v1.2 | September 2022 | TN-2112

TECH NOTE

Kafka on Nutanix

NUTANIX
YOUR ENTERPRISE CLOUD

Copyright

Contents

1. Executive Summary

Kafka is a distributed streaming platform, originally built by LinkedIn, that is now part of the Apache Project. This tech note describes how deploying Kafka on Nutanix with AHV empowers your organization. Together they enable you to store, stream, and process large volumes of data. Using Kafka on Nutanix gives you seamless scalability, data locality, and data tiering. Nutanix also reduces operational complexity with intuitive consumer-grade management.

An IDC study showed that during 2020 the digital universe (which includes all data created, replicated, and consumed) exceeded 64 zettabytes (ZB)—64 trillion GB—of data, or over 9 TB per person. Machines such as sensors and smart devices generate most of this data as the Internet of things (IoT) grows exponentially. The study projected that the amount of digital data created over the next five years is going to be more than twice the amount of data created since the advent of digital storage. Getting all this data into different systems is a major challenge. With Kafka, IT departments can easily centralize large data volumes for multiple systems to subscribe to, which is why Kafka is a central part of most big data infrastructure.

Nutanix, like Kafka, is committed to simplicity, with streamlined infrastructure management, reduced deployment times, and enhanced performance to increase agility in business workflows. When running Kafka on Nutanix with AHV, IT can spend less time in the datacenter and more time cultivating data-driven insights to outperform the competition.

2. Introduction

Audience

This tech note is part of the Nutanix Solutions Library and provides an overview of the Nutanix with AHV and Kafka solution. We wrote it for IT architects and administrators as a technical introduction to the solution.

Purpose

This document covers the following subject areas:

- Overview of the Nutanix solution.
- Overview of Kafka.
- The benefits of implementing Kafka on AHV.

Unless otherwise stated, the solution described in this document is valid on all supported AOS releases.

Document Version History

Version Number	Published	Notes
1.0	July 2020	Original publication.
1.1	July 2021	Refreshed content.
1.2	September 2022	Added Nutanix Objects for Tiered Storage section.

3. Kafka Overview

Kafka is a distributed streaming message platform that lets you send data to and subscribe to data from a central location. Your organization may have thousands of data streams it needs to capture—security sensors, server logs, and user web streams that provide useful insights, to name a few. Organizing these data streams is difficult in a disaggregated environment. Different applications may need the same data and having the application itself send data to five different places is inefficient and not scalable.

Kafka comes in to solve this issue. Multiple endpoints can send data to Kafka in a category called a topic, then multiple consumers can subscribe to that topic to get the data. Let's look at the Kafka components.

Topics

Kafka stores messages from data sources in topics. For example, you can have all your endpoints send their web logs to your web-log topic. Topics are always multisubscriber, so many consumers can subscribe to a topic to receive its data. In the following figure, you can see that this Kafka cluster has multiple topics, with different producers tied to different topics.

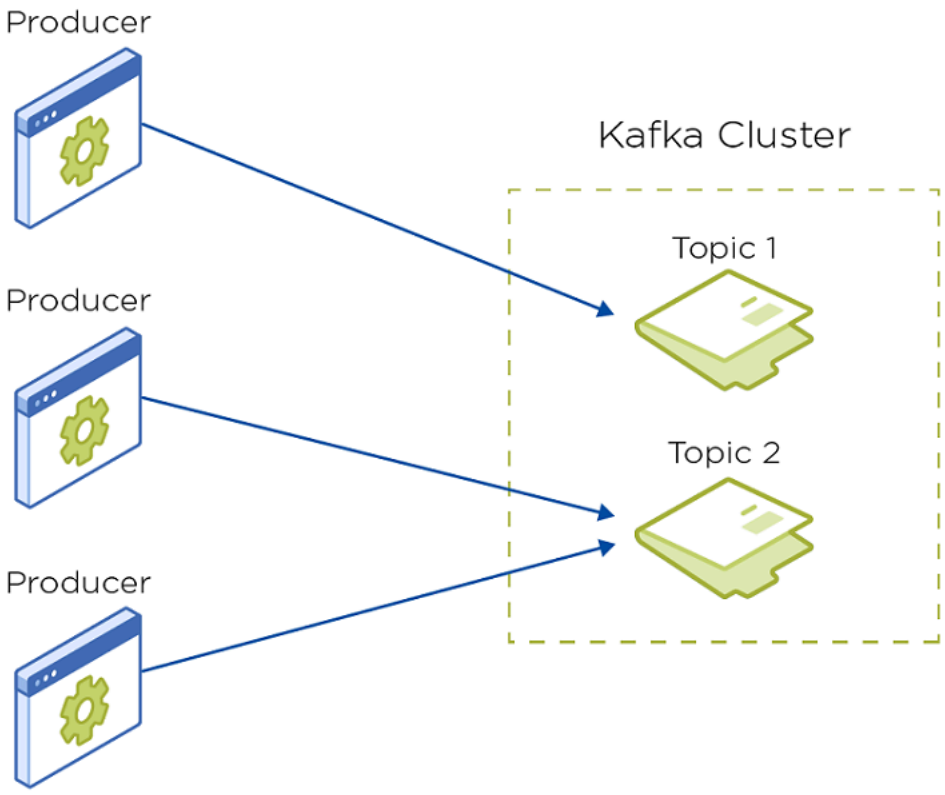

Figure 1: Kafka Cluster with Multiple Topics

Partitions

Partitions are subdivisions of topics. You split topics into multiple partitions for better throughput and to spread them across multiple servers. If you have five servers in your Kafka cluster, you should have five partitions to take advantage of all the servers. Try to spread partitions evenly across the cluster, because if some servers handle more of the load than others, you can end up with an unstable cluster. The following diagram shows Topic 1 divided into three partitions and spread across different brokers (physical servers or VMs) for better performance.

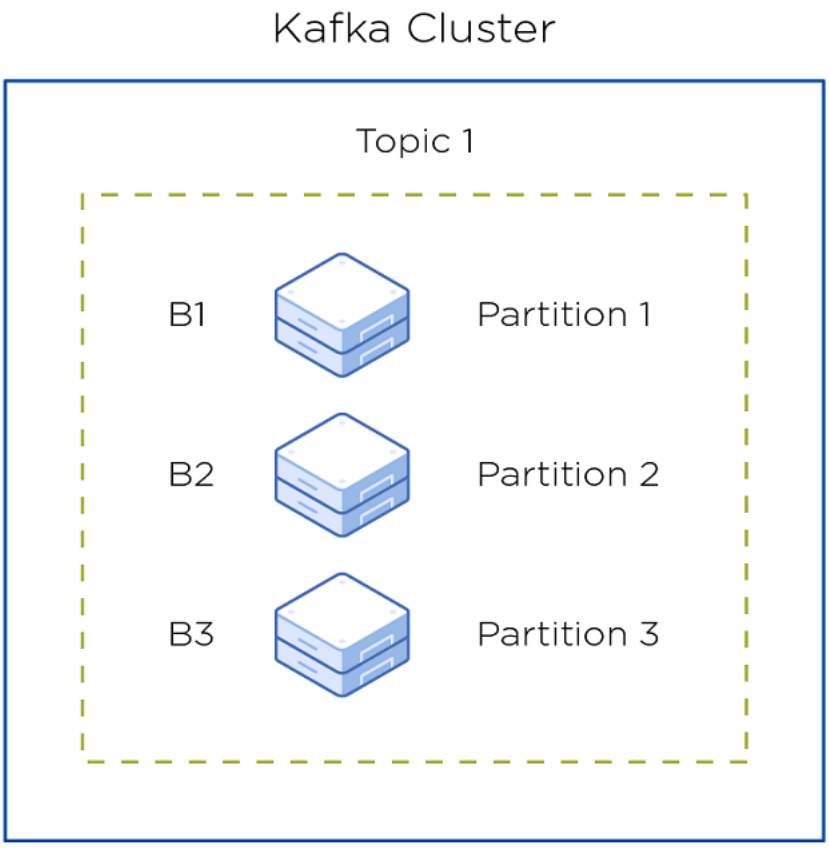

Figure 2: Topic 1 Divided into Three Partitions

Replicas

Replicas are copies of a partition. They enable high availability in Kafka clusters. With Kafka replicas, all data is written to and read from the leader (the primary partition). The followers have copies of the partitions that the system can use if the leader fails. Replicas are never used for read or write operations; the leader handles writes, then sends them to the replicas.

The following diagram shows that each replica is on a different broker, so you don't lose more than one copy if a broker fails.

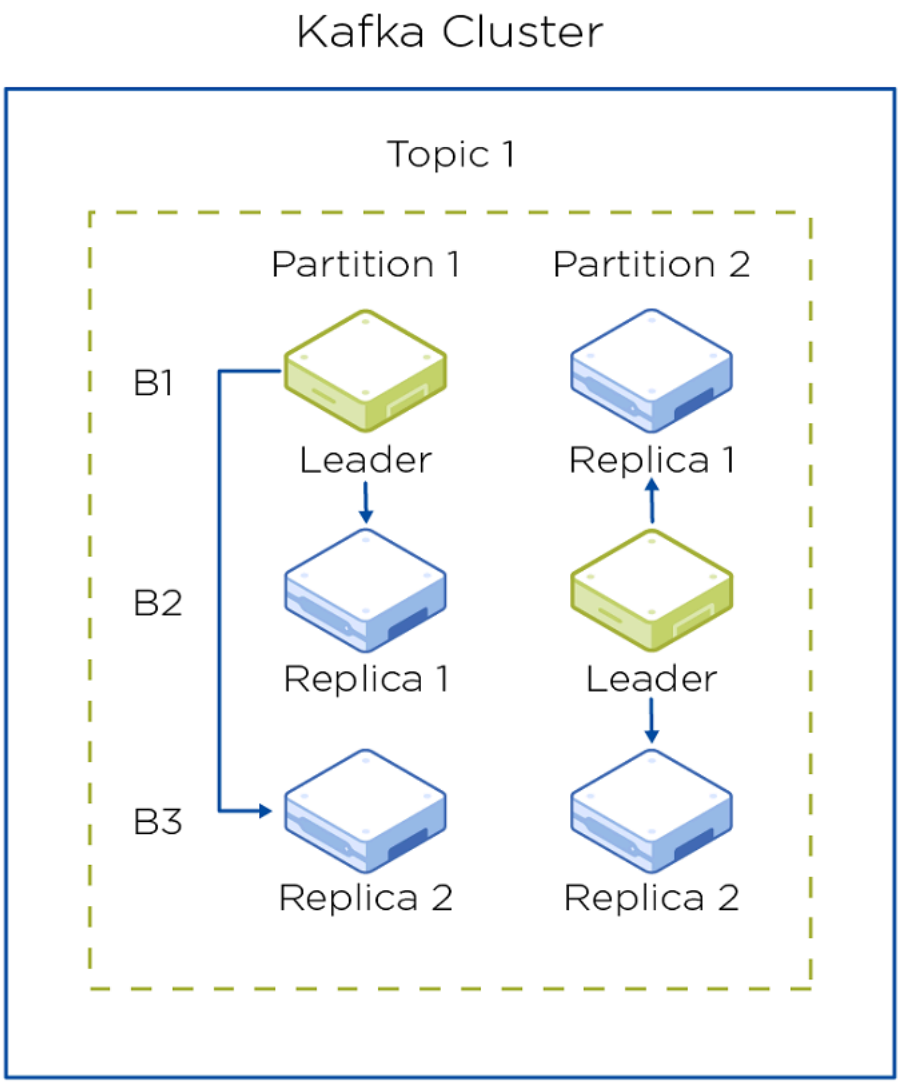

Figure 3: Replicas on Different Brokers

Brokers

Brokers ingest and replicate the data in a Kafka cluster. The number of topic partitions that each broker can have depends on your deployment details, but

in general we recommend fewer than 10,000 per Kafka cluster. Brokers can host partitions from multiple topics. They are stateless and rely on ZooKeeper to maintain their cluster state. When you design your Kafka cluster, look at how many topics, partitions, and replicas you have in order to understand how many brokers you need. In the previous figures, you can see that brokers enable high availability and help with parallel processing by spreading topics across multiple partitions.

Producers

Producers publish data to the Kafka cluster, which converts it into topics for consumers to read. Messages are directed to certain partitions for a topic. The producer chooses the correct partition based on a key or uses a round-robin method if there isn't a key.

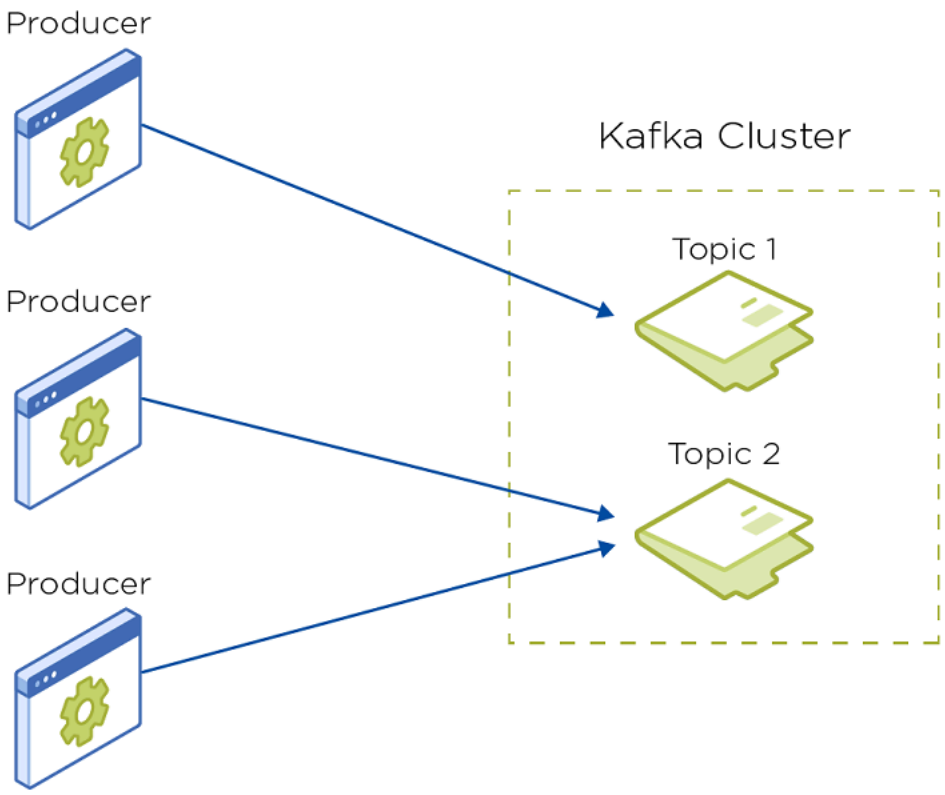

Figure 4: Multiple Producers Sending to Separate Topics

Consumers

Consumers are more complicated than producers. Consumer groups typically subscribe to one or more topics, and each consumer in a group is given an equal share of partitions to read messages from. For example, if there are 20 partitions to consume and 10 consumers, then each consumer is assigned 2 partitions. If there are 10 partitions and 20 consumers, 10 consumers sit idle. Individual consumer groups keep track of the individual messages they've read so that Kafka doesn't have to. This ability means that consumer groups can replay the messages from any point and resume reading from a certain point if the consumers crash.

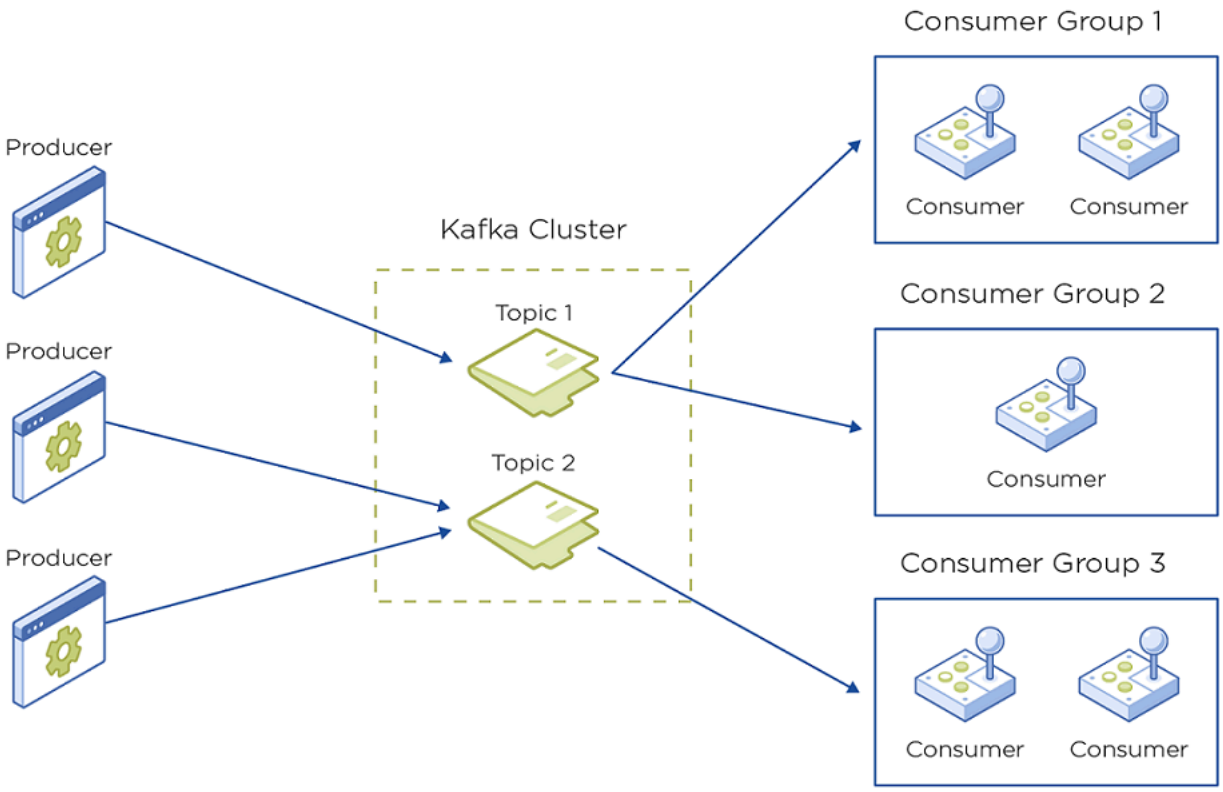

Figure 5: Kafka Cluster with Multiple Consumer Groups

4. Kafka on Nutanix

The Kafka on Nutanix with AHV solution is easy to use and quick to deploy and scale out. Running big-data applications on a hyperconverged platform makes your infrastructure truly invisible, decreasing deployment and expansion times and making maintenance easier.

Scalability

Kafka scales out as your load increases, which means you can start from a few thousand messages a day and grow to many billions of messages a day. Nutanix follows this same principle—you can start with a few nodes and scale out as needed with as little as one node at a time. Nutanix also decreases the amount of time it takes to scale your cluster, as you can add new nodes in under an hour.

Usage and Management

With Nutanix you can improve overall host usage by mixing components. Brokers are very I/O heavy but can underuse CPU on hosts. Consumers mainly use CPU, so if you mix consumers and brokers on a Nutanix cluster you can achieve better overall usage of the entire environment.

Kafka is the messaging intermediary, but there are other components in the big-data pipeline, and Nutanix provides a single pane of glass to manage Kafka brokers, consumers, and other software components. Running Kafka on Nutanix makes management easy and removes silos, which means that extra capacity doesn't go to waste. Nutanix enables you to quickly scale individual big-data components up or down as needed.

Performance Insights and Automation

Troubleshooting is also easier on Nutanix. It can be difficult to narrow down and fix performance problems on bare-metal deployments because there aren't any granular tools. With Nutanix you can go down to the individual host disk level to look at performance.

Bare-metal deployments also don't automatically mitigate performance issues. Nutanix handles many performance issues automatically, which greatly increases uptime. If a single disk starts having performance issues or SMART alerts indicate a disk is going bad, Nutanix automatically stops using that disk and copies the data off. Nutanix health checks indicate when an individual host becomes unstable and migrate the VMs proactively before you have a critical issue.

Failure scenarios are important in Kafka. As stated earlier, topics are broken down into partitions, and each partition has a leader and followers. When the broker storing a partition leader fails, one of the followers becomes the new partition leader. Typically, when you start a cluster or create a topic, its resource distribution is balanced. Failure scenarios can cause the broker configuration to become unbalanced, leading to an unstable broker.

When your brokers are on Nutanix, after a physical host fails, the broker VM restarts within minutes. Using automatic leader rebalancing, you can quickly get your Kafka cluster back to a fully functioning state. This instance is just one example of how running Kafka on Nutanix can help with data availability and data safety. Planning for the same failure scenario in a bare-metal environment means adding in extra overhead that tends to be left idle.

Nutanix Objects for Tiered Storage

Confluent provides an enterprise-ready event streaming platform built on Kafka with additional features that drastically simplify Kafka's management and a customer support wrapper. Confluent Platform 6.0.0 introduced support for Tiered Storage, which allows you to offload completed log segments from the Kafka brokers to a remote tier where you can retain them for longer periods at a more efficient cost. Confluent certified Nutanix Objects as a Tiered Storage

endpoint. While you can retain data for longer periods by increasing the storage capacity on the brokers themselves, this approach results in several problems:

- Cost: Kafka was designed to support real-time applications, and to achieve the performance required for this support, the brokers must be equipped with high-performance storage. However, this class of storage is expensive so costs quickly increase when you add much more of it. The increased cost is particularly difficult to justify given that older data doesn't need fast performance.
- You can't scale storage and compute independently: Each broker can usually only accommodate a limited number of drives. Therefore, you may need additional brokers to hit the required capacity point, even if you don't need additional compute resources, resulting an excess of expensive resources.
- Increased replication traffic between brokers: Because the brokers are data-heavy in this scenario, when you add new brokers, a large amount of data is replicated between the existing and new brokers, causing increased consumption of compute resources and network bandwidth. This data heaviness also slows down reprotect times when a broker fails.
- Requests for older data can block requests for real time data: When consumers request older data, it must be fetched from disk, which is a slower process than fetching data from the page cache. This process can block network threads, slowing down access for consumers of real-time data using the same network thread.

Fortunately, you can address each of these issues by offloading the older data to Nutanix Objects using Tiered Storage:

- Use low-cost object storage for the older data.
- You can scale the object storage tier independently of the brokers, eliminating any danger of being forced to overprovision compute.
- By remaining data-light, brokers can allocate more resources to producer and consumer requests rather than background replication tasks. Furthermore, data reprotection completes more quickly when a broker fails.

- Tiered data is served on a separate path to local data. As a result, consumer requests for real time data are isolated from consumer requests for older data, eliminating blockages that otherwise occur on the I/O path.

5. Conclusion

Nutanix and AHV provide a high-performing and stable foundation for Kafka. Nutanix simplifies and enhances the infrastructure and overall deployment, eliminating the need for complex and time-consuming bare-metal environments. Nutanix can handle any Kafka performance needs as the architecture scales linearly.

Nutanix provides an on-premises cloud architecture that gives you the fast and simple deployment you need with Prism, a single management plane that enables you to expand, monitor, and update your environment. The Nutanix software's automated self-healing features give customers more uptime and better overall performance by keeping the Kafka environment healthy.

Nutanix can handle all your big data needs, enabling you to use the time you normally spend on infrastructure to get insights from your data instead.

About Nutanix

Nutanix is a global leader in cloud software and a pioneer in hyperconverged infrastructure solutions, making clouds invisible and freeing customers to focus on their business outcomes. Organizations around the world use Nutanix software to leverage a single platform to manage any app at any location for their hybrid multicloud environments. Learn more at www.nutanix.com or follow us on Twitter @nutanix.

List of Figures