클라우드 환경 기반 실시간 데이터 처리의 유실 없는 Geo-Replication 구축

김영진, 이모원, 신수인

Data Insight Center, Log Data

NAVER DEVIEW 2023

CONTENTS

NAVER
DEVIEW
2023

NAVER
DEVIEW
2023

1. IDC 장애의 심각성
그리고 Geo Replication

1.1 10/15 사태가 주는 교훈

NAVER
DEVIEW
2023

IDC 장애 그리고 대응

- 10/15 화재가 주는 교훈

- 빠른 장애 대응력이 곧 기업의 경쟁력
  - 기업의 기술 경쟁력에 대한 재고
  - 대응 방식에 따라 Brand 가치의 상승 or 하락

- IT 빅테크 기업이 갖춘 고도화된 장애 대응 시스템
  - 국내 여러 데이터 센터 분산 구조
  - 구글은 1년에 2번 이상 재해 복구 훈련 진행

과기정통부, 모레 '민간 데이터센터 안정성' 긴급 점검

https://m.yna.co.kr/view/AKR20210180807000 17

[IT 줌인] 국내 데이터센터 '외연 확장' 치중, 재해복구 능력 제한

국내 데이터센터 60% 수도권 집중, 이원화 미비
해외 빅테크 재해복구 대책 기반 '회복력' 강조
데이터센터 규제 입법 소위원회 통과, 향방 주목

https://biz.newdaily.co.kr/site/data/html/2022/11/29/2022112900095.html

〈 재난·재해 사고에 대비한 글로벌 빅테크 기업의 데이터센터 관리 운영 〉

기업	대응 현황
amazon	• 물리적으로 분리된 데이터센터 여러 개를 '가용영역'으로 관리 • 인접 가용영역들은 지역별 리전으로 묶어 서비스 연속성 보장 • 서울에 가용영역 4개를 갖추고 서로 연동
Microsoft	• 세계 140개 국가에 데이터센터 분산, 총 60개 리전 운영 • 국내에서는 서울·부산 두 곳의 데이터센터가 상호 백업역할 담당 • 데이터센터 내부 온도 유지에 도움이 되는 스코틀랜드 인근 해저에 데이터센터 구축
Google	• 정확한 데이터센터 위치와 수를 보안 이유로 미공개 • 지난 5년간 미국에서만 370억 달러 투자해 데이터센터 증설(2022년 테네시, 버지니아, 오클라호마 등에 데이터센터 추가 건립) • 1년에 두 차례 이상 재해 복구 훈련

자료 : 한겨레, 2022.10.19. / 한국경제, 2022.10.17

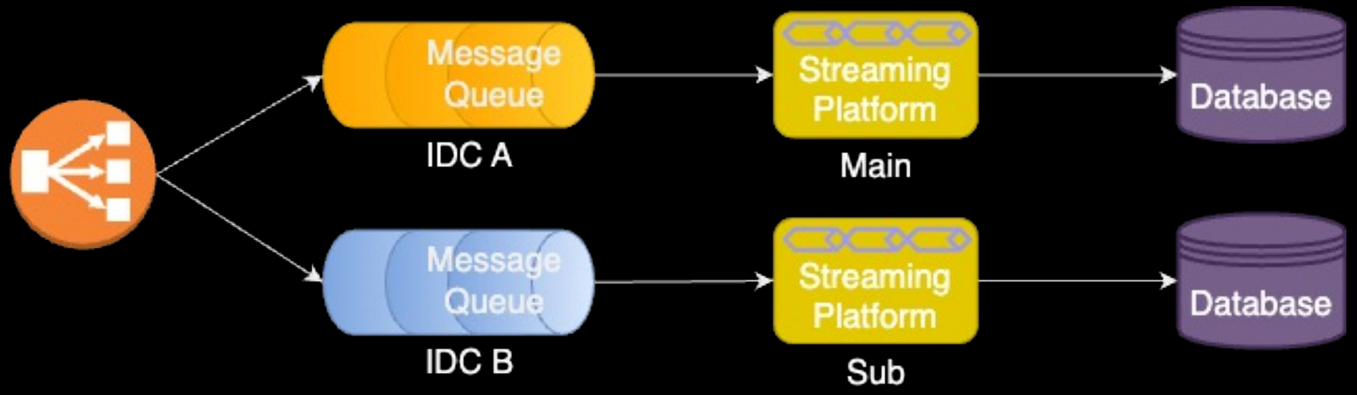
https://now.k2base.re.kr/portal/trend/mainTrend/view.do?poliTrndId=TRND0000000000048121&menuNo=2000&pageUnit=10&pageIndex=4

1.2 Geo Replication의 기능

NAVER
DEVIEW
2023

Geo-Replication의 여러 기능들

- Disaster Recovery
  - RTO (Recovery Time Objective)
- Compliance
  - GDPR, HIPPA and SEC
- Data Migration
  - Copy of Data
- Data Distribution
  - Read Only copy of data
- Data co-location
  - for latency sensitive workload



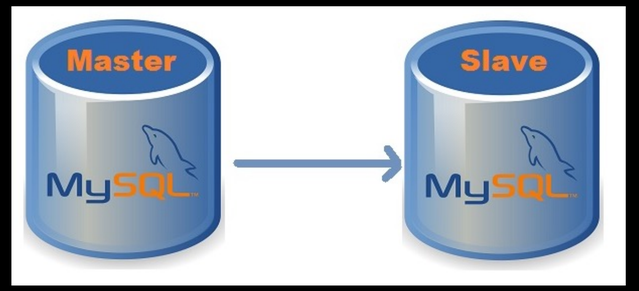
1.3 Geo Replication의 현황

NAVER DEVIEW 2023

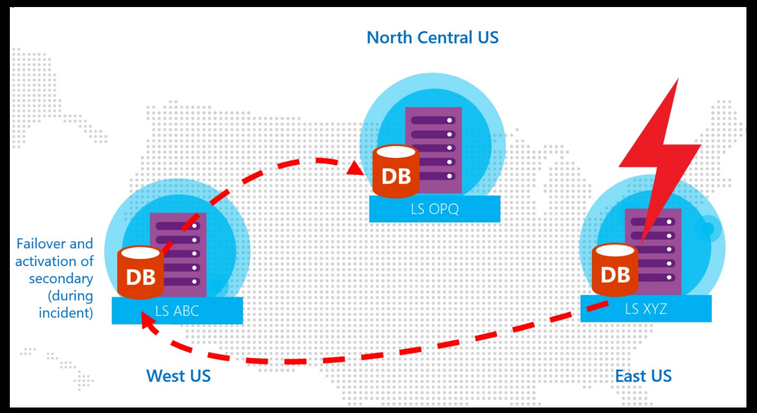
Geo Replication 인식

- 다양한 Platform 에서 이미 지원
  - 자연재해가 많은 국가에서는 보편적
  - 국토의 면적이 넓은 기업에게도 보편적
- 우리는 "Geo Replication" 이라는 단어 조차 익숙하지 않은 실정

Mysql의 GeoReplication



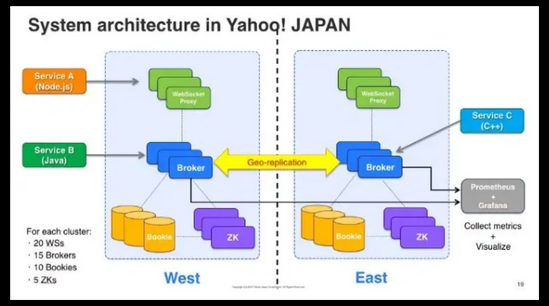
https://thecustomizewindows.com/2019/10/mysql-configuration-to-set-up-master-slave-replication/



https://mscloud.be/azure/configuring-azure-sql-db-active-geo-replication/

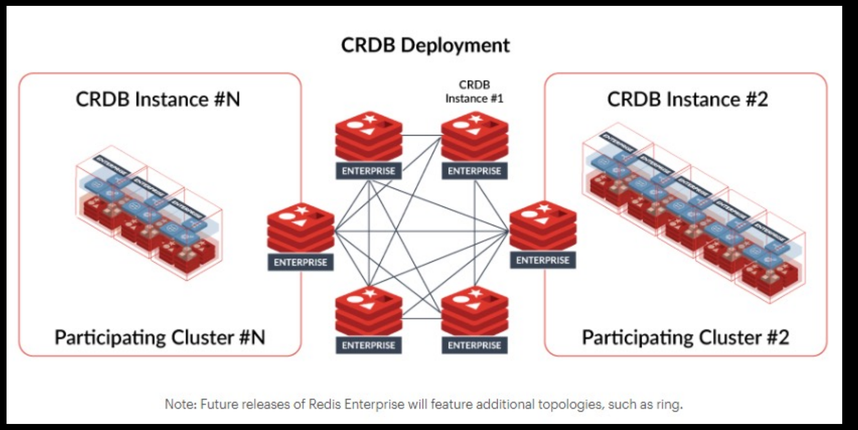
왜? Geo Replication 이 생소할까?

- Multi-IDC Deployment 를 수반
  - 비용증가
  - 관리 Point 증가
- 자연 재해에 대한 경험 부족



https://medium.com/streamnative/apache-pulsar-at-yahoo-japan-b7765bb7b58c

Redis의 GeoReplication



https://redis.com/redis-enterprise/technology/active-active-geo-distribution/

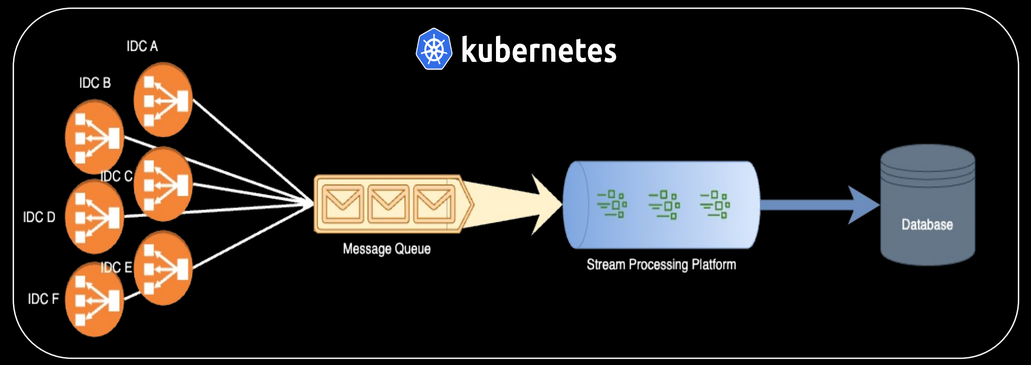
1.4 Geo Replication의 구성

NAVER
DEVIEW
2023

완벽하고 효율적인 Geo Replication을 위해서

- End-to-End 에 걸친 섬세한 설계가 필요
  - End-to-end Transaction
  - Message Delivery Guarantees
  - Exactly Once by Post Processing (eg. Deduplication)

- Trade Off 에 대한 면밀한 분석이 필요
  - Loss-Tolerant <-> Lossless Data Guarantee
  - Automatic <-> Manual Cluster 전환
  - Active-Active <-> Active-Standby 등



Cloud를 기반으로 Source 에서 Database 까지 이르는 전체 Pipeline 에 대한 Geo Replication 이 필요

1.5 Geo Replication의 분류

NAVER
DEVIEW
2023

Geo Replication 적용의 3가지 분류

Performance

- Local Read를 통한 Low Latency

Scalability

- 여러 사이트에 걸친 Load Balance

Availability

- 중단 없는 서비스 유지

※참조 문헌: Fine-grained consistency for geo-replicated systems, Cheng Li, Nuno Preguica, Rodrigo Rodrigues

Availability 중점의 Cost Efficient Geo Replication

- Eventual consistency 모델에 가까운 실용적인 Geo Replication

Cost

Recovery Speed

Data Loss

NAVER
DEVIEW
2023

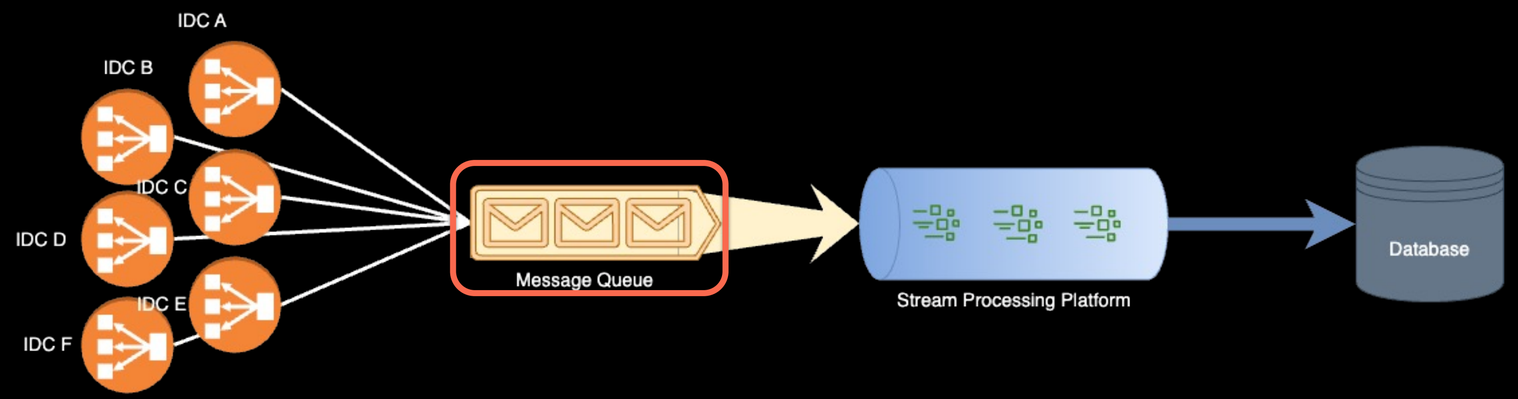
2. Message Queue in Geo Replication

2.1 Message Queue in Geo-Replication

NAVER
DEVIEW
2023

Key roles of Message Queue in Realtime Data Pipeline

- 1st Level Data Aggregation
- Sequential I/O with High TPS
- 찰나의 장애도 Critical



Revisiting Kafka in terms of Geo Replication

- Out-of-the-box: Native & built-in feature
- A/Synchronous Replication Support
- Various Metrics Visibility
- Automatic Cluster Switching API

2.2 Message Queue Geo Replication Key Elements

NAVER DEVIEW 2023

Key Element 1 : Out-of-the-box feature

- 개발 난이도와 관리 측면에서 중요한 포인트

Key Element 2: Synchronous & Asynchronous Replication

- Synchronous Replication은 데이터 정합성에 강점
- Asynchronous Replication은 Dynamic Scalability 적용 가능

Key Element 3 : Visible/Measurable Metrics

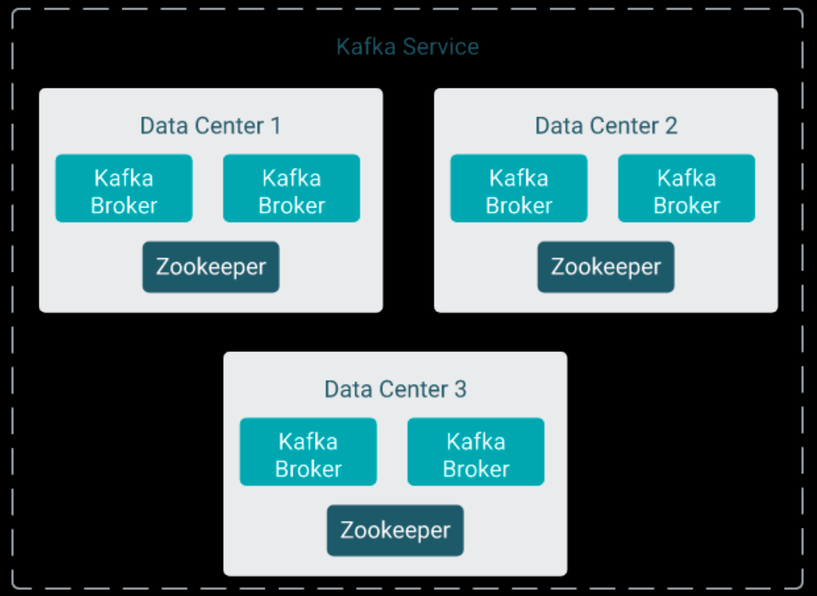
- Asynchronous Replication 의 Replication Lag 측정
- Subscription offset 정보 등의 확인 가능 여부

2.3 Kafka Stretch Cluster

NAVER
DEVIEW
2023

Synchronous Replication on Kafka

- Single Kafka Cluster on Multiple IDC
- Zookeeper Majority Quorums
- Inter Cluster Transfer Latency
- Static Resources



https://docs.cloudera.com/runtime/latest/kafka-overview/topics/kafka-overview-stretch-overview.html

Nominal Write $T_{(client \rightarrow broker)} + (T_{(broker \rightarrow broker)} * min.insync.replicas-1)$

Stretched Write $T_{(client \rightarrow broker)} + (T_{(broker \rightarrow broker)} * min.insync.replicas-1)$

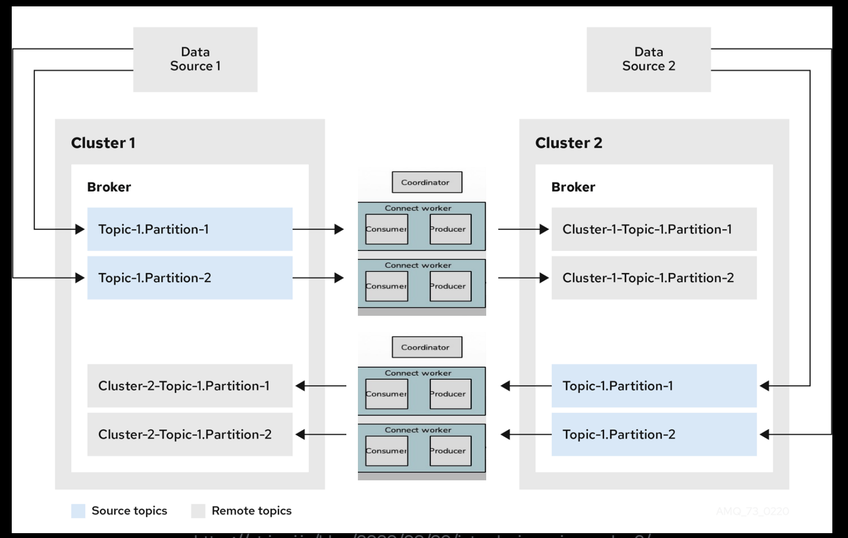
IDC 간 Network Latency 가 추가

2.4 Kafka MirrorMaker

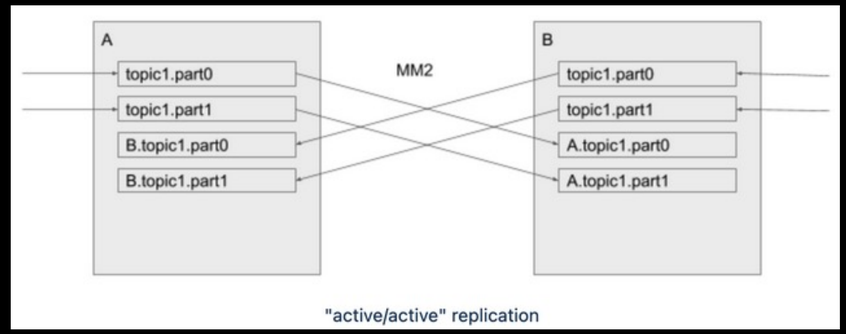
NAVER
DEVIEW
2023

- Asynchronous Replication on Kafka
- MirrorMaker2 features
  - Auto topic discovery & replication
  - Offset replication
  - At least once guarantee
  - Active-Active cluster support
  - Cycle detection

- Limitation of external feature
  - Active–Active 구성 시,
    동일 토픽 간 Multi-named Topic Consuming 필요



https://strimzi.io/blog/2020/03/30/introducing-mirrormaker2/



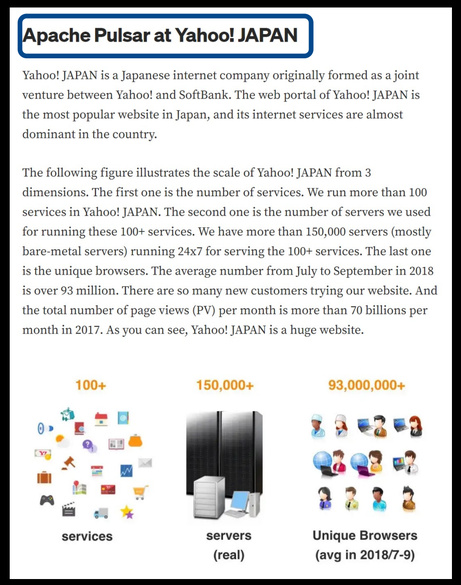
https://cwiki.apache.org/confluence/display/KAFKA/KIP-382%3A+MirrorMaker+2.0#KIP382:MirrorMaker2.0-Cycledetection

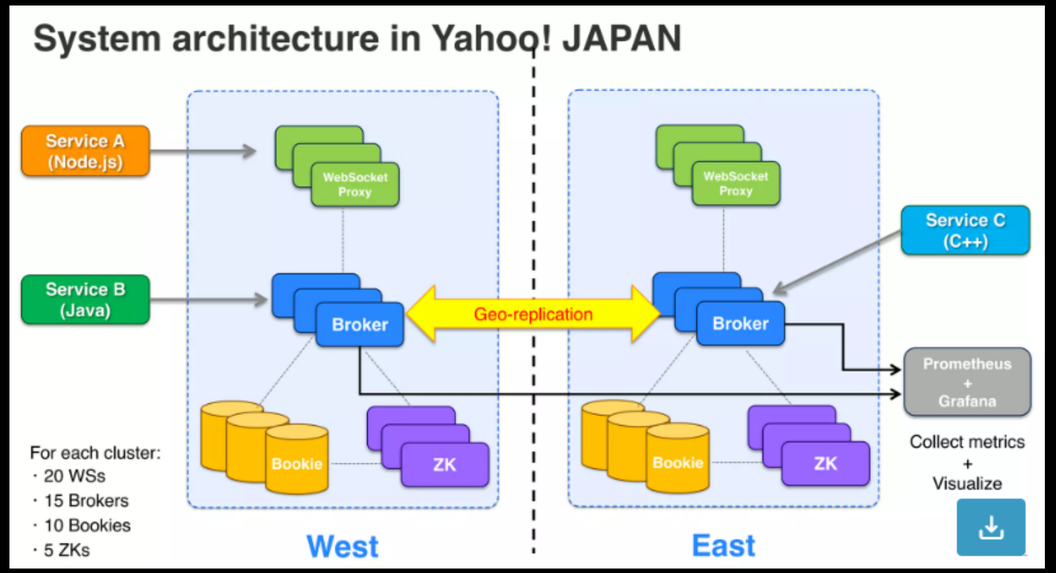
2.5 Alternative Platform Research

NAVER
DEVIEW
2023

Geo Replication in Yahoo! Japan

- Yahoo! Japan 에서의 Geo Replication Story에 주목
  - 미국, 중국의 Replication 기법 보다는 일본이 우리에게 상대적으로 Realistic
    - 자연재해가 비교적 우리보다 많아 경험 풍부
- Verified Geo Replication functionalities on Production Level





https://medium.com/streamnative/apache-pulsar-at-yahoo-japan-b7765bb7b58c

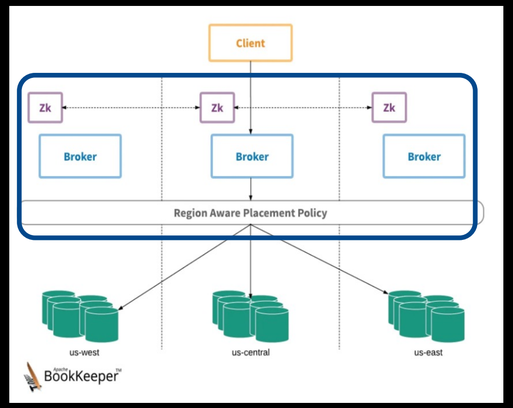
https://www.google.co.kr/url?sa=i&url=https%3A%2F%2Fwww.docswell.com%2Fs%2Fydnjp%2FK46YJZ-2021-10-05-150329&psig=AOvVaw100tMeJ3XqYsMpluvQoVAw&ust=1673946176443000&source=images&cd=vfe&ved=2ahUKEwig5YPL3cv8AhVJFYgKHWc8D8sQr4kDegUIARDQAQ

2.6 Pulsar Geo Replication

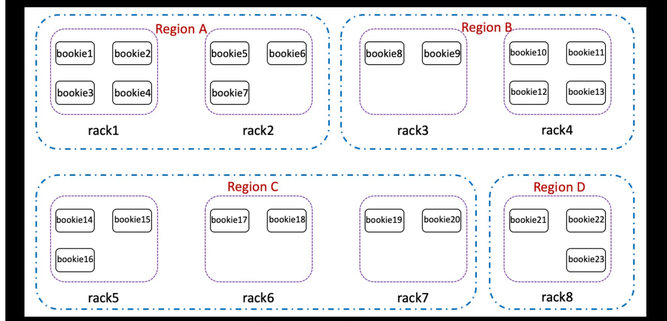
NAVER DEVIEW 2023

Synchronous Replication

- Region Aware Placement Policy
- Muti-Cluster + 단일 Zookeeper ensemble
- Multi IDC Data Write Ack



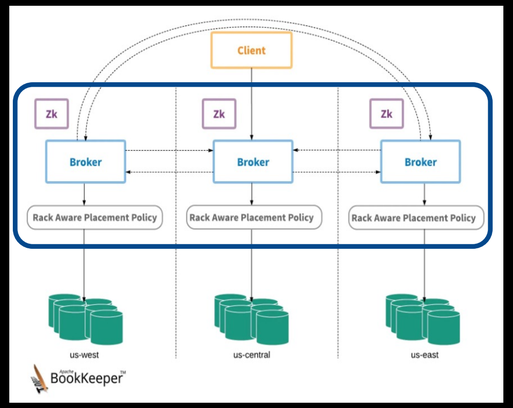
Synchronous Replication



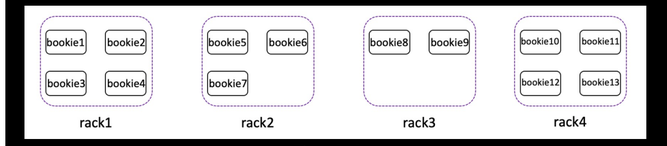
RegionAware

Asynchronous Replication

- Rack Aware Placement Policy
- Write Ack 없이 동작
- Replication Lag 발생 가능



Asynchronous Replication (Full Mesh)



RackAware

https://www.splunk.com/en_us/blog/it/geo-replication-in-apache-pulsar-part-1-concepts-and-features.html

2.7 Pulsar Asynchronous Replication 적용

NAVER DEVIEW 2023

Geo-Replication Command Example

- Asynchronous Replication across multiple cluster

Geo Replication (3 types of commands)

1. 다음과 같은 명령어를 사용하여 2개의 Cluster 를 서로 연결

```
[irteam@command1.server pulsar]$ cat connect_to_cluster_2.sh
#!/bin/sh

bin/pulsar-admin clusters create \
        --broker-url pulsar://command2.server:8080 \
        --url http://command2.server:6650/ \
        pulsar-nds
```

```
[irteam@command2.server pulsar]$ cat connect_to_cluster_1.sh
#!/bin/sh

bin/pulsar-admin clusters create \
        --broker-url pulsar://command1.server:8080 \
        --url http://command1.server:6650/ \
        pulsar-nlog
```

2. 복제 하려고하는 tenants 에 대해서 다음과 같은 명령어 입력

- 데이터를 Push 하려고 하는 cluster 에서 실행할 명령어 (원본 cluster)

```
bin/pulsar-admin tenants update nlog --admin-roles my-admin-role --allowed-clusters pulsar-nlog,pulsar-nds
```

- 데이터를 Pull 하려고 하는 cluster 에서 실행할 명령어 (복사본 cluster)

```
bin/pulsar-admin tenants create nlog --admin-roles my-admin-role --allowed-clusters pulsar-nlog,pulsar-nds
```

3. 복제 하려고 하는 namespace 에 대해서 다음과 같은 명령어 입력

- 데이터를 Push 하려고 하는 cluster 에서 실행할 명령어 (원본 cluster)

```
bin/pulsar-admin namespaces set-clusters nlog/nlog --clusters pulsar-nds,pulsar-nlog
bin/pulsar-admin namespaces set-clusters nlog/source --clusters pulsar-nds,pulsar-nlog
bin/pulsar-admin namespaces set-clusters nlog/refined --clusters pulsar-nds,pulsar-nlog
```

- 데이터를 Pull 하려고 하는 cluster 에서 실행할 명령어 (복사본 cluster)

```
bin/pulsar-admin namespaces create nlog/nlog
bin/pulsar-admin namespaces set-clusters nlog/nlog --clusters pulsar-nds,pulsar-nlog
bin/pulsar-admin namespaces create nlog/source
bin/pulsar-admin namespaces set-clusters nlog/source --clusters pulsar-nds,pulsar-nlog
bin/pulsar-admin namespaces create nlog/refined
bin/pulsar-admin namespaces set-clusters nlog/refined --clusters pulsar-nds,pulsar-nlog
```

Visibility of Replication Status

Search Tenants | + New Tenant

Tenant	Namespaces	Allowed Clusters	Admin Roles	In Rate	Out Rate	In Throughput	Out Throughput	Storage Size	Actions
nlog	2	pulsar-nlog pulsar-nds	my-admin-role	↓ 10.48	↑ 7.63	↓ 20K	↑ 14K	2.2G	Edit Delete
public	2	pulsar-nlog		↓ 0.08	↑ 0.00	↓ 59	↑ 0	49.2M	Edit Delete
pulsar	1	pulsar-nlog		↓ 0.00	↑ 0.00	↓ 0 Bytes	↑ 0 Bytes	0 Bytes	Edit Delete

Search Tenants | + New Tenant

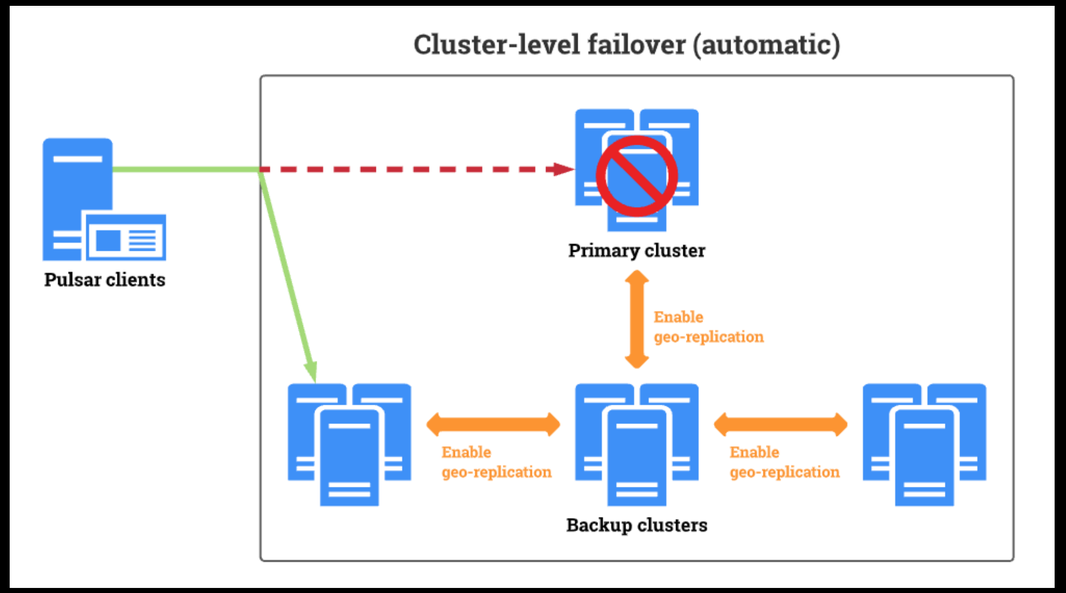
Tenant	Namespaces	Allowed Clusters	Admin Roles	In Rate	Out Rate	In Throughput	Out Throughput	Storage Size	Actions
nds	4	pulsar-nds		↓ 161.1K	↑ 80.45K	↓ 143M	↑ 76M	83.94G	Edit Delete
nlog	2	pulsar-nlog pulsar-nds	my-admin-role	↓ 10.12	↑ 7.63	↓ 19K	↑ 14K	2.2G	Edit Delete
public	2	pulsar-nds		↓ 0.05	↑ 0.00	↓ 56	↑ 0	49.2M	Edit Delete
pulsar	1	pulsar-nds		↓ 0.00	↑ 0.00	↓ 0 Bytes	↑ 0 Bytes	0 Bytes	Edit

2.8 Pulsar Important Features

NAVER
DEVIEW
2023

Pulsar Automatic Cluster Failover API

- 데이터 수집 서버에서 Pulsar Client를 활용하여 Message Produce
- 버전 2.10 부터 Native API 로 Secondary Cluster를 지정하여 활용 가능
- Primary Cluster의 장애 탐지 시 Secondary Cluster로 자동 전환



https://pulsar.apache.org/docs/2.11.x/concepts-cluster-level-failover/

Failover Trigger 조건

- Network failure
- Power failure
- Service error
- Crashed storage space

```
1
2   private PulsarClient getAutoFailoverClient() throws PulsarClientException {
3
4         ServiceUrlProvider failover = AutoClusterFailover.builder()
5                .primary("pulsar://localhost:6650")
6                .secondary(Collections.singletonList("pulsar://other1:6650","pulsar://other2:6650"))
7                .failoverDelay(30, TimeUnit.SECONDS)
8                .switchBackDelay(60, TimeUnit.SECONDS)
9                .checkInterval(1000, TimeUnit.MILLISECONDS)
10               .secondaryTlsTrustCertsFilePath("/path/to/ca.cert.pem")
11     .secondaryAuthentication("org.apache.pulsar.client.impl.auth.AuthenticationTls",
12 "tlsCertFile:/path/to/my-role.cert.pem,tlsKeyFile:/path/to/my-role.key-pk8.pem")
```

2.9 Message Queue 관점의 Pulsar vs Kafka

NAVER DEVIEW 2023

PULSAR

Pros

- Out of the box feature
- Asynchronous Replication 의 경우 간단히 구축 가능
- A/A 구성 시 동일 Topic Name 이슈 없음
- 2.10 버전부터는 Automatic Cluster Failover API 활용 가능

Cons

- Synchronous Replication 은 Topology 적으로 복잡한 구조
- Subscription Offset 에 대한 Exactly Once 지원 계획 없음
- 문서화, 사례 정보 부족

Apache Kafka

Pros

- 다양한 구축 사례 등의 많은 정보
- Subscription offset 에 대한 Exactly Once (PIP-656) 진행중

Cons

- MirrorMaker2는 별도의 설치 및 Cluster 구축 필요
- A/A 구성에서 동일 Topic-Name 구성 불가
- Static Resource 기반의 Replication 구조

NAVER
DEVIEW
2023

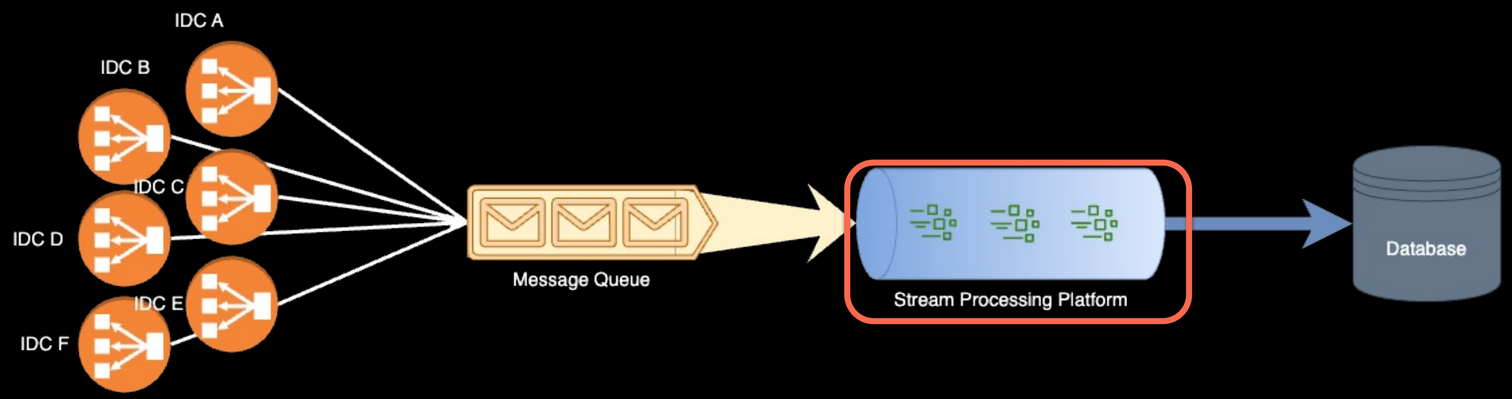
3. Realtime Data Processing in Geo Replication

3.1 Realtime Data Processing in Geo-Replication

NAVER
DEVIEW
2023

Realtime Streaming Data Enrichment

- Geo Replication 의 관점에서 관리 포인트
  - Network Endpoints
  - Offset Management



Revisiting Stream Processing Platforms in terms of Geo Replication

- Message Queue 시스템과 얼마나 단순하게, 효율적으로 동작하는가?
- Dynamic scale in/out 이 가능한지?
- 얼마나 Data Reusability가 높은가?

3.2 Realtime Data Processing Geo Replication Key Elements

NAVER
DEVIEW
2023

Key Element 1 : Cloud Friendliness

- 장애시 Catch-up reads와 평상시의 Tailing reads 사이의 탄력적 Resource Scaling
- Cost Efficient 관점에서 중요한 Feature

Key Element 2: Reduced Management with Integrated Processing

- Message Queue System 과 연결 point를 최소화
- Fewer Platforms Pipeline이 Geo Replication에 보다 유리

Key Element 3: Data Reusability

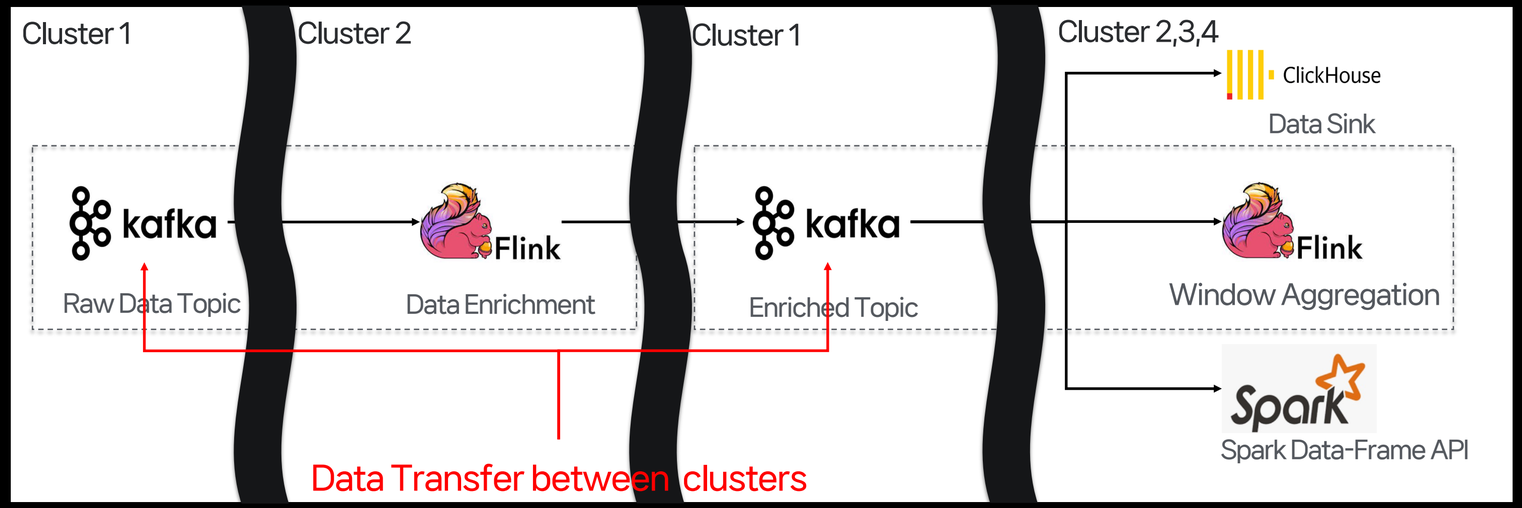
- Data Enrichment 결과 데이터를 얼마나 효과적으로 재사용할 수 있는가?
- 장애 복구 시 효과적인 리소스 사용

3.3 Revisiting Typical Realtime Processing

NAVER
DEVIEW
2023

Low Data Reusability Problem

- Kafka -> Flink 의 Chain은 Cluster to Cluster 통신을 통해서 동작
  -> 높은 네트워크 비용
  -> API를 통한 데이터 전송 코드 개발 및 최적화 필요
- Cluster 간 통신을 줄이고 Data Reusability 가 높은 Platform 에 대해서 Research

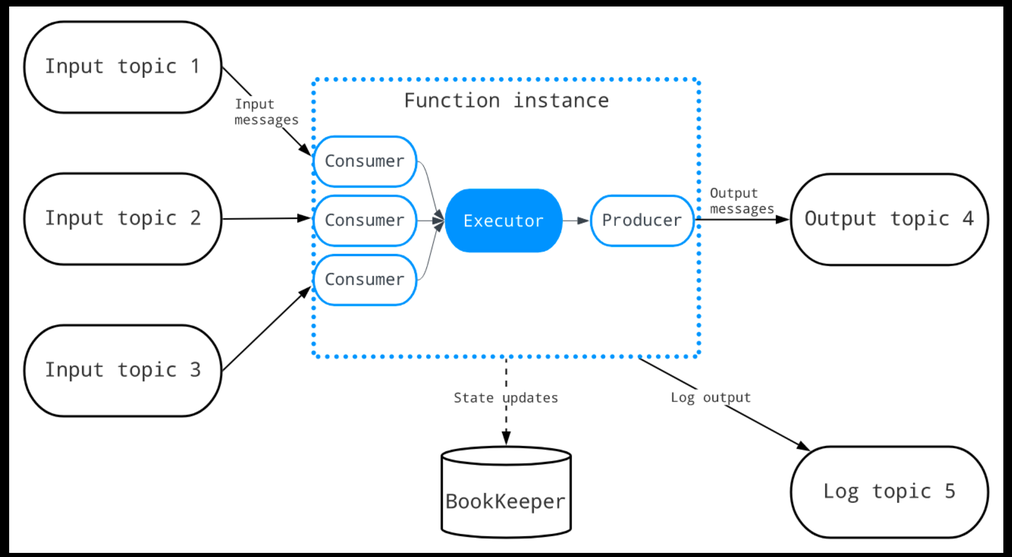


3.4 Alternative Platform Research

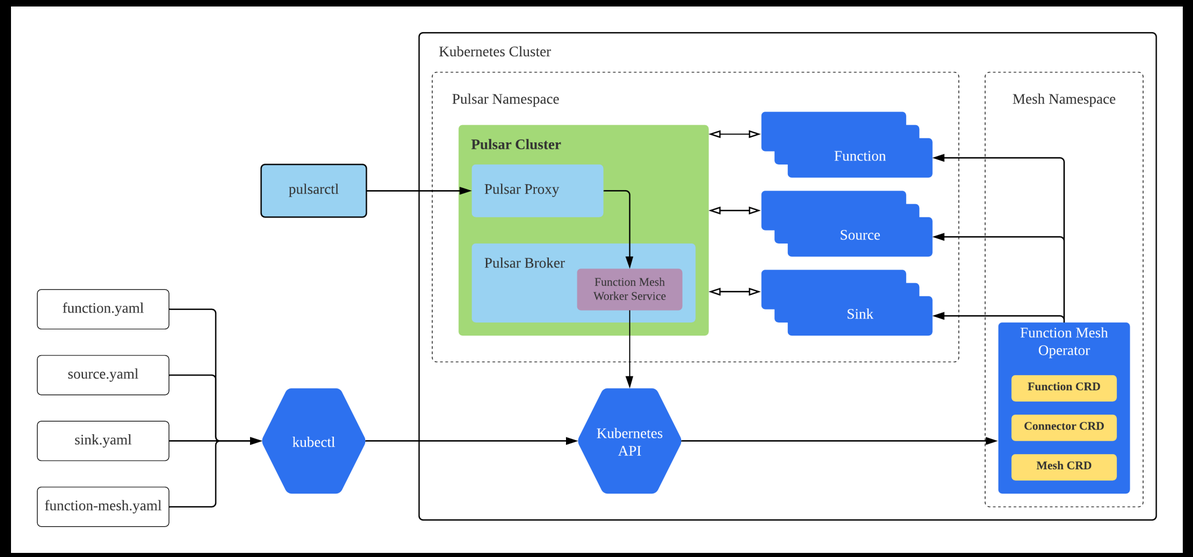
NAVER
DEVIEW
2023

Function and FunctionMesh in Pulsar

- Function: Dynamically Scalable Lambda Like Light-weight processing with Chaining
- Output Topic 을 통한 간편한 In-Cluster 데이터 이동
- Resource Utilization, Prometheus Metrics 연동해 HPA 구성이 가능
- Source, Functions, Sink로 나눠 단계적인 데이터 처리가 가능한 FunctionMesh 적용



https://pulsar.apache.org/docs/2.11.x/functions-concepts/



https://functionmesh.io/docs/

3.5 Streaming Engines Comparison

NAVER DEVIEW 2023

Summarized Comparison of Streaming Engines

	Flink	Spark	Pulsar
API Set	Rich (WindowJoin, SessionWindow...)	Rich (Spark Structured Streaming...)	Relatively Compact (복잡한 Window 연산 불가)
Realtime/ MicroBatch	Realtime	MicroBatch	Realtime
Productability	Need to build complete Application	Leverage Spark's dataframe API	Fast and easy

3.6 Messaging and Processing with Pulsar

NAVER DEVIEW 2023

Pulsar를 통한 Window Aggregation

- Key_Shared Subscription을 활용하여 기존 keyBy()연산을 대체
- 자체 Window Function 활용

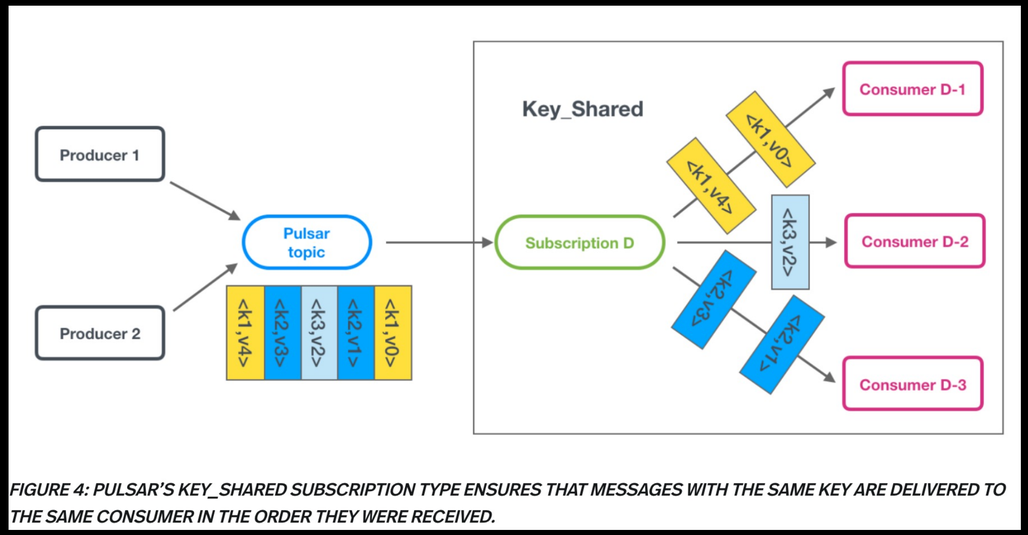


FIGURE 4: PULSAR'S KEY_SHARED SUBSCRIPTION TYPE ENSURES THAT MESSAGES WITH THE SAME KEY ARE DELIVERED TO THE SAME CONSUMER IN THE ORDER THEY WERE RECEIVED.

https://streamnative.io/blog/scalable-stream-processing-pulsars-key-shared-subscription

∴ Messaging + Processing + Sinking을 Pulsar FunctionMesh 하나로 대체

- Data Reusability 향상
- In Cluster 연산을 통한 Network 관리포인트 감소

3.7 Drawback of using HPA on Consumer

NAVER DEVIEW 2023

HPA로 인한 Window Split

1) HPA로 인해서 Consumer 구성 변경 발생
2) Consumer 구성 변경 → KeyBy() 결과에 변화 발생
3) Window Split 현상 발생

2022-01-05	12:03:10	12:03:20	네이버 블로그	30-40	F	Android	Browser	4	사용자 A, 사용자 B 사용자 C, 사용자 D

2022-01-05	12:03:10	12:03:20	네이버 블로그	30-40	F	Android	Browser	2	사용자 A, 사용자 B	s1
2022-01-05	12:03:10	12:03:20	네이버 블로그	30-40	F	Android	Browser	2	사용자 C, 사용자 D	s2

- 따라서 Window Aggregation 함수가 다음의 수식을 만족하지 않는다면 문제 발생

$$F(s1\ UNION\ ALL\ s2)\ =\ F(F(s1)\ UNION\ ALL\ F(s2))$$

- 해결 방법 예시: Rolled up Cube 구성 시 Group By 필요

NAVER
DEVIEW
2023

4. Cloud Friendliness in Geo Replication

4.1 Geo Replication in Cloud

NAVER
DEVIEW
2023

Platforms with Geo Replication

- Cloud 환경에 적합한 Data Processing Platform 이 필요
  - Storage Architecture에 Dependent 할 수록 Dynamic Scalability 저하
  - Cost Efficient Geo Replication 을 위해선 Cloud Friendly Platform 이 필요
    - 예시) 1. Repartition 없이 Scaling 될 수 있는 Messaging Platform
    - 2. 장애 시 HPA 를 통해 빠르게 Catch-up Read 가능한 Processing Platform
    - 3. Standby 시 Resource 사용을 최소화 할 수 있는 Messaging Platform

Apache Kafka

kubernetes

PULSAR

APACHE Spark™

Flink

4.2 Geo Replication in Cloud Key Elements

NAVER
DEVIEW
2023

Key Element 1: HPA 기반의 Dynamic Resource 할당

- HPA를 기반으로 한 탄력적 Resource Scaling 을 통해서 빠르게 장애 대응
- Active/Active, Active/Standby 와 같은 다양한 Resource 구성에 효과적으로 대응
- Cost Efficient Geo Replication을 위한 필수적 특징

Key Element 2: Modularly Designed Sub-system

- Monolithic Design 의 경우 Resource 를 효과적으로 Scaling 하기에 불리한 구조
- Micro Service Architecture 에 유리한 Modular design

Key Element 3: No Penalty after Scaling In/Out

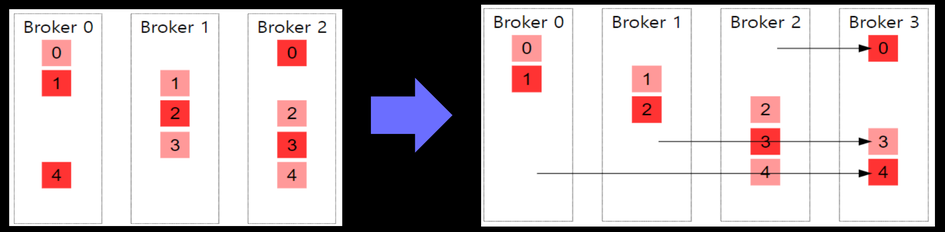
- Resource Scaling 으로 성능 저하/기능 제약이 있는 경우 Cloud의 Resource 환경에 부적합

4.3 Dynamic Scalability of Kafka

NAVER
DEVIEW
2023

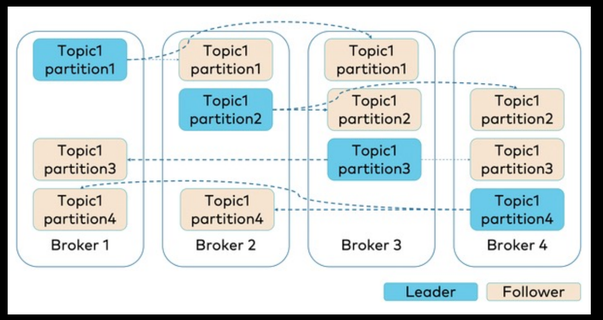
Kafka's architecture and Dynamic Scalability

- Strongly combined broker and partitions
- 증설 시 새로운 노드에 온전한 Partition을 위해 Data Migration 필요
  - 매 증설마다 Performance Degradation 이 발생 (Disk, Network, Computation cost)



https://teke42.wordpress.com/2018/08/29/scaling-kafka/

- Storage and Broker 가 결합되어 있는 구조
  - Message Broker와 Storage 를 독립적으로 Scale 할 수 없는 구조



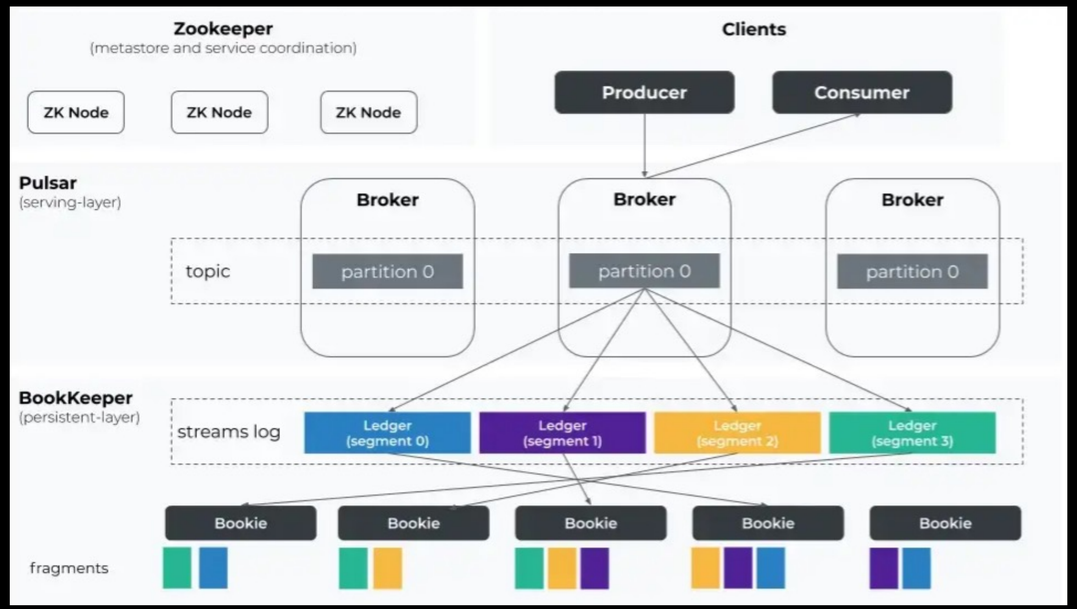
4.4 Dynamic Scalability of Pulsar

NAVER DEVIEW 2023

Pulsar's architecture and Dynamic Scalability

- Layered Architecture and Independent Storage Sub-system
  - Node 추가 시 Partition 구조 없이 새로운 Bookie 로 저장되는 구조
  - Bookie 추가 시 Data Migration 으로 인한 성능 저하가 없음
  - 새 데이터는 새 Bookie 로 저장되는 구조

- Broker 의 Independent Scalability
  - 독립적으로 동작
  - Stateless
  - Auto-Scaling in/out 가능



https://medium.com/streamthoughts/introduction-to-apache-pulsar-concepts-architecture-java-clients-71f1a30b75d6

4.5 Comparison of Cloud Friendliness

NAVER DEVIEW 2023

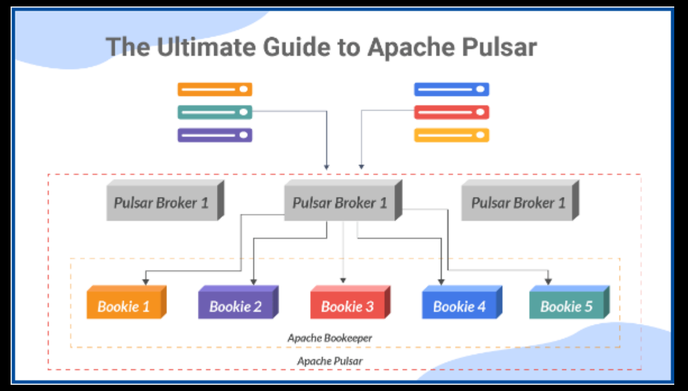
Pulsar vs Kafka

- Monolithic architecture vs Multi-layered architecture
  - Kafka : Broker와 Storage 가 결합된 구조
  - Pulsar : Broker와 Storage (BookKeeper)와의 완전한 분리

https://www.splunk.com/en_us/blog/it/apache-pulsar-architecture-designing-for-streaming-performance-and-scalability.html

- Broker
  - Kafka : Storage 와 결합되어 있어 Stateless 함에도 불구, Dynamic Scaling 불가
  - Pulsar : Storage 와 분리되어 있어 Storage와 별개로 독립적 Dynamic Scaling 가능

- Storage
  - Kafka : 특정 Topic의 특정 Partition을 특정 노드에 할당
  - Pulsar(BookKeeper) : 데이터를 Segment화 하고 Stripping하여 분산 저장



https://medium.com/streamthoughts/introduction-to-apache-pulsar-concepts-architecture-java-clients-71f1a30b75d6

4.6 Pulsar's feature summary

NAVER
DEVIEW
2023

Pulsar's Pros

- "Cloud Native" : Modularity design 과 Cloud Friendly
- "Out-of-box Geo replication" : MM2 와 같은 외부 툴 없이 Geo Replication 을 자체 제공
- "Function" : In Cluster Processing Runtime 을 제공
    - 외부 Connector 의 지원이 부족한 Pulsar 의 단점을 대부분 Cover
    - 자체 Lambda Processing Platform 제공
    - 데이터를 외부로 꺼내지 않더라도 자체적으로 처리 하는 것이 가능
    - Out-of-box feature
- "Multi-Tenant" : Topic Hierarchy가 Kafka 에는 없음

4.7 Integrated Package Manager

NAVER
DEVIEW
2023

Package Manager 를 통한 Application/배포 관리

- 장애시에 배포 시스템이 문제가 있다면?
  - Git 를 통한 Version Control -> Pulsar Package Manager

- Pulsar Package manager 를 통한 Version 관리 장점
  - 외부에 배포 시스템으로 부터 독립적
  - Out of the box feature

내장된 Package Manger

```
enablePackagesManagement=true
packagesManagementStorageProvider=org.apache.pulsar.packages.management.storage.bookkeeper.BookKeeperPackagesStorageProvider
packagesReplicas=1
packagesManagementLedgerRootPath=/ledgers
```

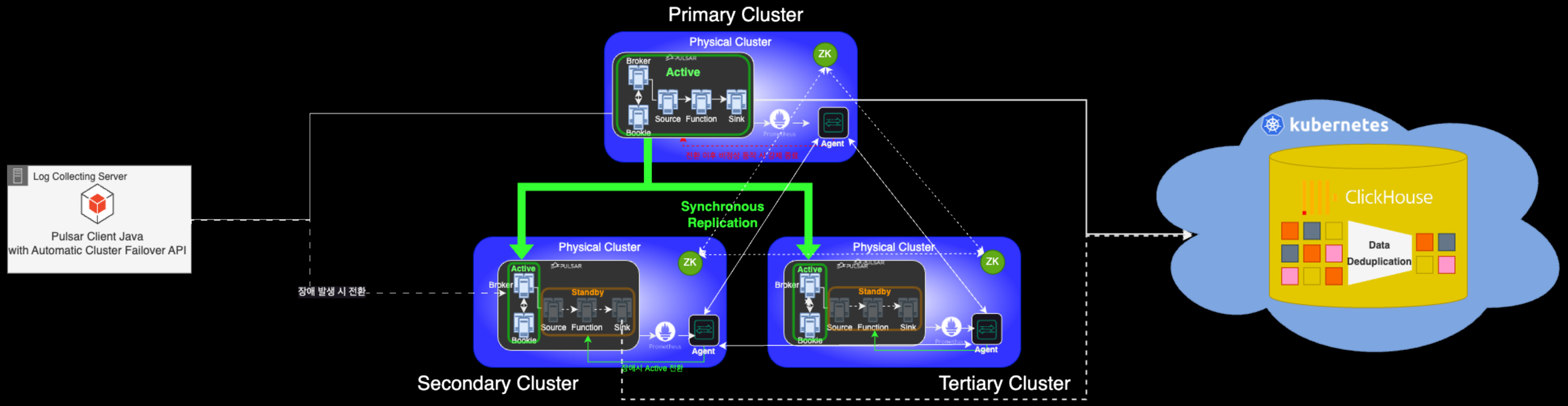
```
sink://public/default/mysql-sink@1.0
function://my-tenant/my-ns/my-function@0.1
source://my-tenant/my-ns/mysql-cdc-source@2.3
```

NAVER
DEVIEW
2023

5. Geo Replication in Practice

5.1 Standard Synchronous Geo-Replication (Type 0)

NAVER DEVIEW 2023



Features

- Full 3 IDC Cluster
- Region-Aware Ensemble Placement Policy
- Automatic Cluster Failover API

Pros

- 높은 Fault Tolerant

Cons

- High Resource
- Inter-Cluster Transfer Latency
- Non Dynamic Scalable

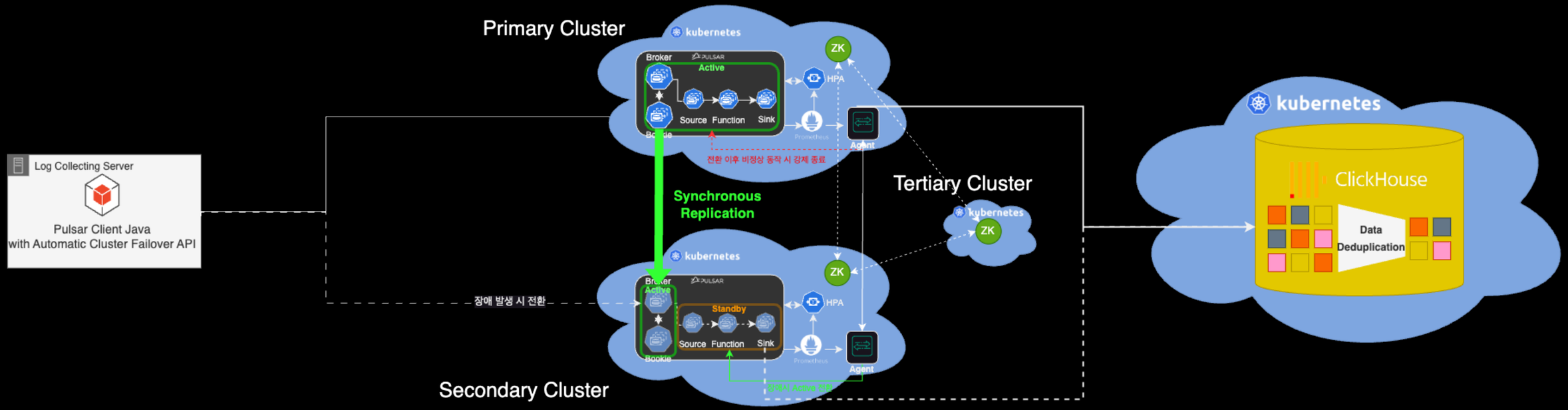
Inter-Cluster Latency

단위: ms

	A	B	C
A	0.02	0.48	2.52
B	0.46	0.01	1.6
C	2.6	1.5	0.03

5.2 Two-point-Five Cluster Synchronous Geo-Replication (Type 1)

NAVER DEVIEW 2023



Features

- 3 IDC Stretched Zookeeper
- Region-Aware Ensemble Placement Policy
- Active-Active Topic Data
- Active-Standby Processing Function
- Data deduplication on DB side

Pros

- Lossless Geo Replication
- Dynamic Scalable on Pulsar Cluster

Cons

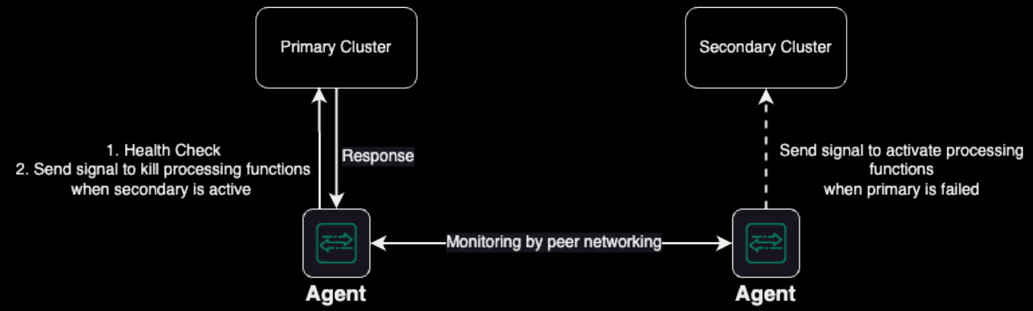
- Due to Zookeeper Ensemble & Quorum, at least 3 Cluster needed
- Inter-Cluster Transfer Latency

5-3. Agent의 Secondary Pulsar Function Trigger 조건, 처리

NAVER DEVIEW 2023

Secondary Pulsar Function이 시작하는 조건

- Trigger Condition
  - 1초마다 Service Ping Check
  - Storage Fail Check
  - Clickhouse Insert rate Outlier detection



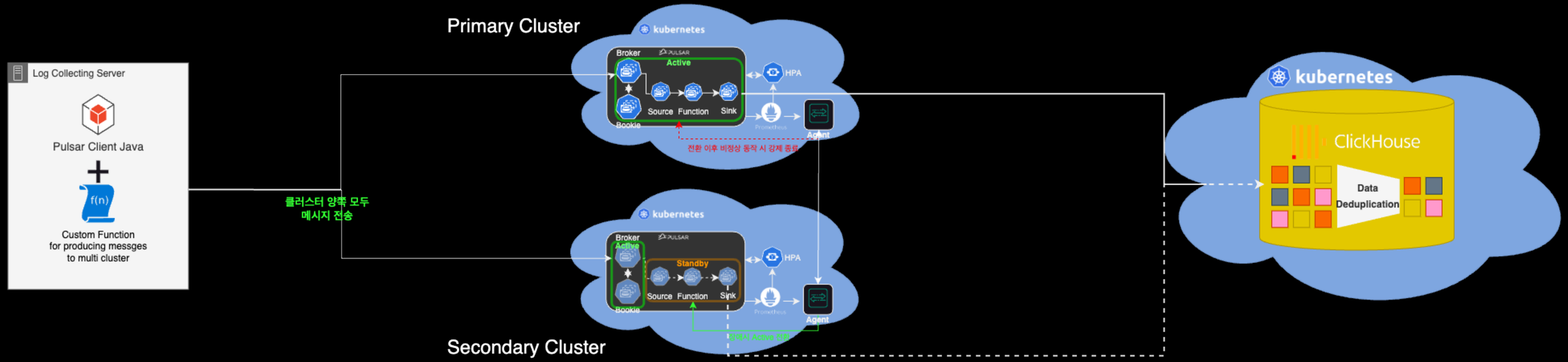
- What is triggered
  - 단일 YAML로 실시간 Processing Pipeline 전체 배포
  - Replicated Application using Pulsar Package manager
  - Cluster 전환 시 Circuit Break 로
    기존 Pulsar Function Instance 삭제

```yaml
functions:
  - name: function-lcs
    namespace: pulsar-nds
    className: refine_lcs.RefineFunctionLCS
    forwardSourceMessageProperty: true
    replicas: 40
    maxReplicas: 80
    input:
      topics:
      - persistent://nds/source/lcs-topic
    output:
      topic: persistent://nds/refined/lcs-refined-topic
    resources:
      requests:
        cpu: "0.8"
        memory: 524M
      limits:
        cpu: "0.8"
        memory: 524M
    pod:
      autoScalingMetrics:
      - type: Resource
        resource:
          name: memory
          target:
            type: Utilization
            averageUtilization: 60
      - type: Pods
        pods:
          metric:
            name: pulsar_function_received_total_1min_total
            selector:
              matchLabels:
                name: functionmesh-nds-realtime-function-lcs
                namespace: pulsar-nds
          target:
            type: AverageValue
            averageValue: "40000"
    pulsar:
      pulsarConfig: "pulsar-nds"
    python:
      pyLocation: function://public/functions/lcs-refine-function@0.1
```

```yaml
sinks:
  - name: sink-lcs
    className: com.riferrei.pulsar.sink.custom.CustomSinkConnector
    replicas: 40
    maxReplicas: 40
    input:
      topics:
      - persistent://nds/refined/lcs-refined-topic
      typeClassName: java.lang.String
    sinkConfig:
      columnList: "sp,url,os_type,br_type,br,age,gender,idno,bc"
      tableName: "secret"
      clickhouseJDBCUrl: "secret"
      clickhouseUser: "secret"
      clickhousePassword: "secret"
    resources:
      requests:
        cpu: "0.8"
        memory: 1G
      limits:
        cpu: "0.8"
        memory: 1G
    pod:
      autoScalingMetrics:
      - type: Resource
        resource:
          name: memory
          target:
            type: Utilization
            averageUtilization: 60
      - type: Pods
        pods:
          metric:
            name: pulsar_sink_received_total_1min
            selector:
              matchLabels:
                name: functionmesh-nds-realtime-sink-lcs
                namespace: pulsar-nds
          target:
            type: AverageValue
            averageValue: "40000"
    pulsar:
      pulsarConfig: "pulsar-nds"
    java:
      jarLocation: function://public/functions/nds-sink@0.1
      jar: custom-sink-connector-0.0.1.jar
    clusterName: pulsar-nds
    autoAck: true
```

5.4 Client-Side Synchronous Geo Replication (Type 2)

NAVER DEVIEW 2023



Features

- Transaction API on Client Side
- Active-Active Topic Data
- Active-Standby Processing Function
- Data deduplication on DB side

Pros

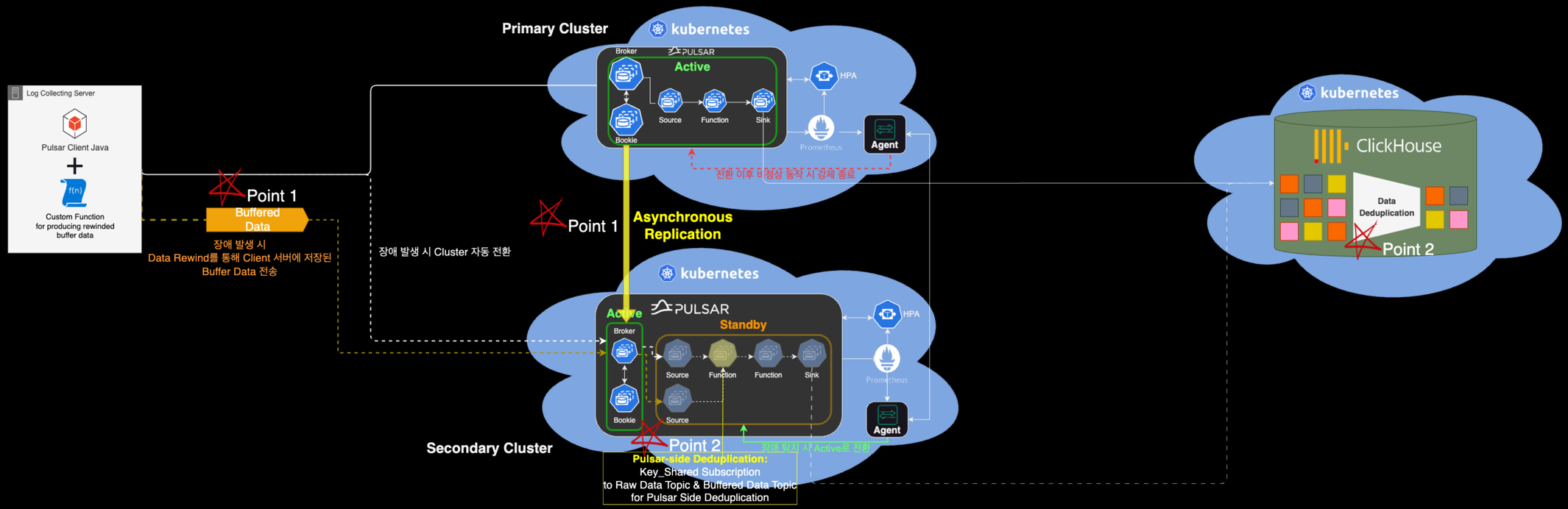
- 2 IDC Synchronous Data Replication

Cons

- Client Side High Resource & Latency
- Not using Automatic Cluster Failover API

5.5 Lossless Asynchronous Geo Replication (Type 3)

NAVER DEVIEW 2023



Features

- Automatic Cluster Failover API
- Active-Active Topic Data
- Active-Standby Processing Function
- Data deduplication on Pulsar & DB side
- Client Side Buffer 활용 Data Rewind

Pros

- Easy to develop and manage
- Low Resources
- No Inter-Cluster latency

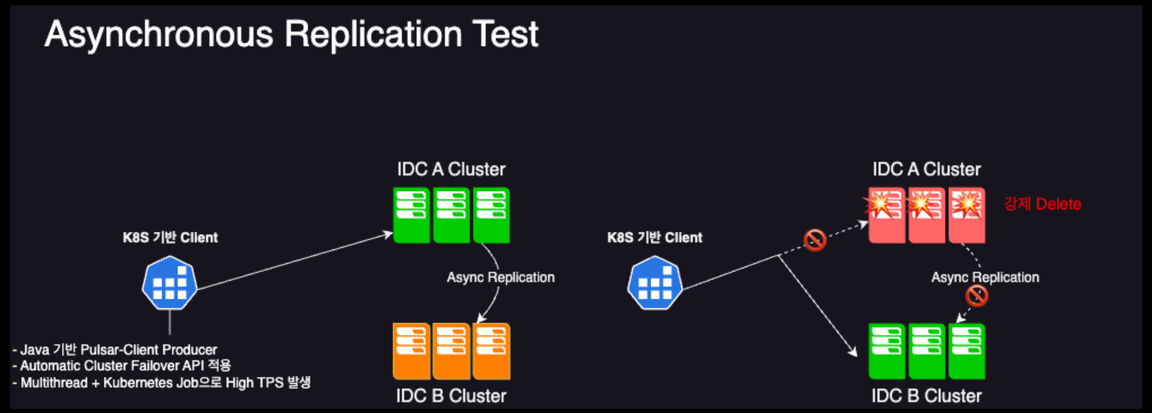
Cons

- Need to manage the Client Buffer Size

5.6 Point 1: Replication Lag & Data Rewind

NAVER DEVIEW 2023

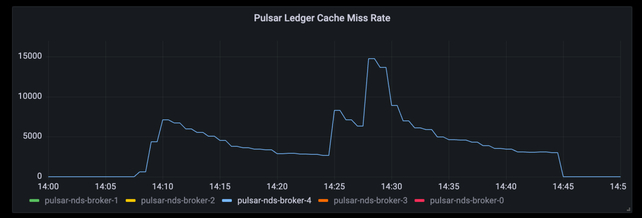
Replication Lag



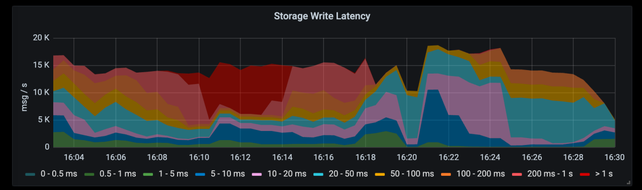
TPS	총 유입 시간	유실 비율
1k	2분	0%
10k	2분	0%
10k	30분	0.1%
30k	30분	1%
50k	30분	3%

TPS	Replication Lag
10k	
30k	
50k	

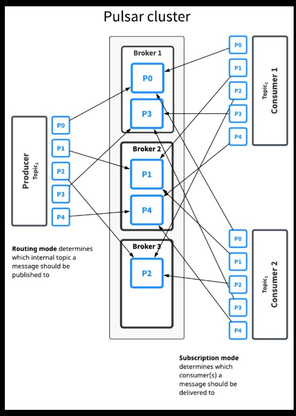
➢ Factors which affect the Replication Lags on Pulsar



- Data Journaling and Ledger



- Storage I/O Performance



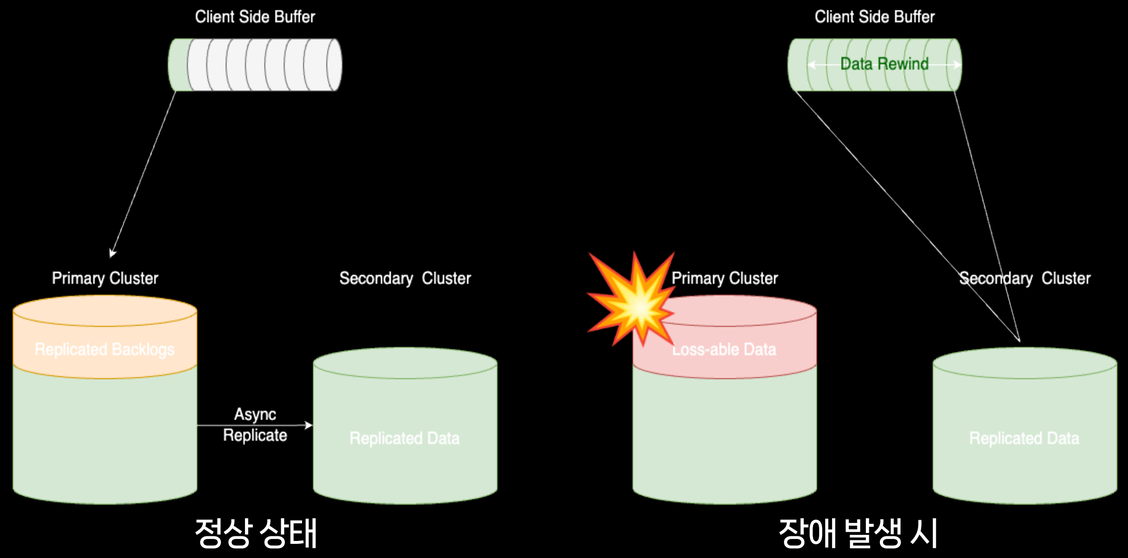
- Pulsar Message Routing Mode

5.6 Point 1: Replication Lag & Data Rewind

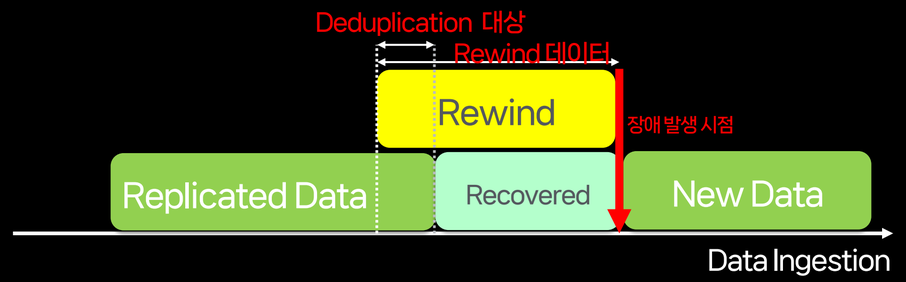
NAVER DEVIEW 2023

Data Rewind on Producers

- Client Side Buffering을 이용한 Data Rewind 적용
  - Fixed-size buffering
  - Adaptive buffering
  - Cluster Switch 시 File Offset 변경을 통해서 데이터 Rewind



- Side Effects
  - Data Deduplication 필요
  - Windows 연산에 대한 시간 경계와 이로 인한 Row Split
  - Rollup Style Database는 일부 산술 연산 시 잘못된 결과 발생 가능
  - Message Ordering Guarantee 가 필요한 경우 추가 작업이 필요



5.7 Point 2: Data Deduplication

NAVER DEVIEW 2023

2 Way Data Deduplication

DB-Side Deduplication

- Data Deduplication on Append Only Database (OLAP)
  - Clickhouse의 Replacing Merge Tree 를 이용한 DB-Side Deduplication

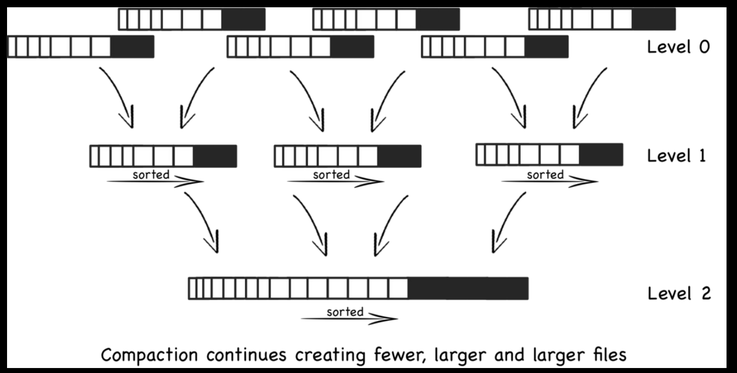
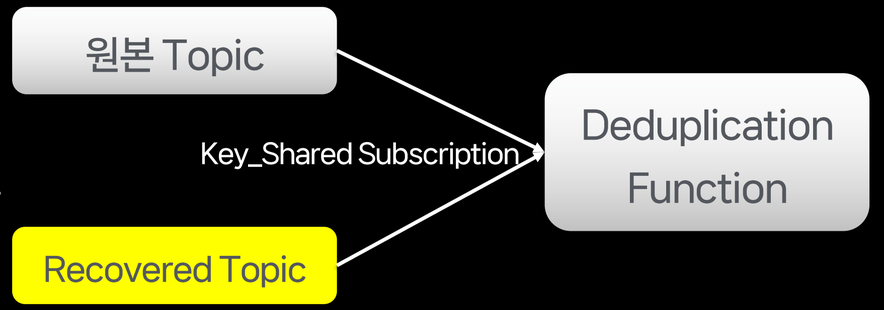


Diagram illustrating compaction of data in a log-structured merge tree

Pulsar-Side Deduplication

- Rewind 된 데이터를 별도의 Topic 에 저장
- 별도의 Platform 없이 Deduplication 을 위해서 Pulsar Function 을 이용
- Key_Shared Subscription + Sequence ID 로 구현
- Replication Lag으로 삭제된 데이터를 찾는 것이 가능



Sequence ID Each Pulsar message belongs to an ordered sequence on its topic. A message's sequence ID is its ordering in that sequence.

5.8 Comparison of 4 Types

NAVER DEVIEW 2023

	Main Features	Development	Resource Cost	Scalability	Replicated Data Consistency	Performance
Type 0		Easy	$$$	Low	High	Low
Type 1		Middle	$$	Middle	High	Middle
Type 2		Middle	$$	Middle	Middle	Middle
Type 3		Hard	$	High	Low	High

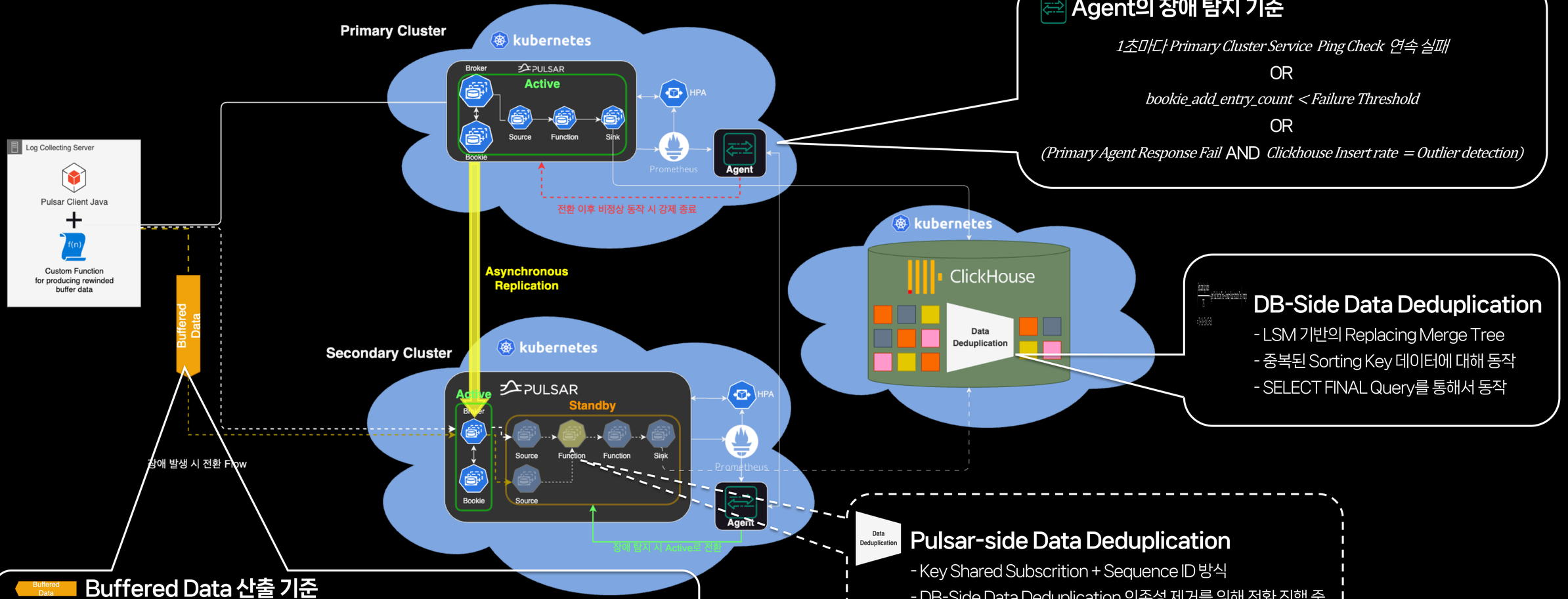
NAVER
DEVIEW
2023

6. Efficiently Lossless Realtime Processing in Log-data

NAVER DEVIEW 2023

6. Efficiently Lossless Realtime Processing in Log-data

Async Type + Cost Efficient + Lossless Features



Agent의 장애 탐지 기준

1초마다 Primary Cluster Service Ping Check 연속 실패

OR

bookie_add_entry_count < Failure Threshold

OR

(Primary Agent Response Fail AND *Clickhouse Insert rate = Outlier detection)*

DB-Side Data Deduplication

- LSM 기반의 Replacing Merge Tree
- 중복된 Sorting Key 데이터에 대해 동작
- SELECT FINAL Query를 통해서 동작

Pulsar-side Data Deduplication

- Key Shared Subscription + Sequence ID 방식
- DB-Side Data Deduplication 의존성 제거를 위해 전환 진행 중

Buffered Data 산출 기준

Buffered Data

$$\left(\frac{Replication\ Lag\ volume}{TPS} + Agent\ \ Fault\ Detection\ Time + Secondary\ Pulsar\ Activation\ Time\right) \times margin$$

$$= (7s + 10s + 5s) \times 1.5 = 33s$$

NAVER
DEVIEW
2023

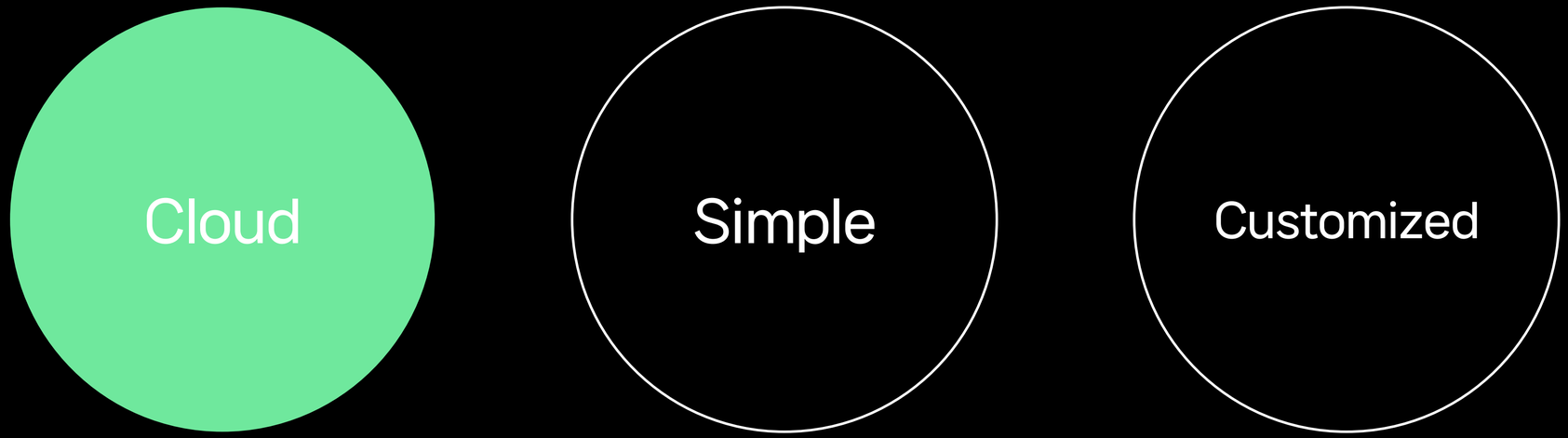
7. Wrap up

7.1 Keys of Geo Replication #1

NAVER
DEVIEW
2023

Cloud

- Cloud 의 유연한 환경에 기반을 둔 비용 효과적인 Resource 사용
- HPA에 기반한 시스템 인프라 구축

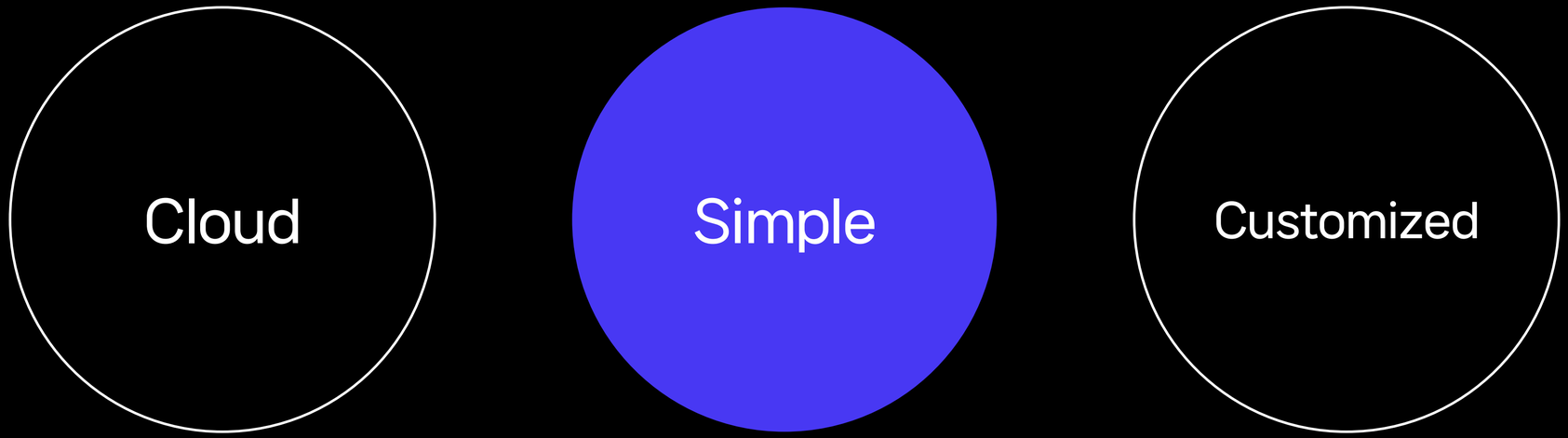


7.2 Keys of Geo Replication #2

NAVER
DEVIEW
2023

Simple

- 불필요한 데이터 흐름을 최소화한 Platform 구조
- Single Cluster 내에서의 데이터 처리

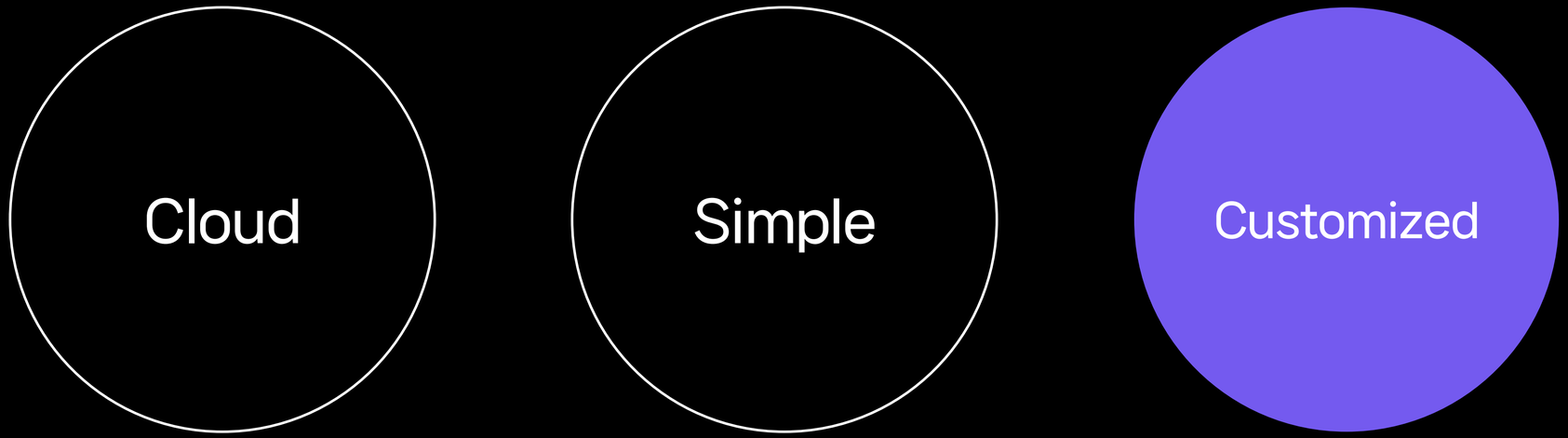


7.3 Keys of Geo Replication #3

NAVER
DEVIEW
2023

Customized

- Business Model에 맞는 Tolerant Control
- Client Level Data Rewind, DB side Deduplication 등 다양한 기술 활용



NAVER
DEVIEW
2023

Thank You / Q&A

References

NAVER
DEVIEW
2023

- Fine-grained consistency for geo-replicated systems - https://www.usenix.org/system/files/conference/atc18/atc18-li_cheng.pdf
- Apache Pulsar - https://pulsar.apache.org/
- Mirror Maker2 - https://cwiki.apache.org/confluence/display/KAFKA/KIP-382%3A+MirrorMaker+2.0
- Apache Kafka - https://kafka.apache.org/
- Apache Pulsar vs Apache Kafka - https://streamnative.io/blog/engineering/2022-04-07-pulsar-vs-kafka-benchmark/
- Apache Flink - https://flink.apache.org/
- Pulsar Replication Lag and Latency - https://www.splunk.com/en_us/blog/it/geo-replication-in-apache-pulsar-part-1-concepts-and-features.html
- Pulsar Package Manager - https://pulsar.apache.org/docs/next/admin-api-packages/
- If(kakao) - https://if.kakao.com/
- A Hitchhiker's Guide to Apache Kafka Geo-Replication - https://www.confluent.io/events/kafka-summit-london-2022/a-hitchhikers-guide-to-apache-kafka-geo-replication/
- Kafka Stretched Cluster more than 3 region - https://docs.confluent.io/platform/current/multi-dc-deployments/multi-region-architectures.html#stretched-cluster-3-region
- Why Zookeeper needs an odd number of nodes - https://bikas-katwal.medium.com/why-zookeeper-needs-an-odd-number-of-nodes-bb8d6020e9e9
- Kafka Rack Aware Feature - https://kafka.apache.org/documentation.html#basic_ops_racks
- Replication Backlog - https://www.splunk.com/en_us/blog/devops/geo-replication-in-apache-pulsar-part-2-patterns-and-practices.html