Istio Service Mesh

Istio Service Mesh

What is Service Mesh?

What is Service Mesh?

— Network services for **microservices**

— Policy-based, enforces behaviors

— Manages topology, changing conditions: load, configuration, resources

— Spans clusters/clouds (Mesos clusters, VMs, Cloud, K8s, Consul)

— Provides discovery, load balancing, failure recovery, and observability

What is Service Mesh?

- Addressable infrastructure layer
- Developer-driven, services-first network
- Takes network concerns out of service / application code (e.g., resiliency, circuit breakers, etc.)
- Declaratively define network behavior by policy: node identity, routing, balancing, and traffic flow
- Language and framework agnostic support for observability and tracing

Istio Service Mesh

Network Services

Istio Service Mesh

Network Services

- Traffic Management
- Observability
- Policy Enforcement
- Service Identity & Security

Traffic Management

- Control traffic flow
  - REST/RPC calls between services
  - Level 7 and Level 4
- Reliable & Robust
  - Load balancers
  - Health-checks
  - Circuit breakers

Observability

- Understanding topology of service dependencies
- Visualize traffic flow between services
- Tracing
- Log aggregation
- KPIs/Metrics

Policy Enforcement

- mTLS and Zero trust networking
- Control interaction between services
- Enforce access policies
- Ensure resources fairly distributed
- Keeps policy concerns out of application code
- Services have verifiable identity
- Rate limiting based on identity of service traffic

TODO look at section in Istio book value of Service Mesh

Istio Service Mesh

Architecture

Service Mesh - Architecture

— Management Plane
— Control Plane
— Data Plane
— Ingress Gateway
— Egress Gateway

Service Mesh - Architecture

Management Plane

- Governance - Multi-cluster and multi-mesh
- Integration with existing clouds and environments
- Federation
- Intelligent services

Control Plane

— Policy, config, routes, rules and platform integration
— Configs Set of Side car service proxies (per pod)
— Does not intercept or route any network traffic
— Config and config propagation

Data Plane

- Uses configured policies, routes, rules and platform integration
- Intercepts and routes network traffic
- Encryption (TLS), Authentication, and Authorization
- Health checks, routing, load balancing, and observability
- Service Discovery

Service Mesh - Architecture

- Service meshes consist of service proxies (data plane), which are configured by the control plane
- Traffic is intercepted using iptables rules per pod
- Service Mesh: Combination of policies managed by control plane plus data plane that implements policy behavior plus service deployments
- Istio turns disparate microservices into an integrated service mesh

Service Mesh versus Client Libs

— a

— b

— c

Why service meshes are being adopted

— a

— b

— c

Service Mesh Capabilities

- a
- b
- c

Service Mesh Landscape

- a
- b
- c

Service Mesh Ecosystem

- API gateways
- L4 to L7
- East west vs. North South
- HAProxy, Traefik, NGINX, Envoy, Ambassador, GKE, EKS, AKS,

Service Mesh Landscape

- a
- b
- c

Service Mesh - Layer 5

— a

— b

— c

mTLS explained

— mTLS

——

TLS explained

— TLS headers

— SNI

——

Origins of Istio

— a

— b

— c

Istio Architecture

- a
- b
- c

Service Mesh - Observability

- a

Installing Istio

— a

— b

— c

Service Proxy

— a

— b

— c

Envoy

— gRPC

— HTTP 2

— REST to gRPC

Sidecar Proxy

- Injection
- Manual Injection
- Ad Hoc

Security and Identity

— a

— b

— c

SPIFFE

- Istio implements the Secure Production Identity Framework for Everyone (SPIFFE) specification
- Spec wrt issuing identities
- Istio creates X.509 certificate
- Istio uses certificate's subject alternative name (SAN)
- SAN is a URI for the service
- Uses pod service account name
- URI: spiffe://$ClusterName/ns/$Namespace/sa/

Istio Config

— All
— Connection models
— JWT support

Istio RBSC

— Config

— A

— B

Pilot

— Config

— A

— B

Pilot Mesh Config

— Config

— A

— B

Pilot Networking Config

— Config

— A

— B

———

Pilot Service Discovery

— Config

Traffic Management

— Config

— A

— B

Traffic Management Config

— ServiceEntry
— DestinationRule
— VirtualService
— Gateway

Telemetry

— Config

— A

— B

Deployments

— Green / Blue

— Canary

— Advanced Canary

Resiliency

- Client Side Load Balancing
- Circuit breaking
- Auto Retry
- Request timeouts
- Injecting faults to test resiliency

Ingress and Egress

— A

— B

— C

Mixers

— Policies

— Attributes

— Adapters

— Config

Telemetry

— Adapters
— Metrics
— Logs
— Visualization

Trouble shooting

— A

— B

— C

Multi-clusters

— Single Mesh

— Mesh Federation

Multi-clusters

- Flat networking not needed since Istio 1.1
- EDS (Pilot API that mirrors Envoy)
  - Local only config
- TLS SNI routing
  - inter-cluster connectivity
- Split Horizon

Istio Multi-clusters

- Istio determines locality based on names for routing
- Uses Kube labels and annotations for cluster awareness
- Supports geo locality, cluster locality for LB
- Proxy assigns services cluster labels
- Use GitOps to sync names and config for multiple control planes

Use Cases Multi cluster

— HA
— Cross-provider
— Deployment strategies (A/B, Canary, Blue/Green)
— Migration