OOM Priorities for Dublin

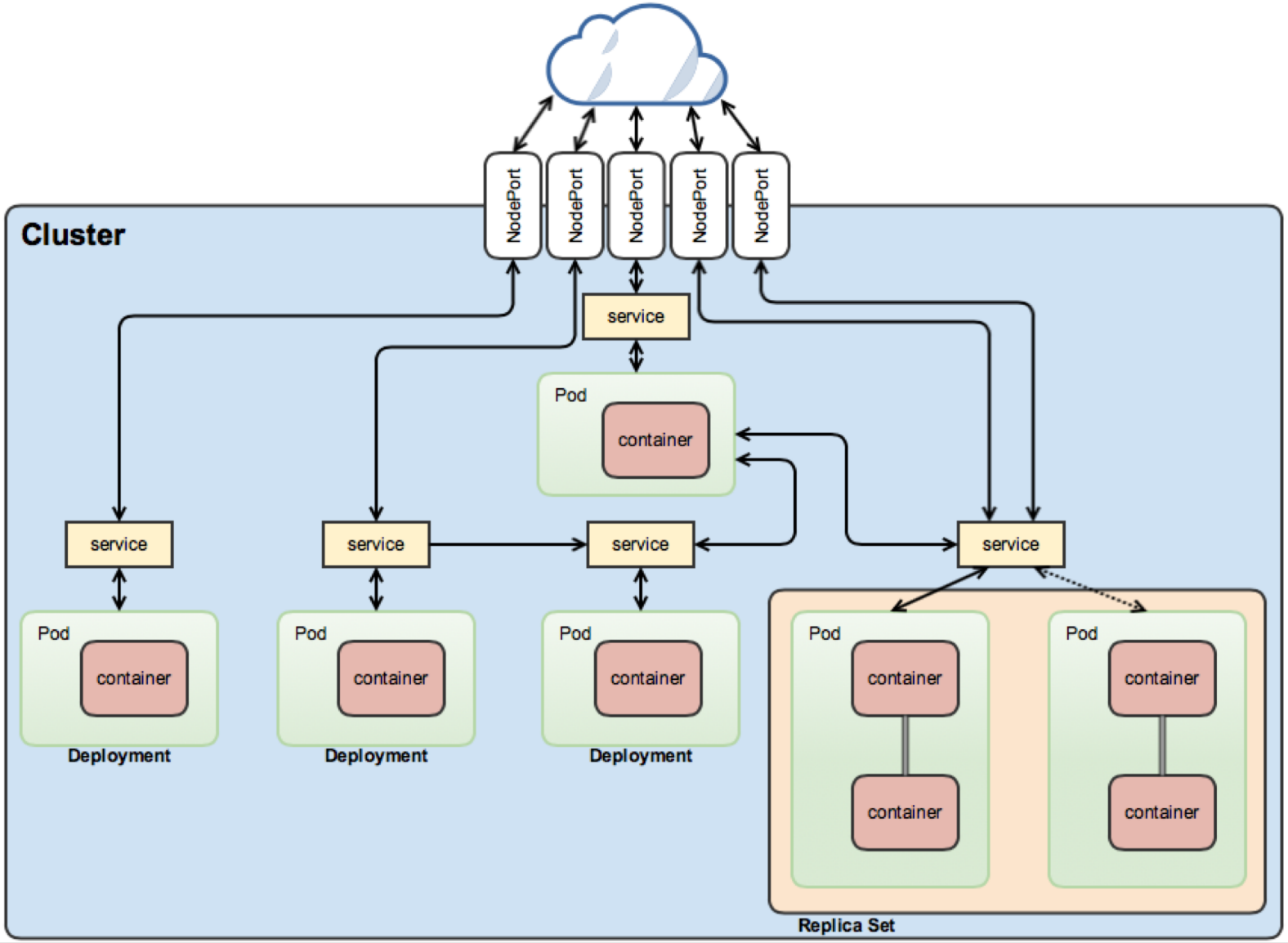
- Footprint Optimization
  - Dec.12 2018 vF2F:
    - https://wiki.onap.org/pages/editpage.action?pageId=45293323
- Security
  - Ingress Controller (eliminate node port pain)
  - Network Policies
  - TLS (Istio/AAF)
- Geo-diversity
  - Container Networking Interface (CNI) Plugin (BGP, VxLAN)
- Production Grade Deployments
  - Automatic Upgrade from Casablanca to Dublin
  - Production Grade Storage Options
  - Improved Platform Monitoring & Reporting
  - Offline Installer
- Development Improvements
  - Helm Chart Ownership
  - H/A Kubernetes Cluster Best Practices

ONAP
OPEN NETWORK AUTOMATION PLATFORM

Security

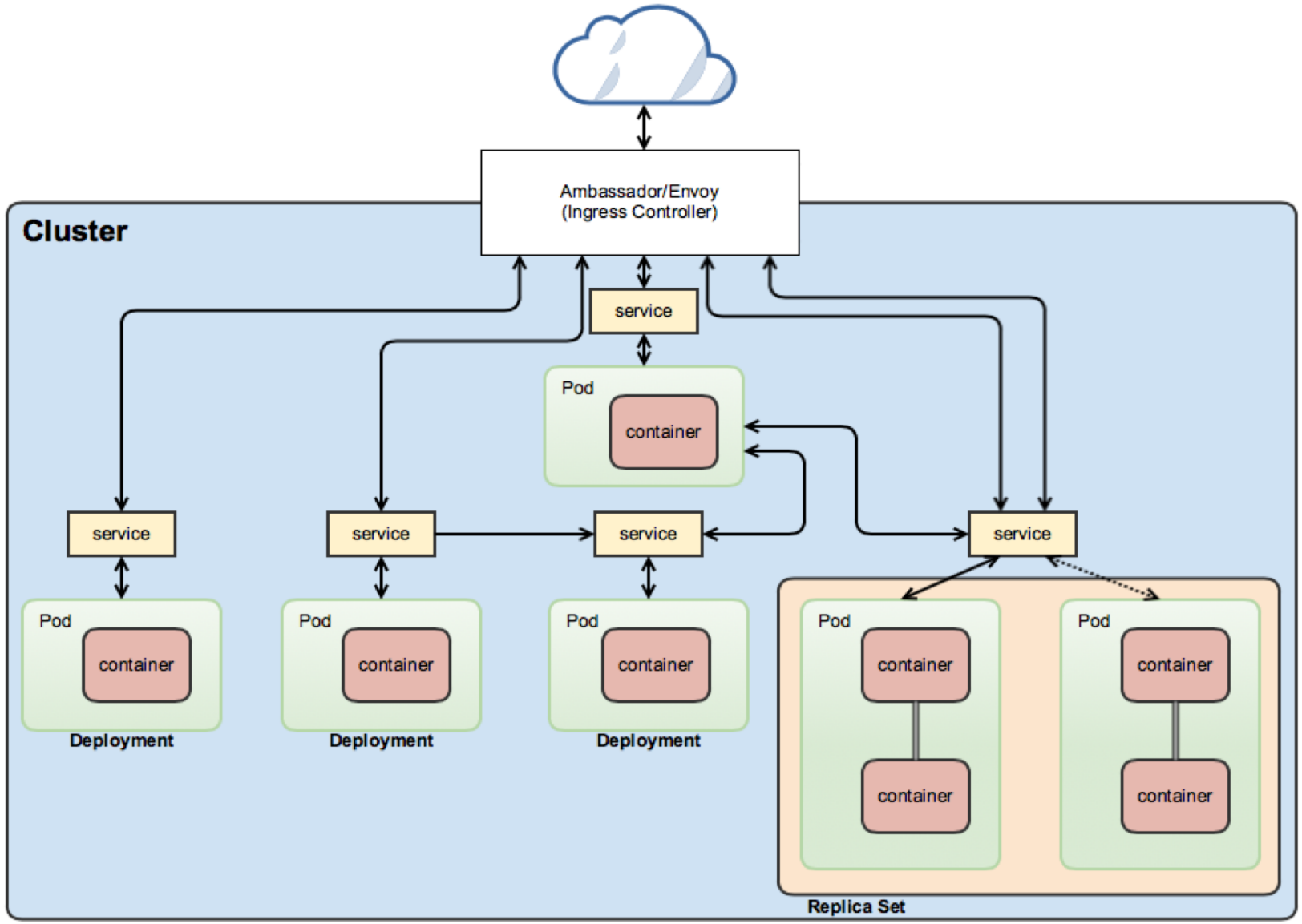
Securing Cluster Access

- 100+ Node Ports
- Huge attack surface
- Administrative nightmare



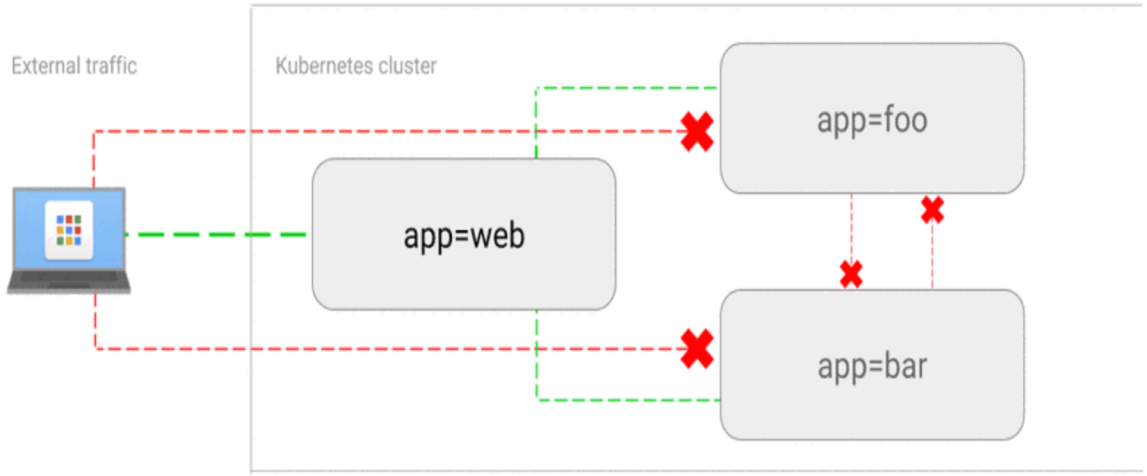
Securing Cluster Access

- Cluster Edge Proxy/Load Balancer
- Dramatic reduction in “attack surface”
- Improved traffic control
- Ambassador/Envoy
  - Kubernetes-native microservice gateway
  - Traffic routed through Envoy Proxy
  - K8s as state store and for resiliency
  - Authentication and TLS termination
  - Rate limiting
  - Observability
  - Traffic routing



Control of traffic behavior

- Network Policies
  - fine-grained Traffic Control
    - rich routing rules
  - restrict Pod-to-Pod communication
  - enforcement via CNI plugin



example Network Policy:

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

labels & roles:

```
...
 ingress:
 - from:
   - namespaceSelector:
       matchLabels:
         user: alice
     podSelector:
       matchLabels:
         role: client
...
```

TLS

- Pluggable Authentication/Authorization/Certificate Management
  - SECCOM approved ✓
- Istio as reference integration
  - Support for out-of-the-box plugins (what are they?)
  - Support for custom plugins to legacy/proprietary auth/cert solutions
- AAF integration implemented as Istio Plugin
- Dublin Release will be validated on Istio with AAF integration
- Optional Istio deployment
- Optional AAF integration

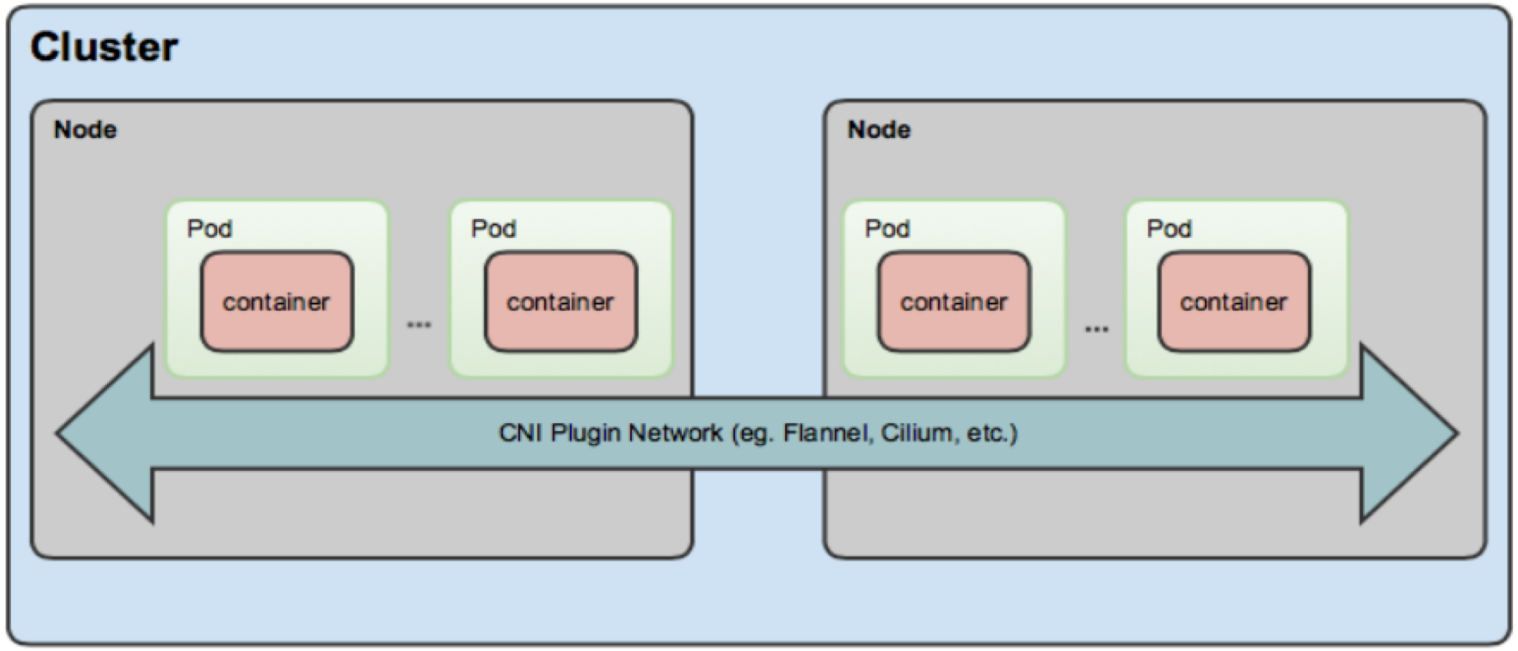
* Istio memory and latency optimizations being evaluated

ONAP
OPEN NETWORK AUTOMATION PLATFORM

Geo-diversity

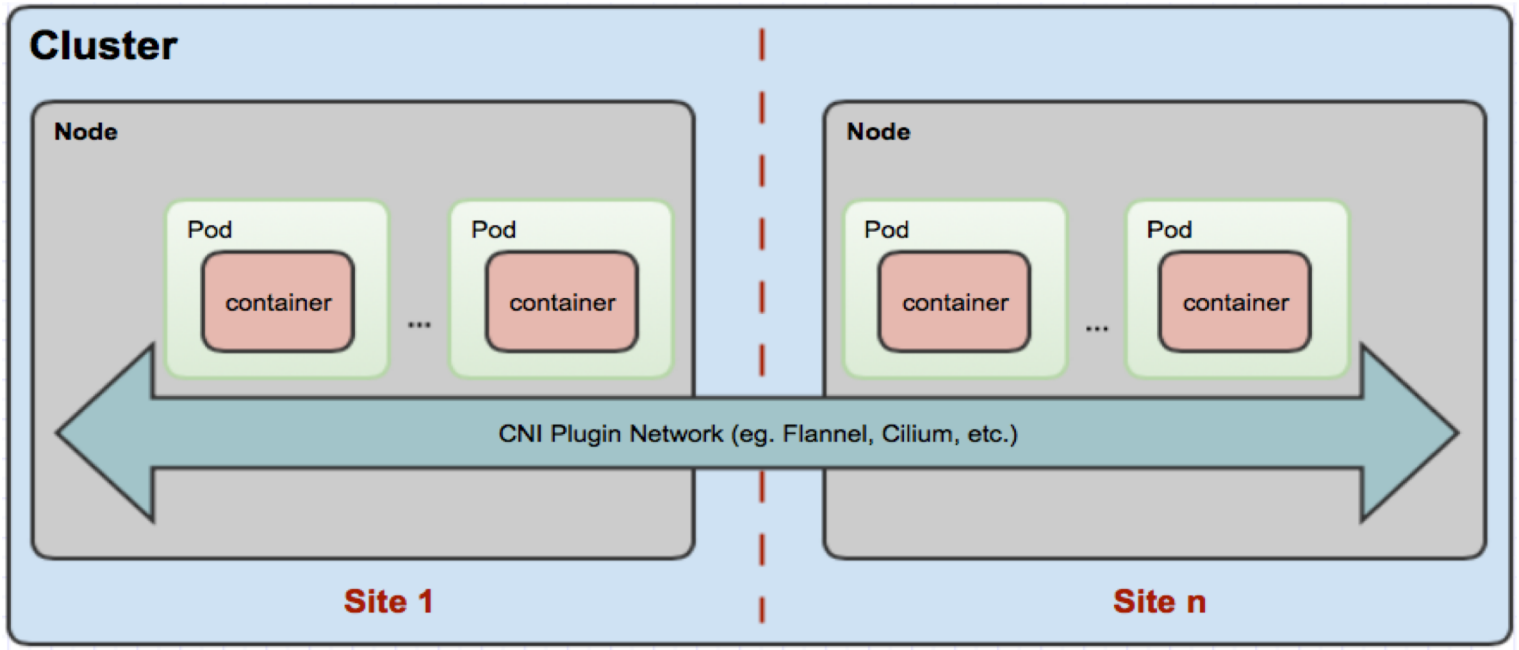
Container Network Interface (CNI)

- Cloud Native Computing Foundation project
- CNI defines specification/API for writing container networking plugins
- CNI Plugins
  - configure network interfaces in containers
  - applies routing rules defined in Network Policies
  - many open source and commercial plugins available
  - provides choice for operators



Container Network Interface (CNI)

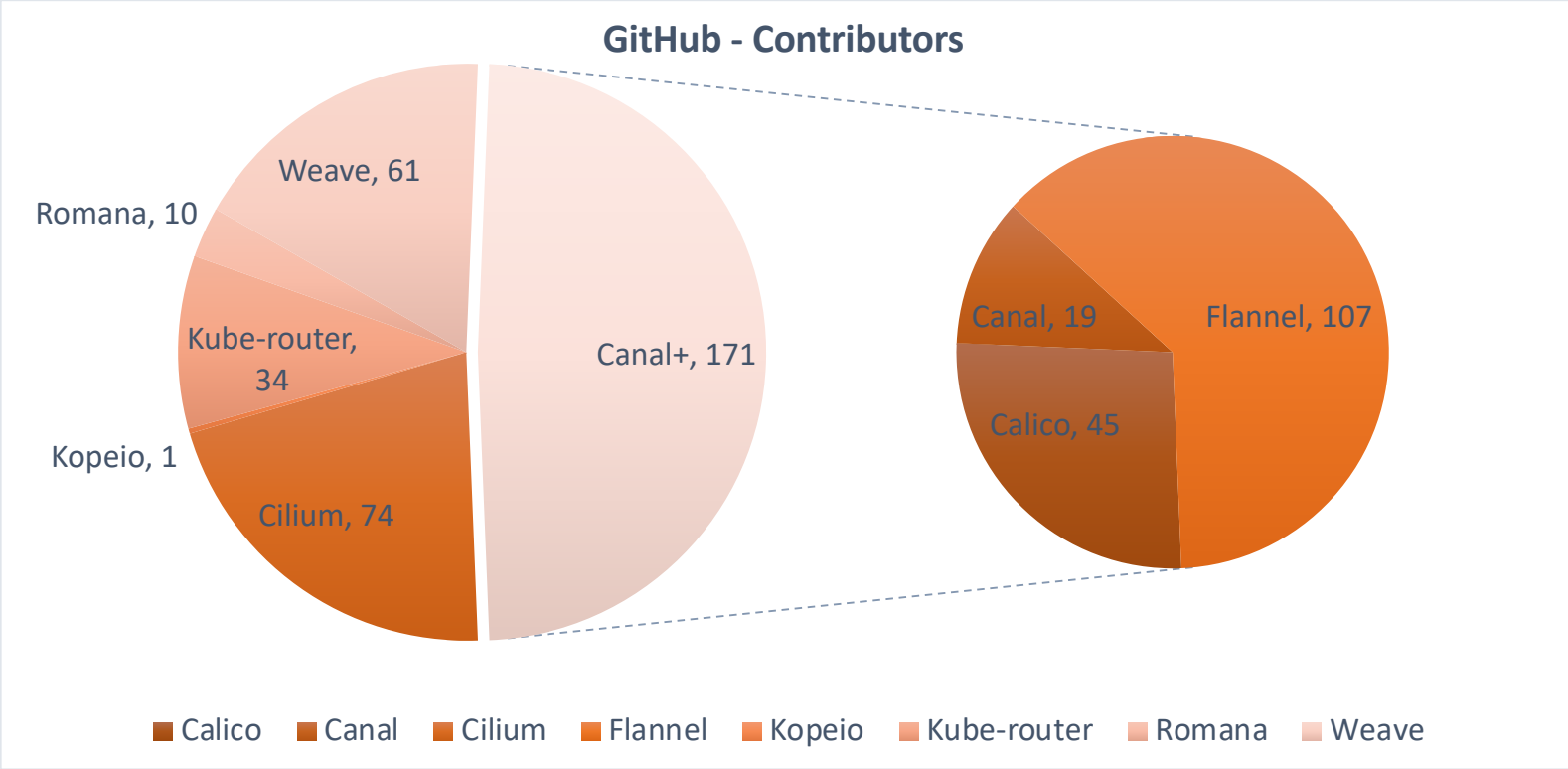
- Cloud Native Computing Foundation project
- CNI defines specification/API for writing container networking plugins
- CNI Plugins
  - configure network interfaces in containers
  - applies routing rules defined in Network Policies
  - many open source and commercial plugins available
  - provides choice for operators



- Nodes may be in the same site or across Geo-separated sites
- Still 1 cluster

Container Network Interface (CNI)

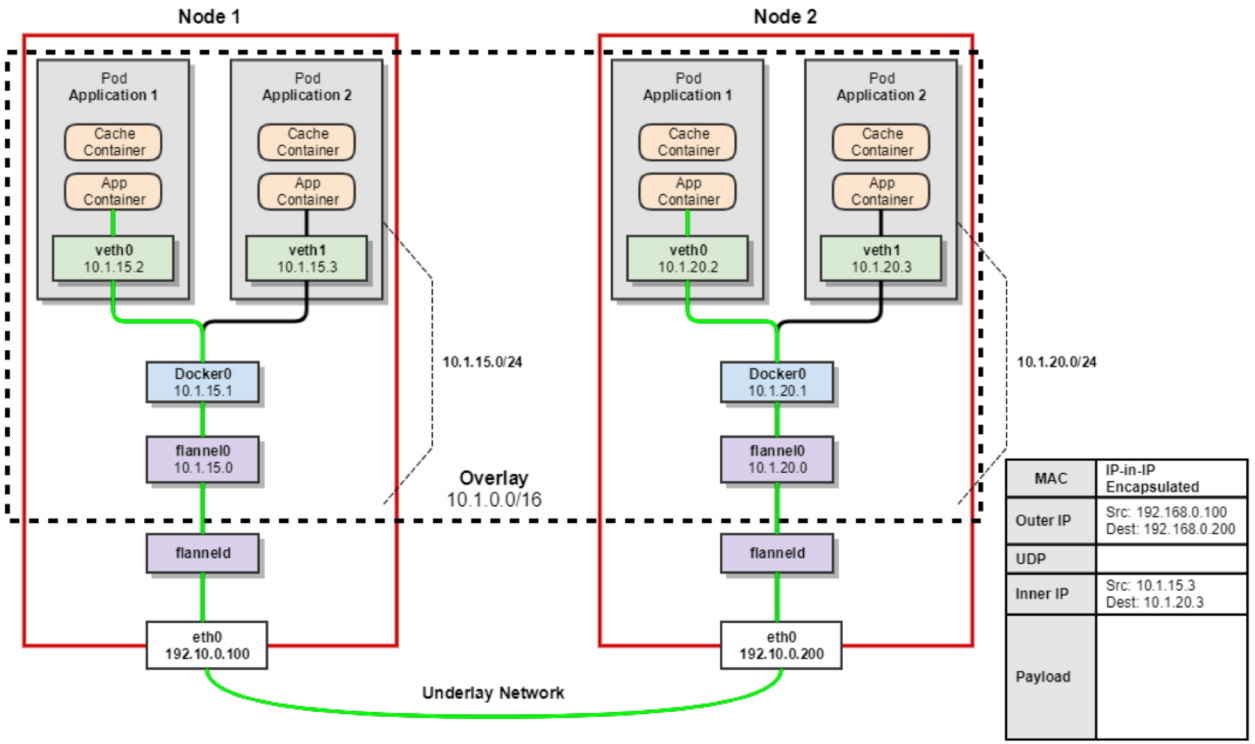
- Kubernetes uses CNI as an interface between network providers and Kubernetes pod networking
- Several popular CNI plugins available:



CNI Network Models

Encapsulated/Overlay Network

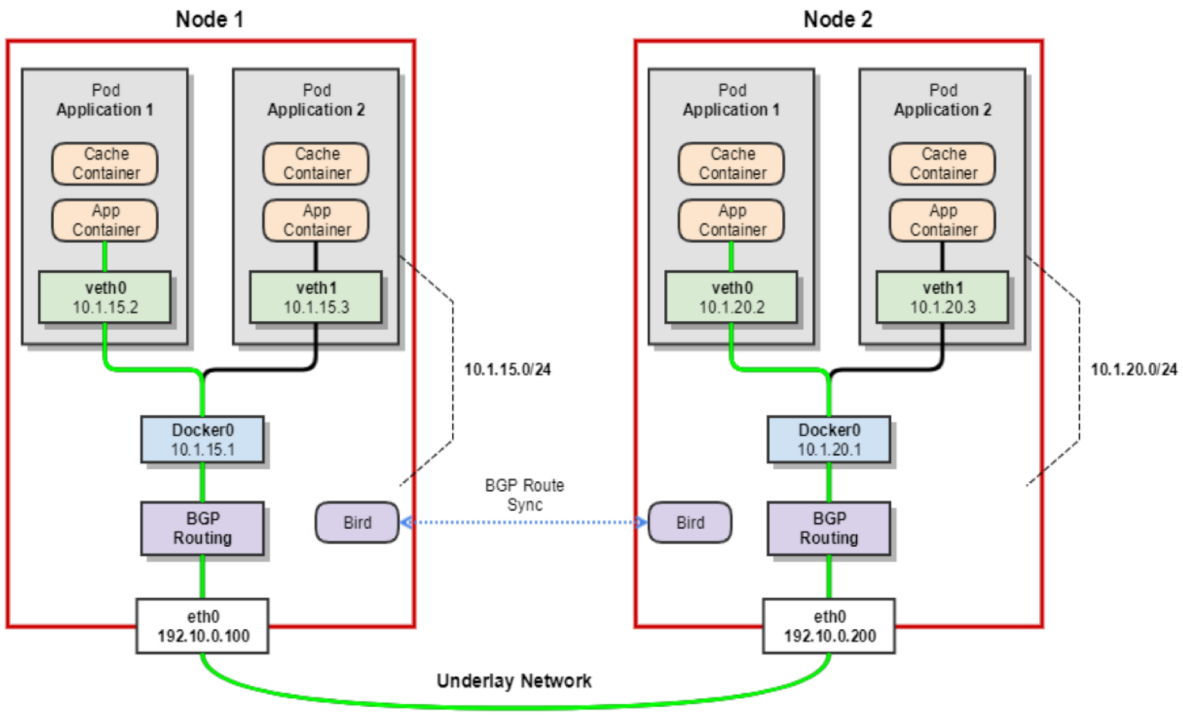
- Layer 2 (L2) network encapsulated over the existing Layer 3 (L3) network (i.e. VXLAN)
- flannel, Canal, Weave & Cilium



MAC	IP-in-IP Encapsulated
Outer IP	Src: 192.168.0.100 Dest: 192.168.0.200
UDP	
Inner IP	Src: 10.1.15.3 Dest: 10.1.20.3
Payload	

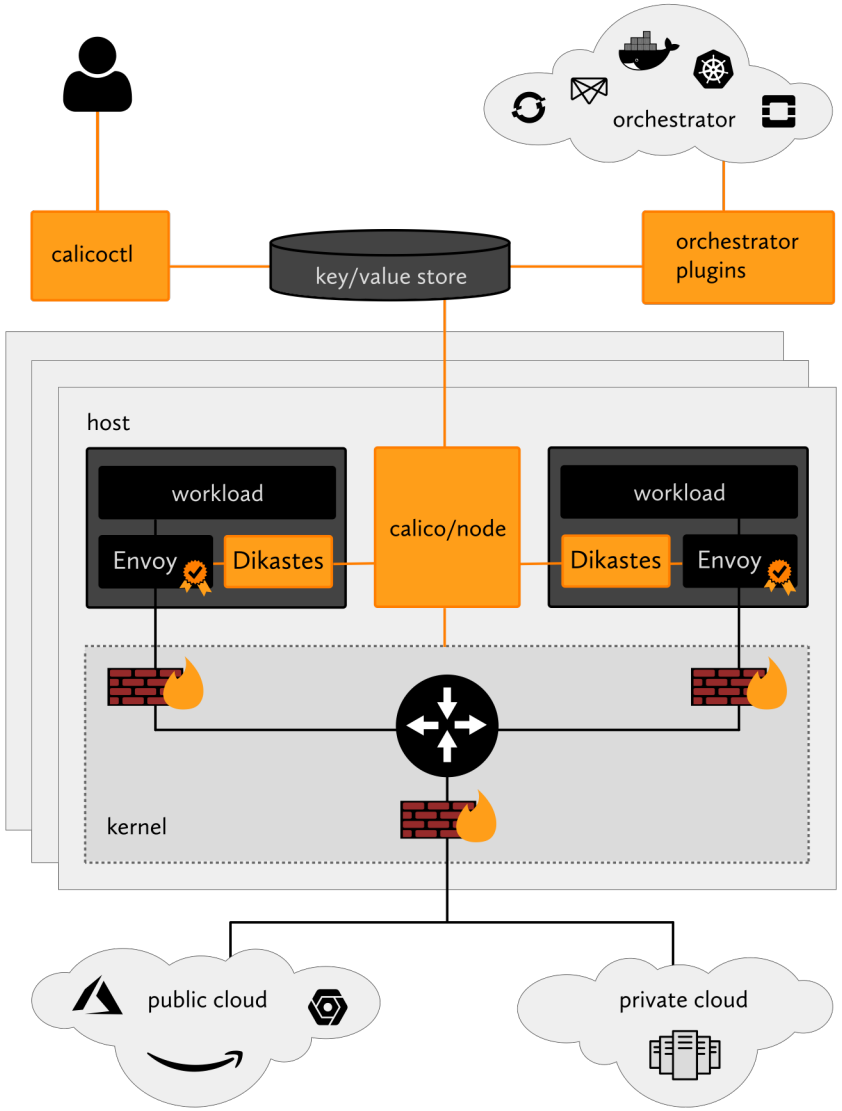
Unencapsulated Network

- This network model provides an L3 network to route packets between containers. (i.e. BGP)
- Calico, Romana & Cilium



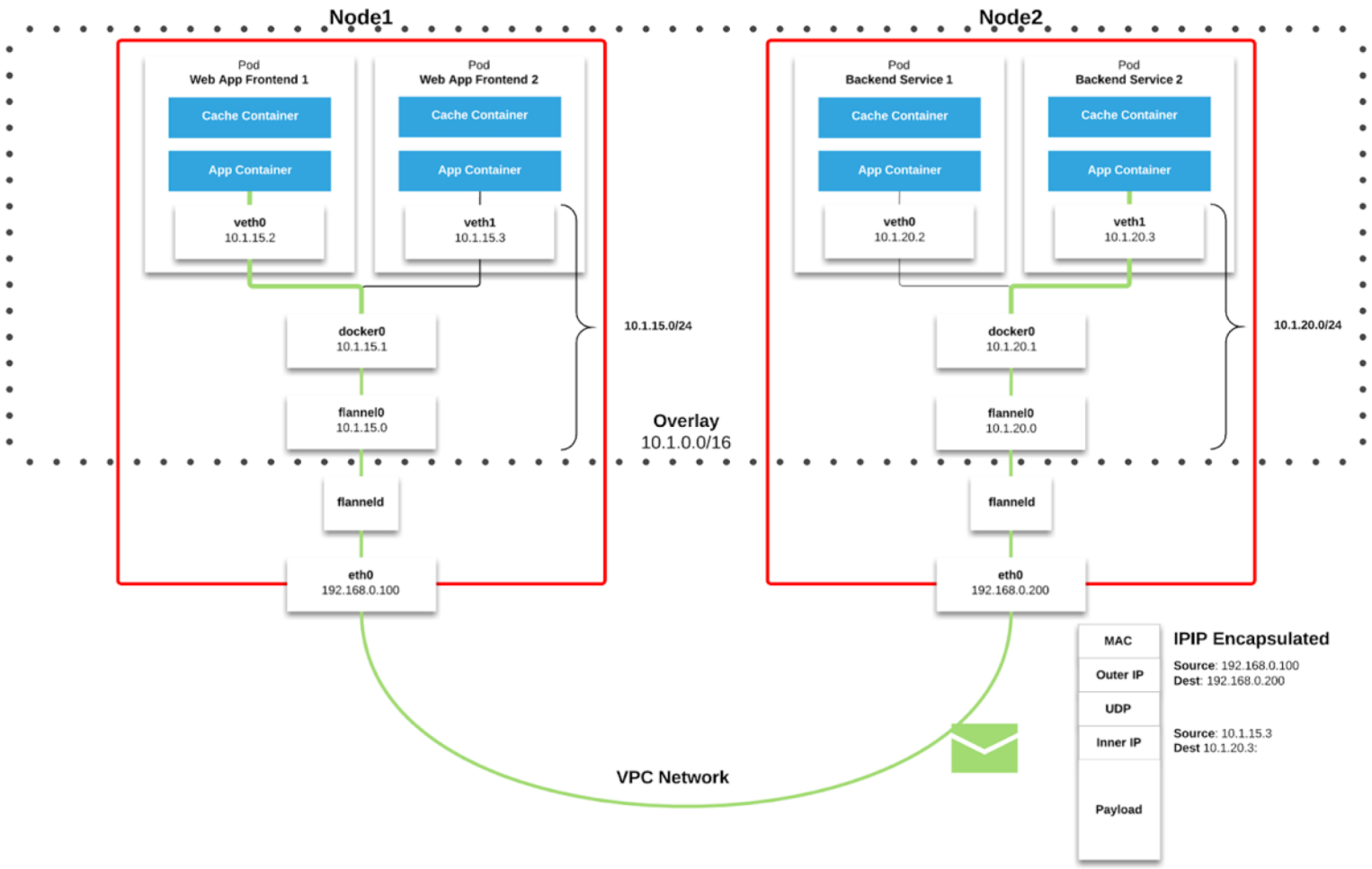
Calico

- Calico provides secure network connectivity for containers and virtual machine workloads
- Calico uses IP-in-IP tunneling or can work with other overlay networking such as flannel
- Calico also provides dynamic enforcement of network security rules
- Optional Envoy sidecars that secure workload-to-workload communications with mutual TLS authentication and enforce application layer policy (K8s NetworkPolicy)
- Used by Akraino

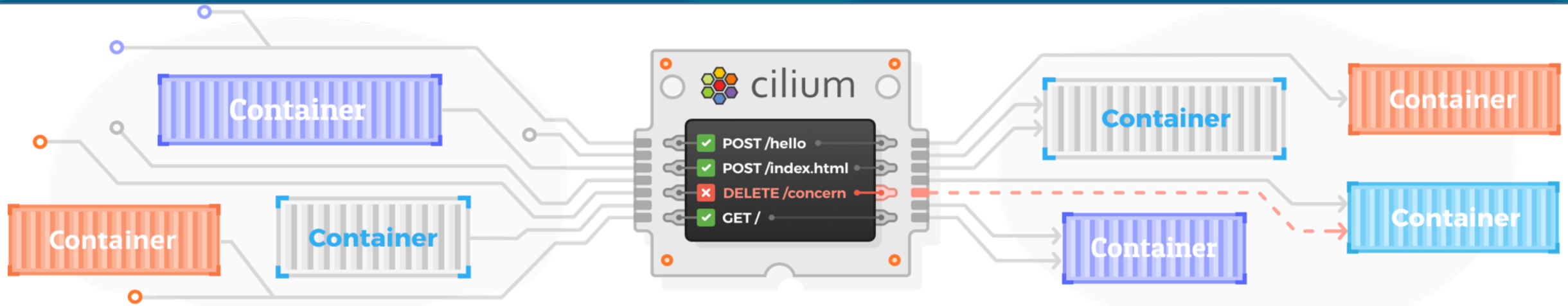


Canal / flannel

- An integration of Calico and flannel
- flannel creates a overlay network (typically VXLAN) to interconnect K8s hosts
- flannel is focused on networking and uses Calico for network policies



Cilium

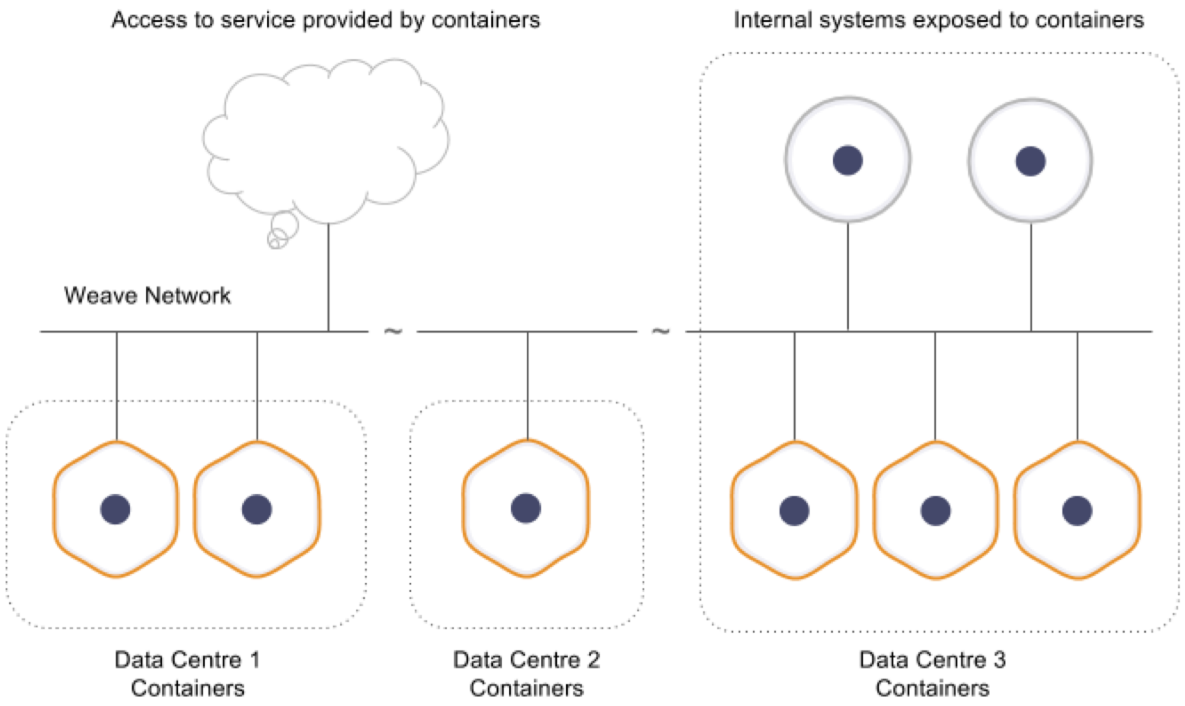


- Cilium brings API-aware network security filtering to Linux containers
- Uses a Linux kernel technology called Berkeley Packet Filters (BPF) – fast!
- Enforce network & application layer security policies based on container identity
- Optional integration with Istio/Envoy (e.g. 1 proxy/host)
- Encapsulated/Overlay & Unencapsulated mode
- Supports K8s: Network Policies, Labels, Ingress, Egress Service
- cilium-agent can be configured to serve Prometheus metrics

See: https://github.com/iovisor/bcc for BPF tracing tools

Weave weaveworks

- Weave (from Weaveworks) is an VXLAN overlay technology
- Includes a “micro DNS” server on each node providing fast service discovery
- Supports Linux kernel Open vSwitch to improve performance
- Doesn’t require an external cluster store (typically etcd)



ONAP
OPEN NETWORK AUTOMATION PLATFORM

Production Grade Deployments

4th release of ONAP – It's Time

Production Grade Storage Options

- Replacing hostPath with a **default** Storage Class for Dublin
  - NFS
  - GlusterFS (delivered in R3 - GlusterFS Infra Demo)
  - CephFS
  - Cinder

- Casablanca enabled storage class configuration

```
global:                                   (or per helm chart)
  persistence:
    storageClass: glusterfs-sc
```

- Different classes provide
  - quality-of-service levels
  - backup and restore policies

AWSElasticBlockStore
AzureFile
AzureDisk
CephFS
Cinder
FC
Flexvolume
Flocker
GCEPersistentDisk
Glusterfs
iSCSI
Quobyte
NFS
RBD
VsphereVolume
PortworxVolume
ScaleIO
StorageOS
Local

Automatic Upgrade from Casablanca to Dublin

Holy Grail: Non-disruptive Rolling Upgrades (no service disruption)

```
helm deploy prod dublin/onap -f onap-prod.yaml
```

Dublin Proposal:

- Automated schema and data migration from Casablanca to Dublin
- Automated component upgrade (mostly there – some issues i.e. jobs)

Stretch Goals:

- Automated component downgrade/rollback – not tested
  - uses old data prior to migration
- No service interruption during upgrade

Automatic Upgrade from Casablanca to Dublin

- Collaborating with PTLs/Project teams to deliver schema upgrades with data migration to Casablanca – will continue through to Dublin
  - AAI
  - SO
  - SDC (under development now)
- Automated using Helm hooks & rolling upgrade/downgrade strategies (POC underway)
  - Some applications already had upgrade capabilities

What are the challenges?

- Migrating from independent db instances to shared db cluster
  - (i.e. 12 Mariadb -> 1 Mariadb-Galera)
- Upgrade/Rollback built into architecture roadmap and design of every new feature
  - api versioning (already underway)
  - upgrade first then enable new features based on latest API versions
  - evolutionary changes
  - message queuing

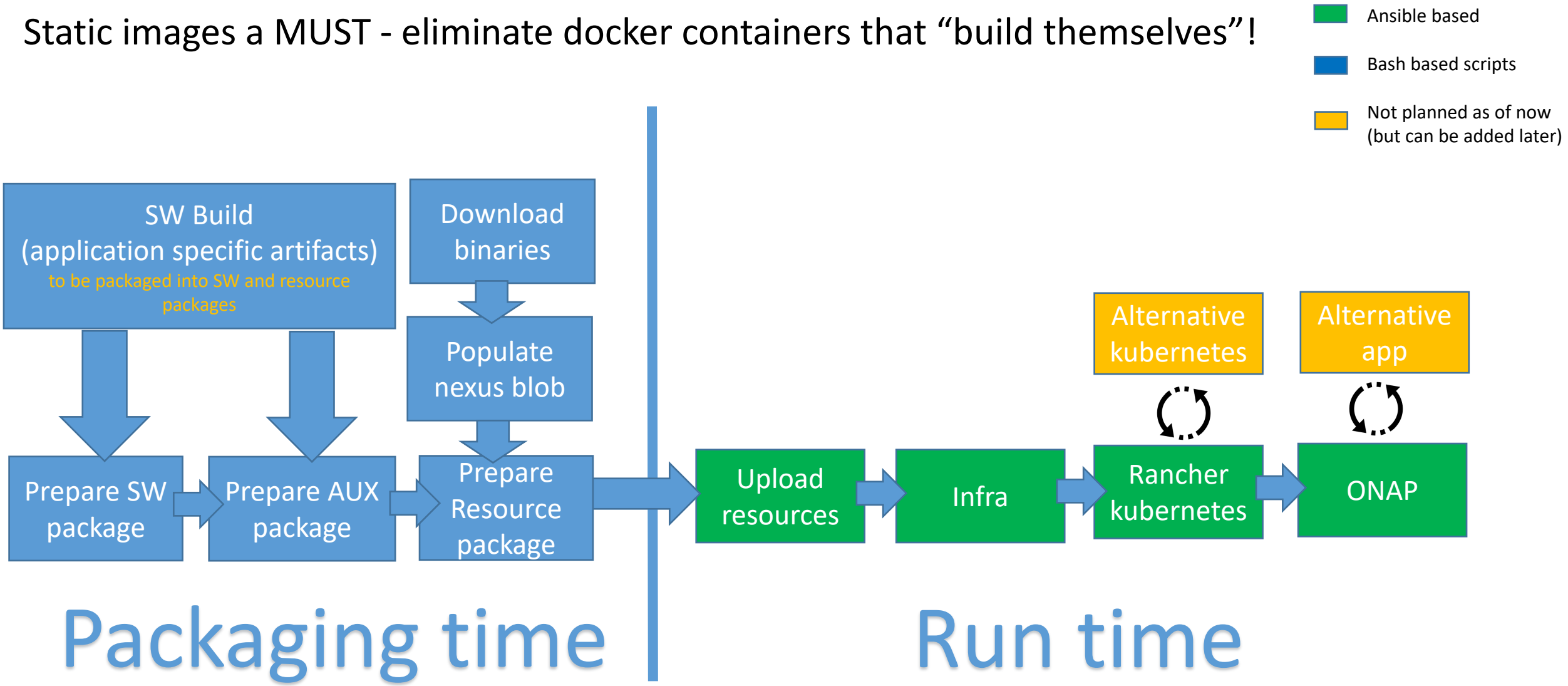
Improved Platform Monitoring and Reporting

- Better visibility into health of the platform
- Platform Dashboard - all green = platform is good
  - replacement of Consul for monitoring
  - expansion/integration of existing Logging Project Dashboard(s)
  - ability to track down and debug issues
- Enable Monitoring of important Metrics
- Logs (and traces)
  - integration with existing Logging project that uses Elastic Stack to centralize & parse logs
- Trigger notifications to alert on states of interest or concern

Tools
  - Prometheus Operator, Zabbix, Prometheus+Grafana, Timelion, Nagios

Offline installer – architecture vision
(Fresh installation)

Static images a MUST - eliminate docker containers that “build themselves”!



ONAP
OPEN NETWORK AUTOMATION PLATFORM

Development Improvements

Helm Chart Ownership

Need for project teams to take ownership of Helm charts for their projects in Dublin

Project teams know best how to update their own charts to address:

- Configuration changes (in files and inject Helm configuration)
- Software Upgrade and Roll-back support (including DB migration)
- Resource Limits
- Network Policies
- (Anti-)affinity rules

OOM team will continue to own:

- Common “shared” helm charts
- Helm plugins
- Offline Installer
- Service Mesh evolution

Helm Chart Ownership

Challenges:

- Creation of oom repo per project
- Build process support (underway – weekly meetings with LF)
- How to maintain consistency with Standardized Helm Charts?
  - need for test suite to validate standardized charts
  - inclusion of OOM Team in patch reviews (necessary during transition)
  - integration team assistance to help enforce global configuration hierarchy

H/A Kubernetes Cluster

- Infrastructure setup is Operator/Service Provider choice
- Integration lab instabilities has made this a necessity
  - No fault of lab – just over subscribed and ONAP increasingly more demanding of resources
  - Must be able to have resilient infrastructure via H/A k8s
- Provide H/A Kubernetes Cluster Best Practices
- Migration to RKE for H/A integration environment
  - deploy Rancher 2.x server in HA with a single command