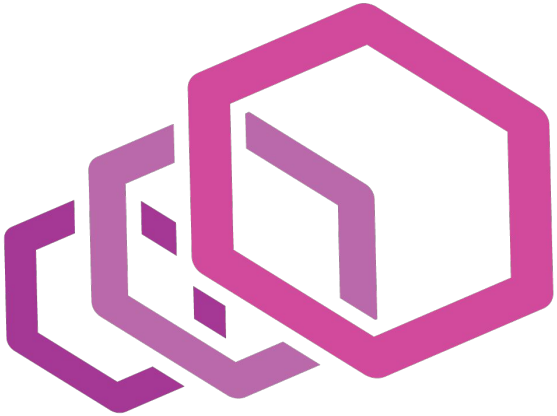
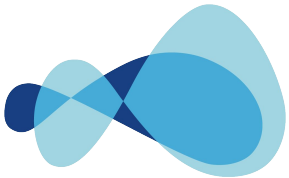


envoy

Envoy Intro



Tetrate

Lizan Zhou

06-25-2019

State of microservice networking in industry

Tetrate

- **Languages** and frameworks.
- **Protocols** (HTTP/1, HTTP/2, gRPC, databases, caching, etc.).
- **Infrastructures** (IaaS, CaaS, on premise, etc.).
- Intermediate **load balancers** (AWS ELB, F5, etc.).
- **Observability** output (stats, tracing, and logging).
- Implementations (often partial) of **retry**, **circuit breaking**, **rate limiting**, **timeouts**, and other distributed systems best practices.
- **Authentication** and **Authorization**.
- Per language **libraries** for service calls.

State of microservice networking in industry

Tetrate

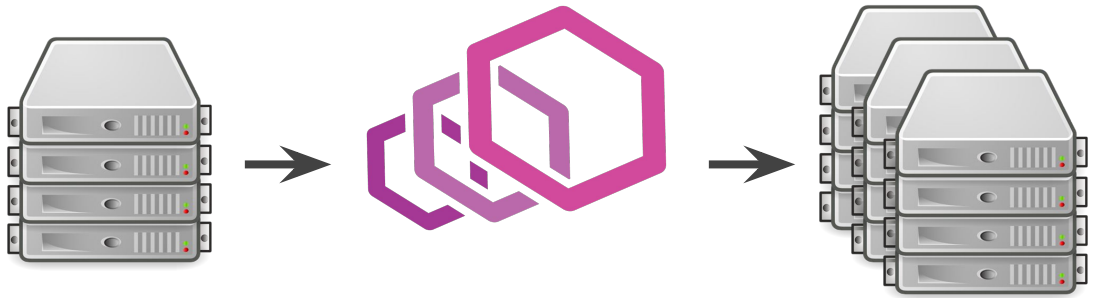
- Likely already in a **world of hurt** or rapidly approaching that point.
- **Debugging** is difficult or impossible (each application exposes different stats and logs with no tracing).
- **Limited visibility** into infra components such as hosted load balancers, databases, caches, network topologies, etc.).
- Multiple and **partial implementations** of circuit breaking, retry, and rate limiting (If I had a $ for every time someone told me that retries are “easy” …).
- Furthermore, if you do have a good solution, you are likely using a **library** and are locked into a particular technology stack essentially forever.
- Libraries are incredibly **painful to upgrade**. (Think CVEs).

What is Envoy?

Tetrate

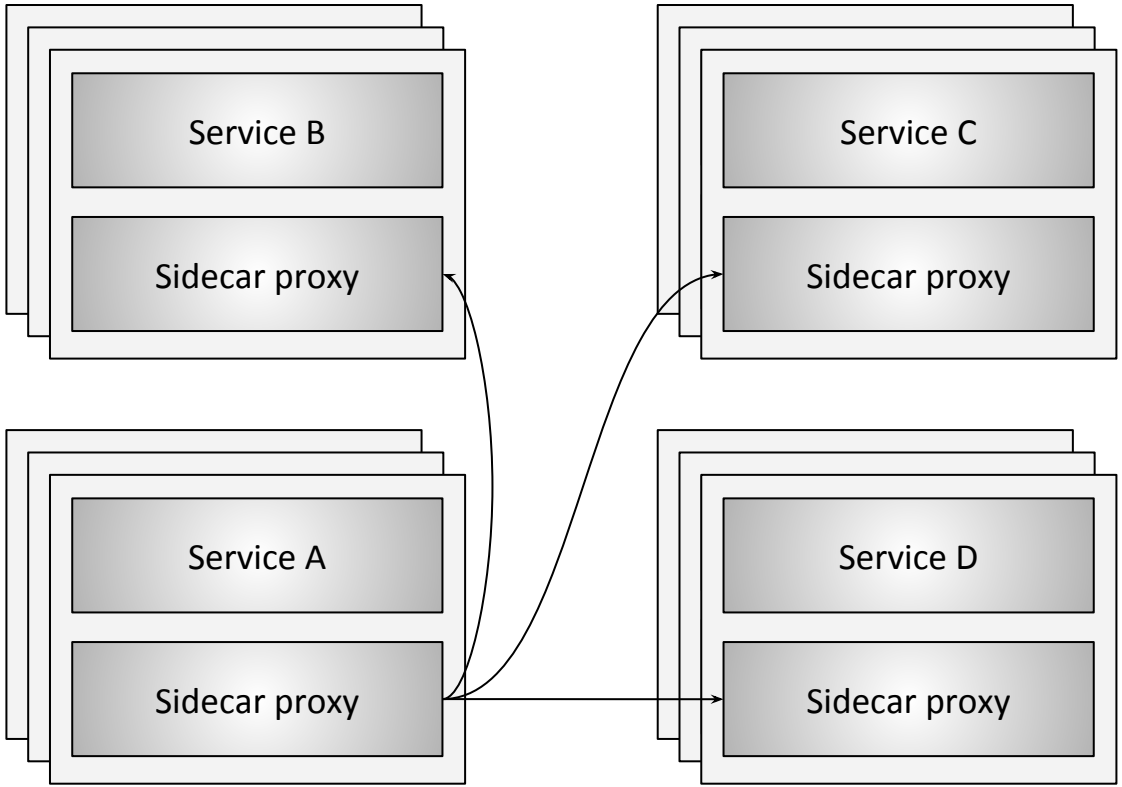
The network should be transparent to applications.

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



Service mesh refresher

Tetrate



Envoy design goals

- **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**

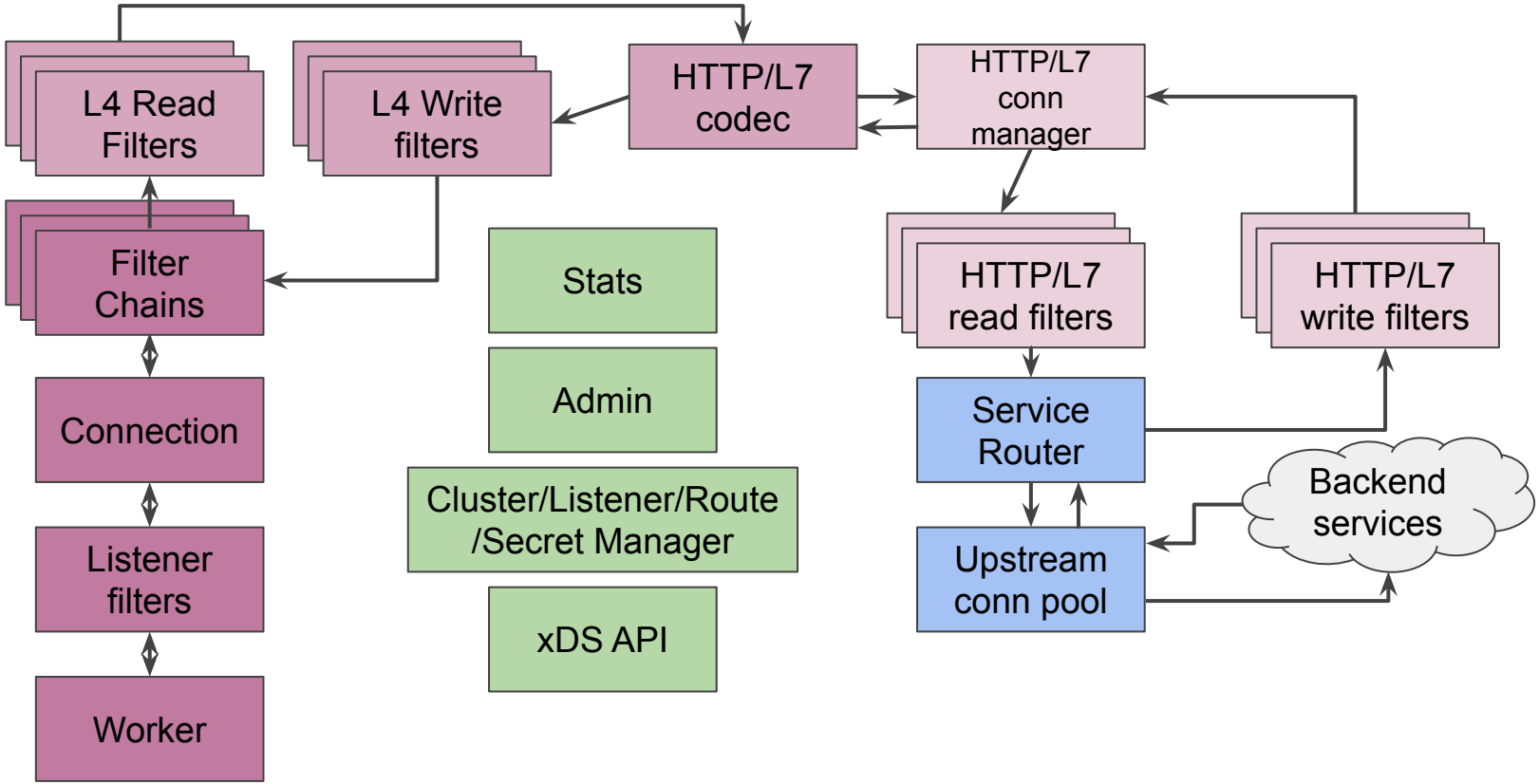
Envoy config management via xDS APIs

Tetrate

- Envoy is a **universal data plane**
- xDS == * Discovery Service (various configuration APIs). E.g.,:
  - LDS == Listener Discovery Service
  - CDS == Cluster Discovery Service
- Both gRPC streaming and JSON/YAML REST via proto3!
- Central management system can control a fleet of Envoys **avoiding per-proxy config file hell**
- **Global bootstrap config** for every Envoy, rest taken care of by the management server
- Envoys + xDS + management system == **fleet wide traffic management distributed system**

Envoy architecture

Tetrate



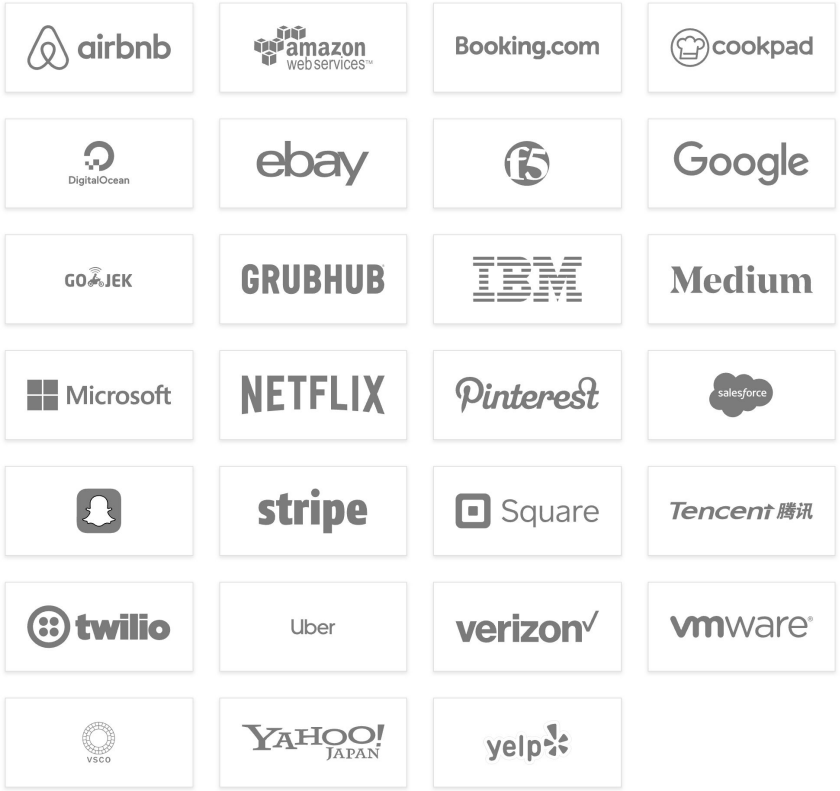
Extension and pluggability

Envoy is designed to have multiple extension point. E.g.:

- L4/L7 filters
- Access loggers
- Tracers
- Health checkers
- Transport sockets
- Retry policy
- Resource monitors
- Stats sink

Envoy Adoption

Tetrate



Adoption in China

NetEase, Inc.

网易云
163yun.com

Envoy 在网易内部的实践

曾宇星 网易资深架构师

目录

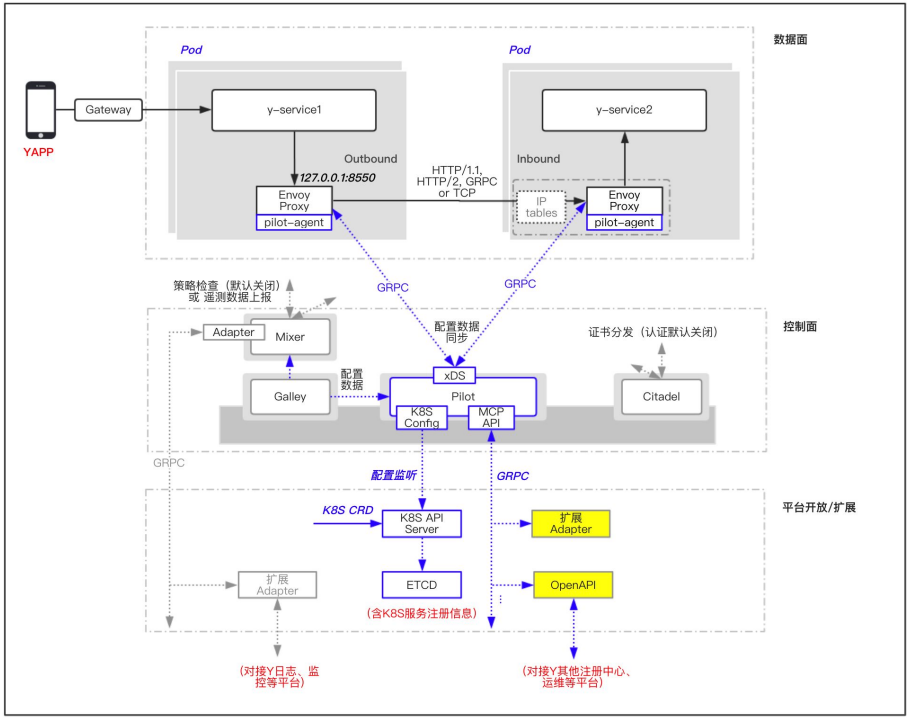
现状

- 以Nginx作为流量出口代理
- 与Consul紧密结合
- 服务发现基本实现业务无感知

挑战

- 数据面能力弱，无控制面
- 维护成本高
- 开源社区偏离

- 整体基于Envoy+Istio方案
- Outbound指向Sidecar
- Inbound可配置流量拦截
- 控制面以Pilot为核心
- 注册中心以K8S原生方式
- 通过MCP机制扩展



- 适配原有服务调用配置

127.0.0.1:8550/proxy/servicea/path/xxx

127.0.0.1:8550/hash/serviceb/path/xxx

proxy 和 hash 为prefix，可配置

- Istio 适配

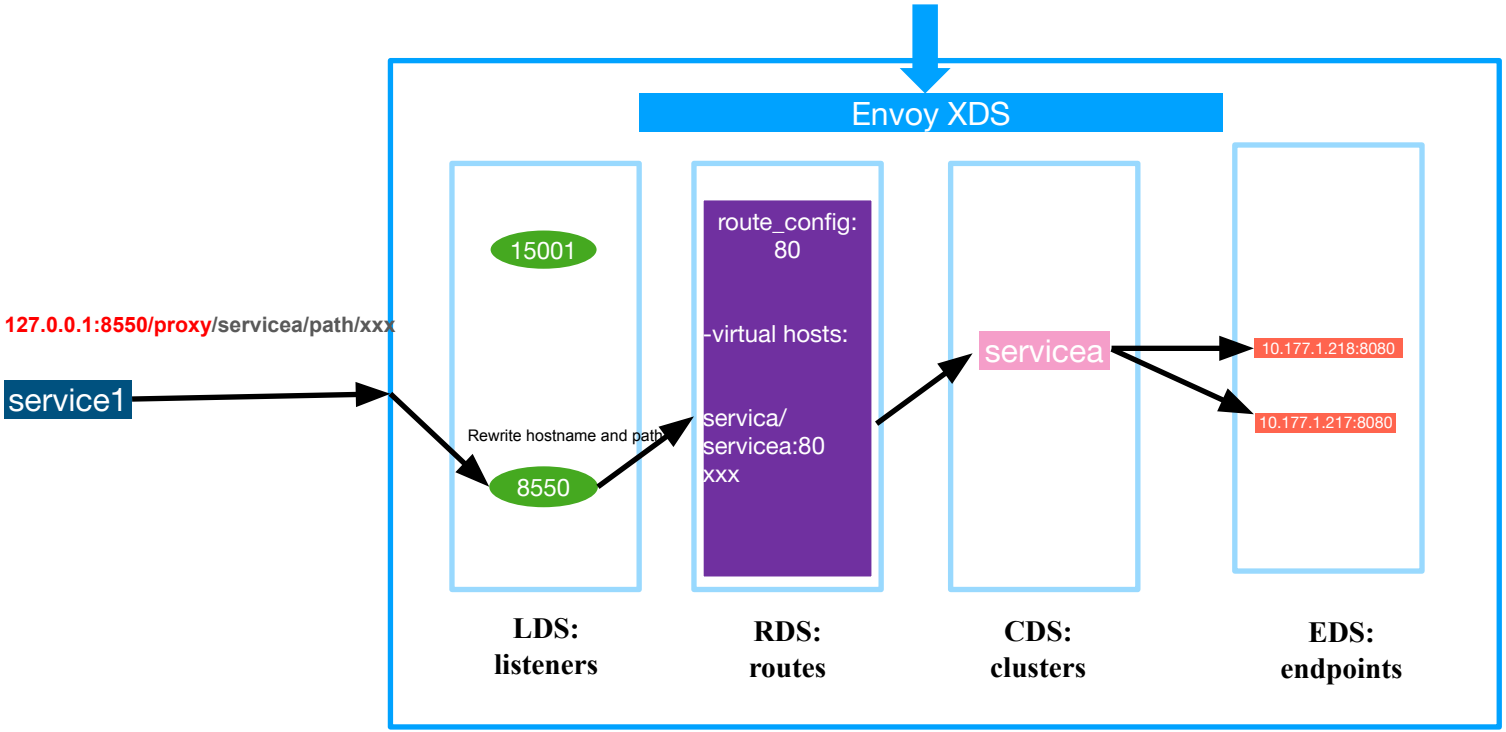
默认下发8550的LDS至Envoy，其中route_config_name 配置为80

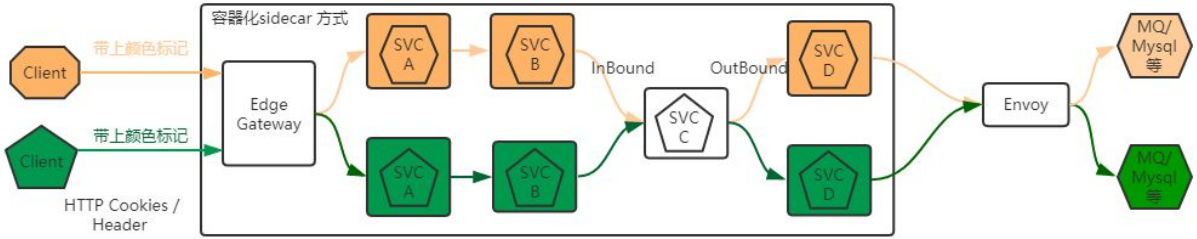
- Envoy适配

在http_connection_manager 下增加url_transformer 配置选项，针对类似URL 127.0.0.1:8550/proxy/servicea/path/xxx 更改http请求的host 和 path 分别为 “servicea“ 和 “path/xxx”

```
listeners:
- name: listener_0
  address:
    socket_address:
      address: 0.0.0.0
      port_value: 8550
  filter_chains:
  - filters:
    - name: envoy.http_connection_manager
      typed_config:
        "@type": type.googleapis.com/envoy.config.filter.network.http_connection_manager.v2.HttpConnectionManager
        stat_prefix: ingress_http
        url_transformer:
          - prefix: "/proxy"
          - prefix: /p
          - prefix: /hash
          - prefix: /chash
          - prefix: /least

        route_config:
          name: local_route
          virtual_hosts:
          - name: service_b
            domains: ["*"]
            routes:
            - match:
                prefix: "/world"
              route:
                cluster: cluster_01
                timeout: "0.005s"
        http_filters:
        - name: envoy.router
```

Rewrite Host 和 Path 、默认路由适配



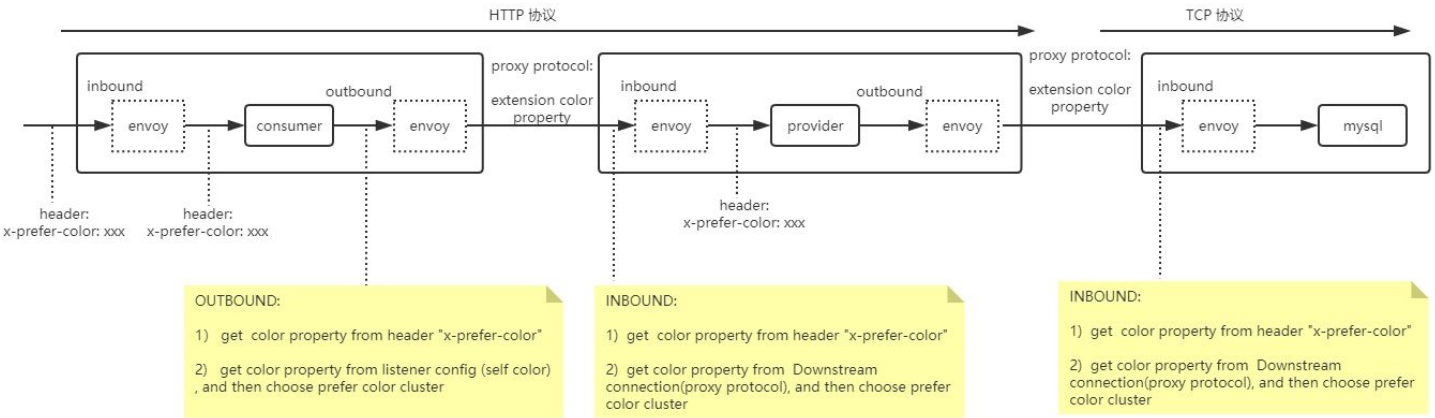


需求：

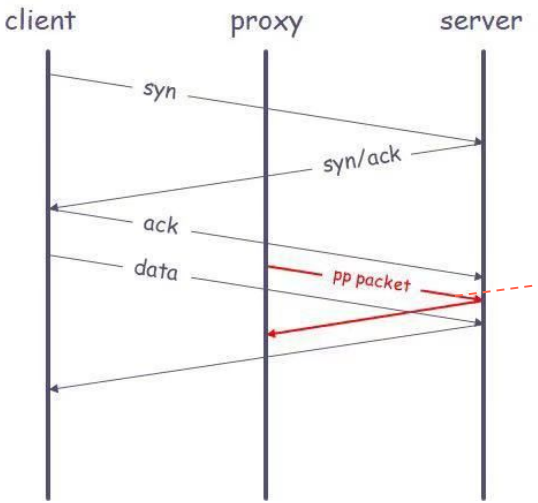
- **流量染色**
- **流量穿梭**
- **业务无感知**

```yaml
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  name: ratings
spec:
  hosts:
  - ratings
  http:
  - match:
    - sourceLabels:
        corlor: red
    route:
    - destination:
        host: ratings
        subset: red


apiVersion: networking.istio.io/v1alpha3
kind: DestinationRule
metadata:
  name: ratings
spec:
  host: ratings
  subsets:
  - name: red
    labels:
      color: red
  - name: green
    labels:
      color: green
```

控制面通过 sourceLabels 下发定向路由，环境较多时配置太麻烦，而且不能解决 subset 不存在时cluster 降级问题



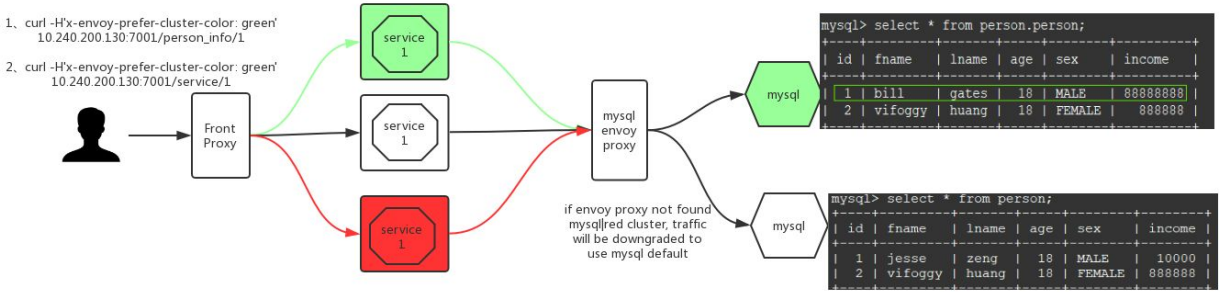
- **支持L4&L7**
- **支持流量染色**
- **支持流量穿梭**
- **应用无侵入**




```
// get the downstream color
auto down_stream_color = downStreamConn->getPreferClusterColor();
auto listener_config_color = transport_socket_options->getDefaultDownStreamColor();

if (!down_stream_color.empty()) {
  send_color = down_stream_color.data();
  ENVOY_LOG(debug, "using downStream connection color: {}", down_stream_color);
} else if (!listener_config_color.empty()) {
  send_color = listener_config_color;
  ENVOY_LOG(debug, "using listener config default color: {}", send_color);
}

if (!send_color.empty()) {
  // add extesion color property
  // defined in <type>
  proxy_data->tlv.type = 0x30; // PP2_TYPE_NETNS;

  int len = send_color.length();
  len = std::min(len, 16);
  proxy_data->tlv.length = ::htons(len);

  strncpy(reinterpret_cast<char*>(proxy_data->tlv.value), send_color.data(), len);

  proxy_data->length += len + PP2_TLV_HEADER_SIZE;

  uint8_t* proxy_data_buf = reinterpret_cast<uint8_t*>(proxy_data.get());
  ////
  memmove(&proxy_data_buf[28], &(proxy_data->tlv), sizeof(Network::ProxyProtocol::pp2_tlv));
} else {
  ENVOY_LOG(debug, "proxy protocol : not color data--");
}
```

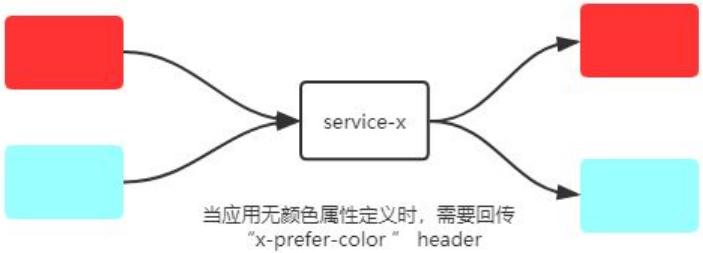
Sidecar 通过header (x-prefer-color) 、Proxy protocol 协议的扩展字段传递 color 属性

多环境治理-example




```python
@app.route('/service/<service_number>')
def hello(service_number):
  return ('Hello from behind Envoy (service {})! hostname: {} resolved'
          'hostname: {}\n'.format(os.environ['SERVICE_NAME'], socket.gethostname(),
                                  socket.gethostbyname(socket.gethostname())))


@app.route('/person_info/<id>')
def get_person_info(id):
  db = MySQLdb.connect("mysql-envoy-proxy", "demo", "demo123", "person", charset='utf8', port=13306)
  cursor = db.cursor()
  sql = "SELECT `fname`,`lname`,`age`,`sex`,`income` FROM person WHERE id = %s" % (id)
  try:
    cursor.execute(sql)
    results = cursor.fetchall()
    for row in results:
      fname = row[0]
      lname = row[1]
      age = row[2]
      sex = row[3]
      income = row[4]
      return ('Person information: id: {}, firstName: {}, lastName: {}, age: {},'
              'sex: {}, income: {}\n'.format(id, fname, lname, age, sex, income))
  except:
    return ("Error: unable to fetch data")
```

流量穿梭：



注意：类似分布式追踪应用回传x-request-id相关头，流量穿梭功能需要业务应用回传x-prefer-color 头

共创云上精彩世界

Q&A

Tetrate

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