Google が提供する フルマネージドのサービスメッシュ 「Anthos Service Mesh（ASM）」 とは

Google Cloud Japan
Application Modernization Specialist
内間 和季

アジェンダ

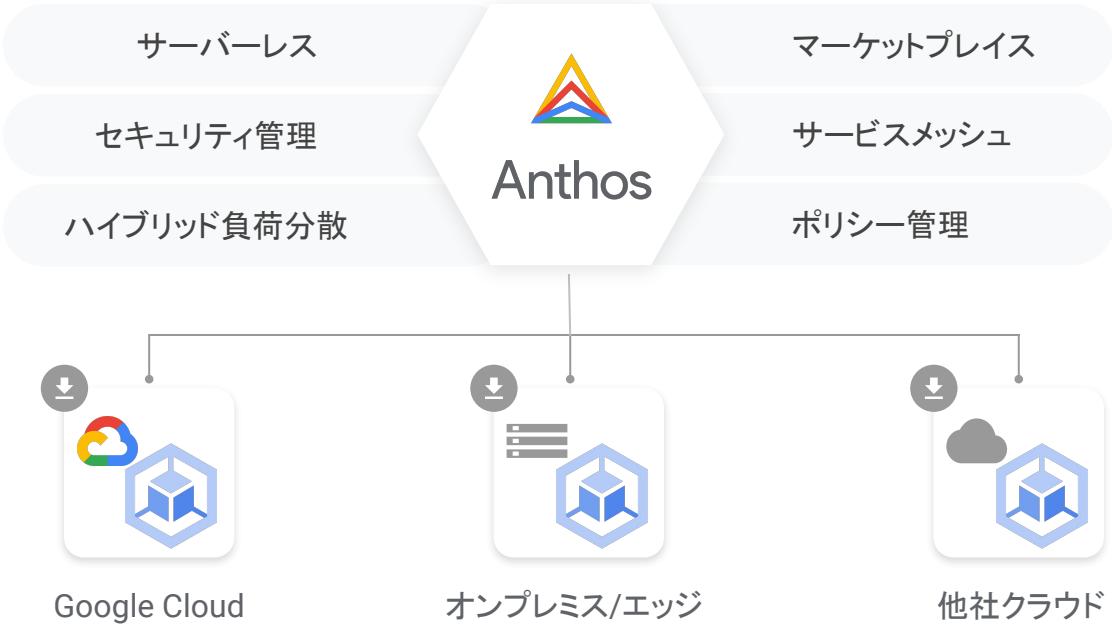
1. Anthos 概要
2. サービスメッシュが注目される背景
3. Anthos Service Mesh 概要
4. Anthos Service Mesh 機能紹介
5. まとめ

Anthos 概要

Google Cloud

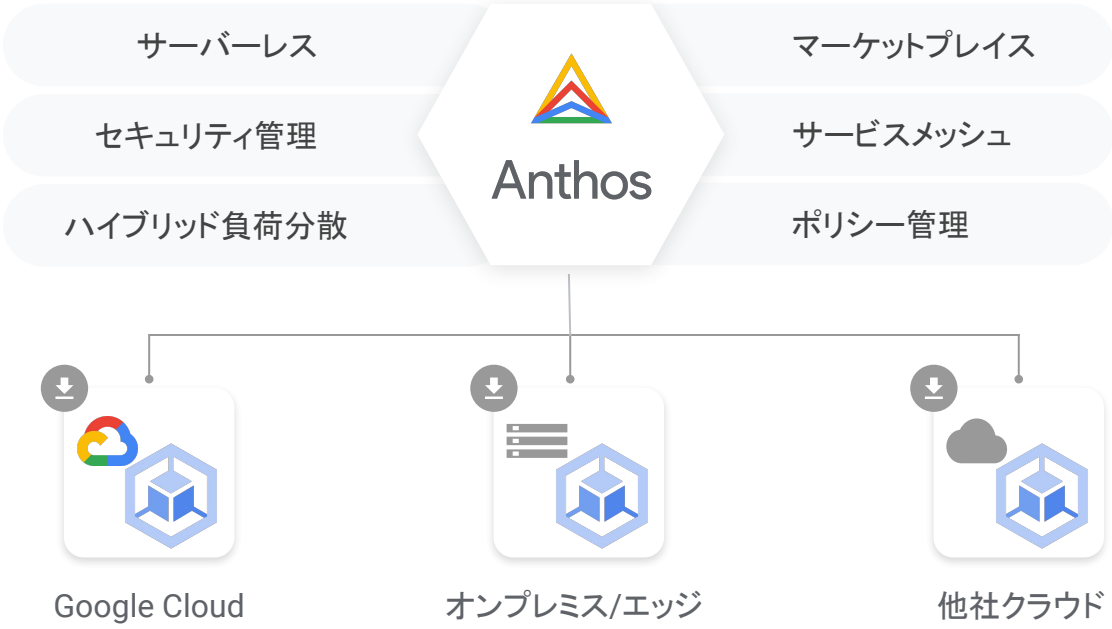
Anthos とは？

ハイブリッド / マルチクラウド 環境に配置した GKE をベースに以下の機能を提供



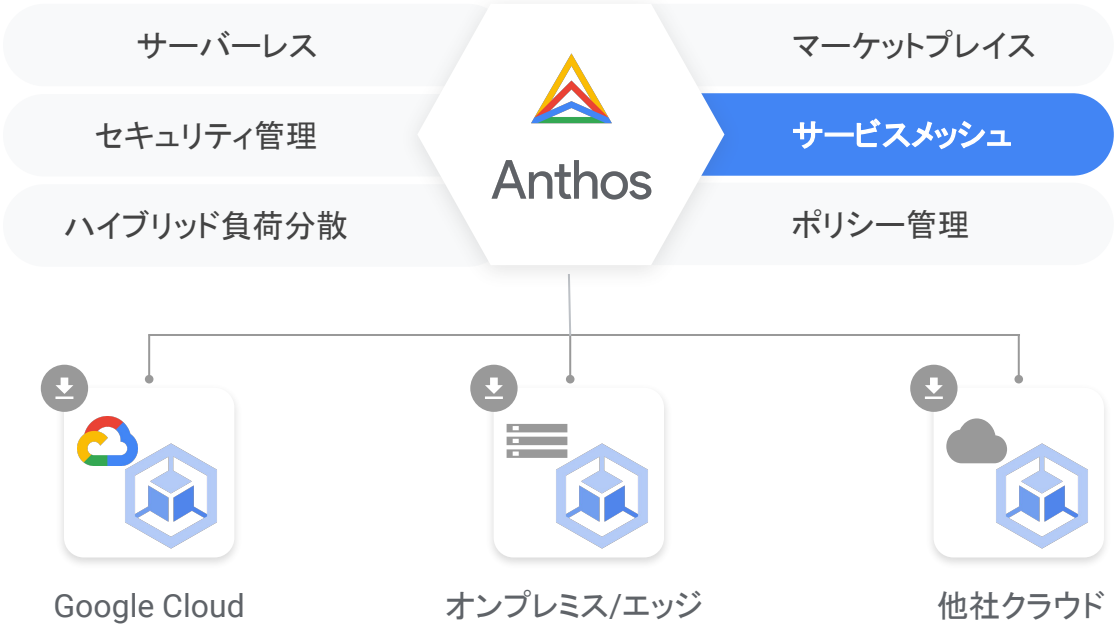
Anthos とは？

ハイブリッド / マルチクラウド 環境に配置した GKE をベースに以下の機能を提供



本セッションで紹介するサービス

本セッションでは、Anthos Service Mesh について説明します



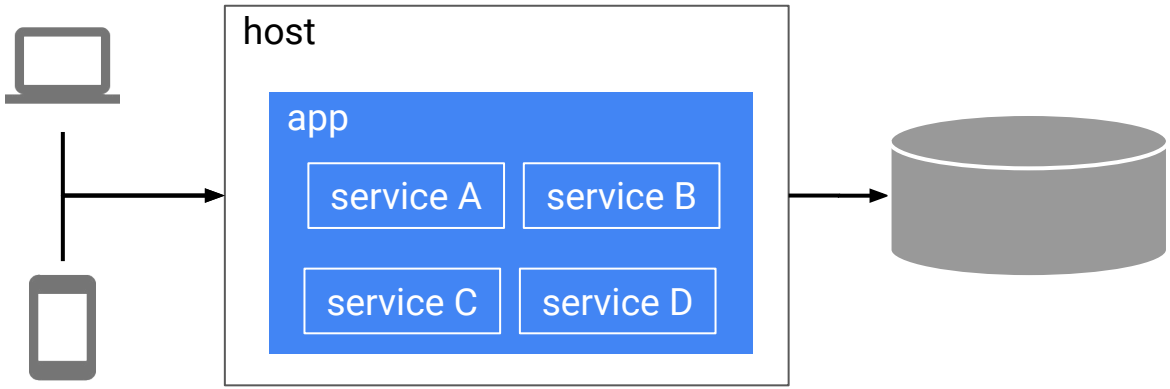
サービスメッシュが注目される背景

02

Google Cloud

モノリシックなアプリケーション例

- 複数の機能が全て 1 つのアプリケーションとして動作
- 大きなコードベース



モノリシックなアプリケーションへの機能追加が続くと

- 機能追加や障害時の影響分析が困難
- コードベースの複雑さ増（開発者のラーニングコスト増）
- ビルド、テストにかかる時間増
- 非効率なスケーリング

変更容易性が低くなり、

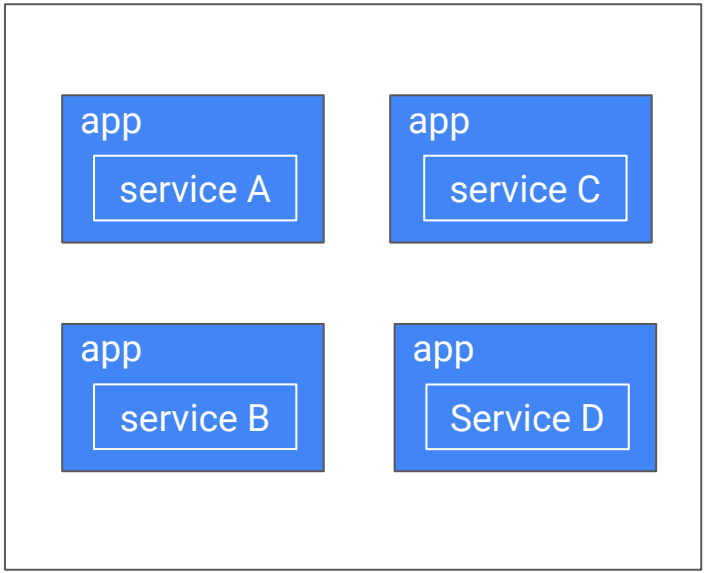
新機能をタイムリーにリリースできない（時間がかかる）

事業の成長に合わせてシステムがスケール出来ずボトルネックに

マイクロサービスとは

Thoughtworks 社の James Lewis 氏と Martin Fowler 氏が 2014 年に提唱

- 機能毎に独立したアプリケーション（サービス）に分割
- 各サービスは単一の目的をもつ
- 分散システム、サービス間は疎結合、
  軽量な API などでやり取り



マイクロサービス化により得るもの

1. 機能追加が容易に
2. 可用性・耐障害性向上
3. スケールアウトが容易に
4. 新技術の導入がしやすい
5. DevOps / Agile との親和性

新機能をタイムリーに高速にリリースすることが可能 に

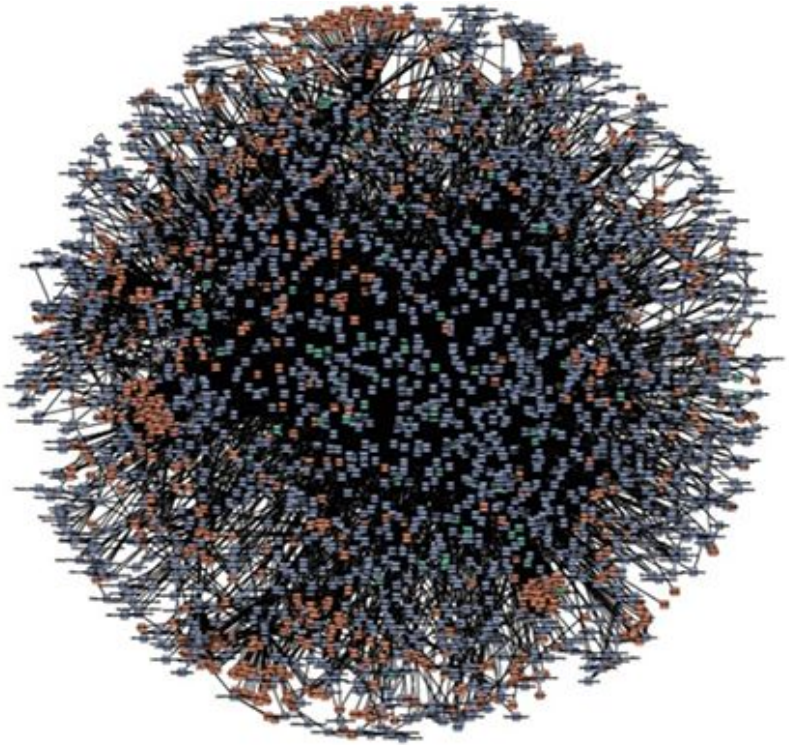
事業の成長に合わせシステムも柔軟にスケール

マイクロサービス化の技術的課題

1. サービスディスカバリー
2. トラフィックコントロール
3. Fault Isolation（障害の分離）
4. Observability（可観測性）
5. サービス間の認証・認可

マイクロサービスを実現するためには、
上記を解決するプラットフォームが必要となる

サービスメッシュ



サービスメッシュ

マイクロサービス環境において、

サービスディスカバリ、

トラフィックコントロール、

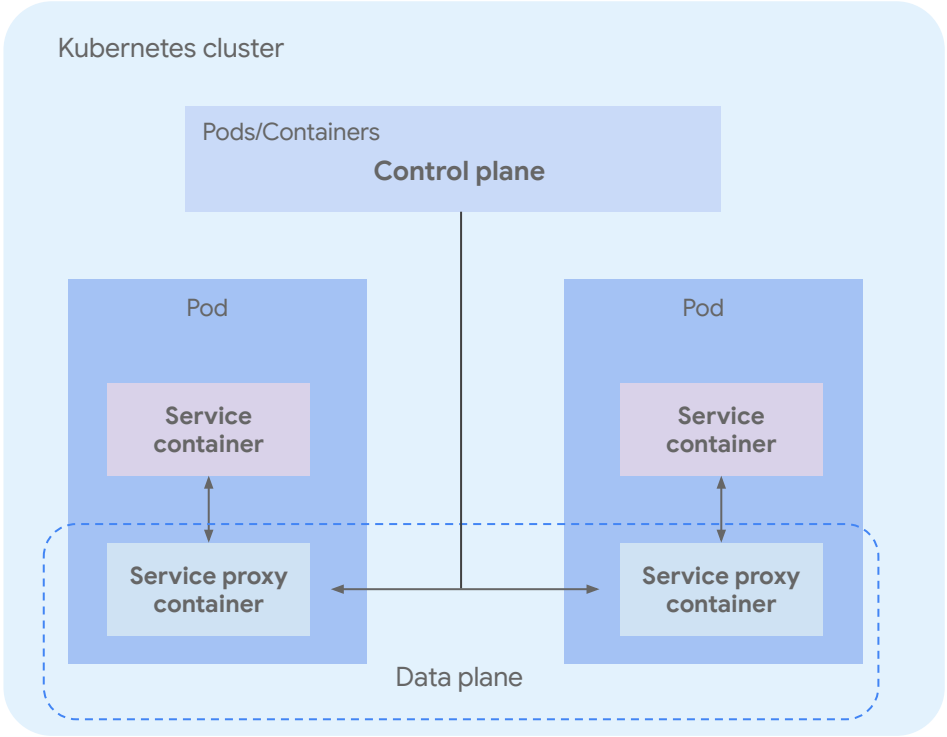
認証・認可、**メトリクス収集**

などの機能を担うソフトウェア

アプリケーション自体に手を入れるのではなく、

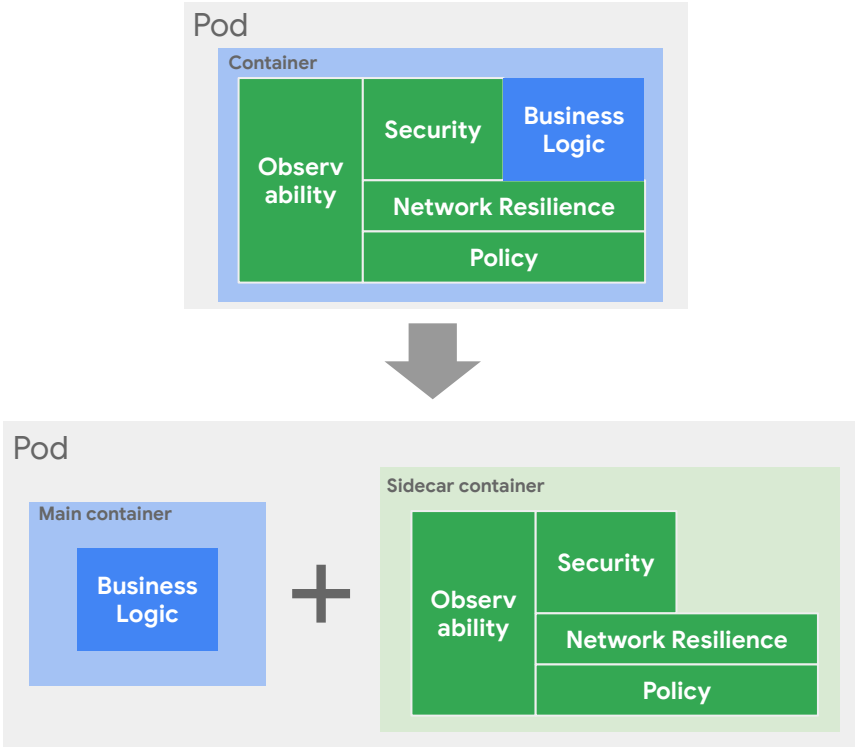
サイドカーと呼ばれるネットワーク

プロキシを使うことで機能を提供



サイドカー

- メインとなるコンテナに加えて、補助的な機能を持ったコンテナを内包した Pod デザイン
- メインコンテナに手を加えずに、トラフィック コントロールやセキュリティ機能、メトリクス収集といった高度な補助機能を実装することが可能（コードの改修不要）



Istio

- Google と IBM を中心に開発している
  サービスメッシュを提供するための
  オープンソースソフトウェア
- プロキシとして Envoy を利用

Istio

Connect, secure, control, and observe services.

Anthos Service Mesh 概要

Google Cloud

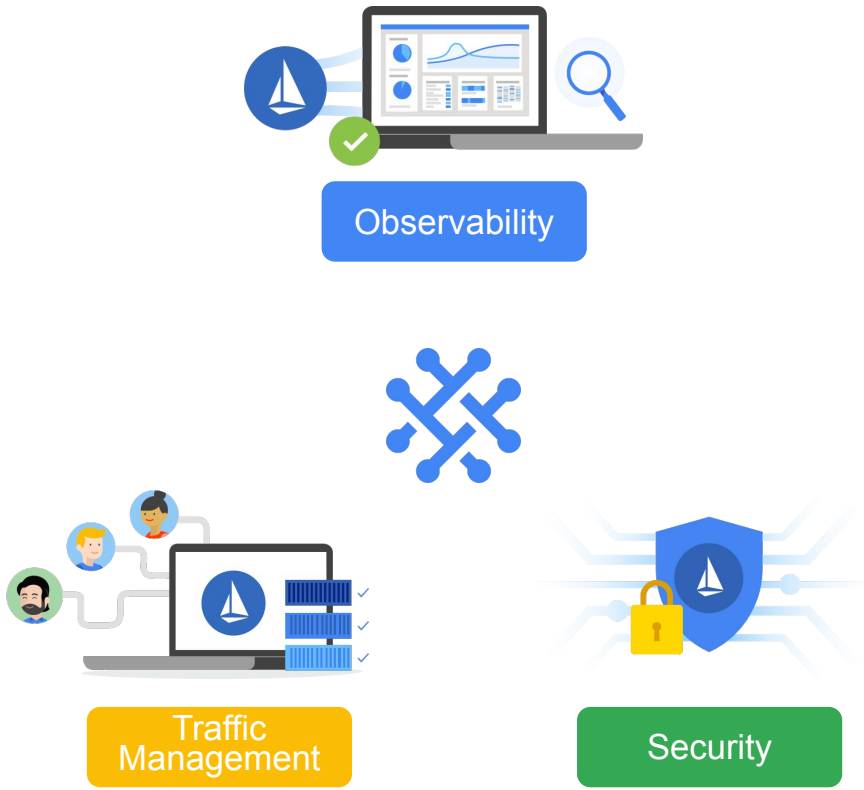
Anthos Service Mesh (ASM)

Google が提供するフルマネージドの
サービスメッシュ

Istio 互換の API, コントロール
プレーンを Google が管理

Operations suite とネイティブな連携

Istio 1.5.5 に対応（2020/06 現在）



対応プラットフォーム

2020.06 現在、ASM は下記プラットフォームをサポートしている

- Anthos GKE on GCP
  - リリースチャネルを利用しているクラスタ
  - 最低リソース要件： e2-standard-4, 4 ノード以上
- Anthos GKE on-prem
  - バージョン 1.2.0-gke.6 以上のクラスタ
  - 最低リソース要件：4 vCPUs, RAM 15 GB

本日は Anthos GKE on GCP を対象に説明します

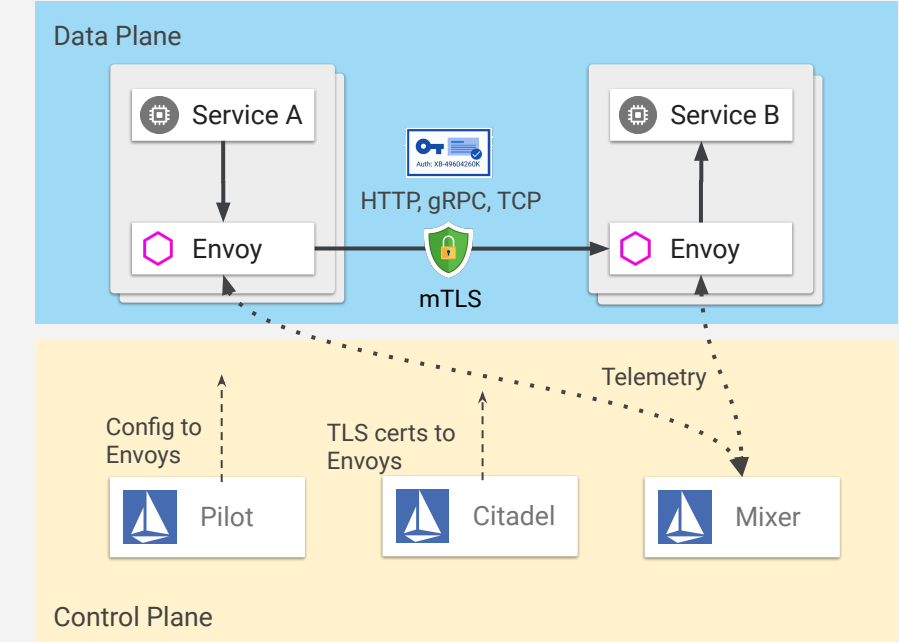
Istio のアーキテクチャ

一般的に Istio のコントロール プレーンは
データプレーンと同じ Kubernetes の
クラスタ上で動作する

主なコンポーネント

- Pilot
  - トラフィック コントロール
- Citadel
  - mTLS 用の CA
- Mixer
  - メトリクス収集
- Envoy
  - サイドカープロキシ

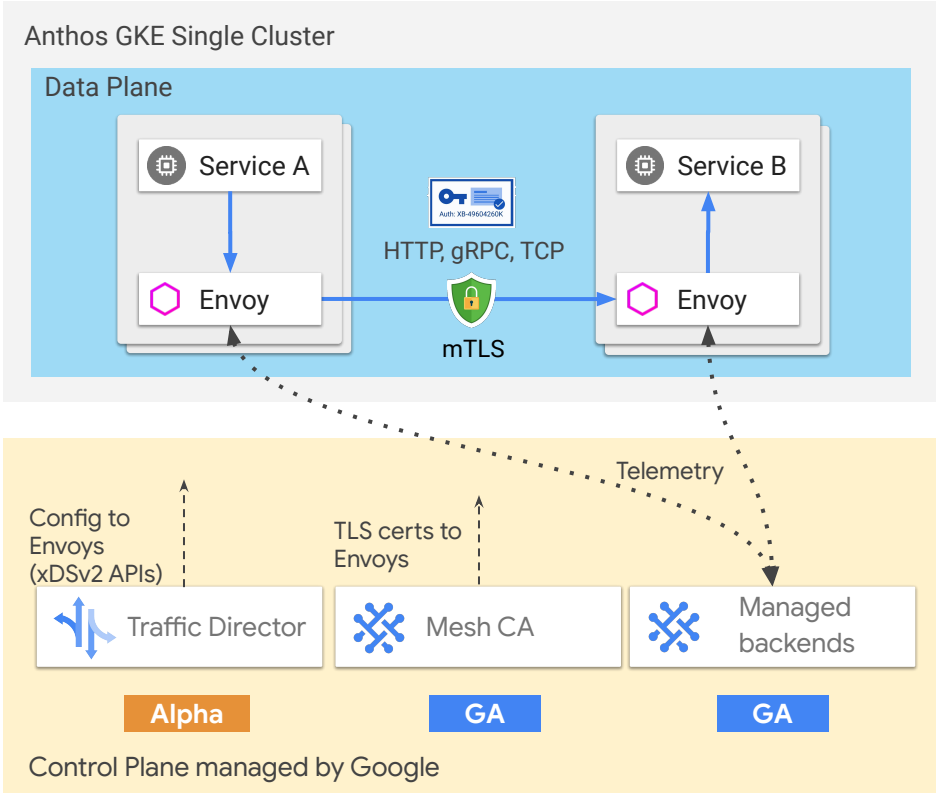
Istio 1.5 より上記コンポーネントは istiod として
シングルバイナリで動作



ASM のアーキテクチャ

OSS 版 Istio や Istio on GKE と違い、
コントロール プレーンがユーザー クラスタから
分離され、Google 管理となる

- Istio Pilot >> **Traffic Director**
- Istio Citadel >> **Mesh CA**
- Istio Mixer >> **Managed backends**



GKE クラスタ上で稼働する Istio コンポーネント（v1.5.x）

```
$ kubectl get pods -n istio-system
NAME                                READY   STATUS    RESTARTS   AGE
istio-ingressgateway-8f4454fb4-vl5t8   1/1     Running   0          5h32m
istiod-7bfcfb589b-2gd7x             1/1     Running   0          5h32m
promsd-786b67d77d-4skbx             2/2     Running   0          5h32m
```

- **istio-ingressgateway**: Ingress トラフィックの負荷分散や制御を行う Envoy Proxy
- **istiod**: Istio コントロールプレーン。ASM v1.5.x では主に Pilot / Galley の機能を利用
- **promsd**: Operations suite 連携用 Prometheus

Anthos Service Mesh
機能紹介

04

Google Cloud

Traffic Management

トラフィック コントロール

ASM では Traffic Director（現状は Pilot）を利用し、トラフィック コントロール機能を提供

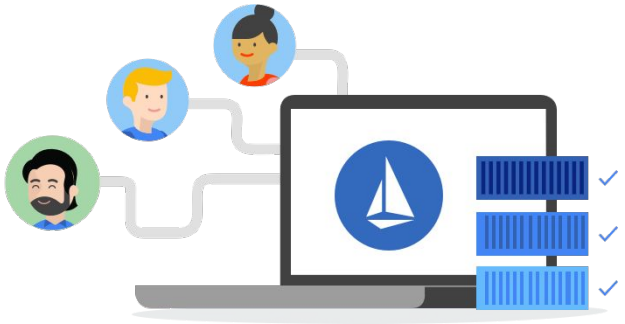
Traffic Splitting 機能等により Canary リリース や、Blue / Green デプロイメントのような高度な

ルーティングが実現

また、アプリケーションのコードを変更すること無く下記機能を実装し、アプリケーションの

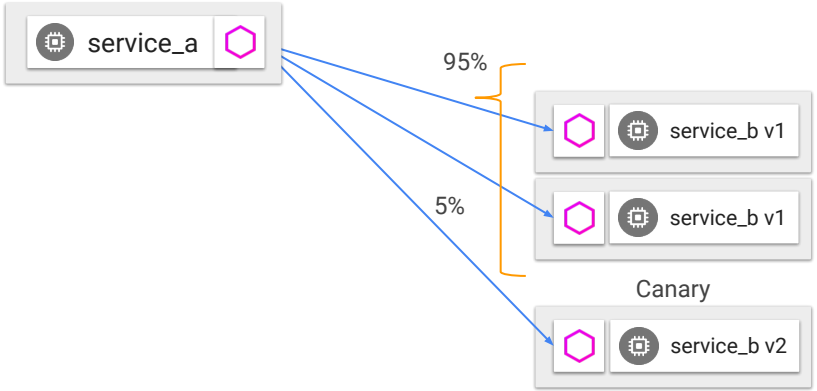
レジリエンシーを向上させることも可能

- Timeout
- Retry
- Fault Injection
- Circuit Breaker



Traffic Splitting による Canary リリース

```
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  name: service_b
spec:
  hosts:
    - service_b
  http:
  - route:
    - destination:
        host: service_b
        subset: v1
      weight: 95
    - destination:
        host: service_b
        subset: v2
      weight: 5
```

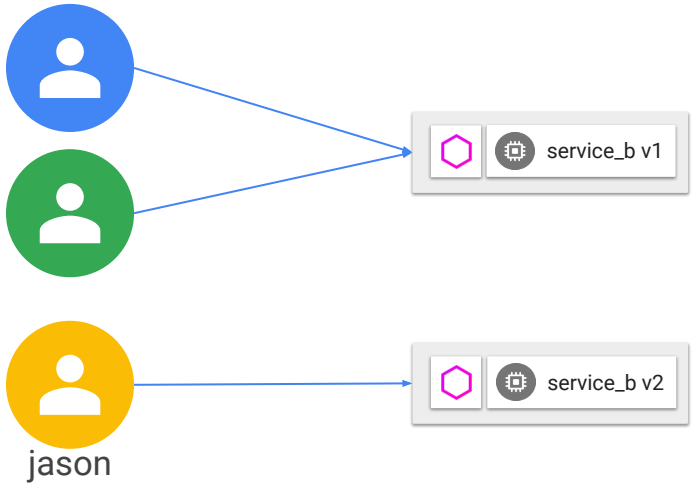


全体の 95% を service_b v1 へルーティングし、

残り 5% のトラフィックのみ service_b v2 へルーティングする

Traffic Steering によるユーザ情報に基づくルーティング

```yaml
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  name: service_b
spec:
  hosts:
  - service_b
  http:
  - match:
    - headers:
        end-user:
          exact: jason
    route:
    - destination:
        host: service_b
        subset: v2
  - route:
    - destination:
        host: service_b
        subset: v1
```



jason というユーザのみ service_b v2 へルーティングし、

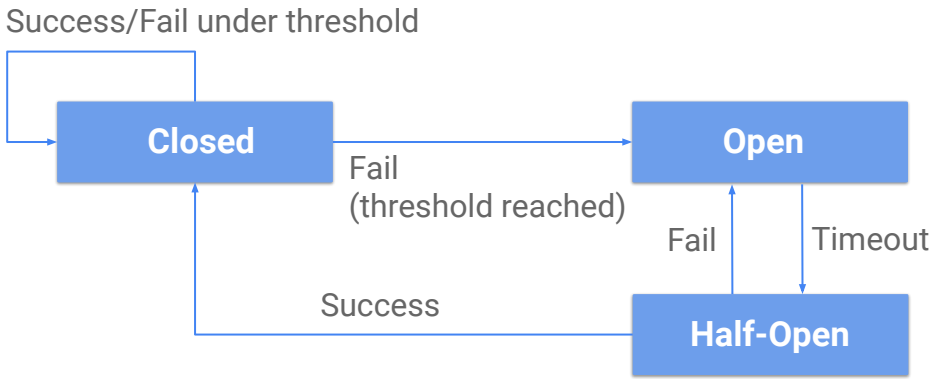
それ以外のユーザは service_b v1 へルーティングする

Circuit Breaker による障害影響範囲の極小化

```
apiVersion: networking.istio.io/v1alpha3
kind: DestinationRule
metadata:
  name: httpbin
spec:
  host: httpbin
  trafficPolicy:
    connectionPool:
      tcp:
        maxConnections: 1
      http:
        http1MaxPendingRequests: 1
        maxRequestsPerConnection: 1
    outlierDetection:
      consecutiveErrors: 1
      interval: 1s
      baseEjectionTime: 3m
      maxEjectionPercent: 100
```

サービス障害の連鎖的に波及を防ぎ、障害影響を抑えるために有効な手法

- リクエスト数がしきい値を超えるとエラーを返却する
- リクエスト先からエラーが返ってきた場合に一定期間、負荷分散対象から除外する、等



サポートされているプロトコル（2020.06 現在）

プロトコル	サポート
IPv4	Yes
HTTP / 1.1	Yes
HTTP / 2	Yes
TCP byte streams	Yes
gRPC	Yes
IPv6	No
L7 support for WebSocket, MongoDB, Redis, Kafka, Cassandra RabbitMQ, Cloud SQL	No (TCP byte streams で一部代替可)

https://cloud.google.com/service-mesh/docs/supported-features#networking

サポートされているロードバランシングポリシー（2020.06 現在）

ロードバランシングポリシー	サポート
Round robin	Yes
Least connections	Yes
Random	Yes
Passthrough	Yes
Consistent Hash	No
Locality-weighted	No

https://cloud.google.com/service-mesh/docs/supported-features#networking

Security

ASM が提供するセキュリティ機能

ASM では以下セキュリティ機能を提供

- mTLS によるセキュアな通信
- 証明書 / 鍵の自動ローテーション ASM
- マネージドなプライベート CA（Mesh CA） ASM
- 認証ポリシー
  - mTLS
  - JWT
  - Cloud IAP
- 認可ポリシー

Mesh CA

Multi-Regional なマネージドプライベート CA

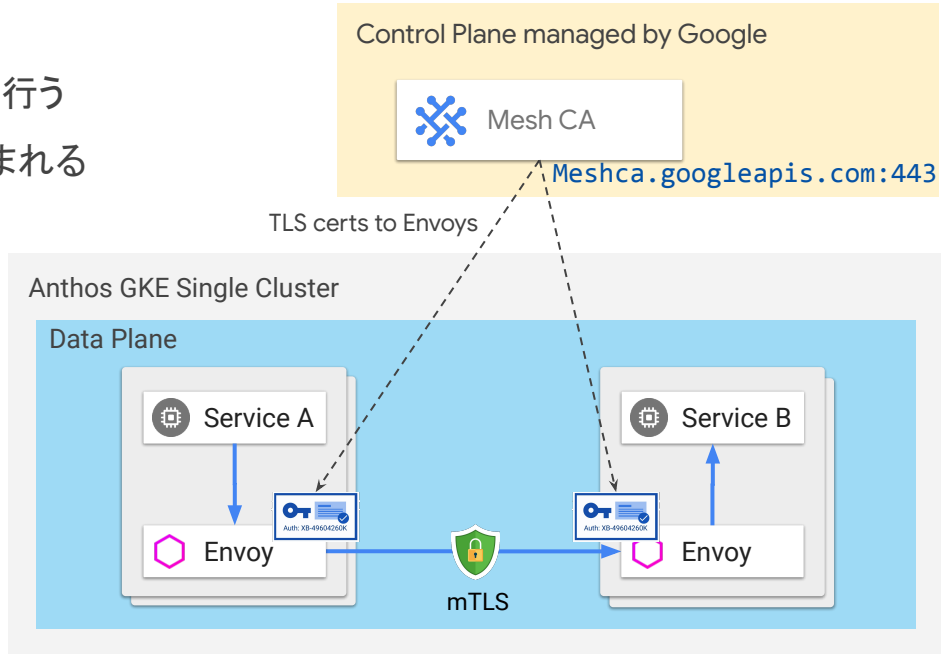
mTLS で利用される証明書の発行やローテーションを行う

Mesh CA から発行される証明書には以下情報が含まれる

- Google Cloud Project ID
- GKE Namespace
- GKE Service Account Name

※ Mesh CA 発行の証明書は

mTLS 以外の用途で利用不可



Mutual TLS (mTLS)

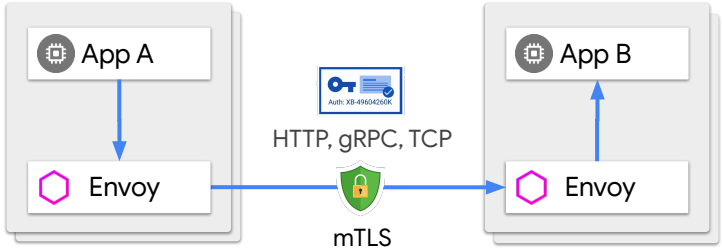
サーバ ⇔ クライアント間の相互認証（HTTPS + TLS クライアント認証）

クライアント側が、サーバ側の実行を許可されていることを確認して接続

相互認証により、接続先だけでなく、接続元の信頼も保証できる

mTLS は以下 2 つのモードが存在する：

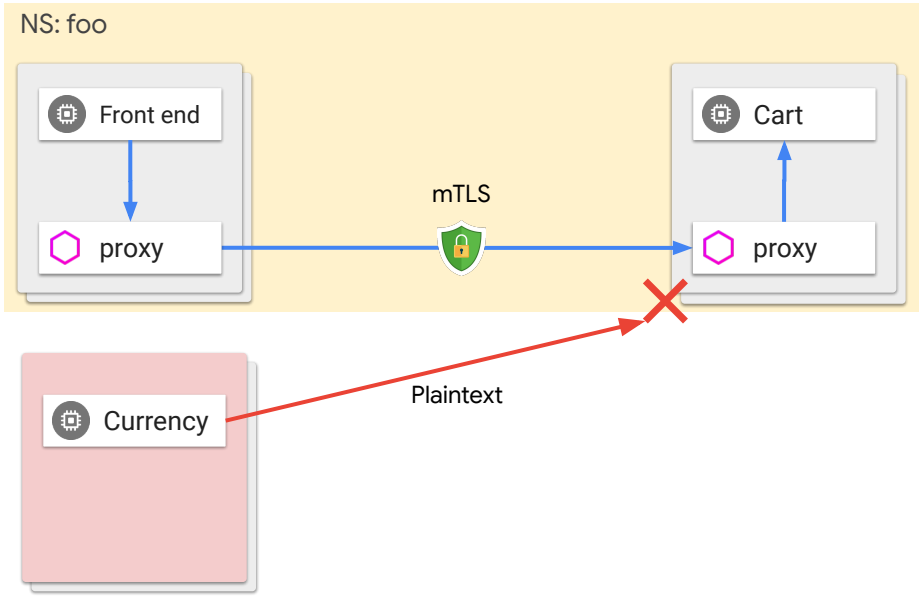
- **Permissive:** 暗号化された通信と平文どちらも許容する
- **Strict:** 暗号化された通信のみ許容する



mTLS の STRICT mode を有効化する例

```yaml
apiVersion: "security.istio.io/v1beta1"
kind: "PeerAuthentication"
metadata:
  name: "namespaced-peer-policy"
  namespace: "foo"
spec:
  mtls:
    mode: STRICT
```

foo namespace 内のワークロードは mTLS による通信を必須とする
サービス間通信を暗号化する必要のあるケース等で利用



認可ポリシーによるアクセス制御の例

```
apiVersion: security.istio.io/v1beta1
kind: AuthorizationPolicy
metadata:
 name: httpbin
 namespace: foo
spec:
 action: ALLOW
 rules:
 - from:
   - source:
       principals: ["cluster.local/ns/default/sa/sleep"]
   - source:
       namespaces: ["test"]
   to:
   - operation:
       methods: ["GET"]
       paths: ["/info*"]
   - operation:
       methods: ["POST"]
       paths: ["/data"]
   when:
   - key: request.auth.claims[iss]
     values: ["https://accounts.google.com"]
```

条件にマッチしない要求は拒否

From:（いずれかにマッチ）
- KSA が cluster.local/ns/default/sa/sleep
- Namespace が test

To:（いずれかにマッチ）
- /info* に対する GET リクエスト
- /data に対する POST リクエスト

When:
- https://accounts.google.com　によって発行された有効な JWT トークンがリクエストに含まれている場合

Observability

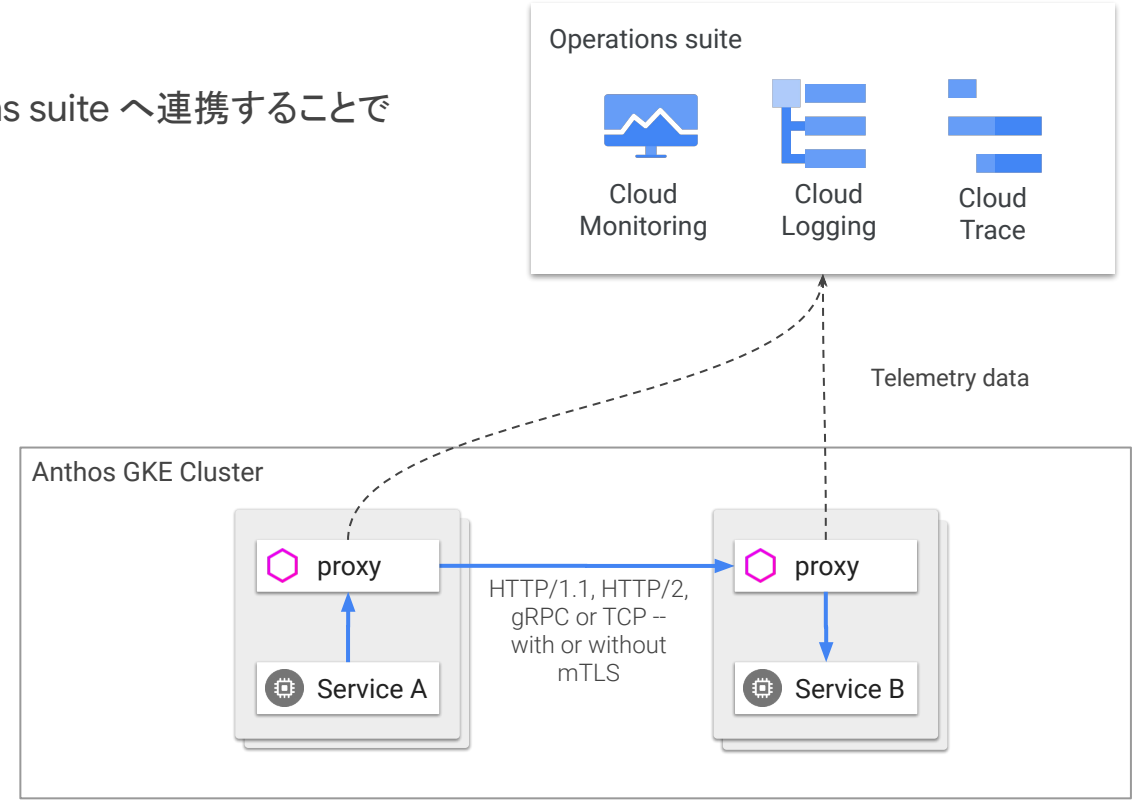
ASM が提供する可観測性

ASM では以下の可観測性に関する機能を提供

- レイテンシやリクエスト数、エラーレート等のメトリクスによる監視
- SLO を使ったサービスレベルの監視
- サービス間依存関係の可視化
- エラーログによる障害原因の調査

ASM と Operations suite の連携

Envoy のテレメトリデータを Operations suite へ連携することで
一元的なモニタリングを実現

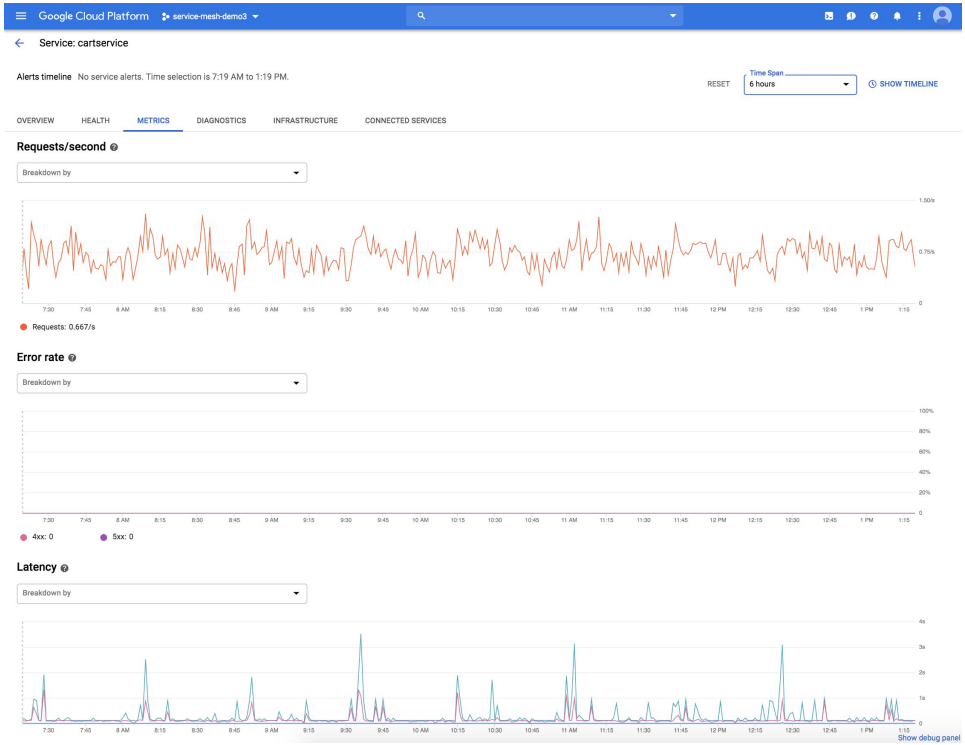


Observability

メトリクス監視

Cloud Monitoring と連携し、各サービスの主要メトリクスを監視

- Requests per second
- Error rate
- Latency
- Request size
- Response size 等



Google Cloud

Observability

SLO によるサービスレベル監視

SLO（Service Level Objectives）を定義し、

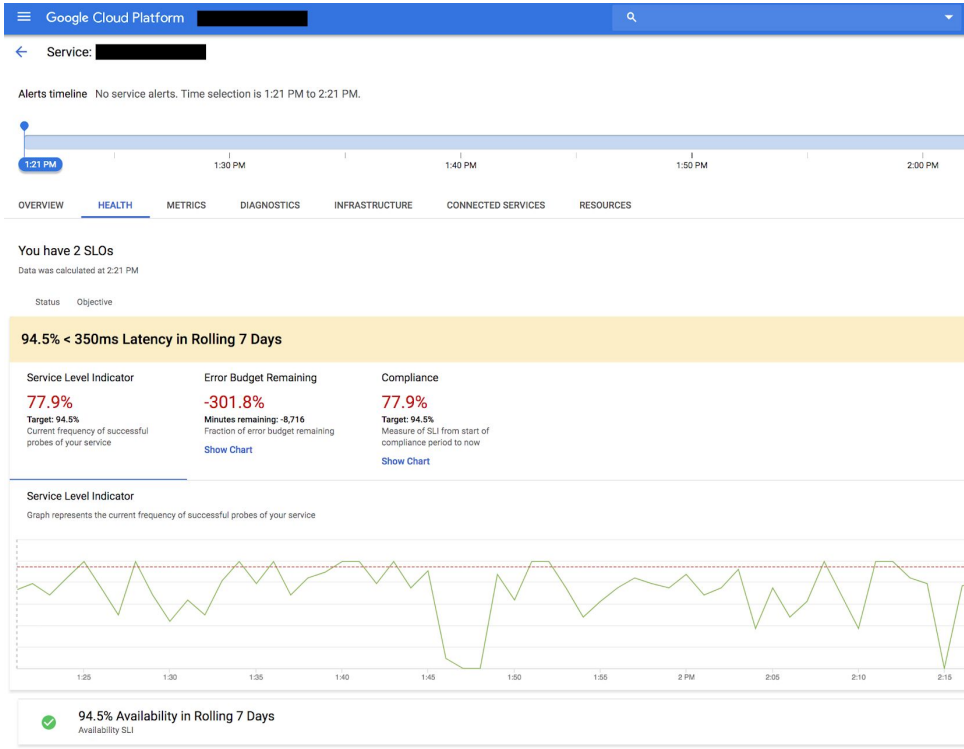
各サービスが SLO を遵守できているかを監視

➡サービスの可用性、品質を維持できているか

2020.06 現在、定義可能な SLI

（Service Level Indicator）は以下 3 種類:

- Availability
- Latency
- Custom

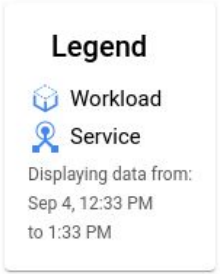


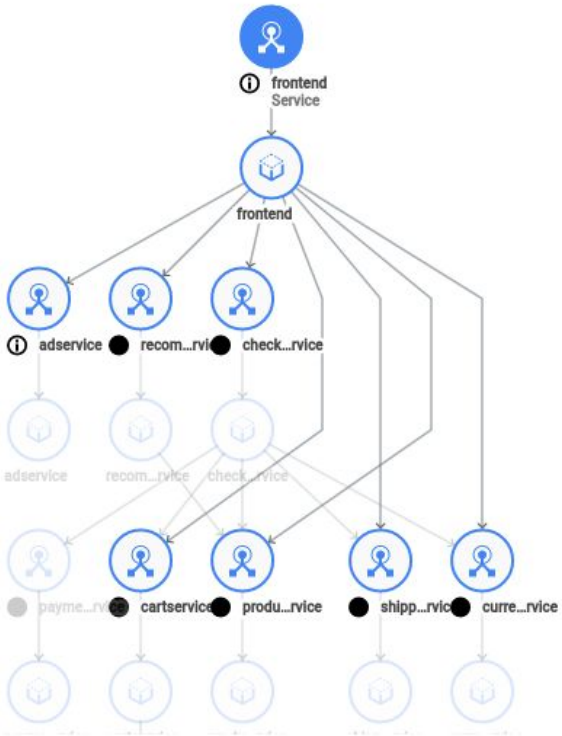
Google Cloud

サービストポロジグラフ

ASM Dashboard ページのトポロジグラフ
により以下サービス情報を可視化

- サービス間の依存関係
- 各サービスとサービス配下の Kubernetes ワークロード情報





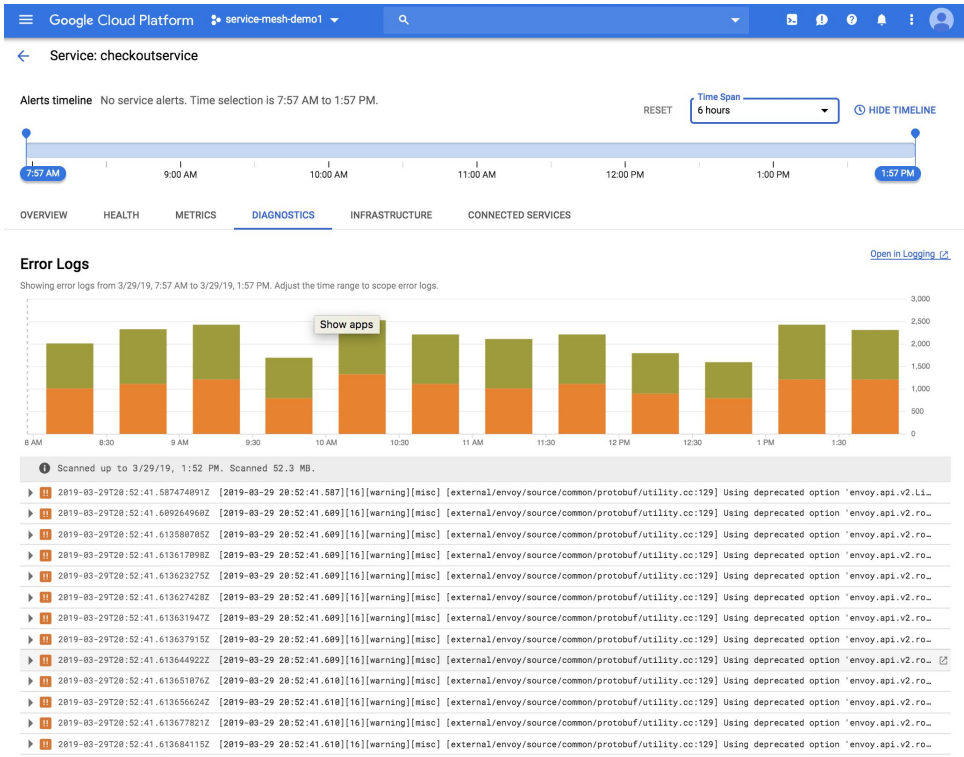
Observability

アクセスログ / エラーログ分析

Cloud Logging と連携し、各サービスのアクセスログやエラーログを確認

アクセスログには以下情報が含まれる

- HTTP リクエストプロパティ（URL, Latency, Header 等）
- アクセス元/アクセス先のワークロード情報（Name, Namespace, Identity, Common Label）
- Tracing が有効な場合、Trace ID や Span ID 等の Tracing 情報



Google Cloud

その他:ASM vs. OSS Istio

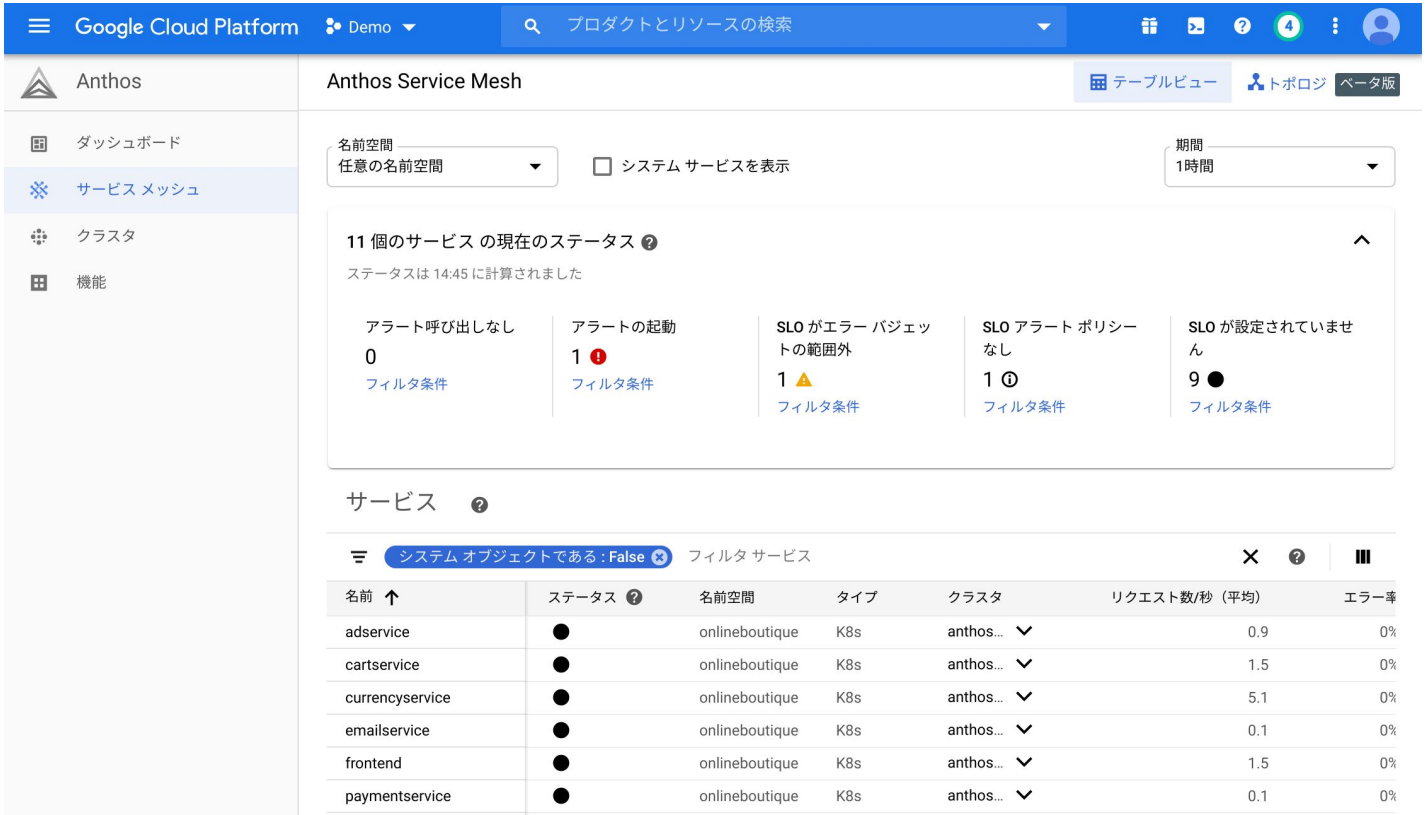
項目	ASM (on GCP)	OSS Istio
サポート	Google	コミュニティサポート
コントロールプレーン管理	Google マネージド	ユーザ管理(リソース管理、死活監視)
コントロールプレーンデプロイ先	Google Cloud (マネージド)	ユーザクラスタ内[*1]
mTLS 証明書 / 鍵の管理	Google マネージド	ユーザ管理
トレーシング	Cloud Trace	Zipkin, Jaeger and LightStep [*2]
メトリクス監視	Service Mesh Dashboard	Prometheus + Grafana [*2]
サービスメッシュ可視化	Service Mesh Dashboard	Kiali [*2]

*1... コントロールプレーンを他クラスタ上へデプロイすることも可能

*2... 記載されている製品以外も利用可能

Observability

ASM デモ - Service Mesh Dashboard



Google Cloud

まとめ

Google Cloud

まとめ

- サービスメッシュが注目される背景
  - 新機能をタイムリーに高速にリリースする → **マイクロサービス**
  - マイクロサービス化の課題を解決するためのソフトウェア → **サービスメッシュ**

- Anthos Service Mesh（ASM）とは Google が提供する**フルマネージドのサービスメッシュ**
  - マネージドサービスを利用することで、**コントロール プレーンの管理やリソース周りの 考慮等から開放**
  - GCP サービスとのネイティブな連携

Next steps

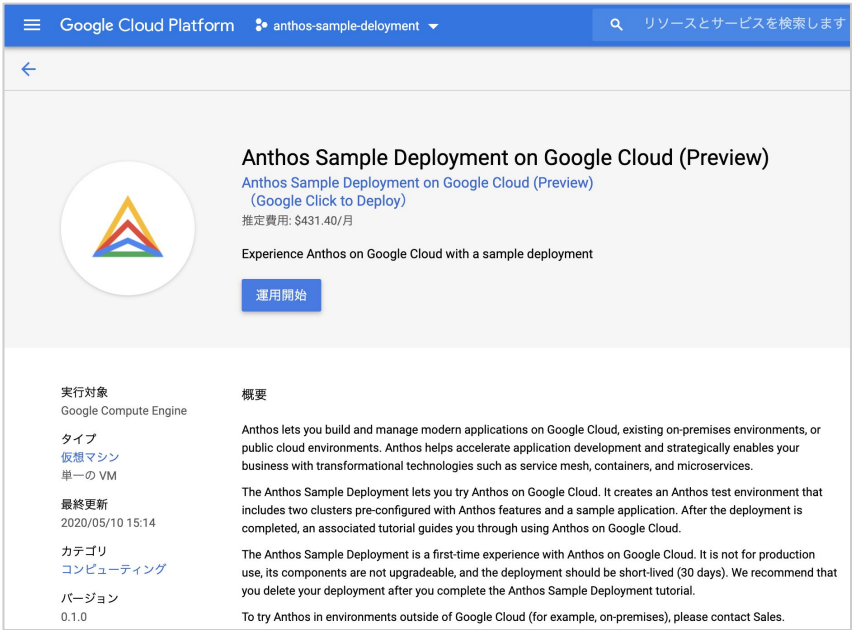
- オンラインの **ハンズオン** があります。Anthos をはじめとした Google Cloud の製品群を体験いただけます
  - Anthos: Service Mesh on Qwiklabs (https://www.qwiklabs.com/quests/100)
  - etc..
- Anthos Sample Deployment で実際に GCP 上で Anthos の各機能をハンズオンで体験することができます

Anthos Sample Deployment はじめ方

GCP の Marketplace から簡単に環境構築可能
（所要時間 15 分ほど、別途スクリプト版もあり）

2020 年 6 月現在、新規 GCP プロジェクトのみ
サポート（既存 GCP プロジェクトはサポート
されないので注意 ）

環境を構築する Zone や利用する Service Account
を選択することも可能 だが、
基本デフォルトのものを選択することを推奨



次回のセッションの紹介

- 次回のテーマは Apigee
- 日程:2020/07/03 20:00〜
- 疎結合なアーキテクチャを実現するための API 基盤である Apigee について説明

Thank you