Twistlock™

Basic Security for Deployment

Subtitle if applicable

kubernetes

Presenter’s Name

Presenter’s Title, Twistlock

What We'll Cover Today

- Quick overview of securing a host
- Secure deployment and configuration of Kubernetes
- Shift-Left Container Security
- Q&A

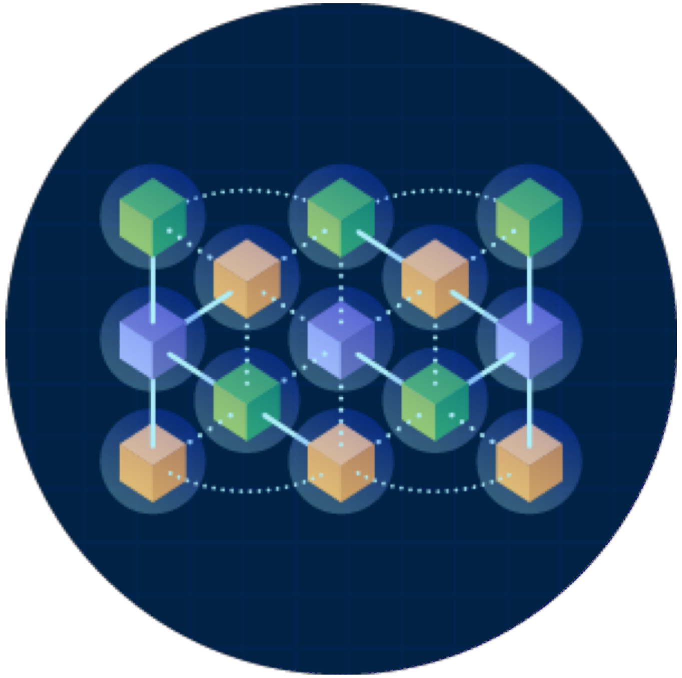
Twistlock

Secure the host

- Minimal install of the OS
- Update your packages
- Remove unneeded services
- Close any open ports you don’t need
- Secure login by certificates
- Intrusion detection systems

Secure Kubernetes

- RBAC
- Layer 3 Protection
- Configure the Master Nodes
- Configure the Worker Nodes
- ETCD
- Secrets Injection



Kubernetes RBAC

- White list security model that is additive
- Role / RoleBinding
- ClusterRole / ClusterRole Binding
- Limit access to your nodes
- Access to containers in the cluster should be done through kubectl

Network Security

- Layer 3 segmentation
- Kubernetes Network Policy
- Realtime network profiling
- Automatic rule creation

Kubernetes and Docker CIS Benchmarks

- Apply to the Docker Engine
- Apply to the Master Nodes
- Apply to the Worker Nodes

CIS Master Node Configuration

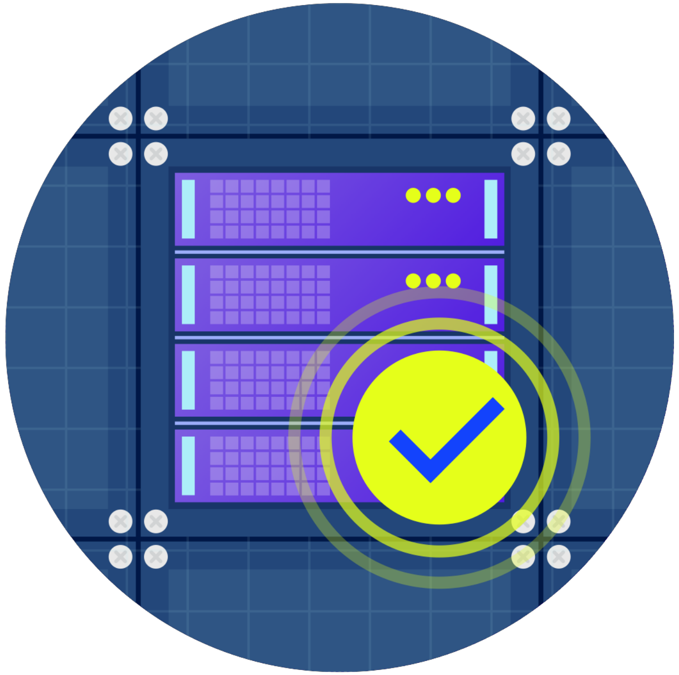
- Configuration changes can be made in apiserver.yaml
- Examples:
  - anonymous-auth
  - insecure-bind-address
- The recommendations also apply to the cluster Federation api-server as well

CIS Worker Node Configuration

- Ensure that the kubelet (Kubernetes Agent) is configured
- Examples:
    - allow-privileged set to 0
    - Disable cAdvisor
    - Ensure proper permissions and ownership to configuration files

Lock down ETCD Server

- It holds a lot of sensitive cluster information
- Unauthenticated so it needs to be protected
- It is recommended that it is placed behind its own firewall



Kubernetes Secret Injection

- Do not include secrets in the images
- Inject them at startup as files or environment variables
- Rotate secrets often
- Ensure secrets are encrypted at rest
- Encrypt them from view when a pod is inspected

What else can we do?

- Secure the images
- Secure the containers
- Secure the entire CI/CD pipeline

Minimal

Typically single process entities

Declarative

Built from images that are machine readable

Predictable

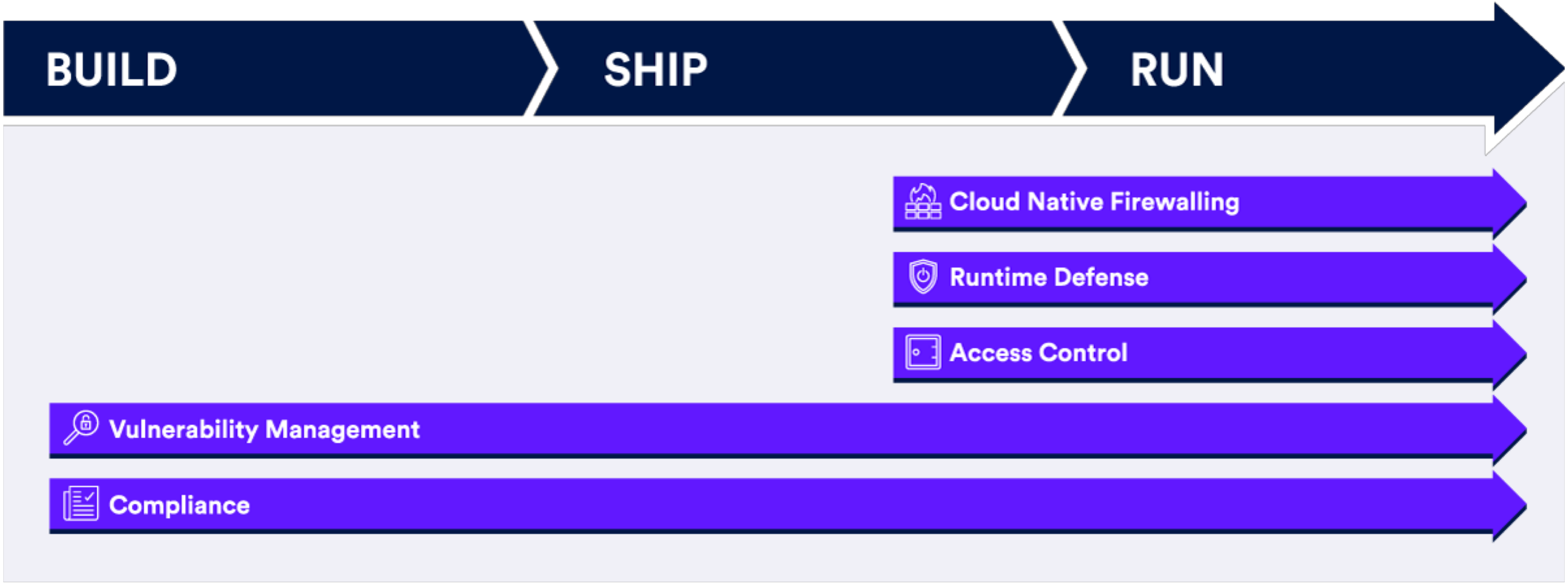
Do exactly the same thing from **`run`** to **`kill`**

1. The minimal nature simplifies security requirements for each artifact

2. The declarative nature allows automated analysis of vulnerabilities and compliance

3. The predictable nature simplifies automation of policy creation and enforcement

Security for the entire CI/CD Pipeline



Thank You

Any Questions?