Service Mesh with gRPC and xDS

—

Megan Yahya, Product Manager @ Google

Agenda

- gRPC in microservices
- gRPC in service mesh
- gRPC without service mesh integration
- Limitations with side-car proxies
- Extensibility with interceptors
- High industry adoption

gRPC in Microservices

gRPC is very popular in microservices based applications.

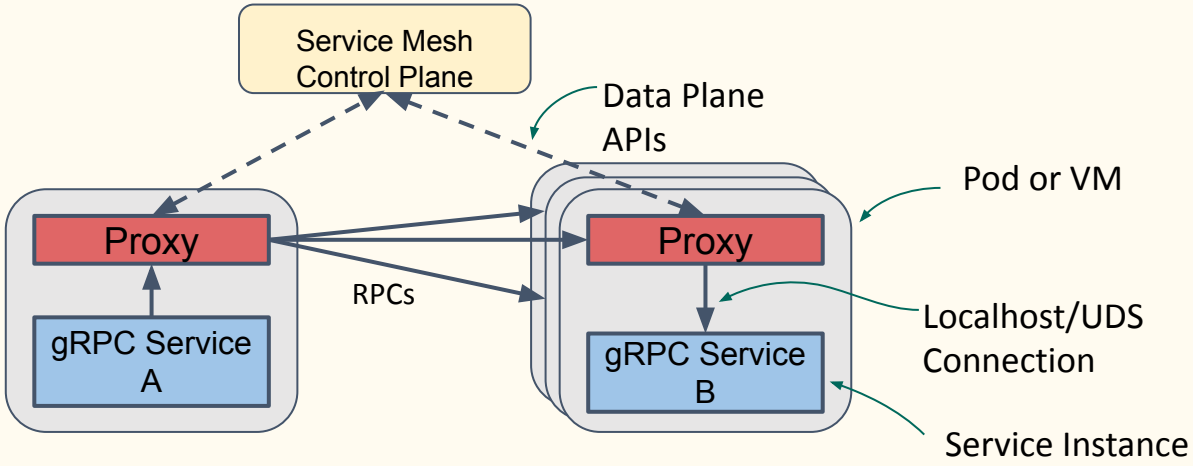
- High performance and efficiency
- Available in multiple languages
- Easy backward/forward compatibility with Protocol Buffers
- Features like deadline, cancellation and metadata
- Extensibility with interceptors
- High industry adoption

gRPC in Service Mesh

gRPC lacks service mesh functionality.

- Service discovery - only DNS resolver
- Load balancing - only pick-first and round-robin policy
- Security - user managed with TLS
- Observability - user managed with OpenCensus

gRPC Without Service Mesh Integration

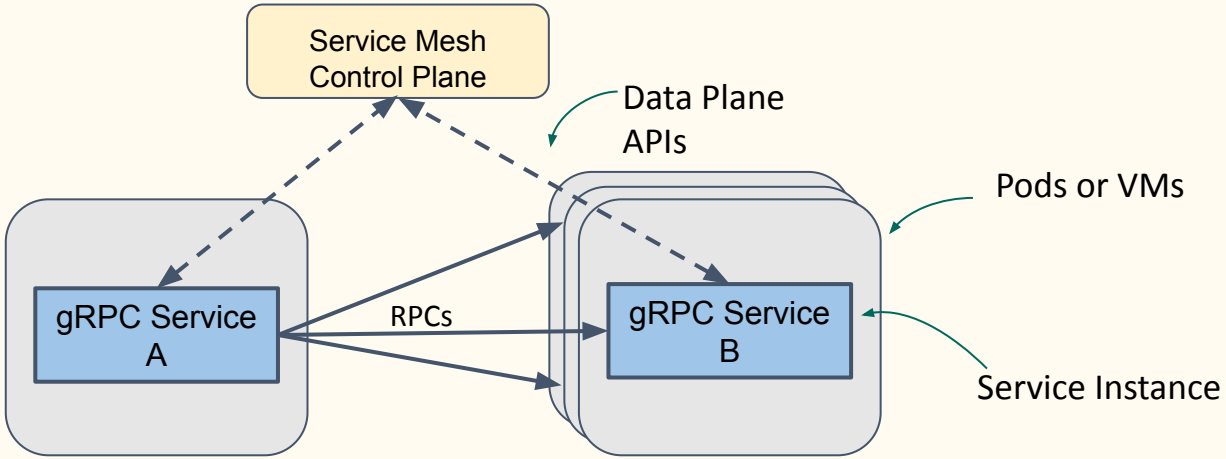


- Sidecar proxies get service mesh policies from the control plane.
- gRPC applications use DNS lookup and send requests to the virtual IP of the service.
- Sidecar proxies intercept requests, apply service mesh policies and route accordingly.

Limitations With Sidecar Proxies

- Performance overhead
- CPU cost overhead
- Added complexity due to traffic interception
- Overhead of managing additional binaries in the data plane
- No lifecycle management of proxies
- No end-to-end security

gRPC Service Mesh - Proxyless!



- gRPC applications get service mesh policies directly from the control plane.
- No sidecar proxies. Services talk to each other directly.

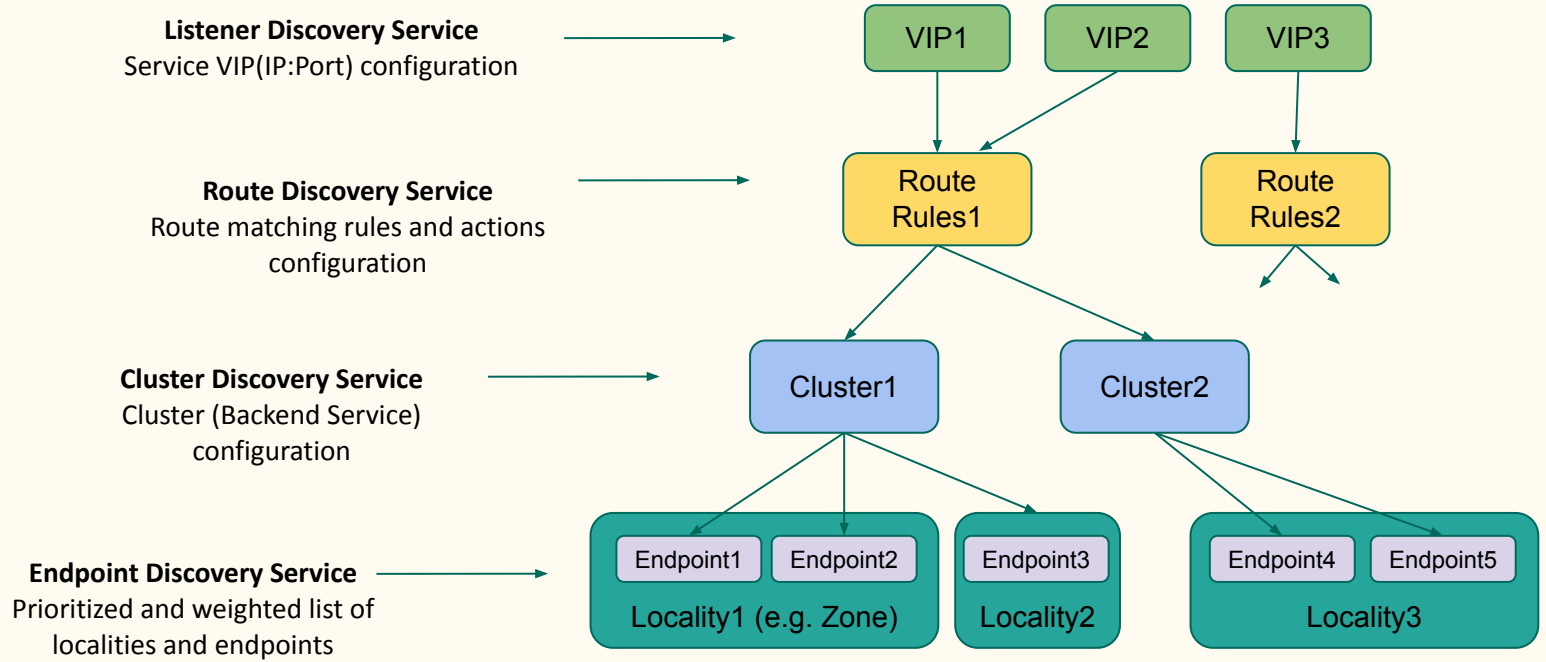
xDS APIs

- xDS is a set of data plane APIs - APIs between mesh control plane and the proxies.
- Developed for Envoy proxy - a popular proxy used in many open source and proprietary service meshes.
- xDS is open, extensible and has strong community support.

xDS is the right choice for service mesh integration in gRPC.

xDS Explained

- It's all about discovering!
- (x)Discovery Service - Listener, Route, Cluster, Endpoint, Health, Secret etc.



xDS in gRPC

- Build a gRPC channel with ‘xds’ resolver scheme.
  - Example: ManagedChannelBuilder.forTarget("xds:///foo.myservice")
- Provide a bootstrap file with xDS server address, credentials and node info via an environment variable.

That’s it!

- Easy to adopt xDS.
- Easy to mix proxied and proxyless deployments.

Plan ahead and write mesh-ready gRPC applications!

Limitations

- Feature gap with Envoy
  - But, gRPC is catching up
- Ecosystem around Envoy filters and observability tools
  - gRPC has interceptors and OpenCensus integration
- gRPC applications need some changes
  - xDS dependency
  - Bootstrap
- Limited language support
  - C++, Java, Go, Python, PHP, Ruby, C# and Node.js

Current Status

- First released in v1.30.0 in June'20
- Features currently supported as of v1.35.0
  - LDS, RDS, CDS and EDS
  - Load reporting via LRS
  - Weighted locality picking and round robin endpoint LB within the locality
  - Route matching with path and headers field
  - Traffic splitting between weighted clusters
- In development
  - xDS v3 support
  - Timeout, circuit breaking and fault injection
  - gRPC server xDS integration
  - Security with mTLS
- Checkout xDS features in gRPC for latest updates.

Control Planes For Proxyless gRPC Applications

- Google Cloud's Traffic Director service mesh control plane
  - Global load balancing with request routing based on geographical proximity, health and capacity of backends.
  - Automatic routing of overflow traffic across regions based on health and capacity of backends.
  - Automatic failover of traffic across regions based on health and capacity of backends.
  - Scalable centralized gRPC health checks of backends in the mesh.
  - Automatic endpoint discovery when backend instances come and go.
  - Demand driven auto-scaling of backends.
  - Works on GCE and GKE.
- Istio
  - Experimental support with no documentation. See this test for an example
- go-control-plane
  - This issue is requesting support for proxyless gRPC.

Resources

- gRFCs
  - xDS load balancing design
  - xDS traffic splitting and routing design
  - xDS timeout, circuit breaking, fault injection
  - xDS-enabled servers
  - xDS based security
- xDS features in gRPC by release
- Data plane vs. control plane, Concepts and terminology
- Envoy xDS APIs, xDS support in gRPC
- Traffic Director, a production ready control plane for proxyless gRPC

Thank you!

Questions?