$ whoami - Orlix

Orlin Vasilev

 - Software Infrastructure Engineer at VMware

- father of two(girl and boy)

- snowboarder/DIY person/home BBQ pitmaster

twitter: @OrlinVasilev

github: @OrlinVasilev

kubernetes

Agenda

- Cloud Native Computing Foundation and CNCF Bulgaria

- Short Intro to Kubernetes

- Container security and lifecycle

- Kubernetes Security benchmarking

- Demo – kube-bench/kubeaudit/sonoboy

- Q&A

kubernetes

CNCF - "few" other projects

Graduated

kubernetes
Orchestration

Prometheus
Monitoring

envoy
Service Proxy

Incubating

OPENTRACING
Distributed Tracing API

fluentd
Logging

GRPC
Remote Procedure Call

containerd
Container Runtime

rkt
Container Runtime

JAEGER
Distributed Tracing

Notary
Security

Vitess
Storage

CoreDNS
Service Discovery

NATS
Messaging

LINKERD
Service Mesh

HARBOR
Registry

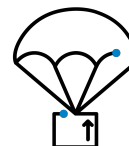HELM
Package Management

CNI
Networking API

ROOK
Storage

etcd
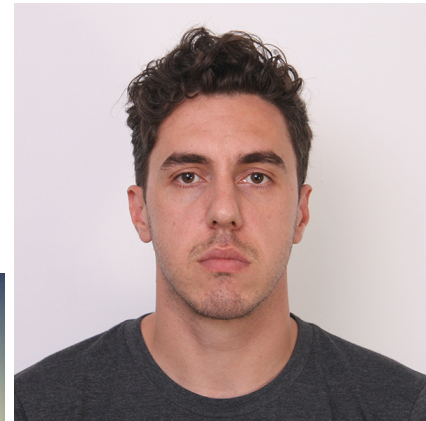Key/Value Store

TUF
Software Update Spec

kubernetes

CNCF Bulgaria

- 7 meetups

- +500 members(525 on 20.01.2019)

- Meetup: https://goo.gl/16bF3X

- YouTube: https://goo.gl/88yH4n

- CFP: https://goo.gl/TfVEMc

Vladimir Dimov

Spas Atanasov

What Does "Kubernetes" Mean?

Greek for "pilot" or
"Helmsman of a ship"
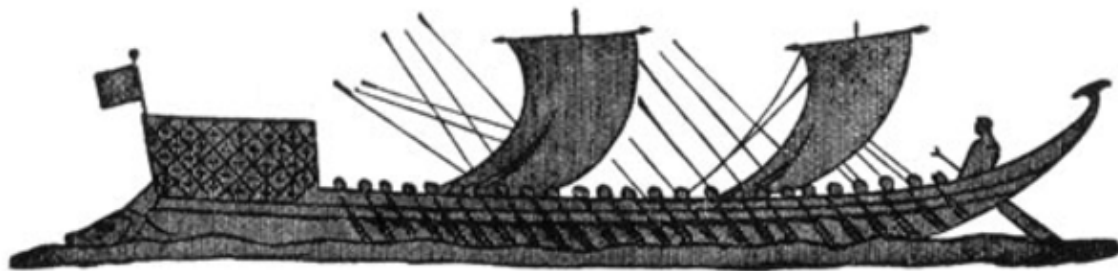
Image Source

kubernetes

What is Kubernetes?

- Project that was spun out of Google as an open source container orchestration platform.

- Built from the lessons learned in the experiences of developing and running Google's Borg and Omega.

- Designed from the ground-up as a **loosely coupled** collection of components centred around deploying, maintaining and scaling workloads.

kubernetes

Self Healing

Kubernetes will **ALWAYS** try and steer the cluster to its desired state.

- **Me:** "I want 3 healthy instances of redis to always be running."
- **Kubernetes:** "Okay, I'll ensure there are always 3 instances up and running."
- **Kubernetes:** "Oh look, one has died. I'm going to attempt to spin up a new one."

kubernetes

Project Stats

- Over 42,000 stars on Github

- 1800+ Contributors to K8s Core

- Most discussed Repository by a large margin

- **50,000+** users in Slack Team

kubernetes

What can Kubernetes REALLY do?

- Autoscale Workloads

- Blue/Green Deployments

- Fire off jobs and scheduled cronjobs

- Manage Stateless and Stateful Applications

- Provide native methods of service discovery

- Easily integrate and support 3rd party apps

kubernetes

Architecture Overview
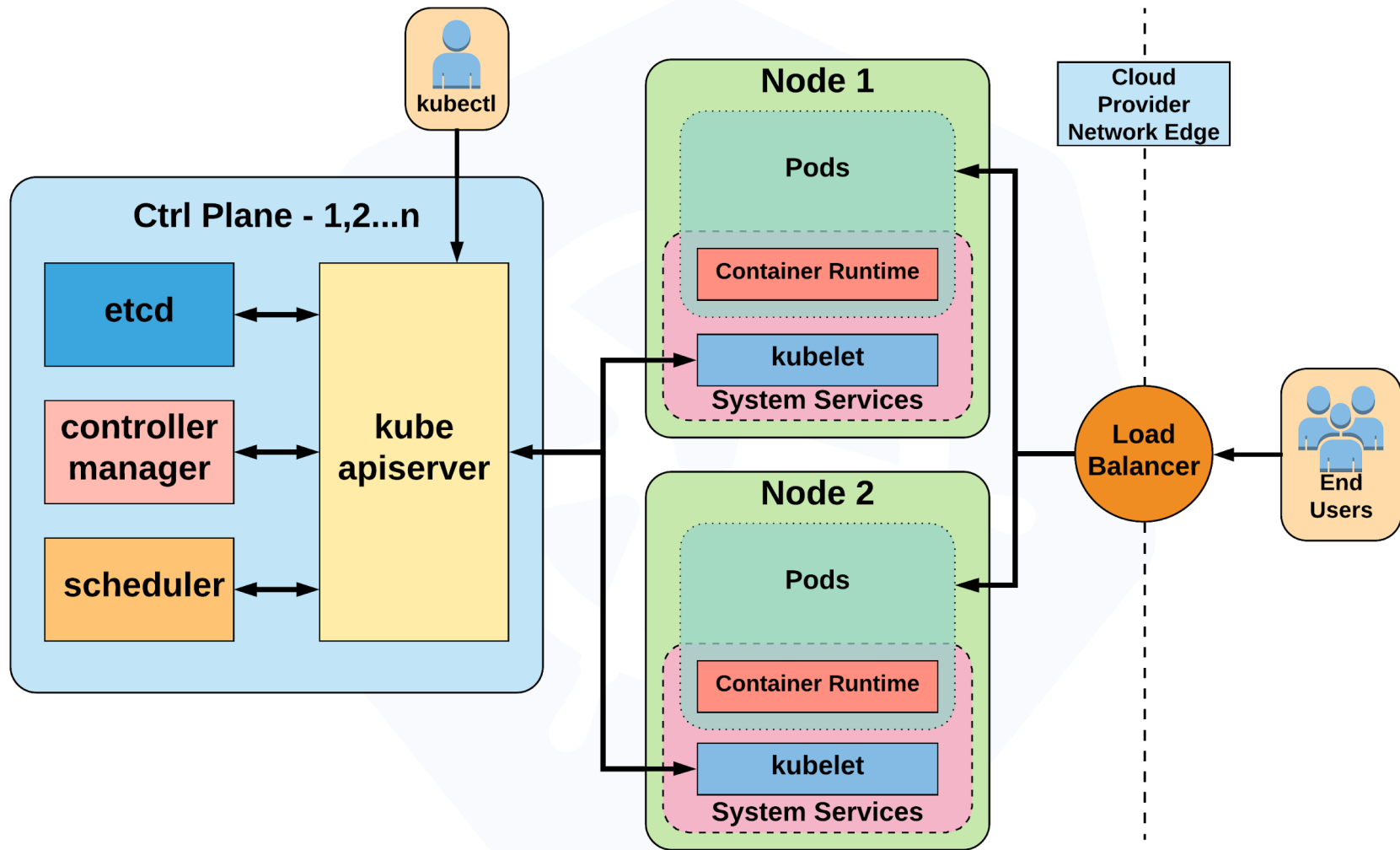
Control Plane Components

- kube-apiserver
- etcd
- kube-controller-manager
- kube-scheduler
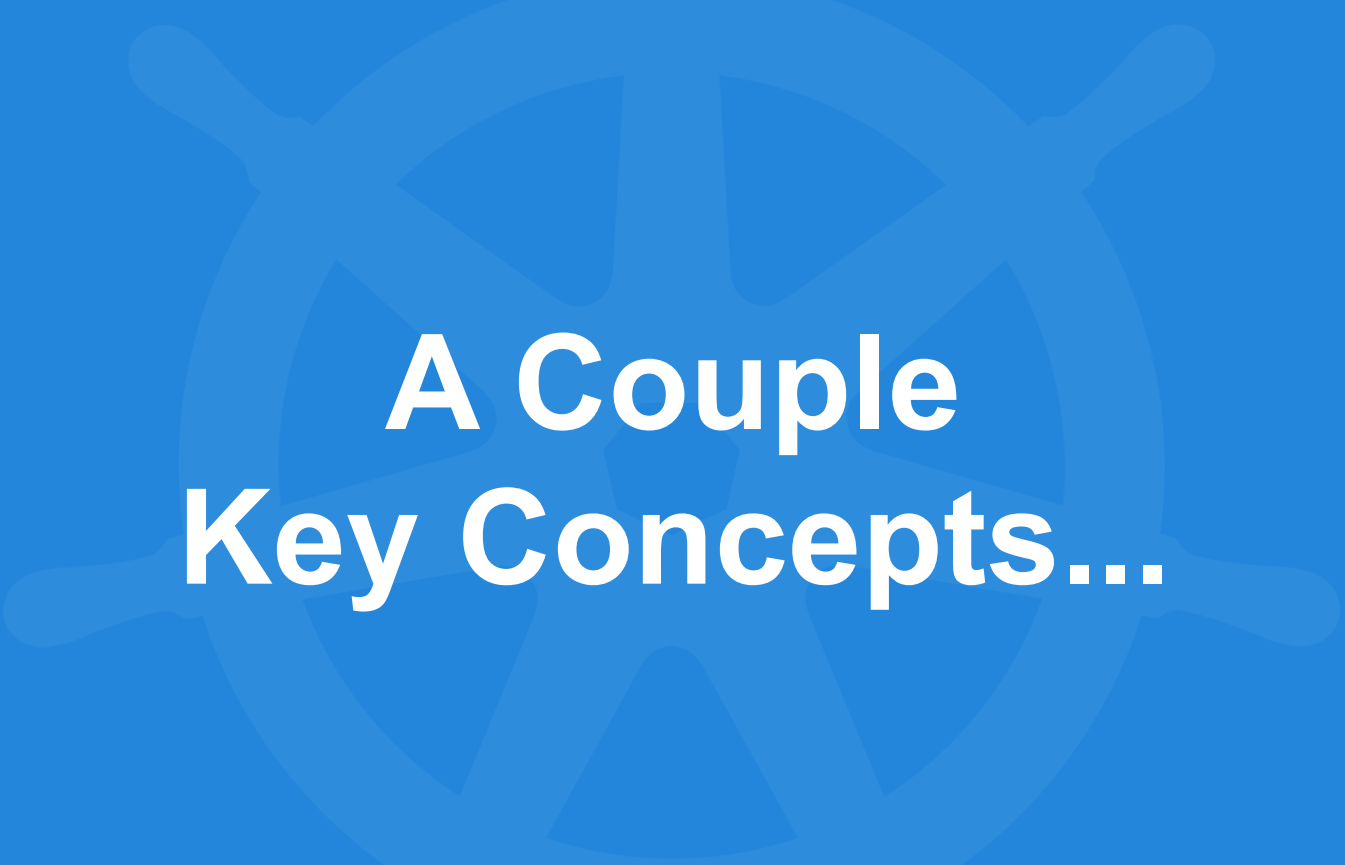
Node Components

- kubelet
- kube-proxy
- Container Runtime Engine

A Couple Key Concepts...

Kubernetes Objects

Basic:

- Pod
- Service
- Volume
- Namespace
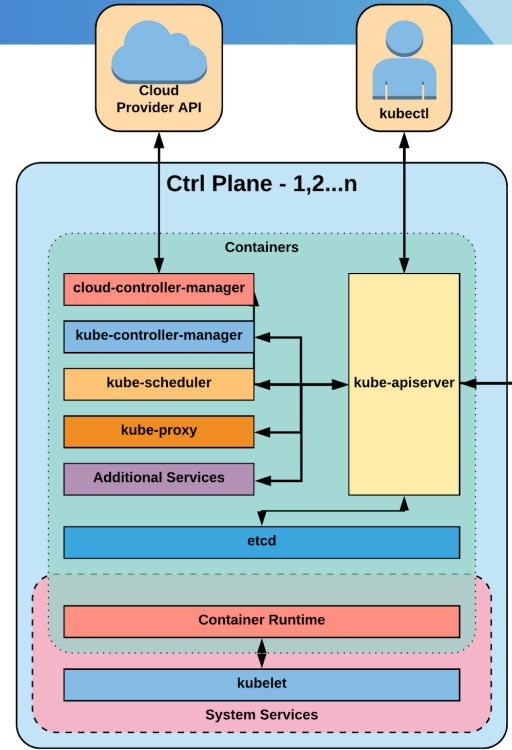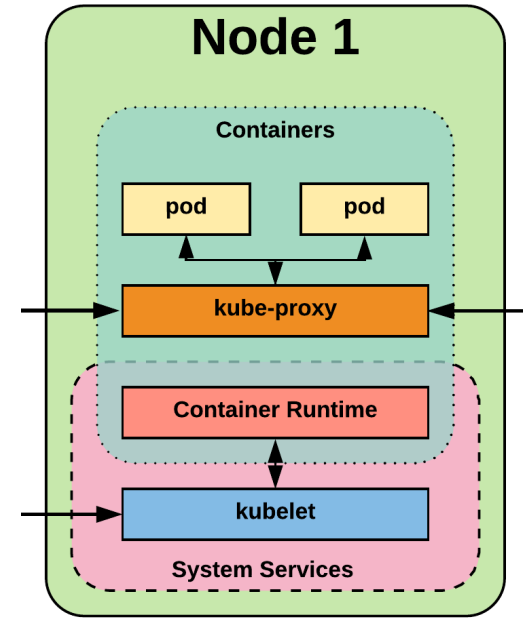
More:

- ReplicaSet
- Deployment
- StatefulSet
- DaemonSet
- Job
- ….

kubernetes

Namespaces

Namespaces are a logical cluster or environment, and are the primary method of partitioning a cluster or scoping access.

```
apiVersion: v1
kind: Namespace
metadata:
  name: prod
  labels:
    app: MyBigWebApp
```

```
$ kubectl get ns --show-labels
NAME          STATUS   AGE      LABELS
default       Active   11h      <none>
kube-public   Active   11h      <none>
kube-system   Active   11h      <none>
prod          Active   6s       app=MyBigWebApp
```

kubernetes

Pod

- **Atomic unit** or smallest "*unit of work*" of Kubernetes.

- Pods are **one or MORE containers** that share volumes, a network namespace, and are a part of a **single context**.

- **Ephemeral**
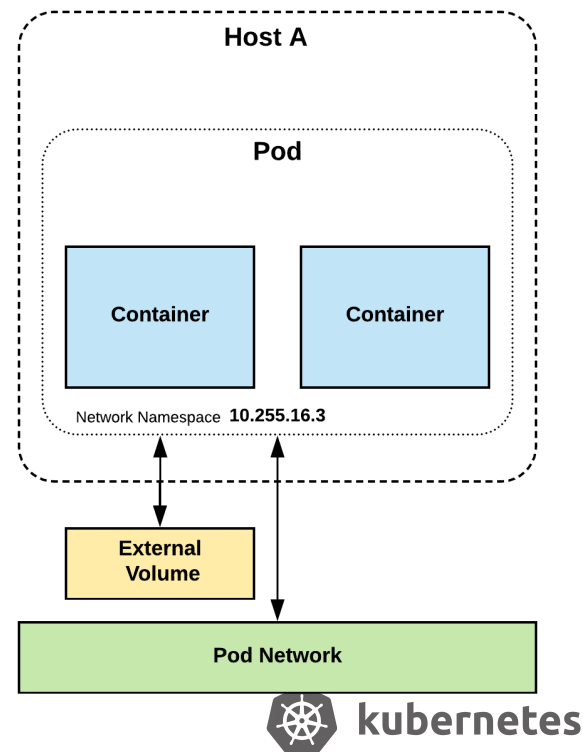
Services

- **Unified method of accessing** the exposed workloads of Pods.

- **Durable resource**
  - static cluster IP
  - static namespaced DNS name
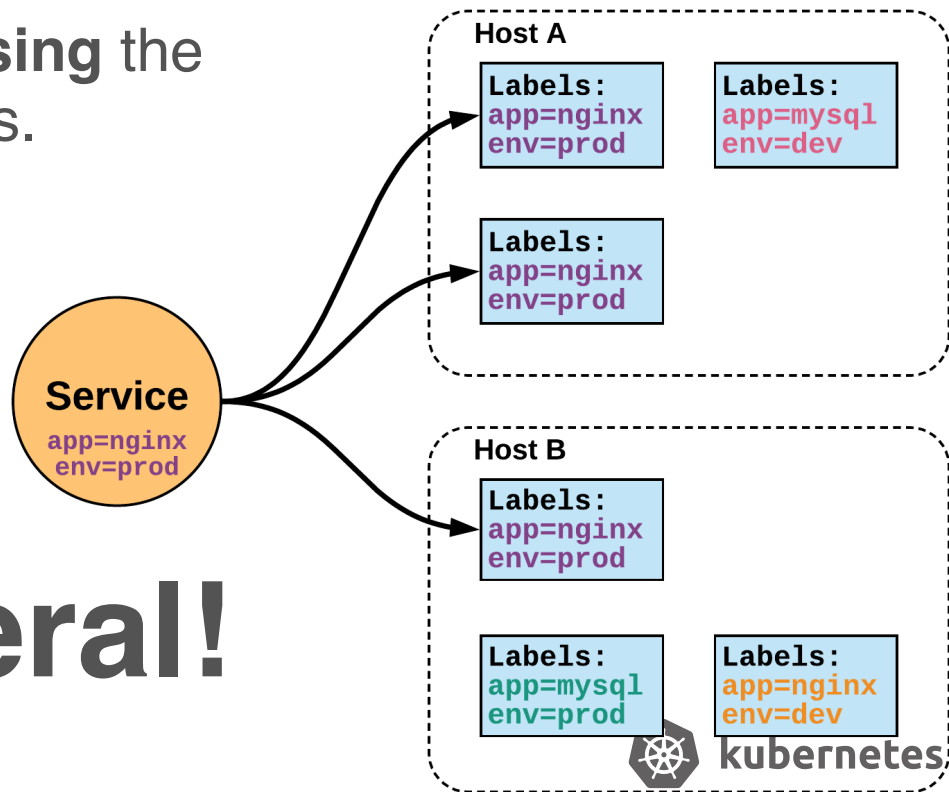
NOT Ephemeral!

Service
app=nginx
env=prod

Host A

Labels:
app=nginx
env=prod

Labels:
app=mysql
env=dev

Labels:
app=nginx
env=prod

Host B

Labels:
app=nginx
env=prod

Labels:
app=mysql
env=prod

Labels:
app=nginx
env=dev

kubernetes

Containers and Kubernetes Security and Benchmarking

Possible Attack Surface

Launch too many pods / high consume CPU/RAM/ Disk

Malicious Image, Compromised Container

Escape the container

Service Account Tokens

Dashboard Access

Direct Etcd Access

Kubelet Exploit

Application Tampering

Metrics Scraping

Metadata API

kubernetes

Container and Kubernetes Security Aspects

- Underlying Infrastructure(On-Prem or Public Cloud)
- Host security(OS)
- Docker daemon security(or containerd, rkt, CRI-O …)
- Container security
- Kubernetes security:
  - Properly configured RBACs
  - Secrets
  - Pod Security Policy
  - Network Policy
  - Admission Controllers
  - etc…

kubernetes

Container Security

- Container Runtime - Least Privileges
- Base Image
- Image Builder/Maintainer
- Image Scanning
- Image Signing
- do not run as root in container

```
RUN useradd -r -u 1001 -g appuser appuser
USER appuser
```

kubernetes

Kubernetes RBAC

- enabled in kube-api : —authorization-mode=RBAC
- Role and ClusterRole
- RoleBinding and ClusterRoleBinding
- ServiceAccounts

kubernetes

Role and RoleBinding

```
kind: Role
apiVersion: rbac.authorization.k8s.io/v1
metadata:
 namespace: default
 name: pod-reader
rules:
- apiGroups: [""] # "" indicates the core API group
 resources: ["pods"]
 verbs: ["get", "watch", "list"]
```

```
kind: RoleBinding
apiVersion: rbac.authorization.k8s.io/v1
metadata:
 name: read-pods
 namespace: default
subjects:
- kind: User
 name: jane # Name is case sensitive
 apiGroup: rbac.authorization.k8s.io
roleRef:
 kind: Role #this must be Role or ClusterRole
 name: pod-reader #must match name
 apiGroup: rbac.authorization.k8s.io
```

kubernetes

Kubernetes Secrets

- A Secret is an object that contains a small amount of sensitive data such as a password, a token, or a key.
- base64ed
- consumed within a pod as volumeMounts
- secrets used to keep docker login information for private registry
- use secrets instead of writing sensitive data in the containers

```yaml
apiVersion: v1
kind: Secret
metadata:
  name: mysecret
type: Opaque
data:
  username: YWRtaW4=
  password: MWYyZDFlMmU2N2Rm
```

kubernetes

PodSecurityPolicy

- Enforced by the PodSecurityPolicy admission controller enabled on kube-api
- PSP is an ClusterLevel resource
- Set of conditions which allows a pod to be run
- some examples
  - [privileged](#)
  - [runAsUser](#)
  - [hostPID](#)
  - [volumes](#)

Tips:
- create PSP before enabling PSP admission controller
- PSP are applied alphabetically

Usage via RBAC with ClusterRoles and ClusterRoleBindings to ServiceAccount or User

```
apiVersion: policy/v1beta1
kind: PodSecurityPolicy
metadata:
 name: privileged
 annotations:
   seccomp.security.alpha.kubernetes.io/allowedProfileNames: '*'
spec:
 privileged: true
 allowPrivilegeEscalation: true
 allowedCapabilities:
 - '*'
 volumes:
 - '*'
 hostNetwork: true
 hostPorts:
 - min: 0
   max: 65535
 hostIPC: true
 hostPID: true
 runAsUser:
   rule: 'RunAsAny'
 seLinux:
   rule: 'RunAsAny'
 supplementalGroups:
   rule: 'RunAsAny'
 fsGroup:
   rule: 'RunAsAny'
```

kubernetes

Network Policy

A network policy is a specification of how groups of pods are allowed to communicate with each other and other network endpoints.

Uses labels to select pods and define rules.

NetworkPolicy needs network plugin which supports it.(calico/weave..)

By default, if no policies exist in a namespace, then all ingress and egress traffic is allowed to and from pods in that namespace.

kubernetes

NetowrkPolicy example

```yaml
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
 name: test-network-policy
 namespace: default
spec:
 podSelector:
  matchLabels:
    role: db
 policyTypes:
 - Ingress
 - Egress
 ingress:
 - from:
  - ipBlock:
     cidr: 172.17.0.0/16
     except:
     - 172.17.1.0/24
```

```yaml
   - namespaceSelector:
     matchLabels:
      project: myproject
   - podSelector:
     matchLabels:
      role: frontend
  ports:
  - protocol: TCP
   port: 6379
 egress:
 - to:
  - ipBlock:
     cidr: 10.0.0.0/24
  ports:
  - protocol: TCP
   port: 5978
```

kubernetes

CIS Kubernetes Benchmark

- 200+ pages of best practices and tests
    https://www.cisecurity.org/benchmark/kubernetes/
- Tests for workers, masters and federated nodes

- **kube-bench** (Aqua Security)
    - 1265 stars at github
    - ~30 contributors
    - https://github.com/aquasecurity/kube-bench

kubernetes

more tools

- **sonobuoy**(heptio)
    - 1141 stars at github
    - 33 contributors
    - https://github.com/heptio/sonobuoy
- **kubeaudit** (shopify)
    - 306 stars at github
    - 16 contributors
    - https://github.com/Shopify/kubeaudit
- **k8guard**
    - 122 stars at github
    - 8 contributors
    - https://github.com/k8guard/k8guard-start-from-here

kubernetes

CVE-2018-1002105 - Remote Privilege Escalation Vulnerability

With a specially crafted request, users that are authorized to establish a connection through the Kubernetes API server to a backend server can then send arbitrary requests over the same connection directly to that backend, authenticated with the Kubernetes API server's TLS credentials used to establish the backend connection.

- https://github.com/kubernetes/kubernetes/issues/71411
- https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2018-1002105
- https://www.youtube.com/watch?v=1M4oXPgxYyE
- https://github.com/evict/poc_CVE-2018-1002105

Affected versions:

- Kubernetes v1.0.x-1.9.x
- Kubernetes v1.10.0-1.10.10 (fixed in v1.10.11)
- Kubernetes v1.11.0-1.11.4 (fixed in v1.11.5)
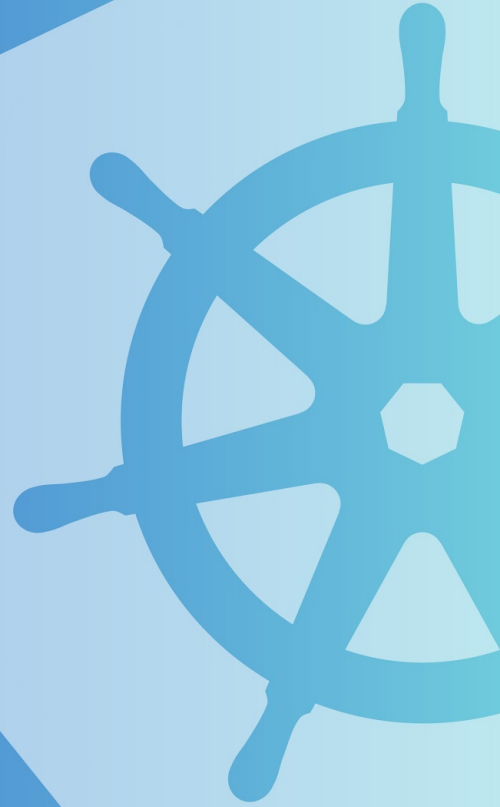- Kubernetes v1.12.0-1.12.2 (fixed in v1.12.3)

kubernetes

Demo

Demo 1 - scraping unsecured metrics
Demo 2 - exploiting misconfigured taints for master
Demo 3 - exploiting the lack of PSP

aliases:
k = kubectl
kdp = kubectl describe pods
kep = kubectl edit pod
kgn = kubectl get nodes —show-lables -o wide
lp = kubectl get pods
lpa = kubectl get pods —all-namespaces
kshell = kubectl exec -it $$ bash

Resources and goodreads

- https://github.com/kelseyhightower/kubernetes-the-hard-way
- https://github.com/hardening-kubernetes/from-scratch
- https://github.com/cncf/presentations/tree/master/kubernetes

kubernetes

Q&A