



# Micro services

- Intro to the Pattern language
- Service Mesh & Gateways
- Distributed tracing



# Micro services

- Let's watch
  - Microservices Architectural Pattern
    - <https://www.youtube.com/watch?v=8BPDv038oMI>



# PolyGlot

- Allow to use several software stacks, programming languages, or frameworks
- Each team adapt to their skills



# Microservices

- It won't magically solves all your problems, it's even possible to make it worse
  - Got time ? Check this out
    - <http://chrisrichardson.net/post/antipatterns/2019/01/28/melbourne-microservices.html>



# Microservices patterns

- Discovery
- Load Balancing
- Circuit breaking
- From REST to Objects (Feign)
- And a lot more ....



# Discovery “intro”

- A registry where services can find each other
  - Health check
- Consul , Eureka , zookeeper...
- We'll study Consul in detail later



# Load Balancing


- Server Side
- Client side



# Service Mesh

- It's much more than that
  - let's watch:
    - What is a service mesh| Armon Dagar
      - <https://www.youtube.com/watch?v=vh1YtWjfcyk>



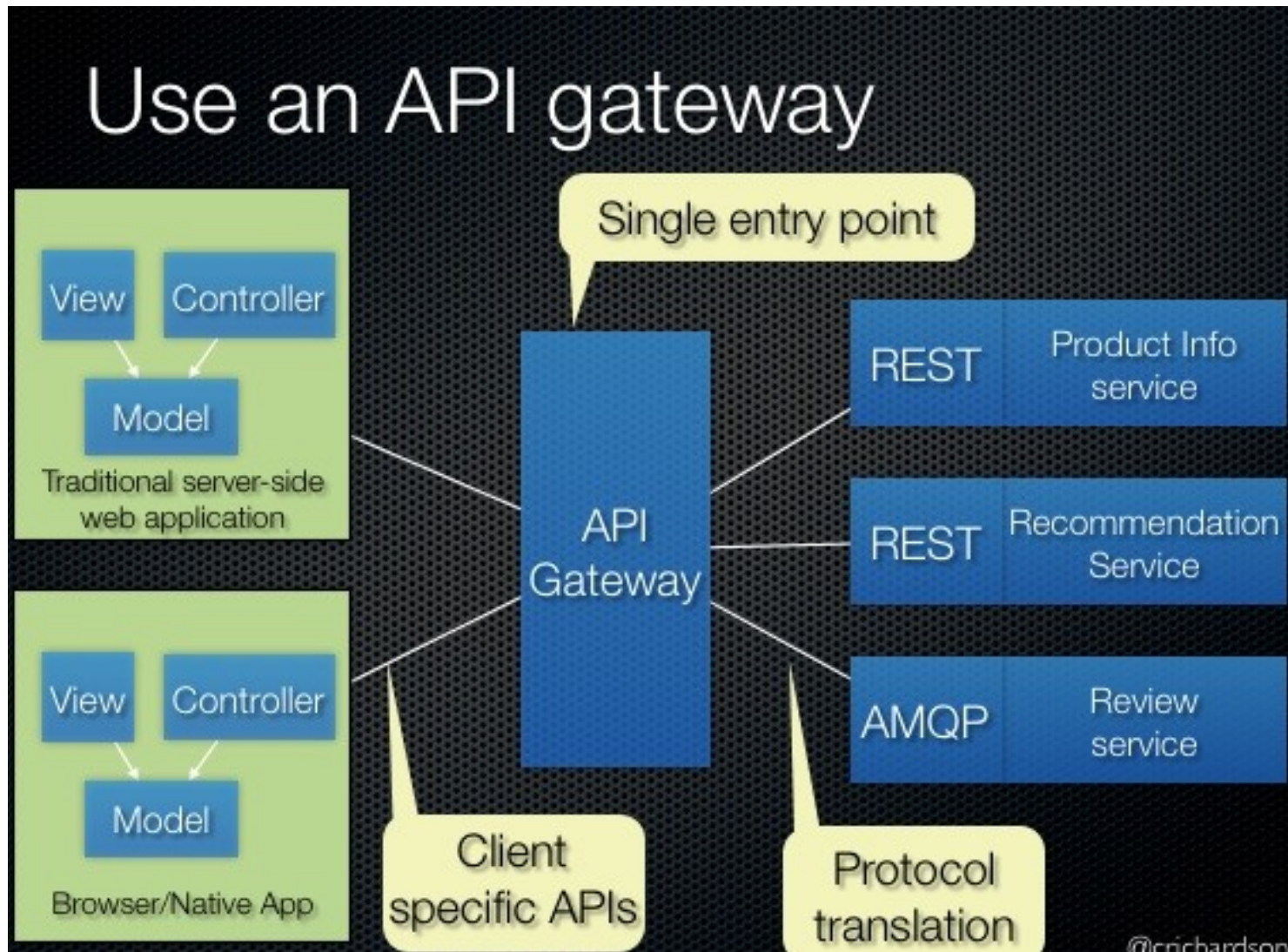
- 
- But we have many services to do a certain operation
    - Do clients need to know about all of them?!



# API Gateway

- A single endpoint for UI
- Works as Router and Filter to the rest of internal services
  - Hide internal endpoints for 100s of services

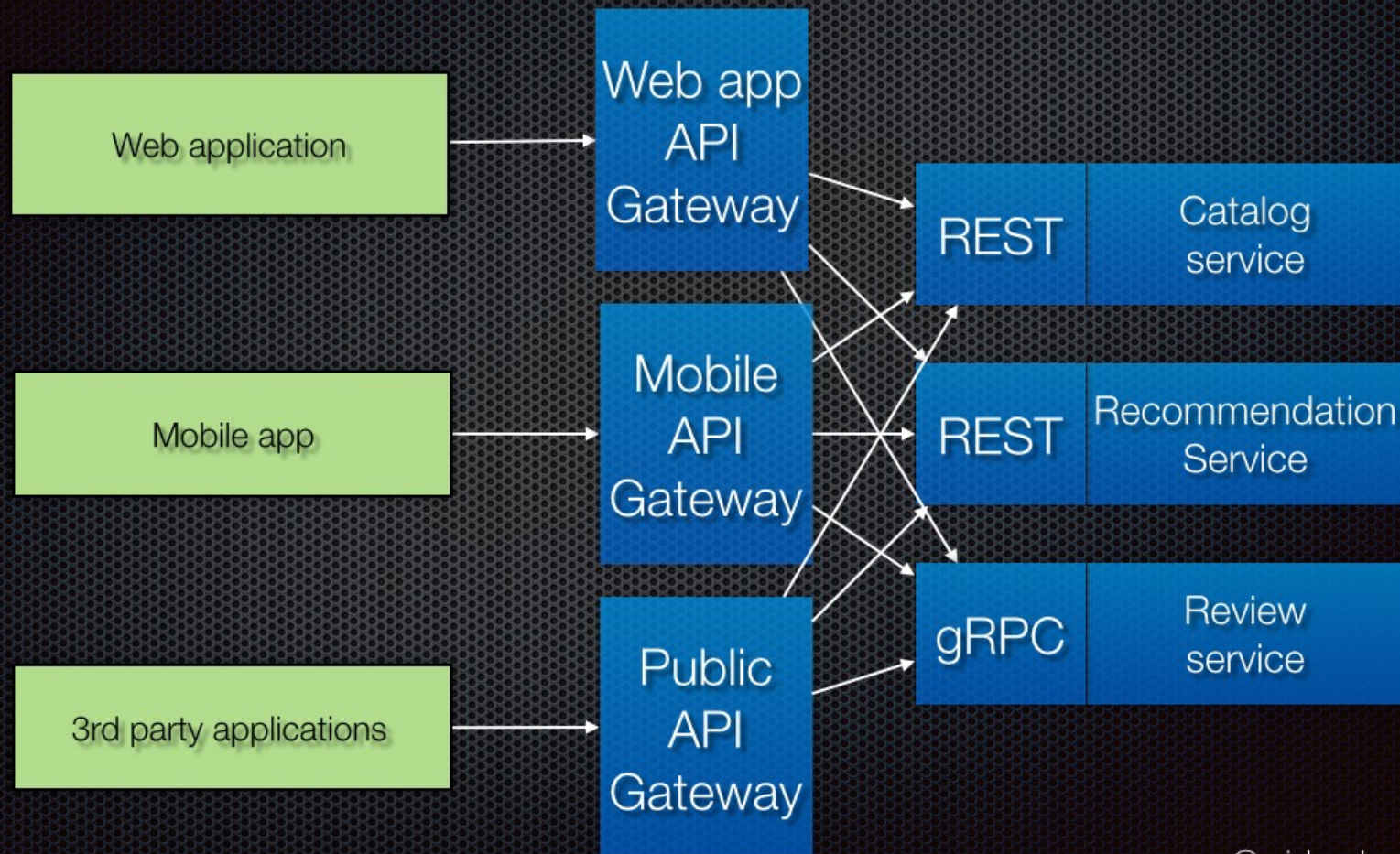
# API Gateway





# API Gateway

## Variation: Backends for frontends





# API gateway

- Let's read:
  - <https://microservices.io/patterns/apigateway.html>



# API Gateway

- Spring cloud gateway
  - <https://spring.io/projects/spring-cloud-gateway>
- ZUUL
- ....



# From Gateway to service Mesh

- Watch:
  - API Gateway to Service Mesh: Navigating a Changing Landscape | Zhamak Dehghani
    - <https://www.youtube.com/watch?v=QYdOJ0QJptE>



# Still confused ?!


- Let's read:
  - <https://avinetworks.com/what-are-microservices-and-containers/>





# Service Mesh

- Be careful!!, it can be easily an Anti-pattern
  - Let's check this out:
    - <https://akfpartners.com/growth-blog/microservice-anti-pattern-service-mesh>



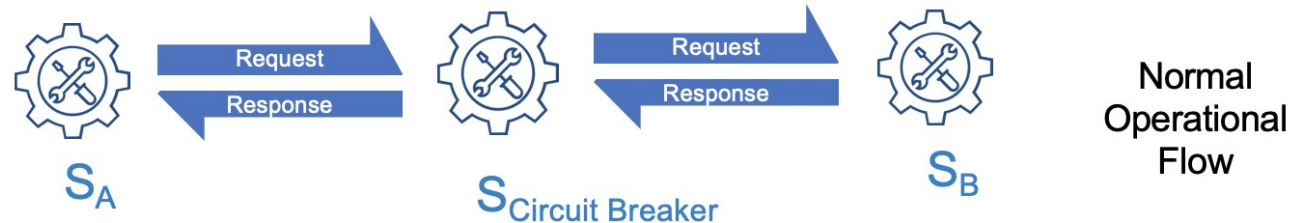
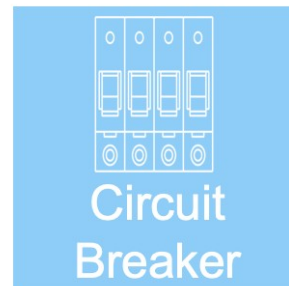
How to prevent a failing service from  
cascading to other services?



# Circuit breaker

- When error rates or response times exceed a desired threshold, the breaker “pops” and returns an appropriate error
- Read:
  - <https://microservices.io/patterns/reliability/circuit-breaker.html>

# Circuit Breaker Pattern



The Circuit Breaker monitors for timeouts or repeated failures – closes the circuit and immediately responds



# Circuit breaker

- Read:
  - <https://akfpartners.com/growth-blog/the-circuit-breaker-pattern-dos-and-donts>



# Check out

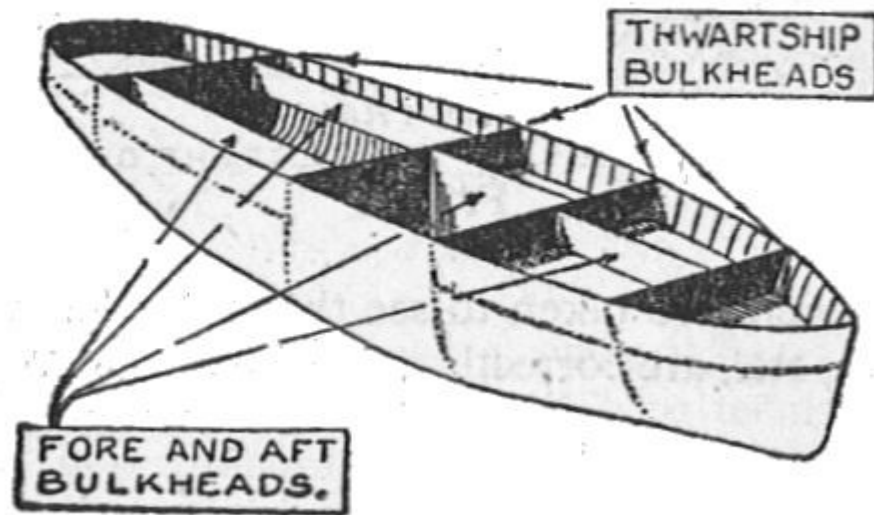
- Spring circuit breaker
  - <https://spring.io/projects/spring-cloud-circuitbreaker>



# Resilience patterns

- Retry, Rate Limiter ,...
- Got more time?, check :
  - <https://github.com/resilience4j/resilience4j>

# Bulkhead



[https://en.wikipedia.org/wiki/Bulkhead\\_\(partition\)](https://en.wikipedia.org/wiki/Bulkhead_(partition))

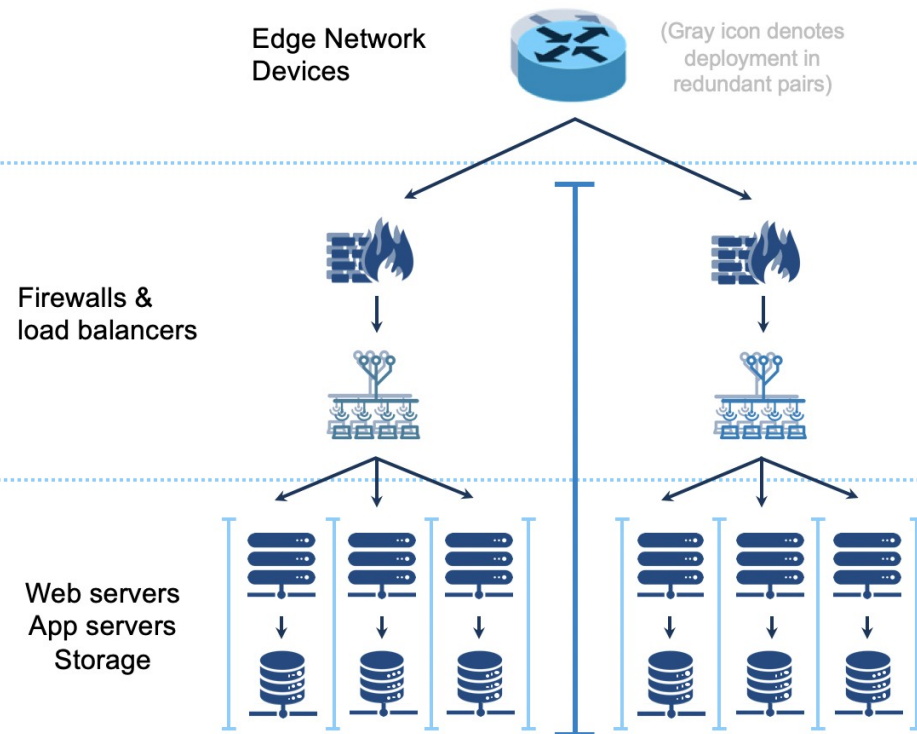




# Bulkhead pattern

- Fault Isolation
  - Prevent failures of some component from propagating to other components
- Read
  - <https://akfpartners.com/growth-blog/bulkhead-pattern>

# Isolate Faults – Swim Lanes



## *Generally Do Not Isolate –*

Cost prohibitive to justify full isolation

## *Isolate if Practical –*

- Firewalls and load balancers **bottleneck and fail**
- Ideally **each swim lane** will have its own firewalls and load balancers (in redundant pairs)

## *Always Isolate –*

- Top of rack switches
- Compute
- Storage



# Swim Lanes

- Got some time? , read :
  - <https://akfpartners.com/growth-blog/fault-isolation-swim-lane>



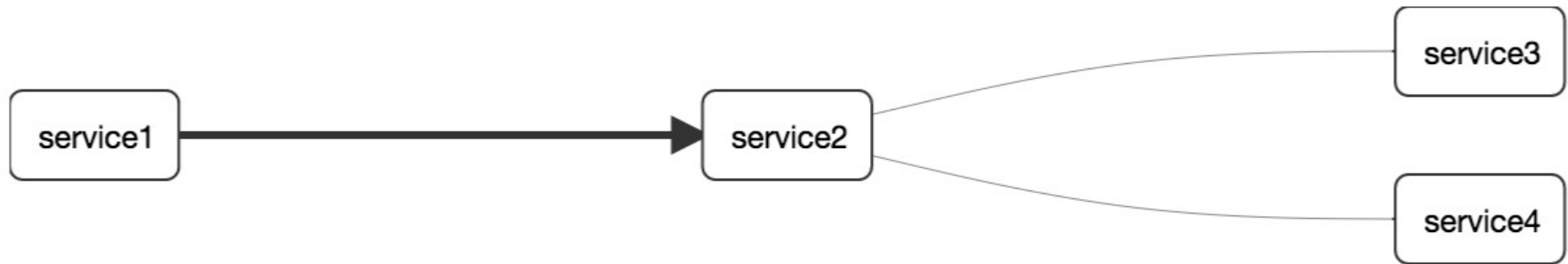
But how we trace services, how we detect  
slow/failing ones ?!



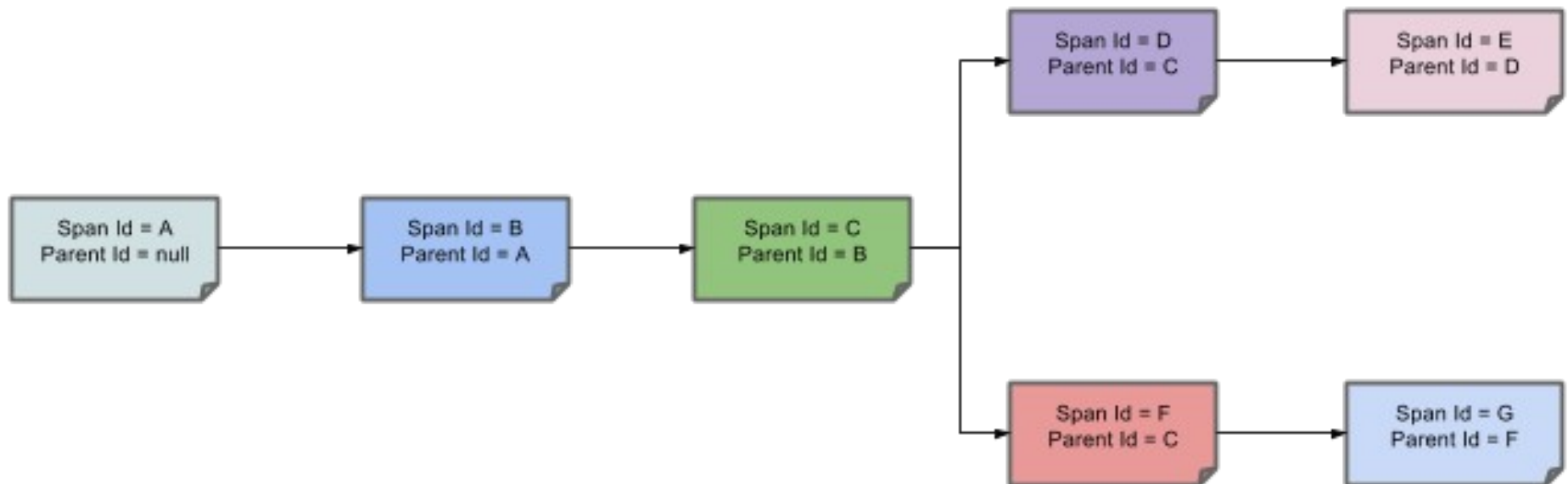
But how we trace services, how we detect  
slow/failing ones ?!

Distributed tracing

# Dependency Graph

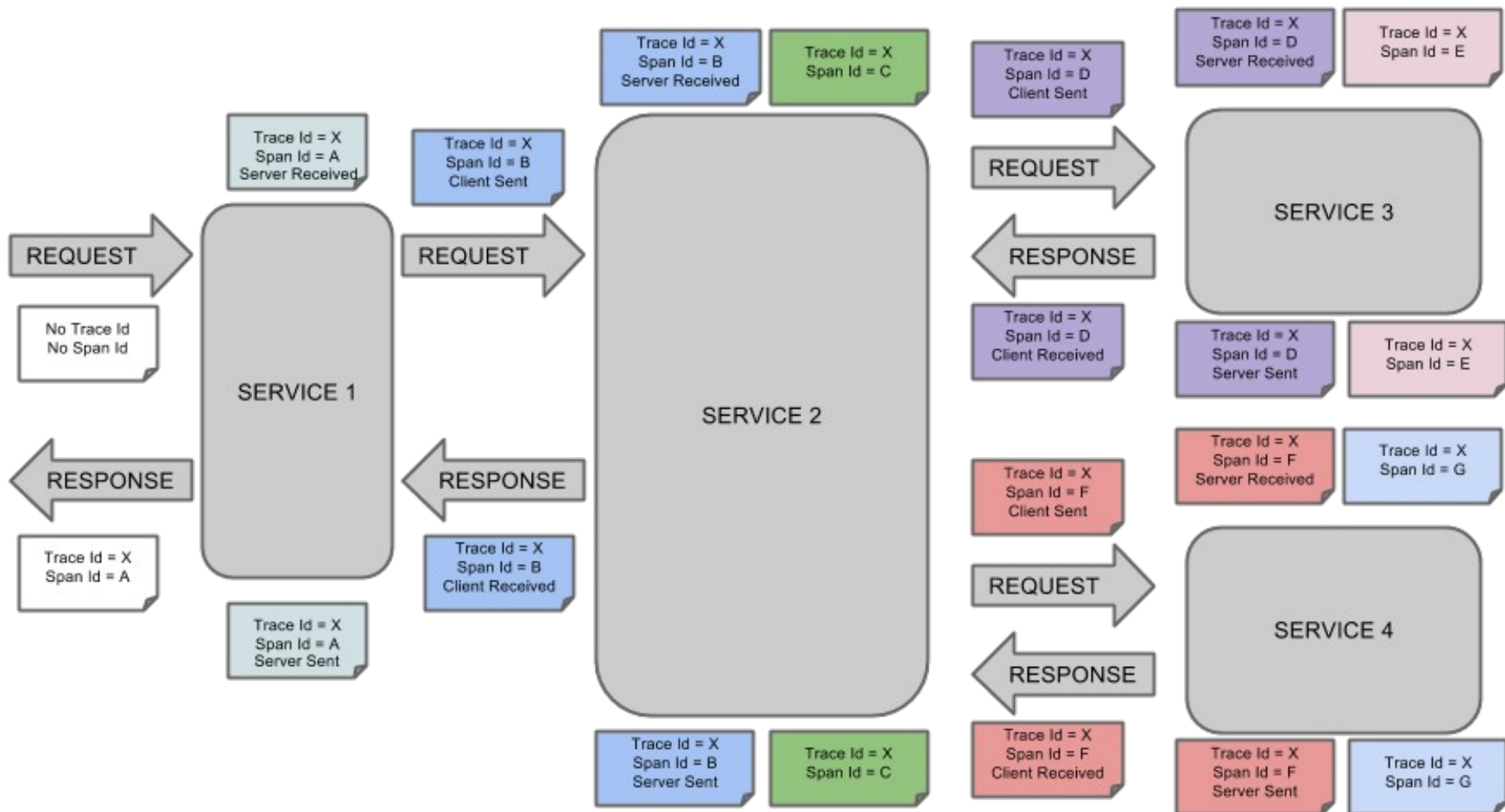


# Spans



Span: unique ID , represents a request or response  
Can have tags, timestamps ,...

# Spans







# Distributed tracing

- Spring Cloud Sleuth
  - <https://cloud.spring.io/spring-cloud-sleuth/reference/html/>
- Zipkin
  - <https://zipkin.io/>



# Distributed tracing

- Let's watch
  - An Introduction to Distributed Tracing and Zipkin | Adrian Cole
    - <https://www.youtube.com/watch?v=jkSm-652UPo>



# Batch Processing

- Your first assignment



# Distributed Batch

- Cloud-Native Batch Processing with Spring Batch 4 - Michael Minella
  - <https://www.youtube.com/watch?v=-lcd-s2JoAw>
- High Performance Batch Processing
  - <https://www.youtube.com/watch?v=J6IPIfm7N6w>