svDevOps

Zero Trust Networking

Why Security shouldn't always be tied to your network

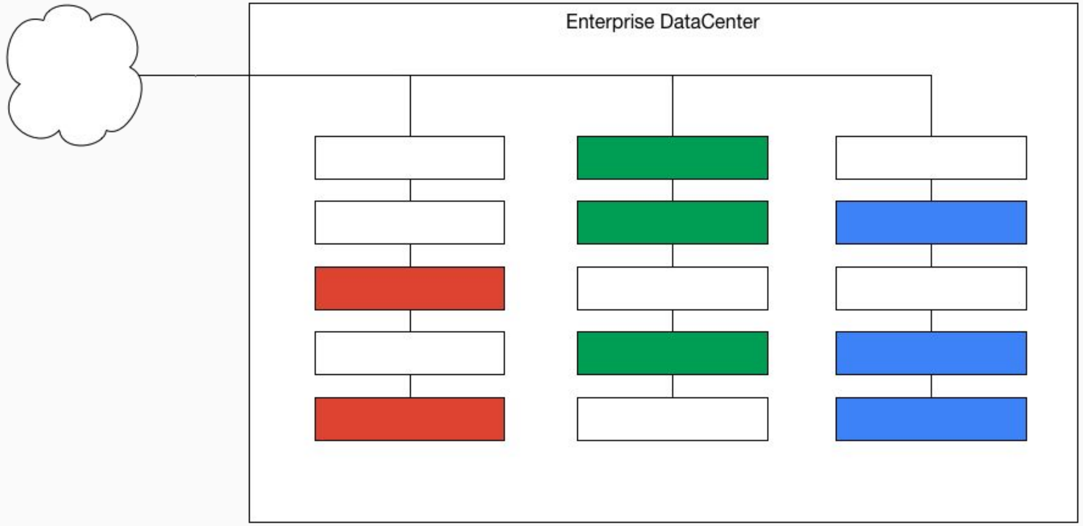
Bernard Van De Walle, Aporeto

TRIREME
APORETO OPEN SOURCE

Aporeto

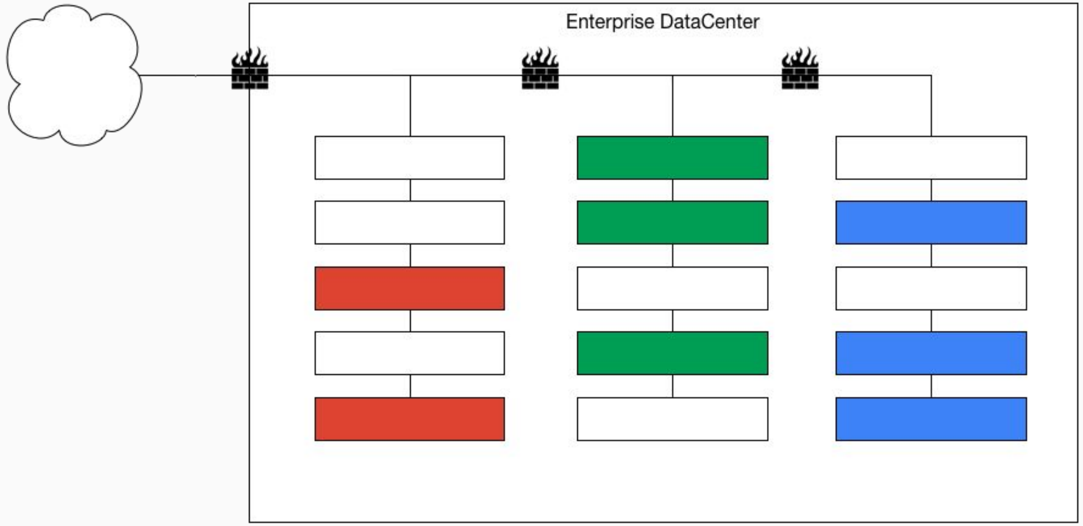
@bvandewa

Security levels

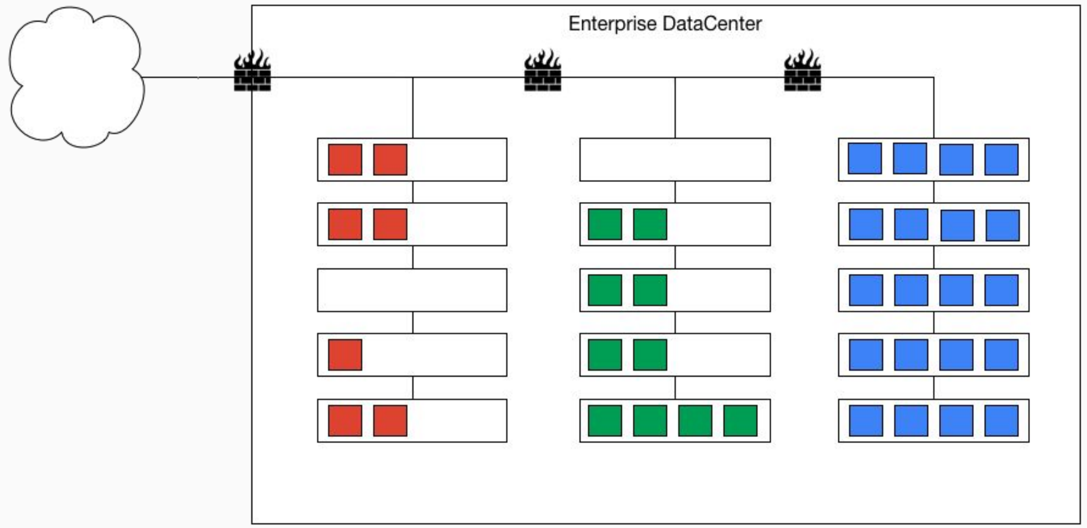


Perimeter security

```
# iptables -I INPUT -s 20.0.0.0/8 -j ALLOW
# iptables -I INPUT -s 10.20.0.0/16 -j DROP
# iptables -I INPUT -s 30.0.0.0/8 -j ALLOW
# iptables -I INPUT -s 10.0.0.0/8 -j ALLOW
# iptables -I INPUT -s 0.0.0.0/0 -j DROP
```

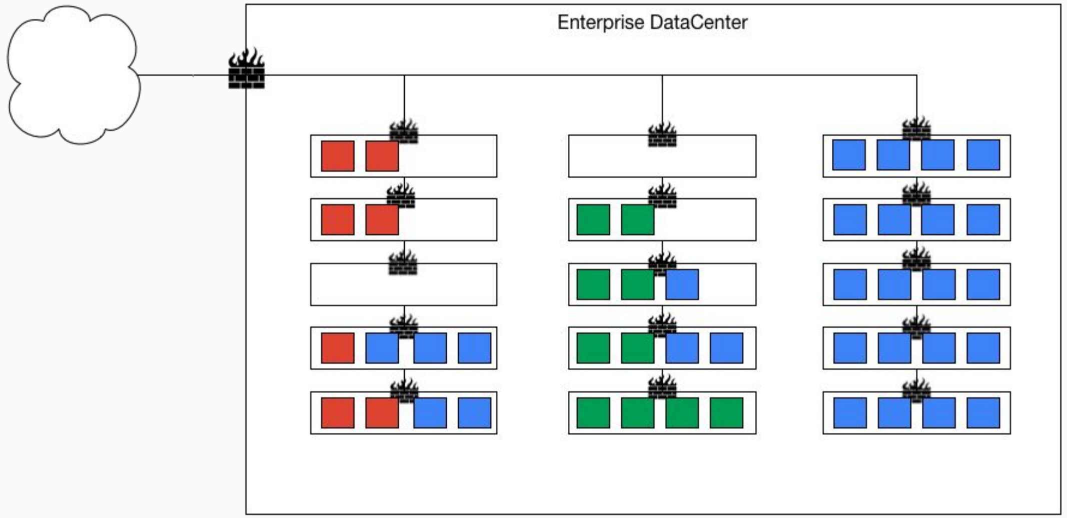


Micro-Services



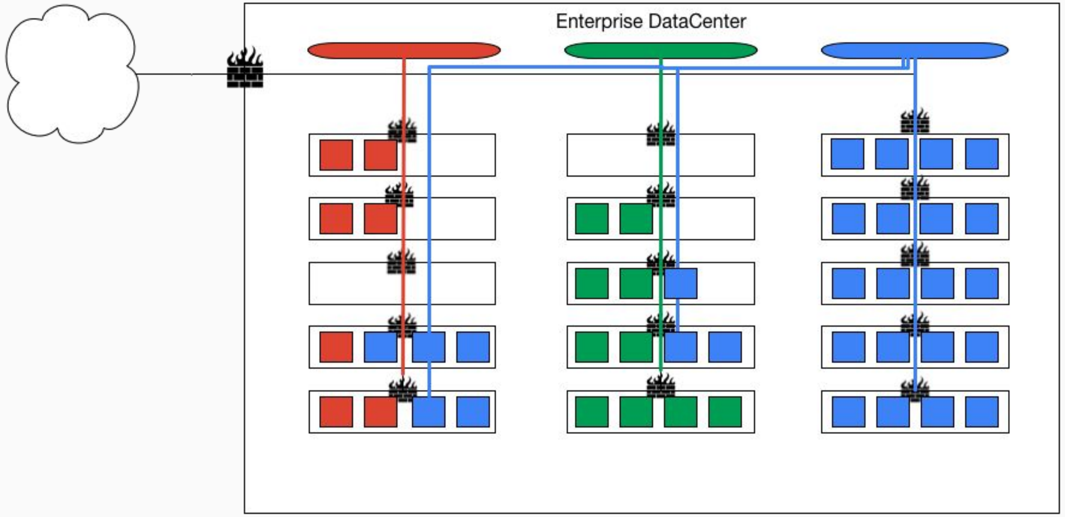
Distributed firewalls

```
# iptables -I INPUT -s 20.10.15.54/32 -j ALLOW
# iptables -I INPUT -s 30.15.21.64/32 -j ALLOW
# iptables -I INPUT -s 20.10.16.54/32 -j ALLOW
# iptables -I INPUT -s 30.15.28.64/32 -j ALLOW
# iptables -I INPUT -s 20.10.13.57/32 -j ALLOW
# iptables -I INPUT -s 30.15.24.64/32 -j ALLOW
# iptables -I INPUT -s 20.10.15.55/32 -j ALLOW
# iptables -I INPUT -s 30.15.26.64/32 -j ALLOW
# iptables -I INPUT -s 20.10.1s.54/32 -j ALLOW
# iptables -I INPUT -s 30.15.28.64/32 -j ALLOW
# iptables -I INPUT -s 20.10.35.53/32 -j ALLOW
# iptables -I INPUT -s 30.15.21.61/32 -j ALLOW
[....
....
...]
# iptables -I INPUT -s 0.0.0.0/0 -j DROP
```

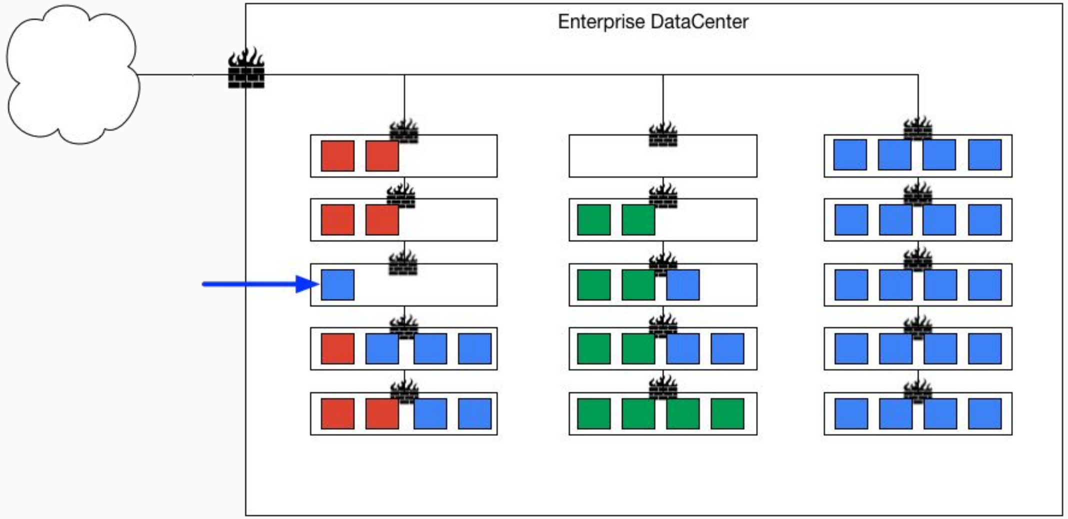


SDN and VPN solutions

```
# iptables -I INPUT -s 20.10.15.54/32 -j ALLOW
# iptables -I INPUT -s 30.15.21.64/32 -j ALLOW
# iptables -I INPUT -s 20.10.16.54/32 -j ALLOW
# iptables -I INPUT -s 30.15.28.64/32 -j ALLOW
# iptables -I INPUT -s 20.10.13.57/32 -j ALLOW
# iptables -I INPUT -s 30.15.24.64/32 -j ALLOW
# iptables -I INPUT -s 20.10.15.55/32 -j ALLOW
# iptables -I INPUT -s 30.15.26.64/32 -j ALLOW
# iptables -I INPUT -s 20.10.1s.54/32 -j ALLOW
# iptables -I INPUT -s 30.15.28.64/32 -j ALLOW
# iptables -I INPUT -s 20.10.35.53/32 -j ALLOW
# iptables -I INPUT -s 30.15.21.61/32 -j ALLOW
[....
....
...]
# iptables -I INPUT -s 0.0.0.0/0 -j DROP
```

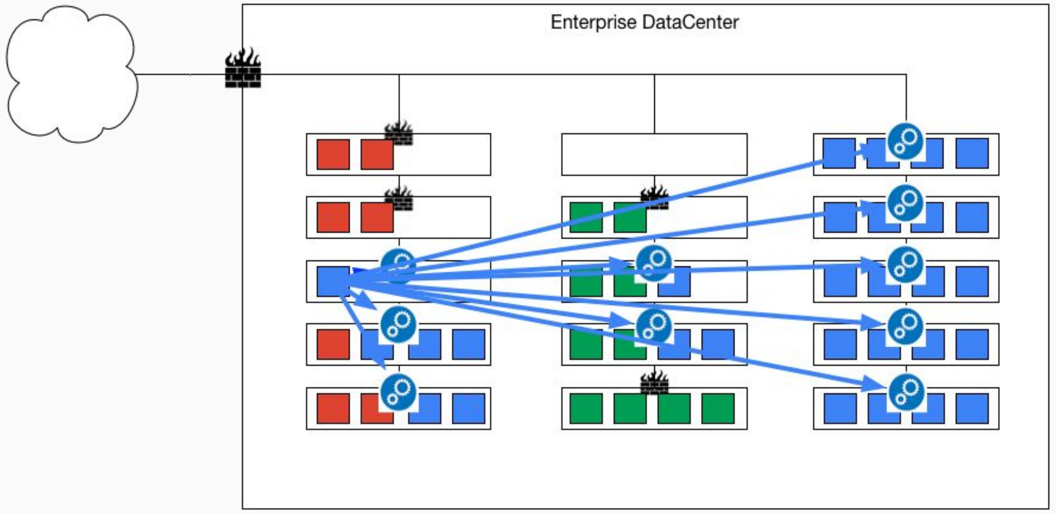


Provisioning assets



Exponential Complexity

```
# iptables -I INPUT -s 20.10.15.54/32 -j ALLOW
# iptables -I INPUT -s 30.15.21.64/32 -j ALLOW
# iptables -I INPUT -s 20.10.16.54/32 -j ALLOW
# iptables -I INPUT -s 30.15.28.64/32 -j ALLOW
# iptables -I INPUT -s 20.10.13.57/32 -j ALLOW
# iptables -I INPUT -s 30.15.24.64/32 -j ALLOW
# iptables -I INPUT -s 30.15.26.64/32 -j ALLOW
# iptables -I INPUT -s 20.10.1s.54/32 -j ALLOW
# iptables -I INPUT -s 30.15.28.64/32 -j ALLOW
# iptables -I INPUT -s 20.10.35.53/32 -j ALLOW
# iptables -I INPUT -s 30.15.21.61/32 -j ALLOW
[....
....
...]
# iptables -I INPUT -s 0.0.0.0/0 -j DROP
```



Network

≠

Network security

Zero Trust Networking

Network is insecure by default

Threat model: inside network as insecure as outside network

Network primitives are irrelevant

IP and Port numbers do not carry any information

Flows need to be authorized

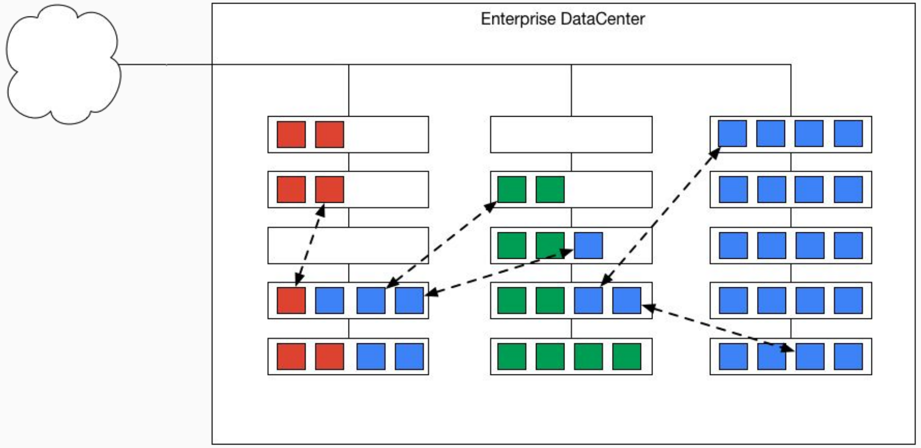
Every connection results from a successful authorization/authentication

Declarative policy language

High-level language to automate policy creation/deployment
(Yet Another Policy Language)

Zero Trust Networking

- Context and Identity used for flow authentication
- Network identity ≠ Endpoint identity
- Secure by default
- Keep the network **simple**



Kubernetes

Zero-Trust networking in Kubernetes

Kubernetes Networking (reachability)

- Based on CNI
- Built-in (GKE, ...) or plugin based
- IP doesn’t carry any information

Kubernetes objects

- Associated Identity
  - Name
  - Namespace
  - Labels

```
apiVersion: v1
kind: Pod
metadata:
  name: external
  namespace: demo
  labels:
    role: external
    app: nginx
spec:
  [...]
```

Kubernetes network policies

- White list model
- No default implementation
- Ingress only

```
apiVersion: extensions/v1beta1
kind: NetworkPolicy
metadata:
  name: backend-policy
  namespace: demo
Spec:
  [...]
```

Kubernetes network policies

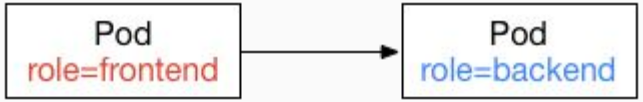
- Explicit activation per **namespace**
- Annotation for activation

```
kind: Namespace
metadata:
  name: demo
  Annotations:
     net.beta.kubernetes.io
     /network-policy: |
      {
        "ingress": {
         "isolation":"DefaultDeny"
        }
      }
```

Kubernetes network policies

- Rules apply to specific Pods
- Pods selected based on labels

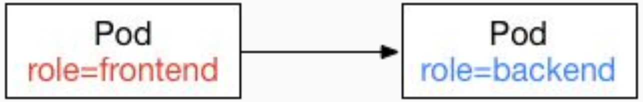
`role=backend`



```
apiVersion: extensions/v1beta1
kind: NetworkPolicy
spec:
  podSelector:
    matchLabels:
      role: backend
  ingress:
    - from:
      - podSelector:
        matchLabels:
          role: frontend
```

Kubernetes network policies

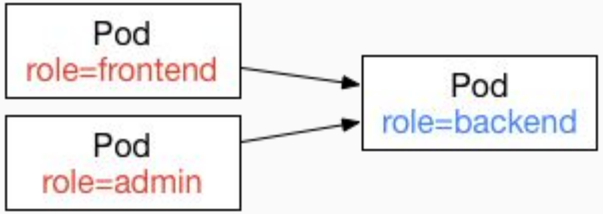
- Rule defines Pods allowed to **send traffic**
- Allowed traffic selected based on labels `role=frontend`



```
apiVersion: extensions/v1beta1
kind: NetworkPolicy
spec:
  podSelector:
    matchLabels:
      role: backend
  ingress:
    - from:
      - podSelector:
        matchLabels:
          role: frontend
```

Kubernetes network policies

- Rules are additive
- Each rule allows additional traffic



```
apiVersion: extensions/v1beta1
kind: NetworkPolicy
spec:
  podSelector:
    matchLabels:
      role: backend
  ingress:
    - from:
      - podSelector:
        matchLabels:
          role: admin
```

Implementations

Tied to networking backend

Because Policing is based on IPs

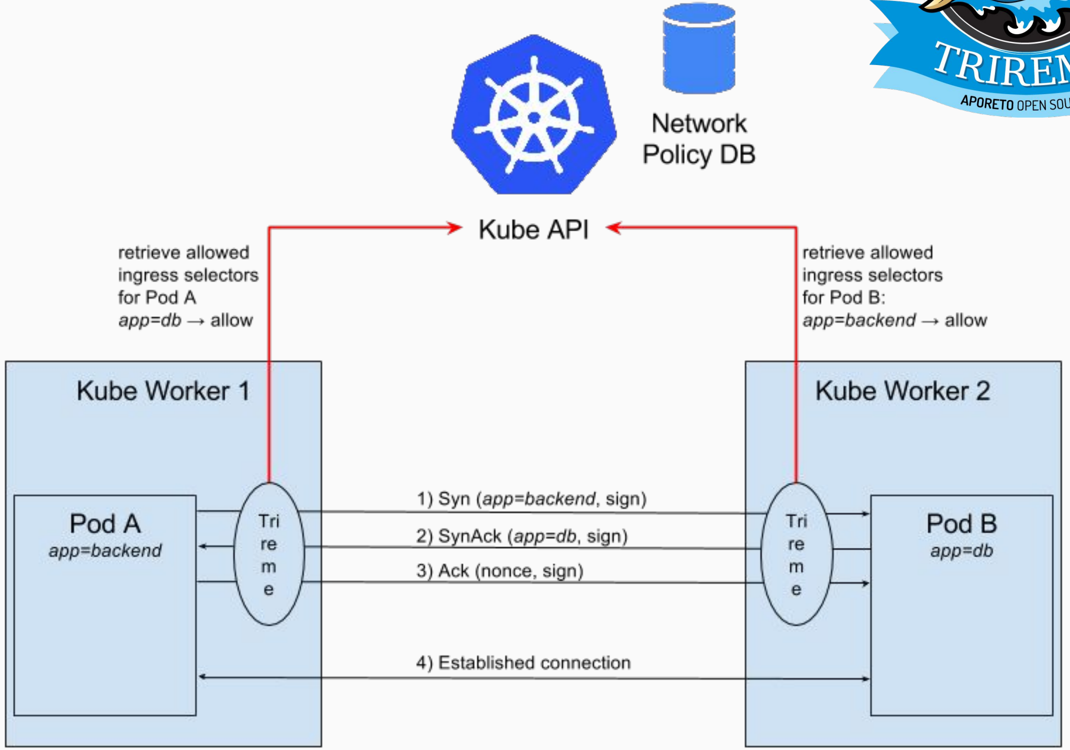
Trireme

https://github.com/aporeto-inc/trireme-kubernetes

- Identity is the pod label
- **IP** irrelevant. Network independent
- Compatible with any Networking backend

E2E authentication

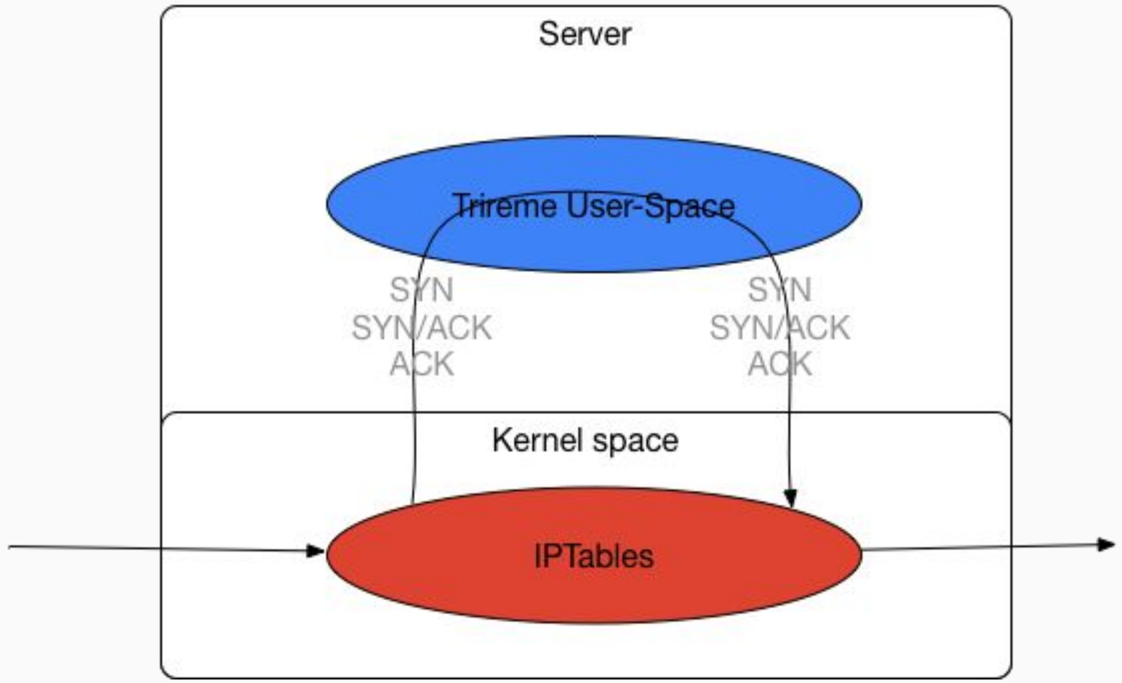
- Identity added on TCP flows handshake
- Identity signed


Network Policy DB
Kube API
retrieve allowed ingress selectors for Pod A *app=db* → allow
retrieve allowed ingress selectors for Pod B: *app=backend* → allow
Kube Worker 1
Kube Worker 2
Pod A *app=backend*
Pod B *app=db*
Tri re m e
Tri re m e
1) Syn (*app=backend*, sign)
2) SynAck (*app=db*, sign)
3) Ack (nonce, sign)
4) Established connection

TRIREME
APORETO OPEN SOURCE

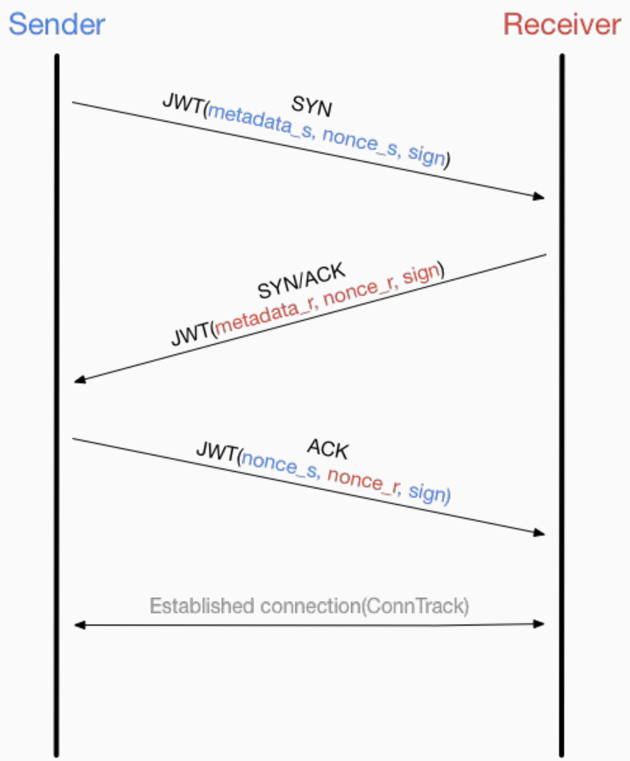
IPTables

- LibNetFilter_Queue
- Redirect to user-space
- Attach **endpoint identity** in user-space



TCP Handshake

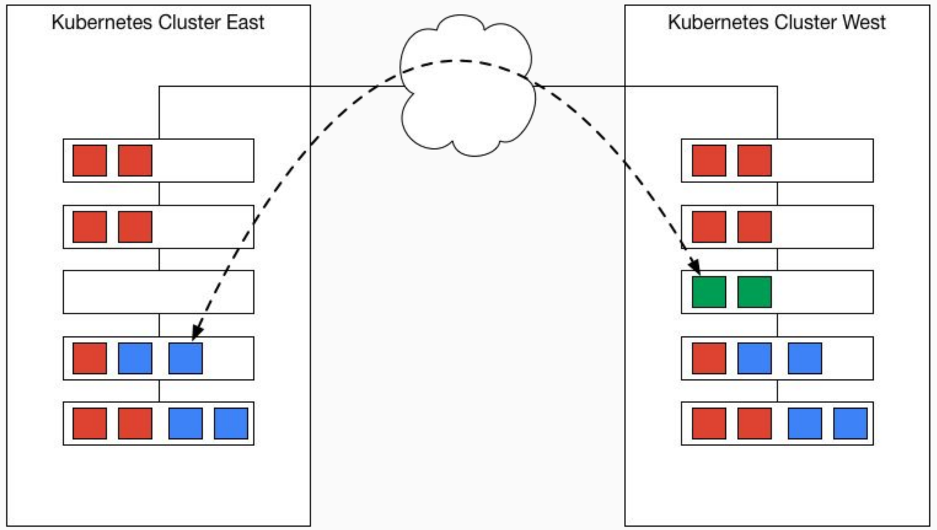
- Sender/Receiver add metadata
- sign and nonces to avoid replay//MITM


Sender
Receiver
SYN
JWT(metadata_s, nonce_s, sign)
SYN/ACK
JWT(metadata_r, nonce_r, sign)
ACK
JWT(nonce_s, nonce_r, sign)
Established connection(ConnTrack)

“Demo Time”

Cluster federation

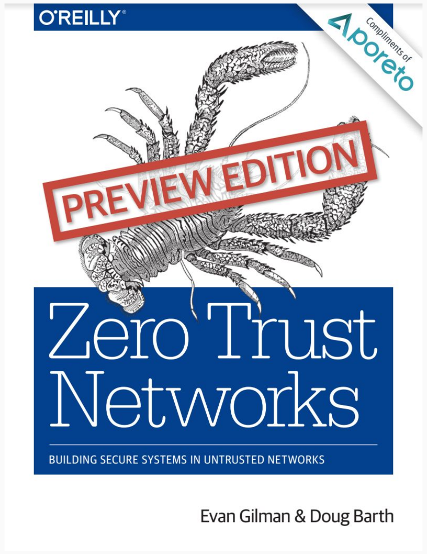
With Zero-Trust Networking



Network reachability
$\neq$
Network security

More about zero-trust

- Encryption
- Visibility
- Auditing



www.aporeto.com/oreilly

Thanks!

@bvandewa

Trireme on Github:
https://github.com/aporeto-inc/trireme-kubernetes

Demo code and slides:
https://github.com/bvandewalle/kubecon-zerotrust