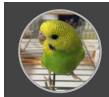
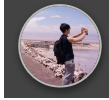
◆ OpenTelemetry Casual Talk ◆

サービスメッシュ環境における OpenTelemetry 活用

#otel_casual


柏原 由紀


逆井(さかさい) @ k6s4i53rx

お前だれ（逆井）

さかさい
逆井 啓佑

```
1
2   Kind: バックエンドエンジニア と SRE
3
4   Hobby:
5     - スノーボード：バックカントリー、気になっています
6     - ゴルフ：絶望的なカットスイング軌道を、
7               強烈なストロンググリップで相殺するスタイル
8     - 最近、Otel への eBPF 活用につよく興味があります
9
10  Community:
11
12    - OpenTelemetry Meetup スタッフ
13
14    X 逆井(さかさい) @ k6s4i53rx
15
```

intro_po.yaml

柏原 由紀

- NTTデータグループ
- 技術革新統括本部　クラウド技術部

- これまでの技術領域
  - クラウド(AWS/Azure)
  - PaaS(K8S/istio、PCF)
  - MSA(Go/Nodejs/React)
  - アジャイル（Scrum、SAFe)

現在の案件

- 社内向けのシステム開発を複数チームで実施

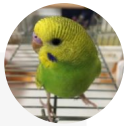
本日 30 分くらいでお話しすること

▶▷▶ 第一部：理論編

10分

サービスメッシュ環境で OTel を活用した分散トレース手法について紹介

- 分散トレースの基礎知識（コンテキスト伝播など）
- サービスメッシュ環境における Envoy を用いた分散トレース



▶▷▶ 第二部：事例編

20分

実際のサービスメッシュ案件での OpenTelemetry を使った分散トレースや、
オブザーバビリティ周辺のシステム構成について紹介

- オブザーバビリティ周りの課題
- 各種ツールを用いてどのように解決をしたか(しようとしているか)

第一部：サービスメッシュ環境で OTel を活用した分散トレース手法

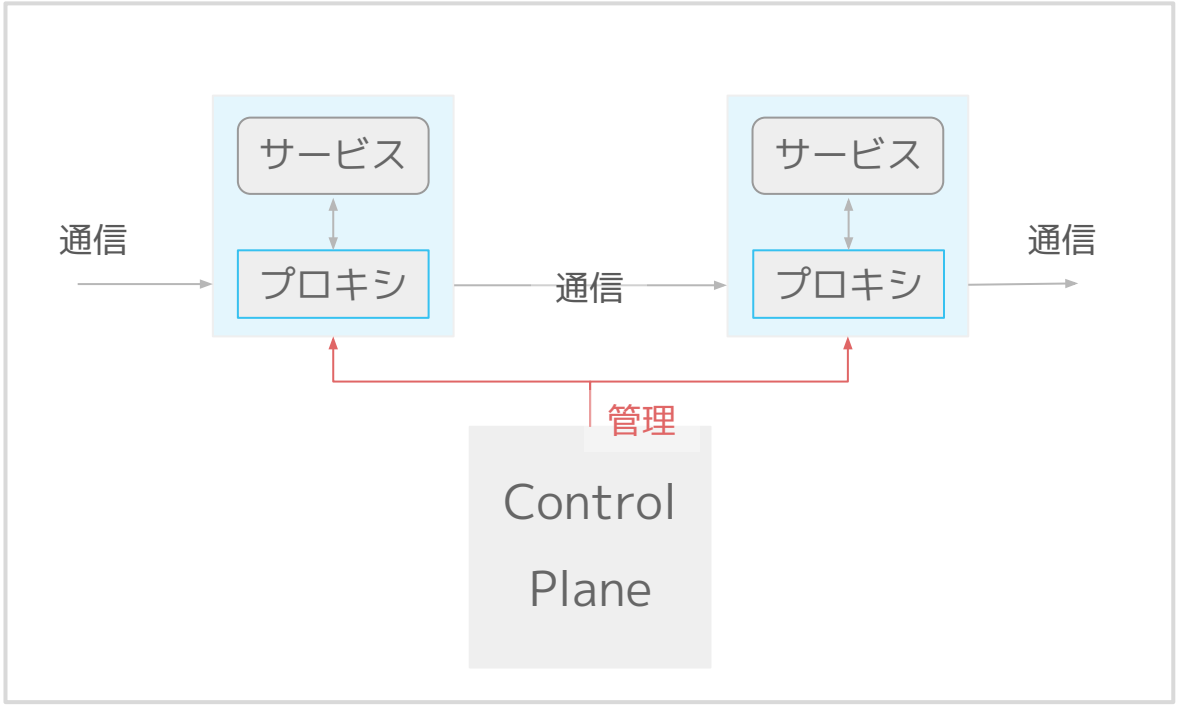
第一部：サービスメッシュについて

サービスメッシュは、

- サービス間通信にプロキシ配置
- プロキシ群をコントロール
  プレーンが管理する

ネットワークモデル

マイクロサービスに
共通的な機能を透過的に具備
することができる。



図：サービスメッシュの概要

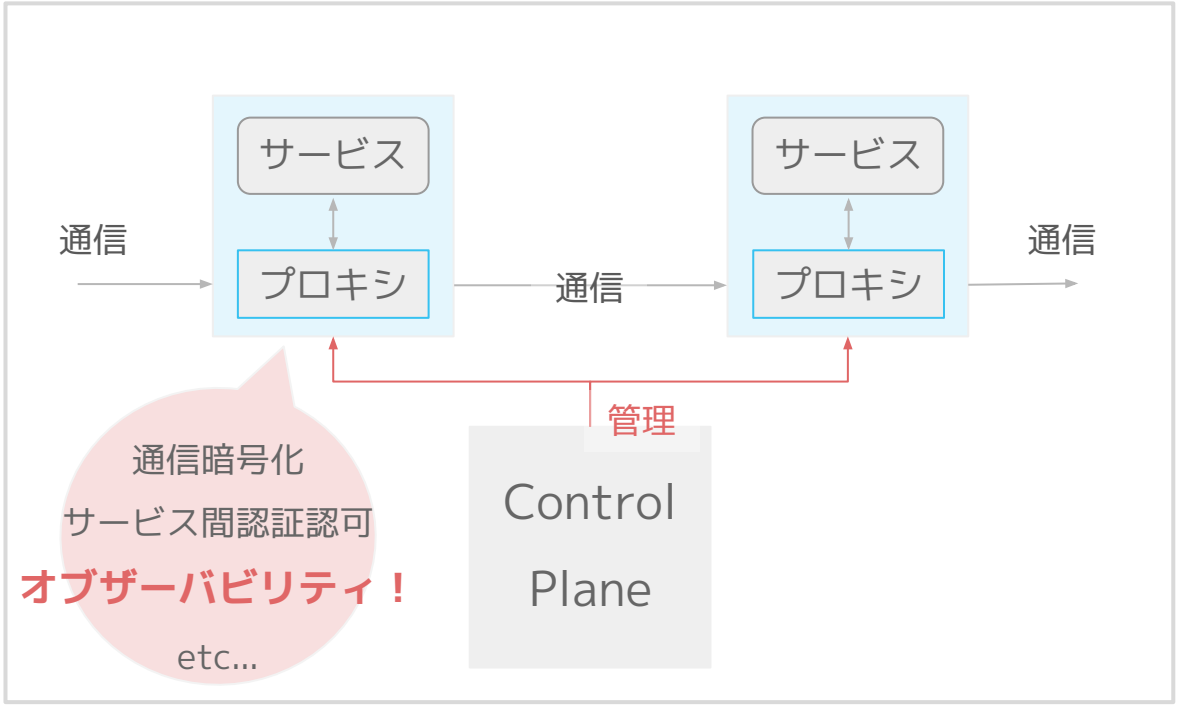
第一部：サービスメッシュについて

サービスメッシュは、

- サービス間通信にプロキシ配置
- プロキシ群をコントロール
  プレーンが管理する

ネットワークモデル

マイクロサービスに
共通的な機能を透過的に具備
することができる。



図：サービスメッシュの概要

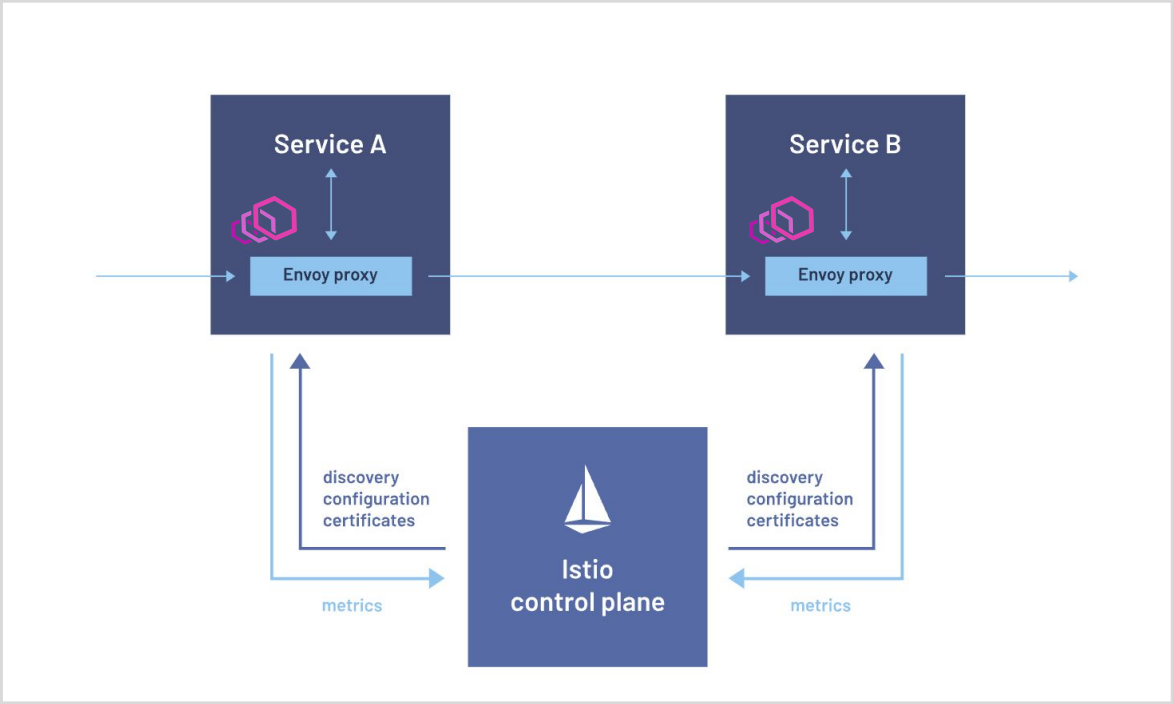
第一部：サービスメッシュについて

サービスメッシュの実装として、

- **Istio**, Linkerd, ... (OSS)
- Anthos Service Mesh (GKE)
  App Mesh (EKS)
  ...

▶▷ Istio

- プロキシに Envoy を使用
- Envoy の設定を抽象化してくれる
  - Istio CR として設定できる
- Envoy が実現するたくさんの機能の一つに**「分散トレース」**がある

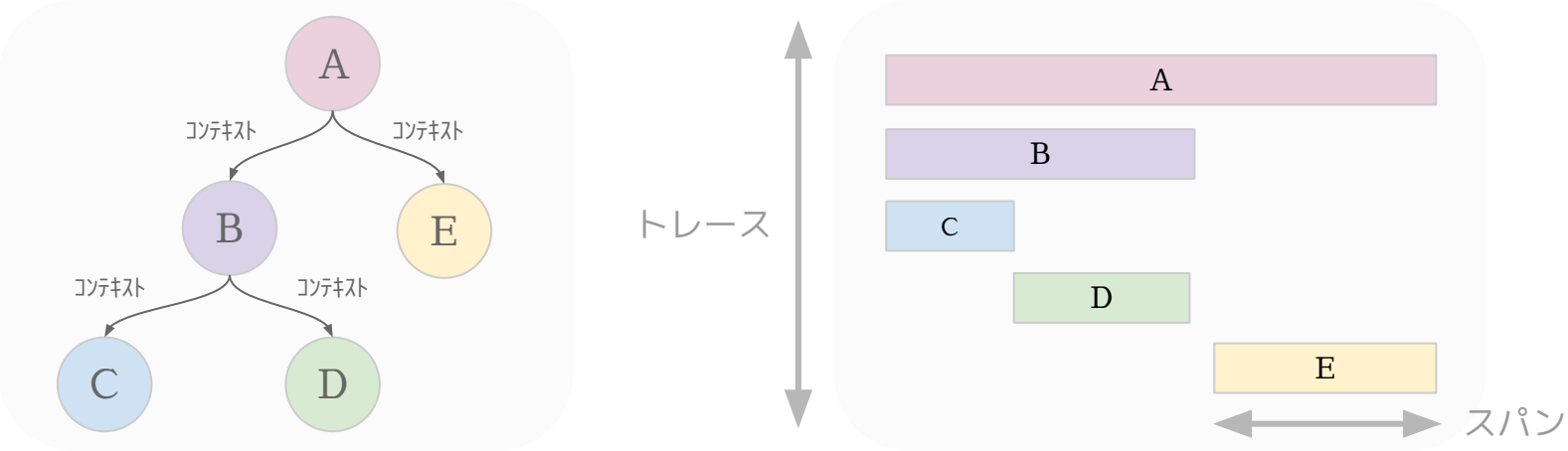


The Istio service mesh:
https://istio.io/latest/about/service-mesh/

参考:【Istio⛵】Istioによって抽象化されるEnvoyのHTTPSリクエスト処理の仕組み
https://hiroki-hasegawa.hatenablog.jp/entry/2024/01/16/013404

分散トレースの主な構成要素

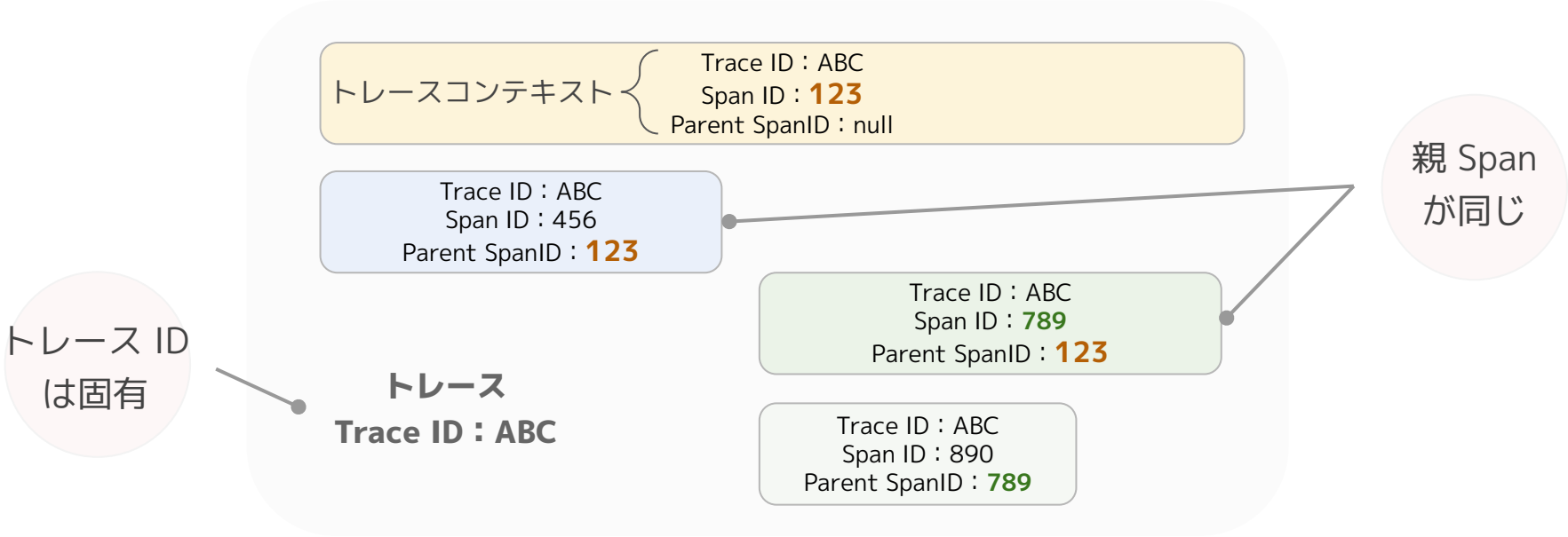
- トレース・・・・・・・・・リクエストの **一連の流れ** を示す
- スパン・・・・・・・・・・トレースの構成要素で、**特定の処理** を示す
- トレースコンテキスト・・分散トレースする上で必要な情報



参考：https://www.jaegertracing.io/docs/1.45/architecture/

トレースコンテキストの伝播 (Context Propagation)

トレースコンテキストにより Waterfall Graph の描画に必要な情報が伝播していく。



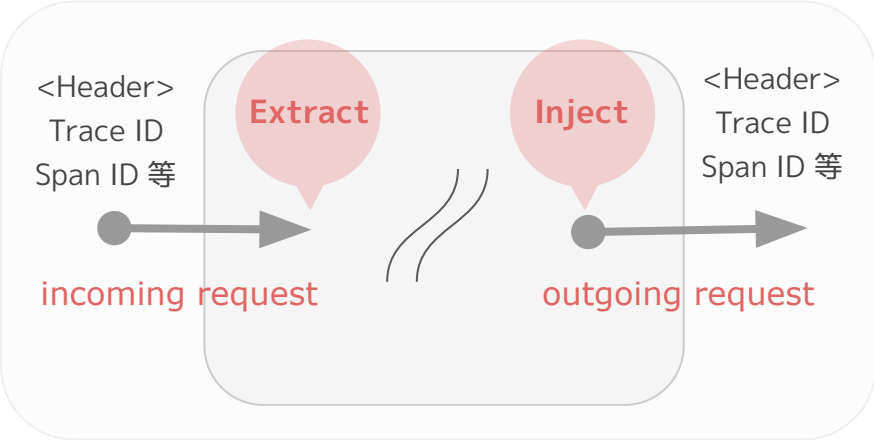
参考：https://docs.lightstep.com/docs/understand-distributed-tracing

サービス間のトレースコンテキスト伝播

サービス間では HTTP ヘッダを使って、トレースコンテキストを伝播させます。

トレースコンテキストを伝播するフォーマット

- W3C Trace Context
- B3 multi-header
- Datadog
- e.t.c...

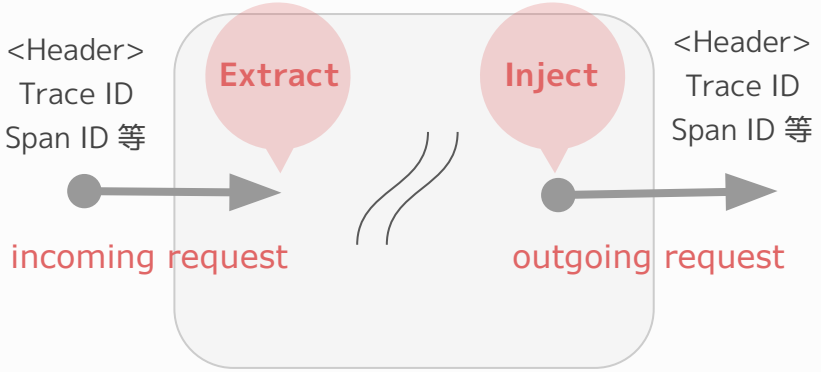


サービス間のトレースコンテキスト伝播

サービス間では HTTP ヘッダを使って、トレースコンテキストを伝播させます。

トレースコンテキストを伝播するフォーマット

- **W3C Trace Context**
- B3 multi-header
- Datadog
- e.t.c...



W3C Trace Context 形式で伝播する場合、

```
{traceparent: <version>-<trace-id>-<span-id>-01}
```

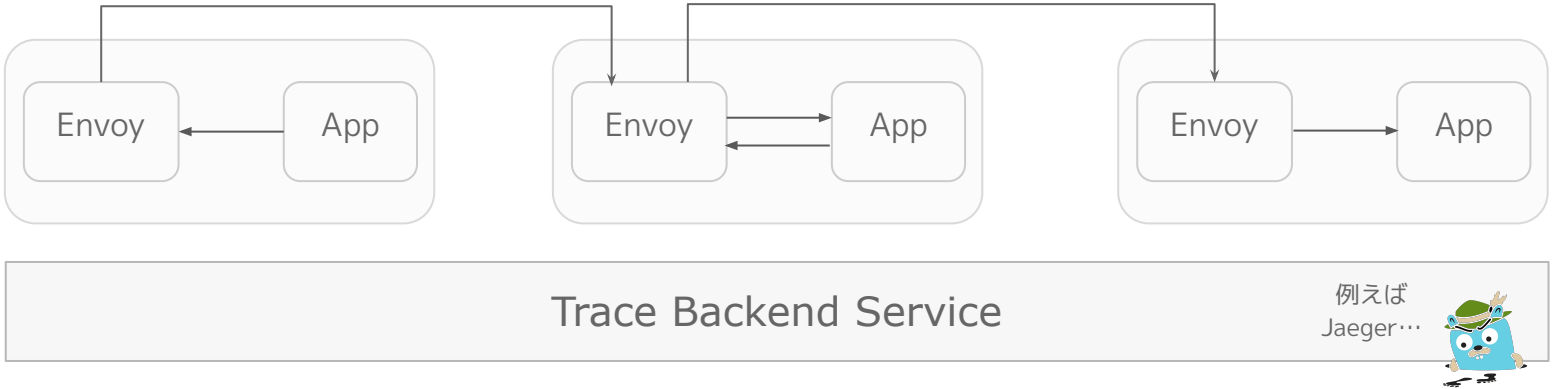
（例）：00-0af7651916cd43dd8448eb211c80319c-b7ad6b7169203331-01

参考：https://www.w3.org/TR/trace-context/

第一部：サービスメッシュと OpenTelemetry？

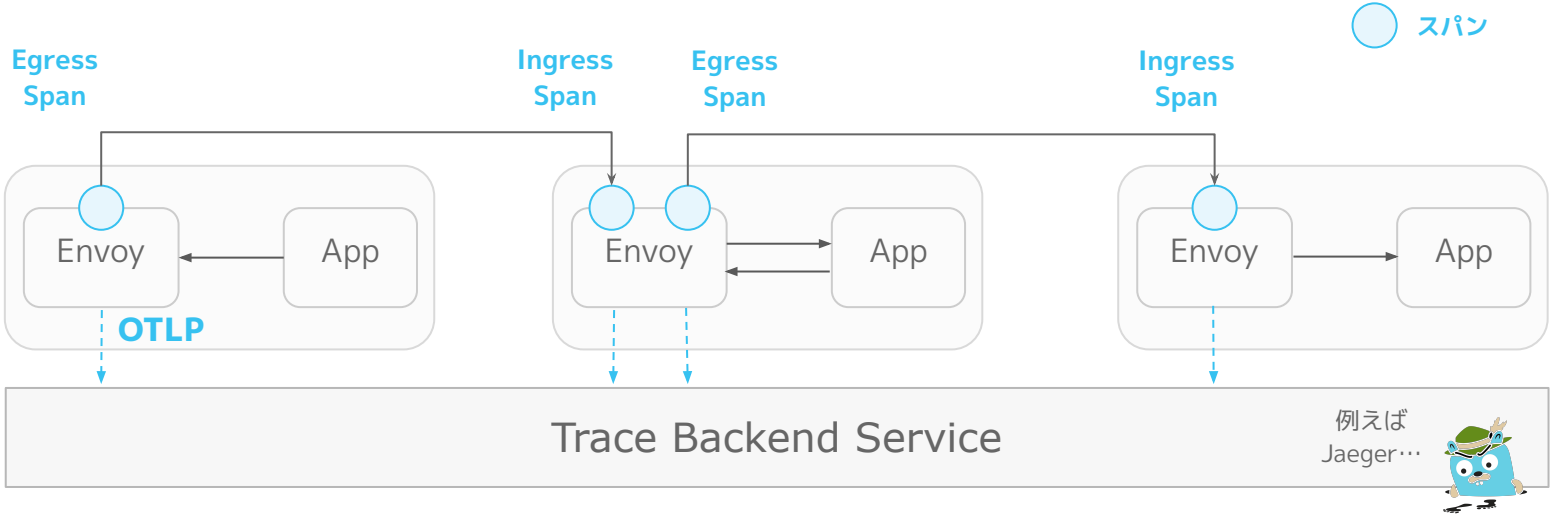
https://istio.io/latest/docs/tasks/observability/telemetry/

- Envoy の機能の一つの「分散トレース」がある
  - リクエストの ingress / egress でスパンの生成 / コンテキスト伝播
- OpenTelemetry とのインテグレーションがある！
  - W3C Trace Context でヘッダを伝播 (B3 や Datadog 形式の設定も可能)
  - Envoy の生成したスパンを OTLP 形式でトレースバックエンドに送信



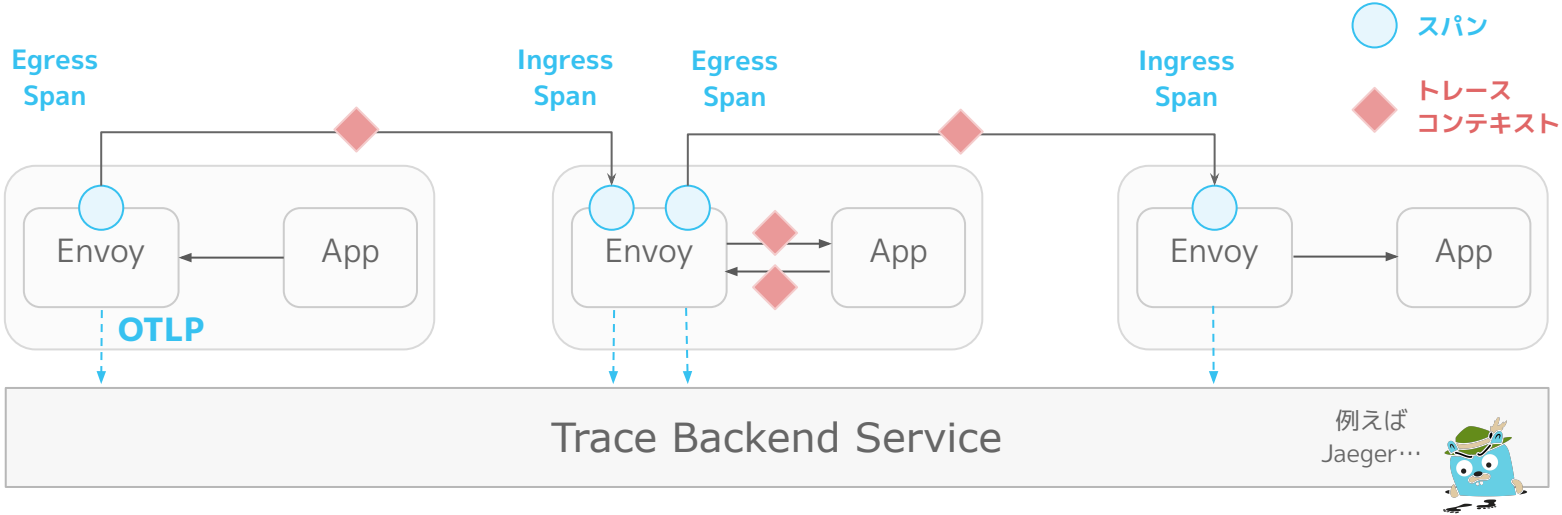
第一部：サービスメッシュと OpenTelemetry？

- Envoy の機能の一つの「分散トレース」がある
  - リクエストの ingress / egress でスパンの生成 / コンテキスト伝播
- OpenTelemetry とのインテグレーションがある！
  - W3C Trace Context でヘッダを伝播 (B3 や Datadog 形式の設定も可能)
  - Envoy の生成したスパンを OTLP 形式でトレースバックエンドに送信



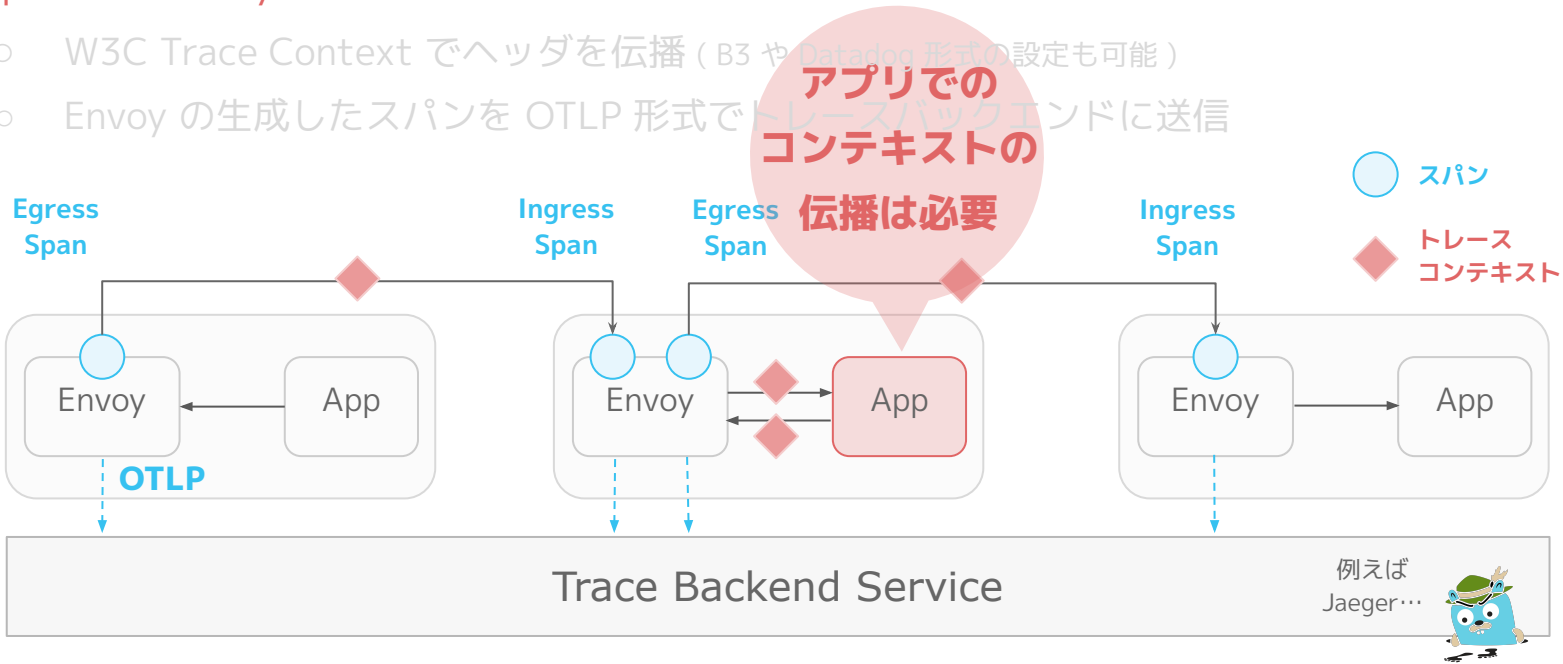
第一部：サービスメッシュと OpenTelemetry？

- Envoy の機能の一つの「分散トレース」がある
  - リクエストの ingress / egress でスパンの生成 / コンテキスト伝播
- OpenTelemetry とのインテグレーションがある！
  - W3C Trace Context でヘッダを伝播 (B3 や Datadog 形式の設定も可能)
  - Envoy の生成したスパンを OTLP 形式でトレースバックエンドに送信



第一部：サービスメッシュと OpenTelemetry？

- Envoy の機能の一つの「分散トレース」がある
  - リクエストの ingress / egress でスパンの生成 / コンテキスト伝播
- OpenTelemetry とのインテグレーションがある！
  - W3C Trace Context でヘッダを伝播（B3 や Datadog 形式の設定も可能）
  - Envoy の生成したスパンを OTLP 形式でトレースバックエンドに送信



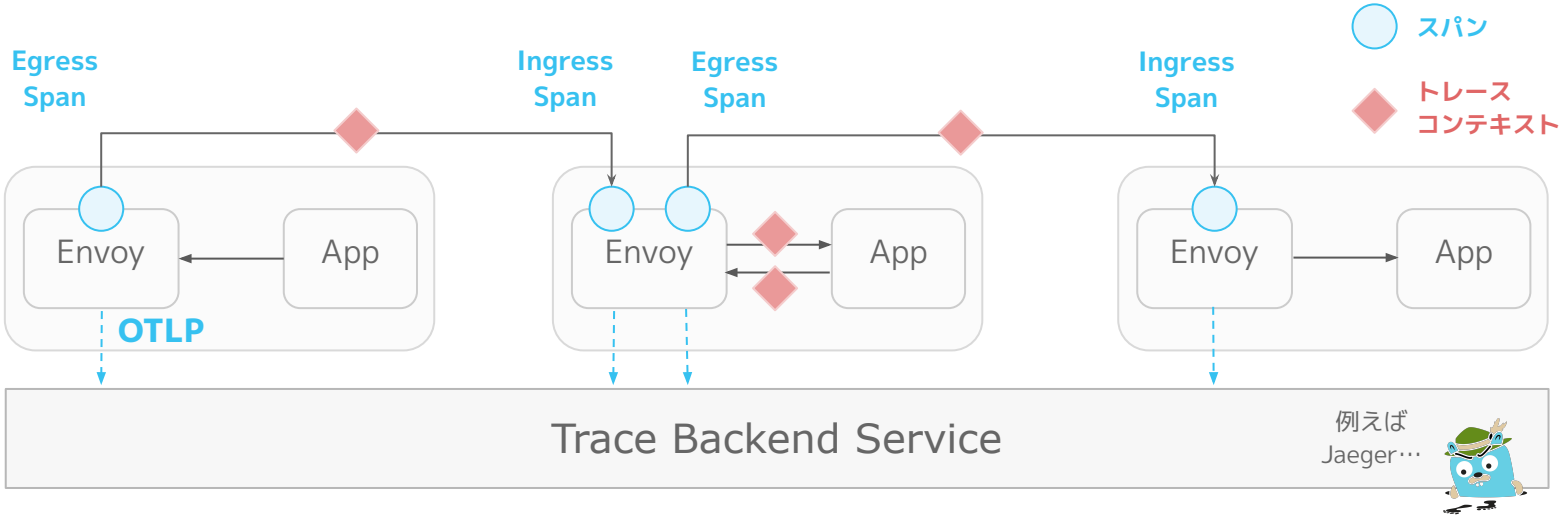
第一部：サービスメッシュと OpenTelemetry？

アプリケーション側で、スパンの生成やエクスポーターを行わず、

アプリケーションへのコード変更のコスト低く、

サービスメッシュ環境で透過的に分散トレースの仕組みをマイクロサービスに実装

※ アプリケーションでコンテキスト伝播は必要であることに注意 (OTel の Propagate API, 自前実装)



第一部：ここまでのまとめ

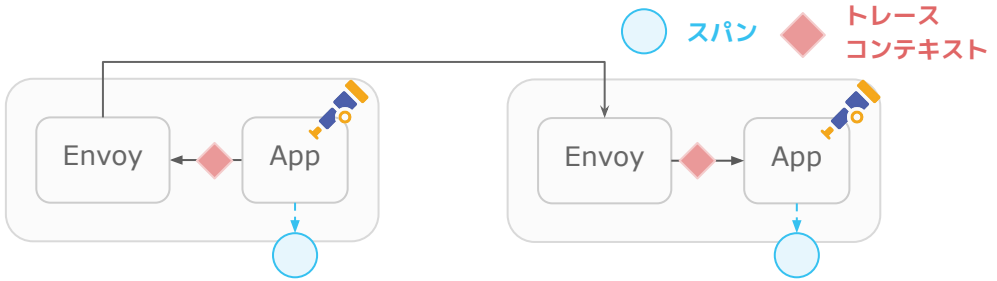
ー ここまでのまとめ ー

- Envoy に分散トレースの一部をオフロードして、アプリ側の計装コスト減
  - サービスメッシュ環境で透過的に分散トレース
  - サービス単位でのボトルネックを探る上では Good
- とはいえ、可能ならばアプリ側の計装も行う方が良い
  - テレメトリーの関連付け (correlate) や、ディメンションを高める観点においても重要
  - 特殊事情でアプリ変更をフッ軽にできない場合はサービスメッシュで「荒く、横断的に」でも有効
- アプリ計装 + Envoy のトレースの場合、Envoy 起因の性能問題にも有効
  - アプリを OTel で計装して、Envoy のトレースと結合してきめ細かいトレース取得が可能
  - トレースデータの増加や Waterfall Graph の複雑化はある

第一部：サービスメッシュ環境における OpenTelemetry

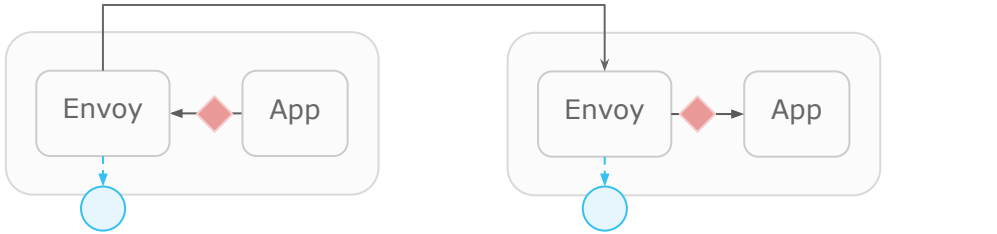
アプリを計装して分散トレース

- アプリの性能解析をより詳細に（関数や処理の粒度で）見ることができ得る
- Envoy との通信は隠れてしまう



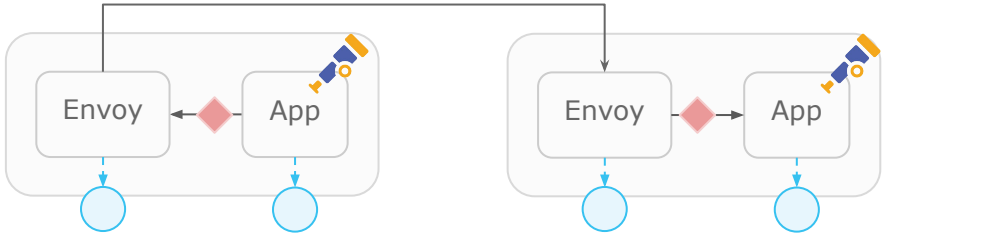
Envoy の分散トレースを有効化

- アプリ側の計装コストを下げて、サービス単位での分散トレースを行うことができる
- スパンはサービス単位(Envoy 間)になる



アプリの計装と Envoy の分散トレースを結合

- Envoy における性能影響を調査することができる
- トレースデータの増加や、それに伴う Waterfall Graph の視認性は複雑になるかも



◆ 補足：サービスメッシュ（Envoy）のトレースについて

- サービスメッシュ自体の遅延や問題を分散トレースできるので、トラブルシューティングに役立つ
- 一方で、以下のような課題もある
  - サービスメッシュのスパンによりデータ量が増加
  - アプリケーションで生成するスパンほど、情報量を付与することができない

Yuri Shkuro 氏 による、サービスメッシュにおける分散トレースに関連する記事があります。
サービスメッシュを分散トレースに含めるメリデメの議論がありわかりやすいので紹介します。

Myth: service mesh can do distributed tracing of your application

https://medium.com/@YuriShkuro/myth-service-mesh-can-do-distributed-tracing-of-your-application-7a5cb5e3b617

第一部：まとめ

サービスメッシュ環境における分散トレースの技術紹介をしました。

柏原さんにバトンタッチして実際の案件での OpenTelemetry 活用話にうつります。

第二部：事例紹介

2024.3.25

NTTデータグループ
柏原 由紀、岡本 隆史

第二部は別 URL での掲載になります。

Speaker Deck の概要欄に第二部のスライドリンクを掲載します。