Migrating Transactions Worth Billions of 💰 to Service Mesh With No Downtime

gojek

KubeCon | CloudNativeCon

Europe 2020

Virtual

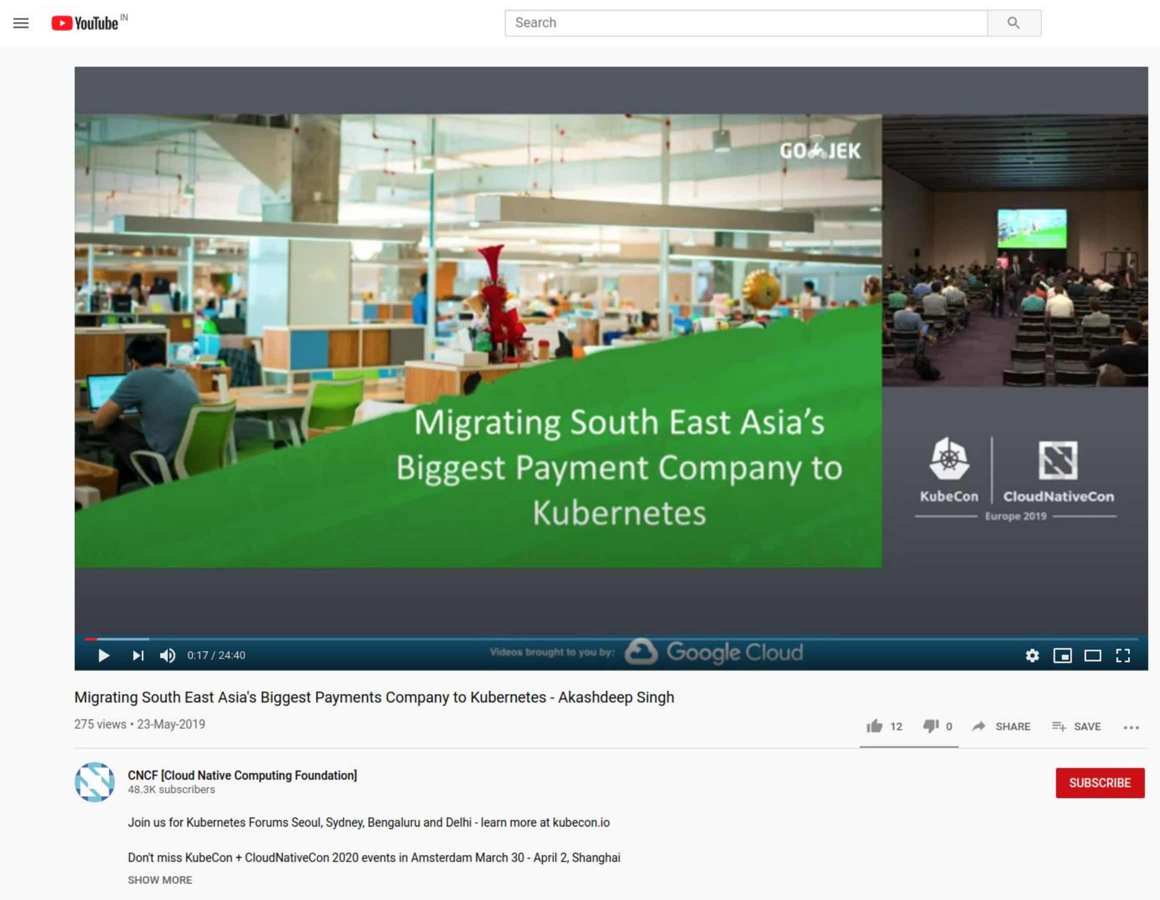
Mahendra Kariya

🐦/@mahendrakariya

Shishir Joshi

🐦/@shishir127

gojek


YouTube IN
Search
GO-JEK
Migrating South East Asia's Biggest Payment Company to Kubernetes
KubeCon
CloudNativeCon
Europe 2019
0:17 / 24:40
Videos brought to you by: Google Cloud
12
0
SHARE
SAVE
SUBSCRIBE

Migrating South East Asia's Biggest Payments Company to Kubernetes - Akashdeep Singh

275 views • 23-May-2019

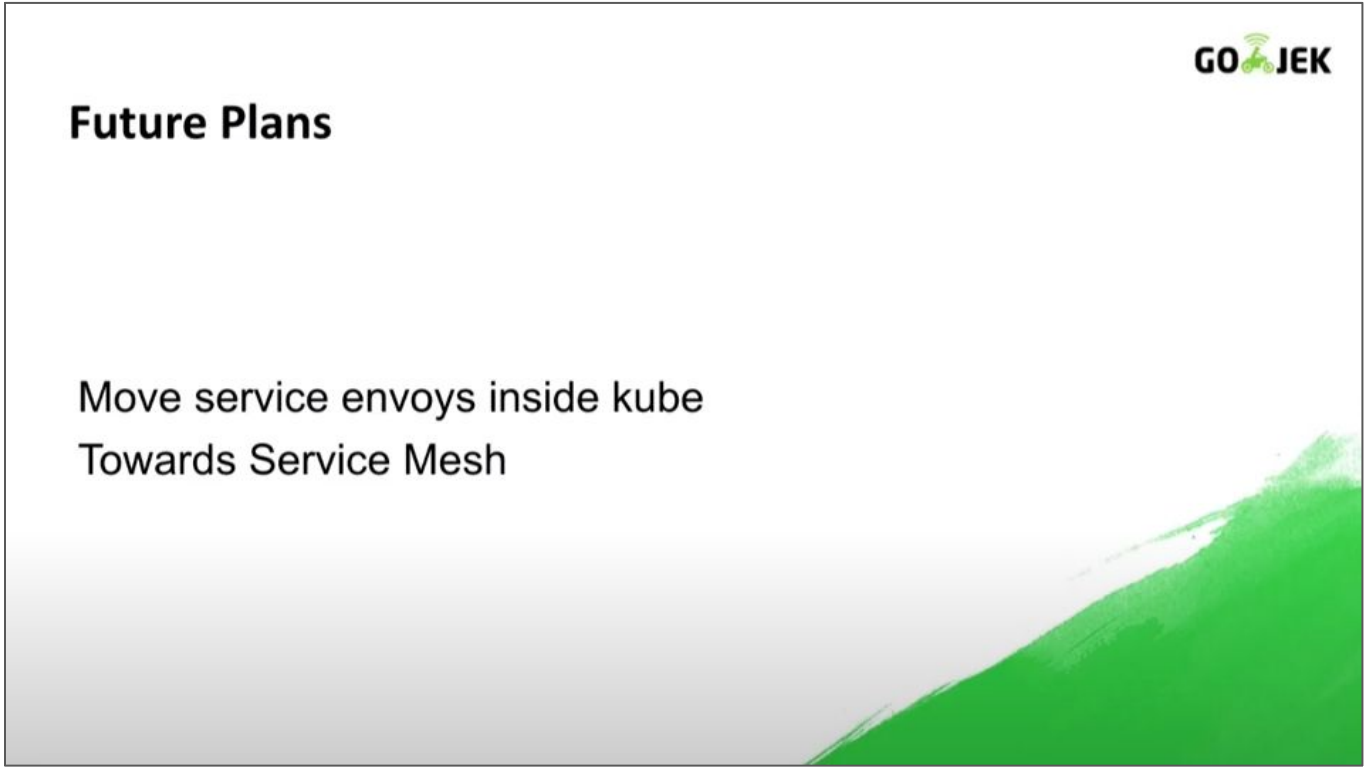
CNCF [Cloud Native Computing Foundation]
48.3K subscribers

Join us for Kubernetes Forums Seoul, Sydney, Bengaluru and Delhi - learn more at kubecon.io

Don't miss KubeCon + CloudNativeCon 2020 events in Amsterdam March 30 - April 2, Shanghai

SHOW MORE

https://www.youtube.com/watch?v=eYb--4iOSCY



GO-JEK

Future Plans

Move service envoys inside kube

Towards Service Mesh

https://www.youtube.com/watch?v=eYb--4iOSCY gopay

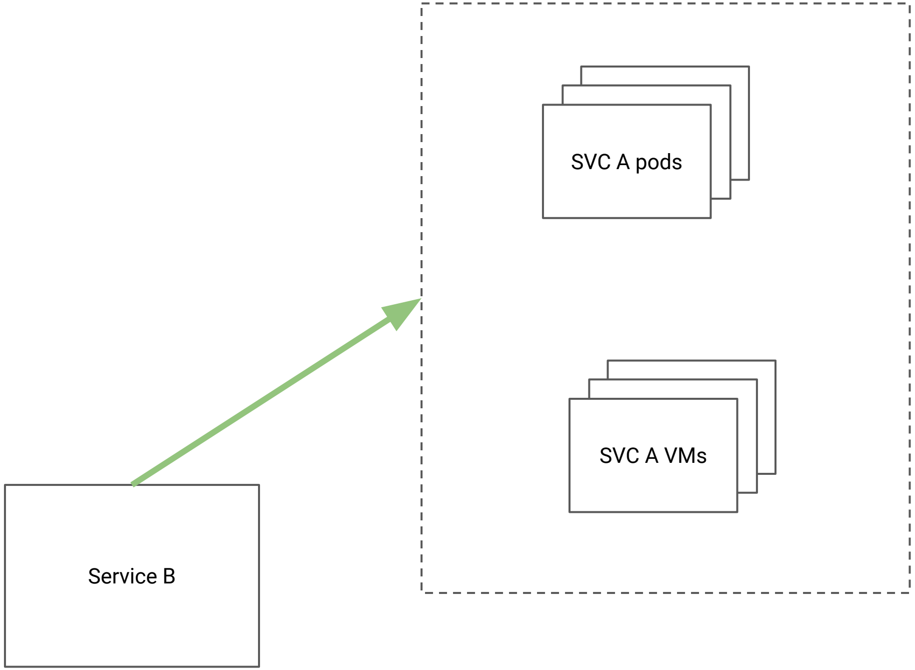
About gopay

- Leading digital payments provider in Indonesia
- Has largest MAU in Indonesia since Q4 2017
- Processed $7.8 billion in transactions in 2019
- Accepted at 300,000+ online and offline merchants
- Has integrations with 28+ financial institutions
- 100M+ transactions every month
- First e-money payment option on Google Play Store in Indonesia

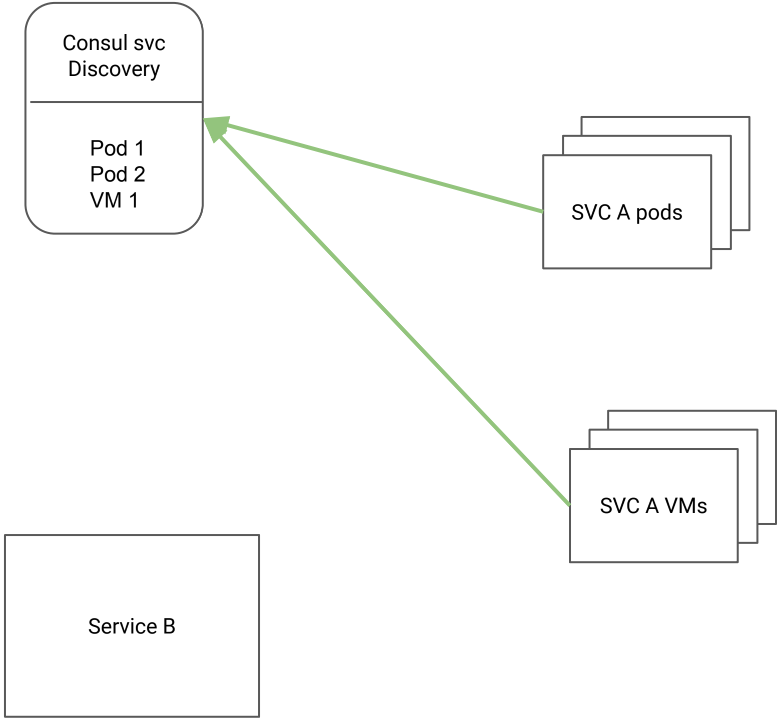
About gopay

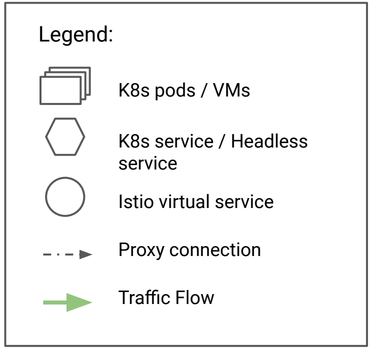
- A few hundred developers
- Multiple Kubernetes Clusters
- 150+ microservices
- 130M+ internal API calls
- 100+ deployments every week
- REST as well as gRPC services
- Services written in Golang, Java, Clojure, Ruby

Before introducing the service mesh

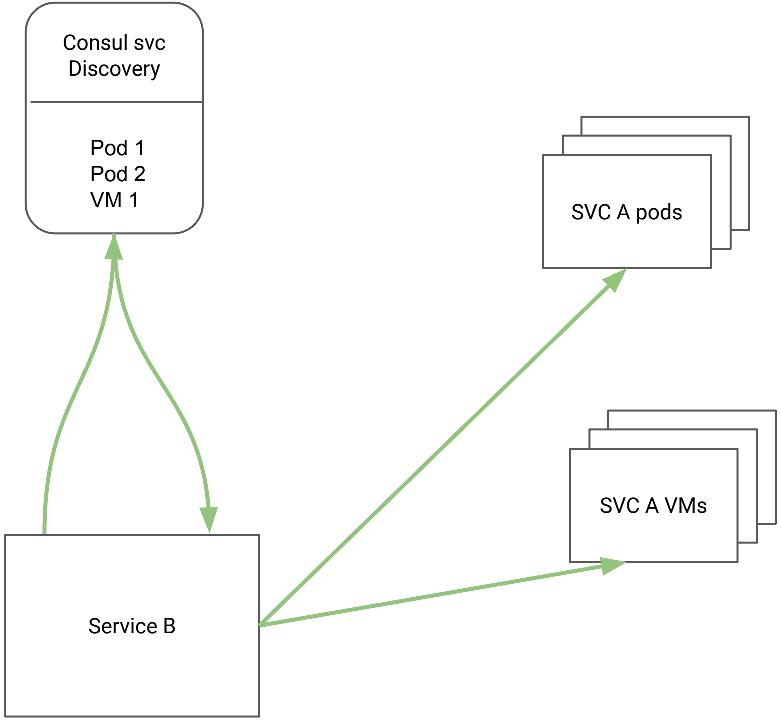

SVC A pods
SVC A VMs
Service B

Service discovery using Consul





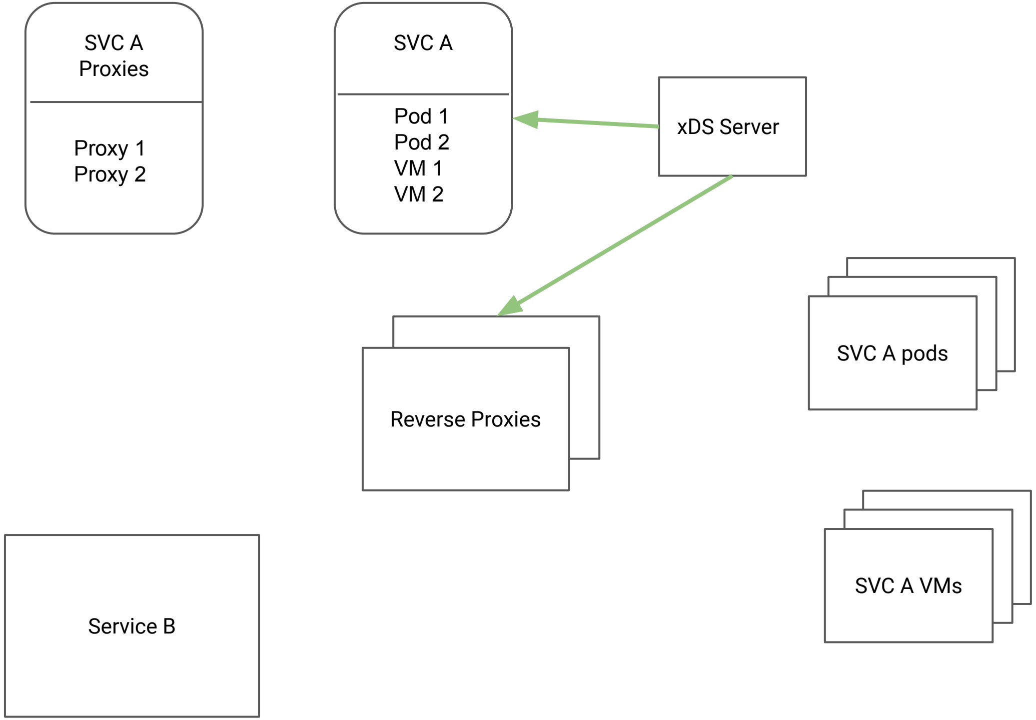
Service discovery using Consul

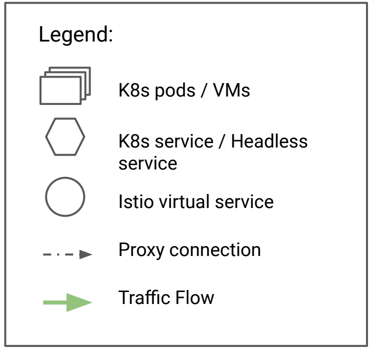


Legend:

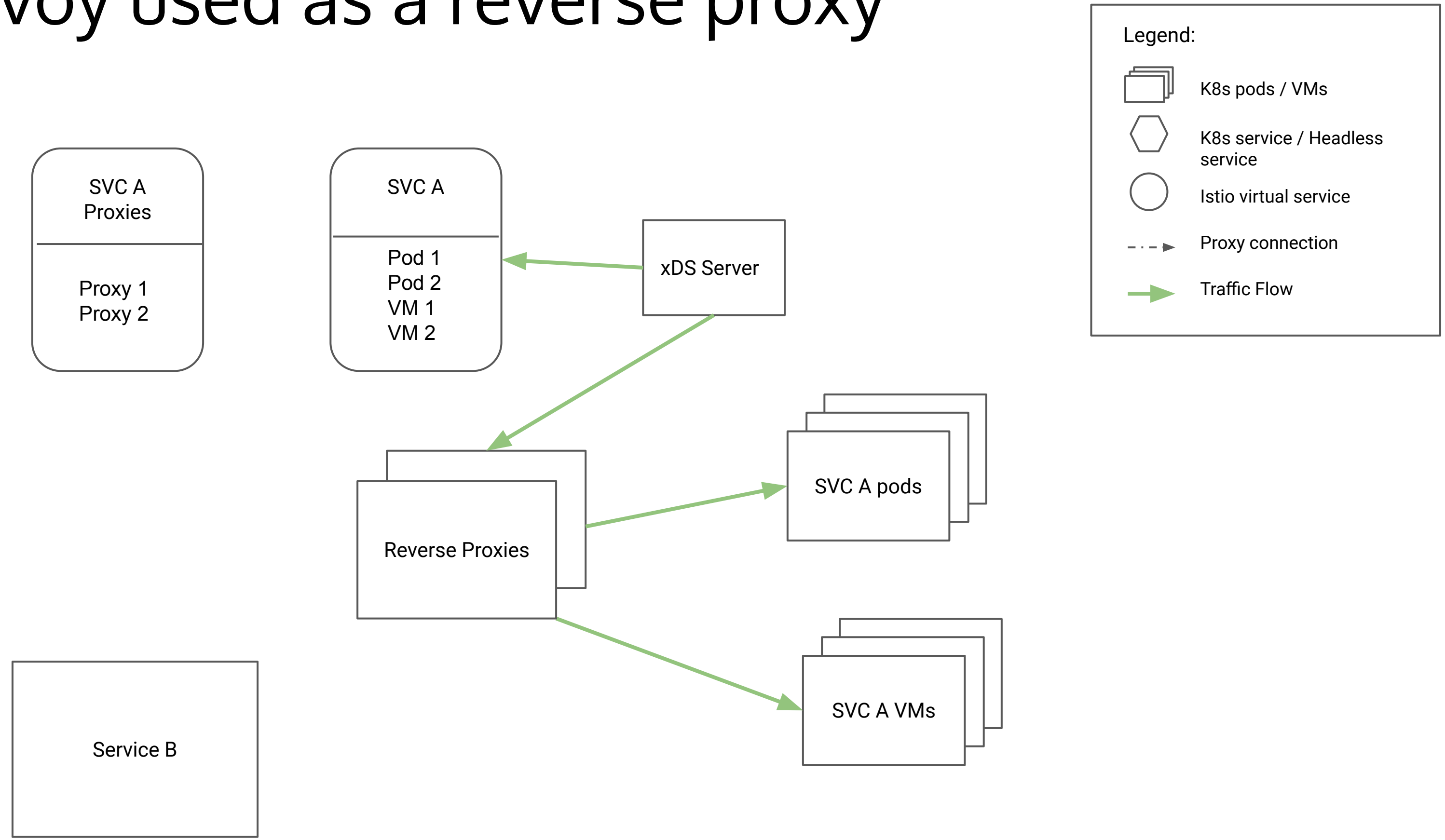
- ❐ K8s pods / VMs
- ⬡ K8s service / Headless service
- ○ Istio virtual service
- -·-▸ Proxy connection
- ➡ Traffic Flow

With Envoy used as a reverse proxy





With Envoy used as a reverse proxy



SVC A Proxies

Proxy 1
Proxy 2

SVC A

Pod 1
Pod 2
VM 1
VM 2

xDS Server

Reverse Proxies

SVC A pods

SVC A VMs

Service B

Legend:

K8s pods / VMs

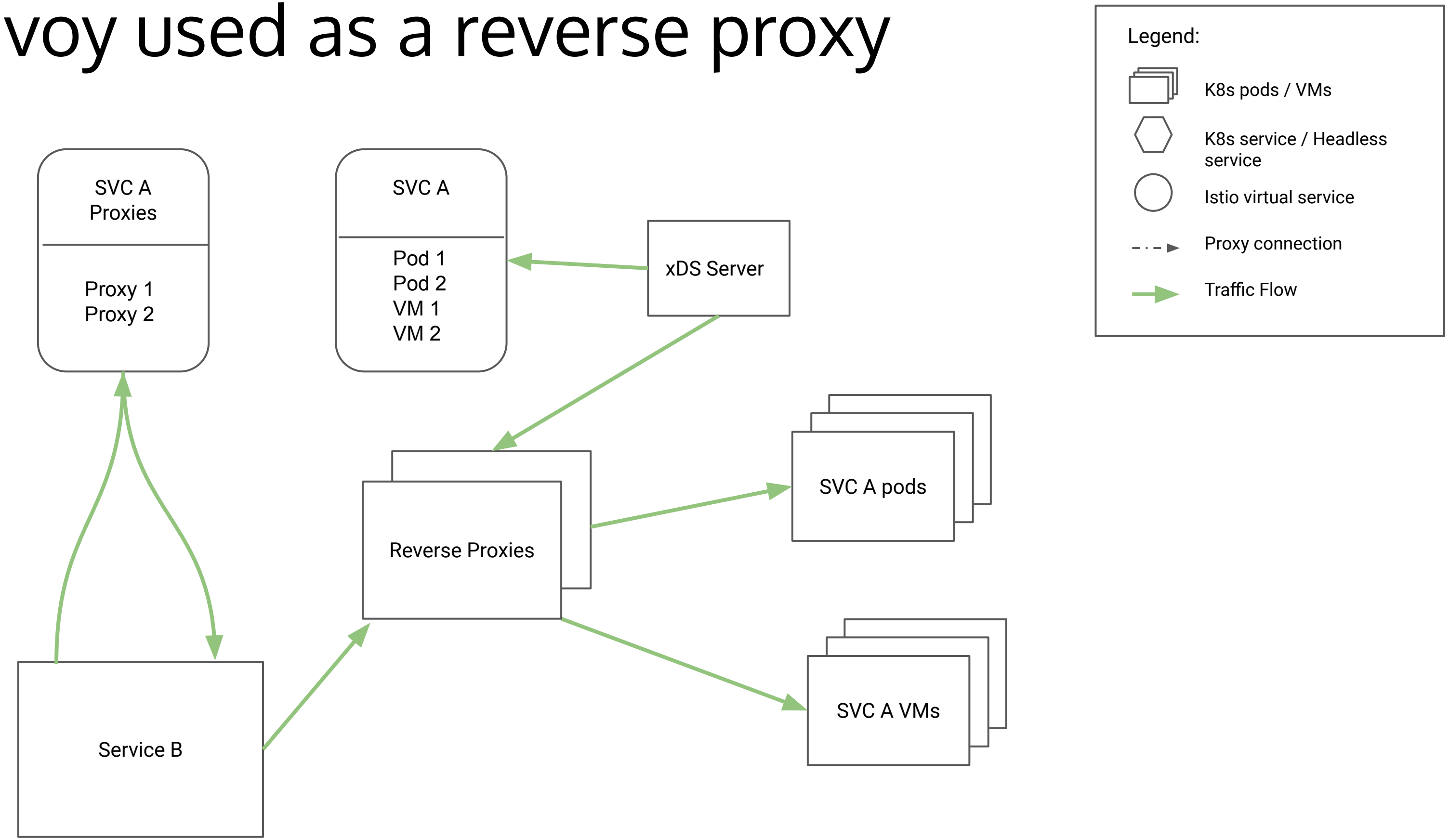
⬡ K8s service / Headless service

○ Istio virtual service

- · - ▸ Proxy connection

➞ Traffic Flow

With Envoy used as a reverse proxy



Challenges with this setup

- Fronting Envoy needs to be deployed for every service

Challenges with this setup

- Fronting Envoy needs to be deployed for every service
- Fronting Envoy versions drifted over time

Challenges with this setup

- Fronting Envoy needs to be deployed for every service
- Fronting Envoy versions drifted over time
- Latency in syncing Envoy with Consul

Challenges with this setup

- Fronting Envoy needs to be deployed for every service
- Fronting Envoy versions drifted over time
- Latency in syncing Envoy with Consul
- Terminated pod didn’t get deleted from Consul

Challenges with this setup

- Fronting Envoy needs to be deployed for every service
- Fronting Envoy versions drifted over time
- Latency in syncing Envoy with Consul
- Terminated pod didn’t get deleted from Consul
- Overhead of ensuring client libraries are updated

Challenges with this setup

- Fronting Envoy needs to be deployed for every service
- Fronting Envoy versions drifted over time
- Latency in syncing Envoy with Consul
- Terminated pod didn’t get deleted from Consul
- Overhead of ensuring client libraries are updated
- Consul was no longer the single source of truth for service discovery

Challenges with this setup

- Fronting Envoy needs to be deployed for every service
- Fronting Envoy versions drifted over time
- Latency in syncing Envoy with Consul
- Terminated pod didn’t get deleted from Consul
- Overhead of ensuring client libraries are updated
- Consul was no longer the single source of truth for service discovery
- Concept of canary deployment was broken

Business Expansion Plans

- Different data localization and processing regulations in different countries

Business Expansion Plans

- Different data localization and processing regulations in different countries
- Same set of services and fronting Envoys need to deployed across regions

Business Expansion Plans

- Different data localization and processing regulations in different countries
- Same set of services and fronting Envoys need to deployed across regions
- Cross region and cross DC traffic needs to be handled securely
    - Certificate management
    - mTLS
    - Ingress Gateways

Business Expansion Plans

- Different data localization and processing regulations in different countries
- Same set of services and fronting Envoys need to deployed across regions
- Cross region and cross DC traffic needs to be handled securely
  - Certificate management
  - mTLS
  - Ingress Gateways
- Rate limiting

Business Expansion Plans

- Different data localization and processing regulations in different countries
- Same set of services and fronting Envoys need to deployed across regions
- Cross region and cross DC traffic needs to be handled securely
  - Certificate management
  - mTLS
  - Ingress Gateways
- Rate limiting
- Quota management

What does service mesh solve?

- Handles the client side load balancing and service discovery

What does service mesh solve?

- Handles the client side load balancing and service discovery
- Deprecate Consul for service discovery

What does service mesh solve?

- Handles the client side load balancing and service discovery
- Deprecate Consul for service discovery
- Better telemetry

What does service mesh solve?

- Handles the client side load balancing and service discovery
- Deprecate Consul for service discovery
- Better telemetry
- Provides better traffic splitting abilities

What does service mesh solve?

- Handles the client side load balancing and service discovery
- Deprecate Consul for service discovery
- Better telemetry
- Provides better traffic splitting abilities
- Eliminates Envoy fronting version drifts

What does service mesh solve?

- Handles the client side load balancing and service discovery
- Deprecate Consul for service discovery
- Better telemetry
- Provides better traffic splitting abilities
- Eliminates Envoy fronting version drifts
- Rate limiting, distributed tracing, transparent mTLS, etc.

What does service mesh solve?

- Handles the client side load balancing and service discovery
- Deprecate Consul for service discovery
- Better telemetry
- Provides better traffic splitting abilities
- Eliminates Envoy fronting version drifts
- Rate limiting, distributed tracing, transparent mTLS, etc.
- Reduces overhead and cost of infrastructure management

Recap

- Client libraries with service discovery and load balancing.
- Keeping client libraries and Envoy updated is tedious. Faced issues because of bugs in older versions.
- Setup needs to be replicated across regions. Infra needs to be minimalistic.
- Service mesh solves these issues and provides more. Can build better tooling on top.

What service mesh should we choose?

Istio

Why Istio?

- We wanted an Envoy based service mesh, because of our prior experience with Envoy

Why Istio?

- We wanted an Envoy based service mesh, because of our prior experience with Envoy

- We had Envoy filters which we ideally wanted to avoid porting

Why Istio?

- We wanted an Envoy based service mesh, because of our prior experience with Envoy
- We had Envoy filters which we ideally wanted to avoid porting
- Didn’t want to hand roll the control plane

Why Istio?

- We wanted an Envoy based service mesh, because of our prior experience with Envoy
- We had Envoy filters which we ideally wanted to avoid porting
- Didn’t want to hand roll the control plane
- First class support for Envoy filters

Why Istio?

- We wanted an Envoy based service mesh, because of our prior experience with Envoy
- We had Envoy filters which we ideally wanted to avoid porting
- Didn’t want to hand roll the control plane
- First class support for Envoy filters
- Features of a service mesh we were interested in were best supported by Istio e.g. policy management

Requirements for introducing service mesh

- Support seamless traffic flow
  - Within the mesh
  - From inside the mesh to outside world
  - From outside world to inside the mesh

Requirements for introducing service mesh

- Support seamless traffic flow
  - Within the mesh
  - From inside the mesh to outside world
  - From outside world to inside the mesh
- Transparency for callee services calling services migrated to Istio

Requirements for introducing service mesh

- Support seamless traffic flow
    - Within the mesh
    - From inside the mesh to outside world
    - From outside world to inside the mesh
- Transparency for callee services calling services migrated to Istio
- Support for staggered migration to Istio

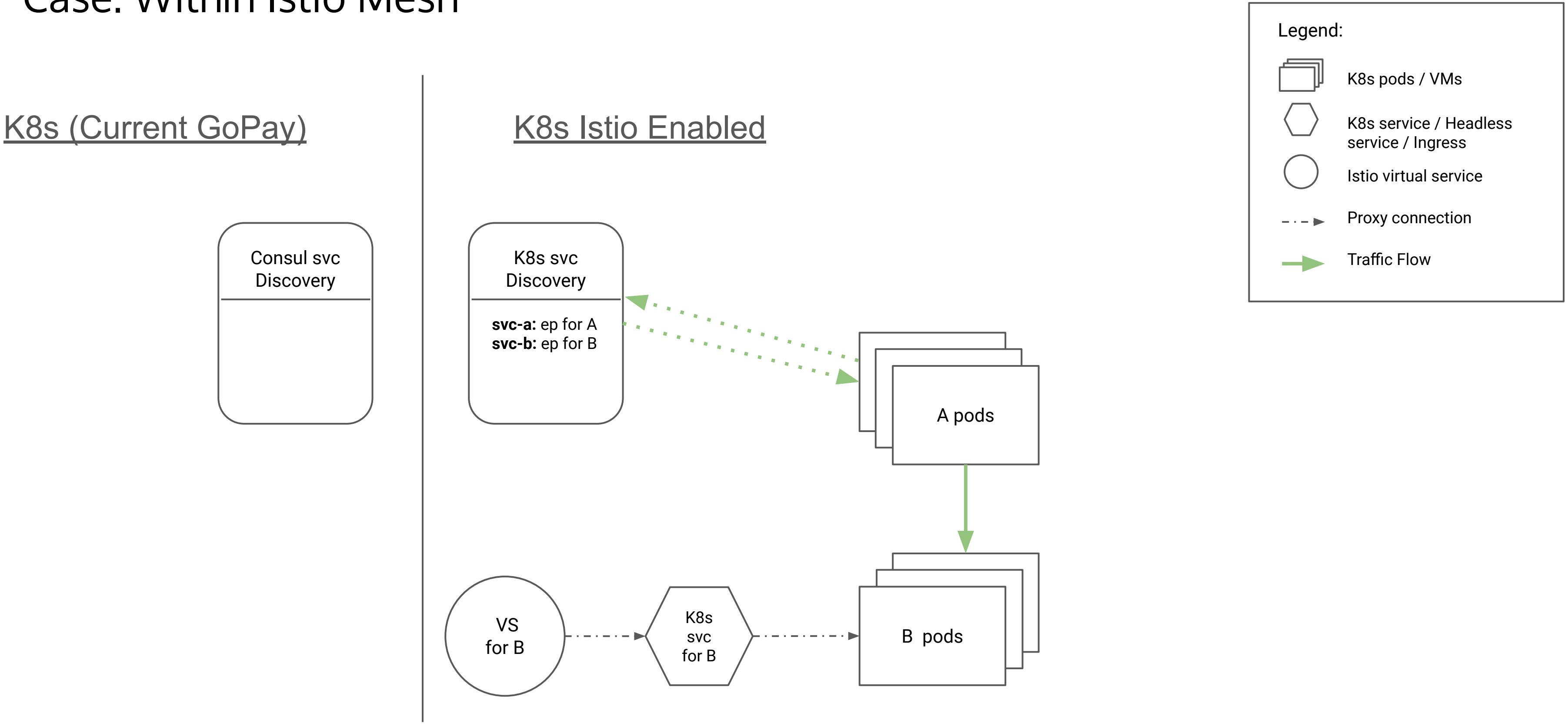
Requirements for introducing service mesh

- Support seamless traffic flow
  - Within the mesh
  - From inside the mesh to outside world
  - From outside world to inside the mesh
- Transparency for callee services calling services migrated to Istio
- Support for staggered migration to Istio
- Robust rollback strategy in case of any failures

Requirements for introducing service mesh

- Support seamless traffic flow
  - Within the mesh
  - From inside the mesh to outside world
  - From outside world to inside the mesh
- Transparency for callee services calling services migrated to Istio
- Support for staggered migration to Istio
- Robust rollback strategy in case of any failures
- Detect any possible performance issues with Istio for our use-case while minimizing impact

Case: Within Istio Mesh



K8s (Current GoPay)

K8s Istio Enabled

Consul svc Discovery

K8s svc Discovery

svc-a: ep for A
svc-b: ep for B

A pods

VS for B

K8s svc for B

B pods

Legend:

K8s pods / VMs

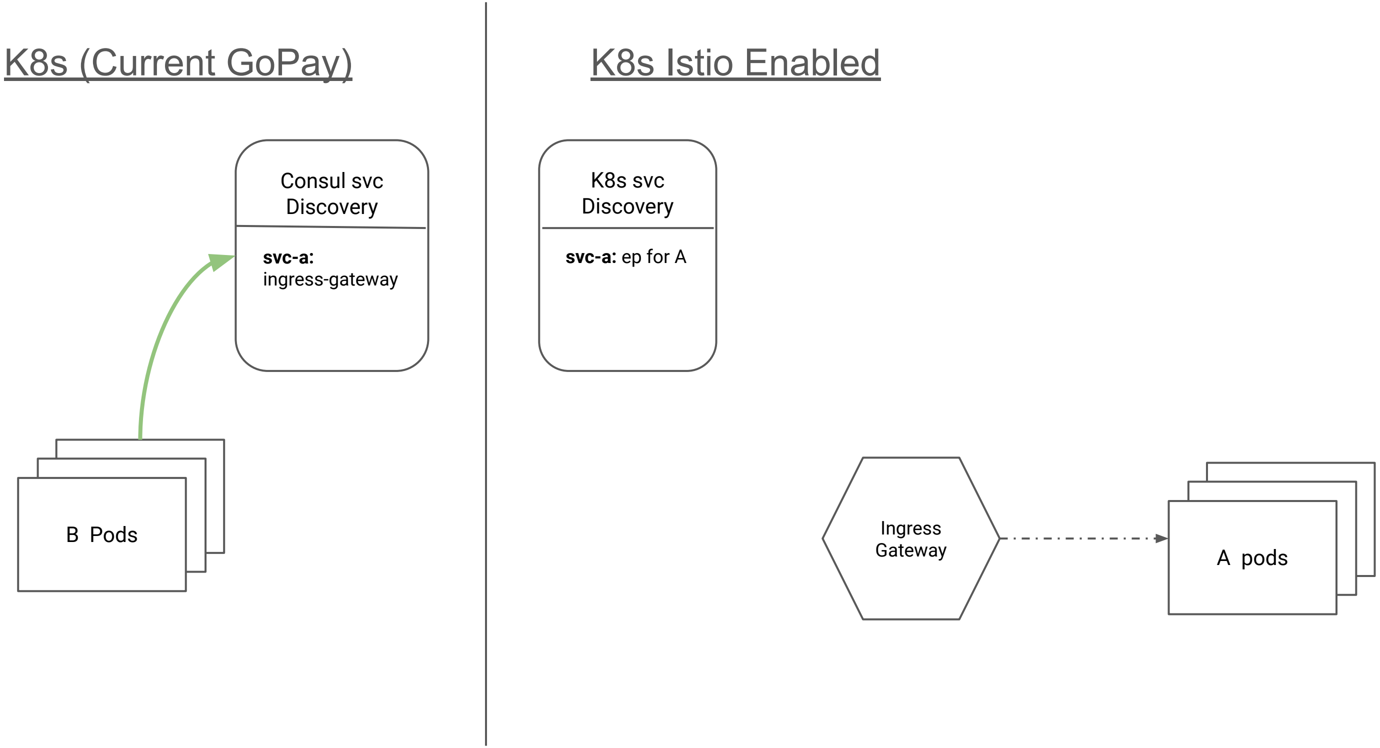
K8s service / Headless service / Ingress

Istio virtual service

Proxy connection

Traffic Flow

Case: From non-Istio Env to Istio Mesh



K8s (Current GoPay)

Consul svc Discovery

svc-a: ingress-gateway

B Pods

K8s Istio Enabled

K8s svc Discovery

svc-a: ep for A

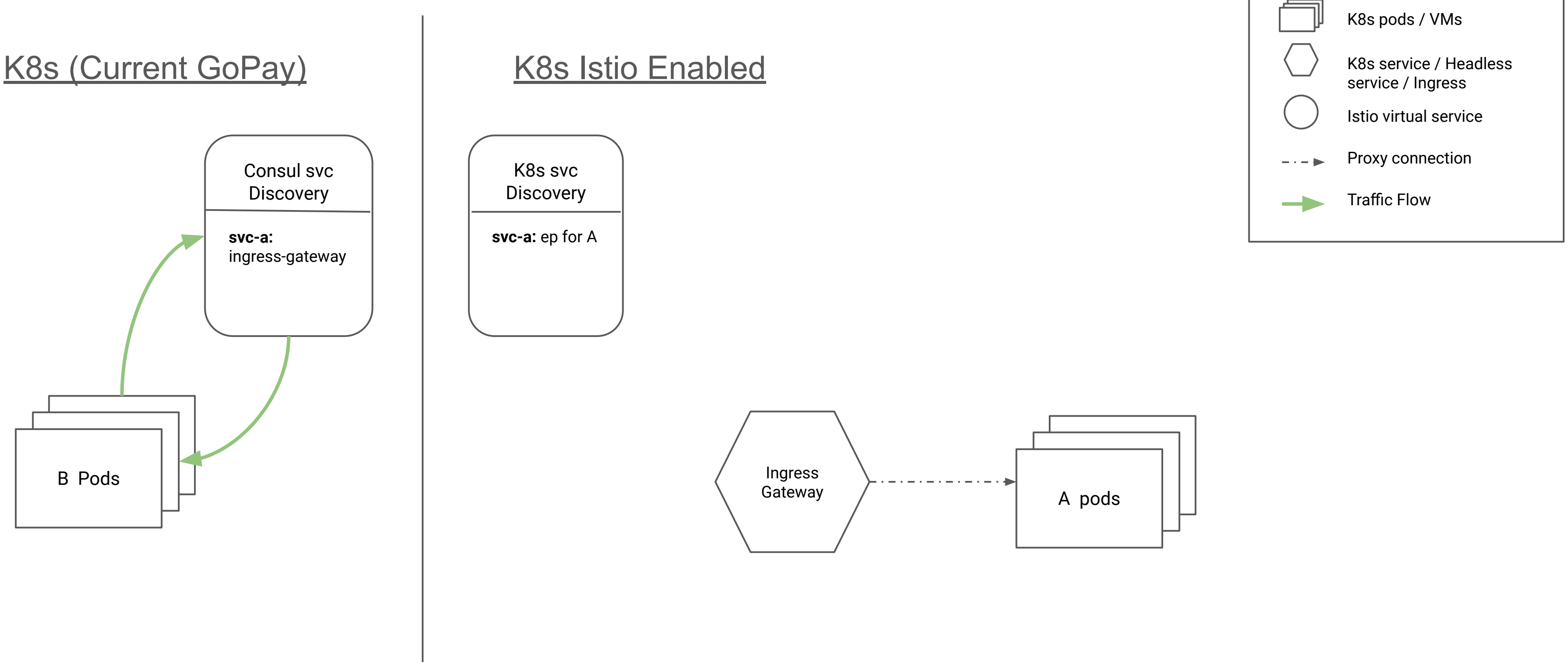
Ingress Gateway

A pods

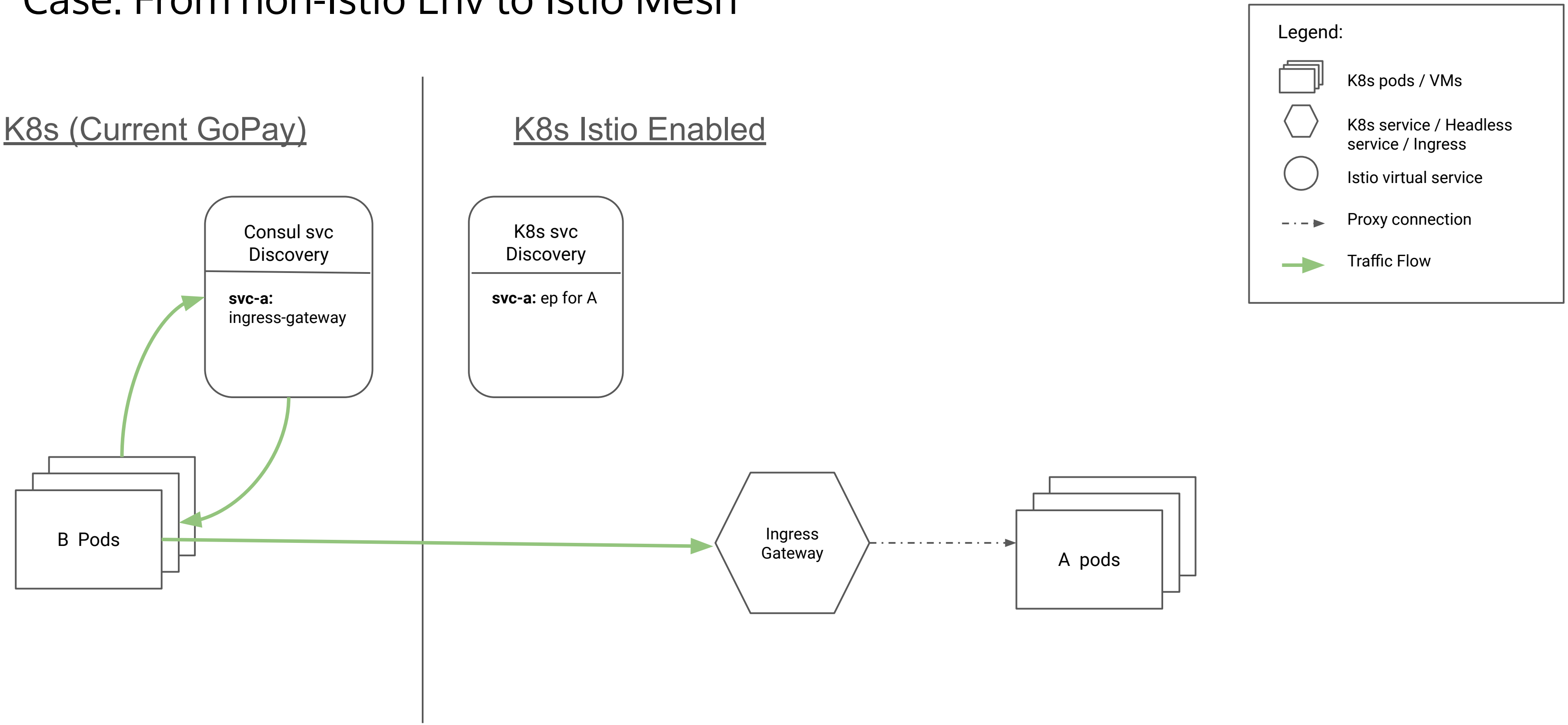
Legend:

- ▭ K8s pods / VMs
- ⬡ K8s service / Headless service / Ingress
- ○ Istio virtual service
- -·-▸ Proxy connection
- → Traffic Flow

Case: From non-Istio Env to Istio Mesh



Case: From non-Istio Env to Istio Mesh



K8s (Current GoPay)

K8s Istio Enabled

Consul svc Discovery

svc-a: ingress-gateway

K8s svc Discovery

svc-a: ep for A

B Pods

Ingress Gateway

A pods

Legend:

K8s pods / VMs

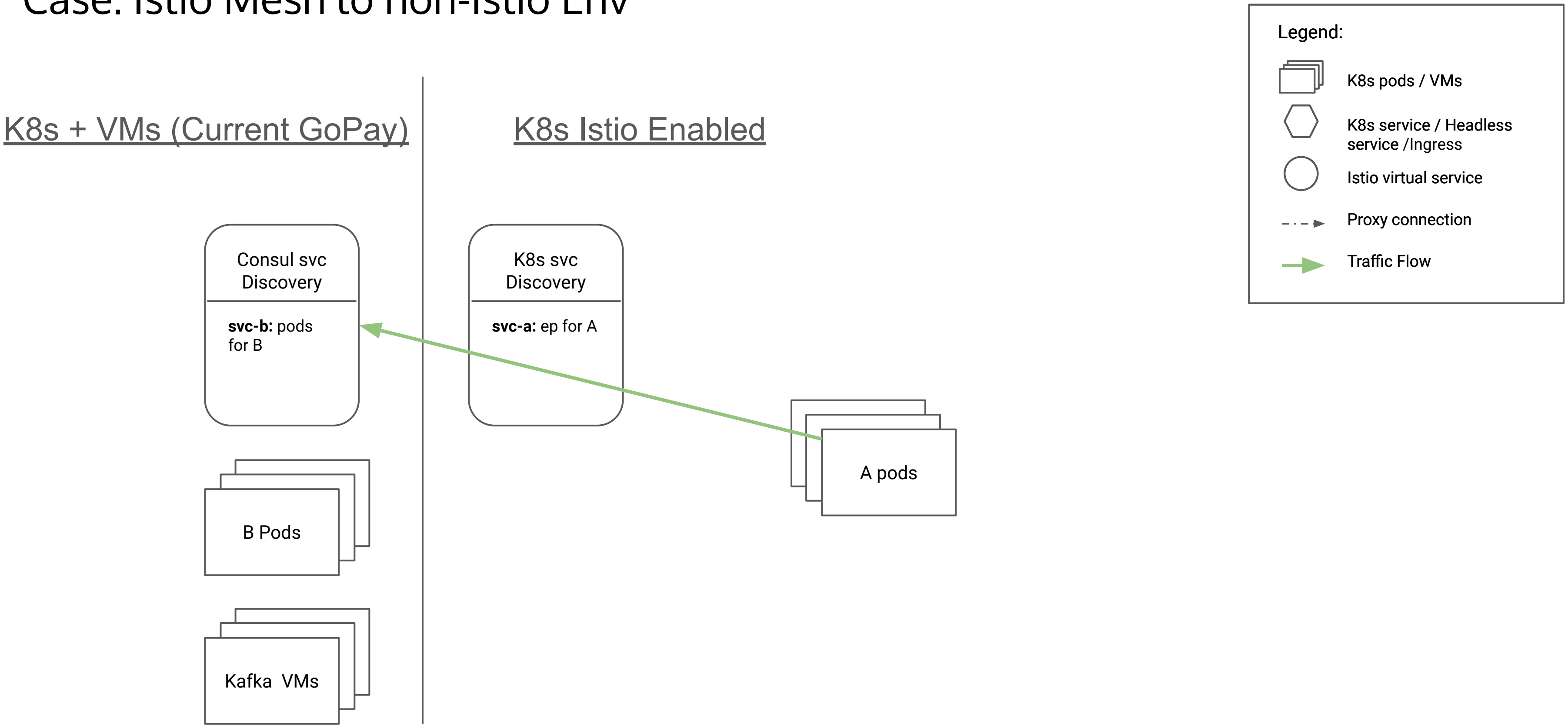
K8s service / Headless service / Ingress

Istio virtual service

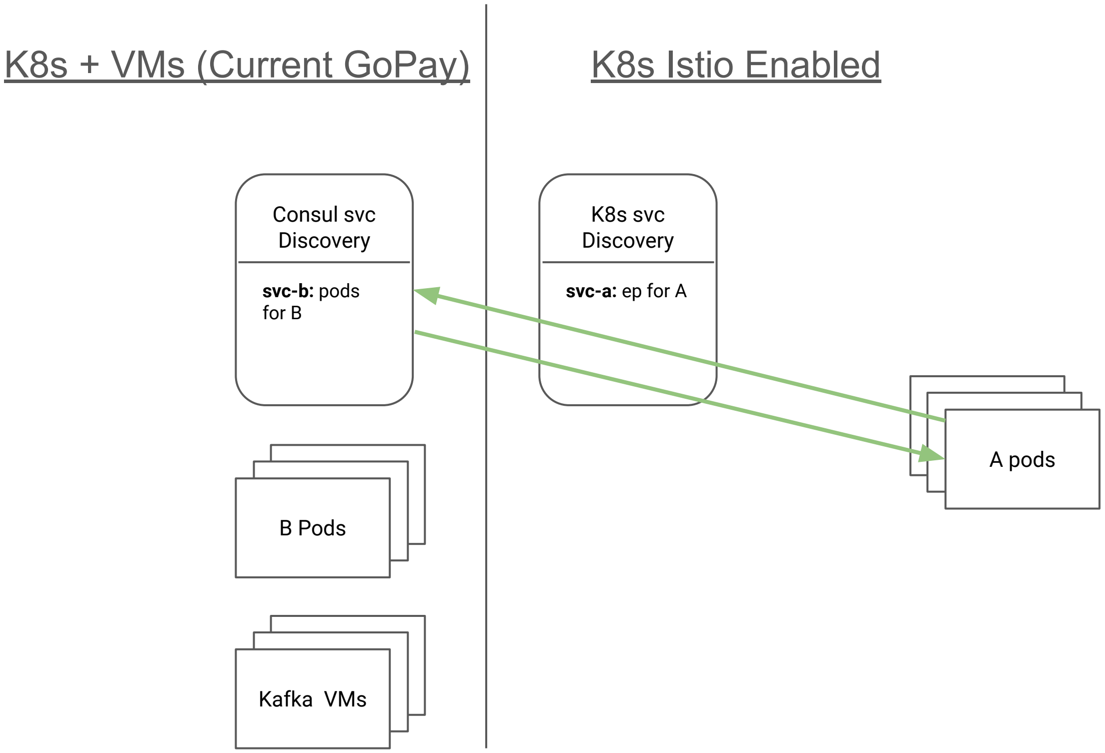
Proxy connection

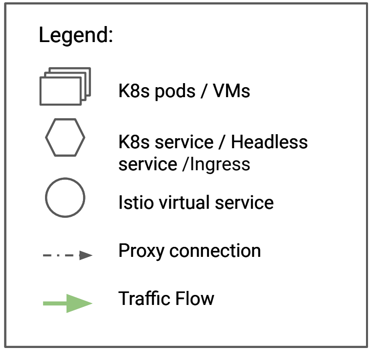
Traffic Flow

Case: Istio Mesh to non-Istio Env

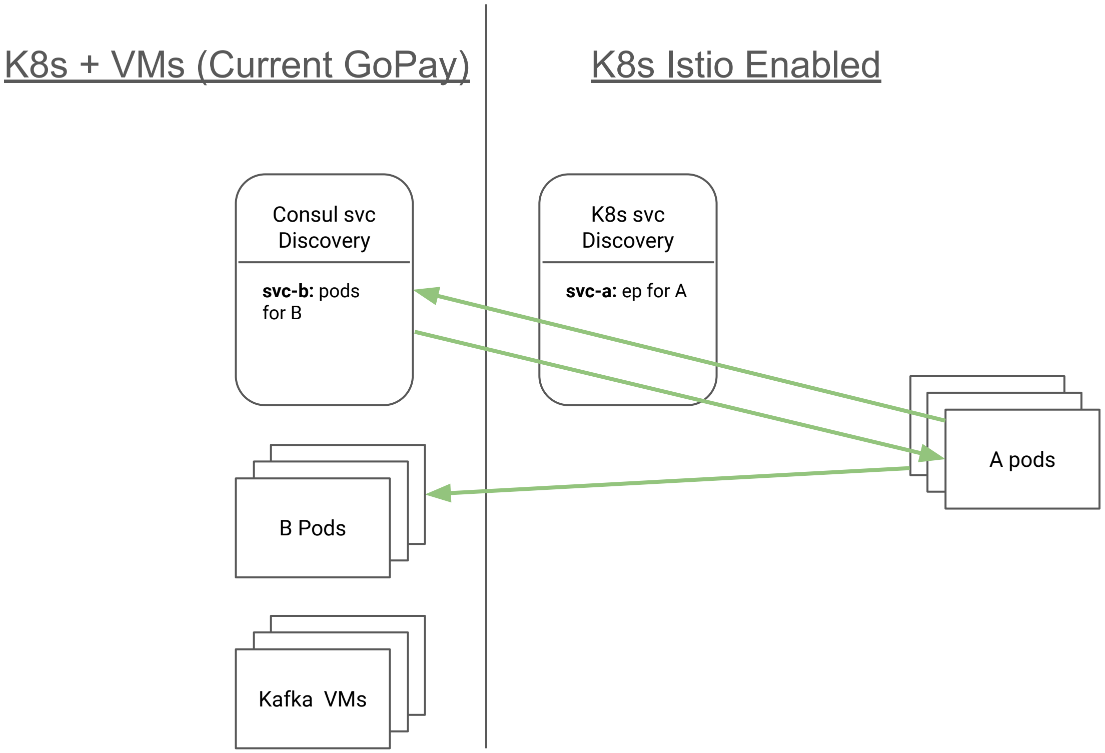


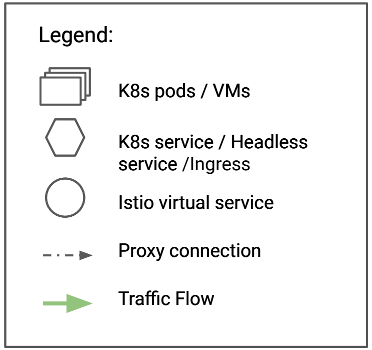
Case: Istio Mesh to non-Istio Env



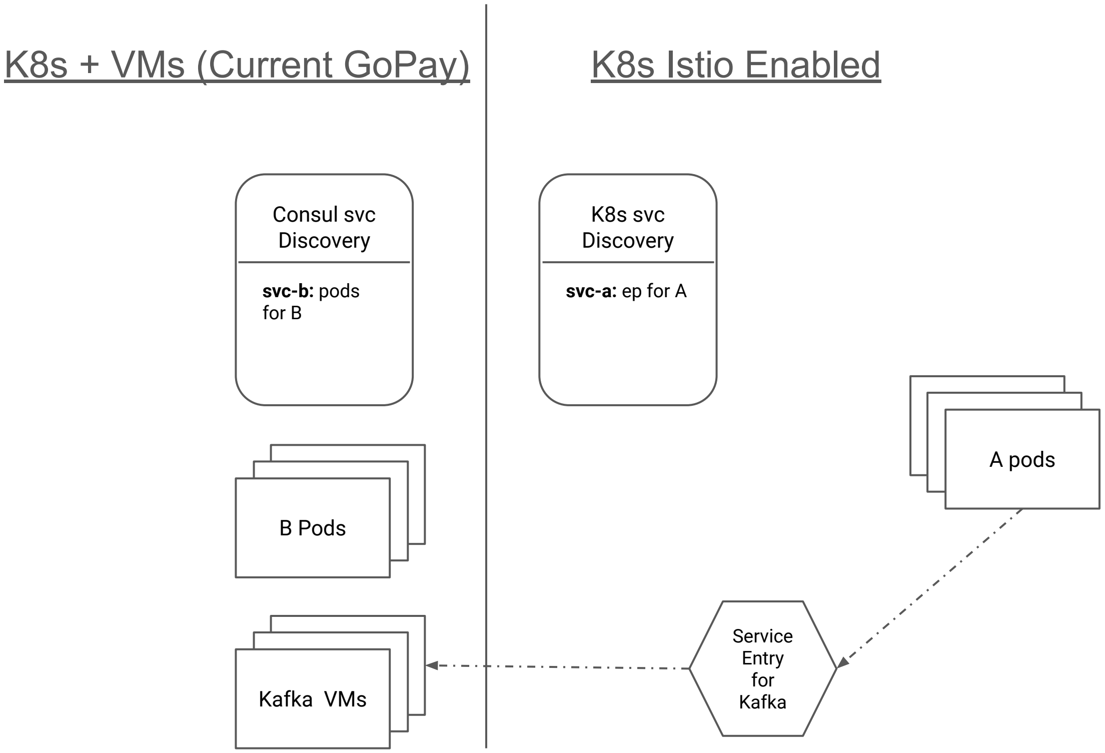


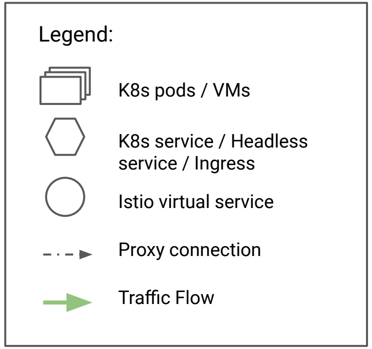
Case: Istio Mesh to non-Istio Env



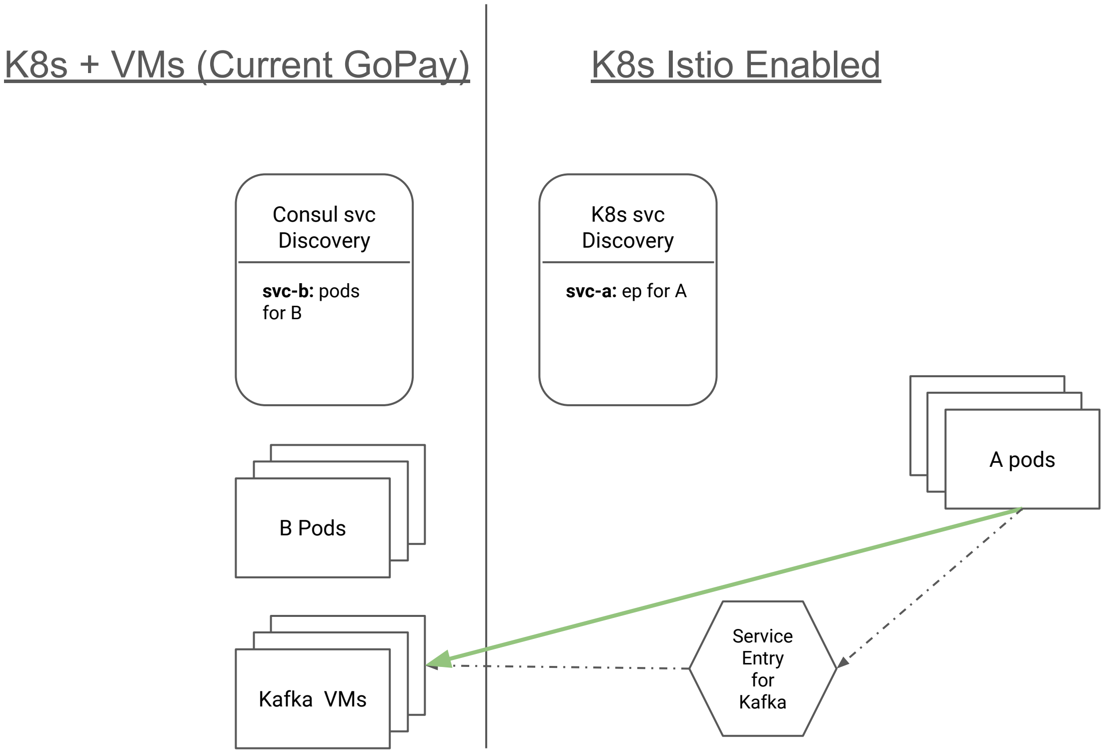


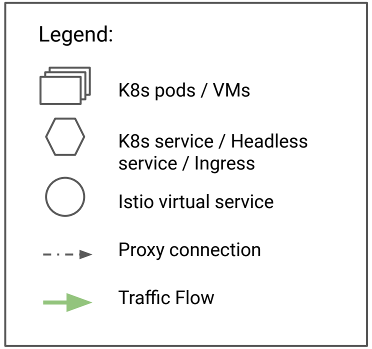
Case: Istio Mesh to non-Istio Env





Case: Istio Mesh to non-Istio Env





Recap

- We chose Istio.
- We wanted seamless rollout and rollback and a staggered rollout option.
- Primarily had 3 cases to handle during rollout
    - Within the mesh
    - From inside the mesh to outside world
    - From outside world to inside the mesh
- We used existing service discovery where possible with Istio to make the rollout seamless .

How we rolled out?

How we rolled out?

1

- Supported just a few basic functionalities of Istio
- Core team initially migrated a few critical services along with devs
- Ironed out any issues based on feedback from devs

How we rolled out?

1

- Supported just a few basic functionalities of Istio
- Core team initially migrated a few critical services along with devs
- Ironed out any issues based on feedback from devs

2

- Created robust documentation around how migrate
- Established SOPs and processes
- Started letting devs do the migration, with active support from core team

How we rolled out?

1

- Supported just a few basic functionalities of Istio
- Core team initially migrated a few critical services along with devs
- Ironed out any issues based on feedback from devs

2

- Created robust documentation around how migrate
- Established SOPs and processes
- Started letting devs do the migration, with active support from core team

3

- Migration in autopilot mode
- Started enabling more Istio features (like mTLS, rate limiting, etc.)

Automation + Documentation FTW!

- Created Helm charts with support for Istio resources

Automation + Documentation FTW!

- Created Helm charts with support for Istio resources
- Added support for rolling back to non-Istio environment in the Helm charts

Automation + Documentation FTW!

- Created Helm charts with support for Istio resources
- Added support for rolling back to non-Istio environment in the Helm charts
- Automated validation of Istio resources configuration in CI pipelines

Automation + Documentation FTW!

- Created Helm charts with support for Istio resources
- Added support for rolling back to non-Istio environment in the Helm charts
- Automated validation of Istio resources configuration in CI pipelines

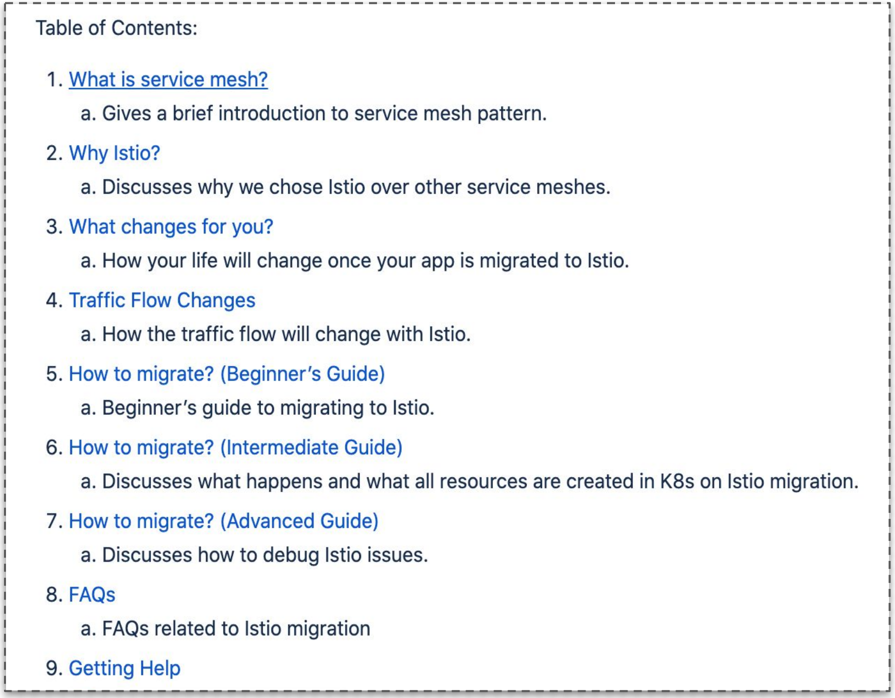


Table of Contents:

1. What is service mesh?
    a. Gives a brief introduction to service mesh pattern.
2. Why Istio?
    a. Discusses why we chose Istio over other service meshes.
3. What changes for you?
    a. How your life will change once your app is migrated to Istio.
4. Traffic Flow Changes
    a. How the traffic flow will change with Istio.
5. How to migrate? (Beginner’s Guide)
    a. Beginner’s guide to migrating to Istio.
6. How to migrate? (Intermediate Guide)
    a. Discusses what happens and what all resources are created in K8s on Istio migration.
7. How to migrate? (Advanced Guide)
    a. Discusses how to debug Istio issues.
8. FAQs
    a. FAQs related to Istio migration
9. Getting Help

Monitoring and Alerting

- Internal Prometheus + Grafana based setup for monitoring and alerting

Monitoring and Alerting

- Internal Prometheus + Grafana based setup for monitoring and alerting
- Cortex from long term metrics storage and horizontal scalability

Monitoring and Alerting

- Internal Prometheus + Grafana based setup for monitoring and alerting
- Cortex from long term metrics storage and horizontal scalability
- Separate dashboards for Istio control plane and data plane

Monitoring and Alerting

- Internal Prometheus + Grafana based setup for monitoring and alerting
- Cortex from long term metrics storage and horizontal scalability
- Separate dashboards for Istio control plane and data plane
- Default dashboard and alerts for any service migrated to Istio

Monitoring and Alerting

- Internal Prometheus + Grafana based setup for monitoring and alerting
- Cortex from long term metrics storage and horizontal scalability
- Separate dashboards for Istio control plane and data plane
- Default dashboard and alerts for any service migrated to Istio
- Service Graph visualization for services fully on Istio

Example Metrics

Control Plane

- xDS Latency
- xDS Error Rate
- Resource usage for control plane pods
- Cert related errors
- Number of out of sync sidecars
- Istio sidecar version drift

Example Metrics

Control Plane

- xDS Latency
- xDS Error Rate
- Resource usage for control plane pods
- Cert related errors
- Number of out of sync sidecars
- Istio sidecar version drift

Data Plane

“Golden Signals” of monitoring

- Latency
- Traffic
- Errors
- Saturation

Challenges Faced

DISCLAIMER - Some of these are specific to our environment and use-cases

- Getting devs comfortable with the new environment & new concepts

Challenges Faced

DISCLAIMER - Some of these are specific to our environment and use-cases

- Getting devs comfortable with the new environment & new concepts
- Confusion with Helm charts for Istio installation vs using istioctl

Challenges Faced

DISCLAIMER - Some of these are specific to our environment and use-cases

- Getting devs comfortable with the new environment & new concepts
- Confusion with Helm charts for Istio installation vs using istioctl
- Understanding service entries

Challenges Faced

DISCLAIMER - Some of these are specific to our environment and use-cases

- Getting devs comfortable with the new environment & new concepts
- Confusion with Helm charts for Istio installation vs using istioctl
- Understanding service entries
- Using Helm to deploy instead of generating templates

Recap

- 3 phases of rollout. Improvements in each phase targeted towards empowering a self-serve migration
- Used staggered migration to discover any issues or limitations
- Created documentation and a guide for migration
- Added automation for validations in CI pipelines

Current State of Rollout

1

- Supported just a few basic functionalities of Istio
- Core team initially migrated a few critical services along with devs
- Ironed out any issues based on feedback from devs

2

- Created robust documentation around how migrate
- Established SOPs and processes
- Started letting devs do the migration, with active support from core team

3

- Migration in autopilot mode
- Started enabling more Istio features (like mTLS, rate limiting, etc.)

Thank You Community!

Thank You Community!

Big Thanks to

- Neeraj Poddar
- Shriram Rajagoplan

Thank you

Mahendra Kariya

@mahendrakariya

Shishir Joshi

@shishir127

KubeCon | CloudNativeCon

Europe 2020

Virtual