IBM Event Streams
Apache Kafka® for the Enterprise

A core part of Cloud Pak for Integration

Subhajit Maitra
maitras@us.ibm.com

IBM Event Streams is fully supported Apache Kafka® with value-add capabilities

IBM Event Streams

- Award-Winning User Experience
- Powerful Ops Tooling
- Schema Registry
- Geo-replication for DR
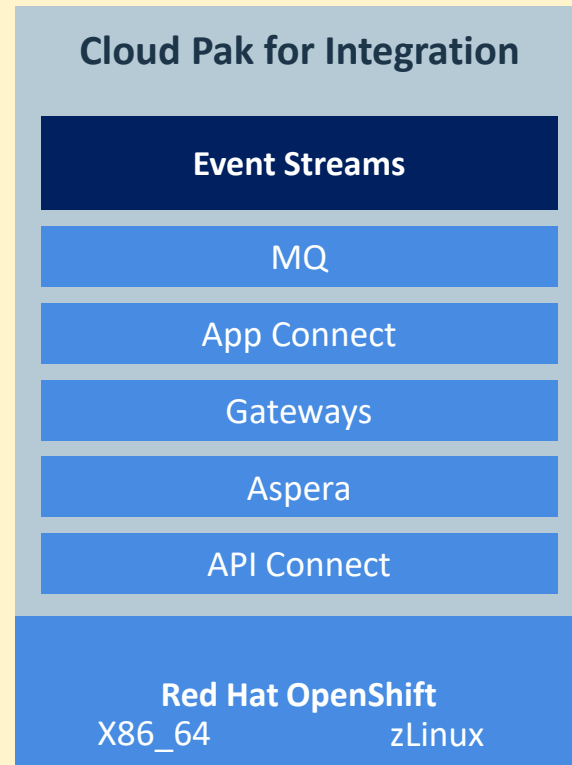- Connector Catalog
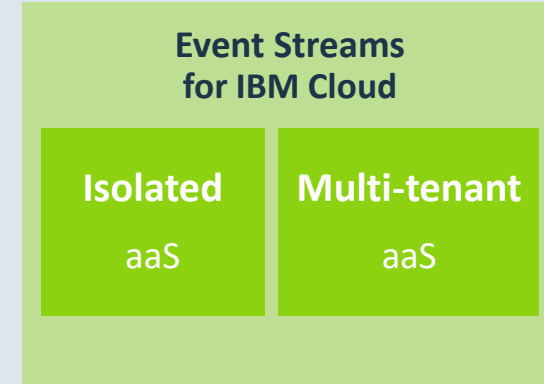- Unrivalled MQ connectivity
- 24 x 7 Support

INDIGO · SILVER WINNER ·

Packaging and Deployment Options to Suit Different Needs

Self Managed Software

Cloud Pak for Integration

Event Streams

MQ

App Connect

Gateways

Aspera

API Connect

Red Hat OpenShift
X86_64   zLinux

Fully Managed Service

Event Streams for IBM Cloud

Isolated
aaS

Multi-tenant
aaS

Hosted service on IBM public cloud

Getting started

Self Managed Software

Strimzi

Red Hat OpenShift

https://strimzi.io

Fully Managed Service

**Event Streams
for IBM Cloud**

Lite

Hosted service on IBM public cloud

Get Started at: ibm.com/cloud/event-streams/get-started

Apache Kafka orchestrated with Kubernetes

- **Highly available by design**
  - Brokers are spread across worker nodes using anti-affinity policies
  - Minimizes the risk of down-time in the event of a node outage

- **Scale the Kafka cluster up with one command**
  - Safely grows the stateful set, reconfigures the network interfaces and gives you more capacity

- **Roll out Kafka cluster configuration changes easily**
  - Make a single configuration change and Event Streams rolls it out across the brokers in the cluster
  - Broker availability is managed using health checks to ensure that availability is maintained
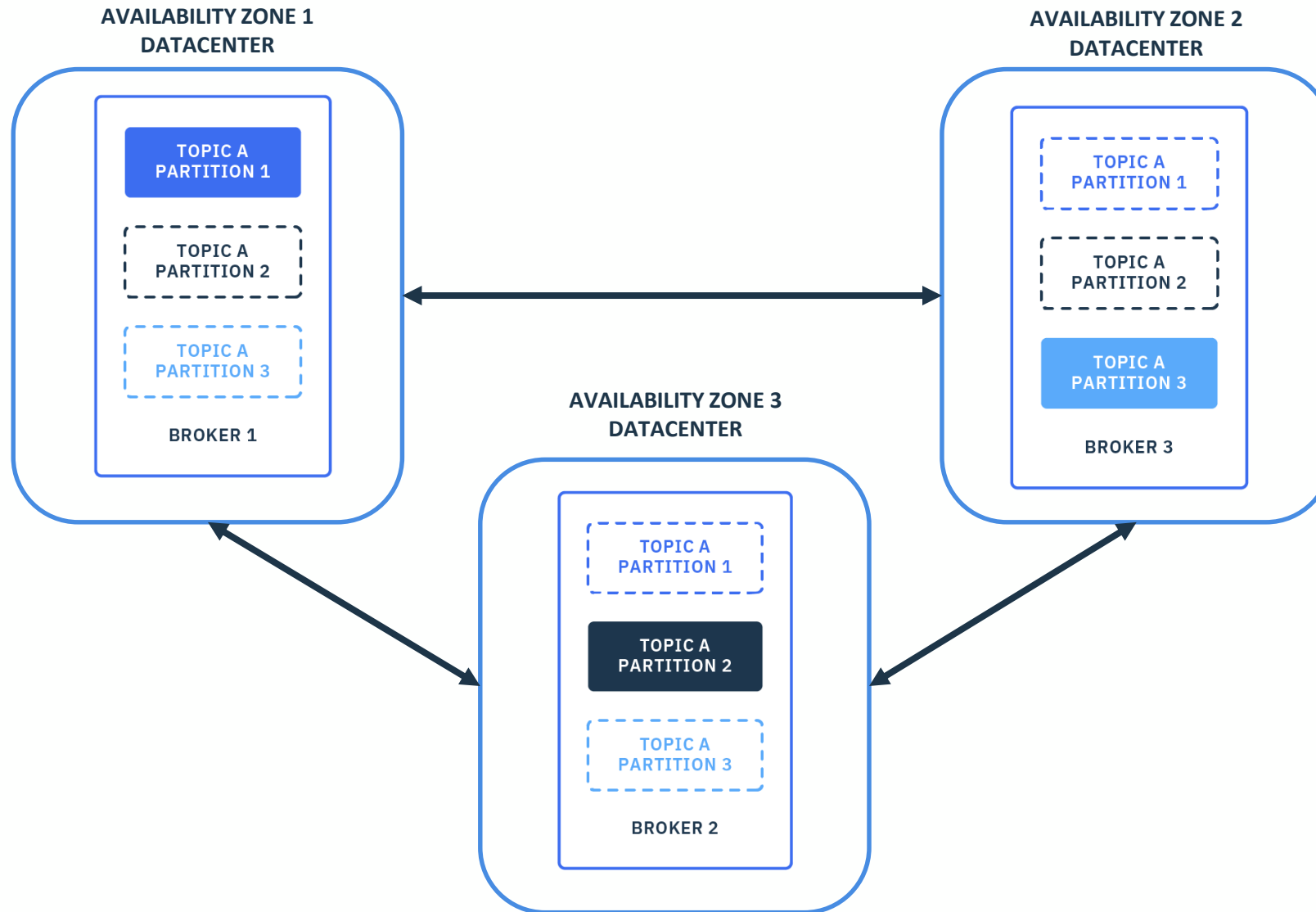
Safe, Planned Upgrade of Apache Kafka

Upgrade Kafka versions safely and without hassle

- First, upgrade the release to a newer version of IBM Event Streams
  - Rolling update of the Kafka brokers minimizes disruption

- As a separate step, upgrade the broker data and protocol version to complete the upgrade
  - Until this point, you can roll back

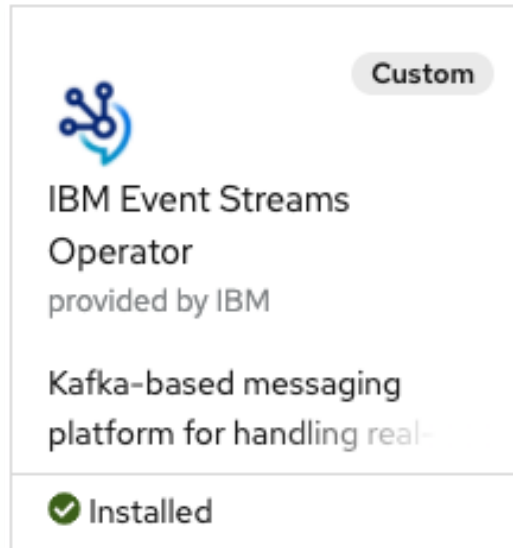# Enhanced resilience with clusters across multiple zones

AVAILABILITY ZONE 1 DATACENTER

TOPIC A PARTITION 1

TOPIC A PARTITION 2

TOPIC A PARTITION 3

BROKER 1

AVAILABILITY ZONE 2 DATACENTER

TOPIC A PARTITION 1

TOPIC A PARTITION 2

TOPIC A PARTITION 3

BROKER 3

AVAILABILITY ZONE 3 DATACENTER

TOPIC A PARTITION 1

TOPIC A PARTITION 2

TOPIC A PARTITION 3

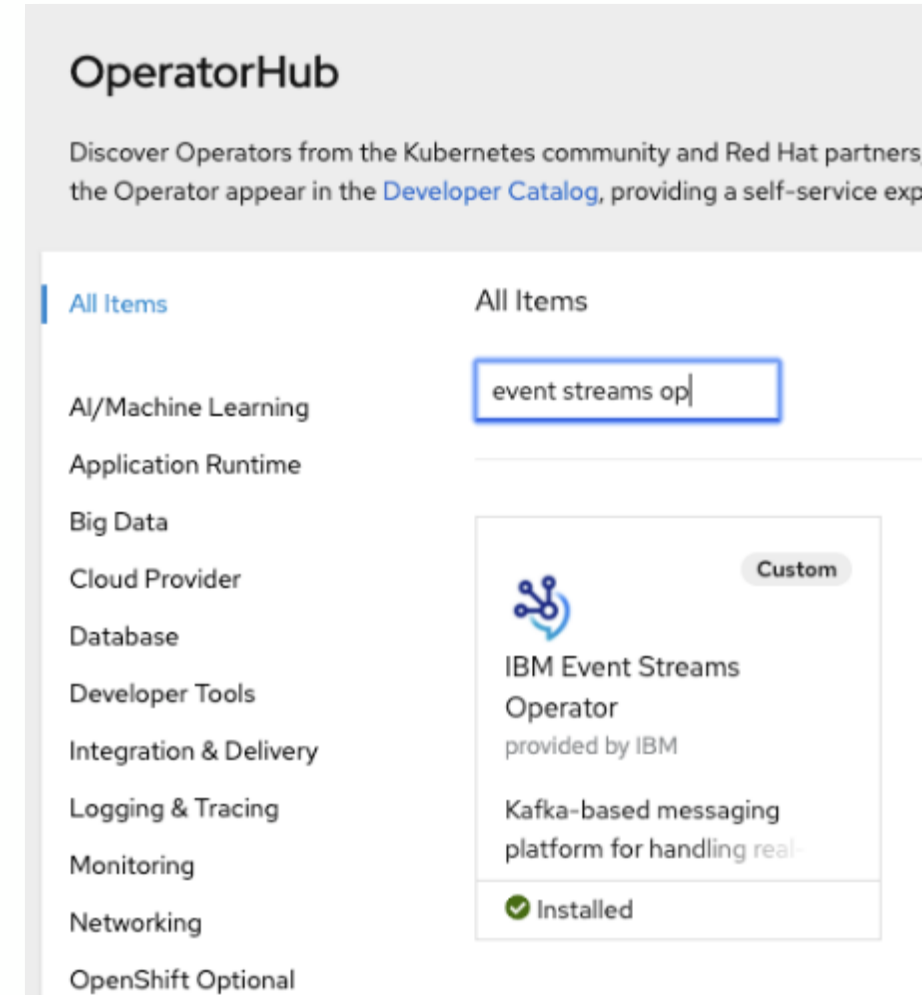BROKER 2

Multi-Availability Zone

- Must have at least 3 zones
- Kafka brokers and ZooKeeper servers span across zones
- Can tolerate failure of a zone with no service degradation
- High-speed network with low latency between zones required (< 20ms)

IBM **Event Streams**

7

Making Apache Kafka Intuitive and Easy
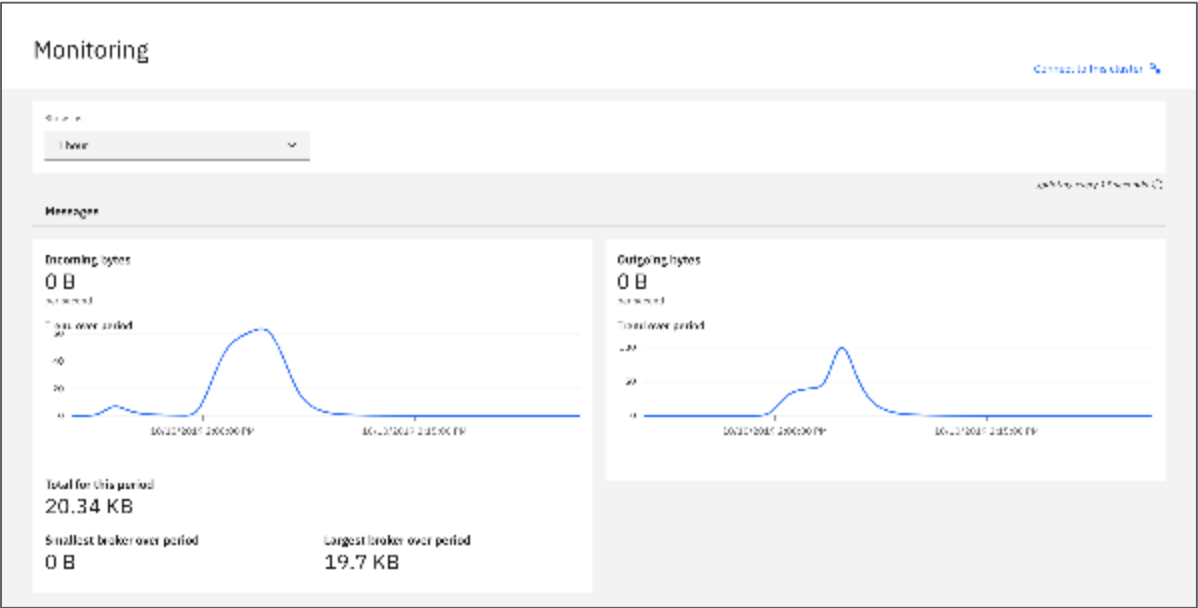## *Native deployment on OpenShift with Operators*

- Kafka has many distinct components to deploy, configure and coordinate for secure connectivity
- Container placement critical to ensure production-level availability
- Secured network traffic ingress
- Ensuring consistent and repeatable deployment

Simple to deploy and Manage

Making Apache Kafka Intuitive and Easy

Monitor status at a glance

Integrated feedback and support

Integrated with Key Monitoring Tools

IBM **Event Streams**

External monitoring tools
Datadog, Splunk, etc

Geo-Replication Makes Disaster Recovery Simple

Target is take-over of workload on the destination cluster by business applications within 15 minutes

Easy configuration using the Event Streams UI from the origin cluster sets up the replicator and security credentials

At-least-once reliability so messages are not lost

Use Existing Data in New Ways that Yield Competitive Advantage
Unmatched Connectivity to Core Systems

Kafka Connect framework

IBM MQ

Connector Catalog

Databases

Couchbase

redis

IBM Cloud Object Storage

REST API

ENHANCED FOR CICS

IBM Z

IBM DataPower

Applications

Direct integrations

IBM Watson IoT Platform

IBM Streams

Cloud Pak for Integration

IBM App Connect Enterprise

IBM Data Replication Change Data Capture

IBM **Event Streams**

12

© 2020 IBM Corporation

Connector catalog

Welcome to the IBM Event Streams

Kafka Connect is a framework for connecting Kafka to external systems. It uses source connectors to move data into Kafka, and sink connectors to move data out of Kafka.

The Event Streams connector catalog contains a list of tried and tested connectors from both the community and IBM.

Find out more about Kafka Connect →

✅ IBM supported

✅ IBM supported

Source connector | Kafka Connect
IBM MQ

Sink connector | Kafka Connect
IBM MQ

Sink connector | Kafka Connect
ArangoDB

Sink connector | Kafka Connect
IBM Cloud Object Storage

Source connector | Kafka Connect
Couchbase

Sink connector | Kafka Connect
Couchbase

Sink connector | Kafka Connect
HTTP

Sink connector | Kafka Connect
Memcached

Unlocking events from existing systems

Key usage pattern
Provide data to applications while protecting your backend

Systems of Record

IBM **Event Streams**

Protected

Scalable, High speed Multiple readers

Expand to multi-cloud

- IBM Event Streams allows you to emit **streams of data** from a mainframe system, providing a high speed decoupled **buffer** – allowing many readers to absorb the data
- Allows microservices to be developed, acting against the stream with **no impact** on **critical systems** of record

- Expand the pattern to multi-cloud – Event streams can be used to create a **local buffer** of data in each cloud environment
- **Minimize** on-prem to cloud **data transfer**
- Give the **fastest response** for cloud applications

Finance Customer Use Case | Transaction Alerts

Problem Statement:

Bank A. is a bank facing growing competition, who wants to modernize its customer experience to improve customer retention

⟲ The cost and time it takes to switch account providers is so low now that customers will simply move if they aren't getting the best possible service

Bank A has a new initiative to offer a service where customers can be alerted in *real-time* when "noteworthy" financial events occur

They do not want to disrupt or add load to their core systems which handle millions of transactions a day & are costly/complex to change

Goals:

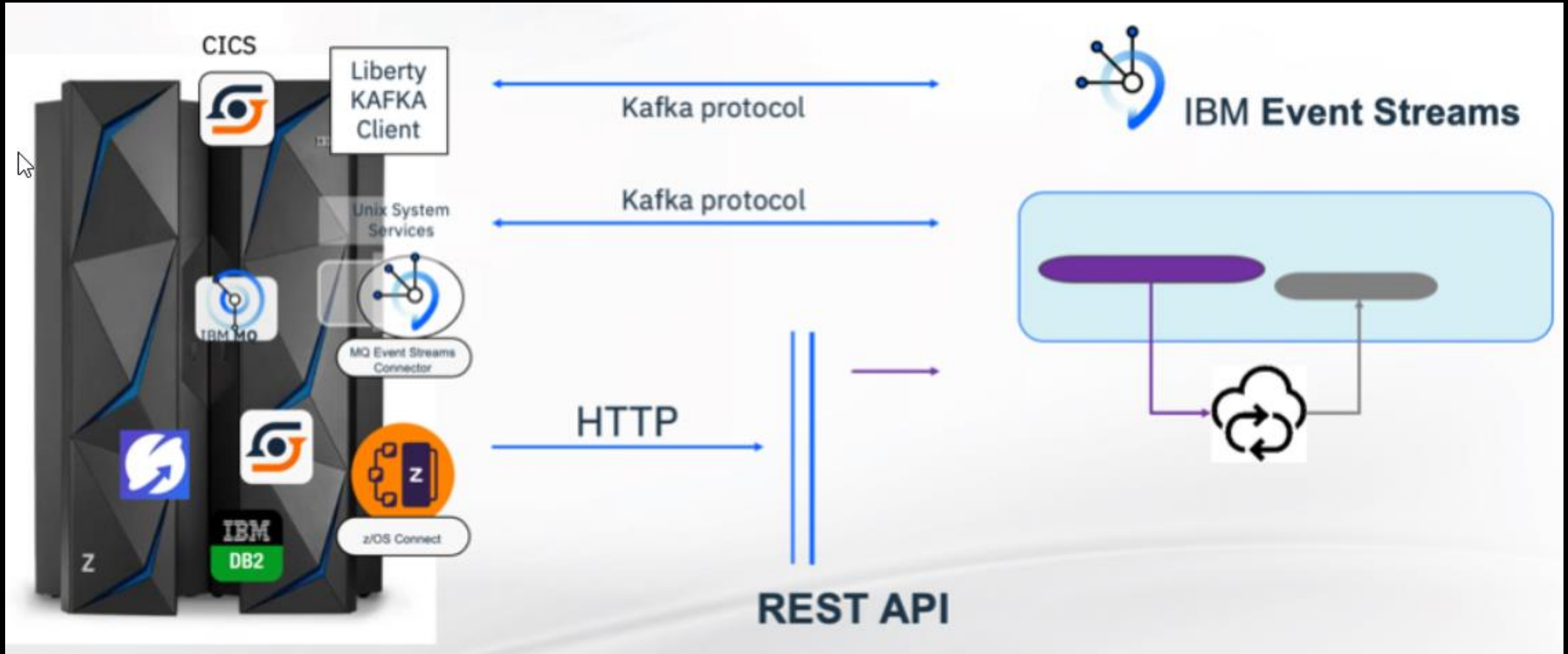- Issue alerts in real-time, so customers receive relevant information *immediately* and not when batch-processing is complete

- Make it possible to add new features E.g. more complex financial events and new types of alerts, quickly and easily over time
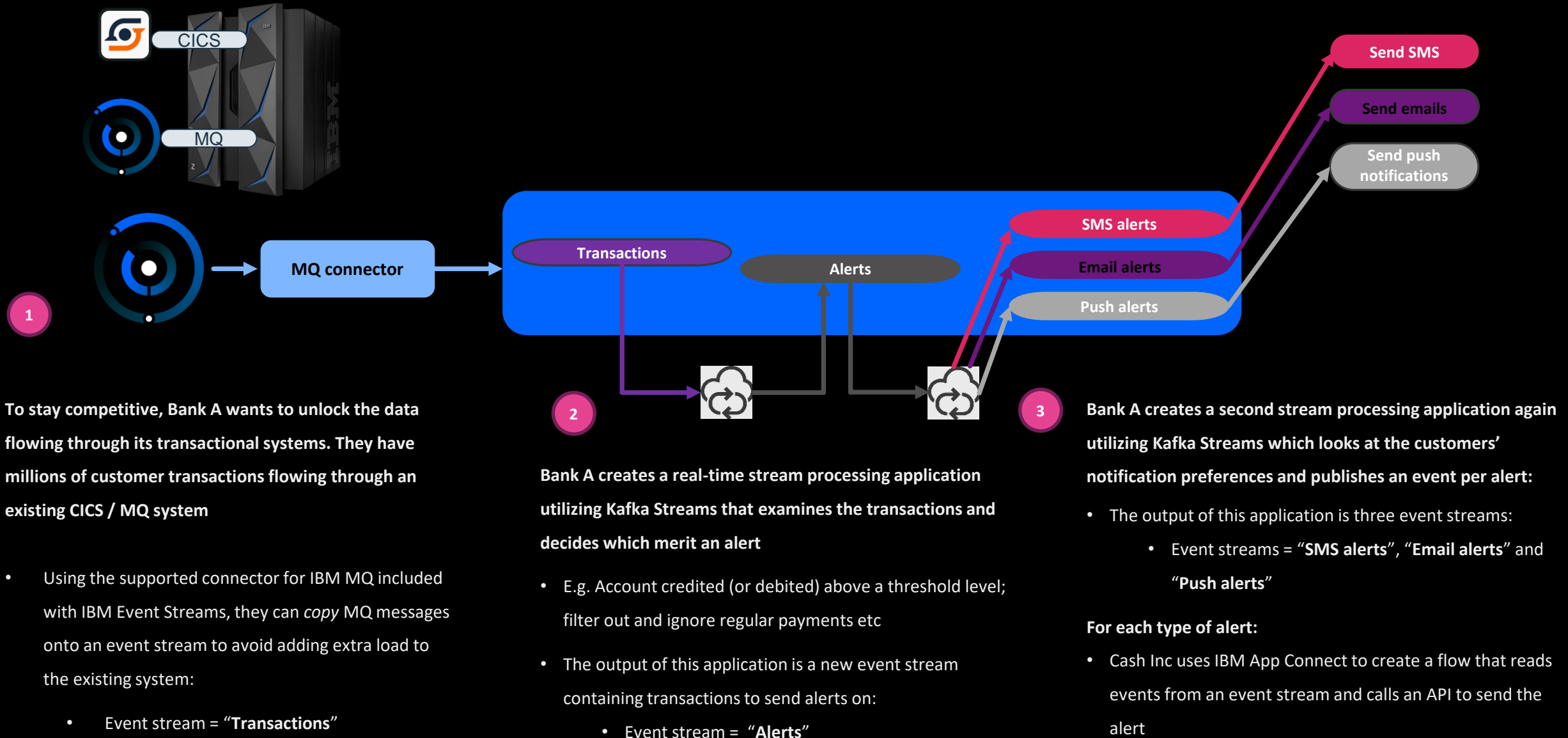

A withdrawal of over $1000 has been made from your account!

Unlock Data from Z to Event Streams

Moving data from Mainframe to Event Streams

Finance Customer Use Case | IBM Event Streams Solution

To stay competitive, Bank A wants to unlock the data flowing through its transactional systems. They have millions of customer transactions flowing through an existing CICS / MQ system

- Using the supported connector for IBM MQ included with IBM Event Streams, they can *copy* MQ messages onto an event stream to avoid adding extra load to the existing system:

  - Event stream = "**Transactions**"

Bank A creates a real-time stream processing application utilizing Kafka Streams that examines the transactions and decides which merit an alert

- E.g. Account credited (or debited) above a threshold level; filter out and ignore regular payments etc

- The output of this application is a new event stream containing transactions to send alerts on:

  - Event stream = "**Alerts**"

Bank A creates a second stream processing application again utilizing Kafka Streams which looks at the customers' notification preferences and publishes an event per alert:

- The output of this application is three event streams:

  - Event streams = "**SMS alerts**", "**Email alerts**" and "**Push alerts**"

For each type of alert:

- Cash Inc uses IBM App Connect to create a flow that reads events from an event stream and calls an API to send the alert

It's Easy to Connect IBM MQ to Apache Kafka

IBM has created a pair of connectors, available as source code or as part of IBM Event Streams
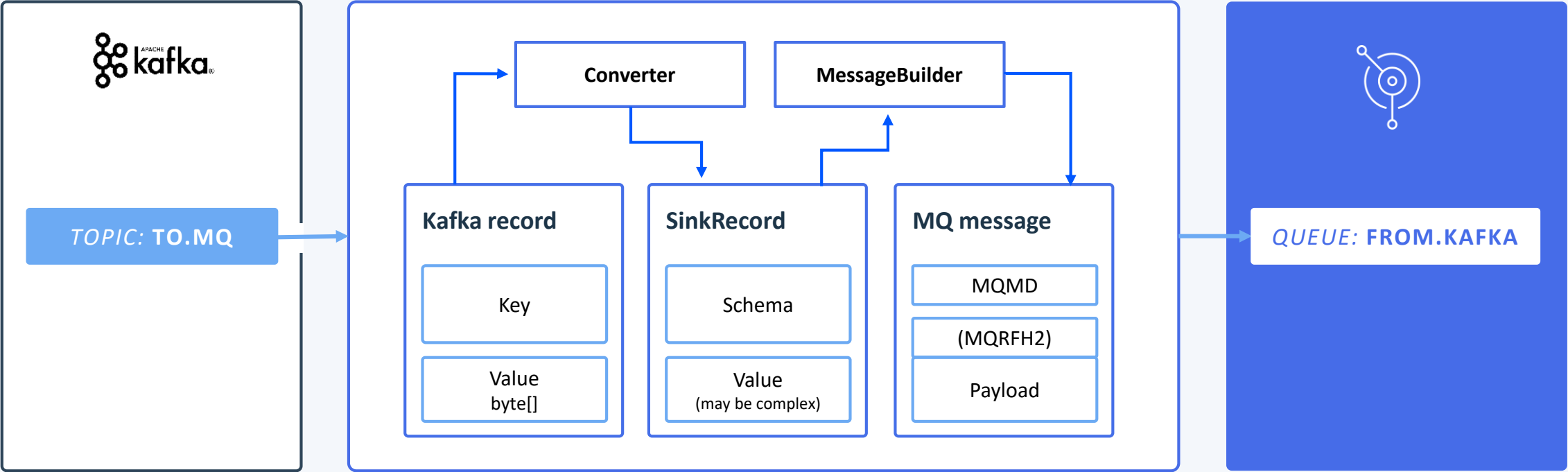
Source Connector

From MQ queue to Kafka topic

https://github.com/ibm-messaging/kafka-connect-mq-source

Sink Connector

From Kafka topic to MQ queue

https://github.com/ibm-messaging/kafka-connect-mq-sink

- Copies messages from MQ queues to Event Streams topics and vice versa

- Supports all current MQ versions

- Extend the connector to support any business-specific message format

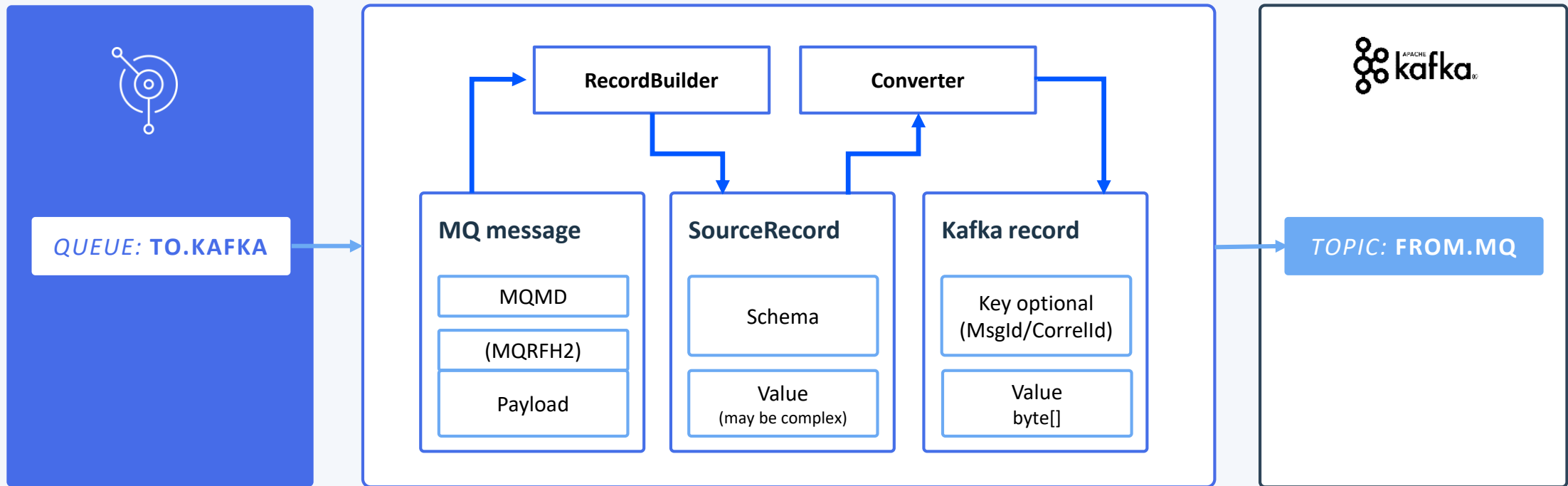- Fully supported by IBM for customers with support entitlement for IBM Event Streams
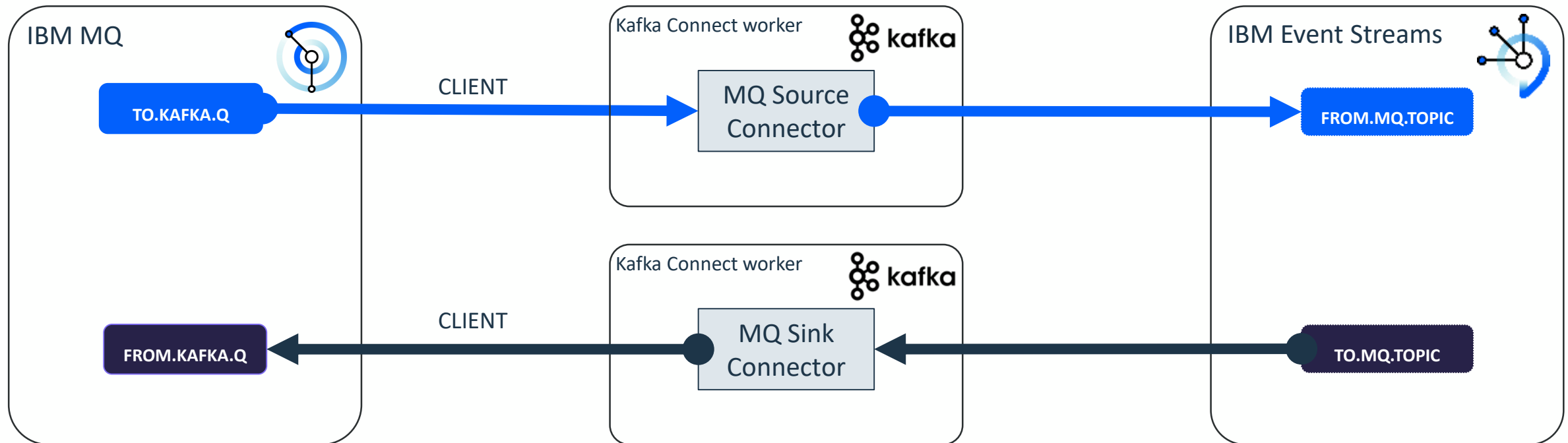
IBM **Event Streams**

MQ sink connector

MQ source connector

Running the Connectors for IBM MQ

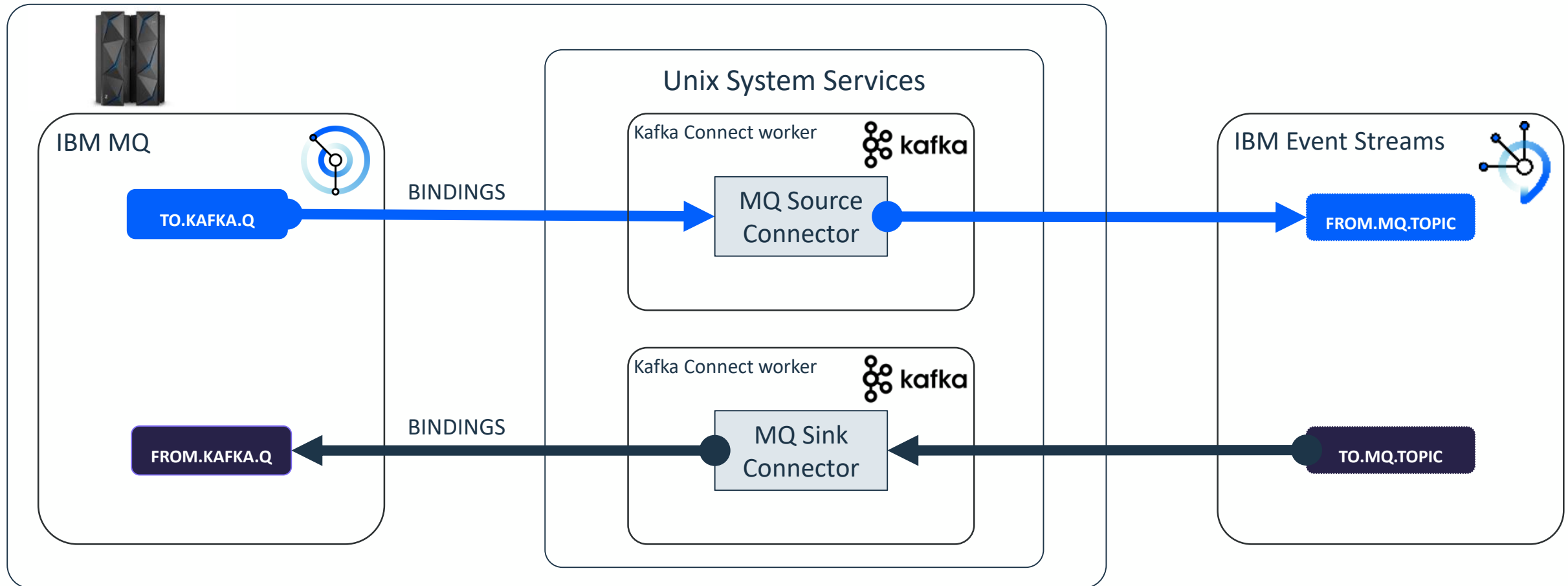The connectors are deployed into a component of Apache Kafka called a Kafka Connect worker

This runs between IBM MQ and IBM Event Streams (or open-source Apache Kafka)

Running the Connectors for IBM MQ on z/OS

The Kafka Connect workers can be deployed onto z/OS Unix System Services

Then, the connection to MQ can be a bindings connection

Advantages of running the MQ connector on z/OS

1) **Lower workload costs**
   - Local bindings are 3x less CPU intensive than client bindings
   - Use of bindings mode reduces latency as removes one network hop, important for real-time analytics use cases
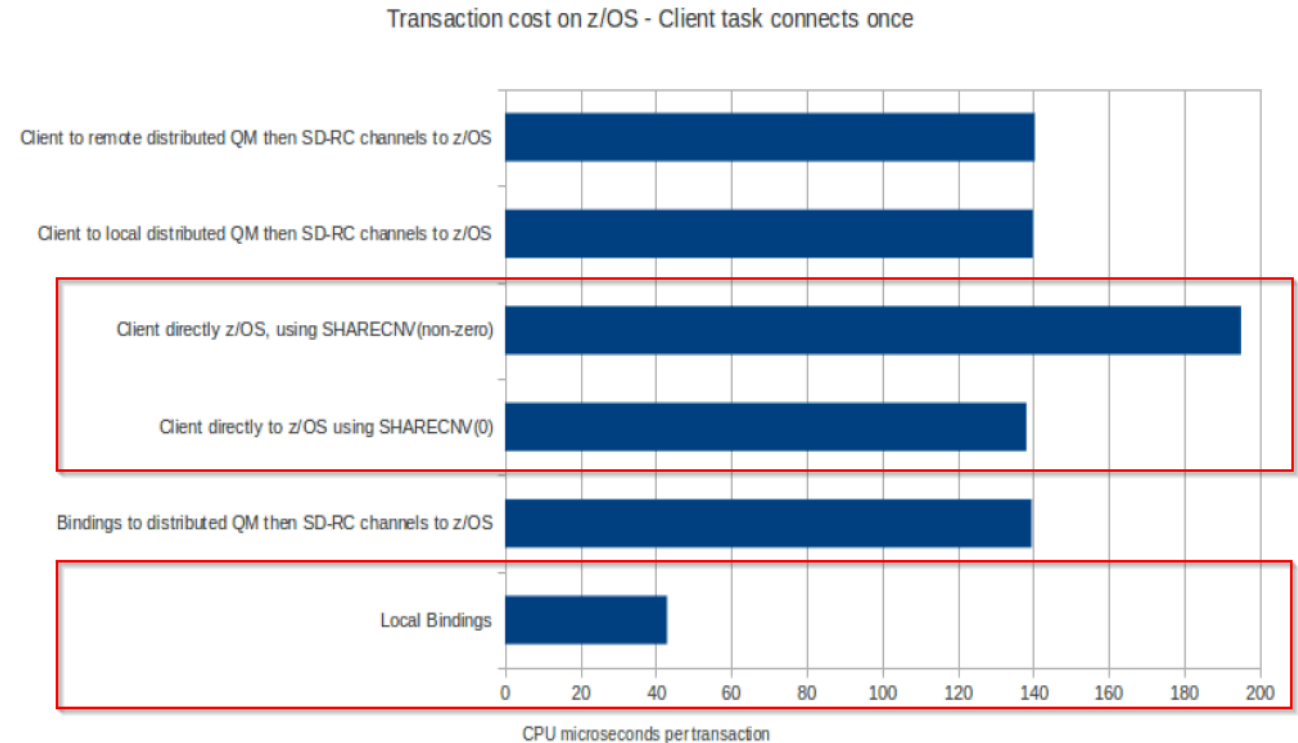
2) **Better performance in bindings mode**

3) **Kafka Connect on z/OS is offloadable**
   - This is pure Java-based workload, and so is eligible for offload to zIIPS
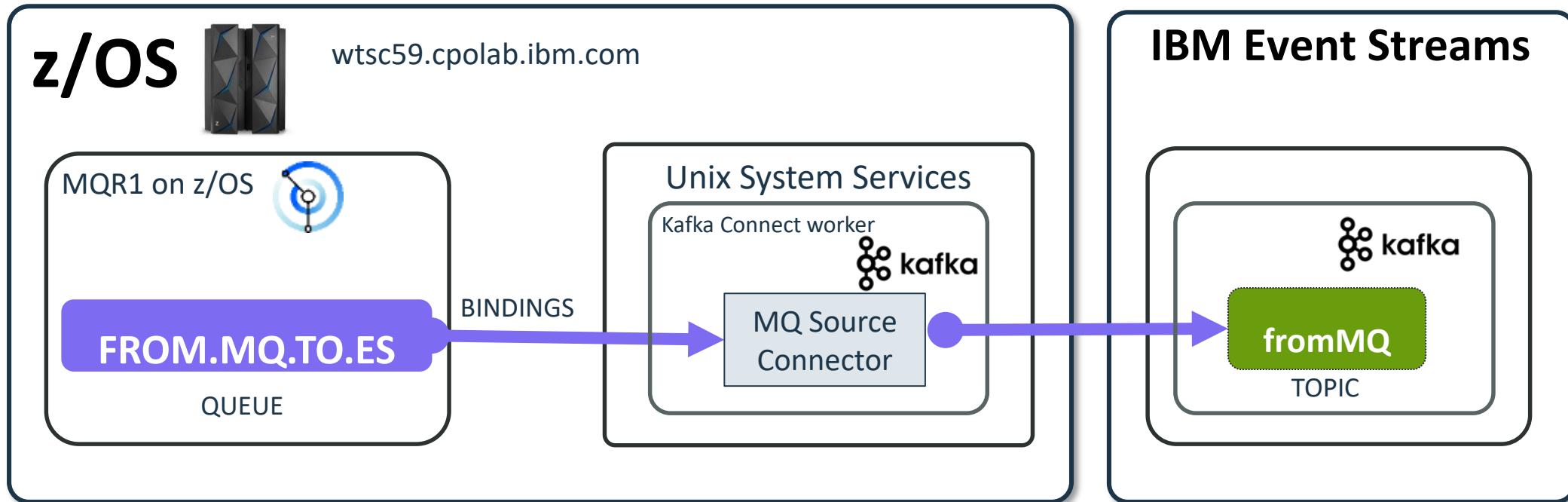
4) **Simplified configuration**
   - One less set of channel and TLS configuration required

CHART: Transaction cost with model: connect, [put, get]*100,000, disconnect

Transaction cost on z/OS - Client task connects once

Client to remote distributed QM then SD-RC channels to z/OS	
Client to local distributed QM then SD-RC channels to z/OS	
Client directly z/OS, using SHARECNV(non-zero)	
Client directly to z/OS using SHARECNV(0)	
Bindings to distributed QM then SD-RC channels to z/OS	
Local Bindings	

CPU microseconds per transaction

Demo

Improved integration for IBM MQ for z/OS - Demo

Find Out More

Explore IBM Event Streams at

https://ibm.github.io/event-streams/

Contact us: eventstreams@uk.ibm.com

IBM **Event Streams**

Thank You