RSAConference2019
San Francisco | March 4 – 8 | Moscone Center

SESSION ID: CSV-R02

Kubernetes Runtime Security

Jen Tong
Security Advocate, Google

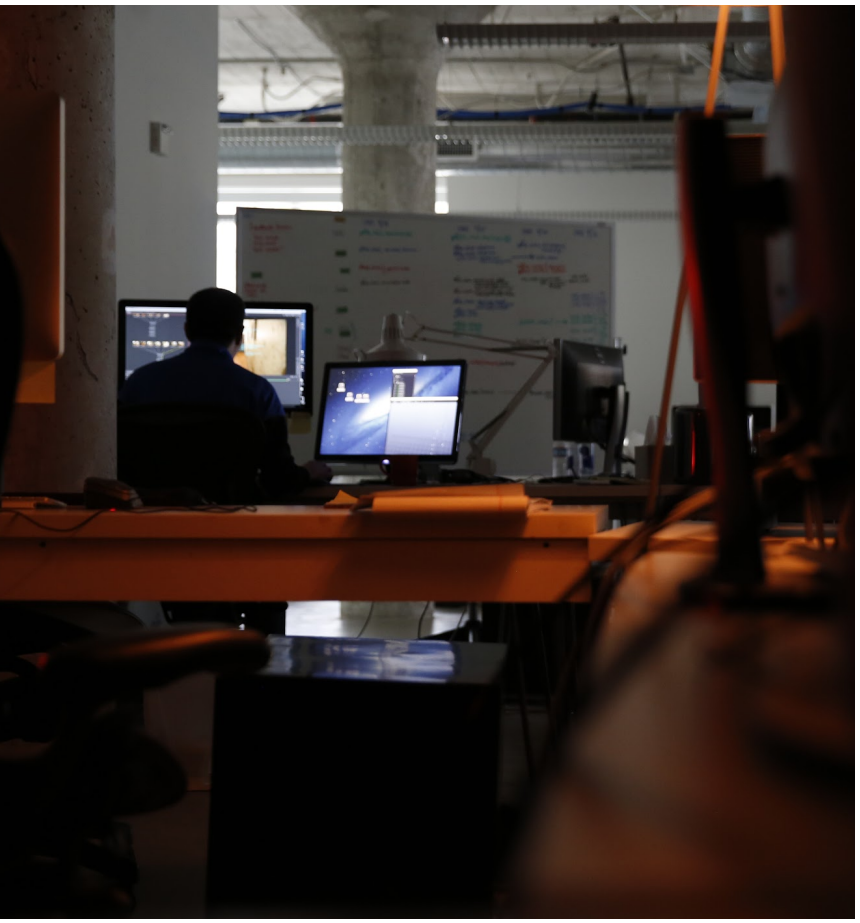
About me

Jen Tong
Security Advocate
Google Cloud Platform

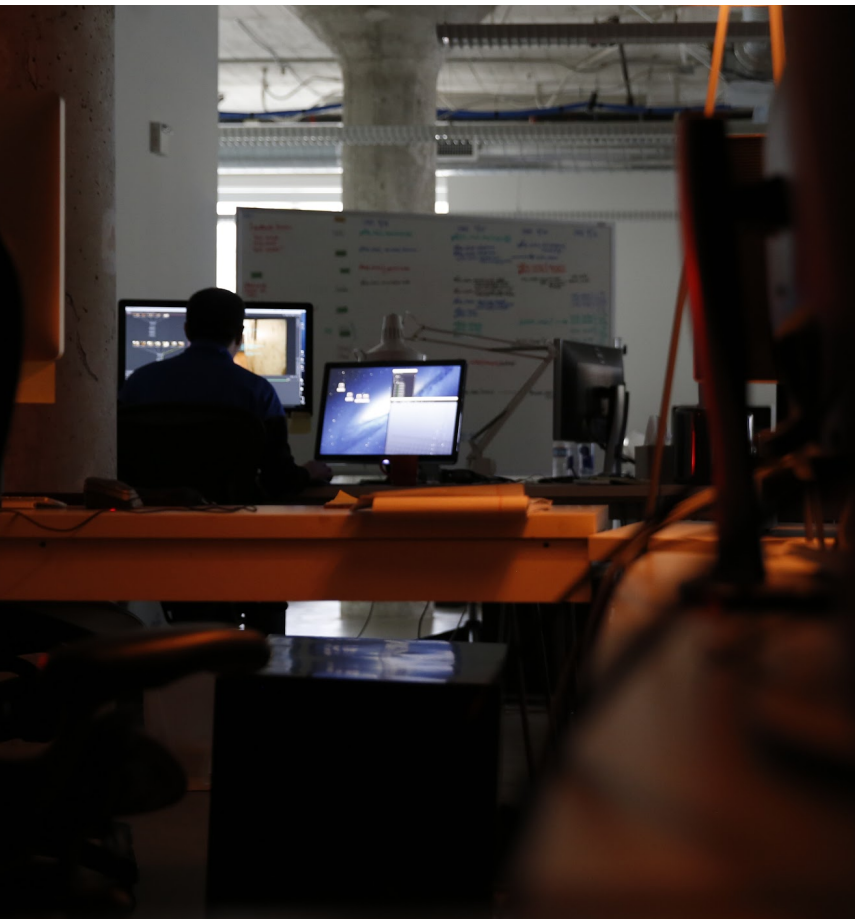
@MimmingCodes

mimming.com

How many of you…

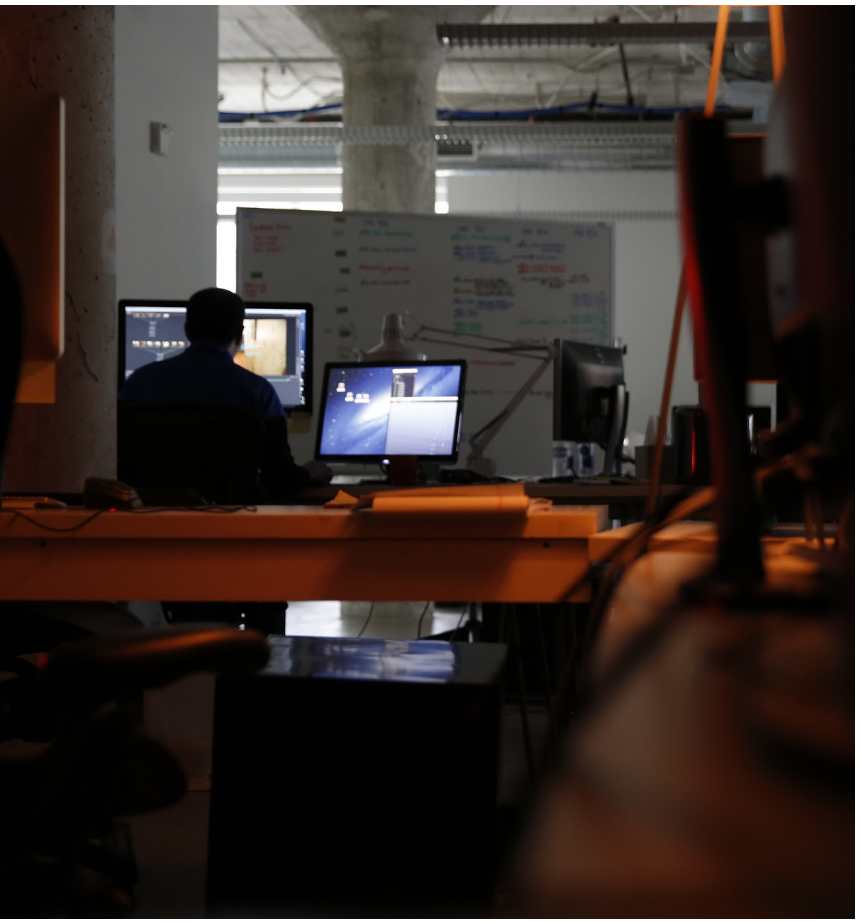


How many of you…



...are familiar with the NIST cybersecurity framework?

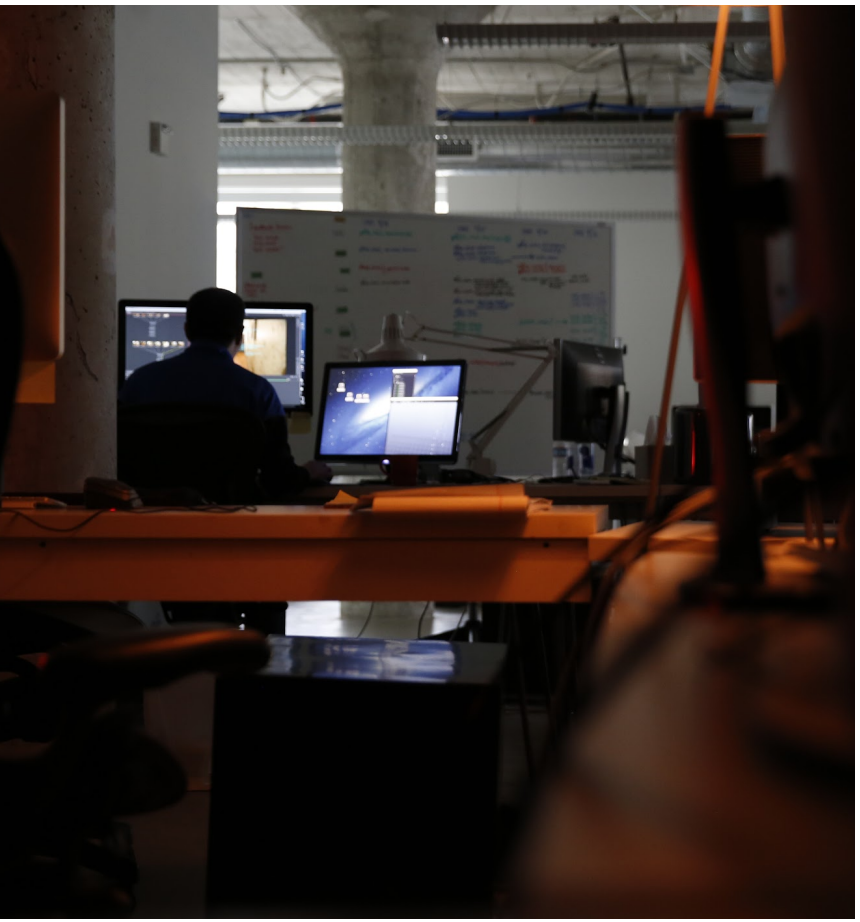
How many of you…



...are familiar with the NIST cybersecurity framework?

...run containers in production?

How many of you…



...are familiar with the NIST cybersecurity framework?

...run containers in production?

...monitor containers for security issues?

Agenda

Container security overview

Containers differ from VMs

How to detect bad things at runtime

Demo

RSAConference2019

Container security overview

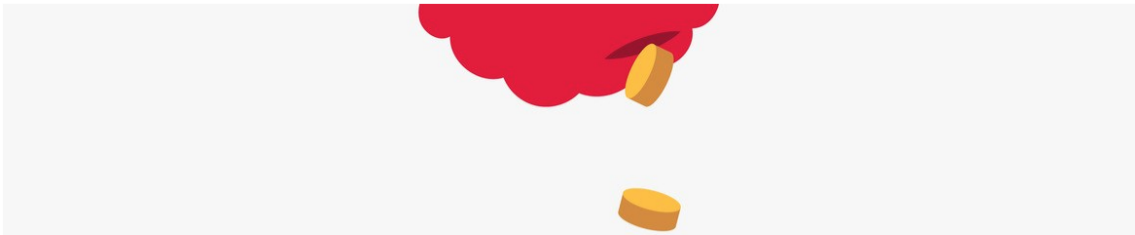
Kubernetes is so new that lots of practitioners don't know what security controls come with it.

> So one of the first things to do is study up on what controls are there and use them to strengthen your security posture
>
> *– Chenxi Wang, Jane Bond Project*

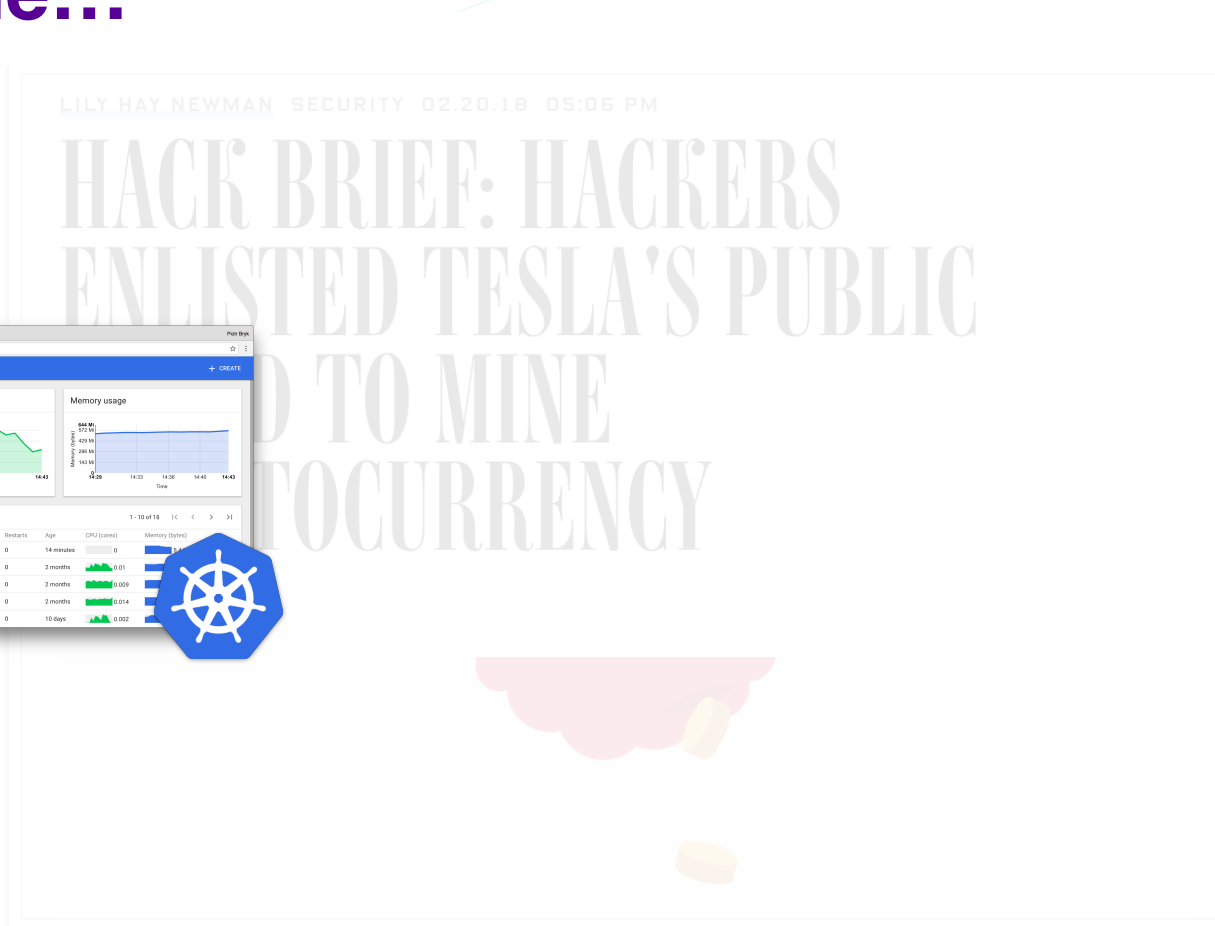
Story time…

LILY HAY NEWMAN SECURITY 02.20.18 05:06 PM

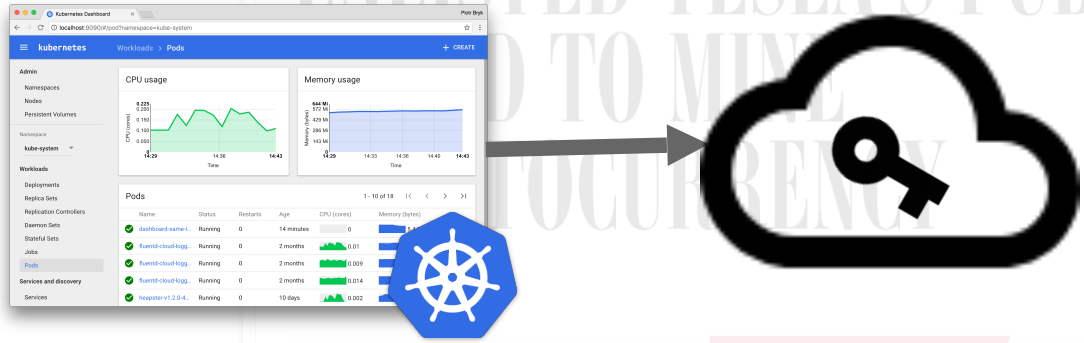
HACK BRIEF: HACKERS ENLISTED TESLA'S PUBLIC CLOUD TO MINE CRYPTOCURRENCY



Story time…


LILY HAY NEWMAN SECURITY 02.20.18 05:06 PM
HACK BRIEF: HACKERS
ENLISTED TESLA'S PUBLIC
D TO MINE
TOCURRENCY
kubernetes
Workloads › Pods
CREATE
CPU usage
Memory usage
Pods

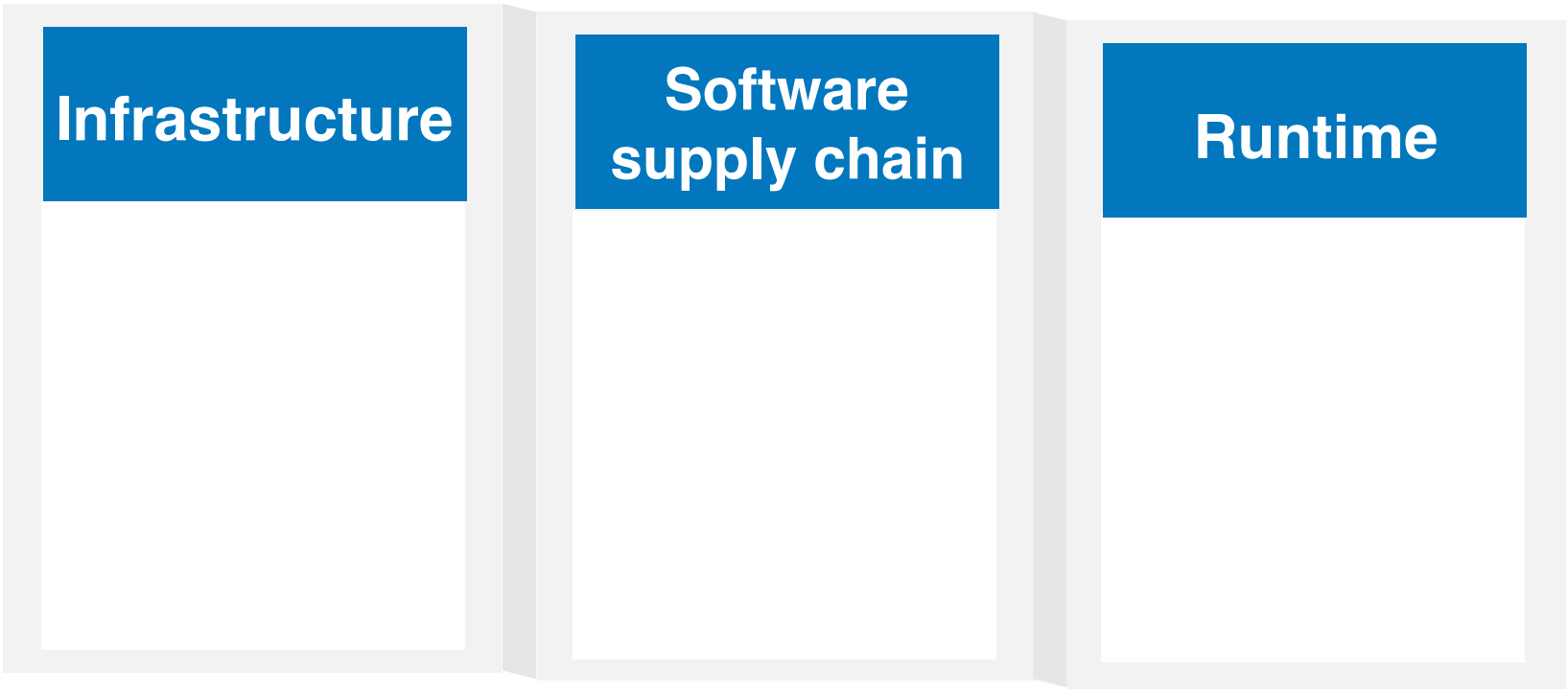
Story time…


kubernetes
Workloads › Pods
CREATE
CPU usage
Memory usage
Pods

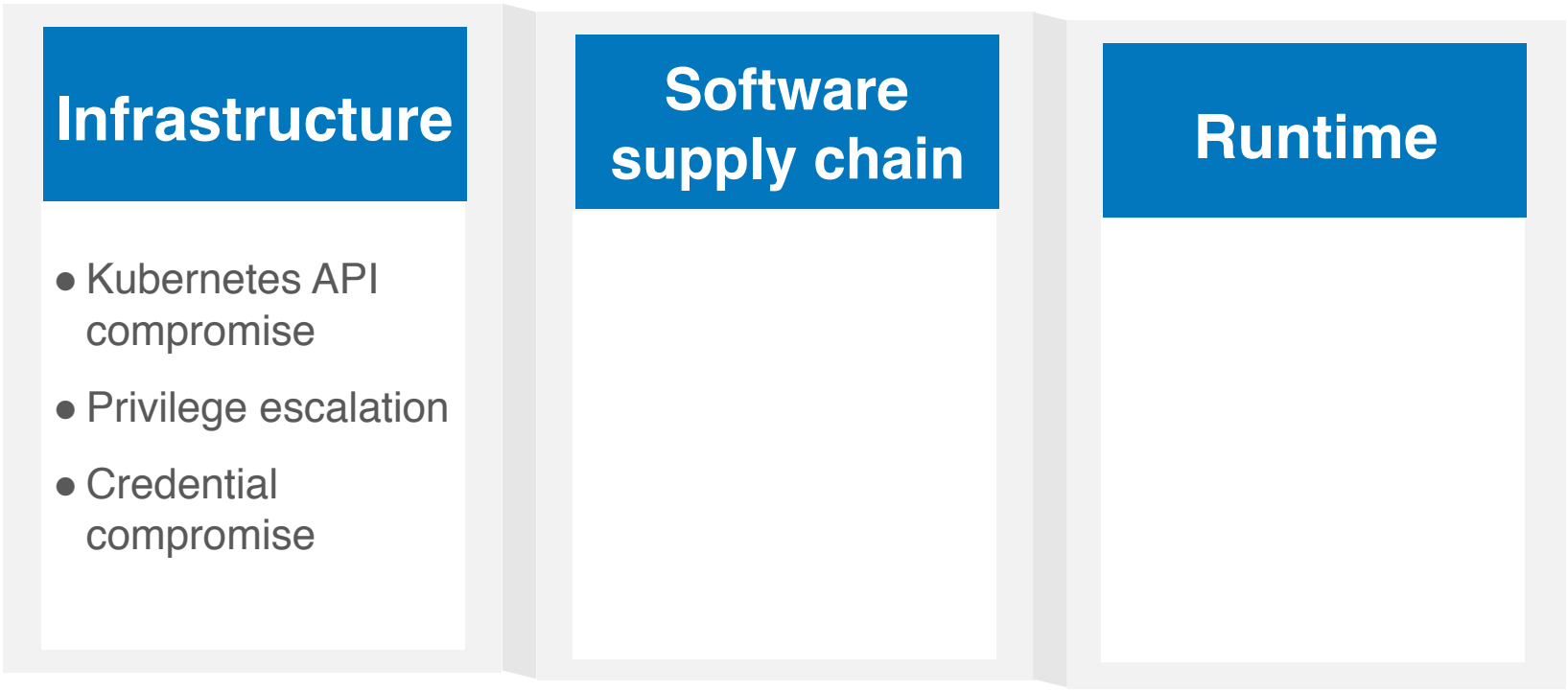
Story time…


kubernetes
Workloads › Pods
CREATE
CPU usage
Memory usage
Pods
$

Threat grouping



Threat grouping



Threat grouping

Infrastructure

- Kubernetes API compromise
- Privilege escalation
- Credential compromise

Software supply chain

- Unpatched vulnerability
- Supply chain vulnerability

Runtime

Threat grouping

Infrastructure

- Kubernetes API compromise
- Privilege escalation
- Credential compromise

Software supply chain

- Unpatched vulnerability
- Supply chain vulnerability

Runtime

- DDoS
- Node compromise
- Container escape
- Zero day

Threat grouping

Infrastructure

- Kubernetes API compromise
- Privilege escalation
- Credential compromise

Software supply chain

- Unpatched vulnerability
- Supply chain vulnerability

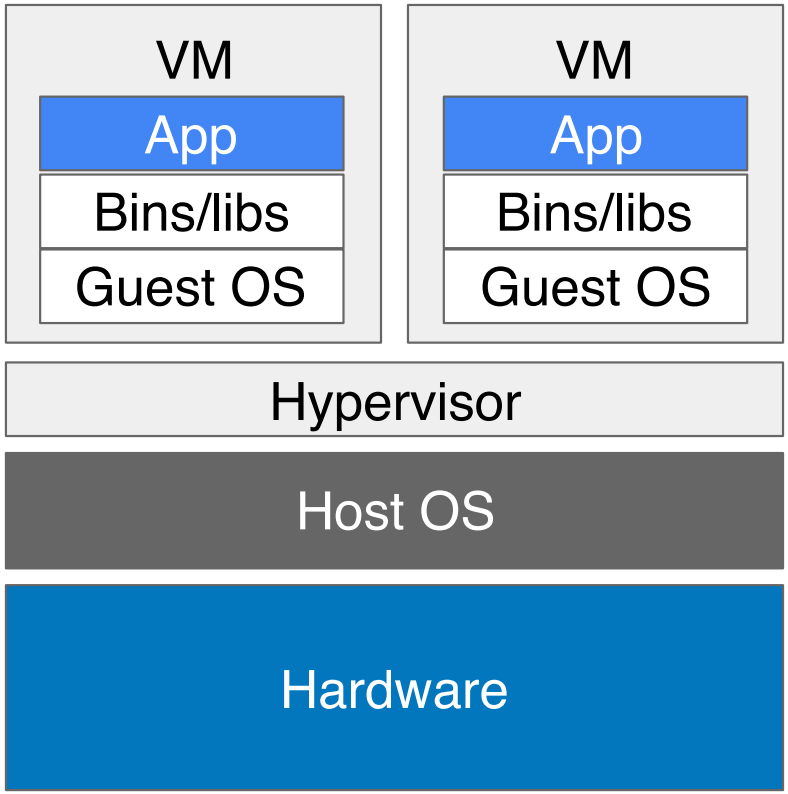
Runtime

- DDoS
- Node compromise
- Container escape
- Zero day

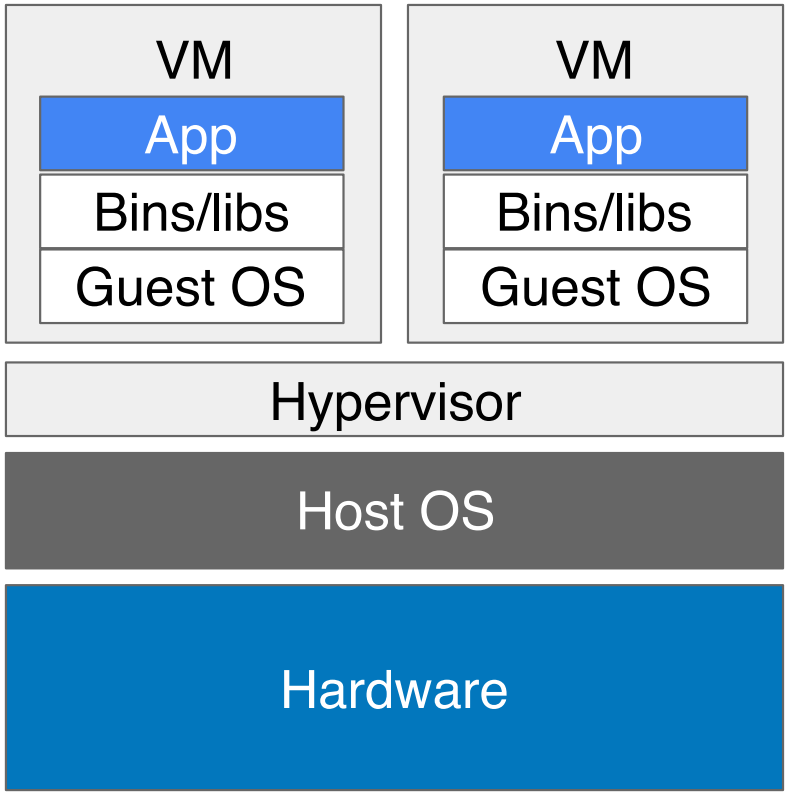
RSAConference2019

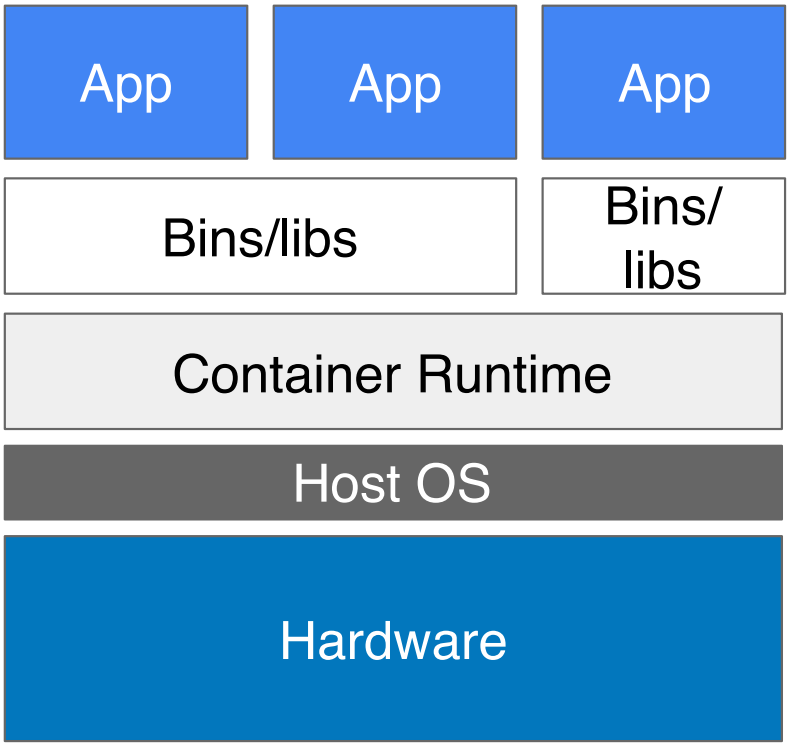
VMs vs Containers

Virtual machine

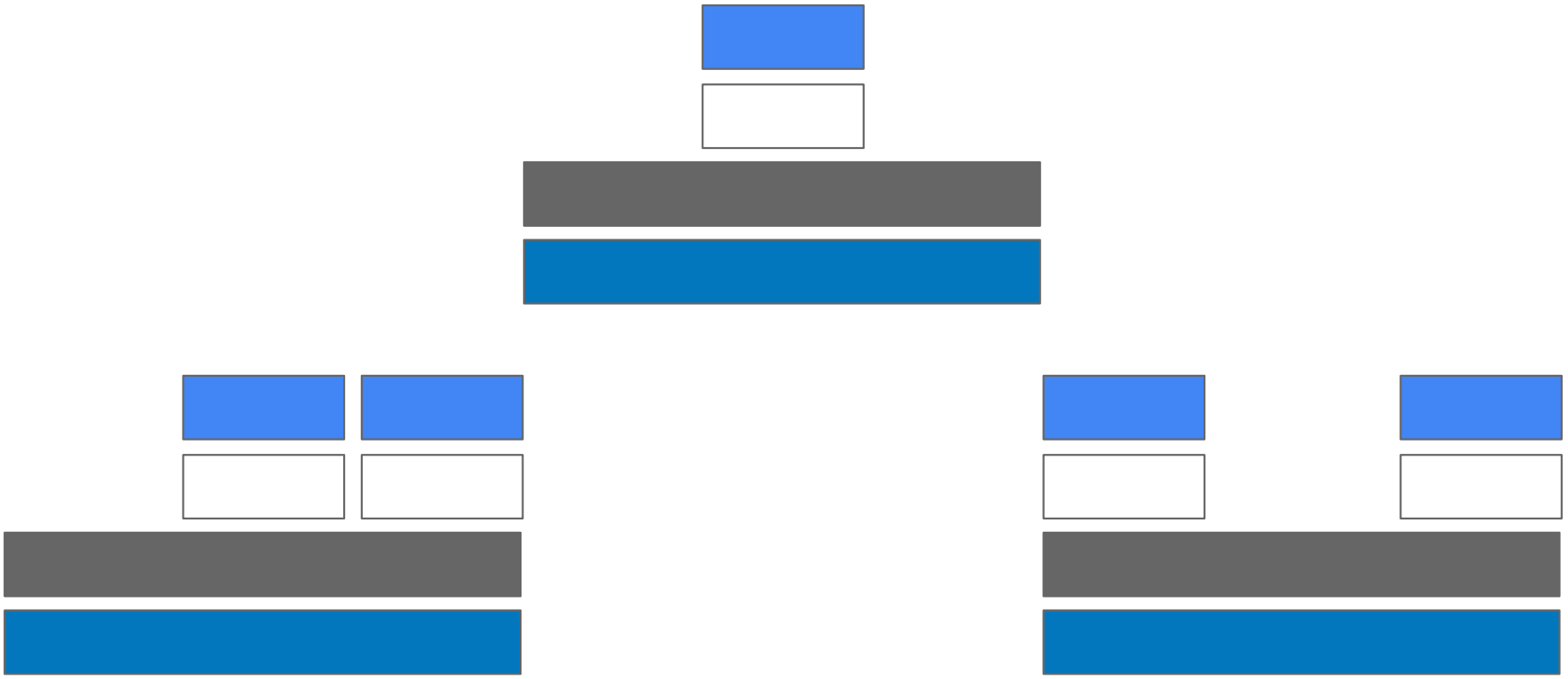


Virtual machine vs Container

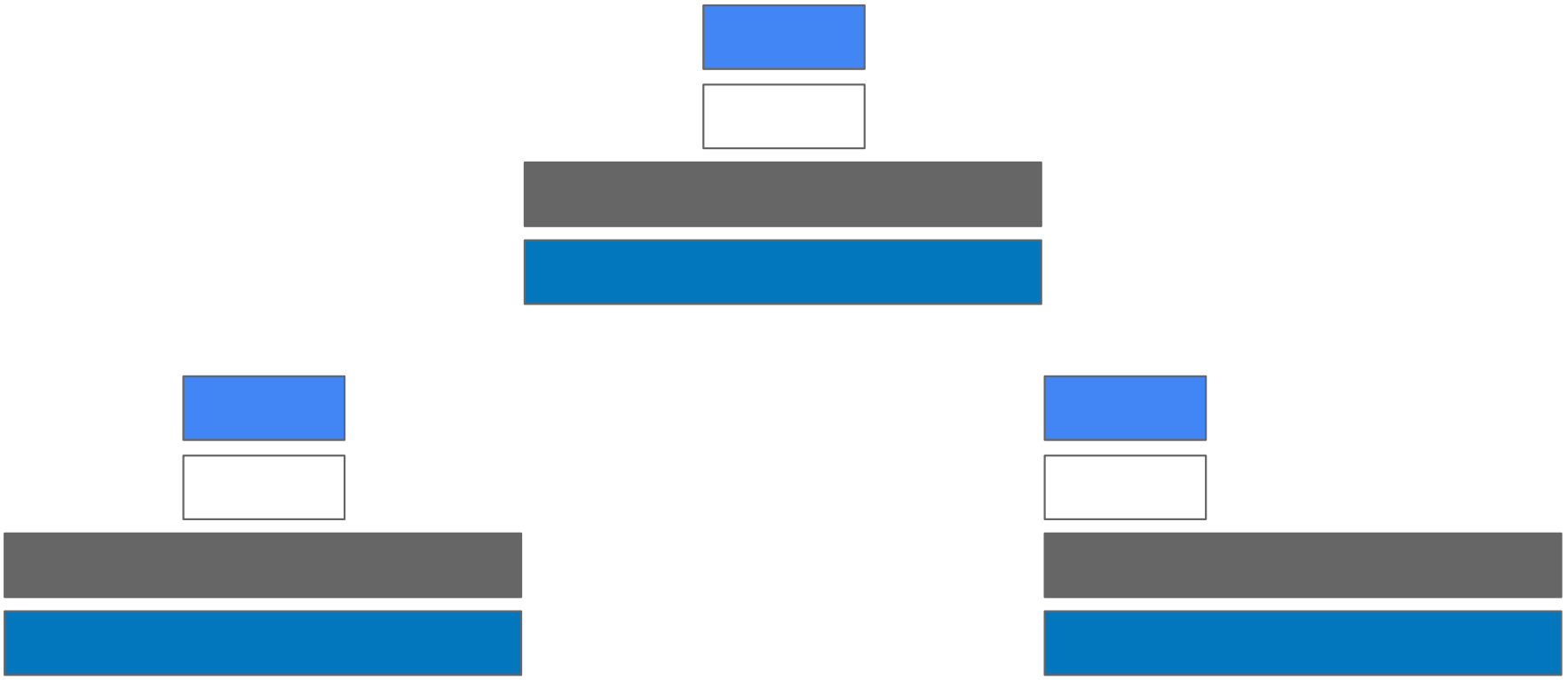




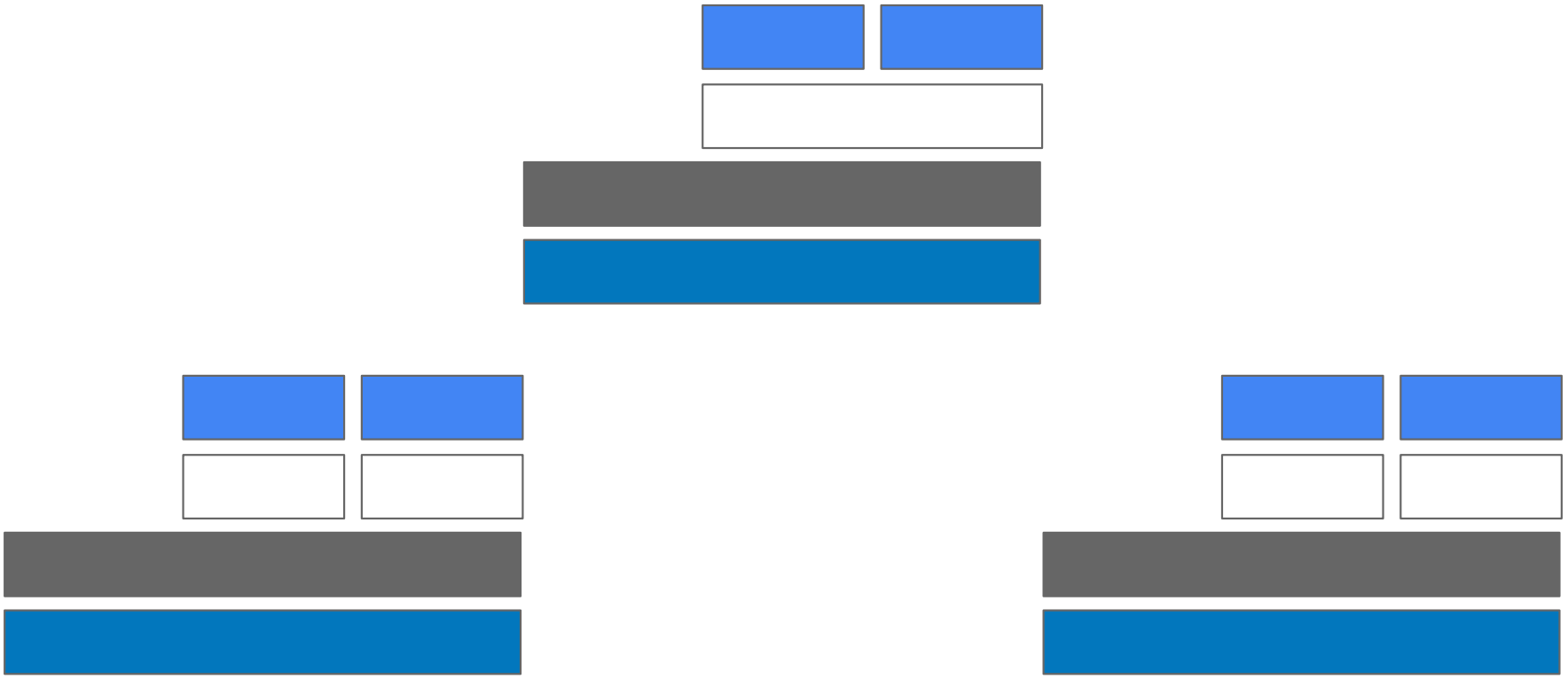
Containers are dynamic



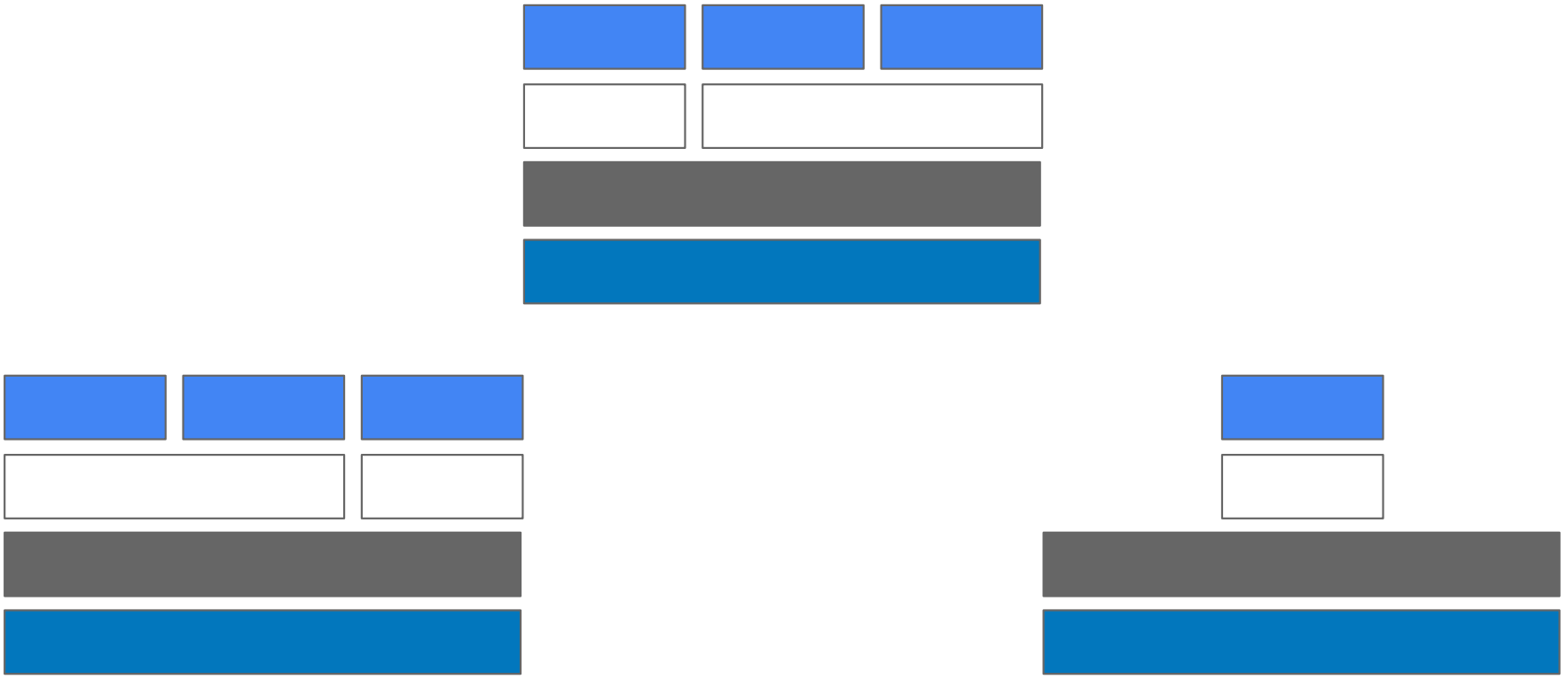
Containers are dynamic



Containers are dynamic



Containers are dynamic



Security implications

	Better	Worse
Attack surface	**Minimalist host OS** limits the surface of attack	**Hypervisors** are a strong security boundary

Security implications

	Better	Worse
Attack surface	**Minimalist host OS** limits the surface of attack	**Hypervisors** are a strong security boundary
Resource isolation	Host resources are **separated using namespaces and cgroups**	Host resources are **not all well separated**

Security implications

	Better	Worse
Attack surface	**Minimalist host OS** limits the surface of attack	**Hypervisors** are a strong security boundary
Resource isolation	Host resources are **separated using namespaces and cgroups**	Host resources are **not all well separated**
Root permissions	**Access controls** for app privileges and shared resources	Containers have access to **wider set of syscalls** to the kernel

Security implications

	Better	Worse
Attack surface	**Minimalist host OS** limits the surface of attack	**Hypervisors** are a strong security boundary
Resource isolation	Host resources are **separated using namespaces and cgroups**	Host resources are **not all well separated**
Root permissions	**Access controls** for app privileges and shared resources	Containers have access to **wider set of syscalls** to the kernel
Lifetime	Containers have a **shorter average lifetime**	It's **harder to do forensics** on a container that isn't there

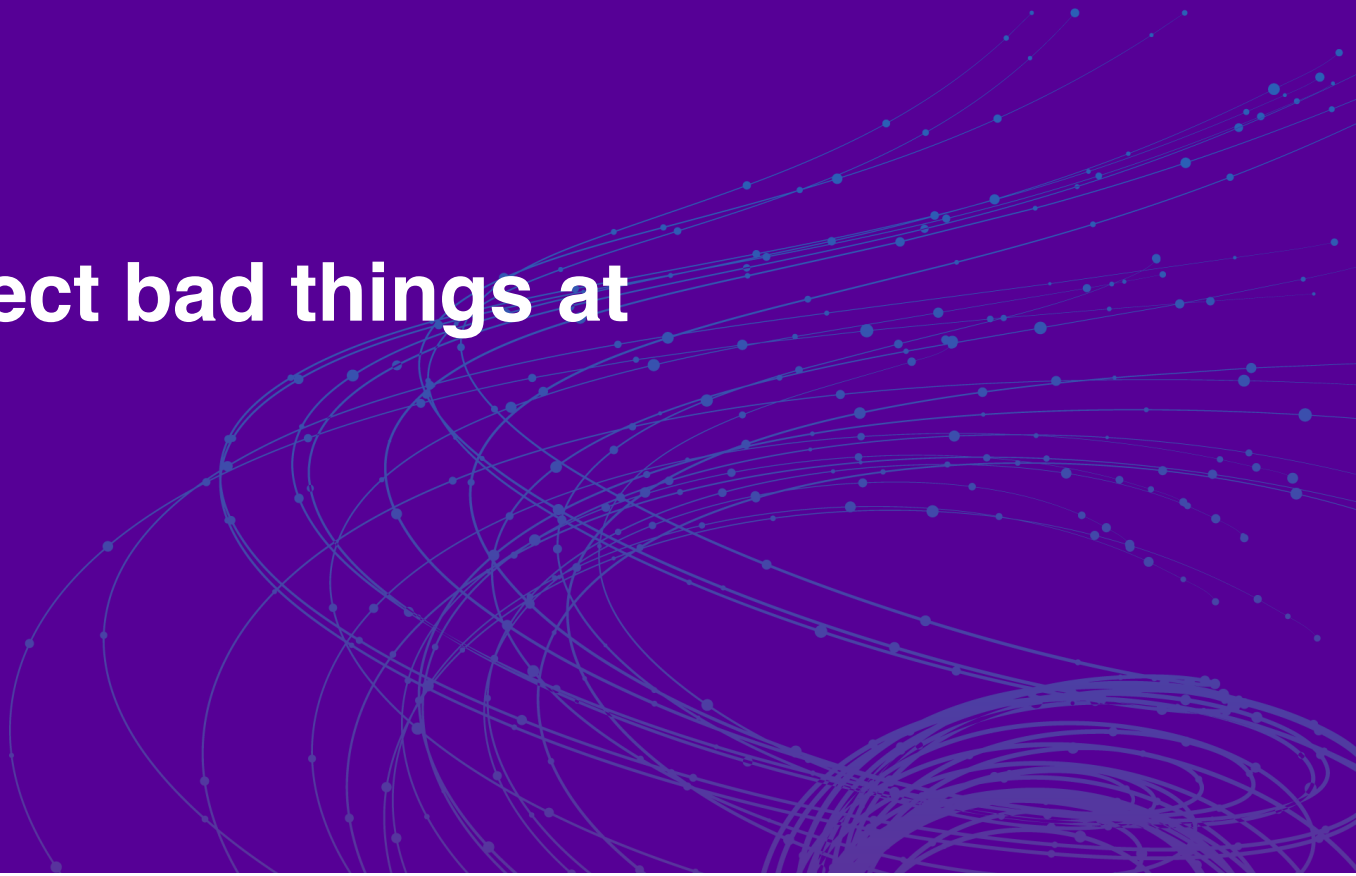
Security implications

	Better	Worse
Attack surface	**Minimalist host OS** limits the surface of attack	**Hypervisors** are a strong security boundary
Resource isolation	Host resources are **separated using namespaces and cgroups**	Host resources are **not all well separated**
Root permissions	**Access controls** for app privileges and shared resources	Containers have access to **wider set of syscalls** to the kernel
Lifetime	Containers have a **shorter average lifetime**	It's **harder to do forensics** on a container that isn't there

… but it's more the same than different

RSAConference2019

How to detect bad things at runtime



Why bother?

My secure supply chain prevents vulnerabilities!

But...

- Incomplete vuln scans
- Misconfigurations
- Zero days

Software supply chain is not perfect.
A fence is better than tall fence posts

NIST Cybersecurity Framework

Identify	Know your assets
Protect	Use security features and defaults
Detect	Detect unusual behavior
Respond	Respond to suspicious events
Recover	Figure out what happened and fix things

NIST Cybersecurity Framework

Identify

Protect

Detect

Respond

Recover

NIST Cybersecurity Framework

Identify	Know what your containers are
Protect	Use secure defaults to protect your containers
Detect	
Respond	
Recover	

NIST Cybersecurity Framework

Identify	Know what your containers are
Protect	Use secure defaults to protect your containers
Detect	Detect container behavior that deviates from the norm
Respond	
Recover	

NIST Cybersecurity Framework

Identify	Know what your containers are
Protect	Use secure defaults to protect your containers
Detect	Detect container behavior that deviates from the norm
Respond	Respond to a suspicious event in your container and mitigate the threat
Recover	

NIST Cybersecurity Framework

Identify	Know what your containers are
Protect	Use secure defaults to protect your containers
Detect	Detect container behavior that deviates from the norm
Respond	Respond to a suspicious event in your container and mitigate the threat
Recover	Complete forensics and fix things so this doesn't happen to your container again

NIST Cybersecurity Framework

Identify	Know what your ~~containers~~ assets are
Protect	Use secure defaults to protect your ~~containers~~ applications
Detect	Detect ~~container~~ behavior that deviates from the norm
Respond	Respond to a suspicious event ~~in your container~~ and mitigate the threat
Recover	Complete forensics and fix things so this doesn't happen ~~to your container~~ again

NIST Cybersecurity Framework

Identify	Know what your containers are
Protect	Use secure defaults to protect your containers
Detect	Detect container behavior that deviates from the norm
Respond	Respond to a suspicious event in your container and mitigate the threat
Recover	Complete forensics and fix things so this doesn't happen to your container again

Detect: container monitoring designs

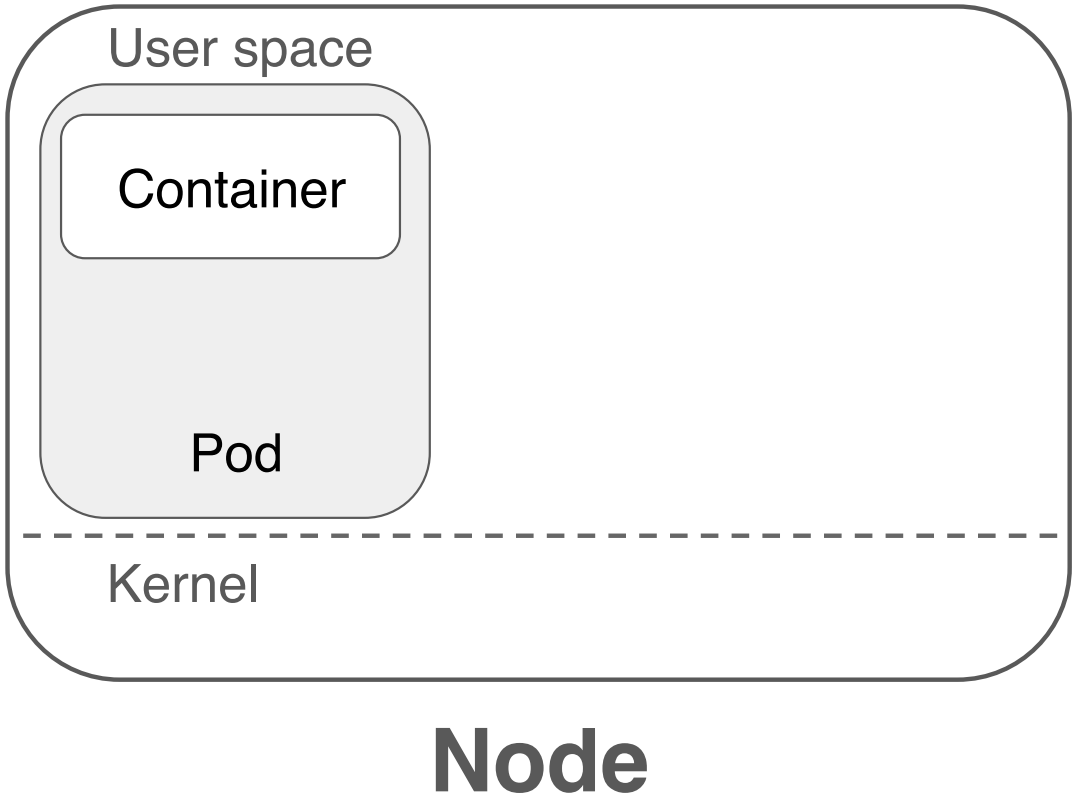
- Hook into your container
- **Log** a bunch of stuff
- **Policies** for:
  - alerts
  - automatic remediation
- **Allow forensics** afterwards

Detect options

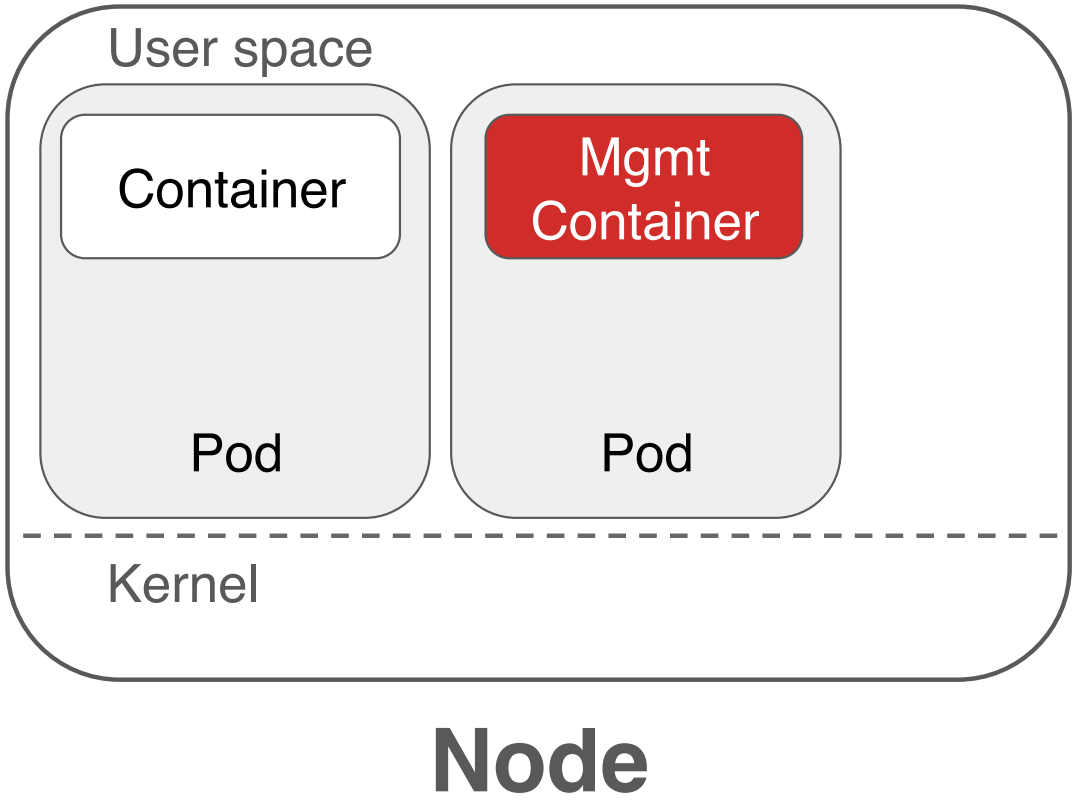
Examine process activity, network activity, file activity, … **HUGE VOLUME**

- **ptrace, kprobes, tracepoints**
- **Audit logs**
- **eBPF**: kernel introspection
- **XDP**: uses eBPF for filtering network packets
- **User-mode API**: for kernel events like inotify

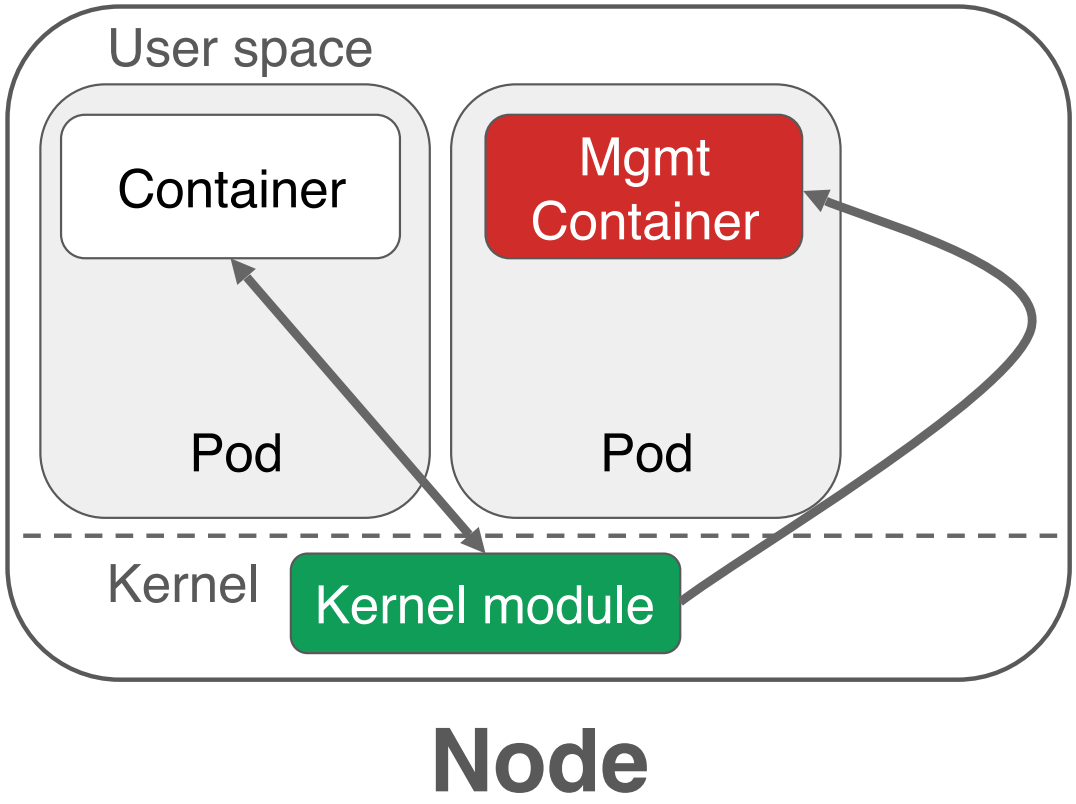
Deployment models



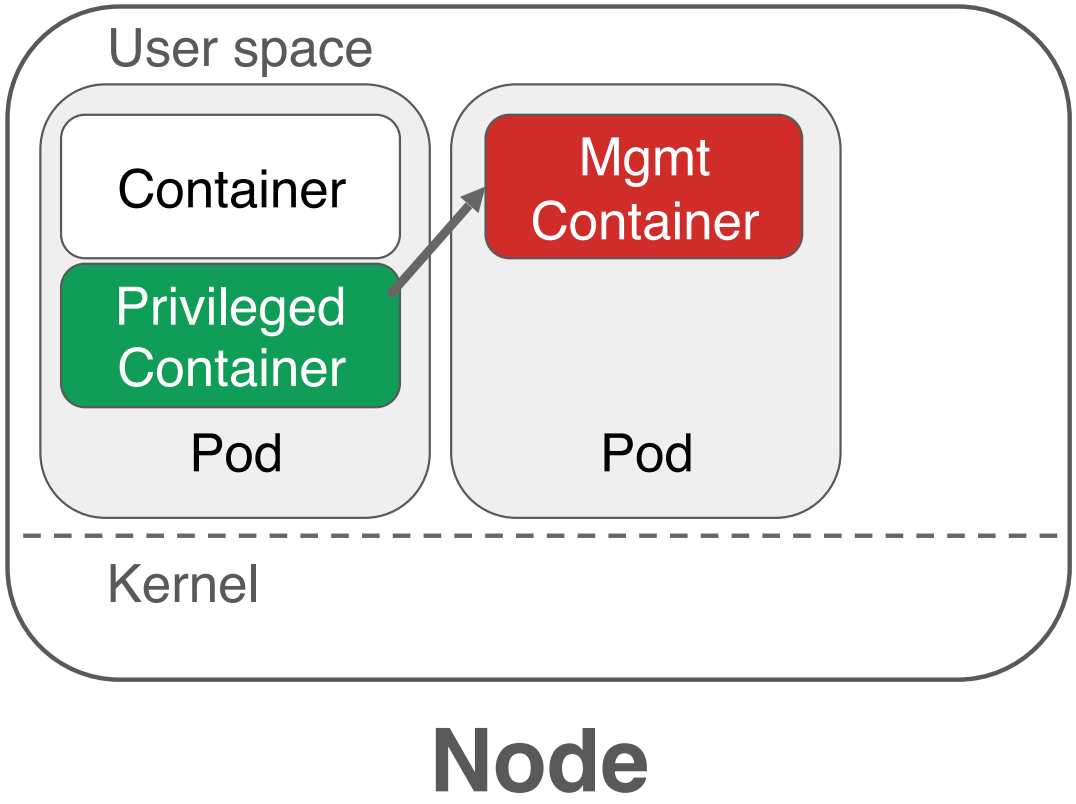
Detect and capture



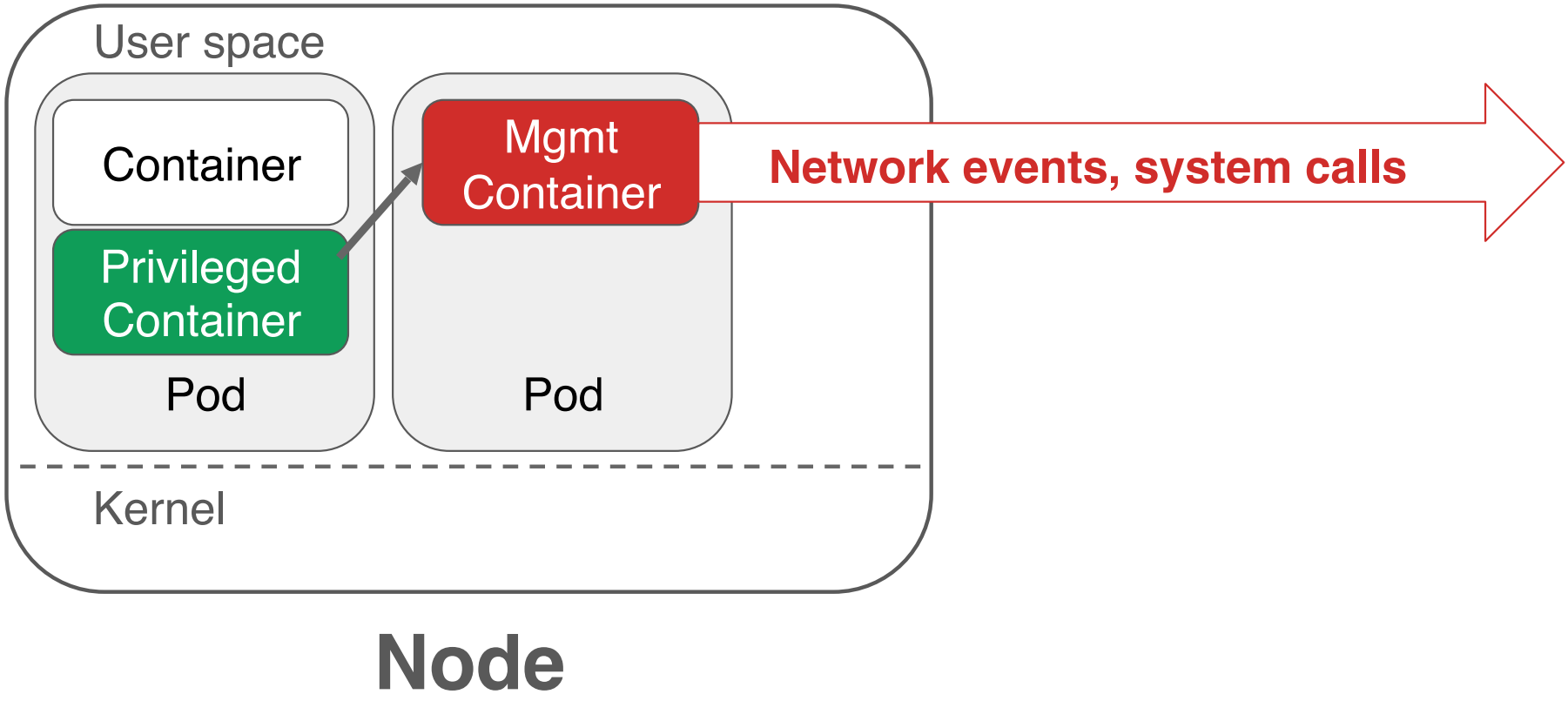
Detect and capture



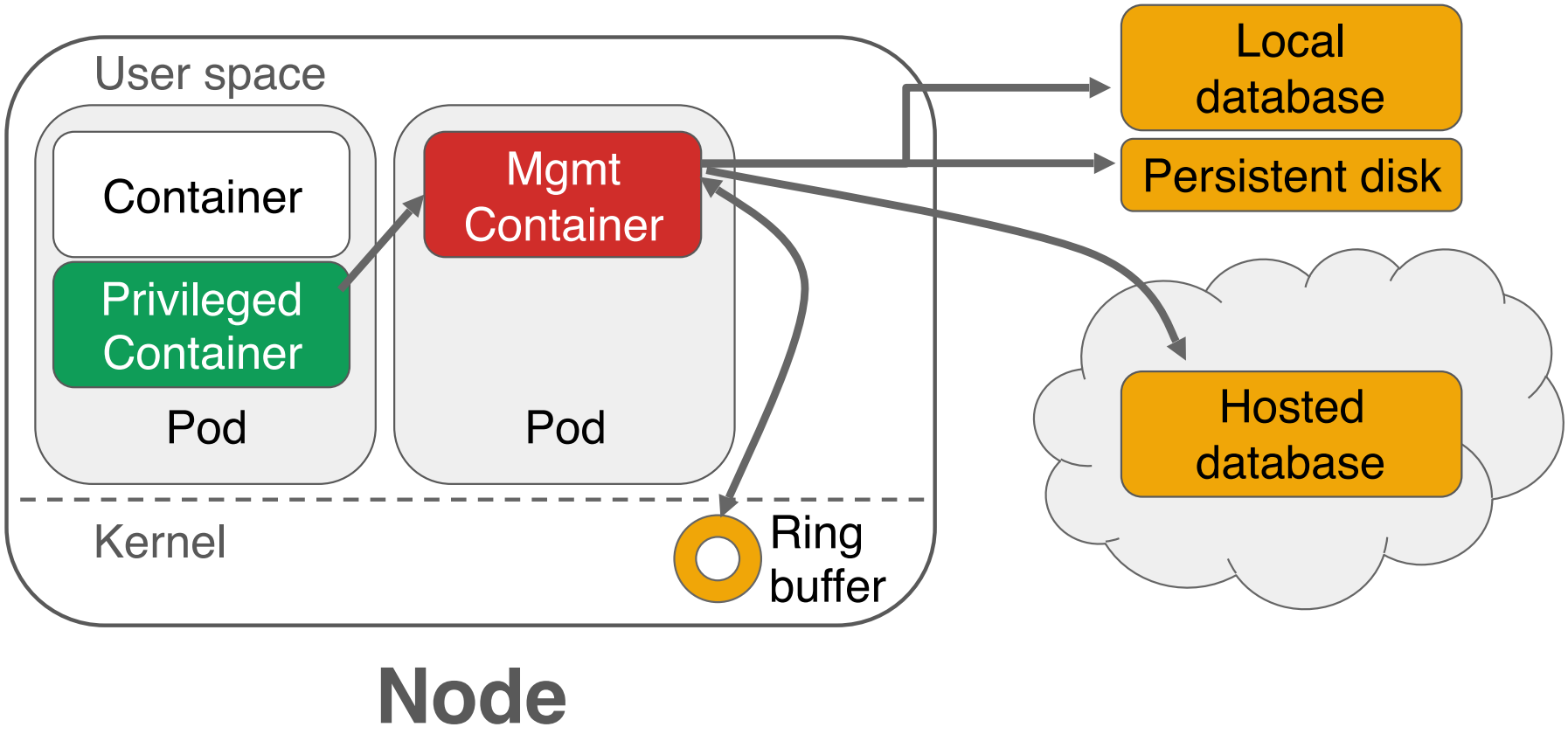
Detect and capture



Manage


User space
Container
Privileged Container
Pod
Mgmt Container
Pod
Kernel
Network events, system calls
Node

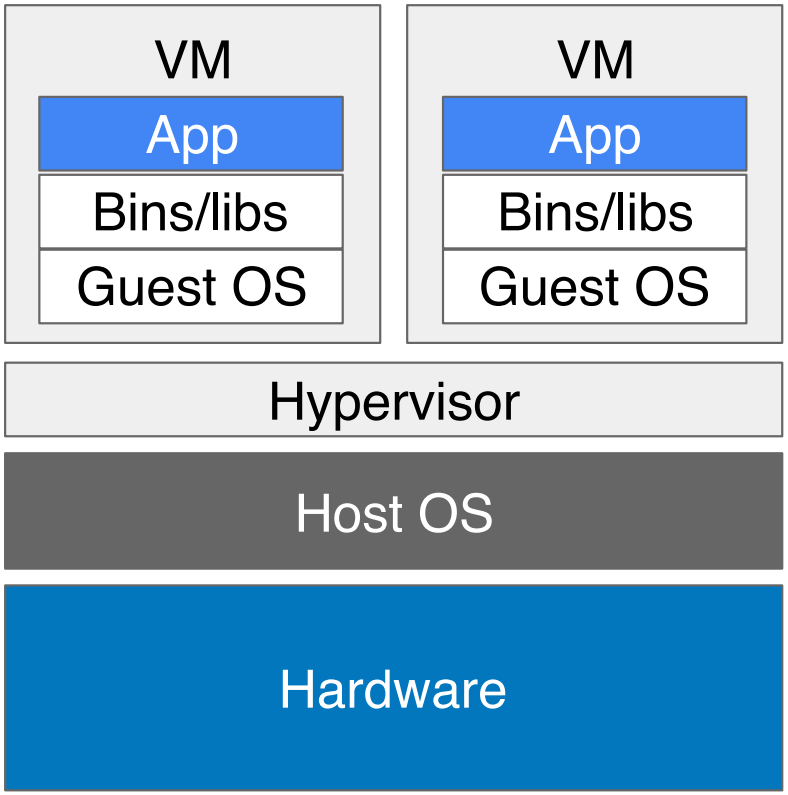
Store


User space
Container
Privileged Container
Pod
Mgmt Container
Pod
Kernel
Ring buffer
Local database
Persistent disk
Hosted database
Node

Respond

- **Send an alert**
- **Isolate a container**, i.e. move it to a new network
- **Pause a container**, i.e. stop all running processes
- **Restart a container**, i.e. kill and restart processes
- **Kill a container**, i.e. kill processes without restart

So, why are containers special again?



Virtual machine

Long lived systems

- Manual security patches and reviews

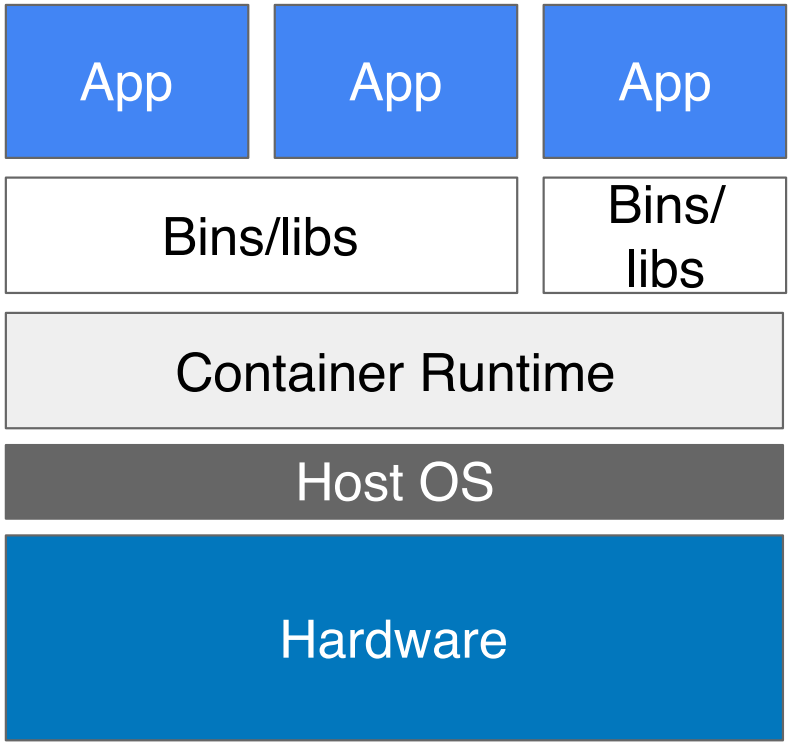
Per-host software

- IDS for host software

Shared, physical network

- Host-centric appliance for network traffic

So, why are containers special again?



Container

Dynamic short-lived containers

- Need to redeploy often

Load isolation by container

- Need container IDS

Overlay network

- Need container network monitoring

Apply slide - What you can do today

- Make it part of your security plan
  - Try out open source options
  - Evaluate commercial options
- Deploy early
  - Get baseline readings
  - Tune your signals
- Rehearse an event

RSAConference2019

Demo

Demo on YouTube

http://jen.run/krs-demo

What we discussed

Container security overview

Containers differ from VMs

Don't build fence posts

What you can do today

Thank you!

Slides: https://mimming.com/krs