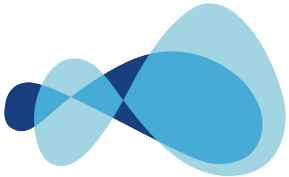
Observability in Service Mesh

Powered by Envoy and Apache SkyWalking



Sheng Wu, Lizan Zhou

06-25-2019

Agenda

Tetrate

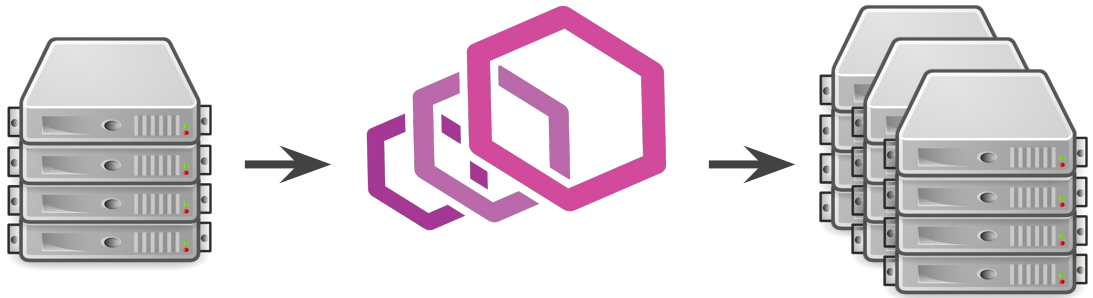
- What is Service Mesh?
- Envoy Overview
- Envoy Data Plane API
- Observability
- Connect Envoy and Istio to SkyWalking
- Q&A

What is Service Mesh?

Tetrate

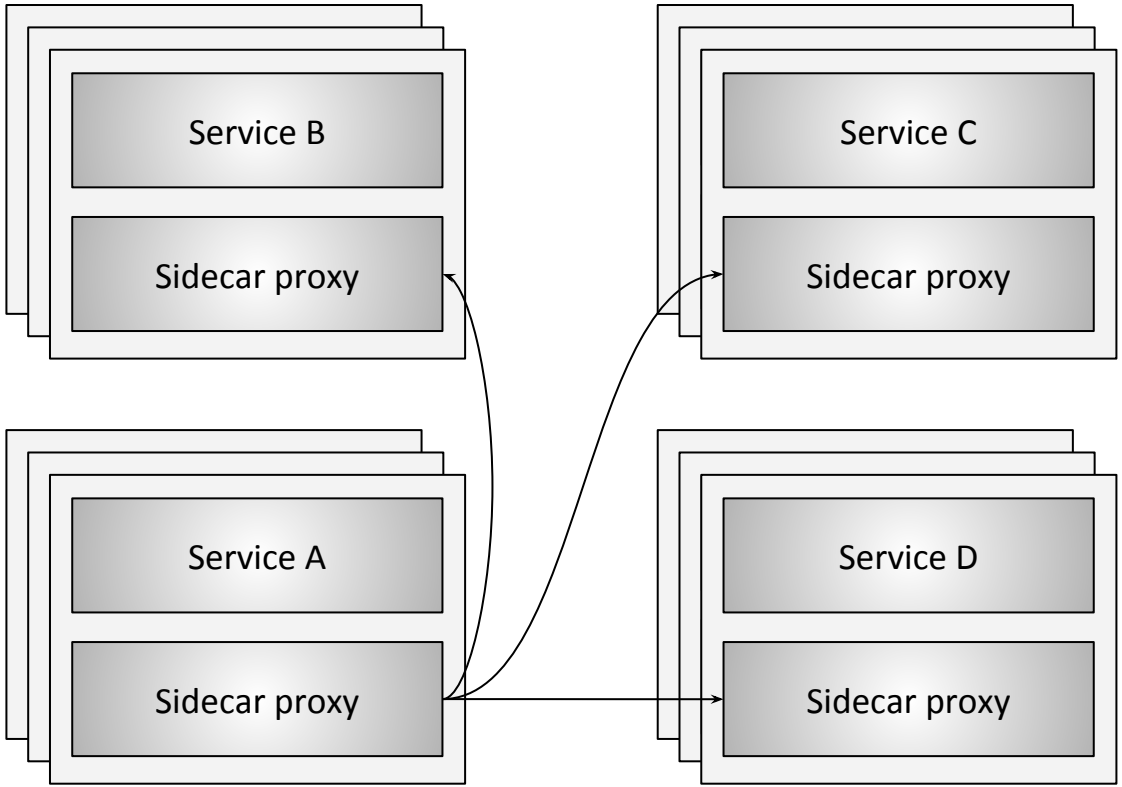
The network should be transparent to applications.

When network and application problems do occur it should be easy to determine the source of the problem.



Service mesh refresher

Tetrate

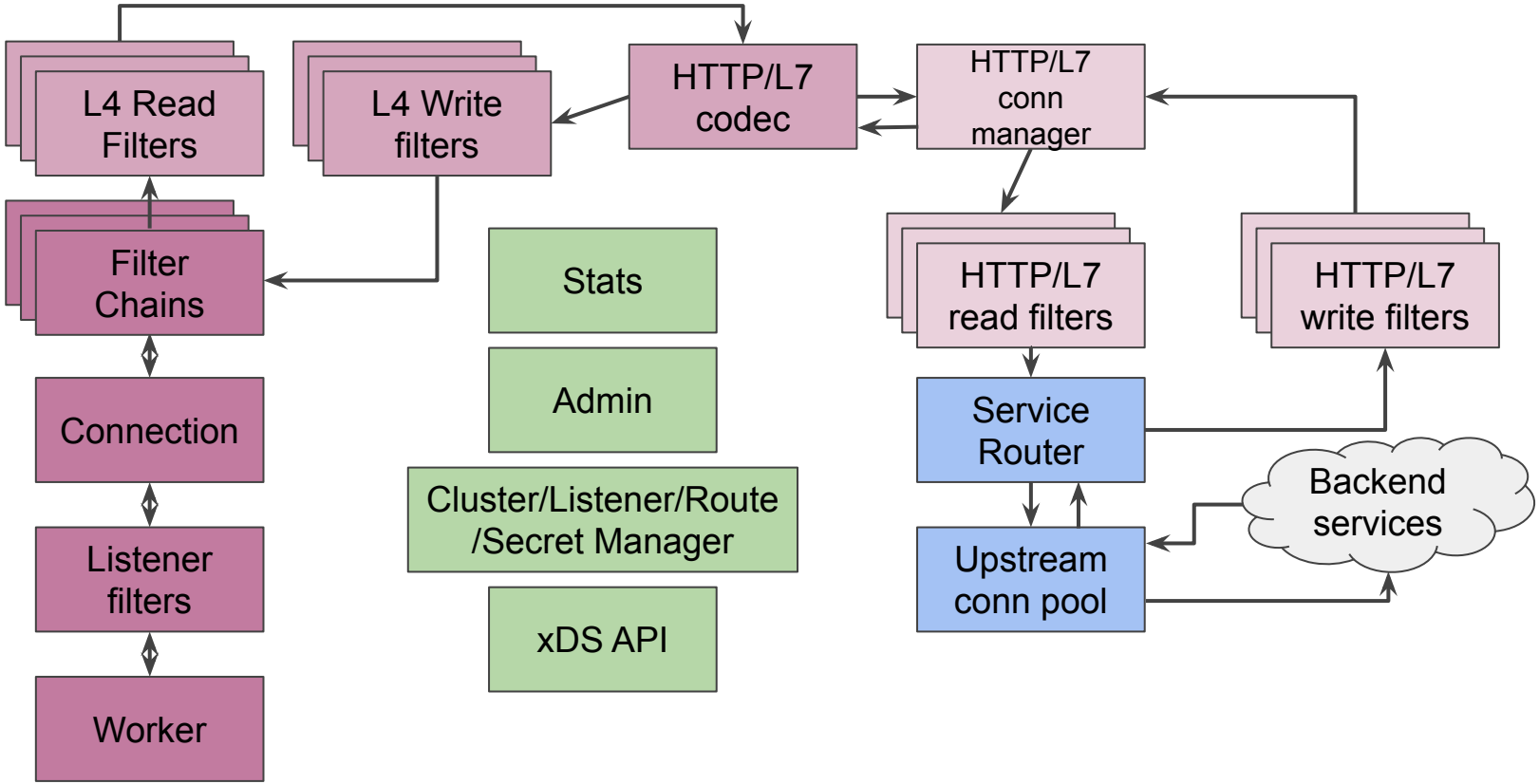


What is Envoy

- **Out of process architecture**
- **High performance / low latency code base**
- **L3/L4 filter architecture**
- **HTTP L7 filter architecture**
- **HTTP/2** first
- **Service discovery** and **active/passive health checking**
- **Advanced load balancing**
- Best in class **observability** (stats, logging, and tracing)
- **Authentication** and **authorization**
- **Edge proxy**

Envoy architecture

Tetrate



Envoy's data plane API

Tetrate

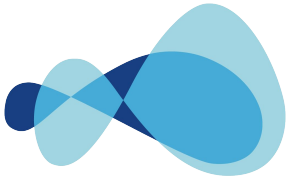
- Envoy is a **universal data plane**
- xDS == * Discovery Service (various configuration APIs). where Envoy retrieve configurations from E.g.,:
  - LDS == Listener Discovery Service
  - CDS == Cluster Discovery Service
  - RDS == Route Discovery Service
- Observability
  - Metrics Service
  - Access Log Service
  - Trace Service (OpenCensus, WIP)

Envoy's data plane API

Tetrate

- Envoy is a **universal data plane**
- xDS == * Discovery Service (various configuration APIs). where Envoy retrieve configurations from E.g.,:
  - LDS == Listener Discovery Service
  - CDS == Cluster Discovery Service
  - RDS == Route Discovery Service
- **Observability**
  - **Metrics Service**
  - **Access Log Service**
  - Trace Service (OpenCensus, WIP)

Observability



Tetrate

Metrics Service


```
service MetricsService {
  // Envoy will connect and send StreamMetricsMessage messages forever. It does not expect any
  // response to be sent as nothing would be done in the case of failure.
  rpc StreamMetrics(stream StreamMetricsMessage) returns (StreamMetricsResponse) {
  }
}

message StreamMetricsMessage {
  // Identifier data effectively is a structured metadata. As a performance optimization this will
  // only be sent in the first message on the stream.
  Identifier identifier = 1;


  // A list of metric entries
  repeated io.prometheus.client.MetricFamily envoy_metrics = 2;
}
```

Access Log Service (ALS)

Tetrate

```
service AccessLogService {
  rpc StreamAccessLogs(stream StreamAccessLogsMessage) returns (StreamAccessLogsResponse) {
  }
}

message StreamAccessLogsMessage {
  Identifier identifier = 1;

  // Wrapper for batches of HTTP access log entries.
  message HTTPAccessLogEntries {
    repeated envoy.data.accesslog.v2.HTTPAccessLogEntry log_entry = 1
        [(validate.rules).repeated .min_items = 1];
  }

  oneof log_entries {
    HTTPAccessLogEntries http_logs = 2;
  }
}
```

envoy-metric receiver in SkyWalking

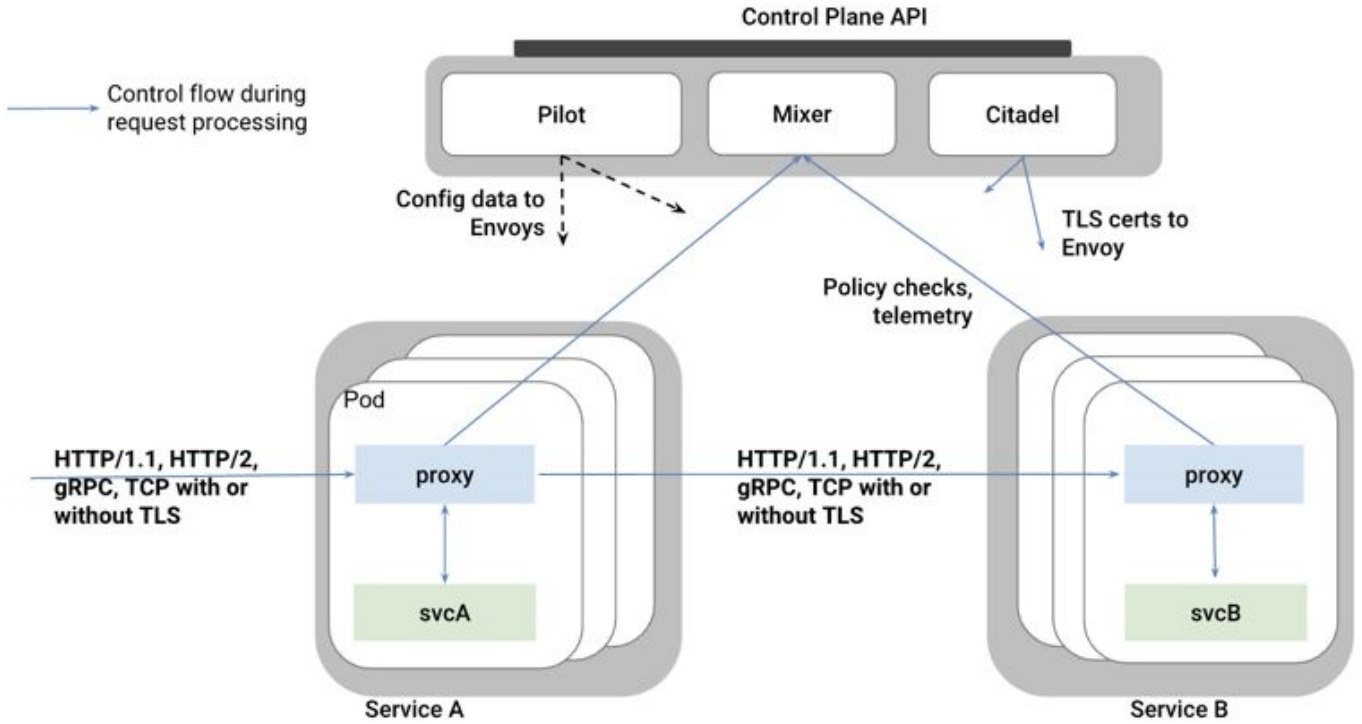
Tetrate

- Apache SkyWalking includes
  - envoy-metric receiver --- supports both Envoy Metrics service and Access log service

- Envoy configuration needed
  - Metrics Service: In bootstrap config
  - Access Log Service: In HTTP Connection Manager

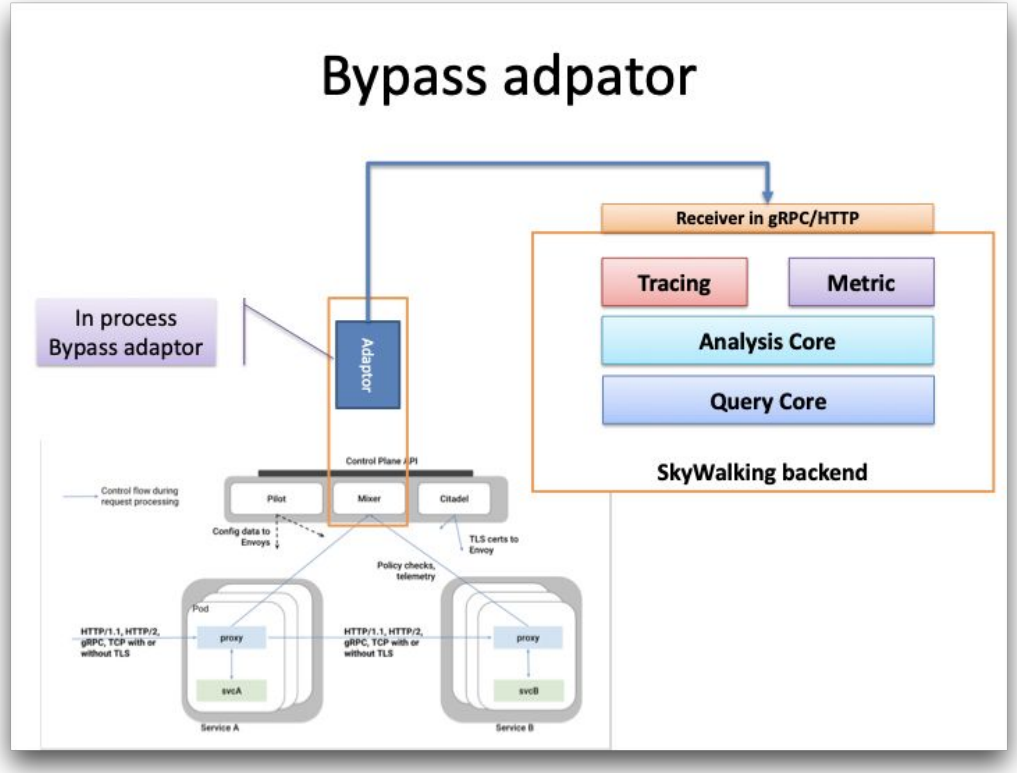
Today, Istio Mixer

Tetrate



Istio + Apache SkyWalking

Tetrate



Istio Mixer Performance Concerns

Tetrate

- Istio Mixer Envoy Filter Cost
  - More CPU of Envoy
  - Latency of RPC
- Istio Mixer Server Cost.
  - Unnecessary cost when using BYPASS
- Istio filter to Server
  - Complex attribute report protocol

What are the key metrics of mesh observability

- Topology
- Metrics of nodes and lines.
  - Nodes mean service.
  - Lines mean distributed dependency.

Options

1. Implement a SkyWalking filter in Envoy
2. Implement the Istio’s Attribute Protocol
3. Introduce ALS to SkyWalking

ALS of Sidecar and Ingress

Tetrate

- **KEY, How to identify Envoy role.**
- **WHY?**
- **Solution, using special words of Envoy node name.**
  - **router~ -> Ingress**
  - **sidecar~ -> sidecar**
    - **OUTBOUND in** upstreamCluster **-> Client side Sidecar**
    - **INBOUND in** upstreamCluster **-> Server side Sidecar**

ALS is only working well for observability when it is controlled by Istio.

Identify NODE by IP

K8s and Istio manage service mesh

Generate Topology line

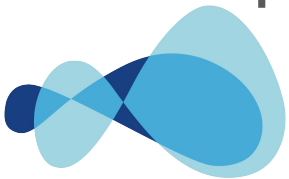
Tetrate

LINE of topology map has two detected points, client side and server side

- Ingress
  - Incoming request(server side): downstreamRemoteAddress -> downstreamLocalAddress
  - Outgoing request(client side): downstreamLocalAddress -> upstreamRemoteAddress
- Server Side sidecar
  - ingress -> sidecar(server side): Don’t generate, can’t identify downstreamRemoteAddress
  - sidecar -> sidecar(server side): downstreamRemoteAddress -> downstreamLocalAddress
- Client Side sidecar
  - sidecar(client side) -> sidecar: downstreamRemoteAddress -> upstreamRemoteAddress

Implementation Open Source in Apache SkyWalking

Project repo, https://github.com/apache/skywalking



Tetrate

Q&A

Tetrate

- Thanks for coming!
- https://envoyproxy.io/
- https://skywalking.apache.org/
- Talk to us if you need help getting started with Envoy and/or Apache SkyWalking.
- **Tetrate is hiring**: Contact us if you want to work on the problems and solutions based on Service Mesh.