HashiCorp Consul

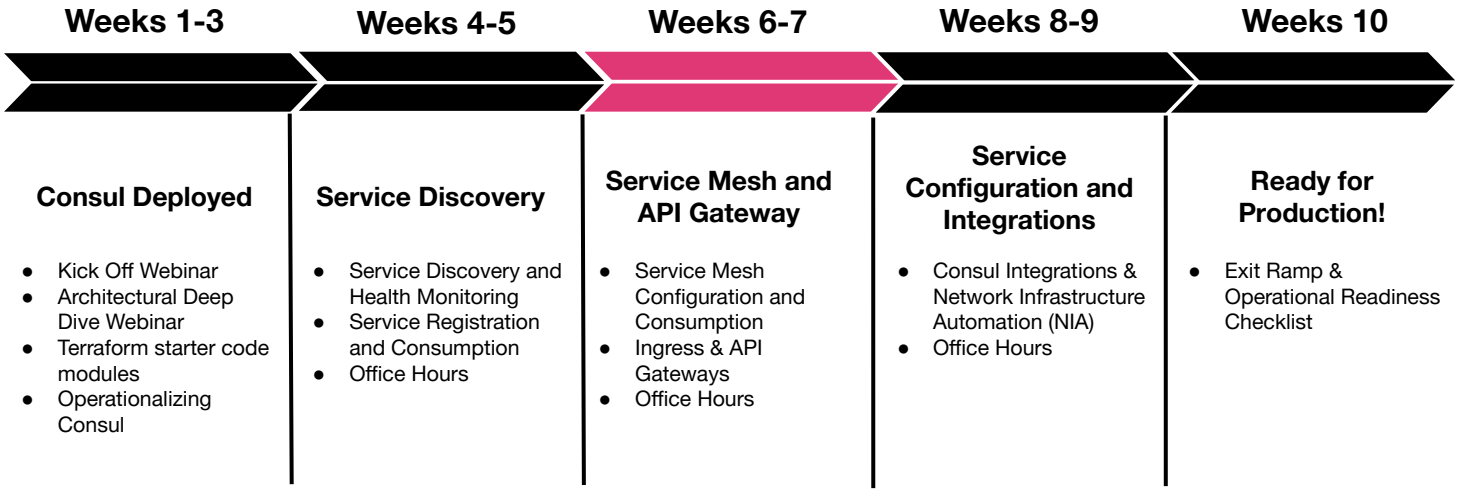
Service Mesh with Consul

August 2022

Agenda

1. What is a Service Mesh?
2. Service Mesh Components
3. Traffic Control
4. Service Resiliency
5. Observability
6. Gateways

Consul Enterprise Path to Production



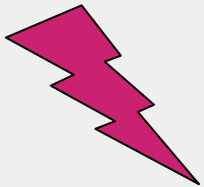
01

What is a Service Mesh?

Method Call became Network Call

Network is reliable, homogeneous and secure

- Latency is zero and bandwidth is infinite



Countermeasures

- Load Balancing
- Traffic Management
- Health Checks
- Service Discovery
- Encryption

Re-Implementation is expensive

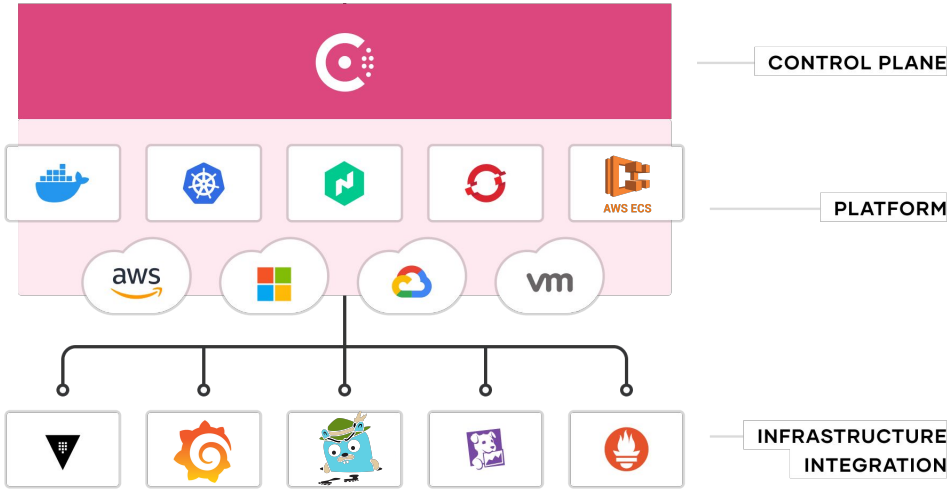
- Global rules and configurations are hard to enforce
- Originally Hystrix and Finagle, but JVM-only

What is a Service Mesh?

- A dedicated infrastructure layer for facilitating service-to-service communication
- Features Include:
  - Consistent Discovery
  - Security
  - Tracing, Monitoring and Failure Handling
  - Does not require a shared asset such as an API gateway
- Consul Connect is used interchangeably with Consul Service Mesh

Multi-Platform Service Mesh

- Automatic load balancing
- Fine-grained control of traffic behavior with routing rules, retries and failovers
- Metrics, logs, and traces for all traffic within a cluster
- Identity-based authentication and authorization for service-to-service communication



02

Components

How Does It Work?

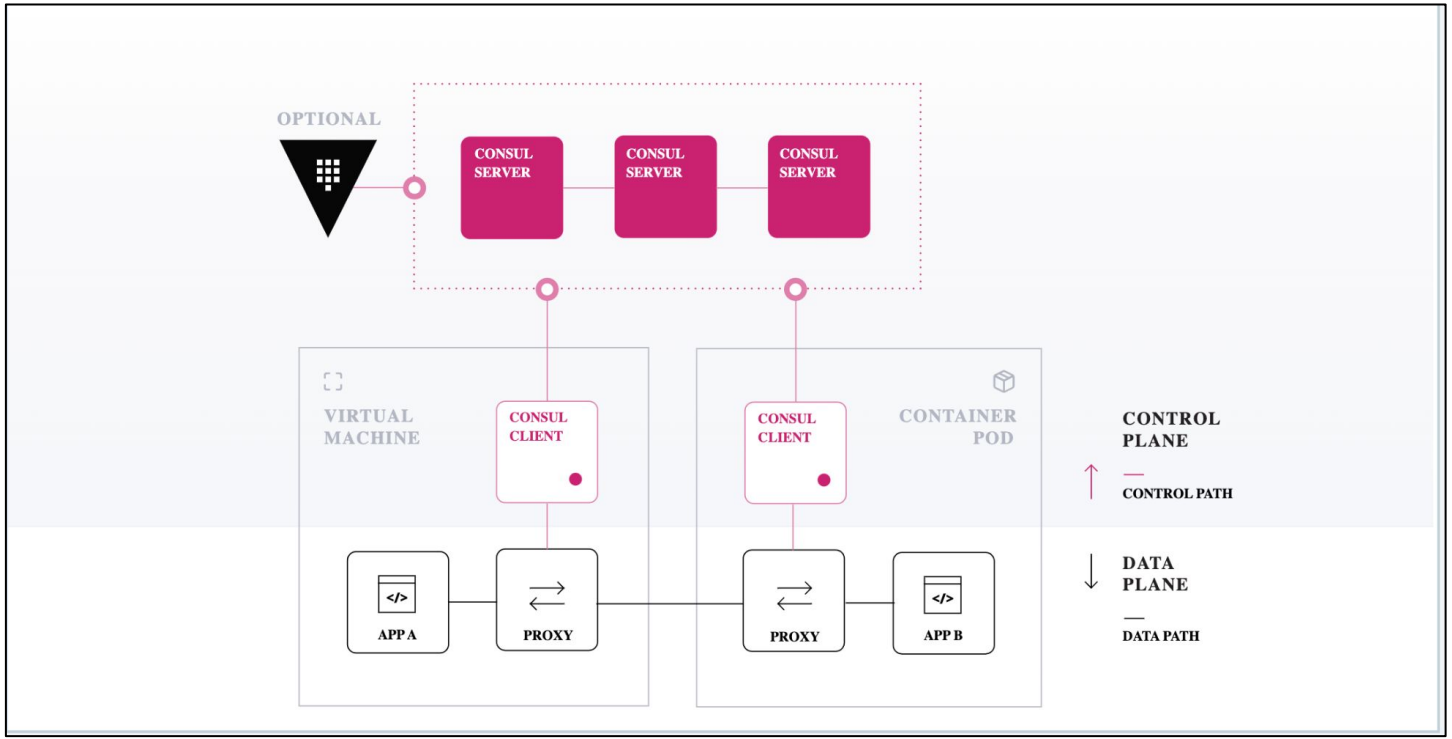
Control Plane

- Manages and configures the proxies to route traffic.
- Configures policies and collect telemetry

Data Plane

- Composed of a set of intelligent proxies (Envoy) deployed as sidecars
- Control all network communication between microservices

Control Plane



02 A

Component 1 - Control Plane

Consul Service Mesh Components

1. **Consul internal certificate authority -**

   Signs services with certificates for mTLS communication

2. **Service sidecar proxies**

   Control and encrypt service traffic

3. **Consul intentions**

   Authorize communications between sidecar proxy enabled (Service Mesh) services

Certificate Authority (CA)

- Provides services with identities
- Certificates are SPIFFE compliant
- Used to establish mutual TLS
- Certificate authority options
  - Built-in CA (not encrypted on disk)
  - Vault
  - AWS ACM Private CA

https://www.consul.io/docs/connect/ca.html

02 B

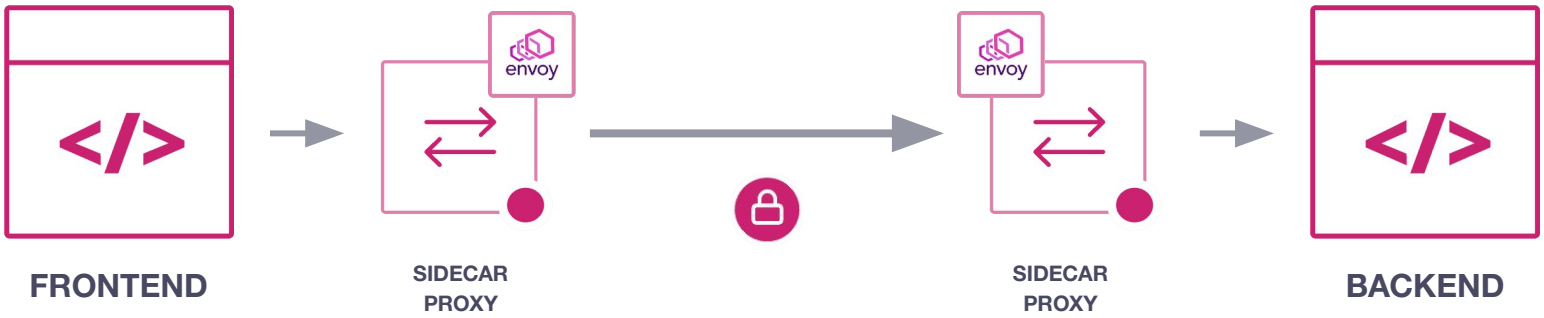
Component 2 - Sidecar Proxy

Sidecar Proxies

- Typically implemented with reverse-proxy processes deployed alongside each service process
- Communicate with service registries, identity providers, log aggregators, etc
- Automatically establish TLS connections
- Secure inbound and outbound connections
- Applications are unaware (connect via localhost)
- Supported Proxies
  - Envoy preferred - (enables Layer 7 policies)
  - Built-in L4 only for dev and testing, use Envoy in Prod deployments
  - Extensible for 3rd party proxies

Service Mesh Overview

Service Mesh Overview



Envoy - Preferred Proxy

- Service proxy written in C++
- Controls all inbound and outbound traffic
- Deployed as sidecar
- Consul leverages many built-in features, for example:
  - Load balancing
  - TLS termination
  - Health checks
  - Traffic Split
  - Metrics

02 C

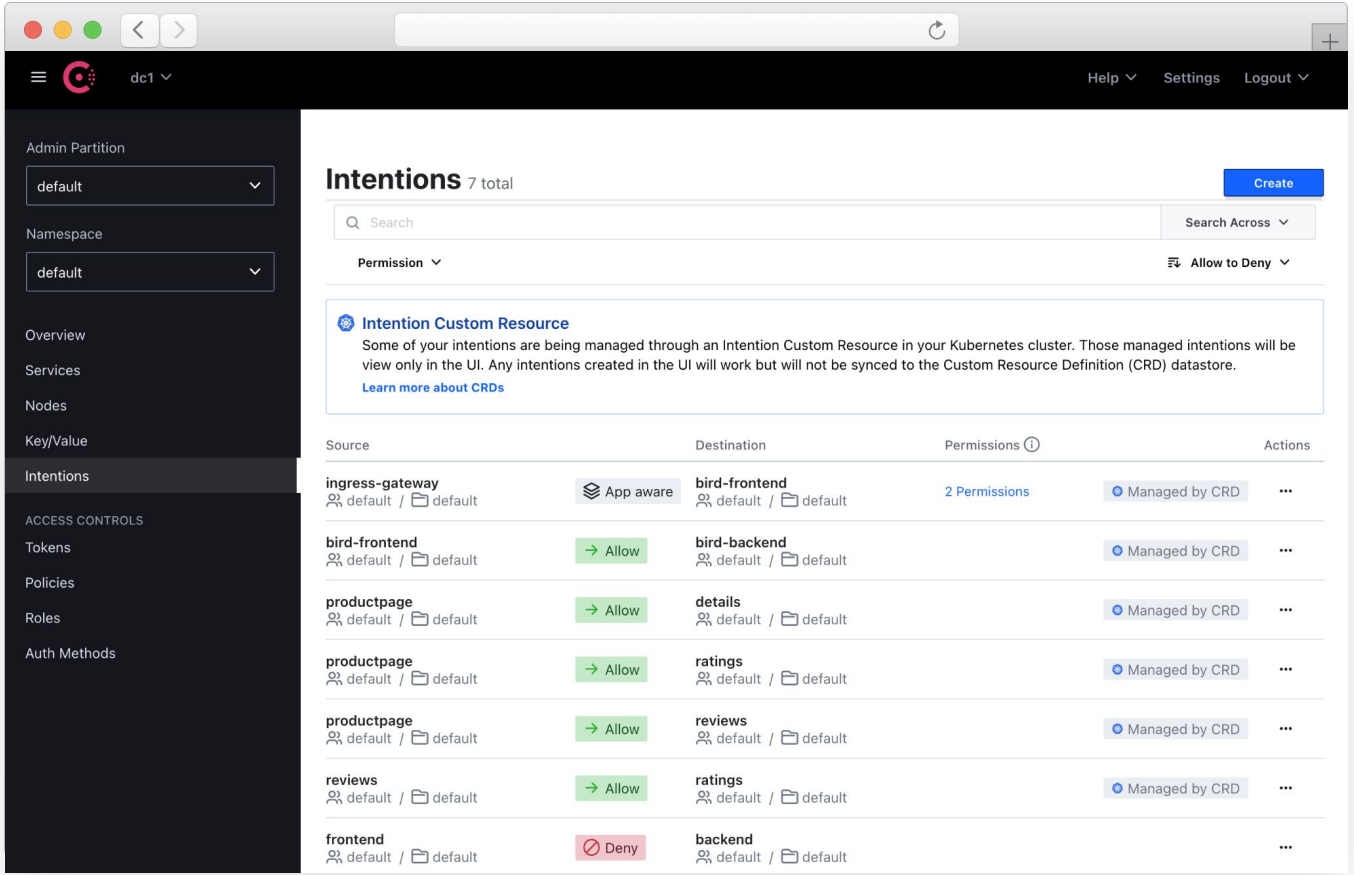
Component 3 - Intentions

Consul Intentions

Intentions **control service-to-service communication**

- Allow or deny service traffic
- Logical service name (not IP)
- Scales independent of instances
- Consistency ensured with Raft
- Manage with web UI, CLI, API and Kubernetes CRD
- Multi-datacenter support

Intentions - UI



Intentions


CODE EDITOR
Kind = "service-intentions"
Name = "web"
Sources = [
{
Name = "ingress-service"
Action = "allow"
},
{
Name = "api"
Action = "deny"
}
]

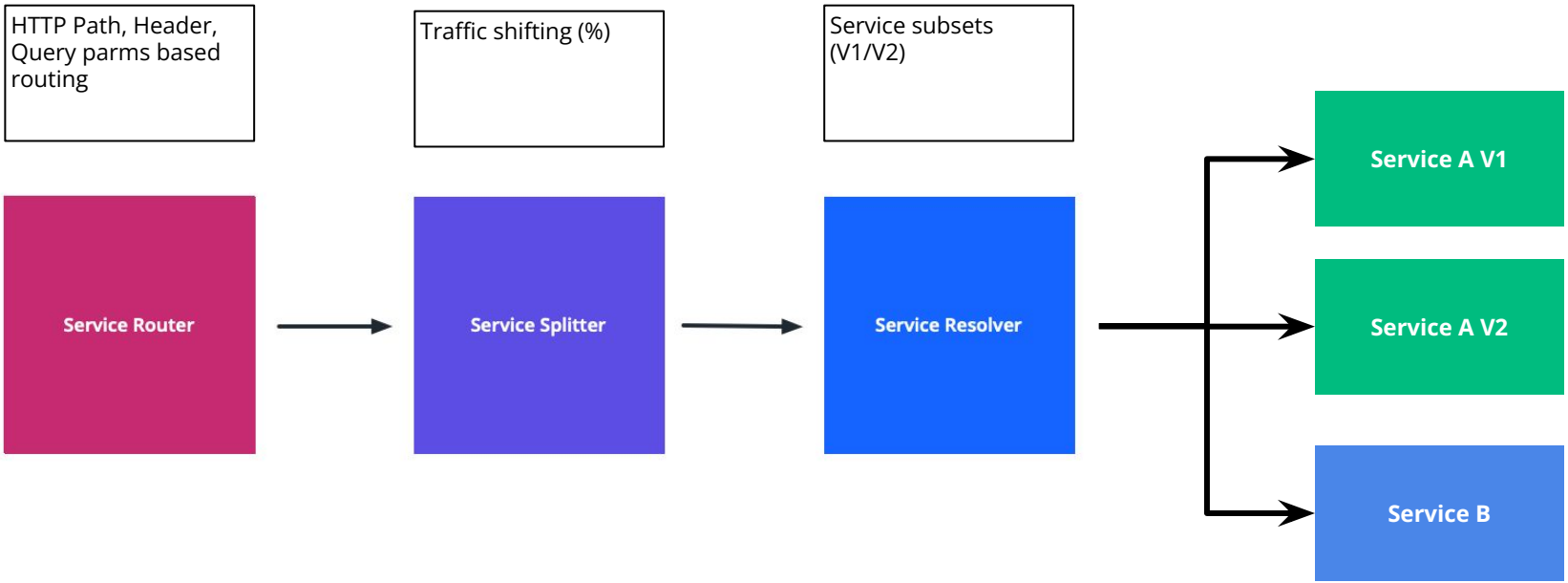
Intentions (Application Aware)

03

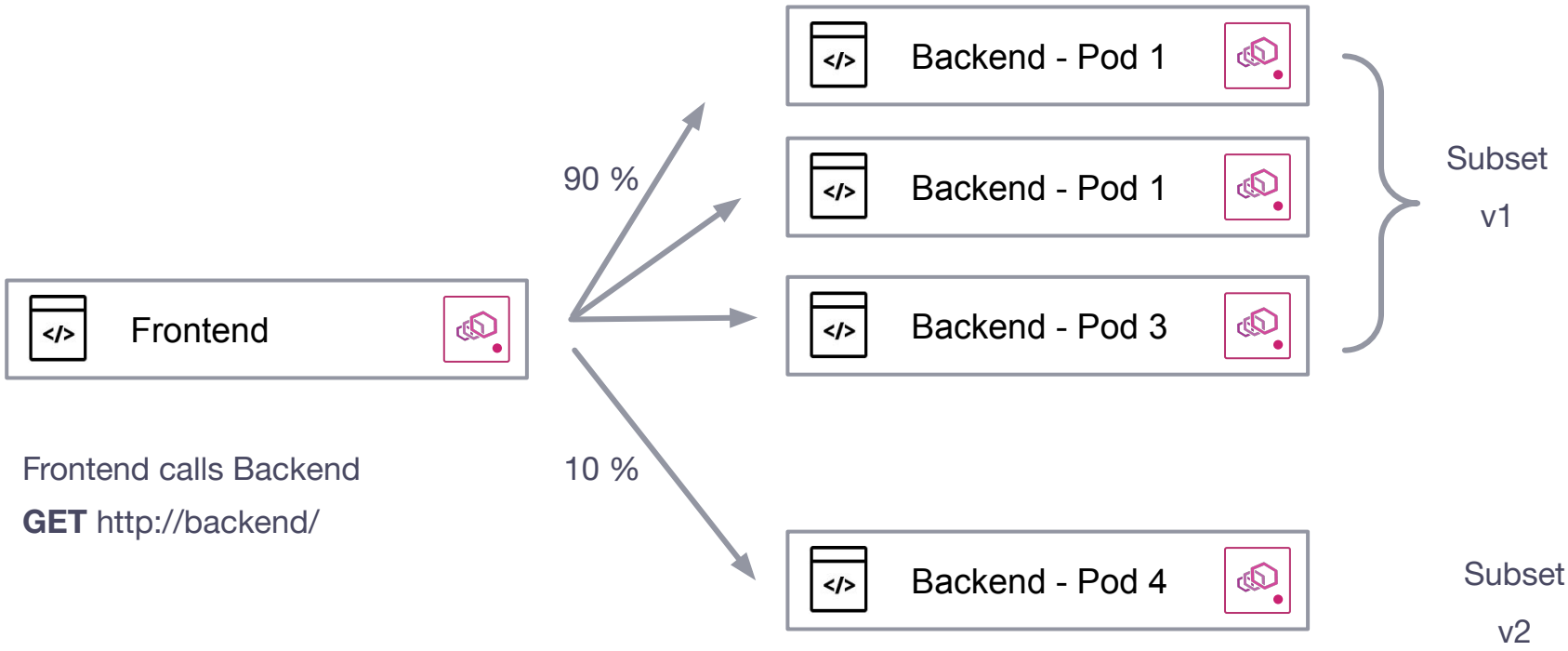
Traffic Control

Consul Discovery Chain

Consul Discovery Chain



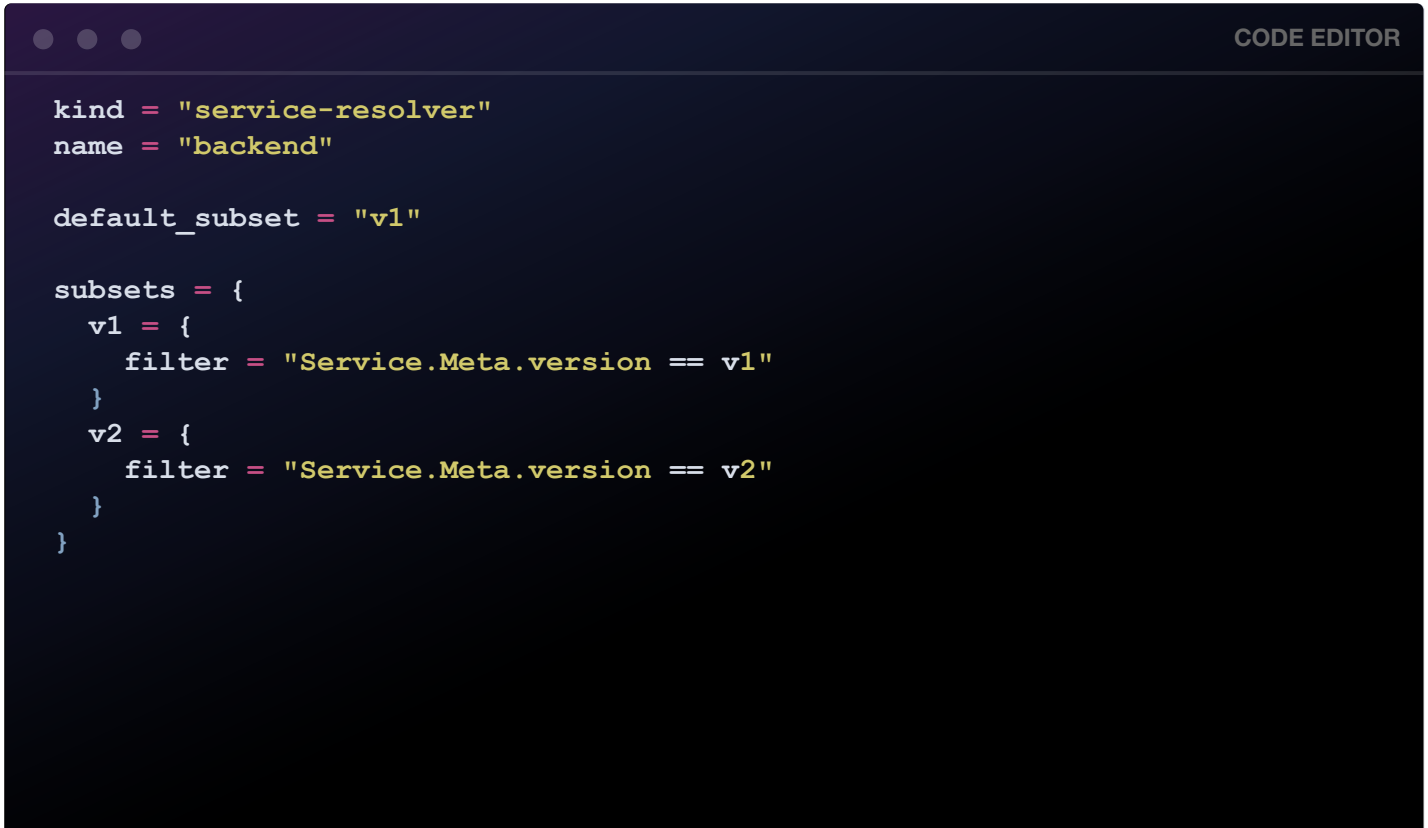
Traffic Splitting - Weights



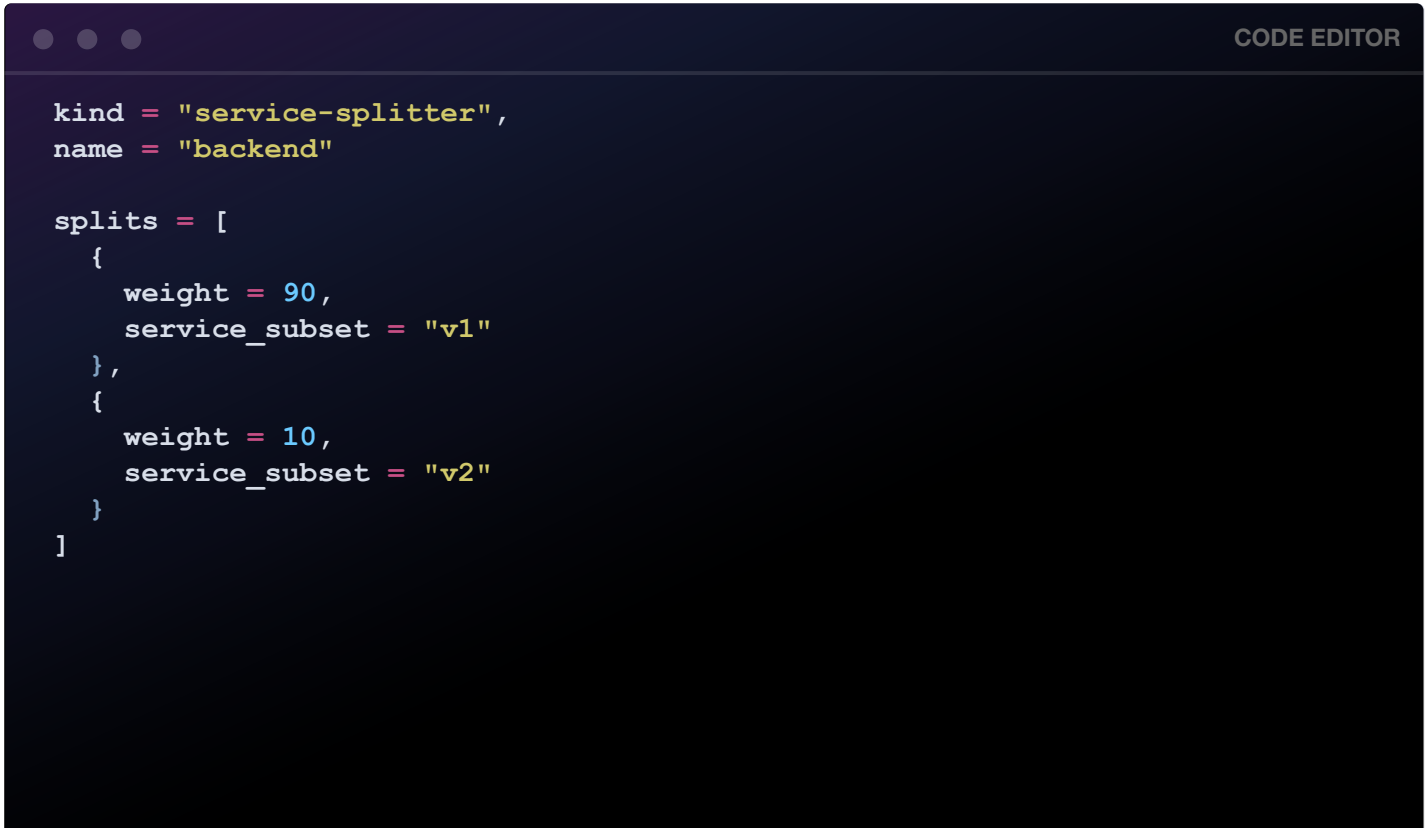
Frontend calls Backend

GET http://backend/

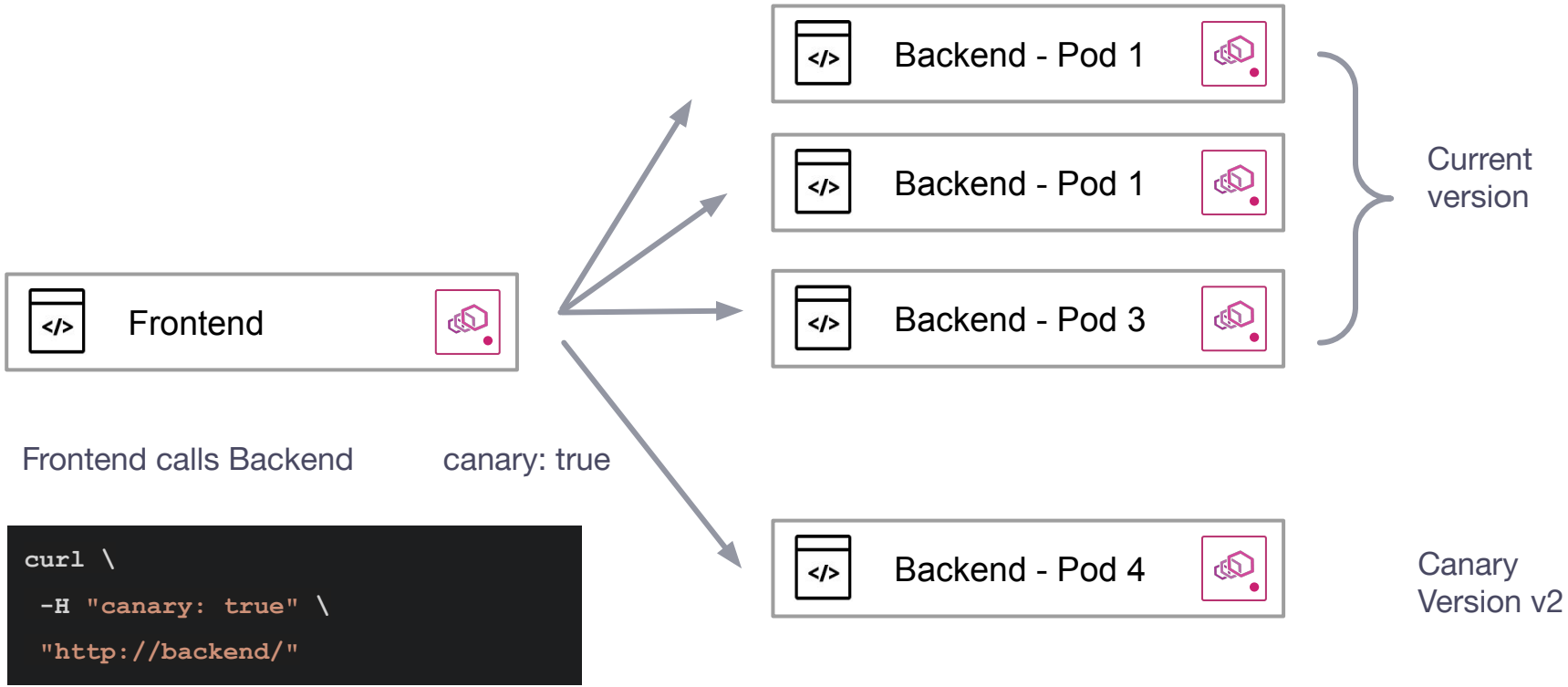
Service Resolver



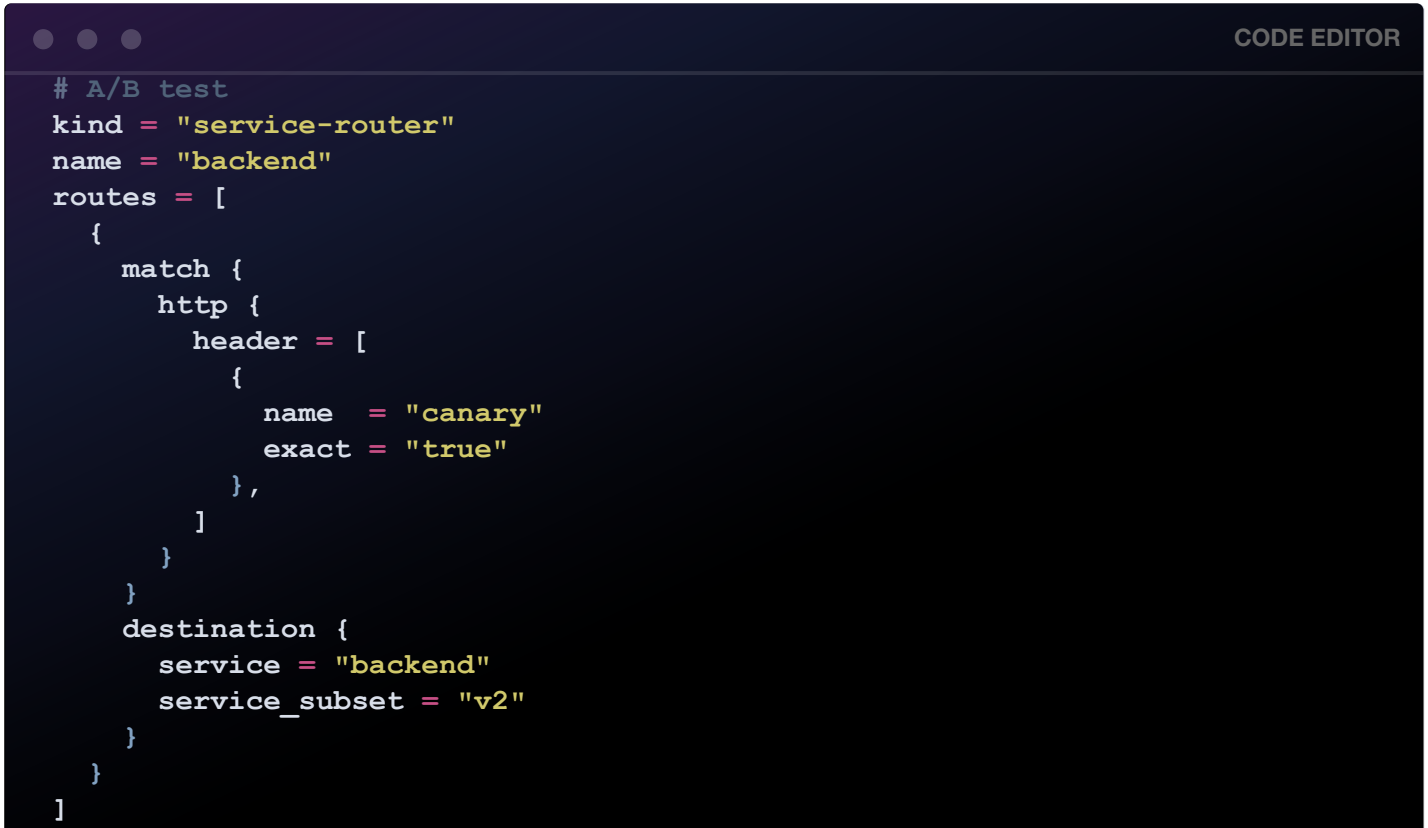
Service Splitter



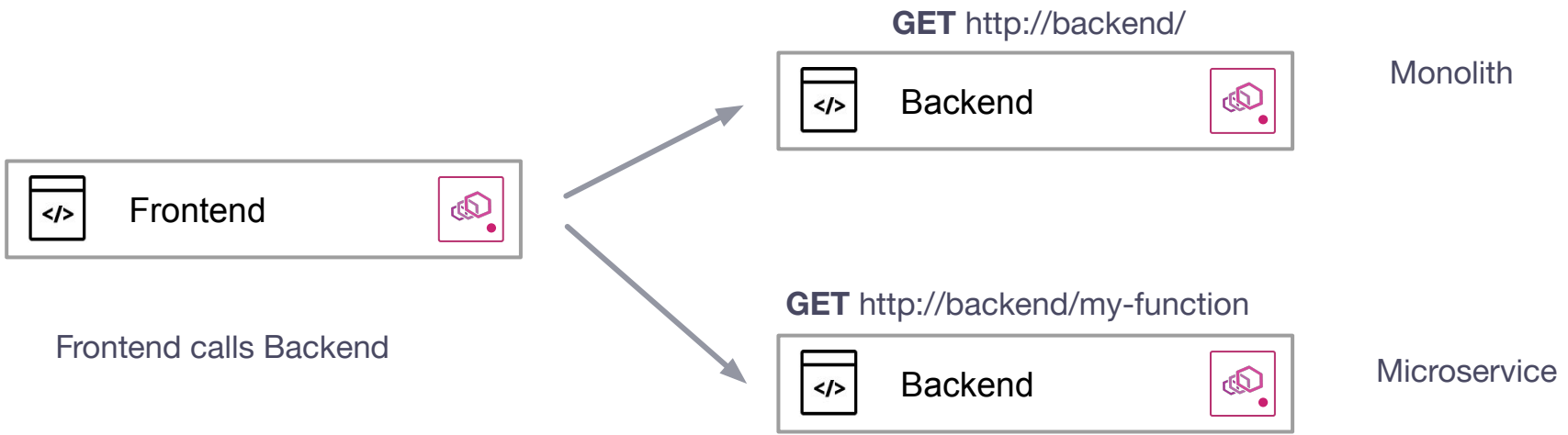
Traffic Splitting - Request Header



Service Router



Traffic Splitting - Request Path

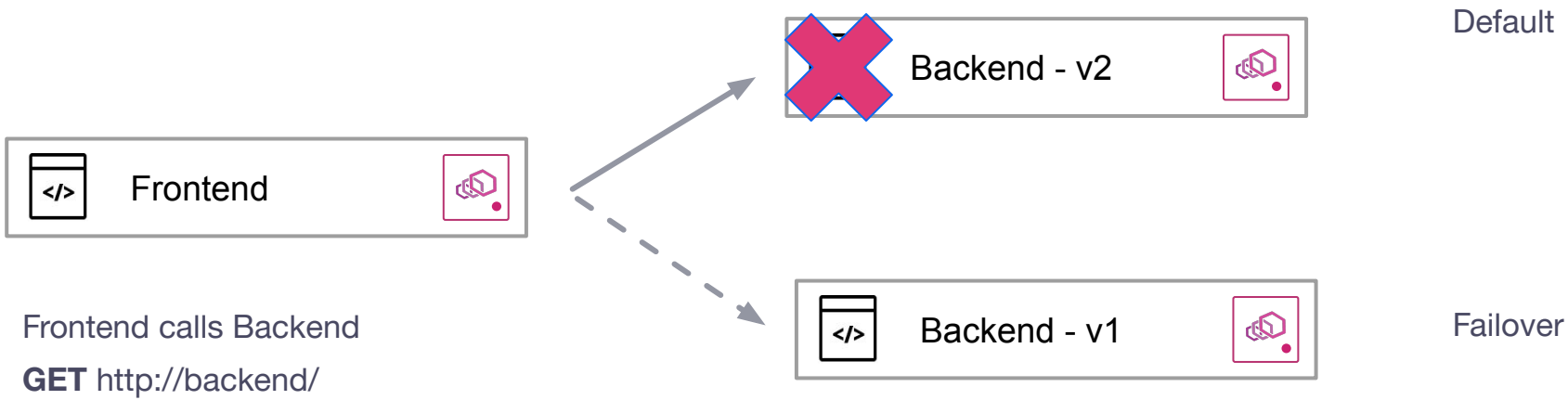


Service Router

04

Service Resiliency

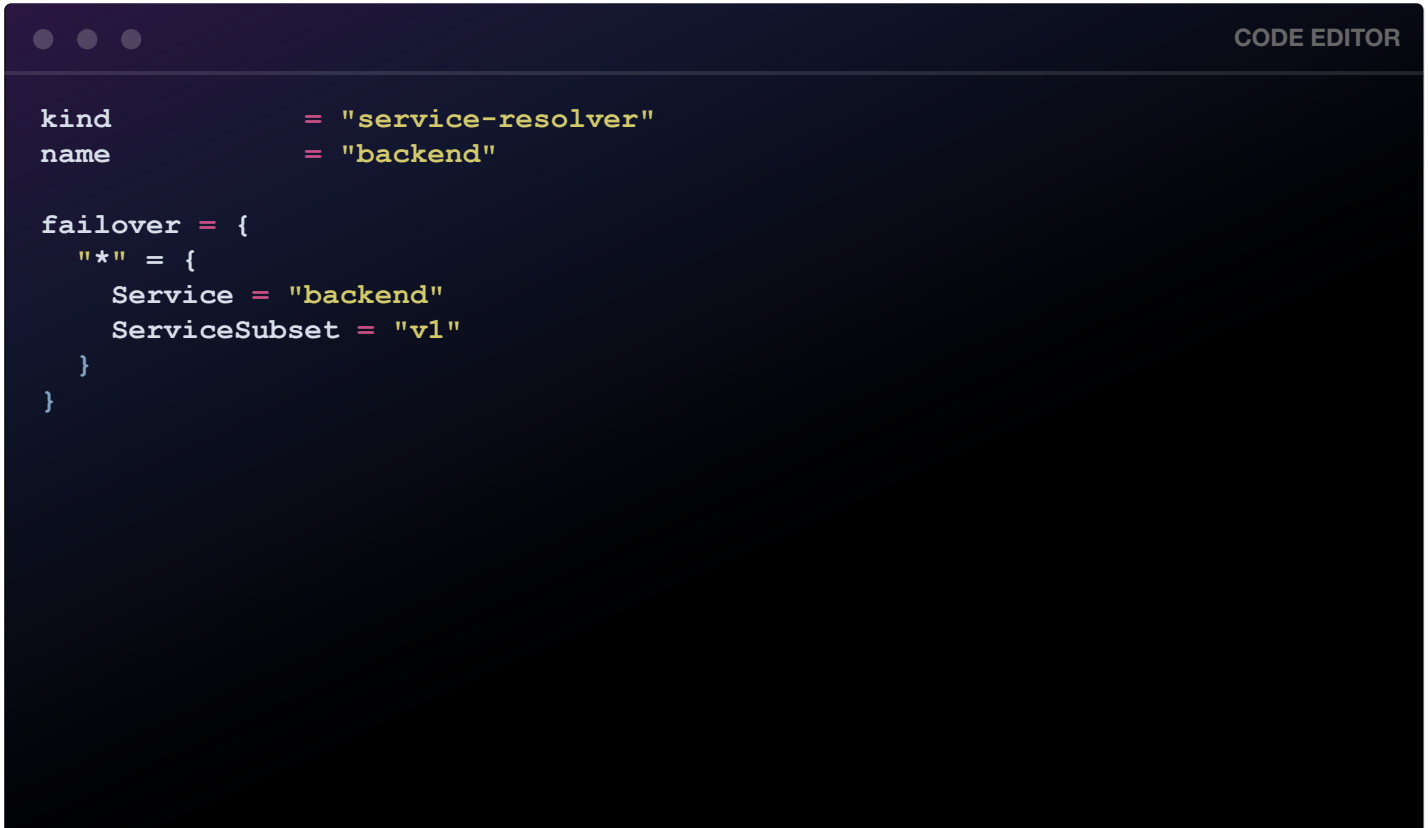
Failover



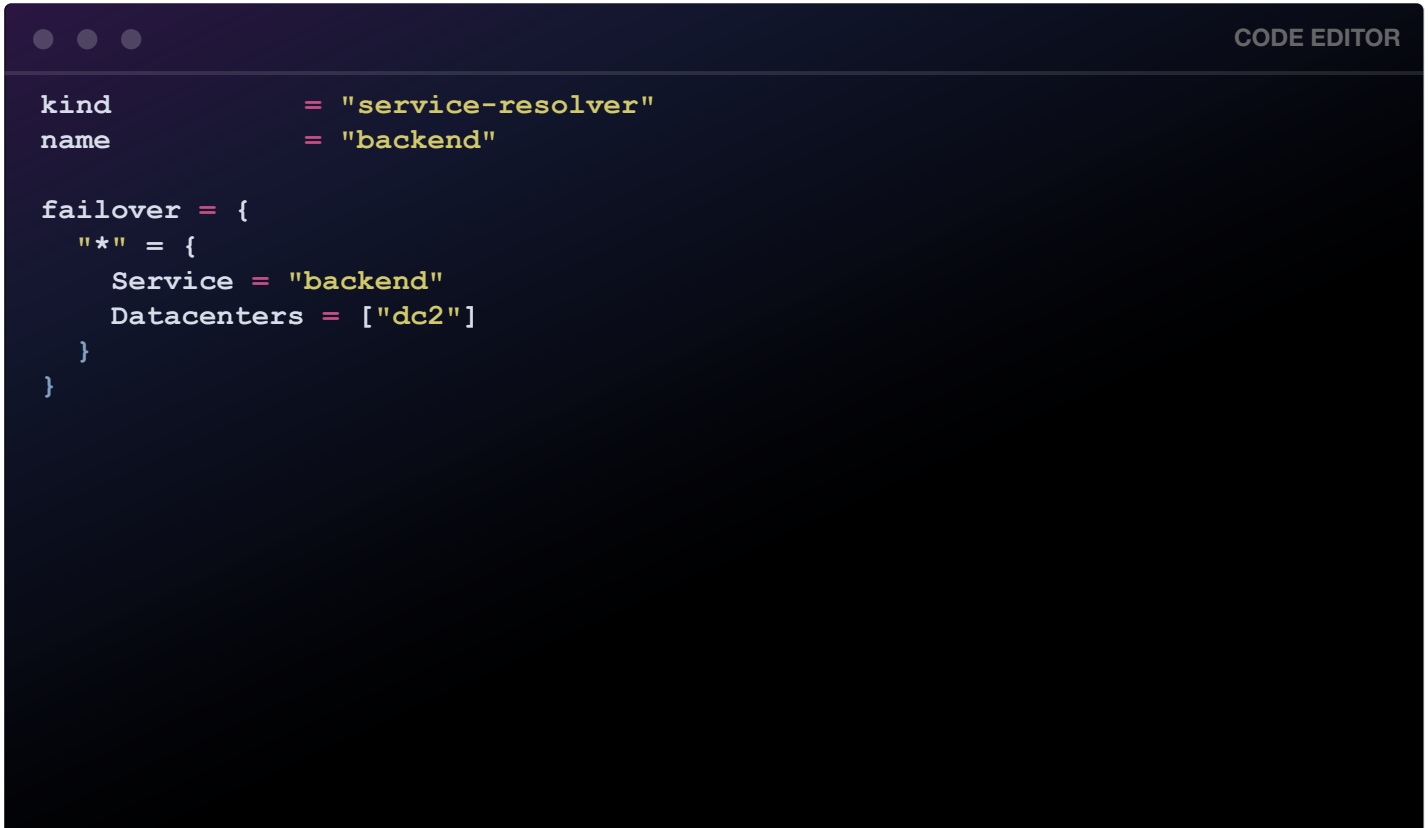
Frontend calls Backend

GET http://backend/

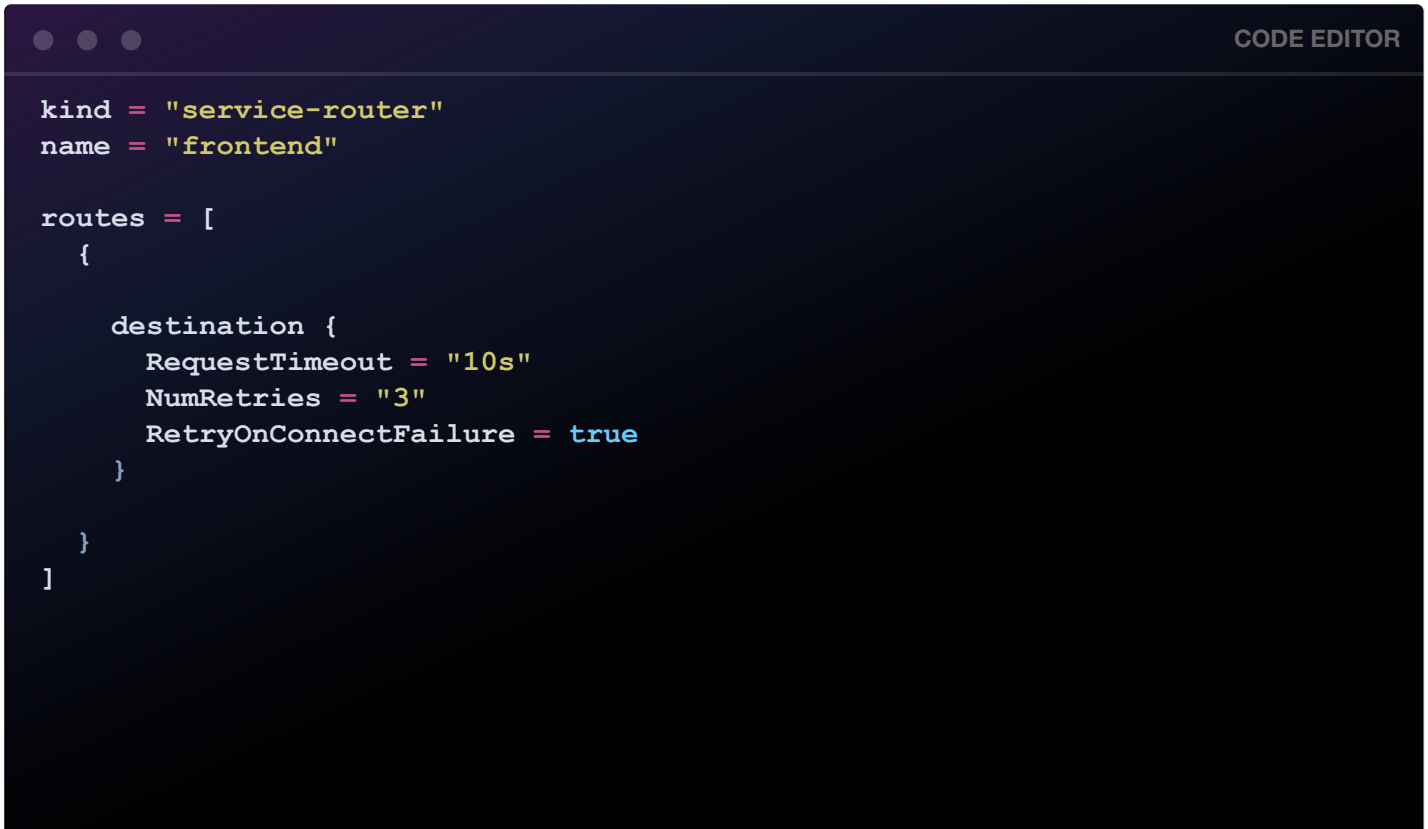
Failover Subset



Service Datacenter



Retry and Timeout



05

Observability

Observability

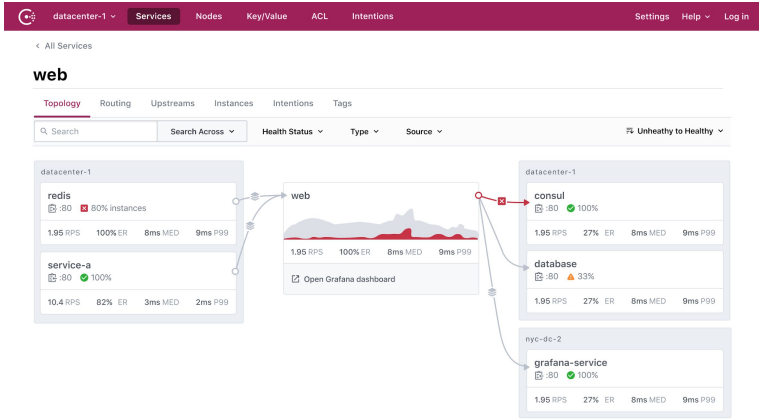
A measure of how well you can infer a system's internal state by observing its outputs

- Checks that increase observability include:
  - HTTP status codes
  - Response latency
  - Tracing data
- Traditionally, these methods were built into services themselves.

Service Mesh Visualization

Ability to see what services are communicating with which services within the service mesh.

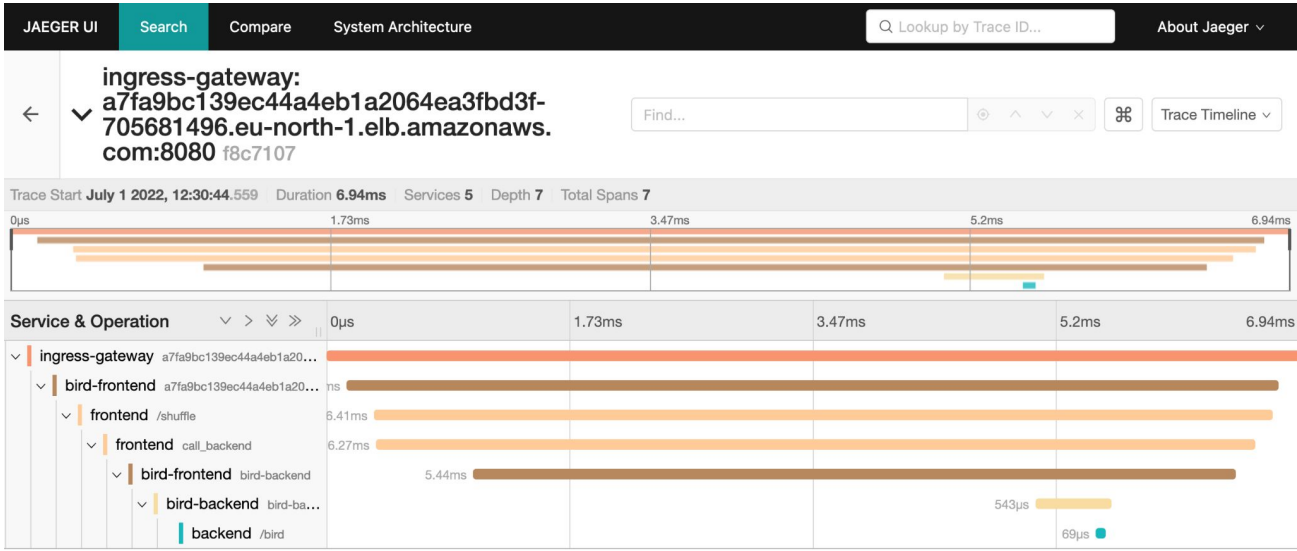
The side-car proxies are emitting metrics, and you can scrape these via Prometheus and visualize them via Grafana dashboards.



Distributed Tracing

Observe requests that span multiple microservices.

Integration with Tracing Collectors like Jaeger by emitting spans via the Zipkin format.

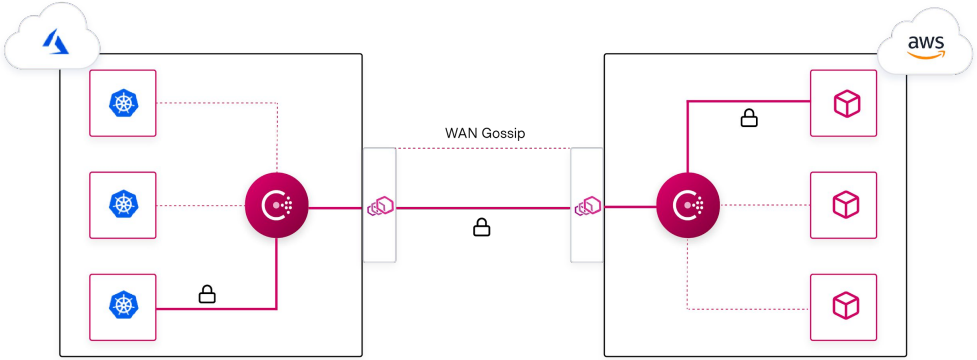


06

Gateways

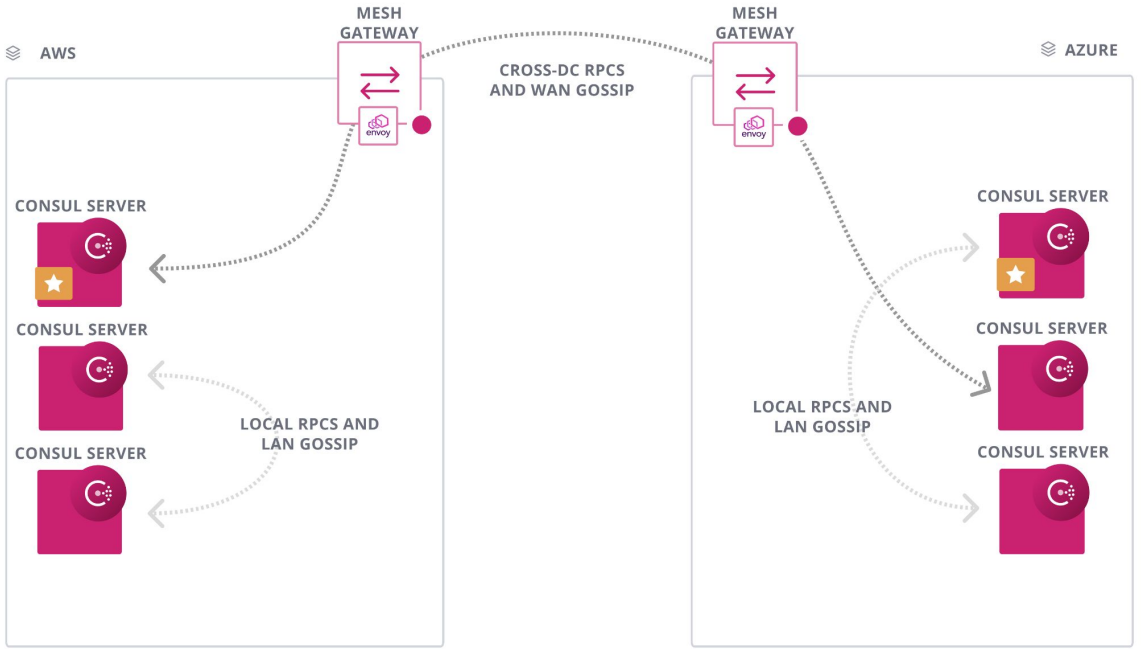
Mesh Gateway

- Extend Consul service mesh across potentially complex environments:
  - Isolated networks
  - Overlapping IP addresses
  - Kubernetes environments
- Automatic mTLS Encryption between Consul datacenters
- Layer 7 Traffic Policies between environments



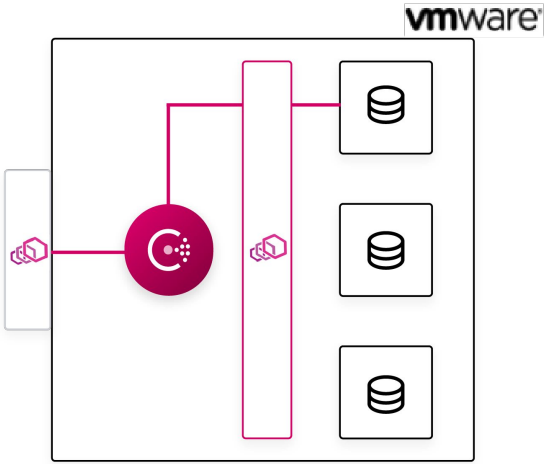
WAN Federation over Mesh Gateways

- All Consul traffic passes through Mesh Gateway - including WAN Gossip
- Secure service communication between clouds without VPN or cloud specific services
- Deliver redundancy and lifecycle capabilities across cloud environments



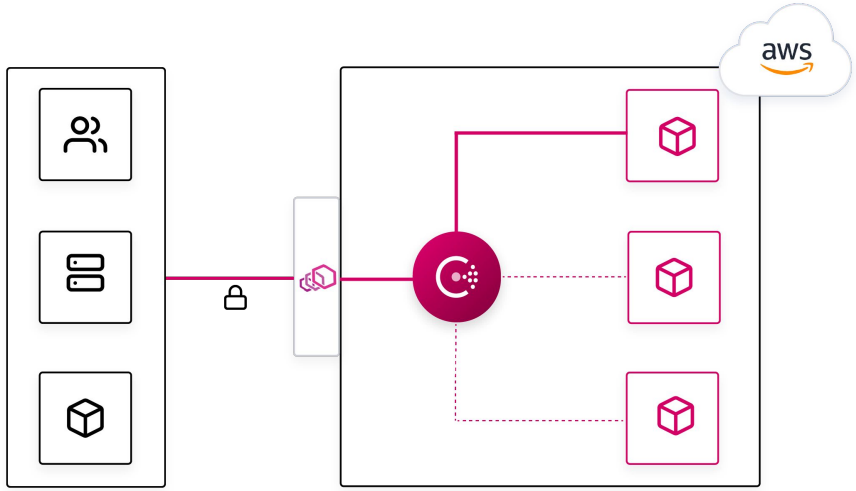
Terminating Gateway

- Outbound access from service mesh workloads
- Extend Consul security and traffic policies to external infrastructure and managed services
- Acts as an egress proxy for the service mesh environment

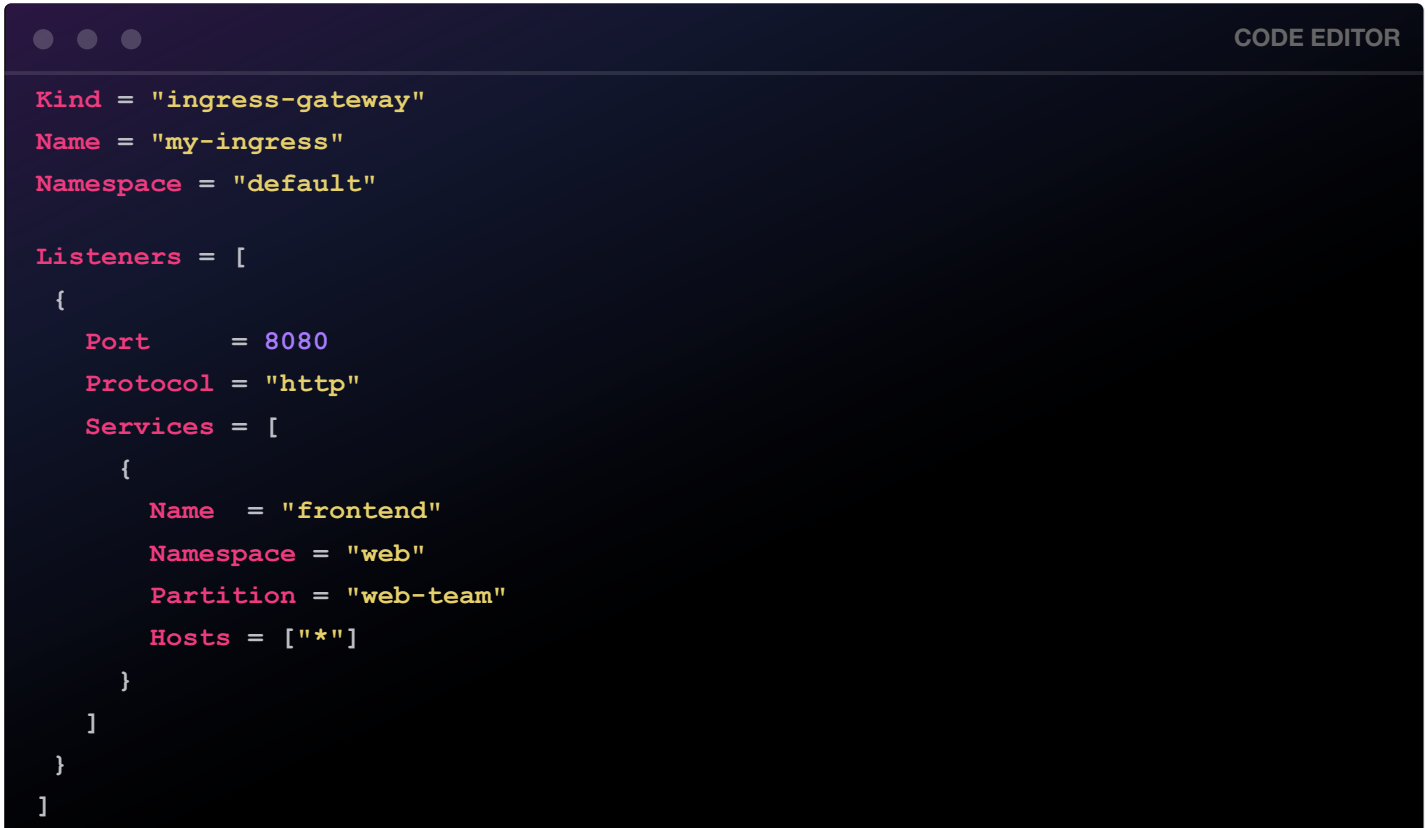


Ingress Gateway

- Inbound access to service mesh workloads
- Enables one-to-many access between ingress gateway and service mesh services
- Logical point of layer 7 traffic policies
- Route traffic to services based on URI path, header, etc...



Example: Ingress Gateway

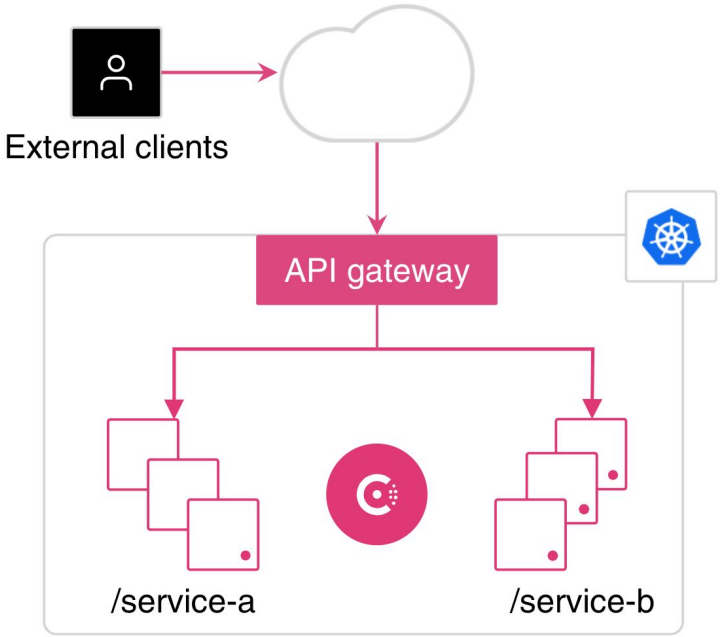


API Gateway

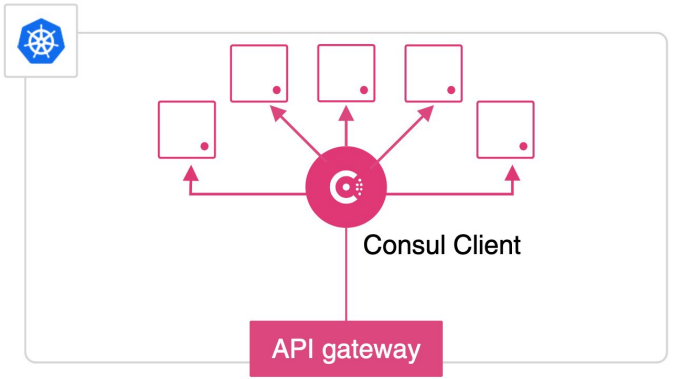
Control access to the service mesh

Consul API Gateway is designed to be deployed alongside Consul service mesh helping users control access into and out of the service mesh. It provides organizations the ability to:

- Control access at the point of entry
- Centralize traffic management
- Route requests based on metadata



Consul API Gateway Installation



Step 1: Install CRDs

```
$ kubectl apply
--kustomize="github.com/hashicorp/consul-api-gateway/config
/crd?ref=v0.1.0"
```

Step 2: Configure & Install via Helm

```
apiGateway:
  enabled: true
  image: hashicorp/consul-api-gateway:0.1.0
```

```
$ helm install consul ... –version “0.40.0”
```

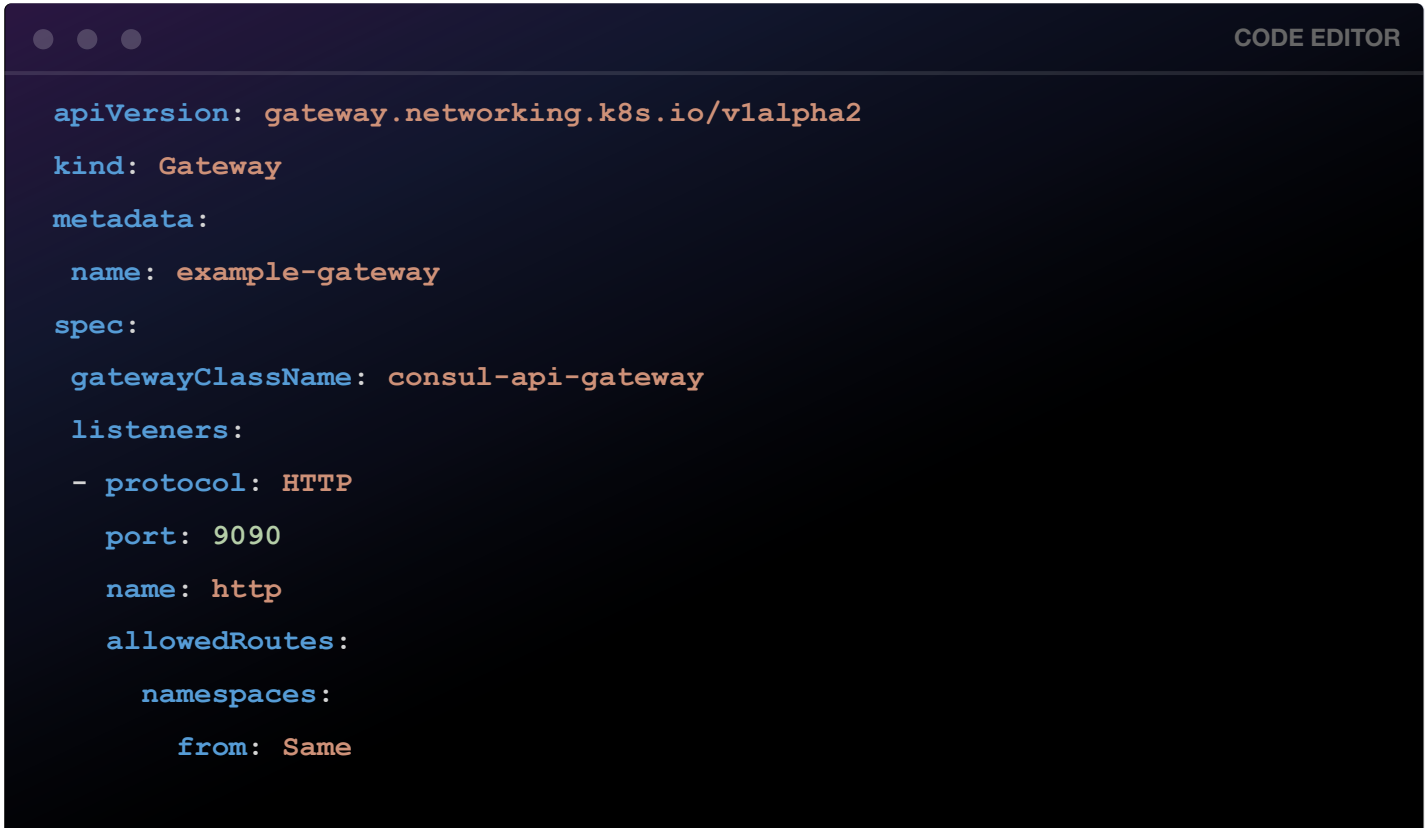
API Gateway Components

Feature	Description
Gateway	Defines the main infrastructure resource that links API gateway components. It specifies the name of the GatewayClass and one or more Listeners, which specify the logical endpoints bound to the gateway's addresses.
GatewayClass	Defines a class of gateway resources that you can use as a template for creating gateways.
GatewayClassConfig	Describes additional Consul API Gateway-related configuration parameters for the GatewayClass resource.
Routes	Specifies the path from the client to the services. They attach to Listeners on the Gateways.
API Gateway Controller	Controller required for deploying and managing the API Gateway service. Retrieves configuration and authentication credentials from the GatewayClassConfig definition.

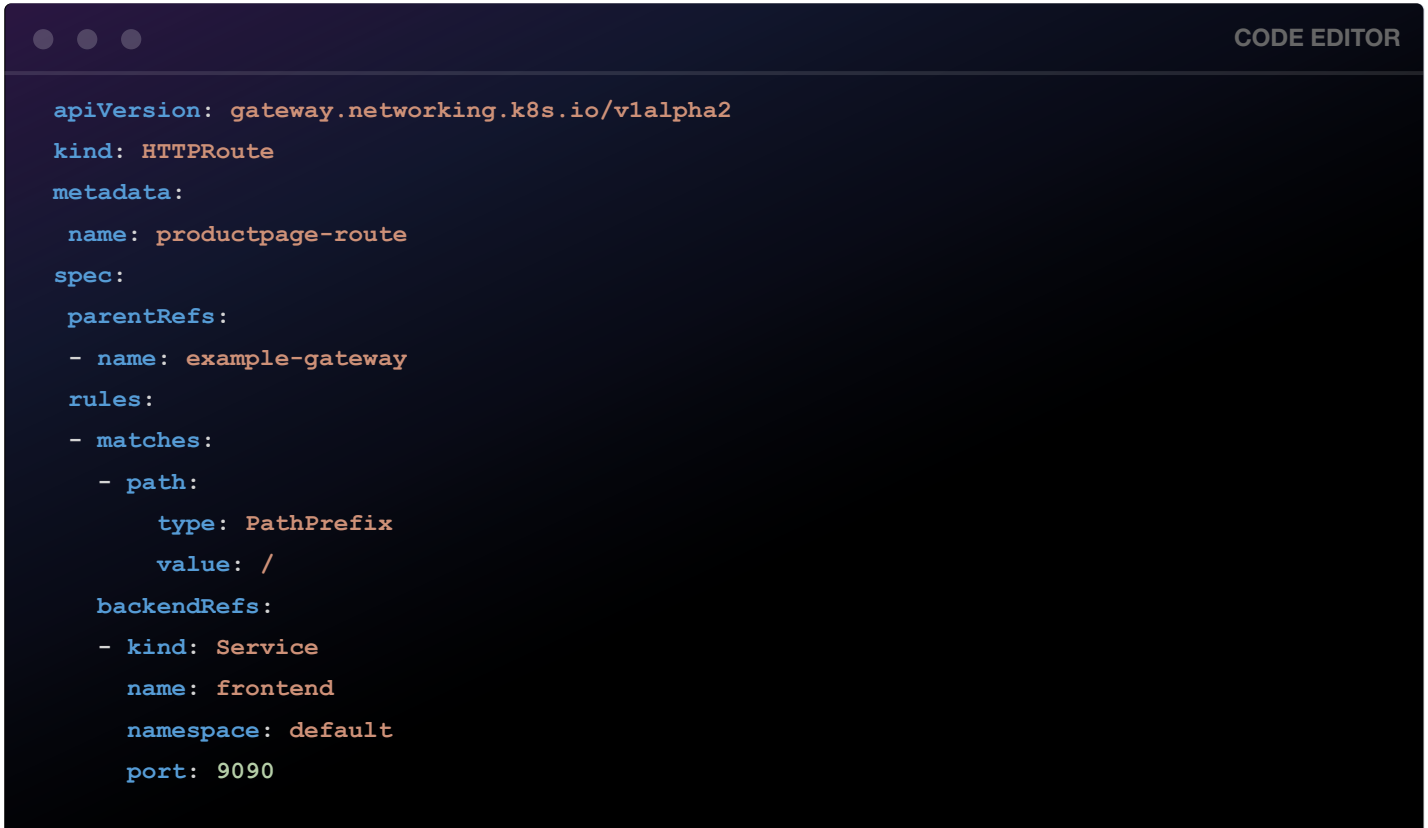
API Gateway Supported Features

- Multiple Listeners (HTTP, HTTPS, TCP, and TCP + TLS)
- Load Balancing
- Client Traffic Routing based on matching criteria
  (URL, Hostname, Headers & values, HTTP Method and Query string)
- Use TLS Certs signed by external CA
- Split traffic based on weighted ratios
- Add/Delete/Modify HTTP headers and values
- Kubernetes

Example: Gateway



Example: HTTPRoute



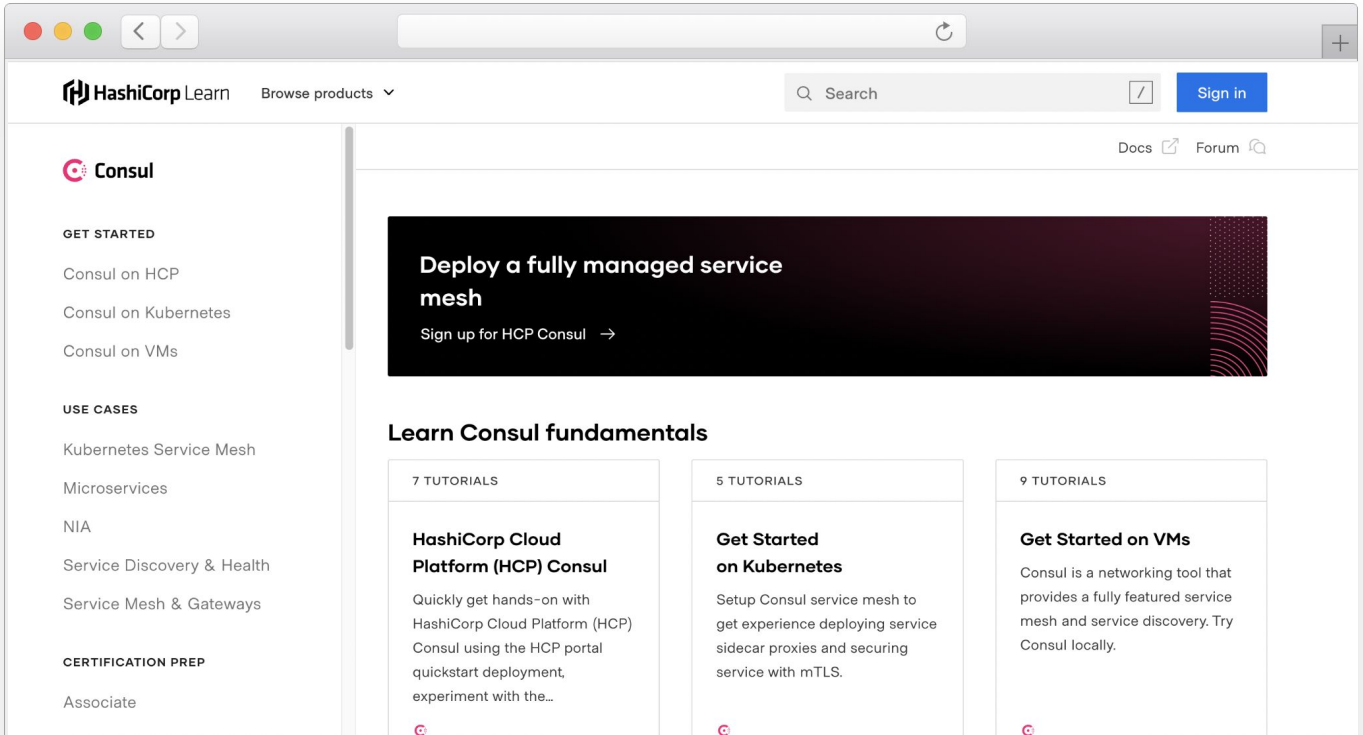
06

Next Steps

Learn

https://learn.hashicorp.com/consul

Step-by-step guides to accelerate deployment of Consul



Resources

- Consul Service Mesh
- Consul Service Mesh with Kubernetes
- Certificate Management
- Intentions
- Discovery Chain
- Observability
- Observability with Kubernetes, Tutorial
- Distributed Tracing
- Multi-Cluster Federation via Mesh Gateways
- API Gateway

Need Additional Help?

Customer Success

Contact our Customer Success Management team with any questions. We will help coordinate the right resources for you to get your questions answered.

customer.success@hashicorp.com

Technical Support

Something not working quite right? Engage with HashiCorp Technical Support by opening a ticket for your issue at support.hashicorp.com.

Discuss

Engage with the HashiCorp Cloud community including HashiCorp Architects and Engineers

discuss.hashicorp.com

Upcoming Onboarding Webinars

Office Hour

An interactive open forum to discuss specific questions about your environment and Use Cases.

Please bring your questions!

Webinar:

Service Configuration and Integrations

Topics include: Network Infrastructure Automation & Consul Terraform Sync

Webinar:

Program Closing

A wrap up of the program content and Consul Production Readiness Checklist

Q & A

Thank You

customer.success@hashicorp.com
www.hashicorp.com/customer-success