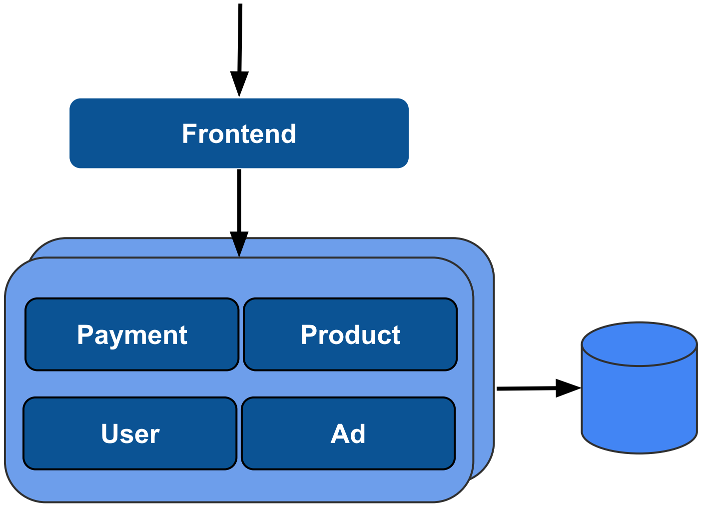
Application Networks

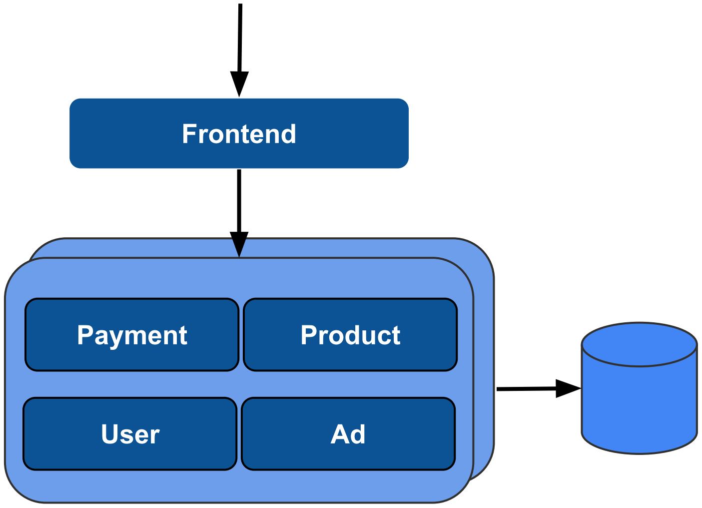
Xiangfeng Zhu
CSE461

From monolith to microservices

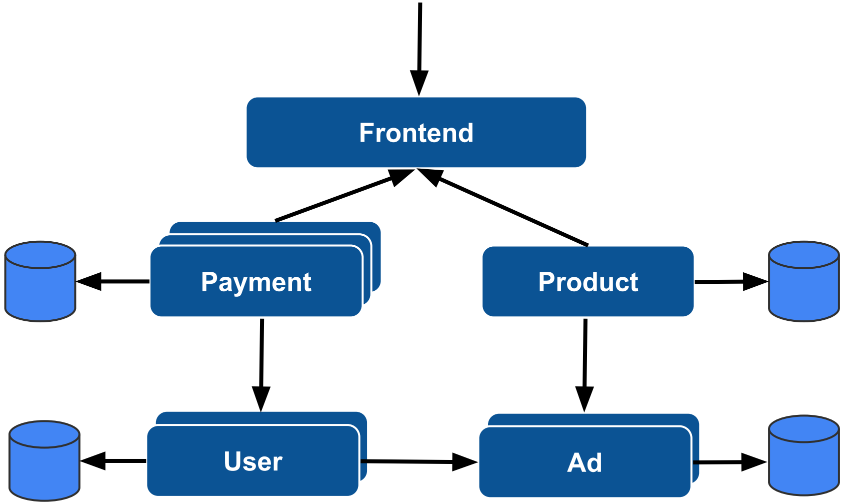


Monolithic Application

From monolith to microservices

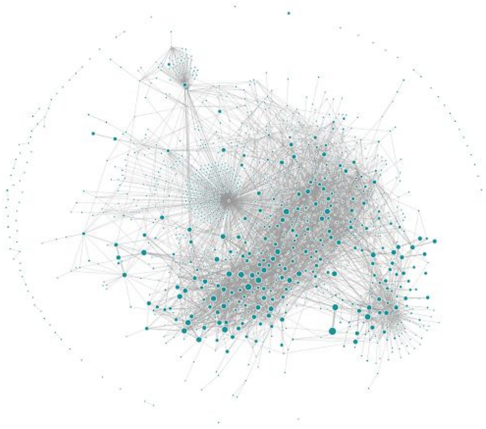


Monolithic Application



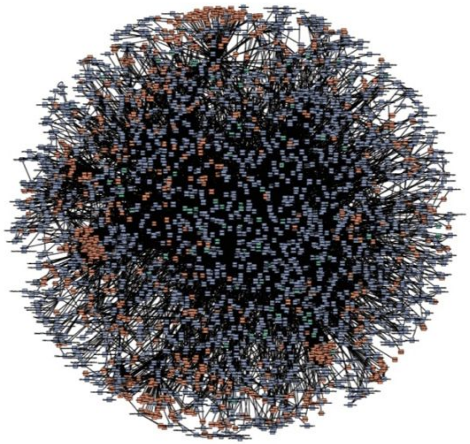
Microservices

From monolith to microservices



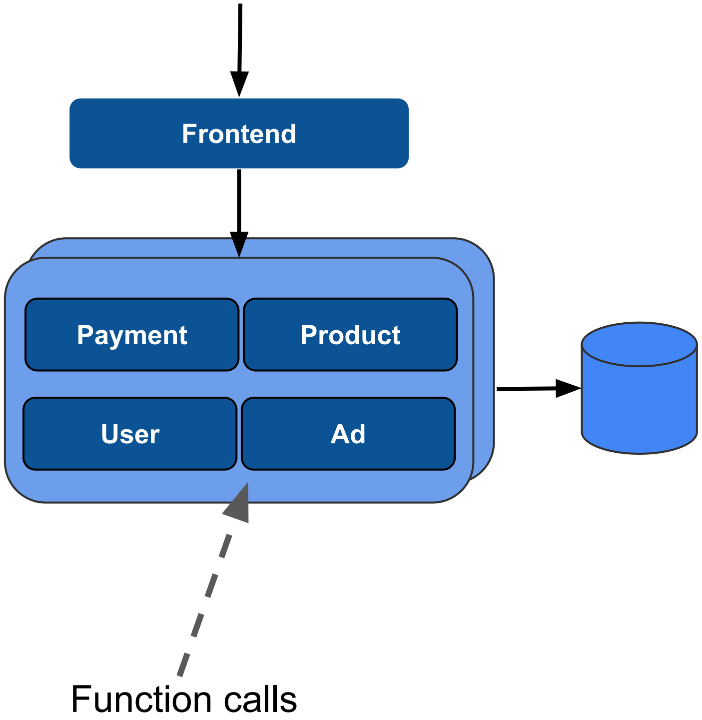
Uber

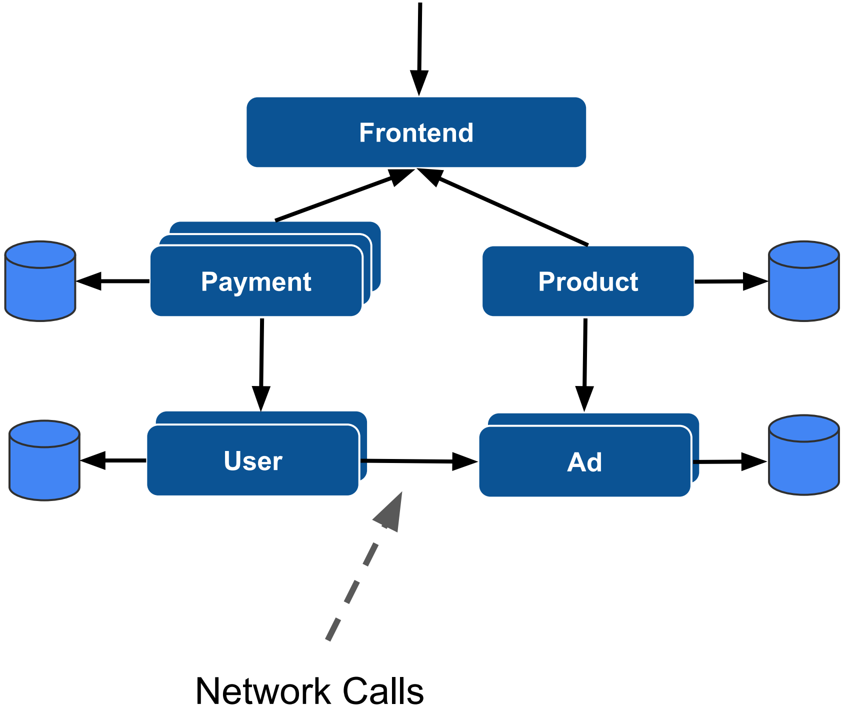
NETFLIX



amazon

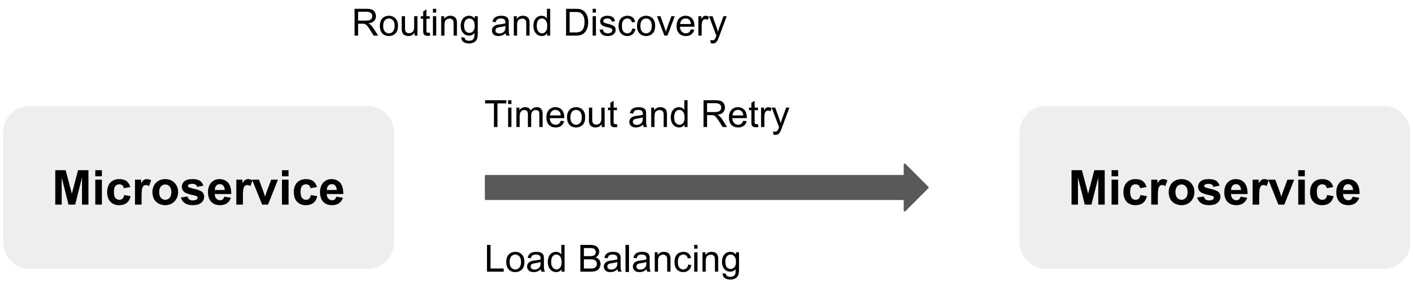
From monolith to microservices



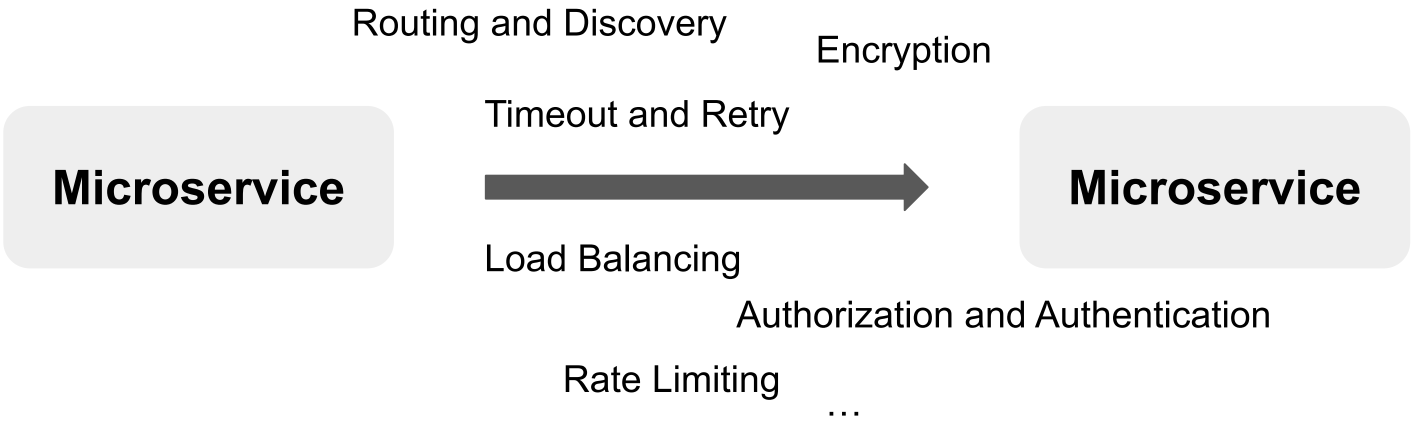


Microservices are distributed systems

Microservices are distributed systems



Microservices are distributed systems



Application networks: A new class of networks

Connect **endpoints of an application**, not anyone

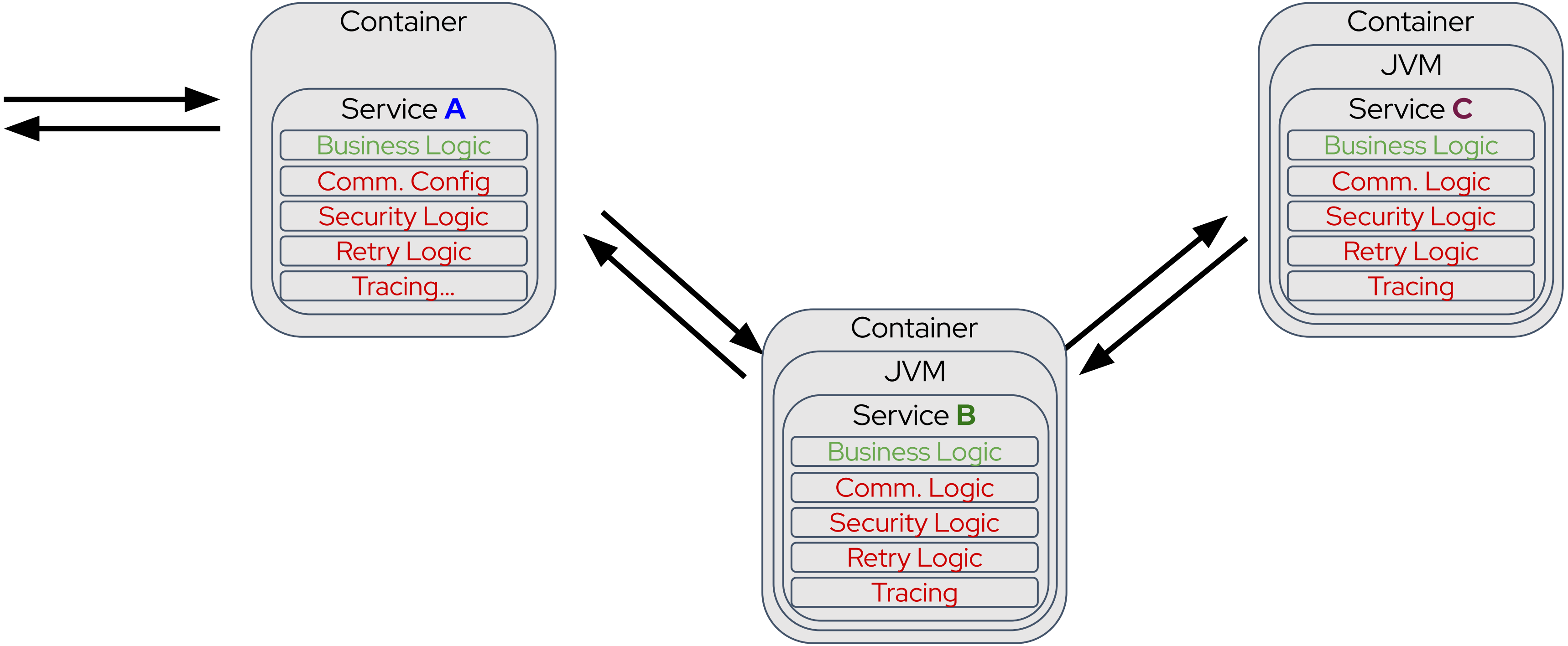
Need **rich message processing**, not just IP

Built by **application developers**, not network engineers

Outline

- Background
- **Service Mesh**
- Application Defined Networks

Building application networks



Building application networks

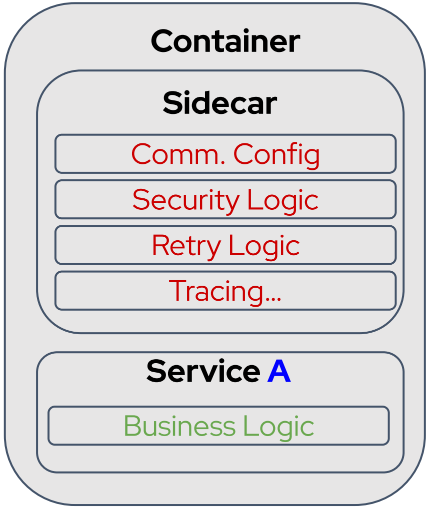
Earlier: Custom code for each microservices

Problems:

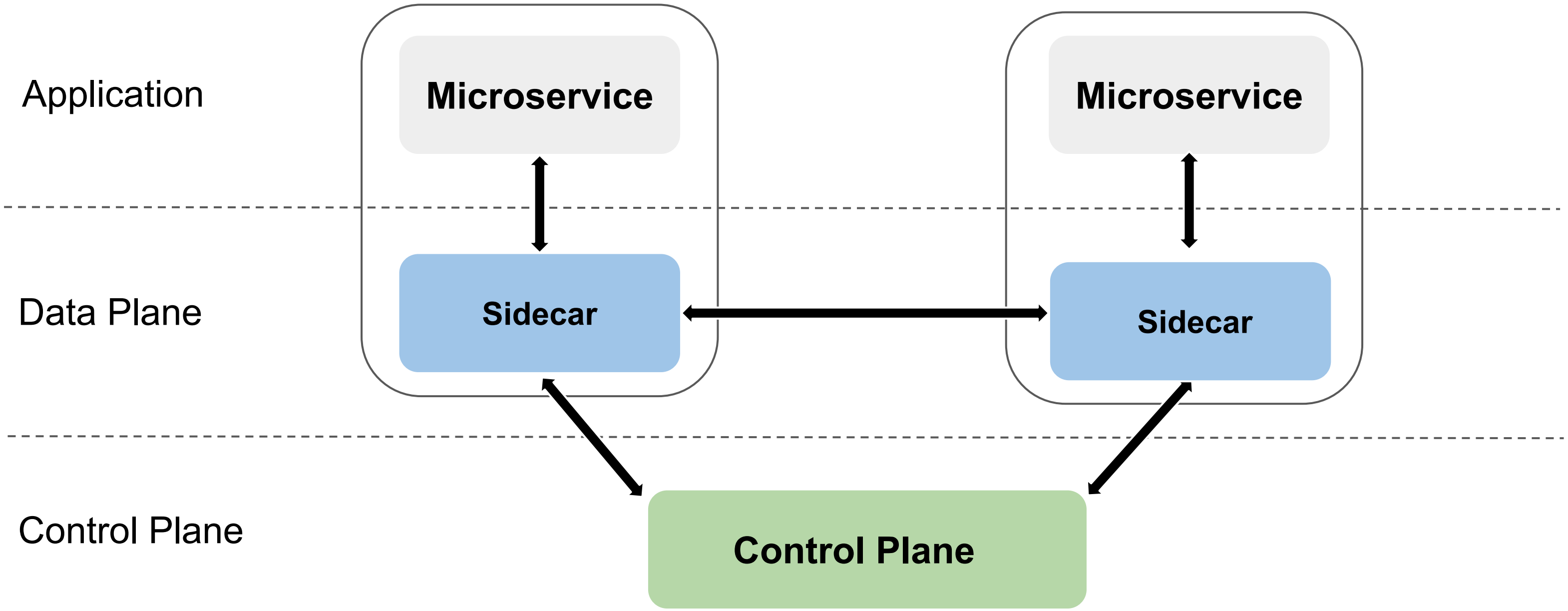
- Huge developer burden
- Network policies evolves independently
- Trust Issues
- …

Solution: Service Mesh with Sidecar Pattern

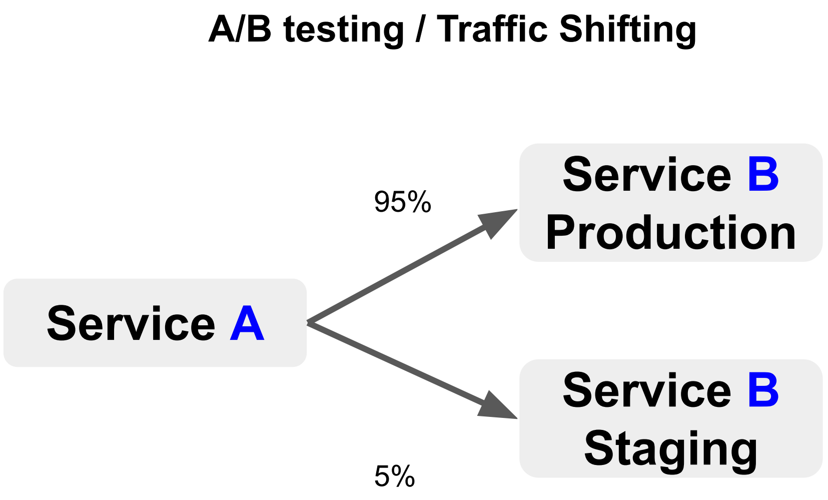
- Sidecar proxy handles all network logics
  - Traffic control / Routing
  - Resilience
  - Observability
  - Security
  - Policy Enforcement
  - …

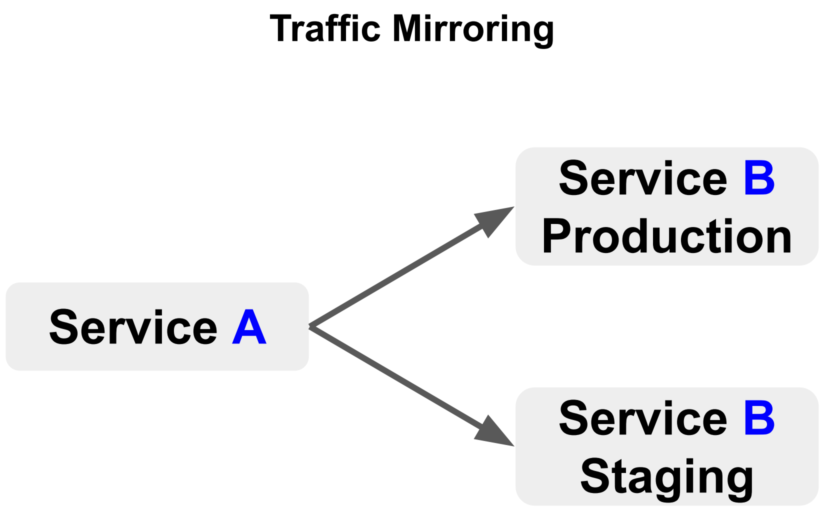


Service Mesh Architecture

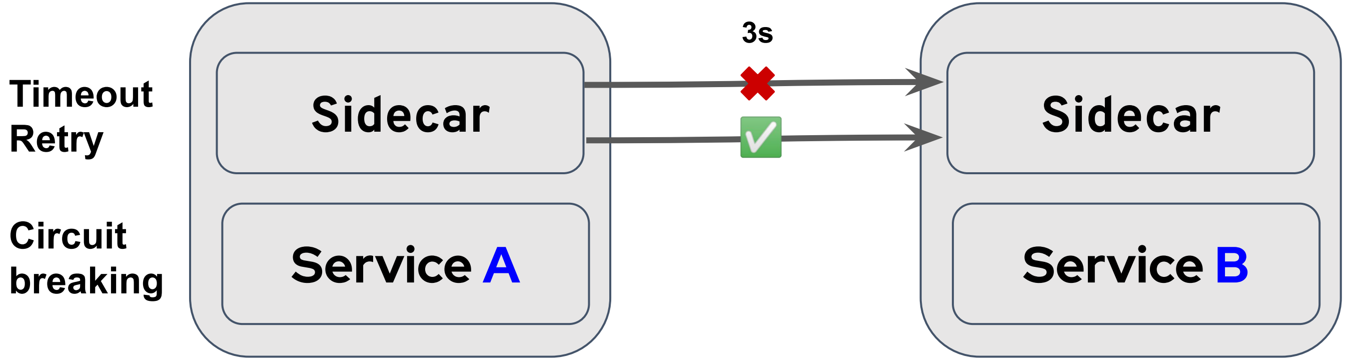


Traffic Management





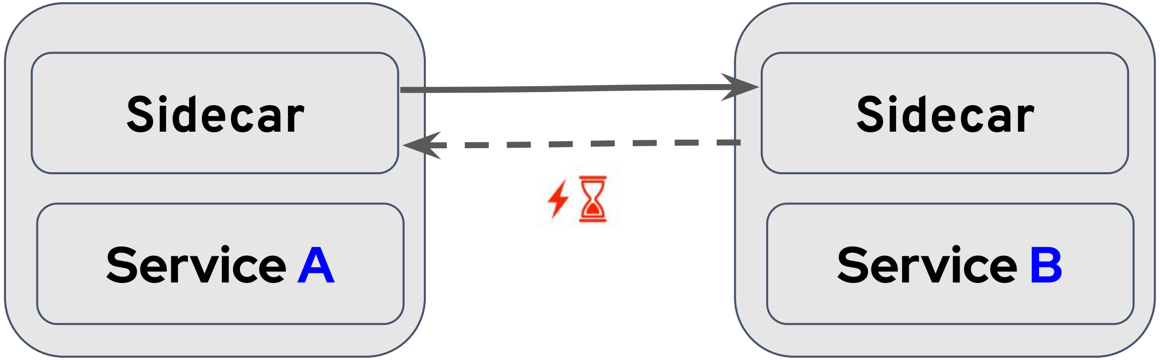
Resilience


Timeout
Retry
Circuit
breaking
3s
Sidecar
Service A
Sidecar
Service B

Chaos Engineering

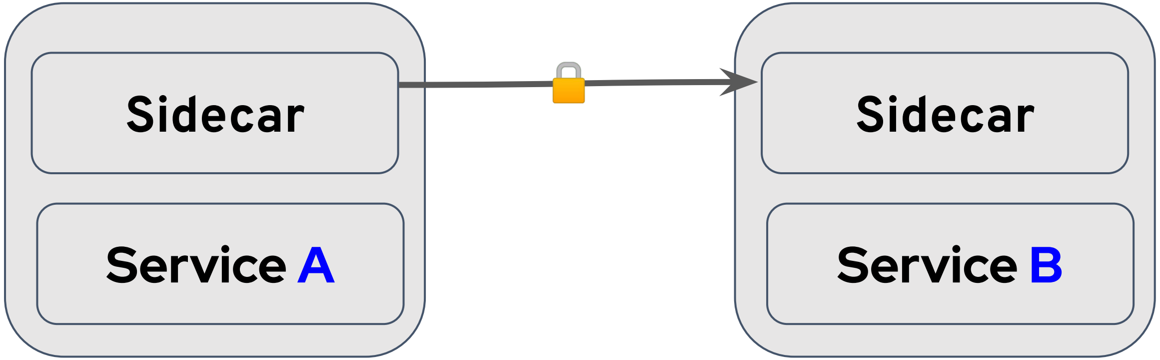
Fault Injection

Delay Injection

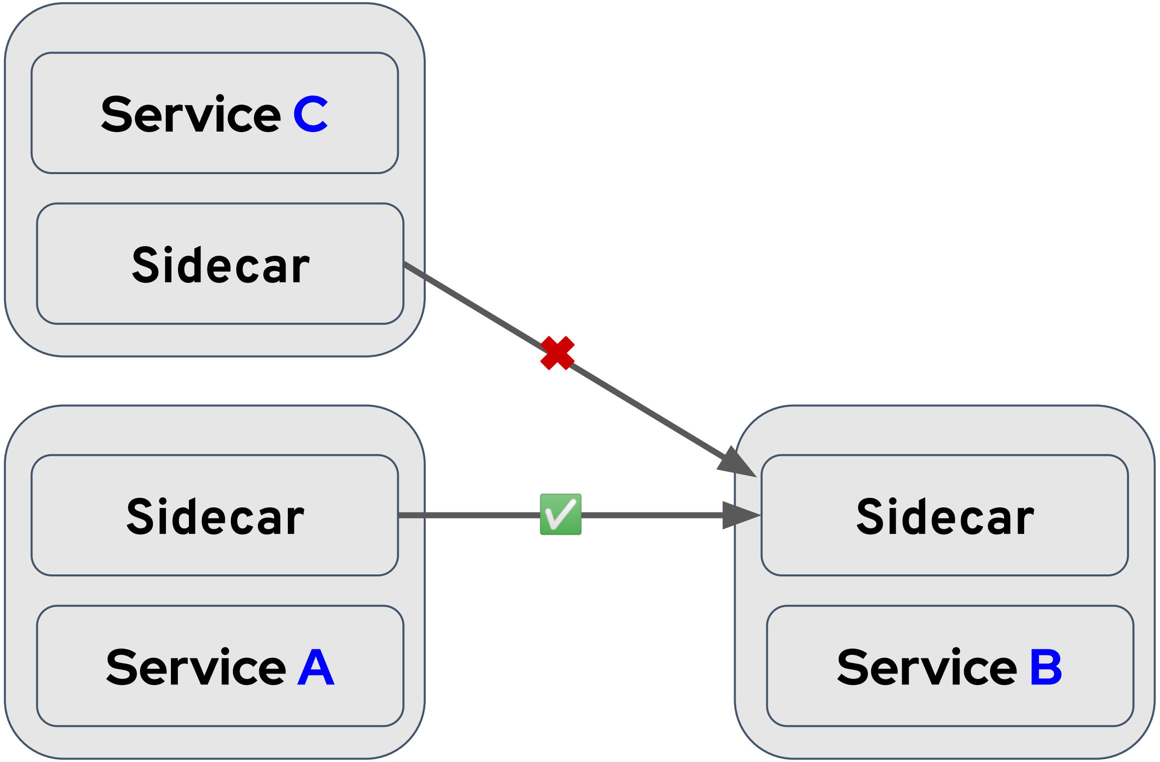


Security

mTLS Encryption
& Authentication



Authorization



Observability

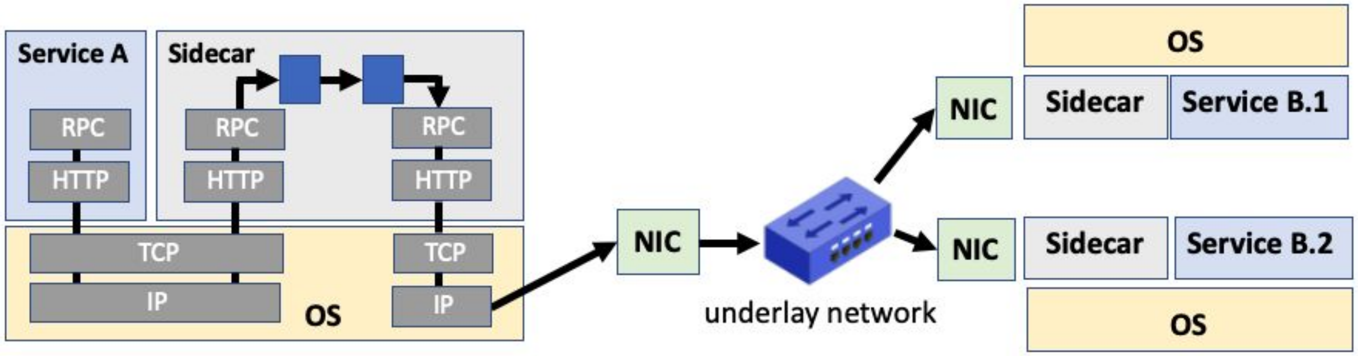
Service Mesh

- 90% organization uses service mesh according to a recent CNCF survey[1]

[1] Service meshes are on the rise - CNCF '22

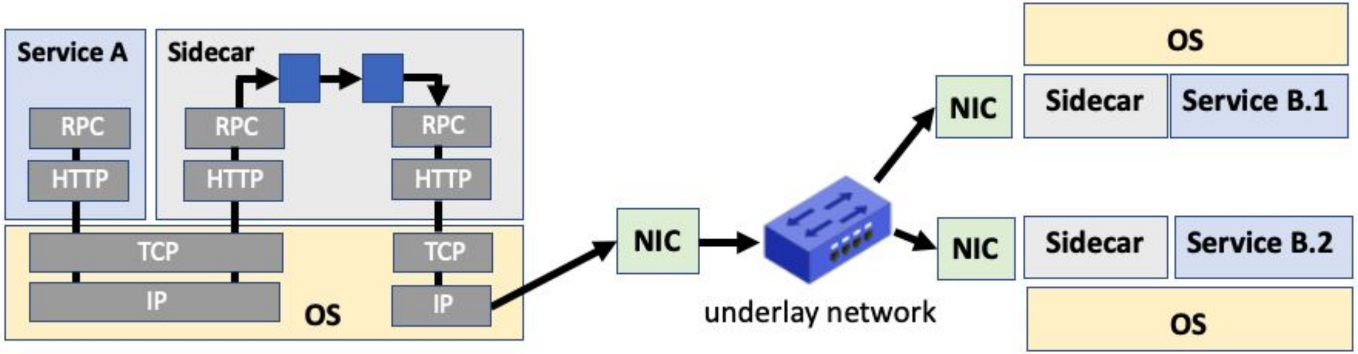
Service Mesh

- Build on general network architecture use by the Internet
  - gRPC/HTTP/TCP/IP



Service Mesh Challenges

- High Overheads
  - Throughput / Latency / CPU



Service Mesh Data Path

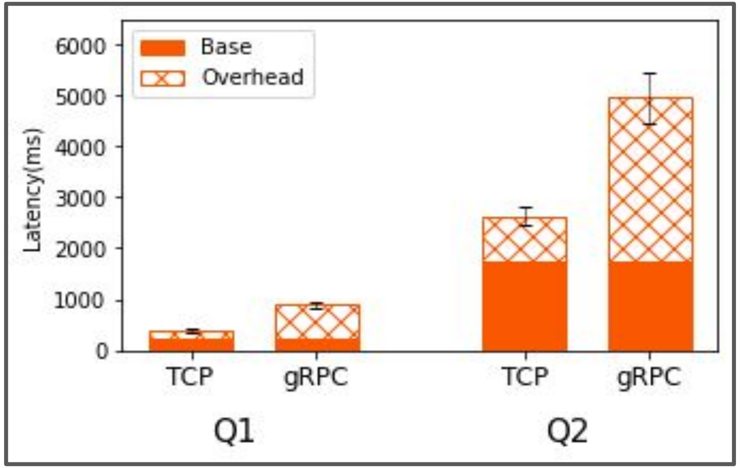
Kelsey Hightower
@kelseyhightower

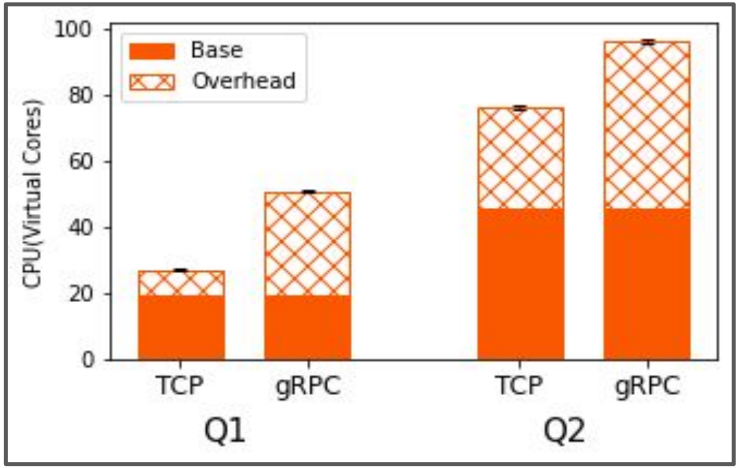
service mess /ˈsərvəs mes/
noun

1. the result of spending more compute resources than your actual business logic dynamically generating and distributing Envoy proxy configs and TLS certificates.

End-to-End Performance Overhead

- TCP mode can increase the latency by 0.6X and CPU usage by 0.9X
- gRPC mode can increase the latency by up to 2.7X and CPU usage by 1.6X





Latency and CPU overhead of Envoy

Service Mesh Challenges

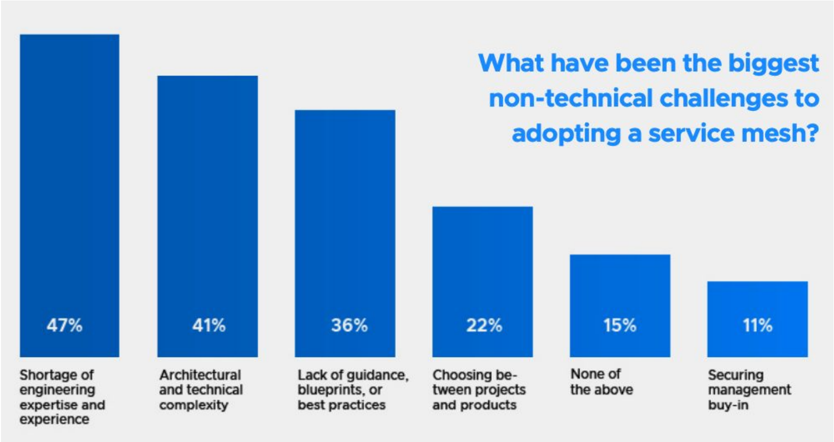
- High Overheads
  - Throughput/Latency/CPU
  - Overlapping/Unnecessary functionalities
  - Information hiding

Service Mesh Challenges

- High Overheads
    - Throughput/Latency/CPU
    - Overlapping/Unnecessary functionalities
    - Information hiding
- Non-portability
    - Difficult to offload to kernel and hardware

Service Mesh Challenges

- High Overheads
  - Throughput/Latency/CPU
  - Overlapping/Unnecessary functionalities
  - Information hiding
- Non-portability
  - Difficult to offload to kernel and hardware
- Difficult to use
  - API is complex and evolving
  - Poor extensibility



CNCF Survey 2022

Service Mesh Challenges

- High Overheads
  - Throughput/Latency/CPU
  - Overlapping/Unnecessary functionalities
  - Information hiding
- Non-portability
  - Difficult to offload to kernel and hardware
- Difficult to use
  - API is complex and evolving
  - Poor extensibility

```yaml
apiVersion: networking.istio.io/v1alpha3
kind: EnvoyFilter
metadata:
  name: reviews-lua
  namespace: bookinfo
spec:
  workloadSelector:
    labels:
      app: reviews
  configPatches:
    # The first patch adds the lua filter to the listener/http connection manager
  - applyTo: HTTP_FILTER
    match:
      context: SIDECAR_INBOUND
      listener:
        portNumber: 8080
        filterChain:
          filter:
            name: "envoy.filters.network.http_connection_manager"
            subFilter:
              name: "envoy.filters.http.router"
    patch:
      operation: INSERT_BEFORE
      value: # lua filter specification
       name: envoy.filters.http.lua
       typed_config:
          "@type": "type.googleapis.com/envoy.extensions.filters.http.lua.v3.Lua"
          inlineCode: |
            function envoy_on_request(request_handle)
              -- Make an HTTP call to an upstream host with the following headers, body, and timeout.
              local headers, body = request_handle:httpCall(
               "lua_cluster",
               {
                [":method"] = "POST",
                [":path"] = "/acl",
                [":authority"] = "internal.org.net"
               },
              "authorize call",
              5000)
            end
  # The second patch adds the cluster that is referenced by the lua code
  # cds match is omitted as a new cluster is being added
  - applyTo: CLUSTER
    match:
      context: SIDECAR_OUTBOUND
    patch:
      operation: ADD
      value: # cluster specification
        name: "lua_cluster"
        type: STRICT_DNS
        connect_timeout: 0.5s
        lb_policy: ROUND_ROBIN
        load_assignment:
          cluster_name: lua_cluster
          endpoints:
          - lb_endpoints:
            - endpoint:
                address:
                  socket_address:
                    protocol: TCP
                    address: "internal.org.net"
                    port_value: 8888
```

Customize Istio Configuration

Application Defined Networks

Idea: Application Defined Networks (ADN)

Developers specify what the network should do at a high level

- Application-relevant abstractions
- Declarative, portable

A controller auto generate an optimized application-specific implementation

- Determine what processing happens and how (incl. hardware offload)
- Determine message headers

Idea: Application Defined Networks (ADN)

Developers specify what the network should do at a high level

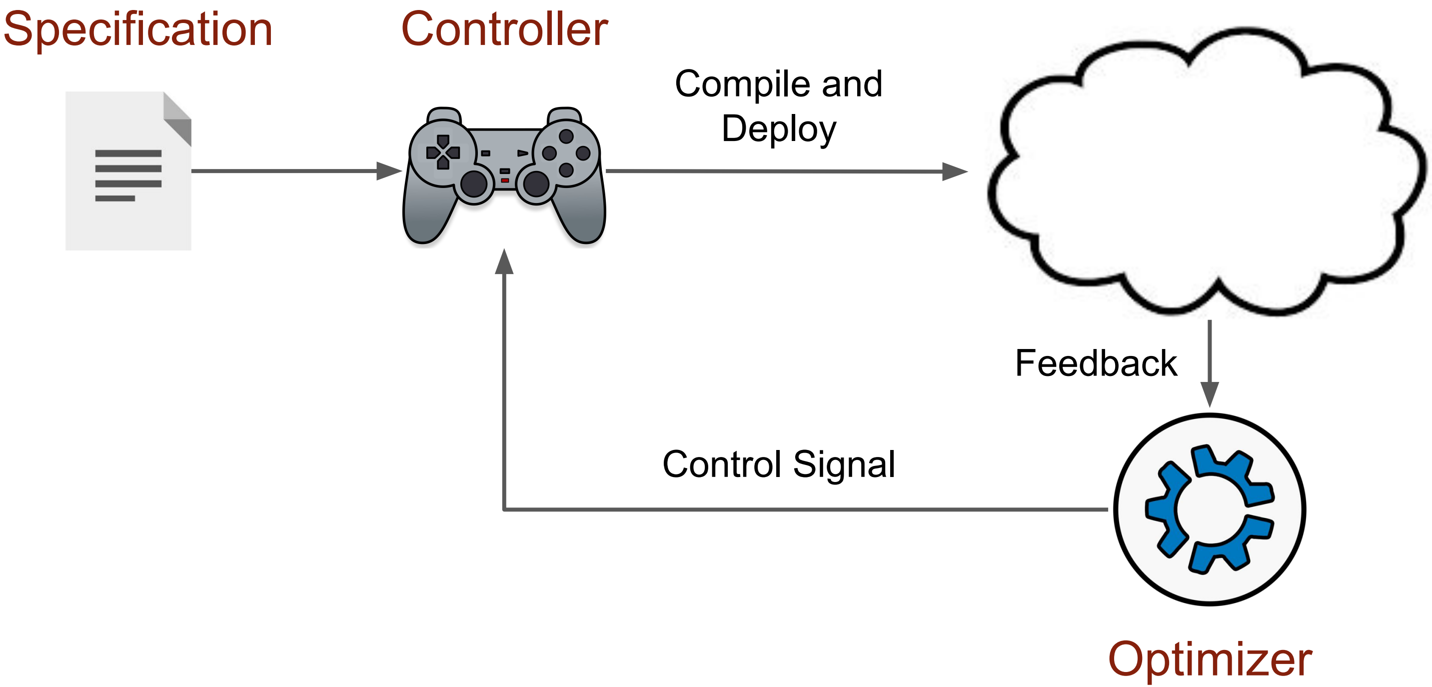
- ❑ Application-relevant abstractions
- ❑ Declarative, portable

A controller auto generate an optimized application-specific implementation

- ❑ Determine what processing happens and how (incl. hardware offload)
- ❑ Determine message headers

Meets application-specific needs
without a burdened implementation that does it all

ADN architecture



Example

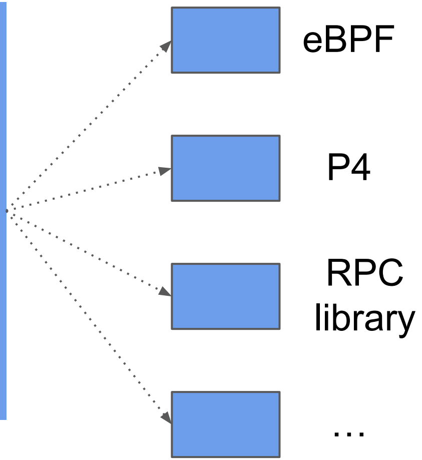
```
S1→S2: LoadBalancing→Logging→Compression→FaultInjection(0.1)
```

```
FaultInjection(probability)


SELECT * from input

WHERE input.ver == 1 AND

      random() < probability
```

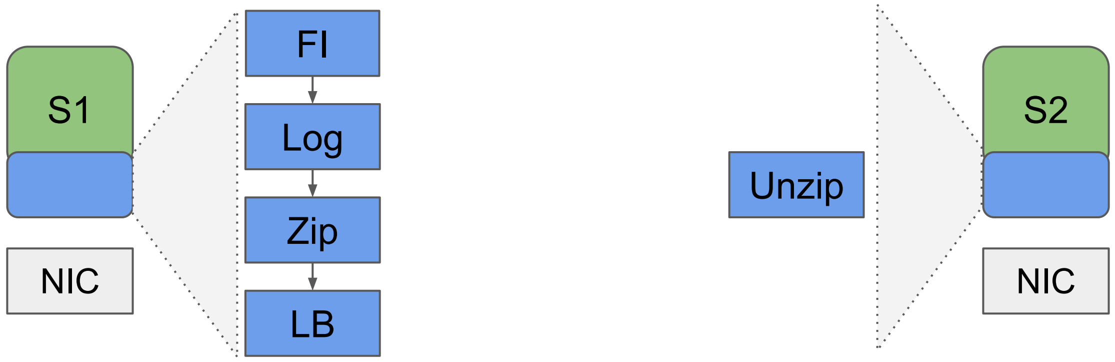


Example

`S1→S2: LoadBalancing→Logging→Compression→FaultInjection(0.1)`

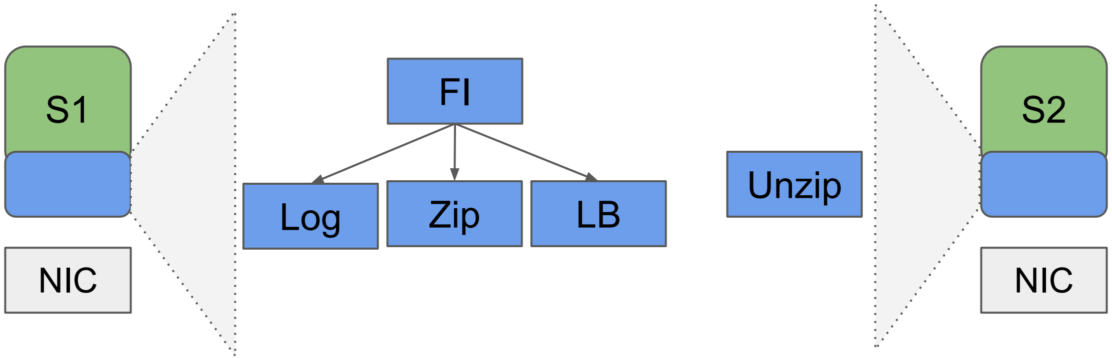
Example

`S1→S2: LoadBalancing→Logging→Compression→FaultInjection(0.1)`



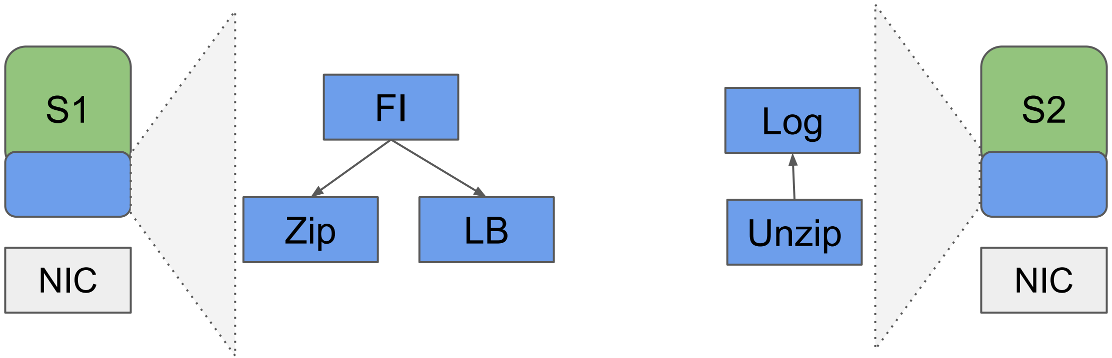
Example

`S1→S2: LoadBalancing→Logging→Compression→FaultInjection(0.1)`



Example

`S1→S2: LoadBalancing→Logging→Compression→FaultInjection(0.1)`



Example

`S1→S2: LoadBalancing→Logging→Compression→FaultInjection(0.1)`