How Istio helped us investigate failures on our microservices

Shota Shirayama
Rakuten, Inc.

The goal of this session

I believe that people, who are considering Service Mesh, think that Istio looks good by giving an example of the usefulness of Istio in our system.

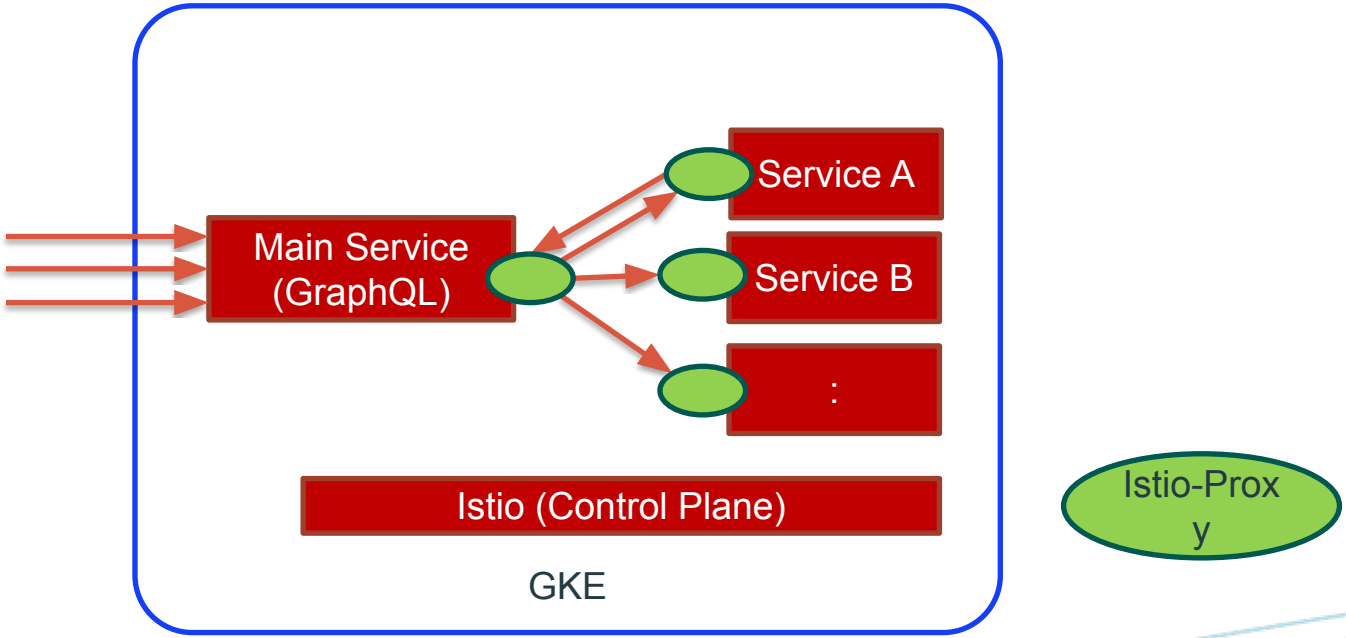
Background

- Microservices increase system complexity in general.
- It wasn't easy to improve logging or architecture for the development team because of their focus on service development.
- SRE decided to deploy Istio to combat the system complexity.

Today’s Story

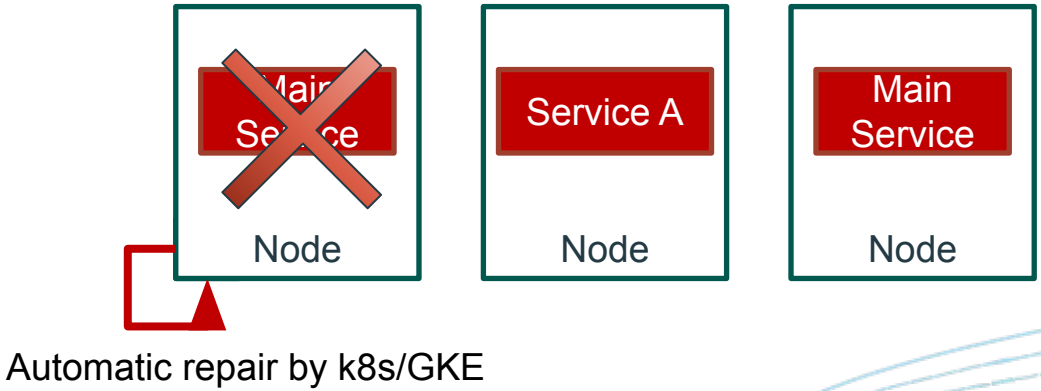
Istio brought us the network's observability and testability, which led us to solve the complex system failure.

Simple diagram of service architecture



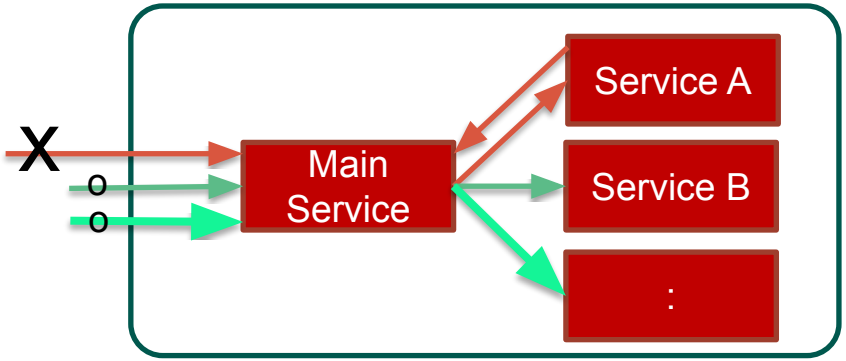
One day, the node went down

- Main Service pods were redundant.
- The node on which Main Service pod was running went down.
- Pod and Node recovered automatically after a while.



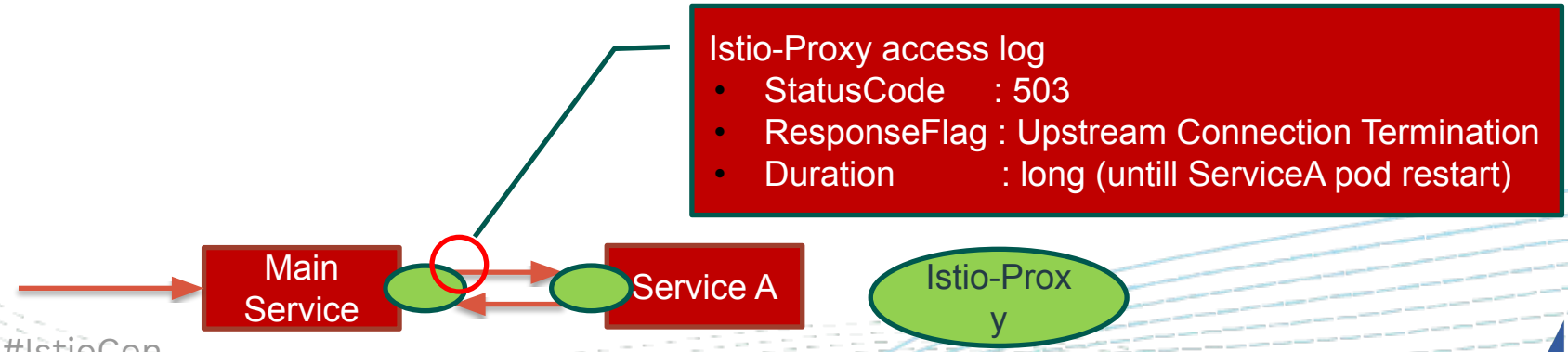
But, one endpoint on Main Service didn't work

- One endpoint on Main Service (which calls ServiceA internally) didn't respond.
- ServiceA pod was restarted manually. Then, it started working properly.



The additional clue for the root cause from Istio-Proxy's log

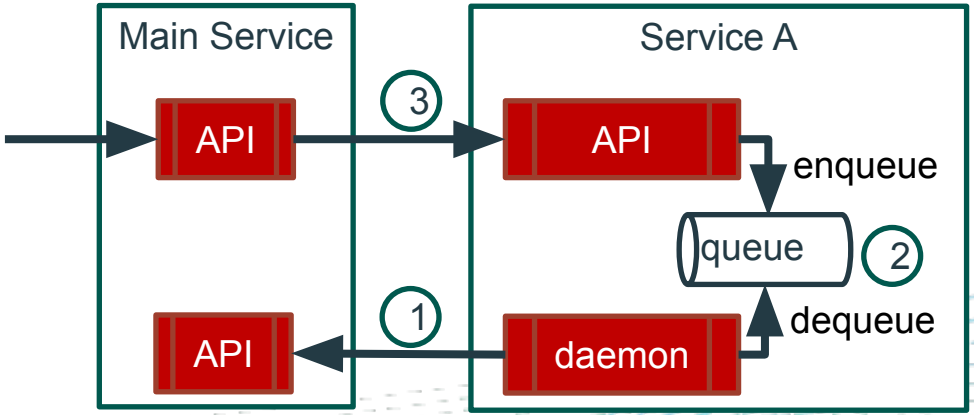
- Fact 1: The node on which Main Service pod was running went down.
- Fact 2: One Main Service endpoint stopped working until manual ServiceA pod restart.
- (No useful application logs to reveal what was happening...)
- **Fact3: Main Service waited for responses from ServiceA.**



Why did Main Service wait for responses from ServiceA ?

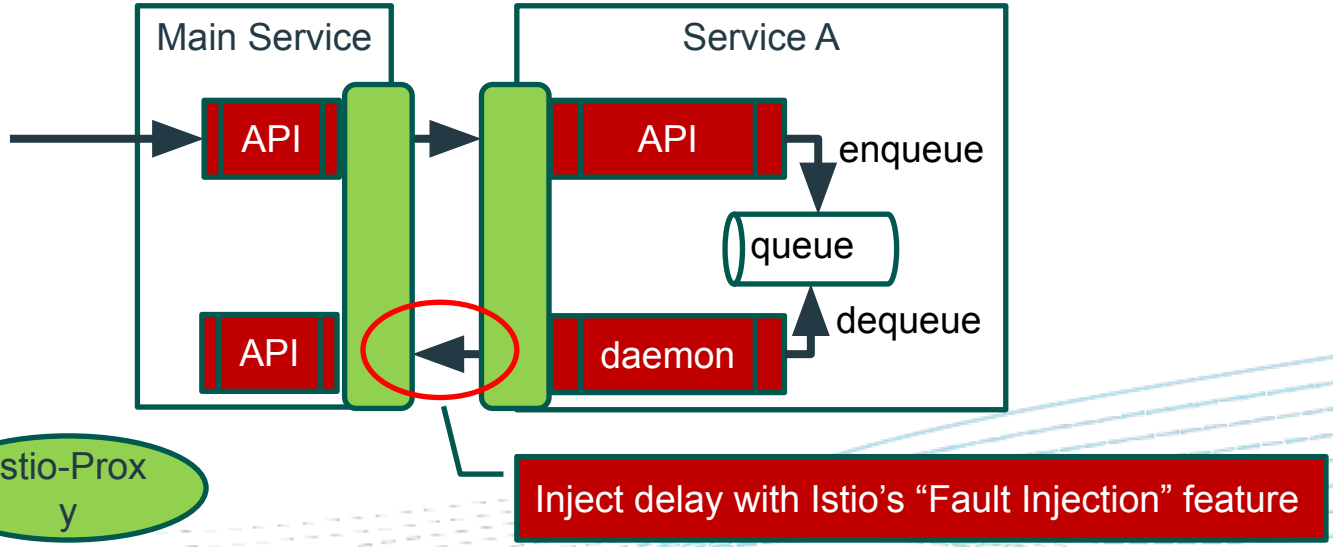
Hypothesis:

1. Node down caused that ServiceA waited for Main Service's response forever.
2. The Queue was full, and ServiceA waited to enqueue the data.
3. ServiceA didn't return responses to Main Service



Reproduce the failure in a test environment

- Difficult to reproduce the same situation...
- Istio's "Fault injection" makes it easy to reproduce the same situation in a test environment.



Recap:
Istio helped us investigate failure on our microservices

Istio improves observability
→ Istio-Proxy's log helped us find the hypothesis of the cause of failure.

Istio improves testability
→ Istio's feature (fault injection) made it easy to reproduce the failure.

Thank you!