KubeCon | CloudNativeCon
North America 2020
Virtual

Proxyless Service Mesh with gRPC

Menghan Li, Google

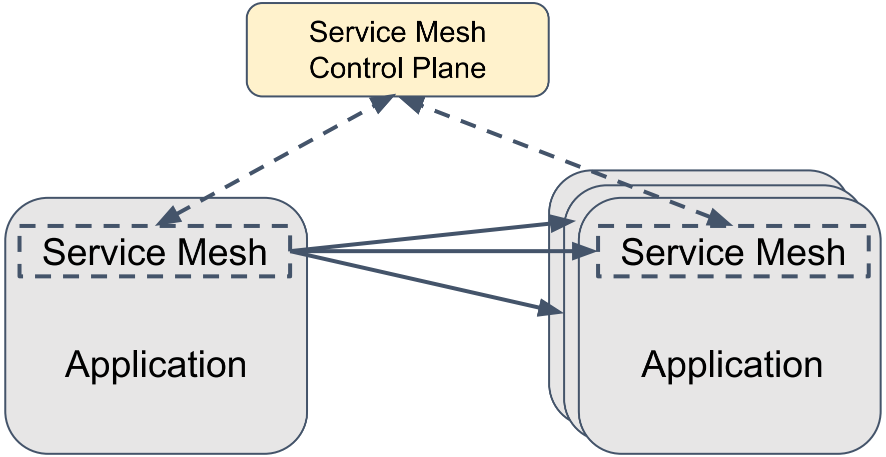
gRPC

- High performance, open source
- High industry adoption
- Features
  - Connection management, multiplexing, bidi-streaming, flow control
  - Deadlines, cancellation, metadata
  - Plugins, interceptors etc.
- Multi-language, multi-platform
- Works great with Protocol Buffers and other wire formats
- Awesome framework for microservices

Before Service Meshes

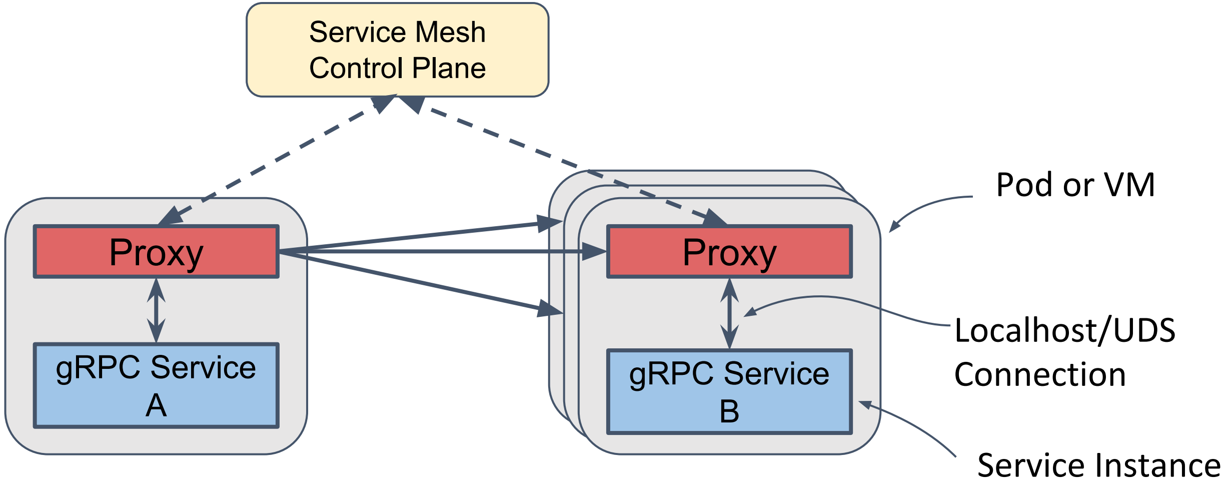
- Before Service Mesh integration in gRPC
  - Service Discovery - only a DNS name resolver
  - Traffic management - pick-first and round-robin load balancing
  - Security - TLS
  - Observability - no built-in solution
- Advanced features require custom plugins
  - Resolver/Balancer interfaces
  - Stats APIs

What is a Service Mesh?



- Infrastructure layer to control how different parts interact
- Solves complexity of microservices architecture

Proxy based Service Mesh



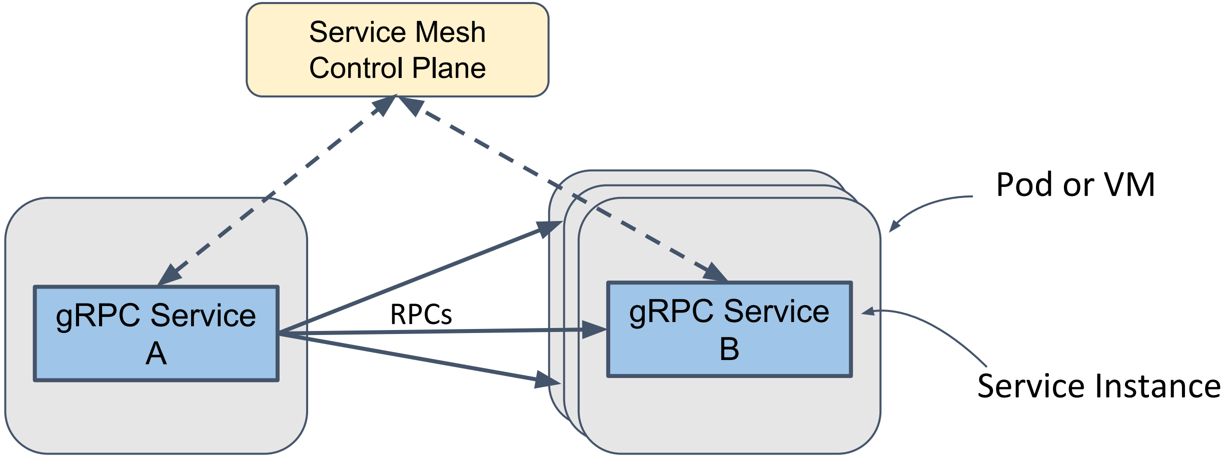
- Sidecar proxies get service mesh configuration from the control plane
- Requests are intercepted by the proxies

Problems with proxies

- Performance overhead
  - Potential bottleneck
- Lifecycle management of proxies
- No end-to-end security

Support Service Meshes in gRPC

Proxyless gRPC Service Mesh



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

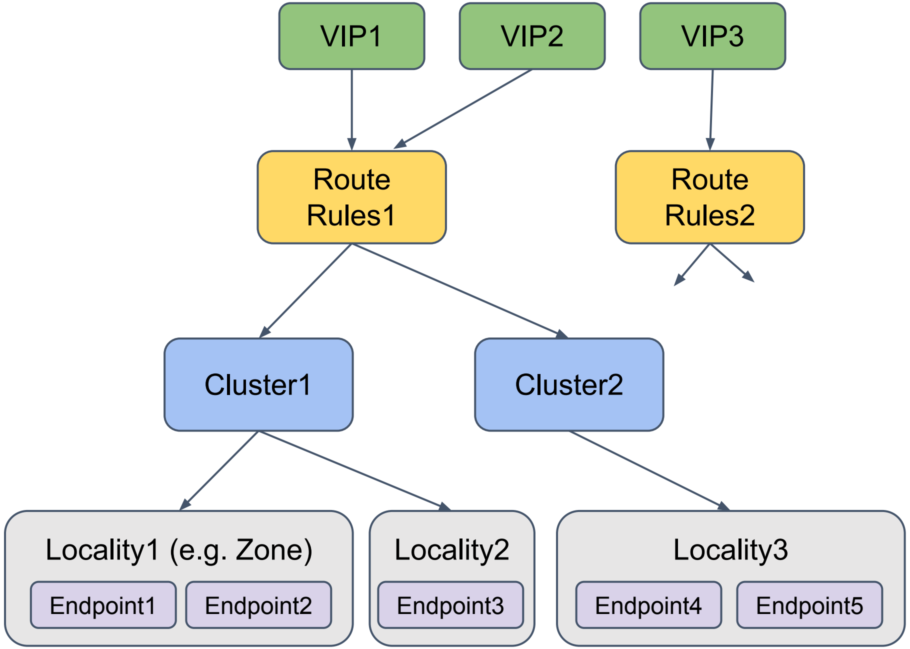
Which Service Mesh

- Choose the right data plane APIs - APIs between mesh control plane and the applications (proxies).
- Attributes: open, extensible, strong community support and widely used.
  - Works with any control plane that supports such data plane APIs.
  - Helps prevent vendor lock-in.

xDS APIs - the wildly popular data plane APIs used by Envoy proxy and istio.

Overview of xDS APIs

- Endpoint
  - A server instance
  - Health status
- Locality
  - A group, a zone
  - Priority (demo)
- Cluster
  - A deployment
    - Different services
    - Different versions of the same service
  - Load balancing
- Route
  - Request routing
    - Path matching, header matching (demo)
    - Traffic splitting (demo)
    - Retry, timeout
- Listener/VIP
  - Start of any traffic from proxy's point of view
  - Doesn't apply very well in gRPC



Enabling xDS in gRPC

- Pull in the xds dependencies
  - E.g. in gRPC-Go, `import _ “google.golang.org/grpc/xds”`
- Build a gRPC channel with “xds” resolver scheme
  - E.g. in gRPC-Go, `grpc.DialContext(ctx, “xds:///foo.myservice”, …)`
- Provide a bootstrap file with xDS server address and configuration
  - Set `GRPC_XDS_BOOTSTRAP` env variable to the bootstrap file

Limitations

- Feature gap
    - Active development going on
- Deploy bootstrap file
- Ecosystem (observability) around Envoy
    - gRPC has interceptors and OpenCensus integration
    - Observability work in progress
- Must recompile applications
    - Not a problem with CI/CD

The resolver scheme is per channel - Easy to migrate and mix’n’match proxied and proxyless deployment.

Current status

Released v1.33 (October 20, 2020)

- xDS client with LDS, RDS, CDS and EDS, Load reporting via LRS
  - Support xDS v2 and v3
- Weighted locality picking and round robin endpoint LB within the locality
- Route matching with path and headers field
- Traffic splitting between weighted clusters

What’s next?

- Timeout, circuit breaking, fault injection
- gRPC server side xDS integration
- Security features like service-to-service mTLS
- Observability

Resources

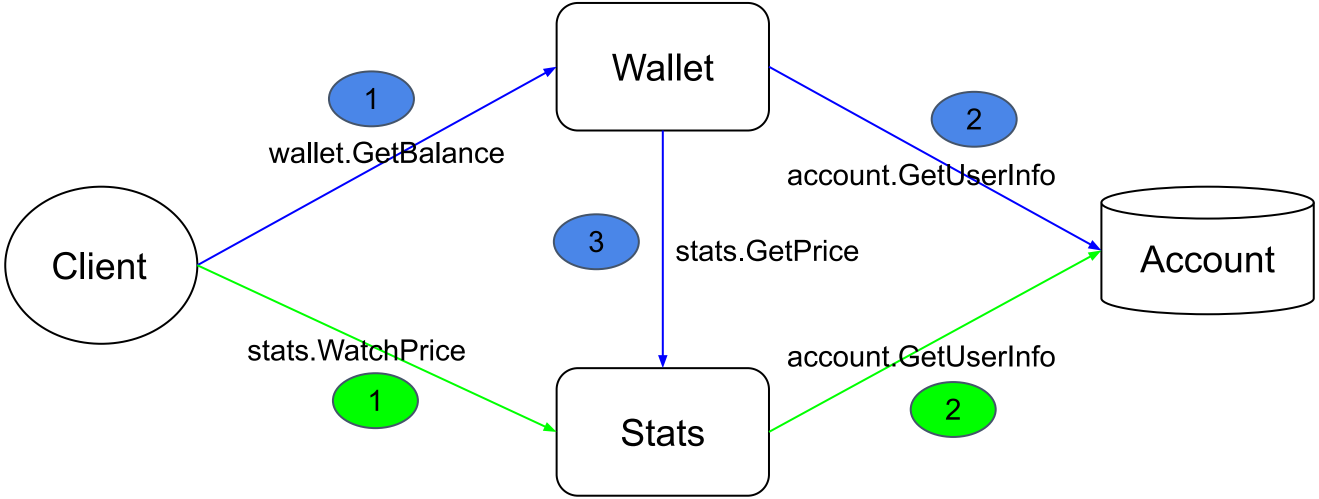
- gRFCs
  - xDS load balancing design
  - xDS traffic splitting and routing design
  - xDS timeout support
  - xDS circuit breaking
- xDS features in gRPC by release
- Envoy xDS APIs, Universal Data Plane APIs
- Data plane vs. control plane, Concepts and terminology
- Traffic Director

Demo

- Application: gRPC Wallet
- Control plane: Traffic Director, Google Cloud's managed control plane for service mesh.
    - Traffic Director uses xDS to communicate with gRPC clients.

gRPC Wallet

- A wallet for gRPC-Coin
- Services
  - Account Service - database for user id and information
  - Stats Service - price for gRPC-Coin
  - Wallet Service - number of gRPC-Coins for each user



Demo: traffic splitting

- Client connected to “wallet.grpcwallet.io”
- Two deployments of Wallet service
    - wallet-v1
    - wallet-v2
- Split traffic for RPC “FetchBalance”
    - v1: 60%
    - v2: 40%
- Useful when migrating from v1 to v2
    - Gradually increase the traffic to v2

Demo: header matching

- Client connected to “stats.grpcwallet.io”
- Two deployments of Stats service
  - stats
  - stats-premium
    - Premium users receive price update with higher frequency
- Match header for user information
  - {“membership”: “premium”}
    - route to stats-premium
    - verified with the Account service

Demo: failover

- Client is in “us-central”
- Two server localities
  - “us-central”, will be priority 0
    - because they are in the same zone as the client
  - “us-west”, will be priority 1
- All traffic go to “us-central”
- When “us-central” is down, traffic will go to “us-west”

Thanks

- Contact
  - menghanl@google.com
  - github @menghanl
- gRPC (https://grpc.io/community/)
  - grpc-io mailing list
  - grpc/grpc gitter

KEEP CLOUD NATIVE EVERYWHERE

KubeCon | CloudNativeCon

North America 2020

Virtual

Title

- body

Title

- body

Title

- body

What is xDS

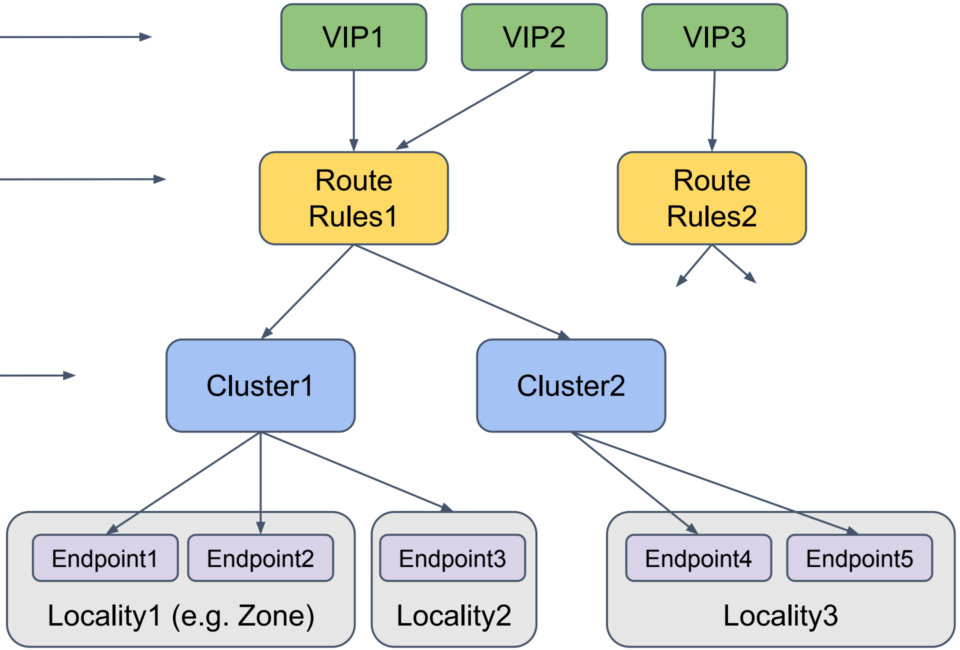
- (x) Discovery Service - Listener, Route, Cluster, Endpoint, Secret etc

Listener Discovery Service
Service VIP(IP:Port) configuration

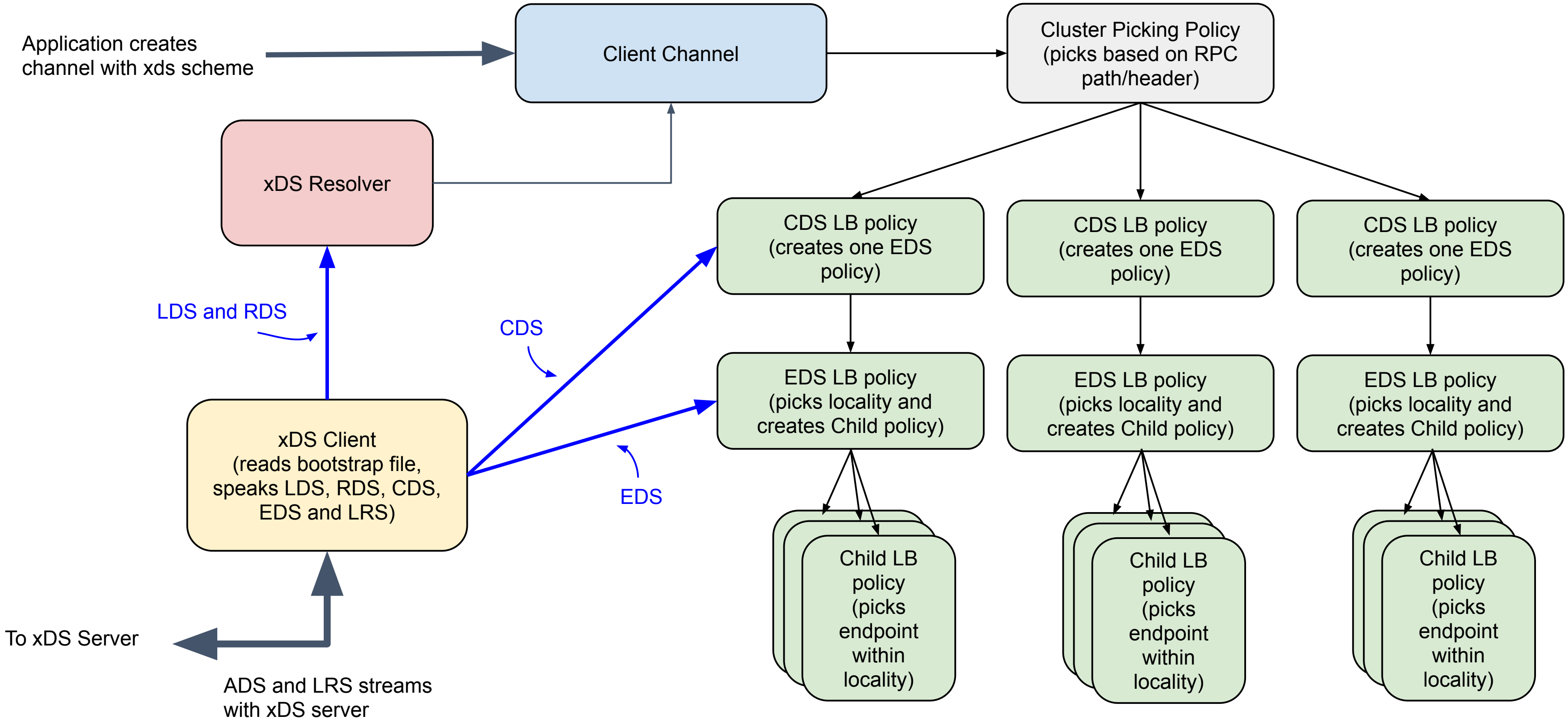
Route Discovery Service
Route matching rules and actions configuration

Cluster Discovery Service
Cluster (Backend Service) configuration

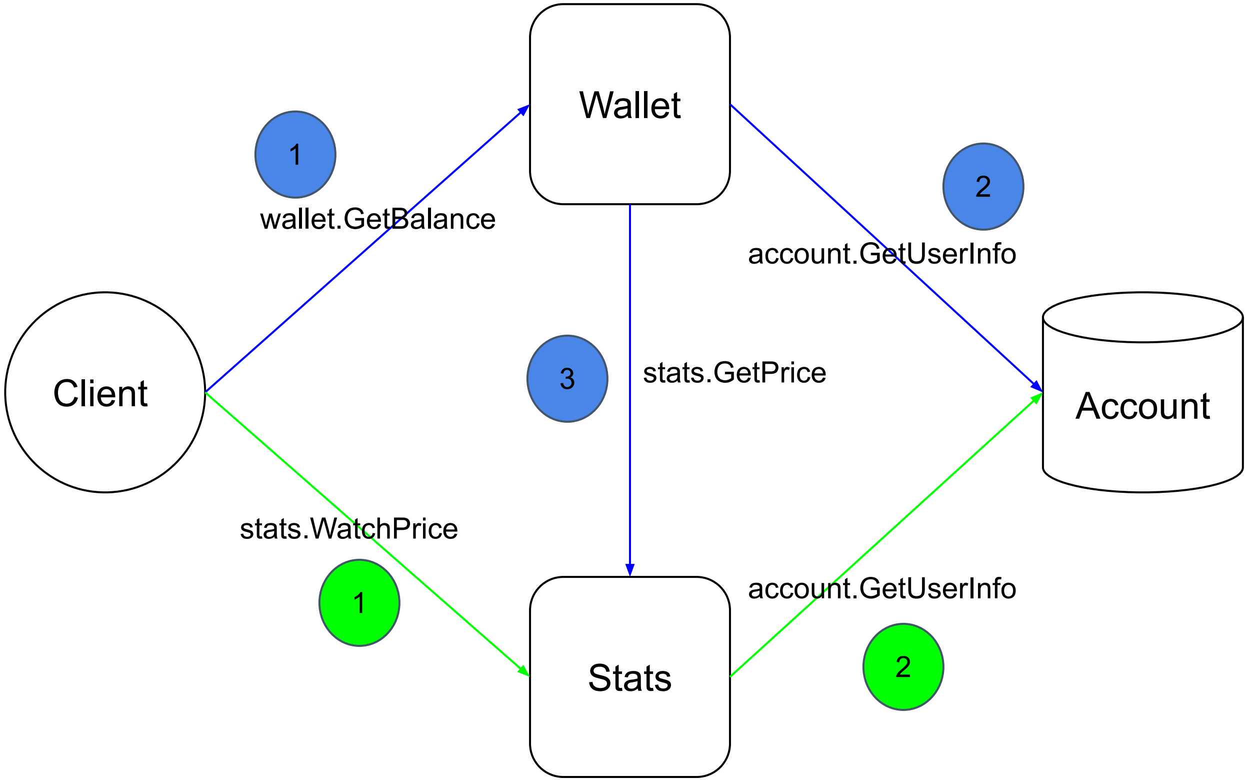
Endpoint Discovery Service
Prioritized and weighted list of localities and endpoints



xDS architecture in gRPC


Application creates channel with xds scheme
Client Channel
Cluster Picking Policy (picks based on RPC path/header)
xDS Resolver
CDS LB policy (creates one EDS policy)
CDS LB policy (creates one EDS policy)
CDS LB policy (creates one EDS policy)
LDS and RDS
CDS
EDS LB policy (picks locality and creates Child policy)
EDS LB policy (picks locality and creates Child policy)
EDS LB policy (picks locality and creates Child policy)
xDS Client (reads bootstrap file, speaks LDS, RDS, CDS, EDS and LRS)
EDS
Child LB policy (picks endpoint within locality)
Child LB policy (picks endpoint within locality)
Child LB policy (picks endpoint within locality)
To xDS Server
ADS and LRS streams with xDS server

gRPC Wallet


Wallet
1
wallet.GetBalance
2
account.GetUserInfo
Client
3
stats.GetPrice
Account
stats.WatchPrice
1
account.GetUserInfo
Stats
2