KubeCon | CloudNativeCon

Europe 2022

WELCOME TO VALENCIA

KubeCon CloudNativeCon Europe 2022

Full Mesh Encryption in Kubernetes with WireGuard and Calico

Peter Kelly, Tigera

About Me

- Tigera (S24)
- Cork, Ireland
- First time in Valencia
- First time speaking at KubeCon
- Not a security researcher!

Peter Kelly
Director of Engineering
Tigera

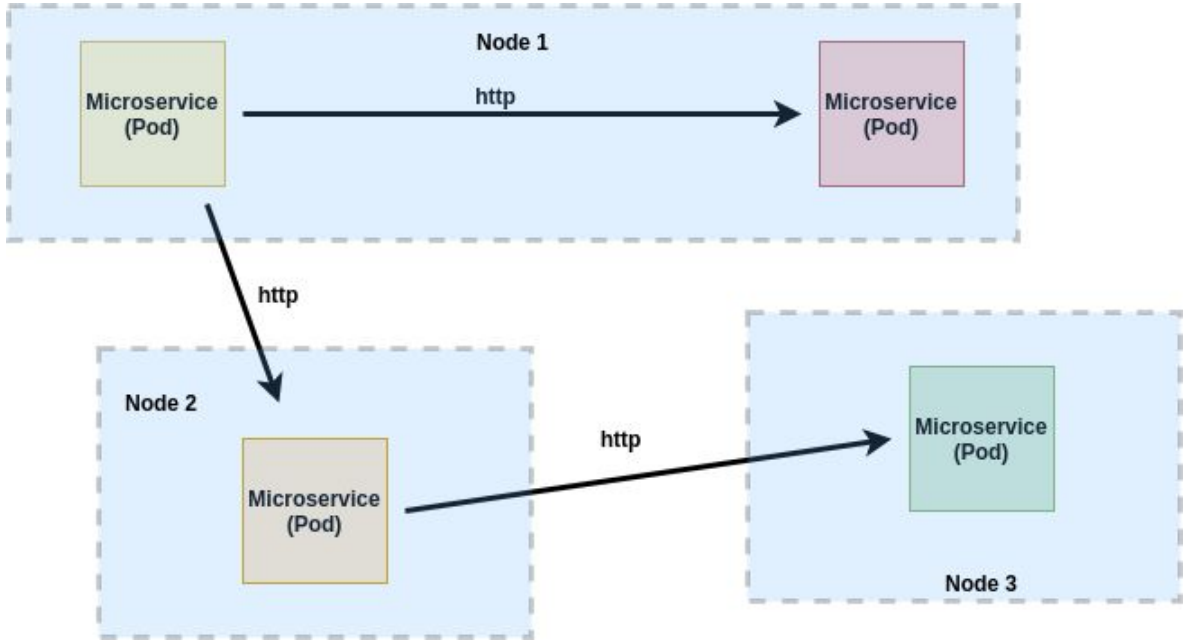
Goals

- Encryption in Kubernetes
- Calico
- WireGuard
- How we built a feature to automatically encrypt traffic in Kubernetes
- Gotchas and future work

Encryption in Kubernetes

- Compliance
- Zero-trust strategy
- Untrusted or shared environments

- Data-in-transit (not data-at-rest)
- None by default
- mTLS (often via Service Mesh)
- IPSec (host-to-host encryption)
- Custom



Encryption in Kubernetes

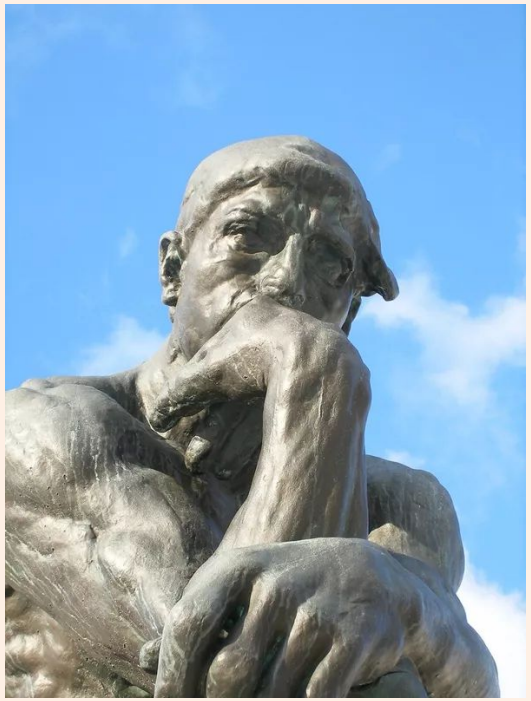
mTLS

- Client / server mutual authentication
- Certificate management & rotation
- Lock down specific namespaces (Istio)
- Automatically enabled including control plan traffic (Linkerd)

Encryption in Kubernetes

What if you do not use a Service Mesh?

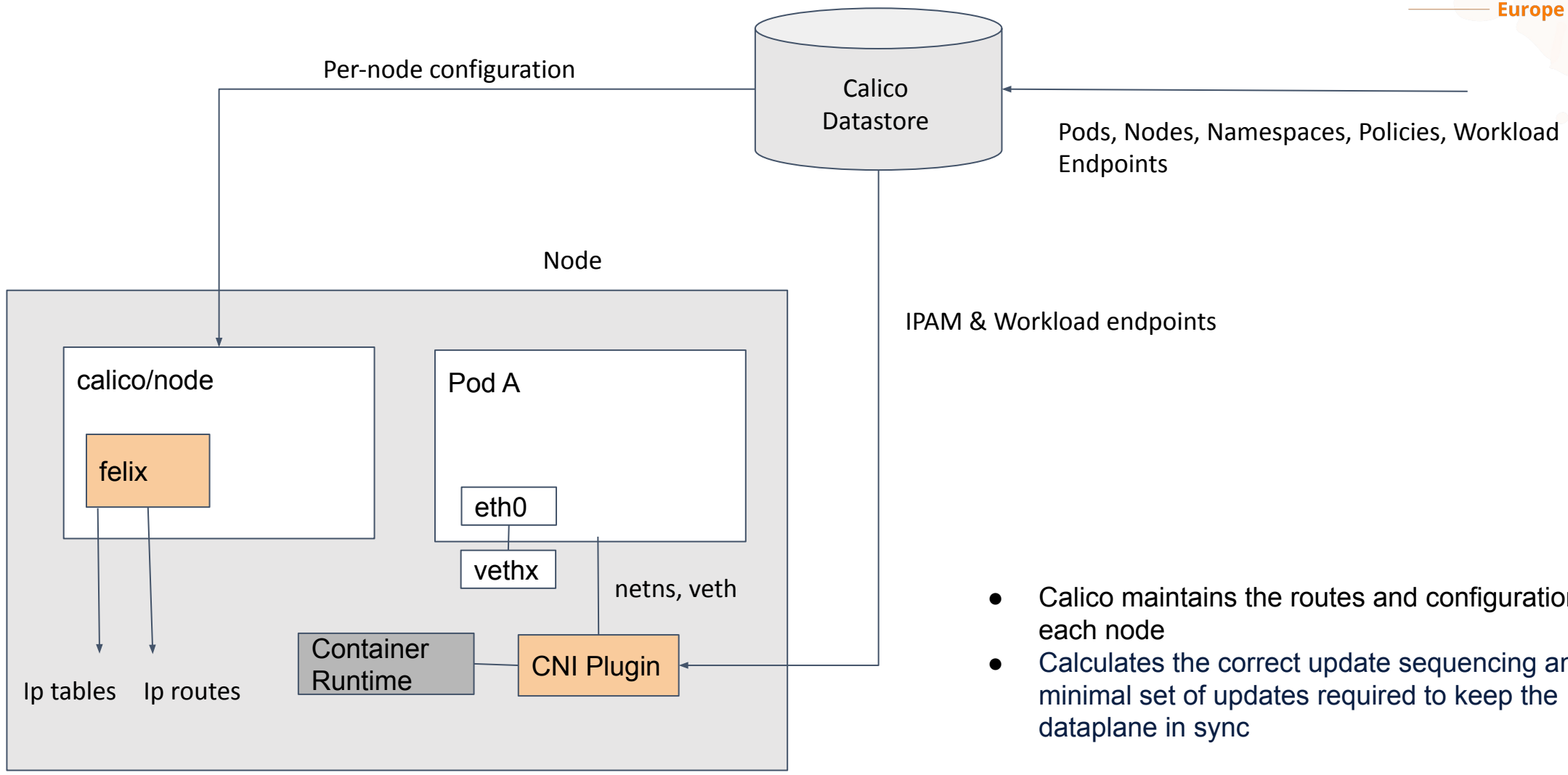
Is there an alternative to IPSec?



Calico

- Popular, battle-hardened CNI with large platform support
- Programs the dataplane to allow packets to flow between containers
- Supports linux and eBPF dataplane
- Easy to configure

Calico



- Calico maintains the routes and configuration on each node
- Calculates the correct update sequencing and minimal set of updates required to keep the dataplane in sync

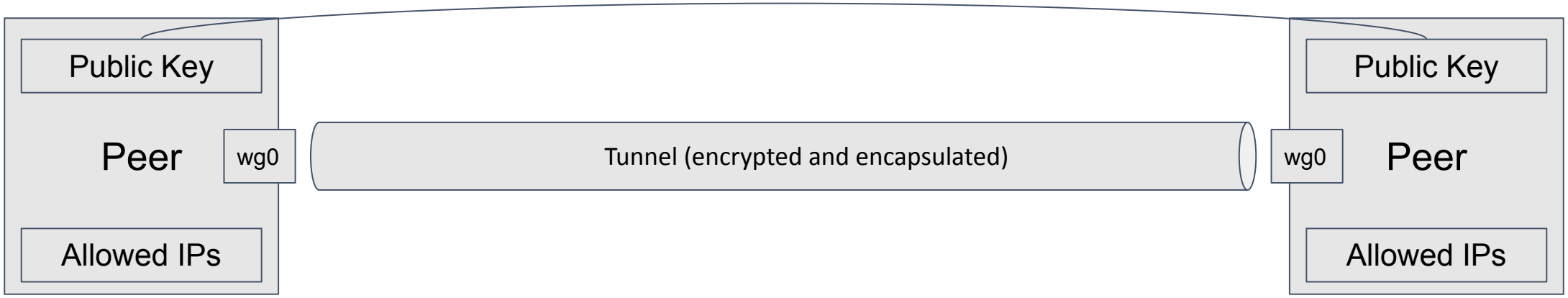
WireGuard

“uses state-of-the-art cryptography and network code to create an encrypted tunnel between two devices based on symmetric encryption”

- Lean (3000 lines of code)
- Simple (mostly transparent)
- Opinionated
  - Not responsible for how peers get public keys
  - Focused state-of-the-art cryptography
- Linux primitives (like Calico)
  - Network interfaces
  - Routing tables, ip routes
- Part of the Linux kernel 5.6+

WireGuard

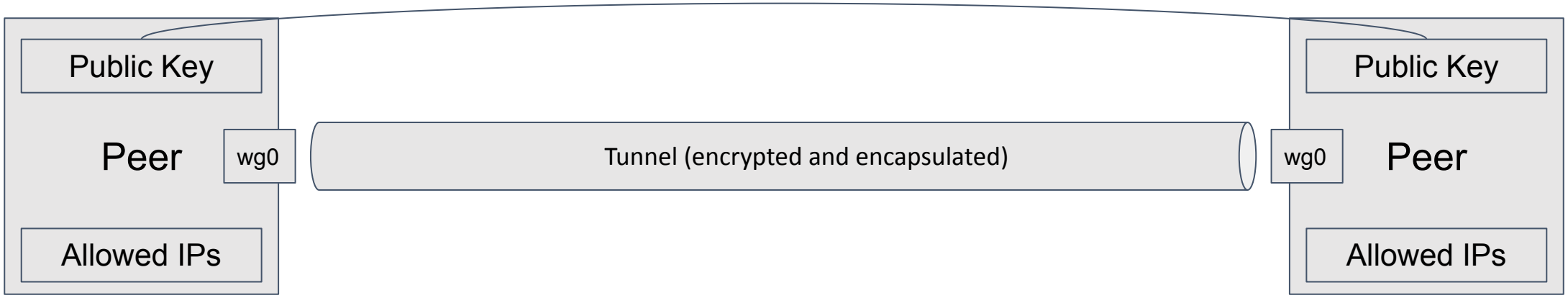
- WireGuard works by adding a network interface like eth0 or wlan0, called wg0
- This network interface can then be configured normally using ifconfig or ip-address . Add and remove routes for it using route or ip-route
- The specific WireGuard aspects of the interface are configured using the wg tool. This interface acts as a tunnel interface.
- Each peer has public key and private key, and a list IPs allowed for the tunnel.
- WireGuard associates tunnel IP addresses with public keys and remote endpoints



WireGuard

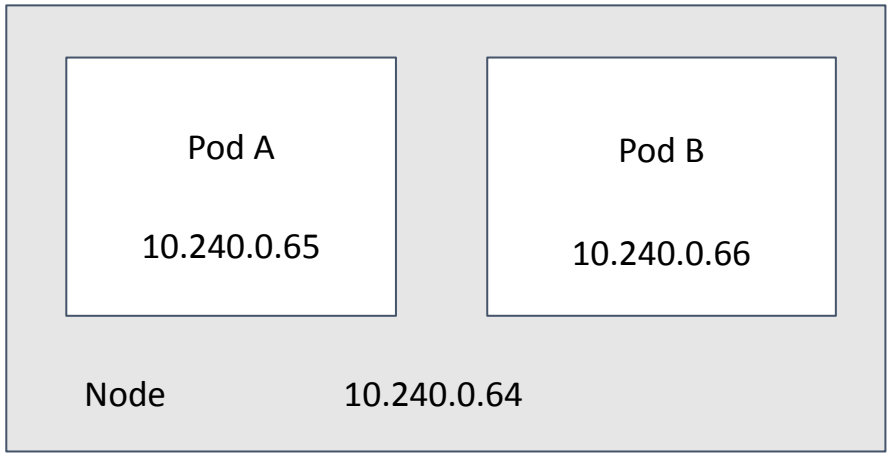
Peer … Node !

AllowedIPs associated with Public Keys (i.e. a Peer) and allowed in the tunnel … Pod IPs !



Calico + WireGuard

1. This packet is meant for 10.240.0.65. Which peer is that? Let me look... Okay, it's for peer `/r0PzTX…` (Or if it's not for any configured peer, drop the packet.)
2. Encrypt entire IP packet using peer `/r0PzTX…` public key.
3. What is the remote endpoint of peer `/r0PzTX…`? Let me look... Okay, the endpoint is UDP port 53133 on host 10.240.0.64.
4. Send encrypted bytes from step 2 over the network (eth0) to 10.240.0.64:53133 using UDP




```
interface: wireguard.cali
  public key: bbcKpAY+Q9VpmIRLT+yPaaOALxqnonxBuk5LRlvKClA=
  private key: (hidden)
  listening port: 51820
  fwmark: 0x100000

peer: /r0PzTX6FOZrW9ExPQE8zou2rh1vb2OIU6SrXMiKImw=
  endpoint: 10.240.0.64:51820
  allowed ips: 10.240.0.64/32, 10.240.0.65/32, 10.240.0.66/32
  latest handshake: 11 seconds ago
  transfer: 1.17 MiB received, 3.04 MiB sent

peer: QfUXYghyJWDcy+xLWOo+xJVsQhurVNdqtbstTsdOp2O=
  endpoint: 10.240.0.4:51820
  allowed ips: 10.240.0.4/32, 10.240.0.5/32, 10.240.0.6/32
  latest handshake: 46 seconds ago
  transfer: 83.48 KiB received, 365.77 KiB sent
```

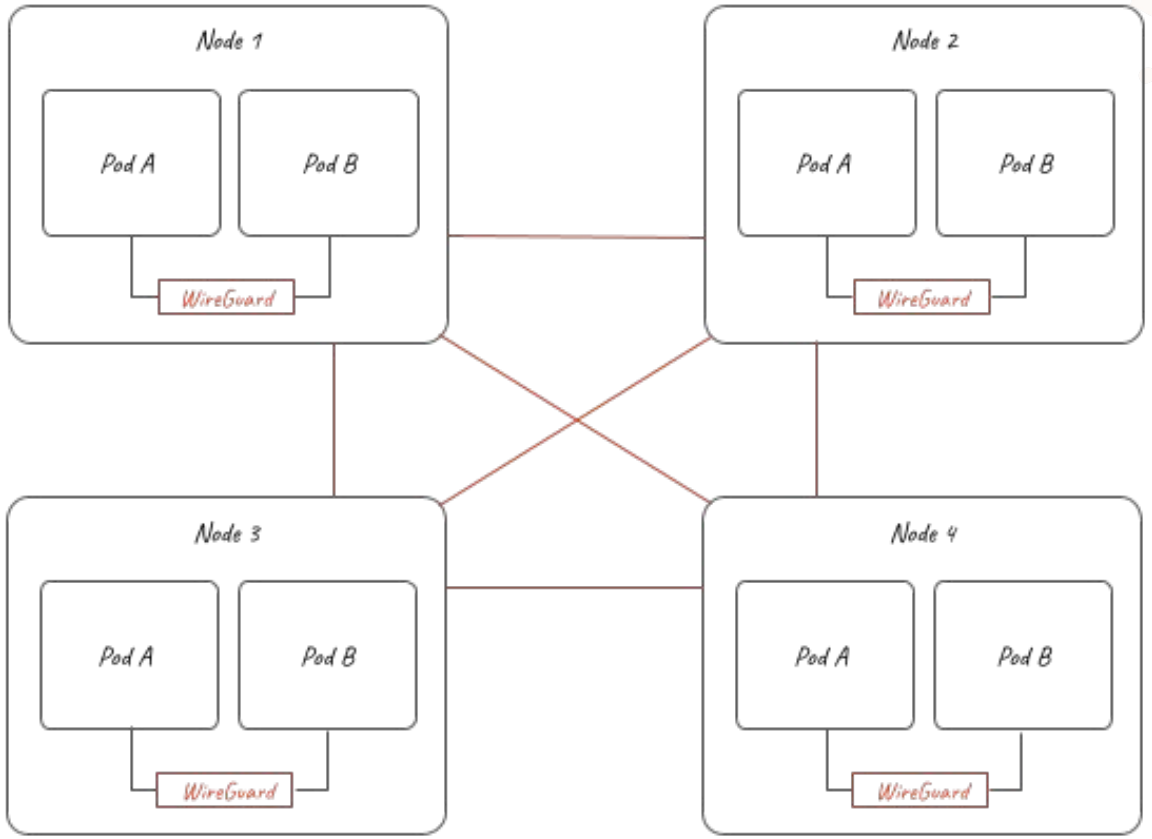
Calico + WireGuard

In Summary…

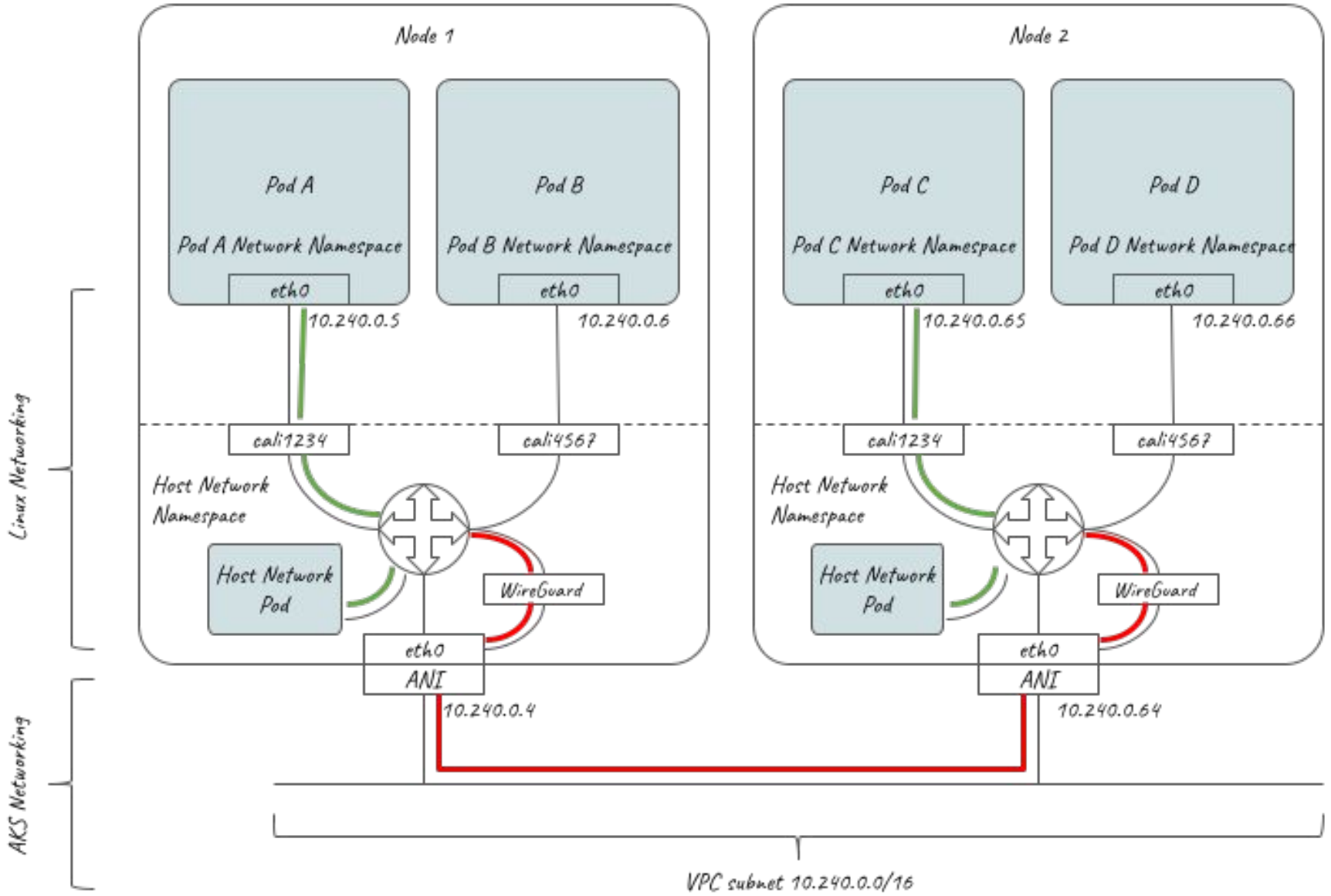
- Calico maintains an eventually consistent dataplane
- Calico and WireGuard like programming with linux primitives
- WireGuard's Peer and AllowedIP concepts map nicely to Nodes and Pods in Kubernetes

Calico + WireGuard

- As nodes come online, they are added as Peers in wg configuration
- Public keys become part of the node manifest and shared with other nodes
- Pod IPs become part of the Allowed IP lists in wg configuration associated with that new Peer
- Results in an encrypted mesh for pod-to-pod traffic between nodes



Calico + WireGuard



Calico + WireGuard

Enable WireGuard

```
kubectl patch felixconfiguration default --type='merge' -p
'{"spec":{"wireguardEnabled":true}}'
```

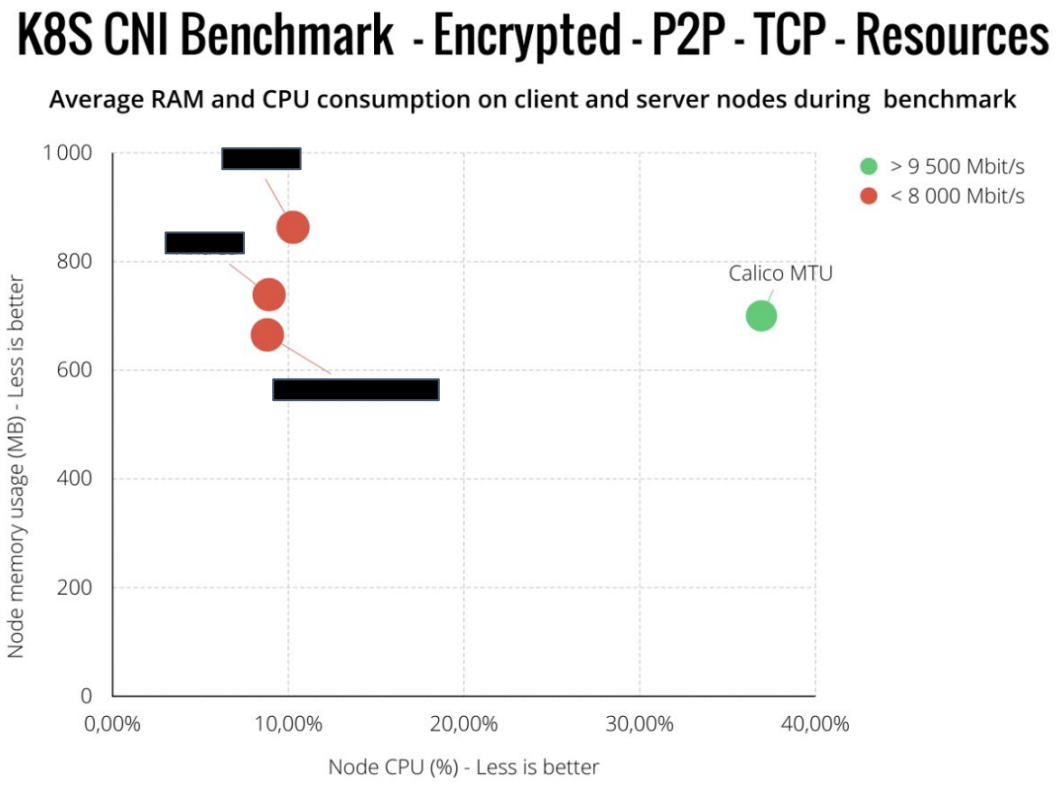
Fetch the node, see the key…

```
calicoctl get node -o yaml
```

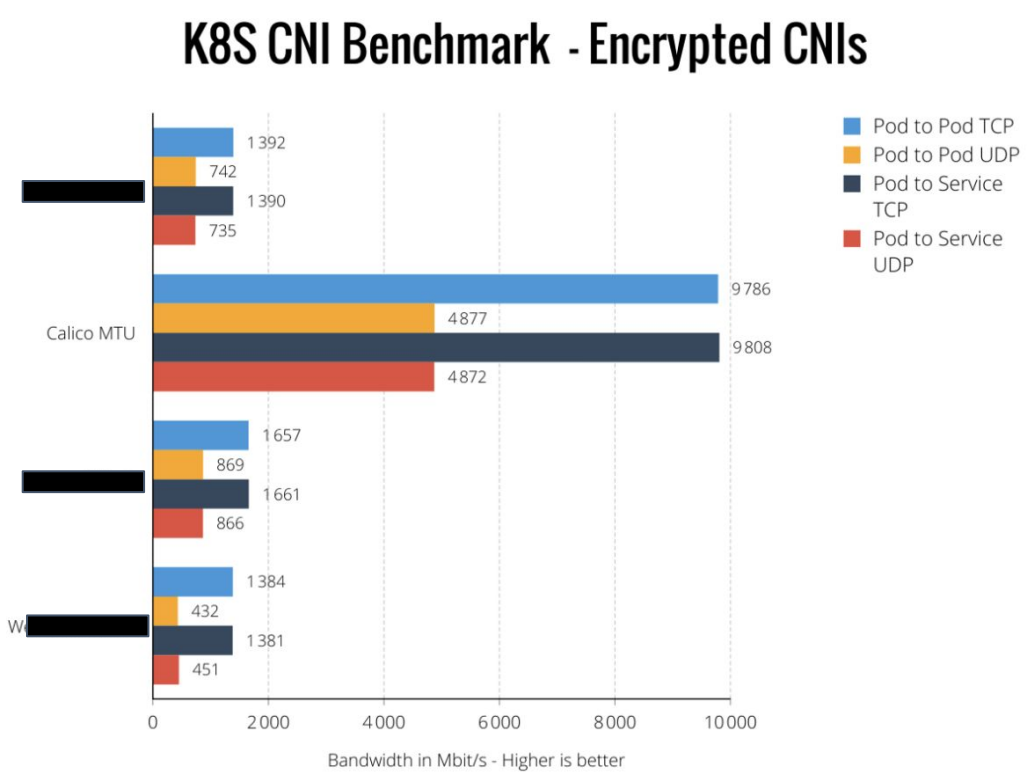
```
...
 status:
   ..
   wireguardPublicKey: jlkVyQYooZYzI2wFfNhSZez5eWh44yfq1wKVjLvSXgY=
   ...
```

Performance

Resources



Bandwidth



Gotchas and future work

- Pod-to-pod on same host
- Pre-flight checks
- Race conditions
- More fine-grained control - policy-based encryption

Thanks!

- https://www.tigera.io/blog/introducing-wireguard-encryption-with-calico/
- https://thenewstack.io/calico-wireguard-support-with-azure-cni/
- https://projectcalico.docs.tigera.io/security/encrypt-cluster-pod-traffic