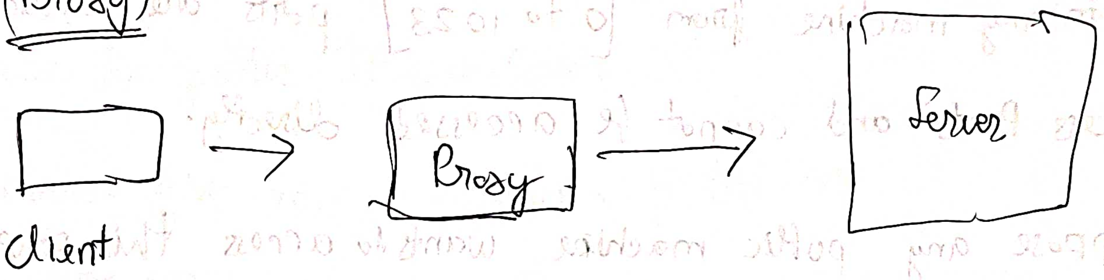
(1.&2) (Proxy V/s Reverse Proxy)

Proxy

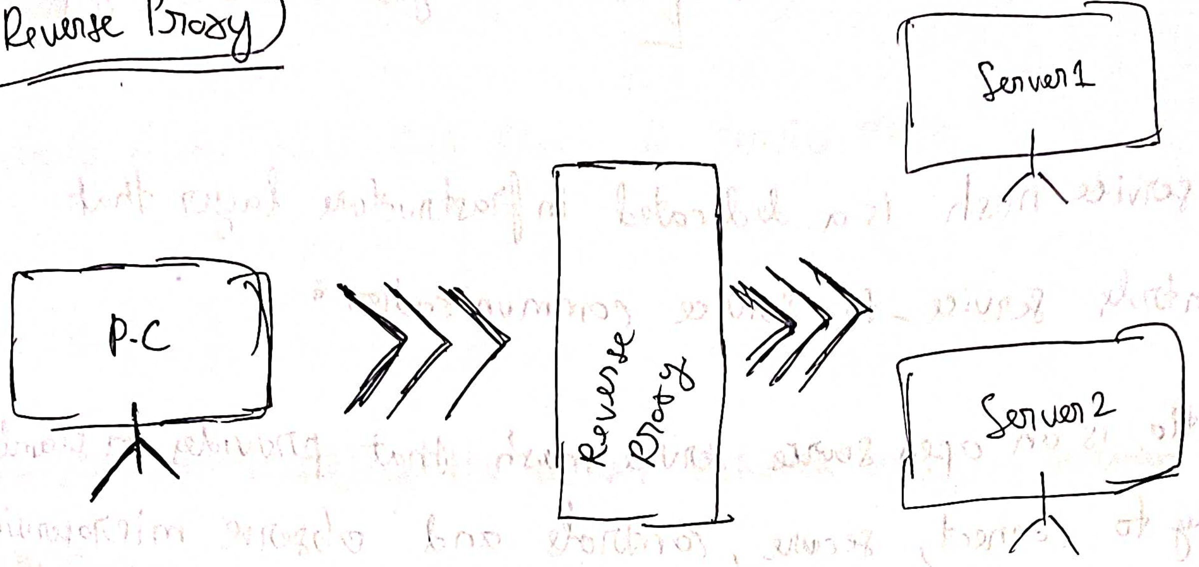


{Same as Low Level Proxy Design Pattern}

Use Cases

* Caching
* Anonymity
* Logging
* Block Sites
* Microservices ⟹ This is a fairly new use case added. If our application is running, we can add another container in the pod, it acts as a side-car proxy. Whichever request is coming, lets says HTTP 1.0 is ~~xx~~ coming we can upgrade it to HTTP 2.0 and then send it to application.

(Reverse Proxy)



In proxy the server does not know who the client is.

In reverse proxy, the client does not know who the server is

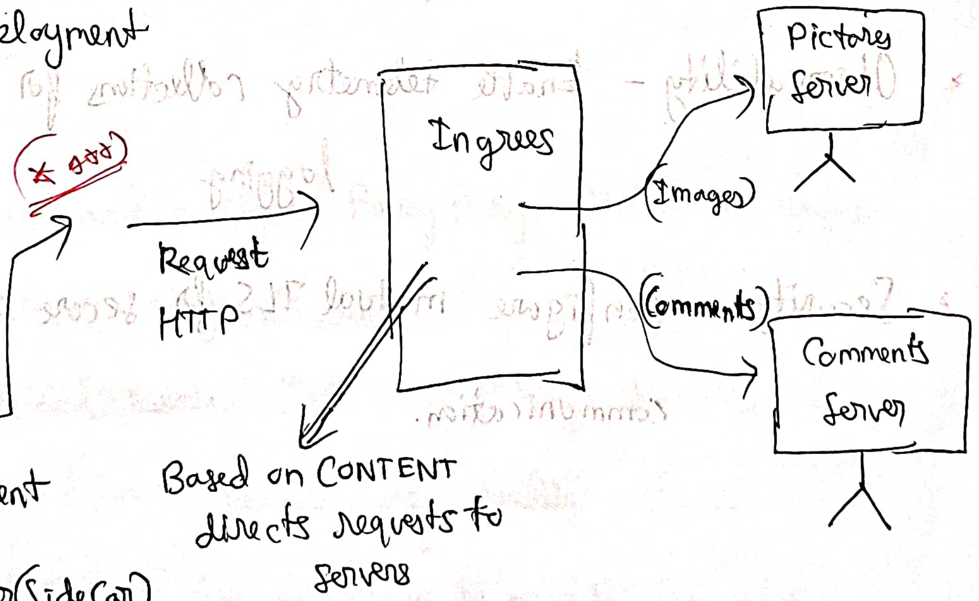
(Example)

{CANARY Deployment}

The Reverse Proxy / Side Car Container tells the new version to serve 8% of customer traffic and rest is server by the actual deployment

USE CASES

* Caching
* Load Balancing
* InGress
* Canary Deployment
* Microservices → (SideCar)



Service Mesh [Both Proxy & Reverse Proxy]

A service mesh is a dedicated infrastructure layer that controls service-to-service communication.

Istio is an open source service mesh that provides a seamless way to connect, secure, controle and observe microservices. It uses (Envoy proxy) as its (data plane) proxy which intercepts all network traffic between microservices.

[Envoy Proxy container will be injected to each of pods of the namespace]

[Kubectl label namespace <your-namespace> istio-injection=enabled]

[Istio Configuration]

* Traffic Management - Define Istio destination rules, and gateways to manage traffic routing
* Observability - Enable telemetry collections for monitoring and logging
* Security - Configure mutual TLS for secure service-to-service communication.

(Two main components of Service Mesh)

[Controle Plane and Data Plane in Service mesh]

[Data Plane :-]

Responsible for handling the actual network traffic between microservices

* Envoy proxies are deployed as sidecars alongside each microservice instance

* They intercept and manage all inbound and outbound traffic

* they can enforce policies like load balancing, retries timeouts, and circuit breaking

[Control Plane] :-

Responsible for managing and configuring the data plane Proxies.

1. Pilot → Manages and configures Envoy Proxy. Distributes config updates to envoy sidecars.

2. Citadel → Issues and Rotates TLS certs among services

3. Galley → Ensures Pilot has done its work correctly.

4. Mixer → Collect metrics, logs and traces to provide observability.

{Benifits of Proxy}

* Anonymity → Server does not know who the client is.

* Caching

* Blocking unwanted Sites → ~~Might~~ ISP might have some rules not to allow access to specific site. But using VPN, we can trick the proxy into saying.

"We want to actually access site B and not site A, which is unrestricted, but after the proxy is bypassed, the packet has the destination i.p. Address and mac Address of the site A only".

* Geofencing → Only Specific Content is visible to Specific users. Proxy checks where the request is coming from and what the Proxy whats.

(Ex.). Someone from India wants to acess a Netflix content, which is not available /allowed. Proxy will block it from accessing that server.