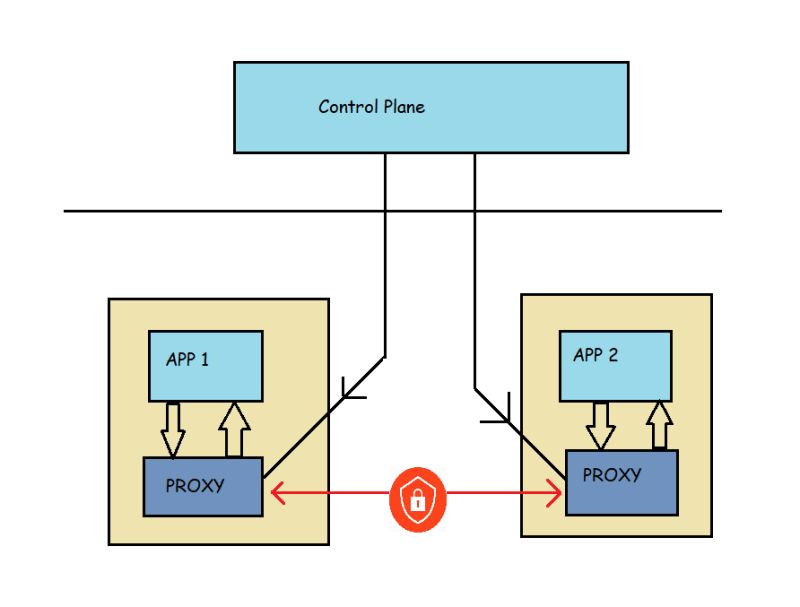
**What is Service Mesh Anyway?**

* A service mesh is a programmable framework that allows you to observe, secure and connect microservices.
* It doesn’t establish connectivity b/w microservices, but instead has policies and controls that are applied on top of existing network to govern how microservice’s interact.
* A service mesh ultimately, shifts implementation responsibilities out of the application and moves them to the network. This is accomplished by injecting behaviour and controls with int application that are applied to the network
* This is how you can accomplish things such as metrics collection, communication tracing and secure communication without changing applications themselves

**How will Service Mesh Work?**

* Basic workflow by injecting a sidecar container 
* Following are the list of behaviours commonly found in a service mesh implementation
  + Traffic shaping with dynamic routing controls b/w services
  + Resiliency support for service communications such as circuit breakers, timeouts and retries
  + Observability of traffic b/w services
  + Tracing of communication flows
  + Secure communication b/w services

**The Service Mesh Ecosystem**

* Envoy:
  + This Envoy proxy is an open-source project create at Lyft.
  + This is an edge and service proxy that was custom built with the complexities and challenges of cloud native applications.
  + While Envoy itself doesn’t constitute a service mesh, it is a key component of service mesh eco system
* Istio:
  + This is an opensource project co-founded by IBM, Google and Lyft in 2017.
  + Istio makes it possible to connect, secure and observe your microservices
  + Istio itself builds upon many other open-source projects such as
    - Envoy
    - Kubernetes
    - Jaeger
    - Prometheus
  + Istio control plane extends the k8s API Server and utilizes Envoy proxy for client side proxies
  + Istio supports mutual TLS (mTLS) authentication communication b/w services, traffic shifting, mesh gateways, monitoring and metrics with Prometheus & Grafana, As well as custom policy injection
* Consul Connect:
  + This is service mesh developed by HashiCorp
  + This extends existing product offering Consul, which has service discovery as primary feature & built in features such as key value store, health checking & service segmentation (for secure TLS communication b/w services)
  + Consul Connect is an open source project with Hashicorp as predominant contributor
  + Consul Connect uses Envoy as sidecar proxy and Consul Server and control plane for programming side cards
  + Consul Connect includes secure mTLS support b/w microservices and observability with Prometheus and Grafana Projects.
* Linkerd:
  + The Linkerd service mesh project is an opensource as well as CNCF project with focus on providing an ultra-lightweight mesh implementation.
  + Linkerd has key capabilities of a service mesh including observability using Prometheus and Grafana, secure mTLS communication and support for traffic shift.
  + The client-side proxy used with Linkerd was developed specifically within the Linkerd project & was written in Rust.
* App Mesh: This is a cloud service hosted by AWS to provide service mesh with application-level networking support for compute services within AWS such as ECS, Fargate, EC2 and EKS

**Why Istio?**

* Istio is a mature service mesh implementation that allows you break down the complexity of distributed cloud native deployments by taking complex functionality out of the application code & moving it into network
* Istio is most feature rich and it’s also built to serve the enterprise use cases
* Istio has following features
  + Automatic metrics and network tracing collected b/w services within the mesh, as well as inbound and outbound network communication with external clients and services
  + Advanced rule-based traffic routing & control with automatic load balancing for HTTP, gRPC, WebSocket and TCP traffic across service versions
  + Automatic mTLS management for secure service-to-service communication using strong identity base authentication & authorization
  + Built-in service communication resiliency with retries, circuit breaking failover and fault injection

**Istio Architecture**

* Architecture
* The components of the control plane
  + Pilot:
    - This is essential component that programs Envoy sidecars. This converts Istio defined APIs into Envoy-Specific configurations
    - Responsible for service discovery with in service mesh
  + Citadel: This provides critical security capabilities within the Istio service by managing certificates and provide strong service identities
  + Galley: Galley manages istio’s configuration, It validates, ingests, processes and distributes istio configuration to other control-plane services
  + Envoy: Istio use Envoy proxy for sidecars as well as gateways

