ISOVALENT

Unlocking Next-Gen Networking and Security with Cilium Service Mesh

Cloud Native and Kubernetes Meetup

Speaker: **Raymond de Jong**

@dejongraymond

- Open Source Projects

ISOVALENT

- Company behind Cilium
- Provides Cilium Enterprise

Agenda

- eBPF & Cilium Introduction
- Cilium Service Mesh
- New Features for 1.14
- Demo

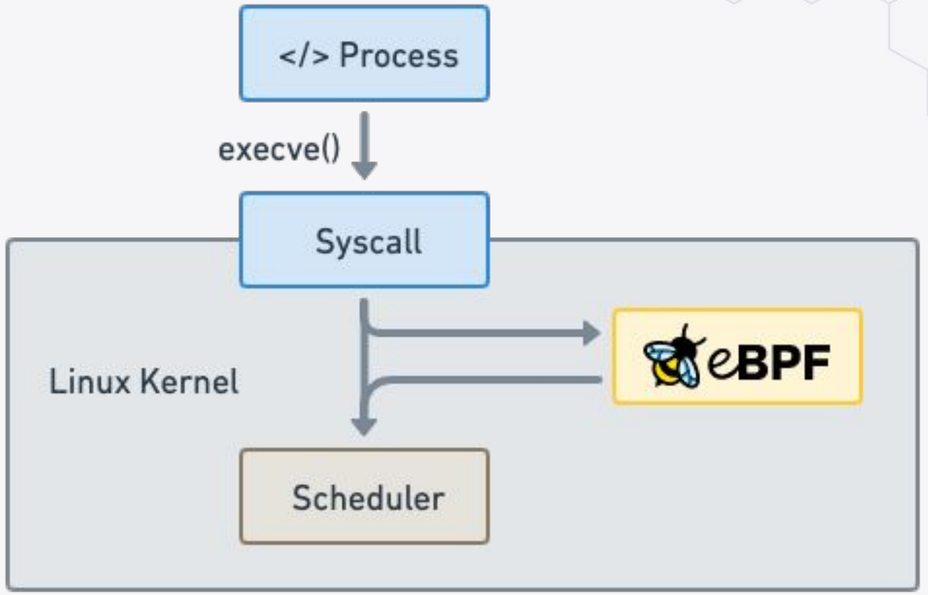
eBPF & Cilium Introduction


eBPF

eBPF

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

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

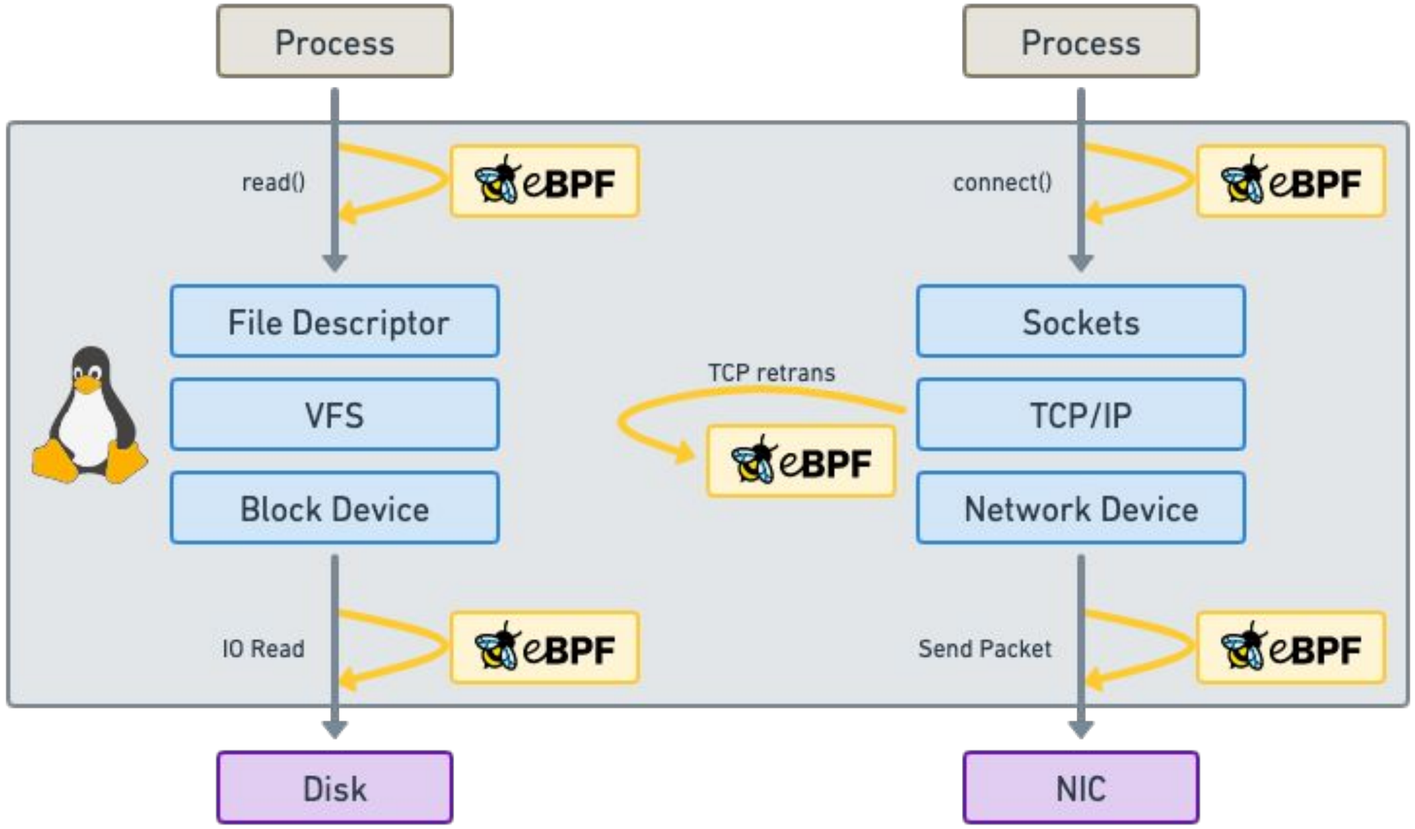



```c
int syscall__ret_execve(struct pt_regs *ctx)
{
        struct comm_event event = {
                .pid = bpf_get_current_pid_tgid() >> 32,
                .type = TYPE_RETURN,
        };

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

        return 0;
}
```

Run eBPF programs on events



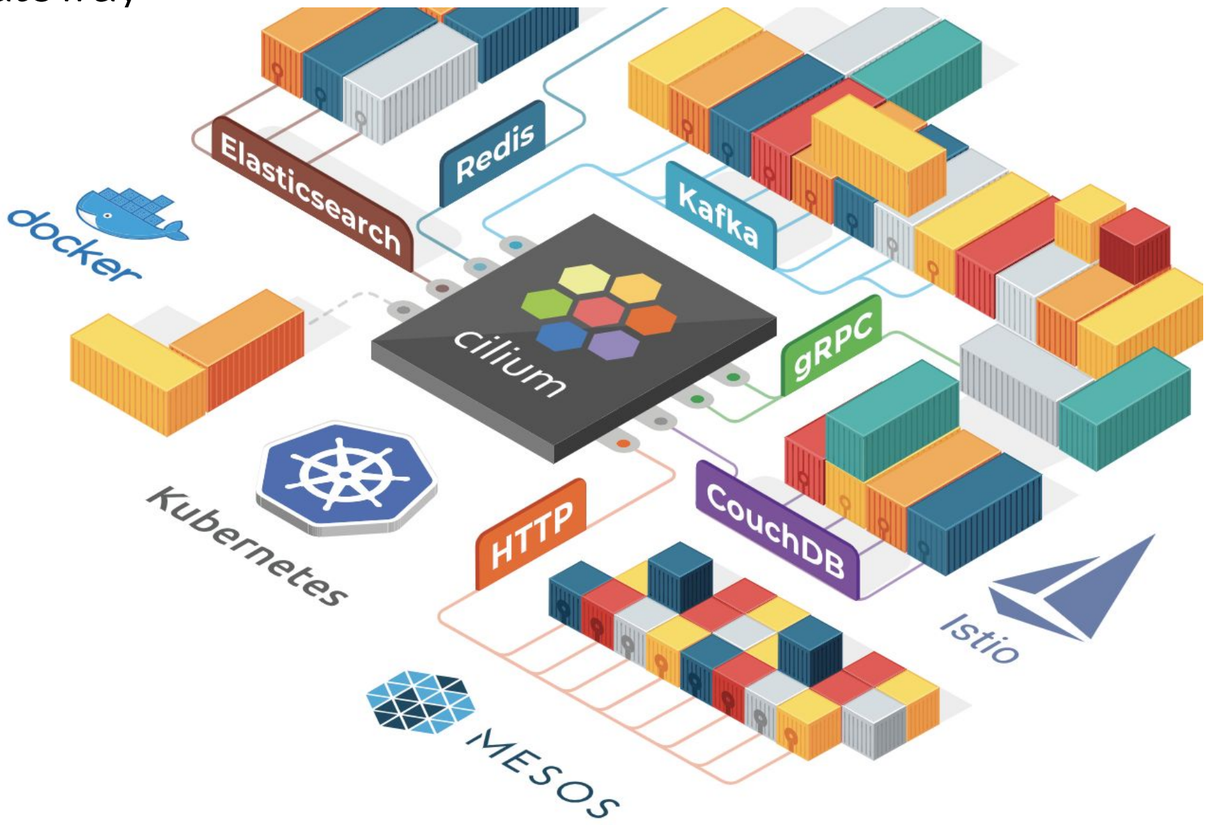
Attachment points

- Kernel functions (kprobes)
- Userspace functions (uprobe)
- System calls
- Tracepoints
- Sockets (data level)
- Network devices (packet level)
- Network device (DMA level) [XDP]
- ...

What is Cilium?

- **Networking & Load-Balancing**
  - CNI, Kubernetes Services, Multi-cluster, VM Gateway
- **Network Security**
  - Network Policy, Identity-based, Encryption
- **Observability**
  - Metrics, Flow Visibility, Service Dependency

At the foundation of Cilium is the new Linux kernel technology eBPF, which enables the dynamic insertion of powerful security, visibility, and networking control logic within Linux itself. Besides providing traditional network level security, the flexibility of BPF enables security on API and process level to secure communication within a container or pod.
Read More



cilium Service Mesh

Service Mesh

Ingress | Authentication | Traffic Management | spiffe | Gateway API

cilium hubble

Observability

Metrics | Tracing | Service Map | Logs | SIEM | fluentd | Grafana | OpenTelemetry

cilium CNI

Networking

Network Policy: DNS | L3/L4 | L7

Encryption: IPsec | Wireguard

Load-Balancing: K8s | Maglev | DSR

Networking: Multi-Cluster | NAT46 | IPv4 | IPv6 | Cloud SDN | BGP | Overlay | SRv6 | Egress Gateway

Runtime Security

Tetragon

SIEM | fluentd | Grafana

Observability

Enforcement

Kubernetes | Container | VM | Metal

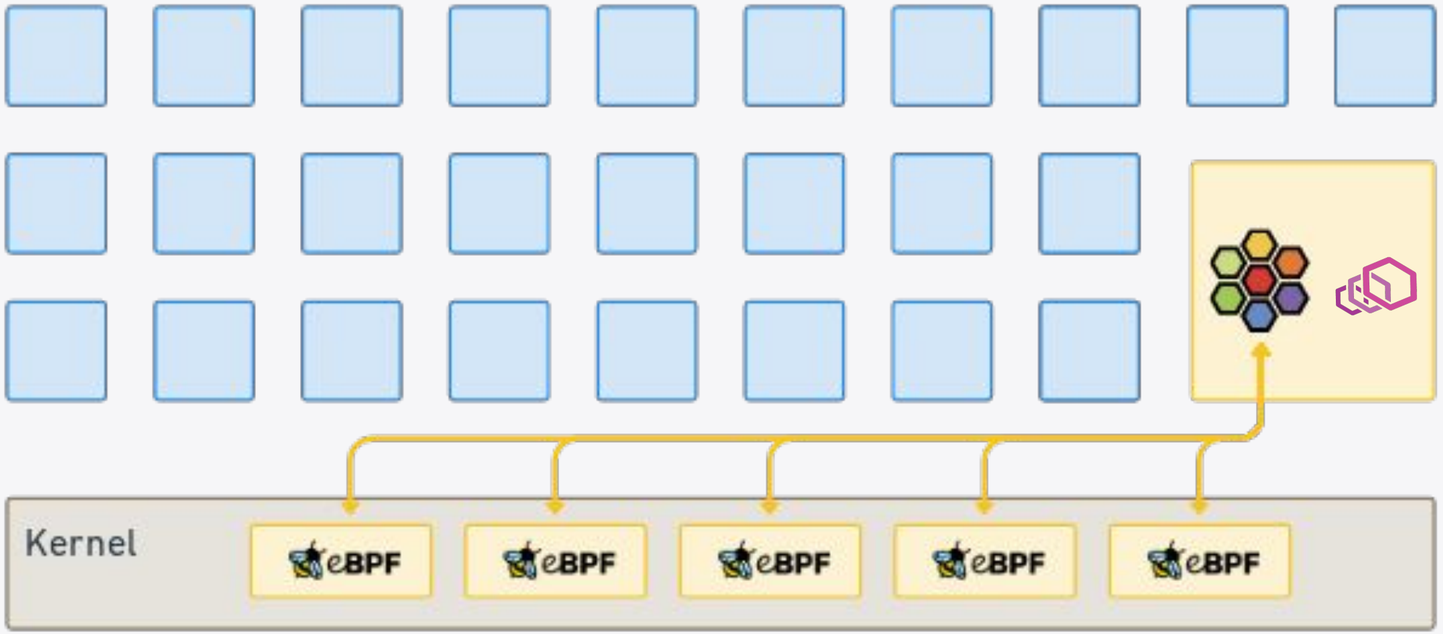
aws | Google Cloud | Azure | Alibaba Cloud | RED HAT OPENSHIFT | vmware

Cilium Service Mesh

What is different with Cilium Service Mesh?

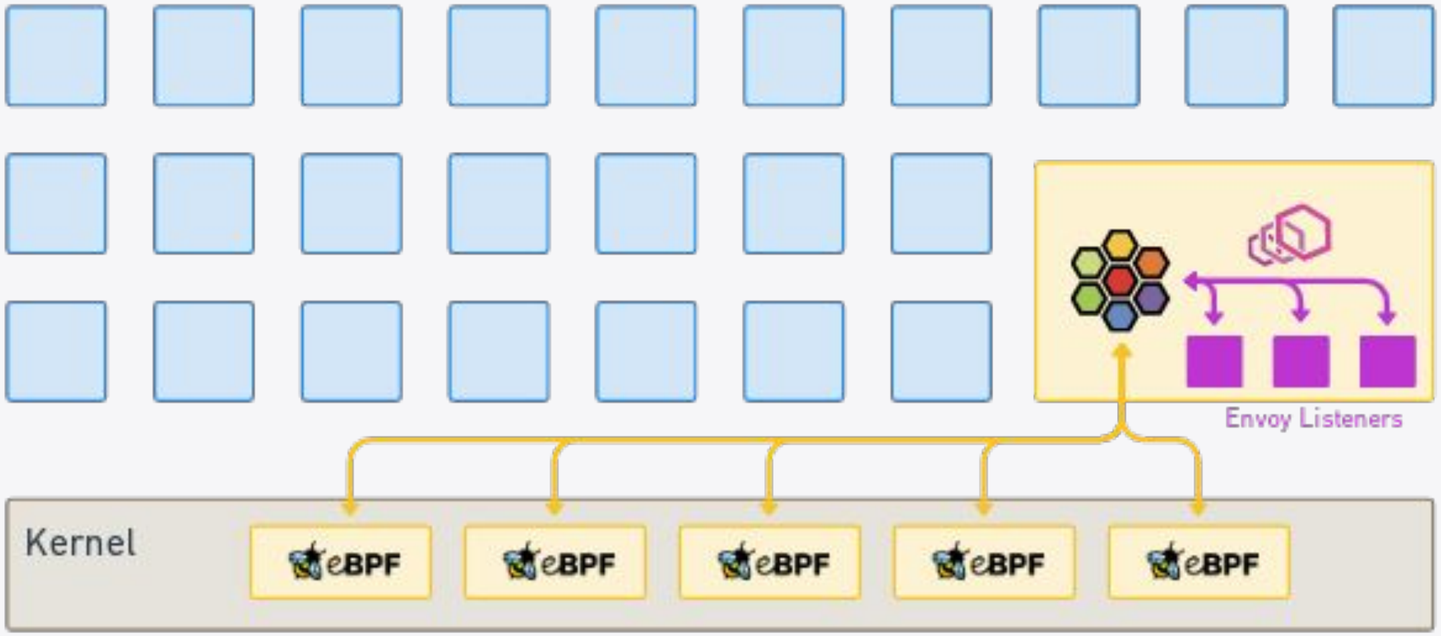
- Reduced operational complexity
- Reduced resource usage
- Better performance
- Avoid sidecar startup/shutdown race conditions

Cilium agent per node



- Dynamic eBPF programs
- Envoy for L7 policies & observability

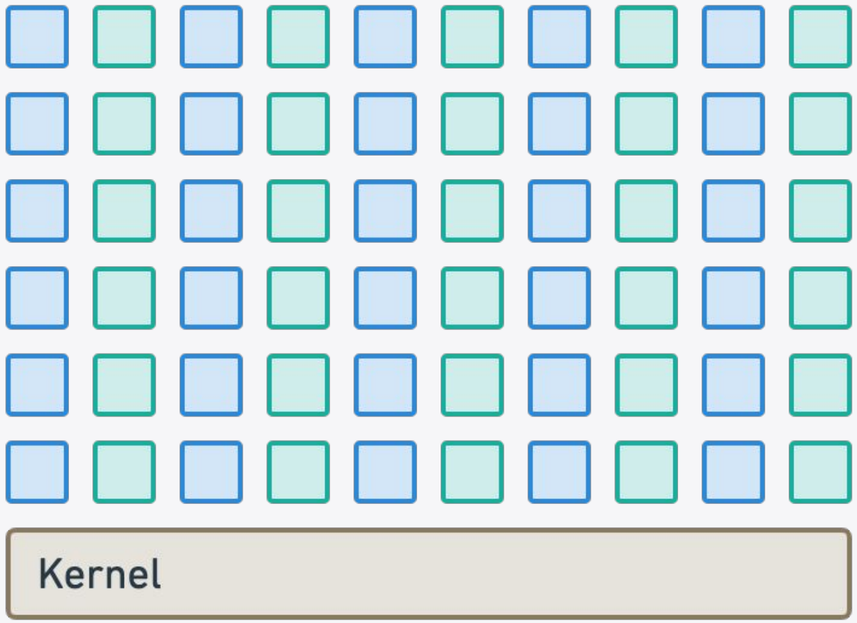
Cilium for sidecarless service mesh



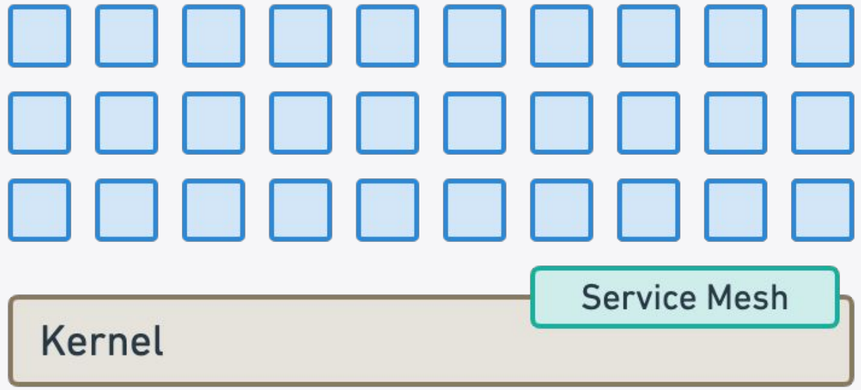
- Dynamic eBPF programs
- Envoy for L7 policies & observability and **traffic management rules** etc

Reduce resource usage - sidecar vs proxy per node

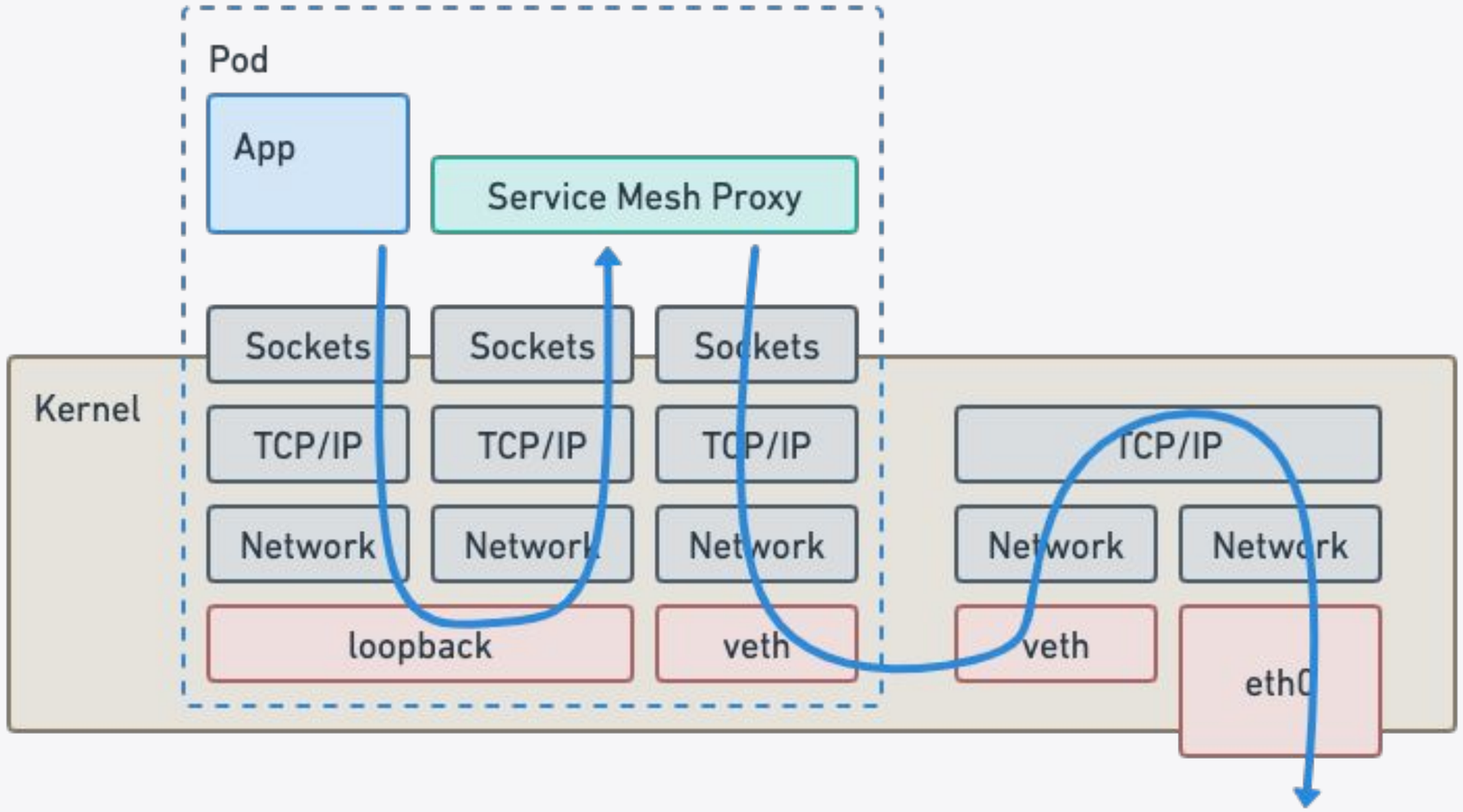
Total number of proxies required



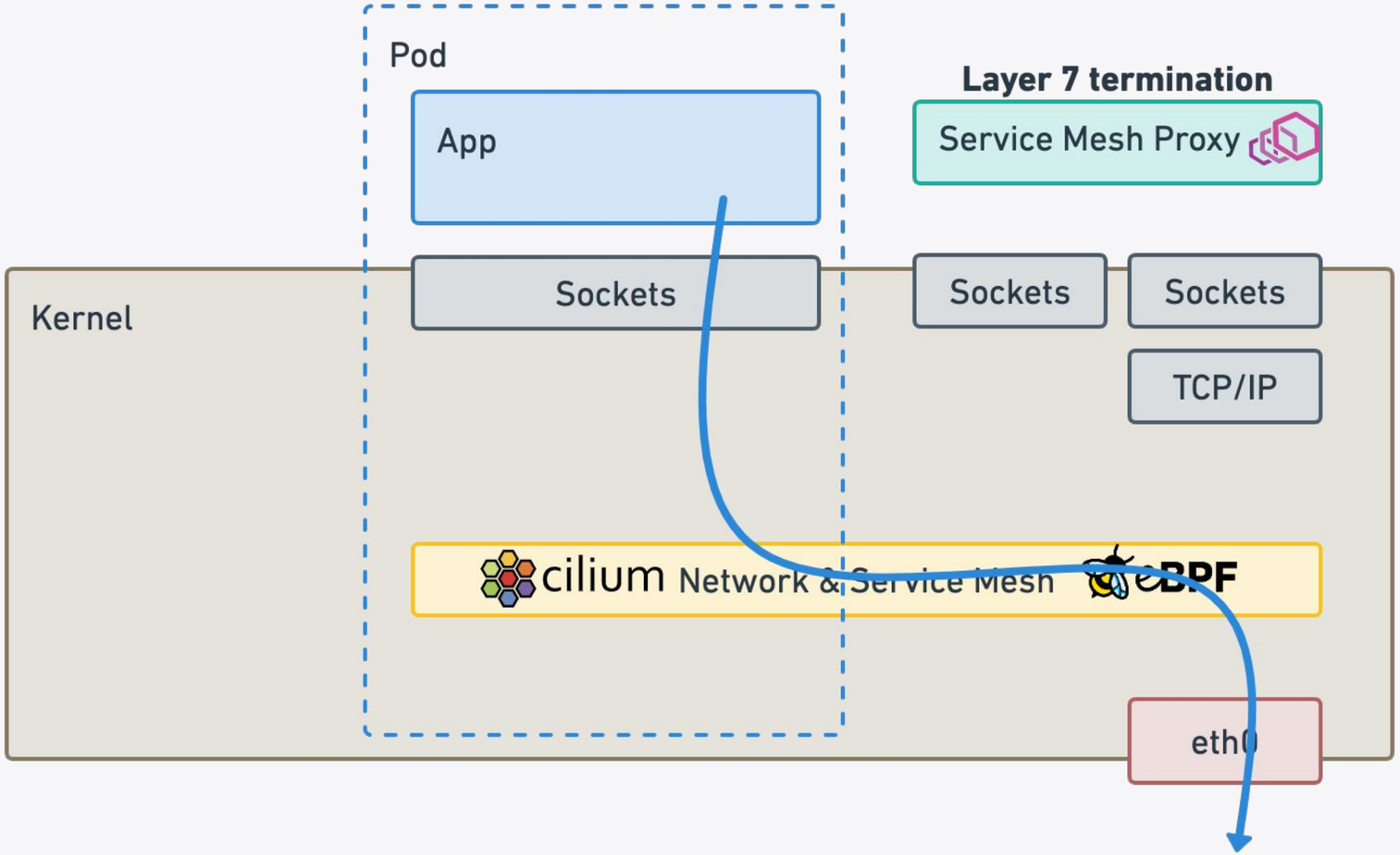
30 pods/node ⇒ 30 proxies/node



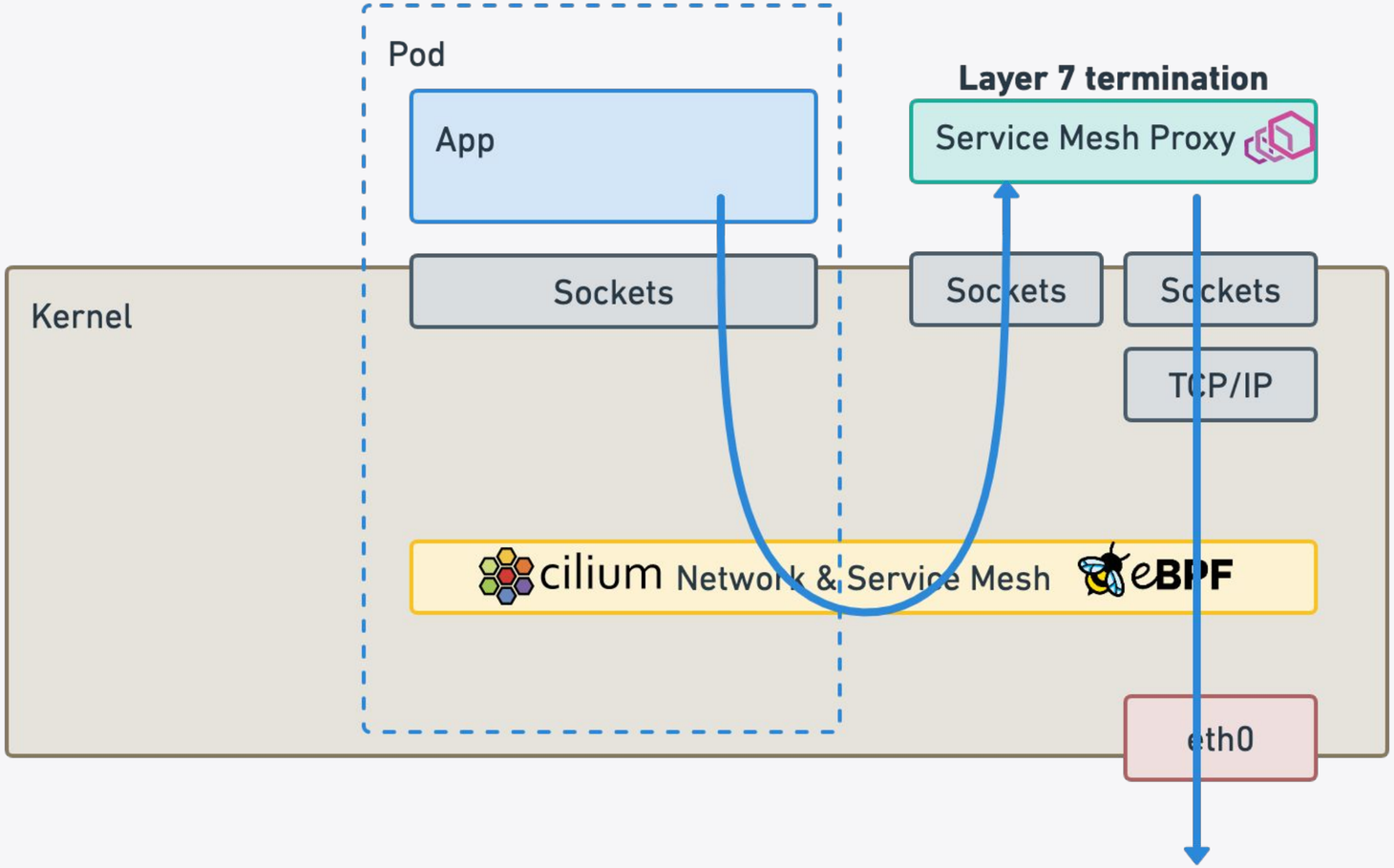
Cost of sidecar injection



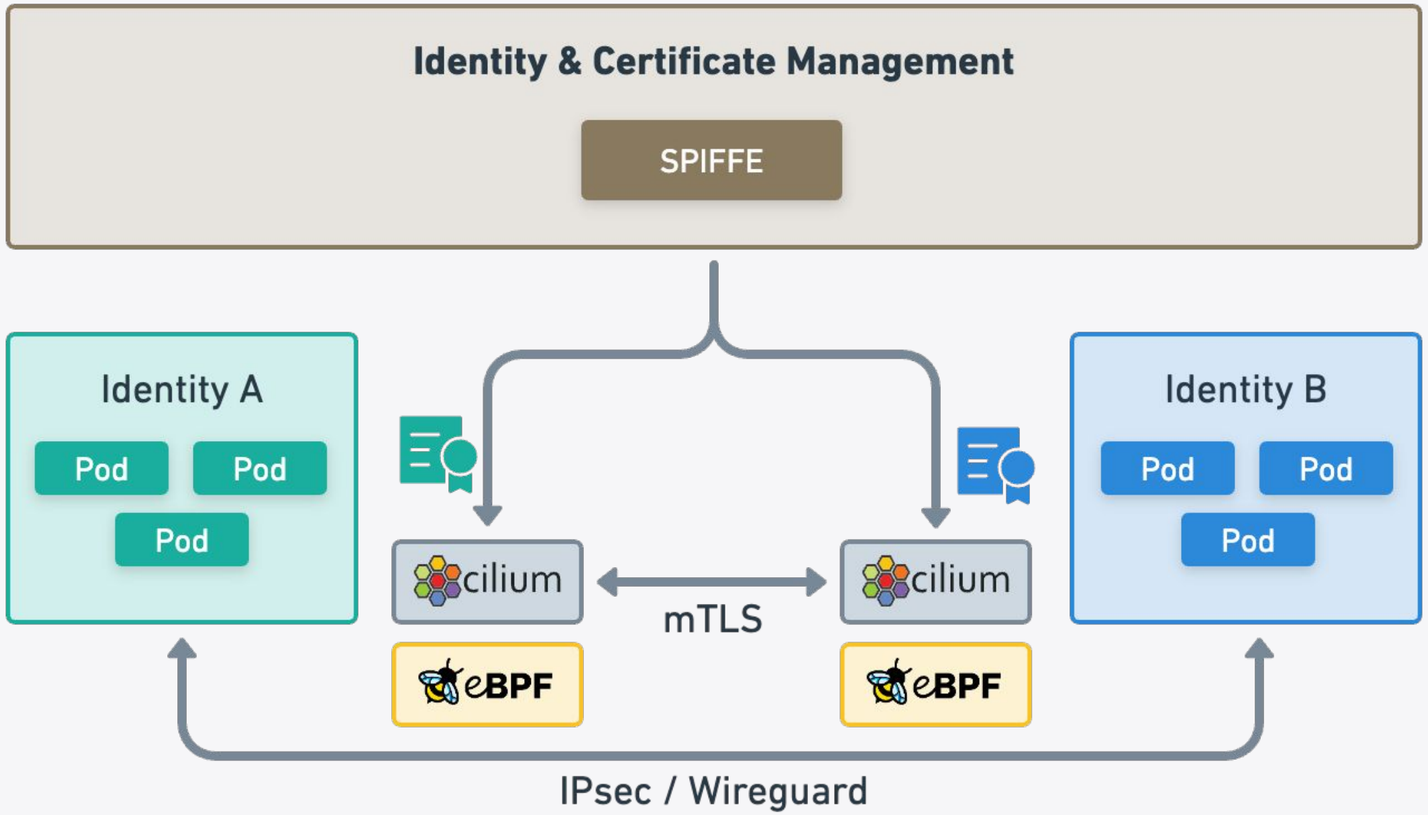
eBPF powered network path for L3/L4 traffic



Envoy for Layer 7 termination when needed



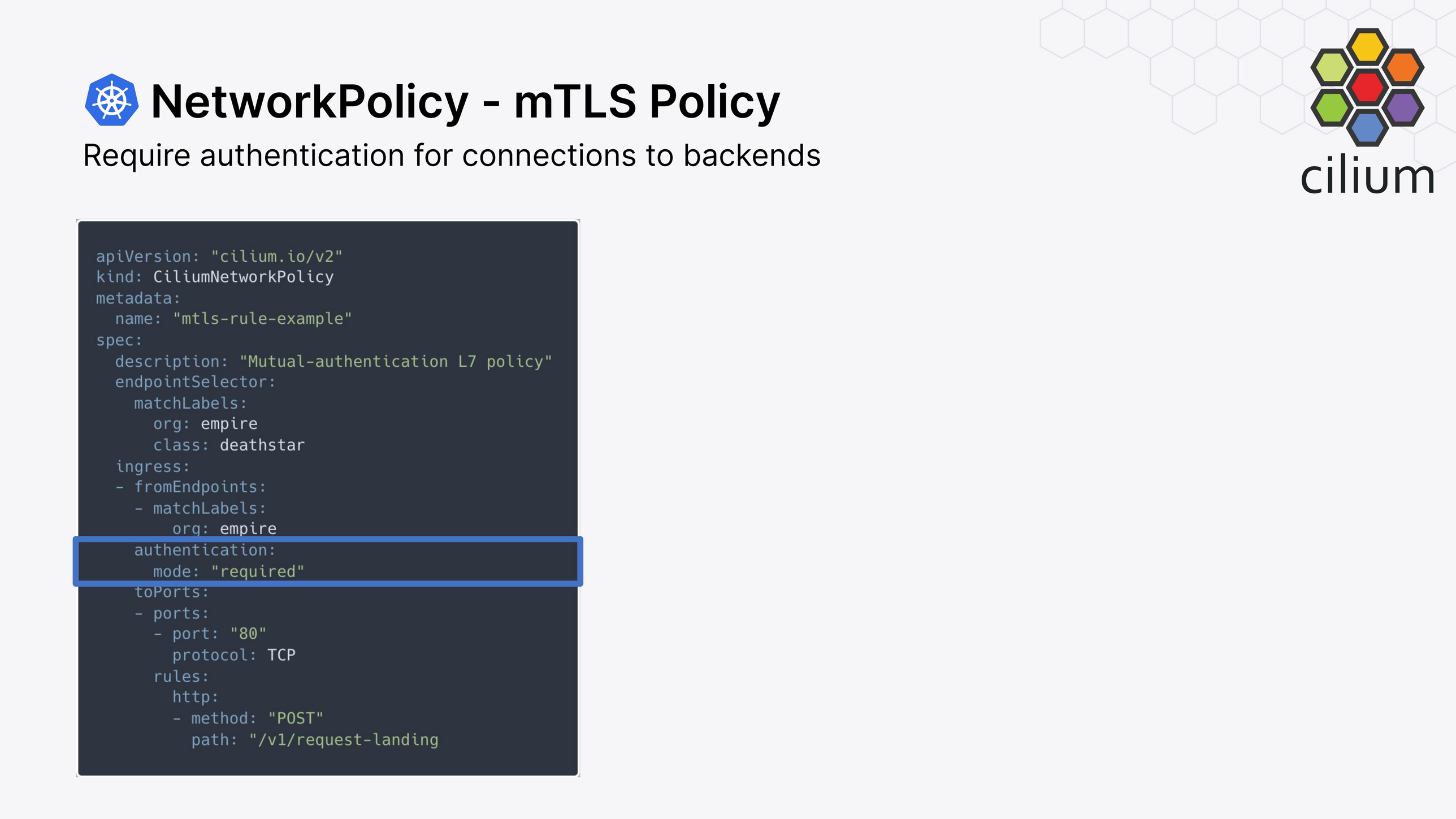
Mutual Authentication



- Not limited to TCP only.
- Works for any protocol (UDP, SCTP, ...)
- Handshake split from the Datapath
- Keeps secrets out of L7 proxies

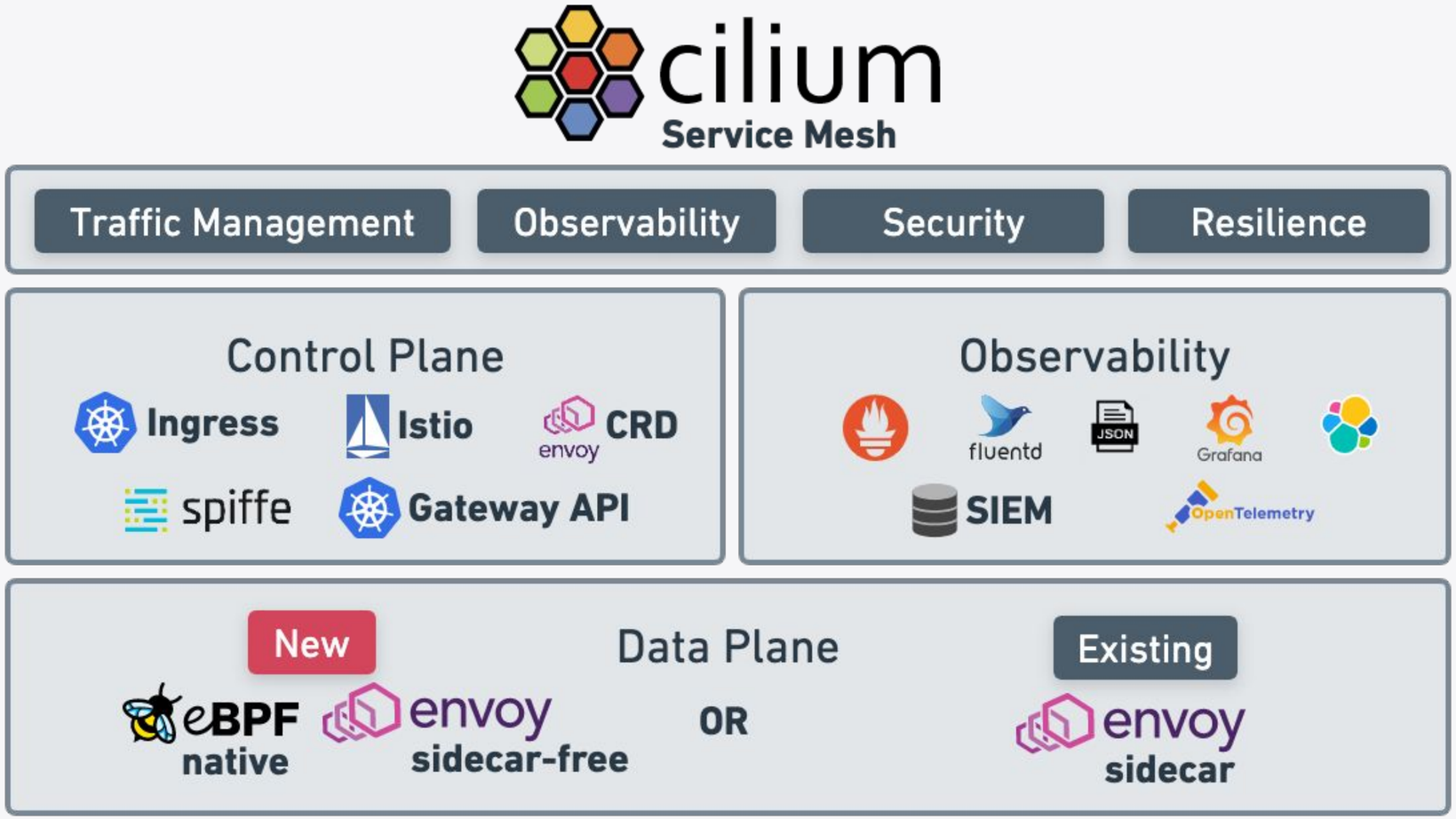
NetworkPolicy - mTLS Policy

Require authentication for connections to backends



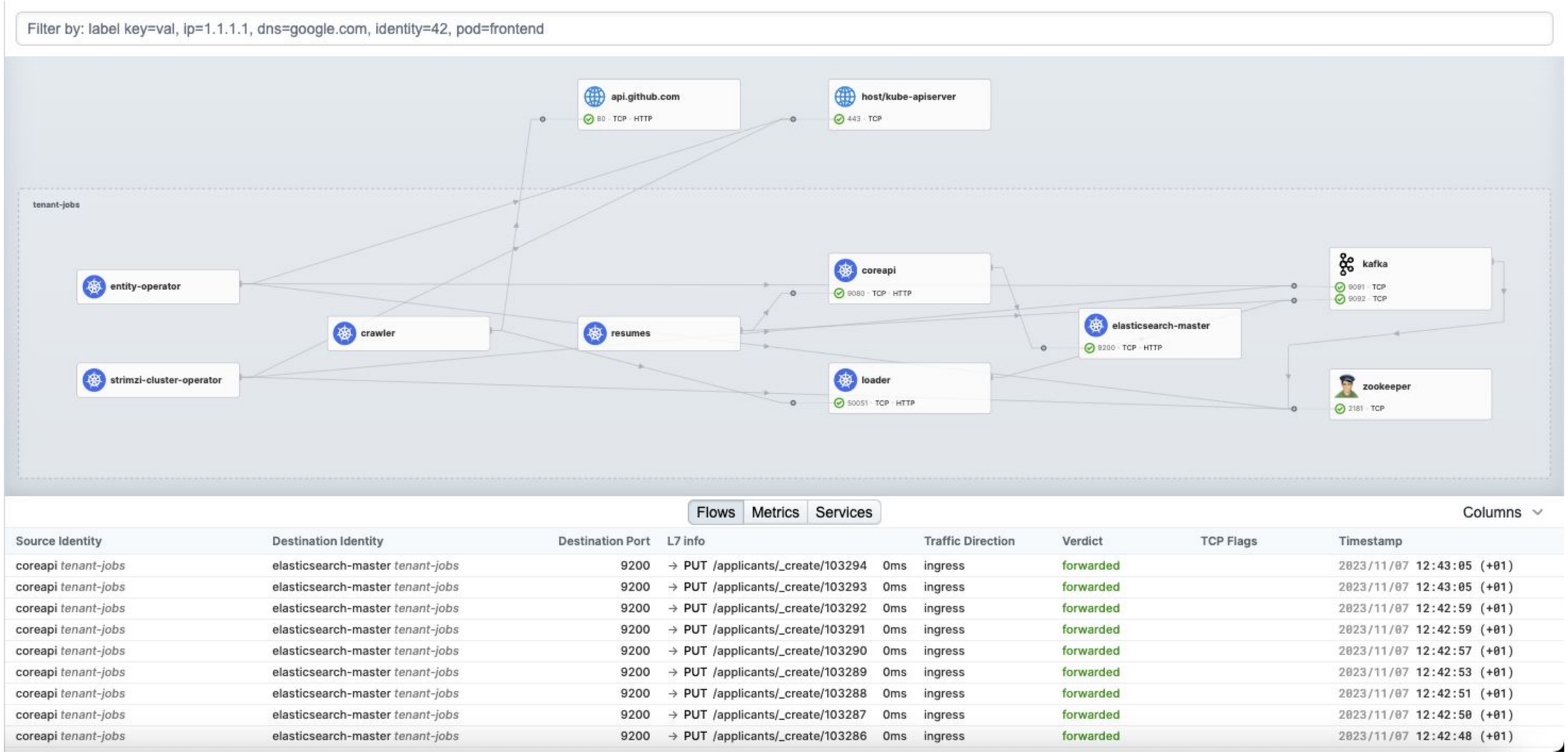
```yaml
apiVersion: "cilium.io/v2"
kind: CiliumNetworkPolicy
metadata:
  name: "mtls-rule-example"
spec:
  description: "Mutual-authentication L7 policy"
  endpointSelector:
    matchLabels:
      org: empire
      class: deathstar
  ingress:
  - fromEndpoints:
    - matchLabels:
        org: empire
    authentication:
      mode: "required"
    toPorts:
    - ports:
      - port: "80"
        protocol: TCP
      rules:
        http:
        - method: "POST"
          path: "/v1/request-landing
```

Cilium Service Mesh

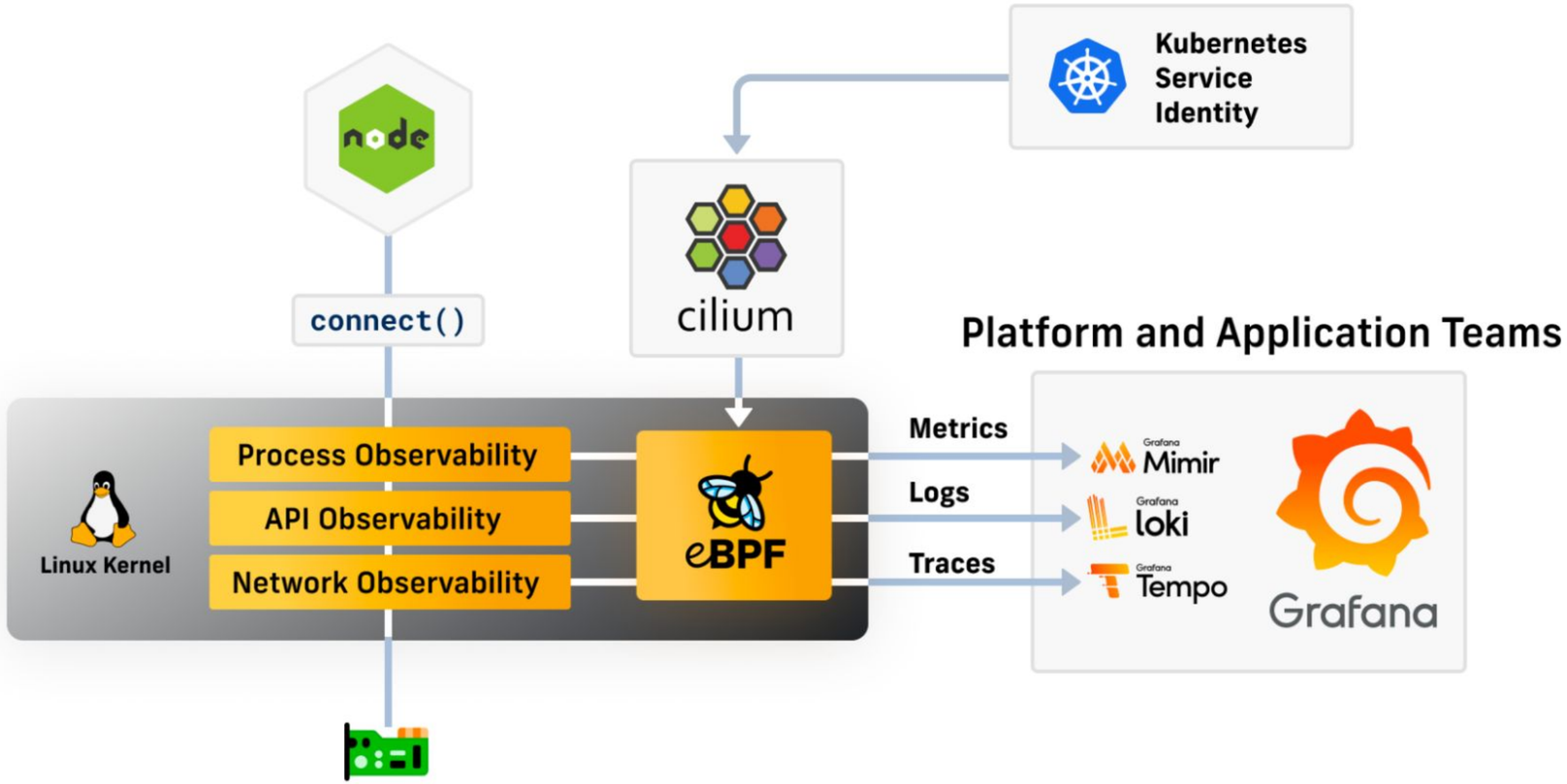


Visibility and Monitoring

Real-Time Flow Visibility and API-Level Monitoring



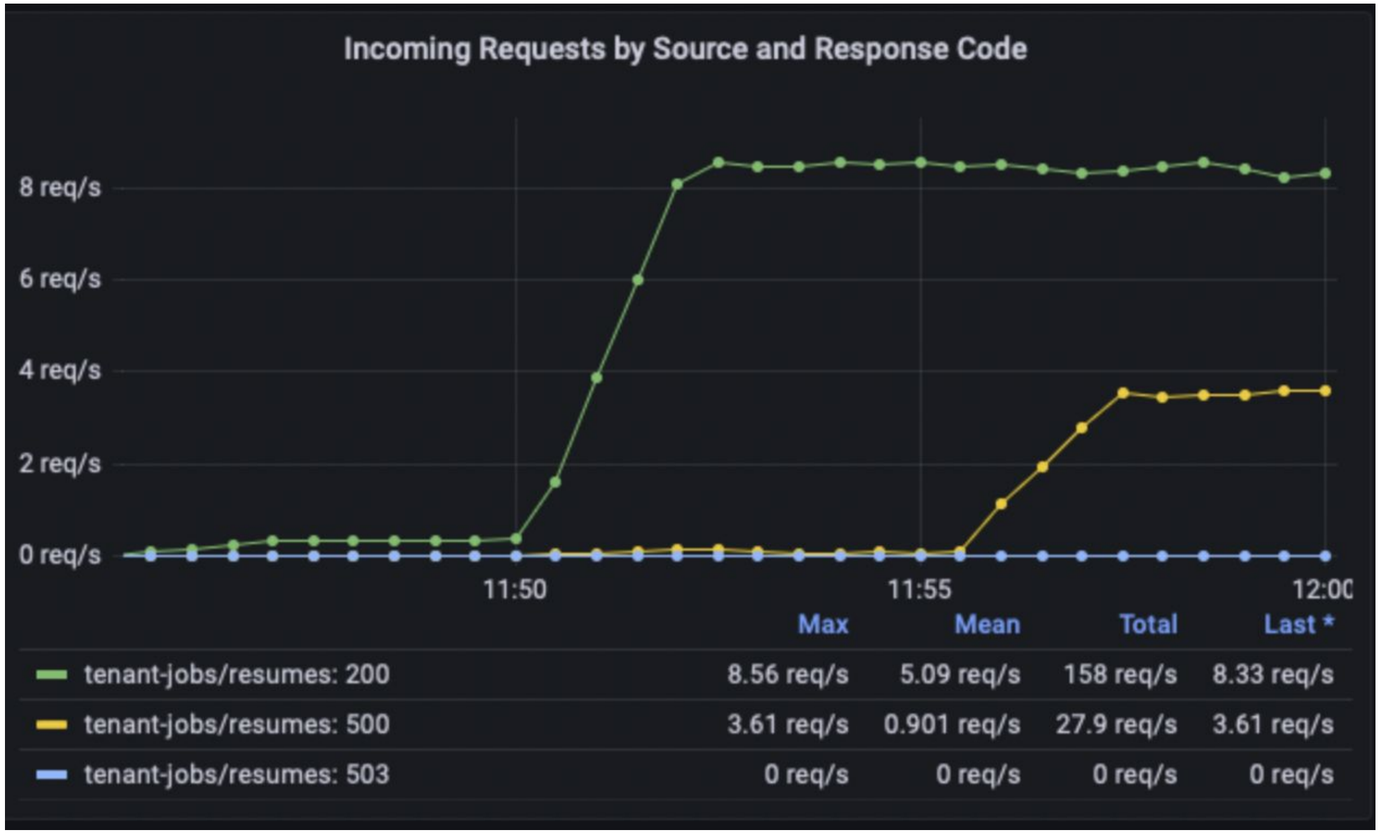
Service identity-aware network and API-layer observability with eBPF & Cilium



HTTP Golden Signals

eBPF powered metrics without Application changes or Sidecars required:

- HTTP Request Rate
- HTTP Request Latency
- HTTP Request Response Codes / Errors



Cilium 1.12 Release

- Production Ready Cilium Service Mesh
- Conformant Ingress Controller
- Using Kubernetes as Service Mesh Control Plane
  - Simple to use sidecar-free Service Mesh configured using Kubernetes Services and Ingress
- Prometheus metrics and OpenTelemetry
- CiliumEnvoyConfig and CiliumClusterEnvoyConfig CRD
- Extended Grafana dashboards for L7 visibility

Cilium 1.13 Release

- Gateway API
  - HTTP Routing
  - TLS Termination
  - HTTP Traffic Splitting / Weighting
  - HTTP Header Modification
- Shared LoadBalancer for Ingress Resources
- mTLS datapath
- L7 Load Balancing for Kubernetes Services with Annotations
- IPAM for LoadBalancer Services and BGP Services Advertisement

Cilium 1.14 Release

- Gateway API
  - Cross-Namespace Routing
  - HTTP Request & Response Header Modifiers
  - TLS Passthrough & TLS Route
- Support for Gateway API 0.7.0
- Mutual Authentication
- Wireguard Enhancements
- Envoy as a Daemonset

Features

Layer 7 Traffic Management Options

Ingress

Original L7 load-balancing standard in K8s

Simple

Supported since Cilium 1.12

Services

Use of K8s services with annotations

Simple

Supported since Cilium 1.13

Gateway API

Originally labelled Ingress v2. Richer in features.

Simple

Support for v0.7.0 and since Cilium 1.13

EnvoyConfig

Raw Envoy Config via CustomResource

Advanced Users & Integrations

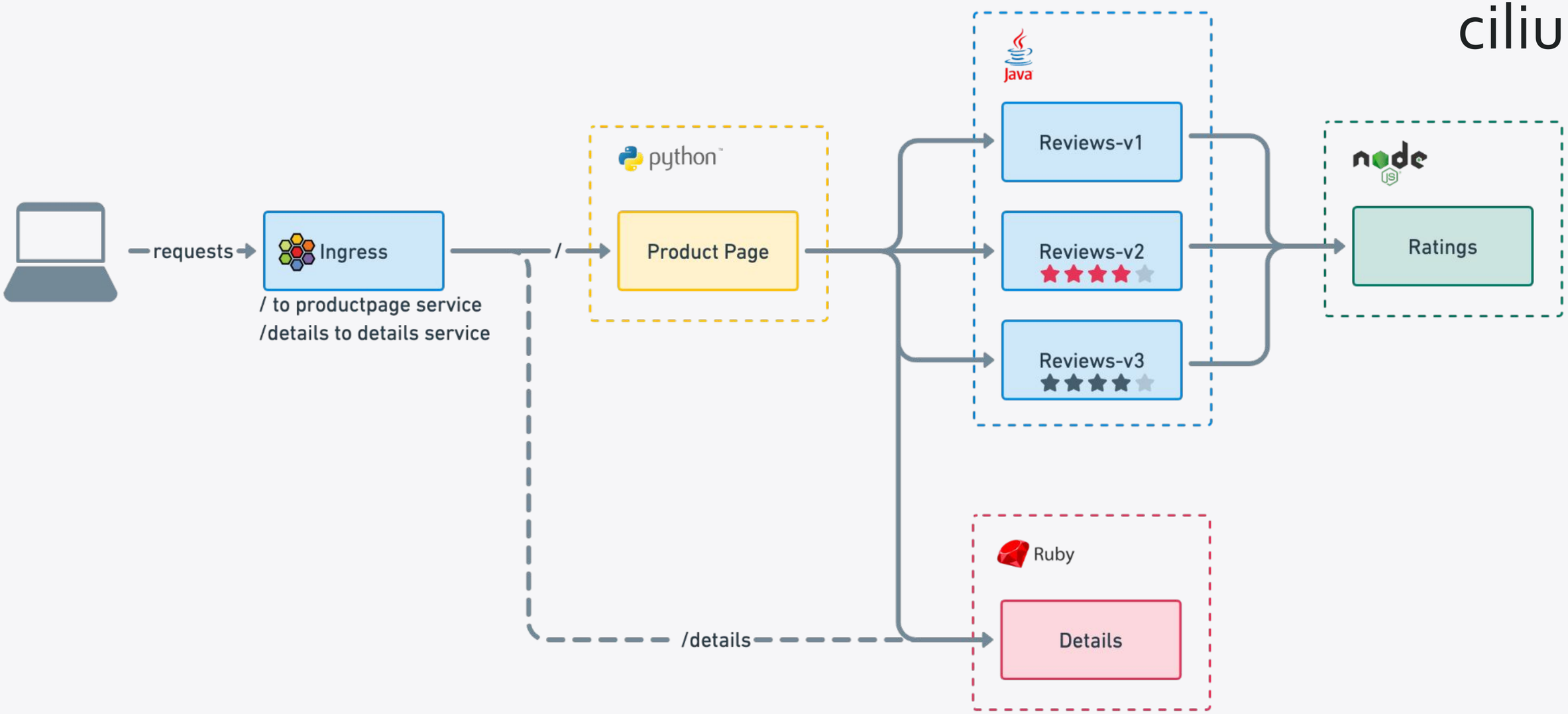
Supported since Cilium 1.12

Ingress

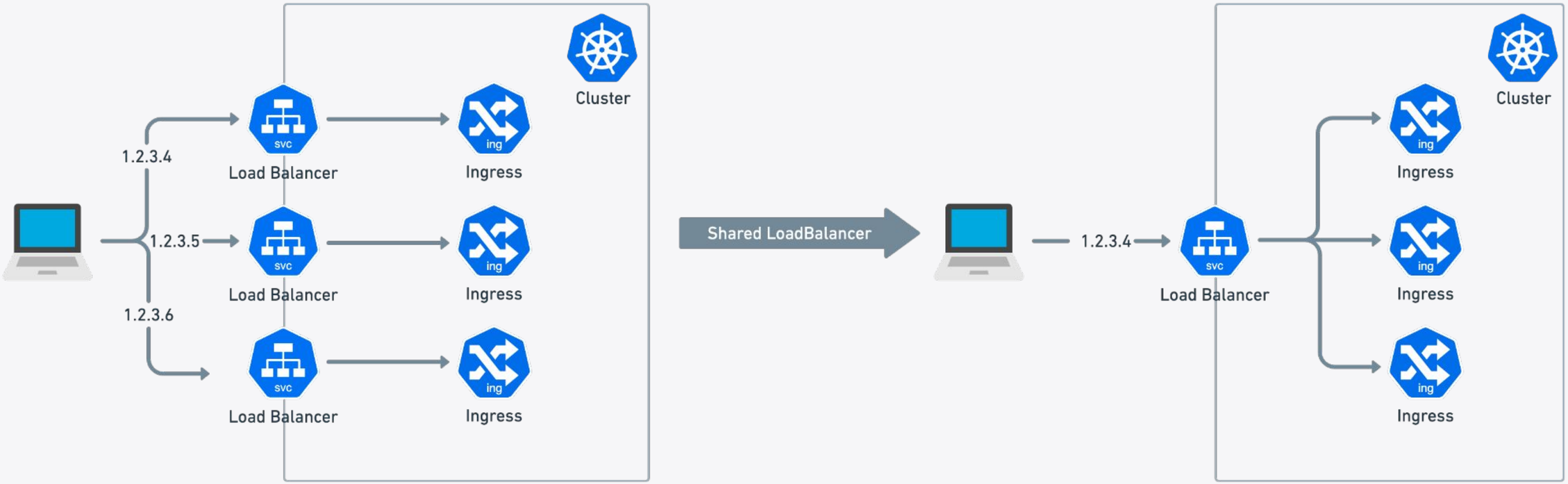
- Ingress can be used for path-based routing and TLS termination
- Cilium manages Ingress resources without external Ingress Controller
- Cilium Service Mesh Ingress Controller requires ability to create Service of Type LoadBalancer using either Cloud Provider integration or e.g. MetalLB
- Ingress CRD with `ingressClassName: cilium`

```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: basic-ingress
  namespace: default
spec:
  ingressClassName: cilium
  rules:
    - http:
        paths:
          - backend:
              service:
                name: details
                port:
                  number: 9080
            path: /details
            pathType: Prefix
          - backend:
              service:
                name: productpage
                port:
                  number: 9080
            path: /
            pathType: Prefix
```

Ingress HTTP Example

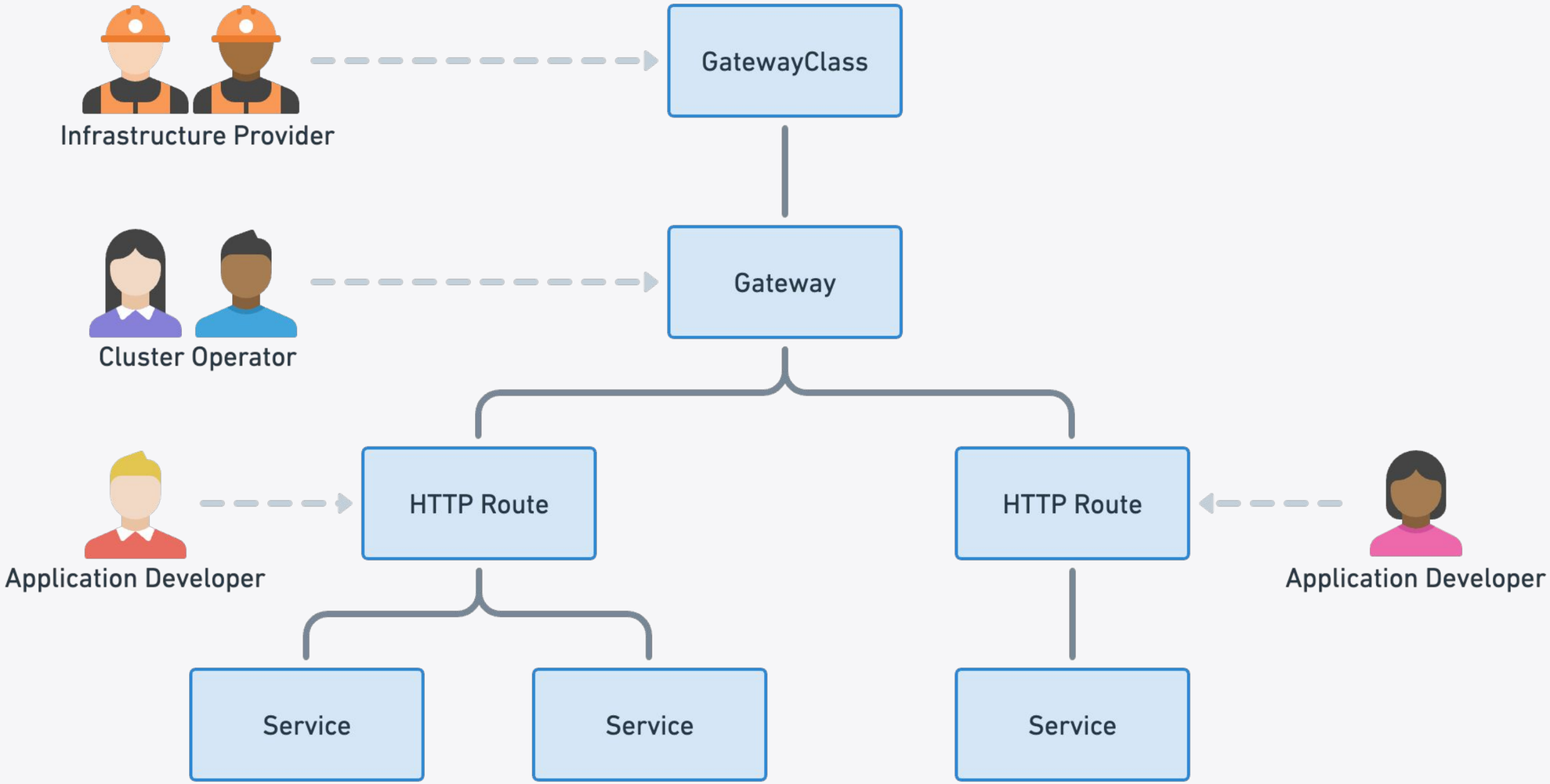


Shared LoadBalancer for Ingress Resources



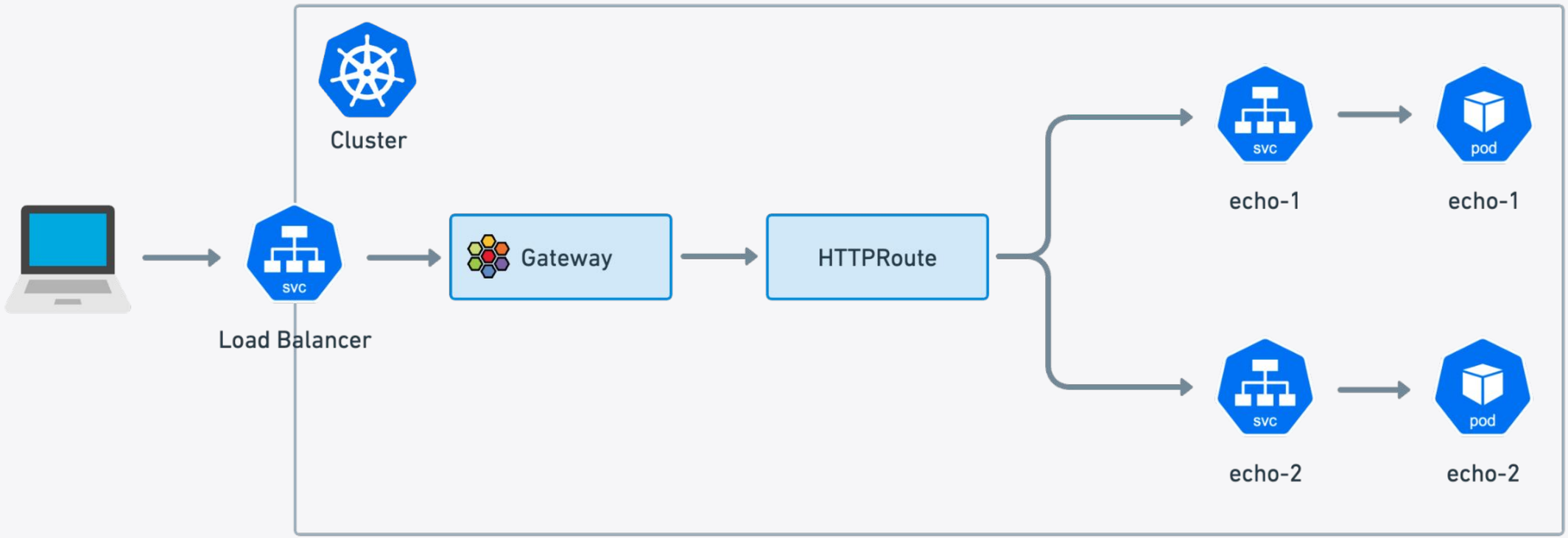
Gateway API

Introduction



Gateway API

Overview



Gateway API

Use of Gateway and HTTPRoute objects for path-based routing


```yaml
apiVersion: gateway.networking.k8s.io/v1beta1
kind: Gateway
metadata:
  name: my-gateway
spec:
  gatewayClassName: cilium
  listeners:
  - protocol: HTTP
    port: 80
    name: web-gw
    allowedRoutes:
      namespaces:
        from: Same
```

```yaml
apiVersion: gateway.networking.k8s.io/v1alpha2
kind: HTTPRoute
metadata:
  name: http-app-1
spec:
  parentRefs:
  - name: my-gateway
    namespace: default
  rules:
  - matches:
    - path:
        type: PathPrefix
        value: /details
    backendRefs:
    - name: details
      port: 9080
```

Gateway API

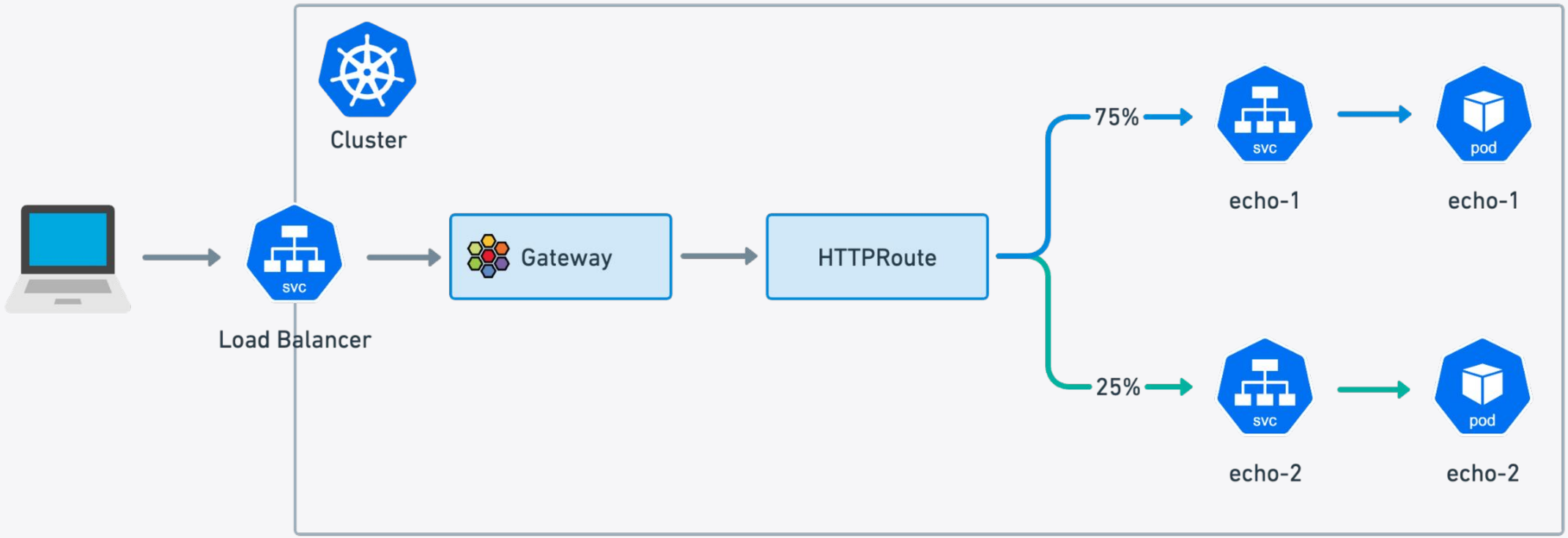
Use of Gateway and HTTPRoute for TLS Termination

```yaml
apiVersion: gateway.networking.k8s.io/v1beta1
kind: Gateway
metadata:
  name: tls-gateway
spec:
  gatewayClassName: cilium
  listeners:
  - name: https
    protocol: HTTPS
    port: 443
    hostname: "bookinfo.cilium.rocks"
    tls:
      certificateRefs:
      - kind: Secret
        name: demo-cert
```

```yaml
apiVersion: gateway.networking.k8s.io/v1beta1
kind: HTTPRoute
metadata:
  name: https-app-route
spec:
  parentRefs:
  - name: tls-gateway
  hostnames:
  - "bookinfo.cilium.rocks"
  rules:
  - matches:
    - path:
        type: PathPrefix
        value: /details
    backendRefs:
    - name: details
      port: 9080
```

Gateway API

Traffic Splitting with Weighted Routes



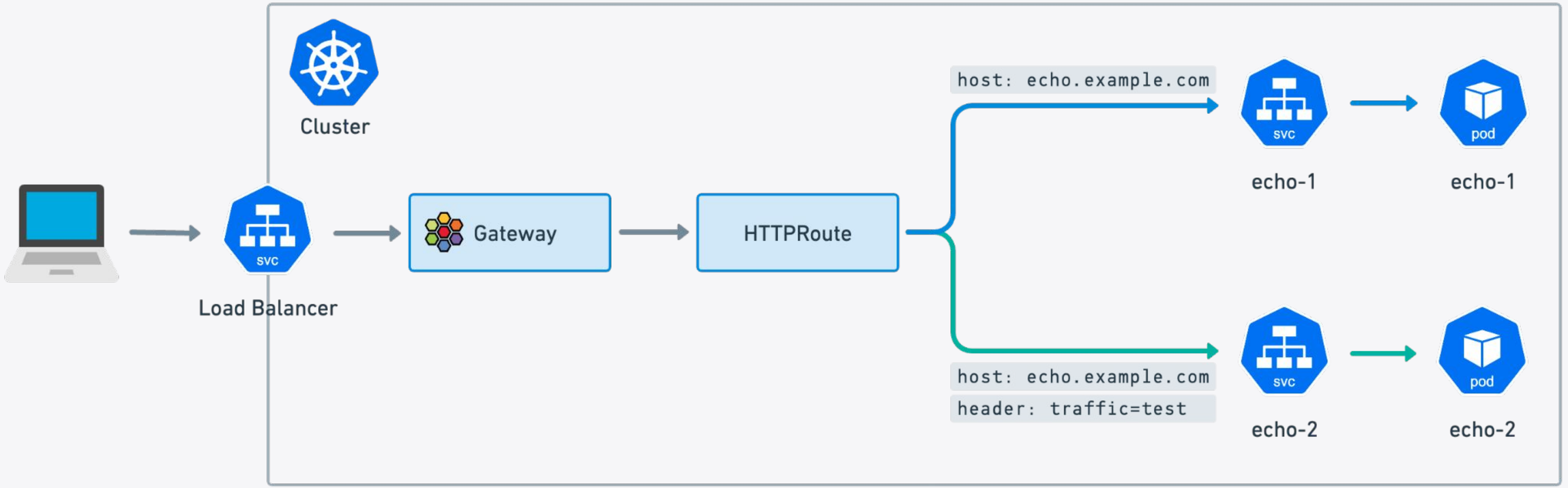
Gateway API

Traffic Splitting with Weighted Routes


```yaml
apiVersion: gateway.networking.k8s.io/v1alpha2
kind: HTTPRoute
metadata:
  name: example-weighted-route
spec:
  parentRefs:
  - name: my-gateway
  rules:
  - matches:
    - path:
        type: PathPrefix
        value: /echo
    backendRefs:
    - kind: Service
      name: echo-1
      port: 8080
      weight: 75
    - kind: Service
      name: echo-2
      port: 8090
      weight: 25
```

Gateway API

Canary Deployment Rollout



Gateway API

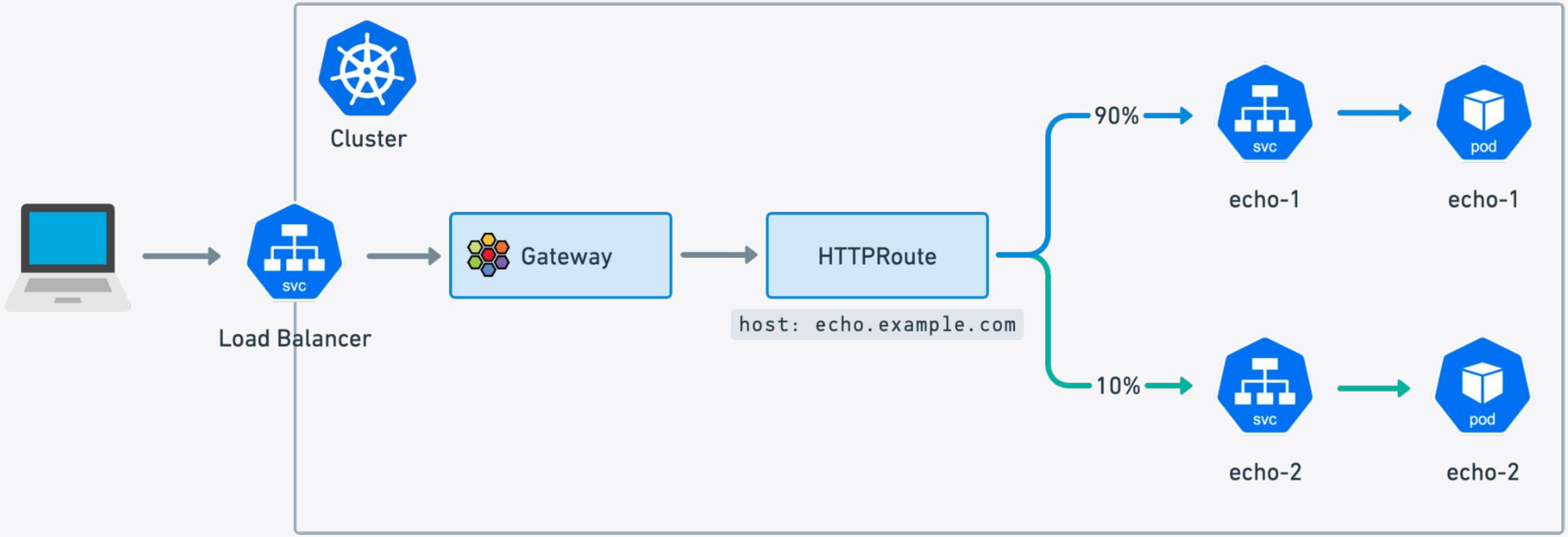
Canary Traffic Rollout


```yaml
apiVersion: gateway.networking.k8s.io/v1beta1
kind: HTTPRoute
metadata:
  name: echo-route
  labels:
    gateway: prod-web-gw
spec:
  hostnames:
  - echo.example.com
  rules:
  - backendRefs:
    - name: echo-1
      port: 8080
  - matches:
    - headers:
      - name: traffic
        value: test
    backendRefs:
    - name: echo-2
      port: 8080
```

Gateway API

Blue-Green Traffic Rollout

cilium



Gateway API

Blue-Green Traffic Rollout


```yaml
apiVersion: gateway.networking.k8s.io/v1beta1
kind: HTTPRoute
metadata:
  name: echo-route
  labels:
    gateway: prod-web-gw
spec:
  hostnames:
  - echo.example.com
  rules:
  - backendRefs:
    - name: echo-1
      port: 8080
      weight: 90
    - name: echo-2
      port: 8080
      weight: 10
```

Gateway API

HTTP Request Header Modifier - Add, Edit or Remove Headers

cilium

```yaml
apiVersion: gateway.networking.k8s.io/v1beta1
kind: HTTPRoute
metadata:
  name: header-http-echo
spec:
  parentRefs:
    - name: my-gateway
  rules:
    - matches:
        - path:
            type: PathPrefix
            value: /add-a-request-header
      filters:
        - type: RequestHeaderModifier
          requestHeaderModifier:
            add:
              - name: my-header-name
                value: my-header-value
      backendRefs:
        - name: echo
          port: 8080
```

Gateway API

HTTP Response Header Modifier

```yaml
apiVersion: gateway.networking.k8s.io/v1beta1
kind: HTTPRoute
metadata:
  name: header-http-echo
spec:
  parentRefs:
    - name: my-gateway
  rules:
    - matches:
        - path:
            type: PathPrefix
            value: /add-a-request-header
      filters:
      - type: ResponseHeaderModifier
        responseHeaderModifier:
          add:
          - name: X-Header-Add-1
            value: header-add-1
          - name: X-Header-Add-2
            value: header-add-2
      backendRefs:
        - name: echo
          port: 8080
```

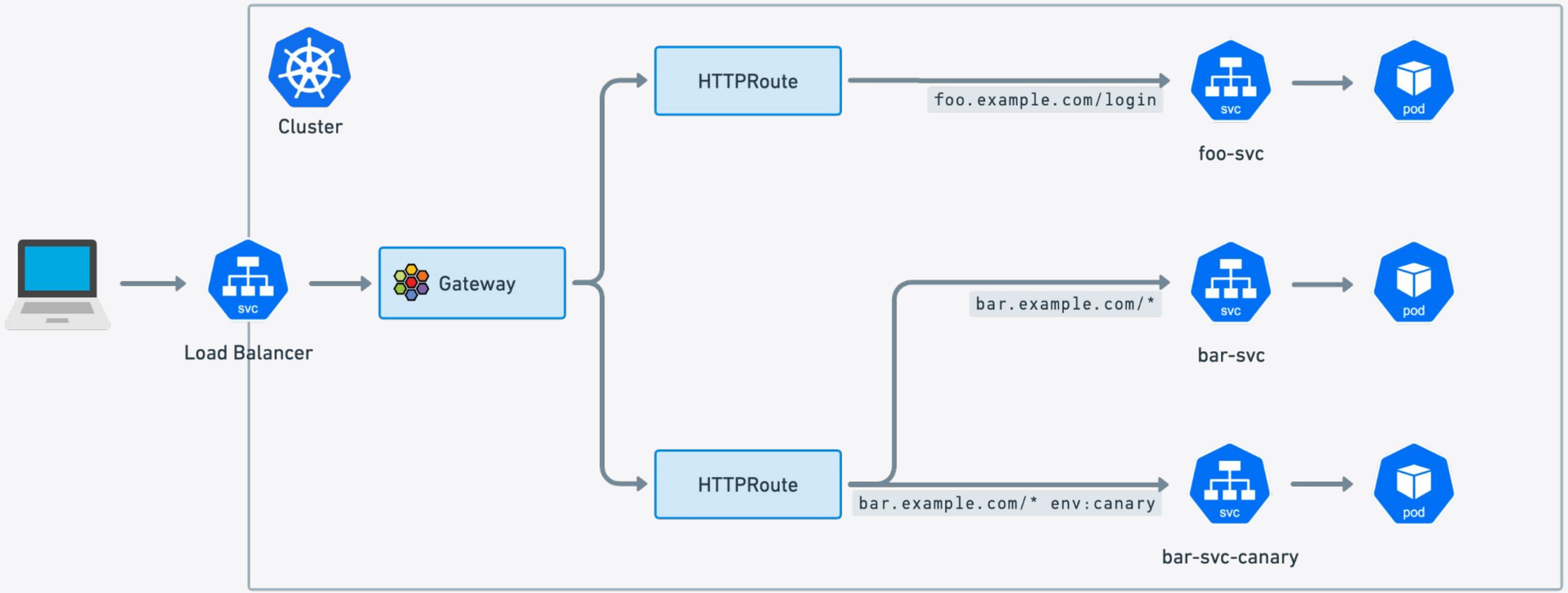
cilium

Demo

Gateway API using Cilium Service Mesh

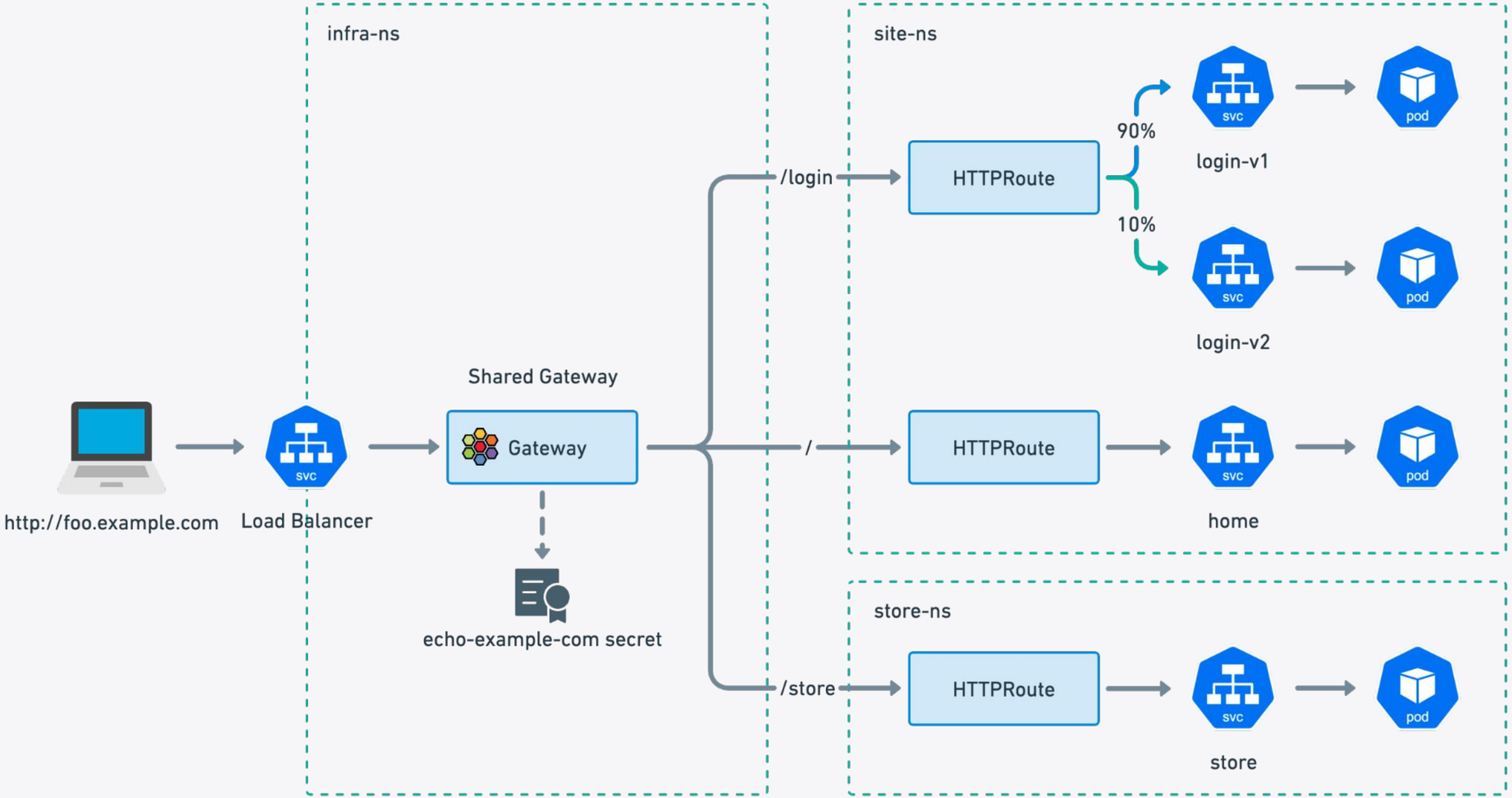
Gateway API

HTTP Routing



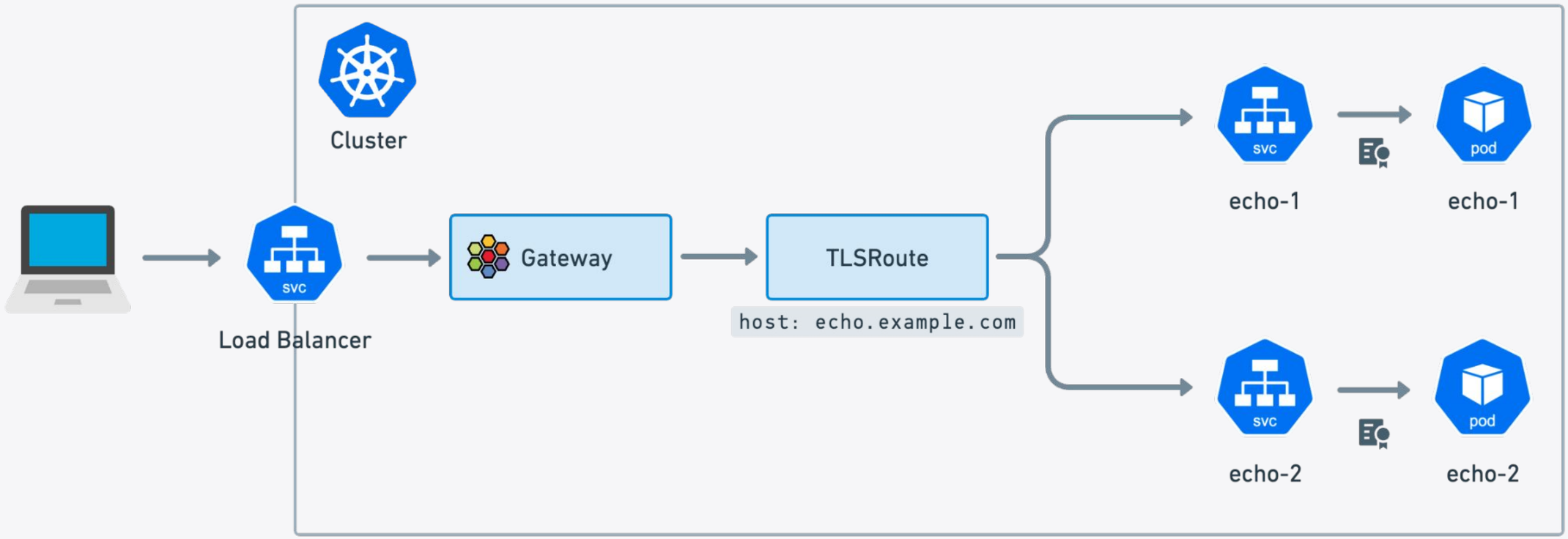
Gateway API

Cross-Namespace Routing



Gateway API

TLS Passthrough & TLS Route



ISOVALENT

Creators of cilium and eBPF

Learn more:

Webinar

Tetragon 1.0 has Landed: What's New and Exciting in Kubernetes Security?

November 30 at 18:00 CET

ISOVALENT

Creators of cilium and eBPF

Cilium 1.14 release!

Webinar: What's new in Cilium 1.14 with Thomas Graf

November 30 at 18:00 CET

Virtual workshop: Cilium 1.14 labs

December 14 at 10:00 CET

Learn more:

Learn more!

ISOVALENT

For the Enterprise

Hardened, enterprise-grade eBPF-powered networking, observability, and security.

isovalent.com/product
isovalent.com/labs

OSS Community

eBPF-based Networking, Observability, Security

cilium.io
cilium.slack.com
Regular news

Base technology

The revolution in the Linux kernel, safely and efficiently extending the capabilities of the kernel.

ebpf.io
What is eBPF? - ebook

ISOVALENT

Thank you!