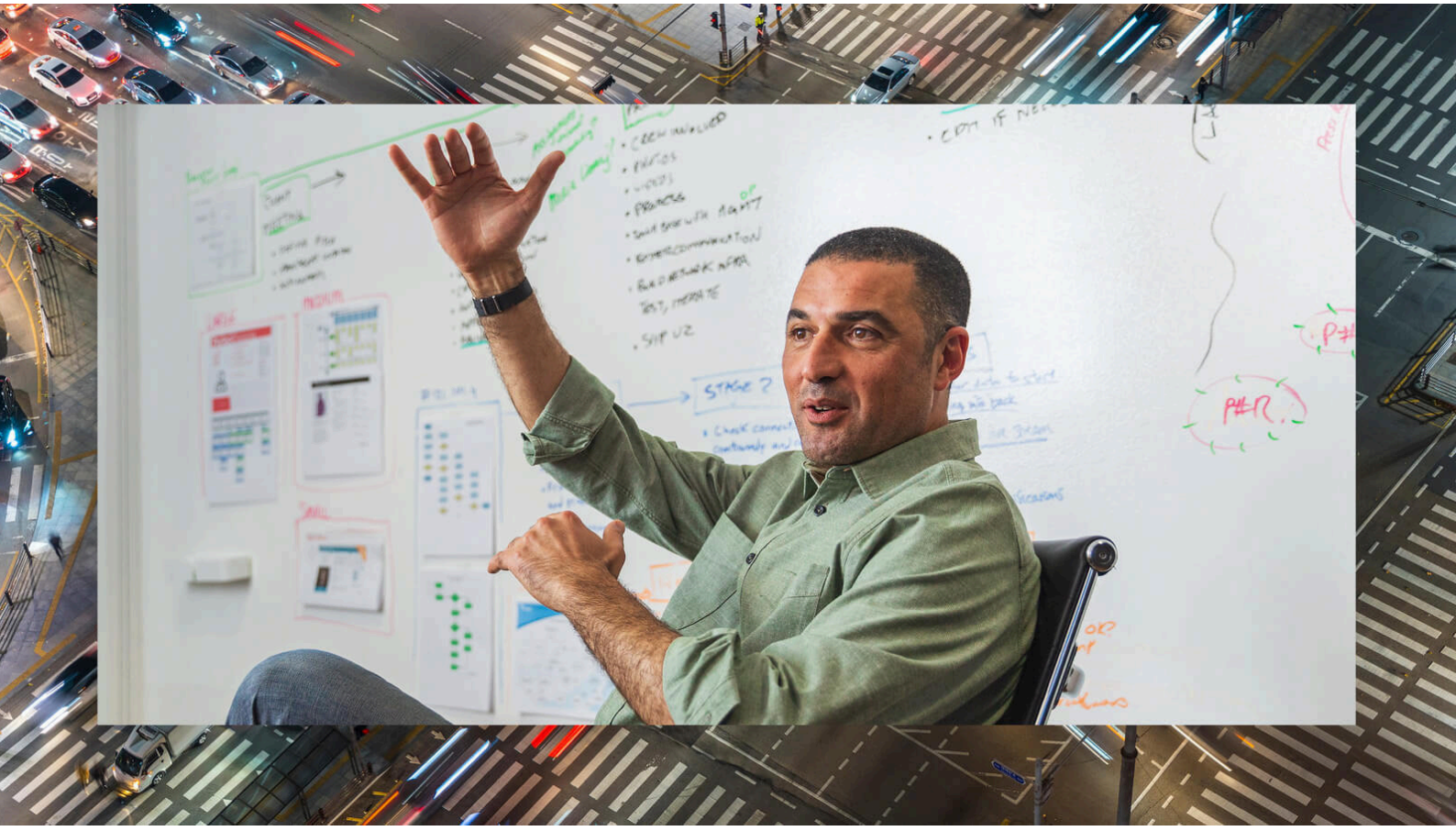
HPE GreenLake

Technical white paper

HPE Ezmeral Data Fabric Software

Event streaming and Apache Kafka support



Contents

Introduction

Event streaming and Apache Kafka functions are built into the HPE Ezmeral Data Fabric Software. It requires no additional process to manage and leverages the same architecture as the rest of the platform while requiring minimal additional configuration. Data managed by HPE Ezmeral Data Fabric Software can be accessed through the Apache Kafka API—although the data fabric streams are implemented quite differently—giving them capabilities beyond Kafka.

Streams overview

Data from many sources (producers) are written to many topics—potentially thousands or more—and are unified into a single data fabric stream. Topics are partitioned for throughput and scalability. Rather than broadcasting messages to consumers or applications, such as Apache Spark, the user can subscribe to topics and read messages from streams without deleting the message. Consumers of messages can be decoupled providing the flexibility for producers and consumers to be developed and deployed independently. Topics can expire after a certain period or retained indefinitely.

Producers

Producers are data-generating applications, such as sensors in automobiles or activity loggers in servers. Producers create messages with the collected data and publish the messages to HPE Ezmeral Data Fabric Software topic partitions. Producers create messages about the collected data, then send it to the producer client library specifying the topic destination of the message, and provide the optional partition ID. The producer client buffers incoming messages and sends them in batches to the HPE Ezmeral Data Fabric Software server. The producer client library sends the records to the server when the following conditions are met:

- The producer client library has batched enough messages to make an efficient remote procedure call (RPC) to the server
- A message has been queued for a specified amount of time
- The producer client library has batched messages beyond the value of the buffer memory configuration
- The application explicitly flushes messages
- If the data fabric node is not available to receive messages, they are buffered in the client library until a node within the cluster comes online and the producer can continue sending messages

The producer client library batches message into multiple publish requests, which are sent to the Streams server, see Figure 1.

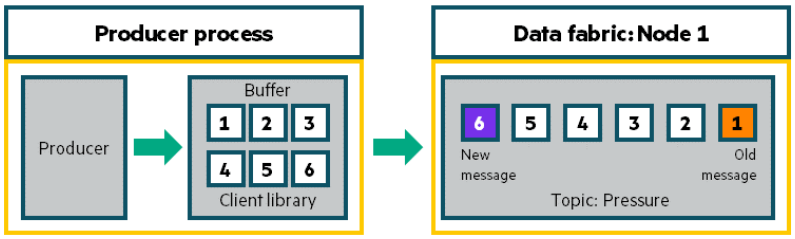


Figure 1. Producer process

Messages can be sent

- "at-least-one": The message delivery guarantees that the message is published at least once on the Streams server. Messages are never lost and can be re-delivered.
- "exactly-once": The message delivery is sent without duplication. Each message is delivered only once. "exactly-once" delivery uniquely identifies a group of messages that are atomically persisted. The message delivery is set with the producer idempotence option.

Failure scenarios are addressed with idempotence:

- The stream processor may take input from multiple source topics and the ordering across these source topics is not deterministic across multiple runs. So, if the stream processor that takes input from multiple source topics is re-run, it might produce different results.
- The stream processor may produce output to multiple destination topics. If the producer cannot do an atomic write across multiple topics, then the producer output can be incorrect if writes to some (but not all) partitions fail.
- The stream processor may aggregate or join data across multiple inputs. If one of its instances fails, then it will be necessary to roll back the state materialized by that stream processor instance. On restarting the instance, it will be necessary to resume processing and recreate its state.
- The stream processor may look up enriching information in an external database or by calling out to a service that is updated out of band. By depending on an external service, the stream processor can be fundamentally non-deterministic. For example, if the external service changes its internal state between two runs of the stream processor, it can lead to incorrect results downstream.

Topics are logical collections of messages that are managed by streams. Topics are partitioned for throughput and scalability. Partitions make topics scalable by spreading the load for a topic across multiple nodes in the cluster. To organize data, all stream producers are load balanced between partitions. Consumers can be grouped to read in parallel from multiple partitions within a topic for faster performance.

In Figure 2, the producer is an oil rig, which creates a stream of sensor data composed of three topics—pressure, temperature, and warnings. The default partition size is 1 when a topic is created but, in this case, Pressure and Temperature are configured with three partitions as there will be a lot of data. For Warnings, two partitions are created as there will be fewer ones when compared to Pressure and Temperature readings.

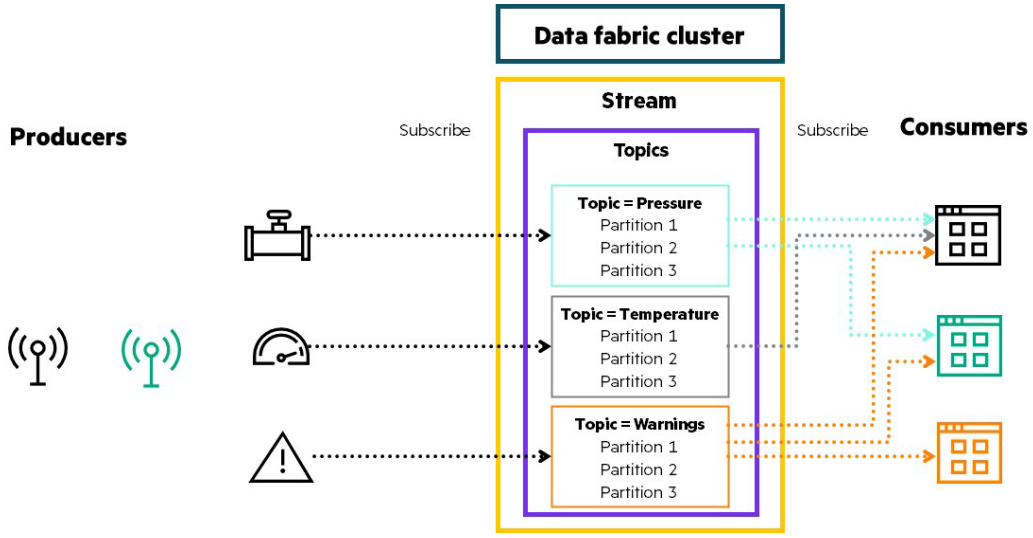


Figure 2. Producer and consumer publish, subscribe flow

Consumers

Consumers use the HPE Ezmeral Data Fabric Software APIs to request messages from the topics in which they are interested. If the server fails, the consumer client automatically retries the message request. A consumer client library sends unread messages from which consumers extract data and can run as separate processes on a single machine or as processes on different machines.

Consumers subscribe to topics. When a consumer subscribes to a topic or partition, it means that they want to receive messages from that topic or partition. For example, an analytics application might subscribe to the topics "rfids_productA", "rfids_productB", and more to track the movement of products from factories to distribution centers. A reporting tool might subscribe to the topics: "meters_NW", "meters_SW", and more to get a report on electricity usage in different geographic regions that a power company services.

Consumers can subscribe to:

Topics

When a consumer subscribes to a topic, it reads messages from all the partitions that are in the topic. The exception is when a consumer is part of a consumer group. Consumer groups are explained in the following section.

Consumers can subscribe to topics in two ways:

- By name: Consumers specify the names of the topics to which they subscribe
- By regular expression: Consumers specify a regular expression and subscribe to all topics with names that match the regular expression

The ability to use regular expressions is helpful when the "-autocreate" parameter for a stream is set to true and producers are allowed to create topics automatically at runtime.

Partitions

Consumers can subscribe to individual partitions within topics. This is helpful when you want a consumer to read the messages published to a specific partition. For example, a producer might publish messages for high-priority data to a specific partition for processing by a dedicated consumer. When a consumer subscribes to individual partitions within a topic, the consumer does not receive messages from any of the other partitions in the topic. To unsubscribe from topics to which you are subscribed with regular expressions, you must use the same regular expressions.

Consuming messages

Consumers request the HPE Ezmeral Data Fabric Software consumer client library to check whether any new messages have been published in the topics or partitions to which they are subscribed, or the partitions that they are assigned. Consumers can do this at any time. If a message with a minimum number of bytes is waiting across a consumer's subscription, HPE Ezmeral Data Fabric Software sends those messages to the consumer, up to a maximum number of bytes. These minimum and maximum values can be configured in the parameters for each consumer. The HPE Ezmeral Data Fabric Software consumer client library sends the messages that have been published by producers but not yet flushed to disk. If a consumer can consume data at the rate the producer publishes messages, the consumer library continuously sends messages to consumers from its memory, increasing the speed of throughput from producer to consumer, see Figure 3.

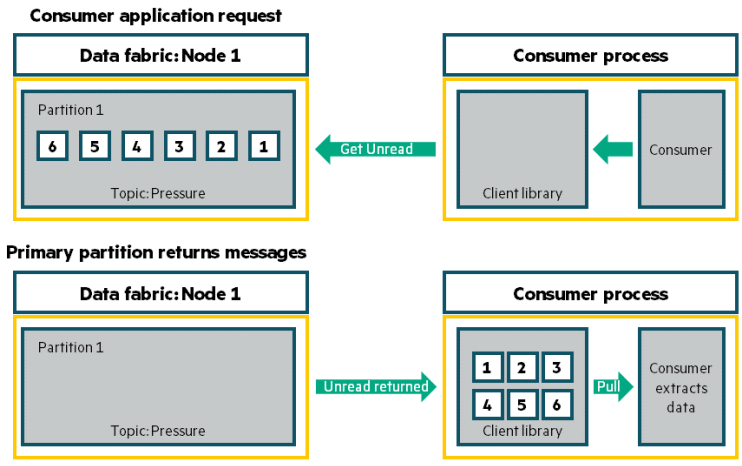


Figure 3. Consumer read messages process

Consumer failure and recovery

When a consumer who is not associated with a consumer-group ID recovers from failure and is back online, it can either start reading its partitions from the earliest offsets or from the latest offset. The consumer may read from the earliest offset in a partition, which is the offset of the message that has been in the partition longest without being deleted because of the expiration of the time-to-live interval for the stream. In this scenario, many messages might be re-read before reading messages that were published after it failed. The consumer may read from the latest offset in a partition, which is the offset of the most current message when the consumer requests new messages from HPE Ezmeral Data Fabric Software. In this scenario, the consumer is up to date but skips the messages between its time of failure and the current time.

Consumer groups

Consumers can be grouped together by setting the same value for the group.id configuration parameter when each consumer is created. For example, if three consumers are created and each of them is assigned to the same group ID clickstream_consumers, together these consumers form the group clickstream_consumers. HPE Ezmeral Data Fabric Software does not generate IDs for consumer groups. IDs can be created by specifying the group.id configuration parameter when creating a consumer. The partitions in each topic to which all the consumers are subscribed are assigned dynamically to the consumers in a round-robin fashion. For example, in Figure 4, there are three consumers in a group and each consumer is subscribed to the same topic. There are five partitions in the topic. HPE Ezmeral Data Fabric Software assigns each partition to a consumer, with two consumers being assigned two partitions. If one of the consumers goes offline, the partitions are reassigned dynamically among the remaining consumers in the group. If the offline consumer comes back online or a different consumer is added to the group, again the partitions are redistributed among the consumers in the group. This parallelism and dynamic reassignment are possible only when none of the consumers in a consumer group subscribe to individual partitions. (See Figure 4.)

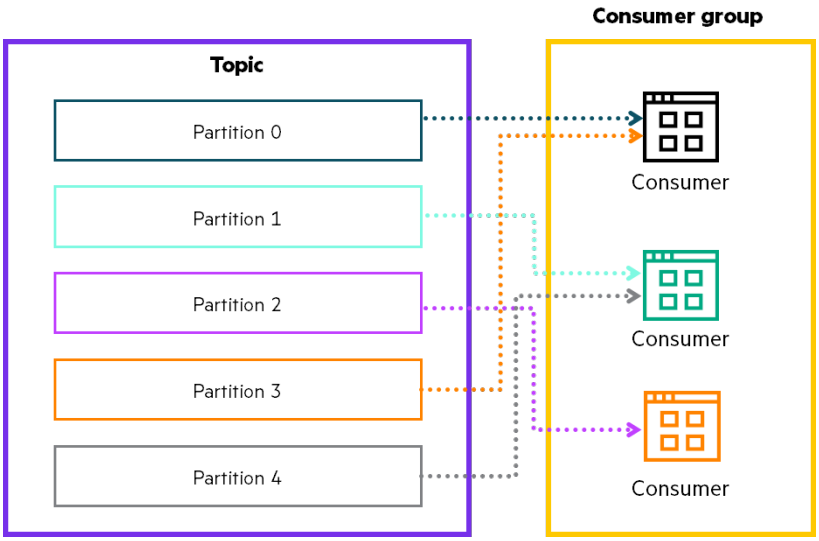


Figure 4. Consumer group partition allocation

Message recovery

Fault tolerance is built into the architecture of HPE Ezmeral Data Fabric Software. Each partition and all its messages are replicated for fault tolerance. The server owning the primary partition for the topic replicates the message to replica containers. Producers and consumers write and read from the primary partition, shown in the example in Figure 5 in orange. If a node goes down and the primary partition is longer available, a replica partition becomes the new primary for the failed partition; producers and consumers will automatically be rerouted to the new primary. Streams are completely reliable by synchronously replicating all rights to at least three nodes while retaining high performance of up to a billion messages a second at millisecond delivery times. (See Figure 5.)

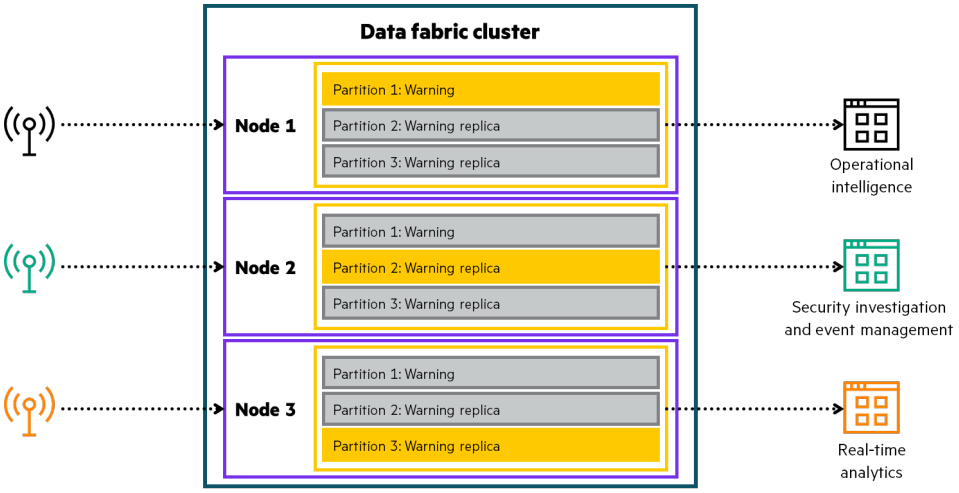


Figure 5. Message recovery after node failure and loss of primary partition

Modes of stream replication

Streams can be replicated in one of two replication modes. The mode is specified per source-replica pair.

Asynchronous replication

In this replication mode, HPE Ezmeral Data Fabric Software confirms to producers that messages are published after the messages are placed in partitions. Messages are replicated in the background. Therefore, the latency of message publishing is not affected by the time required for the network round trip between the source cluster and the destination cluster.

This type of replication is well-suited for clusters that are geographically separated in wide-area networks. Asynchronous replication is the default replication mode.

Synchronous replication

In this replication mode, HPE Ezmeral Data Fabric Software confirms that messages have been placed in partitions only after the messages are sent to a gateway in the destination cluster. Due to the confirmations that HPE Ezmeral Data Fabric Software receives on source clusters, synchronous replication is especially well-suited for creating a backup of the data for disaster recovery.

When the latency of a replication stream is high, HPE Ezmeral Data Fabric Software switches to asynchronous replication temporarily so that producers are not blocked indefinitely. After the latency is sufficiently reduced, it switches back to synchronous replication. The same switching from synchronous to asynchronous replication occurs if all gateways fail. HPE Ezmeral Data Fabric Software does not resume synchronous replication until a new gateway is established or at least one of the failed gateways is restarted.

Replication

Each partition and all its messages are replicated for fault tolerance. The node owning the primary partition for the topic replicates the message to replica containers. Producers and consumers send and read from the primary partition. Replicas are used for fault tolerance so that if a primary portion goes down a new primary partition is selected. Producers and consumers will be pointed to the new primary. HPE Ezmeral Data Fabric Software synchronously replicates all rights to at least three nodes while maintaining high performance of up to a billion messages per second at millisecond-level delivery times.

To prevent loss of access if a whole cluster goes down, streams can be asynchronously or synchronously replicated between data fabric clusters. A backup copy of a stream can be created so that producers and users have a stream to failover if the stream goes offline.

HPE Ezmeral Data Fabric Software replication across clusters

Replication is an effective way to connect geo-distributed data fabric clusters, however, event streams can also be moved between data fabric clusters by mirroring at the volume level.

Basic primary-secondary replication

Streams can be replicated to other data fabric clusters worldwide, or to other streams within a data fabric cluster. An example is shown in Figure 6 where producers in Texas are producing a stream metrics which is being replicated to the Texas_HA cluster, as is also happening for the cluster in New Mexico. This type of replication is called basic primary-secondary replication because replication is in one direction only. The metrics stream in the Texas_HA cluster is a replica. The original metrics stream is the upstream source for the replica. (See Figure 6.)

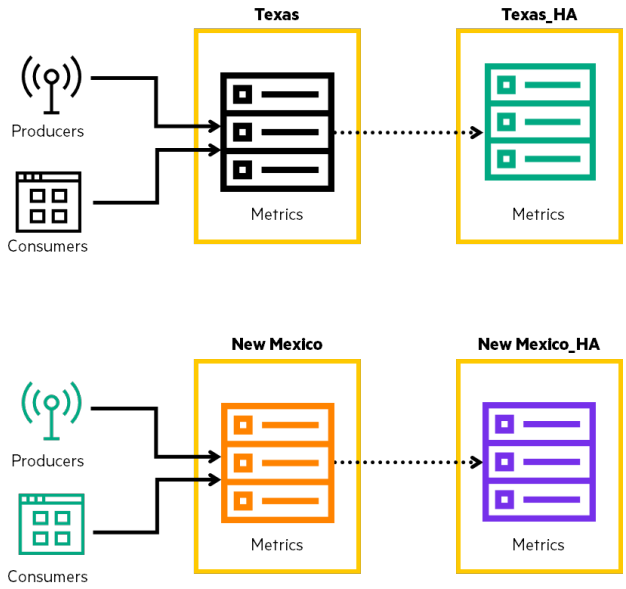


Figure 6. Primary-secondary replication

Many-to-one replication

In Figure 7, the metric stream from Texas and the metric stream from New Mexico are replicated to the metric stream in the California cluster.

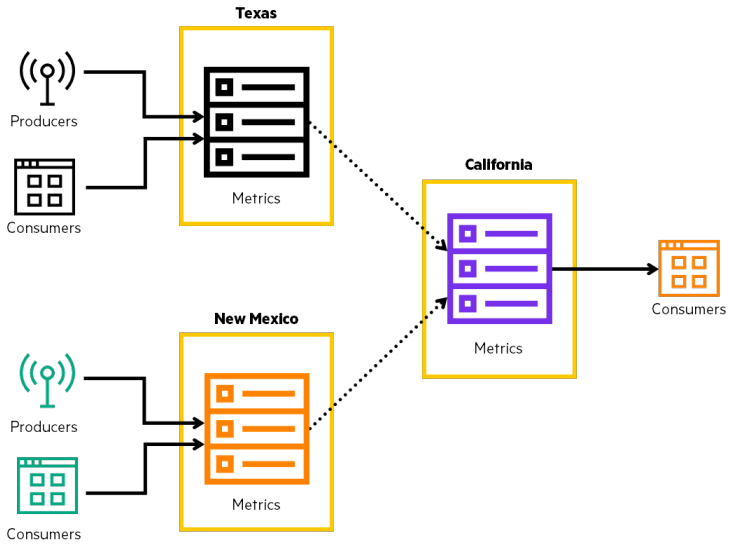


Figure 7. Many-to-one replication

This replica has two upstream sources and this type of replication, called many-to-one replication, requires the topics in each stream to have unique names, so that message offsets do not conflict. For example, suppose both oil wells have a pressure valve named valve_2 and the topic in each oil-well stream for collecting metrics from this valve is named valve_2, at some point, the Texas oil well and the New Mexico oil well both replicate messages that use the same offsets. Since offsets are replicated together with messages, messages can be overwritten in this case.

To avoid this type of problem, the sensors for valve_2 in the Texas oil well can publish to a topic named valve_2_texas, the sensors for valve_2 in the New Mexico oil well can publish to a topic named valve_2_new-mexico, and so on. The consolidated stream in California would contain the topics valve_2_texas and valve_2_new-mexico.

Multi-primary replication

Another kind of replication that can be useful is multi-primary replication. It can be used when there are two streams, both to send updates to and receive updates from the other stream. Each stream is a replica and an upstream source. HPE Ezmeral Data Fabric Software keeps both streams synchronized with each other.

As with many-to-one replication, the names of the topics in each stream must be unique across both streams, so that offsets for messages do not conflict. (See Figure 8.)

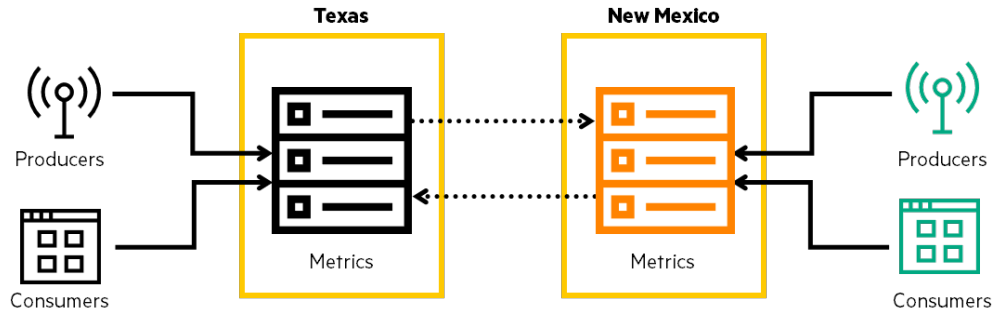


Figure 8. Multi-primary replication

Streams security

Data can be sent encrypted or unencrypted when replicating streams by using the "-networkencryption" parameter when a replica is created or edited. Security permissions can protect topics in a stream from unauthorized access. In addition, user impersonation is also supported.

Access control expressions

Access control expressions (ACEs) are used to protect topics in a stream from unauthorized access. ACEs are set when you create or edit a stream.

The following role definitions are supported:

- **adminperm**
  Determines, which users can modify ACEs for a stream, set up replication of a stream, and modify other attributes of a stream. By default, the stream owner and the data fabric user can modify this setting.
- **copyperm**
  Governs the users who can run the mapr copystream and mapr diffstreams utilities on the stream. Users with this permission can publish messages to topics in a stream, read messages in topics from a stream, and create or remove topics in a stream. This permission is a combination of the consumeperm, produceperm, and topicperm permissions.
- **consumeperm**
  Determines the users who can read messages in topics from a stream.
- **produceperm**
  Controls the users who can publish messages to topics in a stream.
- **topicperm**
  Governs the users who can create topics in a stream or remove them.

The example in Figure 9 shows the adminperm, consumeperm, produceperm, and topicperm permissions on a stream named temp_sensors, which includes the topics temp_sensors_internal and temp_sensors_external.

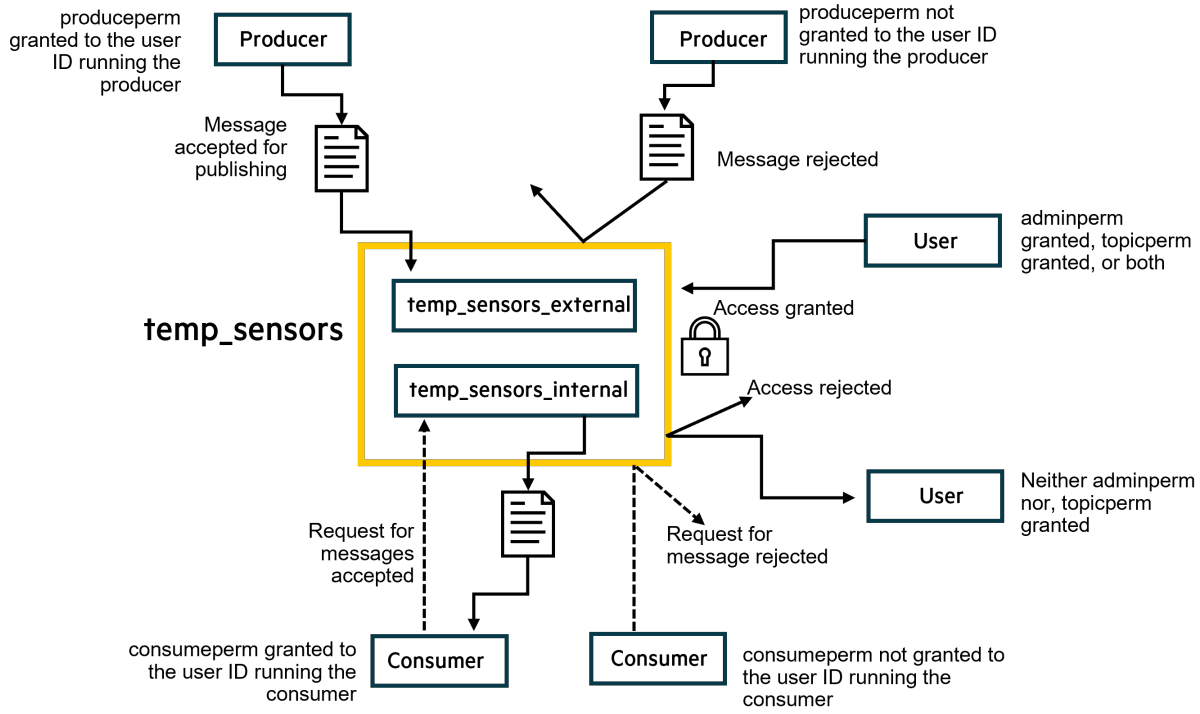


Figure 9. HPE Ezmeral Data Fabric Software security using ACEs

If streams are separated by using unique volumes, then ACEs can be used to create security separation and control.

Stream creation

HPE Ezmeral Data Fabric Software—Customer managed

Creating a new stream is as easy as creating a file; and adding new topics to a data fabric stream can be as easy as simply using them when writing a message, making streams convenient, and efficient for developers to use.

To create a new stream with the UI, see Figure 10.

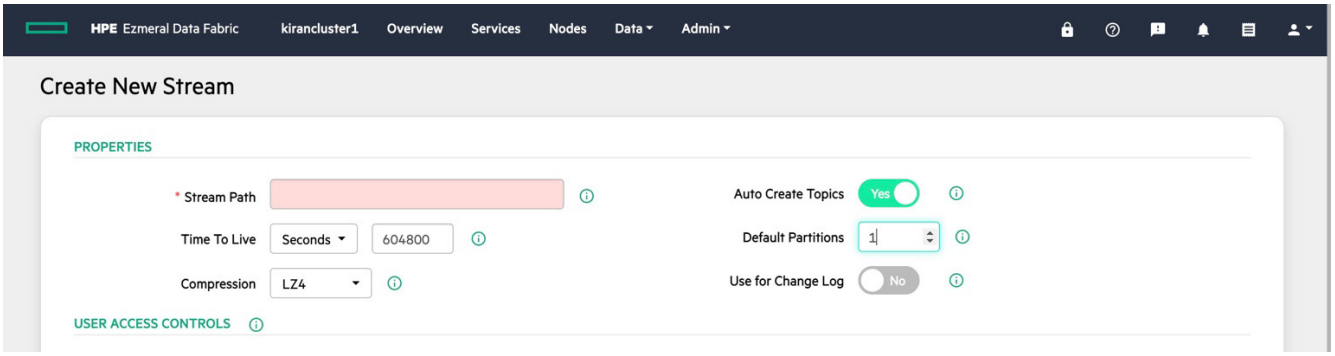


Figure 10. To create a new stream with the MCS UI

When creating multiple streams, it is advisable to simultaneously create multiple volumes for those streams and distribute the streams evenly across the volumes. Depending on the data types/usage in the streams, this will allow greater scalability and security governance configuration as the data scales with the application in the future.

When HPE Ezmeral Data Fabric is deployed on HPE GreenLake, there is a web-based UI that provides a unified interface for object tore installation, configuration, administration, management, and monitoring.

HPE Ezmeral Data Fabric Software

With HPE Ezmeral Data Fabric Software, there is a web-based UI that provides a unified interface for topic creation and configuration. When a user logs in to the UI it presents a summary page that shows all the Data Fabrics, and a summary of the number of topics configured in that Global Namespace, see Figure 11.

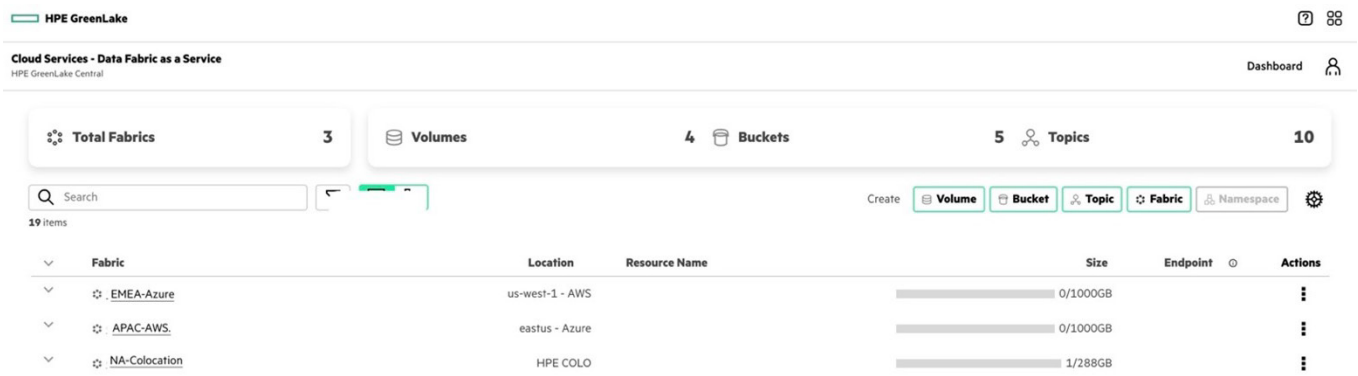


Figure 11. Stream creation using HPE Ezmeral Data Fabric Software UI

Event creation

To create a new event in HPE Ezmeral Data Fabric click the topic button. Enter the name for the topic and data fabric instance in which to create. See Figure 12.

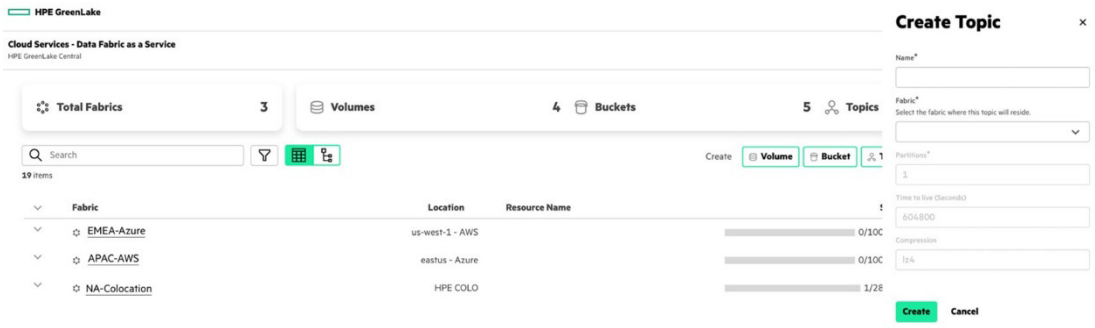


Figure 12. Screenshot showing how to create a topic in HPE Ezmeral Data Fabric

Use cases

HPE Ezmeral Data Fabric Software is ideal for a variety of use cases, including:

Application event pipelines

Many types of applications generate event or log data that must be centrally stored and analyzed to gain insights into user activity or application performance. HPE Ezmeral Data Fabric Software simplifies these pipelines by transporting events to a central location, from which they can undergo event-by-event transformation and analysis.

Database change capture

Most modern databases enable users to generate an event each time an entry is added or modified. Using Change Data Capture (CDC), these events can be published to HPE Ezmeral Data Fabric Software to keep systems like search indexes and caches synchronized, as well as to feed security or notification applications.

Tools like Oracle® GoldenGate and Attunity can be used to capture the changes in RDBMS and then send those as events to the topics. This data can then be used for analytics, as well as for creating reports and serving them to end users.

Internet of Things

The explosion in the number of smart devices and sensors has created many situations in which billions of data points are created by millions of geographically dispersed sensors. HPE Ezmeral Data Fabric Software provides a reliable, global transport for these messages, enabling you to perform analytics both at the source and at a central location.

Predictive maintenance

Near real-time alerts for predictive maintenance in case of device failure indications.

Kafka Wire Protocol Service

HPE Ezmeral Data Fabric Software supports Apache Kafka Wire Protocol Service. Apache Kafka Wire Protocol Service is a TCP/IP service that emulates a Kafka cluster backed by HPE Ezmeral Data Fabric Software. The service makes it possible for Apache Kafka clients written in any programming language to access topics in HPE Ezmeral Data Fabric Software.

Apache Kafka Wire Protocol Service allows user applications to connect, publish, and subscribe to HPE Ezmeral Data Fabric Software topics using standard Apache Kafka client libraries. User applications developed using Apache Kafka clients do not require any modification to work, including recompilation, reconfiguration, or dependency management.

Conclusion

HPE Ezmeral Data Fabric Software brings integrated publish and subscribe messaging to the data fabric converged data platform. It is built into the data fabric platform and requires no additional process to manage, leverages the same architecture as the rest of the platform, and requires minimal additional management.

a00124769ENW, Rev. 3

Hewlett Packard Enterprise