
KubeCon
CloudNativeCon
OPEN SOURCE SUMMIT
China 2023

Scaling up and Scaling Out Networking, Observability, and Security with Cilium

Bill Mulligan, Isovalent & Jaff Cheng, Trip.com Group

Agenda

- Cilium at Trip.com
- Cilium on Alibaba Cloud
- Challenges & Solutions at Scale
- Takeaways

cilium

Created by ISOVALENT

eBPF-based:

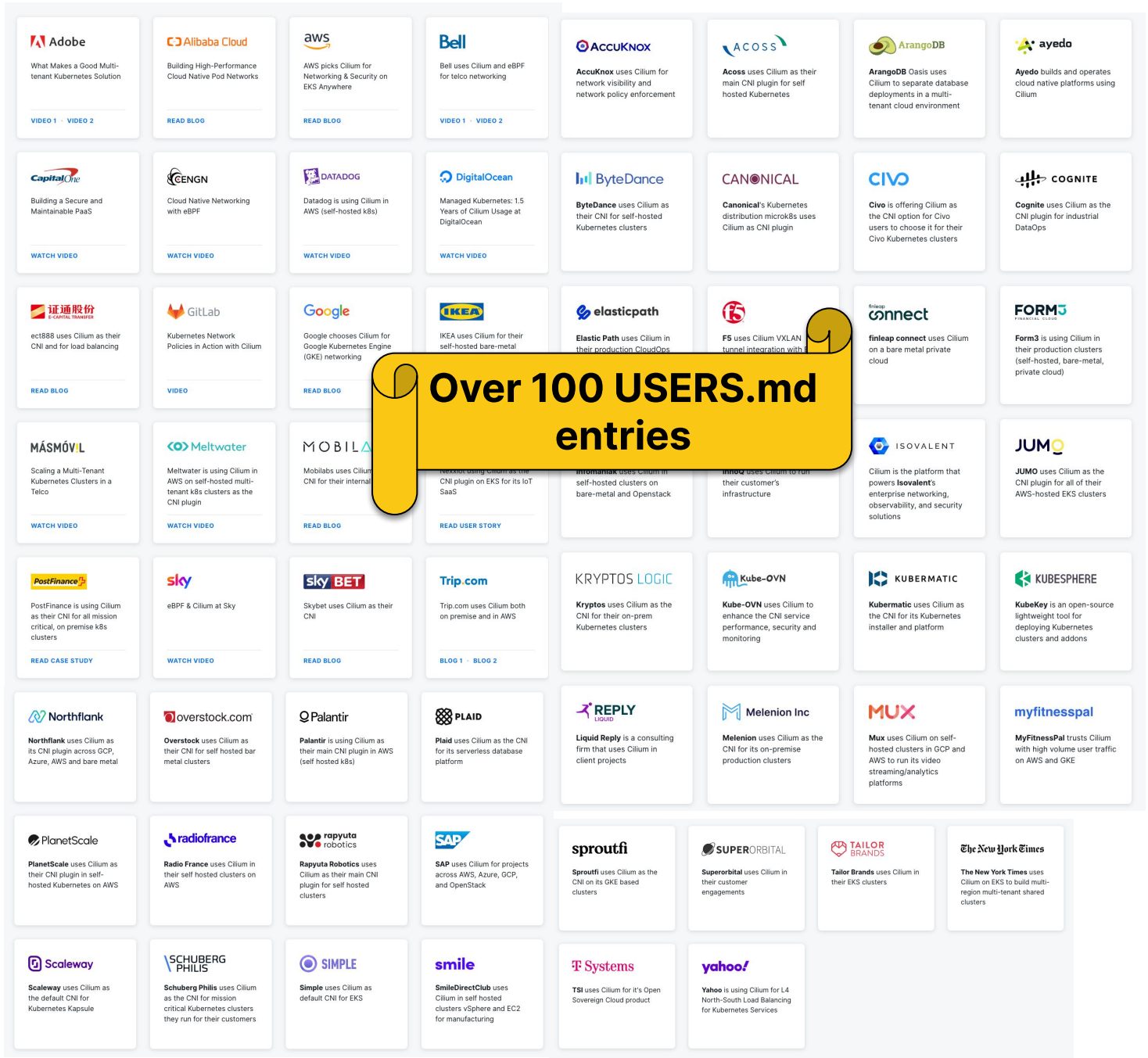
- Networking
- Security
- Observability
- Service Mesh & Ingress

Foundation

CLOUD NATIVE COMPUTING FOUNDATION

Technology

eBPF envoy



eBPF-based:

- Networking
- Security
- Observability
- Service Mesh & Ingress

Foundation

CLOUD NATIVE COMPUTING FOUNDATION

Technology

eBPF envoy

Deploy on your prefered cloud

aws · Sovereign Cloud Stack · Tencent Cloud · openstack · CIVO · nine cloud navigators

EQUINIX METAL · Alibaba Cloud · Scaleway · Google Cloud

DigitalOcean · DaoCloud · Azure · EXOSCALE · APPUiO SWISS CONTAINER PLATFORM · HETZNER

Use your favorite Kubernetes distribution

Azure Kubernetes Service · OPENSHIFT · Amazon EKS · RKE 2 · Google Kubernetes Engine

Gardener · kind · CRANE · kops · KUBEKEY · k8e

kubeasz · KUBERMATIC KubeOne · KUBESPRAY · FLANT Deckhouse Kubernetes Platform

Major cloud providers have now picked for Networking & Security in their Kubernetes platforms

aws

Azure

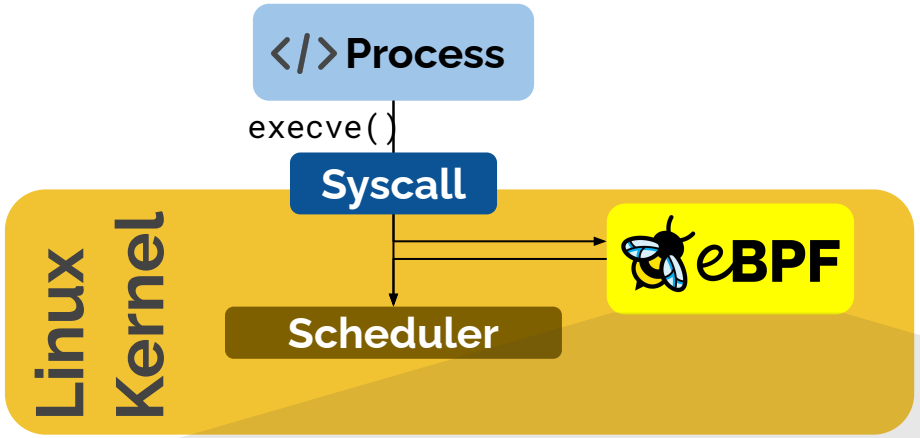
Google Cloud

Alibaba Cloud

eBPF

Makes the Linux kernel programmable in a secure and efficient way.

“What JavaScript is to the browser, eBPF is to the Linux Kernel”




```c
int syscall__ret_execve(struct pt_regs *ctx)
{
        struct comm_event event = {
                .pid = bpf_get_current_pid_tgid() >> 32,
                .type = TYPE_RETURN,
        };

        bpf_get_current_comm(&event.comm, sizeof(event.comm));
        comm_events.perf_submit(ctx, &event, sizeof(event));

        return 0;
}
```

Cilium CNI

Scalable, Secure, High Performance CNI Plugin

Cilium CNI

Scalable, Secure, High Performance CNI Plugin

Cilium Service Mesh

Sidecar-free Mesh & Ingress

Cilium CNI

Scalable, Secure, High Performance CNI Plugin

Cilium Service Mesh

Sidecar-free Mesh & Ingress

Hubble

Network Observability

Cilium CNI

Scalable, Secure, High Performance CNI Plugin

Cilium Service Mesh

Sidecar-free Mesh & Ingress

Hubble

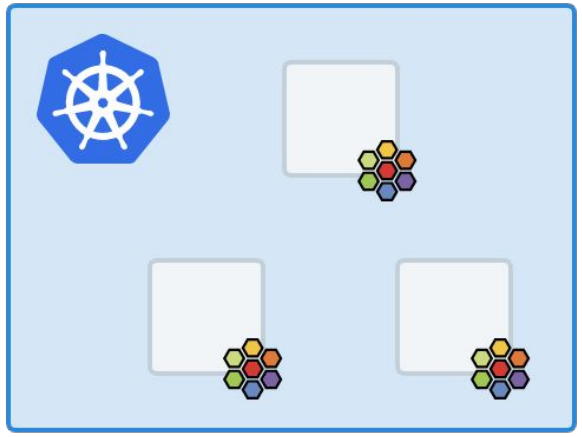
Network Observability

Tetragon

Security Observability & Runtime Enforcement

The Origins

The Beginning



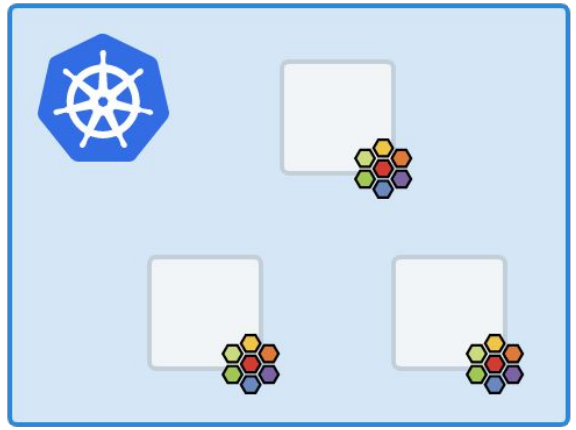
Networking

- Container Networking
- L3-L4 Load Balancing

Vision

Inent & identity-based, high performance container networking platform built using eBPF

Network Security 1.0



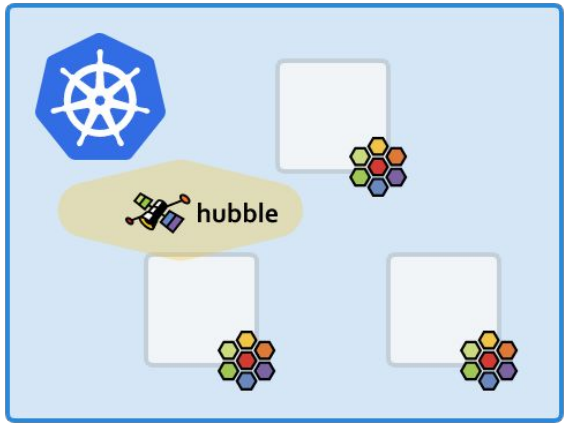
Networking

- Container Networking
- L3-L4 Load Balancing

Network Security

- Network Policy L3-L7
- Encryption

Hubble



Networking

- Container Networking
- L3-L4 Load Balancing

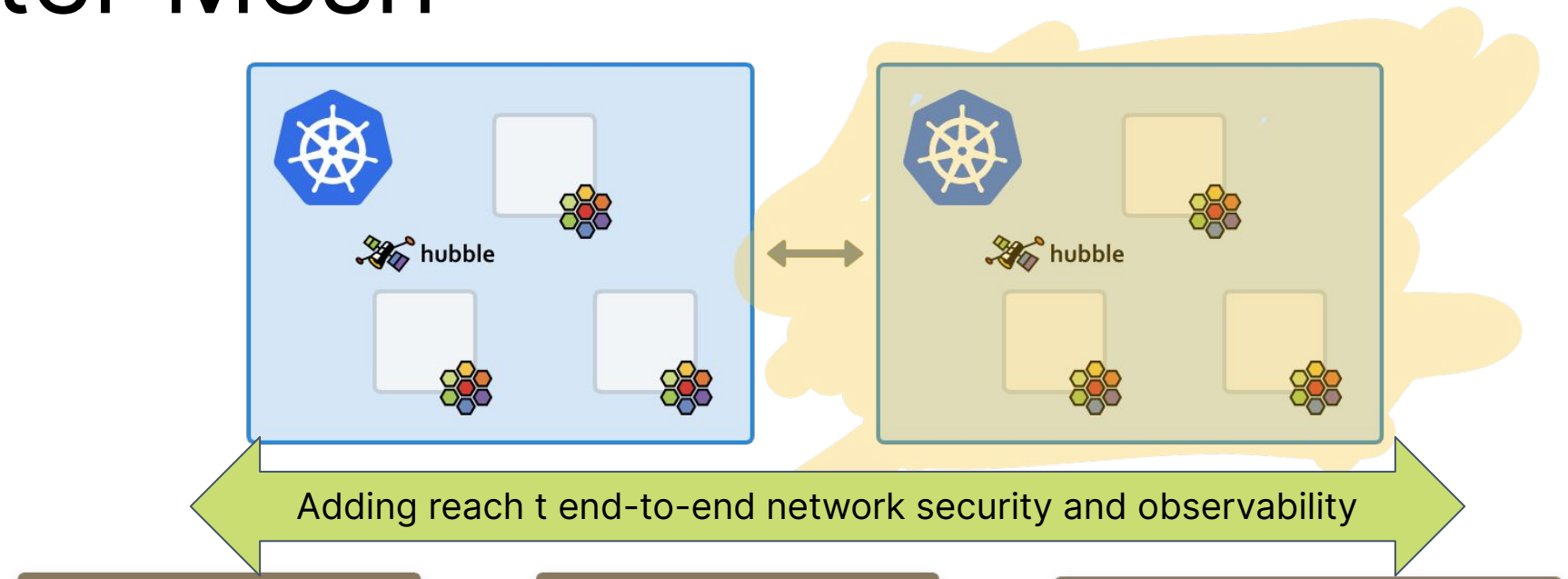
Network Security

- Network Policy L3-L7
- Encryption
- SIEM Integration

Network Observability

- Flow Logs
- Metrics
- Troubleshooting

Cluster Mesh



Networking

- Container Networking
- L3-L4 Load Balancing
- Multi-Cluster

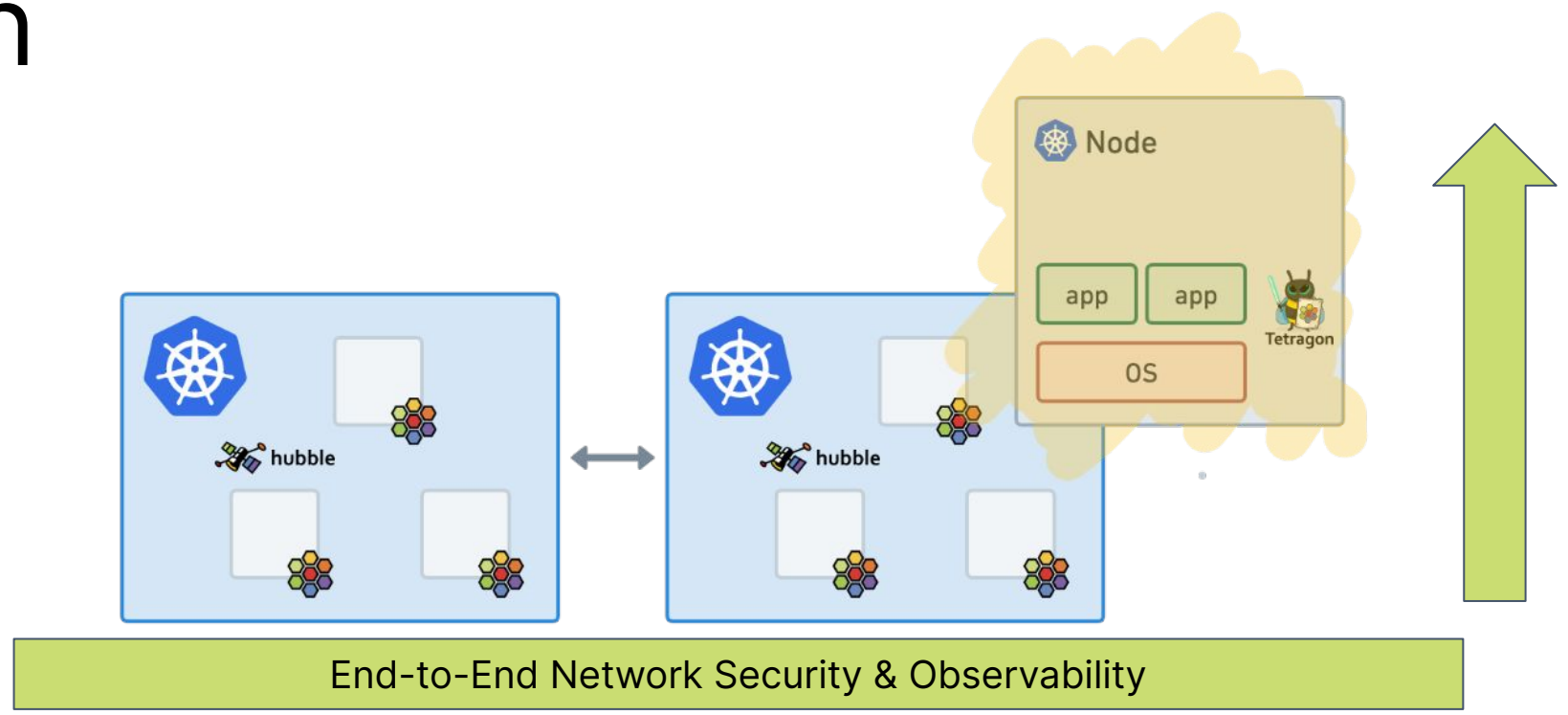
Network Security

- Network Policy L3-L7
- Encryption
- SIEM Integration

Network Observability

- Flow Logs
- Metrics
- Troubleshooting

Tetragon



Networking

- Container Networking
- L3-L4 Load Balancing
- Multi-Cluster

Network Security

- Network Policy L3-L7
- Encryption
- SIEM Integration

Network Observability

- Flow Logs
- Metrics
- Troubleshooting

Runtime Security

- Runtime Enforcement

Runtime Observability

- Syscall, File, Privilege, and Network Observability
- SIEM Integration

Tetragon

Security Observability & Runtime Enforcement

CLOUD NATIVE COMPUTING FOUNDATION

Metrics

Events

SIEM

fluentd

Logs

Traces

JSON

Tetragon Agent

Linux Kernel

Kernel Runtime

Smart Collector

Stack Traces

Ring Buffer

Metrics

Hash Maps

Process Execution

Syscall Activity

System Calls

File Access

VFS

Seq Attack

TCP/IP

NS Escapes

Priv Escalations

Namespaces

Data Access

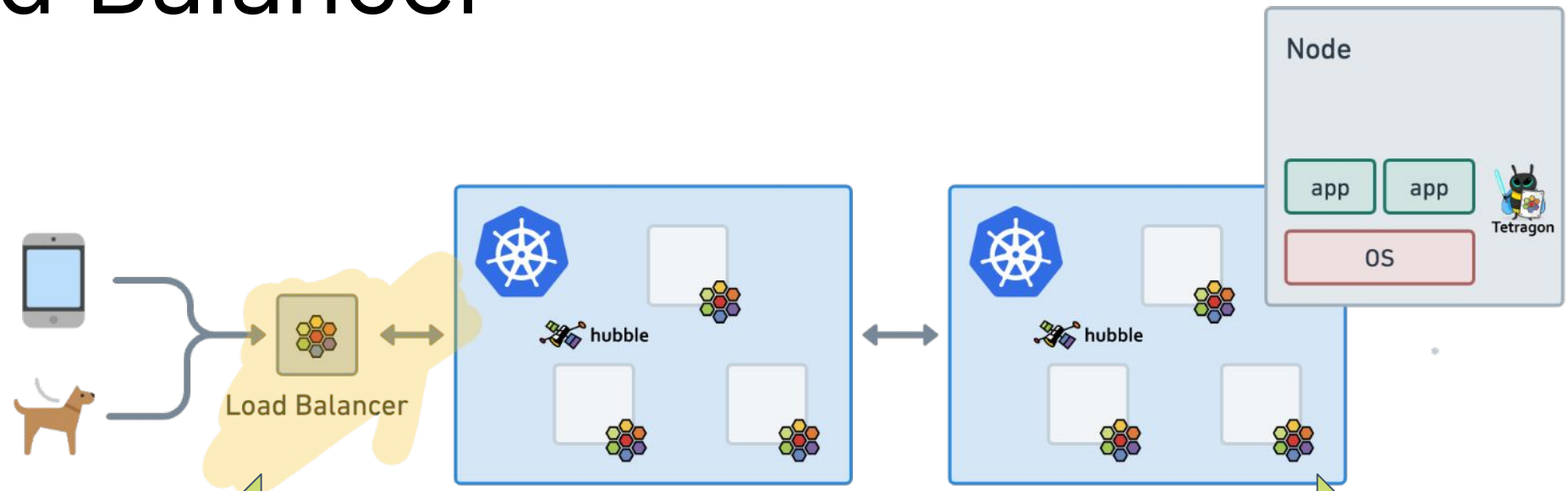
Storage

HTTP, DNS, TLS

eth0

Network

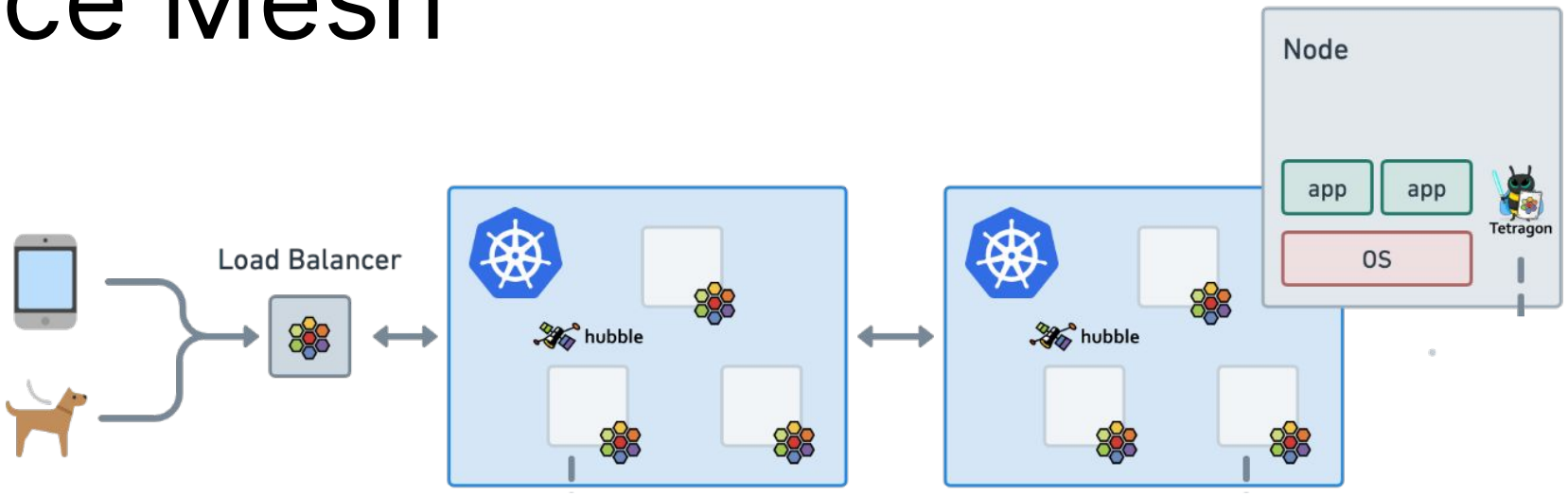
Load Balancer



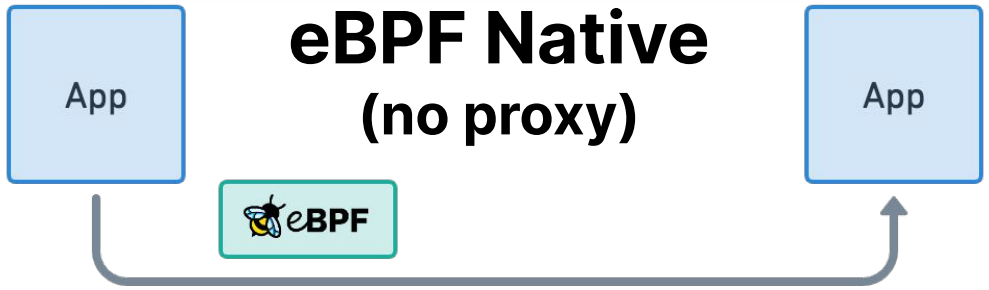
Adding reach to end-to-end network security and observability

Networking	Network Security	Network Observability	Runtime Security	Runtime Observability
• Container Networking • L3-L4 Load Balancing • Standalone LB • Multi-Cluster	• Network Policy L3-L7 • Encryption • SIEM Integration	• Flow Logs • Metrics • Troubleshooting	• Runtime Enforcement	• Syscall, File, Privilege, and Network Observability • SIEM Integration

Service Mesh



Networking	Network Security	Network Observability	Service Mesh	Runtime Security	Runtime Observability
• Container Networking • L3-L4 Load Balancing • Standalone LB • Multi-Cluster	• Network Policy L3-L7 • Encryption • SIEM Integration	• Flow Logs • Metrics • Troubleshooting • Network Time Machine	• L7 Load Balancing • Tracing	• Runtime Enforcement	• Syscall, File, Privilege, and Network Observability • SIEM Integration • Analytics



Whenever possible

Traffic Management

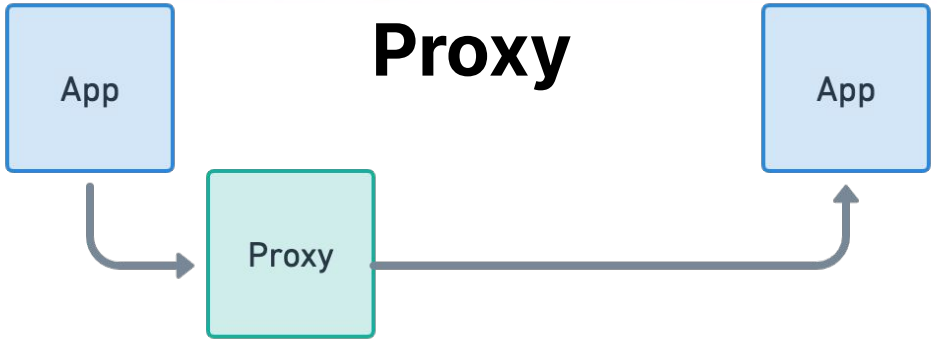
- L3/L4 forwarding & Load-balancing
- Canary, Topology Aware Routing
- Multi-cluster

Security

- Network Policy
- mTLS

Observability

- Tracing, OpenTelemetry, & Metrics
- HTTP, TLS, DNS, TCP, UDP, ...



When eBPF cannot do it

Traffic Management

- L7 Load-balancing & Ingress

Resilience

- Retries, L7 Rate Limiting

Security

- TLS Termination & Origination
- L7 Network Policy*

Cilium 1.14 and beyond

Roadmap Highlights

- Mutual Authentication for NetworkPolicy
- SPIFFE Integration
- Day 2 Operations Enhancements
- Grafana Dashboards in Hubble UI
- Istio Ambient Mesh & zTunnel integration
- Did we mention more Grafana already?
- Cilium Mesh

Roadmap Highlights

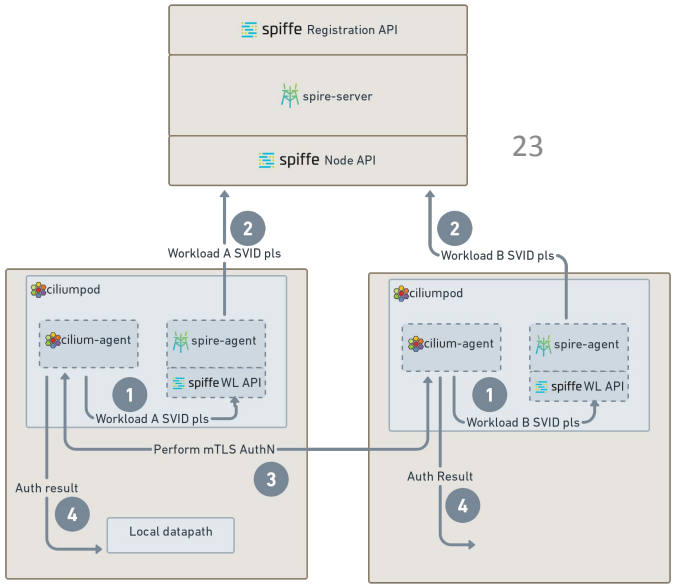
Mutual Authentication

Mutual Auth via just Network Policy

```
apiVersion: cilium.io/v2
kind: CiliumNetworkPolicy
metadata:
  name: frontend-backend
spec:
  endpointSelector:
    matchLabels:
      role: backend
  ingress:
    - fromEndpoints:
      - matchLabels:
          role: frontend
      auth:
        required: strict
```

SPIFFE Integration

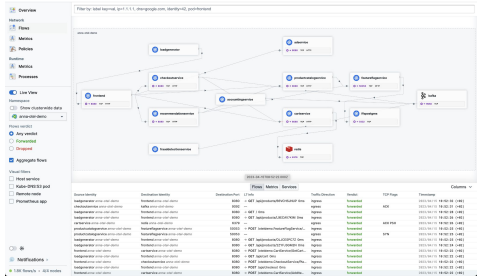
Certificate management via SPIFFE/SPIRE + SPIFFE ID selector matching



Day 2 Ops

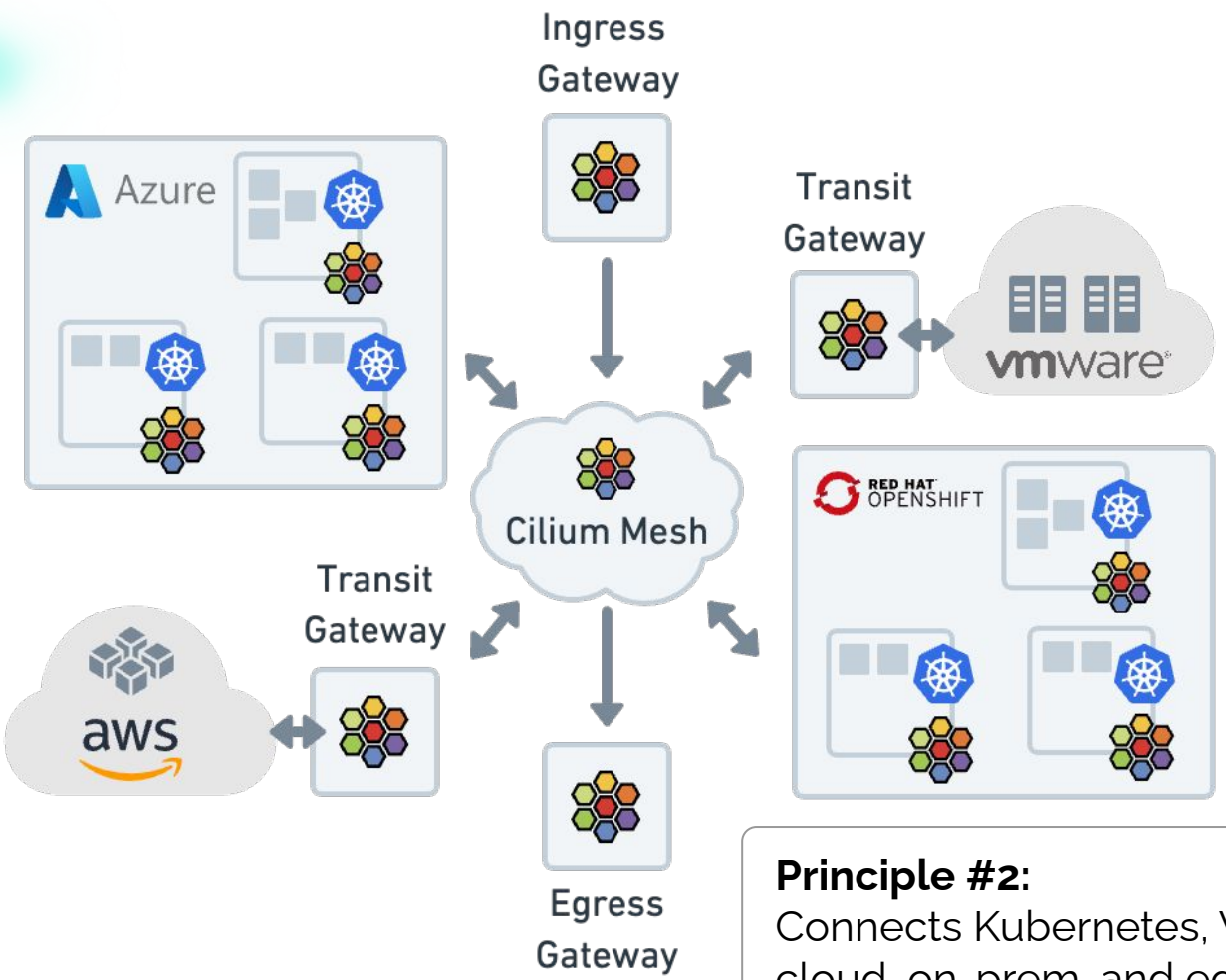
cilium

Assisted monitoring, proactive troubleshooting, simplified day 2 ops



Cilium Mesh

One Mesh to Connect Them All



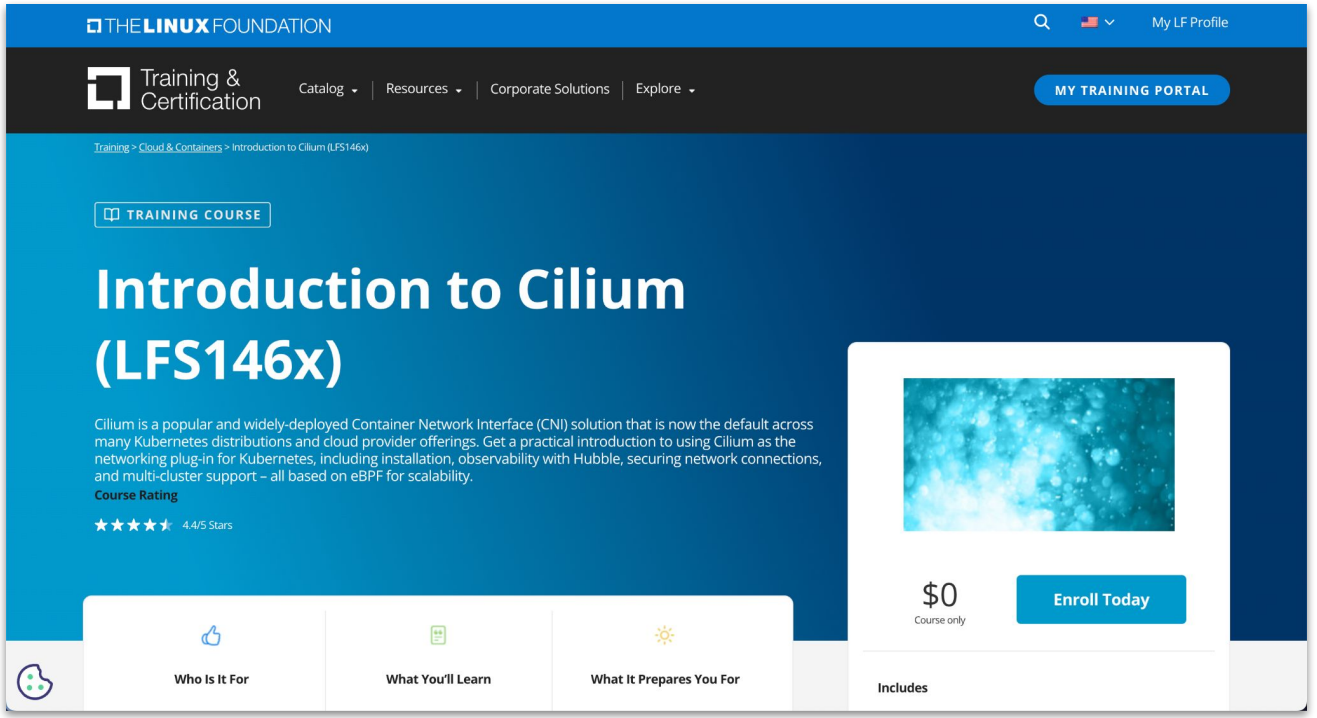
Principle #1:
Combines all Cilium components into a single mesh:

- Kubernetes Networking (CNI)
- Cluster Mesh (Multi-Cluster)
- Ingress & Egress Gateway
- Load Balancer
- Service Mesh

Principle #2:
Connects Kubernetes, VMs, and Servers across cloud, on-prem, and edge.

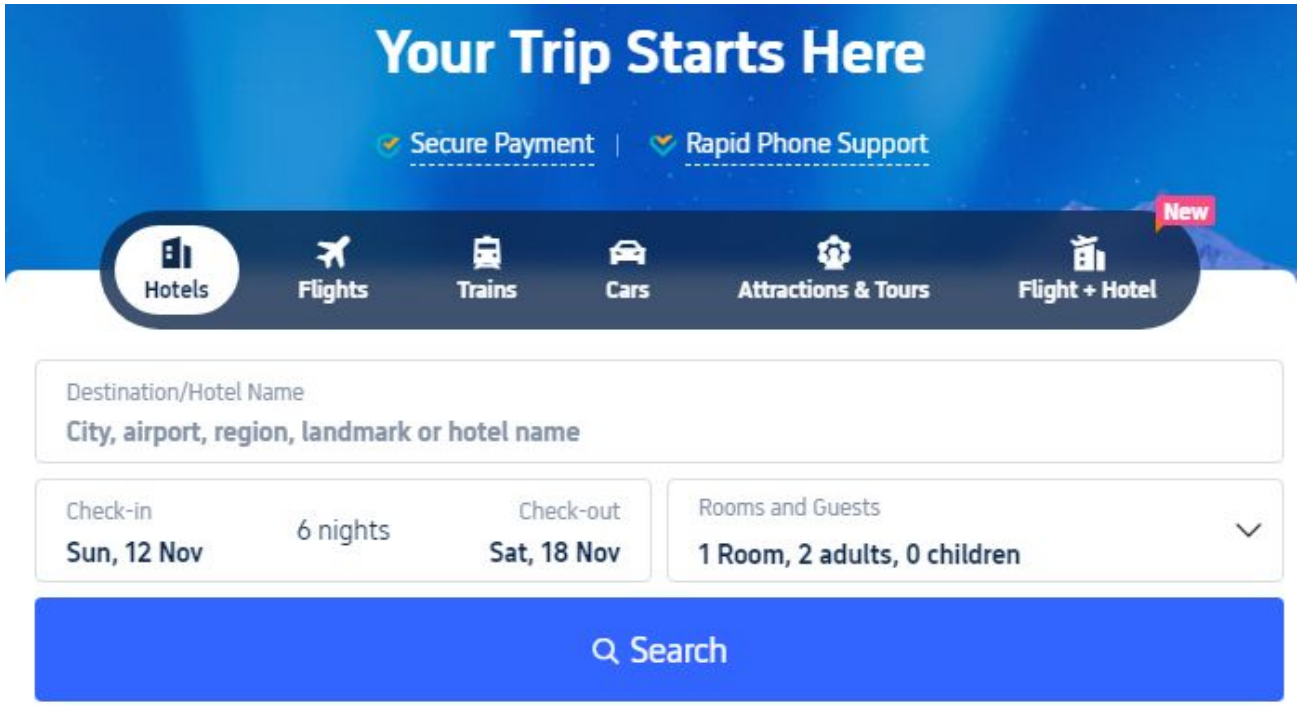
LF Intro to Cilium Course

training.linuxfoundation.org - LFS146x



About Trip.com Group

- Leading online travel platform
- Flights, hotels, tours and more
- 400 million users worldwide



Cloud team @Trip.com Group

- R&D of cloud infra over the globe
- Virtualization, networking, storage, security

Trip.com Ctrip Skyscanner QUNAR Travix

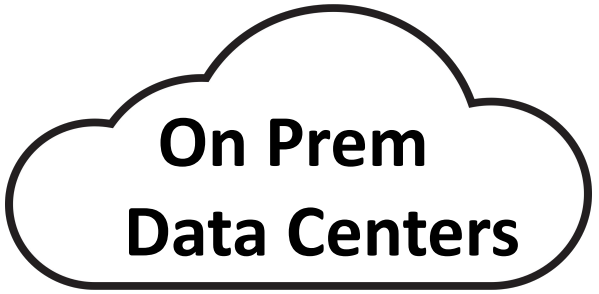
Cilium at Trip.com

20K Nodes

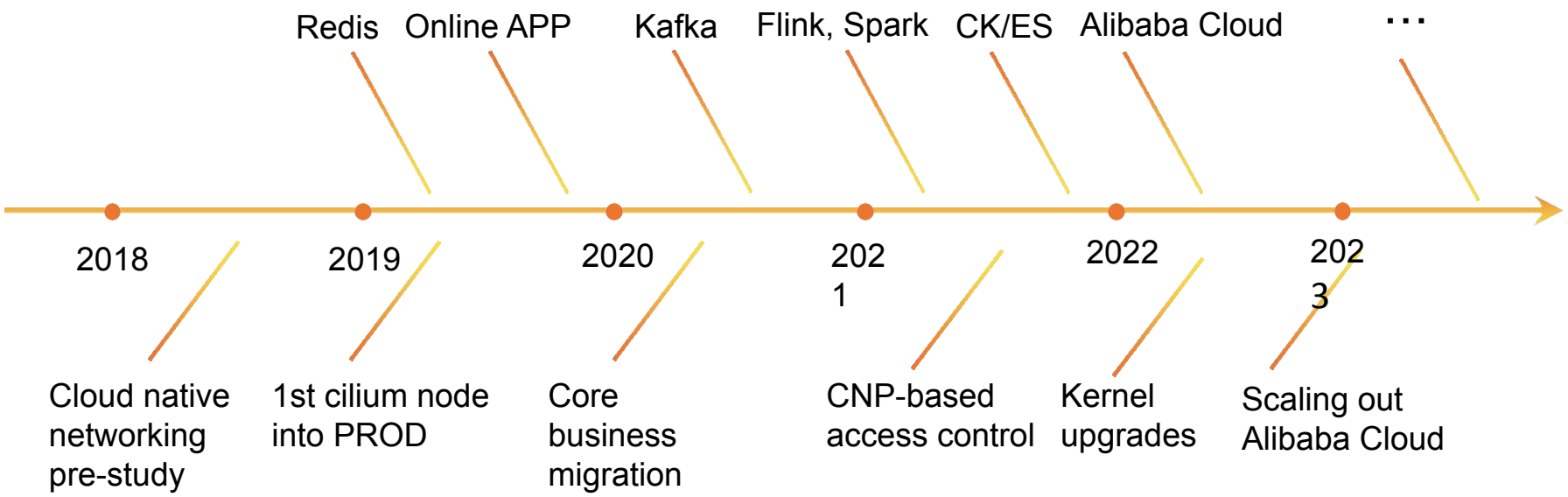
350K Pods

3K Network Policy

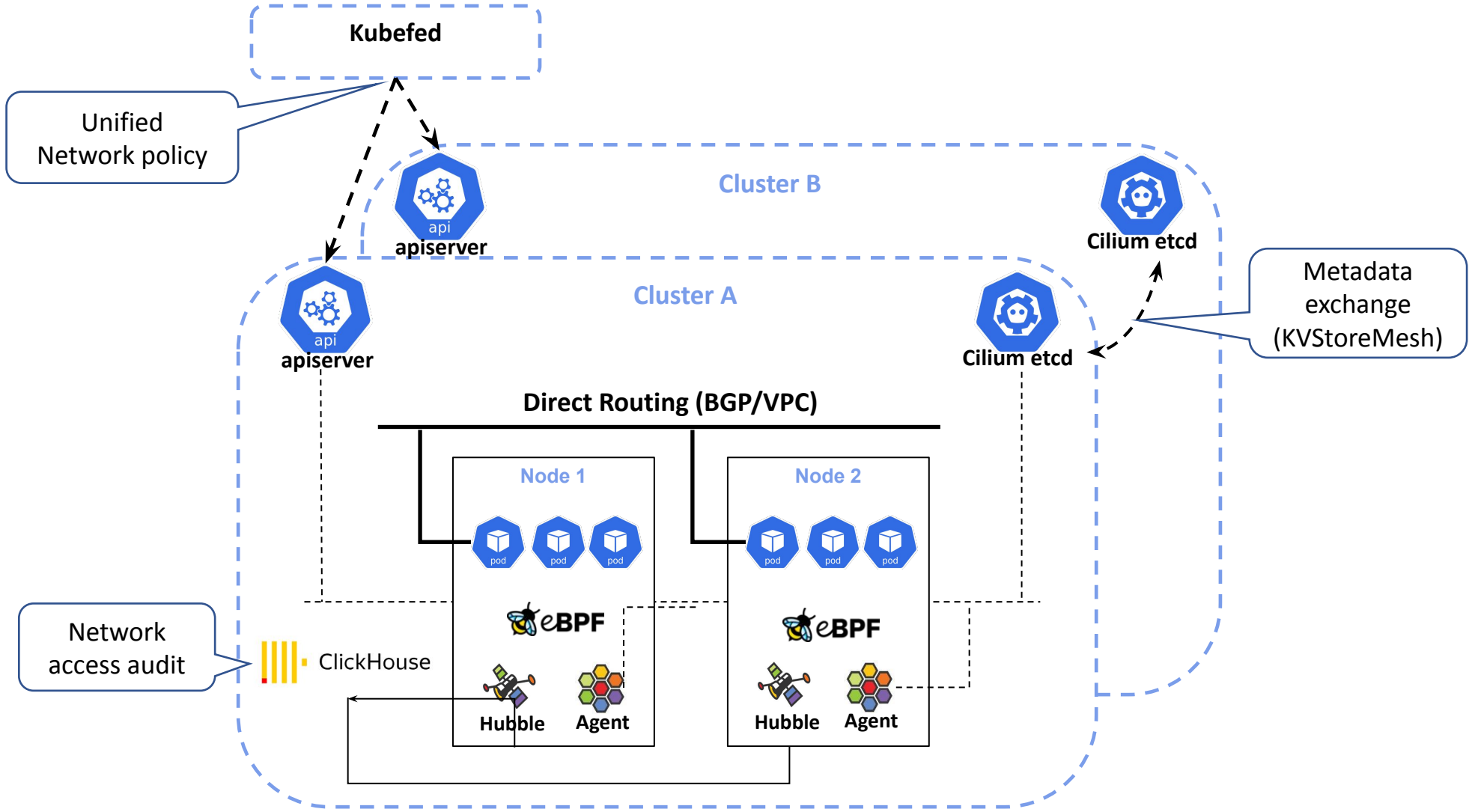
200K Hubble events/s



History


Redis
Online APP
Kafka
Flink, Spark
CK/ES
Alibaba Cloud
...
2018
2019
2020
2021
2022
2023
Cloud native networking pre-study
1st cilium node into PROD
Core business migration
CNP-based access control
Kernel upgrades
Scaling out Alibaba Cloud

Deployment overview

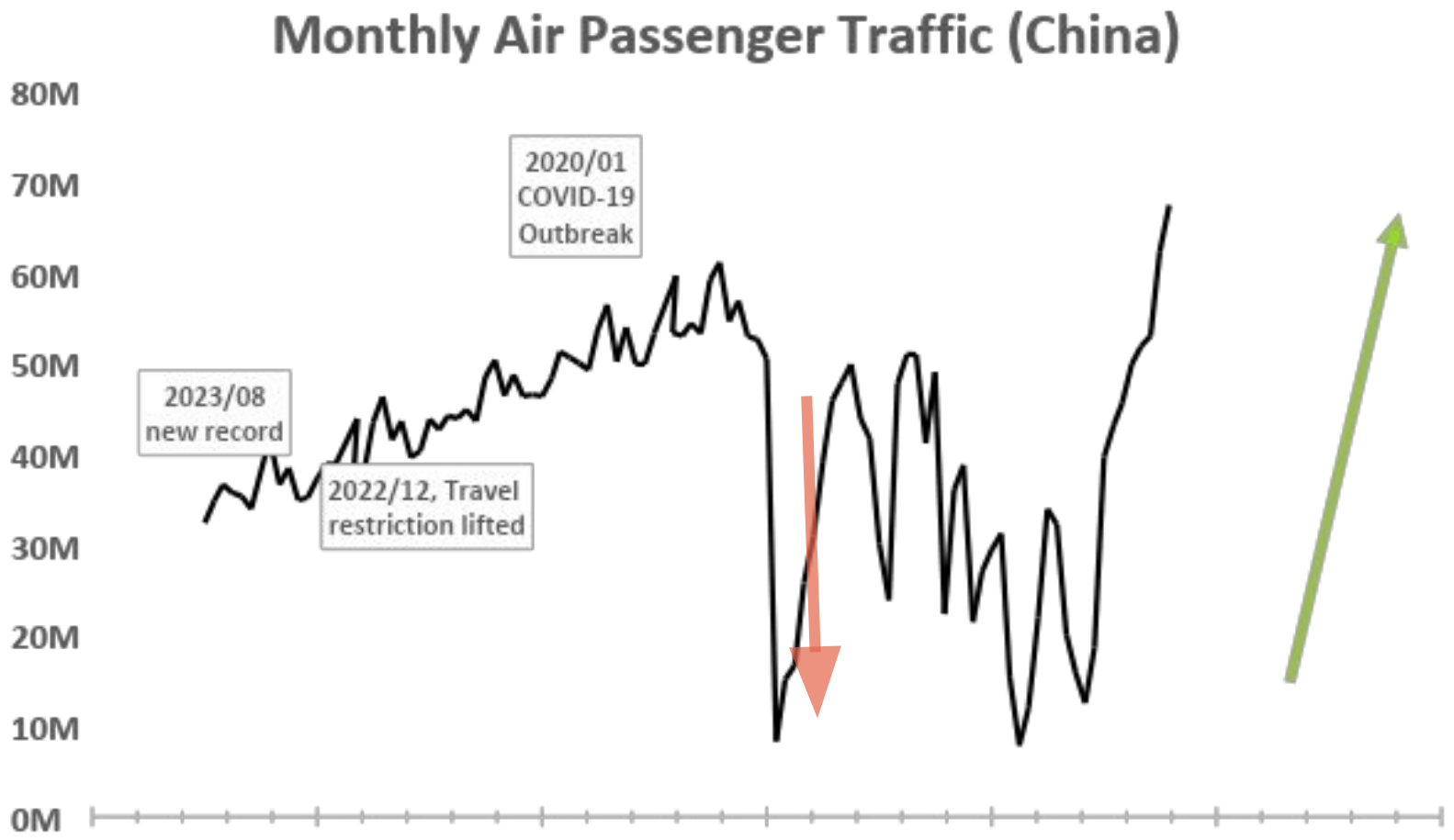


Cilium brings us

- Cluster scalability (up 6000+ nodes/cluster)
- Data plane stability (4+ years in PROD)
- K8s-Native features (network policy, LB)
- Network observability
- Consistent experience (user & admin, tech stack, CNI)

Embracing change

- **Cost Efficiency**
- **Auto-scaling**
- **Capacity**



Data source: Civil Aviation Administration of China

Cilium on Alibaba Cloud

3K ~ 5K Auto-scaling nodes

10K~40K Pods

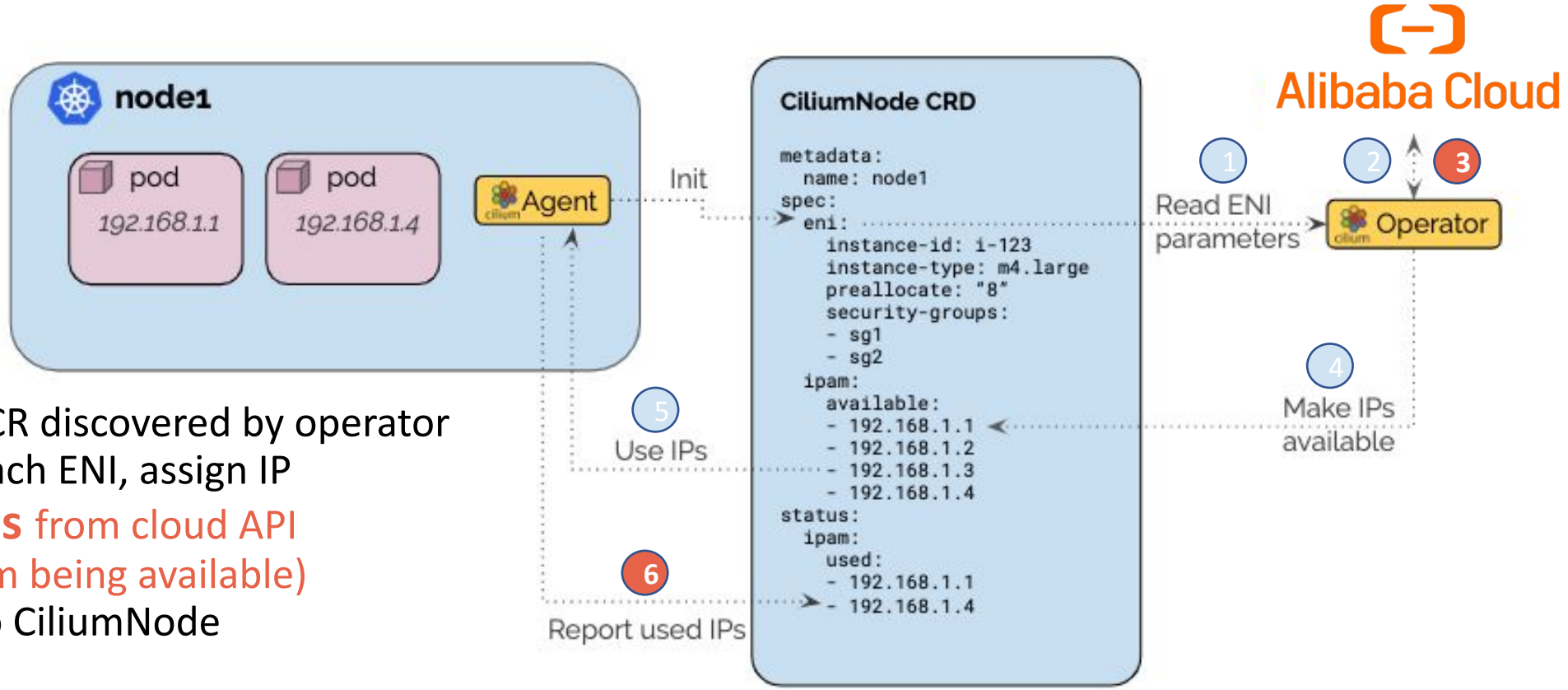
100 Nodes concurrent creation supported

< 1 min From pod creation to running on a triggered node

Challenges

- **Node/pod scaling performance**
  - Cilium operator: slow ENI resync during IP allocation
  - Cilium agent: delayed IP deficit status report
  - Scheduling: IP availability not considered by kube-scheduler
- **Control plane resiliency**
  - Potential thundering herd
  - Network policy degradation plan
- **Early user of *--ipam=alibabacloud***
  - Encountered panic on the first run
  - Bug fixes

ENI IPAM Architecture



1. CiliumNode CR discovered by operator
2. Operator attach ENI, assign IP
3. **Resync ENIs** from cloud API (blocks IP from being available)
4. Update IPs to CiliumNode
5. Agent use IPs
6. **Report** used IPs(Delayed by 15s interval)

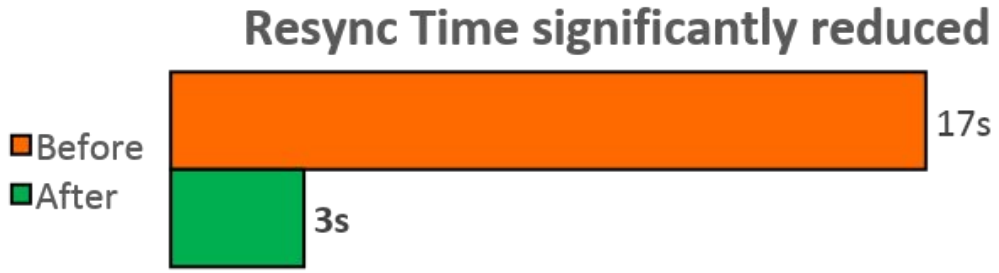
Image source: docs.cilium.io/en/stable/network/concepts/ipam/eni

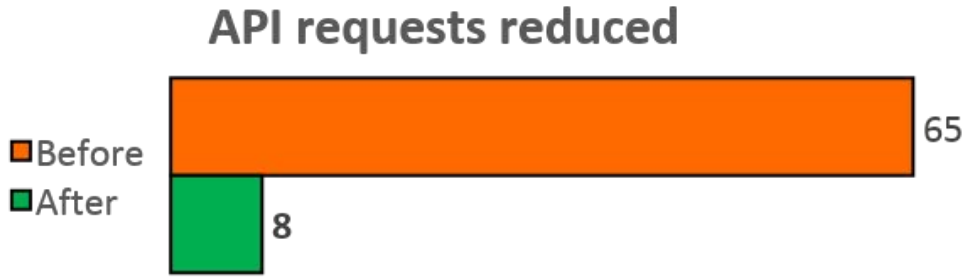
Operator improvements

1. Drop unused subnet filter in ENI resync

```
for subnet in subnets {
    // List ENIs from individual subnets
    DescribeNetworkInterfaces(subnet)
}
```

```
// List all ENIs
DescribeNetworkInterfaces()
```





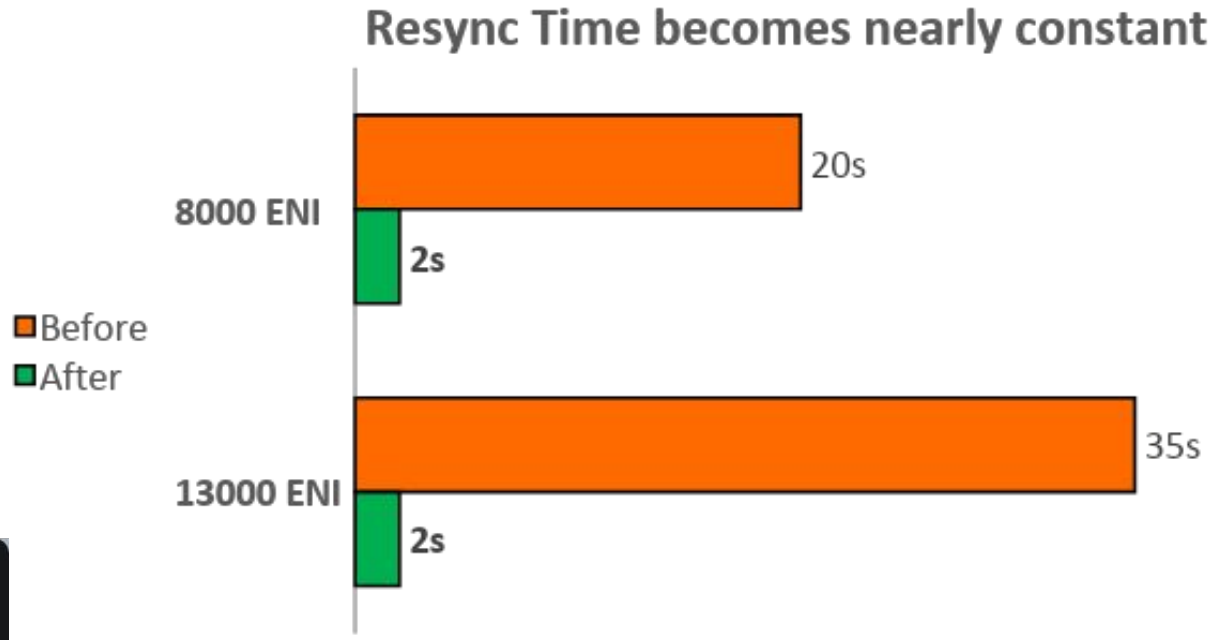
Data from a cluster with 1500 ENIs, 40+ subnets

Operator improvements

2. Resync ENIs selectively on relevant instance

```
// List all ENIs
DescribeNetworkInterfaces()
```

```
// List ENIs on instance
DescribeNetworkInterfaces(instance)
```



Data from a cluster during different periods

Agent improvements

Accelerate agent startup and node bootstrap

- **Add a new flag *--ipam-cilium-node-update-rate* to configure the delay of IP usage report (hardcoded as 15s previously)**

- **Fix multiple issues that cause agent fatal on initialization**
  - Race condition in CiliumNode update
  - Enlarge cloud API quotas to avoid API rate limit

Scheduling Problems

- **IP address availability is not considered by kube-scheduler**
  - Pod could stuck in ***ContainerCreating*** while other nodes are available

  https://github.com/kubernetes/enhancements/issues/611

- **Pod IP pre-warming is limited by IP address space**

- **Different node types have different IP capacities**
  - PodCIDR /24, /25, /26; Cloud provider flavors…

IP capacity-aware Scheduling

Solution: *Extended Resources + Device Plugin*

1. Device plugin read IPAM status from cilium-agent

```
$ curl --unix-socket /path/to/cilium.sock http://localhost//v1/healthz | jq -r .ipam.status
IPv4: 2/10 allocated,
```

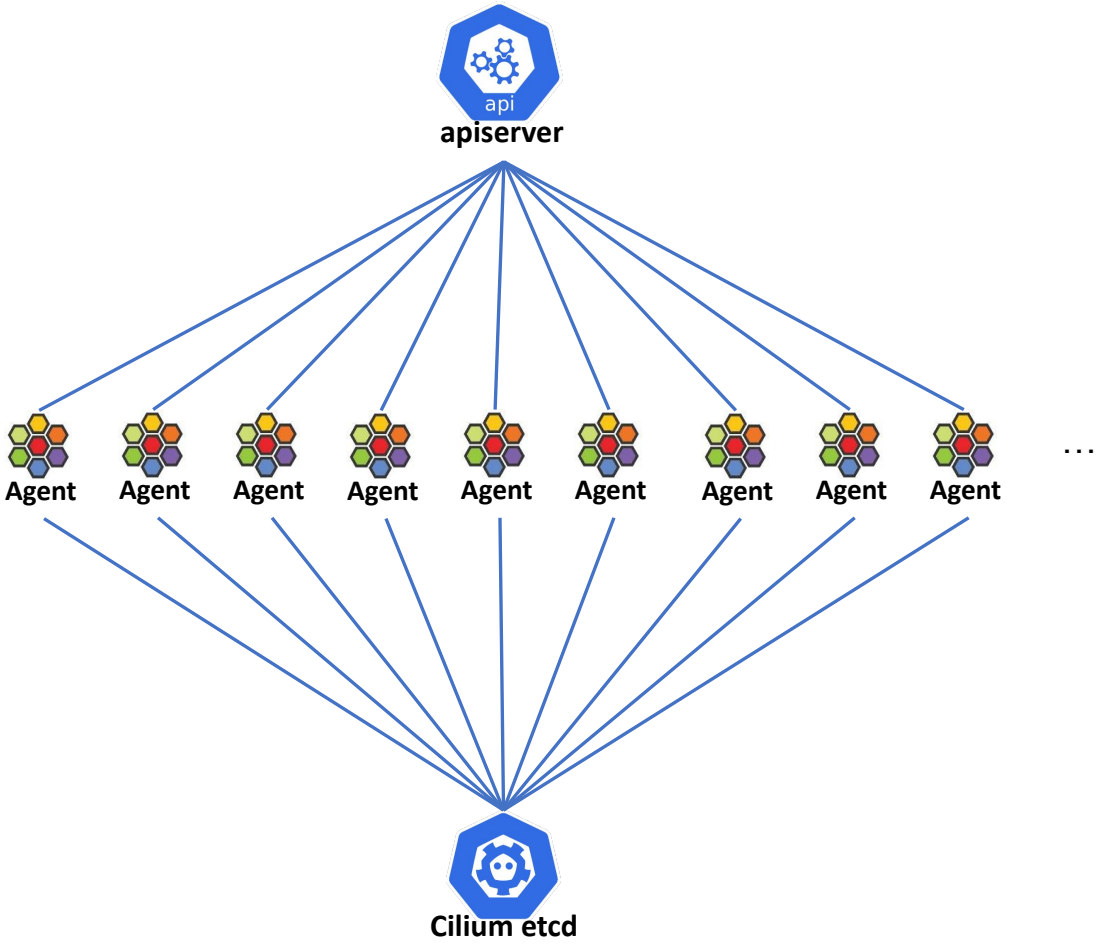
2. Calculate & Advertise IP capacity to apiserver

```
kind: Node
...
status:
  allocatable:
    example.com/ip: "8"
    cpu: "40"
    ...
```

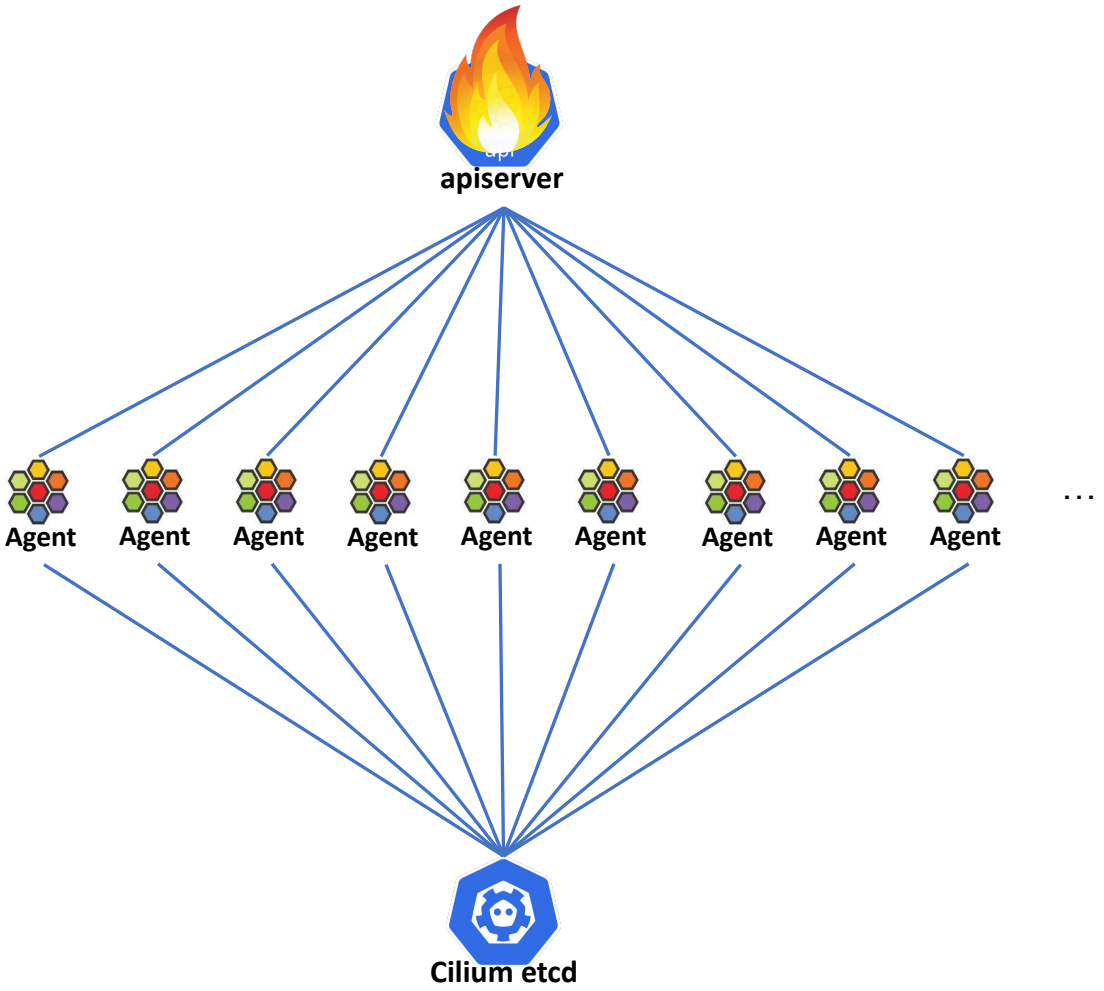
3. Webhook injector adds IP requests to non-hostNetwork pod spec

```
resources:
  limits:
    example.com/ip: "1"
    cpu: "4"
    memory: 8Gi
  requests:
    example.com/ip: "1"
    cpu: "4"
    memory: 8Gi
```

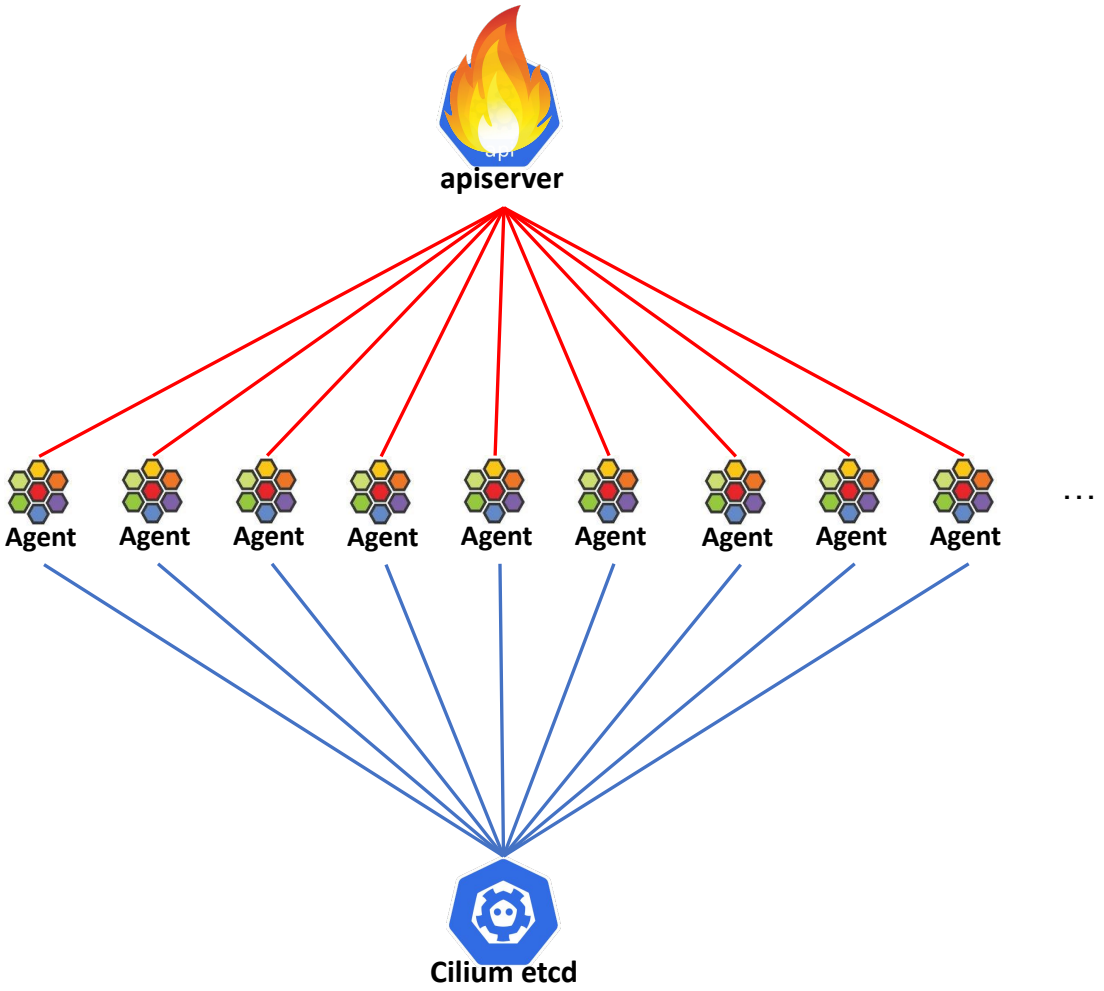
Thundering herd



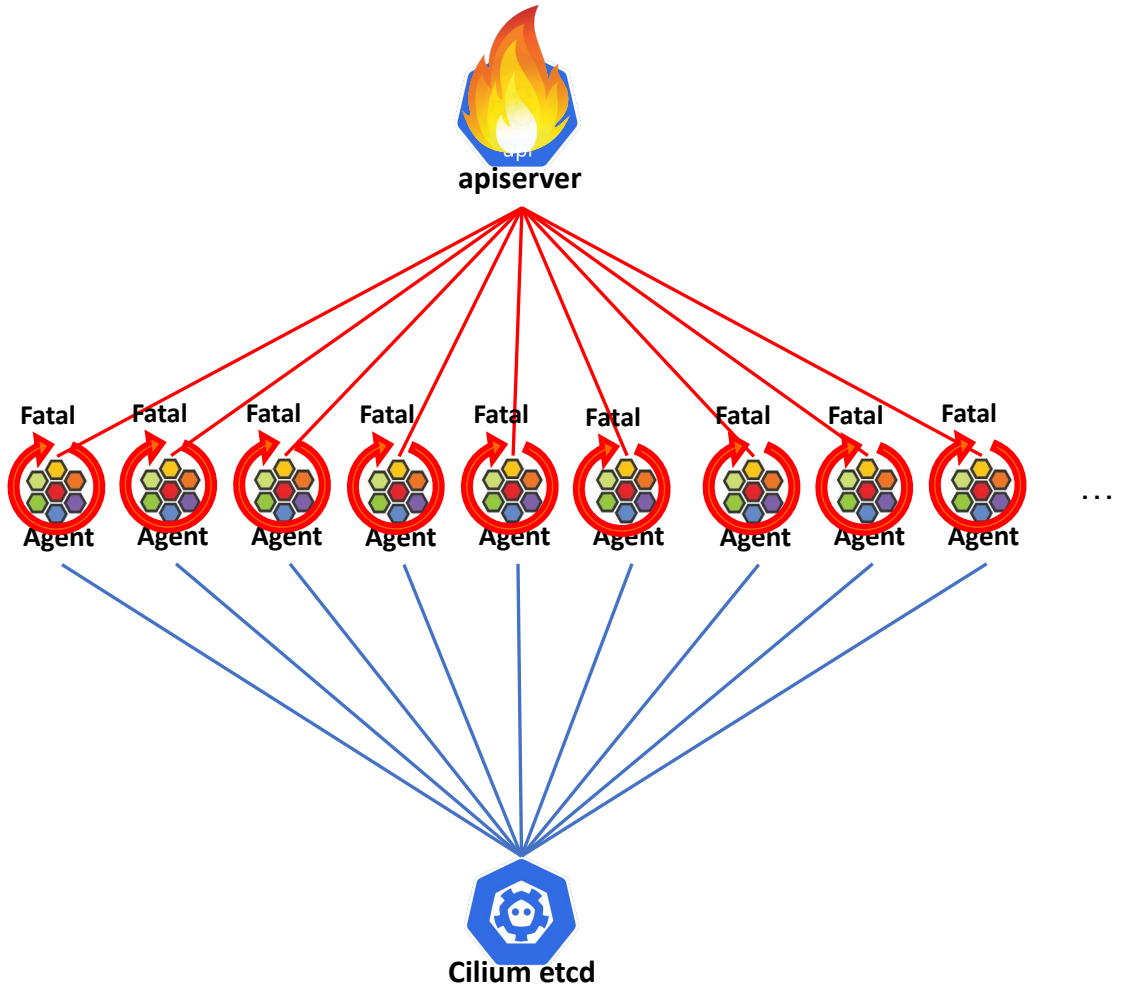
Thundering herd



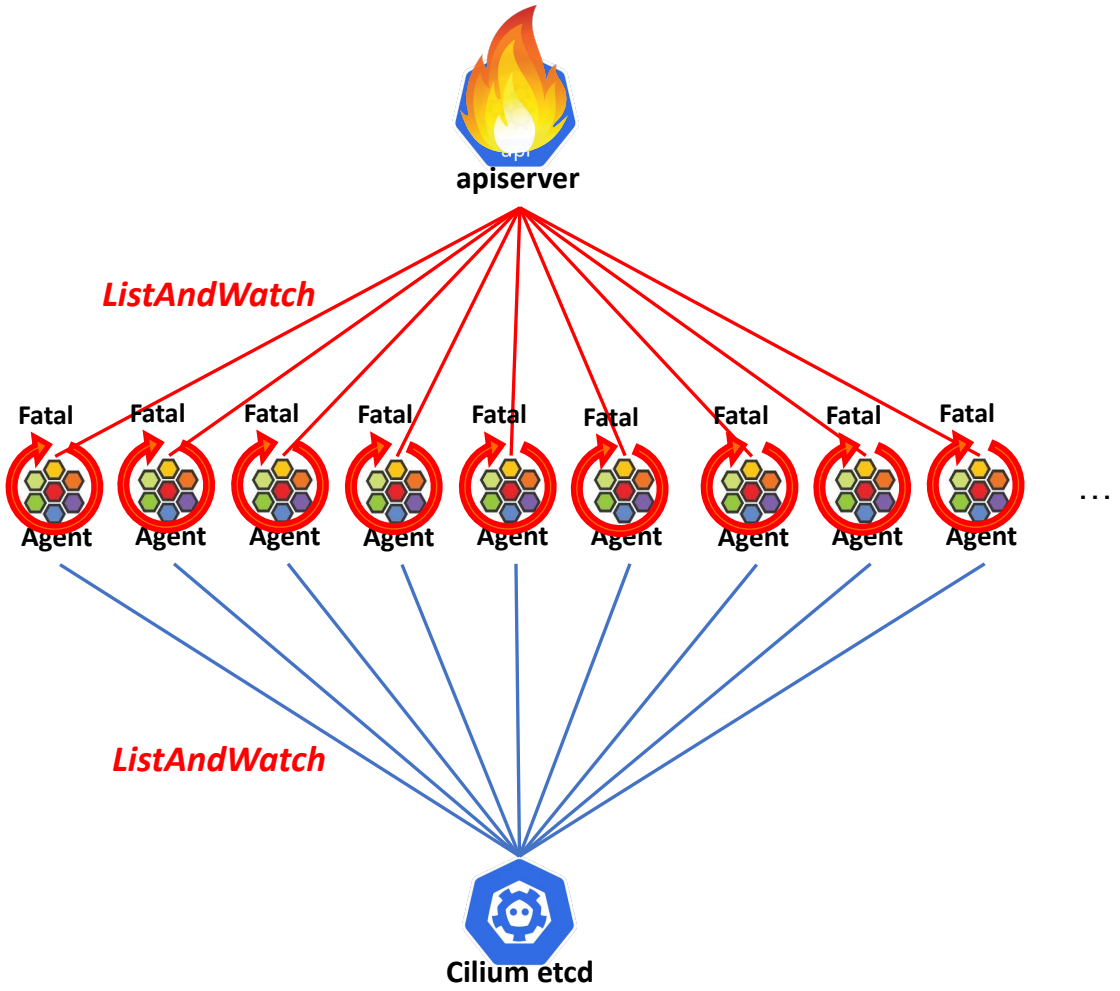
Thundering herd



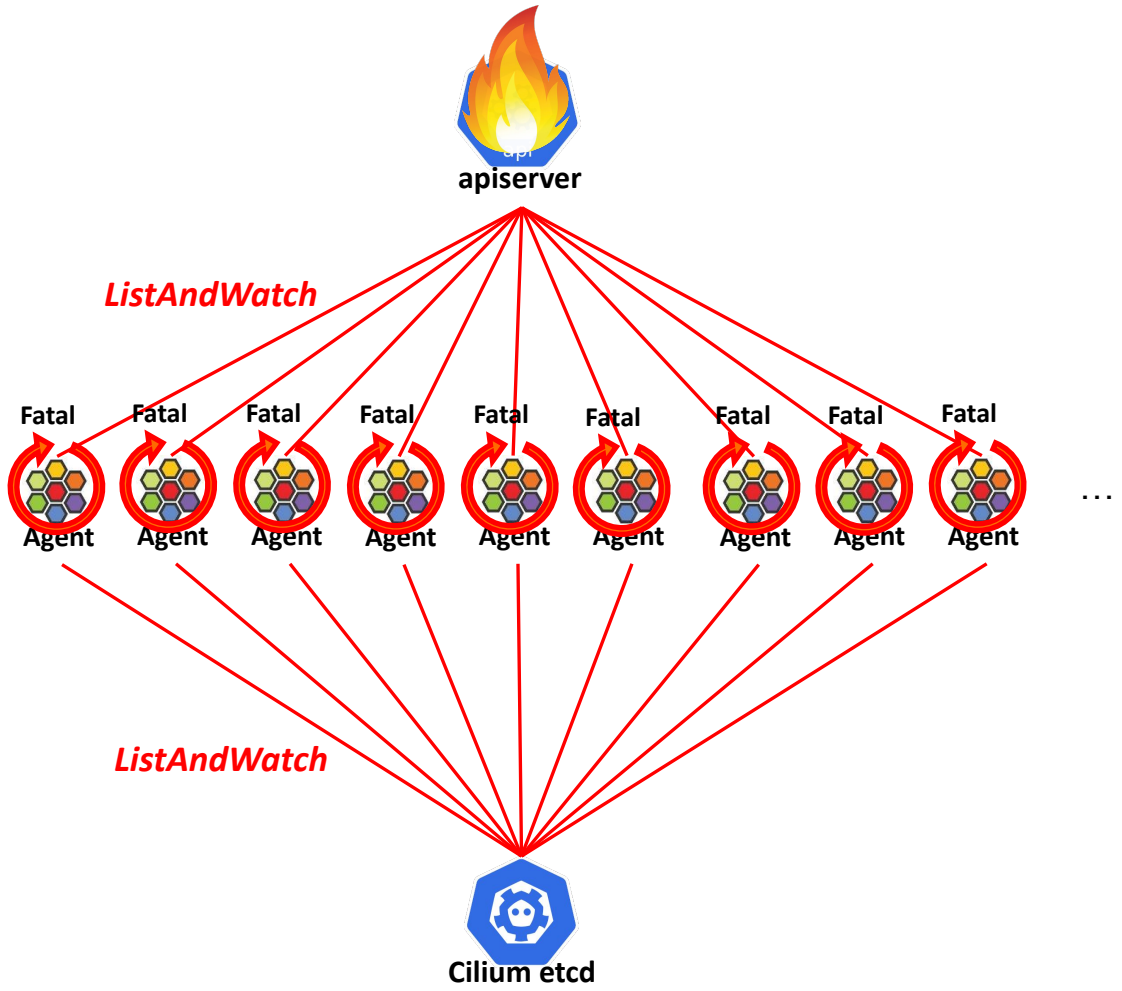
Thundering herd



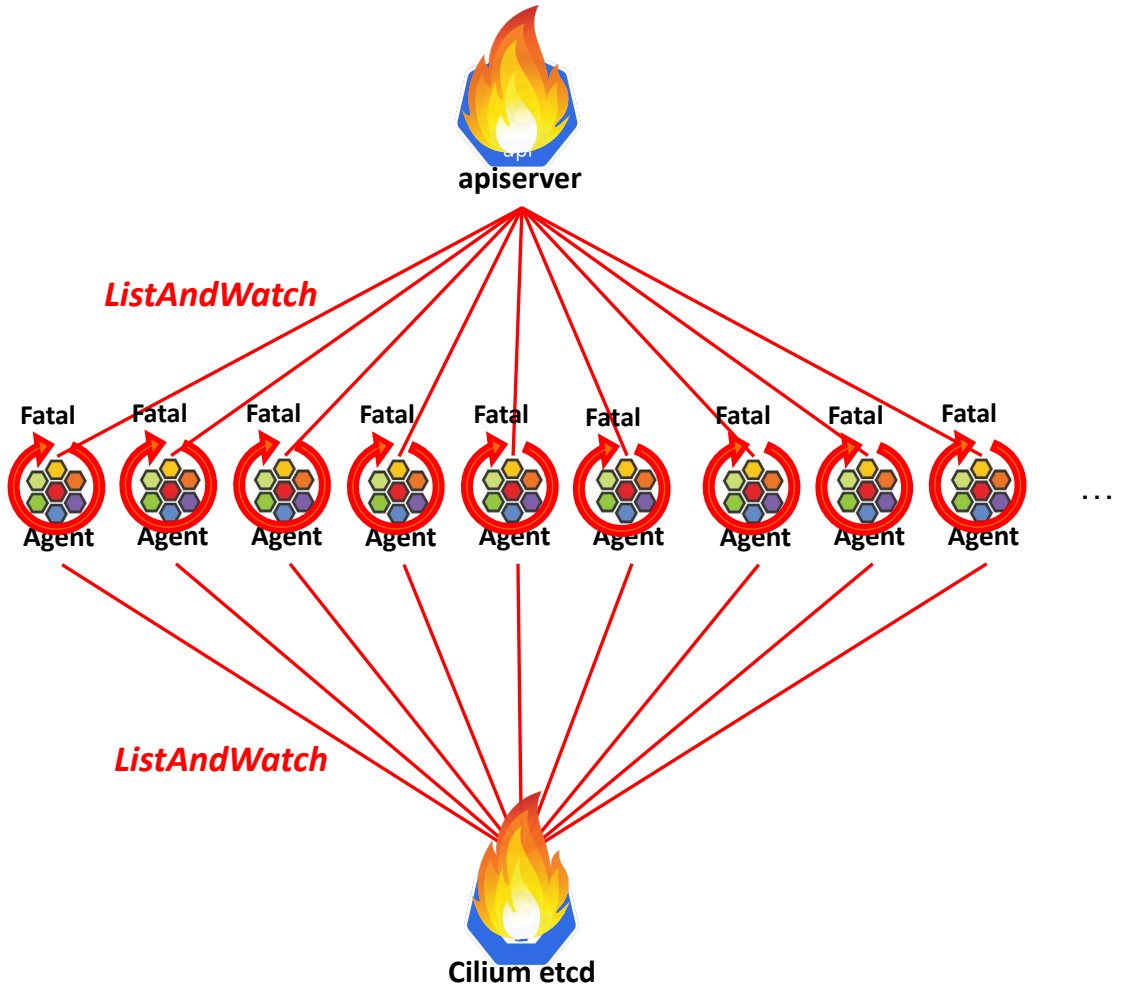
Thundering herd



Thundering herd



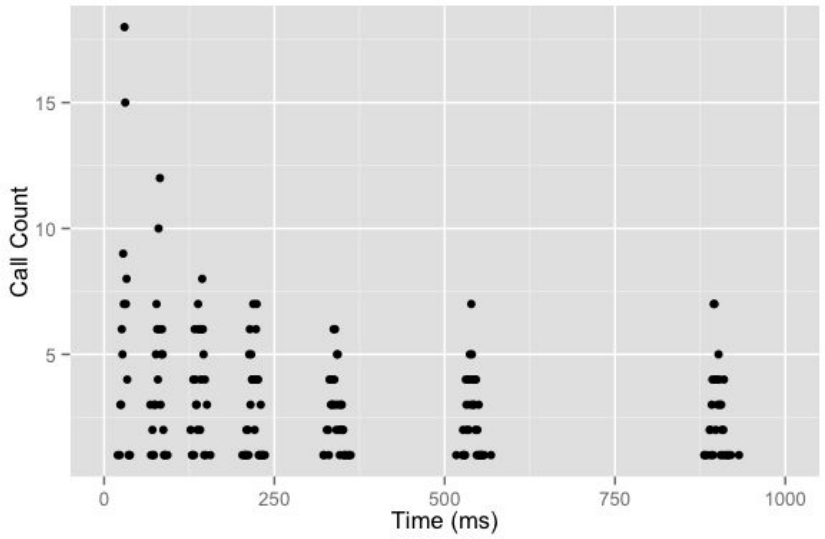
Thundering herd



Limitations of Pod container

- **Exponential backoff**
  - Backoff step too small for a cluster of agents: 10s, 20s, 40s, … capped at 5m
- **Randomness**
  - Restart policy doesn't support jitter

Without jitter



With jitter

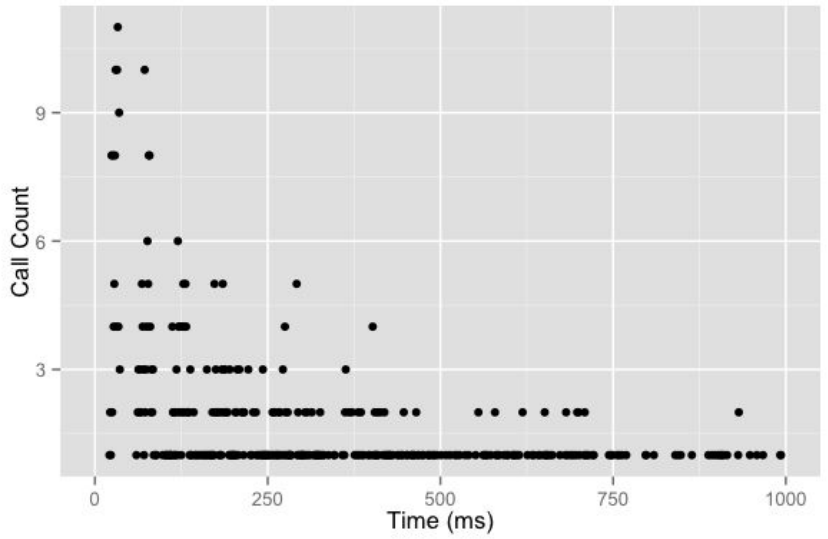
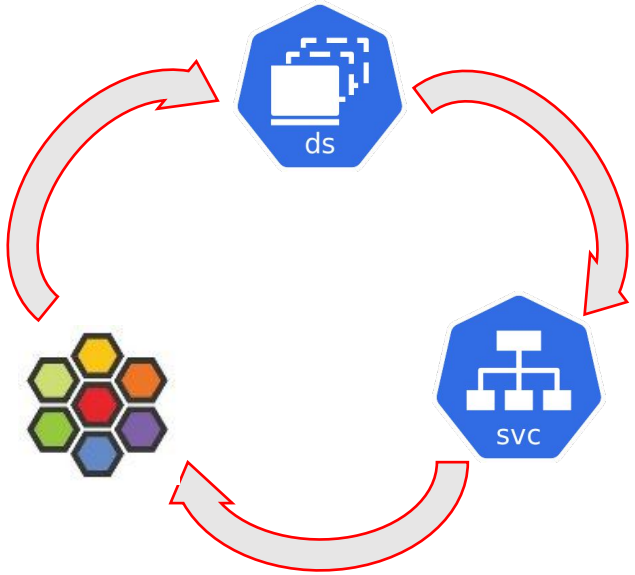
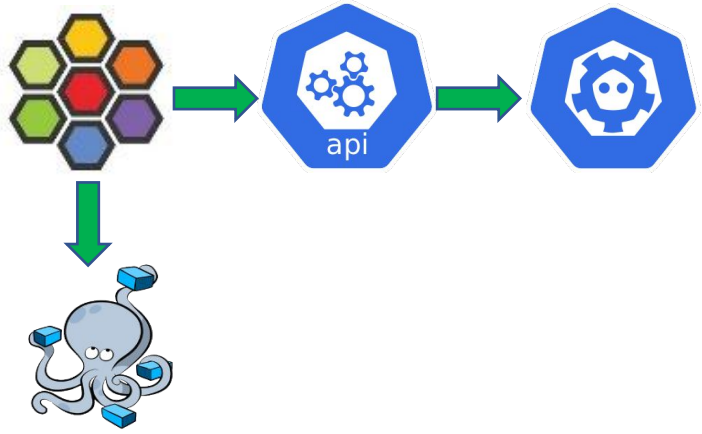


Image source: aws.amazon.com/cn/blogs/architecture/exponential-backoff-and-jitter/

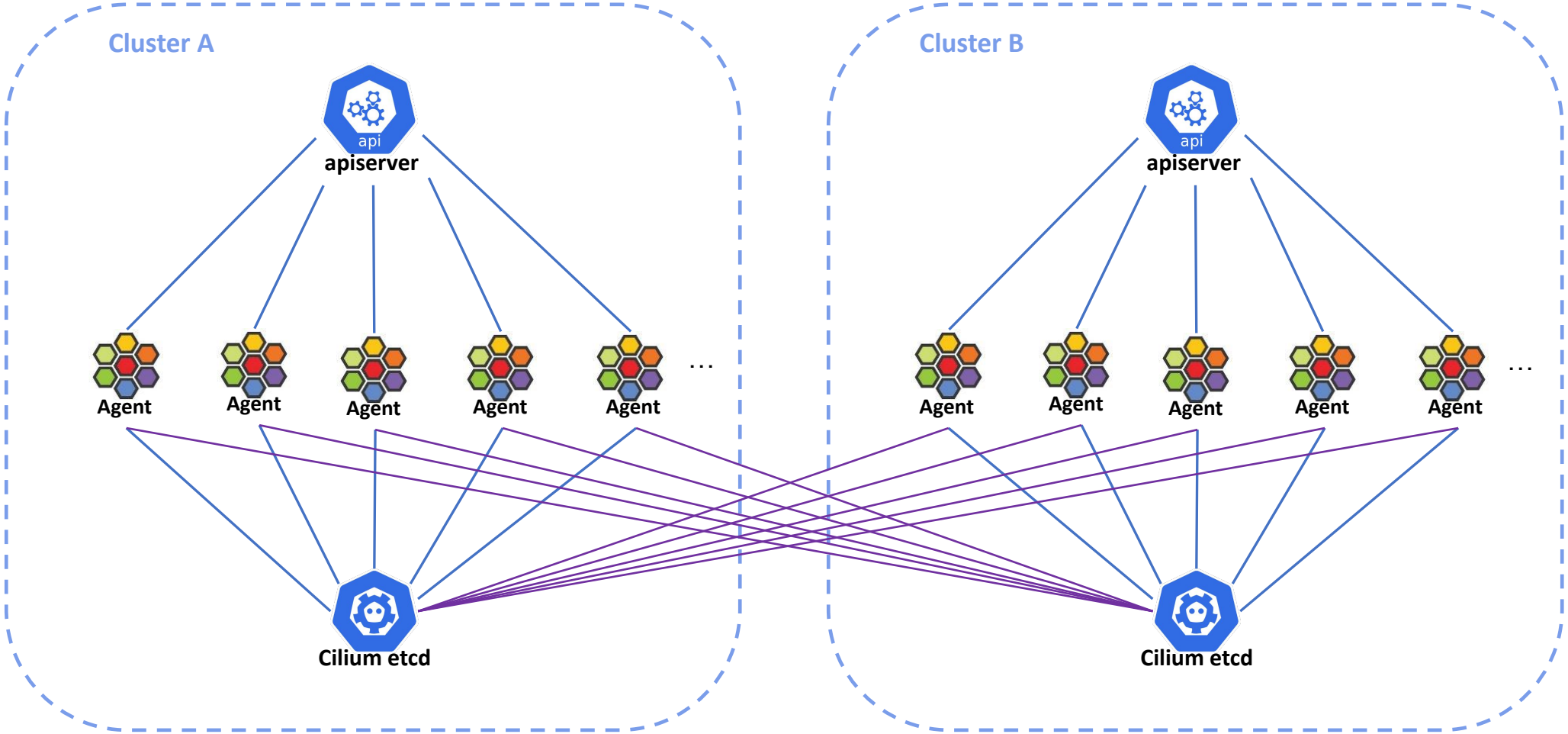
Docker Compose

- **Full control of custom backoff step and jitter**
- **Better control over canary release**
- **Decouple deployment, avoid chicken-egg problems**
  - imagine creating an agent pod depends on a Webhook service (rare but possible)

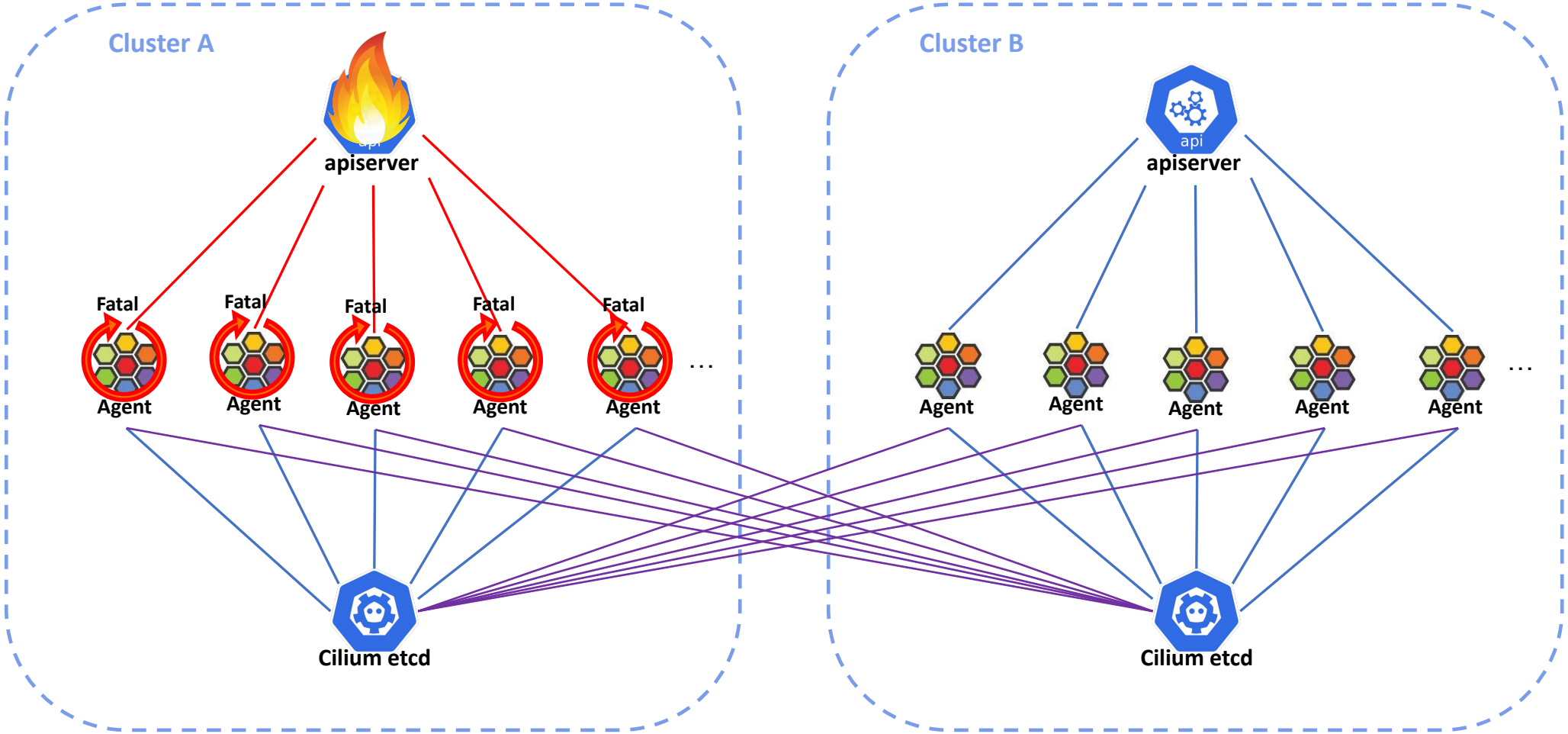




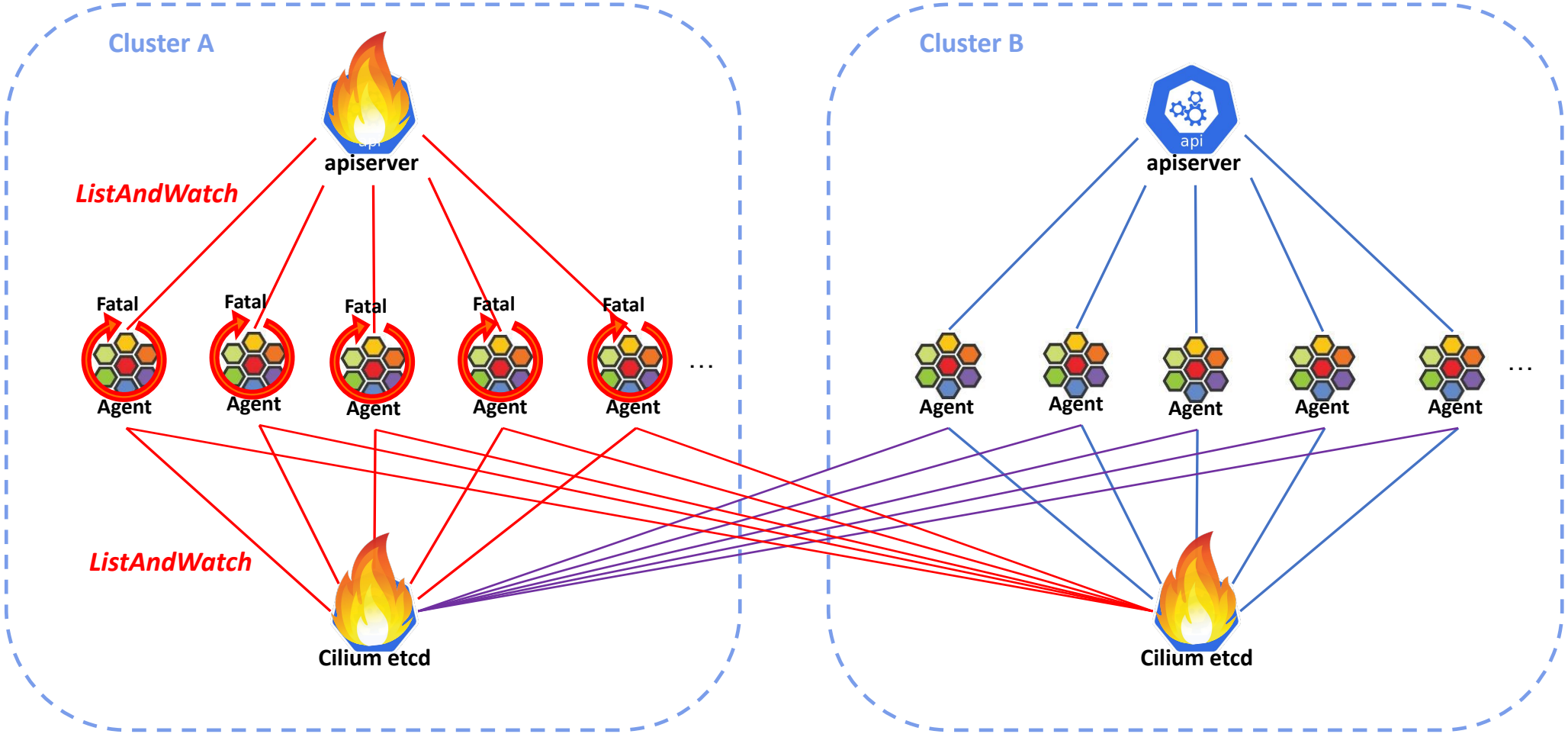
Fault propagation



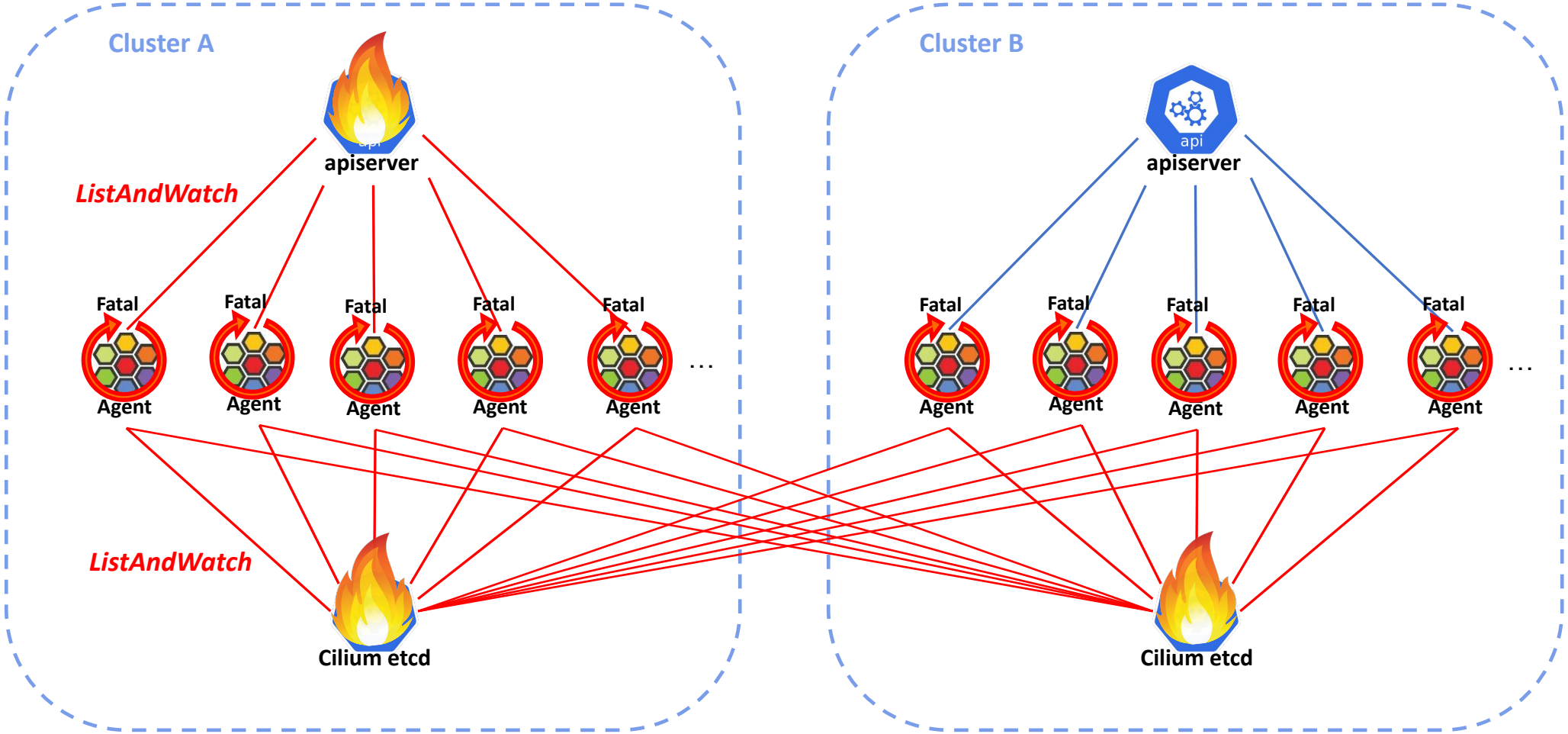
Fault propagation



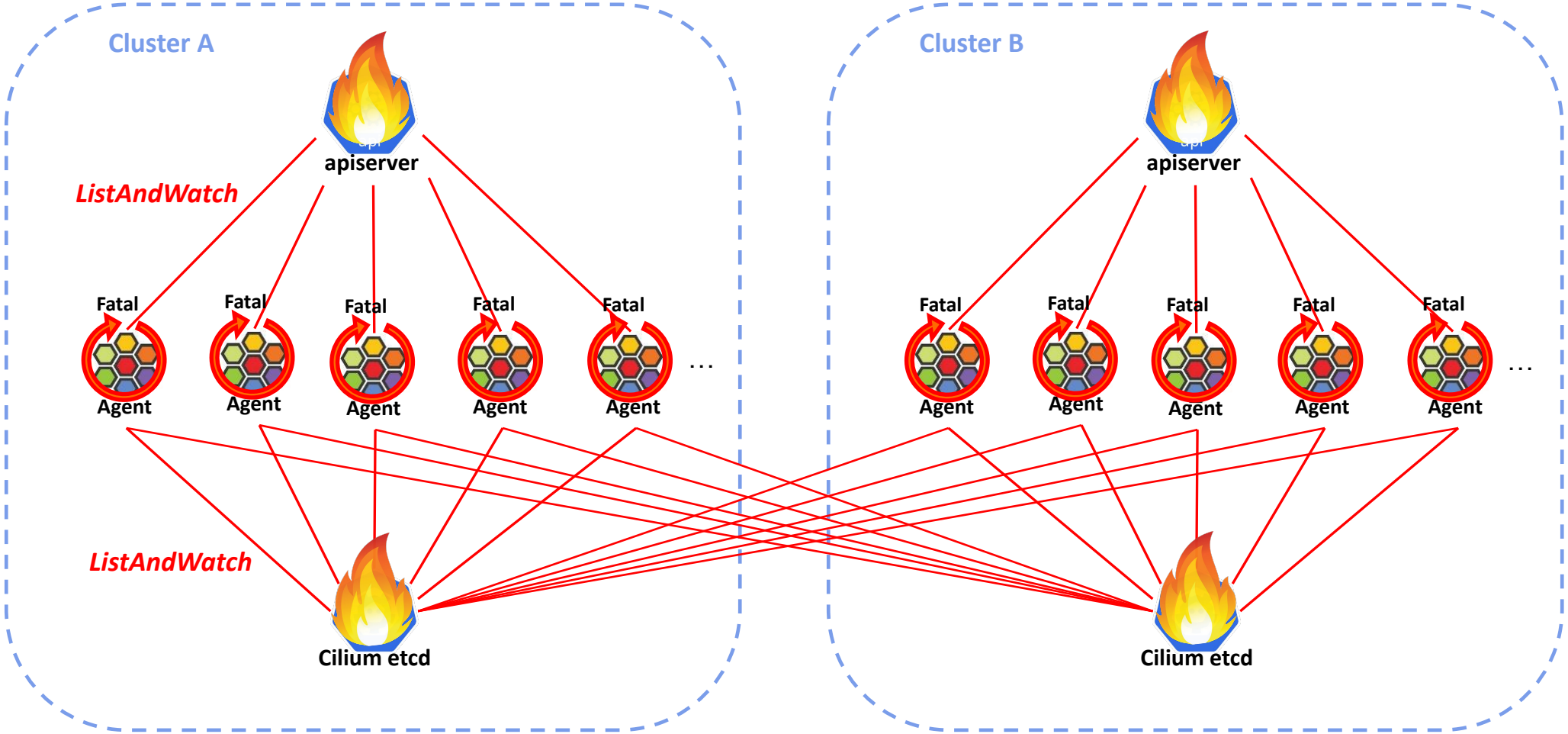
Fault propagation



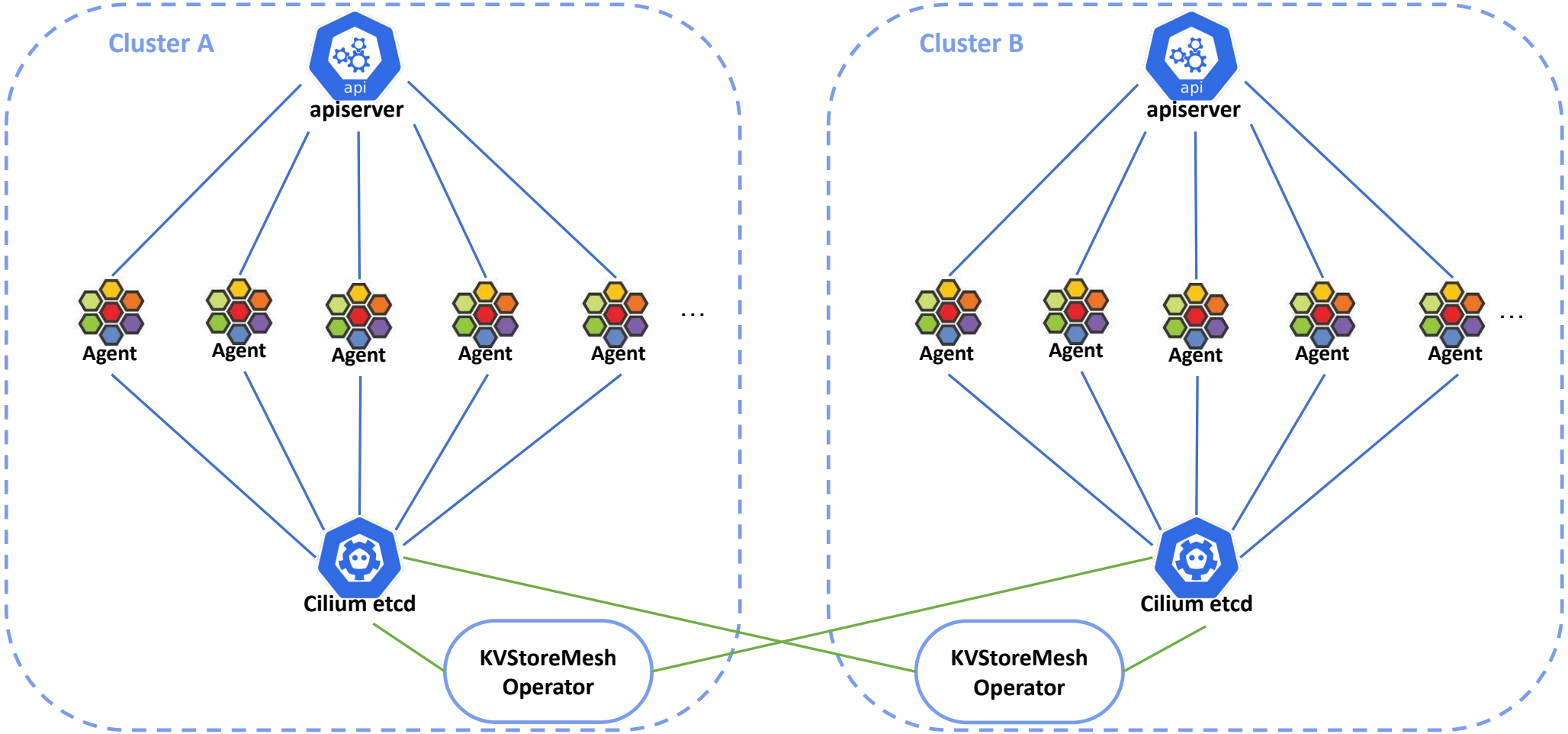
Fault propagation



Fault propagation



Solution: KVStoreMesh



KVStoreMesh has also been implemented in upstream version 1.14, currently in beta.

Network policy degradation

Last resort for policy degradation when control plane and agent are down

```
# Add an allow-any ingress rule to an endpoint
$ bpftool map update pinned ${POLICY_MAP_NAME} \
key hex 00 00 00 00 00 00 00 00 \
value hex 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 noexist
```

Key takeaways

- Prefer Kernel patch version > 100
- Pay attention to cloud API quotas
- For big subnets, pay attention to ARP table capacity
  net.ipv4.neigh.default.gc_thresh{1,2,3}
- Control retry behaviors
- Incident action plan

Thank you!