Shifting com istiops

Estratégias de deploy do Istio com Golang

pismo

O que é o Istio?

Service Mesh

- Segurança e políticas
- Observabilidade
- Escalabilidade e performance
- Modelos de deployment
- Gerenciamento de tráfego

Gerenciamento de tráfego

Funcionalidades

- Auto-discovery de serviços e balanceamento
- Roteamento de tráfego
- Resiliência de rede e testes

Gerenciamento de tráfego

Como opera?

- VirtualServices
- DestinationRules

Barreiras

- Difícil manutenção
- Difícil de conseguir um estado atual dos tráfegos
- Componentes sensíveis

O que é o istiops ?

- Simplifica o gerenciamento de tráfego do Istio
- Agnóstico à plataforma como CLI
- "Plugável"
- Open source 🙌

Por que escolhemos Golang?

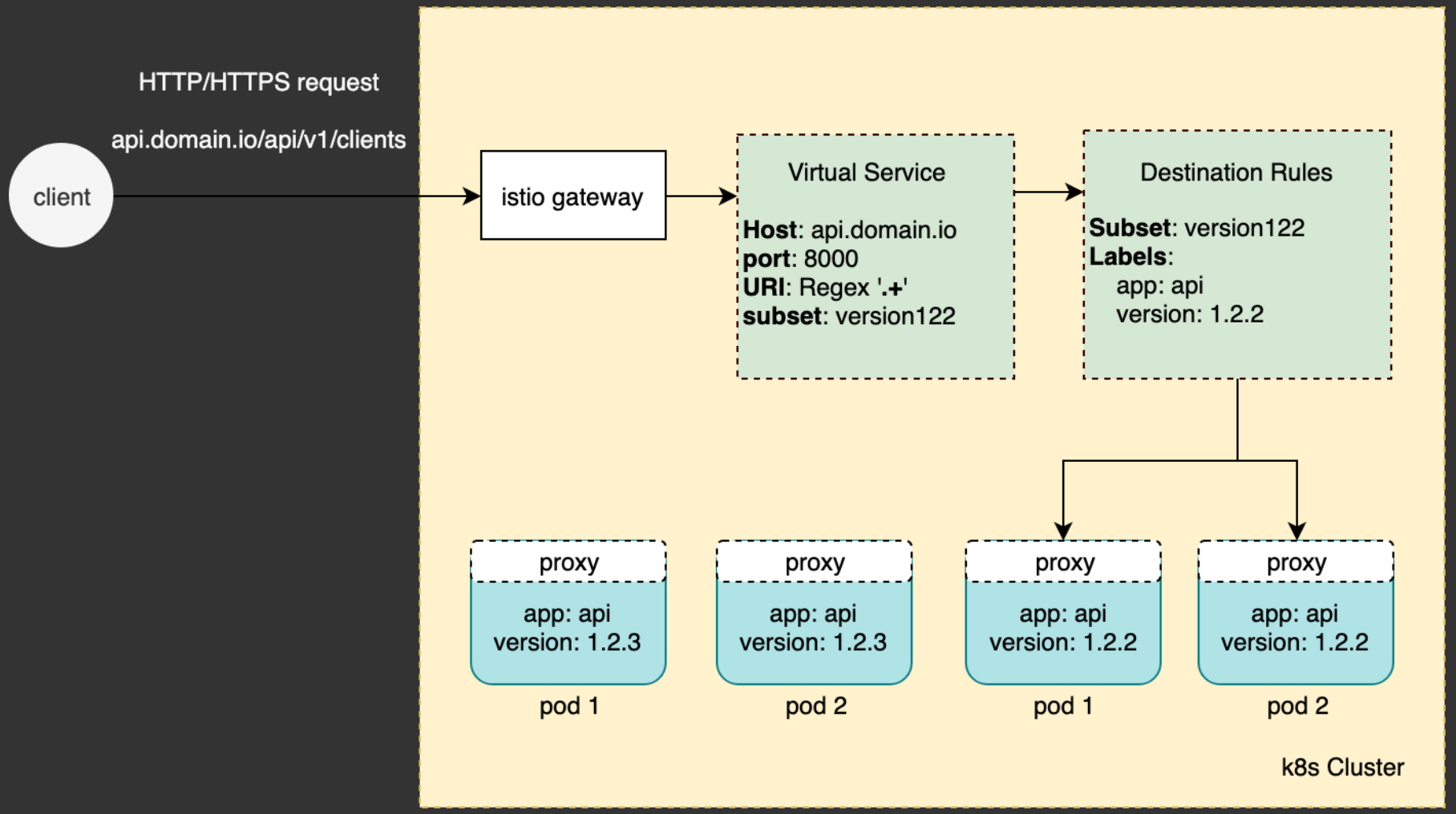
- Fácil de manter
- Fácil de testar
- Velocidade de execução
- Binário multi-plataforma como artefato
- Orientado a Interfaces
- Boa curva de aprendizagem
- Debugging

Estratégias de deploy

Como o istiops é capaz de me ajudar com isso?

Como eu uso o istiops ?

A deep dive ao CLI & uso geral


HTTP/HTTPS request
api.domain.io/api/v1/clients
client
istio gateway
Virtual Service
Host: api.domain.io
port: 8000
URI: Regex '.+'
subset: version122
Destination Rules
Subset: version122
Labels:
app: api
version: 1.2.2
proxy
app: api
version: 1.2.3
pod 1
proxy
app: api
version: 1.2.3
pod 2
proxy
app: api
version: 1.2.2
pod 1
proxy
app: api
version: 1.2.2
pod 2
k8s Cluster

```
 _       _    _
(_)___| |_(_) ___   _ __   ___
| / __| __| |/ _ \| '_ \ / __|
| \__ \ |_| | (_) | |_) \__ \
|_|___/\__|_|\___/| .__/|___/
                  |_|

Istiops is a CLI library for Go that manages istio's traffic shifting easily.

Usage:
  istiops [command]

Available Commands:
  help        Help about any command
  traffic     Manage istio's traffic rules
  version     Shows current build version

Flags:
  -h, --help   help for istiops

Use "istiops [command] --help" for more information about a command.
```

show

```
$ istiops traffic show \
--label-selector environment=pipeline-go \
--namespace default \
--output pretty
```

Get all active traffic rules

clear

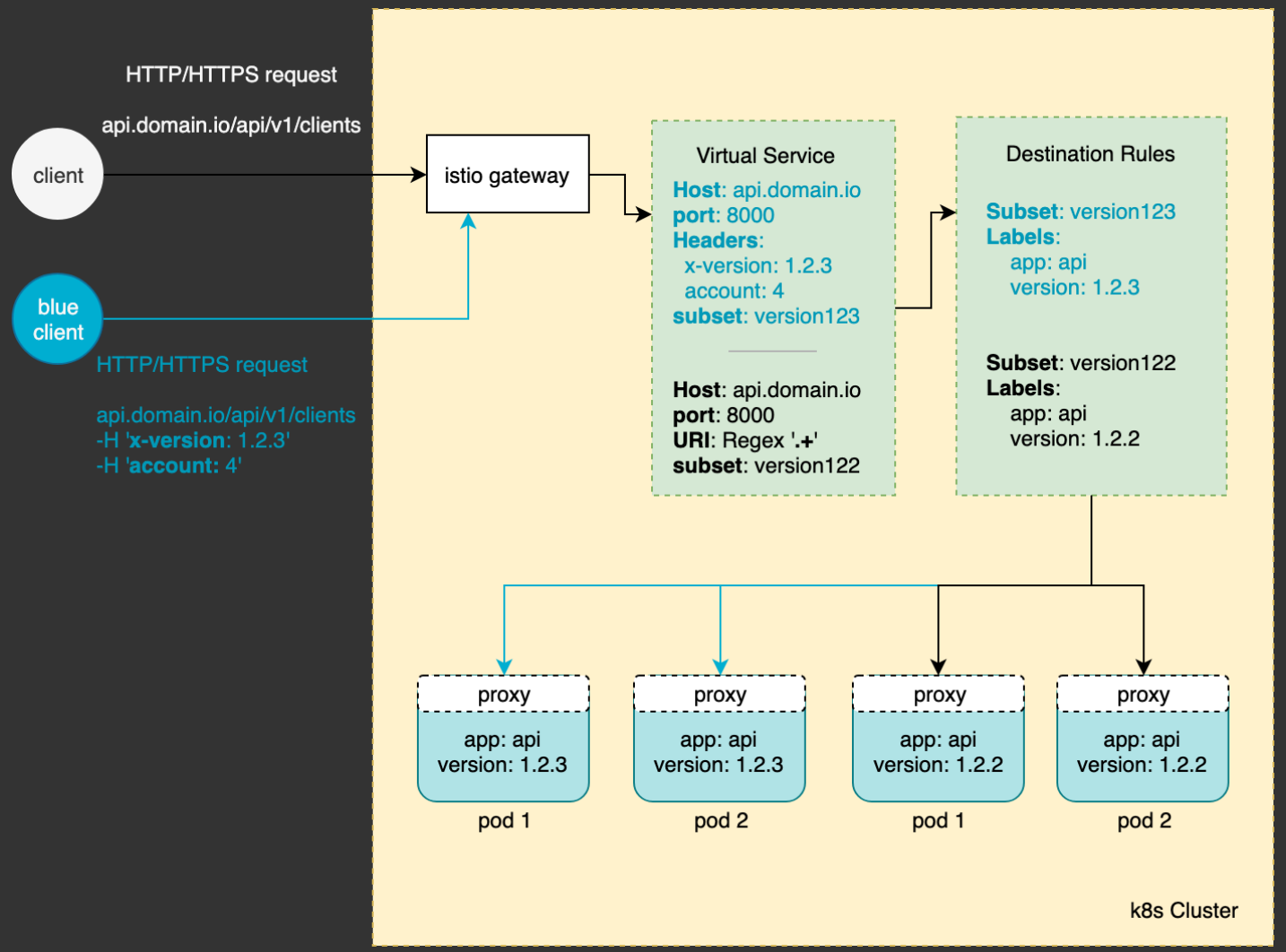
```
$ istiops traffic clear \
--label-selector environment=pipeline-go \
--namespace default
```

Clear all rules (except master one)

headers

```
$ istiops traffic shift \
    --build 12 \
    --label-selector environment=pipeline-go \
    --headers x-version=1.2.3,account=4 \
    --destination api-domain:8000 \
    --pod-selector app=api,version=1.2.3
```

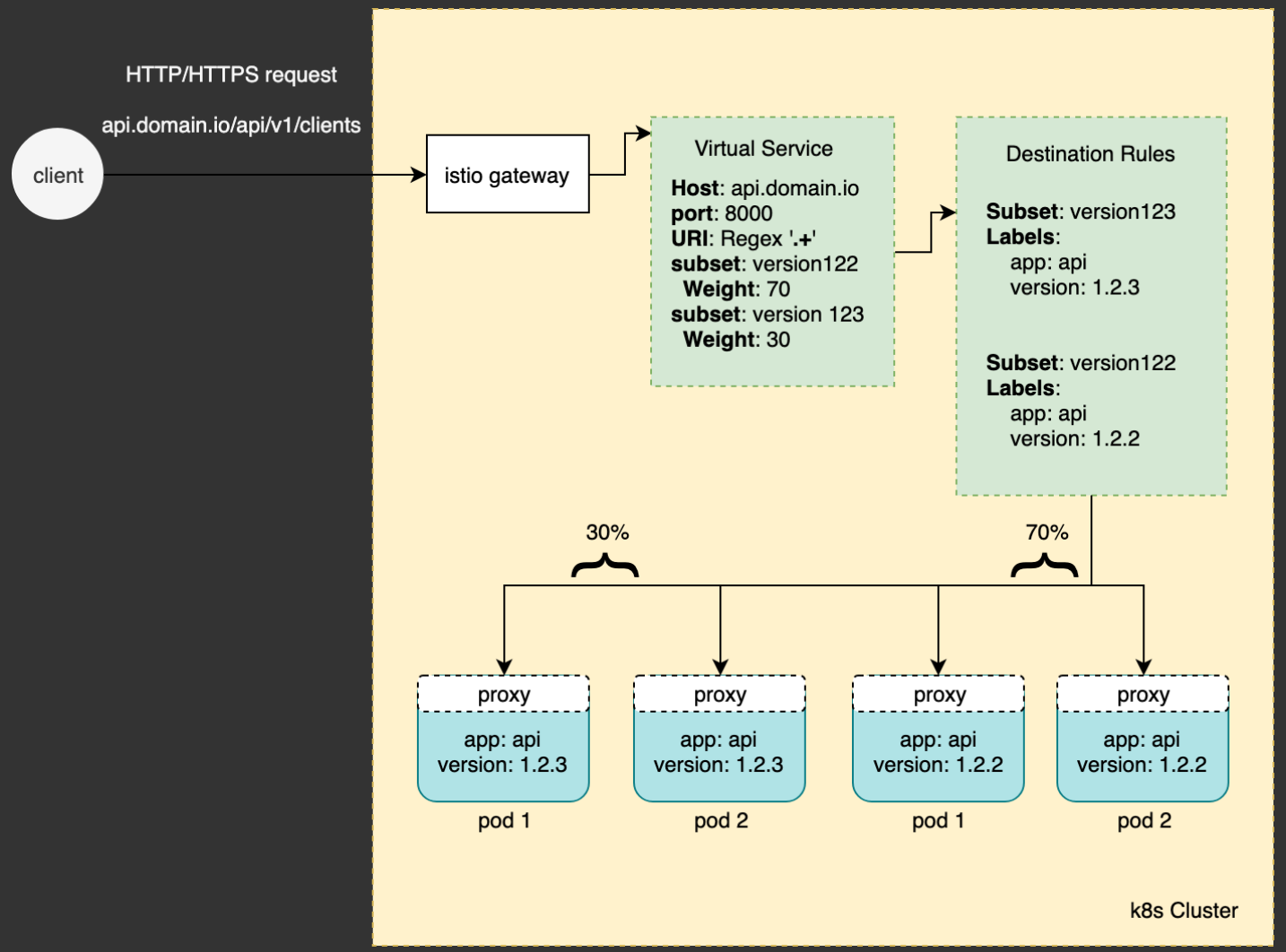
Set new traffic based on headers


HTTP/HTTPS request
api.domain.io/api/v1/clients
client
istio gateway
blue client
HTTP/HTTPS request
api.domain.io/api/v1/clients
-H 'x-version: 1.2.3'
-H 'account: 4'
Virtual Service
Host: api.domain.io
port: 8000
Headers:
x-version: 1.2.3
account: 4
subset: version123
Host: api.domain.io
port: 8000
URI: Regex '.+'
subset: version122
Destination Rules
Subset: version123
Labels:
app: api
version: 1.2.3
Subset: version122
Labels:
app: api
version: 1.2.2
proxy
app: api
version: 1.2.3
pod 1
proxy
app: api
version: 1.2.3
pod 2
proxy
app: api
version: 1.2.2
pod 1
proxy
app: api
version: 1.2.2
pod 2
k8s Cluster

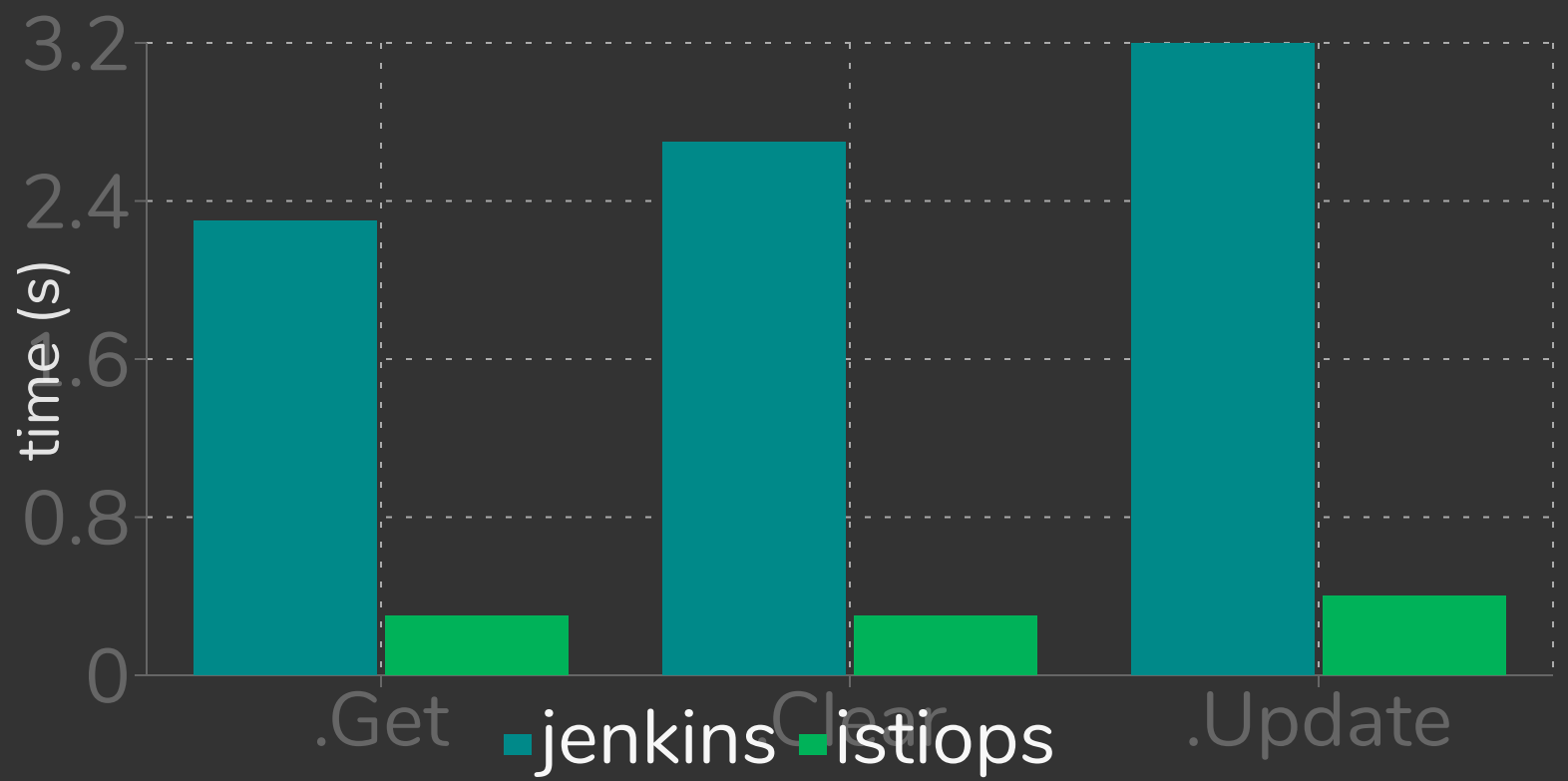
weight

```
$ istiops traffic shift \
    --build 12 \
--label-selector environment=pipeline-go \
--weight 30 \
--destination api-domain:8000 \
--pod-selector app=api,version=1.2.3
```

Set new traffic based on weight


HTTP/HTTPS request
api.domain.io/api/v1/clients
client
istio gateway
Virtual Service
Host: api.domain.io
port: 8000
URI: Regex '.+'
subset: version122
Weight: 70
subset: version 123
Weight: 30
Destination Rules
Subset: version123
Labels:
app: api
version: 1.2.3
Subset: version122
Labels:
app: api
version: 1.2.2
30%
70%
proxy
app: api
version: 1.2.3
pod 1
proxy
app: api
version: 1.2.3
pod 2
proxy
app: api
version: 1.2.2
pod 1
proxy
app: api
version: 1.2.2
pod 2
k8s Cluster

Shared Libraries x istiops



Importing as a pkg

./examples/main.go

pkg/operator/operator.go

```
1. type Operator interface {
2.    Get(selector map[string]string) (router.IstioRouteList, error)
3.    Update(shift router.Shift) error
4.    Clear(shift router.Shift) error
5. }
6.
```

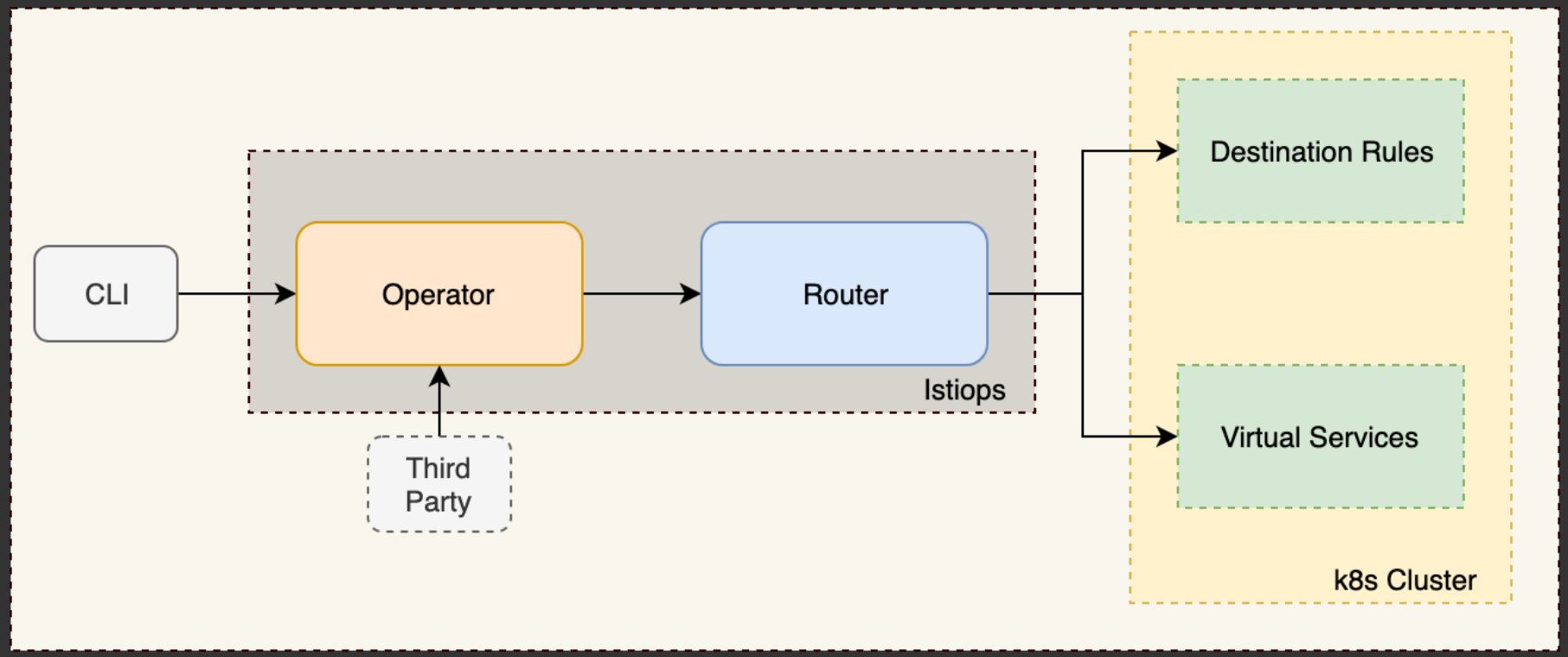
pkg/operator/istiops.go

```
type Istiops struct {
   DrRouter Router
   VsRouter Router
}

```

pkg/operator/istiops.go

```
1. type Router interface {
2.    Create(shift router.Shift) (*router.IstioRules, error)
3.    Validate(shift router.Shift) error
4.    Update(shift router.Shift) error
5.    Clear(shift router.Shift) error
6.    List(selector map[string]string) (*router.IstioRouteList, error)
7. }
8.
```


Destination Rules
CLI
Operator
Router
Istiops
Third
Party
Virtual Services
k8s Cluster

main.go

```go
package main

import (
  "github.com/google/uuid"
  "github.com/pismo/istiops/pkg/client"
  "github.com/pismo/istiops/pkg/operator"
  "github.com/pismo/istiops/pkg/router"
  "k8s.io/client-go/util/homedir"
)

func main() {
  // generate random uuid
  uuid, err := uuid.NewUUID()
  if err != nil {
        panic(err.Error())
  }

  // get istio client based on ~/kube/config
  kubeConfigPath := homedir.HomeDir() + "/kube/config"
  clients, err := client.New(kubeConfigPath)
  if err != nil {
        panic(err.Error())
  }

  trackingID := uuid.NewUUID().String()
  metadataName := "api-xpto"
  metadataNamespace := "default"
  build := 27

  var drr operator.Router
  drr = &router.DestinationRule{
        TrackingId: trackingID,
        Name:       metadataName,
        Namespace:  metadataNamespace,
        Build:      build,
        Istio:      clients.Istio,
  }

  var vsr operator.Router
  vsr = &router.VirtualService{
        TrackingId: trackingID,
        Name:       metadataName,
        Namespace:  metadataNamespace,
        Build:      build,
        Istio:      clients.Istio,
  }

  shift := router.Shift{
        Port:     5000,
        Hostname: "api.domain.io",
        Selector: map[string]string{"environment": "pipeline-go"},
        Traffic: router.Traffic{
              PodSelector: map[string]string{
                    "app":     "api",
```


Live Demo?
I also like to live dangerously

That's all, folks!

https://github.com/pismo/istiops [project]

https://vsantos.github.io/gophercon [presentation]

https://pismo.io/ [We are hiring 👀]