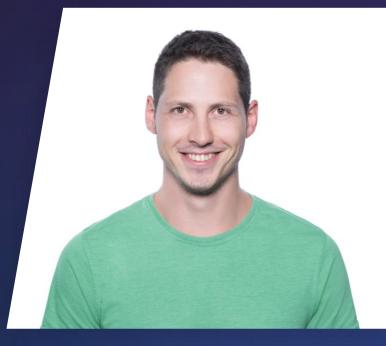
# **Conquering The Jungle of Kubernetes Compliance**

## IT-SECX 07.10.2022



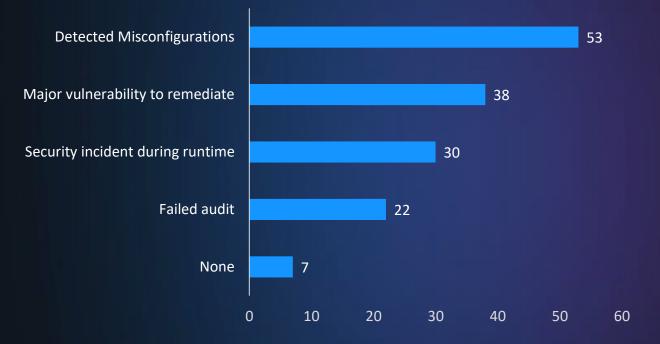
Markus Gierlinger Cloud Native Security Researcher

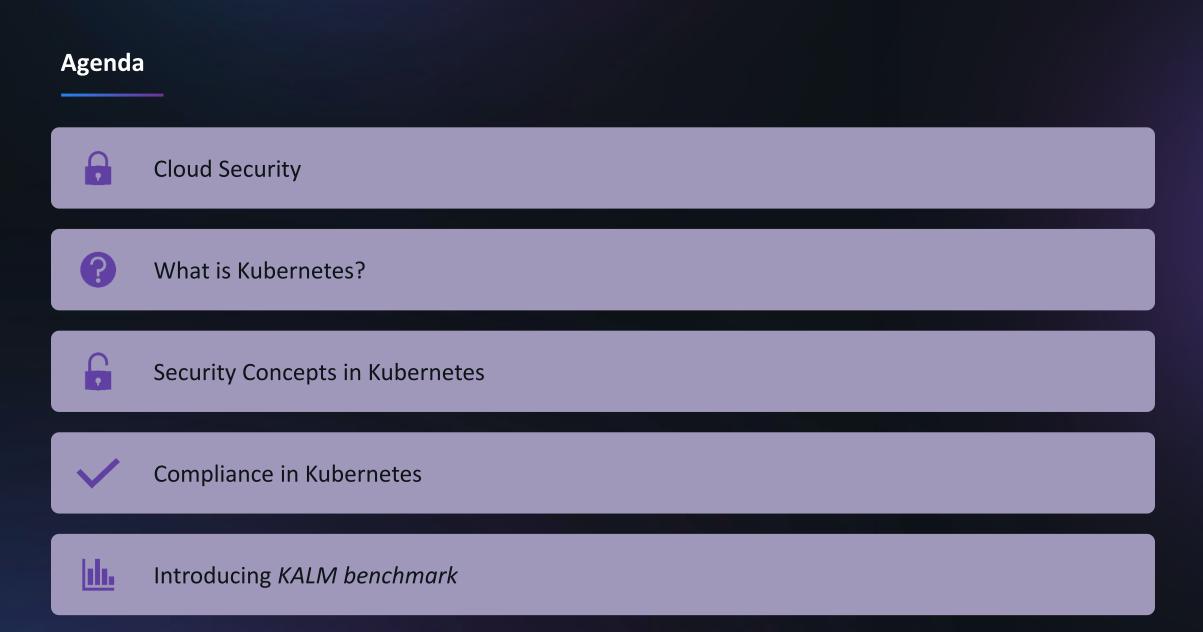
# Johnatrace

## **Kubernetes Security in 2022**

**93 %** 

experienced 1+ security incidents in their Kubernetes environments in the last 12 months





## This talk is and is NOT about

- Focus only on
  - Kubernetes core resources
- Not about:
  - Plugins and 3rd-Party applications in Kubernetes
    - E.g. Service Meshes, CNI-Plugins
  - Not about compliance frameworks (HIPPA, PCI, PHI, and SOC2, etc.) in general
    - Instead it's about compliance w.r.t. to Kubernetes specific security controls
  - Covering peculiarities for certain cloud providers

## 4C's of Cloud Native Security

- Defense-in-depth approach
- Every "C" represents a layer for Dev/DevOps

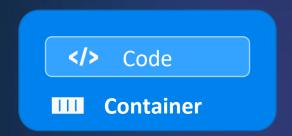
## 4C's of Cloud Native Security - Code

- Primary attack surface
- Shift to micro-services
- Key issues:
  - Insecure code
  - Supply chain
  - Logic flaws

</>
 Code

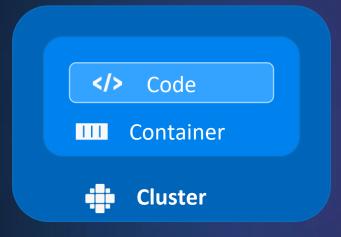
## 4C's of Cloud Native Security - Container

- Clusters use Container Runtime engines
- Packaged apps and micro-services
- Key issues:
  - Vulnerabilities
  - Supply chain
  - Misconfiguration



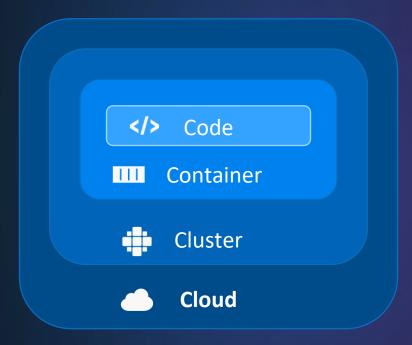
## 4C's of Cloud Native Security - Cluster

- Usually Kubernetes
- Focus areas: <sup>[1]</sup>
  - Securing cluster components
  - Securing workloads in the cluster



## 4C's of Cloud Native Security - Cloud

- Physical infrastructure that runs server
- Typically managed by Cloud Service Providers (CSP)
- Shared Responsibility Model
- Key Issues: <sup>[1]</sup>
  - Misconfigurations
  - Automation loopholes



# **Cloud - Shared Responsibility**

Responsibility	Onipremises laas saas faas
Data classification and accountability	
Client and end-point protection	
Identity and access management	
Application-level controls	
Network controls	
Host infrastructure	
Physical security	
	Cloud Customer Cloud Provider

## 4C's of Cloud Native Security

- Defense-in-depth approach
- Every "C" represents a layer for Dev/DevOps



# **Kubernetes**



## What is Kubernetes?

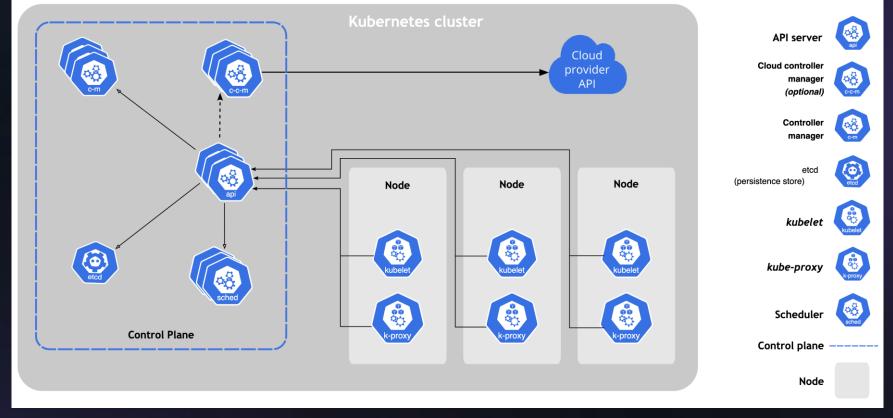
- Platform for managing containerized apps and services
- Open sourced by Google in 2014
- "OS" of the cloud
  - Storage orchestration
  - Automatic bin packing
  - Self-healing
  - Service discovery and load balancing
  - Automated rollouts and rollbacks
  - Secret and configuration management



## **Kubernetes Architecture**

• Control Plane

- API Server
- ETCD
- Controllers
- Data Plane
  - Kubelet
  - Kube-proxy



Source: Kubernetes Components

• Infrastructure as Code (IaC)

- Infrastructure as Code (IaC)
- All resources are Kubernetes objects

- Infrastructure as Code (IaC)
- All resources are Kubernetes objects

	Application.yami	- L ×	
1 apiVe <u>rsion: v</u>	1		
2 kind: Namespa	ce		
3 metadata:			
4 name: my-na	mespace		
5 labels:			
6 pod-secur	ity.kubernetes.io/warn:	restricted	
8			
9			
10			
11			
12			
13			
14			
15			
16			
17			
18			
19			
20			
21			
22			
23			
24 25			
26			
20			
28			
29			
30			
31			
32			
33			
34			
35			
36			
37			
38			
39			
40			
41			

- Infrastructure as Code (IaC)
- All resources are Kubernetes objects

#### 1 apiVersion: v1 2 kind: Namespace 3 metadata: name: my-namespace labels: pod-security.kubernetes.io/warn: restricted 8 apiVe<u>rsion: apps</u>/v1 9 kind: Deployment 10 metadata: name: my-nginx labels: app: nginx 14 spec: 15 replicas: 3 16 selector: matchLabels: app: nginx template: metadata: labels: app: nginx spec: containers: - name: nginx image: nginx:1.14.2 ports: - containerPort: 80

Application.yaml

- Infrastructure as Code (IaC)
- All resources are Kubernetes objects

#### 1 apiVersion: v1 2 kind: Namespace 3 metadata: name: my-namespace labels: pod-security.kubernetes.io/warn: restricted 8 apiVersion: apps/v1 9 kind: Deployment 10 metadata: name: my-nginx labels: app: nginx 14 spec: replicas: 3 selector: matchLabels: app: nginx template: metadata: labels: app: nginx Pod spec: containers: template - name: nginx image: nginx:1.14.2 ports: - containerPort: 80

Application.yaml

- Infrastructure as Code (IaC)
- All resources are Kubernetes objects

# Application.yaml 1 apiVersion: v1 2 kind: Namespace

pod-security.kubernetes.io/warn: restricted

- 8 apiVersion: apps/v1 9 kind: Deployment 10 metadata: name: my-nginx labels: app: nginx 14 spec: 15 replicas: 3 16 selector: matchLabels: app: nginx 19 template: metadata: labels: app: nginx spec: containers: - name: nginx image: nginx:1.14.2 ports: - containerPort: 80 30 apiVe<u>rsion: v1</u> 31 kind: Service 32 metadata: name: my-nginx-svc labels: app: nginx 36 spec: type: LoadBalancer
- 38 ports:

3 metadata:

labels:

Δ

name: my-namespace

- 39 port: 80
- 40 selector:
- 1 app: nginx

- Infrastructure as Code (IaC)
- All resources are Kubernetes objects
- Kubernetes takes care of creating and maintaining the desired state

apiVersion: v1	
kind: Namespace	
metadata:	
name: my-namespace	
labels:	
<pre>pod-security.kubernetes.io/warn:</pre>	restricted
apiVersion: apps/v1 kind: Deployment	
metadata:	
name: my-nginx	
labels:	
app: nginx	
spec:	
replicas: 3	
selector:	
matchLabels:	
app: nginx	
template:	
metadata:	
labels:	
app: nginx	
spec: containers:	
- name: nginx	
image: nginx:1.14.2	
ports:	
- containerPort: 80	
apiVersion: v1	
kind: Service	
metadata:	
name: my-nginx-svc	
labels:	
app: nginx	
spec:	

Application.yaml

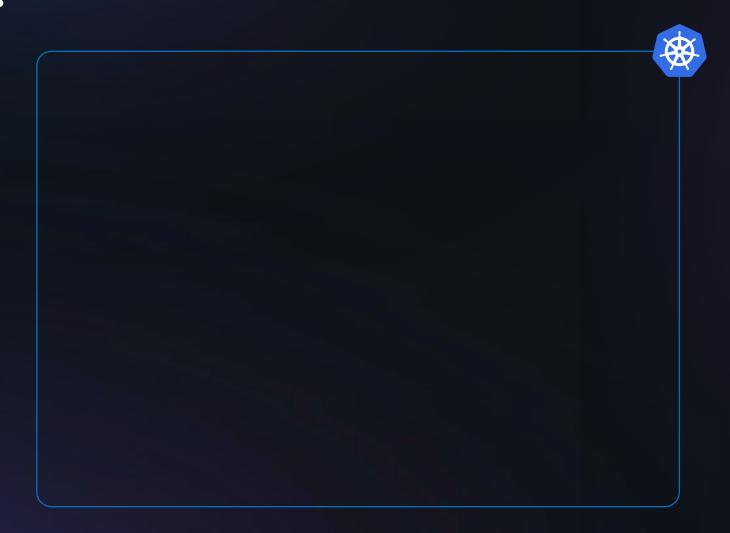
38 ports:

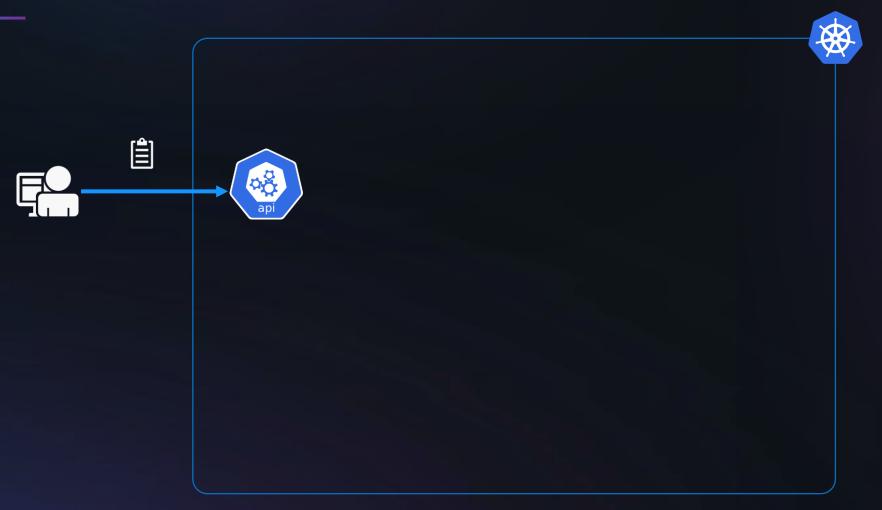
14

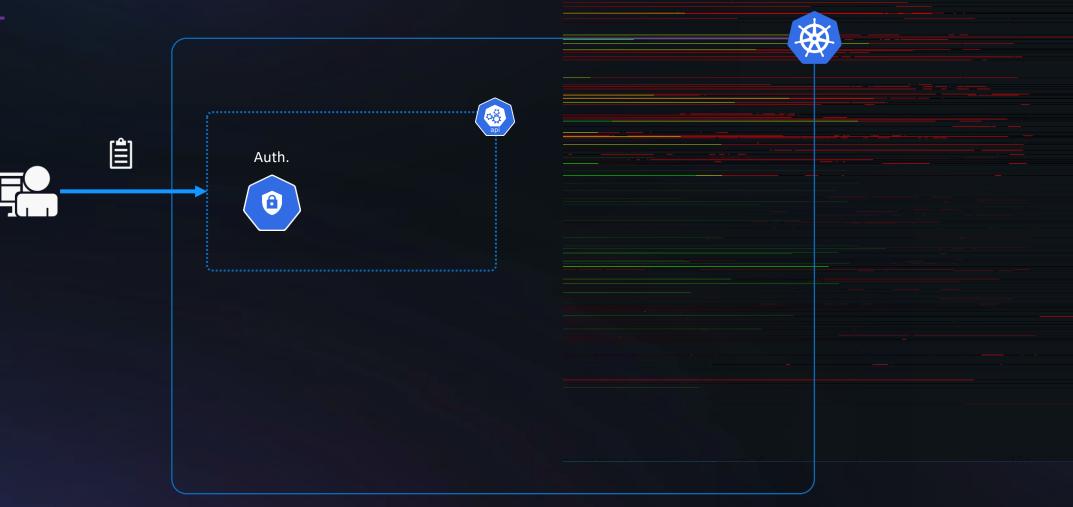
24 25

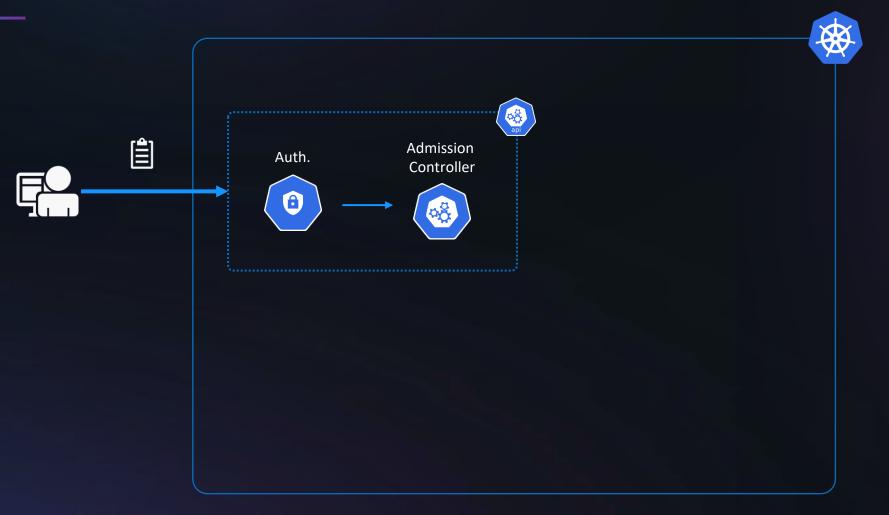
- 39 port: 80
- 40 selector:
- 1 app: nginx

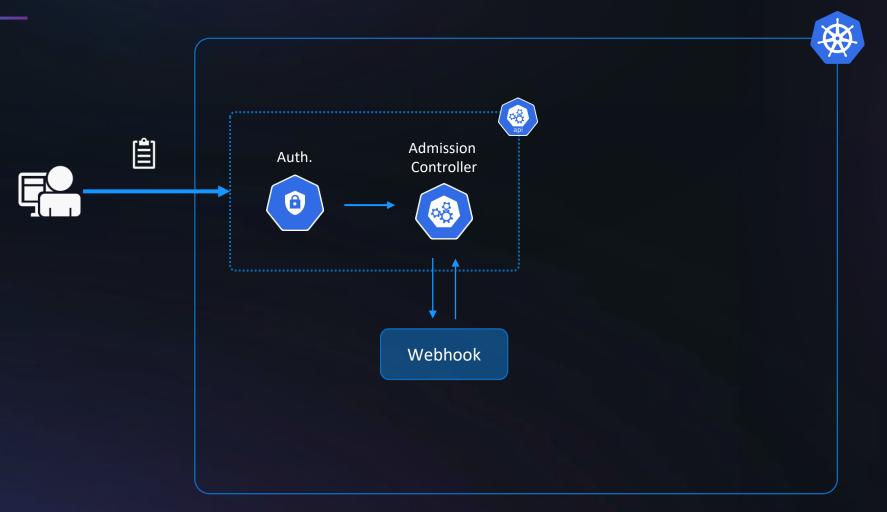


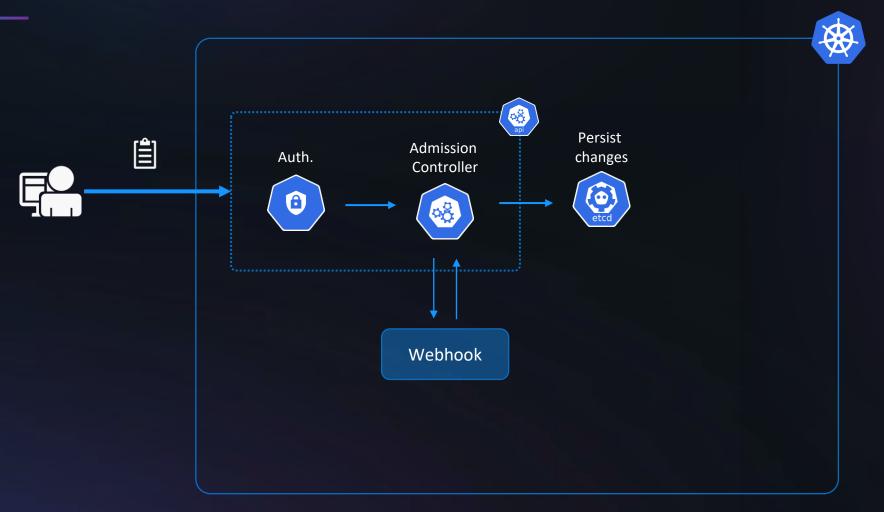


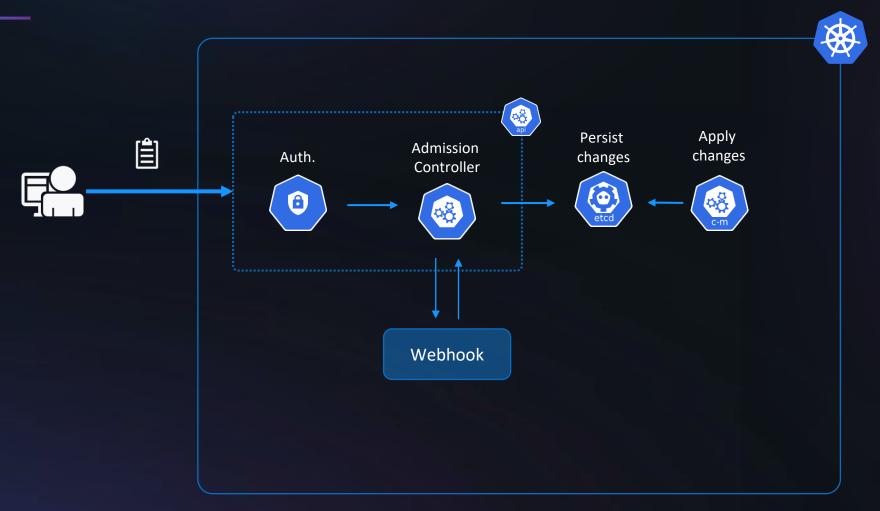


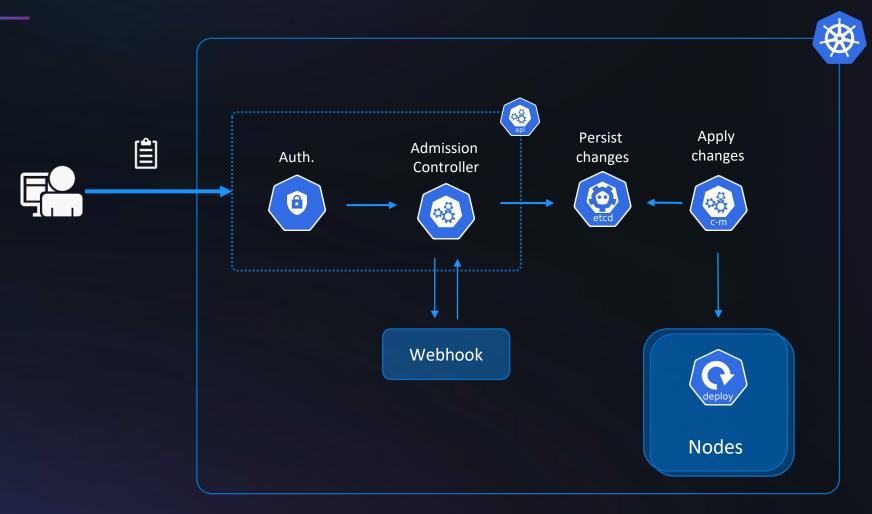












# **Kubernetes Security**

## **OWASP Kubernetes Top 10**

- 1. Insecure Workload Configuration
- 2. Supply Chain Vulnerabilities
- **3**. Overly Permissive RBAC Configurations
- 4. Lack of Centralized Policy Enforcement
- 5. Inadequate Logging and Monitoring
- 6. Broken Authentication Mechanisms
- 7. Missing Network Segmentation Controls
- 8. Secrets Management Failures
- 9. Misconfigured Cluster Components
- **10.** Outdated and Vulnerable Kubernetes Components



## Areas of Kubernetes Security

- Authentication & Authorization
- Pod Security
- Networking
- Workload Configuration
- Supply Chain

- Role-based access control (RBAC)
- Apply Principle of Least Privilege
- 3 types of subjects
  - User, Group (external)
  - ServiceAccount (internal)

- Role-based access control (RBAC)
- Apply Principle of Least Privilege
- 3 types of subjects
  - User, Group (external)
  - ServiceAccount (internal)

```
Pod.yaml
                                      -\Box \times
 1 apiVersion: rbac.authorization.k8s.io/v1
 2 kind: RoleBinding
 3 metadata:
     name: read-secrets
    namespace: development
 6 subjects.
     kind: User
     name: dave
    apigroup: rbac.authorization.k8s.io
10 roleRef:
    kind: ClusterRole
11
12
    name: secret-reader
13
    apiGroup: rbac.authorization.k8s.io
14 ---
15 apiVersion: rbac.authorization.k8s.io/v1
16 kind: ClusterRole
17 metadata:
    name: secret-reader
18
19 rules:
20 - apiGroups: [""]
21 resources: ["secrets"]
    verbs: ["get", "watch", "list"]
```

- Role-based access control (RBAC)
- Apply Principle of Least Privilege
- 3 types of subjects
  - User, Group (external)
  - ServiceAccount (internal)

Pod.yaml — 🗆 🗙
<pre>1 apiVersion: rbac.authorization.k8s.io/v1 2 kind: RoleBinding 3 metadata: 4   name: read-secrets 5   namespace: development 6 subjects: 7 - kind: User 8   name: dave 9   apiGroup: rbac.authorization.k8s.io 10 roleRef:</pre>
<pre>10 Folger: 11 kind: ClusterRole 12 name: secret-reader 13 apiGroup: rbac.authorization.k8s.io 14 15 apiVersion: rbac.authorization.k8s.io/v1 16 kind: ClusterRole 17 metadata: 18 name: secret-reader 19 rules: 20 - apiGroups: [""] 21 resources: ["secrets"] 22 verbs: ["get", "watch", "list"] 23</pre>

- Role-based access control (RBAC)
- Apply Principle of Least Privilege
- 3 types of subjects
  - User, Group (external)
  - ServiceAccount (internal)
- Authorization based on
  - API groups
  - Resources
  - Verbs

Pod.yaml $ \Box$ $\times$
<pre>1 apiVersion: rbac.authorization.k8s.io/v1 2 kind: RoleBinding 3 metadata:</pre>
4 name: read-secrets
5 namespace: development
6 subjects:
7 - kind: User
8 name: dave
<pre>9 apiGroup: rbac.authorization.k8s.io</pre>
10 roleRef:
11 kind: ClusterRole
12 name: secret-reader
13 apiGroup: rbac.authorization.k8s.io
14
<pre>15 apiVersion: rbac.authorization.k8s.io/v1</pre>
16 kind: ClusterRole
17 metadata:
18 name: secret-reader
19 rules:
20 – apiGroups: [""]
21 resources: ["secrets"]
<pre>22 verbs: ["get", "watch", "list"]</pre>
23

Pod.yaml — 🗌 🗙
1 apiVersion: v1
2 kind: Pod
3 metadata:
4 name: my-pod
5 spec:
<pre>6 securityContext:</pre>
7 runAsUser: 1000
8 readOnlyRootFilesystem: true
9 containers:
10 – name:
<pre>11 image: nginx:latest</pre>
<pre>12 securityContext:</pre>
13 runAsUser: 2000
14 allowPrivilegeEscalation: false
<pre>15 hostIPC: false</pre>
<pre>16 hostNetwork: false</pre>
17 hostPID: false

	Pod.yaml $ \Box$ $\times$
1 a	piVersion: v1
2 k	ind: Pod
3 m	etadata:
4	name: my-pod
5 s	pec:
6	securityContext:
7	runAsÜser: 1000
8	<pre>readOnlyRootFilesystem: true</pre>
9	containers:
10	- name:
11	<pre>image: nginx:latest</pre>
12	securityContext:
13	runAsÜser: 2000
14	allowPrivilegeEscalation: false
15	hostIPC: false
16	hostNetwork: false
17	hostPID: false

	Pod.yaml $ \Box$ $\times$
1 ap	piVersion: v1
2 ki	ind: Pod
3 me	etadata:
4	name: my-pod
5 sp	pec:
6	securityContext:
7	runAsUser: 1000
8	readOnlyRootFilesystem: true
9	containers:
10	- name:
11	<pre>image: nginx:latest</pre>
12	securityContext:
13	runAsUser: 2000
14	allowPrivilegeEscalation: false
15	hostIPC: false
16	hostNetwork: false
17	hostPID: false

Pod.yaml — 🗌 🗙
1 apiVersion: v1
2 kind: Pod
3 metadata:
4 name: my-pod
5 spec:
<pre>6 securityContext:</pre>
7 runAsUser: 1000
<pre>8 readOnlyRootFilesystem: true</pre>
9 containers:
10 – name:
11 <u>image: nginx:latest</u>
12 securityContext:
13 runAsUser: 2000
14 allowPrivilegeEscalation: false
15 hostIPC: false
<pre>16 hostNetwork: false</pre>
17 hostPID: false

- Pod Security Policy (PSP) replaced by Pod Security Standards (PSS)
  - Are a set of best-practice profiles for running pods securely [1]
  - are applied at the *Namespace* level when pods are created/modified

Pod.yaml — 🗆 🗙
1 apiVersion: v1
2 kind: Pod
3 metadata:
4 name: my-pod
5 spec:
<pre>6 securityContext:</pre>
7 runAsUser: 1000
<pre>8 readOnlyRootFilesystem: true</pre>
9 containers:
10 - name:
<pre>11 image: nginx:latest</pre>
<pre>12 securityContext:</pre>
13 runAsUser: 2000
<pre>14 allowPrivilegeEscalation: false</pre>
15 hostIPC: false
16 hostNetwork: false
17 hostPID: false

Application.yaml	- 🗆 ×
<pre>1 apiVersion: v1 2 kind: Namespace 3 metadata: 4  name: my-namespace 5  labels: 6     pod-security.kubernetes.io/warn: 7</pre>	restricted

• Kubernetes uses Container Network Interface (CNI) to create virtual networks

- Kubernetes uses Container Network Interface (CNI) to create virtual networks
- Default: all pods can communicate with each other

- Kubernetes uses Container Network Interface (CNI) to create virtual networks
- Default: all pods can communicate with each other
- Namespaces are **not** a separation mechanism!

- Kubernetes uses Container Network Interface (CNI) to create virtual networks
- Default: all pods can communicate with each other
- Namespaces are **not** a separation mechanism!
- Network Policies
  - Define rules how pods can communicate
  - Require support by CNI plugin





Calico

- Kubernetes uses Container Network Interface (CNI) to create virtual networks
- Default: all pods can communicate with each other
- Namespaces are **not** a separation mechanism!
- Network Policies
  - Define rules how pods can communicate
  - Require support by CNI plugin





Calico



• Service Meshes

• Generic configurations

Pod.yaml — 🗌	$\times$
1 apiVersion: v1	
2 kind: Pod	
3 metadata:	
4 name: my-app	
5 spec:	
6 automountServiceAccountToken: false	
7 serviceAccountName: dedicated-service-account	
8 containers:	Í
9 – image: nginx:latest	
10 name: app	
11 ports:	
12 - containerPort: 8080	
13 14	
15	
16	
17	
18	
19	
20	
21	
22	
23	
24	
25	
26	
27	
28	
29	
30 31	
32	
33	
34	

- Generic configurations
- Secrets Management
  - Not in environment variables!
  - ConfigMaps are not secure

	Pod.yaml $ \Box$ $\times$
	apiVersion: v1
2	kind: Pod
3	metadata:
4	name: my-app
5	spec:
6	automountServiceAccountToken: false
7	<pre>serviceAccountName: dedicated-service-account</pre>
8	env:
9	- name: SECRET_PASSWORD
10	valueFrom:
11	secretkeyRef:
12	name: mysecret
13 14	key: password
14 15	<pre>containers: - image: nginx:latest</pre>
16	name: app
17	ports:
18	– containerPort: 8080
19	
20	
21	
22	
23	
24	
25	
26	
27	
28	
29	
30 21	
31 32	
32 33	
34	
54	

- Generic configurations
- Secrets Management
  - Not in environment variables!
  - ConfigMaps are not secure
- Use of labels/annotations <sup>[1]</sup>
  - Configuration of some features
  - Useful semantic information

	Pod.yaml $ \Box$ $\times$
1 ai	piVersion: v1
	ind: Pod
	etadata:
4	labels:
5	<pre>app.kubernetes.io/part-of: webshop</pre>
6	app.kubernetes.io/managed-by: owner
7	name: my-app
	pec:
9	automountServiceAccountToken: false
10	<pre>serviceAccountName: dedicated-service-account</pre>
11	env:
12	- name: SECRET_PASSWORD
13	valueFrom:
14 15	secretKeyRef:
15	name: mysecret
17	key: password containers:
18	<pre>- image: nginx:latest</pre>
19	name: app
20	ports:
21	- containerPort: 8080
22	
23	
24	
25	
26	
27	
28	
29	
30	
31	
32	
33	
34	

- Generic configurations
- Secrets Management
  - Not in environment variables!
  - ConfigMaps are not secure
- Use of labels/annotations <sup>[1]</sup>
  - Configuration of some features
  - Useful semantic information
- Reliability
  - Resource Requests/ Limits
  - Readiness-/ Liveness-Probe

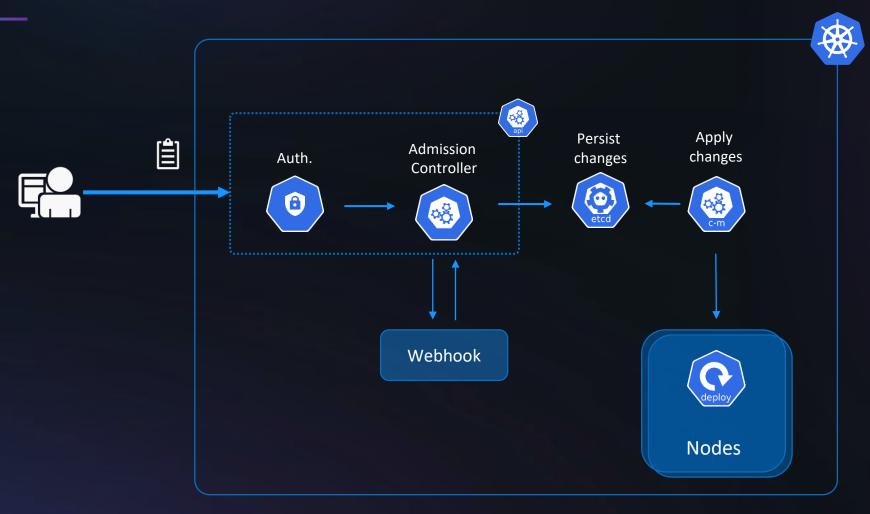
	Pod.yaml $-\Box \times$
	apiVersion: v1
	kind: Pod
3	metadata:
4	labels:
5	<pre>app.kubernetes.io/part-of: webshop</pre>
6	app.kubernetes.io/managed-by: owner
7	
8	
9	automountServiceAccountToken: false
10	<pre>serviceAccountName: dedicated-service-account</pre>
11	env:
12	
13 14	
15	name: mysecret
16	key: password
17	
18	<pre>- image: nginx:latest</pre>
19	name: app
20	ports:
21	_ containerPort: 8080
22	readinessProbe:
23	httpGet:
24	<pre>path: /ready</pre>
25	port: 8080
26	resources:
27	limits:
28	cpu: 1m
29	ephemeral-storage: 1Mi
30	memory: 1Mi
31	requests:
32 33	cpu: 1m enhomenal storage: 1Mi
33 34	ephemeral-storage: 1Mi memory: 1Mi
54	memory. Int

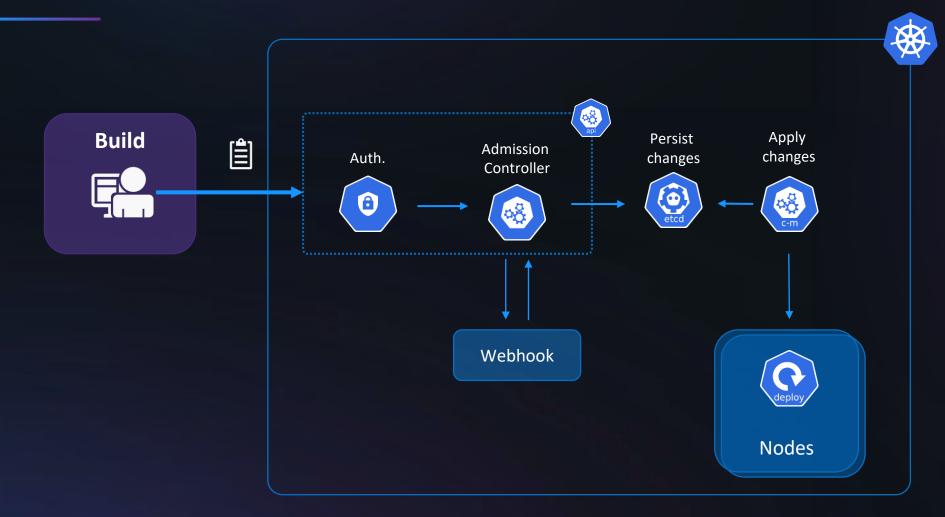
# Supply Chain

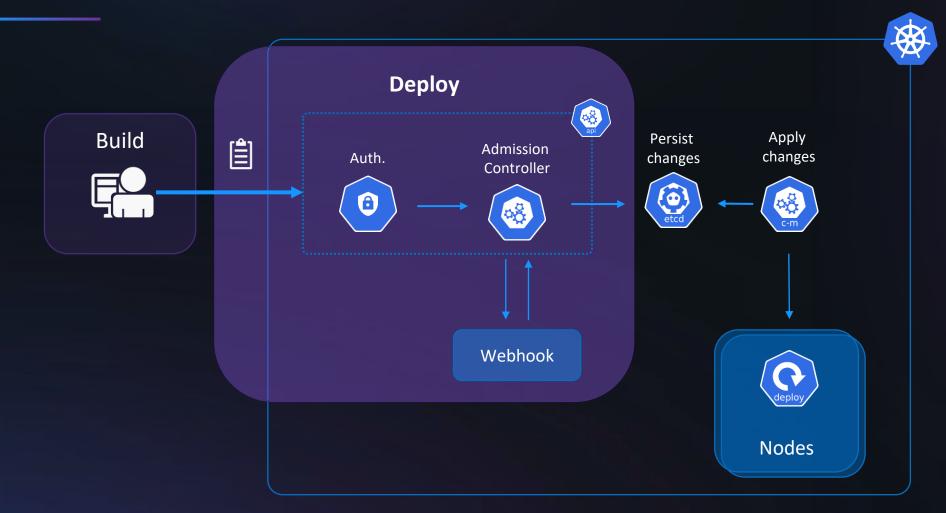
- Primarily about containers
  - Software composition (SBOM)
  - Vulnerabilities
- Use admission controller to check:
  - Limit to trusted registries
  - Image Verification (3<sup>rd</sup> Party)

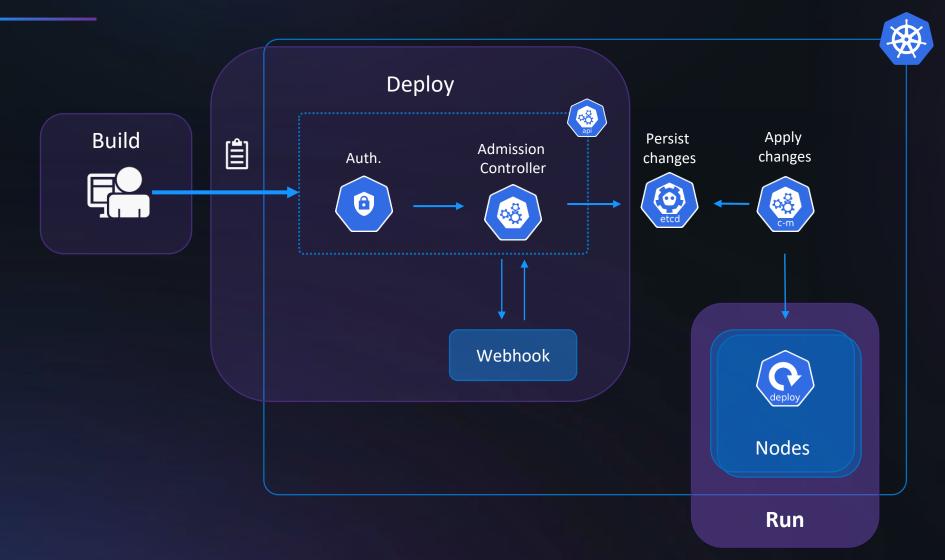
<u>入</u> cosign











#### Problems

- Cloud environments move very fast
- Kubernetes is designed to just work
  - implies security defaults are very lenient
- Kubernetes is evolving quickly
  - <u>3 release per year</u>
  - Hard to keep track of attack vectors <sup>[1]</sup>

# Compliance

# **Compliance Frameworks**

- Regulatory standards
  - NIST
  - GDPR
  - SOC2
  - PCI DSS
  - HIPAA
  - ...



#### **Compliance Frameworks**

- Regulatory standards
  - NIST
  - GDPR
  - SOC2
  - PCI DSS
  - HIPAA
  - •••
- Guides and Benchmarks
  - **CIS Benchmarks**
  - NSA / CISA Kubernetes Hardening Guide
  - <u>Kubernetes STIG</u> DISA
  - <u>Kubernetes Security Checklist</u>



#### **CIS Benchmarks**

- Center for Internet Security
- Are configuration baselines and best practices for securely configuring a system
- Controls map to many established standards and regulatory frameworks
- Dedicated for various technologies, e.g.
  - Docker
  - Kubernetes (for various clouds)

	On premi					Connoanio	Chitos Cloud	indations
Responsibility		292	25	23°2 26	25 28	25	*Uide Ud	arks ons
Data classification and accountability							$\checkmark$	$\checkmark$
Client and end-point protection							$\checkmark$	$\checkmark$
Identity and access management							$\checkmark$	$\checkmark$
Application-level controls							$\checkmark$	$\checkmark$
Network controls							$\checkmark$	$\checkmark$
Host infrastructure							$\checkmark$	
Physical security								
		e Clou	id Custo	mer 🧲	Cloud P	rovider		

Source: CIS: Shared Responsibility for Cloud Security: What You Need to Know

## **CIS Kubernetes Benchmark**

- 1. Control Plane Components
  - Master Node Configuration Files
  - API Server
  - Controller Manager
  - Scheduler
- 2. etcd
- 3. Control Plane Configuration
  - Authentication and Authorization
  - Logging
- 4. Worker Nodes
  - Worker Node Configuration Files
  - Kubelet
- 5. Policies
  - RBAC and Service Accounts
  - Pod Security Standards
  - Network Policies and CNI
  - Secrets Management
  - Extensible Admission Control
  - General Policies

## **CIS Kubernetes Benchmark**

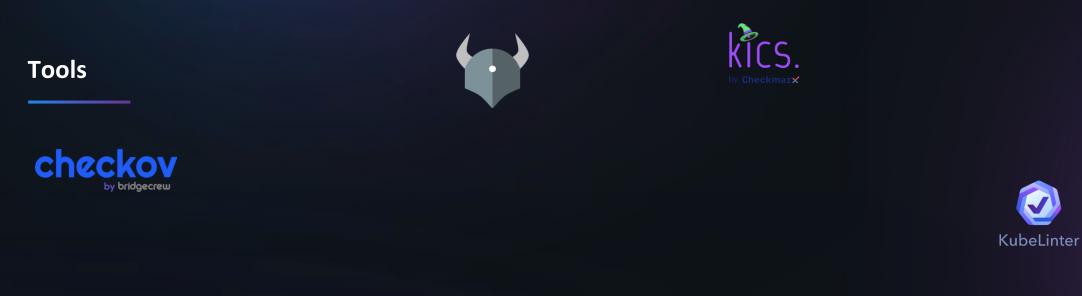
- 1. Control Plane Components
  - Master Node Configuration Files
  - API Server
  - Controller Manager
  - Scheduler
- 2. etcd
- **3.** Control Plane Configuration
  - Authentication and Authorization
  - Logging
- 4. Worker Nodes
  - Worker Node Configuration Files
  - Kubelet
- 5. Policies
  - RBAC and Service Accounts
  - Pod Security Standards
  - Network Policies and CNI
  - Secrets Management
  - Extensible Admission Control
  - General Policies

#### Infrastructure

Workload





















# **KALM Benchmark**

#### **KALM Benchmark**

- Open-Source project (on <u>Github</u>)
- Focus on workload misconfiguration
- Goal:
  - Have a benchmark for comparing tools/scanners
  - Help decision makers find the right fit
  - Help with the development/adjustment of the rule-set

#### • 2 parts

- Benchmark manifests itself
- Tool to analyze results

#### Overview

									Scope		Category				
Scanner	Image	↓ Score	Coverage	CI M	Runs	Cust	Report Formats	is	Scan	Scan	Misc	Net	Pod	Psp	RI
ocs -		67.3%	51.9%	M	2	<b>2</b>	Plain, JSON, Sarif, Cyclone	true	•	×		1/6	61/9	11/31	3
rivy	2	54.5%	41.0%	<b>V</b>	<b>V</b>	🖌 in	JSON, Table, Sarif, Templat	true	<b>v</b>	×	0/0	0/6	63/98	0/31	2
lheckov	checkov	52.7%	37.2%	2	<b>V</b>	<b>V</b>	Plain, JSON, CycloneDX, Ju	true	2	<b>V</b>	0/0	0/6	56/98	11/31	6
inyk	👹 snyk	38.6%	23.4%	2	×	🗹 in	Plain, JSON, SARIF	true	<b>V</b>	х	0/0	0/6 (+5)	19/9	9/31	2
kubeaudit		35.4%	21.3%	<b>V</b>	<b>V</b>	🖌 Б	Plain, JSON, logrus	true	<b>V</b>	<b>V</b>	0/0	0/6 (+5)	47/9	0/31 (	0
(ubescape	<u>0</u>	35.3%	22.3%	x	🗹 ar	<b>V</b>	Plain, JSON, JUnit, Prometh	true	<b>V</b>	<b>V</b>	0/0	0/6 (+5)	42/1	0/31	0
ubeLinter	🐼 KubeLint	33.3%	20.5%	2	<b>V</b>	🖌 Б	Plain, JSON, SARIF	true	<b>V</b>	×	0/0 (+1)	1/6 (+5)	28/9	0/31	3
ferrascan	© terroscori	29.9%	21.3%	<b>V</b>	<b>V</b>	🗹 in	Plain, JSON, YAML, SARIF,	true	<b>V</b>	х	0/0 (+1)	0/6 (	34/98	3/31 (	0
polaris	pelaris	19.1%	28.9%	<b>V</b>	<b>V</b>	🖌 as	Pretty, JSON, YAML	true	<b>V</b>	×	0/0	0/6	53/9	0/31	0
kube-score	30	15.9%	19.2%	<b>V</b>	2	×	Plain, JSON, SARIF, CI	true	2	×	0/0 (+1)	1/5 (+2)	25/9	0/31	-
Datree	9	15.0%	9.6%	V	🗹 w	🗹 w	Plain, JSON, JUnit, YAML, XML	true	2	×	0/0	0/6	8/98	0/31	(
kubiscan	۲	12.9%	5.9%	х	<b>V</b>	×	Pretty	true	×	<b>V</b>	0/0	0/6	0/98	0/31	1
kubesec	0	1.0%	0.4%	<b>v</b>	<b>V</b>	×	JSON, Template	true	<b>V</b>	×	0/0	0/6	1/98	0/31	(
kube-ben		0.0%	0.0%	<b>V</b>	×	×	Plain, JSON, JUnit, Postgre	true	×	~	0/0	0/6	0/98	0/31	

💡 to get a description of a column hover over the column name

Show Details for KICS

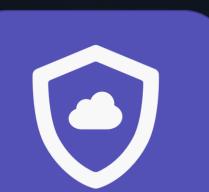
#### **KALM Benchmark**

- Normalized checks across benchmarks and tools
- Design
  - Best practice manifest per default
  - Only 1 explicit misconfiguration per check
- Multiple categories
- Contains > 250 manifests

• *Note: not 100% coverage* 



# Takeaways



Automation is indispensable for cloud security



Lots of **tools** available to achieve compliance

KALM Benchmark can help to pick the right tools

# dynatrace

dynatrace.com