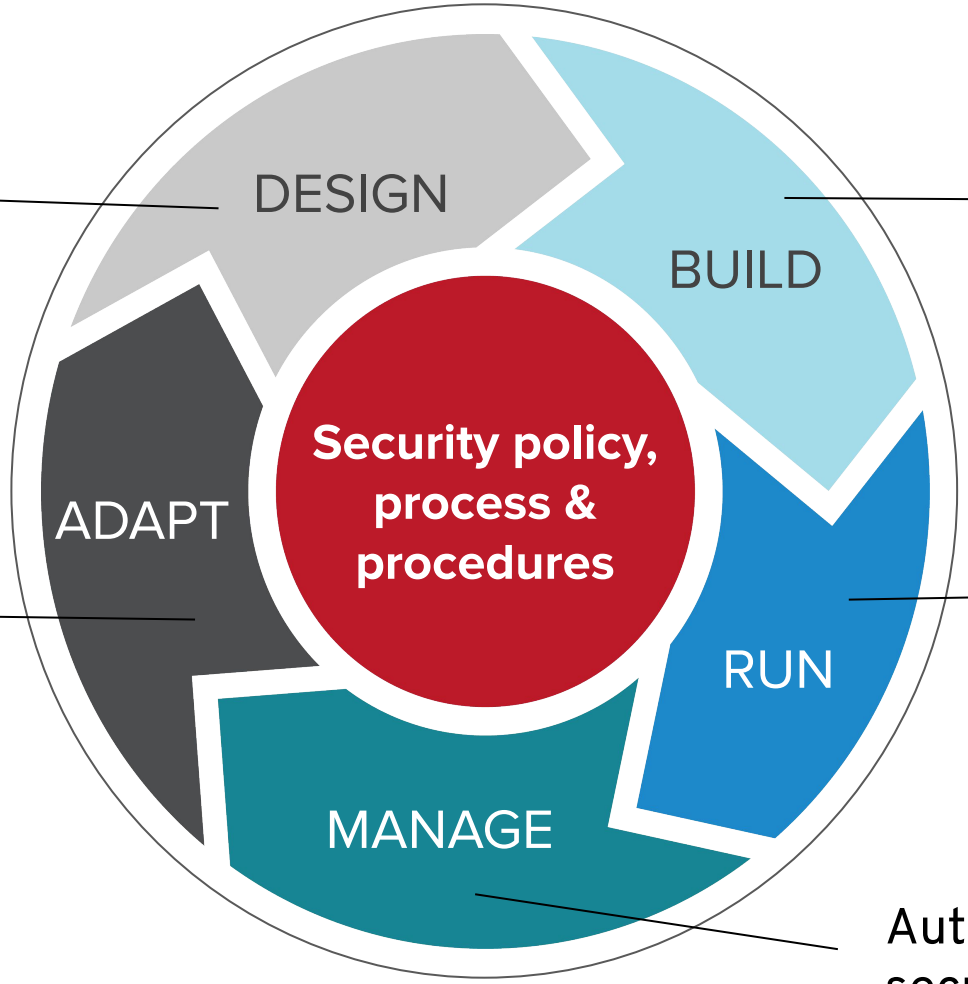
b. The platform: more patches and updates, but across many applications and with potentially higher stakes.

Security as a Continuous Process

SECURITY IS BEST WHEN IT'S A PROCESS

Security is best - most effective *and* easiest to do - when it's a clear process, built into every aspect of platform and application development and care.

Identify security requirements & governance models



Built-in from the start; not bolted-on

Revise, update, remediate as the landscape changes

Deploy to trusted platforms with enhanced security capabilities

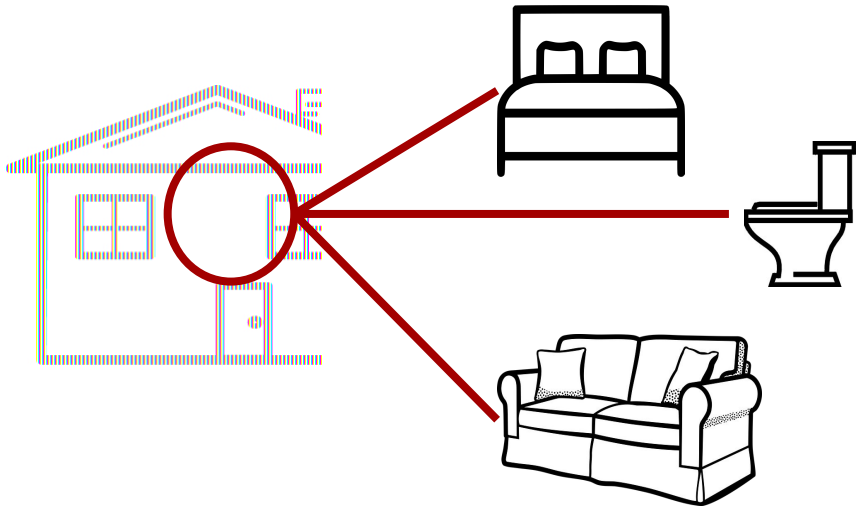
Automate systems for security & compliance

Design:

Identify security requirements and governance models

LOCKED DOORS AND CAREFUL KEYS

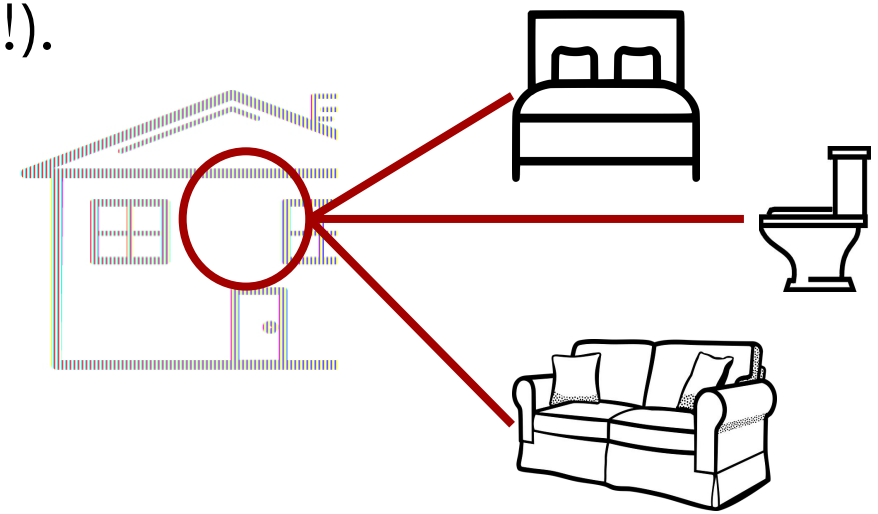
You don't hand out your house keys to just anybody. And some things in your house (valuables safe, gun safe, liquor cabinet, admin account on your PCs, Wi-Fi router), you secure even *more* carefully.



LOCKED DOORS AND CAREFUL KEYS: RBAC

Don't hand out security keys to just anybody!

Use RBAC to restrict who can access your projects, and use encryption for your traffic in flight and at rest. Also, put access controls on your apps and APIs (and IPAs, too!).



A NOTE ABOUT DATA

We thought it worth calling out data security specifically. Data, especially *customer* data, is one of the **most** important things a company needs to try to keep safe - just like you would keep your vital records (social security cards, birth certificates, marriage certificates, etc) in a fireproof safe, in a room, in your house.

Source:
http://www.soul-repairs.com/blog/2019/08/10/what-happened-at-capital-one/
https://www.noobaa.io/private-cloud
https://github.com/gluster/glusterfs-specs/blob/master/done/GlusterFS%203.5/Disk%20Encryption.md

Build:

Security built in from the start, not bolted on

Red Hat

KNOW YOUR MATERIALS

Time to repaint your house? Cool! One can of lead paint, please!

Time to replace the furnace? *Obviously* you'll need one that leaks carbon monoxide...

KNOW YOUR MATERIALS

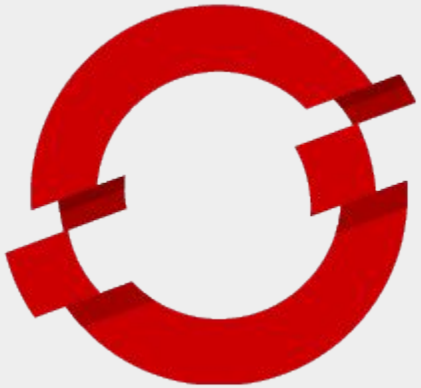
Get your container images from a **trusted** source - Docker Hub, as an example, is rather famously NOT one of these places.

Know who's building the images, and where they're pulling their ingredients from - and stick to places like Maven Central, the Red Hat Container Catalog, or your own internal container registry.

Source:
https://snyk.io/blog/top-ten-most-popular-docker-images-each-contain-at-least-30-vulnerabilities/
https://www.kennasecurity.com/one-fifth-of-the-most-used-docker-containers-have-at-least-one-critical-vulnerability/
https://www.securityweek.com/docker-hub-breach-hits-190000-accounts

Run:

Deploy to trusted platforms, with enhanced security capabilities

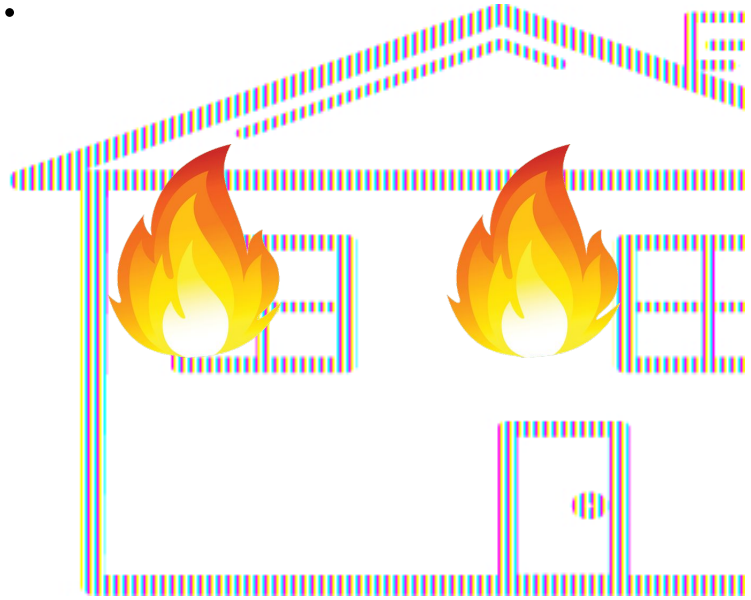


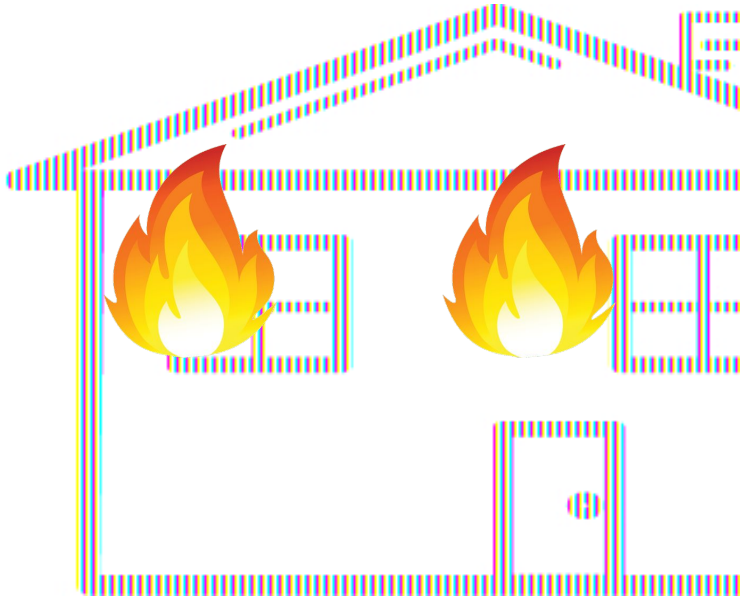
THE NEIGHBORHOOD

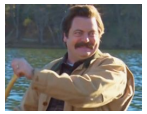
Ideally, people live in houses in roughly family-ish groups, and those houses exist within a neighborhood. Each house doesn't have to BE a neighborhood, and also not *everyone* in the neighborhood has to live in the same house.

THE NEIGHBORHOOD

Everyone in the *same* house sounds...kind of awful. It would have to be a MASSIVE house, and if someone in one room decides that making meth sounds like a *lot of fun* and blows up their room...probably your room will also be affected.







Source:
https://www.youtube.com/watch?v=zGxwbhkDjZM

THE NEIGHBORHOOD

So...separate houses seems legit. But...relying on the neighborhood for common infrastructure (not needing to BE a neighborhood) means that the walls of each house have to be strong enough to keep them safe from *each other*.

TADA MULTITENANCY

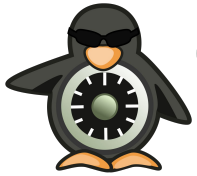
Multitenancy means *multiple groups of users and groups of applications in a shared environment.*

It means less operations setup and maintenance, easier governance and security (because *one* place to secure infrastructure), and the ability to share computing resources - but it *also* means that you need proper isolation.

TADA MULTITENANCY

You don’t want applications stealing resources from each other, and you don’t want apps attacking each other or the platform if compromised.

OpenShift just… *does this*. Quotas, SELinux, and project boundaries act as your house walls, keeping your applications from being used as a starting point to attack the rest of your neighborhood.



OPENSHIFT AND THE *runc* VULNERABILITY

Vulnerability Statement:

"A flaw was found in the way runc handled system file descriptors when running containers. A malicious container could use this flaw to overwrite contents of the runc binary and consequently **run arbitrary commands on the container host system.**"

Mitigation

This vulnerability is **mitigated on Red Hat Enterprise Linux 7 if SELinux is in enforcing mode.** SELinux in enforcing mode is a **pre-requisite for OpenShift** Container Platform 3.x.

Source: https://access.redhat.com/security/cve/cve-2019-5736

THE *runc* VULNERABILITY **WITHOUT** SELINUX

NOT OpenShift's Vulnerability Statement:

"What makes this *troublesome* is that interacting with Linux primitives like namespaces typically requires you to run as root, somewhere. In most installations involving runc (including the default configuration in Docker, Kubernetes, containerd, and CRI-O), the whole setup runs as root.

An alternative to running as root is to leverage a user namespace.

Kernel operations that are user-namespace-aware can delineate privileged actions occurring inside the user namespace from those that occur outside.

However, user namespaces are not yet widely employed and are not enabled by default."

Source: https://aws.amazon.com/blogs/compute/anatomy-of-cve-2019-5736-a-runc-container-escape/

POINTS OF ENTRY

We talked about the necessary vulnerabilities in a neighborhood - doors, windows, a gate into the community, etc.

OpenShift solves this in a couple of different ways...

POINTS OF ENTRY

One is what we just talked about - strong boundaries between a pod (running instance of an application) and other pods, other projects, and the underlying cluster infrastructure.

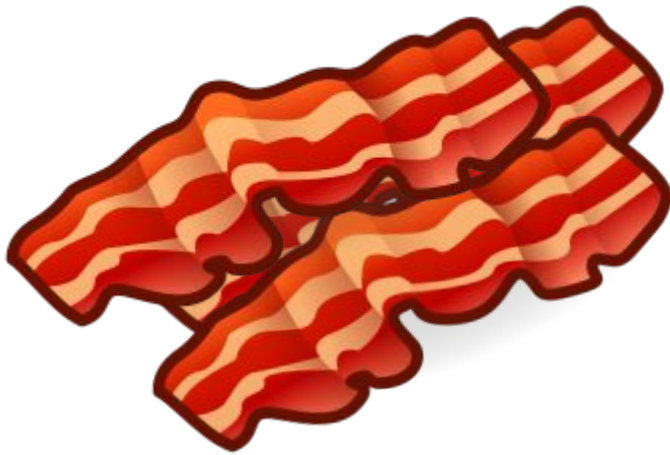
Basically...entry into a room in one house does not guarantee entry into any other room in that house, or any other house in the neighborhood.

POINTS OF ENTRY

Another is supporting **one entry point** into the cluster, or even clusterS, via HAProxy + integrations with enterprise load balancers (like F5).

So...someone can't show up in a house without first going through the gate into the neighborhood.

WHAT IF YOU COULD BUILD YOUR HOUSE ENTIRELY OUT OF STONE?



Source: https://www.kids-pages.com/folders/stories/The_Three_Little_Pigs/page5.htm

Red Hat

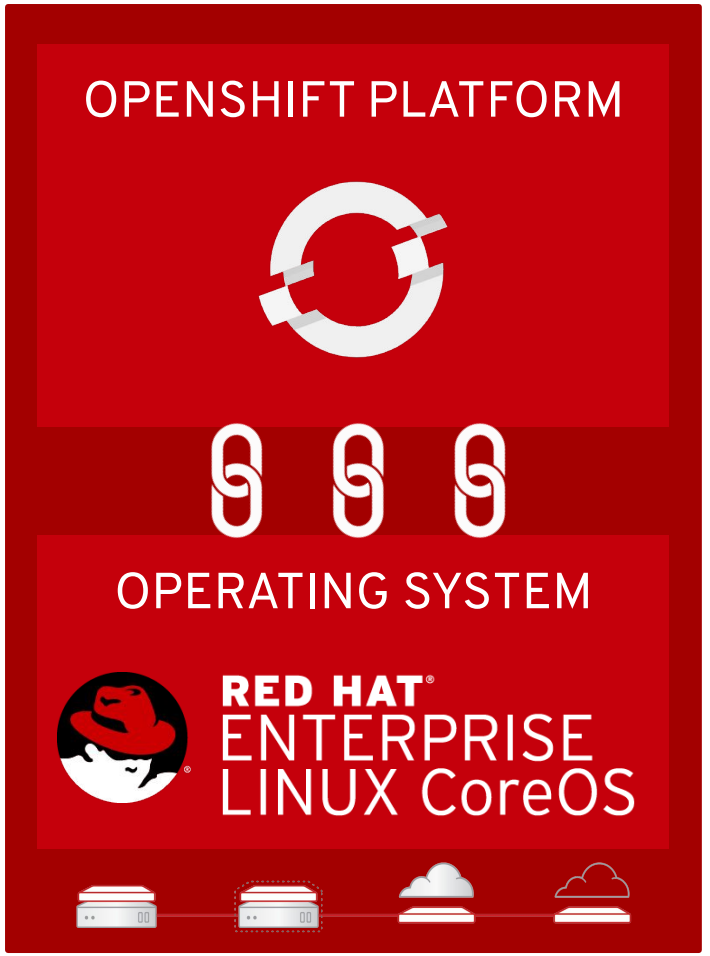
...OR REINFORCED CONCRETE?

Source: https://www.nytimes.com/2018/10/14/us/hurricane-michael-florida-mexico-beach-house.html

SOLID...SOLID AS A ROCK

Preventing changes to the foundation of your systems *completely* would sure be cool...

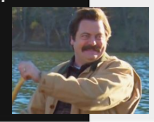
NAILED IT. OpenShift, starting with 4.1, has immutable hosts.



Red Hat

Manage:

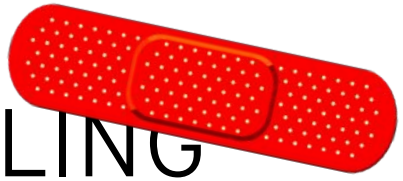
Automate systems for security and compliance



Red Hat

STUFF BREAKS

AUTOMATED HEALING



Wouldn't it be cool if your house notified you when the sump pump broke, and then automatically fixed itself before you had a pond in your basement?

That's basically Kubernetes and Ansible - Red Hat **strongly** believes in the power of automated healing of infrastructure and applications.

AUTOMATED HEALING

They both work by defining what the infrastructure/application format *should* look like, and then correcting things that don't look like that anymore.

This lets you focus on other important things that require human thought, like...*security*.

PATCHING PROCESSES

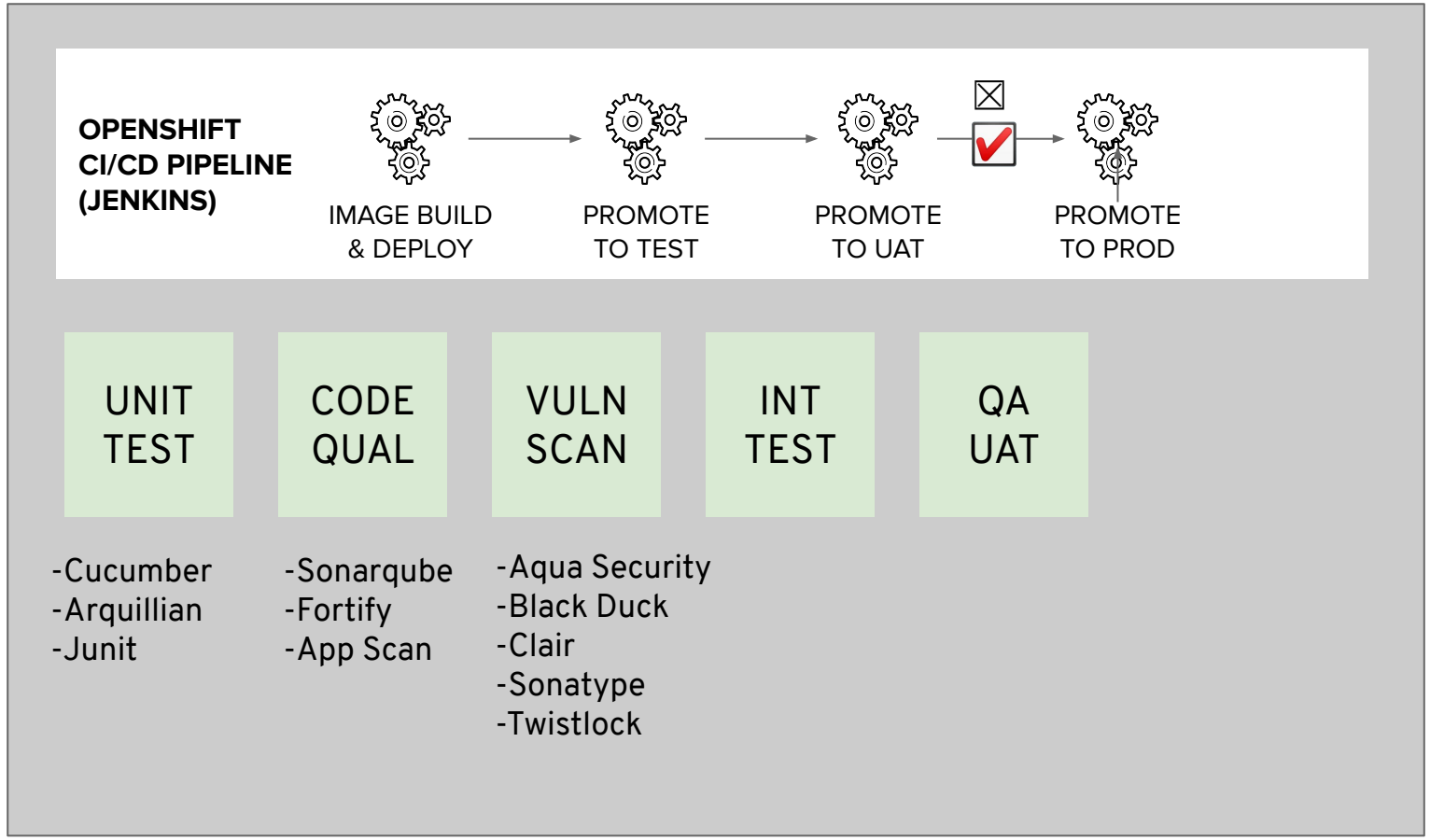
An extension of this is patching - imagine if your house told you when your roof was wearing away and then fixed *that* for you too.

The easier it is to patch infrastructure, the less *anyone* has to worry about it - and the less people can forget about it, or de-prioritize it.

Adapt:

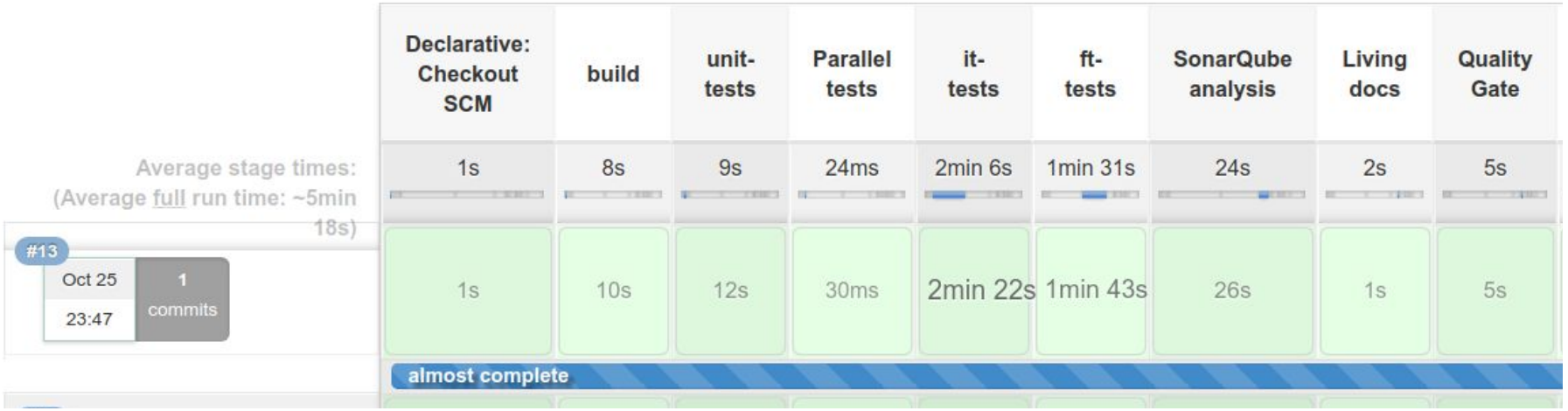
Revise, update, and remediate as the landscape changes

CI/CD WITH SECURITY GATES



CI/CD WITH SECURITY GATES

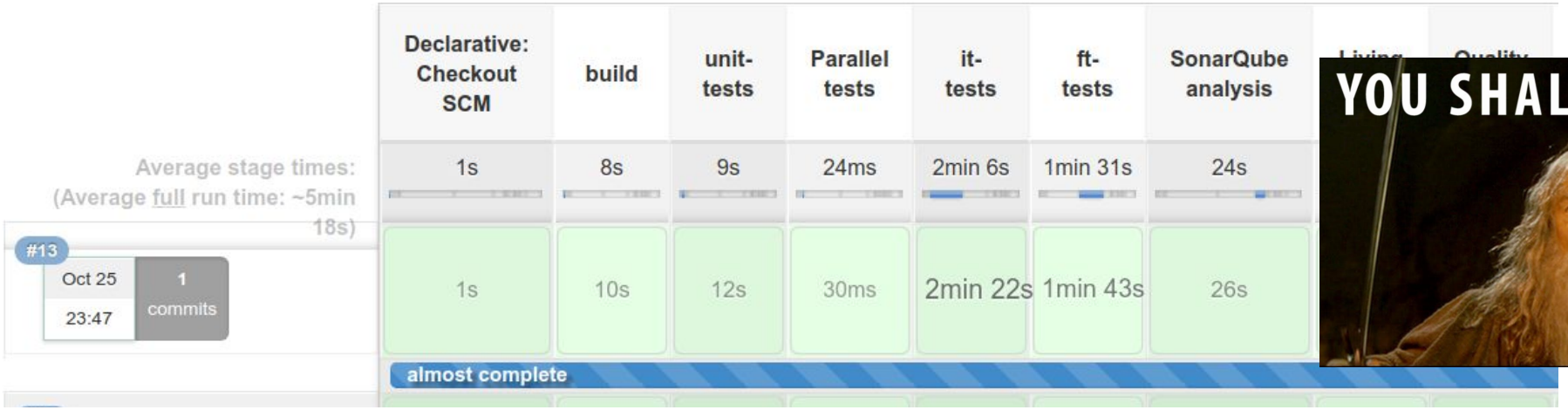
Integrate security testing, code quality assessments, and vulnerability scanning into your build process - these tools are integrated into OpenShift (like Jenkins), can run *on* OpenShift (BlackDuck), or will work *with* OpenShift or the binaries that become running applications (Sonar).



Red Hat

CI/CD WITH SECURITY GATES

Use automated policies to flag builds and containers with security issues - implement a process that clearly explains how quickly to fix artifacts that have security issues, and what level of issue can't go to Production.



Building Alliances -
or, Why Security is Awesome

Red Hat

POSSIBLE RESPONSES TO THREATS

Threats to enterprises via their applications are real, and there are a few likely responses...

POSSIBLE RESPONSES TO THREATS

Ideally, enterprises respond to threats in ways that help everyone see that security is helpful and is **not** the enemy of innovation and delivery speed - in fact, it helps ***enable*** them.

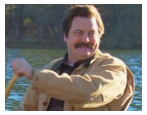
...no, *really*. We totally mean it!

POSSIBLE RESPONSES TO THREATS

1. 

   “We’ll just ignore that there **are** security threats.”

   - Speed in the moment: highest
   - Likelihood of catastrophes: very high 
   - Pain/interruption of daily process: medium, varies with awareness
   - Overall speed: slow



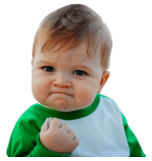
POSSIBLE RESPONSES TO THREATS

2. 👀

"We're afraid of everything - too afraid to make a plan. So...can you go through all of your code, line by line, and report back that there are no security issues (or other illogical, unhelpful things)?"

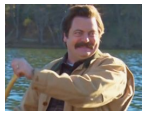
- Speed in the moment: lowest
- Likelihood of catastrophes: highest - false sense of "security"
- Pain/interruption of daily process: highest
- Overall: slowest *and* most painful

POSSIBLE RESPONSES TO THREATS

3. 

"We acknowledge that threats exist, and we've made the best plan we can for proactively addressing them - and we trust that we can adapt that plan if we need to. We just… *do security*, built into our processes."

- Speed in the moment: medium to high, esp with automation
- Likelihood of catastrophes: lowest
- Pain/interruption of daily process: low
- Overall: ***fastest!***



OPENSHIFT IS AN INNOVATION PLATFORM

MODERNIZE APPS

WEB APPS

CLOUD NATIVE DEV

MULTI-CLOUD

MOBILE

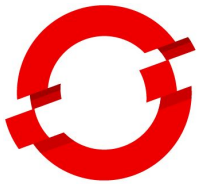
BIG DATA | ANALYTICS

AI | ML

IOT

Red Hat

OPENSHIFT IS AN INNOVATION PLATFORM



Red Hat OpenShift

531%
5-year ROI

66%	36%	8 MONTHS	US$1.29M
Faster development life cycle	More applications per year	Payback period	Average annual benefits per 100 developers

The Business Value of Red Hat OpenShift, IDC #US41845816, October 2017, https://www.redhat.com/en/resources/The-Business-Value-of-Red-Hat-OpenShift.

Red Hat

OPENSHIFT IS AN INNOVATION PLATFORM

CATHAY PACIFIC

10x

Increased application development throughput from 20 to **200 changes a day**

OpenShift on AWS and private cloud

Hilton

Months → Days

Improved time to market by **accelerating development time**

OpenShift on AWS

Schiphol Amsterdam Airport

50%

Reduction in development time for new services and APIs. **Launched a new cloud platform in 10 days**

OpenShift on AWS, Azure, and private cloud

Source:

Cathay Pacific: Red Hat press release, Cathay Pacific Takes Customer Experiences to New Heights with Red Hat's Hybrid Cloud Technologies, May 2018.

Hilton: Red Hat case study, Hilton enhances digital guest experience with Red Hat container and automation technology, October 2018.

Schiphol: Red Hat case study, Amsterdam Airport Schiphol builds agile cloud with Red Hat, August 2017.

Red Hat

...AND OPENSHIFT IS A **SECURE** PLATFORM

It's designed to either handle for you OR enable *you* to handle a lot of the security issues that are inherent to enterprise application development.

- RBAC
- Trusted container base images
- Application and project isolation
- Network segmentation
- Immutable hosts
- Automated healing and patching
- Build and deploy pipelines
- *More, but we totally ran out of time...*

FINALLY...

Security is not the *enemy* of speed - security *enables* speed.

- Security is best when it's a process.
- Build security in from the start, don't bolt it on as an afterthought.
- Pick a platform that's secure by default (so you don't have to think about it as much!)
- Use automation.
- Be willing to adapt.

FINALLY...

Security is super important. So...be deliberate about it, be clear, and ask us to *help* you be clear, about how OpenShift helps.

Q & A

Ask us for our slides!

Joshua.Smith@redhat.com - @architect_josh

Laine.Vyvyan@redhat.com - @lainie_ftw

Thank you!

Red Hat is the world's leading provider of enterprise open source software solutions. Award-winning support, training, and consulting services make Red Hat a trusted adviser to the Fortune 500.


linkedin.com/company/red-hat


youtube.com/user/RedHatVideos

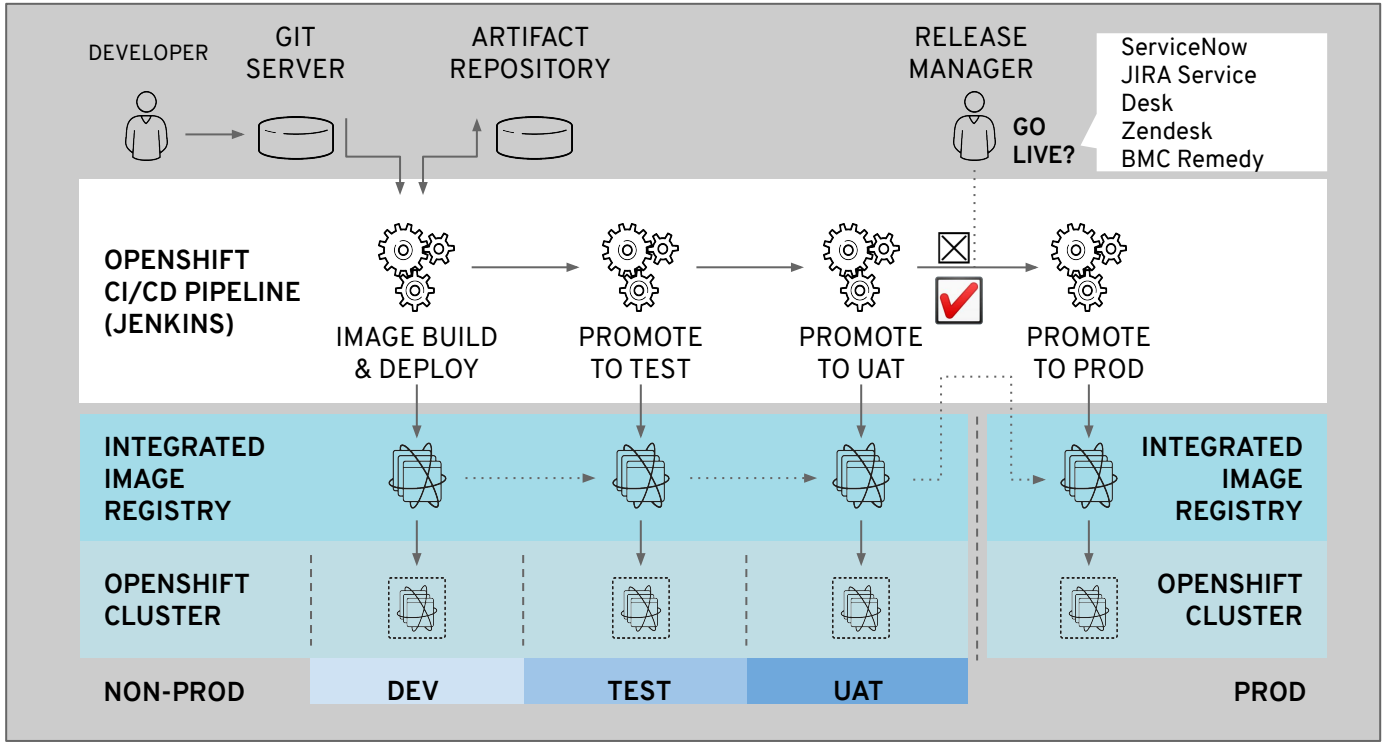

facebook.com/redhatinc


twitter.com/RedHat

Red Hat

MANAGING CONTAINER DEPLOYMENT

- Deployments: Containerized App Configuration as Code
- Whitelist / Blacklist external repos
- Apply runtime security policies
- Validate image signatures
- Monitor for new vulnerabilities
- Trust is temporal:
  rebuild & redeploy as needed



Red Hat

WHERE YOU AT?

Address? Phone number (for texting, not calling, of course...)?

Email address? Discord ID?

How do you find people??

API GATEWAY!

Information about location, contact protocols, etc.

Also...what if you could password-protect people sending you spam, or junk mail? Or robo-dialers calling you?

THAT'S THE VALUE OF API ACCESS TOKENS.

IMMUTABLE OPERATING SYSTEM

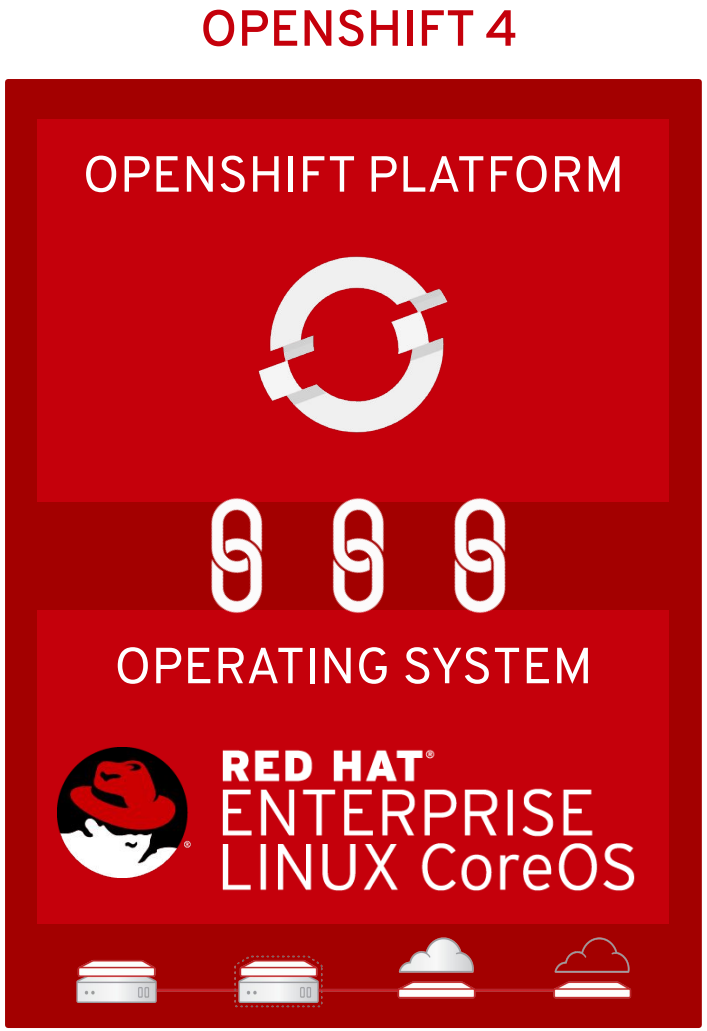
Red Hat Enterprise Linux CoreOS is versioned with OpenShift

CoreOS is tested and shipped in conjunction with the platform. Red Hat runs thousands of tests against these configurations.

Red Hat Enterprise Linux CoreOS is managed by the cluster

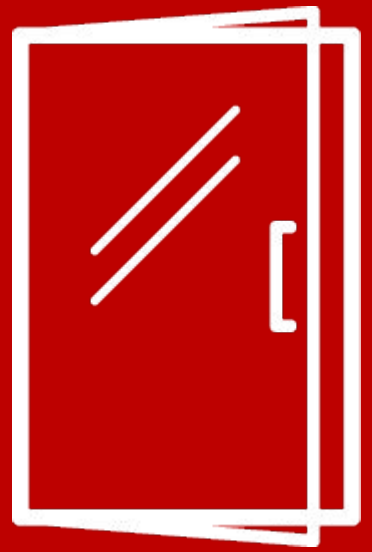
The Operating system is operated as part of the cluster, with the config for components managed by Machine Config Operator:

- CRI-O config
- Kubelet config
- Authorized registries
- SSH config

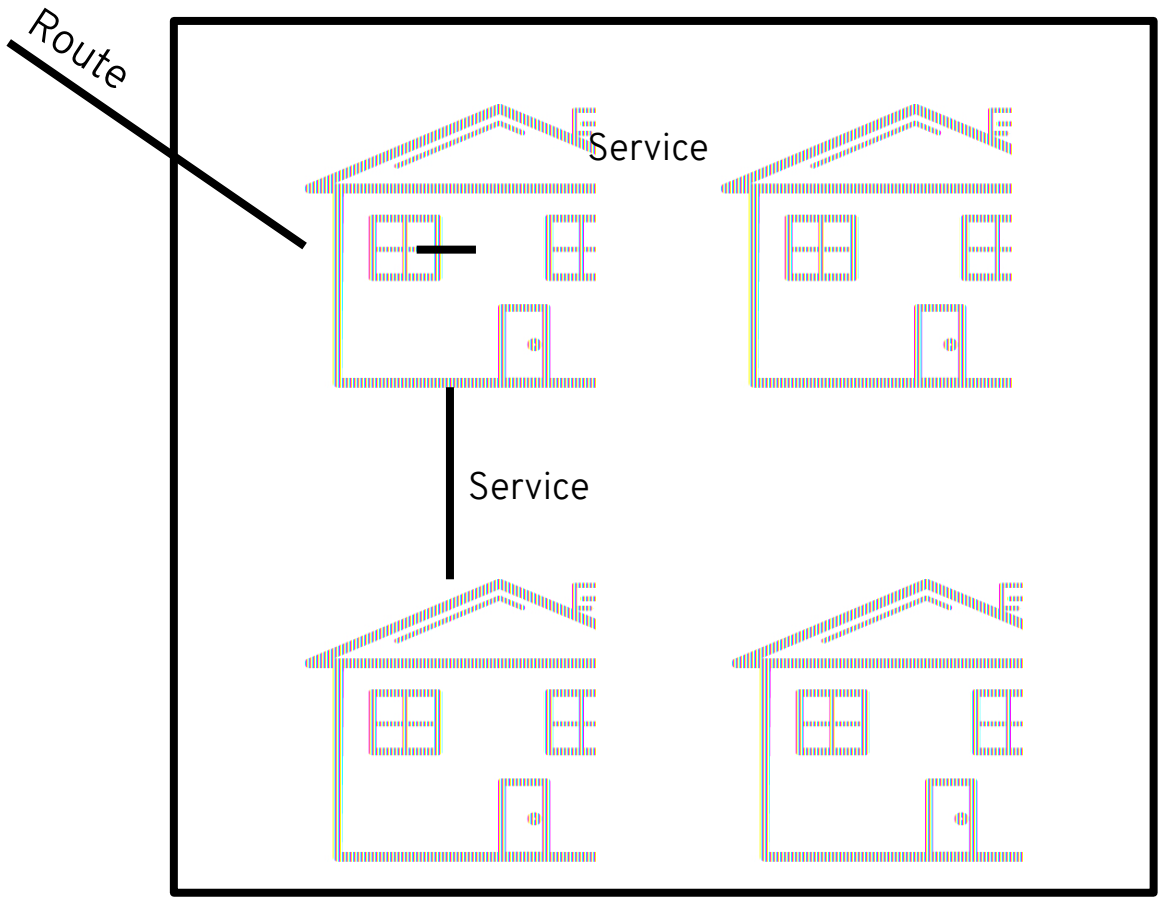


Red Hat

Attack Vectors and Protections



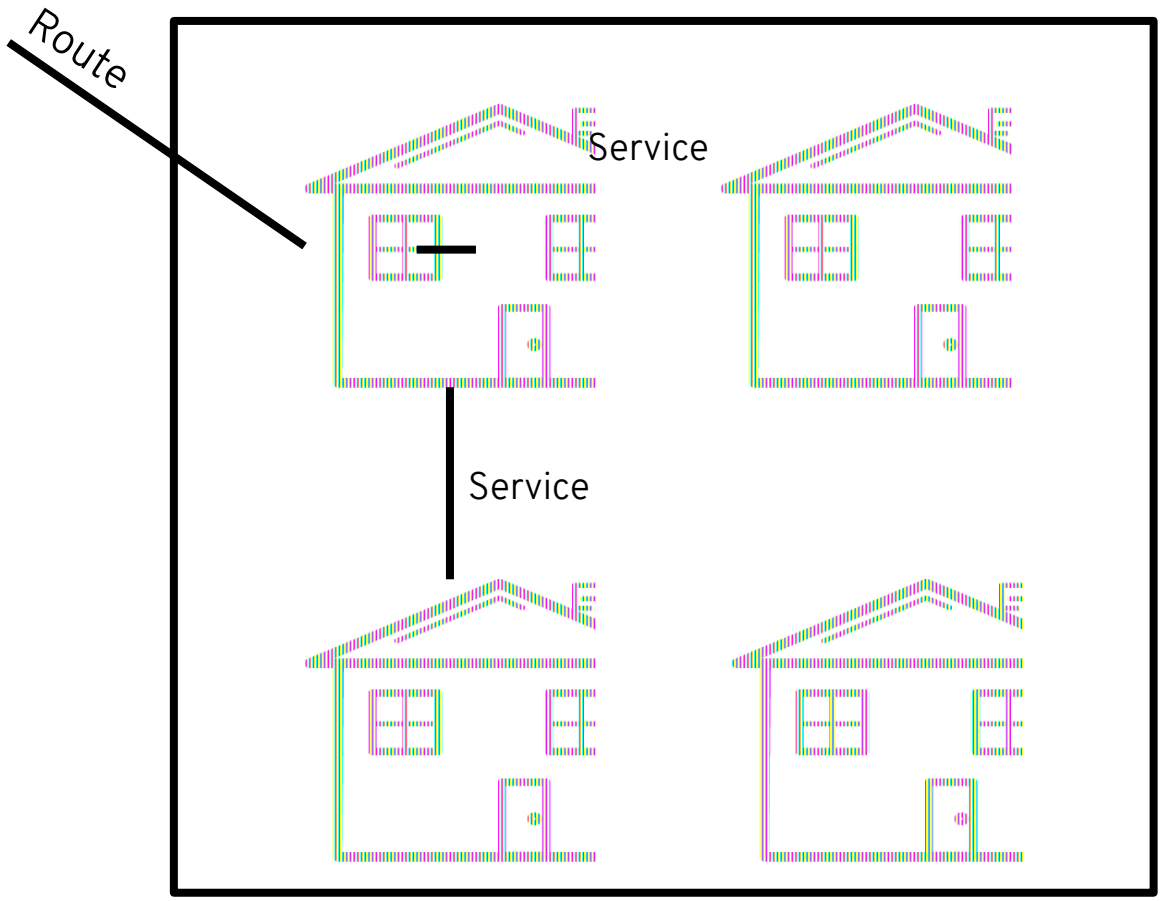
POINTS OF ENTRY



(house to house/app to app)

Red Hat

POINTS OF ENTRY



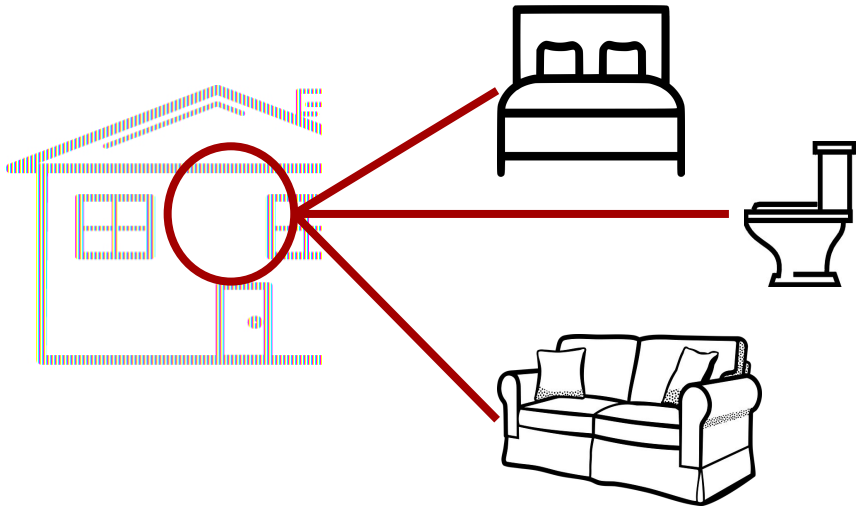
(Project to project)

But we still want to have people over for dinner (invite them inside) sometimes!

DESIGN EXAMPLE: BODA

THIS ISN'T WHAT I ORDERED...

Imagine you're doing some construction on a room in your house.



Red Hat

Secure House/Container Build Process

...continuing the last slide - use a repeatable, trusted process for building containers - and get a house builder you trust.

You *can* rebuild between environments, which is the house equivalent of walking through and inspecting one house and then moving into a separate house that you later find out the plugs aren't wired and the toilet drains into the basement...

Red Hat

IDENTITY AND ACCESS MANAGEMENT

OpenShift includes an OAuth server, which does three things:

- Identifies the person requesting a token, using a configured identity provider
- Determines a mapping from that identity to an OpenShift user
- Issues an OAuth access token which authenticates that user to the API

Managing Users and Groups in OpenShift
Configuring Identity Providers

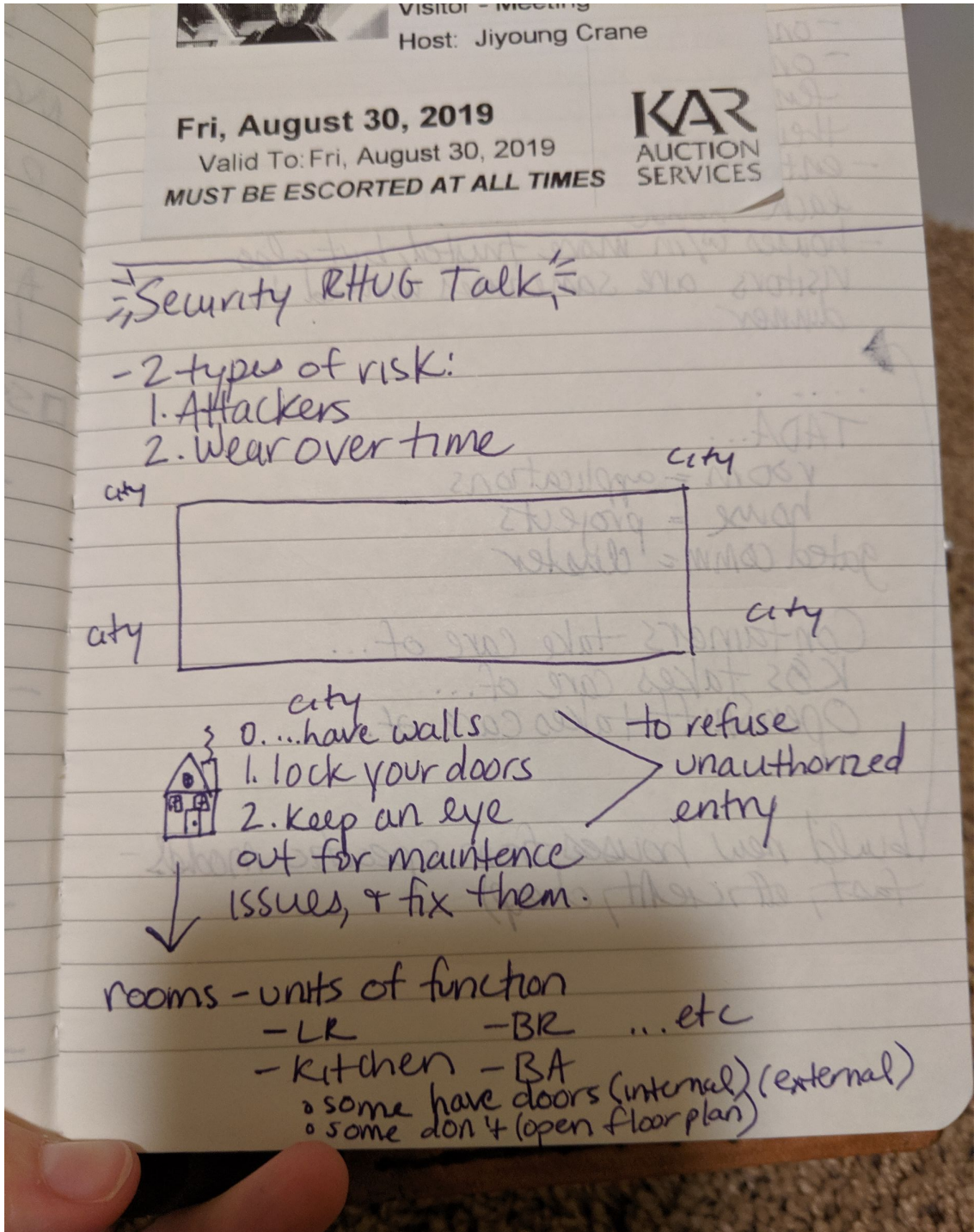
Supported Identity Providers include

- Keystone
- LDAP
- GitHub
- GitLab
- GitHub Enterprise (new with 3.11)
- Google
- OpenID Connect
- Security Support Provider Interface (SSPI) to support SSO flows on Windows (Kerberos)

Multitenancy: the Neighborhood

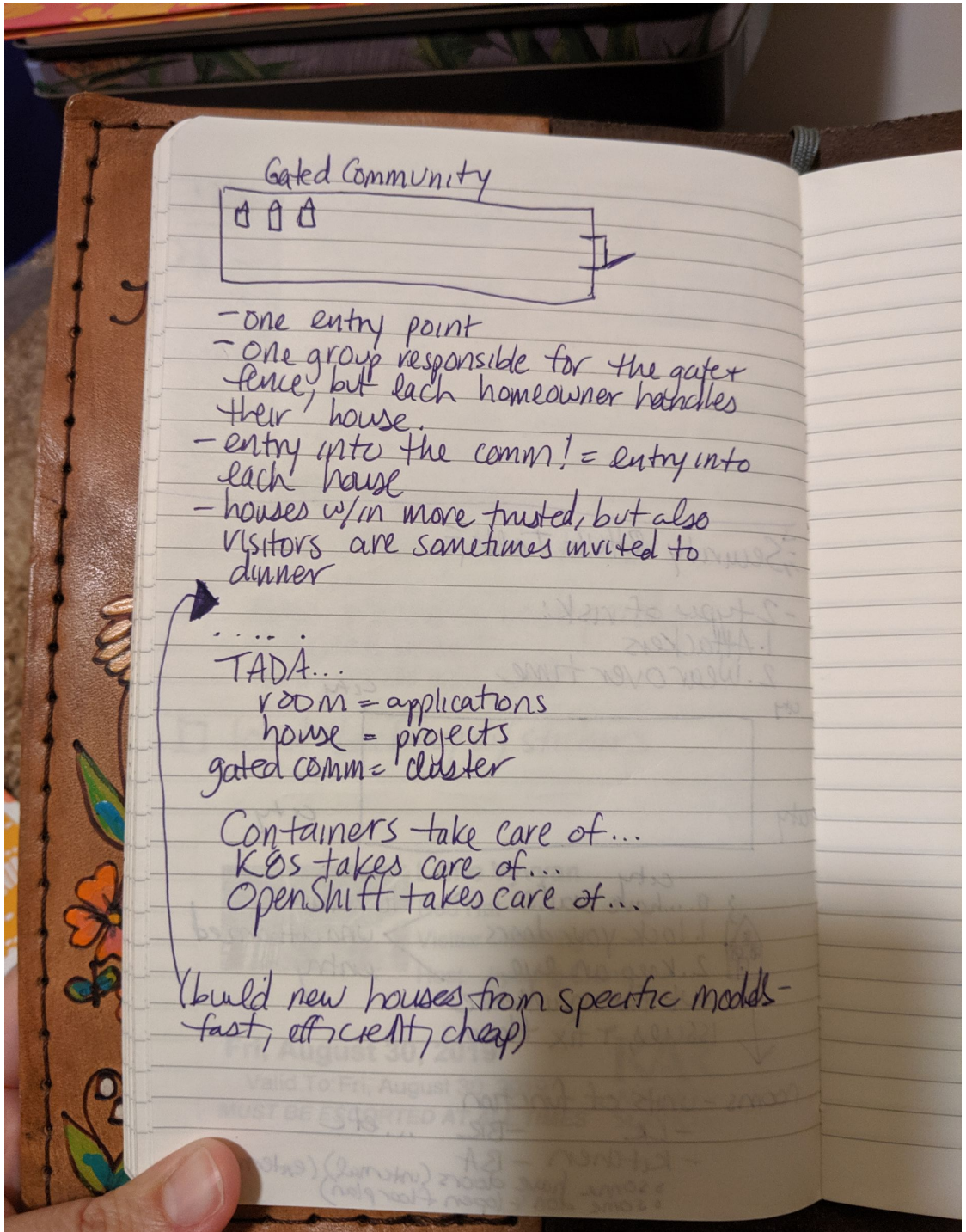
Platform Multitenancy - multiple teams and groups of
Host Multitenancy - communities are better but need good fences and thick walls.

AGENDA



Red Hat

AGENDA



Red Hat

Key Points

Host Multitenancy
Secure Individual Container Content
Secure Container Access
Secure Build & Test Process: reproducible, and deploy-what-you-test
Secure User Access
Secure Network Traffic: Network Segmentation and Encryption
Secure Storage: Stateless/immutable hosts and encryption
Secure API access

Source:

Ten Layers of Container Security

Key Points

Host Multitenancy - communities are better but need good fences and thick walls.
Platform Multitenancy
Secure Individual Container Content - know what you let in your house
Secure Container Access - ??
Secure Build & Test Process: reproducible, and deploy-what-you-test
Secure User Access - RBAC - Walls and interior doors
Secure Network Traffic: Network Segmentation and Encryption - encrypted phone calls and not a party line
Secure Storage: Stateless/immutable hosts and encryption -
Secure API access

Source:

Ten Layers of Container Security

cri-o

A lightweight, OCI-compliant container runtime

Optimized for Kubernetes

Any OCI-compliant container from any OCI registry (including docker)

Improve Security and Performance at scale

[CRI - the Container Runtime Interface]
[OpenShift 4 defaults to CRI-O]
[Red Hat contributes CRI-O to the Cloud Native Computing Foundation]

RUNTIME SECURITY POLICIES

SCC (Security Context Constraints)

Allow administrators to control permissions for pods

Restricted SCC is granted to all users

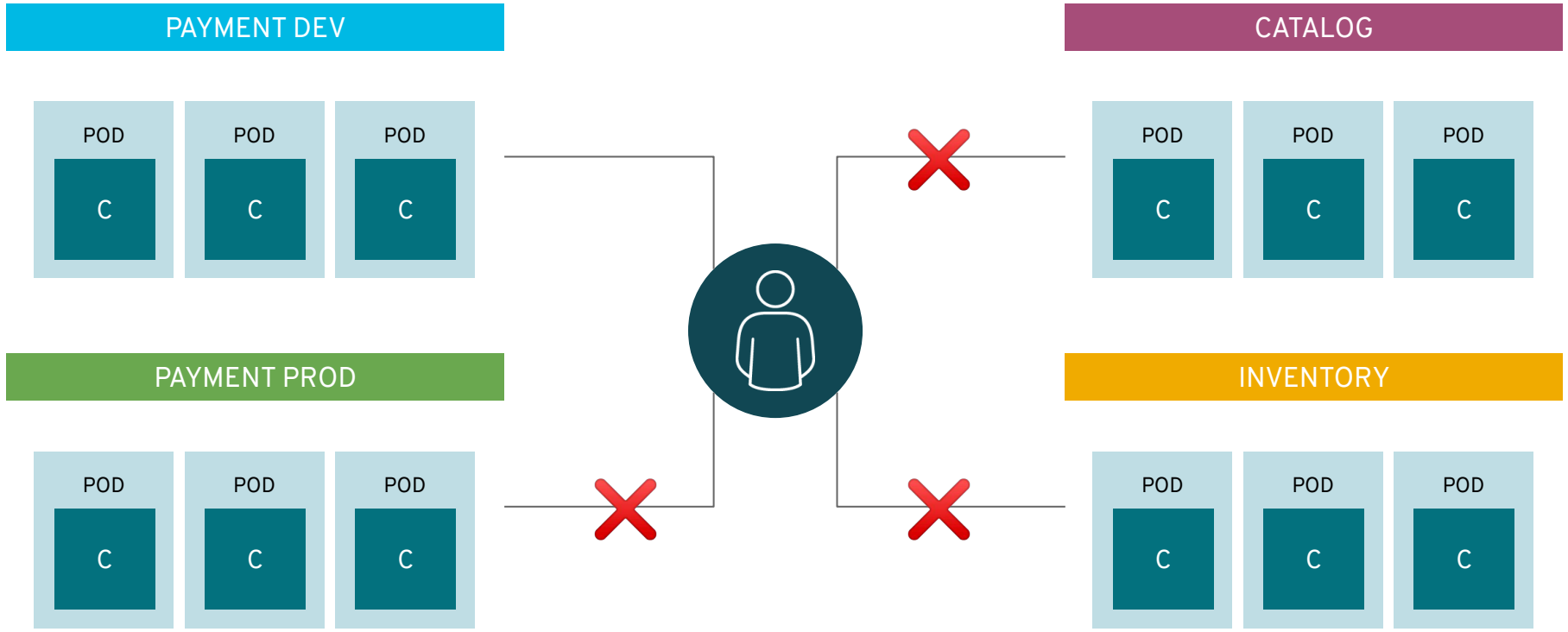
By default, no containers can run as root

Admin can grant access to privileged SCC

Custom SCCs can be created

```
$ oc describe scc restricted
Name:                                   restricted
Priority:                               <none>
Access:
  Users:                                <none>
  Groups:                               system:authenticated
Settings:
  Allow Privileged:                     false
  Default Add Capabilities:             <none>
  Required Drop Capabilities:           KILL,MKNOD,SYS_CHROOT,SETUID,SETGID
  Allowed Capabilities:                 <none>
  Allowed Seccomp Profiles:             <none>
  Allowed Volume Types:                 configMap,downwardAPI,emptyDir,persistentVolumeClaim,projected,
  Allow Host Network:                   false
  Allow Host Ports:                     false
  Allow Host PID:                       false
  Allow Host IPC:                       false
  Read Only Root Filesystem:            false
  Run As User Strategy: MustRunAsRange
```

PROJECTS ISOLATE APPLICATIONS
across teams, groups and departments



Red Hat

RESTRICT ACCESS BY NEED TO KNOW

Role based authorization

- Project scope & cluster scope available
- Matches request attributes (verb,object,etc)
- If no roles match, request is denied (deny by default)
- Operator- and user-level roles are defined by default
- Custom roles are supported

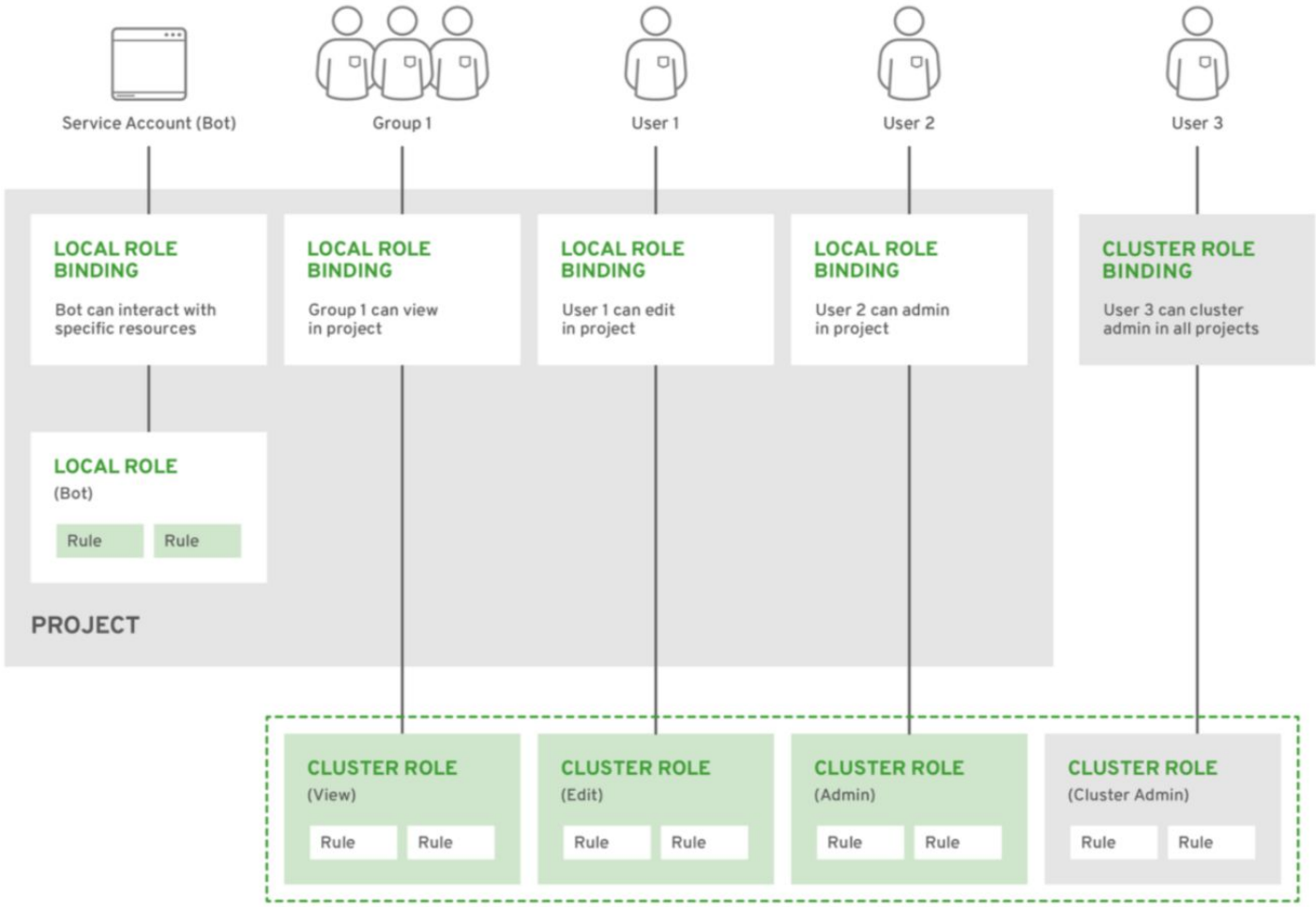
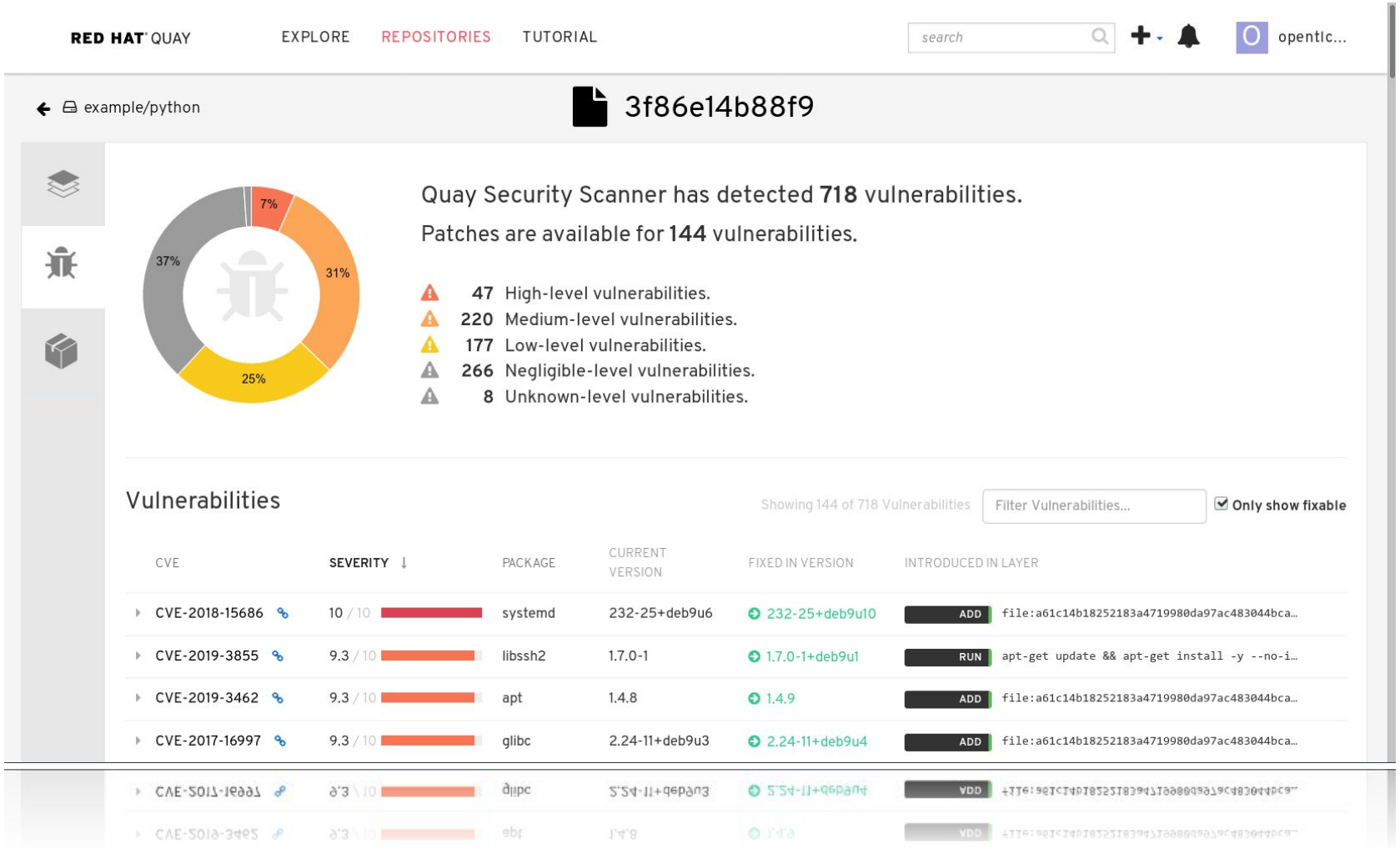


Figure 12 - Authorization Relationships

For more information see: [Managing RBAC in OpenShift]

RED HAT QUAY
ENTERPRISE CONTAINER REGISTRY

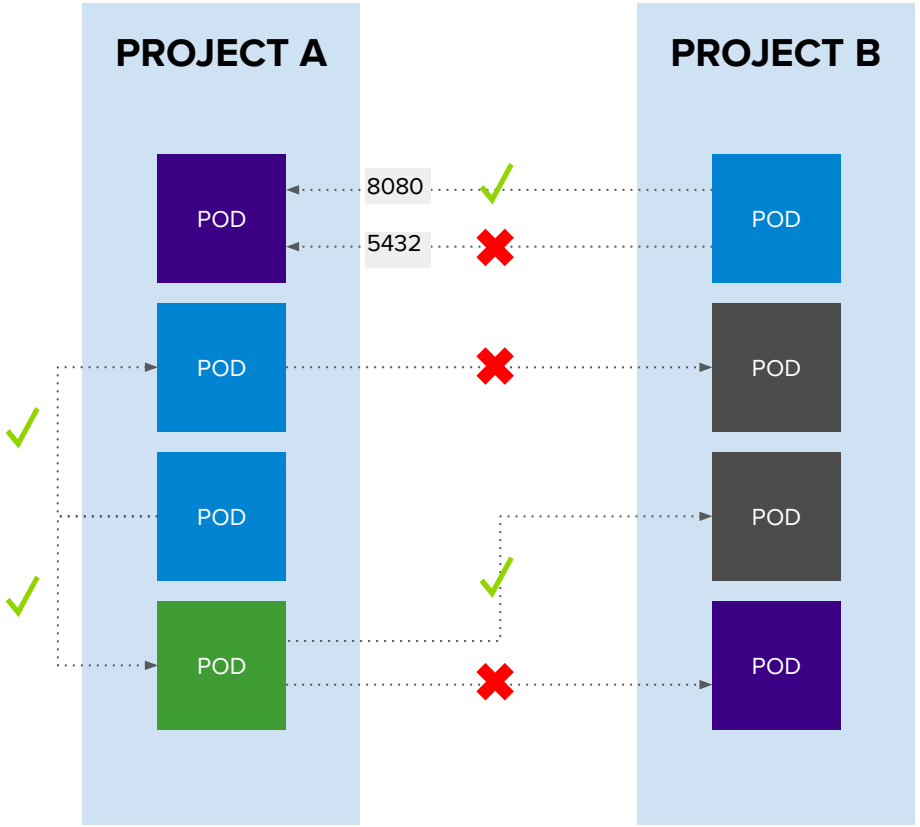
- Offered as self-managed and as-a-service
- Vulnerability Scanning (Clair)
- Geographic Replication
- Build Image Triggers
- Image Rollback with Time Machine



Red Hat

OPENSHIFT SDN

Network Policy enabled by default in OpenShift 4



Example Policies

- Allow all traffic inside the project
- Allow traffic from green to gray
- Allow traffic to purple on 8080

```
apiVersion: extensions/v1beta1
kind: NetworkPolicy
metadata:
  name: allow-to-purple-on-8080
spec:
  podSelector:
    matchLabels:
      color: purple
  ingress:
  - ports:
    - protocol: tcp
      port: 8080
```


Red Hat

Secure User Access: RBAC, walls, and interior doors

(is this duplicate of previous content?)